Uniwersytet im. Adama Mickiewicza w Poznaniu

Wydział Neofilologii

Instytut Językoznawstwa

Communicative Alignment of Synthetic Speech

Jolanta Bachan

Rozprawa doktorska

Opiekun naukowy:

prof. UAM dr hab. inż. Grażyna Demenko

Poznań 2011

Acknowledgements

I wholeheartedly thank Professor Grażyna Demenko for her help, support and supervision of my work over the years of cooperation. I would like to thank Professor Piotra Łobacz for her continuing support, belief and trust in me. I would like to extend special thanks to Professor Dafydd Gibbon for his invaluable help and discussions about my work. Further thanks to Professor Maciej Karpiński for providing his dialogue corpus and teaching me about human-computer interaction and dialogue analysis in various classes. Thanks to Professor Władysław Zabrocki for being always helpful in solving problems connected with the thesis academic procedures.

I would also like to thank my colleagues and all the students, friends and relatives who willingly took part in my experiments. Without their input, my work would not have been possible.

Further thanks to the Bielefeld University, the Kulczyk Family Foundation, the Scholarship Foundation of Professor Władysław Kuraszkiewicz and the International Speech Communication Association for awarding scholarships to me, which helped me to focus on my academic work and develop my scientific interests.

And finally, immeasurable thanks to my parents and my brother for their unfailing support and love for me.

The research presented in this thesis was partly carried out within the scope of the research grant no. N N104 11 98 38 received from the Minister of Science and Higher Education.

Badania przedstawione w pracy zostały częściowo zrealizowane w ramach grantu promotorskiego nr N N104 11 98 38 przyznanego przez Ministra Nauki i Szkolnictwa Wyższego.

Table of Contents

Index of Tables

Index of Figures

Chapter 1: Introduction

1.1 Objectives of the thesis

The central claim of the thesis is that a dialogue system should be well-motivated by dialogue theory and by analysis of actual dialogues, and that the resulting system should be tested in a real-world scenario. Based on this claim, the thesis concentrates on methodology and investigates a wide range of methods required for fulfilling these requirements adequately. The operational aim is to provide a simple proof-of-concept dialogue system based on the claim and combining written and spoken communication. The operational aim is therefore not to develop a fully functional dialogue system, but a prototype which illustrates the main claim and the methodology of the thesis in a simulated stressful emergency scenario.

The alignment theories discussed by Pickering and Garrod (2004) will be the focus of the present work. According to the alignment theories, alignment in dialogue takes place on semantic, syntactic and pragmatic levels. In the present thesis the work is focused on the semantic level and the thesis claim is:

> Alignment of semantic representations is essential for successful communication in a dialogue.

The intention is to test semantic alignment both descriptively, using the dialogue act approach of Bunt (2000) and with two corpus linguistic studies, and operationally, with a finite state text-in-voice-out dialogue system which has been specially designed for this purpose. The finite state dialogue system uses a male Polish synthetic voice which was created for this application, and an innovative combination of two finite state systems: a finite state dialogue manager which controls a finite state map traversal system. To assure success in communication, routines for recovery from misalignment have also been addressed in the dialogue manager.

The methodologies which are dealt with include:

1. Linguistic dialogue theories.
2. Theory-based corpus linguistic description of dialogue.
3. Dialogue modelling with automata.
4. Speech synthesis component of a dialogue system and voice creation for speech synthesis module and its evaluation.
5. Dialogue corpus creation and evaluation with microvoices (synthetic voices which only cover a restricted range of the language, for experimental purposes).
6. Dialogue system demonstration prototype and evaluation.

In order to create the demonstration prototype, the specific computational linguistic issues to be addressed include:

1. Dialogue design based on a formal analysis of the dialogue act in the first corpus linguistic study, with finite state modelling, and on a scenario-specific dialogue act analysis in the second corpus linguistic study.
2. Formal-informal speech style selection in a realistic stress scenario (emergency dialogue with a hospital call-centre).
3. Formal properties of automaton models.
4. Information extraction from two corpora for dialogue modelling.
5. Information extraction from text and speech corpora and a speech corpus creation for synthetic speech voice creation.

1.2 Motivation of the thesis

In the information society people need to cooperate more and more with computer systems, and therefore computer systems need to be designed which make this cooperation easier. Typical activities such as looking for timetables on the internet, booking flights via online forms and changing the settings of a mobile phone in call centres are very common. The human user has to follow automatic instructions because in general there is no human operator. However, such communication systems are not natural, often the processes are lengthy and time-consuming, and they are always restricted to the pre-defined options of the system. In certain situations, when these options fail the customers are redirected to

human operators as the required tasks are too complex for the system. Two main issues are involved here: first, the 'intelligence' of the system, and second, the 'naturalness' of the input-output interaction. The present study concentrates on input-output interaction with text-in-voice-out dialogue, a common configuration in commercial information systems such as satellite navigation devices and screen readers for the blind.

Because talking is more natural than dialling numbers or filling in text forms, many institutions provide call centres where people can choose to talk about their problems or requests with a human operator. However, human work time is very expensive and one person can basically deal with just one customer at a time. Therefore in the technologies concerned with making input-output issues more natural much effort is being put into the development of dialogue systems which can communicate with a human being via the speech signal and deal with more than one customer at a time (cf. the Vermobil project, Wahlster 2000, and the SmartKom project, Wahlster 2006). Such a dialogue system has a speech recognition module which receives human speech input and converts it to a form which is understandable by the computer and produces synthetic speech to provide information back to the user. The communication between the human user and the computer system is administered by a dialogue manager which decides on the next actions the system should take. In addition to acoustic speech recogniser and speech synthesiser components, the system also includes computational linguistic components such as a machine-readable lexicon together with a parser which extracts meaning from the pre-processed human speech, and a natural language module generation which converts the reply created by the dialogue manager into the natural language form. An example of a dialogue system architecture is shown on Figure 1.

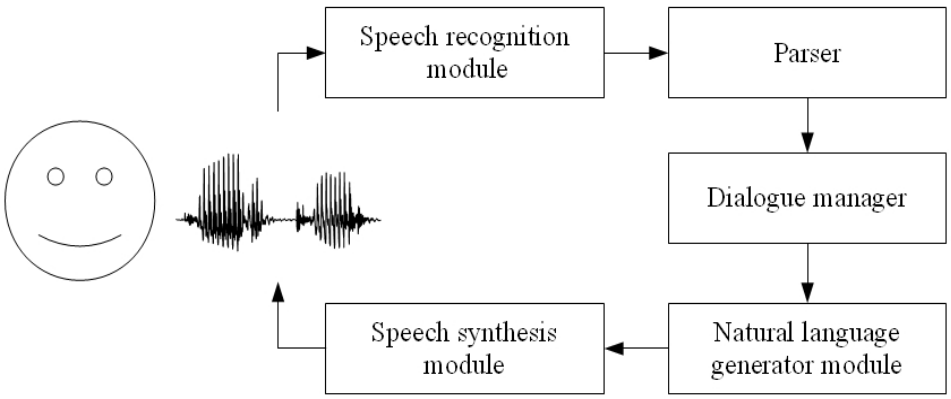


Figure 1: Simplified architecture of a spoken dialogue system

1.3 Alignment and accommodation

In recent years new aspects of communication have been investigated which are relevant for developing natural human-computer dialogue interaction. These include alignment of communication form and content between the interlocutors (Pickering & Garrod 2004) and accommodation of interlocutors to each other (Giles et al. 1992). It has been noticed that while communicating, interlocutors tend to adopt each other's behaviour such as style of speaking, vocabulary, gestures.

In the present context, alignment is meant here as adaptation on the syntactic, semantic and pragmatic levels of communication between the two interlocutors, including the choice of similar lexical items and speaking style. However, it needs to be emphasised that the form, content and degree of alignment depends on the communication situation and status relations between the interlocutors. The main distinction for emergency scenarios to be made is between alignment in public and private situations. In public situations in which interlocutors do not know each other the degree of alignment of their behaviours has been found to be smaller than in face-to-face conversations between two close friends (Batliner et al. 2008). In fact, there may be deliberate non-alignment between a call-centre operator and an emotional caller, in order to calm the caller.

Table 1 presents a dialogue excerpt with an example of alignment. In the dialogue excerpt coming from the dialogue corpus recorded for the present study, an example of

lexical alignment is shown. Here Speaker A, while giving instructions, talks about the roundabout. Speaker B does not see the roundabout, so Speaker A defines it as a ʻcircular flower bed'. In order to be understood, because Speaker A is nervous, Speaker B adapts the word ʻflower bed' to refer to the roundabout, but then immediately uses again the regular word ʻroundabout'. Speaker A starts to use the word roundabout again appartently unintentionally because her focus later in the dialogue is on giving the next instructions of the route and does not think of the roundabout anymore.

Table 1: Dialogue excerpt with lexical alignment

Polish	***English***
A: [route description] Przy *rondzie* są roboty, więc trzeba będzie je objechać [route description]	A: [route description] At the *roundabout* there are roadworks, so they must be passed by [route description]
B: Może Pani powtórzyć. Nie widzę tutaj ronda po drodze.	B: Can you repeat. I don't see any *roundabout* on the way.
A: Znaczy... rondo, to jest taki, taki okrągły kwietnik. [yyy] jest [yyy] między sklepem a lodziarnią.	A: It means... the roundabout, this is, such a *circular flower bed* [yyy] is [yyy] between the shop and the ice cream parlor.
[route description]	A: [route description]
A: Następnie objechać rondo – ten taki okrągły kwietnik.	A: Then go round the *roundabout* – this c*ircular flower bed*.
B: Czyli po tym jak skręce w prawo...	B: So after having turned right...
A: Tak.	A: Yes.
B: Muszę jeszcze skręcić w lewo, żeby dojechać do tego kwietnika.	B: I again have to turn left to get to this *flower bed*.
A: Tak, tak, tak. Jestem dość zdenerwowana iii iii wszystko... wszystko wydaje mi się takie... Przykro mi.	A: Yes, yes, yes. I'm quite nervous aaand aaand everything... everything seems to me so... I'm sorry.
B: Dobrze. Proszę się uspokoić. Czyli na rondzie gdzie muszę skręcić?	B: Good. Please, calm donw. So at the *roundabout*, where do I have to turn?
A: [yyy][um] Na rondzie musi Pani [yyy] na rondzie musi Pani skręcić w [y] obok lodziarni [route description]	A: [yyy] [um] At the *roundabout* you have to [yyy] at the *roundabout* you have to turn at [y] next to the ice cream parlor.

The present study focusses on basic aspects of alignment which are relevant for human-computer communication in stressful emergency scenarios in public. In public stress situations it is necessary to know the conversation is conducted in terms of formal and informal styles, and not what emotions, in the usual senses of the term ('fear', 'anger', 'sadness', 'happiness' …, 'neutral'; cf. Ortony & Turner 1990, Murry & Arnott 2008, Bachan & Surmanowicz 2008), are expressed in the interlocutors' speech: for present purposes, negative emotions such as 'fear', 'anger', 'sadness' are included in the concept of 'stress'. The 'informal' and 'formal' styles are more related to private versus public

situations than to emotion, and both may occur in stress scenarios. These distinctions are taken into account in the dialogue system demonstration prototype.

If one of the interlocutors becomes involved in a difficult position and undergoes great stress, the interlocutor to which the stressed person talks to will try to align their speech on the syntactic (including lexical), semantic and pragmatic levels (Branigan et al. 2000), but will not try to empathise with the emotional state of their interlocutor. In the course of the conversation, the interlocutors will start to use the same vocabulary (Brennan & Clark 1996; Clark & Wilkes-Gibbs 1986; Wilkes-Gibbs & Clark 1992), but not necessarily both speaker's voices will start sounding nervous because of the stress affecting one of the interlocutors. However, this is not necessarily the case with a professional call centre operator.

It is assumed that the speaking style towards a stressed person (or a person in any other emotional state) is different than toward a person who does not show any emotions. The dialogue system should be able to recognise the emotional states of its users and based on the prosodic and lexical speech characteristics apply a speech style which will be aligned with these emotional states (cf. Batliner et al. 2003).

1.4 Modelling dialogue

The goals of the present investigation include providing explicit models for relevant aspects of human-human communication connected with alignment and accommodation. The literature on these topics does not consider ways of aligning synthetic speech with the human interlocutor in their interaction, focussing specifically on stressed and emotional speech in crisis situations, although acceptable human-computer interaction is the subject of much research. The models should enable appropriate speech style selection in these situations, based on the observations that existing models of emotion are both too simple and too speculative, that actors imitating crisis speech are not producing authentic crisis speech, and that in public stress scenarios the formal-informal style dimension is more relevant than emotion space.

The general working hypothesis is that it is possible to replace traditional emotion label sets with a generic model of the following type (which would also apply to 'emotion' in addition to 'style' if required):

TRIGGER_SITUATION → STYLE → STYLE_MANIFESTATION

The trigger situation is the particular public stress scenario which requires a certain formal or informal communication style. The style manifestation is the set of syntactic, lexical and phonological conventions which are associated with the chosen style. The specific hypothesis is that it is possible to design and implement a speech style selection module based on this model to drive synthesiser-interlocutor alignment, and to implement it in speech synthesis software. Such a module should improve the naturalness and efficiency of human-computer communication. In the spoken dialogue demonstration prototype, the styles and style manifestations are considered, but an automatic recognition of alternative trigger situations (age, gender, social status, task etc.) is not included since a specific single simulation scenario (a variety of map task with university graduate students) is used.

For the spoken dialogue demonstration prototype, the focus is on the dialogue manager and speech synthesis modules.

In human communication the interlocutors tend to align their behaviours, not only speech, but also gestures and body movements. The present investigation is not concerned with multimodal communication of this kind; consequently, the selected scenario is a telephone-like scenario with no visual contact between interlocutors. The present study is also not concerned with recognising and manipulating phonetic features of speech, e.g. prosodic and paralinguistic features such as voice quality, intonation, rhythm and tempo of human speech. However, styles are also characterised by lexical items and other markers such as hesitation phenomena, repetitions and curses, suggesting different behavioural and expressive states of the interlocutor. Based on the analysis of these items, a dialogue system can generate a kind of output which would be expected in human-human communication. These stylistic markers in human-human communication may also indicate that the communication is not successful; if a recognition module were to be developed, situations when the system cannot understand the speaker would need to be modelled. In such situations the dialogue manager should select a different trigger situation for planning the conversation. Similarly, the dialogue manager may also apply a different speaking style to be generated by the speech synthesis module. Such a system would analyse the trigger situation, for example, domestic violence, and compare this trigger

situation with the phonetic features manifesting human emotions, for example, fear. If the dialogue manager finds a scenario to be used in such a situation, it applies the appropriate scenario and an appropriate speech style, for example a reassuring style.

1.5 Contributions of the present research

First, the Pickering and Garrod (2004) approach to alignment is criticised and modified in the area of semantic alignment. The first criticism is that Pickering and Garrod are not precise about what semantic alignment is. In the present research, two corpus linguistic studies are undertaken for this purpose, and in the operational system a map with certain unforeseeable properties is used as a reference point for semantic alignment, and negotiation of a route through the map requires semantic alignment of different types. The second criticism is that Pickering and Garrod only deal with cooperative alignment. The present research does not deal with non-cooperative alignment, but it deals with cooperative non-alignment to some extent, between a professional call-centre operator and a caller.

Second, the dialogue act approach of Bunt is criticised because in his earlier work, at the time of the corpus linguistic studies, the dialogue acts were simply listed abstractly, with no empirical illustration. A selection of Bunt's dialogue acts was made for the purpose of the present research, and investigated in the corpus linguistic studies. In later work, Bunt (2010) added empirical information, but did not deal with scenarios such as the emergency calling scenario. A second criticism is that in the earlier work, and to a large extent in the later work, Bunt does not deal with sequences of dialogue acts, but only with a hierarchical classification of dialogue acts. In the present research, sequences of dialogue acts in the corpus and also in the operational system are modelled with finite state automata.

Third, the present research has an operational outcome, as a text-input-voice-output dialogue system which is intended to test the points listed above, and an evaluation of this system. The use of two finite state systems, one as a dialogue manager, and the other as a map traversal algorithm, with the dialogue manager controlling the map traversal module. One further original contribution in this context is the new Polish male voice PL2 for the MBROLA (Dutoit et al. 1996) speech synthesis system.

1.6 Overview

Following the introduction to the topic and the research aims presented in this chapter, in Chapter 2 a brief selection of relevant theoretical linguistic approaches on alignment to dialogue description and their implications for development of the spoken dialogue demonstration prototype are discussed. In Chapter 3 dialogue modelling is briefly introduced and components of dialogue systems are presented. In Chapter 4 a pilot study in which theoretical principles are applied to actual dialogue description is undertaken. In this study the research is carried on an existing dialogue corpus. Chapter 5 presents work development of provisional automaton models of the dialogue. The aim is to develop techniques and tools for dialogue modelling in the prototype dialogue system. Chapter 6 is concerned with prerequisites for developing a speech synthesis module for a dialogue system. It presents results of diphone search in existing text and speech corpora as well as introduces two tools for efficient diphone database creations developed for this purpose. The creation of a speech corpus used for Polish male synthetic voice creation is presented together with evaluation of the voice. Chapters 7 and 8 present the test of the thesis claim. Chapter 7 is a corpus linguistic study of the kinds of alignment in public emergency dialogues which are required for designing the spoken dialogue demonstration prototype. In this Chapter, creation of dialogue corpus is presented and prompt materials and recording techniques are discussed. The addressed scenarios are stressful emergency situations and neutral dialogues based on map and diapix tasks. The development of the spoken dialogue demonstration prototype, including evaluation with human users, is dealt with in Chapter 8. The chapter presents an innovating technique combining two finite-state-automata which work together in the dialogue system: one for map traversal, and one for dialogue negotiation. Chapter 9 is concerned with the conclusions from the present work and tasks for the future.

Much of the empirical and technical material (materials for speech corpus recording scenarios, tables with results of empirical studies, automaton models of dialogue structure, code of software tools) is included in Appendices in order to avoid distraction from the main argument in the text.

Chapter 2: Alignment – critical overview

2.1 Chapter overview

The specification of a dialogue system depends partly on linguistic, psycholinguistic and logical specifications of the domain of language in dialogue. The discussion of these concepts will be very selective and brief, because relevant studies tend to be very general, from the point of view of speech technology and are important foundations for dialogue system development but not the focus of attention in the present research. In this chapter, the relevant concepts of 'alignment', 'coordination', 'common ground', 'speech act', dialogue act', 'sign', and 'language as-product vs. 'language-as-action' will be discussed. The discussion mainly follows the approach of Pickering & Garrod (2004) and Levelt (1992). The main thesis of this chapter is that alignment in dialogue takes place on syntactic, lexical, semantic, and pragmatic levels of language as well as on the obvious levels of pronunciation and prosody of speech.

2.2 Basic alignment models

Alignment was defined in the Introduction as adaptation on the syntactic, semantic and pragmatic levels of communication between the two interlocutors, including the choice of lexical items and speaking style; the form, content and degree of alignment depend on the communication situation and status relations between the interlocutors. For the present investigation, the main distinction for emergency scenarios is to be made between alignment in public situations and alignment in private situations, which affect the use of different utterance styles. The problem of emotional alignment is important, but not directly relevant for communication in public situations. Even if a person calling a call-centre is highly stressed and emotional, it is not a good idea for the call-centre response to use the same emotional utterance types, but the response must still be aligned on the basis of appropriate strategies for achieving successful communication with a stressed person.

There are several questions which must be answered clearly.

What function does alignment have in communication? For the present study, the following function is the most important:

> The general function of alignment is coordination between interlocutors in order to achieve a successful outcome of communication.

Alignment in dialogue is a component of communication, is a social activity, and a successful outcome may be defined on many different levels: alignment of pronunciation, alignment of vocabulary, alignment of syntax, and also alignment of descriptive semantic content and alignment of pragmatic functionality. Another issue is whether alignment is a consciously aware strategic behaviour or a subconsciously implicit behaviour is not in the focus of the present study.

What is the purpose of alignment in a dialogue system? Alignment is a kind of behaviour control procedure during communicative interaction. People may use many levels of alignment procedure in communication, including the language features which have been mentioned already, and also gestures of the face, the hands and the position of the body. Clark (1985) suggests that other kinds of non-linguistic coordinated activity, such as dancing, and cooperation on the same practical task, may be subject to the same principles of alignment.

What approaches to modelling alignment have been proposed? Pickering and Garrod (2004) outline four approaches which will be discussed below.

2.2.1 Alignment as a social phenomenon

As a social phenomenon, alignment in communication depends on status relations between the speakers and listeners, who consider the social effect of their utterances. The principle of alignment as a social phenomenon is that people want to communicate politely, cooperatively and successfully with each other (Grice 1975; Giles et al. 1992; Allwood et al. 2000). It is true that there are also types of communication which are not based on cooperation but on conflict and aggressiveness. In these communication scenarios alignment may be deliberately avoided, but in some way alignment is still a reference point for communication. However, in the stressful emergency dialogue scenario involved in the present study, successful communication will be cooperative and potentially supported by alignment.

From the point of view of dialogue system development, an exclusively social view of alignment is too restrictive because it concentrates on the obvious observation that alignment is a social phenomenon. But this is incomplete: by concentrating on pragmatics, the view does not take the necessary formal dimensions of communication such as appropriate formulation (pronunciation, lexicon and syntax), adequate expression of content (semantics) into consideration.

2.2.2 Alignment as Audience Design

Another model of alignment considered by Pickering & Garrod (2004) is the audience design model. In the case of audience design, the speaker chooses expressions most likely to be correctly understood and accepted by the listener. The aim of this is to enhance communication on the basis of beliefs which the speaker has about the hearer. The main problems with the theory about the Audience Design mechanism of alignment are:

1. From a processing point of view, the Audience Design is very complex to compute. Many levels of language, speech and interaction have to be taken into account during the alignment process, involving listener modelling and inference making.
2. The Audience Design model does not provide a robust procedure, since each aspect of alignment depends on many assumptions which may not be true.
3. The Audience Design model does not explain the other pragmatic, social and non-linguistic dimensions of alignment which affect the speaker.

2.2.3 Alignment as Priming

Alignment seen as Priming involves mechanisms of linguistic representation which are generally considered as being automatic, like other priming processes. Priming means the preparation of a speaker or hearer for behaving in a certain way on the basis of previous perception or behaviour. In this view, Pickering & Garrod (2004) claim that alignment automatically falls out of linguistic processing, because priming applies to many other kinds of linguistic behaviour. Pickering & Garrod point out that this view offers the following features:

1. Priming is cognitively economical: the processes involved are those which are involved in regular speech production.

2. Priming is robust: the need for detailed listener models is not present, information is taken from perception of the immediate context.
3. Priming explains linguistic repetitions and imitation.
4. Priming is computationally less complex for common kinds of phonetic and phonological alignment, which is very rapid and is 'resource-free", I.e. does not involve huge cognitive resources.
5. Alignment is a process which takes place below awareness levels.

Alignment is a process which does not only concern normal speakers. It also concerns speakers with some kinds of impairment, such as autism. In an experiment, the alignment of Noun Phrase structure in children was examined (Allen et al. 2011). In this experiment, the syntactic alignment behaviour of autistic (Autistic Spectrum Disorder, ASD) and non-autistic children was compared. The children with Autistic Spectrum Disorder (ASD) spontaneously converge, or align, syntactic structure with an interlocutor. Children with ASD were more likely to produce a passive structure to describe a picture after hearing their interlocutor use a passive structure to describe an unrelated picture when playing a card game. Furthermore, they converged syntactic structure with their interlocutor to approximately the same extent as did both chronological and verbal age-matched controls: autistic children, 24%, age-matched children – 21%, verbal-age-matched controls – 20%. These results suggest that the linguistic impairment that is characteristic of children with ASD, and in particular their difficulty with interactive language usage, cannot be explained in terms of a general deficit in linguistic imitation such as alignment by Priming.

The Priming point of view can also be criticised. Priming does not explain the more abstract levels of alignment, since it is based exclusively on the perception of linguistic input, and it does not account for functional properties of alignment in increasing the chance of cooperative and successful communication.

2.2.4 Alignment as Inter-level Interaction

In the Interaction model, alignment automatically takes place at several different levels of language at the same time. Pickering and Garrod (2004) consider that the Interactive Alignment model is too strong if it is taken literally. For example, it is not always the case

that alignment at one level of representation leads to alignment at other levels. Alignment for example at lexical level may mask an underlying misalignment at the semantic level, for example when ambiguity is involved: “John!” may denote John Brown or John Smith, for example.

The Inter-Level Interaction model will be referred to again below. For the present study, the point is that the model implies that the different views of alignment may not be competitors. They may occur in combination as simultaneous and interacting procedures in a multiple mechanism composed of the described components: social behaviour, audience orientated, primed, interactive or all of these. The components does not have to be mutually exclusive and some context may require any combination of these components, or all the components.

2.2.5 Alignment in human-computer interaction

In studies of human-computer interaction, it has been suggested that the way humans interact with computers is related to beliefs about the social status of interlocutors, beliefs and knowledge about computers, and beliefs about the linguistic capability of interlocutors. It appears that there may be a lower degree of alignment when speakers are to interact with people of lower social status and more alignment when the speaker believes their interlocutors to be linguistically less capable. In human-computer interaction it seems that that people communicate with computers as if computers were like people who are rather stupid and of lower social status.

In an experiment using the Reverse Wizard of Oz scenario, lexical alignment was investigated by Branigan (Branigan 2009, cf. Pearson et al. 2006): 83% of alignment occurred when people believed they were interacting to a computer, which was the truth, and 44% of alignment occurred when people believed they were interacting to a human, which was not true as they were interacting with a computer.

Similarly, in a second experiment, an advertisement of an older dialogue system for $10, and a new system from 2003 for $299, there was 80% of alignment with a basic version of a program, and 42% of alignment with an advanced version of the program.

These experimental results suggest that people align more with computers than with people, and apparently they transfer their beliefs about people they align less with to

computers: they also align more with stupid computers than with more smart ones (or rather computers that they think are stupid or smart).

2.2.6 Alignment, coordination, and situation models

All of the views discussed so far leave many issues open, in particular the functionality of alignment: what actually is successful communication? In the following sections a number of issues in this area will be discussed briefly, mainly based again on Pickering & Garrod (2004).

According Clark (1985), dialogue is a joint activity and coordination is similar in other coordinated activities, such as ballroom dancing or with lumberjacks using a two-handed saw. An obvious case which is not mentioned by Pickering & Garrod or Clark is in some kinds of sports such as tennis, baseball, football, boxing, wrestling.

According to another approach, coordination occurs when interlocutors share the same linguistic representation at some level (Branigan et al. 2000; Garrod & Anderson 1987).

Pickering and Garrod (2004) prefer to call the first case ‘coordination’ and the second case ‘alignment’. Alignment occurs at a particular level when interlocutors have the same representation at that level. So dialogue is coordinated, but also aligned. But it is not clear whether there are other alignment levels in the other activities which are coordinated. This is not discussed by Pickering & Garrod.

Pickering & Garrod (2004) continue their discussion of alignment by introducing situation models and relating them to other alignment concepts:

1. Alignment of situation models (Zwaan & Radvansky 1998) forms the basis of successful dialogue.
2. The way that alignment of situation models is achieved is by a primitive and resource-free priming mechanism.
3. The same priming mechanism for situation models produces alignment at other levels of representation, such as the lexical and syntactic.
4. Interconnections between the levels mean that alignment at one level leads to alignment at other levels.

5. There is another primitive mechanism allows interlocutors to repair misaligned representations interactively.
6. More sophisticated and potentially costly strategies that depend on modelling the hearer's beliefs are only needed if the primitive mechanisms do not succeed in producing alignment.

On this basis, they propose their own version of the Interactive Alignment account of dialogue alignment.

In a dialogue system, the users are in a certain situation which has to be modelled. A situation model as introduced by Pickering & Garrod is described as a multi-dimensional representation of the situation under discussion (Johnson-Laird 1983; Sanford & Garrod 1981; van Dijk & Kintsch 1983; Zwaan & Radvansky 1998). According to Zwaan and Radvansky, the key dimensions encoded in situation models are space, time, causality, intentionality, and reference to main individuals under discussion. This is clearly relevant for the current research.

Although Pickering & Garrod criticise approaches which propose two situation models, one for the speaker and one for the hearer, because they are too complicated and inefficient. But the criterion of complexity and efficiency are not clear. For a dialogue system in which new information has to be communicated, this criticism is not justified. There are also other situations in which two models may be needed: for deception, lying, hiding confidential information. Therefore full alignment of the situation models may not be possible. Lack of alignment also occurs when misunderstandings happen. So misalignment may have to be tolerated, and error-correction mechanisms may be needed.

In the present study, the central questions will be tackled: how (or to which extent) the dialogue system can align with the key dimensions of the situation model, namely space, time, causality, intentionality, and reference to main individuals under discussion.

If the system in the emergency call centre is able to align to these dimensions with a high degree of accuracy, then it should be able to put the appropriate priority to the phone call and classify the call, as well as following instructions about the emergency location: this is situation model alignment. The situation model provides a set of features for the TRIGGER_SITUATION part of the model presented in the Introduction.

In an extreme case if two people are in very different associations, such as a stressed caller and a call-centre employee, or if two people come from different cultures and speak different languages, it is still possible for them to align their situation models through explicit negotiation (Brennan & Clark 1996; Clark & Wilkes-Gibbs 1986; Garrod & Anderson 1987; Schober 1993). According to Pickering & Garrod (2004), the global alignment of the situational models seems to result from the local alignment at the level of the linguistic representations being used, and they propose that this kind of alignment works via a priming mechanism: If a hearer hears an utterance that activates a particular representation, then priming creates an expectation that makes it more likely that the hearer will subsequently produce an utterance that uses that representation when he takes on the speaker role. This kind of interactive priming becomes an essential part of Pickering & Garrod's approach to alignment.

The starting point for the Pickering & Garrod approach was apparently Garrod and Anderson (1987), who introduced a principle of output/input coordination: in a maze game task, players tended to make the same semantic and pragmatic choices that held for the utterances that they had just heard. In other words, what they said, i.e. their outputs, tended to match what they heard, i.e. their inputs at the level of the situation model. During the course of interaction the semantic and pragmatic representations used for producing output and processing input became aligned. The studies provide (cf. Garrod & Anderson 1987, Brown-Schmitt et al. 2005) evidence for alignment of situation models in comprehension.

The conclusion to be drawn for the present study is the interesting fact that if there is a factor constraining the speaker's situation model, it also constrains the listener's situational model.

2.2.7 Levels of alignment

In the Introduction, alignment was defined with reference to different levels of language, and in the literature relations such as repetition and imitation are mentioned in this connection. Transcriptions of dialogues (see the corpus linguistic study in Chapter 4) contain numerous number of repeated linguistic elements and structures, which indicates that there is alignment not only of the situational model, but also at other levels (Aijmer

1996; Schenkein 1980; Tannen 1989). As Pickering & Garrod point out, the following levels may become aligned during dialogue:

1. Lexicon: the same expressions tend to be used while referring to particular objects; the expressions become shorter an more similar when used with the same interlocutor and get modified if the interlocutor changes (Brennan & Clark 1996; Clark & Wilkes-Gibbs 1986; Wilkes-Gibbs & Clark 1992).
2. Syntax: interlocutors tend to use the same syntactic structures (Branigan et al. 2000)
3. Phonetics: the articulation of interlocutors' repeated expressions becomes increasingly reduced, i.e. the expressions developed during a dialogue are shortened and harder to recognise when heard in isolation. Additionally, interlocutors tend to align accent and speech rate (Giles et al. 1992; Giles & Powesland 1975).
4. Semantics and pragmatics: some evidence on comprehension was provided by Levelt and Kelter (1982, Experiment 6) in which subjects were presented with the question-answer pairs and their task was to assess their naturalness. Pairs in which repeated form was used got the best scores. This suggests that people prefer to get responses aligned with their own form.

Pickering and Garrod (2004) say that in successful dialogue the interlocutors develop aligned situation models and aligned representations at all linguistic levels. Additionally, priming at one level leads to priming at other levels.

However, Pickering and Garrod are not very precise about the formal properties of semantic alignment, and they do not underline the importance of alignment on the semantic level being essential for successful communication. Also, they do not deal with cooperative non-alignment, where one person is stressed and the other person does not align but tries to persuade the first to align on the stress-free person, and which is required for scenarios in the present research.

2.2.8 Error trapping with misalignment

An important activity in dialogue is error trapping, in this case recovery from a state of misalignment, when the interlocutors interpretations of utterances differ, for instance with ambiguities. In dialogue it happens that people use the same name, but they think of two different people. These interlocutors align on the *superficial* level, but their situation model is misaligned. In such cases the interlocutors need to use recovery mechanisms which will help them establish alignment, i.e. establish who is the person they refer to.

The recognition of errors and the treatment of errors is a necessary property of a spoken dialogue system.

2.3 Communicative signs: function and processing

Communication uses signs, and alignment means the alignment of signs with all their properties which are involved in communication. Alignment processes cover syntax, semantics and pragmatics. Therefore understanding what alignment is also depends on understanding what a sign is.

The de Saussure sign model (1913) is shown in Figure 2, which shows the meaning-form (signifié-signifiant) relation, which de Saussure sees as a mental relation between the concept and the sound image. The picture in the middle illustrates the relation.

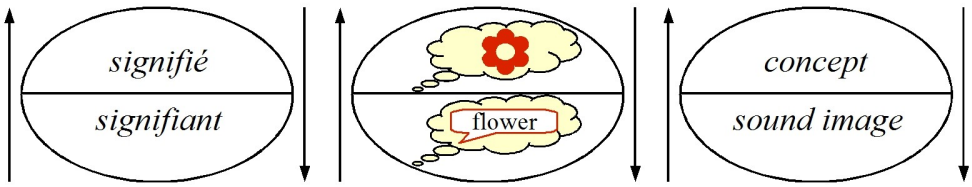


Figure 2: The Saussurean sign model

2.3.1 Levelt & Schriefers's 'sign pie'

The Levelt & Schriefers (1987) model, which is known as the 'sign pie', has three components, unlike de Saussure's model, which has two components. The third component is syntax, which answers a criticism of de Saussure's model (and the models of Bühler and Jakobson) which do not explicitly contain a syntax component. The sign pie model, which is also a mental model, is visualised in Figure 3.

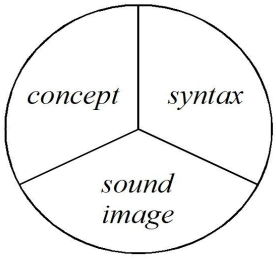


Figure 3: Levelt & Schriefers's 'sign pie' (1987:396).

Levelt & Schriefers 1987: 396) point out:

> An item's syntactic properties always play a crucial role in the sentence generation process. They determine the syntactic environments that must be realized if that item is to be used, and these in turn impose constraints on the syntactic properties of further items to be retrieved. Or to put it differently: where concepts clearly serve as input for lexical access in speech production, yielding sound images as output, syntax plays both input and output roles.

Examples of the importance of syntax are found with prepositions, which may depend more on grammatical relations than on meaning relations. The Levelt & Schriefers model is used as the basis for a model of activation in communication, as shown in Figure 4.

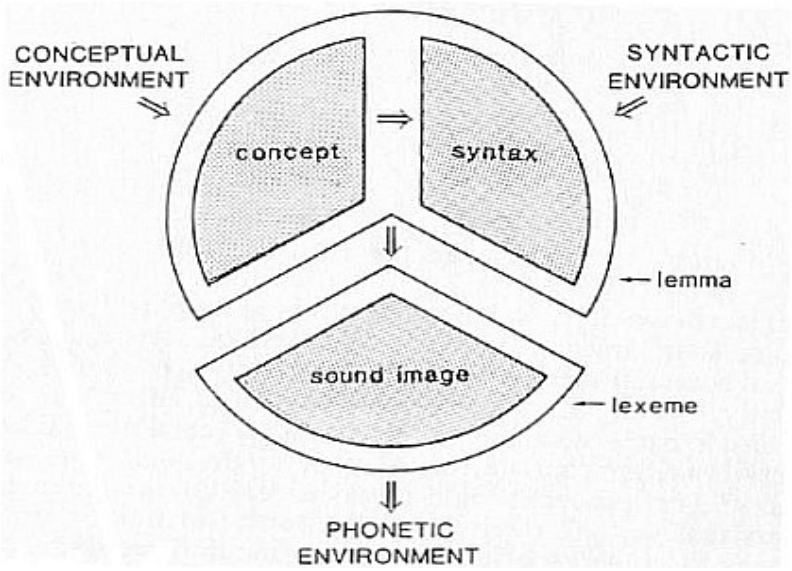


Figure 4: Levelt & Schriefers image of the activation of a linguistic sign in speech production (Levelt & Schriefers 1987: 396)

The extended model of Levelt and Schrievers shows a move from the language-as-product view of traditional sign models to the language-as-action approach which is

necessary in psycholinguistics and speech technology. Pickering & Garrod comment on the language-as-product tradition as follows:

> The language-as-product tradition is derived from the integration of information-processing psychology with generative grammar and focuses on mechanistic accounts of how people compute different levels of representation. (Pickering & Garrod 2004: 170)

They point out that in the language-as-action tradition

> utterances are interpreted with respect to a particular context and takes into account the goals and intentions of the participants. This tradition has typically considered processing in dialogue using apparently natural tasks (e.g., Clark 1992; Fussell & Krauss 1992). (Pickering & Garrod (2004: 170)

Finally they compare the two traditions:

> Whereas psycholinguistic accounts in the language-as-product tradition are admirably well-specified, they are almost entirely decontextualized and, quite possibly, ecologically invalid. On the other hand, accounts in the language-as-action tradition rarely make contact with the basic processes of production or comprehension, but rather present analyses of psycholinguistic processes purely in terms of their goals (e.g., the formulation and use of common ground; Clark 1985; Clark 1996; Clark & Marshall 1981). (Pickering & Garrod (2004: 170)

Although Pickering & Garrod claim that the product approach is not relevant for alignment, this is not true in the context of computation. A product is at the same time a result of processing, and also an input for processing. In spoken dialogue, one product (for example a situation model) is changed into another product (a modified situation model) by processing. So the two approaches are not as incompatible as Pickering & Garrod claim.

The Levelt model is extended in other work. The Levelt production model has three main components, and is the planning, formulation (with two subcomponents) and articulation components (Table 2).

Table 2: Processing modules in speech generation and their relation to phases of lexical access (Levelt & Schriefers 1987: 398)

Processor	***Input***	***Output***	***Relation to Lexical Access***
Conceptualiser	communicative intention	preverbal message	creating a lexical item's conceptual conditions
Grammatical encoder	preverbal message	surface structure	retrieval of lemma, i.e. making the item's syntactic properties available, given appropriate conceptual or syntactic conditions
Sound form encoder	surface structure	phonetic or articulatory plan for utterance	retrieval of the lexeme, i.e. the item's stored sound form specifications, and its phonological integration in the articulatory plan
Articulator	phonetic plan	overt speech	executing the item's context-dependent articulatory program

The Formulator (Figure 5), which is the most relevant component in this context, is characterised as follows by Levelt (1992): In speech production the formulator is described as a process whose input is the lexical concept (the message) and whose output is a phonetic or articulatory plan for the item. The appropriate item for the mental lexicon is selected and is integrated into the developing grammatical encoding. An articulatory program is created for the selected lexical item on the basis of its stored phonological code and the phonological context of the utterance as a whole.

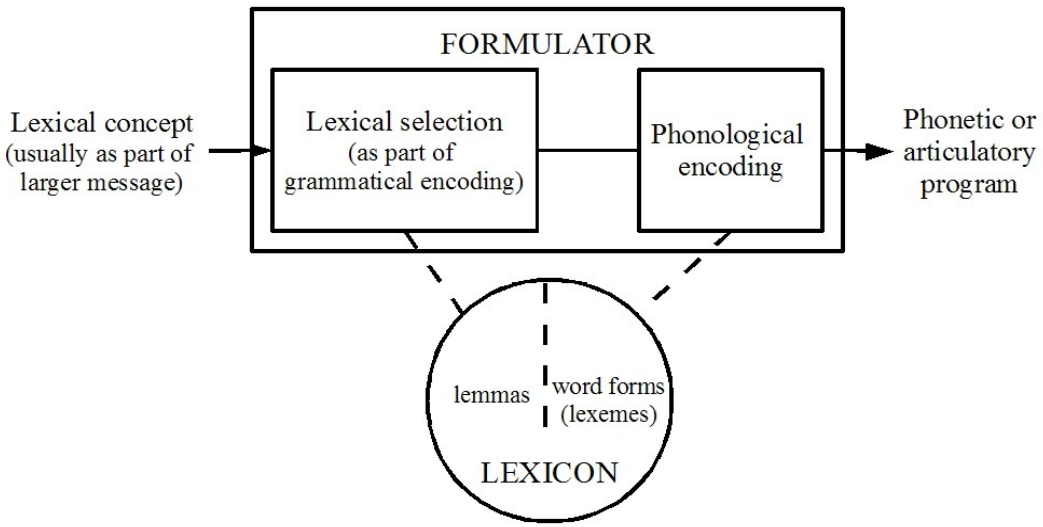


Figure 5: An outline of lexical access in speech production (Levelt 1992: 4)

2.3.2 The revised Interactive Alignment model of dialogue processing

According to Pickering and Garrod (2004: 175)

> the interactive alignment model assumes that successful dialogue involves the development of aligned representations by the interlocutors. This occurs by priming mechanisms at each level of linguistic representation, by percolation between the levels so that alignment at one level enhances alignment at other levels, and by repair mechanisms when alignment goes awry.

Figure 6 illustrates the alignment process. The linguistic levels of two interlocutors are linked. In Figure 6, A and B represent two interlocutors in a dialogue in this schematic representation of the stages of comprehension and production processes according to the interactive alignment model. The horizontal links show the channels by which alignment takes place at these levels by means of the Priming mechanism, including lexical priming, syntactic priming, etc.

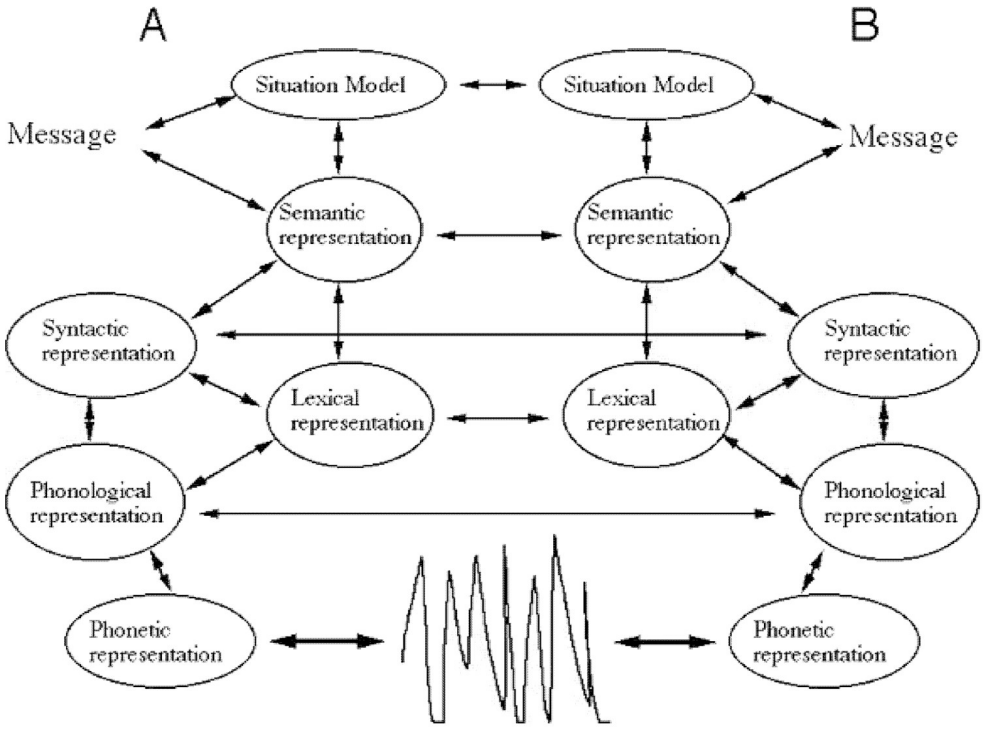


Figure 6: Schematic representation of the stages of comprehension and production processes according to the interactive alignment model (Pickering & Garrod 2004: 176)

The interactive alignment model does not apply in this way to monologues, whose goal is not to become aligned with the listener, although indirect alignment (with previously experienced communication) may occur. In a monologue the speaker tries to formulate the message in such a way that the listener can obtain the appropriate representation corresponding to the speaker's message. The important fact is that in monologue (including writing) the speaker's and the listener's representations may never align, the automatic mechanism of alignment is not present. The alignment mechanism occurs only when the speaker gets regular feedback from the interlocutors and on the basis of this he or she can control the alignment process. In dialogue, priming is the central mechanism in the process of alignment and mutual understanding. Thus dialogue indicates the important functional role of priming (Pickering and Garrod 2004). The process of interactive alignment by priming is supported by further factors:

1. The use of routine procedures in dialogue.
2. The use of implicit common ground (background knowledge which is assumed to be shared) and explicit common ground (which is mentioned in the dialogue).

Pickering & Garrod discuss an alternative model, the autonomous transmission model, in which the transfer of information between producers and comprehenders takes place via decoupled production and comprehension processes that are isolated from each other (see Figure 7). Communication takes place only through the acoustic medium and the messages are constructed independently by the speaker and the hearer.

Pickering and Garrod (2004) say that the autonomous transmission model is not appropriate for dialogue. In dialogue the production and comprehension processes are coupled and this is the core of the interactive alignment model.

However, it is not clear how the interactive alignment model can be represented in a precise model: the horizontal connections between levels do not exist independently of the physical signal transmission. Therefore, contrary to what Pickering and Garrod claim, the interactive alignment processes at different levels in the overall alignment procedure can only be reconstructed from an autonomous transmission model of physical contact via speech.

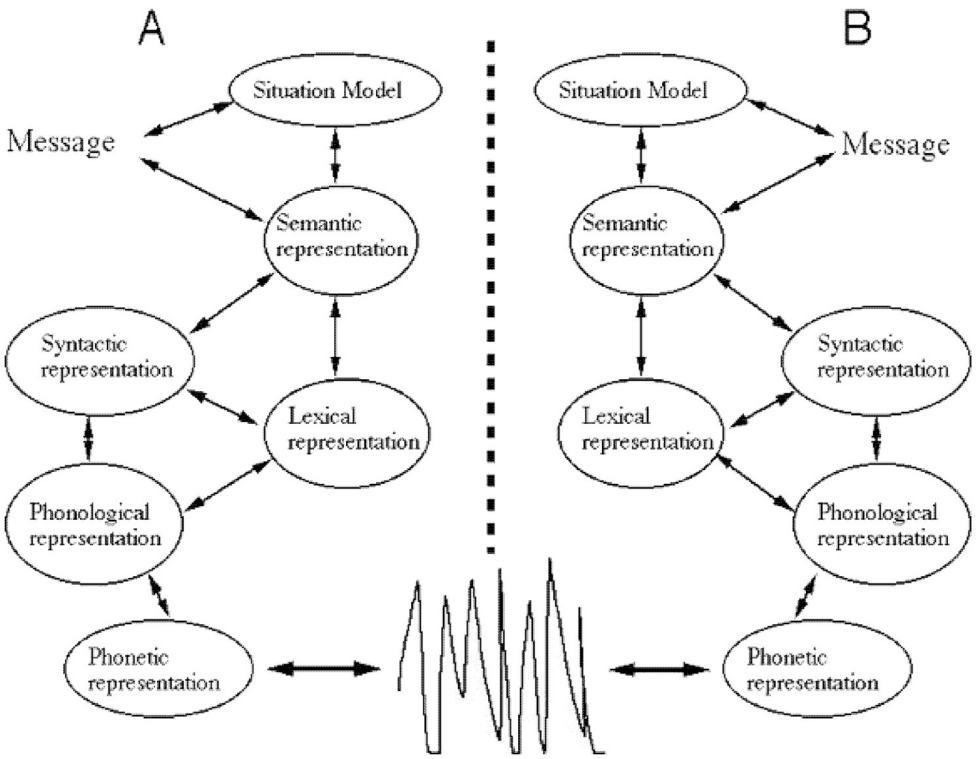


Figure 7: Schematic representation of the stages of comprehension and production processes according to the autonomous transmission account (Pickering & Garrod 2004: 177)

2.4 Speech acts and dialogue acts

Austin (1962) presented two theories of speech acts. In the first theory, he distinguishes between constative and performative utterances. Constative utterances can be true or false, as in traditional propositions. Performative utterances cannot be true or false, but perform some action, for example questions, commands, promises, etc. In the second theory, the functions or 'force' of utterances were treated, and it was claimed that there are no basic distinctions between constative and performative utterances, which all share certain types of force:

1. locutionary force (propositional content of utterance – predicates and arguments),
2. illocutionary force (conventional use of utterances to create social links between the interlocutors), "doing things with words" - "hereby" = "niniejszym", illocutionary verbs = speech act verbs,

3. perlocutionary force (effect the utterance has on the hearer).

The forces are indicated by language forms and structures:

1. word order,
2. stress,
3. intonation contour,
4. punctuation,
5. the mood of the verb,
6. the so-called performative verbs (e.g. ‘say’, ‘tell’, ‘confess’, ‘promise’, ‘warn’, ‘baptise’).

Searle (1969) extended and modified Austin’s theory and developed 9 constitutive rules which define successful utterance, which define the role of the speaker and his beliefs about the hearer in producing a successful utterance, for which he distinguishes between utterance acts (produced words), propositional acts (assigning meaning to the utterance acts) and illocutionary acts (similar to Austin’s ‘illocutionary force’). Searle gives the example of ‘promise’:

> Given that a speaker *S* utters a sentence *T* in the presence of a hearer *H*, then, in the literal utterance of *T*, *S* sincerely and non-defectively promises that *p* to *H* if and only if the following conditions 1-9 obtain. (Searle 1969: 57)

Searle’s formulation of the felicity conditions for promising (1969) are:

1. Normal input and output conditions obtain.
2. S expresses the proposition that p in the utterance of T.
3. In expression that p, S predicates a future act A of S.
4. H would prefer S’s doing A to his not doing A, and S believes H would prefer his doing A to his not doing A..
5. It is not obvious to both S and H that S will do A in the normal course of events.
6. S intends to do A.
7. S intends that the utterance of T will place him under an obligation to do A.
8. S intends (i-1) to produce in H the knowledge (K) that the utterance of T is to count as placing S under an obligation to do A. S intends to produce K by means of the

recognition of i-1k, and he intends i-1 to be recognized in virtue of (by means of) H's knowledge of the meaning of T.

9. The semantical rules of the dialect spoken by S and H are such that T is correctly and sincerely uttered if and only if conditions 1-8 obtain.

Grice extended speech act theory with his conversational maxims (1975), also called Maxims of Cooperation:

1. Maxims of quantity
    1. Make your contribution as informative as required.
    2. Do not make your contribution more informative than is required.
2. Maxims of quality
    1. Do not say what you believe to be false.
    2. Do not say that for which you lack adequate evidence.
3. Maxim of relation
    1. Be relevant.
4. Maxims of manner
    1. Avoid obscurity of expression.
    2. Avoid ambiguity.
    3. Be brief.
    4. Be orderly.

It is true that not all discourse is cooperative, but discourse in emergency dialogue situations has to be cooperative.

Dialogue acts (Bunt 2000) are a further extension of speech act theory, and will be dealt with in the context of the corpus linguistic study in Chapter 4.

> Dialogue acts are the smallest functional units of dialogues, and are utterances corresponding to speech acts such as 'greetings', 'request', 'suggestion', 'accept', 'confirm', 'reject', 'thank', 'feedback'"(Gibbon et al. 2000:6).

The functional annotation of dialogues, referred to as dialogue act annotation, is a means of capturing and encoding different levels of discourse structure, and identifying how they relate to one another at a pragmatic level; what was the speaker's intention and what their pragmatic effect on the dialogue was.

2.5 Summary

In this chapter, the Pickering and Garrod (2004) theory was presented with necessary criticism from the point of view of the speech technology and the scenarios required in the present research. The first criticism is that Pickering and Garrod are not precise about the formal properties of semantic alignment The need for recovery from the misalignment process and for implementing a misalignment recovery technique in the dialogue system was presented. The second criticism of Pickering and Garrod's theory (2004) is that it only deals with cooperative alignment. The present research does not deal with non-cooperative alignment, but it deals with cooperative non-alignment to some extent, between a professional call-centre operator and a caller, in which a call-centre operator may remain calm and try to influence a stressed caller to align with him.

Finally, it was proposed that the alignment concept should be extended to include models of signs and that communication uses signs, therefore alignment means alignment of signs with all their properties. Relevant aspects of sign theories from the Saussurean sign model to dialogue acts were presented, and speech act rules which ensure success of communication were outlined.

Chapter 3: Dialogue modelling

A number of conclusions can be drawn from the descriptive criteria discussed in the previous chapter.

First, the language-as-product approach is suitable for the description of the structural components of language: lexicon, grammar (including text structure), phonology.

Second, the language-as-action approach deals with two essential features for a spoken dialogue system: it covers processing models for accessing and using the language-as-product components, and also it deals with dialogue as action, with the focus on the context, the goals and the intentions of the participants in their interaction. These three features can be interpreted in the emergency dialogue system as follows:

1. Context focus: a person is calling the emergency call centre to ask for help and has knowledge of the geography of the area so he can guide an ambulance to the emergency location.
2. Goals: the caller wishes to get help, and the emergency system has the goal of assessing the threat, assigning priority to the call, and calculating an appropriate route to the emergency location.
3. Intentions: the caller intends to cooperate with the emergency system in order to achieve successful cooperation, and the emergency system has the intention of cooperating with the caller in discovering the problem and a route to the emergency location.

Cooperation is a central feature of many kinds of communication, and is an essential feature of this kind of emergency dialogue system with cooperation between human and computer. The challenging task of the system designer is to model the human-computer interaction in such a way that the computer is able to *sufficiently* understand the meaning of the dialogue in order to reach the goals, even if understanding is not *perfect*.

An important aspect of cooperative communication is alignment. In the following chapter a corpus linguistic study will be described, with the aim of finding patterns in dialogue which can be used to model typical dialogues, and of identifying stimuli which trigger alignment behaviours of particular types in humans. Based on this investigation a model of the human-computer dialogue will be developed.

The aim is to model the alignment between the participants in such a way that the dialogue system will not 'work on its own' independently from the human's lexical, syntactic and emotional behaviour input, but will try to *align with* the speaker's linguistic representations, and, what is the main concern of this study, *align with* the speaker's emotion manifestation behaviours revealing information about the speaker's emotional states and let the system *coordinate* the dialogue to make the dialogue a 'joint activity.'

The components of language which are required in a language-as-product approach are also required in a dialogue system, but the main levels of alignment which are dealt with in this study are the semantic and pragmatic levels: in order to reach the communicative goals, the user's semantic and pragmatic representations will have to become aligned with the system during a process of negotiation. The negotiation is on two levels: first, a semantic level, in aligning knowledge of the route to the emergency location; second, a pragmatic level, in securing uptake and avoiding misunderstandings.

Therefore, when trying to build a comprehensive dialogue system which will align its speech with a caller, it is necessary to design a model which will be able to fulfil the following local goals:

1. Find and recognise the lexical items used by the caller.
2. Compare the lexical items with the dictionary of synonyms available in the system in order to align at the lexical level.
3. Align the situation models (i.e. what the user is talking about) to verify the TRIGGER_SITUATION.
4. Find and recognise the syntactic structures used by the caller.
5. Identify phonological and prosodic features of the caller's voice, such as the accent, intonation, speech rate.

6. Check markers of stress and formality/informality in order to draw conclusions about the caller's state, the STYLE_MANIFESTATION.
7. Adapt the lexical items and syntactic structures used by the caller in order to apply the correct speaking mode of the speech synthesiser based on the verification of the TRIGGER_SITUATION and the STYLE_MANIFESTATION possibilities.
8. Recover from misalignment.

In the spoken language dialogue demonstration prototype which is developed in the present research, a comprehensive system is not the aim; this was emphasised in the introduction. The aim is to pick basic components and demonstrate the methodology involved. In the demonstration dialogue system prototype, points #1, #3, partly #6 and #7, and fully #8 will be addressed. Points #2, #4 and #5 are not addressed as they would require building a comprehensive voice-in-voice-out spoken dialogue system in which human speech would be input to the system. If only #2 and #4 hold, then the input to the system can be text. A comprehensive system would require building a complex syntax parser which is not the subject of the present work.

It is not clear from the literature how the interactive alignment model can be represented in a precise computational model: the horizontal connections between levels certainly do not exist independently of the physical signal transmission. Therefore, contrary to what Pickering and Garrod (2004) appear to claim, the interactive alignment processes at different levels in the overall alignment procedure can only be reconstructed from an autonomous transmission model of physical contact via speech.

The dialogue system discussed in this thesis would have to predict the caller's knowledge and determine the common ground between system and caller. To do that a set of TRIGGER_SITUATIONs needs to be built containing possible situations about which the user may be talking. Additionally, an extensive dictionary of domain specific vocabulary needs to be designed to make sure the system will know the *standard* form of relevant *colloquial* expressions. For example, greetings and apologies may be involved, and in moments of frustration the users in stressful situations may use obscenities. Such words must be included in the dictionary of the system, although the system itself should use their neutral synonyms, i.e. the system should not align to taboo expressions, but

remain on a formal level of style as far as possible. There may also be name misalignment: the same place, such as a reference building or a square in a town, may have many different names used by the local people.

Therefore an essential part of dialogue system design is to make it build up the extensive implicit common ground with its interlocutor. This means that the more information the system and the user share in their dialogue, the more effective conversation is and the more their situation models are aligned. From a semantic point of view, the common ground can be created by using a map, whose semantic relevance has to be negotiated in relation to reality (e.g. there may be temporary blocks due to building construction). From a pragmatic point of view, the theory about common ground and implicit common ground assumes that there are interactive repair mechanisms using implicit and full common ground when the interlocutors' representations are not properly aligned.

The basic strategies for making the implicit common ground more explicit may be repetitions with rising intonation, a repetition with additional query (e.g. if the emergency location is in a shop, which shop?), or a restatement (e.g. reformulating "two chicks" by "two girls"). Such reformulations in dialogues in order to secure uptake are also called *clarification requests* (Ginzburg 2001). These reformulations do not require the speaker to take into account the listener's situational model. They reflect the fact that the listener fails to understand what the speaker is saying in relation to the listener's situation model. The system should therefore be able to generate the clarification requests when something is not clear or if the vocabulary used by the speaker is not in the synonym dictionary. Then the system will try to obtain utterances which it can understand.

The pragmatic alignment procedure of interactive repair using the full common ground in human-human interaction is used when the basic repair using implicit common ground fails, for example, when two people are talking about John, but the listener does not know whether the speaker means John Smith or John Brown. Then the listener has to refer to the full common ground saying "He knows both, but thinks I do not know Brown, so it has to be Smith" (Pickering & Garrod 2004). The use of full common ground also happens when the speaker tries to deceive the listener or wants to conceal information from the

other (e.g. Clark & Schaefler 1987), i.e. when the communication is not cooperative, and when the other speaker tries to uncover the goals of the first speaker.

In a dialogue system the use of full common ground may happen only to the caller. It is hard to imagine that ta computer system, at least at the current state of knowledge, will use the interactive repair procedure of using full common ground. The system will certainly have limitations, however well designed. In the case of the caller, a human interlocutor can infer that the system may not know the vocabulary, names, places mentioned by him and therefore may try to adapt to the limitations of the system by using the standard or most common words. If communication becomes too misaligned for the computer to repair, control must be handed to a human operator. This level will not be dealt with in the present work.

3.1 Dialogue systems

A dialogue system is a computer system capable of interactive turn-by-turn communication with a human user. (Gibbon et al. 1997). Nowadays dialogue systems use many different media to communicate with the human user. There are system which employ as input and output such media as text, speech, graphics, haptics, gestures and combination of those to create multi-modal dialogue systems.

For example, while writing this text the keyboard and the mouse are used as the physical input devices to the computer. This activates the hand muscles in a human being to generate the input. Further, the information sensed by the computer input devices can be processed at different levels of abstraction, providing different levels of the understanding of the intention of the user. Then the computer outputs the processed input on the screen or to the loudspeaker to communicate with the human user. On the screen the human sees text or figures and can hear audio via the loudspeakers. Finally, the computer output stimulates human senses like vision and hearing (Gibbon et al. 1997).

The best well know human dialogue system which was developed in 1966 was ELIZA (Weizenbaum 1966). The intention of the designer, Joseph Weizenbaum, was to parody a Rogerian therapist, mainly by rephrasing the patient's statements as questions and posing them to the patient. The system was able to generate statements which seemed to be very

thoughtful and deep and able to convince the interlocutor that the system was a human being.

The dialogue systems can be categorised as follows according to the modality dimension:

1. Text-based – systems whose communication is based on text input and output.
2. Spoken dialogue systems – systems which communicate with the user via speech.
    1. Spoken language dialogue systems are modelled on human-human conversation in which spoken natural language is used.
    2. Voice Response Units (VRU) – systems much simpler than the spoken language dialogue systems and restricted to a narrow domain and producing canned speech. An example of such a system could be a weather forecast service (Gibbon et al. 1997).
3. Graphical user interface – systems which apply images rather than textual commands for human-computer communication.
4. Multi-modal – systems which combine more than one modality to interact with the human user.

In the present work, the author refers to the dialogue system as a kind of a text-in-voice-out dialogue system.

3.2 Dialogue system components

The basic components of a dialogue system are:

1. Dialogue manager – coordinates the dialogue, adopts certain dialogue strategies and decides on the responses to the user, often implemented as a Finite State Automaton.
2. Natural language understanding (NLU) – processes the input so that it is comprehensible to the machine, for example by identification of proper names, tagging parts of speech and parsing.

3. Input recogniser/decoder – makes the input readable to the computer and may be implemented as automatic speech recogniser, gesture recogniser, handwriting recogniser, etc.
4. Output generator – produces data for the output render component.
5. Output render – produces output which can be implemented as text-to-speech synthesiser, talking head or a robot.
6. Multi-modal fusion – combines together different output media (Liao 2002)

3.3 Spoken dialogue systems

A spoken dialogue system is a dialogue system whose channel of communication is voice. The most essential modules of the system are speech synthesiser for generating speech and speech recogniser for processing the input human speech. The other components are as in the other dialogue systems, so dialogue manager and natural language understanding natural language generator.

The spoken dialogue systems may be of different types. The following spoken dialogues systems are in use nowadays (Wikipedia):

1. Informational: providing information about time, whether, bank accounts, credit cards.
2. Transactional: allowing magazine subscription or bank transactions.
3. Diagnostic: providing technical support.
4. Educational/tutoring: supporting language learning or physics.
5. Entertainment and chatting.

3.4 Human-computer interaction

Human–computer interaction (HCI) studies interaction between human users (HU) and computers. Figure 8 shows a model of human-computer interaction on an abstract level. Communication between the human and the computer is held on several modalities (or channels as it is named in the figure). Humans employ one or more modalities to communicate with another human, also for communication with computers. In the human-

computer communication, the input modalities are constrained by what the computer can process. The computer generates output to the HU choosing among the available media. This computer output is interpreted by the HU on different modalities (or channels) based on human cognitive skills. The loop from human output-input is the auto-feedback loop which is continuously active in the human-human communication. Depending on the kind of the system, the computer may use different size of the feedback channel. For example, in Virtual Reality applications such as advanced technology computer games or flight simulators, the computer will maximise the size of the bandwidth being used for feedback (Gibbon et al. 2000).

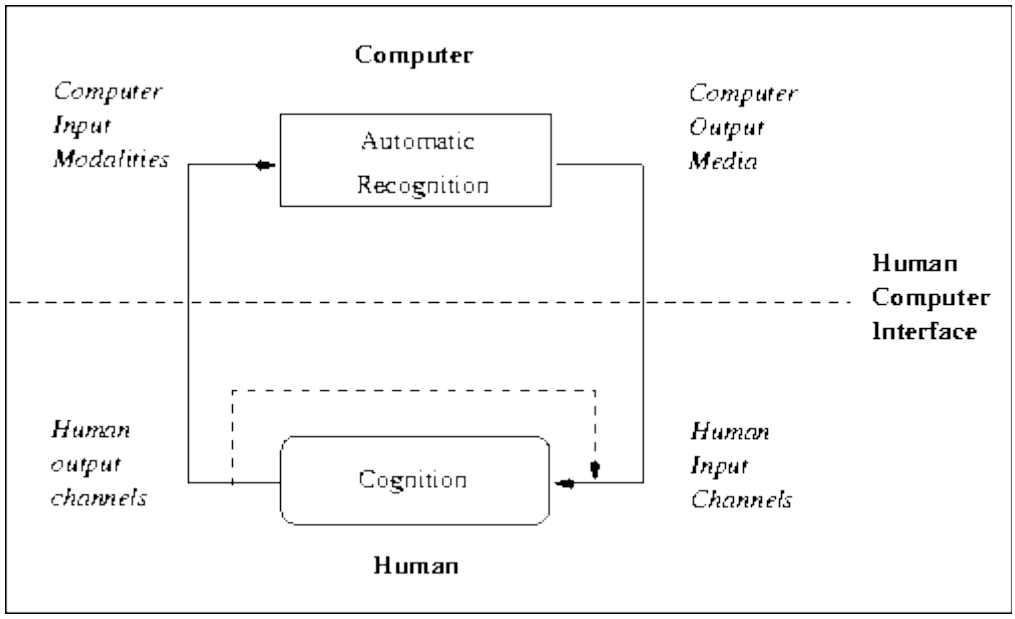


Figure 8: A model of human-computer interaction (Schomaker et al. 1995, from Gibbon, Mertins & Moore 2000)

3.5 Summary

In the present chapter, dialogue modelling was briefly discussed and requirements of a comprehensive emergency-call-centre dialogue system which would align with the caller's speech were specified. From those requirements, optimal components for the demonstration dialogue system prototype were selected. Additionally, relevant aspects of dialogue systems were discussed and specifications for human-computer interaction were presented.

Chapter 4: Corpus linguistic study of dialogue interaction

4.1 Chapter overview

In this chapter, for examining the details of alignment and dialogue act theory, a corpus linguistic study and its results are presented. For the study, the map-task dialogues from the PoInt corpus (Karpiński 2002) are used. Two dialogues (18min of speech) are annotated on the dialogue act level using the selected dialogue act categories from Bunt DIT++ categories (2008). An excerpt from the dialogue is annotated on the phone level for the creation of synthetic voices, and for the Automatic Close Copy Speech (ACCS) synthesis of the dialogue for testing the annotations. The chapter discusses a thorough dialogue analysis for the use of dialogue finite-state-automata creation presented in the next chapter.

4.2 Aim of the corpus linguistic study

The aim of this corpus linguistic pilot study is to find techniques of analysing dialogue as recorded and annotated speech material. The study is oriented at practical work with the corpus of dialogues. Although at this stage only two dialogues have been analysed, and most of the works is done on only one dialogue, the techniques being developed and the findings being uncovered will be very useful when dealing with a big corpus of dialogue recordings.

The assumption is that it is possible to create a dialogue scenario which would serve as a model for a dialogue system. The idea is to find the speakers' techniques to carry out the dialogue, solve problems and sustain communication. The aim is to describe a dialogue and its most frequent dialogue act sequences and communication techniques.

In this pilot study the following are looked at:

1. dialogue annotation at the dialogue act level
2. annotation of turns – dialogue flow
3. finite state automata of dialogue act sequences
4. most frequent dialogue acts sequences

In this study some useful techniques were developed, as well as some incorrect and less-appropriate ways of dialogue analysis were uncovered. Those techniques will need to be improved in the analysis of a big corpus of dialogues and some approaches will need to be changed.

A preliminary study of expressive speech synthesis (Bachan & Surmanowicz 2008) was made using synthetic 'microvoices', i.e. voices which do not cover the whole language but only a restricted set of utterances which are relevant for the study. The methodology introduced there is extended in the present study, but the content is different, since the present study is concerned with public and private styles in stress situations, not with standard emotion sets. The present study is also concerned with developing microvoices for testing purposes.

An additional feature of the present study is that it was designed to provide information for developing formal dialogue models (which will be discussed in detail in the following chapter. The models involve Regular Grammars implemented as Finite State Automata which model interaction management: input analysis, dialogue management, output generation.

A distinction which needs to be considered in the corpus linguistic study design is the difference between reactive systems and initiative systems.

4.3 Speech material - PoInt corpus

For the preliminary analysis an existing corpus of dialogues was used, extracted from the Corpus of Polish Intonational Database (PoInt) (Karpiński 2002). The PoInt corpus contains 3 types of speech recordings: read speech, semi-spontaneous monologues and dialogues. There are 3 different types of dialogues in which the participants try to accomplish different tasks: (a) to guess who is on a presented picture by asking a limited number of polar questions; (b) to perform a special version of the "map task"; (c) to discuss a controversial topic in order to reach and present a common conclusion.

The sound material was recorded digitally on optical discs (CD-R), using Tascam CD-R700 recorders, Shure Beta Series microphones, AKG condenser microphone, and a Spirit Folio F1 mixing console. The sessions, recorded as cda files, were converted into Windows WAV format (16-bit/44.1kHz) and copied to a hard disk. Dialogues were recorded in stereo

mode, although the speakers and their microphones were not completely acoustically isolated. The signals were normalized to ca. 97% of the available dynamics range. (Karpiński 2002).

From the dialogue corpus, the map-task dialogues were chosen for the present pilot study.

4.4 Annotation

4.4.1 Annotation procedure

The data were annotated using the Praat phonetic workbench software tool (Boersma & Weenink 2001). A Praat window displaying the stereo speech signal of the dialogue and its annotation tiers is shown in Figure 9.

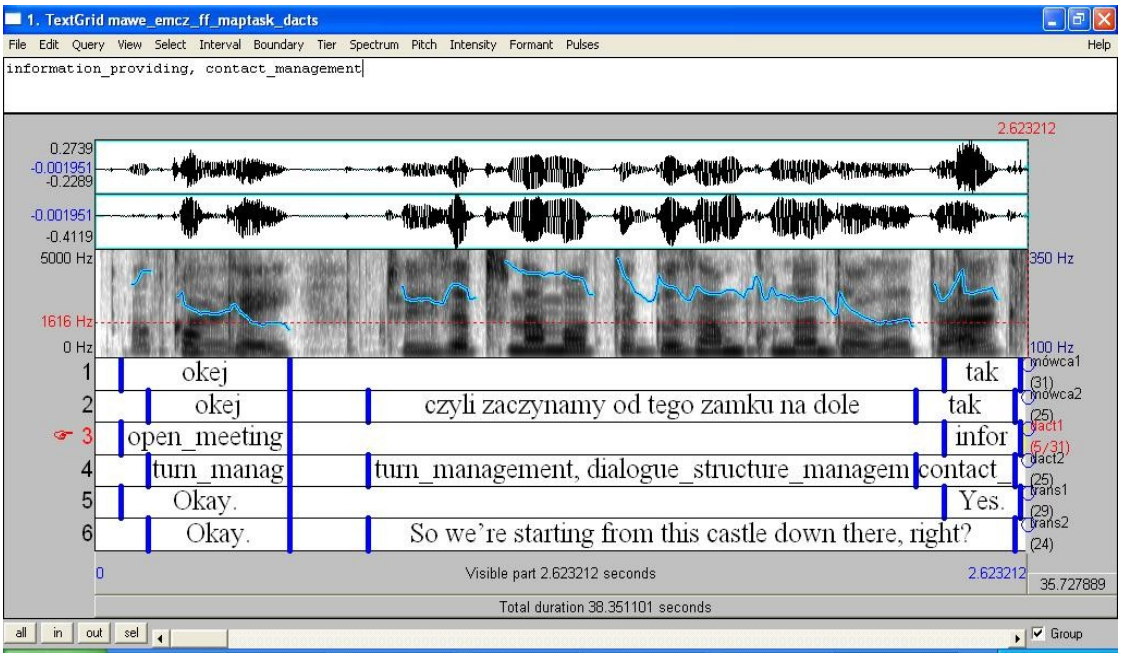


Figure 9: The Praat window displaying the stereo speech signal of the dialogue with its annotation tiers

The dialogues in the PoInt corpus were annotated orthographically in Polish and the translation in English on a separate tier for each speaker was provided. The speakers' turns

were divided into utterances which corresponded to dialogue acts. However, in the available corpus the dialogue act annotation was not provided.

For the present research two kinds of annotation is required:

1. dialogue act annotation – for creating dialogue models as FSA,
2. phonemic annotation – for MBROLA micro-voice creation and for ACCS synthesis (Bachan 2007a, Bachan 2008).

These levels of annotation were performed by the author of this work.

4.4.2 Dialogue act annotation

For the preliminary analysis, two dialogues (18 min) between two females and a male and a female were annotated on the dialogue act level. The dialogue act categories for the annotations were selected from Bunt's main categories of Dynamic Interpretation Theory, DIT (Bunt 2000, cf. Gibbon 2009). The DIT main categories and their functions are:

1. General Purpose communicative functions
    1. Information transfer
        1. Information seeking
        2. Information providing
    2. Action discussion functions
        1. Commissives, e.g. promise
        2. Directives, e.g. dismissal
2. Dimension-specific communication functions
    1. Activity-specific functions
        1. Open meeting
        2. Bet
        3. Congratulation
        4. …
    2. Dialogue control functions
        1. Feedback
            1. Auto-feedback
            2. Allo-feedback
        2. Interaction management

1. Turn management
2. Time management
3. Contact management
4. Own communication management
5. Partner communication management
6. Discourse structure management, e.g. topic shift
7. Social obligations management
    1. Salutation
    2. Self-introduction
    3. Apologising e.g. prayer gesture
    4. Gratitude expressions e.g. thumbs up gesture
    5. Valediction

More than one dialogue act category was assigned to a speaking turn, because one utterance can have more than one communicative function. For the dialogue act analysis, the *multi-layered* labels of dialogue acts were reduced to *one-layered* labels, i.e. only the first dialogue act was preserved, e.g. infpr_dir → infp. Table 3 shows the abbreviations of 12 dialogue act functions chosen from Bunt's categories and used in the annotation of the selected dialogues.

Table 3: Abbreviation of dialogue act functions

Abbreviation	***Dialogue act function***
allo	allo-feedback
auto	auto-feedback
cnt	contact management
dir	directives
infpr	information providing
infsk	information seeking
open	open meeting
own	own communication control
partner	partner communication management
social	social obligations management
time	time management
turn	turn management

Unfortunately, at the time of the annotation creation, Bunt's category set was not illustrated by actual examples, so for the present study the category set had to be empirically interpreted. Bunt also did not relate his categories explicitly to previous work in speech act and discourse theory where turn-taking patterns and speech act sequencing

are also dealt with. Bunt's categories released for example in (2010) include examples of actual dialogue acts in the natural language.

4.4.3 Phonemic annotation

In order to create MBROLA micro-voices and perform ACCS synthesis (Bachan 2007a) annotation on the phone level had to be added. An excerpt of a dialogue (7sec) between two females was selected and the stereo signal was split into two mono channels, one for each speaker. Then the excerpt of dialogue was annotated using the automatic tool SALIAN (Szymański & Grocholewski 2005) integrated into the Annotation Editor program (Klessa 2006). Because SALIAN takes speech signal and its transcription as input, for the selected excerpt a TXT file containing the transcription of the utterances was created. The speakers were not isolated from each other during the recording and the speech of each of them is clearly heard on the channel of one another. For the purpose of the automatic annotation, the utterances of the other speaker were deleted from the annotated channel (Praat function: *Edit → Set selection to zero*) (Boersma & Weenink 2001). Additionally, the SALIAN software is unable to annotate long speech files, therefore the speech excerpt was limited to 7sec.

The automatically generated annotations were then manually checked against the processed recordings for each speaker on corresponding channels.

4.4.4 Processing of annotations for dialogue analysis

To analyse the dialogue for Finite State Automata (FSA) creation the information on the dialogue act annotation and utterance transcription tiers from one map-task dialogue recording was extracted and analysed. The material was prepared using Linux scripts[1] in the following manner:

1. The TextGrid file was saved as plain text. All the special characters used in the Polish language were replaced by the ASCII characters.
2. Dialogue act annotation tier and utterance transcription tier for each speaker were extracted.

1 The scripts were either the author' Python and Perl scripts or were Shell scripts provided by Dafydd Gibbon to whom the author is grateful for making them available

3. Dialogue acts annotation tiers and utterance transcription were merged to create a matrix which is presented in Appendix A.
4. Dialogue acts sequences were divided into 49 *parts* and the beginnings and ends of those parts were used to determine the initial and terminal states of the automata. Those divisions into *parts* are marked in grey in the table in Appendix A and they indicate the first silence after the last dialogue act in a sequence overlapping with the other speaker's speech, which means that the start/end of the *part* may occur within speaker's turn if the speaker made a pause within his turn. Figure 10 illustrates the division of the dialogue into *parts* starting from the beginning of the dialogue till the 51 second.

Such designed matrices were then used for creating automata for each of the speakers separately for the whole dialogue. The silent pauses were not taken into account and are not generated by the automata.

4.4.5 Notes on material preparation

Longer work on the automata and the dialogue itself raised questions and doubts about whether the way of division of the dialogue into *parts* was the appropriate one. Having a closer look at the pauses being chosen for the end of the *part* and at the same time indicating the initial and terminal states of the automata seems not to be correct. Examples on Figure 10 show that the borders 1, 2, 3, 5 and 9 are defined at the end of the turns of both speakers (indicated by '-' on both sides of the border for each of the speakers). However, the borders 4, 6, 7 and 8 split speaker's turn, for example at a time when the speaker stopped talking to take a breath. Additionally, the part between borders 4 and 5 is very long and built on many dialogue acts. At this part there are many places where none of the speakers speaks, so it should be possible to define a border there if a different approach was taken. The most appropriate examples of the borders are borders 1, 3 and 9 where not only both speakers ended their turns, but also the turn was grabbed by the other speaker and he started speaking after the pause.

Such observations will be taken into account in the future work. However, this pilot study is based on the matrices created according to the specifications outlined above.

4.5 Time structure of the dialogue

A computational analysis of the annotation files shows that the time relations between the utterances are very complex. The relations are not purely sequence relations but involve overlaps in time.

In Figure 10 the temporal division of the dialogue into parts is shown. At the first silence after the last dialogue act in a sequence overlapped with the other speaker's speech, the bars represent dialogue act intervals (chunks of speech), the indices indicate indexed borders and the dash indicates the of a turn (silence). The diagram is based on the first 51 seconds of the dialogue.

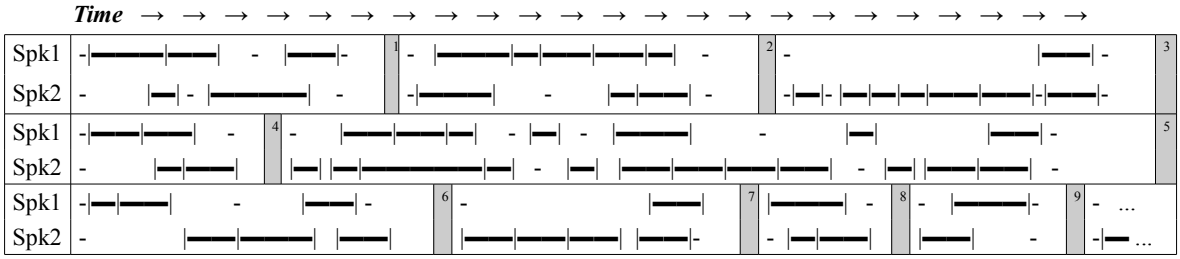


Figure 10: Temporal sequences and overlaps in a dialogue

In preparation for dialogue act automata creation, the dialogue was divided into 49 *parts* indicated by the first silence after the last dialogue act in a sequence overlapped with the other speaker's speech. For the 49 *parts*, 49 simple automata were designed for each speaker generating the dialogue act sequences which occurred in the analysed dialogue. After the multi-layered labels reduction, there were 49 dialogue act sequences for each speaker with one-layered labels, many of which were the same. Table 4 shows the length of the shortest and the longest dialogue act sequences and the frequency of dialogue acts for both speakers in the studied material. The min, max and mean figures correspond to lengths of dialogue act sequences and the other figures present the frequency of dialogue acts for both speakers

In Table 4 the basic information about the map-task dialogue may be read. Speaker's 2 utterances were longer than speaker's 1 on average, as after reduction the mean dialogue act sequence length was by 0.45 longer for speaker 2. Both speakers provided information most frequently. Additionally, speaker 1 used auto-feedback, contact-management and information seeking dialogue acts very often, whereas speaker's 2 speech was

characterised by using time management dialogue acts frequently. Nearly half of all the dialogue act sequences were composed of only one or two dialogue acts in the studied material (cf. column n ≤ 2).

Table 4: Basic statistics of the studied material; N – number of sequences, n ≤ 2 – number of sequences with the length of one or two dialogue acts

Spk	***N***	***n ≤ 2***	***min***	***max***	***mean***	***allo***	***auto***	***cnt***	***dir***	***infpr***	***infsk***	***own***	***partner***	***social***	***time***	***turn***	***All***
1	49	21	1	10	3.2	2	28	27	2	43	27	2	3	0	9	14	**157**
2	49	22	1	10	3.65	3	17	4	14	79	6	3	3	1	37	12	***179***

4.6 Most frequent dialogue act sequences

The following sections describe the dialogue analysis for the purpose of developing a formal dialogue model for an operational text-in-voice-out dialogue system.

4.6.1 Dialogue initiation

Dialogue initiation could not be studied at this stage, because the whole research is based on one dialogue only and it is not enough to make general comments about the issue. Additionally, the real open meeting dialogue acts were cut out as it was not of the interest of the project for which the material was made (the PoInt project, Karpiński 2002). However, the available material shows that the beginning of the map-task dialogue is quite chaotic. The speakers at the beginning try to organise the dialogue, set their roles and start their tasks.

4.6.2 Dialogue termination

Dialogue termination was also not studied, as there was only one dialogue on which the present research is based. What can be said is that one of the speakers closed the session by the social obligation management dialogue act, i.e. thanking the other participant for his cooperation.

4.6.3 Turns

In order to study the turns, the dialogue was examined again and one more annotation mark was added – the additional mark was to signal the boundary of the dialogue turn. The turn

boundary marks were set at the pauses which signalled the end of the turn. As a result, there were 4 different types of pauses classified in the dialogue:

1. normal pauses – these are silences which were found in the dialogue and were made within speaker's turn because the speaker had to think or take a breath, such pauses do not have a special annotation mark;
2. termination pauses – these are pauses made because the speaker finished his turn and gave the turn to the other interlocutor, such pauses are marked with '-' sign;
3. forced-termination pauses – one of the speakers had to stop talking because the other interlocutor interrupted him, however the melody of his speech suggests that the speaker wanted to continue and not to end his turn; such pauses are marked with '+-' sign;
4. inter-termination pauses – the speaker ended his turn, but the other interlocutor did not start talking, so the speaker continues, such pauses divide the chunk of speech into two turns although the other interlocutor did not actually start talking, such pauses are marked with '--' sign.

At this stage of research different types of turn termination were not studied. However, in the future it would be good to differentiate between turns which end up because of different reasons such as the natural handing over of the turn or the interruption. Both, the intonation of such turns as well as the dialogue act sequences should be looked at.

4.7 Frequency of dialogue acts

Table 5 shows the number of turns distinguished in the dialogue for both speakers and the number of dialogue acts in the whole dialogue. The table shows that speaker 1 had more turns than speaker 2 in the dialogue, but speaker 2 had more to say as the numbers of the multiple and single dialogue acts suggest this and the duration on the overall speech tells. One turn may be divided into speech *chunks* and each of these *chunks* may represent different communicative functions. The number of *all dialogue acts* in the table shows the number of dialogue acts when to one speech *chunk* one or more than one communicative function is assigned, so for example one *chunk* of speech may represent "information

providing" and "directives" communication functions. Actually, the number of *all dialogue acts* stands for the number of speech *chunks* to which communicative functions were assigned. The number of *single dialogue acts* shows the number of all different communicative functions which appeared in the dialogue, so for example if the *chunk* of speech represented both "information providing" and "directives", then it would be counted as 2 dialogue acts. The number of s*tudied dialogue acts* shows only the dialogue acts which were studied in the present study, so the dialogue acts which appeared in the following positions:

1. at turns composed of a single speech chunk (column *Single*);
2. initial position of a turn at turns which were composed of more than one speech chunk (column *Start*);
3. final position of a turn at turns which were composed of more than one speech chunk (column *End*).

The figures of *studied dialogue acts* stand for the number of 'single' dialogue acts as it was counted in the column *single dialogue acts*. Whereas the figures in the columns *single*, *start* and *end* show the actual number of positions at which the dialogue acts were counted.

Table 5: Dialogue act length

Speaker	***Turns***	***All DA***	***Time***	***Mean***	***Max***	***Min***	***Single (separate) DA***	***Studied DA***	***Single***	***Start***	***End***
Spk 1	128	233	189.08s	0.81s	3.56s	0.08s	297	232	75	53	53
Spk 2	110	351	356.58s	1.02s	5.72s	0.09s	496	259	35	75	75

According to the quantitative data in Table 5 it can be concluded that speaker 2 had more to say in the dialogue. Evidence for this is not only the speech duration, but also although speaker 1 has more turns, speaker's 2 speech was divided in more chunks and there were much more dialogue acts assigned to his speech. Additionally, speaker 2 had fewer single-utterance-turns (only 35) and all the others were composed of more than one speech chunk. These data support the view that the leader (here speaker 2) in a map-task dialogue speaks much more than the instruction follower (here speaker 1). The same view supports the frequency of different dialogue acts presented in Table 6. 35% of the dialogue acts are directed to provide auto-feedback or manage the contact and the other 35% of

dialogue acts seek for information in speaker's 1 speech. Whereas 45% of speaker's 2 dialogue acts provide information and 23% are categorised as being directives. Only 7% of all dialogue acts provide auto-feedback and manage contact. Also the frequency of the turn management acts is higher for speaker 2 and may suggest that whenever the speaker wanted to say something, he wanted to communicate that he would like to speak for a longer period of time, whereas speaker 1 did not want to take the initiative and his turns were probably reduced to auto-feedback.

Table 6: Frequency of different dialogue acts in the whole dialogue for both speakers

Speaker 1			***Speaker 2***		
allo-feedback	3	1,01%	allo-feedback	5	1,01%
auto-feedback	44	14,81%	auto-feedback	24	4,84%
contact_management	57	19,19%	contact_management	10	2,02%
directives	8	2,69%	directives	114	22,98%
information_seeking	106	35,69%	information_providing	222	44,76%
information_providing	40	13,47%	information_seeking	10	2,02%
open_meeting	1	0,34%	social_obligation_mng	1	0,20%
own_communication_mng	5	1,68%	own_communication_mng	6	1,21%
partner_communication_mng	4	1,35%	partner_communication_mng	3	0,60%
time_management	15	5,05%	time_management	63	12,70%
turn_management	14	4,71%	turn_management	38	7,66%
	297	100,00%		496	100,00%

The frequency of dialogue acts at certain positions in dialogue turns is shown in Table 7. M means a stand alone utterance which constitutes a turn, Start – the start of a turn in turns built by more than one speech chunk and E – the end of a turn in turns built by more than one speech chunk. If one utterance is defined as having more than one communicative function, they are calculated as separate dialogue acts. By looking only at bigger numbers some similarities and differences may be noticed. First of all, information providing dialogue acts are frequent for both speakers at all three positions in a turn. Information seeking acts are more frequent at the end of the turn for both speakers, and occur frequently as stand-alone questions in speaker's 1 speech. Time and turn dialogue acts occur mostly at the beginning of a turn and hardly ever constitute stand alone turns. Directives are mainly given by speaker 2, and auto-feedback and contact management is provided by speaker 1. In the following sections dialogue acts at different positions in a turn will be discussed.

Table 7: Number of dialogue acts at the beginning (S) and end (E) of dialogue act sequences in a turn, and single turns (M) build by one utterance; o - open meeting, s - social communication management

	allo		***auto***			***cnt***			***dir***			***infpr***			***infsk***			***o***	***own***		***part***		***s***	***time***			***turn***			***All***
	E	***M***	***E***	***M***	***S***	***E***	***M***	***S***	***E***	***M***	***S***	***E***	***M***	***S***	***E***	***M***	***S***	***S***	***E***	***M***	***M***	***S***	***E***	***E***	***M***	***S***	***E***	***M***	***S***	
Sp1	1	1	1	30	10	12	31	11	4	1	3	26	23	23	15	10	5	1	1	0	3	1	0	0	1	6	0	2	10	232
Sp2	0	0	1	5	7	3	3	2	28	19	13	52	22	38	7	1	1	0	1	1	1	1	1	7	1	20	2	2	20	259

Research on turn initiation discusses dialogue acts which appeared at the beginning of a turn which was long enough to be divided into two or more chunks of speech, where each chunk of speech was assigned its own communicative functions. Bunt's category 'Turn-unit-initial functions' covers three subcategories: turn accept, turn grab, and turn take, but they are not distinguished within this work. A turn may start with a filled pause which represent the time management communicative function before the speaker starts providing information. In such example the turn is divided into two speech chunks: turn initiation (time management) and turn continuation or turn termination (information providing). In this section the turn initiation dialogue acts will be analysed.

Table 8 and Figure 12 shows frequency of different dialogue acts at the beginning of a turn. The same data presented in a pie chart is shown on Figure 11. The data shows that the occurrence of the information providing dialogue acts is the same frequent for both of the speakers at the beginning of a turn. Time and turn management dialogue acts are more frequent for speaker 2; especially the time management communicative function is dominant in speaker's 2 speech which may be characteristic for the speaker who often starts his turn with "uhm..." The biggest differences, however, are for the auto-feedback, contact management, information seeking and directives acts. The first three are characteristic for speaker 1 who follows the instructions, whereas directives are common for speaker 2, the instruction giver in the map-task dialogue.

Table 9 presents the multi-layered dialogue acts, i.e. all the communicative functions which were assigned to the chunk of speech at the beginning of a turn. In the table we can see the main differences between the speakers such as:

1. Speaker 1:
    1. combines auto-feedback with contact management communicative functions
    2. provides and seeks for information at the same time

3. is more confident taking the turn

2. Speaker 2:
    1. starts a turn with time management dialogue acts
    2. gives informative directives very often, not only directives

Table 8: Number of different dialogue acts at the beginning of a sequence in a turn

DA	*auto*	*cnt*	*dir*	*infpr*	*infsk*	*open*	*partner*	*time*	*turn*	*All*	*Turns*
Spk1	10	11	3	23	5	1	1	6	10	70	53
Spk2	7	2	13	38	1	0	1	20	20	102	75

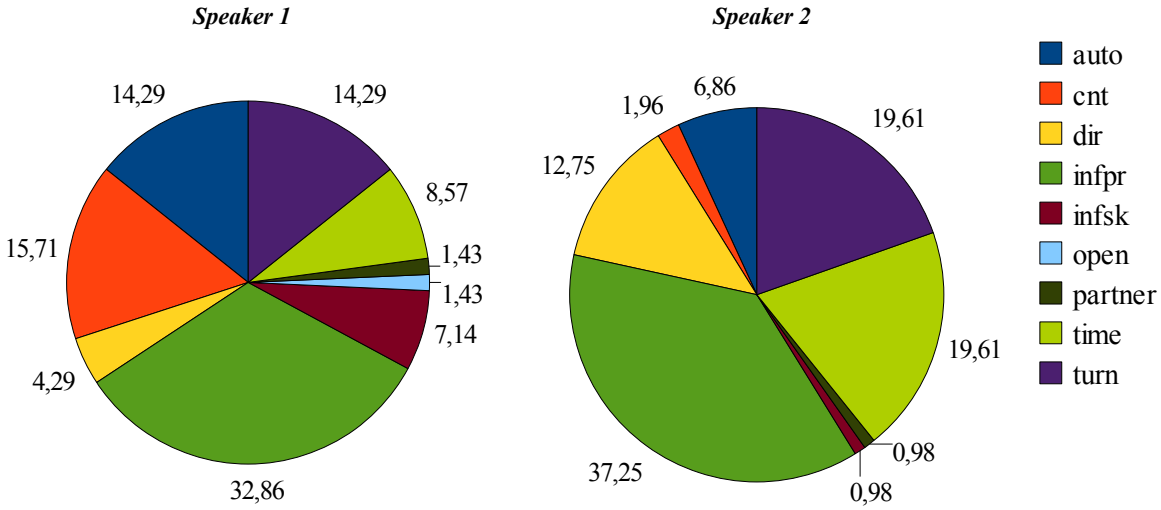


Figure 11: Percentage representation of frequency of dialogue acts at the initial position in a turn

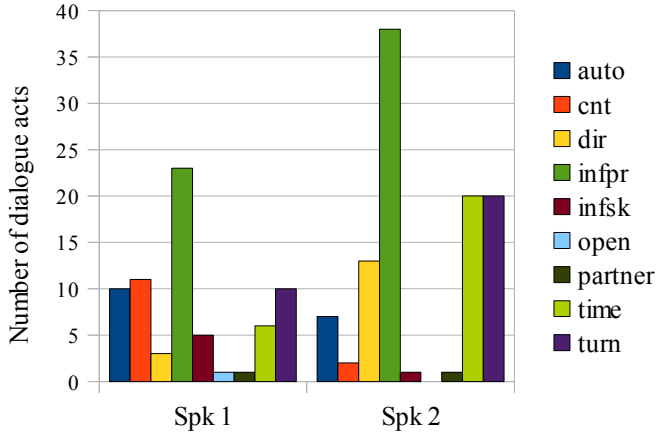


Figure 12: Number of different dialogue acts at the beginning of a sequence in a turn

Table 9: Dialogue acts at the beginning of a turn for speaker 1 and speaker 2

Speaker 1		***Speaker 2***	
DA	***N***	***DA***	***N***
auto-feedback	6	auto-feedback	6
auto-feedback, contact_management	4		
auto-feedback, contact_management, turn_management	1	auto-feedback, turn_management	1
contact_management	3	contact_management	1
contact_management, information_providing	1		
directives	1	directives	4
information_providing	13	information_providing	17
information_providing, directives	2	information_providing, directives	9
information_providing, information_seeking	2		
information_providing, turn_management	4	information_providing, turn_management	13
information_seeking	3		
partner_communication_management	1	partner_communication_management	1
time_management	6	time_management	16
turn_management	4	time_management, contact_management	1
turn_management, contact_management, information_providing, directives, open_meeting	1	time_management, turn_management	5
turn_management, directives	1	turn_management	1
	53		**7**

Turns composed of one utterance, or more precisely – one speech chunk, may have more than one communicative function. In this section single-utterance turns will be looked at. Table 10, Figure 13 and Figure 14 show different representations of the frequency of dialogue acts at single-utterance turns. The numbers indicate that speaker 1 utters mostly short utterances and these are mainly auto-feedback and contact management dialogue acts. Relatively often the speaker provides or seeks for information in such short turns. On the other hand, speaker 2 is much more frequent in providing information or giving directives.

Table 10: Number of different dialogue acts at a single-utterance turn, with time measurements; Dur – duration, Avg – average length

	allo	***auto***	***cnt***	***dir***	***infpr***	***infsk***	***own***	***partner***	***time***	***turn***	***All***	***Turns***	***Dur***	***Avg***
Spk1	1	30	31	1	23	10	0	3	1	2	102	75	50.5s	0.67s
Spk2	0	5	3	19	22	1	1	1	1	2	55	35	47.84s	1.37s

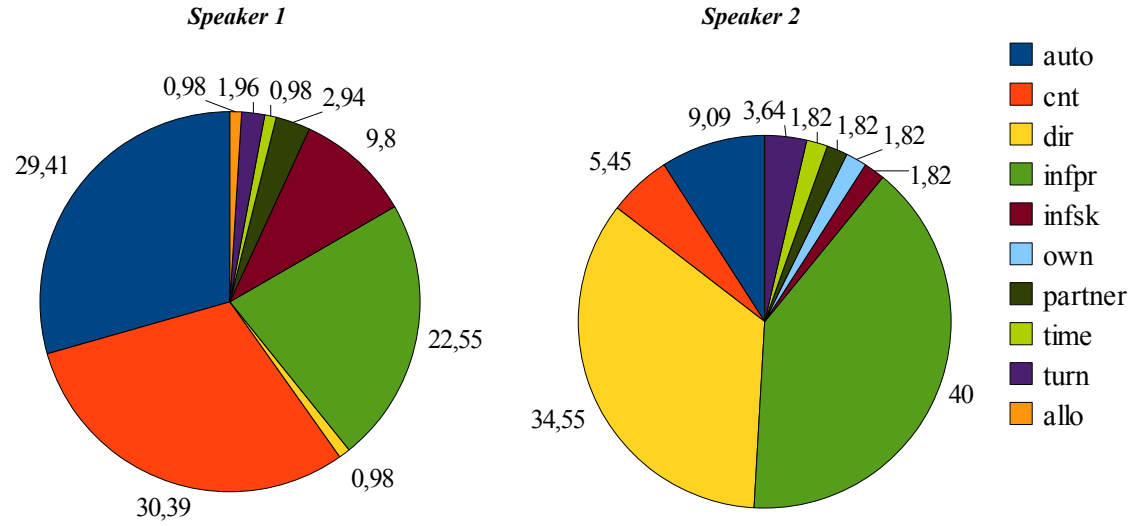


Figure 13: Percentage representation of frequency of dialogue acts in single-utterance turns

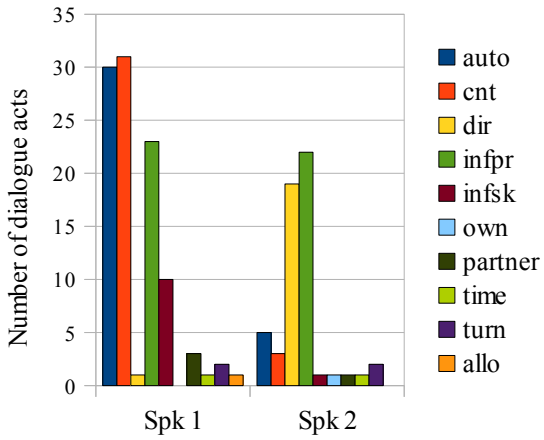


Figure 14: Number of different dialogue acts at a single-utterance turn

Table 11 presents the multi-layered dialogue acts, i.e. all the communicative function which were assigned to the single-utterance turns. In the table the main differences between the speakers can be seen, such as:

1. Speaker 1:
    1. provides auto-feedback which is often combined with the contact management function

2. purely asks for information or provides information, the short utterances do not have other communicate functions

2. Speaker 2:
    1. provides information and gives directives at the same time
    2. rarely asks for information

Table 11: Dialogue acts of single-utterance turns for speaker 1 and speaker 2

Speaker 1		***Speaker 2***	
DA	***N***	***DA***	***N***
auto-feedback	11	auto-feedback	4
auto-feedback, contact_management	16	auto-feedback, contact_management	1
auto-feedback, contact_management, time_management	1		
auto-feedback, information_providing	2		
contact_management	12	contact_management, information_providing	1
		directives	5
information_providing	15	information_providing	5
information_providing, contact_management	2	information_providing, directives	13
information_providing, information_seeking	1	information_providing, turn_management	1
information_providing, partner_communication_management	2		
information_seeking	10	information_seeking	1
		own_communication_management	1
partner_communication_management	1	partner_communication_management	1
		time_management	1
turn_management	1	turn_management, directives	1
turn_management, information_providing	1		
	75		**35**

Summarising, single-utterance turns are characteristic for speaker 1, the instruction follower. The speaker provides a lot of auto-feedback and informs the interlocutor that is ready to get and process new information by generating contact management dialogue acts. On the other hand, speaker 2, the instruction giver, rarely produces short turns and if so, these are mainly information providing or directives dialogue acts.

Research on turn termination looks at dialogue acts which appeared at the end of a turn which was long enough to be divided into two or more chunks of speech, where to each chunk of speech was assigned its own communicative functions. Bunt's category 'Turn-unit-final functions' covers three subcategories: turn assign, keep, release – but they are not distinguished within this work. Table 12, Figure 15 and Figure 16 are different representations of the frequency of different dialogue acts at the terminal position in a turn. Both speakers tended to finish their turns by providing information. Other frequent and

expected communicative functions are: information seeking, contact management and in a smaller degree directives in speaker's 1 speech. Whereas in speaker's 2 speech directives prevail, but also time management and information seeking are quite frequent.

Table 13 presents the multi-layered dialogue acts, i.e. all the communicative functions which were assigned to the chunk of speech at the end of a turn. In the table we can see the following differences between the speakers such as:

1. Speaker 1:
    1. provides information, but at the same time gives a sign that he is looking for confirmation by using the information seeking function.
2. Speaker 2:
    1. gives directives in a 'polite way' making them very informative – speech where information providing is combined with directives is problematic to categorise, because although the speaker provides information, the follower feels that he should follow the route being given by the speaker, without any special indication that following the route is what the listener should be doing;
    2. is more uncertain and gives the listener signs that he needs more time to think on what to say. The time management function was not present at all in speaker's 1 speech at this position.

The analysis of dialogue acts and the differences between speakers' speech is not only caused by the speakers' different roles in the dialogue, but speakers' personalities. The speaker decides whether to be strict and give pure orders, or cover his directives by a veil of information.

Table 12: Number of different dialogue acts at the end of a sequence in a turn

	allo	***auto***	***cnt***	***dir***	***infpr***	***infsk***	***own***	***social***	***time***	***turn***	***All***
Spk1	1	1	12	4	26	15	1		0	0	60
Spk2	0	1	3	28	52	7	1	1	7	2	102

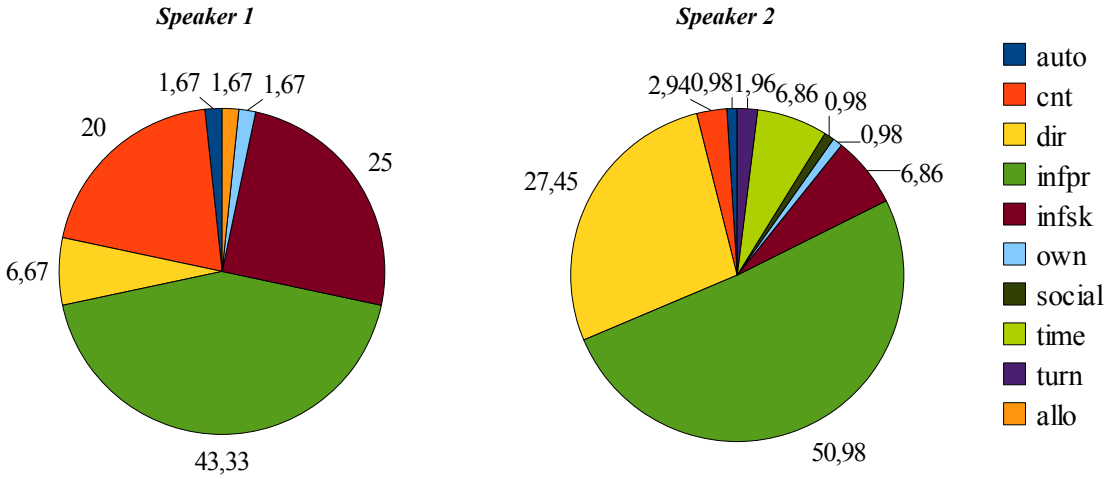


Figure 15: Percentage representation of frequency of dialogue acts at the final position in a turn

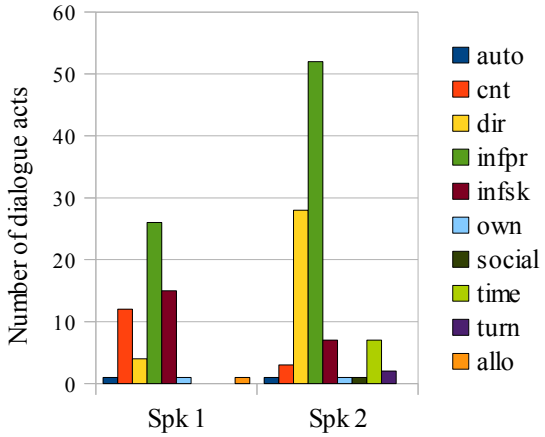


Figure 16: Number of different dialogue acts at the end of a sequence in a turn

Table 13: Dialogue acts at the end of a turn for speaker 1 and speaker 2

Speaker 1		***Speaker 2***	
DA	***N***	***DA***	***N***
auto-feedback	2	auto-feedback	1
contact_management	13	contact_management	1
		contact_management, information_seeking	1
directives	1	directives	6
information_providing	20	information_providing	27
information_providing, directives	2	information_providing, directives	21
		information_providing, directives, time_management	1
information_providing, information_seeking	4	information_providing, information_seeking	1
information_providing, own_coomunication_management	1	information_providing, time_management	1
		information_providing, turn_management	2
information_seeking	10	information_seeking	4
		information_seeking, contact_management	2
		own_communication_management	1
		social_obligation_management	1
		time_management	5
	53		**75**

4.7.1 Dialogue flow

To model a dialogue, it is not only important to study the turns of separate speakers, but also the speech flow between the speakers. The speakers may start speaking while the other speaker has not finished yet, their speech may be separated by a short pause. In both cases, the dialogue speech flow sounds fluent and "aligned".

4.7.2 Overlapping speech

In this section overlapping speech is discussed in terms of the kind of dialogue acts which start being produced by one speaker before the other speaker has finished his turn.

Table 14 presents the overlapping dialogue acts where speaker 2 started talking before speaker 1 had finished his dialogue act (not a turn, as this table shows also examples where the speaker 1 produced longer turns and speaker 2 only provided short replies which were within the timestamps of the long turns of speaker 1). Starting from the middle columns, Table 14 shows the overlapping dialogue acts as a sequence, so for the first row: 'auto-feedback' of speaker 1 overlaps with 'information providing' of speaker 2. N in the left column stands of the actual number of such speaker's 1 dialogue acts overlapping with some dialogue acts of speaker 2. N on the right tells how many sequences of this kind

appeared in the dialogue, so here there were 4 auto-feedback dialogue acts overlapping with 4 information providing dialogue acts. Auto-feedback dialogue acts constitute 7.84% of all dialogue acts which overlapped with speaker's 2 speech. Whereas the 'auto-feedback – information providing' sequence was the only one which started with auto-feedback (shown by 100% in the last column). The next example is more complex as there are more sequences which started with 'auto-feedback, contact management' dialogue act. Here, apart from 'information providing' which occurred 4 times as overlapping with the 'auto-feedback, contact management', there were also 'directives' (3) and 'own communication management' dialogue act (1). Those sequences constitute 50%, 66.67% and 16.67% of the occurrences for dialogue acts which overlapped with 'auto-feedback, contact management'. In this table, the overall is not 100%, because to some overlapping speech chunks more than one dialogue act was assigned. Therefore the 100% entirety has been disrupted – see the last row. However, for the proceeding tables presenting dialogue acts flow, the 100% entirety is sustained.

The analysis of Table 14 starts with the most frequent overlapping dialogue act. For speaker 1 it is 'information providing' which overlaps with speaker's 2 'information providing' dialogue act. 'Directives', 'time management' and 'turn management' also occur very often. Less frequent is 'auto-feedback'.

The next most frequent dialogue act is 'information seeking'. This dialogue act overlaps with 'information providing' most often which means that the interlocutor often started replying before the question was fully produced. Finally, the most frequent overlapping dialogue acts for speaker 1 are 'auto-feedback' and 'contact management'. Those most often overlap with 'information providing' and 'directives'.

Statistical data in Table 14 supports general expectations about dialogue:

1. auto-feedback and contact management dialogue acts overlap with the other speaker's speech as they provide information that the listener is attentive, comprehends what is being said and is willing to obtain more information from the interlocutor – therefore the interlocutor does not wait until the speaker finishes providing auto-feedback or markers for contact management;

2. the dialogue speech and turn change is fluent as can be seen by the number of 'information providing' dialogue acts overlapping with such informative dialogue acts as 'information providing' or 'directives'. Frequent time management and turn management dialogue acts show that the other speaker is also ready to take the turn after being provided with information by the other speaker.
3. 'information seeking' elicits 'information providing'. The interlocutor does not leave the question without an answer.

Table 14: Overlapping dialogue acts: spk 2 starts talking before spk 1 has finished

Speaker 1			***Speaker 2***		
Frequency (%)	***N***	***DA***	***DA***	***N***	***Frequency of the sequence (%)***
7.84	4	auto-feedback	information_providing	4	100
11.76	6	auto-feedback, contact_management	directives	3	50
			information_providing	4	66.67
			own_communication_management	1	16.67
15.69	8	contact_management	directives	2	25
			information_providing	6	75
			turn_management	2	25
1.96	1	directives	auto-feedback	1	100
35.29	18	information_providing	auto-feedback	3	16.67
			directives	5	27.78
			information_providing	6	33.33
			turn_management	5	27.78
			time_management	5	27.78
19.61	10	information_seeking	information_providing	4	40
			time_management	2	20
			turn_management	2	20
			partner_communication_management	1	10
1.96	1	partner_communication_management, information_providing	time_management	1	100
1.96	1	time_management	information_providing	1	100
3.92	2	turn_management	auto-feedback	1	50
			directives	1	50
100	51			60	981.68

Table 15 shows the dialogue acts of speaker 2 which overlapped with dialogue acts of speaker 1, because speaker 1 started talking before speaker 2 had finished his dialogue act. The table is to be read as it was explained for Table 14. The most frequent dialogue acts which overlap are 'information providing', produced by speaker 2, with 'auto-feedback', 'contact management' and 'information seeking'; less frequent is 'information providing'. Much less frequent but noticeable is 'time management, own communication management' of speaker 2 overlapping quite frequently with 'information seeking' of speaker 1. Finally,

in the table ‘auto-feedback’ stands out as being quite frequent. It overlaps with either ‘auto-feedback’ or ‘information providing’.

Statistical data in Table 15 supports general assumptions about the dialogue flow:

1. ‘information providing’ is fuelled by ‘auto-feedback’ and ‘contact-management’ dialogue acts.
2. if information in ‘information providing’ dialogue act is not clear, then the interlocutor asks for information in ‘information seeking’ dialogue acts.
3. in a dialogue interlocutors exchange information as for the ‘information providing’ dialogue act the interlocutor replies with the ‘information providing’ dialogue act.
4. interlocutors help each other solve problems as when one interlocutor encounters problems and try to repair them in ‘own communication management’ dialogue acts, the interlocutor helps him by asking questions in ‘information seeking’ dialogue acts.
5. dialogue flow is smooth as ‘auto-feedback’ overlaps with ‘information providing’.

Table 15: Overlapping dialogue acts: spk 1 starts talking before spk 2 has finished

Speaker 2			***Speaker 1***		
Frequency (%)	***N***	***DA***	***DA***	***N***	***Frequency of the sequence (%)***
7.94	5	auto-feedback	auto-feedback	2	40
			information_providing	3	60
1.59	1	contact_management	information_providing, information_seeking	1	100
7.94	5	directives	information_providing	2	40
			different	3	60
30.16	19	information_providing	auto-feedback	5	26.32
			contact_management	3	15.79
			information_providing	4	21.05
			information_seeking	7	36.84
30.16	19	information_providing, directives	auto-feedback	8	42.11
			contact_management,	5	26.32
			information_providing	2	10.53
			information_seeking	2	10.53
			turn_management	2	10.53
1.59	1	information_providing, information_seeking	turn_management	1	100
6.35	4	information_providing, turn_management	*different*	4	100
1.59	1	information_seeking	information_providing	1	100
1.59	1	own_communication_management	auto-feedback	1	100
1.59	1	partner_communication_management	auto-feedback	1	100
9.52	6	time_management, own_communication_management	information_seeking	3	50
			different	3	50
100	**63**			**63**	**1100.02**

In this section overlapping dialogue acts have been analysed. Statistical data supported some general principles about the dialogue such as:

1. interlocutor provides auto-feedback frequently,
2. information seeking elicits information providing,
3. interlocutors exchange information,
4. interlocutors help each other solve problems.

4.7.3 Non-overlapping speech

In this section the speech flow between the interlocutors is analysed, but here the analysis is on non-overlapping speech where one speaker had finished his speech and a short pause had followed before the other speaker started speaking. The tables in this section are to be read as Table 14 from the previous section.

Table 16 shows the dialogue act flow between the speakers, where speaker 1 has finished speaking, then there was a short silence before speaker 2 started talking. The most frequent last dialogue act before the pause is ‘information providing’. ‘Information providing’ is then followed by either ‘information providing’ or ‘auto-feedback’ most frequently. Very frequent are also ‘auto-feedback’ and ‘contact management’ being followed by ‘information providing’ and ‘directives’. Last but not least, speaker 1 produces an ‘information seeking’ dialogue act and gets either ‘information providing’ in a response or ‘time management’ dialogue act asking for more time to answer the question.

Table 16, although less diverse than Table 14, is very similar in the results which it presents. The main conclusions which can be drawn are:

1. after providing information, the other speaker either provides his own information or provides auto-feedback to show that he is ready to obtain new information.
2. auto-feedback and contact management dialogue acts are signs to the other speaker that he can provide new information or directives.
3. asking questions evokes in the interlocutor the need to provide information or at least to take the turn by giving a sign that he needs more time to formulate the answer.

Table 16: Non-overlapping dialogue acts: spk 2 starts talking after spk 1 has finished

Speaker 1			***Speaker 2***		
Frequency (%)	***N***	***DA***	***DA***	***N***	***Frequency of the sequence (%)***
9.09	5	auto-feedback	information_providing	1	20
			information_providing, directives	3	60
			turn_management	1	20
14.55	8	contact_management, auto-feedback	contact_management, information_providing	1	12.50
			information_providing, directives	5	62,50
			time_management	2	25
20	11	contact_management	information_providing	4	36.36
			information_providing, directives	5	45.45
			information_providing, turn_management	1	9.09
			time_management	1	9.09
41.82	23	information_providing	auto-feedback	5	21.74
			contact_management	1	4.35
			directives	2	8.70
			information_providing	1	4.35
			information_providing, directives	4	17.39
			information_providing, turn_management	6	26.09
			information_seeking	1	4.35
			time_management	3	13.04
14.55	8	information_seeking	auto-feedback, turn_management	1	12.50
			information_providing	1	12.50
			turn_management, information_providing	3	37.50
			time_management	3	37.50
100	**55**			**55**	**500**

Table 17 shows statistical data on non-overlapping dialogue act flow between the speakers. Here the most prevailing are ‘directives’ and ‘information providing’ being followed by ‘auto-feedback’, ‘contact management’ or ‘information providing’ dialogue acts. In the table there is also ‘information seeking’ and ‘time management’ which both occur before the pause after which the speaker’s 1 starts speaking. ‘Information seeking’ is natural at this place, as the speaker 2 simply seeks for information. However, ‘time management’ is more speaker-specific and shows that speaker 2 needs more time to formulate his speech. While waiting, speaker 1 tries to help giving his own directives, providing information or simply giving a sign that he would like to take the turn again.

Similarly as it was with the previous table, Table 17 has a lot in common with its counterpart on overlapping speech in Table 15. In both tables ‘information providing’ prevails, being the dialogue act after which the other interlocutor takes his turn or simply provides (short) auto-feedback. Also, in both tables ‘time management’ appeared and it did not actually elicited any specific dialogue act from the other speaker. ‘Time management’ was a sign that the speaker did not know what to say, so the interlocutor took the turn.

Table 17: Non-overlapping dialogue acts: spk 1 starts talking after spk 2 has finished

Speaker 2			***Speaker 1***		
Frequency (%)	***N***	***DA***	***DA***	***N***	***Frequency of the sequence (%)***
5.88	3	directives	directives	1	33.33
			information_providing	2	66.67
43.14	22	information_providing	auto-feedback	4	18.18
			contact_management	3	13.64
			contact_management, auto-feedback	6	27.27
			information_providing	7	31.82
			information_seeking	1	4.55
			time_management	1	4.55
31.37	16	information_providing, directives	auto-feedback	2	12.5
			auto-feedback, contact_management, time_management	7	43.75
			contact_management	2	12.50
			turn_management	5	31.25
7.84	4	information_seeking	information_providing	3	75
			time_management	1	25
5.88	3	information_seeking, contact_management	information_providing	1	33.33
			time_management	1	33.33
			auto-feedback	1	33.33
5.88	3	time_management	directives	1	33.33
			turn_management	1	33.33
			turn_management, information_providing	1	33.33
100	**51**			**51**	**600**

4.7.3.1 Topic flow

The map-task dialogue is characterised by a specific type of structure. There is only one leading person, who gives instructions where to go and the listener follows the instructions. The instruction giver divides the route into smaller distances which are limited by the landmarks on the map. Once the listener reaches the landmark, the instruction giver moves forward and starts describing the way of how to get to the next landmark. On the way, the interlocutors encounter obstacles because their maps are different and they need to explain the differences to get back on the right track. If their cooperation is successful, they finally reach the destination.

For the purpose of the topic flow analysis, the dialogue was divided into smaller chunks by map landmarks. Once the speakers reached the landmark or they realised there was a misunderstanding, the border dividing the dialogue was put. Such rules divided the dialogue into 28 chunks which were categorised into the following categories (topics):

1. dialogue initiation – the beginning of the dialogue, quite chaotic, when the interlocutors try to settle their new roles;

2. dialogue termination – the interlocutors have reached the goal and want to finish the cooperation;
3. direction description – the leading person gives instructions about the route to the instruction follower;
4. misunderstanding – either the speakers have just realised that they have a communication problem or the instruction follower (speaker 1) lost his way and he tries to explain where he is and why he cannot follow the instructions;
5. misunderstanding explanation_spk2 – speaker 2 tries to explain the misunderstanding;
6. misunderstanding explanation_spk1_spk2 – the interlocutors realised they have a problem and they both try to explain the misunderstanding.

For each of the dialogue chunk, the speech duration for both speakers was calculated and normalised according to the normalised difference formula:

$$normalised\ difference = \frac{dur_{spk2} - dur_{spk1}}{\left(\frac{dur_{spk2} + dur_{spk1}}{2}\right)} * 100$$

where dur_{spk1} and dur_{spk2} are durations of speech in the categorised dialogue chunk.

The results of the normalisation process and the division of the dialogue into 28 chunks are shown in Table 18. The normalisation process showed statistically significant difference between the most numerous types of topics, namely "direction description" and "misunderstanding". To check the statistically significant difference, the dispersion ranges of the result sets, i.e. the closeness to the mean, were compared. For this measure dispersion ranges are estimated starting with the mean scores. From the mean scores one standard deviation is subtracted to define the lower limit, and one standard deviation is added to the mean score to define the upper limit. The formulae for the calculations are:

lower limit: *mean – standard deviation*

upper limit: *mean + standard deviation*

If the dispersion ranges do not overlap, then there is a significant difference between the two sets. If the dispersion ranges do overlap, then there is no statistically significant difference.

No statistical test was carried out on the normalised difference figures, but according to the data in Table 19 it can be gathered that the difference between the categories "direction description" and "misunderstanding" is so high that it is assumed that it is statistically significant. The difference between the two categories is visualised on Figure 17.

Table 18: Normalised difference of speakers' speech at different categories. ID – ID of the dialogue chunk (position in dialogue), Dur – speech duration

ID	***Category***	***Dur_{Skp1}***	***Dur_{Spk2}***	***Normalised difference***
1	Dialogue initiation	11.1	13.82	21.83
28	Dialogue termination	0	0.46	200
2	Direction description	3.51	11.57	106.9
3	Direction description	4.23	18.54	125.69
4	Direction description	1.59	12.11	153.58
6	Direction description	5.65	17.31	101.57
11	Direction description	6.23	11.33	58.09
14	Direction description	5.93	20.48	110.19
16	Direction description	6.31	22.94	113.71
17	Direction description	2.07	11.30	138.07
19	Direction description	5.47	11.45	70.69
20	Direction description	1.90	8.75	128.64
21	Direction description	7.16	17.51	83.91
22	Direction description	2.38	5.87	84.61
23	Direction description	1.04	10.26	163.19
25	Direction description	1.62	4.91	100.77
26	Direction description	5.97	16.65	94.43
27	Direction description	0.97	5.77	142.43
5	Misunderstanding	11.04	5.37	-69.10
8	Misunderstanding	12.94	12.31	-4.99
10	Misunderstanding	18.67	11.26	-49.52
12	Misunderstanding	6.29	10.47	49.88
13	Misunderstanding	14.06	18.40	26.74
15	Misunderstanding	16.48	25.29	42.18
18	Misunderstanding	10.40	7.59	-31.24
24	Misunderstanding	12.38	13.92	11.71
9	Misunderstanding explanation_spk1&spk2	9.99	16.77	50.67
7	Misunderstanding explanation_spk2	3.67	16.42	126.93

Table 19: Difference between the main categories

Topic	***Min***	***Max***	***Average***	***STDEV***
Direction description	58.09	163.19	111.03	29.59
Misunderstanding	-69.10	49.88	-3.04	43.55

Figure 17: Difference between the two most numerous dialogue categories

The analysis of the dialogue chunks divided by the landmarks into different categories (topics), uncovered some dialogue patterns listed below:

1. speaker 2, the instruction giver, was making it explicit that one landmark had been reached and they would be heading for the next one from then on. Speaker 2 was mainly using the word "teraz" (Eng. "now") to underline the shift to the next landmark. Examples of the "speech signs":
    1. "i teraz" (Eng. "and now")
    2. "teraz [pw] skręcamy" (Eng. "now [filled_pause] we turn")
    3. "i teraz tak" (Eng. "and now this way")
    4. "następnie" (Eng. "then/next")
    5. "czyli zakręcamy" (Eng. "so we turn")
    6. "idziemy" (Eng. "we go")
    7. "i my idziemy" (Eng. "and we go")
2. speaker 1, the instruction follower, was giving explicit "speech signs" that he reached the landmark and was ready to receive instructions how to get to the consecutive landmark. Speaker 1 mainly used word "dobra" (Eng. "OK") to say that he reached the goal. Examples of speech signs were:
    1. "dobra" (Eng. "OK")
    2. "mam" (Eng. "I have it")
    3. "tak" (Eng. "yes")
3. auto-feedback was given mainly by using the following "speech signs"

1. “tak” (Eng. “yes”)
2. “acha” (Eng. “aha”)

4. when misunderstanding appeared the speakers underlined the differences by stating about whose map/landmarks they were talking. They expressed this by saying “u mnie” (Eng. “on mine <my map>)”

These results may be compared with Karpiński (2006).

4.8 Conclusions

The corpus linguistic pilot study was carried out to to find an empirical basis for the dialogue acts and alignment, and for formal modelling. A set of 12 dialogue act categories was selected from Bunt’s main categories (2008) and used for the annotation of selected dialogues from the PoInt corpus (Karpiński 2002). The dialogues were analysed for the purpose of creating finite state automaton models of dialogue sequences and a formal dialogue model for the prototype dialogue system.

Chapter 5: Modelling dialogue sequences with finite automata

5.1 Chapter overview

The use of Finite-State-Automata (FSA) for natural language modelling is not a new concept, but no published works are available for the Polish language, and where they exist, they are not driven by modern linguistic theories. More advanced tools for computer modelling of the human language competence were presented by Vetulani (2004) and automatic text processing technologies for applications in public security were described in (Vetulani et al. 2010). However, the solutions using the FSA for spoken language has not yet been applied.

When it comes to dialogue and other languages, Raux and Eskenazi (2009), for example, use a non-deterministic finite-state machine for modelling the control over turn-taking behaviour of conversational agents. Gibbon (1985) describes finite state automata for short-wave radio communication in English, using older speech act concepts and the concept of 'securing of uptake' as a basic version of semantic alignment procedures. But for Polish such solutions have not yet been adopted.

This chapter presents relevant properties of finite state automata and automaton creation for modelling dialogue sequences and discusses their application in the dialogue system.

5.2 Automaton models

The basic model of a dialogue can the represented as the Finite State Automaton (FSA) which is presented in Figure 18. The FSA illustrates the regular expression: (Question Answer) + Confirmation (Gibbon 1981, Gibbon 1985).

The representation in Figure 18 is much too simple. Analysis of the dialogue shows many more complex sequences, and also a lot of overlapping turns, which this model does

not describe. Also, macro-level annotation shows that a dialogue consists of the following phases, which also need to be described:

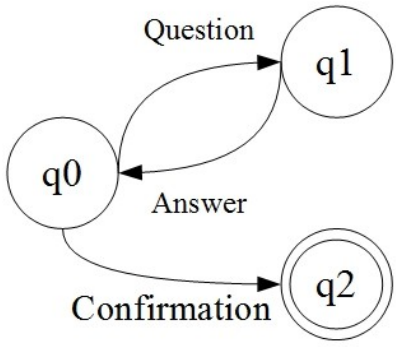


Figure 18: A basic dialogue model implemented as FSA

1. H - Hello greeting.
2. O – Opening.
3. N – Negotiation.
4. C – Closing.
5. G – Goodbye greeting.

This ordering of the phases allows for different variations (Gibbon et al. 2000:62) which a full model must cover.

The following sections introduce observations from the dialogue and the automata theoretic concepts which will be needed for dialogue sequence modelling.

5.3 First steps in realistic automaton creation

The annotation on the dialogue act tier was used to create manually a collection of loop-free automata which modelled each sequence of the dialogue acts for each of the speakers. To create the loop-free automata the matrix presented in Appendix A was analysed separately for each speaker. Each dialogue act sequence served to define the initial node, the terminal node and the transitions between the nodes for each of the loop-free automaton. Such automata were evaluated for correctness using an NDFST interpreter (Gibbon 2008, more about the tool in 5.7.2 NDFST interpreter online tool below). The

length of the input string to the NDFST interpreter was equal to the length of the dialogue act sequence which the automaton was to generate. If the generated output sequence was the same as the dialogue act sequence on which the automaton was based, then the automaton was correct. Exemplar automata creation process for speaker 2 at this stage is shown in Table 20; the full table with the automata for speaker 1 is presented in Appendix B. The columns in the Table stand for: Dur – duration, I – initial state, T – terminal state, ID – automaton ID, grey colour – the moments when neither of the interlocutors was speaking, divides the dialogue into parts which define the sequences on which automata were built.

Table 20 and Table from Appendix B show only correct automata. If the same dialogue act appeared one after the other in the original dialogue act sequence, one of them was removed as such repetition will be modelled later by a loop in automata with iterations. If to one utterance more than one dialogue act was assigned, then such information was coded by connecting the two or more dialogue acts by an underscore, e.g. infpr_dir. Such prepared automata and their outputs were then used in the automata generalisation process.

Table 20: Excerpt of table with loop-free automata for each sequence of dialogue acts for speaker 2

Start	***End***	***Dur***	***Dialogue acts***	***Utterances***	***I***	***T***	***Transitions***	***Input***	***Output***	***ID***
0.00	3.81	3.81								
3.81	4.19	0.38	auto-feedback	tak	q0	q2	q0, x, auto, q1; q1, x, infpr_dir, q2	x x	auto infpr_dir	1
4.19	4.84	0.65								
484	6.27	1.43	information providing, directives	wychodzimy z zamku						
6.27	7.69	1.42								
7.69	8.4	0.72	time management	[pw720]	q0	q2	q0, x, time, q1; q1, x, infpr, q2	x x	time infpr	2
8.40	10.90	2.50								
10.9	11.53	0.63	time management	[pw570]						
11.53	12.86	1.33	information providing	do zamku						
12.86	12.96	0.10								

The output dialogue act sequences and the automata's IDs described above and partly presented in Table 20 (and fully shown in Appendix B) underwent the following processing for the purpose of automata generalisation:

1. alphabetical sorting of dialogue act sequences,

2. reduction of multi-layered labels of dialogue acts to one-layered labels; only the first dialogue act was preserved, e.g. infpr_dir → infpr,
3. deletion of repetition of the same dialogue act,
4. re-sorting of dialogue act sequences.

The reduction process was carried out, because the fact that to the same utterance more than one communicative function could be assigned made the longer sequences unique and limited the generalisation. The partial results of the reduction process sorted alphabetically before and after the reduction are shown in Table 21. Different positions of automata IDs in the table show that the reduction process had an effect. The entire tables for both speakers are shown in Appendix C.

Table 21: Examples of reduction of multi-layered labels to one-layered labels for speaker 2 sorted alphabetically. ID – ID of the automaton

Before		***After***	
ID	***Outputs with multi-layered labels***	***ID***	***Outputs with one-layered labels***
24	auto dir	24	auto dir
34	auto infpr infsk_cnt infpr_turn infpr	1	auto infpr
1	auto infpr_dir	34	auto infpr infsk infpr infpr
17	dir	17	dir
21	dir auto auto_infpr infpr time infpr_turn infpr	21	dir auto auto infpr time infpr infpr
19	dir auto cnt infpr	19	dir auto cnt infpr
8	infpr	8	infpr
3	infpr auto	6	infpr
4	infpr auto infpr time_turn time infpr_dir	11	infpr
40	infpr infpr_dir	3	infpr auto
22	infpr infpr_dir time infsk time_turn	4	infpr auto infpr time time infpr
7	infpr partner	44	infpr cnt infpr infpr time infpr infpr own

5.4 Generalisations over finite regular languages

Many generalisations over finite regular languages can be expressed by loop-free FSAs and can be visualised with directed acyclic graphs. Altogether 25 loop-free automata were created, 10 for speaker 1 and 15 for speaker 2, on the material which was prepared using the automatic method described in Section 4.5

5.4.1 Prefix generalisations

The outputs with one-layered dialogue act labels presented in Table 21 were analysed for the purpose of generalisation of the automata. The following steps concerning repetitions were taken:

1. sequences of the the same single dialogue act were reduced to just one occurrence of this dialogue act,
2. occurrences of the same sequence of dialogue acts in one output were marked in yellow,
3. if the same dialogue act repeated every second time it was marked in green.

The three different kinds of repetitions may be modelled by iterations in the automata. Table 22 shows a fragment of a table with the results of such analysis for speaker 2. The table is called the prefix generalisation table and it served to create combined automata with and without loops. The names of the automata created on the basis on the dialogue act sequences are also presented in the table. The entire tables with analyses for both speakers are presented in Appendix D.

Table 22: A fragment of the prefix generalisation table for speaker 2

Name	***ID***	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
1_auto	24	auto	dir								
1_auto	34	auto	infpr	infsk	infpr						
1_auto	1	auto	infpr								
2_dir	19	dir	auto	cnt	infpr						
2_dir, 3_dir	21	dir	auto	infpr	time	infpr					
2_dir, 3_dir	17	dir									
4_infpr, 5_infpr	4	infpr	auto	infpr	time	infpr					
4_infpr, 5_infpr	9	infpr	auto	infpr	time	infpr	dir				
4_infpr, 5_infpr	3	infpr	auto								
4_infpr, 5_infpr	36	infpr	auto								
4_infpr, 5_infpr	44	infpr	cnt	infpr	time	infpr	own				
6_infpr_a, 6_infpr_b, 7_infpr_b	31	infpr	dir	infpr							
7_infpr_a , 7_infpr_b	26	infpr	dir	own	dir	infpr	infsk	infpr			
8_infpr	7	infpr	partner								
9_infpr_a	12	infpr	time	infpr							
9_infpr_a, 9_infpr_b, 9_infpr_c	30	infpr	time	infpr	time	infpr	time	auto	infpr	turn	infpr
9_infpr_a	46	infpr	time	infpr	time	infpr					
9_infpr_a	48	infpr	time	infpr							

The material of dialogue act sequences sorted alphabetically served to build several loop-free automata which could be visualised as directed acyclic graphs. (The material is shown in Table 22 above and in Appendix D.1 and Appendix D.2.) Sequences with the same dialogue acts at the beginning were grouped together and combined automata without loops were created. In other words, if the simple automata built on the actual dialogue act sequences (cf. Appendix B) shared the same prefix (initial substrings), they were merged together into a tree with a shared prefix.

The following studies of initial dialogue acts were carried out, because nearly half of the dialogue act sequences in the studied material were composed of only one or two dialogue acts (cf. Table 4, column $n \leq 2$). Table 23 shows different single dialogue acts and the number of sequences in which they occurred at the initial position of a *part* of each speaker in the analysed dialogue. Whereas Table 24 presents the most frequent two dialogue acts at the beginning of a *part.*

The numbers in Table 23 show that the prefixes, or rather single initial dialogue acts, for both speakers differ. The numbers stand for the numbers of sequences in which the corresponding prefix occurred. For example, there are much more *parts* in which speaker 1, the instruction follower, starts with auto-feedback utterances, whereas, the total number of sequences starting with the information-providing utterances is half of the number of all of the sequences of speaker's 2, the instruction giver. The data in Table 23 also indicate that there were no sequences starting with directives for speaker 1, and no sequences starting with contact management and partner communication management for speaker 2, therefore automata with such prefixes could not be created for those speakers.

Table 24 presents what appears after the first dialogue act. While speaker 2 mainly provides information as it is indicated by the information providing dialogue act either on the initial or the second position, speaker 1 is a bit more diverse as not only provides he information, but also seeks for it.

The simple analysis of dialogue acts at the beginning of parts shows similarities and differences between speakers. The source of the similarities and differences may come from either the role in the map-task dialogue or the speaker's personality. While speaker 1 seems to be more decisive, speaker 2 needs more time to construct an utterances. Such difference may originate in the speakers' personalities or may be caused by more responsible and complex task assigned to the speaker 2.

Table 23: Initial dialogue acts in sequences for each of the speakers

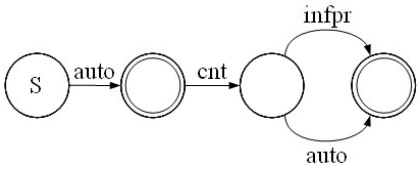
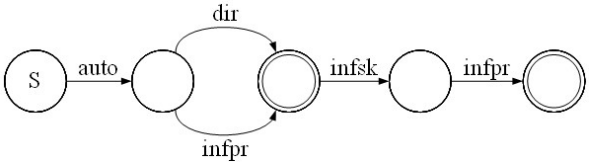
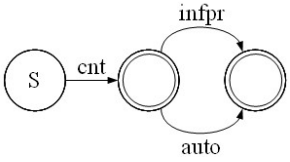
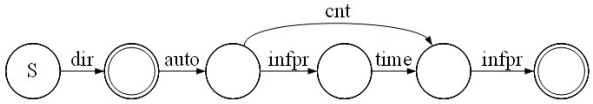
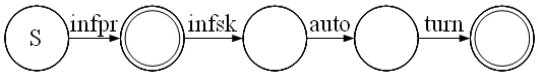
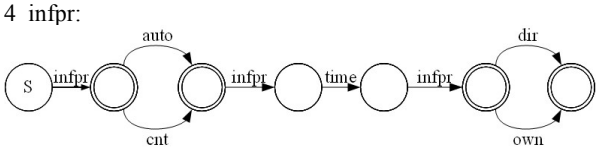
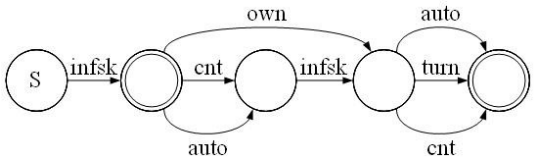
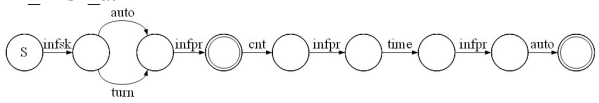
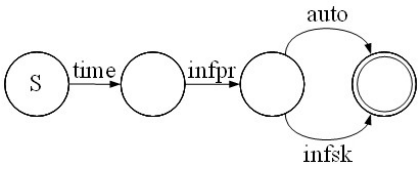
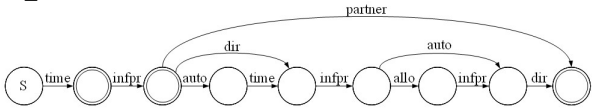
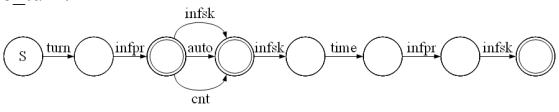
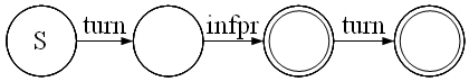
Prefix – initial dialogue act	***Speaker 1***	***Speaker 2***
auto-feedback	11	3
contact management	5	0
directives	0	3
information providing	11	24
information seeking	5	2
partner communication management	2	0
time management	6	13
turn management	9	4
Overall:	***49***	***49***

Table 24: Most frequent two dialogue acts at the beginning of a part for speaker 1 and speaker 2.

Speaker 1		***Speaker 2***	
Sequence	***N***	***Sequence***	***N***
auto cnt	3	auto infpr	2
auto infpr	5	dir auto	2
infpr cnt	2	infpr auto	4
time infpr	3	infpr dir	2
time infsk	2	infpr time	7
turn infpr	7	infpr turn	3
turn infsk	2	time infpr	9
		time turn	2
		turn infpr	3

Table 25 shows examples of loop-free automata built by combining sequences with the same prefix. All the automata can be found in Appendix F. The loop-free automata, although clear, they are not very efficient. Some of them can produce only 2 sequences of dialogue acts. Additionally, some of the automata have many transition states – up to 9 nodes. These results prompted automata reconstruction in order to add loops, reduce the number of nodes and improve efficiency so that one automaton could generate more sequences of dialogue acts.

Table 25: Loop-free automata combining sequences with the same prefix.

Speaker 1	***Speaker 2***
auto-feedback	
1_auto: 	1_auto: 
contact management	***directives***
4_cnt: 	2_dir: 
information providing	
15 infpr: 	4 infpr: 
information seeking	
11_infsk_a: 	12_infsk_a: 
time	
7_time: 	13_time: 
turn	
6_turn: 	17_turn_a: 

5.4.2 Suffix generalisations

In order to generalise suffixes of the automata, a suffix generalisation table was created. A fragment of the suffix generalisation table for speaker 2 is presented in Table 26. The table shows sequences of dialogue acts which occurred at the dialogue *parts*, but this

time they are moved to the furthest right columns, as opposed to the prefix generalisation table. For suffix generalisation the last 3 dialogue acts were taken into account. If the sequences shared the same suffix (final substring), they were merged into a shared suffix. The green colour in Table 26 marks the sequences which could be combined. Only 2 automata were created using the suffix generalisation for speaker 2 – one loop-free automaton, and one automaton with iteration. An automaton without loops built by suffix generalisation is shown on Figure 19.

Table 26: Suffix generalisation table for speaker 2. M – match

Name	***ID***	***10***	***9***	***8***	***7***	***6***	***5***	***4***	***3***	***2***	***1***	***M***
1_auto	34							auto	infpr	infsk	infpr	1
7_infpr_a, 7_infpr_b	26				infpr	dir	own	dir	infpr	infsk	infpr	1
16_time	36	time	turn	infpr	allo	time	cnt	infpr	allo	time	infpr	
2_dir, 3_dir	21						dir	auto	infpr	time	infpr	2
9_infpr_a	12								infpr	time	infpr	2
9_infpr_a	46						infpr	time	infpr	time	infpr	2
9_infpr_a	48								infpr	time	infpr	2
9_infpr_a	14								infpr	time	infpr	2
	2									time	infpr	2
9_infpr_a, 9_infpr_b, 9_infpr_c	30	infpr	time	infpr	time	infpr	time	auto	infpr	turn	infpr	2?
12_infsk_a,12 _infsk_b	23								infsk	auto	infpr	
4_infpr, 5_infpr	4						infpr	auto	infpr	time	infpr	2

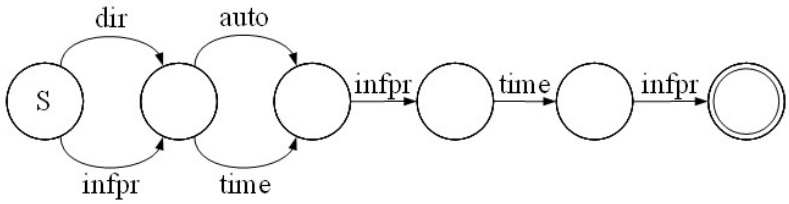


Figure 19: Combined automata 2_back without loops created by suffix generalisation

5.4.2.1 Centre generalisations

If the dialogue act sequences shared the central part, they were merged. The centre generalisations were made while looking at the same sequences within one dialogue act sequence in the speaker's turn (i.e. within the sequence in one row in Table 22) and across the turns (i.e. across the sequences in different rows in Table 22). However, no automata apart from those built in the suffix and prefix generalisation procedures were created using in particular the centre generalisation.

5.5 Generalisations over non-finite regular languages

Generalisations over non-finite regular languages can be expressed with FSAs with loops, and visualised with directed cyclic graphs. Altogether 22 automata with loops were created, 7 for speaker 1 and 15 for speaker 2.

5.5.1 Local generalisations

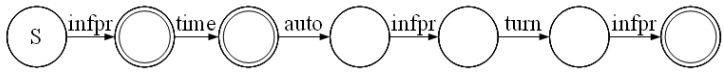
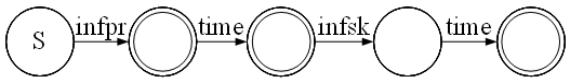
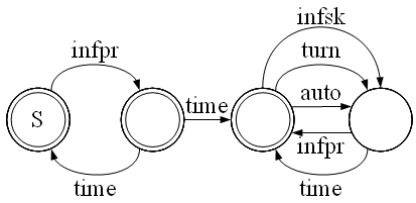
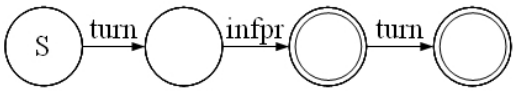
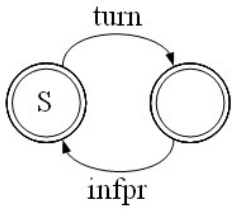
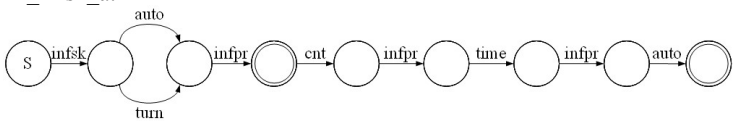
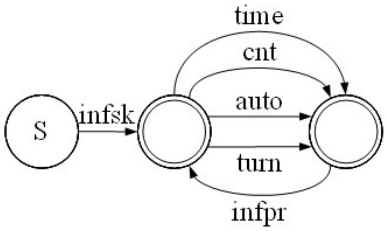
If a transition starts and ends at the same state, it is a local generalisation. However, no automata with local generalisations were needed in this pilot study.

5.5.2 Non-local generalisations

If a transition starts at state j in a path through an automaton, and ends at state j-i, $i > 0$, then the loop represents a non-local generalisation. (Note that local generalisation is when $i=0$).

Analysis of the prefix generalisation tables presented fully in Appendix D and analysis of the loop-free automata allowed to create a whole set of automata with loops which model non-finite regular languages. Exemplar loop-free automata and their counterparts with loops for speaker 2 are shown in Table 27. Additionally, an automaton which combine suffix with a loop is presented on Figure 20. All the iterative automata are presented in Appendix G.

Table 27: Loop-free automata and their counterparts with loops for speaker 2.

Loop-free	***With loops***
9_infpr:  10_infpr:  	10 infpr NEW 9+10: 
17_turn_a:  	17_turn_b: 
12_infsk_a:  	12_infsk_b: 

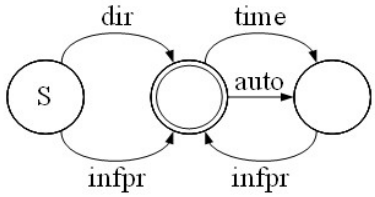


Figure 20: Combined automata 1_back with loops created by suffix generalisation

5.6 Turn automata

In this pilot study a set of dialogue acts automata were created for the future use in dialogue modelling. The material described above allowed to create a set of automata for speaker 1 and speaker 2 based on real dialogue act sequences. Silent pauses were not taken into account in the automata creation procedure. Three types of automata were created:

1. simple automata based only on real dialogue act sequences for each speaker separately;

2. semi-coupled automata based on real dialogue act sequences for both speakers;
3. generalised automata with recursive transitions for each of the speakers separately.

Simple automata generate dialogue act sequences for each speaker, whereas semi-coupled automata present the simple automaton for each speaker and the change of turns between the speakers. An exemplar semi-coupled automaton created by combining two simple automata for each speaker is presented on Figure 21, with semi-coupled automaton 1 made by combining dialogue act automata for speaker 1 (spk1) and speaker 2 (spk2). S stands for the initial state. The node more to the left shows which speaker starts the sequence. The dotted arrows show the transition of turns between the speakers. More than one dialogue act on an arc means that the utterance in the original dialogue had more than one communicative function.

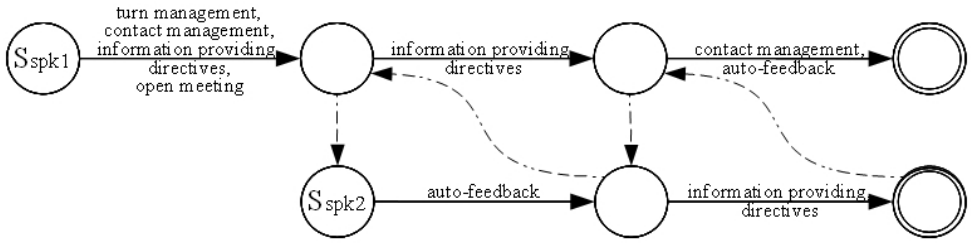


Figure 21: A semi-coupled automaton 1 for spk1 and spk2.

Analysis of the simple automata resulted in creating more general automata for each of the speakers. Generalised automata characterise complexity and recursion. Such automata have only one initial state, but may have more than one terminal state. Exemplars of the generalised automata are presented in Figure 22 for the speaker 1 – the follower of the instructions, and in Figure 23 for the speaker 2 – the leading person in the map task. More examples of semi-coupled automata can be found in Appendix E and of generalised automata for speaker 1 can be found in Appendix F and Appendix G.

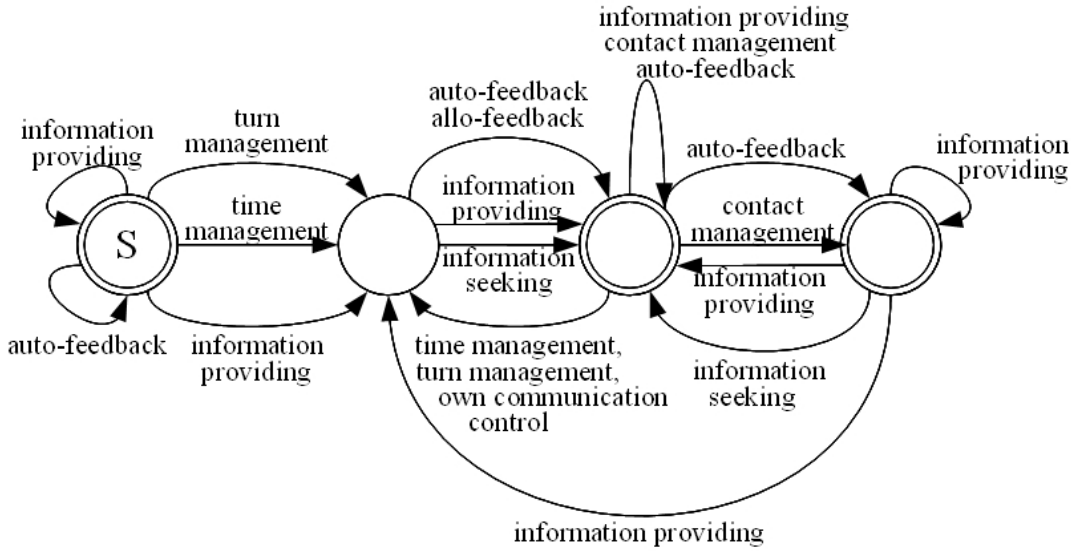


Figure 22: A generalised automaton of dialogue acts for speaker 1, the follower of the instructions in the map task

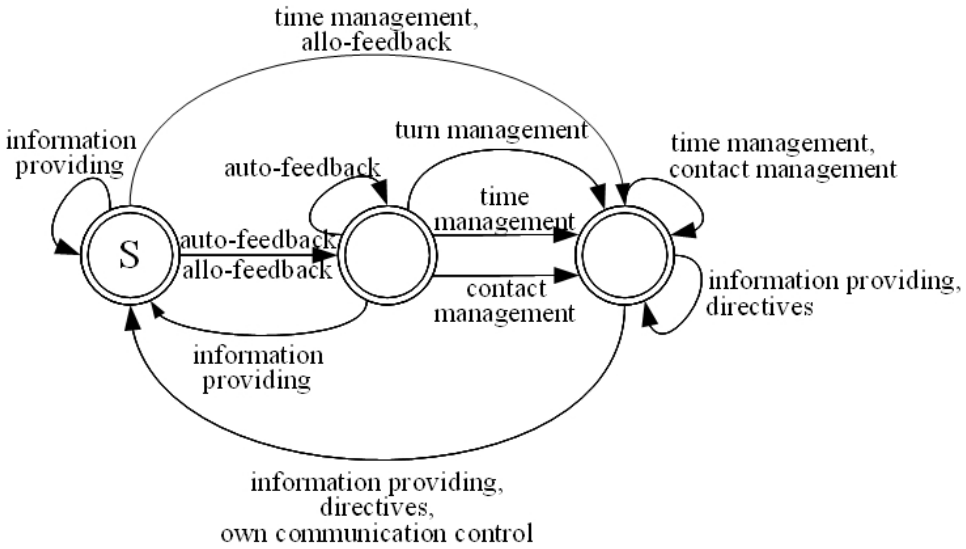


Figure 23: A generalised automaton of dialogue acts for speaker 2, the instructor giver in the map task

5.7 Evaluation of dialogue act automata

5.7.1 General evaluation criteria

Generalisation means that one automaton may generate different sequences. Those sequences may differ in length and output sequences. The output needs to be tested for truth. If the generated sequences occur in the corpus or are evaluated as possible, then they are true. However, if the sequences are empirically impossible (inside or outside the corpus), then this is overgeneralisation.

The automata were evaluated for:

1. Coherence: whether the automata are syntactically correct and actually work when operational.
2. Completeness: whether the automata describe all the phenomena they are intended to describe (not necessarily only restricted to a particular corpus, but including generalisations, and possibly also judged by native speaker intuitions, too).
3. Soundness: whether the automata describe *only* the phenomena they are intended to describe (not necessarily restricted to a particular corpus, but including generalisations, and possibly also judged by native speaker intuitions, too).
4. Consistency: whether the modelling is done in the same way for similar observations of utterances.

5.7.2 NDFST interpreter online tool

The Nondeterministic Finite State Transducer (NDFST) online tool (Gibbon 2008) was used in the present study to evaluate the dialogue acts automata. This NDFST tool allows to evaluate the correctness of the FST transitions and it generates output specified by those transitions. The NDFST takes the information about the initial states and terminal states as input, together with the transition quadruples:

```
<currentstate, inputsymbol, outputsymbol, nextstate>
```

The transducer implements a finite state relation (regular relation), that is, a translation from one regular language into another regular language. It can also be used with empty input or output languages. For the evaluation, the input language 'x*' was used, i.e. the input of the tool was specified with 'x' sequences of different lengths 1, … *n*, which simply

determine the number of the states the transducer is intended to go through, and in this way the transducer specifies the length of the output.

The code input to the system is presented below (Gibbon 2008):

```
# METADATA: Demo FST file for NDFST interpreter.
# The interpreter features generous use of white space, error
messages for FST
# configuration file input, optional prettyprinted trace for
fst run.
# Initial state:
initial = q0
# Set of terminal states:
terminal = q0, q1, q2
# Transition quadruples
<currentstate,inputsymbol,outputsymbol,nextstate>:
fst =
   q0, x, info_seek, q1;
   q1, x, info_seek, q1;
   q1, x, auto-feedback, q0;
   q1, x, contact_management, q0;
   q1, x, turn_management, q2;
   q1, x, info_provide, q2
```

The NDFST interpreter may take unspecified x's as input and then, depending on the state, it generates any dialogue act sequences. Exchanging x's by dialogue acts and combining them with real utterances allows to generate sequences of real utterances which occurred in the dialogue, and only utterances which occurred in the dialogue.

In the above exemplar code the initial state is labelled `q0`, and there are 3 terminal states at `q0`, `q1` and `q2`. For an input equal '`x x x`', the FST tool will generate output seqeuences such as the following:

```
info_seek auto-feedback info_seek
info_seek contact_management info_seek
info_seek info_seek auto-feedback
info_seek info_seek contact_management
info_seek info_seek info_provide
info_seek info_seek info_seek
info_seek info_seek turn_management
```

Graphs of automata created within this pilot study underwent testing using the NDFST interpreter. In order to test the automata, a set of transition quadruples were created based on the graphs and input to the NDFST online tool. The following criteria were evaluated in order to achieve the goals of the general evaluation criteria:

1. FST code must conform to:
    1. correctness of transition quadruples,
    2. correctness of specified initial and terminal states,
    3. ability to specify length of input.
2. Semantics of FST (correspondence with dialogue):
    1. identity of generated output sequences with the sequences in the real dialogue,
    2. ability to handle recursion (loops, iteration),
    3. length of input and its impact on the number of outputs when recursion is eliminated or reduced,
    4. length of input and its impact on the number of outputs when recursion is not eliminated.

5.7.3 Evaluation results

The evaluation involved creating two types of FST:

1. FST which took x's as input and produced a set of dialogue acts sequences as output,
2. FST which took dialogue acts sequence as input and produced real utterances as output.

In the first case only the generalised automata with loops were evaluated. The generated output was correct, however the longer the input the more different sequences were created. However, according to the FSA they all could appear in the real dialogue.

Also the second type of FST was evaluated as generalised automata with loops. This resulted in creating sequences of real utterances. The aim of the evaluation was to see how many different types of sequences need to be created in order to generate the target

sequence, i.e. the sequence which occurred in the real dialogue. If the input length was short, just a few output sequences were created to get the target sequence. However, complex automata generated a lot of overgeneralised 'junk' (files up to 70MB of data) and testing FST with loops was stopped in favour of FST without loops. In formal terms, the automata were 'complete' but not 'sound'.

Evaluation of FST without loops or with a reduced number of loops went successfully. The target sequences were created as the only output sequence or with a few other sequences which were all possible to appear in the real conversation. Even inputs including many transition states did not produce many output sequences.

The evaluation tables of generalised automata are to be found in Appendix H.1 and evaluation tables of semi-coupled automata, *de facto* simple automata without loops are to be found in Appendix H.2.

5.8 Loop-free automata evaluation

The loop-free automata with prefix and suffix generalisations underwent evaluation using the NDFST online tool (Gibbon 2008). Based on the graphs of the automata, for each automaton an initial and final states were defined and transition quadruples as before:

```
<currentstate, inputsymbol, outputsymbol, nextstate>
```

written. Such prepared material were then tested in the NDFTS online tool. The following criteria were evaluated:

1. correctness of the input code, i.e. the initial state, final states and the transition quadruples,
2. truth of the output – if the output matched the dialogue act sequences on which the automaton was built, then the automaton was correct,
3. soundness of the output – if the output did not exactly match the dialogue act sequences on which the automaton was built, but the generated sequences did appear partially in the other sequences, then the automaton was correct.

Table 28 presents a part of an evaluation table with a fragment of an evaluation table of loop-free automata for speaker 1. The abbreviations are defined as follows:

ID – sequence ID,
Sequences – original dialogue act sequences on which the automata were built

Name – name of automaton

I – initial node

T – terminal node

Tr – truth value (contribution to 'completeness' criterion)

S – soundness value (contribution to 'possibility' criterion)

In the table the original sequences on which automata were built are presented, together with the automata code required by the NDFTS online tool. The automata were tested for different input lengths, i.e. different lengths of x's sequences.

The evaluation shows that if the length of the x-sequence was equal the sequences on which the automaton was built, then the true or at least possible output was produced ('1' stands for 'true'). If the x-sequence did not equal the original dialogue act sequences, no output was produced. No output does not mean that the automaton is erroneous, but simply did not match the automaton specifications.

Table 28: Fragment of evaluation table of loop-free automata. for speaker 1

ID	***Sequences***	***Name***	***I***	***T***	***Code***	***Input***	***Output***	***Tr***	***S***
44	auto cnt auto	1_auto	q0	q1, q3	q0, x, auto, q1;	x	auto	1	1
43	auto cnt auto				q1, x, cnt, q2;	x x	–	–	–
13	auto cnt infpr				q2, x, auto, q3;	x x x	auto cnt auto	1	1
6	auto				q2, x, infpr, q3		auto cnt infpr	1	1
10	auto					x x x x	–	–	–
						x x x x x	–	–	–
15	turn infpr auto	6_turn	q0	q2, q3, q7	q0, x, turn, q1;	x	–	–	–
1	turn infpr cnt				q1, x, infpr, q2;	x x	turn infpr	1	1
47	turn infpr auto infsk time infpr infsk				q2, x, infsk, q3; q2, x, auto, q3; q2, x, cnt, q3; q3, x, infsk, q4;	x x x	turn infpr auto	1	1
16	turn infpr				q4, x, time, q5;		turn infpr cnt	1	1
17	turn infpr				q5, x, infpr, q6;		turn infpr infsk	1	1
20	turn infpr				q6, x, infsk, q7	x x x x	–		
29	turn infpr infsk					x x x x x	–	–	–
						x x x x x x	–	–	–
						x x x x x x x	turn infpr auto infsk time infpr infsk	1	1
							turn infpr cnt infsk time infpr infsk	0	1
							turn infpr infsk infsk time infpr infsk	0	1
						x x x x x x x x	–	–	–
						x x x x x x x x x	–	–	–

5.9 Iterative automata

Automata with iterations underwent the same evaluation procedure as the loop-free automata using the NDFST online tool (Gibbon 2008). The same as for the loop-free automata, initial and terminal states were specified and transition quadruples `<currentstate, inputsymbol, outputsymbol, nextstate>` written. Such material was then evaluated for the same criteria of code correctness, output sequences truth or soundness. An evaluation table of one of the automata with loops for speaker 1 is presented in Table 29.

Table 29: An evaluation table of iterative automaton for speaker 1.

ID	*Sequences*	*Name*	*I*	*T*	*Code*	*Input*	*Output*	*Tr*	*S*
5	time infsk cnt	8_time_b	q0	q3, q5	q0, x, time, q1; q1, x, infsk, q2; q2, x, cnt, q3; q3, x, auto, q2; q3, x, infpr, q4; q4, x, infsk, q5	x	–	–	–
25	time infsk cnt auto cnt infpr infsk					x x	–	–	–
						x x x	time infsk cnt	1	1
						x x x x	–	–	–
						x x x x x	time infsk cnt auto cnt	0	1
							time infsk cnt infpr infsk	0	1
						x x x x x x	–	–	–
						x x x x x x x	time infsk cnt auto cnt auto cnt	0	1
							time infsk cnt auto cnt infpr infsk	1	1
						x x x x x x x x	–	–	–
						x x x x x x x x x	time infsk cnt auto cnt auto cnt auto cnt	0	1
							time infsk cnt auto cnt auto cnt infpr infsk	0	1
						x x x x x x x x x x	–	–	–
						x x x x x x x x x x x	time infsk cnt auto cnt auto cnt auto cnt auto cnt	0	1
							time infsk cnt auto cnt auto cnt auto cnt infpr infsk	0	1
						x x x x x x x x x x x x	–	–	–

The evaluation of iterative automata was more complicated than the loop-free automata. Many more outputs were generated by one automaton and these were not the original sequences of dialogue acts on which the automata were built, therefore the possibility criterion was more often evaluated than in the loop-free automata. It was also problematic to decide on the maximum input length. The number of x's ranging from 1 to the biggest number of the dialogue acts in a sequence +2 was usually adopted. So for

example if the longest dialogue act sequence was 8, then the longest input of x's would be 10. However, this rule was not obligatory and also longer lengths were tested.

5.10 Further issues: dialogue flow and alignment

The deep analysis of the dialogue served to describe a typical map-task dialogue. The typical map-task dialogue is composed of smaller dialogues which belong to one of the 4 categories:

1. dialogue initiation
2. direction description
3. misunderstanding
4. dialogue termination

The typical dialogue flow is presented on Figure 24. While dialogue initiation is chaotic, direction description and misunderstanding dialogue categories characterise certain patterns. In the direction description the leading person gives directives and provides information about the route all the time. The instruction follower adds to this dialogue only auto-feedback or interrupts the speaker asking questions. The direction description dialogues may be generated by such an automaton which is presented on Figure 25. In the misunderstanding type of dialogue, the two speakers cooperate trying to solve the problem therefore providing a lot of information to each other about the way they see their maps. An automaton which could model the misunderstanding type of dialogue is presented on Figure 26.

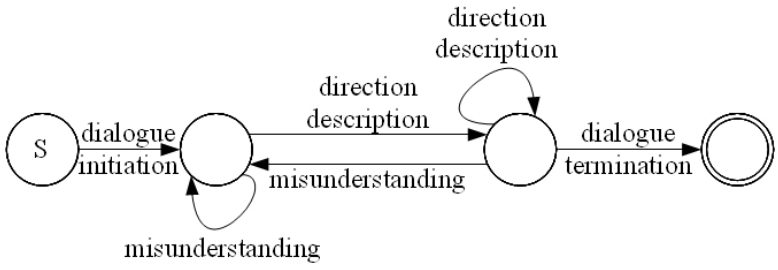


Figure 24: Automaton of typical dialogue flow.

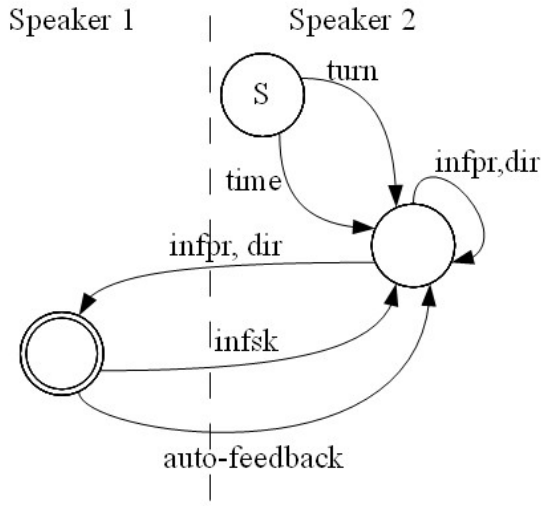


Figure 25: An automaton generating the direction description dialogue type.

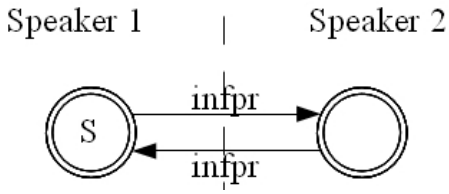


Figure 26: An automaton generating the misunderstanding dialogue type.

The analysis of the map-task dialogue uncovered some dialogue patterns. Once the speakers found themselves in their roles, their dialogue was fluent and could be limited to information providing and directives dialogue acts by the leading person, and auto-feedback and clarification questions by the instruction follower. The task participants would move smoothly from one landmark to another, finally reaching their destination.

However, the differences in the maps caused disruptions in the dialogue and those differences needed to be quickly explained. Once the instruction follower encountered an obstacle to perform a pace on the map, he was informing the speaker about it and started describing his situation on the map. The task participants co-operated the same intensively on solving the task and would not move forward until they both had the same view on where they were and to where would be the following step.

The analysis of the single-utterance turns, turn initiations and terminations triggered creating a general turn automaton, presented on Figure 27. The automaton may generate single utterance or turns built of two or more dialogue acts. The studies carried out in the present work allow to assess the likelihood of different dialogue acts being produced in a turn. Figure 28 and Figure 29 present the generalised turn automata with dialogue act occurrence probability for speaker 1 and speaker 2 respectively. Only dialogue acts with higher than 7% occurrence probability are generated by the automata. "?" in the automata means that any dialogue act may be generated at this iteration. Comparison of these two

automata for each speaker show that each of them generate different dialogue act sequences with a different probability. Speaker's 1 automaton generates single-utterance turns aiming at contact management and auto-feedback, less frequently it would provide information or seek for information. On the other hand, speaker's 2 automaton generate information providing or directive single dialogue acts. When it comes to longer turns, automata for both speakers would most frequently generate information providing dialogue acts at the beginning and at the end of turns. Apart from that similarity, only differences are to be seen. First of all, speaker's 1 automaton would generate a set of dialogue acts aiming at managing contact and providing auto-feedback, terminating in asking for information or indicating that the turn is being passed over to the interlocutor. Whereas, speaker's 2 automaton would generate turn-taking or time-managing dialogue acts at the beginning of a turn, to end up with either in providing information or directives.

These probabilistic automata could be implemented into the dialogue system manager of a dialogue system to generate the turns most adequate for the communication stage. Together with the studies about overlapping and non-overlapping speech presented in 4.7.2 and 4.7.3 the probabilities of different dialogue acts' occurrences could be narrowed down. The most general observation and also the most frequent occurrence is that speaker 2 provides information, and the speaker 1 only generates auto-feedback. Such result raises a question whether the map-task dialogue is the best scenario for acquiring data for dialogue analysis? The answer is yes as it provides a structure-predicted dialogues which can be modelled by simple finite state automata. The research presented later is this thesis shows also other techniques for collecting dialogue speech material at laboratory setting with a less predictable structure.

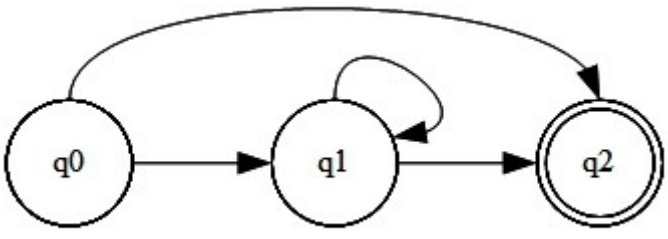


Figure 27: Generalised turn automaton

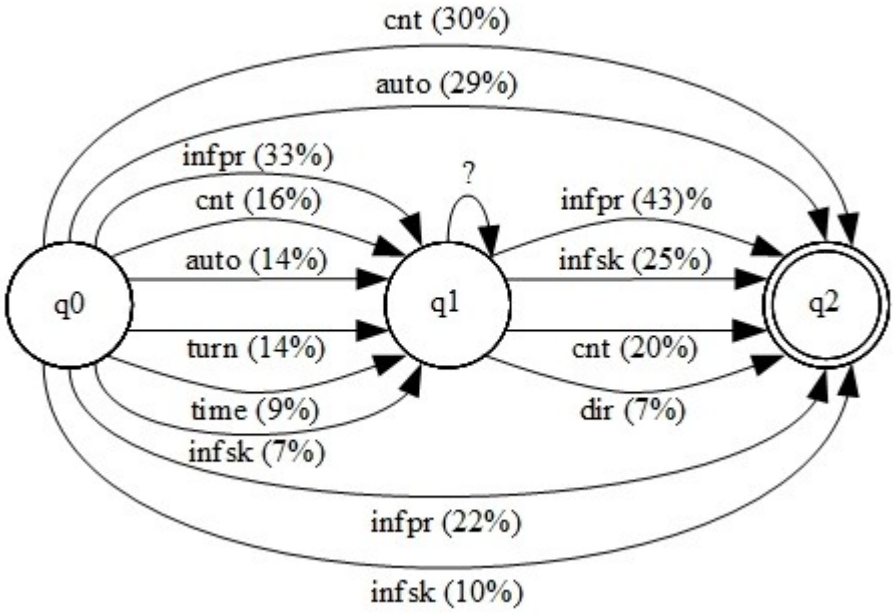


Figure 28: Generalised turn automaton for spk 1 with dialogue act occurrence probability

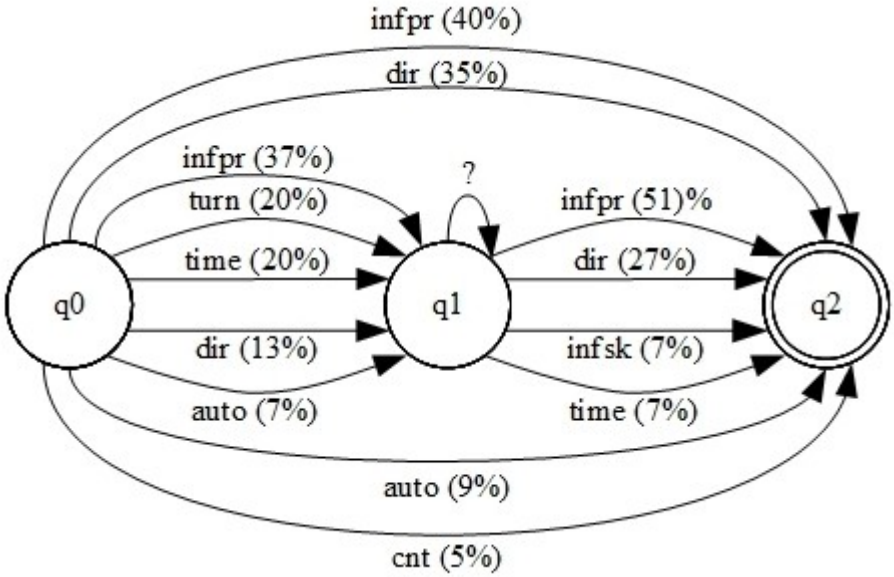


Figure 29: Generalised turn automaton for spk 2 with dialogue act occurrence probability

5.10.1 Generalised turn automaton at time line

The generalised turn automaton was put into the time context. The arcs of the automaton generate dialogue acts. Each speaker's generalised automaton was set one after the other and connected into a chain. The procedure is presented on Figure 30. It suggests that

speakers change turns linearly. However, as discussed before, speakers often do not wait for the other speaker to finish, but they interrupt each other. Therefore, the linear representation of automata flow had to be rearrange to visualise also the overlapping speech. This visualisation is shown on Figure 31. The solid arcs show the generation of dialogue acts by a single speaker, whereas the dotted arcs stand for possible turn changing between speakers.

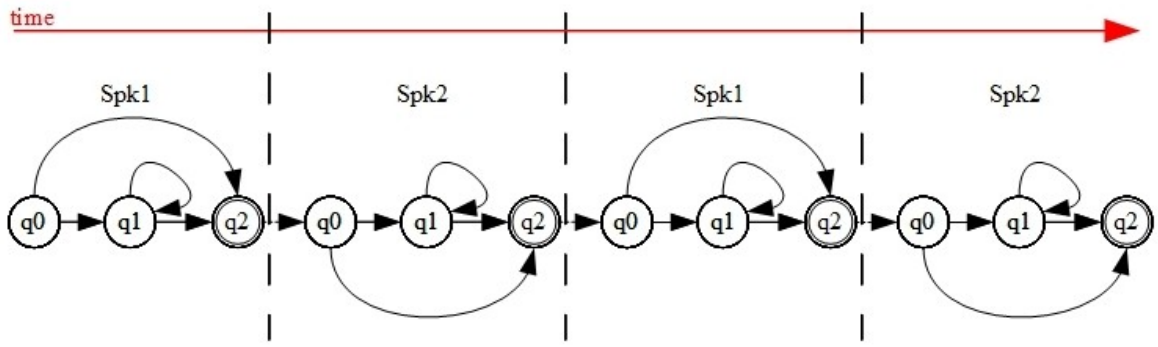


Figure 30: Linear representation of generalised turn automata for spk1 and spk 2

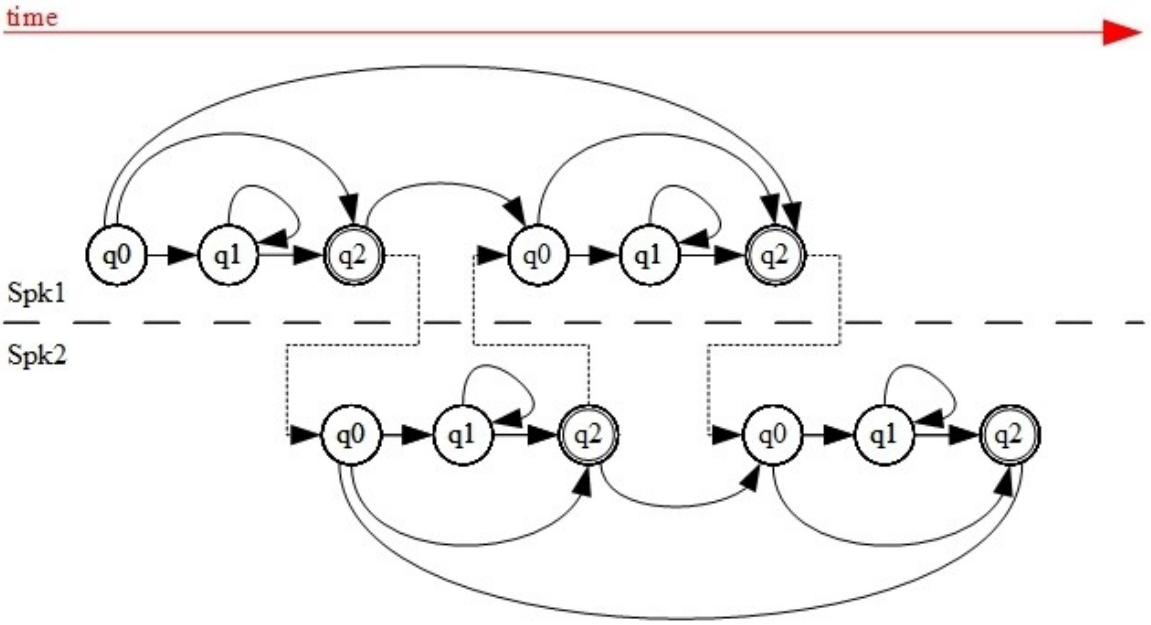


Figure 31: Visualisation of overlapping speech being produced by generalised turn automata for spk 1 and spk 2

The generalisation step was moved forward and the two generalised turn automata with tree stages (q0, q1, q2) can be replaced by a single automata with four stages (q0, q1, q2, q3) where:

q0: initiation state

q0, q2: termination states

q0 → q1: speaker's 1 dialogue act

q0 → q2: speaker's 1 dialogue act

q1 → q1: speaker's 1 dialogue act

q1 → q2: speaker's 1 dialogue act

q2 → q3: speaker's 2 dialogue act

q2 → q0: speaker's 2 dialogue act

q3 → q0: speaker's 2 dialogue act

q3 → q3: speaker's 2 dialogue act

The integrated generalised 4-stage automata is shown on Figure 32. Putting the output into the time context, the automaton could generate such output in reference to each of the speakers:

Time:	→ →	→ →	→ →	→ →	→ →	→ →	→ →
Spk1 output:	DA				DA		DA
Spk2 output:			DA	DA		DA	

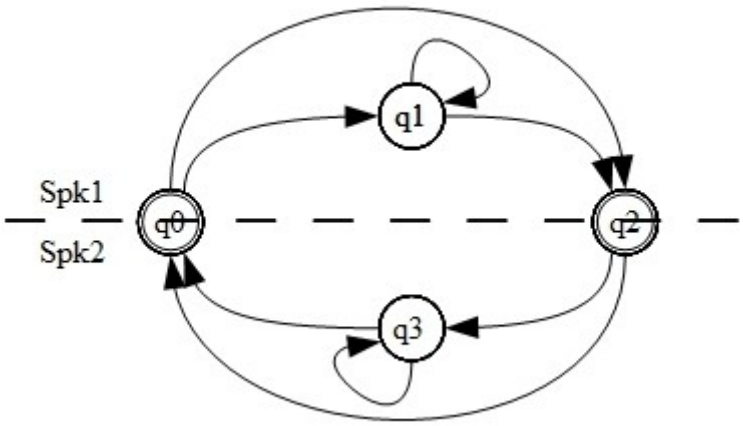


Figure 32: Integrated generalised linear 4-stage turn automata for two speakers

Such visualisation *immediately* uncovers the problem of overlapping speech, therefore the automata is called 'linear' as it does not generates the overlapping speech. Simply, the

finite state automata generate sequences and not overlapping outputs. Therefore, the building automata rules had to be violated and an integrated generalised “overlapping” 4-stage turn automaton for two speakers was created, which is presented on Figure 33. Such automaton generates simultaneously outputs for two different speakers separately, moving between 4 stages, where 2 stages are common, and 2 are exclusive. Precisely:

q0: initiation state

q0, q2: termination states

q0 → q1: speaker’s 1 dialogue act

q1 → q1: speaker’s 1 dialogue act

q1 → q2: speaker’s 1 dialogue act

q0 → q2: speaker’s 1 dialogue act or speaker’s 2 dialogue act

q0 → q3: speaker’s 2 dialogue act

q3 → q3: speaker’s 2 dialogue act

q3 → q2: speaker’s 2 dialogue act

q2 → q0: speaker’s 1 dialogue act or speaker’s 2 dialogue act

However, it has to be underlined that the integrated generalised “overlapping” 4-stage turn automata for two speakers is designed against the linear rule of the automata sequential output. Here, it could be imagined that the output would be:

Time:	→ →	→ →	→ →	→ →	→ →	→ →	→ →
Spk1 output:		DA	DA		DA	DA	
Spk2 output:	DA	DA		DA		DA	

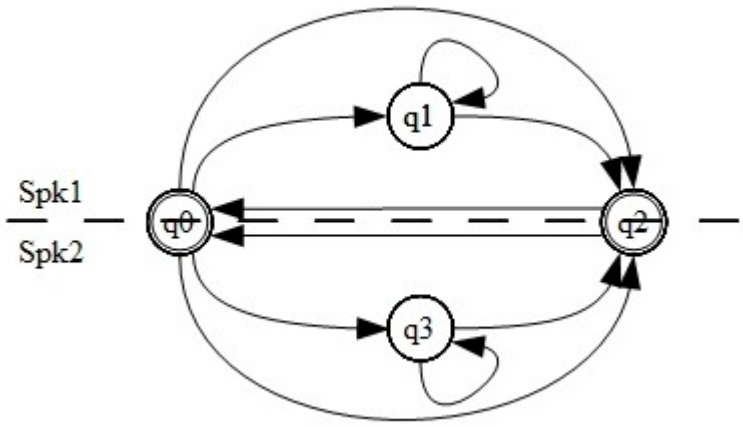


Figure 33: Integrated generalised "overlapping" 4-stage turn automata for two speakers

Dialogue analysis and automata creation showed that not only sequences of dialogue acts are important, but also the time context, as speakers do not change turns successively, but also interrupt each other which leads to overlapping speech which cannot be modelled with a finite state automaton.

5.11 Summary

In this chapter extensive work on finite state automaton creation based on analysis of spoken dialogue was presented. Applying dialogue act categories to finite state automata makes it possible to generalise work on dialogue acts to new scenarios. The finite state automata were successfully evaluated in operational tests using two main criteria:

1. Completeness: how much of the data does the automaton cover? Does it cover all the data and also possible items which are not in the data? I.e. does the item under-generalise (for parts of the data) or generalise correctly?
2. Soundness: does the automaton cover items which are not in the data and are not possible? I.e. does the item over-generalise and produce unacceptable sequences?

The findings from this work will serve for dialogue management in the demonstration dialogue system.

Chapter 6: Speech synthesis module

6.1 Chapter overview

The present chapter reports on the research on several existing text and speech corpora in preparation for creating new linguistic material for the dialogue system. Despite the fact that the Polish language is well documented, creating a new speech corpus for the use for synthetic voice creation was inevitable. To minimise the manual work, two automatic tools are developed, which can be applied to other applications and to other languages.

The procedure of creating a Polish male voice for a diphone based synthesiser is presented and evaluation of this voice is performed in speech perception tests. This task is undertaken because no Polish male synthetic voice for the target diphone based synthesiser is available and its usage is planned for the demonstration dialogue system implementation.

6.2 The role of speech synthesis

In this study, speech synthesis plays two roles:

1. Data-driven microvoices (synthetic voices constructed from restricted data sets) for experimentation with annotated speech corpora together with close-copy synthesis.
2. Full Polish male voice for use with the spoken dialogue demonstration prototype system.

The utterance units in the annotated dialogue resulting from the corpus linguistic study were used to make microvoices for testing purposes, including the evaluation of annotation quality.

The main speech synthesis task in this study aims at developing a demo module of the speech generator of the dialogue system. The idea in the dialogue system is to select a set of real utterances (and information about what category of the dialogue acts they represent) and synthesise them.

The first design choice was to specify basic requirements for selecting a speech synthesis system; the system which fits the requirements best is the MBROLA diphone synthesis system:

1. Free software.
2. Comprehensive documentation (Dutoit et al. 1996; Dutoit 1997[2]).
3. Suitability for multilingual speech synthesis (73 voices for 36 languages are publicly available on the internet).
4. Cross-platform availability (runtime binaries for 37 operating systems and operating system versions are available, including the required Linux and Windows versions).
5. Offline use, independence from internet tools.
6. Ease of installation.
7. Reasonable quality in relation to the other requirements.
8. Simple interface between text parser and diphone synthesis components.

MBROLA was developed in the mid-1990s and is not a state of the art system any longer. More sophisticated techniques such as unit-selection synthesis and HMM synthesis are now used and achieve higher quality, there is no other speech synthesis system which fulfils requirements 1-8 above. The most important criterion from the point of view of this study is criterion 8, the simple interface between text and synthesis processing components, which allows duration and pitch to be processed very easily.

Using the MBROLA synthesiser it was possible to combine the microvoice and the full voice technique: interface files were prepared using the microvoice technique, and these were modified and used with the full Polish voice in the demonstrator system.

The reason for making a new full Polish voice was to be able to align the speech dialogue system with the caller in the emergency scenario: the existing MBROLA voice is for a female speaker, so the new voice was developed using a male speaker. In this way, a choice of gender was available for the system.

2 Cf. also the MBROLA project web page: http://tcts.fpms.ac.be/synthesis/

6.3 Synthesis experiment with corpus linguistic analysis

6.3.1 MBROLA micro-voice creation

The corpus created for the corpus linguistic dialogue analysis was used for making MBROLA microvoices, which were used for Automatic Close Copy Speech synthesis (Bachan 2007a). The ACCS procedure is described after the following description of microvoice creation. The main steps in MBROLA microvoice creation are described by Bachan (2008). The main features are reviewed here. In order to create MBROLA micro-voices, the following steps are required:

1. Manual procedures:
    1. Selection of the data for the purpose of experimentation.
    2. Recording of the data.
    3. Annotation of the data at the phone/phoneme level (also performed automatically if such tools available)
2. Computed procedures:
    1. Analysis of the annotation files and construction of a table of diphones with their beginning and end points (developer: Bachan).
    2. Extraction of individual speech files for each diphone using the diphone table and the speech recording file (developer: Bachan).
    3. Application of the MBROLA voice creation software 'Mbrolator' (developers: Pagel & Dutoit – MBROLA creators).
    4. Creation of MBROLA text-speech interface files ('pho files') from the annotations, for testing purposes by Automatic Close Copy Speech (ACCS) synthesis (developer: Bachan).

The software tools developed by the author were developed independently for each speaker in the dialogue, from each speaker's channel. The diphones were extracted automatically using a Python tool. The diphones and information about the diphones were then input to the MBROLATOR, a program for MBROLA voice creation. As a result, two MBROLA micro-voices were created, containing diphone sets would would allow to

synthesise the utterances which occurred in the recording. Figure 34 visualises the process of MBROLA voice creation.

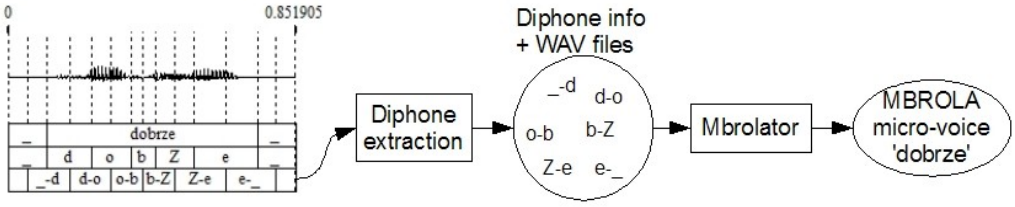


Figure 34: Mbrolation, the MBROLA micro-voice creation procedure.

6.4 Automatic Close Copy Speech synthesis

Automatic Close Copy Speech (ACCS) synthesis with an MBROLA type diphone synthesis is a process of automatically creating pronunciation specification tables (NLP-DSP interfaces[3], implemented as PHO files), making use of recorded and annotated real utterances, and synthesising the pronunciation specification tables using an appropriate voice (diphone database). The voice may be created from the annotated utterances, or may be an independently created voice. In the present study both cases of the voice are used.

The idea of the study was to select a set of utterances from the analysed dialogues coming from the PoInt corpus (Karpiński 2002) and provide PHO files of those utterances, however the work did not get so far. What has been done is a demo synthesis of an excerpt of dialogue which was chosen for the phonemic transcription and used for MBROLA micro-voice creation (described above).

The turns of speech for each interlocutor were ACCS synthesised separately with MBROLA using their micro-voices. Because it was not possible to cut out first diphone in the first utterance by one of the speakers, this utterance could not be synthesised. After the speakers' turns were synthesised, the speech signals were put together to a stereo sound and created a dialogue.

Additionally, because of the problem with one missing diphone, the pl1 female MBROLA voice was used to synthesise all utterances which occurred in the dialogue excerpt. To make the two speakers sound different, the F0 values for one of the

3 NLP-DSP stands for Natural Language Processing – Digital Signal Processing modules of a text-to-speech (TTS) system.

interlocutors were divided by 2 to be more similar to the male F0 values. Such a method was successfully tested and evaluated by Bachan (2007b).

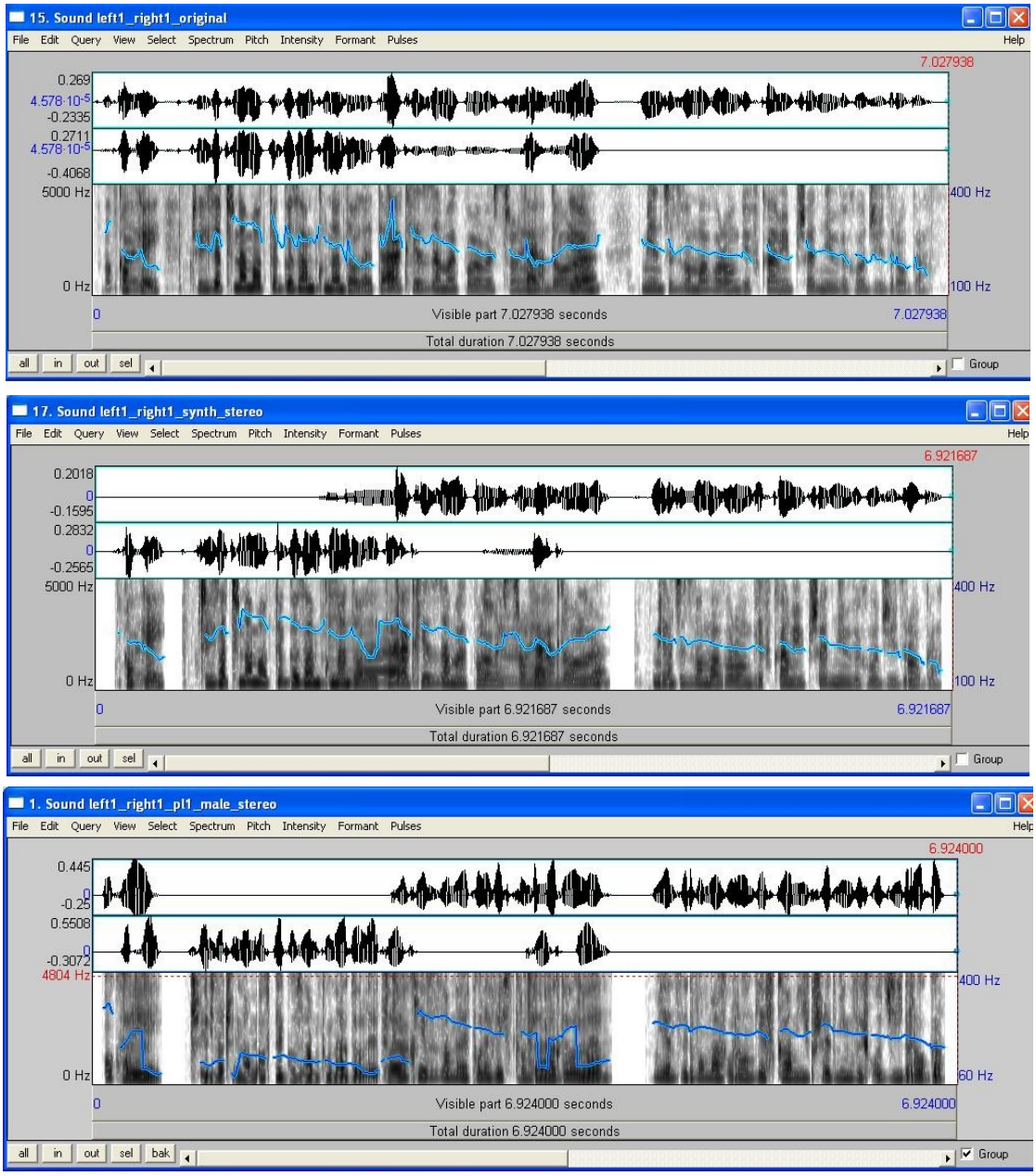


Figure 35: Comparison of original recording with microvoice and PL1 female voice

The waveforms, spectrograms and pitch contours of the original dialogue recording (top), its synthesis with the MBROLA micro-voice (middle) and with the pl1 female voice (bottom) are presented on Figure 35.

6.5 MBROLA full voice creation

6.5.1 MBROLA data flow architecture

The data flow for creating an MBROLA voice is shown in Figure 36. The recorded speech signal is annotated, and the annotations can be used for either creating microvoices and using the ACCS microvoice-based procedure, or for creating the full voice Further details are given by Bachan (2010).

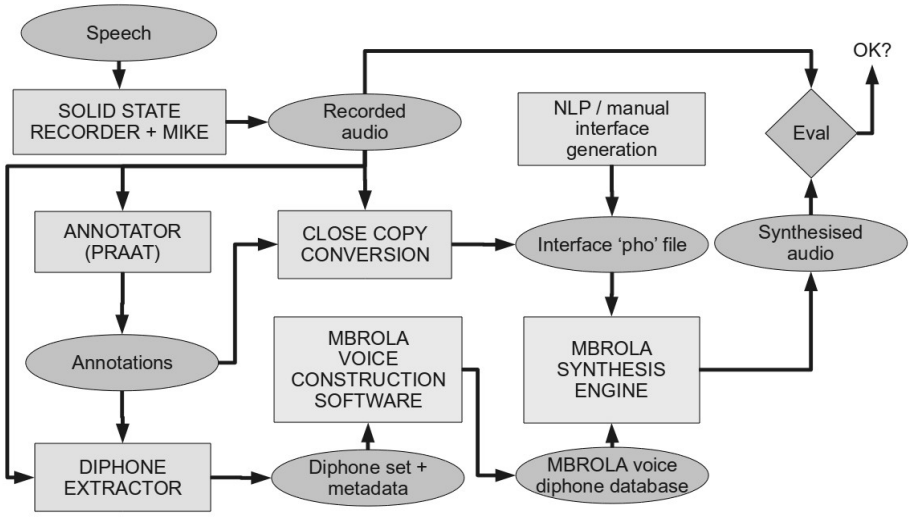


Figure 36: Data flow chart for MBROLA voice creation and runtime synthesis.

6.5.2 Corpus specification

The Polish male MBROLA voice pl2 was created for use in the spoken dialogue demonstration prototype system. The voice creation procedure, and the tools used for the MBROLA voice creation, are described.

The reasons for developing the male voice have been indicated already. For the Polish language only a Polish female voice was available. The original aim of the present study was to create a male voice for the use in a dialogue system being developed for a police emergency call centre; this aim was later changed to the creation of a call centre for hospital emergencies, for which a male voice is also in general a realistic requirement, but for which a choice of male and female gender is desirable. A technical reason for developing a new voice is that the diphone set of the older female voice is based on an

older version of the phoneme set for Polish which contains 37 phonemes, whereas more recent work shows that 40 phonemes are required (Jassem 2003, Demenko et al. 2003, Table 30), so this mismatch with existing data was another reason why the new MBROLA voice was created. However, the mapping of the extended phoneme set onto the older phoneme set is straightforward, with minimal loss of information in converting from 40 to 37 units. So for the present work, after creating the first MBROLA voice it was not a problem to create a second voice from the same recorded data, one with the extended phoneme set and the other with the older phoneme set. The voice using the older phoneme set was created to make the existing tools originally developed by Bachan (2007a) for the older voice also compatible with the new male voice. An additional advantage of having two voices with the same phoneme specifications is that they can be switched directly in the actual application without any runtime modification.

Table 30: Extended SAMPA phoneme labels used for annotation (Demenko et al. 2003)

BLF Polish modified SAMPA	***Orthography***	***Phonemic transcription***	***BLF Polish modified SAMPA***	***Orthography***	***Phonemic transcription***
p	pik	pik	i	kit	kit
b	byt	byt	y	typ	typ
t	test	test	e	test	test
d	dym	dym	a	pat	pat
k	kat	kat	o	pot	pot
g	gen	gen	u	puk	puk
c	kiedy	cjedy	@ - *schwa*		
J	giełda	Jjewda	m	mysz	myS
f	fan	fan	n	nasz	naS
v	wilk	vilk	n'	koń	kon'
s	syk	syk	N	pęk	peNk
z	zbir	zbir	l	luk	luk
S	szyk	Syk	r	ryk	ryk
Z	żyto	Zyto	w	łyk	wyk
s'	świt	s'fit	j	jak	jak
z'	źle	z'le	w~	ciąża	t^s'ow~Za
x	hymn	xymn	j~	więź	vjej~s'
t^s	cyk	t^syk			
d^z	dzwon	d^zvon			
t^S	czyn	t^Syn			
d^Z	dżem	d^Zem			
t^s'	ćma	t^s'ma			
d^z'	dźwig	d^z'vik			

The diphones needed for creating the MBROLA voice, also called the MBROLA diphone database, were originally to be extracted from the existing BOSS corpus (Demenko et al. 2007, Demenko et al. 2008). But very many diphones were missing in this corpus (about 30% of the total number of possible Polish diphones), so it was not possible to create the full male MBROLA voice. However, the material was used to design the tools for automatic diphone extraction, automatic diphone extraction evaluation and for creating a prototype voice for testing, with about 1000 diphones.

For the actual MBROLA voice used in the spoken language demonstration prototype, a new corpus was designed, recorded and annotated.

6.5.3 Text corpus creation

The core of a diphone synthesiser is the diphone database (also called the voice). Creating the database must be well thought-out, because it has to include all the possible diphones in the language. In practice, creation of the corpus for constructing the diphone database is mainly achieved in four steps:

1. Creating a text corpus:
    1. A list of phones (PL), including main allophones if possible, is prepared for the language.
    2. Out of the list of phones PL a list of diphones (DL) is generated: $|DL| \leq |PL|^2$.
    3. Alternative procedures:
        1. A list of words containing all the diphones is created. Each diphone should appear at least once; diphones in such positions as inside strongly stressed syllables or in strongly reduced (i.e. over coarticulated) contexts should be excluded, and the key words are put in a carrier sentence.
        2. A large text corpus is collected and converted (usually automatically) into phonemic representation:
            1. The diphones are extracted and checked for completeness. If the diphone set is incomplete, more sentences are added (or created for the purpose) until all diphones are included.

2. The smallest possible set of sentences containing all the diphones is extracted from the corpus.

2. Recording the corpus:
    1. The corpus is read by a professional or other experienced speaker, if possible with nearly monotonous intonation.
    2. The speech is digitally recorded and stored in a digital format.
3. Segmenting the corpus:
    1. The diphones must be found in the corpus and annotated either manually or automatically by the means of automatic segmentator.
    2. The position of the borders between the phones is marked.
    3. With additional software routines the middle points of the phones are marked in order to be able to create diphones.

When the diphone set for voice creation is made, it also requires metadata about the individual diphones in a table with the following information types:

1. diphone name,
2. diphone speech waveform file name,
3. interval durations: beginning of the diphone from the beginning of the diphone file, mid point of diphone, end of diphone. This format allows the duration of the two phones in each diphone to be modified independently.

For the creation of the new Polish voice the second alternative text procedure was used: the aim is to reduce the number of sentences to be read in order to be able to produce a set of speech data recordings efficiently. The full set of diphone pairs has 1444 members (for 37 phones), but only about 1158 are possible in Polish. Using the phonetically rich sentence reduction algorithm (Bachan 2010), instead of 1158 sentences carrying one target diphone each, much fewer phonetically rich sentences may be recorded, 211 phonetically rich sentences in this case (and 156 target-diphone phrases). This ratio of about 7:1 means a great increase in efficiency for recording time and for annotation time.

The tool is to extract the diphones from an already annotated speech corpus automatically, regardless the original purpose of the corpus.

6.6 The Mbrolator software

The Mbrolator is a software suite for MBROLA voice creation, consisting of a library of tools with three user interface tools for parameter setting (UNIX shell script), metadata identification (Perl Makefile creator script) and a voice creation tool (UNIX make script).

During the make process, the Mbrolator software performs normalisation operations: the energy levels at the beginning and at the end of a segment are modified in order to eliminate amplitude mismatches – the energy of all the phones of a given phoneme is set to phones' average value, and the pitch is normalised in order to make pitch calculation more efficient at runtime (Dutoit 1997).

The requirements of the system are diphone files in the WAV format and diphone database file in the SEG format. The restrictions put on the diphone files are:

1. the diphone WAV files need to be at 16kHz sampling rate;
2. the diphone WAV file cannot be longer than 10,000 samples;
3. for each diphone a context of 500 samples needs to be left on the left and on the right sides, for signal processing purposes in pitch extraction and normalisation.

When the corpus is created and diphones extracted, the diphone database file in the SEG format contains information about: the name of the diphones, the corresponding waveforms, their durations and internal sub-splitting.

6.7 The phone and diphone sets

6.7.1 Phoneme set

The phoneme set for the extended SAMPA was presented in Table 30 and shows 40 phoneme labels. The phoneme set for the older version of the Polish SAMPA is smaller and contains 37 phonemes. The phoneme set is presented in Table 31.

Table 31: Polish SAMPA transcription used in the PL1 Polish female MBROLA voice (Szklanny & Marasek 2002)

PL1 Polish SAMPA	***Orthography***	***Phonemic transcription***	***PL1 Polish SAMPA***	***Orthography***	***Phonemic transcription***
p	pik	pik	i	kit	kit
b	bit	bit	I	typ	tIp
t	test	test	e	test	test
d	dym	dIm	a	pat	pat
k	kat	kat	o	pot	pot
g	gen	gen	u	puk	puk
f	fan	fan	e~	gęś	ge~s'
v	wilk	vilk	o~	wąs	vo~s
s	syk	sIk	m	mysz	mIS
z	zbir	zbir	n	nasz	naS
S	szyk	SIk	n'	koń	kon'
Z	żyto	ZIto	N	pęk	peNk
s'	świt	s'fit	l	luk	luk
z'	źle	z'le	r	ryk	rIk
x	hymn	xImn	w	łyk	wIk
ts	cyk	tsIk	j	jak	jak
dz	dzwon	dzvon			
tS	czyn	tSIn			
dZ	dżem	dZem			
ts'	ćma	ts'ma			
dz'	dźwig	dz'vik			

The differences between the sets and their labels is presented in Table 32.

Table 32: Mismatches between BLF and PL1 SAMPA

Extended SAMPA annotation labels	***Older SAMPA symbols***	***Extended SAMPA annotation labels***	***Older SAMPA symbols***
p	p	i	i
b	b	y	I
t	t	e	e
d	d	a	a
k	k	o	o
g	g	u	u
c	-	*@ - English schwa*	-
J	-	-	e~
f	f	-	o~
v	v	m	m
s	s	n	n
z	z	n'	n'
S	S	N	N
Z	Z	l	l
s'	s'	r	r
z'	z'	w	w
x	x	j	j

Extended SAMPA annotation labels	*Older SAMPA symbols*
t^s	ts
d^z	dz
t^S	tS
d^Z	dZ
t^s'	ts'
d^z'	dz'

Extended SAMPA annotation labels	*Older SAMPA symbols*
w~	-
j~	-

6.7.2 Diphone set

An MBROLA voice for any language should contain a set of all diphones allowed in the language. In the extreme case the number of diphones would be equal PHONELIST x PHONELIST, i.e. each phone would exist in combination with all phones. However, it is not always the case, because all phone combinations are not always allowed. For the male MBROLA voice development, two sets of diphones were created, one based on the Polish extended SAMPA and the other on the older SAMPA set.

Taking a set of 40 phonemes for the Polish language and adding to it a pause gives us a set of 41 elements. In the best case the BOSS corpus (Demenko 2007) should contain 1681 diphones. However, the analysis of the BOSS corpus showed that there are only 1083 diphones available. The lack of the missing diphones does not allow to create a full MABROLA voice. The diphones need to completed before the full MBROLA voice is created.

The diphone set based on the older SAMPA phoneme set would be equal all phonemes plus a pause, i.e 38 elements x 38 elements, which is 1444.

6.7.3 Search for diphones

A tool was designed in order to find the diphones needed for creating the MBROLA diphone database for the male pl2 voice.

The original diphone search was performed on sets of recorded data which were created for different purposes in the last 5 years. The corpora/data consisted of raw texts, recordings with texts of read speech and professionally described recordings with annotations. The archivisation of the data sets was different and therefore some of the diphones exist only in text, which means that the recordings of those diphones were not available at the searching stage. This section presents the data, the preparation of the data,

and the technique and results of the diphone search. Later, a new and more complete corpus was created.

For the original diphone searching, different sets of recordings of the same male voice were used. The sets of recordings were made for different purposes and their specifications were different. The recordings had been made in a professional studio and were read by a male professional speaker. The original corpus of recordings for a male voice was for a unit-selection speech synthesis development scenario for the BOSS speech synthesis system (Demenko 2007). The texts were spoken by a professional speaker and the recordings were made in a professional recording studio. The sampling rate of the data in the available format is 16kHz in a standard WAV format. The corpus consisted of approximately 3240 utterances.

Diphone search was performed on annotations for these sets of corpora. The examined resources were available as:

1. recordings with annotations,
2. only annotations,
3. recordings with text of read sentences
4. only texts of read sentences

These resources underwent thorough examination to search for all the diphones which could be produced in the Polish language.

In the corpus of recordings five bases with different kinds of sentences are found (for more information see Bachan 2007b):

1. Base A – 367 sentences with most frequent Polish consonant clusters.
2. Base B – 114 meaningless sentences with ‘*all* ’Polish diphones.
3. Base C – 676 short sentences grouped into four, in each of which the same keyword appears. The keywords contain Polish triphones CVC in voiced context and in different intonation patterns.

4. Base D – (available) 200 sentences grouped into six, in each of which the same keyword appears. The keywords contain Polish triphones CVC in sonorant context and in different intonation patterns.
5. Base E – (available) 67 compound sentences with most frequent words from the vocabulary.

6.7.4 Annotation of the original synthesis corpus

Annotation of the recordings coming from the BOSS corpus (Demenko et al. 2007, Demenko et al. 2008) at phoneme level was performed automatically using the software tool SALIAN (Szymański & Grocholewski 2005) and checked by trained phoneticians. Phonemic segments which were not correctly handled by the automatic segmentator were manually edited. Additionally, the annotations also contain prosodic information, based partly on functional judgements and partly on prosodic information. The annotation uses the following information types (for more information cf. Bachan 2007b) :

1. Sample serial numbers (column 1).
2. Phonemic/allophonic label tier (column 2):
    1. Labels for 40 phonemes. Table 30 shows a list of phoneme labels used in the annotation (Demenko et al. 2003: 85, cf. Jassem 2003: 103, 105).
    2. Stress and accent types (Demenko et al. 2006: 462):
    3. Word and syllable boundaries (spaces indicate line breaks in the annotation files):
    4. Four additional labels, including labels for paralinguistic information:
    5. Prosodic tier (column 3) - Prosodic phrase boundary labels:

6.7.5 Annotation file format

The annotations are in the BOSS Label File (BLF) format, designed for the Bonn Open Speech Synthesis (BOSS) system (Klabbers et al. 2001). Table 33 shows the structure of the BLF annotation file. The file represents a three column matrix, with sample numbers in the first column, an allophonic representation including word and syllable boundary allophones and lexical stress types in the second column, and a prosodic boundary

representation in the third column. The use of sample numbers and not time stamps makes additional knowledge of sampling rate metadata necessary. The table represents the first part of the Polish sentence *Na szczęście myśl o przeprowadzce była tylko chwilowa i Gosia będzie nadal z nami mieszkać*. (En. *Fortunately, the idea of moving out was only temporary and Gosia will be still living with us.*) from the corpus.

Table 33: Fragment of BLF file input resource.

Sample number (16 kHz rate)	***Segmental labels***	***Prosodic labels***
0	#$p	
5798	#n	-5,.
6863	a	
8008	#S	
9312	t^S	
10047	"e	
10880	j~	
11351	.s'	
12640	t^s'	
13634	e	
14481	#$jm	
15613	|y	
16235	z'	
17214	l	
18843	#o	

In the phonemic/allophonic annotation label column, the following conventions are applied:

1. [#] encodes the beginning of a word, e.g. [#n] stands for a word-initial allophone of the phoneme /n/,
2. [.] encodes the beginning of a syllable, e.g. [.s'] stands for a syllable-initial allophone of the phoneme /s'/,
3. ["] denotes falling accent realised by F0 fall on postaccented syllable/syllables or F0 interval between accented and postaccented vowels, e.g. ["e] stands for the accented allophone of the phoneme /e/ with falling accent,
4. [#$p] stands for a pause ([#$p] is always inserted at the beginning and at the end of a sentence and can also appear in the middle of a sentence),
5. label [#$jm] is read as

1. [#m] - word-initial allophone of the phoneme /m/,
2. [$j] - a segment not to be used for the speech synthesis; the whole word is ignored for the purposes of speech synthesis.

In the prosody label column information about the type of utterance is represented:

1. [-5,.] indicates the beginning of a sentence with falling intonation.

For further information cf. section above and cf. Demenko et al. (2006).

However, the BLF format has one drawback: it is not possible to calculate the duration of the whole recording from the information included in the annotation file. In the BLF annotation file, only the beginnings of the segments are marked, starting with 0 for the first segment. This means that for the last segment only the beginning is marked, without the information on how long the segment lasts. The last segment in all recordings in the BOSS corpus is always a pause. Calculating the durations of the whole recording based on the annotation file is therefore not possible, because the information about the duration of the final pause is not included in the BLF annotation file.

The BOSS corpus contains about 3300 BLF files available, but only 1580 WAV files.

The BOSS corpus with the BLF annotation files was searched through using the Python scripts created for the automatic diphone extraction (Python scripts are discussed in detail in 6.9.6 – 6.9.9). The diphone extractor contains a set of scripts:

1. *BLF2TextGrid.py* – converts the BLF annotation files to the TextGrid format.
2. *PE-TextGrid2PL1-TextGrid.py* – converts the broad SAMPA phoneme set in the annotation files to the PL1 MBROLA voice phoneme set.
3. *FindDiphonesInTextGrids.py* – finds diphone pairs and creates DIPH file for each TextGrid file.
4. *CutOutIndividualDiphones.py* – cuts out *one* instance of each diphone from all the WAV files in a directory, creates the SEG file and a TXT file with information about which phones could not be cut of if the left or right context of the diphone was less than 50ms.

5. *CutOutAllDiphones.py* – cut outs *all* instances of the diphones from all the WAV files in a directory, creates the SEG file and a TXT file with information about which phones could not be cut of if the left or right context of the diphone was less than 50ms.
6. *ConcatenateDiphones.py* – cocatenate diphones being cut out from the same file.

In the diphone search procedure, the script #5 and #6 were not used. The script #4 was modified when only run on the annotations, so that the diphones were not actually extracted from the WAV files, but they were found in the annotation files. 1097 diphones were found in the annotation files, some of which were not correct. The incorrect diphones come from the mistakes in the annotations. The total of 1097 diphones is found in the BOSS corpus of annotations, which is less than the total number of possible diphones for Polish.

6.7.6 Search procedure in available diphone database

Before the availabe diphone database was examined, the target diphones which were in the database spreadsheet were looked at to get an overview of what the 4 sub-databases contain. The next step was to find how many other diphones, apart from the target diphones, the databases contain. In order to do that, the database was prepared in the following manner:

1. Creating separate text files for the recordings for which the annotations were not provided taking the information about the read text (carrier phrase) from the spreadsheet.
2. Automatic annotation: requires WAV file and TXT file with the text which was recorded. Automatic annotator does the following:
    1. converts graphemes to phonemes – the phoneme set used by the automatic annotator has 40 phonemes, i.e. the broad SAMPA set;
    2. adds word and syllable boundary markers;
    3. adds word-accent marker.

Automatic annotation was performed by the SALIAN tool (Szymański & Grocholewski 2005) which is integrated into the Annotation Editor program (Klessa 2006). It generates annotation files in the BLF format.

3. BLF format to TextGrid format conversion: requires BLF and corresponding WAV files.
4. Conversion of the extended phoneme set (broad SAMPA) from the automatic annotation to the pl1 phoneme set.
5. Converting TextGrid files to DIPH files – the DIPH format is used for the diphone search.
6. Diphone search in DIPH files.

The steps #3-6 were performed by the software created for the automatic diphone extraction. The C-C diphone and V-V diphone sub-databases did not undergo such examination, because the read text, i.e. the carrier phrases of the recorded speech was not provided. Therefore, the only information about the diphones contained in the recordings was taken from the WAV filename, which contained the target diphone or triphone, e.g. bb.wav or dZb.wav.

About 350 recordings were corrected manually and therefore the annotation in this corpus is not fully automatic.

The investigation of the other databases brought the following results:

1. Basic database: 988
2. Broad database: 726
3. C-C diphones: 184 different diphones and triphones from the broad SAMPA
4. V-V diphones: 37

Taken together, these diphones did not account for the complete Polish diphone set.

6.7.7 Diphone search in synthesis text and online.

In 2010 the same speaker who provided his voice for the BOSS system was recorded again in order to fill the gap of the missing triphones. The recordings of his speech were not yet

processed, however, but the read text was available. This text was used to search for diphones, which should be present in the raw recordings.

The raw text was first transcribed using the SALIAN (Szymański & Grocholewski 2005) grapheme-to-phoneme converter and then examined using a UNIX shell script for diphone search. The diphone search resulted in finding 1026 diphones from the extended phoneme set.

The diphone set found in the text used for recordings was too small, so a different strategy was used, using the same procedure with several thousand lines of online newspaper text, in which 1100 diphones were found.

The diphone search showed that the maximal number of the diphones which can be found in the available corpora and databases is 1217.

6.8 Phonetically rich sentence extractor

The objective is to select the smallest possible set of sentences from a text corpus which will contain the largest number of diphones. Such a sentence set may be then recorded and annotated, the diphones extracted and a new MBROLA voice may be created. Usually, in order to collect diphones, researchers create a large set of sentences aimed at containing one target diphone in a carrier sentence. The phonetically rich sentence extractor searches for diphones in a text corpus and selects only the sentences which contain the most new diphones in relation to those chosen before.

6.8.1 Diphone set creation

A full MBROLA voice should contain all the diphones of a given language. A diphone (or a dyad) is a unit that begins in the middle of the stable state of a phone and ends in the middle of the following one (Dutoit 1997). The theoretical number of diphones in a given language is $|DL| \leq |PL|^2$, where DL represents a diphone list and PL represents a phone list. However, not all the phones appear in all phone contexts, therefore the actual number of diphones for a natural language may be smaller.

The calculation of the diphones was done to find out the optimal number of diphones in Polish. For the Polish language, two sets of phones are accepted, both being applied to the SAMPA alphabet (Wells 1997): the Polish SAMPA (Wells 1996) and the Polish

Extended-SAMPA (Demenko et al. 2003, Jassem 2003). The Polish SAMPA contains 37 labels. Adding the pause to it, it gives a maximum of 38*38=1444 diphones. The Polish Extended-SAMPA contains 40 labels, with the pause it gives 41*41=1681 diphones.

6.8.2 Available text resources

For the present study, a set of sentences created for speech technology purposes were used. 1623 sentences were taken from the Bonn Open Synthesis System (BOSS) corpus (Demenko et al. 2007) and 8828 sentences came from the Jurisdic database (Demenko et al. 2008), a database made for a speech recognition system creation. Altogether 10451 sentences were written orthographically and saved in a text document in the ANSI encoding, which is required by the automatic transcription software. The sentences were then transcribed using SALIAN (Szymański & Grocholewski 2005).

6.9 Software

6.9.1 Sentence extraction procedure

In Figure 37 the phonetically rich sentence extraction procedure is presented. First, the sentences are automatically transcribed using SALIAN (Szymański & Grocholewski 2005). Then, the sentences are divided into diphones and the number of diphones for each sentence is calculated. Next, the sentences are sorted according to the descending number of diphones. Having the sorted list, the program creates an empty set of diphones and adds to it the diphones which occur in the selected sentences: The algorithm compares the diphones in the processed sentence. If it contains new diphones which have not occurred before, the sentence is selected and the new diphones are added to the diphone set.

The algorithm is designed in such a way that first it can select only the sentences which contain 10 new diphones or more. Then the number may be decreased by 1 in a loop until 1. In this way all the sentences are checked and selected even if they contain only 1 new diphone. However, the precedence is given to the diphone richest sentences.

6.9.2 Results of sentence extraction

The phonetically rich sentence extractor was run on the available text resources. According to the Polish SAMPA, 1008 diphones were found in the corpus and 211 sentences were

selected out of 10451. When the Polish Extended-SAMPA phone set was applied, 1095 diphones were found and 201 sentences were selected from the text corpus.

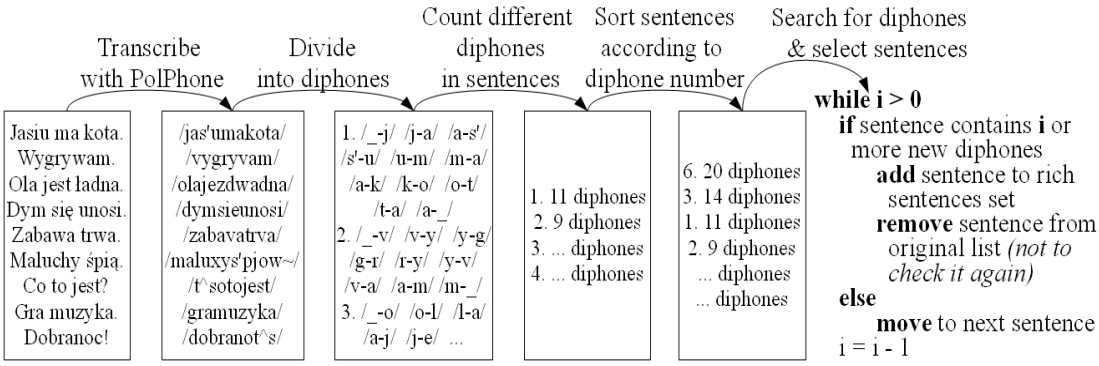


Figure 37: Phonetically rich sentence extraction procedure

The phonetically rich sentences are given in Appendix K.1. In Appendix K.2 the word list with target-diphones which exist in Polish, but were not found in the automatically selected sentences is included.

6.9.3 Automatic diphone extraction system architecture

For automatic diphone extraction a system was designed to automatically cut out diphones from the recordings based on the annotations of those recordings. In the first stage the system searches for the diphones in the annotations on the phone level and cuts them out from the recordings, creating at the same time a database file with information about those diphones. The information in the database file concerns, for example, the labels of the diphones which have been extracted, the names of the files in which the individual diphones are, the placement of the boundary between the half-phones.

The extracted diphones are then put forward to the evaluation stage in which both, the database file and the diphone files are evaluated.

Based on the database file annotation files for separate diphones are created. These diphone annotation files allow to manually compare the annotations with the signal in the diphone files to see if the annotation is correct with the signal. Additionally, the manual investigation of the diphones with their annotations allows to evaluate the quality of the diphone and select diphones of the best quality for the synthetic voice creation.

The next step in the diphone extraction evaluation is the concatenation of the extracted diphones. The extracted diphones from one utterance are glued back together. In the ideal case, the concatenated diphones extracted from one recording with preserved order of the diphones' occurrence should give a synthesised speech output identical with the original recordings, without any glitches or repetitions. If the extraction of diphones were not correct, the synthesised speech signal would be disrupted.

The overall architecture of the automatic diphone extraction system is presented on Figure 38.

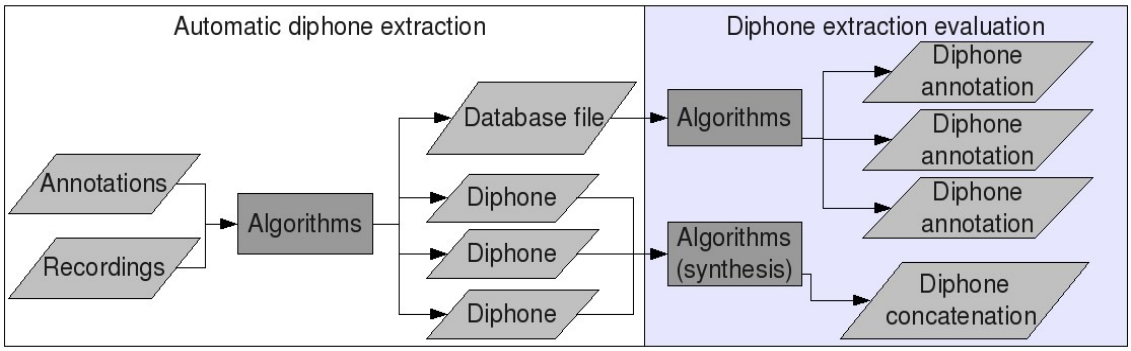


Figure 38: Architecture of the automatic diphone extraction system

6.9.4 Automatic diphone extraction system design

The specificity of the BFL annotation files, i.e. the fact that the BLF format is only used for annotation of the corpora for the BOSS system and the BLF drawback connected with the lack of the information about the length of the annotated recording (see 6.7.5 Annotation file format above) triggered the conversion of the BLF annotation files to more commonly used Praat TextGrid file format. If the decision is to make a diphone database with the older version of a SAMPA set of Polish phonemes, then the set of phonemes in the TextGrid files, which contain the Polish extended SAMPA set (PE-SAMPA TextGrid) from BLF files, may be then converted to the older version of Polish SAMPA set creating new TextGrid files (SAMPA TextGrid). From the TextGrid annotation files information only about the diphones (or more precisely pairs of phones) and their boundaries are extracted and input to the DIPH file (see 6.9.8 Find all diphones in TextGrid files below). Based on the information from the DIPH file, the diphones are extracted and a database file in the

SEG format is created. The design of the automatic diphone extraction software is presented on Figure 39. The conversion flow of the text files into different formats is presented on Figure 40.

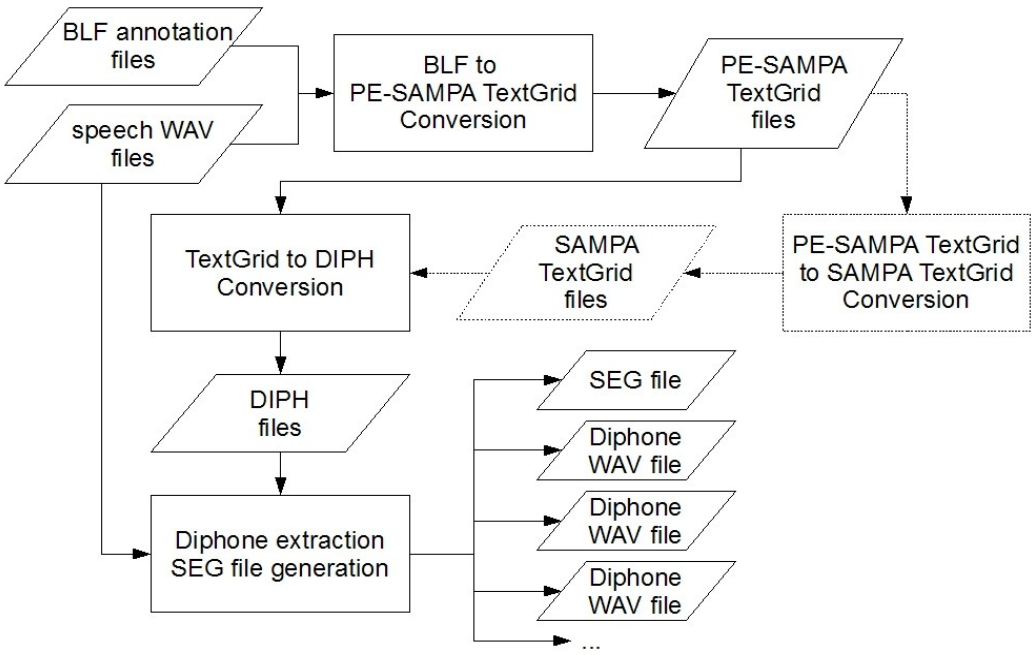


Figure 39: Design of the automatic diphone extraction software. PE-SAMPA – the Polish extended SAMPA

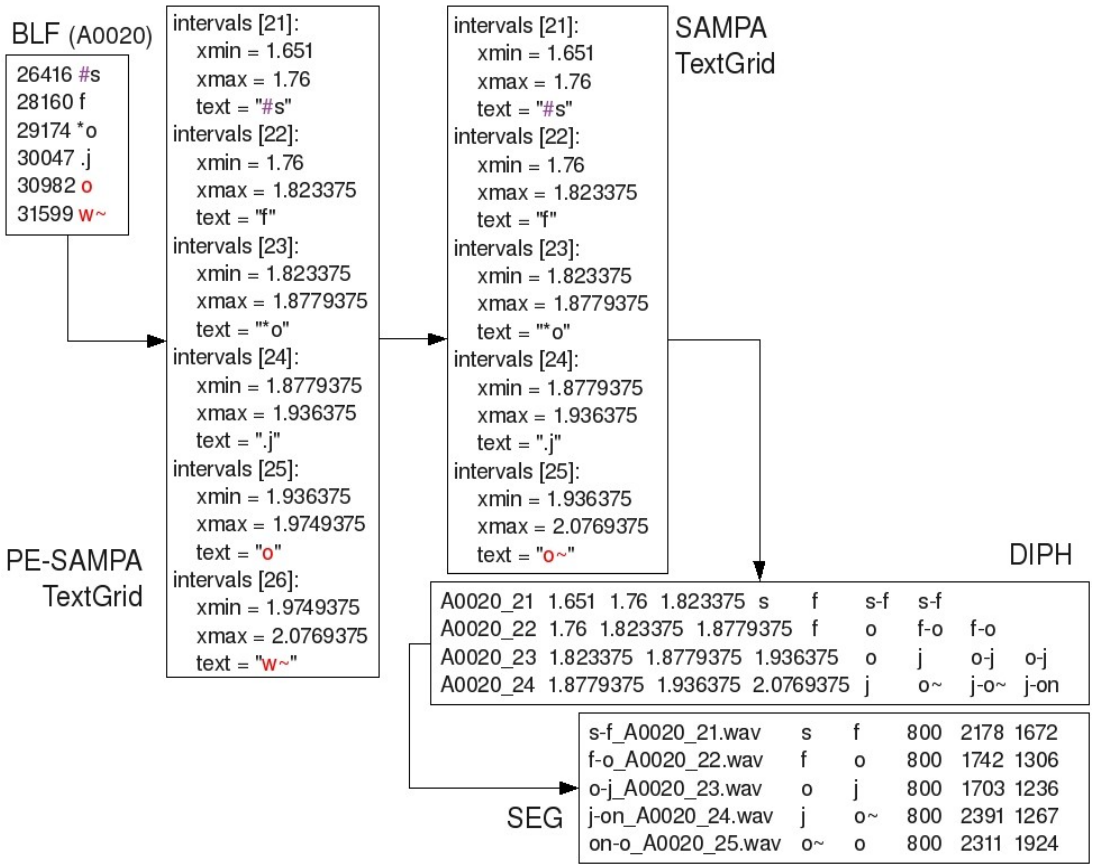


Figure 40: Conversion flow of text files in the automatic diphone extraction system

6.9.5 Automatic diphone extraction system implementation

The automatic diphone extraction system was implemented in Python on the Linux platform, with support of the SOX software for processing of the audio files. The consecutive steps of the diphone extraction are described below.

6.9.6 BLF to TextGrid conversion

The BOSS label file (BLF) format is a format used for annotating corpus for Bonn Open Speech Synthesis (BOSS) system. It is not commonly used annotation format and its main drawback is that it does not contain information about the length of the annotated recording, because only the beginnings of the segments are marked. This means that the beginning of the last segment is known, but it is not known when it finishes. These two reasons of the BLF format specificity pushed toward the conversion of this format to more commonly used annotation format. The format being chosen was TextGrid file which is

used by the Praat software and allows for clear annotation of the speech signal on many tiers. For each interval tier information about the beginning and the end interval is given, together with the text/label assigned to this interval. The format of an interval of the TextGrid file is explained in Table 34:

Table 34: The format of an interval in TextGrid file

TextGrid	***Explanation***	***Example 1***	***Example 2***
intervals	place of the interval	intervals [1]:	intervals [2]:
xmin	start of the interval	xmin = 0.0	xmin = 0.0790625
xmax	end of the interval	xmax = 0.0790625	xmax = 0.17
text	text/label	text = "#$p"	text = "#n'"

For the BLF to TextGrid format conversion a program called *BLF2TextGrid.py* was created. The program reads in all the WAV files and corresponding BLF files in a directory and converts the BLF files to the TextGrid file format creating TextGrid annotation files which contain almost identical information as in the BLF file. The three columns of the BLF file are converted in the following way:

Column 1: Sample numbers from the first column of the BLF files are converted to seconds, as this is the time measurement required by the TextGrid format.

1. The length of the whole recording is extracted from the original recording WAV file.
2. Based on the overall length and the annotation, the duration of the last segment is calculated.

Column 2: The phone labels with their accent markers and word and syllable boundaries are copied to the TextGrid files untouched.

Column 3: The prosodic phrase boundary labels are neglected and are not inserted into the TextGrid files.

6.9.7 PE-SAMPA TextGrid to SAMPA TextGrid conversion

For the Polish language one MBROLA voice already existed. This MBROLA voice, called pl1, was a female voice with a set of diphones corresponding to the set of phonemes of the older version of Polish SAMPA. Since the female MBROLA voice has been available for a few years, the assumption was that some software using that MBROLA voice already

exists.[4] To allow those software use the new male voice without any phoneme label conversion, the automatic diphone extraction software contains a PE-SAMPA TextGrid to SAMPA TextGrid conversion program called *PE-TextGrid2PL1-TextGrid.py*. This program reads in all the TextGrid files in a directory and on the "phones" tier maps the Polish extended SAMPA phoneme labels onto the older version of the SAMPA phoneme labels. The differences in the PE-SAMPA and the older SAMPA (pl1 voice) phoneme set was shown in Table 32 above. Table 35 presents the mapping table of PE-SAMPA set onto SAMPA set.

Table 35: The mapping table of PE-SAMPA set onto SAMPA set

PE-SAMPA	***SAMPA***
^	
y	I
@	e
c	k
J	g
o w~ (sequence)	o~
o j~ (sequence)	o~
e w~ (sequence)	e~
e j~ (sequence)	e~
a w~ (sequence)	a n

The following changes are done:

1. [^] is removed to map phoneme labels such as [d^z] and [t^S] onto [dz] and [tS] respectively;
2. [y] is straightforwardly converted to [I];
3. [@], so called *schwa*, is converted to [e], as [e] is the closest phone to schwa sound in the older SAMPA set;
4. [c] and [J] are converted to [k] and [g]. However, the phones [c] and [J] should be reconstructed by the sequences of phones [kj] or [ki] for the phone [c] and by [gj] or [gi] for the phone [J]. The problem illustrates Table 36. The asterisk "*" at the brackets means that the phones in the brackets may or may not appear after the phones [c] and [J].

4 For example, the author of this work created software called Automatic Close Copy Speech (ACCS) synthesiser.

Table 36: The phones [c] and [J] from the BLF SAMPA annotation convention and their equivalents in the PL1 diphone database.

BLF SAMPA annotation labels		***PL1 SAMPA symbols***	
c	(j/i)*	k	j/i
J	(j/i)*	g	j/i

For the available annotation files, the occurrence of [c] and [J] was transcribed as [c] followed by [j] or [i] and [J] followed by [j] or [i], creating sequences of phones [cj] and [Jj]. Having these sequences of phones, it was easy to replace the phone [c] by the phone [k] and the phone [J] by the phone [g] without any additional sequential split. If the [c] or [J] were not followed by [j] or [i], then there would have to be introduced a process of splitting the phones [c] and [J] into sequences of [kj] or [ki] and [gj] or [gi], respectively. Then the duration of one phone would have to be split and one part of the value of the duration given to the phone [k] or [g] and the other part of the duration given to the phone [j] or [i].

To illustrate the problem, different transcriptions of the word “kiedy” is presented in Table 37:

Table 37: Different transcriptions of the word “kiedy”

BLF notation	***BLF notation with the sequential split procedure needed***	***Older SAMPA notation***
c	c	k
j		j
e	e	e
d	d	d
y	y	I

5. sequence of [o w~] and [o j~] or [e w~] and [e j~] are converted to one label [o~] or [e~] respectively. This step is complicated and requires:

   1. concatenation of special annotation markers attached to either [o] or [e] and [w~] or [j~], for example, [“e] followed by [$jw~] is converted to [“$je~];

   2. assignment of duration of the new concatenated segment starting at the beginning of the first segment, i.e. [o] or [e] and finishing at the end of [w~] or [j~];

3. reduction of the elements of the tier in the TextGrid on the phone level. Each concatenation of the [o w~], [o j~], [e w~] or [e j~] reduces the number of intervals by 1.

6. sequence of [a w~] as in [Saw~sa] is converted straightforwardly to the [a n] sequence as [n] is the most appropriate phone at this point.

6.9.8 Find all diphones in TextGrid files

Before the diphones are extracted, the TextGrid annotation files are processed and consequent phones are put together to find the diphones, or more precisely, pairs of phones, in the annotations. For this step a program called *FindDiphonesInTextGrid.py* was written. The program reads in all the TextGrid files in a directory and for each TextGrid file creates a DIPH file. The format of the DIPH file and three lines from a DIPH file as an example are presented in Table 38.

Table 38: The DIPH file format with exemplar three lines from a DIPH file

Diphone ID	***1st phone start time***	***boundary time***	***2nd phone end time***	***1st phone label***	***2nd phone label***	***diphone label***	***normalised diphone label***
source WAV file + place of occurrence	millisec	milliseconds, place of the boundary between the phones	millisec	only phone label, without accent or word and syllable boundary markers	only phone label, without accent or word and syllable boundary markers	phone labels separated by a hyphen	diphone label without special characters or capital letters used in the further step for the filename
A0009_1	0.0	0.0790625	0.17	_	n'	_-n'	SIL-ni
A0009_2	0.0790625	0.17	0.24	n'	e	n'-e	ni-e
A0009_3	0.17	0.24	0.28	e	j	e-j	e-j

The information in the DIPH file may seem a bit redundant to what is in the TextGrid annotation file, because the start time, end time and the boundary time between the phones is repeated. However, at this step the diphone ID is given which will be later used, for example for the diphone concatenation process. Additionally, the clean phone labels are inserted, without any accent and word or syllable boundary markers. Moreover, the diphone labels are created and those diphone labels are mapped onto the normalised diphone labels which will be used for the diphone filenames. The normalisation table for the diphone filenames is presented in Table 39. The conversion of underscore [_] into [SIL] is performed, because the SOX software does not create files with an underscore at the

beginning of a filename. The conversion of the capital letter into a duplication of its small counterpart is done, because the Windows system does not differentiate between capital and small letters.

Table 39: Diphone label normalisation table

Diphone label	***Normalisation***	***Type of marker***	***Example***
~	n	nasalisation	w~ => wn
'	i	palatalisation	n' => ni
^		conjunction	t^s => ts
?	q	question mark	? => q
capital letter	double small letter	phone label	S => ss
@	ea	schwa	@ => ea
_	SIL	pause	_ => SIL

6.9.9 Diphone extraction

After the diphones are found and DIPH files for all the TextGrid annotation files are generated, the next step is to extract diphones from the recordings and create a diphone database file in the SEG format. The diphones cannot be cut out exactly on the diphone boundary, i.e. the middle of the phone, but some context needs to be left for the MBE analysis. The restrictions of the Mbrolator software were listed in section 6.6 on The Mbrolator software. Comparing with those restrictions, the present automatic diphone extraction program:

1. operates correctly only on files with the 16kHz sampling rate;
2. leaves a context of 800 samples, i.e. 50ms on the right and on the left of the diphone;
3. calculates the overall length of the diphone file, therefore the limit of 10,000 samples is checked. The program does not create diphone files which are longer than the given limit. If one of the diphone is a pause, then this pause is automatically shortened an a correct diphone is created.

The last step of diphone extraction does three things:

1. reads in all the DIPH files in a directory and based on the information in the DIPH files calculates the times of diphone extraction;

2. extracts diphones with a context of 50ms on the right and on the left and writes them to new WAV files whose filenames have the following structure: diphonename_diphoneID.wav, for example *ni-e_A0009_2.wav*. Such a structure of a filename allows for easy alphabetic grouping of the files containing the same diphone in a directory and the diphone ID allows for diphone concatenation in the evaluation step of the automatic diphone extraction process.
3. lists in a separate text file, called *notcutoutdiphones.txt* diphones whose context is less than 50ms or overall length exeeded 10,000 samples.
4. creates diphone database file in the SEG format, called *pl2.seg*.

The SEG file format with three lines taken from a SEG file as examples is presented in Table 40. The text file which lists the diphones which were not cut out has a similar format to that from the SEG file, but additionally contains the category of the error. Three error types are traced by the system:

error 1: the context on the left of the diphone is shorter than 50ms

error 2: the context on the right of the diphone is shorter than 50ms

error 3: the length of the diphone file is longer than 10,000 samples

An example of a line from the *notcutoutdiphones.txt* file is:

```
error 1: SIL-ni_A0009_1.wav _ n' 800 2159 1432
```

Table 40: The SEG file format with three examplar lines from the SEG file.

Diphone filename	***1st half-phone label***	***2nd half-phone label***	***Diphone start time***	***Diphone end time***	***Diphone boundary time***
diphone name + diphone ID + WAV extension	PE-SAMPA or SAMPA	PE-SAMPA or SAMPA	samples (16kHz)	samples (16kHz)	samples (16kHz)
ni-e_A0009_2.wav	n'	e	800	2087	1527
e-j_A0009_3.wav	e	j	800	1679	1359
j-e_A0009_4.wav	j	e	800	1520	1120

The present diphone extraction software gives two options for diphone extraction:

1. extraction of all the diphones from each recording – program called *CutOutDiphones.py* ;
2. extraction of only one instance of each diphone from the whole database – program called *FindIndividualDiphones.py*.

Both programs read in all the DIPH files in a directory and based on the information about the diphones, the SEG file is created and commands for SOX, a software for audio files processing, are build up. The structure of the SOX command for diphone extraction is as follows:

```
sox wavfilename diphonefilename trim starttime lenghttime
```

The command is structured as follows: `sox` is the sound processing program called, `wavefilename` is the source filename of the recording from which the diphone is to be extracted, `diphonefilename` is the name of the file into which the diphone is to be written; `trim` is the sox parameter for extraction of a segment interval; `starttime` is the point from which the extraction should begin, in seconds; `lengthtime` is the time for which the extraction should last from the start point, in seconds.

6.9.10 Evaluation of the automatically extracted diphones

Process of automatic diphone extraction involves many steps and conversion of annotation files to other formats. Diphone extraction from a big corpus may lead to extracting diphones of not the best quality or, if the calculations were wrong, incorrect segments could be extracted. Therefore in the final steps, the result of annotation files conversion to the SEG file format and the quality of the diphones and diphone extraction are evaluated.

6.9.11 Generate TextGrids for diphones

The conversions of annotation file formats can trigger an error. As a result, the final product of such conversions, i.e. the diphone database SEG file, may contain errors. To check the quality of the SEG file, a program for generating TextGrid files based on the information in the SEG file, called *GenTextGridsForDiphones.py*, was created. The additional function of generating TextGrid annotation files for the diphones is that it allows to manually investigate diphone files with their annotation to check:

1. segmentation:
    1. the placement of the diphone boundaries if they are in a stable place
    2. the placement of the boundary between a half-phones
2. signal quality – to see if there are no disruptions in the signal or the diphone is pronounced correctly

3. annotation – to see if the diphone label corresponds to the speech signal.

Such investigation allows to finely select only the best diphones for the MBROLA diphone database. Figure 41 presents an example of a diphone waveform and its annotation. The left context of the diphone suggests a pause before the [n'] phone. Additionally, the filename of this diphone, i.e. ni-e_A0009_2, confirms that the [n'] phone occurred at the beginning of the utterance. Therefore, the selection of this diphone may not be the best for the MBROLA diphone database, because of the unstable work of the vocal folds at the beginning of an utterance after a pause. For the MBROLA diphone database diphones with monotonous pitch contour are to be chosen.

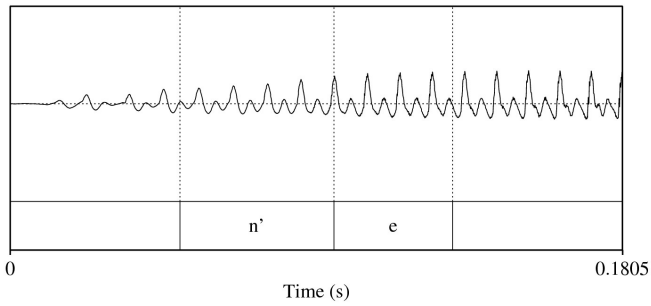


Figure 41: Diphone WAV file with automatically generated annotation

6.9.12 Concatenate diphones

To check the extraction of the diphones, a program called *ConcatenateDiphones.py* was created. The program reads in all the diphone files in the directory and based on the diphone filenames, glues them back together into one utterance. If the extraction is correct, no disruptions or repetitions should be met in the concatenated signal. For the diphone concatenation a program SOX was used. The available SOX commands does not allow to concatenate excerpts of audio files, therefore before the diphones are concatenated, the diphone files prepared for the MBROLA diphone database with the context of 50ms on the right and 50ms on the left are processed. This processing means extracting only the diphones, without the context. The SOX command for this is:

```
sox diphonefilename onlydiphonefile trim 0.05 diphonelength
```

The parameter settings for the sox audio tool are defined as follows:

1. `diphonefilename` – the name of the file from which the pure diphone is to be extracted;
2. `onlydiphonefile` – the name of the file into which the pure diphone without the context is to be written. The filename of the pure diphone is the same as the old diphone WAV filename with an extension '_diphone', for example ni-e_A0009_2_diphone.wav.
3. `trim` – the SOX command for extraction;
4. `0.05` – the point from which the extraction should begin, in seconds. 0.05 is equal the context of 50ms, which means that the value 0.05 omits the first 50ms of the diphone context;
5. `diphonelength` – the time for which the extraction should last from the start point, in seconds, calculated as: diphonelength = diphonefilelength – 0.1s
    1. diphonefilelength – is the length of the file with the diphone in context
    2. 0.1s is equal the summary of the left and right context of the diphone

Such prepared pure diphone files are then concatenated according to the diphone IDs. The program reads in all the diphone files in a directory and orders them according to the diphone ID in the filename. The process of diphone ordering is shown on Figure 42.

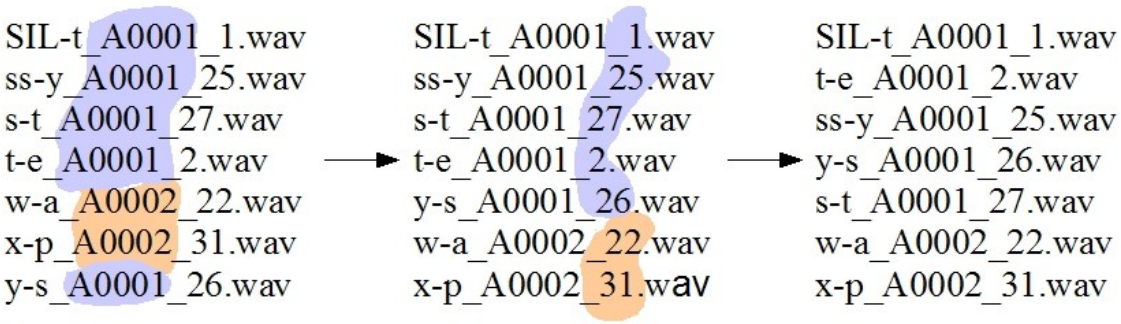


Figure 42: Diphone files ordering according to the diphone ID.

After the diphones files are ordered, the process of concatenation starts. For concatenation two commands are used; the first one is a SOX command, the second one is the Linux command. This sequence of commands work in the program as a loop.:

```
sox concatwavfilename onlydiphonefile new.wav
```

```
mv new.wav concatwavfilename
```

where:

1. `sox` - the program called, i.e. SOX;
2. `concatwavfilename` – the filename of the concatenated file. The filename structure of the concatenated file consists of diphones' source filename with '_concat' extension, for example A0009_concat.wav
3. `onlydiphonefile` – the name of the file with the pure diphone which is to be glued to the concatenated file.
4. `new.wav` – the transitional WAV file into which the concatenation of the concatenated file, i.e. concatwavfilename, and the diphone, i.e. onlydiphonefile, is to be written.
5. `mv` – Linux function which moves new.wav file into the the concatenated file.

The transitional new.wav file was added, because it is not possible to concatenate two files and write the result of the concatenation to one of those files, i.e. it was not possible to perform such an operation:

```
sox concatwavfilename onlydiphonefile concatwavfilename
```

If any of the diphones are not extracted, then the ordering function omits such a diphone. It may happen that the first diphone is not extracted, because the initial pause does not give context of 50ms for the diphone. In such a case the diphone with number 1 in the filename will not be created and concatenation will start from diphone 2 and so on. Such diphone number 1 will be missing in the concatenated file.

Additionally, it should be stated that the concatenation function is intended to be used when all the possible diphones are extracted from each recording – this is done with program *CutOutDiphones.py,* not when only once instance of each diphone is extracted, which is done with program *FindIndividualDiphones.py.*

6.9.13 PL2 synthetic Polish male voice evaluation

Male PL2 MBROLA voice was evaluated in diagnostic tests and speech output assessment tests. Diagnostic evaluation was carried out to see whether the new PL2 MBROLA voice works in both Windows and Linux environments.

Additionally, about 100 sentences were synthesised to see whether the PL2 contains all the diphones. It was found out, that the PL2 voice contains all the common diphones, but it does not contain some rare diphones, especially those which theoretically do not exist in Polish thanks to the voicing and devoicing processes. These diphones were not included while preparing the material for creation of diphone database for the PL2 voice. However, these diphones might be replaced by either voiced or voiceless diphones, therefore all the tested material, i.e. 100 sentences were synthesised successfully in both Windows and Linux environments.

The synthesised sentences came from different sources. Some of them were synthesised using the ACCS synthesis (Bachan 2007a) and existed in the corpus created for the PL2 voice creation, other sentences were coming from outside sources or were created for the evaluation.

The next step was to carry out two speech perception tests in order to assess the speech output quality. The first test aimed at speech recognition and the second assessed the overall quality of the synthetic speech. Both tests were designed not only to evaluate the PL2 voice, but also to compare it with the PL1 female voice (Szklanny and Marasek 2002). In 2007 the author of the present work carried out speech quality test of the ACCS synthesis within her work (Bachan 2007b). At that time the PL1 female voice was used for the ACCS synthesis. In the present work, the same sentences taken from the BOSS corpus (Demenko et al. 2007, Demenko el al. 2008) were ACCS synthesised using the PL2 male voice, with the pitch values coming from the original male speaker. Test 1 presented here is based on the ACCS evaluation coming from (Bachan 2007b). Test 2 contains ACCS synthesised sentences with the PL2 male voice and the PL1 female voice. For the sentences with the PL1 female voice, the original male pitch values coming from the original speaker were multiplied by 2 to imitate female pitch values. Additionally, the original recording from the BOSS corpus were added to Test 2. Test 1 and Test 2 are presented below. The test materials are to be found in the appendices.

Test 1: word recognition – functional testing of intelligibility of speech

Method:	Semantically predictable and unpredictable synthesised sentences from the corpus (Base A and Base B of the BOSS corpus, Demenko et al.****) are presented to the subjects. The subjects write down what they hear in the answer sheet. The set of unpredictable sentences (meaningless sentences) is used to eliminate the influence of the topdown processing (Clark & Yallop 1995: 312, Ryalls 1996: 94).
Material:	10 semantically predictable and 10 semantically unpredictable synthesised sentences with the PL2 male voice.
Instructions:	Listen to the recording and write down what you hear.

Test 2: subjective sentence quality test – judgement testing of speech quality

Method:	The subjects are asked to evaluate the quality of isolated long (multiple) sentences from the corpus (Base E of the BOSS corpus, Demenko et al. ****) at 5-point rating scale from 1 to 5, where 1 is the lowest grade and 5 is the highest grade.
Material:	5 different compound sentences have been chosen from the Base E of the BOSS corpus and prepared in the following fashion: 1. 5 compound sentences were ACCS synthesised using the PL1 female voice, with male pitch values multilied by 2. 2. 5 compound sentences were ACCS synthesised using the PL2 male voice, with pitch values coming from the original male speaker. 3. 5 sentences were taken from the BOSS corpus and were not synthesised, but these were the original recordings of the male professional speaker. Altogether, 15 stimuli were used in the test, but only 5 different sentences. All the sentences are played in a random order.
Instructions:	Listen to the recording and evaluate the quality of speech at the 5-point rating scale, where 5 is the highest grade.

The two speech quality tests were administered to 20 Polish subjects, 10 females and 10 males, at the age between 18 and 25 years old (average age: 21 years). The test materials were to be downloaded from a server and the subjects were taking the test on their own, in their homes or in a classroom. The general instruction to the subjects was to listen to the audio files via the headphones or, if they are not available, via the loudspeakers in a quiet room. Additionally, the subjects were asked to listen to the audio files once or twice, if necessary 3 times, but not more. Each testing session lasted about 20min.

Speech perception test results are presented in Table 41 and Table 42 and include results of PL2 voice in comparison with PL1 voice.

Test 1: word recognition test results are presented in Table 41. The evaluation of word recognition of PL2 was carried out within this Ph.D. thesis work, whereas results of PL1 voice come from the author's M.A. thesis and were carried out in 2007 on a different population (Bachan 2007b). For ACCS evaluation using the PL1 voice 19 people (11 females and 8 males) at the age between 8 and 55 years took part (average age: 24 years).

The test results show that the average recognition of words synthesised with PL2 is high (93.85%) and the test stimuli, i.e. words in semantically predictable and unpredictable sentences affect perception. Recognition of words in semantically unpredictable (meaningless) sentences is worse by 10.58% than in the semantically predictable (meaningful) sentences. Overall, females' recognition was higher.

Compared with the speech perception tests in which the PL1 voice was used (Bachan 2007b), recognition scores for the PL2 is higher, especially when it comes to the semantically unpredictable sentences. However, the population who took part in the test in 2011 and 2007 is different, so the conclusion cannot be drawn that the PL2 voice is of higher quality. However, it is very encouraging that the results are slightly better.

Table 41: Results for Test 1 – average correctly recognised words in predictable and unpredictable sentences. N – number of words

	PL2			***PL1 (Bachan 2007b)***		
	Predictable (%)	***Unpredictable (%)***	***All (%)***	***Predictable (%)***	***Unpredictable (%)***	***All (%)***
N	75	51	126	75	51	126
Female	98.40	90.20	95.08	*not available*	*not available*	91.92
Male	97.87	84.90	92.62	*not available*	*not available*	88.10
Overall	98.13	87.55	93.85	96.28	81.53	90.01

Table 42 shows the results for male and female subjects for Test 2: subjective sentence quality. In the test tree different voices were evaluated: PL1 and PL2 synthetic voices and the original (natural) male voice. The original voice received the highest scores. These are the recordings of the professional speaker recorded in the radio studio, so the high scores were expected. When it comes to the assessment of PL1 and PL2, both voices received very similar scores just a bit under 3 overall. However, the rating of the PL2 voice was slightly higher which indicates that the synthesised sentences with the PL2 male voice sounded a bit better. Females and males assessed the thee voices similarly and did not show any sex preferences – males gave lower grades to all thee voices, and females were less critical and their grades were higher.

To comparison it can be added that in the speech output assessment test in 2007 (Bachan 2007b), the synthetic speech generated with the PL1 female voice received on average 2.73 score (STDV – 0.89, Min:Max – 1:4), whereas the original professional

speaker recordings were assessed on average with 4,69 score (STDV – 0.51, Min:Max – 3:5). The MOS for PL1 in the tests from 2007 and 2011 differ only by 0.06 point. This may support the view that the speech quality of synthetic speech generated with the PL2 voice is better, despite the population among which the test is carried out.

Table 42: Test results for Test 2. MOS/5 – Mean Opinion Score out of 5, STDV – standard deviation, Max:Min scores given by subjects

	PL1			***PL2***			***Original***		
	MOS/5	***STDV***	***Min:Max***	***MOS/5***	***STDV***	***Min:Max***	***MOS/5***	***STDV***	***Min:Max***
Female	2.86	1.07	1:5	2.98	0.94	1:4	4.98	0.14	4:5
Male	2.72	0.81	1:4	2.86	0.95	1:4	4.96	0.20	4:5
Overall	2.79	0.95	1:5	2.92	0.94	1:4	4.97	0.17	4:5

PL2 evaluation showed that the newly created synthetic voice is of good quality and the methods developed within this work are effective and may be used in the future MBROLA voice creations, as well as for other purposes where the diphone databases are used. The phonetically rich sentence extractor as well as the automatic diphone extractor software saved a lot of time and the time consuming manual work carried by the phonetician was automatised. However, the final selection of the diphones was carried out manually. Maybe thanks to this manual selection of the best diphones the PL2 voice scored better in the speech perception tests than the PL1 voice. However, this question cannot be answered having carried out those two speech perception tests.

Having only just 200 sentences to be recorded and annotated, it should be efficient to create an MBROLA voice for any language. This should enhance the experimental work with speech synthesis in different environments, in small student groups in linguistic, phonetic, psycholinguistic and technical studies in technologically less well equipped countries.

6.10 Summary

In this chapter several text and speech corpora were investigated for diphone extraction with insufficient results for a full diphone database creation. Because only a female voice PL1 for the MBROLA speech synthesis system was available, the aim was to develop a male PL2 voice. The male PL2 voice is going to be used for speech synthesis in the demonstration dialogue system.

Chapter 7: Dialogue corpus for demonstration prototype

7.1 Chapter overview

The present chapter reports on steps leading to a dialogue corpus of recordings, its annotation and analysis. The analysis of the corpus is focused on the alignment phenomenon between the speakers, and the observations are used to model human-computer communication, paying attention to the stress situations. A set of conversation scenarios is created in order to observe and test alignment in different communication situations.

Previous chapters reported on theories, methodologies and tools which were needed to test the thesis underlying this research, as presented in the first chapter. In the following chapters, the thesis is tested and new solutions for dialogue corpus recordings are presented. It will be shown that finite state automata may not only be used for dialogue control, but also for map traversal.

7.2 Corpus design

For the present research, two types of dialogues were recorded in laboratory conditions: a map task dialogue and a picture description dialogue, so-called diapix task (Bradlow et al. 2007; Lewandowski 2009; Baker & Hazan 2009). The map task and diapix tasks are source of semi-spontaenous speech. Both dialogues are directed at crisis situations and communication in public setting, especially between people who do not know each other. The aim was to record 12 dialogues of each kind and 3 dialogues performed by a control group (on description of the control group see below 7.2.2 Subjects). Additionally, a neutral reading task was added in order to obtain the same speech material from all the subjects.

7.2.1 Prompt speech material and the recording scenarios

Prompt speech material was intended to invoke stress. The idea was to simulate a conversation which could happen in a crisis situation. Additionally, neutral prompt speech material is added to create a corpus of control dialogues.

1. Map task:
    1. emergency scenario – each of the subjects gets a street map. In this task one person has to lead the other person to get to a place in which a man with a heart attack is waiting for help. The person who is chosen to lead the other person gets a map with a marked route on it which leads among different landmarks. The other person gets a map with the landmarks only. The interlocutors cannot see each other. The task is to describe a route of how an ambulance should get to the emergency location. At the beginning one of the subjects gets a description of the situation underlining the tragic situation to invoke emotions such as fear or sadness. Additionally, the two maps the interlocutors get slightly differ in respect of the landmarks to create trouble in communication. Moreover, on the route of the ambulance there are such obstacles as an accident, a traffic jam, road works, school race. These are not seen by the other person. To boost the stress degree, the leading subject gets a limited time of 5 minutes to perform the task (cf. Johnstone & Scherer 1999).
    2. neutral scenario – a route description of how to get to a cinema. In this task one person has to lead the other person to the cinema along the streets. The maps are identical, however, only on the following person's map the starting point is mark, and on the leading person's map the final point (the cinema) is marked. There is no time limit set to the task.
2. Diapix task – description of a picture in order to look for differences: in this task there is no leading person. Both interlocutors have the same status. Their task is to describe the picture and find the differences between them. The interlocutors cannot see each other.

1. emergency scenario – the picture presents an accident site. The subjects get a time limit of 5 minutes to finish the task in order to raise the stress level. The same countdown clock as for the map task with the emergency scenario is used.
2. neutral situation – the picture of a shopping area of a town.

3. Reading – the task is to read a text in a neutral style. The subjects are asked to read the text slowly and without any poetic interpretation.

Although the tasks are divided into two categories: map task and diapix, and these task are divided into emergency and neutral scenarios, their actual design is different to try to reduce the subjects' familiarisation with the task. Obviously, the longer the subjects will cooperate, the better their strategies will be. Also, the stress level originating from the fact of taking part in the recordings will probably be fading out through out the recording session. Therefore, it was decided to carry out the emergency scenarios first. And then to proceed to the neutral tasks. Reading was to be recorded at the end of the recording session.

7.2.2 Subjects

Subjects who do not know each other were selected, in order to assure that the dialogue would have a public character (Batliner et al. 2008). If possible it was intended to choose pairs of interlocutors with a big age gap between them and with different academic status. It was intended to confront young people (students) with older people (academic teachers). This confrontation could also affect the young people's stress level.

Additionally, pairs of people who know each other, possible good friends, are planned to be recorded as a control group.

For the project, 15 males and 15 females were chosen and recorded in pairs: male – male, male – female, female – female.

For the task people who do not have experience with emergency task were chosen, because it was assumed that the emergency call centre operator's expertise was not needed for the task. One of the scenarios could be to ask the professional emergency call centre operators to participate in the recordings, but their acquired behaviours from work do not need to help in the alignment experiment. They could sound commanding from the very

beginning of the conversation and do not try to align with the speaker neither phonetically (speech expressiveness) nor linguistically (learned phrases and formulae). They carry out cooperative non-alignment, as described in Chapter 1. It is not claimed that the call centre operators do not align with the interlocutors, as this is unconditional behaviour, but it is expected that their work experience made them better control their behaviours. Additionally, their authoritative behaviour may disturb the equality of subjects expected in the diapix task. Last but not least, it might be difficult to find the real emergency call centre operators who would like to participate in the recordings. Taking these issues into account, it was decided to ask inexperienced people to participate in the task. It was expected that the corpus would contain more natural dialogues, the emotions evoked in both interlocutors would be real and their alignment would be uncontrolled. Nevertheless, investigating the alignment between the real emergency call centre operators and the callers sound as a very interesting task for the future. But in this scenario, the best would be to acquire the real data from the call centres and not set up laboratory tests. All the phone calls at the call centres are recorded and stored in databases, the only problem is that they contain sensitive data and therefore are hard to obtain.

Dialogue corpora with spontaneous and semi-spontaneous speech like the one described by Karpiński (2002) do not meet the requirements of the present study because of the special kind scenario which is involved. Developing a dialogue system prototype limits the range of topics and tasks the dialogue system may deal with. Also, the available corpora for Polish does not take into account the public and emergency setting of communication.

7.2.3 Recordings

The recordings were performed in two adjacent offices at the university. At the beginning, ot was planned to use head microphones, one for each of the speakers. The head microphones would assure freedom of movements and would not block subjects' view on the pictures or the maps provided. Moreover, a table microphone was to be used to provide a back-up recording. However, this idea was abandoned and a more handy approach was taken to use build-in laptop microphones as this was the most suitable choice for the

recording software described later in this section and reduced the required hardware to minimum

It was not clear which method of communication to choose, because the idea was that the subjects did not see each other, but could hear each other, so some isolation had to be assured, but also the audio communication channel had to be provided. The choice was made between the telephone and a Skype call. On one hand, the telephone conversation seemed more reliable, but holding a handset would limit subjects' movements. Additionally, the speech signal from the handset could get into the microphone which was unwanted. However, the telephone conversation simulates the situation of telephone conversation with the call centre operator.

The Skype call, on the other hand, might not be so reliable. The connection may hang up if the Internet connection is not fast enough. But the pro for of using skype was that it is allows to use the headphones which may reduce or eliminate the signal coming from the other interlocutor normally heard via the telephone handset. Usually, the speech signal coming from the handset is so loud that the person standing next to the person talking on the phone can hear what is being said by the person on the other side on the telephone. This speech might reach easily the microphone and be recorded. This was to be avoided in the present corpus creation as the aim was to get clear speech signal from each of the interlocutors separately.

Table 43 lists pros and cons using either the telephone or the skype call for communication between the interlocutors in the recording session.

Table 43: Pros & cons using either the telephone or the skype call for communication between interlocutors

Telephone	***+/-***	***Skype call***	***+/-***
Free at university	+	Free	+
Reliable	+	Hangs up	-
Signal coming out the handset	-	Headphones reduce/eliminate signal coming from the other interlocutor	+
Does not emit noise (no fan)	+	Additional computers/laptops emit additional noise	-
Holding a handset limits subject's movements who will have to hold a pen for writing in their second hand	-	Sets hands free if used with a free-standing microphone and headphones	+
A handset may be source of noise if it is knocked at the overhead microphone frame or even at the microphone itself if the subject is not careful	-		
Simulates holding a handset when calling a call centre	+		

7.3 Implementation

7.3.1 Creation of maps

For the two map tasks, four maps were created according to their design in the OpenOffice.org Draw program.

In the *emergency scenario*, the map shows a hospital and an ambulance ready to set out for the patient. On the leading person's map, there is the route marked and a few obstacles which prevent the ambulance from taking the shortest route. On the following person's man there are neither marked route nor obstacles on the way which could complicate the task. Additionally, both maps differ in the positions of the landmarks to confuse the interlocutors. The landmarks are such typical buildings as a cinema, school or a shop. There are also trees, a lake and a car park. The maps for the scenario are shown in Appendix M.

For the *neutral scenario*, two identical maps were created, which differ only in the initial and termination points. For the following person, the initial point is marked. For the leading person, the termination point is marked. The first step of the conversation is then to arrange where to start from. There is no route marked on the map and it depends on the interlocutors which route to take. Although, one of the persons is to be the leader, it does not eliminate the fact that the follower can suggest their own ideas of which way to take.

The street arrangement is fairly varied in order to encourage interlocutors to make decisions whether to turn left or right, whether to take street A or B. There are no landmarks on the map, only street names. Most of the names were taken from the map of Poznań. But there are also some made up names.

The maps which differ only in the initial and the termination points marked on separate maps are presented in Appendix N.

7.3.2 Creation of diapixes

For the diapix task, two sets of pictures were used. The emergency pictures are actual photos arranged for the task. The neutral pictures are adopted from the diapix task created for recording the Wildcat Corpus of Native- and Foreign-Accented English (Bradlow et al. 2007).

On the emergency scenario diapix, there is a boy being injured sledging on a hill. The pictures were taken using standard Canon digital camera and made a bit brighter using the standard OpenOffice.org Picture processing tools: brightness and contrast. Both pictures differ in 10 details: 5 changed items and 5 missing items. The differences between the photos are listed in Table 44. The photos are presented on Figure 43.

Table 44: Difference between diapixes from the emergency scenario

Changed Items		***Missing Items***	
Version A	***Version B***	***Version A***	***Version B***
boy lying nearby sledge	boy lying on sledge	glasses	no glasses
boy keeps legs apart	boy keeps legs together	no girl on sledge	girl on sledge
blue sledge	green sledge	no red car	red car
snow green spade	red broom	no bike	bike
rockers on fence	rucksack on fence	no light	light in the window
	rockers on bike		

Figure 43: Diapixes from the emergency scenario

The diapixes for the neutral scenario present a shopping area a town. They were processed in OpenOffice.org Draw slightly in order to change English names for the Polish names. The pictures differ in 10 details. The differences are presented in Table 45. The diapixes are shown on Figure 44.

Table 45: Difference between diapixes from the neutral scenario

Changed Items		***Missing Items***	
Version A	***Version B***	***Version A***	***Version B***
cat on pet shop sign	sheep on pet shop sign	no beehive	beehive
'Schab' sign	'Kotlety mielone' sign	paw prints on door	no paw prints on door
'Barszcz' sign	'Rosół' sign	'Dyskoteka'	no sign
woman wears red shoes	woman has green shoes	just 'Sklep ZOO'	'Sklep ZOOlogiczny'
		no bench	bench
		boy carrying box	boy not carrying box

Figure 44: Diapixes for the neutral scenario (adopted from Bradlow et al. 2007)

7.3.3 Reading task

The excerpt for the reading task was taken from the first page of "One Hundred Years of Solitude" (1967) by Gabriel García Márquez. It contains a site description and does not require any expressive poetic interpretation. The text, although has quite complex sentences, has quite simple vocabulary which should not be problematic for subjects. The excerpt being chosen for the reading task is:

Macondo było wówczas niewielką osadą – dwadzieścia chat z trzciny oblepionej gliną, zbudowanych na brzegu rzeki, której przezroczyste wody bystro toczyły się po gładkich białych kamieniach koryta, wielkich jak jaja przedhistorycznych ptaków. Świat był jeszcze tak młody, że wiele rzeczy nie miało nazwy i mówiąc o nich, trzeba było wskazywać palcem. Co roku w marcu rodzina Cyganów rozbijała namiot niedaleko wioski i pośród zgiełku piszczałek i bębnów prezentowała mieszkańcom nowe wynalazki. Najpierw przywieźli magnes.

7.3.4 Instruction to the subjects

The subjects are asked to perform the following tasks:

1. Map task: emergency scenario

Leading person:	Imagine that your close relative have just had a heart attack. You are calling an emergency call centre asking for help. You need to explain to the operator how to get to the suffering person. On the map you see the route which the ambulance needs to take. On the route there are marked different events which make it impossible for the ambulance to take the shortest way. Your task is to lead the ambulance along the marked route. You have 5 minutes to finish the task.
Following person:	Imagine that you are an emergency call centre operator who needs to send an ambulance to a sufferer. In a moment you will get a phone call from a person asking for help. She/He will explain to you the way the ambulance will have to take to get to the spot. Your task is to follow the route and mark on the map with a marker pen the way you are following.

2. Map task: neutral scenario

Leading person:	Imagine that you are working in a cinema. A person is calling you asking how to get to the spot. Your task is to help the person to get from the place from which he/she is calling to the cinema along the marked streets. While explaining the route, mark on the map with a marker pen the way.
Following person:	You are going to a cinema. However, you do not know the way. You are calling to the cinema asking how to get to the spot. Your task is to follow the route and mark on the map with a marker pen the way you are following.

3. Diapix: emergency scenario

   In a moment, you will see a picture presenting an accident. A young boy got injured and needs medical help. Your task is to cooperate with your partner and find 10 differences between yours and your partner's pictures and mark them on the picture with a marker pen. You have 5 minutes to finish the task. (Only you see the countdown clock).

4. Diapix: neutral scenario

   In a moment, you will see a picture presenting a shopping area in a town. Your task is to cooperate with your partner and find 10 differences between yours and your partner's pictures and mark them on the picture with a marker pen.

5. Reading:

   Read slowly and clearly the following text

7.3.5 Recording scenario

The recordings were performed in two quiet university offices. The subjects sat alone in each of the rooms. In each of the rooms there was a laptop used for the recordings and the communication between the interlocutors. The interlocutors communicate via Skype. The recording setting is presented in Figure 45.

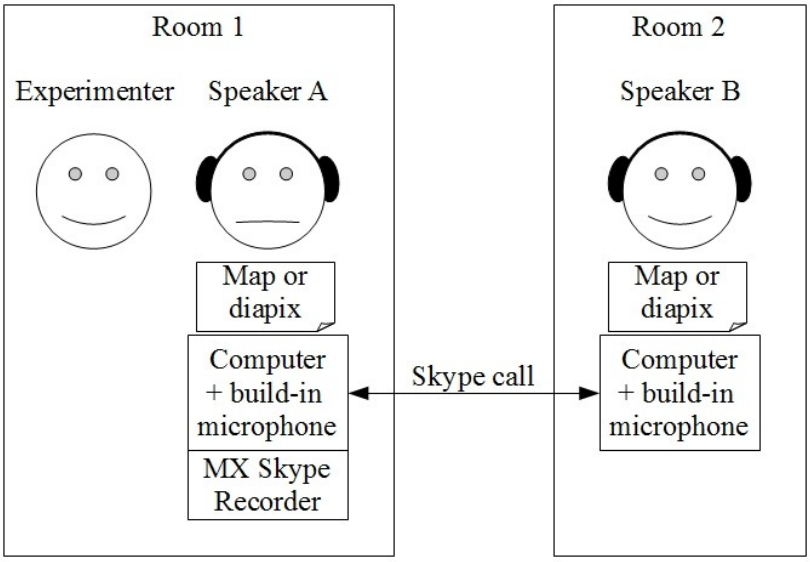


Figure 45: Recording setting of the dialogue corpus

For the recordings, very quiet laptops were used whose fans do not produce loud stationary noise: DELL Latitude E4310 and ASUS EeePC 1000H. Both laptops were connected to the Internet. To each of the laptops headphones were attached to make it possible for the interlocutors to hear each other, but to exclude the speech signal from one of the channels. i.e. if the headphones were not used, the speech signal would be recorded on two channels – once on the speaker's input, and then coming from the loudspeakers it would be recorded on the other channel at the receiver's laptop.

For the recordings laptop's built-in microphones were used. They were tested and the sound quality of the recordings was good.

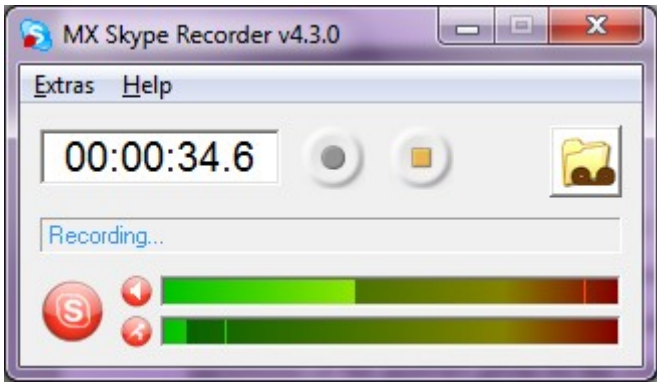


Figure 46: MX Skype Recorder window

The recordings were performed by the MX Skype Recorder software on one of the laptops (the laptop of the leading person, the caller, i.e. the student). The MX Skype Recorder is a shareware. A standard license for the program was purchased to carry out the recordings within the present work. The MX Skype Recorder allows to record unlimited time audio skype calls on two separate channels for two speakers in the stereo WAV format. It is also possible to record more speakers. The general functionality is that MX Skype Recorder records conversations either in "single" or "dual" audio track – one channel for input and the other for output. On Figure 46 the MX Skype Recorder window is presented. MX Skype Recorder window shows the time of the recording, the Record and Stop buttons. There is also a button linked to the folder in which the recordings are saved. Additionally, there are three red indicators showing (1) Skype is active, (2) Skype is sending audio data to "audio out" device and (3) Skype is getting audio data from "audio in" device. The are also two level lines which give information about the volume of the audio signal on in- and out-channels.

On the computer of the following person, the younger person, there was a big timer which showed the time left for the tasks. The timer was provided by the TimeLeft desktop utility distributed as a freeware and a shareware (NesterSoft Inc.). For the project, the freeware version was used. The timer image is shown on Figure 47. The red background of the clock was set to underline the seriousness of the task and an effect 'left-right side-slip' for the digit changing was added in order to distract the person being recorded. When the time finished, i.e. was counted down to 0:00, an audio alarm switched on. It was a well-known alarm tune which wakes up millions of people every day for work and people do not always have pleasant associations with it.

Figure 47: TimeLeft timer used for the recording of the emergency scenarios.

Each of the subject got a marker pen to mark the route on the map or the differences between the pictures. The leading subject, i.e. the student received the black pen and the other subject received blue pen. Their maps and diapixes were inserted into a transparent plastic cover. The subjects were asked to draw on these plastic covers. The covers were coded, so that the visual answers given by each of the subjects can be accessed and it is possible to easily compare agreement of the answers given by the subjects within one pair and across the other subjects.

The recording session was carried out under a supervision of the reasearcher being present only in one of the rooms – the room of in which the student was sitting. Before each of the tasks, the supervisor provided the instructions and the maps and the diapixes. The subjects were asked whether the instructions for the task were clear and whether the recording could be started.

7.4 Corpus creation

The corpus was recorded according to the specifications described above. 15 pairs carried out four dialogues based on the given tasks. Additionally, the reading task was performed by each of the subjects. Altogether, 60 dialogues recorded at 48000kHz sampling frequency by MX Skype recorder and 30 reading tasks recorded with Praat at the 44100kHz sampling frequency.

One recording session lasted about 30min. On Figure 48 there is a picture of a person being recorded at the emergency scenario. On the computer screen the big TimeLeft timer can be seen. The dialogue partner is in the next room and is not shown on the picture. The two interlocutors communicate via skype.

Figure 48: A person in the emergency setting at the corpus recording

Altogether, 15 pairs were recorded. 15 women and 15 men were arranged into 5 female-female pairs, 5 female-male pairs and 5 male-male pairs. 12 pairs were composed of people who did not know each other or were in the superior-inferior relation, and 3 control pairs were composed of friends. Into the leader's position, young students were put. Into the follower's position, people with a higher degree or in a superior position at work were put. Although it was planned to record students with academic teachers, this idea was abandoned, and instead mainly Ph.D. students teaching at the university were asked to participate in the experiment. Altogether, 8 Ph.D. students took part in the recordings, one Ph.D. holder, one secretary, one acoustician, one acoustic technician, a taxi driver and a medical technician. The rest were students, either with secondary education or with the B.A. degree. The youngest subjects were 19 years old, and the eldest was 53. The biggest age difference between the subjects was 32 years. More detailed data of the pairs of people who participated in the corpus recording and the dialogues are presented in Table 46. "A" subjects are the leaders' group, i.e. the students. The "B" group is the follower's group. All the data concerning the recordings were not attached to the present work because of the printing page limitation. However, such metadata about the recordings go with the corpus:

1. Date of the recordings

2. Folder name in which the recordings coming from one pair are collected, e.g. 2011-02-15_03_MM. The folder name codes information about the date of the recording (2011-02-15), the pair ID (03) and the pair sex (MM stands for male-male).
3. Subject's ID
4. Subject's sex
5. Subject's name
6. Subject's age
7. Subject's height
8. Subject's weight
9. Subject's education
10. Subjects occupation, e.g. Ph.D. student
11. Notes, e.g. whether the recorded pair are friends
12. Place of the recording, i.e. Collegium Novum or Poznań Supercomputing and Networking Center
13. Length of each recording with 14 decimal places (Praat format)
14. Size of each recording in KB

Additionally, from all dialogue recordings the channels for each speaker were extracted and saved with extensions "_ch1" for speaker B and "_ch2" for speaker A. The split of channels was done in order to perform annotation and extract phonetic information about speech for each of the speakers separately.

Table 46: Data of the corpus recording. Age diff – stands for age difference between the interlocutors counted as B's age – A's age.

Pairs	*ID*	*F/M*	*Age*	*Age diff: B -A*	*Degree*	*Map task : emergency (sec)*	*Diapix: emergency (sec)*	*Map task: neutral (sec)*	*Diapix: neutral (sec)*	*Reading (sec)*
female-female	01A	F	27	-4	secondary	80	291	131	535	49
	01B	F	23		B.A.					38
	04A	F	23	6	B.A.	94	290	72	295	38
	04B	F	29		M.A.					36
	08A	F	27	-2	M.A.	70	310	175	412	37
	08B	F	25		M.A.					34
	09A	F	23	8	secondary	128	228	113	504	41
	09B	F	31		Ph.D.					40
	13A	F	21	7	B.A.	170	306	124	327	39
	13B	F	28		M.A.					33
female-male	02A	M	21	32	secondary	138	59	181	323	33
	02B	F	53		M.A.					35
	05A	F	25	2	M.A.	266	257	241	284	37
	05B	M	27		M.A.					35
	11A	F	22	4	secondary	325	198	163	279	38
	11B	M	26		M.A.					31
	06A	F	23	10	B.A.	72	293	251	307	35
	06B	M	33		M.A.					31
	07A	F	23	3	B.A.	228	303	159	522	36
	07B	M	26		M.A.					38
male-male	03A	M	24	28	B.A.	148	303	185	345	40
	03B	M	52		M.A.					43
	12A	M	19	9	secondary	156	305	196	245	38
	12B	M	28		M.A.					37
	15A	M	22	3	secondary	226	303	167	568	38
	15B	M	25		B.A.					42
	10A	M	30	-1	secondary	170	213	116	295	36
	10B	M	29		B.A.					35
	14A	M	20	-1	secondary	54	298	79	150	32
	14B	M	19		secondary					36
15137sec = 4h 12min 2726298KB = 2.6GB					***All:***	2325	3957	2353	5391	1111
					Min:	54	59	72	150	31
					Max:	325	310	251	568	49
					Average:	155	264	157	359	37

7.5 Corpus annotation

A selected set of dialogues is chosen for annotation. Because the speech material is quite big, annotation of all the dialogues would go beyond the realms of this thesis work. The preliminary annotation was performed by the author on two levels for each of the speaker. The proposed annotation seems to be the most optimal for the basic studies of the dialogue. These levels are:

1. speech – speech vs. non-speech is marked, together with information about speaker noises.
2. special – on this tier speech events, such as filled-pauses or hesitations are marked.

The speech level tier is either called speechA or speechB for either of the speakes. 4 types of makers may apper on this tier:

1. S – for speech signal
2. sil – for silence
3. spk – for speaker's noise (breath or smack)
4. N – for non-human noise (knock or paper murmur)

The special tier is either called specialA or specialB for either of the speakes. On this tier all events bearing information other than textual is marked, but not exactly paralinguistic. There is information about:

1. special words, e.g. "tak" (Eng. yes), "nie" (Eng. no), "mhm"
2. filled-pauses – they are "spelled", meaning that long filled-pauses would be marked as "y" and longer as "yyy".
3. lenghtened phones – a word with lenghtened phone would be spelled "aleee", "mammm" or "iiiglasty"
4. mistakes – a speech signal where a speaker made a slip or cannot utter what he or she has in mind is marked as "mistake"
5. laughter – the part of the signal where the speaker is laughing or it is heard that while speaking the speaker is lauging is marked as "laughter"

6. unfinished – the speech chunks which were not finished but broken off are marked as “unfinished” (Pl. “urwane”)
7. “–” was inserted in the intervals between the intervals bearing the “special” information. This marker was added not to have empty intervals.

The dialogues were first annotated roughly on the *speech* tier for both speakers, in order to follow both interlocutors in the dialogue and analyse the exchange between them. Then the channel for each interlocutor was annotated on the *special* tier and the boundaries on the speech tier was adjusted according to the waveform and spectrogram. After the annotation was performed for each speaker, the annotation tiers, speechA, specialA, speechB and specialB were put again together. The annotation on these four tiers can be seen on Figure 49.

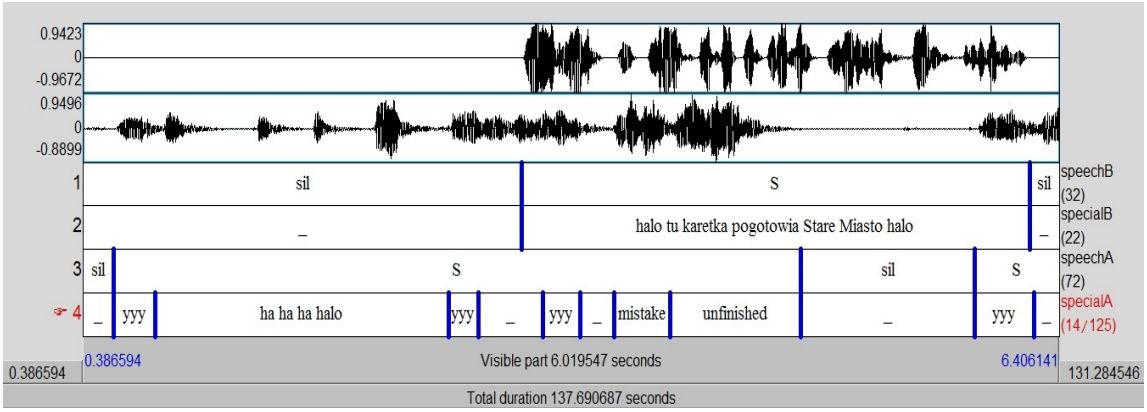


Figure 49: Annotation of dialogues on speech and special tiers for each speaker

One typical emergency map task dialogue (Speakers’ ID: 12A and 12B) was annotated on six tiers:

1. phones – extended SAMPA phoneme set and #-mark for phones of worse quality used for annotation of phones; additionally, the beginnings of words are marked with ## and the beginnings of the syllables within the words are marked with “.”
2. syllables - “s” for syllables, “yyy” for filled-pauses for syllable tier annotation
3. speech – orthographic transcription in Polish
4. English – translation of the Polish speech tier into English

5. dialogue acts – Bunt's dialogue acts categories used for the pilot study annotation (Bunt 2000, cf. 4.4.2 Dialogue act annotation)
6. special – on this tier speech events, such as filled-pauses, confirmations or hesitations are marked.

The phone annotation was performed in order to create MBROLA micro-voices. First, the text of the dialogue was written and later the phonemic annotation was performed manually using the extended SAMPA phoneme set presented in Figure 29. Apart from the extended SAMPA phoneme set, one new (phoneme) segment was introduced, i.e. the phoneme /h/ which appears in "mhm" which means confirmation or agreement. Phones which were not pronounced clearly enough were marked with additional #-mark, e.g. /a#/ or /n#/. This #-mark will be used to neglect the diphones of the worse quality if possible, and choose the better diphones.

The syllable annotation was performed in order to carry out measurements on this speech unit. The transcription and translation of speech was done in order to analyse the speakers' utterances and use them later in the dialogue model for the demo dialogue system.

Additionally, annotation of dialogue acts was performed. For dialogue acts annotation Bunt's categories of Dynamic Interpretation Theory (DIT) (Bunt 2000, cf. Gibbon 2009). Unlike it was in the pilot study, described in (4.4.2 Dialogue act annotation), not only the main categories were annotated, but also more detailed information about the dialogue acts was marked. For example, instead of the main category "auto-feedback", it was specified whether the auto-feedback was positive or negative. Altogether, 20 different dialogue act categories were selected for the purpose of the present research and used for the annotation. They are presented in Table 47 together with their frequency of appearance. (The dialogue acts frequency is visualised in a chart in Figure 51.) In the last column of Table 47, main DA categories, there are categories used for the pilot study above. It does not mean that the "main" categories are highest in the dialogue act category hierarchy, but they were chosen to distinguish different dialogue acts, and to generalise other dialogue acts. For example, "request" belongs to the category "directives", which belongs to the category "action discussion functions", which belongs to the main category "general-

purpose communicative functions". Because in the pilot study, the "directives" category was used, therefore in the table the "request" category was assigned to the category "directives" and not to the "general-purpose communicative functions".

The annotation on 6 levels: phones, syllables, speech, English, dialogue acts and special for Speaker A and Speaker B is presented in Figure 50.

Table 47: Dialogue acts frequencies and their statistics used for dialogue annotation. N is the number of DA

	Speaker A	***N***	***Speaker B***	***N***	***DA full name***	***Main DA categories***
1	allo_neg	1			negative allo-feedback	allo-feedback
2			agree	1	agreement	information providing
3	auto	3	auto	4	auto-feedback	auto-feedback
4	auto_pos	4	auto_pos	10	positive auto-feedback	auto-feedback
5			closing	1	valediction: goodbye	social obligations management
6	cnt	1	cnt	6	contact management	contact management
7	confirm	6	confirm	5	confirm	information providing
8	dir	17	dir	1	directives	directives
9	disagree	2			disagreement	information providing
10	infpr	38	infpr	9	information providing	information providing
11	infsk	2	infsk	17	information seeking	information seeking
12	open	1	open	2	opening	opening
13	own	6	own	1	own communication management	own communication management
14	preclosing	2			preclosing	discourse structure management
15			promise	2	promise	commissives
16			request	1	request	directives
17	social	1	social	2	social obligations management	social obligations management
18	time	20	time	4	time management	time management
19	topic	6	topic	1	topic introduction / shift	discourse structure management
20	turn	3	turn	3	turn management	turn management
		113		***70***		

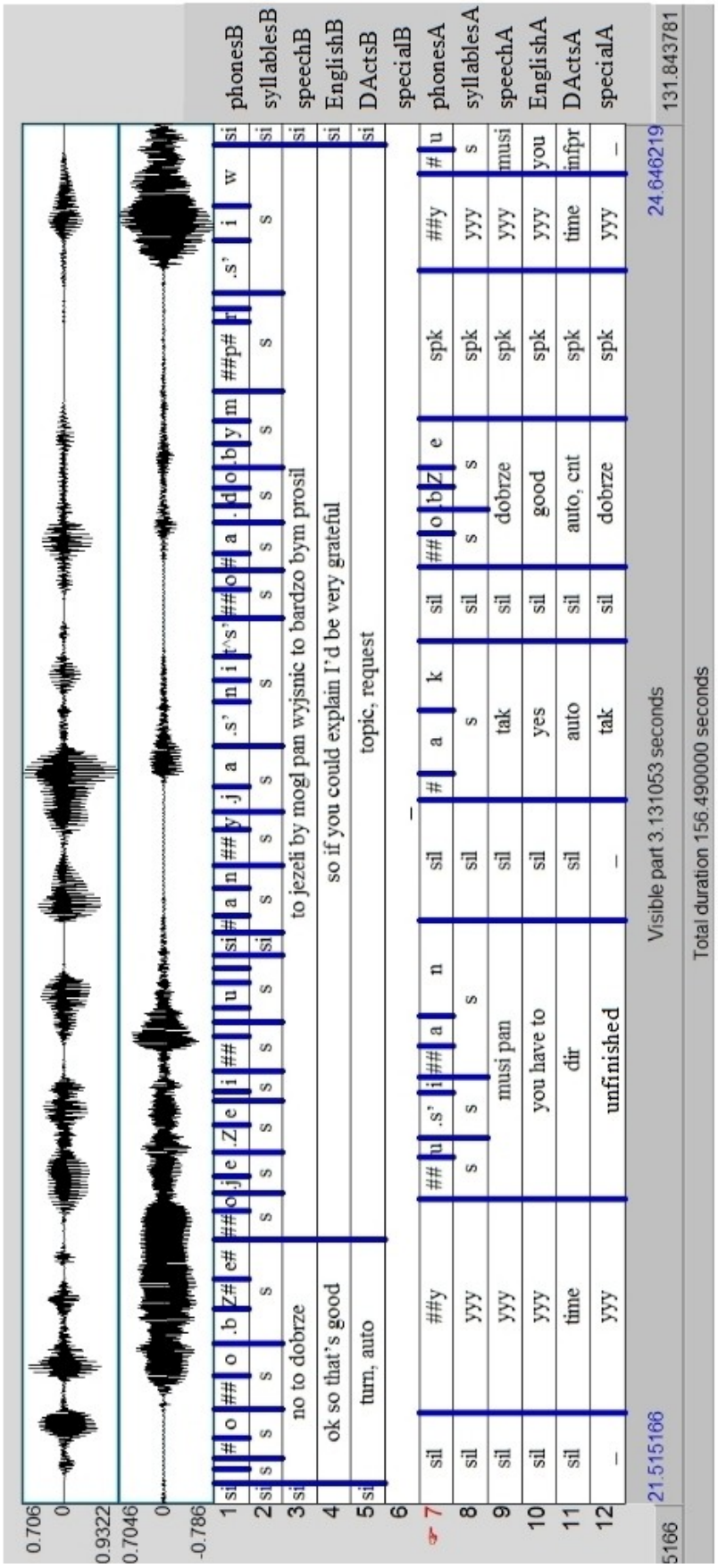


Figure 50: Annotation of dialogues on several tiers for Speaker A (channel 2, bottom) and Speaker B (channel 1, top)

7.5.1 General analysis of the corpus

The full corpus consists of four parts:

1. Map task: emergency – fast speech, very formal, Speaker A domination
2. Diapix: emergency – formal speech, cooperative dialogue
3. Map task: cinema – informal speech, a lot of auto-feedback coming from Speaker B, laughter, fun talk
4. Diapix: shopping area – informal speech, cooperative dialogue, diminutives

The created scenarios turned out to be very suitable settings for different kinds of dialogues. The public character of the conversations as well as the stress elicitation techniques worked as planned. In the first emergency map task dialogue, the A speakers were under stress. Their speech was fast, they were making mistakes, but aimed at finishing the task quickly. They were dominant in this task taking much of the dialogue time. On the other hand, the B speakers were calm and cooperatively non-aligned with the interlocutors. They tried to finish to the task successfully, but their speech was not affected by stress or fear. In this scenario, both speakers were very formal and used honorific forms of address.

In the diapix emergency dialogues, the A speakers were not so dominant. They started the description of their picture in order to find differences, but when their ideas finished, they let the B speakers talk. In these dialogues, the interlocutos were also very formal and their speech was affected by the stress factor.

The neutral tasks with the route description to the cinema totally changed the A speakers. Although the pairs of speakers were the same people and still they did not know each other previously, their style of speaking changed completely. First, their speech was relaxed, it was much slower and it could be noticed that they were happy with the idea of helping others with such a leisurely task as a visit to the cinema. Second, in their speech many colloquial words and phrases appeared which were not present in the previous dialogues. The speech of the A speakers was not dominant to the same extent as in the emergency scenario. Both speakers were making additional comments not connected with the task itself which could be described as ‘fun talk’.

The recordings of the last scenario, the diapix of the shopping area, also differed from the other dialogues. In many cases, Speaker B was dominant, as in a normal setting where he or she would be socially superior to the student. However, these dialogues were still very cooperative, including colloquial vocabulary, diminutives and laughter.

7.5.2 Analysis of the selected dialogue

General measurements on one selected emergency dialogue annotated on 6 tiers (Speakers IDs: 12A and 12B) were carried out. These measurements are presented in Table 48. On the speech tier, 139 intervals for Speaker A were annotated and 85 intervals for Speaker B. These figures include also silences. Looking at the Speech duration column, a huge difference is to be found between the inputs given from both of the speakers. A bit more about the dialogue flow can be said when looking at the Longest silence figures. 28.87sec of Speaker's B complete silence says that the speaker let his interlocutor talk without any interruption for a long period of time. More than one dialogue act could be assigned to one stretch of speech (speech interval), therefore there is a bigger number of dialogue acts than there is the speech intervals annotated in the dialogue. Syllables and Phones columns show the number of these units annotated in the selected dialogue. Finally, the Special column contains the number of special speech events such as filled-pauses, simple confirmation, speaker noises, etc.

Table 48: Dialogue statistics of emergency dialogue (pair ID: 12). Total dialogue duration 156.49sec

	Intervals on Speech tier	***Speech duration***	***Longest silence***	***Speech intervals***	***Dialogue acts***	***Syllables***	***Phones***	***Special***
SpkA	139	106.71sec	4.74sec	79	113	506	1231	94
SpkB	85	42.02sec	28.87sec	52	70	241	579	36

This is the visualisation of the numerical data presented in Figure 51. To the "other" category belong the dialogue acts which appeared only once or twice in the dialogue for both speakers. The distribution of different dialogue acts differ for both speakers. Speaker A produces many information providing dialogue acts and directives, whereas Speaker B mainly seeks for information or gives positive auto-feedback. Additionally, in Speaker's A speech many time management dialogue acts are found, these are the filled pauses where the speaker tried to formulate his utterance.

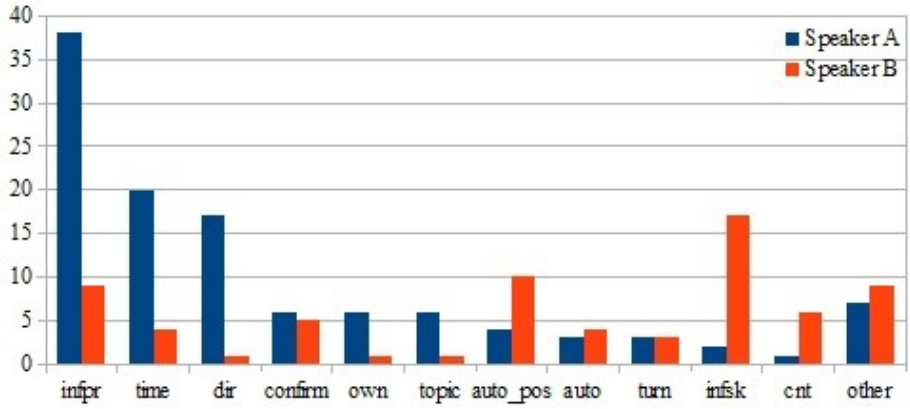


Figure 51: Dialogue acts frequency

The speech events annotated on the Special tier were grouped into 6 categories. These categories together with their frequencies of occurrence are presented in Table 49. Altogether, much more special speech events were annotated on the Speaker's A tier. There were 38 filled-pauses, 10 mistakes, and 24 speaker noises marked in Speaker's A speech. Also, the number of events in the "other" category is high and equals 14. These were the unfinished turns as well as many prolonged phones connected with hesitations, similar to filled-pauses. For example, Speaker A would say "do budkiii z lodami" (*Eng.* to the ice-cream stand), where the prolonged "iii" originated from the fact the speaker needed a few milliseconds in order to find the word for "lody" (ice-cream).

Speaker's B Special tier is less rich and the high number of confirmations attract attention. These are the positive auto-feedback dialogue acts. There are quite many confirmations in the Speaker's A speech. Both speakers used similar words for the confirmation among which were: "tak" (*Eng*. yes), "dobrze" (*Eng*. good), "acha" (Eng. aha), "rozumiem" (*Eng*. I understand), "okej" (*Eng*. okay), "mhm", and combinations of those like "tak, rozumiem" or "tak tak tak".

Speker's B also frequently finished his question with the "tak?" (*Eng.* yes?) question mark underlining the fact that he was seeking information.

Table 49: Special events frequencies

	All	*filled pause "yyy"*	*confirmation*	*yes?*	*mistake*	*spk*	*other*
SpkA	99	38	13	0	10	24	14
SpkB	36	5	16	5	2	5	3

Additionally, the measurement of the Min, Max and Mean values of Speakers' pitch (F0) were extracted using the Praat tool (Boersma & Weenink 2001) for all the five recording tasks. The pitch values are presented in Table 50.

Table 50: Min, Max and Mean (M) pitch values (F0) for Speaker A and Speaker B across the five recording tasks.

	Map task: emergency			*Diapix: emergency*			*Map task: cinema*			*Diapix: shopping area*			*Reading*		
	Min	***Max***	***M***	***Min***	***Max***	***M***	***Min***	***Max***	***M***	***Min***	***Max***	***M***	***Min***	***Max***	***M***
SpkA	52	295	165	49	255	158	59	274	177	58	295	165	69	220	161
SpkB	63	245	134	60	274	129	61	278	136	63	273	126	54	223	121

7.5.3 Duration analysis: the *nPVI* index

The *nPVI* index was calculated to find phonetic duration differences between the speakers (and perhaps the dialogue acts) to use in the speech synthesis module for alignment purposes. The normalised pairwise variability index, *nPVI*, is a measure of smoothness, i.e. evenness of duration: if all durations are the same, then the index is be 0 (Asu & Nolan 2005). The formula for the pairwise variability index is:

$$nPVI = 100 \times \left[\sum_{k-1}^{m-1} \left| \frac{d_k - d_{k+1}}{(d_k + d_{k+1})/2} \right| / (m-1) \right]$$

The formula calculates smoothness of durations by calculating average differences between the durations of neighbouring units in a corpus (generally phonemes P, syllables S or inter-stress units ISU). The duration differences are divided by the average of the two durations. This normalisation is intended to reduce the effect of speech rate differences.

The whole equation is the average normalised duration difference between all neighbours in the utterance. The measure is interpreted as a measure of similarity/difference, i.e. smoothness/roughness, between all neighbours:

1. If there is no difference in the durations, i.e. if each difference between neighbours is 0, then the index is 0.
2. The larger the normalised average difference, the more uneven the utterance. The index tends asymptotically towards 200. (I.e. it never quite reaches 200, there is never an index which is 200 or higher).
3. Smoothness (a low index) is usually interpreted as syllable timing, while roughness (a high index) is usually interpreted as timing which involves a larger unit, such as a foot, or a stress unit.
4. A small number may indicate syllable timing, but it is impossible to tell what kind of unevenness a larger number expresses.

The *nPVI* results for phones, syllables and pitch values of filled pauses are presented in Table 51. The test material was divided into three groups: All, Without filled pauses ("yyy") and only filled pauses ("yyy"). Theoretically, the number of filled pause for phones and syllables should be the same, however, there is a slight difference in the Speaker A figures and it results probably with a bit different annotation on both tiers.

The *nPVI* for phones and syllable duration, despite the fact whether the filled pauses are included or not, the results are similar and probably are not statistically significant. However, there is a big difference in the *nPVI* for filled pause duration. This result suggests that the Speaker A speech is less fluent and the speaker needs different amounts of time to formulate his utterances. Whereas, Speaker's B *nPVI* result for filled pauses are indifferent in comparison to other duration measures.

The *nPVI* was also calculated on filled pauses pitch values (fundamental frequency F0). For each filled pause, if existent, 16 mean pitch values were automatically extracted and used for nPVI calculation. The results show that the pitch contours of the filled pauses are very smooth and even. (More about it can be read in 7.6.3 ACCS synthesis of the filled pauses "yyy")

The measurements of the *nPVI* was calculated on quite small material, but for the present research the selected material seems to be representative. The comparison between the speakers shows tendency of alignment of acoustic parameters of speech such as segment durations, but further research need to be carried out in this field.

Table 51: nPVI for duration of phones, syllables and pitch values of filled pauses ("yyy"). N is number of items

		Phones		***Syllables***		***Filled pauses pitch***	
		N	***nPVI***	***N***	***nPVI***	***N***	***nPVI***
All	***SpkA***	1231	44.90	506	47.76	–	–
	SpkB	579	42.58	241	51.15	–	–
Without filled pauses	***SpkA***	1193	42.42	467	45.59	–	–
	SpkB	574	44.12	236	51.47	–	–
Only filled pauses	***SpkA***	38	77.36	39	78.22	525	4.93
	SpkB	5	43.28	5	43.28	89	1.19

7.6 Prototype dialogue synthesis

Annotation on the phone level was used to automatically extract diphones and to create two synthetic micro-voices for each of the speaker. Additionally, the phone annotation was used for the Automatic Close Copy Speech (ACCS) synthesis of the dialogue with the two independent Polish MBROLA voices, the already existing PL1 female voice and the PL2 male voice created within the present dissertation.

7.6.1 Diphone extraction for prototype MBROLA micro-voices

From the two-channelled dialogue recording annotated on the phone level, the channels were extracted for each of the speaker. Then, for each speaker the diphones have been extracted using the automatic diphone extractor presented in (6.9.3 Automatic diphone extraction system architecture). Before extraction, the extended SAMPA phoneme set was converted to the Polish SAMPA set used in the PL1 female voice. Only one instance of a diphone was extracted. If a phone was marked with # meaning worse quality of a phone, it was extracted as a part of a different diphone. For example, a-n and a#-n were different diphones. After the diphones were extracted, the diphone database SEG file was checked, and if possible, the diphones with # were manually removed. If , however, diphones containing # in the filename were the only instances of the diphones, they were renamed and # was removed from both, the filename and the diphone name. # had to be removed from the filename, because MBROLA did not want to mbrolate files with # in the name. Diphones, whose one segment was marked as /junk/ were omitted. The selection process for diphones starting with /b/ is presented in Table 52.

Table 52: Diphone manual selection process

Before						***After***					
Filename	***1st***	***2nd***	***S***	***E***	***M***	***Filename***	***1st***	***2nd***	***S***	***E***	***M***
b#-ii#_A_129.wav	b#	I#	800	1567	1190	b-ii_A_129.wav	b	I	800	1567	1190
b#-l_A_59.wav	b#	l	800	1639	1400	b-l_A_59.wav	b	l	800	1639	1400
b#-l#_A_364.wav	b#	l#	800	1363	1063						
b#-r#_A_8.wav	b#	r#	800	1250	1048	b-r_A_8.wav	b	r	800	1250	1048
b-zz_A_208.wav	b	Z	800	1568	1211	b-zz_A_208.wav	b	Z	800	1568	1211
b-a_A_934.wav	b	a	800	2762	1297	b-a_A_934.wav	b	a	800	2762	1297
b-j_A_1097.wav	b	j	800	1515	1352	b-j_A_1097.wav	b	j	800	1515	1352
b-o_A_660.wav	b	o	800	2591	1569	b-o_A_660.wav	b	o	800	2591	1569
b-on_A_69.wav	b	o~	800	2226	1430	b-on_A_69.wav	b	o~	800	2226	1430
b-u_A_755.wav	b	u	800	2220	1411	b-u_A_755.wav	b	u	800	2220	1411

The numbers of all extracted diphones and then their selection were as follows:

	All	Selected
Speaker A:	419	314
Speaker B:	264	238

This means that for creation of Speaker A MBROLA micro-voice, 314 diphones were used. And for creation of the micro-voice for Speaker B, 238 diphones were used.

During the mbrolation process, it was noticed that the diphone built of two silences /_-_/ should not be present in the SEG file or at least, it must be put at the end of the diphone list. MBROLA assumes that /_-_/ diphone is the last one and terminates the voice creation at this diphone. MBROLA adds automatically the /_-_/ diphone, so to avoid this problem, it is advised to remove the /_-_/ diphone from the SEG file.

7.6.2 ACCS synthesis of the dialogue

The selected dialogue annotated on the phone tier was synthesised using the Automatic Close Copy Speech (ACCS) synthesis system (Bachan 2007a). The speech channel for each speaker was synthesised separately using different voices, and then combined in order to get a 2-channel dialogue audio file. For the synthesis, 4 different voices were used:

1. pl1 female voice
2. pl2 male voice

3. marcin micro-voice – Speaker's A voice
4. pawel micro-voice – Speaker's B voice

A synthesised excerpt of a dialogue is presented on Figure 52. It shows waveforms of Speaker's A and Speaker's B speech signal, their pitch contours and the annotation. The Figure was generated using a Praat script which allowed to draw pitch contours for each speaker separately.[5] (The script is to be found in Appendix O.)

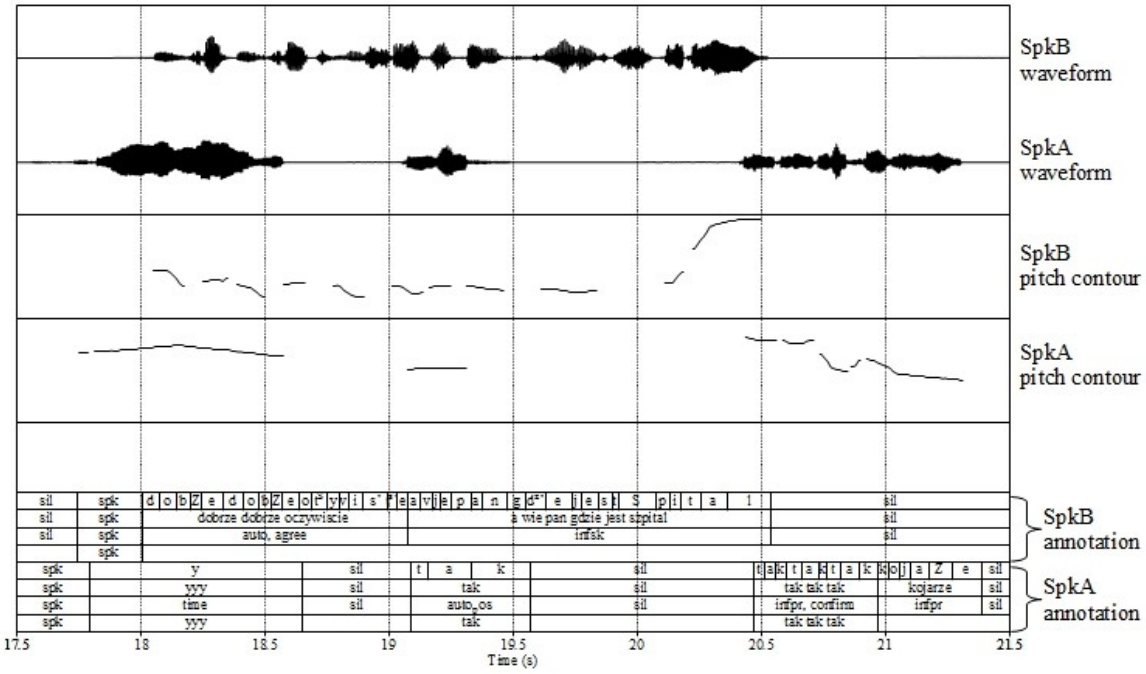


Figure 52: Speaker's A and Speaker's B waveforms, pitch contours and annotation tiers of a synthesised dialogue excerpt at 17.5 to 21.5 second

7.6.3 ACCS synthesis of the filled pauses "yyy"

Filled pauses pauses are interesting speech units for analysis for many researchers (e.g. Štefan Beňuš 2009). In the present work they were also looked at to analyse their melody and their potential use for generating feedback by the dialogue system. For each Speaker's filled pauses, 16 *mean* pitch values were automatically extracted and the MBROLA PHO files were created. (The PHO file is the format accepted by the MBROLA speech synthesiser). Each filled pause was separated from the other by a 100msec pause. If a filled pause was long and more than one segment could be distinguish (e.g. "yyy yyy yyy" in the

5 The author is grateful to David Weenink for help in creating the Praat script at Interspeech 2009 conference.

annotation), then these consecutive "yyy" were synthesised without the 100msec pause in between. The synthesised filled pauses with the Speaker A and Speaker B MBROLA micro-voices are presented on Figure 53.

The general conclusion may be drawn that the synthesised filled pauses for both speakers look very similar on the spectrogram and their pitch contour does not differ a lot. However, Speaker's A filled pause pitch contours are richer. There are filled pauses with a lower and higher pitch, as well long filled pause in which up to three "yyy" segments can be distinguished. Within these long filled pauses the pitch contour changes dramatically, falling from mean to very low values and then rising again to the mean pitch values.

The smoothness and evenness of the pitch values of the filled pauses was confirmed by the *nPVI* results presented in section 7.5.3 Duration analysis: the nPVI index and were equal 4.93 for Speaker A and 1.19 for Speaker B.

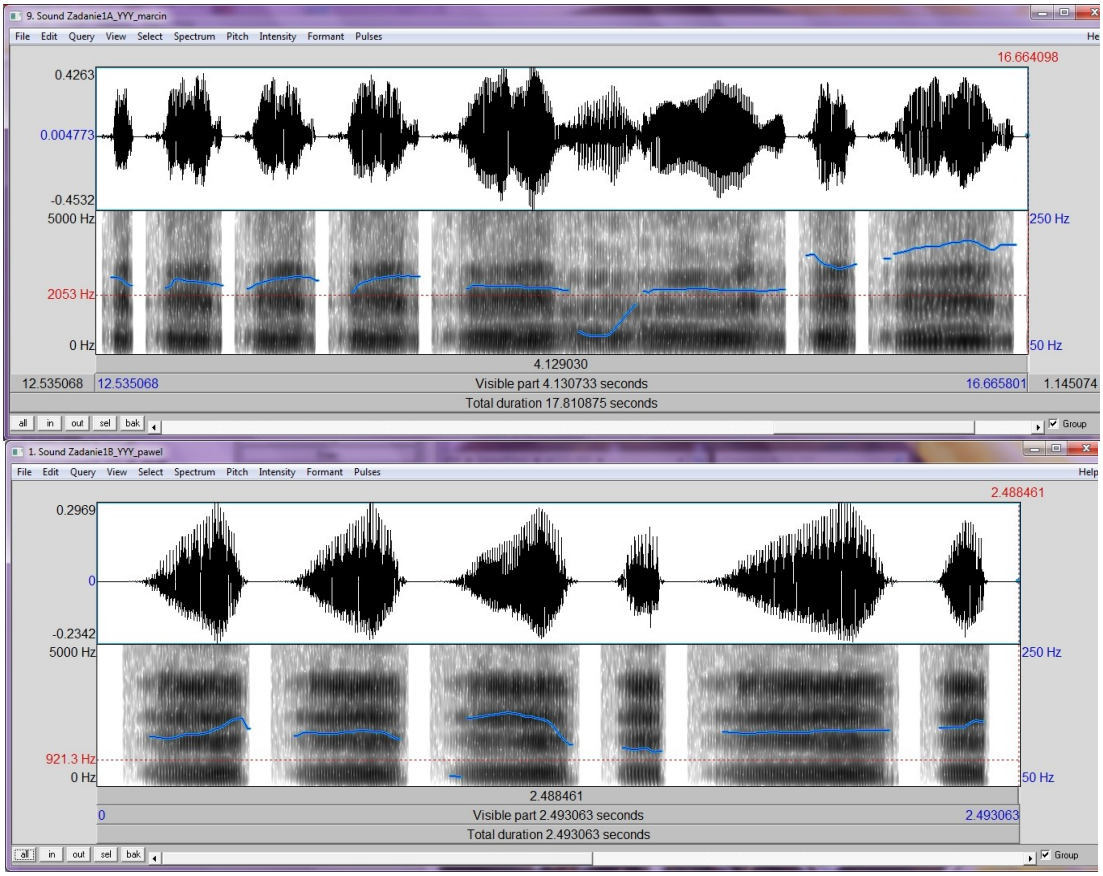


Figure 53: Examples of the ACCS synthesised filled pauses for Speaker A (top) and Speaker B (bottom)

7.7 Finite State Transducer model of the map

In Chapter 5, the finite state automata were presented as models for dialogue sequences. In the present section, the map used for the dialogue recordings will be mapped onto the FST to control the traverse along the map.

The emergency map can be represented as a finite state transducer (FST) where each junction corresponds to the transition node. However, not all the streets are open. Some junctions cannot be reached, because there is no way through. There is a traffic jam on the way or roadworks, and even at one place the street has been blocked because of a school race. Such junctions are not taken into account when designing the FST. Traffic on all the other streets is two-way, so turnings back are not hampered. Moving along the map, some route is followed. At a normal map, the route can be tracked thanks to the street names or the landmarks being passed on the way. In the FST, the street names and the landmarks can be replaced by Latin letters for simplification. Such an analysis of the map resulted in creation of a FST, modelling the movements along the map in order to reach the goal. The design process and the FST are presented on Figure 54. Figure (A) presents the emergency map with the marked junctions for selections. All the junctions which cannot be reached because of the obstacles on the way were not selected for the nodes of the FST. On figure (B) there is the FST, with the enumerated transition nodes and the transitions producing Latin letters. q0 is the start node and q13 is the end node.

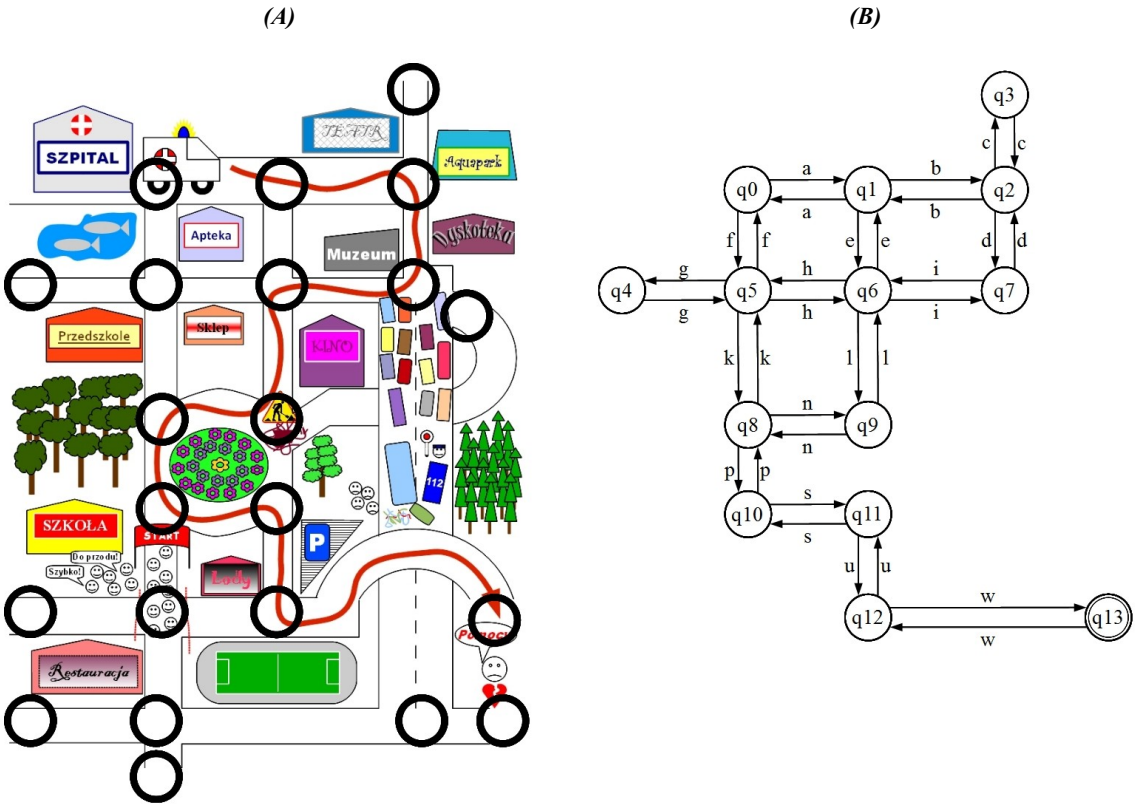


Figure 54: (A) Emergency map with all junctions marked for selection for the FST nodes; (B) Emergency dialogue automaton with the nodes representing the reachable junctions selected

The emergency dialogue was analysed in order to find correspondence with the FST. Dialogue was divided into 29 utterance exchanges according to the specification described below. The first 2 exchanges were aimed to open the dialogue and set the topic. The last exchange aimed at closing the dialogue. These 3 utterance exchanges were not taken into account in the current analysis as they did not include instructions about how to move on the map. But it is important to note that in the second exchange when the topic was set, the speakers agreed from where to start the route, so they described the start node. The other 26 utterance exchanges were analysed and instructions were compared with the transitions of the FST. Each instruction led to moving from one node to another node of the FST, resulting in creating a route as a sequence of Latin letters. However, spoken instructions were not structured as in would be expected for the FST. Spoken instructions led to jumps over one or more nodes, neglecting the nodes which were on the way. Although unclear for the FST, the instructions were understood by the human partner. Sometimes the

instructions led to jumps back in order to explain the route again. Finally, there were utterance exchanges clarifying the current position on the map.

The division of the dialogue into instructions of the route is presented in Table 53. In the ID column, there are the different ID numbers of the utterance exchanges. S and E are the start and end nodes assigned to the utterance exchange. In the T column, there is the transition route represented as a letter or a sequence of letters if the instruction leads to jump over a few nodes. If the instructions lead to turn back to some earlier node, also such transition is represented as a sequence of letters. ? stands for no transition and appears if the utterance exchange correspond to a local loop. Spk, Polish and English column include information about the speaker who is speaking, what is being said and its translation into English respectively. // stands for the borders in the annotation. DA contains information about the dialogue act categories assigned to different utterances. sil stands for silence and * means unfinished utterances

Table 53: Utterance exchange in the emergency map task dialogue

ID	***S***	***E***	***T***	***Spk***	***Polish***	***English***	***DA***
3	q0	q2	a b	A:	musi pan//sil//tak//sil//dobrze//spk//yy y//musi pan jechac yyy prosto//spk//az dojedzie pan dooo yyy do* do do* [spk] do najblizszej ulicy	you have to//sil//yes//sil//good//spk//yyy// you have to drive yyy straight//spk//until you get toooo yyy to* to to* [spk] the closest street	dir//sil//auto//sil//a uto, cnt//spk//time//inf pr, dir//spk//infpr (own)
4	q2	q7	d	A:	spk//czyli tam//yyy//gdzie mamy muzeum musi pan skre'cic w prawo	spk//that is there//yyy//where we have the museum you have to turn right//sil	infpr (own)//spk//infpr// time//infpr, dir
				B:	dobrze tak rozumiem	good yes I understand	auto_pos, confirm
5	q7	q6	i	A:	nast*epnie//sil//tak//nastepnie gdy juz pan yyy yyy skreci w prawo [spk] yyy jeszcze yyy na* najblizszy skret w prawo	next//sil//yes//next when you already yyy yyy you turn right [spk] yyy again yyy the clo* closest right turn	topic, turn//sil//auto//infp r//time//infpr (own)
6	q6	q9	l	A:	spk//minie pan kino//sil//nastepnie yyy w lewo//yyy	you pass the cinema//sil//then yyy left//sil//yyy	spk//infpr, dir//sil//infpr, dir//sil//time
7	q9	q2	l i d	B:	i ja rozumiem ze za dyskoteka z powrotem w prawo//tak?	I understand that after the disco back to the right//yes?	infsk, infpr//cnt, infsk
8	q2	q7	d	A:	spk//zaaa muzeum	spk//aaafter museum	spk//infpr
				B:	najblizszym//sil//za muze* y//sil//ycha acha	the closest//sil//after muse* y//sil//aha aha	infsk//sil//auto//sil/ /confirm
				A:	za muzeum	after the museum	infpr
				B:	przy muzeum w prawo//rozumiem	at the museum right//I understand	infpr//confirm

ID	*S*	*E*	*T*	*Spk*	*Polish*	*English*	*DA*
				A:	tak	yes	confirm
				B:	mijam aqua park//spk	I pass the aqua park//spk	infpr//spk
				A:	kino//	the cinema	dir, disagree
9	q7	q7	?	B:	i wtedy?//w prawo?	and then?//right?	infsk//infsk
				A:	nie nie nie nie nie nie nie	no no no no no no no	allo_neg
				B:	tak?	yes?	cnt, infsk
				A:	spk//yyy musi* y zje*	spk//yyy you have to* y go*	spk//turn, time, own
				B:	mhm	mhm	auto_pos
10	q7	q0	d b a	A:	jeszcze raz	one more time	topic
11	q0	q2	a b	A:	spk//yyy szyb*ko yyy sk*reca ym mija//jedzie pan prosto ze szpitala	spk//yyy quick*ly yyy you turn* ym you pass//you drive straight from the hospital//sil	spk//time, own, infpr//infpr, dir
				B:	tak	yes	auto_pos
				A:	prosto	straight	confirm
12	q2	q7	d	A:	spk//y mija pan yyy muzeum skreca pan w prawo	spk//y you pass yyy museum you turn right	spk//infpr, dir
				B:	tak	yes	auto_pos
13	q7	q6	i	A:	nastepnie jeszcze raz skreca pan w prawo//yyy mijane jest muzeum	then once again you turn right //yyy the museum is passed	infpr, dir//infpr
				B:	rozumiem	I understand	auto_pos
				A:	spk//yyy i ma po lewej stronie ma pan kino	spk//yyy and you have on the left side you have the cinema	spk//infpr, infsk
				B:	tak	yes	auto_pos, confirm
14	q6	q9	l	A:	i* w* nas*tepnie//jeszcze raz//i* w* sk*//przy tym kinie skreca pan w lewo	and* in* ne*xt//one more time//and* in* t*urn//at this cinema you turn left	time, own//topic//time, own//infpr, dir
				B:	okej	okay	auto_pos
15	q9	q8	n	A:	yyy tam ma pan naaa rondzie roboty//sil//yyy musi pan je o* o* omina'c'	yyy there you have on the roundabout roadworks//sil//yyy you have to g* g* go around them	time, infpr//sil//infpr
16	q8	q11	p s	A:	spk//i nastepnie yyy gdy wyjedzie pan yyy z* z ronda	spk//and next yyy when you leave yyy the* the roundabout//sil	spk//infpr//sil
17	q11	q12	u	A:	y skrecajac yyy na rondzie w lewo//spk//yyymmm//dojedzie pan do* yyy budkiii z lodami//spk//iii //do parkingu po lewej stronie	y turning yyy at the roundabout left//spk//yyymmm//you will get to* yyy an ice-cream stand//spk//aaand//to the parking lot on the left side	infpr//spk//time//infpr//spk//time//infpr
18	q12	q13	w	A:	i nastepnie musi pan skrecic jeszcze raz w lewo//spk//i mamy tu yyy sk* yyy iii	and then you need to turn left one more time //spk//and we have here yyy sk* yyy aaand	infpr, dir//spk//infpr, own//sil//infpr

ID	*S*	*E*	*T*	*Spk*	*Polish*	*English*	*DA*
					s*//sil//przejazd przez yyy wiaduktem przez yyy yyy yyy nad wiadu*//spk//nad wiaduktem i tam po tej drugiej stronie wiaduktu na* znajduje sie ta osoba	s*//sil//a level crossing through yyy a viaduct through yyyy yyy yyy over the viadu*//spk//over the viaduck and there on the other side of the viaduct on* there is that person	(own)//spk//infpr
19	q13	q9	w u s p n	B:	a czyli rozumiem ze ja na rondzie mam skrecic w lewo	and so I understand that at the roundabout I need to turn left	infsk
				A:	spk	spk	spk
				B:	tak?	yes?	cnt, infsk
20	q9	q8	n	A:	tak	yes	confirm
				B:	na rondzie w lewo	at the roundabout left	infpr
				A:	mh//sil//w prawo	mh//sil//right	time//sil//dir, disagree
21	q8	q9	n	B:	a potem?	and then?	infsk
22	q9	q8	n	A:	spk//znaczy* oczywiscie nie ma nie ma pan wlasnie wyjscia tam na rondzie yyy sa roboty wiec w prawo	spk//it means* of course you don't you don't actually have a choice there at the roundabout yyy there are roadworks so right	spk//infpr
				B:	acha	aha	auto_pos
				A:	spk	spk	spk
				B:	spk//bo wlasnie w* z* z mojej mapy wynika ze nie ma robo't	spk//because actually at* on* on my map it says there are no roadworks	spk//infpr
23	q8	q11	p s	A:	omija pan rondo i do konca	you go around the roundabout and then to the end	dir
				B:	ale widocznie cos sie musialo zmienic	but apparently something must have changed	infpr, infsk
				A:	acha	aha	auto_pos
				B:	spk//yyy//dobrze wobec tego na rondzie skrecam w lewo	spkyyy//dobrze wobec tego na rondzie skrecam w lewo	spk//time//infpr
24	q11	q11	?	B:	i wtedy?//za rondem?	i wtedy?//za rondem?	infsk//infsk
				A:	spk//yyy//(po) prostu mija pan bo tam nie ma nie ma wyjscia//jest tam (po) prostu objezdza pan rondo i*	spk//yyy//sim* simply* you pass because there is no there is no choice//there is (sim*) simply you go around the roundabout and*	spk//time//infpr, dir//infpr, dir
25	q11	q12	u	A:	spk//y//dalej jedzie pan yyy prosto maj'ac' po lewej stronie praking i po prawej stronie budke z lodami	spk//y//further you drive yyy straight passing on the left side the parking lot and on the right the ice-cream stand	spk//time//infpr, dir
26	q12	q13	w	A:	spk//nastepnie skreca pan jeszcze raz yyy w lewo poprzez wiadukt//i taaa* i za wiaduktem znajduje sie ta osoba	spk//next you turn once again yyy left throughout the viaduct//and there* and after the viaduct there is this person	spk//infpr//infpr, own

ID	*S*	*E*	*T*	*Spk*	*Polish*	*English*	*DA*
27	q13	q12	w	B:	acha//czyli rozumiem musze przejechac przez wiadukt//tak?	aha//so I understand that I need to drive through the viaduct//yes?	auto, turn//infsk//cnt, infsk
				A:	tak//to wszystko	yes//that's all	confirm//preclosing
28	q12	q13	w	B:	y//sil//nad wiaduktem//tak?	y//sil//above the viaduct//yes?	turn, time//sil//infsk//cnt, infsk
				A:	mhm tak	mhm yes	confirm
				B:	acha acha acha	aha aha aha	auto_pos

The data about transitions from Table 53 were transferred onto the map FST. The instructions in the utterance exchanges led to moving along the FST and these moves are visualised as red arcs with utterance exchange IDs on Figure 55. Following the utterance exchange IDs shows how the dialogue proceeded. Analysis of the data about the transitions and utterances and dialogue act categories shows the following characteristics of the dialogue:

1. Most of the instructions *forward* expressed in one dialogue utterance exchange led to move forward from one node to adjacent node. Only the instruction leading from q0 to q2 nodes (IDs: 3,11), did not explicitly underline the transition at q1 node. Also the instruction which led to go around the roundabout did not underline the intermediate q10 node (IDs: 16, 23).
2. The turnings back (loops) lead to moving far *backward* over a few nodes (IDs: 7, 10, 19). There are turnings back to the adjacent nodes (IDs: 21, 27) but other go backward over up to 4 nodes (ID: 19).
3. Every move *forward* is repeated, therefore the are always at least 2 different IDs over each forward arc. This means that the instruction *forward* were repeated to make sure the mistake is avoided and the move is correct.
4. Utterance exchanges could be visualised as local loops (IDs: 9, 24) and they did not lead to any move either forward or backward.
5. Each move *backward* and each local loop was initiated by information seeking dialogue act.

6. Speaker A produced many information providing dialogue acts and directives and Speaker B role was limited to giving positive auto-feedback and confirmation dialogue acts.

7. Disagreement dialogue acts by Speaker A was preceded with information providing coming from Speaker B (IDs: 8, 20). This means that whenever Speaker B was uncertain about the route, he presented his reasoning in information providing dialogue acts and this led to disagreement and Speaker A tried to explain again the route. Disagreement (ID: 8, 20) led either to a local loop (ID: 9) or a turning back (ID: 21).

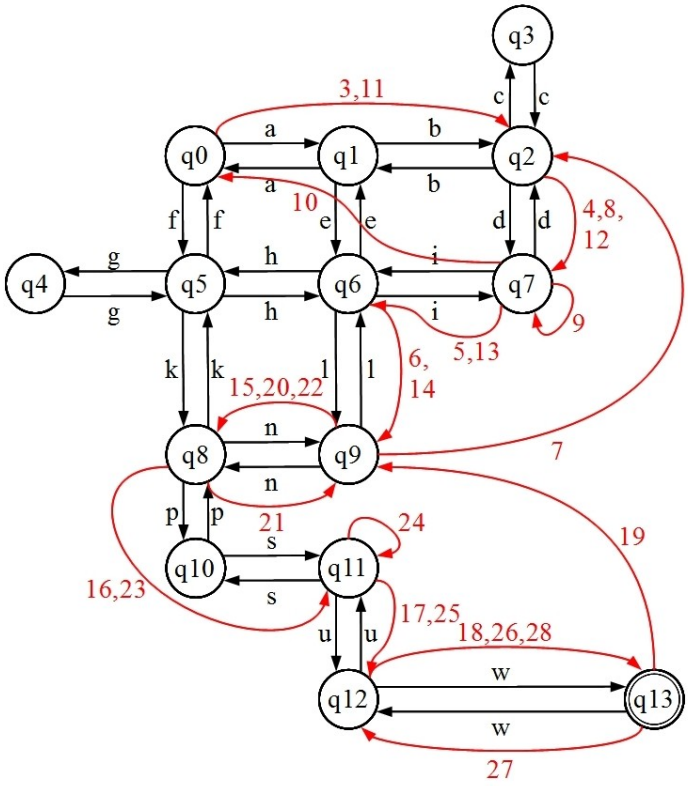


Figure 55: Map FST with utterance exchanges IDs

In this dialogue, the alignment phenomenon may be observed as a smooth move along the map, i.e. the FST. As long as the alignment is not disrupted, the interlocutors exchange information – instructions and positive auto-feedback, and move from one landmark to the next landmark on the map. However, immediately when a deviation from the alignment

takes place, a misalignment is reported and the interlocutors try to clarify the misunderstandings. The misalignment correction phenomenon relates to the finite state model as the backward movements in the traversal of the automaton.

The alignment on the semantic level is interpreted as obtaining the same semantic representation of the map. Because the maps which the interlocutors see differ, a misalignment must take place sooner or later. The recovery process from misalignment is very quick and intuitive and does not require many explanations from the speakers. Examples of misalignment from the semantic representation of the map are to be found it Table 53 at dialogue exchange IDs: 9, 22.

The data from Table 53 were rearranged in order to be used in the online NDFST interpreter (Nondeterministic Finite State Transducer) (Gibbon 2008). The instructions for the transitions which appeared in the dialogue are represented by an instruction or a set of utterance exchange between Speaker A and Speaker B inserted into { } brackets. If the instruction led to transition over one or more nodes, there the transitions are also inserted into { } brackets, for example {a b}. The transitions for which there were no instructions in the dialogue are replaced by an input symbol “x”. The code designed for use in the online NDFST interpreter is presented below:

```
# Initial state:
initial =     q0
# Set of terminal states:
terminal =     q13
# Transition     quadruples
<currentstate,inputsymbol,outputsymbol,nextstate>:
fst =
    q0, x, a, q1;
    q1, x, a, q0;
    q1, x, b, q2;
    q2, x, b, q1;
    q2, x, c, q3;
    q3, x, c, q2;
    q7, x, d, q2;
    q6, x, i, q7;
    q6, x, h, q5;
    q5, x, h, q6;
    q5, x, g, q4;
    q4, x, g, q5;
    q0, x, f, q5;
    q5, x, f, q0;
    q1, x, e, q6;
    q6, x, e, q1;
    q5, x, k, q8;
    q8, x, k, q5;
```

```
    q9, x, l, q6;
    q8, x, n, q9;
    q8, x, p, q10;
    q10, x, p, q8;
    q10, x, s, q11;
    q11, x, s, q10;
    q12, x, u, q11;
    q0, {SpkA: musi pan//sil//tak//sil//dobrze//spk//yyy//musi
pan jechac yyy prosto//spk//az dojedzie pan dooo yyy do* do do*
[spk] do najblizszej ulicy}, {a b}, q2 ;
    q2, {SpkA: spk//czyli tam//yyy//gdzie mamy muzeum musi pan
skre'cic w prawo
         SpkB: dobrze tak rozumiem}, d, q7 ;
    q7, {SpkA: nast*epnie//sil//tak//nastepnie gdy juz pan yyy
yyy skreci w prawo [spk] yyy jeszcze yyy na* najblizszy skret w
prawo}, i , q6 ;
    q6, {SpkA: spk//minie pan kino//sil//nastepnie yyy w
lewo//yyy}, l, q9 ;
    q9, {SpkB: i ja rozumiem ze za dyskoteka z powrotem w
prawo//tak?}, {l i d}, q2 ;
    q2, {SpkA: spk//zaaa muzeum
         SpkB: najblizszym//sil//za muze* y//sil//ycha acha
         SpkA: za muzeum
         SpkB: przy muzeum w prawo//rozumiem
         SpkA: tak
         SpkB: mijam aqua park//spk
         SpkA: kino}, d, q7 ;
    q7, {SpkB: i wtedy?//w prawo?
         SpkA: nie nie nie nie nie nie nie
         SpkB: tak?
         SpkA: spk//yyy musi* y zje*
         SpkB: mhm}, ?, q7 ;
    q7, {SpkA: jeszcze raz}, {d b a}, q0 ;
    q0, {SpkA: spk//yyy szyb*ko yyy sk*reca ym mija//jedzie pan
prosto ze szpitala
         SpkB: tak
         SpkA: prosto}, {a b}, q2 ;
    q2, {SpkA: spk//y mija pan yyy muzeum skreca pan w prawo
         SpkB: tak}, d, q7 ;
    q7, {SpkA: nastepnie jeszcze raz skreca pan w prawo//yyy
mijane jest muzeum
         SpkB: rozumiem
         SpkA: spk//yyy i ma po lewej stronie ma pan kino
         SpkB: tak}, i , q6 ;
    q6, {SpkA: i* w* nas*tepnie//jeszcze raz//i* w* sk*//przy
tym kinie skreca pan w lewo
         SpkB: okej}, l, q9 ;
    q9, {SpkA: yyy tam ma pan naaa rondzie roboty//sil//yyy
musi pan je o* o* omina'c'}, n, q8 ;
    q8, {SpkA: spk//i nastepnie yyy gdy wyjedzie pan yyy z* z
ronda}, {p s}, q11 ;
    q11, {SpkA: y skrecajac yyy na rondzie w
lewo//spk//yyymmm//dojedzie pan do* yyy budkiii z
lodami//spk//iii //do parkingu po lewej stronie}, u, q12 ;
    q12, {SpkA: i nastepnie musi pan skrecic jeszcze raz w
lewo//spk//i mamy tu yyy sk* yyy iii s*//sil//przejazd przez
yyy wiaduktem przez yyy yyy yyy nad wiadu*//spk//nad wiaduktem
```

```
i tam po tej drugiej stronie wiaduktu na* znajduje sie ta
osoba}, w, q13 ;
    q13, {SpkB: a czyli rozumiem ze ja na rondzie mam skrecic w
lewo
          SpkA: spk
          SpkB: tak?}, {w u s p n}, q9 ;
    q9, {SpkA: tak
         SpkB: na rondzie w lewo
         SpkA: mh//sil//w prawo}, n, q8 ;
    q8, SpkB: a potem?}, n, q9 ;
    q9, {SpkA: spk//znaczy* oczywiscie nie ma nie ma pan
wlasnie wyjscia tam na rondzie yyy sa roboty wiec w prawo
         SpkB: acha
         SpkA: spk
         SpkB: spk//bo wlasnie w* z* z mojej mapy wynika ze nie
ma robo't}, n, q8 ;
    q8, {SpkA: omija pan rondo i do konca
         SpkB: ale widocznie cos sie musialo zmienic
         SpkA: acha
         SpkB: spk//yyy//dobrze wobec tego na rondzie skrecam w
lewo}, {p s}, q11 ;
    q11, {SpkB: i wtedy?//za rondem?
          SpkA: spk//yyy//(po) prostu mija pan bo tam nie ma
nie ma wyjscia//jest tam (po) prostu objezdza pan rondo i*}, ?,
q11 ;
    q11, {SpkA: spk//y//dalej jedzie pan yyy prosto maj'ac' po
lewej stronie praking i po prawej stronie budke z lodami}, u,
q12 ;
    q12, {SpkA: spk//nastepnie skreca pan jeszcze raz yyy w
lewo poprzez wiadukt//i taaa* i za wiaduktem znajduje sie ta
osoba}, w, q13 ;
    q13, {SpkB: acha//czyli rozumiem musze przejechac przez
wiadukt//tak?
          SpkA: tak//to wszystko}, w, q12 ;
    q12, {SpkB: y//sil//nad wiaduktem//tak?
          SpkA: mhm tak
          SpkB: acha acha acha}, w, q13
```

7.8 Summary

In this chapter, a dialogue corpus was created for the purpose of the testing the thesis. A technique of collecting semi-spontaneous and cooperative speech with diapixes was used – a technique which is complementary to the map-task and in which usually only one speaker plays the dominant role. The recorded dialogues were aligned on many levels across 4 scenarios (emergency and neutral) but only semantic alignment was discussed in detail. The alignment features are used for building the dialogue model for the demonstration dialogue system.

The thesis that the alignment on the semantic level is essential for successful communication and any misalignment need to be recovered quickly was confirmed.

Chapter 8: Demonstration dialogue system

8.1 Overview

The operational objective of the thesis is to provide a text-in-voice-out demonstration system as a proof of concept for the principles of dialogue management, dialogue act sequencing and alignment which have been investigated in previous chapters, using the Polish male synthetic voice which was developed for use in the dialogue system. First, the requirement specifications for the system are summarised; second, the design of the system is presented; third, the implementation is described; finally, the evaluation procedure and results are provided.

8.2 Requirement specifications

The operational goal of the demonstration spoken dialogue system is to communicate with the human user, a caller, via spoken and written media in order to get from point A to point B on a street map. The scenario is therefore multimodal, and involves speech and writing: The task for the human user is to explain a route in writing to a computer. The computer acts as a telephone operator at an emergency call centre, and the human caller explains to the SDS computer how an ambulance should follow a route in order to get to a person with a heart attack. The starting point is therefore a hospital and the end point is a place where the person with the heart attack is. The caller explains the route and the dialogue system draws it on its map and checks with the caller. However, the ambulance operator discovers that not all the streets on the dialogue system's map are passable and it is the task of the human user to figure out which route the ambulance can take, bearing the blockages in mind. The caller tells the system which route to take, and the system checks the input and either gives the caller positive feedback that it can move forward on the map, or negative feedback that the route is blocked or the suggested move is incorrect. The caller sees the map with street names, the start point (the hospital) and the end point (the ill person). However the caller is not provided with a graphical interpretation of the actual moves of the system. Only if the user asks for clarification of where the ambulance is now on the

map, information of where the ambulance is is shown. This scenario has been chosen because in communication on the phone, one person also does not see what the other person is drawing on their map in similar situations. Only when misunderstanding appears do both parties try to clarify exactly what there is on their maps.

The structure of the dialogue is in three main parts: an information exchange with greetings, the route negation, and the farewell. The dialogue between the computer and the caller starts with short greetings from the computer, asking for basic information about name and sex in order for the system to know whether it is talking to a female or a male and then, after making sure the user knows where the start point is, i.e. the hospital, the user is asked to explain the route.

The dialogue strategy is as follows. The caller provides text input into a command line in a terminal window and the dialogue system provides synthetic audio output. The user can insert any text, but in order to make the system move along the map, the human user needs to insert relevant information, i.e. the names of the streets on the map. When the system makes a move, then audio output is produced with positive feedback. Negative feedback is generated when the user inserts incorrect text (it is recognised as non-existent name of a street) or the street is blocked according to the computer's map, or the user suggests taking a street which violates the street arrangement on the map. For example, it is not possible to move directly from A to C, but the correct move is from A to B and then from B to C. The dialogue system contains a speech style selection module with two speech styles: formal and informal. The formal speech consists of formal utterances, whereas the informal speech consists of utterances with colloquial expressions, repetitions and filled pauses.

The interface of the systems is specified in terms of its dialogue syntax, semantics and pragmatics:

1. Syntax – the actual selection of utterances for the computer call centre synthesis come from the emergency map-task dialogue recorded for the dialogue corpus (dialogue ID: 12). These are either dialogue opening and closing sentences, or positive and negative feedback. The selected utterances produced either by Speaker A or Speaker B were synthesised using the ACCS method (Bachan 2007a). Some

utterances were modified according to the system's needs or were invented and synthesised for use in the demonstration system.

2. Semantics – apart from the opening and closing utterances, the feedback utterances describe dialogue system's reaction to the perceived correctness of the input provided by the human user. Positive feedback is produced when there is a match and the computer can make a move on the map following the human user's instruction. If the computer does not find a match and therefore cannot make a move, negative feedback is generated.
3. Pragmatics – the selection of dialogue acts and styles has been made to meet the needs of the emergency scenario and also the human users (young students). The emergency scenario requires formal style. However, the fact that the dialogue system is going to be tested with students demands taking into account an informal scenario. Therefore two speech styles have been developed, formal and informal, with expectations that the dialogue with informal utterances will be assessed as being more natural. The dialogue acts consist of opening dialogue utterances, asking for information (name, sex, readiness to start explaining the route), request to explain the route and at the end a promise that the ambulance will arrive soon and closing dialogue utterance.

   The feedback utterances within the route description dialogue part consist of positive auto-feedback or negative auto-feedback with information providing dialogue acts informing that the move cannot be made.

At present only the command line application is specified. In the future, it is planned to develop an online web system, with more advanced text parser and real time speech synthesis.

8.3 Design

8.3.1 The street map and data elicitation

The emergency map created and used for the map task dialogue in the dialogue corpus recording has been modelled as a transition diagram of a Finite State Automaton. The FSA transition states were assigned to the junctions, and the streets were named as transition

labels. The traffic between all the junctions is two-way. (For one-way streets, the inverse transition could be deleted.) However, on the original map, not all the streets were passable, so these would have to be treated the same way by the program. The transitions for impassable streets are not deleted, this is the task to be solved by the caller which streets are passable and which are not.

The instruction to the human caller is:

> Imagine that you are talking to a person from an emergency call centre. Your task is to direct an ambulance from the hospital to the person with the heart attack along the streets marked on the map. Write the name of the street to move the ambulance from one junction to the other. ATTENTION: Not all the streets are passable!

The human user inputs 'chat' text into the system in writing. The dialogue system communicates with the caller via audio output producing synthetic speech.

The map shown to the user is presented in Figure 56.

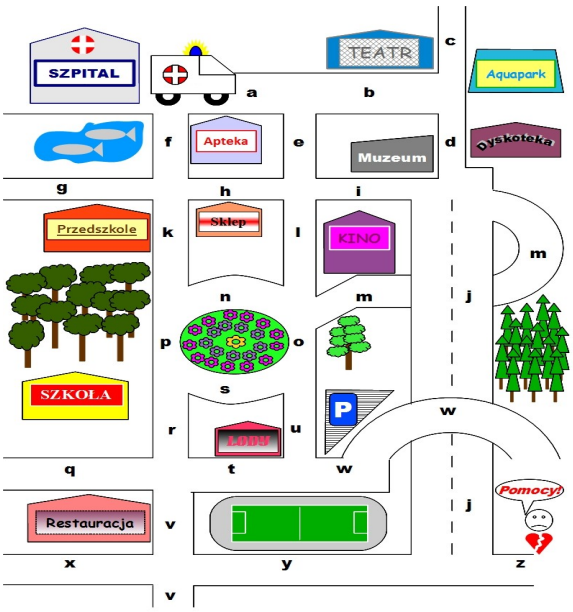


Figure 56: Emergency map presented to the human user for the communication scenario with the dialogue system

In Figure 57 the different transition states and the actual map traversal (MT) finite state automaton designed for the demo dialogue is presented. Because all the streets are two-way, the transitions are possible in two directions.

The transitions between different states are defined in a transition table. All the transition possibilities are shown in Table 54. The transition backward from the final state, q13, was removed, i.e. ['q13','w','q12'].

Table 54: Transitions of FSA designed for the dialogue system.

['q0','a','q1']	['q1','a','q0']	['q0','f','q5']	['q5','f','q0']	['q1','b','q2']	['q2','b','q1']
['q1','e','q6']	['q6','e','q1']	['q2','c','q3']	['q3','c','q2']	['q2','d','q7']	['q7','d','q2']
['q4','g','q5']	['q5','g','q4']	['q5','h','q6']	['q6','h','q5']	['q5','k','q8']	['q8','k','q5']
['q6','i','q7']	['q7','i','q6']	['q6','l','q9']	['q9','l','q6']	['q8','n','q9']	['q9','n','q8']
['q8','p','q10']	['q10','p','q8']	['q10','s','q11']	['q11','s','q10']	['q11','u','q12']	['q12','u','q11']
['q12','w','q13']					

Additionally, all the streets on the map as saved in a set of streets which contains: {'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'}

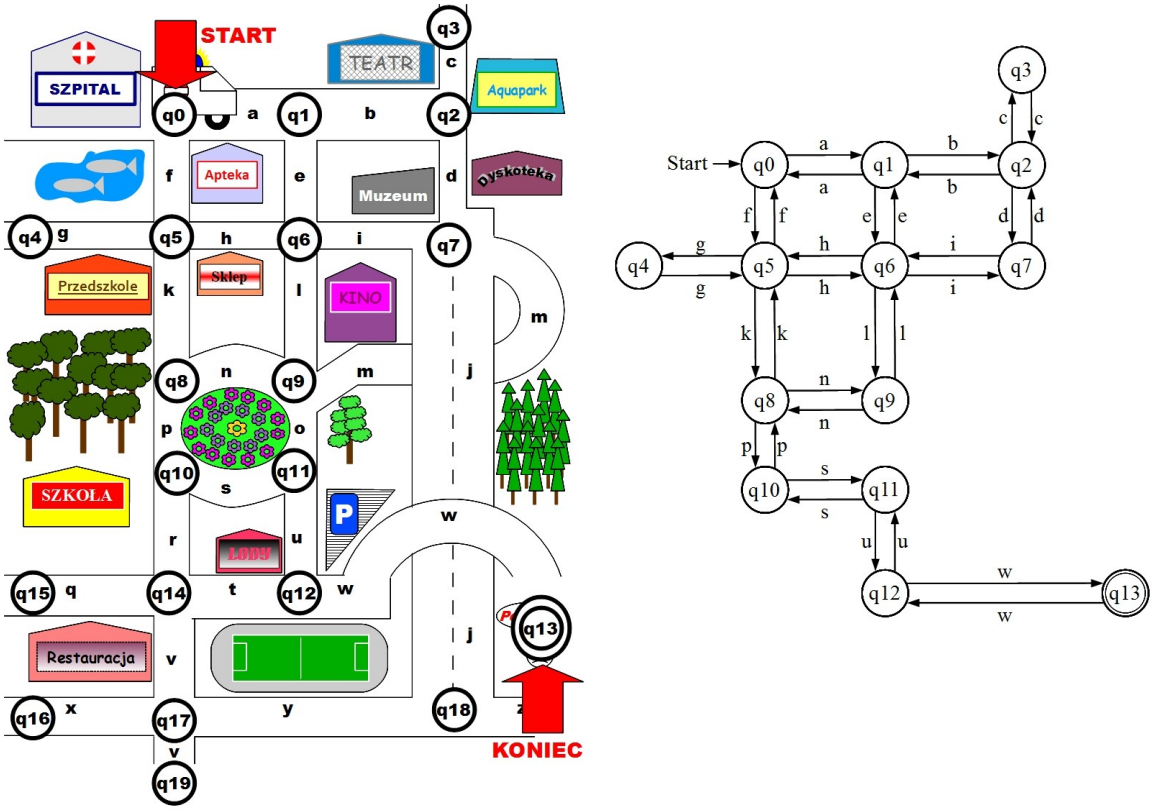


Figure 57: Map task dialogue as a basis for map traversal automaton

For the preliminary testing, a command line input is designed for the human user and synthetic audio output from the dialogue system to the user was planned. In Figure 58 the architecture of the dialogue system is presented. The interface in human-computer communication is based on text input from the caller into the terminal window. As already

noted, the computer communicates with the user via synthetic speech played via loudspeakers. Additionally, the caller is provided with a map with street on a computer screen. Based on personal characteristics of the user or randomly either formal or informal speech style is selected for the dialogue. This selection is made by the human experimenter.

The text input inserted by the human user is sent to a text analyser. The text analyser checks whether the input text matches the street_names set or if it is one of the function words (*?*, *what*, *exit*). The text analysis is sent to the dialogue manager which decides which speech reply to produce. The dialogue manager can either produce one of the opening or closing dialogue utterances or may produce different kinds of feedback following instructions from the user of where to move on the map. If the current state matches the transition triple <initial_state, street_name, next_state>, then positive feedback is produced. Otherwise, negative feedback from set II is selected. If the street name does not match any of the street names and none of the function words, then the negative feedback from set I is selected. If the selected text is recognised as one of the function words, then an appropriate action is undertaken by the dialogue manager.

The meanings of function words are:

1. *what* – repeat the last utterance (Polish ‘co’)
2. *?*– show the current state at which the ambulance is and display the map with states
3. *exit* – end the dialogue and move to the farewell section of the program

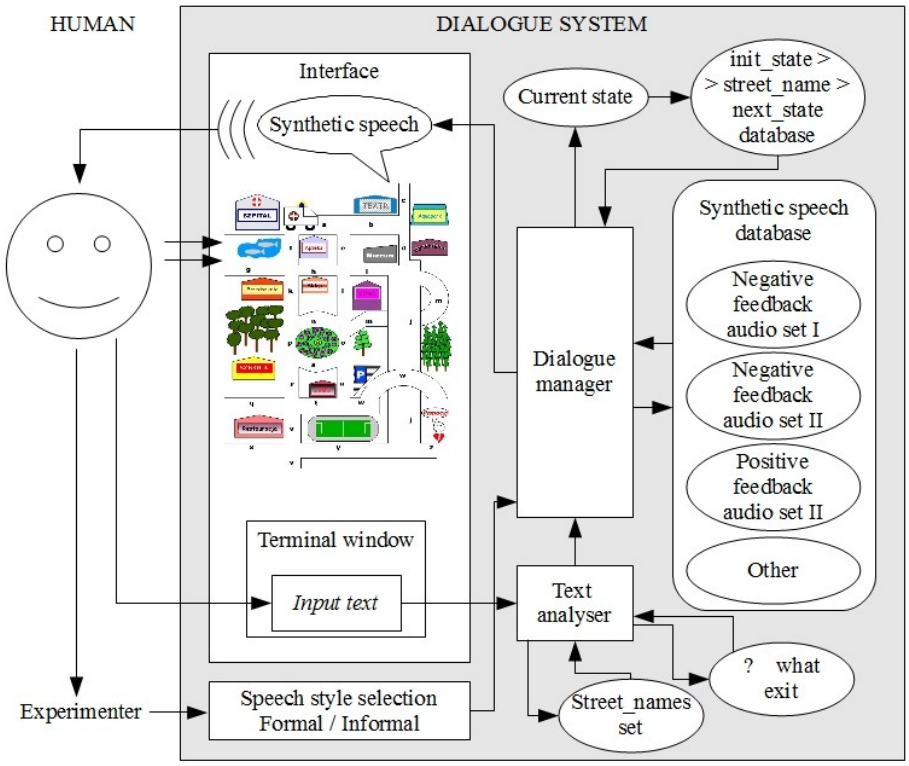


Figure 58: Dialogue system architecture

8.4 Implementation

The demonstration dialogue system was implemented as a command line application. The written input from the human user is entered on the command line and the output from the system comes from the loudspeakers and it is synthetic speech. In two special cases the system produces text. These two cases are introduced later in this section.

The user needs to insert street names which do not violate the street arrangement on the map to make the system move from one state to the next state. Additionally, the user has the following options:

4. input "*what*" – repeats the last audio output
5. input "*?*"– the current state at which the system (the ambulance) is printed as text and a special map with transition states is displayed
6. input "*exit*" – exits the dialogue loop and moves to the farewell section of the program

The pseudo-code of the implementation of the algorithm is presented in Figure 61. The dialogue manager (DM) finite state automaton is presented in Figure 59. The figure presents a semi-coupled DM automaton of speech act categories in which the automaton on the top is the dialogue system automaton and the automaton at the bottom is the expected human caller input automaton.

The transitions between the interlocutors, i.e. the turn change, is marked with dashed arrows. The red arrows (also loops) show the dialogue flow where the caller input does not match the expected input. For example, the dialogue system will not move to the request dialogue act until it does not get the positive feedback from the caller that he knows where the hospital is. The DM automaton with example utterances is presented in Figure 60. In general, there is no turning back from one state to the other on the individual automata, i.e. caller and dialogue sequence automata. The local loops are implemented, but no backward arrows are designed.. However, when working together, the turning backs are expected. These backward transitions are from DS:q6 to HU:q3 and HU:q4 to DS:q5. At this part of the dialogue, the DM automaton is connected with the MT automaton presented in Figure 57. As long as the caller moves through the dialogue, he is also moving through the map.

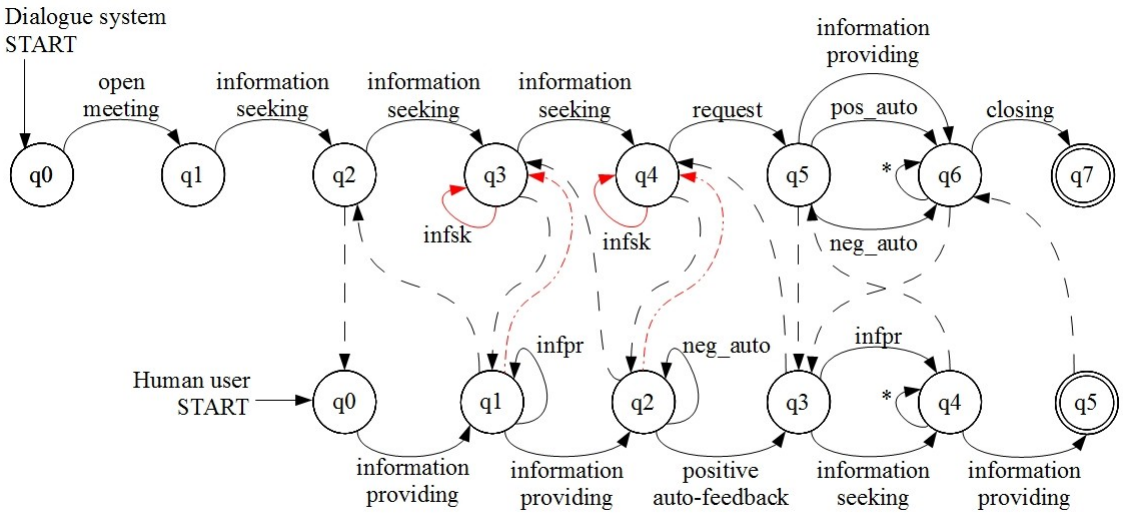


Figure 59: Dialogue manager automaton with dialogue acts

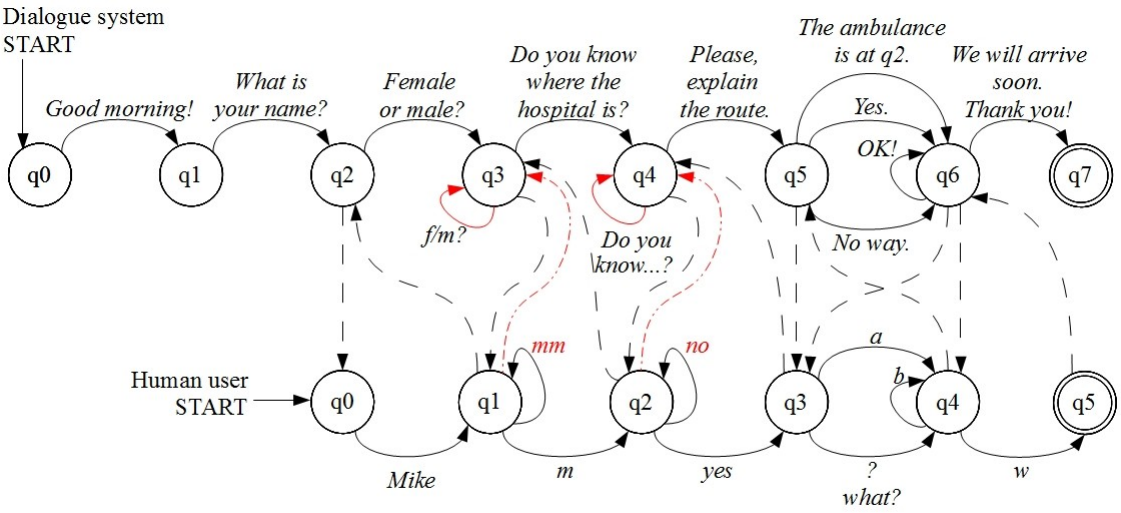


Figure 60: Dialogue manager automaton with exemplar utterances

As already noted, the dialogue is divided into three sections: opening, direction description in the while-loop and the farewell.

In the opening section, the system greets the caller, asks for name and sex. Then using the Polish honorific form addressed to a lady or a gentleman, the system asks whether it is clear where the hospital is and if the answer is "yes", it asks to explain the route. At the opening dialogue section, any text can be inserted for the name, then either "f" or "m" is suggested on screen for specifying the user's sex. The reply "yes" is expected to the question "Do you know where the hospital is?". After "yes" is inserted, the system asks for describing the route the ambulance needs to take from the hospital to the person with the heart attack.

The opening section is designed in highly structured way. First of all, after "Good morning" there is quite a long pause before the next utterance to see the caller's reaction to the spoken system's output. Although the users are informed that they communicate with the computer via text, the experimenter wants to see whether the users will want to use speech to communicate with the system. Second, the question is asked: "female or male" and a text is printed on the screen "f/m". This text is added to see whether the users will understand the textual suggestion and will write either "f" or "m" to specify their sex, or will rather try to write the full words "female" or "male". The last opening question is the yes/no question: "Do you know where the hospital is?" The simplest answer is "yes" or

"no", but no textual suggestion is given, because the experimenter wants to see what kind of answers are to be input by the callers to such question. After the yes/no question is answered positively, the system produces a request to the caller to explain the route.

The negotiation of the route is processed by one while-loop. The caller inputs the name of the streets or any function words in order to carry out the task and direct the system to drive its ambulance from the hospital to the ill person. If the suggested street name was correct and the system made a move on its map, positive feedback is produced. If the suggested street is impassable or it violates the street arrangement on the map (e.g. jump from A to C is impossible; to move from A to C first the move from A to B needs to be made and then the move from B to C) negative feedback from audio set II is produced. When the text input is neither the street name nor the function word, it is recognised as non-existent street and negative feedback from audio set I is produced. When the function word "what" (Polish: "co") is inserted, then the system's last utterance is repeated. If the function word "?" is input, then on the screen the state at which the ambulance is is printed (e.g. "The ambulance is at state: q4") and a map with states is displayed.

After the system gets to the state q13, the farewell is produced. The system promises that the emergency service will do everything they can to arrive quickly and informs that if they have any trouble to get to the place, they will call again.

At any point, the function word "exit" can be input in order to exit the program. At the end section, the inputs provided by the caller are printed to the TXT log file. In the log file the information about the name, sex, the users path as well as other inputs are saved, together with the information about the dialogue start and end times. Additionally, each run of the program is timed using the built-in Unix function "*time*" which measures the exact run of the program. This duration time is inserted manually into the log file by the experimenter after each run of the program. The structure of an entry of the program run printed into the log file is:

Duration: 2m42.126s	duration time of the program run measured with Unix time function (inserted manually)
a e i i l l l n p s u w	user's path
----------	divisional line
2011-04-19 11:40:38.690220 Mirek mm m tak	start time and inputs from the opening section (name, male/female, readiness yes/no)
----------	divisional line
a e i m j i l l l o nn n p r s u w	user's inputs from the while-loop section
----------	divisional line
21	number of all inputs
2011-04-19 11:43:20.796291	end time
##########	divisional line from the next user's entry

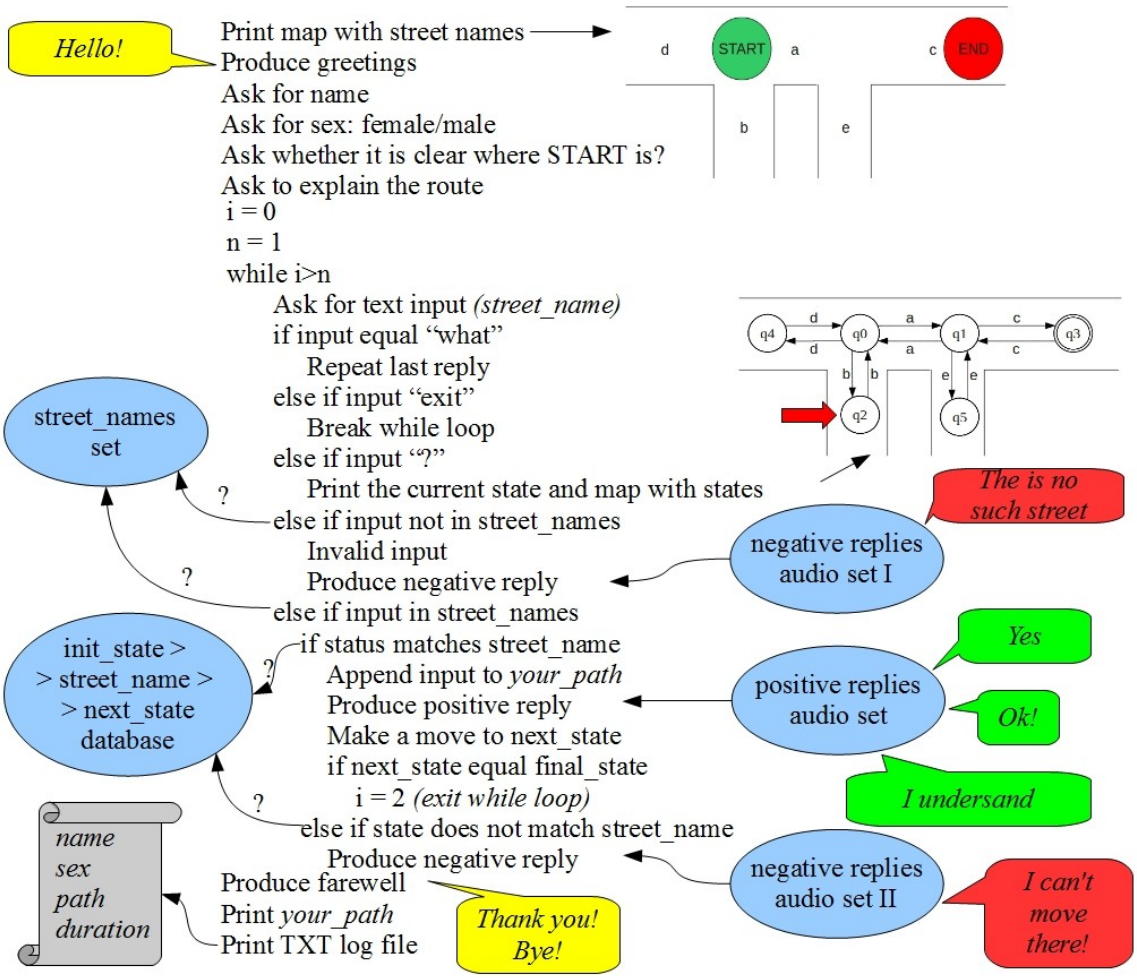


Figure 61: Visualisation of the implementation of the dialogue system main algorithm

The dialogue system has two speech styles: formal and informal. These speech styles were implemented into formal and informal versions of the system. Depending on user's characteristics, e.g. cool look, sporty clothes, tattoos, the human experimenter chooses the informal speech style for the dialogue. This style is also chosen randomly for dialogues with people with no special characteristics. The formal speech style is chosen when the

user looks quite strict or behaves very formally. Also, this speech style is chosen randomly for dialogues with people who does not reveal such characteristics, but behave as ordinary students.

The utterances used in formal and informal speech styles were selected from the exemplar emergency dialogue described in the previous chapter. The analysis of the dialogue corpus, made it possible to differentiate between the formal and informal utterances in this kind of a dialogue.

8.4.1 Implemented utterances

The observations made on the alignment phenomenon between the speakers in the dialogue corpus, especially results of the communication in the emergency scenario, made it possible to select utterances which will match best the requirements of the aligned or cooperatively non-aligned dialogue system. Those requirements (cf. Grice 1975) are:

1. informativeness
2. relevance
3. briefness
4. politeness.

The first three requirements correspond to Gricean Maxims of Cooperation. The fourth requirement results from the observations of how the dialogues in the dialogue corpus recordings were carried out.

The original utterances from the exemplar dialogue were annotated on the phone level and then synthesised using the automatic close copy speech (ACCS) synthesis (Bachan 2007a) with the PL2 male voice. The selected utterances were then left untouched or were slightly modified in order to meet formal and informal criteria. Some utterances were invented by the author to match the dialogue scenario. All the utterances were synthesised using MBROLA (Dutoit et al. 1996) and exported to the WAV audio format and integrated with the dialogue system. All the feedback utterances are chosen randomly from the audio sets at a runtime of the dialogue system. The other utterances at the dialogue opening and closure are generated according to the scenario. All the utterances available to the system are presented in Table 55.

Table 55: Informal and formal utterances and their English translations available to the dialogue system

	Formal		***Informal***	
	Polish	***English***	***Polish***	***English***
Introduction / Farewell	Dzień dobry!	Good morning!	Dzień dobry! Witam!	Good morning! Welcome!
	Jak się nazywasz?	What is your name?	Jak się nazywasz?	What is your name?
	Kobieta czy mężczyzna?	Female or male?	Kobieta czy mężczyzna?	Female or male?
	Wie Pani gdzie jest szpital?	Do you know where the hospital is? (To Mrs)	A wie Pani gdzie jest szpital?	And do you know where the hospital is? (To Mrs)
	To dobrze. To jeżeli mogłaby Pani wyjaśnić dorgę, to bardzo bym prosił.	So good. So if you could explain the route, I would be very grateful. (To Mrs)	No to dobrze. To jeżeli mogłaby Pani wyjaśnić drogę, to bardzo bym prosił.	So good. So if you could explain the route, I would be very grateful. (To Mrs)
	Wie Pan gdzie jest szpital?	Do you know where hospital is? (To Mr)	A wie Pan gdzie jest szpital?	And do you know where hospital is? (To Mr)
	To dobrze. To jeżeli mogłby Pan wyjaśnić dorgę, to bardzo bym prosił.	So good. So if you could explain the route, I would be very grateful. (To Mr)	No to dobrze. To jeżeli mogłby Pan wyjaśnić drogę, to bardzo bym prosił.	So good. So if you could explain the route, I would be very grateful. (To Mr)
	Postaramy się. W razie czego będziemy jeszcze dzwonić. Proszę czekać. Niedługo przyjeżdżamy. Dziękuję.	We will try. If necessary, we will call you. Please, wait. We will arrive soon. Thank you.	No postaramy się. W razie czego będziemy jeszcze dzwonić. Także... Proszę czekać. Niedługo przyjeżdżamy. Dziękuję.	So we will try. If necessary, we will call you. So... Please, wait. We will arrive soon. Thank you.
Positive feedback	Oczywiście.	Of course.	[yyy] tak.	[yyy] yes.
	Dobrze.	Good.	No dobrze.	So good.
	Dobrze, dobrze, tak, rozumiem.	Good, good, yes, I understand.	Ja rozumiem, że [yyy] prosto.	I understand, that [yyy] straight.
	Dobrze, tak.	Good, yes.	Acha, acha.	Aha, aha.
	Tak.	Yes.	Acha.	Aha.
	Tak, rozumiem.	Yes, I understand.	Tak, tak, rozumiem.	Yes, yes, I understand.
	Tak, następnie.	Yes, next.	Tak, okej.	Yes, okay.
	Prosto.	Straight.	Okej.	Okay.
	A potem?	And then?	[yyy] dobrze, wobec tego... dalej.	[yyy] good, so in this case... next.
	I wtedy?	And then?	Acha, czyli rozumiem muszę przejechać prosto.	Aha, so I understand that I need to go straight.
	Dobrze, dobrze, oczywiście.	Good, good, of course.	Acha, acha, acha, dobrze.	Aha, aha, aha, good.
			Dobra.	Good.
			Dobra, okej.	Good, okay.
			A potem?	And then?
			I wtedy?	And then?
			Dobrze, dobrze, tak, rozumiem.	Good, good, yes, I understand.
			Tak, rozumiem.	Yes, I understand.
			Dobrze, dobrze, oczywiście.	Good, good, of course.
			Tak.	Yes.
			Tak, następnie.	Yes, next.
			Dobrze, tak.	Good, yes.
	Nie rozumiem.	I don't understand.	A czyli...? Nie rozumiem.	And so...? I don't understand.
	Z mojej mapy wynika, że nie ma takiej ulicy.	According to my map, there is no such street.	[yyy] Mam tutaj problem – nie ma takiej ulicy.	[yyy] I have a problem here – there is no such street.
	Nie znam tej nazwy ulicy.	I don't know this street name.	Nie nie nie nie nie nie nie. Nie ma takiej ulicy.	No no no no no no no. There is no such street.

	Formal		*Informal*	
	Polish	***English***	***Polish***	***English***
Negative Feedback: No Street (set I)	Jeszcze raz, proszę.	One more time, please.	Jeszcze raz.	Again.
			[yyy] jeszcze raz, proszę.	[yyy] One more time, please.
			[yyy] nie znam tej nazwy ulicy.	[yyy] I don't know this street name.
			[yyy] No z mojej mapy wynika, że nie ma takiej ulicy.	[yyy] so according to my map, there is no such street.
Negative Feedback: Wrong Way (set II)	Tam właśnie nie mogę.	I can't there, actually.	No chyba coś się musiało zmienić. Nie mogę tam.	So I guess something must have changed. I can't there.
	Z mojej mapy wynika, że nie mogę.	According to my map, I can't there.	No tam właśnie nie mogę.	So I can't there, actually.
	Coś się musiało zmienić. Nie mogę tam.	Something must have changed. I can't there.	Ojej! Z mojej mapy wynika, że nie mogę.	Oops! According to my map, I can't there.
	Nie, nie ma właśnie wejścia tam.	No, there is, actually, no entrance there.	[e-e] Nie, nie ma właśnie wejścia tam.	No, there is, actually, no entrance there.

8.5 Evaluation

The dialogue system underwent thorough evaluation according to the EAGLES standards (Gibbon et al. 2000). First, the author performed diagnostic evaluation to check whether the system runs without failures. It was checked whether the audio outputs were generated correctly, whether the function words did their jobs and whether the user's inputs were printed correctly into the TXT log file.

Having underwent successfully the diagnostic evaluation, the dialogue system faced functional testing and judgement testing with the human users. The setting of the evaluation is presented on Figure 62. 52 people took part in the evaluation whose general data is presented in Table 56. In the evaluation, mainly young students took part around between 19 and 23 years old. However, there were also a few older people in their late 20's and one 52-year-old man. This man was asked to take part in the evaluation because his voice was used for the pl2 creation and therefore the synthesis of the utterances of the dialogue system.

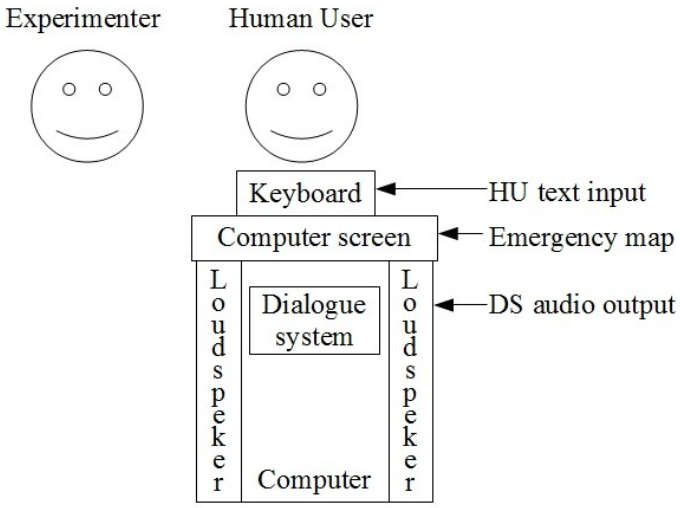


Figure 62: Dialogue system evaluation setting

Table 56: General data of people who participated in the dialogue system evaluation

	Formal				***Informal***			
	N	***Age***			***N***	***Age***		
		Average	***Min***	***Max***		***Average***	***Min***	***Max***
Female	14	22	19	26	14	20	19	23
Male	12	22	20	27	12	26	20	52
Overall	26	22	19	27	26	23	19	52

The functional testing was based on evaluation of the dialogue between the human and the computer. The dialogue model was tested, the scenario and the successfulness of communication by the means of the actual conversation between the test participant and the computer. After the dialogue was finished, the test participant was asked to assess different domains of the system on the 5-point rating scale, where 1 was the lower grade and 5 was the highest grade. The system's 4 domains were evaluated: speech style selection module, speech synthesis, dialogue manager and system design. However, the test participants were asked to evaluate 7 categories: friendliness, speech quality, speech intelligibility, dialogue, dialogue naturalness, system attractiveness and ease of usage. The correspondence between the assessed 7 categories to the dialogue system domains is presented in Table 57.

Table 57: Questionnaire of assessment of 7 areas of the dialogue system and their correspondence to the dialogue system domains

Assessments	***Dialogue system domain***
friendliness	speech style selection module
speech quality	speech synthesis
speech intelligibility	speech synthesis
dialogue	dialogue manager
dialogue naturalness	dialogue manager, speech style selection module
system attractiveness	system design
ease of usage	system design

An exemplar course of the dialogue is presented in Table 58. The dialogue was reconstructed based on the TXT log file entry:

```
Duration: 2m32.093s
f k p s u w
--------------------------
2011-04-19 11:32:41.235151 Anna k tak
--------------------------
f k p r ? r co co co ? s co uu u w
--------------------------
18
11-04-19 11:35
##########################
```

Only the user's inputs are saved in the log file therefore the actual dialogue system outputs can only be guessed. The opening utterances are always the same, but the positive and negative feedback utterances are chosen randomly from different sets, so for the dialogue reconstruction they might not match exactly the utterances which were spoken by the system at its runtime.

In the exemplar dialogue the user entered her personal data and then moved smoothly to the giving direction part of the dialogue. She tried stubbornly to enter the r street, but the system refused. The user asked two times for clarification of whether the ambulance is ("?") and had great difficulties to understand one utterance as it asked three times in a row for repetition ("what"). However, in the end the communication turned out to be successful and the user was informed that the ambulance should arrive soon.

Table 58: Dialogue reconstruction based on one log file entry for informal speech style

	Dialogue system	***User***
1	Good morning! Welcome!	–
2	What is your name?	Anna
3	Female or male? *Printed text:* `m / f`)	f
4	And do you know where the hospital is? (To Mrs)	yes
5	So good. So if you could explain the route, I would be very grateful. (To Mrs)	f
6	Yes, okay.	k
7	Aha, so I understand that I need to go straight.	p
8	And then?	r
9	So I guess something must have changed. I can't there.	?
10	*Printed text:* `The ambulance is at: q10` (map with states is displayed)	r
11	No, there is, actually, no entrance there.	what
12	No, there is, actually, no entrance there.	what
13	No, there is, actually, no entrance there.	what
14	No, there is, actually, no entrance there.	?
15	*Printed text:* `The ambulance is at: q10` (map with states is displayed)	s
16	Aha, aha, aha, good.	what
17	Aha, aha, aha, good.	u
18	And then?	w
19	Good, good, of course.	–
20	So we will try. If necessary, we will call you. So... Please, wait. We will arrive soon. Thank you!	–

8.6 Results

In Table 59 the basic statistics of the functional testing are presented. As stated above, in the test 14 females and 12 males took part to evaluate each of the two scenarios: formal and informal. Altogether 52 people took part in the evaluation. The duration time of all the dialogues in formal and informal scenarios lasted about 75min 34sec and 76min 48sec respectively. The max and min duration times were very similar, but in both cases, the duration times for the formal scenario were longer by a few seconds. However, the mean duration time turned out to be shorter for the formal scenario by 2seconds. The number of inputs inserted during one dialogue is almost the same and equals 20.54 inputs for the formal and 20.26 inputs for informal scenarios. The same equality applies to the average length of the path and the number of inputs of the "what" function word. The "?" word

appeared much more frequent in dialogue with the informal speech styles and suggest that the informal speech was less intelligible.

The basic statistics of the functional testing show that both, the formal and informal speech styles provided similar conversational circumstances to the dialogue participants, with the tendency of the informal speech to be less intelligible.

Table 59: Basic statistics of functional testing of the dialogue system

	Formal					***Informal***				
	Time	***Inputs***	***Path***	***what***	***?***	***Time***	***Inputs***	***Path***	***what***	***?***
Overall	75min 34s	534	264	26	7	76min 48s	538	248	23	15
Max	4min 47s	30	14	5	3	4min 21s	33	14	3	4
Min	2min 1s	12	6	0	0	1min 51s	10	6	0	0
Mean	2min 59s	20.54	10.15	1.00	0.26	3min 1s	20.69	9.54	0.88	0.57
									Altogether	**152min 22s**
									All inputs	**1072**

All the subjects accomplished the communication with the computer successfully, which means that they were aligned at least on the essential semantic level. According to the semantic level interpretation adopted for the map-task scenario, the human user as well as the computer, i.e. the dialogue system must have had the same or at least a compatible semantic representation of the map to move along the map and assure success of communication by getting to the end point. Misalignments happened, but the dialogue system effectively recovered from misalignments.

The results of the judgement testing are presented in Table 60. Overall the system received very high scores: 4.11 for the formal dialogue scenario and 4.30 for the informal scenario, where 5 was the highest grade. Looking at the separate categories and the test groups more details can be said. The difference in male and female assessment is revealed in scenarios assessments. On average, males gave higher scores to the formal scenario in 6 categories, whereas females assessed higher than male the informal scenario in 4 categories. However, overall, males gave higher scores (4.25) than the females (4.15). Speech quality and dialogue naturalness received the lowest scores when the formal speech style was applied. When the informal speech style was used, the lowest assessments were assigned to the speech quality and the speech intelligibility. In both scenarios, friendliness

received the highest scores, where friendliness of the informal speech style was evaluated higher by 0.16 point than in the formal speech style. This shows that an "imperfect" system with hesitations, filled-pauses and colloquial expression seems more friendly to the users than a system with "perfect" speech. Also naturalness of such "imperfect" system is evaluated higher, although these imperfections affect speech intelligibility. What is surprising is the fact that the assessment of speech quality does not go hand in hand with the assessment of speech intelligibility. In the informal scenario, the speech quality was assessed higher than the speech intelligibility. It was noticed that the test participants had problems understanding the filled-pauses, which are unwittingly produced by the humans. Also, such negation as "e-e" also surprised the users who had problems understanding it. On the other hand, the formal speech scenario received higher scores in the speech intelligibility category, but speech quality was assessed by 0.46 point lower on average. This means that although users had less problem understanding the formal speech, they were not less pleased with their quality than the imperfect speech used in the informal scenario.

The system attractiveness and ease of usage were assessed high which means that the system design was correct and did not have negative influence on the human-computer communication.

The assessment results show that the speech styles were designed correctly and higher results of the informal scenario show that this style aligns more with the young dialogue system users.

The general observation made during the evaluation of the system with the human users is that the young people were very enthusiastic about the communication with the computer system. Although a little bit uncertain about the task, they felt comfortable because the human experimenter were always present and ready to help with the system.

Although the test participants were not informed about the fact that their dialogue is timed, if they encountered problems getting to the final point, they were making comments that the patient would die if that task was real. This shows that the users themselves understood the importance of the task and wanted to finish the task as quickly as possible.

Table 60: Results of the judgement testing of the dialogue system in 7 categories. Numbers in brackets stand for average assessment across the 7 categories and 2 scenarios for females (F), males (M) and overall (All)

		Formal							***Informal***						
		Friendliness	***Speech quality***	***Speech intelligibility***	***Dialogue***	***Dialogue naturalness***	***System attractiveness***	***Ease of usage***	***Friendliness***	***Speech quality***	***Speech intelligibility***	***Dialogue***	***Dialogue naturalness***	***System attractiveness***	***Ease of usage***
F (4.15)	***Mean***	4.64	3.50	4.21	4.07	3.43	4	4.07	4.71	4.29	4.14	4.21	4.21	4.50	4.14
	Min	3	3	3	3	2	2	3	4	3	3	3	3	3	3
	Max	5	5	5	5	5	5	5	5	5	5	5	5	5	5
M (4.25)	***Mean***	4.25	4.08	4.25	4.42	4.08	4.33	4.33	4.50	4.00	4.00	4.25	4.25	4.42	4.33
	Min	3	3	3	3	2	3	3	3	3	3	3	3	3	3
	Max	5	5	5	5	5	5	5	5	5	5	5	5	5	5
All (4.20)	***Mean***	4.46	3.77	4.23	4.23	3.73	4.15	4.19	4.62	4.15	4.08	4.23	4.23	4.46	4.23
							Overall:	**4.11**						***Overall:***	**4.30**

The users' reaction to the opening section question also brought interesting results. Many people after hearing "Good morning" replied by saying "Good morning" or asked the experimenter whether they should say "hello". Only very few did not say anything and waited for the next utterance. This shows that the natural way of communication is speech, and not a combination of text and speech.

In the question about sex, the textual suggestion "f/m" had a great impact on the users and nearly no one after seeing that text tried to write the full words for the male and the female sexes. This shows that redundancy in the human-computer interaction does not have to be avoided, but redundant information may help in communication.

Finally, the yes/no question "Do you know where the hospital is?" apart from the answer "yes", the users input "I know" (Polish: "wiem") or asked the experimenter whether it would be fine to write only "y" for "yes" as it was suggested in the previous answer.

The very interesting communication phenomenon was achieved unplanned and had a great impact on the human-computer interaction. The design of the dialogue system required the program sox to play the synthesised utterances. However, sox, after finishing

playing the audio file, takes a few seconds to finish processing. The result is that, although sox finishes playing the audio file, the command line accepts the input text, but the text is not shown on the screen. This results in the communication problem, because although the command line seems to be empty, it accepted "invisible" text if it was inserted by the user immediately after the system's utterance finished. For example, the user thinks the input is "p" because this is on his screen, however the actual input is "pp" because the user pressed "p" for the fist time when sox was still working. This problem tried to be solved by printing a line of dashes ("------") on the screen immediately after sox stopped. The test participants were instructed:

> "The system will be ready for your answer when on the screen appears ----------"

However, many people were impatient and did not wait for the line of dashes to appear on the screen. This resulted in communication problems and also incorrect system performance which instead of one utterance could produce two utterances, because of double text insertion without realising it (double press on ENTER before the line of dashes was printed onto the screen). This inconvenience of waiting for the line of dashes was first interpreted by the experimenter as a nuisance, but later it indicated how important the alignment of the interlocutors is. The users who from the beginning applied the insertion of their inputs to the pace of the computer, i.e. appearance of the line of dashes on the screen, finished the task quicker because no misunderstanding appeared. People who rushed and started to have communication problems sooner or later understood the importance of line of dashes and adapted their pace to the computer's pace. Only a handful of users had to be stopped by the experimenter to wait for the appearance of the line of dashes till the task was accomplished.

This unplanned 'nuisance' showed how important alignment is in communication. Not only the dialogue system needs to align its speech with the user, but also, the user needs to align with the system requirements to carry out successful communication. An additional observation is that the dialogue system has not only to take into account consecutive input, but also input overlapping with system's output as it was presented on Figure 31 and Figure 33. The kind of alignment discussed here goes beyond semantic alignment, and involves pragmatic alignment.

8.7 Summary

The analysis and conclusions drawn from the dialogue corpus presented in Chapter 7 served to design a demonstration dialogue system prototype which was created to test the the main claim of the thesis. In this dialogue system, not only a finite state automaton for dialogue control was implemented, following the concepts presented in Chapter 5, but also a finite state automaton was built to control the moves along the map. These two finite state automata worked together to enable communication and success with the task.

The laboratory setting designed for the human-computer interaction ensured alignment of the semantic level which resulted in successful performance of the communication task. The human users carried in their minds the same semantic representation (or a compatible semantic representation) of the map, and therefore could move along the map until the end point. Any misalignment problems were reported to the system, and the recovery technique implemented in the system made it able to recover successfully from misalignment.

The choice of utterances synthesised with the PL2 voice created in Chapter 6 turned out to be correct and served for developing two speech styles in the dialogue system: formal and informal. The synthetic speech of the dialogue system was assessed by the human users and very high scores show that both communicativeness and cooperativeness of the synthetic speech was assured in the dialogue system.

The overall design of the dialogue system was evaluated in subjective tests with very positive results. This means that simple solutions with command line input do not need to be avoided and still may look attractive to young users, perhaps because text systems appear to be familiar to these users, since text systems look like the chat applications which are frequently used on the internet. The present dialogue system has the advantage that it has a text-in-voice-out and not simply a text-in-text-out interface.

Chapter 9: Summary and conclusions

The main thesis of the thesis, as presented in the first chapter, is:

> Alignment of semantic representations is essential for successful communication in a dialogue.

The validity of this thesis was demonstrated. An emergency scenario and a map-task dialogue were chosen as an example and alignment of semantic representations of the map was claimed to be essential for successful communication.

In order to test the thesis, a dialogue system was developed which would conform with modern dialogue theory on alignment between interlocutors. The operational goal was to develop a proof-of-concept dialogue system together and the methodology of the thesis in a simulated stressful emergency scenario. The dialogue analysis was performed on two dialogue corpora.

In the thesis linguistic specifications were outlined and their implications for the spoken dialogue system development were discussed. It was stated that general function of alignment is coordination between interlocutors in order to achieve successful outcome of communication.

A thorough pilot study was carried out in order to apply the theoretical principles to the actual dialogue material and provisional automaton models of the dialogue were developed. A dialogue annotation scheme was proposed, following the DIT++ taxonomy (Bunt 2000), but not so detailed and including more general categories and leaving out many detailed ones which were not relevant for the present purpose. Instead of increasing the number of dialogue acts in the taxonomy to cover the present scenario, a separate semantic task manager for map traversal was designed, which was controlled by the dialogue act manager.

For the demonstration dialogue system, a Polish male synthetic voice (PL2) was created using speech material and automatic tools were developed within this work. PL2 voice was created with the aim of implementing it in the speech synthesis module of the

demonstration dialogue system. Several available text and speech corpora were searched for diphone extraction, but the results were not satisfying, therefore new speech material was recorded and annotated for the need of PL2 creation. Automatic tools such as phonetically rich diphone extractor and automatic diphone extractors were developed (Bachan 2010) . Finally, the synthetic voice was positively evaluated in perception tests by human subjects.

The new corpus linguistic study on alignment was carried on a dialogue corpus recorded for this purpose. This corpus was recorded because the dialogue material used in the pilot study did not provide enough information for dialogue modelling needed for the demonstration system. Four different dialogue scenarios were arranged and prompt speech material (maps for the map-tasks and the emergency diapix) was created for three of these scenarios. A dialogue corpus was recorded taking into account the public character of conversations in the emergency setting. The linguistic study of alignment in this kind of dialogue made it possible to design a dialogue scenario needed for the demonstration dialogue system.

Finally, a prototype dialogue system was developed and evaluated with human users. The prototype dialogue system combined text input with speech output and its core was based on two linked finite state automata: one for the dialogue manager and one for map traversal. The laboratory setting of the evaluation task demonstrated alignment of the semantic representation of the map, as all the human users finished the task successfully.

Additionally, the alignment of the dialogue system was based on speech style selections: formal and informal. The speech style selection demonstrated the claim that the traditional emotion label sets used in general speech synthesis may be replaced by the speech *style* in the dialogue systems. Such a system not only takes into account the communication in public situations, but also aligns with the user on the same levels as the human would do, so not emotional, but semantic, syntactic and pragmatic.

The synthetic speech of the prototype dialogue system was aligned on the speech style level, however it was not designed to align on lexical or phonetic levels. This is the task of future research and development, in order to develop a system with a more advanced text parser which can process longer and more complex text input. The alignment on the

phonetic level seems impossible at the moment as such a system would not only have to take spontaneous speech as input, but also understand it (by the means of a speech recogniser) and analyse its phonetic properties at a runtime, an,d based on this information, produce phonetically aligned synthetic speech output. Building a more complex speech synthesis module to align with the input lexical items at a runtime, however, is planned for future research.

Bibliography

Aijmer, K. 1996. *Conversational routines in English: Convention and creativity*. London: Longman

Allen, M.L, Haywood, S., Rajendran, G & Branigan, H. 2011. Evidence for syntactic alignment in children with autism. In: *Developmental Science 14:3*, pp. 540–548

Allwood, J., Traum, D., & Jokinen, K. 2000. Cooperation, Dialogue and Ethics. *International Journal of Human Computer Studies,* 53, pp. 871-914

Asu, E. L. & Nolan, F. 2005. Estonian rhythm and the Pairwise Variability Index. Proc. Fonetik 2005, Gothenburg, pp. 29-32

Austin, J.L. 1962. How to Do Things With Words. Oxford: Clarendon Press

Bachan, J. 2007a. Automatic Close Copy Speech Synthesis. In: *Speech and Language Technology. Vol. 9/10*. Ed. Grażyna Demenko. Poznań: Polish Phonetic Association. 2006/2007, pp. 107-121

Bachan, J. 2007b. *Close Copy Speech Synthesis for Perception Testing and Annotation Validation*. M.A. Thesis. Institute of Linguistics. Adam Mickiewicz University

Bachan, J.. 2010. Efficient diphone database creation for MBROLA, a multilingual speech synthesiser. In: *XII International PhD Workshop (OWD 2010)*. *Conference Archives PTETiS, Vol. 28*. 23-26 October 2010, pp. 303-308

Bachan, J. & Surmanowicz, B. 2008. Preliminary results of expressive speech synthesis in Polish. In: G. Demenko, K. Jassem, S. Szpakowicz (Eds.) *Speech and Language Technology, Vol 11*. Poznań: Polish Phonetic Association, pp. 103-112

Baker, R. & Hazan, V. 2009. Acoustic-phonetic characteristics of naturally-elicited clear speech in British English. (A) In: *J. Acoust. Soc. Am.*, 125, p. 2729

Batliner, A. Fischer, K., Huber, R., Spilker, J. & Nöth, E 2003. How to find trouble in communication. In: *Speech Communication 40 (1-2)*, pp. 117-143

Batliner, A., Steidl, S., Hacker, Ch. & Nöth, E. 2008. Private emotions versus social interaction: a data-driven approach towards analysing emotion in speech. In: *User Modelling and User-Adapted Interaction - The Journal of Personalization Research 18*, pp. 175-206

Boersma, P. & Weenink, D. 2001. PRAAT, a system for doing phonetics by computer. Glot International 5(9/10), pp. 341-345

BOSS, the Bonn Open Synthesis System. <http://www.ikp.uni-bonn.de/forschung/phonetik/sprachsynthese/boss/>, accessed on 2010-09-19

Bradlow, A. R., Baker, R. E., Choi, A., Kim, M. and van Engen, K. J. 2007. The Wildcat Corpus of Native- and Foreign-Accented English. In: *Journal of the Acoustical Society of America*, 121(5), Pt.2, p. 3072

Branigan, H. 2009. Identifying the causes of linguistic alignment in dialogue. Presentation at Bielefeld University, 2009-01-26

Branigan, H.P., Pickering, M.J. & Cleland, A.A. 2000. Syntactic coordination in dialogue. In: *Cognition, 75*, pp. B13-B25

Brennan, S.E. & Clark, H.H. 1996. Conceptual pacts and lexical choice in conversation. In: *Journal of Experimental Psychology: Learning, Memory, and Cognition, 22*, pp. 1482-1493

Brown-Schmidt, S., Campana, E. & Tanenhaus, M.K. 2005. Real-time reference resolution by naive participants during a task-based unscripted conversation. In J.C. Trueswell & M.K. Tanenhaus (eds.). *Approaches to studying world-situated language use: Bridging the language as product and language as action traditions* (MIT Press)

Bunt, H. 2000. Dialogue pragmatics and context specification. In: H. Bunt & W. Black, (Eds.) *Abduction, Belief and Context in Dialogue. Studies in Computational Pragmatics*. John Benjamins, Amsterdam, pp. 81–150

Bunt, H. DIT++ Taxonomy of Dialogue Acts. 2008. (Release 3, version 2, February 8, 2008) <https://let.uvt.nl/general/people/bunt/docs/dit-schema3-2.html>, accessed on 2009-10-15.

Bunt, H. DIT++ Taxonomy of Dialogue Acts. (Release 5, May 2010) <http://dit.uvt.nl/>, accessed on 2011-30-04

Clark, H.H. 1985. Language and language users. In G. Lindzey & E. Aronson (Eds.), *The Handbook of Social Psychology* (3rd ed.). New York: Harper Row, pp. 179-231

Clark, H.H. 1992. *Arenas of Language Use*. Chicago, IL: University of Chicago Press

Clark, H.H. 1996. *Using language*. Cambridge: Cambridge University Press

Clark, H.H., & Marshall, C.R. 1981. Definite reference and mutual knowledge. In: A.K. Joshi, I.A. Sag, & B.L. Webber (Eds.), *Elements of discourse understanding* (pp. 10-63). Cambridge: Cambridge University Press

Clark, H.H., & Schaefer, E.F. 1987. Concealing one's meaning from overhearers. *Journal of Memory and Language, 26*, pp. 209-225

Clark, H.H., & Wilkes-Gibbs, D. 1986. Referring as a collaborative process. *Cognition, 22*, pp. 1-39

de Saussure, F. 1913. *Cours de linguistique générale*, éd. Payot

Demenko, G., Wypych, M. & Baranowska, E. 2003. Implementation of Grapheme-to-Phoneme Rules and Extended SAMPA Alphabet in Polish Text-to-Speech Synthesis. In: G. Demenko & M. Karpiński (Eds.) *Speech and Language Technology, Vol. 7*. Poznań: Polish Phonetic Association, pp. 79-95

Demenko, G., Grocholewski, S., Wagner, A., Szymanski M. 2006. Prosody annotation for corpus based speech synthesis. In: *Proceedings of the Eleventh Australasian International Conference on Speech Science and Technology*, pp. 460-465. Auckland, New Zealand

Demenko, G., Klessa, K. Szymański, M. & Bachan, J. 2007. The design of Polish speech corpora for speech synthesis in BOSS system. In: *Preceedings of XII Sympozjum Podstawowe Problemy Energoelektroniki, Elektromechaniki i Mechatroniki (PPEEm 2007)*. Wisła, Poland, pp. 253-258

Demenko, G., Grocholewski, S., Klessa, K., Ogórkiewicz, J., Wagner, A. Lange, M., Śledziński, D. & Cylwik, N. 2008. JURISDIC: Polish Speech Database for Taking

Dictation of Legal Texts. In: *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*. 28-30 May 2008, Marrakech, Morocco

Demenko, G. Möbius, B. & Klessa, K. 2008. The design of Polish speech corpus for unit selection speech synthesis. In: G. Demenko, K. Jassem, S. Szpakowicz (Eds.) *Speech and Language Technology, Vol 11*. Poznań: Polish Phonetic Association, 85-10

Dutoit, T. 1997. *An Introduction To Text-To-Speech Synthesis*. Dordrecht: Kluwer Academic Publishers

Dutoit, T., Pagel, V., Pierret, N., Bataille, F. & van der Vrecken O. 1996. The MBROLA Project: Towards a Set of High-Quality Speech Synthesizers Free of Use for Non-Commercial Purposes. In: *Proceedings of ICSLP 96, Vol. 3*. Philadelphia, pp.1393-1396

García Márquez, G. 1967. *Cien años de soledad.* Polish: *Sto lat samotności*. Translated by Grażyna Grudzińska. Warszawskie Wydawnictwo Literackie Muza SA. Warszawa 2004

Garrod, S., & Anderson, A. 1987. Saying what you mean in dialogue: A study in conceptual and semantic co-ordination. *Cognition, 27*, 181-218

Gibbon, D. 1981. Idiomaticity and functional variation. A case study of international amateur radio talk. In: *Language and Society 10*, 1981:21-42

Gibbon, D. 1985. Context and variation in two-way radio discourse. In: Charles A. Ferguson (ed.). *Discourse Processes 8*, 4:391-420

Gibbon, D. 2009. Gesture theory is linguistics: modelling multimodality as prosody. *Proceedings of PACLIC 23 Conference*, Hong Kong.

Gibbon, D., Moore, R. & Winski, R. 1997. *Handbook of Standards and Resources for Spoken Language Systems*. Berlin: Mouton de Gruyter

Gibbon, D., Mertins, I. & Moore, R. 2000. *Handbook of Multimodal and Spoken Dialogue Systems: Terminology, Resources and Product Evaluation*. New York: Kluwer Academic Publishers

Giles, H., Coupland, N., & Coupland, J. 1992. Accomodation theory: Communication, context and consequences. In H. Giles, J. Coupland, & N. Coupland (Eds.), *Contexts of accommodation* (pp. 1-68). Cambridge: Cambridge University Press

Giles, H., & Powesland, P. F. 1975. *Speech styles and social evaluation*. New York: Academic Press

Ginzburg, J. 2001. Fragmenting Meaning: Clarification Ellipsis and Nominal Anaphora. In: H. Bunt (Ed.) *Computing meaning 2: Current issues in computational semantics*. Dordrecht: Kluwer, pp. 247-270

Grice, H.P. 1975. Logic and conversation. In: Cole, P. & Morgan, J. (Eds.) *Syntax and Semantics, Vol. 3*. New York: Academic Press. pp. 41-58

Jassem, W. 2003. Polish. In: *Journal of the International Phonetic Association, Vol. 33*. Cambridge: Cambridge University Press, pp. 103-107

Johnson-Laird, P.N. 1983. *Mental models: Toward a cognitive science of language, inference and consciousness.* Cambridge, MA: Harvard University Press

Johnstone, T. & Scherer, K.R. 1999. The effects of emotions on voice quality. In: *Proceedings of the XIVth International Congress of Phonetic Sciences.*

Karpiński, M. 2002. The Corpus of the Polish Intonational Database (PoInt). In: *Investigationes Linguisticae, Vol. 8*, pp. 24-25, <http://www.staff.amu.edu.pl/~inveling/pdf/maciej_karpinski_inve8.pdf>, accessed on 2010-05-20

Karpiński, M. 2006. S*truktura i intonacja polskiego dialogu zadaniowego*. Poznań: Wydawnictwo Naukowe UAM

Klabbers, E., Stöber, K., Veldhuis, R, Wagner, P. & Breuer, S. 2001. Speech Synthesis Development Made Easy: the Bonn Open Synthesis System. In: *Eurospeech-2001*, pp. 521-525

Levelt, W.J.M. 1992. Accessing words in speech production: Stages, processes and representations. *Cognition 42*, pp. 1-22

Levelt, W.J.M., & Kelter, S. 1982. Surface form and memory in question answering. *Cognitive Psychology, 14*, 78-106

Levelt, W.J.M., & Schriefers, H. 1987. Stages of lexical access. In: G.A.M. Kempen (Ed.), *Natural Language Generation*. Dordrecht: Martinus Nijhoff, pp. 395-404

Lewandowski, N. 2009. Sociolinguistic factors in language proficiency: phonetic convergence as a signature of pronunciation talent. In: Dogil, G., Reiterer, S. M. (Eds.). *Language Talent and Brain Activity*. Berlin, New York: Mouton de Gruyter, pp. 257-278

Murray, I.R. & Arnott, J.L. 2008. Applying an analysis of acted vocal emotions to improve the simulation of synthetic speech. In: *Computer Speech and Language, Vol. 22, Issue 2.* Academic Press Ltd.

Ortony, A. & Turner, T. J. 1990. What's basic about basic emotions? In: *Psychological Review, 97*, pp. 315-331

Pearson, J., Hu, J., Branigan, H.P., Pickering, M.J., & Nass, C. 2006. Adaptive language behavior in HCI: how expectations and beliefs about a system affect users' word choice. In: *Proceedings of the SIGCHI conference on Human Factors in computing systems*. Montréal, April 2006, pp.1177-1180

Pickering, M.J. & Garrod, S. 2004. Toward a mechanistic psychology of dialogue. In: *Behavioral and Brain Sciences, 27*, pp. 169-225

Raux, A. & Eskenazi, M. 2009. A Finite-State Turn-Taking Model for Spoken Dialog Systems. In: *Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the ACL*, Boulder, Colorado, June 2009. pp. 629–637

SAMPA - computer readable phonetic alphabet. Polish. Maintained by J.C. Wells. Created on 1996-09-06. <http://www.phon.ucl.ac.uk/home/sampa/polish.htm>, accessed on 2010-09-20

Schenkein, J. 1980. A taxonomy fo repeating action sequences in natural conversation. In B. Butterworth (Ed.), *Language production Vol. 1*. London: Academic Press, pp. 21-47

Schober, M.F. 1993. Spatial perspective-taking in conversation. *Cognition, 47,* 1-24

Schomaker, L., Nijtmans J., Camurri, A., Lavagetto, F., Morasso, P., Benoît, C., Guiard-Marigny, T., LeGoff, B., Robert-Ribes, J., Adjoudani, A., Defée, I., Münch, S.,

Hartung J. Blauert, K. 1995. A taxonomy of multimodal interaction in the human information processing system. Esprit Project 8579 Miami

Searle, J.R. 1969. *Speech Acts*. London: Cambridge University Press

SmartKom: <http://www.smartkom.org/>, accessed on 2009-07-11

Szklanny, K. & Marasek, K. 2002. PL1 - A Polish female voice for the MBROLA synthesizer. Copying the MBROLA Bin and Databases. <http://tcts.fpms.ac.be/synthesis/mbrola/mbrcopybin.html>, accessed on 2008-05-13

Szymański, M. & Grocholewski, S. 2005. Transcription-based automatic segmentation of speech. In: *Proceedings of 2nd Language and Technology Conference,* Poznań, pp. 11–14

Tannen, D. 1989. *Talking voices: Repetition, dialogue, and imagery in conversational discourse*. Cambridge: Cambridge University Press

van Dijk, T.A., & Kintsch, W. 1983. *Strategies in discourse comprehension*. New York: Academic Press

Vermobil project. <http://verbmobil.dfki.de/overview-us.html>, accessed on 2011-04-04

Vetulani, Z. 2004. *Komunikacja człowieka z maszyną. Komputerowe modelowanie kompetencji językowej*. Akademicka Oficyna Wydawnicza EXIT.

Vetulani, Z., Marciniak, J., Obrębski, T. Vetulani, G., Dąbrowski, A. Kubis, M. Osiński, J., Walkowska, J., Kubacki, P., Witalewski, K. 2010. *Zasoby językowe i technologie przetwarzania tekstu. POLINT-112-SMS jako przykład aplikacji z zakresu bezpieczeństwa publicznego*. Wydawnictwo Nauowe UAM.

Wahlster, W. 2000. (Ed.) *Verbmobil: Foundations of Speech-to-Speech Translation*. Berlin, Heidelberg, New York, Barcelona, Hong Kong, London, Milan, Paris, Singapore, Tokyo: July 2000, Springer

Wahlster, W. 2006. (Ed.) *SmartKom: Foundations of Multimodal Dialogue Systems*. Cognitive Technologies Series, Heidelberg, Germany: Springer

Weizenbaum, J. 1966. ELIZA - A Computer Program For the Study of Natural Language Communication Between Man And Machine. *Communications of the ACM 9(1)*, pp. 36-45

Wikipedia. the free encyclopedia: Dialogue system. <http://en.wikipedia.org/wiki/Dialog_system>, accessed on 2011-04-03

Wilkes-Gibbs, D., & Clark, H. H. 1992. Coordinating beliefs in conversation. *Journal of Memory and Language, 31*, pp. 183-194

Zwaan, R.A., & Radvansky, G.A. 1998. Situation models in language comprehension and memory. *Psychological Bulletin, 123*, 162-185

Software

ActivePerl. 1996-2006 ActiveState Software Inc.

Dutoit, T. 2005. The MBROLA Project. January 4th, 2005. <http://www.tcts.fpms.ac.be/synthesis/mbrola.html>, accessed 2006-10-15

Gibbon, D. 2008. Nondeterministic Finite State Transducer. Version 2008-08-12. <http://wwwhomes.uni-bielefeld.de/gibbon/Forms/Python/FSM/generator.html> , accessed on 2009-10-12

Klessa, W. Annotation Editor. WK Software. 10 September 2006.

Lennes, M. 2003. Praat script – collect_pitch_data_from_files.praat. <http://www.helsinki.fi/~lennes/praat-scripts/public/collect_pitch_data_from_files.praat>, accessed 2006-02-18

MX Skype Recorder v4.3.0 Jan 30 2010. Copyright © 2006-2010

OpenOffice.org Draw. Copyright © 2000-2010 Oracle and/or its affiliates.

Praat – doing phonetics by computer. Copyright © 1992-2011 by Paul Boersma & David Weenink. <www.praat.org>, accessed on 2011-04-04

Python Programming Language. <http://www.python.org/>, accessed on 2010-09-20

Skype. Copyright 2003-2011 Skype Limited.

SoX - Sound eXchange <http://sox.sourceforge.net/>, accessed on 2010-09-20

TimeLeft, Version 3.55. Copyright © 1999-2010 by NesterSoft Inc, <www.nestersoft.com/timeleft>, <http://www.timeleft.info/>, accessed 2011-01-25

Appendix A Dialogue act matrix

An excerpt of a matrix of dialogue acts for Speaker 1 and Speaker 2. Start – start time of the dialogue act, End – end time of the dialogue act, Dur – duration of the dialogue act. [pw(number)] stands for the filled pause (Pl. *pauza wypełniona*) and the number is the duration of the pause in milliseconds. Grey colour divides the dialogue acts sequences into parts and indicate the moments in the dialogue when neither of the interlocutors was speaking; more precisely – when one of the speakers was finishing his turn and the other did not yet start speaking.

Speaker 1					***Speaker 2***				
Start	***End***	***Dur***	***Dialogue act***	***Utterance***	***Start***	***End***	***Dur***	***Dialogue act***	***Utterance***
0	2.56	2.56							
					0	3.81	3.81		
2.56	4.02	1.46	turn management, contact management, information providing, directives, open meeting	czyli idziemy od lewego dolnego naroznika					
					3.81	4.19	0.38	auto-feedback	tak
4.02	5.02	1.00	information providing, directives	do prawego gornego					
					4.19	4.84	0.65		
					4.84	6.27	1.43	information providing, directives	wychodzimy z zamku
5.02	6.26	1.24							
6.26	6.58	0.32	contact management, auto-feedback	tak					
					6.27	7.69	1.42		
6.58	8.09	1.51							
					7.69	8.40	0.72	time management	[pw720]
8.09	9.3	1.21	information seeking	a mamy wejsc tam na gorze					
					8.40	10.90	2.50		
9.30	9.71	0.42	information seeking	do je					
9.71	10.82	1.11	information seeking, own communication management	do tego budynku po prostu					
10.82	11.74	0.92	information seeking	obojetnie od ktorej strony					
					10.90	11.53	0.63	time management	[pw570]
					11.53	12.86	1.33	information	do zamku

Speaker 1					*Speaker 2*				
Start	***End***	***Dur***	***Dialogue act***	***Utterance***	***Start***	***End***	***Dur***	***Dialogue act***	***Utterance***
								providing	
11.74	12.00	0.26	contact management	tak					
12.00	17.24	5.24							
					12.86	12.96	0.10		
					12.96	13.91	0.95	information providing	tez musimy
					13.91	14.22	0.31		
					14.22	14.55	0.33	auto-feedback	tak
					14.55	14.81	0.25	auto-feedback	tak
					14.81	14.99	0.18	auto-feedback	tak
					14.99	16.20	1.21	information providing	od strony fosy tutaj
					16.20	17.2	1.00	information providing	mostu zwodzonego
					17.20	18.07	0.87		
17.24	18.20	0.96	auto-feedback	mostu zwodzonego					
					18.07	19.57	1.50	information providing	no tak taka malutka no
18.20	19.60	1.40							
					19.57	20.51	0.94		
19.60	20.26	0.66	information providing	(ale) cos tam jest					
20.26	21.12	0.86	information providing	takie moze no					
					20.51	20.88	0.37	auto-feedback	tak
					20.88	21.20	0.32	auto-feedback	tak
21.12	22.31	1.19							
					21.20	21.28	0.08		

Appendix B Loop-free automata for speaker 1

Table with loop-free automata for each sequence of dialogue acts for speaker 1; Dur – duration, I – initial state, T – terminal state, ID – automaton ID, grey colour – the moments when neither of the interlocutors was speaking, divides the dialogue into parts which define the sequences on which automata were built.

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
0.00	2.56	2.56								
2.56	4.02	1.46	turn, cnt, infpr, dir, open	czy(li) idziemy od lewego dolnego naroznika	q0	q3	q0, x, turn_cnt_infpr_dir_ open, q1;	x x x	turn_cnt_in fpr_dir_op en infpr_dir cnt_auto	1
							q1, x, infpr_dir, q2;			
4.02	5.02	1.00	infpr, dir	do prawego gornego			q2, x, cnt_auto, q3			
5.02	6.26	1.24								
6.26	6.58	0.32	cnt, auto	tak						
6.58	8.09	1.51								
					q0	q3	q0, x, infsk, q1;	x x x	infsk own cnt	2
8.09	9.30	1.21	infsk	a mamy wejsc tam na gorze			q1, x, own, q2;			
							q2, x, cnt, q3			
9.30	9.71	0.42	infsk	do je						
9.71	10.82	1.11	infsk, own	do tego budynku po prostu						
10.82	11.74	0.92	infsk	obojetnie od ktorej strony						
11.74	12.00	0.26	cnt	tak						
12.00	17.24	5.24								
17.24	18.20	0.96	auto	mostu zwodzonego	q0	q2	q0, x, auto, q1;	x x	auto infpr	3
							q1, x, infpr, q2			
18.20	19.60	1.40								
19.60	20.26	0.66	infpr	(ale) cos tam jest						
20.26	21.12	0.86	infpr	takie moze no						
21.12	22.31	1.19								
22.31	22.99	0.68	infpr	kolo konia (w kazdym razie)	q0	q3	q0, x, infpr, q1;	x x x	infpr cnt cnt_auto	4
							q1, x, cnt, q2;			
22.99	23.49	0.50	infpr	kon jest			q2, x, cnt_auto, q3			
23.49	23.84	0.35	cnt	tak						
23.84	24.42	0.58								
24.42	24.83	0.41	cnt, auto	dobra						
24.83	25.68	0.85								
25,68	26,59	0,91	infpr	(to) gdzie mam isc						
26,59	31,43	4,83								
31,43	31,66	0,23	auto, cnt	tak						
31,66	36,92	5,26								
36,92	37,47	0,55	auto	[aha]						
37,47	37,65	0,18								
37,65	38,12	0,47	time_ma nagemet	[pw550]	q0	q3	q0, x, time, q1;	x x x	time infsk cnt_auto	
38,12	38,93	0,81	infsk	w poziomym kierunku			q1, x, infsk, q2;			
38,93	40,34	1,42					q2, x, cnt_auto, q3			
40,34	40,88	0,54	auto, contact_	dobra						

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
			mamage ment							
40,88	46,28	5,40								
46,28	47,19	0,90	auto	[mhm]	q0	q1	q0, x, auto, q1	x	auto	
47,19	47,25	0,06								
47,25	48,33	1,08	infsk	obok tego drzewa	q0	q1	q0, x, infsk, q1	x	infsk	
48,33	49,01	0,68								
49,01	50,21	1,21	infsk	czyli z lewej strony tego drzewa						
50,21	51,24	1,03								
51,24	52,28	1,04	infsk	przy tym drzewie mam przejsc	q0	q4	q0, x, infsk, q1;	x x x x	infsk auto infsk turn_infpr	
52,28	66,04	13,76					q1, x, auto, q2;			
							q2, x, infsk, q3;			
							q3, x, turn_infpr, q4			
66,04	66,64	0,60	auto	[aha]						
66,64	67,63	1,00	infsk	taka baszta jakby						
67,63	76,71	9,08								
76,71	77,96	1,25	turn, infpr	moment bo nie mam baobabu						
77,96	78,55	0,59								
78,55	79,60	1,04	infpr	przeszedlem kolo tego drzewa	q0	q1	q0, x, infpr, q1;	x x	infpr infpr_infsk	
79,60	81,28	1,68	infpr, infsk	na ktorym tam chyba jakies gniazdo bocianie jest			q1, x, infpr_infsk, q2			
81,28	82,16	0,88	infpr	przeszedlem z lewej strony						
82,16	83,50	1,34	infpr	ide w kierunku tej baszty						
83,50	84,80	1,30								
84,80	85,42	0,62	auto, cnt	prosto	q0	q1	q0, x, auto_cnt, q1	x	auto_cnt	
85,42	89,78	4,36								
89,78	90,08	0,30	auto, cnt	(tak)	q0	q2	q0, x, auto_cnt, q1;	x x	auto_cnt infpr	
							q1, x, infpr, q2			
90,08	91,70	1,62								
91,70	92,57	0,87	infpr	i tam nie mam baobabu						
92,57	92,84	0,27								
92,84	93,70	0,86	infpr	ja widze taka baszte						
93,70	95,05	1,35	infpr	a za basztami sa jakies gory						
95,05	95,61	0,57								
95,61	96,46	0,85	infpr	za baszta sa gory						
96,46	98,32	1,86								
98,32	98,70	0,38	auto	[aha]	q0	q2	q0, x, auto, q1;	x x	auto infpr_cnt	
98,70	101,24	2,53					q1, x, infpr_cnt, q2			
101,24	101,66	0,42	infpr, cnt	tak						
101,66	107,78	6,12								
107,78	108,12	0,34	auto, cnt	tak	q0	q3	q0, x, auto_cnt, q1;	x x x	auto_cnt cnt infpr	
							q1, x, cnt, q2;			
108,12	108,67	0,55	cnt	okej			q2, x, infpr, q3			
108,67	109,32	0,65	infpr	to jest dobre (nie)						
109,32	112,74	3,41								
112,74	113,58	0,84	infsk,	czyli w tak jakby w	q0	q4	q0, x, infsk_infpr,	x x x x	infsk_infpr	

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
			infpr	dol			q1;		cnt infsk auto_infpr	
113,58	113,85	0,27	cnt	tak			q1, x, cnt, q2;			
113,85	114,60	0,75					q2, x, infsk, q3;			
							q3, x, auto_infpr, q4			
114,60	115,33	0,73	infsk	w dol rysunku						
115,33	117,23	1,90								
117,23	118,69	1,45	auto, infpr	tak idziemy poziomo do wiezy						
118,69	120,48	1,79								
120,48	121,38	0,90	turn, infpr	i teraz skrecamy w prawo	q0	q3	q0, x, turn_infpr, q1;	x x x	turn_infpr infpr_infsk auto_cnt	
121,38	122,45	1,07	infpr, infsk	czyli tak jakby w dol rusynku			q1, x, infpr_infsk, q2;			
122,45	139,74	17,30					q2, x, auto_cnt, q3			
139,74	141,44	1,70	auto, cnt	[aha]						
141,44	142,19	0,75								
142,19	142,83	0,64	infpr, turn	czyli juz wiem	q0	q2	q0, x, turn_infpr, q1;	x x	turn_infpr infpr	
							q1, x, infpr, q2			
142,83	143,46	0,62	infpr	mielismy isc						
143,46	144,27	0,81								
144,27	144,53	0,26	infpr, turn	tak	q0	q3	q0, x, turn_infpr, q1;	x x x	turn_infpr infpr_infsk infpr	
144,53	146,11	1,58	infpr, infsk	i w gore od drzewa isc			q1, x, infpr_infsk, q2;			
							q2, x, infpr, q3			
146,11	146,71	0,61								
146,71	147,59	0,88	infpr	to nie ta baszta						
147,59	147,90	0,30								
147,90	149,21	1,31	infpr	to ja musze wy wylizac droge	q0	q6	q0, x, infpr, q1;	x x x x x x	infpr dir infpr infpr_infsk auto infsk	
							q1, x, dir, q2;			
							q2, x, infpr, q3;			
149,21	149,76	0,55	dir	poczekaj			q3, x, infpr_infsk, q4;			
							q4, x, auto, q5;			
149,76	150,26	0,50					q5, x, infsk, q6			
150,26	151,21	0,95	infpr	z koszulki						
151,21	151,75	0,54								
151,75	154,19	2,44	infpr, infsk	taka baszta z takimi trzema wiezyczkami						
154,19	155,26	1,07	auto	[aha]						
155,26	156,64	1,37	infsk	taka obok koloseum						
156,64	158,64	2,00								
158,64	159,56	0,92	auto	nie masz koloseum						
159,56	159,89	0,33	auto	dobra						
159,89	160,35	0,46								
160,35	162,10	1,75	infpr, dir	w kazdym razie idziemy do tej u gory	q0	q7	q0, x, infpr_dir, q1;	x x x x x x	infpr_dir auto infpr infpr_time infpr_own cnt infpr_infsk	
162,10	163,74	1,64	infpr, dir	w takim razie idziemy na polnoc			q1, x, auto, q2;			

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
							q2, x, infpr, q3;			
							q3, x, infpr_time, q4;			
163,74	165,46	1,72					q4, x, infpr_own, q5;			
							q5, x, cnt, q6;			
							q6, x, infpr_infsk, q7			
165,46	165,92	0,46	auto	dobra						
165,92	167,39	1,47								
167,39	168,16	0,77	infpr	i teraz jestem						
168,16	168,65	0,48	infpr	jestem						
168,65	169,11	0,46	infpr, time	przy						
169,11	170,04	0,93	infpr, time	przy tej						
170,04	171,81	1,77	infpr, own_comunication_management	przy tym domku z tymi trzema wiezyczkami						
171,81	176,47	4,66								
176,47	176,84	0,37	cnt	tak						
176,84	177,23	0,39								
177,23	177,77	0,54	cnt	tak						
177,77	178,59	0,82	infpr, infsk	przy tej wschodniej						
178,59	186,57	7,98								
186,57	186,96	0,39	turn, dir	ale (czekaj)	q0	q2	q0, x, turn_dir, q1;	x x	turn_dir infpr	
							q1, x, infpr, q2			
186,96	187,92	0,96	infpr	bo jak ja skrece w prawo						
187,92	188,93	1,02	infpr	to jak mam tam koloseum						
188,93	191,48	2,55								
191,48	191,67	0,18	infpr	tak						
191,67	193,49	1,82	infpr	takie normalne koloseum jak jak w rzymie						
193,49	195,31	1,82								
195,31	196,02	0,71	infpr	nie mam baobabu	q0	q2	q0, x, infpr, q1;	x x	infpr turn	
196,02	197,60	1,58					q1, x, turn, q2			
197,60	198,87	1,26	infpr	a mam jakiegos dziadka co stoi						
198,87	201,05	2,19	infpr	jakiegos mnicha na polnoc od od od koloseum						
201,05	203,74	2,69								
203,74	203,99	0,25	turn	[pw250]						
203,99	204,66	0,67								
204,66	205,33	0,66	time	[pw660]	q0	q3	q0, x, time, q1;	x x x	time infpr infsk	
205,33	206,28	0,96	infpr	no tak kawaleczek pod mnichem			q1, x, infpr, q2;			
206,28	207,59	1,31	infpr	powiedzialbym ze na rowni dokladnie			q2, x, infsk, q3			
207,59	209,20	1,61	infpr	jezeli chodzi o rownolezniki						
209,20	209,81	0,61	time	ysys	q0	q6	q0, x, time, q1;	x x x x x	time infpr time	

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
									time_own infpr infsk	
209,81	210,18	0,37	time, own	[ys]			q1, x, infpr, q2;			
210,18	210,41	0,23	time, own	[y]			q2, x, time, q3;			
210,41	210,97	0,56	infpr	ta baszta			q3, x, time_own, q4;			
210,97	211,19	0,22	infpr	ide			q4, x, infpr, q5;			
211,19	211,90	0,72	infpr	idealnie na rowni			q5, x, infsk, q6			
211,90	213,00	1,09	infpr	jezeli chodzi o rownolezniki						
213,00	213,71	0,72								
213,71	214,66	0,94	infpr	czyli na tym samym poziomie						
214,66	216,15	1,50								
216,15	216,77	0,62	infsk	a mnicha masz						
216,77	225,21	8,43								
225,21	226,63	1,42	infpr	dwa centymetry pod mnichem bedzie mozliwe	q0	q3	q0, x, infpr, q1;	xxx	infpr infpr_partner allo	
226,63	228,11	1,48	infpr	to przejde za koloseum akurat			q1, x, infpr_partner, q2;			
							q2, x, allo, q3			
228,11	231,74	3,63								
231,74	232,50	0,76	infpr, partner	dwa bylo mowa						
232,50	233,21	0,71								
233,21	234,16	0,95	allo	[smiech]						
234,16	234,25	0,09								
234,25	234,56	0,32	time	na	q0	q7	q0, x, time, q1;	x x x x x x x	time infsk cnt auto_cnt cnt_infpr infpr infsk	
234,56	235,14	0,58	infsk	na wschod			q1, x, infsk, q2;			
235,14	235,32	0,18	cnt	tak			q2, x, cnt, q3;			
235,32	236,53	1,21					q3, x, auto_cnt, q4;			
							q4, x, cnt_infpr, q5;			
							q5, x, infpr, q6;			
							q6, x, infsk, q7			
236,53	237,07	0,54	auto, cnt	dobra	q0	q7	q0, x, time, q1;	x x x x x x x	time infsk cnt auto_cnt cnt_infpr infpr infsk	
237,07	243,00	5,93					q1, x, infsk, q2;		time infsk cnt auto_cnt infpr infpr infsk	
							q2, x, cnt, q3;		time infsk cnt infpr infpr infpr infsk	
							q3, x, auto_cnt, q4;	x x x x x x x x x x x x x x x	time infsk cnt auto_cnt cnt_infpr infpr infsk cnt infpr infpr	

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
									auto_cnt infpr infpr infpr infsk	
243,00	243,96	0,95	cnt, infpr	dwa centymetry			q4, x, cnt_infpr, q5;		time infsk cnt auto_cnt cnt_infpr infpr infsk cnt infpr infpr auto_cnt infpr infpr infpr infsk	
243,96	245,29	1,33	infpr	na poludniowy wschod			q5, x, infpr, q6;		time infsk cnt auto_cnt infpr infpr infpr infpr infpr infpr infpr infpr infpr infpr infsk	
245,29	246,63	1,34	infsk	czyli tak w dol po skosie			q5, x, infpr, q5;		time infsk cnt auto_cnt infpr infpr infpr infpr infsk cnt auto_cnt infpr infpr infpr infsk	
246,63	246,88	0,25	cnt	tak			q6, x, infsk, q7;		time infsk cnt auto_cnt infpr infpr infpr infpr auto_cnt cnt_infpr infpr auto_cnt infpr infpr infsk	
							q5, x, infpr, q3;		time infsk cnt infpr infpr infpr infpr infpr auto_cnt cnt_infpr infpr auto_cnt infpr infpr infsk	
246,88	247,85	0,96					q5, x, infpr, q1;		time infsk cnt infpr infpr infpr infsk cnt auto_cnt infpr infpr infpr infpr infpr infsk	
							q4, x, infpr, q5;		time infsk cnt infpr	

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
									infpr infpr auto_cnt infpr infpr infpr infpr auto_cnt infpr infpr infsk	
247,85	248,47	0,62	infpr	to ja tu nie moge			q3, x, infpr, q5		time infsk cnt infpr infpr infpr infpr infpr infsk cnt auto_cnt cnt_infpr infpr infsk	
248,47	249,06	0,59	infpr	bo ja tu mam gory						
249,06	253,12	4,06								
253,12	253,62	0,50	auto, cnt	no dobra						
253,62	254,32	0,70	infpr	jestem przy gorach						
254,32	262,85	8,53								
262,85	263,44	0,59	infpr	to ja mam gory						
263,44	267,11	3,67								
267,11	267,69	0,58	infpr	pod (w) gorami						
267,69	268,53	0,84								
268,53	270,52	1,99	infsk	to ja mam przejsc te gory z lewej strony czy z prawej						
270,52	271,96	1,44								
271,96	272,55	0,59	time	[pw590]	q0	q6	q0, x, time, q1;	x x x x x x	time auto infsk auto turn infsk_infpr	
272,55	273,02	0,47	auto	z lewej strony			q1, x, auto, q2;			
273,02	275,80	2,77	infsk	czyli blizej tego mojego koloseum a twojego baobabu			q2, x, infsk, q3;			
275,80	278,83	3,03					q3, x, auto, q4;			
							q4, x, turn, q5;			
							q5, x, infsk_infpr, q6			
278,83	279,08	0,25	auto	tak						
279,08	280,37	1,28								
280,37	280,72	0,35	turn	i ba						
280,72	284,27	3,56	infsk, infpr	i baszte mam miec jak patrze na rysunek [p710] po prawej stronie drogi						
284,27	286,97	2,69								
286,97	287,26	0,29	turn	czyli takie	q0	q3	q0, x, turn, q1;	x x x	turn infpr infsk	
287,26	288,37	1,11	infpr	jakby taka petle zatoczylem			q1, x, infpr, q2;			
288,37	289,66	1,30	infpr	wokol tego mojego koloseum			q2, x, infsk, q3			
289,66	289,87	0,21	infsk	tak						
289,87	302,63	12,76								
302,63	303,35	0,72	infpr	i pijemy wino	q0	q3	q0, x, infpr, q1;	x x x	infpr cnt auto	
303,35	306,39	3,04					q1, x, cnt, q2;			

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
							q2, x, auto, q3			
306,39	307,00	0,60	cnt	(dobra)						
307,00	310,27	3,27								
310,27	310,66	0,39	auto	tak						
310,66	310,95	0,29								
310,95	311,23	0,28	auto	tak	q0	q3	q0, x, auto, q1;	x x x	auto infpr infsk	
311,23	313,40	2,17					q1, x, infpr, q2;			
							q2, x, infsk, q3			
313,40	313,90	0,50	auto	tak						
313,90	317,53	3,62								
317,53	317,96	0,43	auto	dobra						
317,96	318,48	0,53	infpr	oczko mam						
318,48	318,78	0,30								
318,78	320,34	1,56	infsk	caly czas poziomo idziemy do oczka						
320,34	321,43	1,09								
321,43	322,35	0,92	infpr	stoje przed oczkiem						
322,35	327,12	4,77								
327,12	328,08	0,96	infsk	od strony polnocnej						
328,08	328,73	0,66								
328,73	329,23	0,50	auto	dobra	q0	q6	q0, x, auto, q1;	x x x x x	auto infpr dir cnt infpr_dir allo	
329,23	332,34	3,10					q1, x, infpr, q2;			
							q2, x, dir, q3;			
							q3, x, cnt, q4;			
							q4, x, infpr_dir, q5;			
							q5, x, allo, q6			
332,34	332,71	0,37	infpr	jedno						
332,71	336,61	3,90								
336,61	337,57	0,96	dir	to przejdY miedzy drzewami						
337,57	337,73	0,15	cnt	tak						
337,73	338,14	0,42								
338,14	338,79	0,65	infpr, dir	jak masz jedno						
338,79	338,88	0,08								
338,88	338,96	0,08	allo	[smiech]						
338,96	339,11	0,15								
339,11	339,84	0,73	allo	[smiech]						
339,84	341,07	1,24								
341,07	341,47	0,40	partner	grube	q0	q1	q0, x, partner, q1	x	partner	
341,47	345,86	4,40								
345,86	346,59	0,73	time	[pw]	q0	q3	q0, x, time, q1;	x x x	time infpr auto	
346,59	346,98	0,39	infpr	nie			q1, x, infpr, q2;			
346,98	347,04	0,07					q2, x, auto, q3			
347,04	347,72	0,68	infpr	po lewej stronie oczka						
347,72	348,84	1,12	infpr	stoja dwa cienkie drzewa						
348,84	349,35	0,51								
349,35	351,51	2,16	infpr	tak z trzy centymetry na na na zachod od oczka						
351,51	352,01	0,50	infpr	przy gorach						
352,01	353,07	1,06	infpr	stoja dwa cienkie drzewa						
353,07	353,53	0,46								
353,53	354,31	0,78	infpr	a(y) grube drzewo						

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
354,31	356,54	2,23	infpr	stoi na polnoc od oczka jakies trzy centymetry						
356,54	358,68	2,14								
358,68	359,08	0,40	auto	no						
359,08	362,19	3,12								
362,19	362,71	0,51	auto	a						
362,71	366,61	3,90								
366,61	367,73	1,13	infpr	no to u mnie jest odwrotnie	q0	q1	q0, x, infpr, q1	x	infpr	
367,73	371,62	3,89								
371,62	371,89	0,27	cnt, auto	ta(k)	q0	q1	q0, x, cnt, q1;	x	cnt	
371,89	378,34	6,45								
378,34	378,63	0,29	cnt	tak						
378,63	382,67	4,03								
382,67	383,04	0,37	cnt	tak						
383,04	390,72	7,68								
390,72	391,19	0,47	auto, cnt, turn	[aha]	q0	q5	q0, x, auto_cnt_turn, q1;	x x x x x	auto_cnt_turn infsk cnt auto infpr	
							q1, x, infsk, q2;			
391,19	392,96	1,77	infsk	to drzewko mam w ktorym momencies ominac			q2, x, cnt, q3;			
							q3, x, auto, q4;			
							q4, x, infpr, q5			
392,96	393,24	0,29	cnt	tak						
393,24	396,53	3,29								
396,53	397,34	0,81	cnt	po prawej stronie						
397,34	397,73	0,39	auto	dobra						
397,73	399,39	1,66	infpr	to przechodze w takim razie z lewej strony drzewa						
399,39	401,74	2,35								
401,74	402,29	0,55	partner, infpr	duze gory	q0	q1	q0, x, partner_infpr, q1	x	partner_infpr	
402,29	409,30	7,01								
409,30	409,51	0,21	cnt	tak	q0	q2	q0, x, cnt, q1;	x	cnt infpr_auto	
409,51	412,27	2,75					q1, x, infpr_auto, q2			
412,27	413,18	0,92	infpr, auto	czyli od zachodu						
413,18	414,24	1,06								
414,24	414,64	0,40	cnt,	dobra	q0	q1	q0, x, cnt, q1	x	cnt	
414,64	415,01	0,37								
415,01	416,18	1,17	infpr	jak ja tam pojde dwa centymetry	q0	q10	q0, x, infpr, q1;	x x x x x x x x x x	infpr time infpr own_infpr infpr cnt infsk cnt infpr_infsk cnt	
416,18	417,28	1,11	infpr	tam stoi namiot jakis			q1, x, time, q2;			
417,28	419,58	2,29					q2, x, infpr, q3;			
							q3, x, own_infpr, q4;			
							q4, x, infpr, q5;			
							q5, x, cnt, q6;			
							q6, x, infsk, q7;			

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
419,58	420,23	0,66	time	[pw660]			q7, x, cnt, q8;			
							q8, x, infpr_infsk, q9;			
420,23	420,99	0,76	infpr	no nie			q9, x, cnt, q10			
420,99	422,32	1,32								
422,32	423,10	0,78	infpr	choragiewka jest						
423,10	425,21	2,12	infpr	ale jest to taki namiot plocienny wedlug mnie						
425,21	425,63	0,42	infpr	odz						
425,63	426,08	0,45	own, infpr	odwijany						
426,08	427,45	1,37	infpr	no w pasy jest bialoniebieskie						
427,45	427,87	0,42	cnt	takze						
427,87	430,49	2,62								
430,49	431,40	0,91	infsk	na lewo od tych gor						
431,40	431,64	0,24	cnt	tak						
431,64	440,14	8,50								
440,14	441,88	1,74	cnt	na polnocny zachod						
441,88	442,22	0,34	cnt	dobrze						
442,22	445,02	2,80	infpr, infsk	i przechodzimy miedzy tym tym namiotem a tymi gorami jakos						
445,02	445,23	0,20	cnt	tak						
445,23	447,40	2,17								
447,40	447,78	0,38	cnt	tak	q0	q1	q0, x, cnt, q1	x	cnt	
447,78	452,05	4,27								
452,05	452,57	0,52	auto, cnt	[aha]	q0	q3	q0, x, auto_cnt, q1;	x x x	auto_cnt cnt auto_cnt	
452,57	452,93	0,36	cnt	dobra			q1, x, cnt, q2;			
452,93	457,42	4,49					q2, x, auto_cnt, q3			
457,42	457,99	0,57	auto, cnt	[aha]						
457,99	459,62	1,63								
459,62	460,07	0,45	auto, cnt	tak						
460,07	465,12	5,06								
465,12	465,83	0,71	cnt, auto	na zachod	q0	q4	q0, x, auto_cnt, q1;	x x x x	auto_cnt cnt auto_cnt auto_cnt_time	
465,83	466,73	0,89					q1, x, cnt, q2;			
							q2, x, auto_cnt, q3;			
							q3, x, auto_cnt_time, q4			
466,73	467,17	0,44	cnt	dobra						
467,17	472,91	5,74	??	DEL						
472,91	473,29	0,38	auto, cnt	tak						
473,29	476,24	2,95								
476,24	477,27	1,03	auto, cnt, time	[pw1030]						
477,27	480,97	3,70								
480,97	482,38	1,42	infsk	taka przy takich drzewach jakby	q0	q9	q0, x, infsk, q1;	x x x x x x x x x	infsk cnt partner cnt_infpr infpr cnt auto_cnt turn_time infsk	

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
							q1, x, cnt, q2;			
482,38	483,31	0,93	infsk	taka mala skalka			q2, x, partner, q3;			
							q3, x, cnt_infpr, q4;			
							q4, x, infpr, q5;			
483,31	483,64	0,33	cnt	tak			q5, x, cnt, q6;			
							q6, x, auto_cnt, q7;			
483,64	484,70	1,06					q7, x, turn_time, q8;			
							q8, x, infsk, q9			
484,70	485,10	0,40	cnt	[mhm]						
485,10	487,69	2,59								
487,69	488,05	0,36	partner	kepa						
488,05	488,30	0,24	cnt, infpr	tak						
488,30	489,00	0,71	infpr	taka dosyc duza						
489,00	489,21	0,21	cnt	[mhm]						
489,21	492,05	2,84								
492,05	492,36	0,31	auto, cnt	tak						
492,36	492,57	0,21	turn, time	[pw210]						
492,57	494,10	1,53	infsk	omijajac ta skalke z lewej strony						
494,10	499,70	5,60								
499,70	500,03	0,33	infpr, cnt	tak	q0	q4	q0, x, infpr_cnt, q1;	x x x x	infpr_cnt infsk auto_cnt turn_infpr	
500,03	501,78	1,75					q1, x, infsk, q2;			
							q2, x, auto_cnt, q3;			
							q3, x, turn_infpr, q4			
501,78	502,52	0,74	infsk	za ta kepe						
502,52	507,68	5,16								
507,68	507,99	0,30	auto, cnt	tak						
507,99	510,61	2,62								
510,61	511,83	1,22	turn, infpr	i tu stoi jakis						
511,83	512,25	0,42								
512,25	512,49	0,25	turn	tak	q0	q9	q0, x, turn, q1;	x x x x x x x x x	turn infpr infpr_time infpr auto infsk time infpr infsk	
512,49	513,56	1,06	infpr	ale tu (z tylu) ktos stoi			q1, x, infpr, q2;			
513,56	516,05	2,49					q2, x, infpr_time, q3;			
							q3, x, infpr, q4;			
516,05	517,95	1,90	infpr	no tam ke ja mijam kepe tam ktos stoi ale			q4, x, auto, q5;			
							q5, x, infsk, q6;			
517,95	518,29	0,33	infpr, time	ale			q6, x, time, q7;			
518,29	519,01	0,73					q7, x, infpr, q8;			
519,01	519,94	0,93	infpr	bo jest taki domek tam			q8, x, infsk, q9			
519,94	521,07	1,13	infpr	na prawo od tej kepy						
521,07	525,48	4,41								
525,48	526,13	0,65	auto	niebieskiego						
526,13	526,61	0,48	infsk	w domku						
526,61	531,59	4,98								

Start	*End*	*Dur*	*DAs*	*Utterances*	*I*	*T*	*Transitions*	*Input*	*Output*	*ID*
531,59	531,73	0,14	time	[pw140]						
531,73	532,43	0,70	infpr	takie cos tam stoi						
532,43	533,53	1,11	infsk	nie wiem czy to jest facet czy						
533,53	537,21	3,67								
537,21	538,07	0,86	infpr	on stoi na sciezce	q0	q4	q0, x, infpr, q1;	x x x x	infpr infpr_infsk cnt_auto infsk	
538,07	538,45	0,39					q1, x, infpr_infsk, q2;			
							q2, x, cnt_auto, q3;			
							q3, x, infsk, q4			
538,45	539,62	1,16	infpr, infsk	czyli pod girami mu przechodze						
539,62	540,74	1,13								
540,74	541,20	0,46	cnt, auto	dobra						
541,20	545,12	3,91								
545,12	545,56	0,44	cnt	[mhm]						
545,56	547,13	1,58								
547,13	548,31	1,18	infsk	jakas wyrwa jest w ziemi						
548,31	552,89	4,58								
552,89	553,30	0,42	cnt	[mhm]	q0	q2	q0, x, cnt, q1;	x x	cnt auto_cnt	
							q1, x, auto_cnt, q2			
553,30	560,03	6,72								
560,03	561,21	1,18	auto, cnt	poludniowy wschod						
561,21	562,19	0,98								
562,19	562,42	0,23	turn	[aha]	q0	q6	q0, x, turn, q1;	x x x x x x	turn infsk infpr cnt infsk infpr	
562,42	563,72	1,30	infsk	od polnocnej strony wyrwy			q1, x, infsk, q2;			
563,72	564,46	0,74	infsk	czy od poludniowej			q2, x, infpr, q3;			
564,46	565,99	1,53					q3, x, cnt, q4;			
							q4, x, infsk, q5;			
							q5, x, infpr, q6			
565,99	567,60	1,61	infpr	czyli nad wyrwa przechodze	q0	q3	q0, x, turn, q1;	x x x x x x	turn infsk infpr cnt infsk infpr	
							q1, x, infsk, q2;	x x x	turn infsk infpr	
							q2, x, infpr, q3;	x x x x x x x x x	turn infsk infpr cnt infsk infpr cnt infsk infpr	
567,60	572,35	4,74					q3, x, cnt, q1	x x x x x x x x x x x x	turn infsk infpr cnt infsk infpr cnt infsk infpr cnt infsk infpr	
572,35	572,84	0,50	cnt	[aha]						
572,84	577,10	4,26								
577,10	577,56	0,46	infsk	do konia						
577,56	579,55	1,99								
579,55	580,06	0,51	infpr	mam konia						
580,06	583,84	3,78								

Appendix C Reduction of multi-layered labels

Reduction of multi-layered labels to one-layered labels in the outputs of dialogue act automata of each sequence in the dialogue for speaker 1 and speaker 2 sorted alphabetically. ID stands for the ID of the automaton.

Appendix C.1 Speaker 1

Before		***After***	
ID	***Outputs with multi-layered labels***	***ID***	***Outputs with one-layered labels***
6	auto	6	auto
3	auto infpr	10	auto
32	auto infpr dir cnt infpr_dir allo	43	auto cnt auto
31	auto infpr infsk	44	auto cnt auto auto
12	auto infpr_cnt	13	auto cnt infpr
10	auto_cnt	11	auto infpr
43	auto_cnt cnt auto_cnt	3	auto infpr
44	auto_cnt cnt auto_cnt auto_cnt_time	12	auto infpr
13	auto_cnt cnt infpr	32	auto infpr dir cnt infpr allo
11	auto_cnt infpr	31	auto infpr infsk
37	auto_cnt_turn infsk cnt auto infpr	37	auto infsk cnt auto infpr
36	cnt	36	cnt
40	cnt	40	cnt
42	cnt	42	cnt
49	cnt auto_cnt	49	cnt auto
39	cnt infpr_auto	39	cnt infpr
35	infpr	35	infpr
30	infpr cnt auto	19	infpr auto infpr infpr infpr cnt infpr
4	infpr cnt cnt_auto	30	infpr cnt auto
18	infpr dir infpr infpr_infsk auto infsk	4	infpr cnt cnt
9	infpr infpr_infsk	18	infpr dir infpr infpr auto infsk
48	infpr infpr_infsk cnt_auto infsk	9	infpr infpr
24	infpr infpr_partner allo	24	infpr infpr allo
41	infpr time infpr own_infpr infpr cnt infsk cnt infpr_infsk cnt	48	infpr infpr cnt infsk
21	infpr turn	46	infpr infsk auto turn
46	infpr_cnt infsk auto_cnt turn_infpr	41	infpr time infpr own infpr cnt infsk cnt infpr cnt
19	infpr_dir auto infpr infpr_time infpr_own cnt infpr_infsk	21	infpr turn
7	infsk	7	infsk
8	infsk auto infsk turn_infpr	8	infsk auto infsk turn
45	infsk cnt partner cnt_infpr infpr cnt auto_cnt turn_time infsk	14	infsk cnt infsk auto
2	infsk own cnt	45	infsk cnt partner cnt infpr cnt auto turn infsk
14	infsk_infpr cnt infsk auto_infpr	2	infsk own cnt
33	partner	33	partner
38	partner_infpr	38	partner
28	time auto infsk auto turn infsk_infpr	28	time auto infsk auto turn infsk
34	time infpr auto	34	time infpr auto
22	time infpr infsk	22	time infpr infsk
23	time infpr time time_own infpr infsk	23	time infpr time time infpr infsk
25	time infsk cnt auto_cnt cnt_infpr infpr infsk	5	time infsk cnt
5	time infsk cnt_auto	25	time infsk cnt auto cnt infpr infsk
47	turn infpr infpr_time infpr auto infsk time infpr infsk	20	turn infpr
29	turn infpr infsk	16	turn infpr
50	turn infsk infpr cnt infsk infpr	15	turn infpr auto
20	turn_dir infpr	1	turn infpr cnt
16	turn_infpr infpr	17	turn infpr infpr
15	turn_infpr infpr_infsk auto_cnt	47	turn infpr infpr infpr auto infsk time infpr infsk

Before		*After*	
ID	***Outputs with multi-layered labels***	***ID***	***Outputs with one-layered labels***
17	turn_infpr infpr_infsk infpr	29	turn infpr infsk
1	turn_open infpr_dir cnt_auto	50	turn infsk infpr cnt infsk infpr

Appendix C.2 Speaker 2

Before		*After*	
ID	***Outputs with multi-layered labels***	***ID***	***Outputs with one-layered labels***
24	auto dir	24	auto dir
34	auto infpr infsk_cnt infpr_turn infpr	1	auto infpr
1	auto infpr_dir	34	auto infpr infsk infpr infpr
17	dir	17	dir
21	dir auto auto_infpr infpr time infpr_turn infpr	21	dir auto auto infpr time infpr infpr
19	dir auto cnt infpr	19	dir auto cnt infpr
8	infpr	8	infpr
3	infpr auto	6	infpr
4	infpr auto infpr time_turn time infpr_dir	11	infpr
40	infpr infpr_dir	3	infpr auto
22	infpr infpr_dir time infsk time_turn	4	infpr auto infpr time time infpr
7	infpr partner	44	infpr cnt infpr infpr time infpr infpr own
28	infpr time	13	infpr infpr
29	infpr time auto infpr turn infpr_dir	40	infpr infpr
29	infpr time infpr_dir infpr turn infpr_dir	9	infpr infpr auto infpr infpr infpr infpr time infpr dir
29	infpr time infpr_time infpr turn infpr_dir	31	infpr infpr dir infpr infpr infpr
12	infpr time infpr_time infpr_turn	43	infpr infpr infpr
30	infpr time infpr_time time infpr_dir time auto infpr turn infpr_dir	36	infpr infpr infpr auto
46	infpr time_allo infpr infpr_dir time infpr_dir	26	infpr infpr infpr dir own dir infpr infpr infsk infpr
32	infpr turn infpr_dir infpr auto time_cnt time infpr infpr_time auto infpr	22	infpr infpr time infsk time
45	infpr turn infpr_turn inpr infpr_dir time	48	infpr inpr infpr infpr infpr time time infpr
6	infpr_dir	7	infpr partner
44	infpr_dir cnt_infpr infpr_dir infpr time infpr infpr_dir own	28	infpr time
43	infpr_dir infpr infpr_dir	29	infpr time auto infpr turn infpr
36	infpr_dir infpr infpr_dir auto	12	infpr time infpr infpr
31	infpr_dir infpr_turn dir infpr_turn infpr infpr_dir	14	infpr time infpr infpr
48	infpr_dir inpr infpr_dir infpr_turn infpr time time_turn infpr_dir	46	infpr time infpr infpr time infpr
14	infpr_dir time infpr infpr_dir	29	infpr time infpr infpr turn infpr
11	infpr_turn	29	infpr time infpr infpr turn infpr
13	infpr_turn infpr	30	infpr time infpr time infpr time auto infpr turn infpr
9	infpr_turn infpr auto_turn infpr_dir infpr_dir_time infpr_dir infpr time infpr_dir dir	10	infpr turn
26	infpr_turn infpr_dir infpr dir own dir infpr_turn infpr infsk_cnt infpr	32	infpr turn infpr infpr auto time time infpr infpr auto infpr
10	infpr_turn turn	45	infpr turn infpr infpr infpr time
23	infsk auto_cnt auto infpr_infsk	23	infsk auto auto infpr
47	infsk turn_infpr infpr_time cnt_infsk infpr_turn infpr time infpr auto_turn	47	infsk turn infpr cnt infpr infpr time infpr auto
33	time	33	time
2	time infpr	2	time infpr
20	time infpr	20	time infpr
35	time infpr	35	time infpr
25	time infpr dir infpr_dir auto dir	25	time infpr dir infpr auto dir
39	time infpr infpr_dir	39	time infpr infpr
49	time infpr infpr_dir infpr_dir_time infpr_dir	16	time infpr infpr auto time infpr allo infpr infpr dir
16	time infpr_dir infpr auto time infpr allo infpr_dir infpr dir	49	time infpr infpr infpr infpr

Before		*After*	
ID	***Outputs with multi-layered labels***	***ID***	***Outputs with one-layered labels***
27	time infpr_dir infpr infpr_time infpr_dir partner	27	time infpr infpr infpr infpr partner
41	time infsk infpr infpr_dir time infpr_dir partner dir infpr_turn	41	time infsk infpr infpr time infpr partner dir infpr
15	time time_own time infpr_dir	15	time time time infpr
38	time turn dir time	38	time turn dir time
36	time turn infpr_dir infpr allo time cnt infpr_dir allo time infpr infpr_dir_own own	36	time turn infpr infpr allo time cnt infpr allo time infpr infpr own
18	turn_dir dir auto	18	turn dir auto
42	turn_infpr infpr_dir	42	turn infpr
5	turn_infpr infpr_dir turn_infpr	50	turn infpr infpr own infpr time infpr infpr infpr social
50	turn_time infpr_dir infpr own infpr_dir time infpr_turn infpr infpr_turn social	5	turn infpr turn

Appendix D Generalisation tables

Generalisation tables show sequences of dialogue acts generated by the simple loop-free automata based on the actual dialogue act sequences which appeared in the dialogue (cf. Appendix B and Appendix C)

1. sequences of the the same single dialogue act were reduced to just one occurrence of this dialogue act,
2. occurrences of the same sequence of dialogue acts in one output were marked in yellow,
3. if the same dialogue act repeated every second time it was marked in green.

Name stands for the name of the automata created on the basis on the dialogue act sequences, ID is the automaton ID.

Appendix D.1 Prefix generalisation table for speaker 1

Name	*ID*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1_auto	44	auto	cnt	auto							
1_auto	43	auto	cnt	auto							
1_auto	13	auto	cnt	infpr							
2_auto	32	auto	infpr	dir	cnt	infpr	allo				
2_auto	31	auto	infpr	infsk							
2_auto	11	auto	infpr								
2_auto	3	auto	infpr								
2_auto	12	auto	infpr								
3_auto	37	auto	infsk	cnt	auto	infpr					
1_auto, 2_auto	6	auto									
1_auto, 2_auto	10	auto									
4_cnt	49	cnt	auto								
4_cnt	39	cnt	infpr								
4_cnt	36	cnt									
4_cnt	40	cnt									
4_cnt	42	cnt									
13_infpr	19	infpr	auto	infpr	cnt	infpr					
14_infpr	30	infpr	cnt	auto							
14_infpr	4	infpr	cnt								
13_infpr	18	infpr	dir	infpr	auto	infsk					
14_infpr	24	infpr	allo								
14_infpr	48	infpr	cnt	infsk							
13_infpr, 14_infpr	9	infpr									
15_infpr	46	infpr	infsk	auto	turn						
14_infpr	41	infpr	time	infpr	own	infpr	cnt	infsk	cnt	infpr	cnt
14_infpr	21	infpr	turn								
	35	infpr									
11_infsk_a, 11_infsk_b	8	infsk	auto	infsk	turn						
11_infsk_a, 11_infsk_b	14	infsk	cnt	infsk	auto						
12_infsk	45	infsk	cnt	partner	cnt	infpr	cnt	auto	turn	infsk	
11_infsk_a	2	infsk	own	cnt							
11_infsk_a, 11_infsk_b	7	infsk									

Name	*ID*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
9_partner	33	partner									
9_partner	38	partner									
10_time	28	time	auto	infsk	auto	turn	infsk				
7_time	34	time	infpr	auto							
7_time	22	time	infpr	infsk							
7_time	23	time	infpr	time	infpr	infsk					
8_time_a, 8_time_b	5	time	infsk	cnt							
8_time_a, 8_time_b	25	time	infsk	cnt	auto	cnt	infpr	infsk			
6_turn	15	turn	infpr	auto							
6_turn	1	turn	infpr	cnt							
6_turn	47	turn	infpr	auto	infsk	time	infpr	infsk			
6_turn	17	turn	infpr								
6_turn	29	turn	infpr	infsk							
6_turn	20	turn	infpr								
6_turn	16	turn	infpr								
5_turn	50	turn	infsk	infpr	cnt	infsk	infpr				

Appendix D.2 Prefix generalisation table for speaker 2

Name	*ID*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
1_auto	24	auto	dir								
1_auto	34	auto	infpr	infsk	infpr						
1_auto	1	auto	infpr								
2_dir	19	dir	auto	cnt	infpr						
2_dir, 3_dir	21	dir	auto	infpr	time	infpr					
2_dir, 3_dir	17	dir									
4_infpr, 5_infpr	4	infpr	auto	infpr	time	infpr					
4_infpr, 5_infpr	9	infpr	auto	infpr	time	infpr	dir				
4_infpr, 5_infpr	3	infpr	auto								
4_infpr, 5_infpr	36	infpr	auto								
4_infpr, (5_infpr)	44	infpr	cnt	infpr	time	infpr	own				
6_infpr_a, 6_infpr_b, 7_infpr_b	31	infpr	dir	infpr							
7_infpr_a, 7_infpr_b	26	infpr	dir	own	dir	infpr	infsk	infpr			
8_infpr	7	infpr	partner								
9_infpr_a	12	infpr	time	infpr							
9_infpr_a, 9_infpr_b, 9_infpr_c	30	infpr	time	infpr	time	infpr	time	auto	infpr	turn	infpr
9_infpr_a	46	infpr	time	infpr	time	infpr					
9_infpr_a	48	infpr	time	infpr							
9_infpr_a	14	infpr	time	infpr							
10_infpr	22	infpr	time	infsk	time						
9_infpr_a	28	infpr	time								
11_infpr_a, 11_infpr_b, 11_infpr_c	32	infpr	turn	infpr	auto	time	infpr	auto	infpr		
11_infpr_b	45	infpr	turn	infpr	time						
11_infpr_a, 11_infpr_b, 11_infpr_c	10	infpr	turn								
	8	infpr									
	40	infpr									
	6	infpr									

Name	*ID*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
	43	infpr									
	11	infpr									
	13	infpr									
12_infsk_a_b	23	infsk	auto	infpr							
12_infsk_a_b	47	infsk	turn	infpr	cnt	infpr	time	infpr	auto		
13_time	16	time	infpr	auto	time	infpr	allo	infpr	dir		
13_time	25	time	infpr	dir	infpr	auto	dir				
13_time	27	time	infpr	partner							
13_time	2	time	infpr								
13_time	20	time	infpr								
13_time	35	time	infpr								
13_time	39	time	infpr								
13_time	49	time	infpr								
13_time	15	time	infpr								
14_time	41	time	infsk	infpr	time	infpr	partner	dir	infpr		
15_time	38	time	turn	dir	time						
16_time	36	time	turn	infpr	allo	time	cnt	infpr	allo	time	infpr
	33	time									
18_turn	18	turn	dir	auto							
19_turn	50	turn	infpr	own	infpr	time	infpr	social			
17_turn_a_b, 19_turn	5	turn	infpr	turn							
17_turn_a_b	42	turn	infpr								

Appendix D.3 Suffix generalisation table for speaker 1

M – match; the green colour marks match of 2 or more matching dialogue acts in a sequence from the back. The number in the matching column M stands for the ID of an automaton which was built on the suffix match.

Name	*ID*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*M*
2_auto	32					auto	infpr	dir	cnt	infpr	allo	
15_infpr	24									infpr	allo	
1_auto	44								auto	cnt	auto	
15_infpr	30								infpr	cnt	auto	
4_cnt	49									cnt	auto	
7_time	34								time	infpr	auto	
11_infsk_a, 11_infsk_b	14							infsk	cnt	infsk	auto	
1_auto, 2_auto	6										auto	
1_auto, 2_auto	10										auto	
1_auto	43								auto	cnt	auto	
6_turn	15								turn	infpr	auto	
15_infpr	41	infpr	time	infpr	own	infpr	cnt	infsk	cnt	infpr	cnt	
6_turn	1								turn	infpr	cnt	
15_infpr	4									infpr	cnt	
8_time_a, 8_time_b	5								time	infsk	cnt	
11_infsk_a	2								infsk	own	cnt	
4_cnt	36										cnt	
4_cnt	40										cnt	
4_cnt	42										cnt	
3_auto	37						auto	infsk	cnt	auto	infpr	
2_auto	11									auto	infpr	
2_auto	3									auto	infpr	

Name	*ID*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*M*
4_cnt	39									cnt	infpr	
5_turn	50					turn	infsk	infpr	cnt	infsk	infpr	
5_turn	51								turn	infsk	infpr	
6_turn	17									turn	infpr	
6_turn	20									turn	infpr	
14_infpr, 15_infpr	9										infpr	
	35										infpr	
2_auto	12									auto	infpr	
1_auto	13								auto	cnt	infpr	
14_infpr	19						infpr	auto	infpr	cnt	infpr	
6_turn	16									turn	infpr	
14_infpr	18						infpr	dir	infpr	auto	infsk	
15_infpr	48								infpr	cnt	infsk	
2_auto	31								auto	infpr	infsk	
8_time_a, 8_time_b	25				time	infsk	cnt	auto	cnt	infpr	infsk	
7_time	22								time	infpr	infsk	
7_time	23						time	infpr	time	infpr	infsk	
6_turn	29								turn	infpr	infsk	
12_infsk	45		infsk	cnt	partner	cnt	infpr	cnt	auto	turn	infsk	
11_infsk_a, 11_infsk_b, 12_infsk	7										infsk	
6_turn	47				turn	infpr	auto	infsk	time	infpr	infsk	
10_time	28					time	auto	infsk	auto	turn	infsk	
9_partner	33										partner	
9_partner	38										partner	
16_infpr	46							infpr	infsk	auto	turn	
15_infpr	21									infpr	turn	
11_infsk_a, 11_infsk_b	8							infsk	auto	infsk	turn	

Appendix D.4 Suffix generalisation table for speaker 2

M – match; the green colour marks match of 2 or more matching dialogue acts in a sequence from the back. The number in the matching column M stands for the ID of an automaton which was built on the suffix match.

Name	*ID*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*M*
4_infpr, 5_infpr	3									infpr	auto	
4_infpr, 5_infpr	36									infpr	auto	
18_turn	18								turn	dir	auto	
12_infsk_a_b	47			infsk	turn	infpr	cnt	infpr	time	infpr	auto	
13_time	25					time	infpr	dir	infpr	auto	dir	
1_auto	24									auto	dir	
13_time	16			time	infpr	auto	time	infpr	allo	infpr	dir	
4_infpr, 5_infpr	9					infpr	auto	infpr	time	infpr	dir	
2_dir, 3_dir	17										dir	
1_auto	1									auto	infpr	

Name	*ID*	*10*	*9*	*8*	*7*	*6*	*5*	*4*	*3*	*2*	*1*	*M*
2_dir	19							dir	auto	cnt	infpr	
6_infpr_a, 6_infpr_b, 7_infpr_b	31								infpr	dir	infpr	
14_time	41			time	infsk	infpr	time	infpr	partner	dir	infpr	
1_auto	34							auto	infpr	infsk	infpr	1
7_infpr_a, 7_infpr_b	26				infpr	dir	own	dir	infpr	infsk	infpr	1
16_time	36	time	turn	infpr	allo	time	cnt	infpr	allo	time	infpr	
2_dir, 3_dir	21						dir	auto	infpr	time	infpr	2
9_infpr_a	12								infpr	time	infpr	2
9_infpr_a	46						infpr	time	infpr	time	infpr	2
9_infpr_a	48								infpr	time	infpr	2
9_infpr_a	14								infpr	time	infpr	2
	2									time	infpr	2
	20									time	infpr	2
	35									time	infpr	2
	39									time	infpr	2
	49									time	infpr	2
	15									time	infpr	2
9_infpr_a, 9_infpr_b, 9_infpr_c	30	infpr	time	infpr	time	infpr	time	auto	infpr	turn	infpr	2?
	8										infpr	
	40										infpr	
	6										infpr	
	43										infpr	
	11										infpr	
	13										infpr	
11_inppr_a, 11_infpr_b, 11_infpr_c	32			infpr	turn	infpr	auto	time	infpr	auto	infpr	
12_infsk_a_b	23								infsk	auto	infpr	
4_infpr, 5_infpr	4						infpr	auto	infpr	time	infpr	2
17_turn_a_b	42									turn	infpr	
4_infpr, 5_infpr	44					infpr	cnt	infpr	time	infpr	own	
13_time	27								time	infpr	partner	
8_infpr	7									infpr	partner	
19_turn	50				turn	infpr	own	infpr	time	infpr	social	
15_time	38							time	turn	dir	time	
9_infpr_a	28									infpr	time	
10_infpr	22							infpr	time	infsk	time	
	33										time	
11_infpr_b	45							infpr	turn	infpr	time	
11_infpr_a, 11_infpr_b, 11_infpr_c	10									infpr	turn	
17_turn_a_b, 19_turn	5								turn	infpr	turn	

Appendix E Semi-coupled automata for speaker 1 and speaker 2

Turn automata made by combining dialogue act automata for speaker 1 (spk1) and speaker 2 (spk2). S stands for the initial state. The node more to the left shows which speaker starts the sequence. If the initial nodes are one under the other, then it means that the speaker on the top starts. The dotted arrows show the transition of turns between the speakers. More than one dialogue act on an arc means that the utterance in the original dialogue had more than one communicative function.

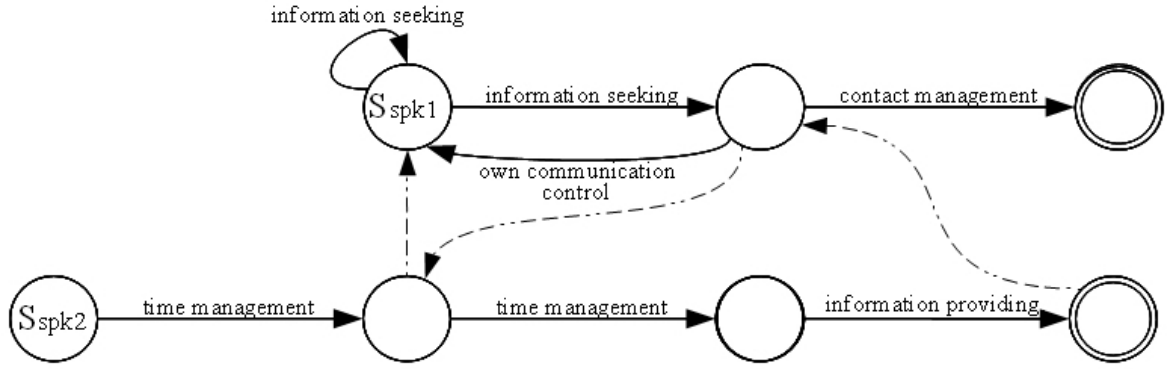


Figure 63: Semi-coupled automaton 2

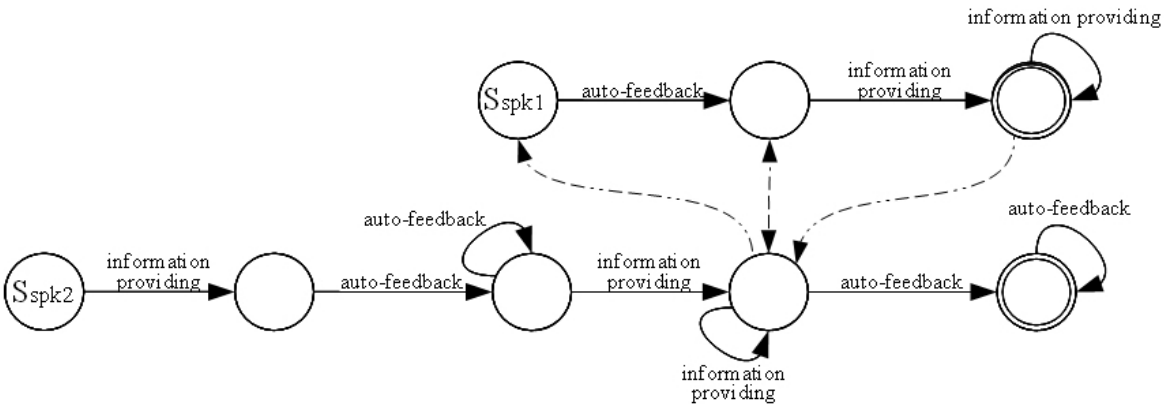


Figure 64: Semi-coupled automaton 3

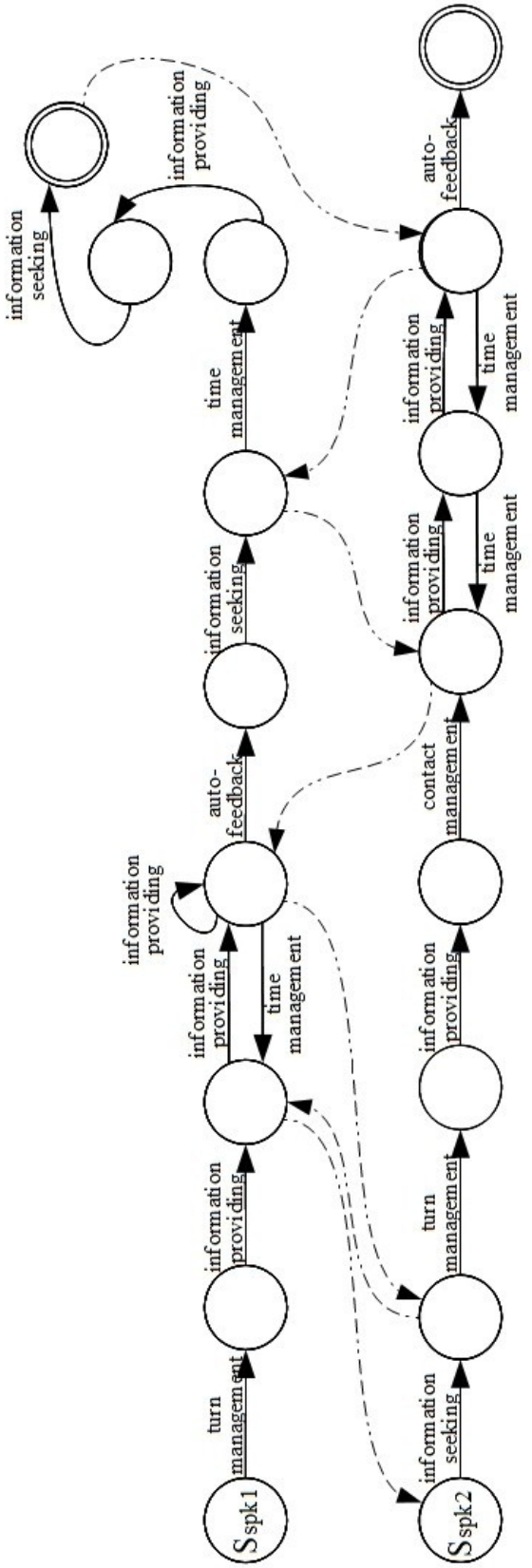
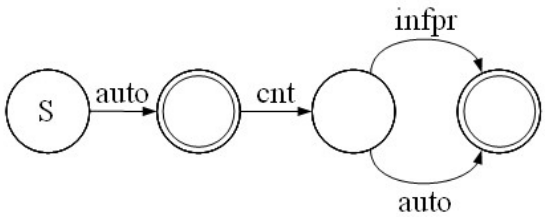


Figure 65: Semi-coupled automata 4

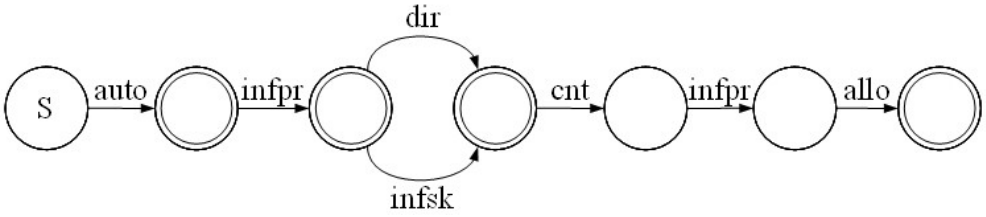
Appendix F Loop-free automata

Appendix F.1 Loop-free automata for speaker 1

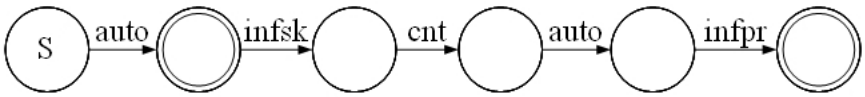
1_auto:



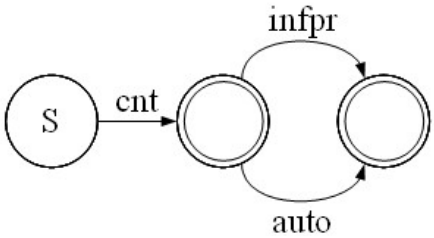
2_auto:



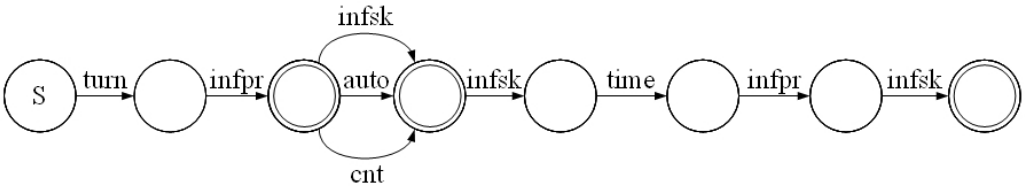
3_auto:



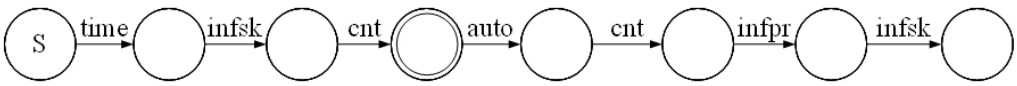
4_cnt:



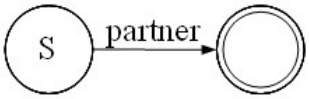
6_turn:



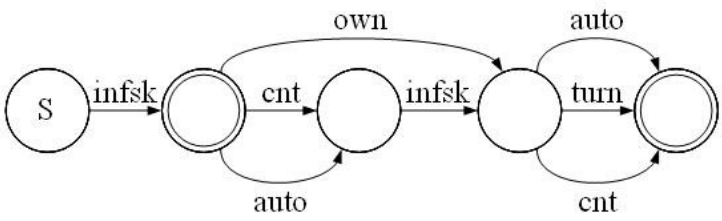
8_time:



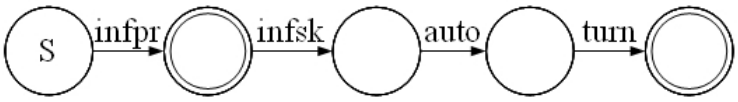
9_partner:



11_infsk:

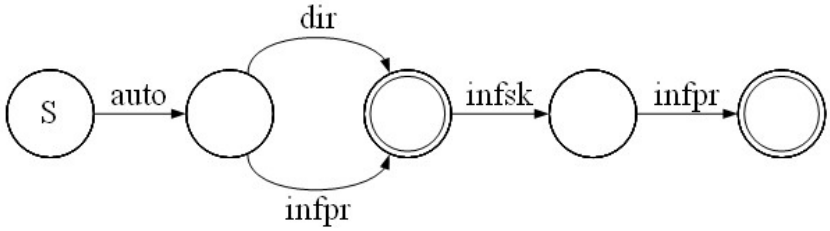


15_infpr:

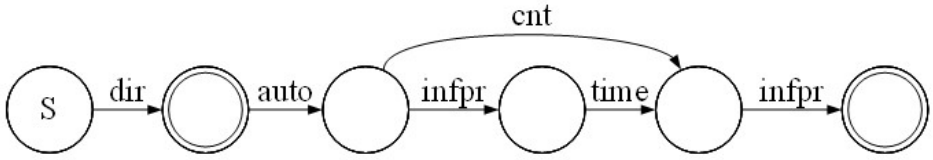


Appendix F.2 Loop free automata for speaker 2

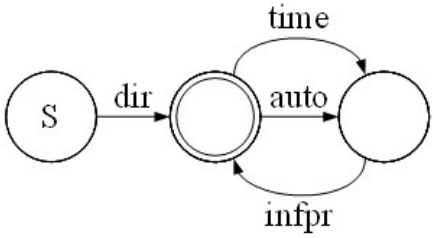
1_auto:



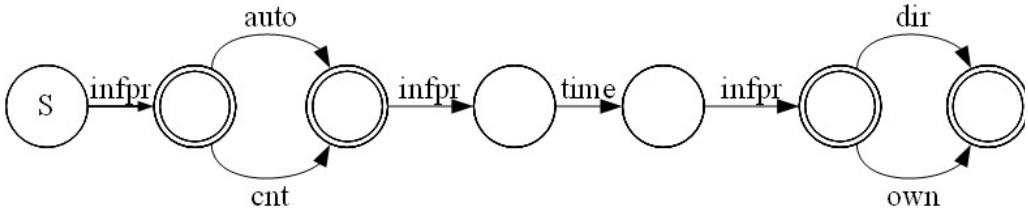
2_dir:



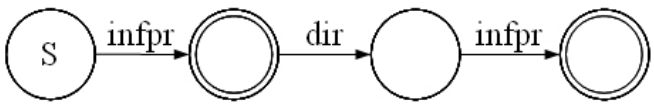
3_dir:



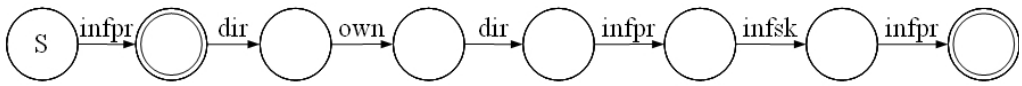
4_infpr:



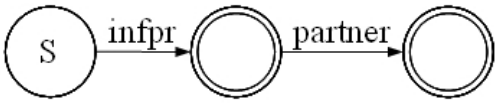
6_infpr_a:



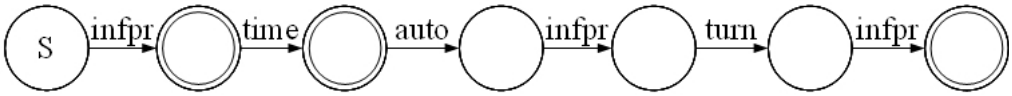
7_infpr_a:



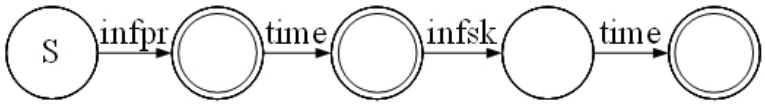
8_infpr:



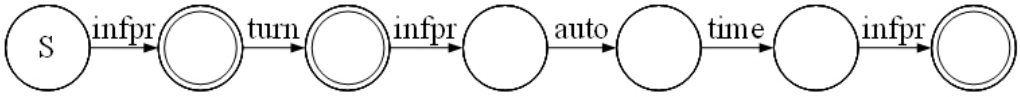
9_infpr_b:



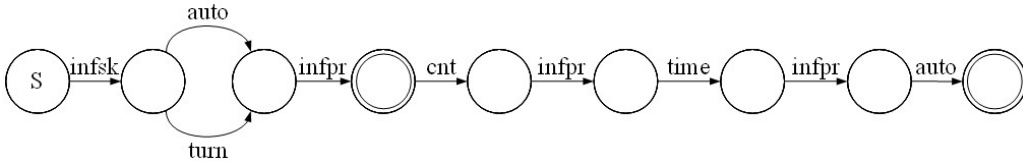
10_infpr:



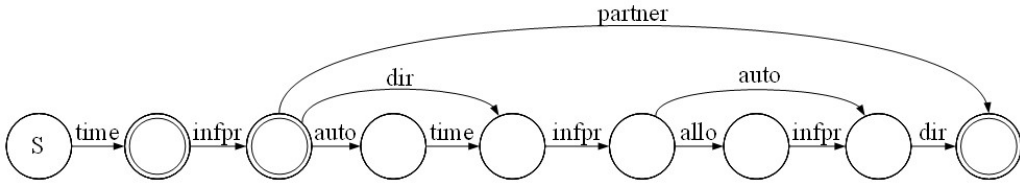
11_infpr_a:



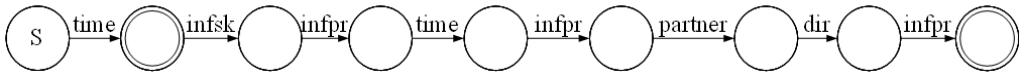
12_infsk_a:



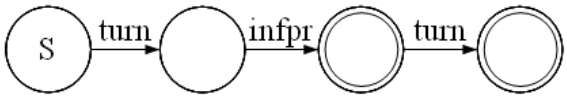
13_time:



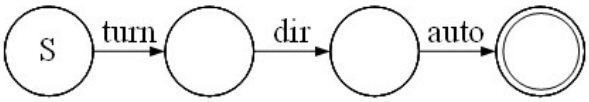
14_time:



17_turn_a:



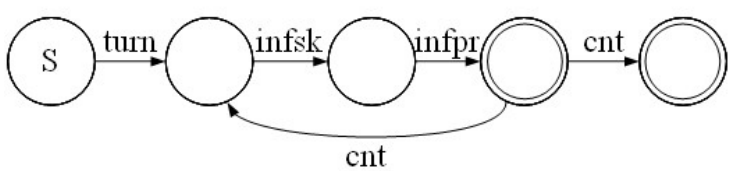
18_turn:



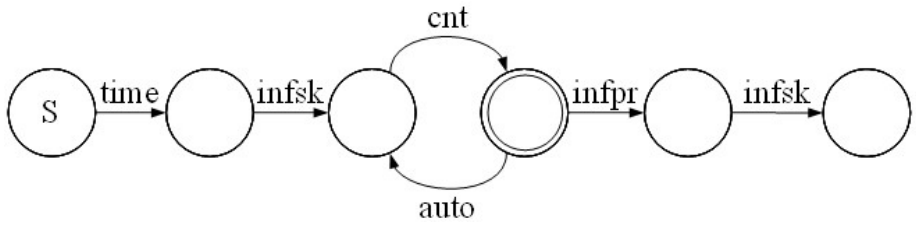
Appendix G Iterative automata

Appendix G.1 Iterative automata for speaker 1

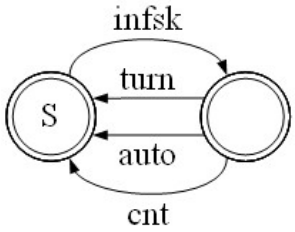
5_turn:



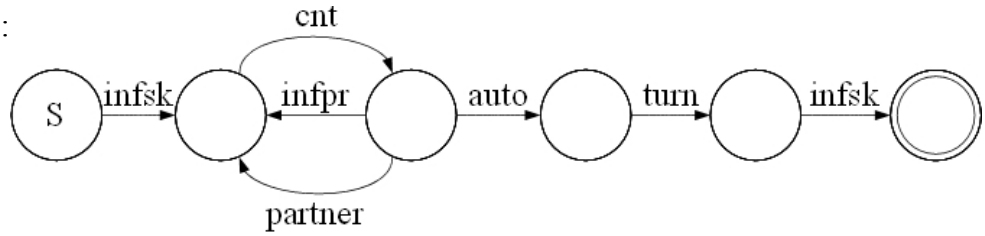
8_time_b:



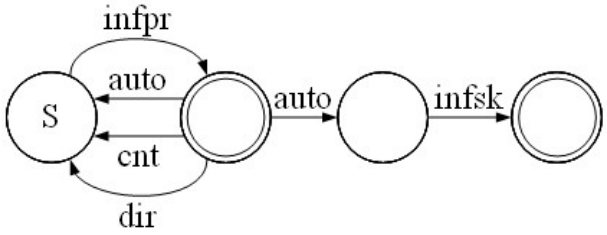
11_infsk_b:



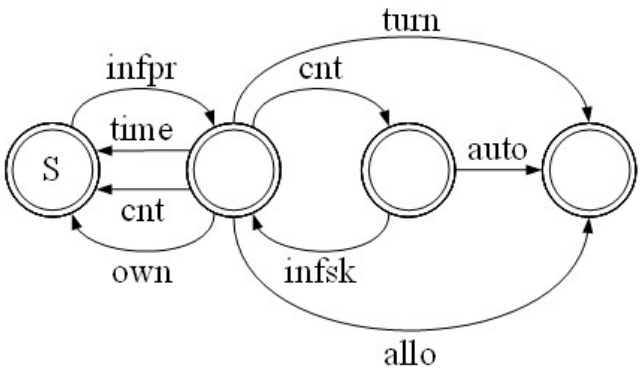
12_infsk_b:



13_infpr:

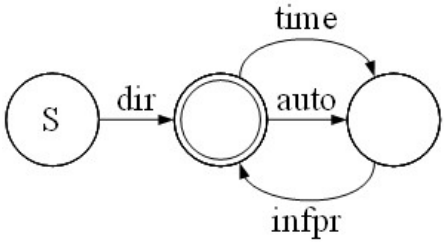


14_infpr:

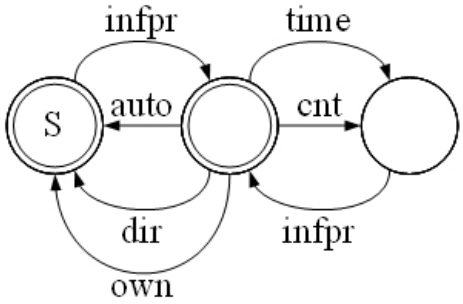


Appendix G.2 Iterative automata for speaker 2

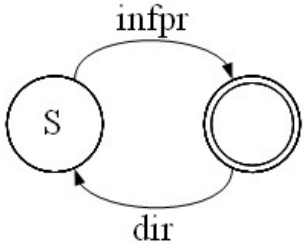
3_dir:



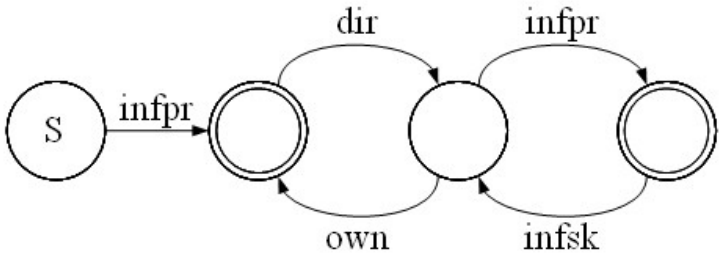
5_infpr:



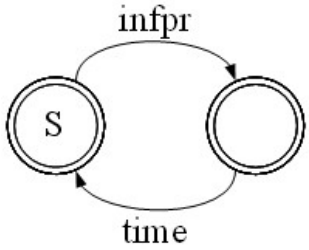
6_infpr_b:



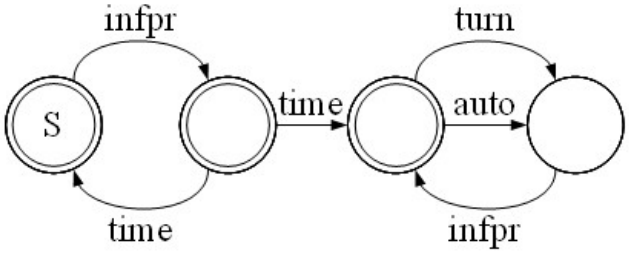
7_infpr_b:



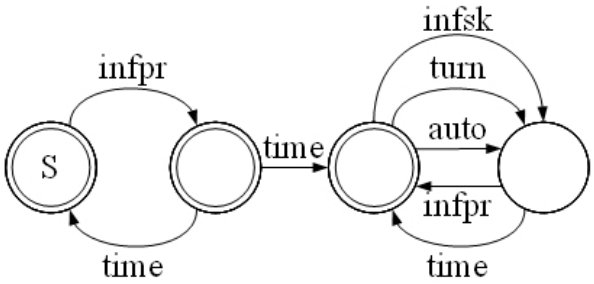
9_infpr_a:



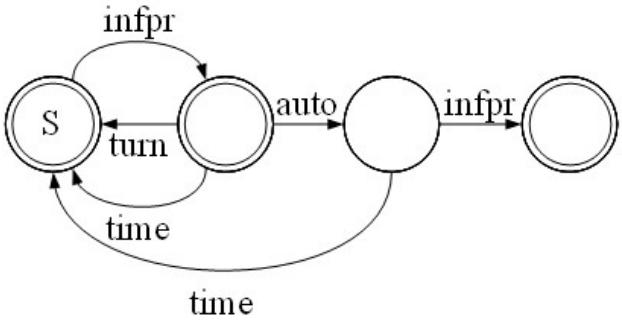
9_infpr_c:



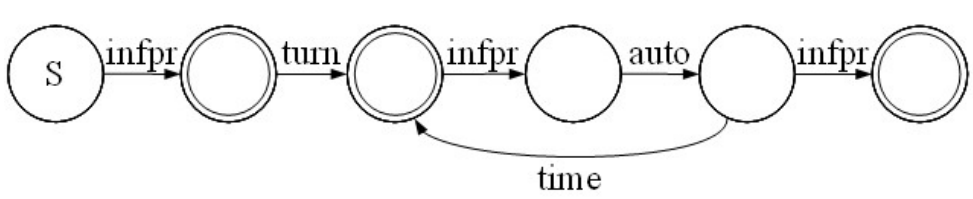
10_infpr_
new_9+10:



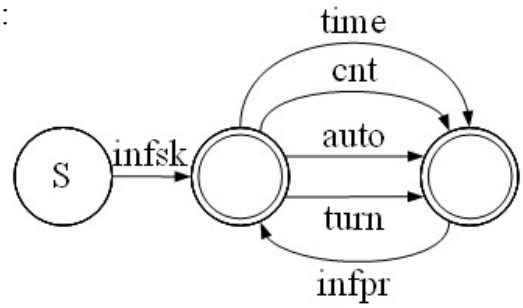
11_infpr_b:



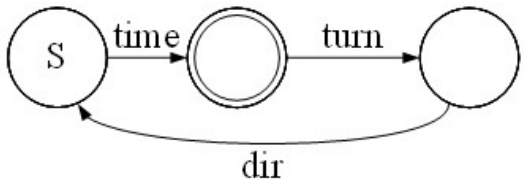
11_infpr_c:



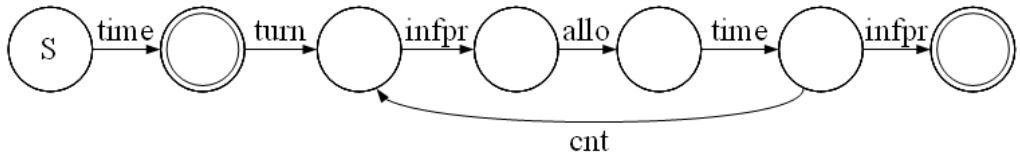
12_infpr_b:



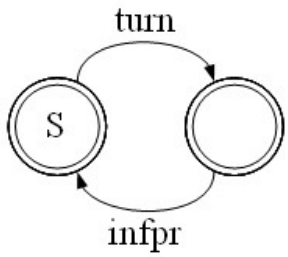
15_time:



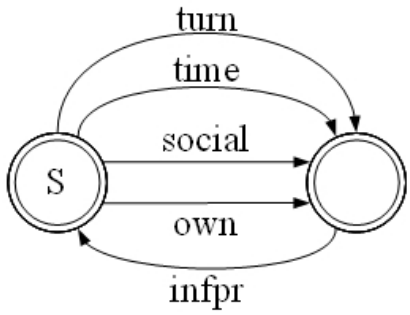
16_time:



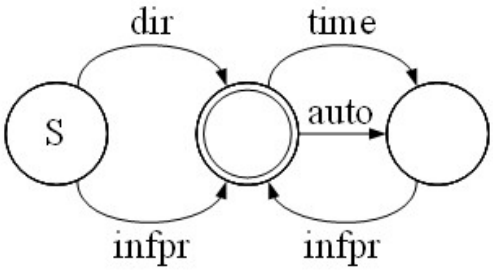
17_turn_b:



19_turn:



back1_
3_dir+infpr



Appendix G.3 Generalised automata for speaker 1

S stands for the initial state. More than one dialogue act on an arc means that the utterance in the original dialogue had more than one communicative function.

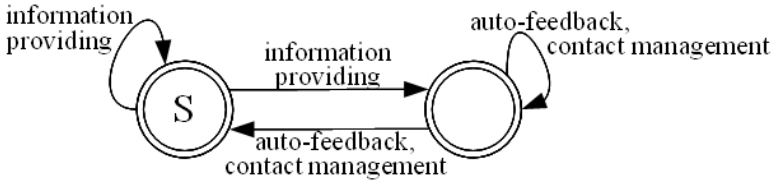


Figure 66: Generalised automaton 1 for speaker 1.

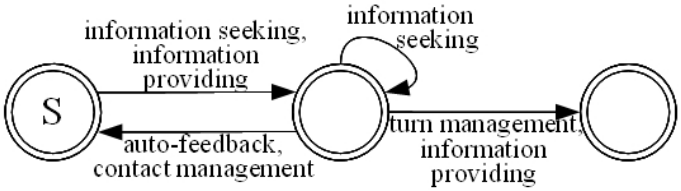


Figure 67: Generalised automaton 2 for speaker 1.

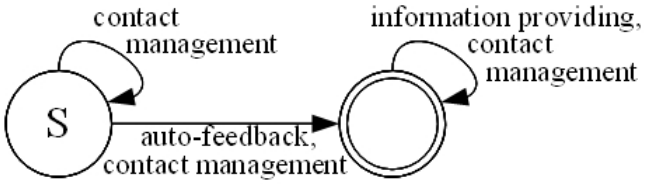


Figure 68: Generalised automaton 3 for speaker 1.

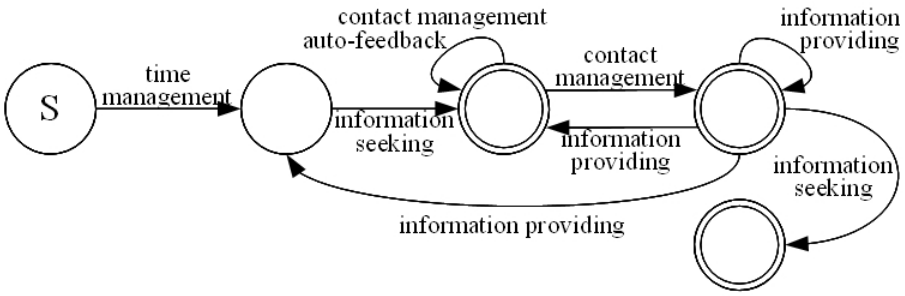


Figure 69: Generalised automaton 4 for speaker 1.

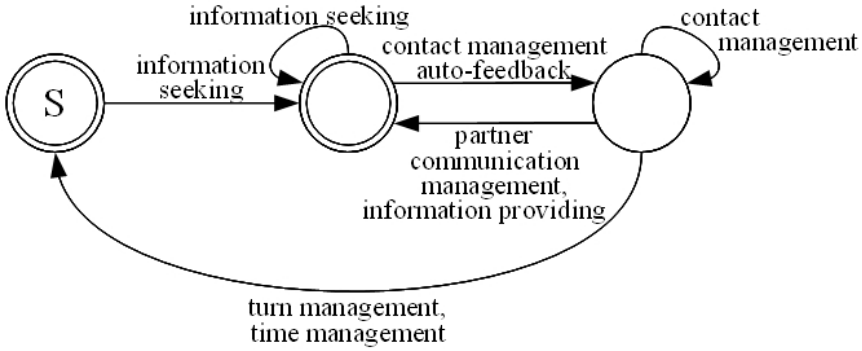


Figure 70: Generalised automaton 5 for speaker 1.

Appendix H Automata evaluation

Automata evaluation performed using the NDFST online tool. F – figure on which the automaton is presented, Autom – type of automaton and information about the speaker, S – initial state, T – terminal state. The target output sequence, i.e. the original sequence of utterances which occurred in the original dialogue is marked in italics. Table 61 shows the explanation of abbreviations of dialogue act types.

Table 61: Explenation of abbrieviations of dialogue act types.

infsk	information seeking
infpr	information providing
dir	directives
autof	auto-feedback
opm	open meeting
turn	turn management
cnt	contact management
time	time management
allo	allo-feedback
own	own communication control
infsk_own	info seeking, own communication control

Appendix H.1 Generalised automata

F	***Autom***	***Sequence***	***S***	***T***	***Transition quadruples***	***Output***
Figure 66	generalised 1, spk1	x x x	q0	q0, q1	q0, x, infpr, q0; q0, x, infpr, q1; q1, x, auto, q0; q1, x, auto, q0; q1, x, cnt, q0; q1, x, cnt, q0	infpr infpr infpr infpr infpr infpr infpr infpr auto infpr infpr auto infpr infpr cnt infpr infpr cnt infpr auto infpr infpr auto infpr infpr auto infpr infpr auto infpr infpr cnt infpr infpr cnt infpr infpr cnt infpr infpr cnt infpr
Figure 67	generalised 2, skp1	x x x x x	q0	q0, q1, q2	q0, x, infsk, q1; q1, x, infsk, q1; q1, x, auto, q0; q1, x, cnt, q0; q1, x, turn, q2; q1, x, infpr, q2	infsk auto infsk auto infsk infsk auto infsk cnt infsk infsk auto infsk infsk auto infsk cnt infsk infsk cnt infsk cnt infsk infsk infpr infsk cnt infsk infsk infsk infsk infsk auto infsk turn infsk infsk cnt infsk auto infsk infsk cnt infsk cnt infsk infsk infsk infsk infpr infsk infsk infsk infsk infsk infsk infsk infsk infsk turn ...

F	*Autom*	*Sequence*	*S*	*T*	*Transition quadruples*	*Output*
Figure 68	generalised 3, spk1	x x x x	q0	q1	q0, x, cnt, q0; q0, x, cnt, q1; q0, x, auto, q1; q1, x, infpr, q1	auto infpr infpr infpr cnt auto infpr infpr cnt cnt auto infpr cnt cnt cnt auto cnt cnt cnt cnt cnt cnt cnt infpr cnt cnt infpr infpr cnt infpr infpr infpr
Figure 69	generalised 4, spk1	time infsk cnt cnt infpr infpr infsk cnt infpr infpr cnt infpr infpr infsk	q0	q2, q3, q4	q0, time, naaa, q1; q1, infsk, czyli_tak_w_dol_po_skosie, q2 ; q1, infsk, na_wschod?, q2; q1, infsk, w_poziomym_kierunku?, q2; q2, cnt, dobra, q3 ; q2, cnt, tak?, q3 ; q3, cnt, dobra, q3; q3, infpr, bo_ja_tu_mam_gory, q1 ; q3, infpr, bo_ja_tu_mam_gory, q4 ; q3, infpr, dwa_centymetry, q4; q3, infpr, to_ja_tu_nie_moge, q3 ; q4, cnt, no_dobra, q3 ; q4, infpr, na_poludniowy_wschod, q1 ; q4, infpr, na_poludniowy_wschod, q4; q3, infpr, jestem_przy_gorach, q4; q4, infpr, to_ja_mam_gory, q4; q4, infpr, pod_gorami, q1; q1, infsk, to_ja_mam_przejsc_te_gory_z_lewej_stro ny_czy_z_prawej, q2	*naaa na_wschod? tak? dobra* *dwa_centymetry* *na_poludniowy_wschod* *czyli_tak_w_dol_po_skosie tak?* *to_ja_tu_nie_moge* *bo_ja_tu_mam_gory no_dobra* *jestem_przy_gorach* *to_ja_mam_gory pod_gorami* *to_ja_mam_przejsc_te_gory_z_lewej* *_strony_czy_z_prawej* … (70MB)
	generalised 4, spk1, reduced loops	time infsk cnt cnt infpr infpr infsk cnt infpr infpr cnt infpr infpr infsk	q0	q2, q3, q4	q0, time, naaa, q1; q1, infsk, czyli_tak_w_dol_po_skosie, q2 ; q1, infk, na_wschod?, q2; q2, cnt, tak?, q3 ; q3, cnt, dobra, q3; q3, infpr, bo_ja_tu_mam_gory, q4 ; q3, infpr, dwa_centymetry, q4; q3, infpr, to_ja_tu_nie_moge, q3 ; q4, cnt, no_dobra, q3 ; q4, infpr, na_poludniowy_wschod, q1 ; q3, infpr, jestem_przy_gorach, q4; q4, infpr, to_ja_mam_gory, q4; q4, infpr, pod_gorami, q1; q1, infsk, to_ja_mam_przejsc_te_gory_z_lewej_stron y_czy_z_prawej, q2	naaa czyli_tak_w_dol_po_skosie tak? dobra bo_ja_tu_mam_gory na_poludniowy_wschod czyli_tak_w_dol_po_skosie tak? jestem_przy_gorach to_ja_mam_gory no_dobra bo_ja_tu_mam_gory to_ja_mam_gory pod_gorami czyli_tak_w_dol_po_skosie *naaa na_wschod? tak? dobra* *dwa_centymetry* *na_poludniowy_wschod* *czyli_tak_w_dol_po_skosie tak?* *to_ja_tu_nie_moge* *bo_ja_tu_mam_gory no_dobra* *jestem_przy_gorach* *to_ja_mam_gory pod_gorami* *to_ja_mam_przejsc_te_gory_z_lewej* *_strony_czy_z_prawej* naaa czyli_tak_w_dol_po_skosie tak? dobra bo_ja_tu_mam_gory pod_gorami czyli_tak_w_dol_po_skosie tak? dwa_centymetry to_ja_mam_gory no_dobra to_ja_tu_nie_moge jestem_przy_gorach pod_gorami czyli_tak_w_dol_po_skosie
						naaa na_wschod? tak? dobra

F	*Autom*	*Sequence*	*S*	*T*	*Transition quadruples*	*Output*
						bo_ja_tu_mam_gory pod_gorami to_ja_mam_przejsc_te_gory_z_lewej _strony_czy_z_prawej tak? to_ja_tu_nie_moge dwa_centymetry no_dobra to_ja_tu_nie_moge bo_ja_tu_mam_gory na_poludniowy_wschod na_wschod?

Appendix H.2 Semi-coupled automata

F	*Autom*	*Sequence*	*S*	*T*	*Transition quadruples*	*Output*
Figure 21	semi-coupled 1, spk1	turn infpr cnt	q0	q3	q0, turn, czyli_idziemy_od_lewego_dolnego_narożnika, q1; q1, infpr, do_prawego_gornego, q2; q2, cnt, tak, q3	*czyli_idziemy_od_lewego_dolnego_narożnika do_prawego_gornego tak*
	semi-coupled 1, spk2	auto infpr	q0	q2	q0, auto, tak, q1; q1, infpr, wychodzimy_z_zamku, q2	*tak wychodzimy_z_zamku*
Figure 63	semi-coupled 2, spk1	infsk infsk infsk_own infsk cnt	q0	q2	q0, infsk, a_mamy_wejsc_tam_na_gorze, q0; q0, infsk, do_je, q1; q1, infsk_own, do_tego_budynku_po_prostu, q0; q0, infsk, obojetnie_od_ktorej_strony, q1; q1, cnt, tak, q2	a_mamy_wejsc_tam_na_gorze do_je do_tego_budynku_po_prostu do_je tak *a_mamy_wejsc_tam_na_gorze do_je do_tego_budynku_po_prostu obojetnie_od_ktorej_strony tak* a_mamy_wejsc_tam_na_gorze obojetnie_od_ktorej_strony do_tego_budynku_po_prostu do_je tak a_mamy_wejsc_tam_na_gorze obojetnie_od_ktorej_strony do_tego_budynku_po_prostu obojetnie_od_ktorej_strony tak
	semi-coupled 2, spk2	time time infpr	q0	q2	q0, time, filled_pause, q1; q0, time, filled_pause, q0; q1, infpr, do_zamku, q2	*filled_pause filled_pause do_zamku*
Figure 64	semi-coupled 3, spk1	auto infpr infpr	q0	q2	q0, auto, mostu_zwodzonego, q1; q1, infpr, ale_cos_tam_jest, q2; q2, infpr, takie_moze_no, q2	*mostu_zwodzonego ale_cos_tam_jest takie_moze_no*
	semi-coupled 3, spk2	infpr auto auto auto infpr infpr infpr auto auto	q0	q4	q0, infpr, tez_musimy, q1; q1, auto, tak, q2; q2, auto, tak, q2; q2, infpr, od_strony_fosy_tutaj, q3; q3, infpr, mostu_zwodzonego, q3; q3, infpr, no_tak_taka_malutka_no, q3; q3, auto, tak, q4; q4, auto, tak, q4	tez_musimy tak tak tak od_strony_fosy_tutaj mostu_zwodzonego mostu_zwodzonego tak tak *tez_musimy tak tak tak od_strony_fosy_tutaj mostu_zwodzonego no_tak_taka_malutka_no tak tak* tez_musimy tak tak tak od_strony_fosy_tutaj no_tak_taka_malutka_no mostu_zwodzonego tak tak
						tez_musimy tak tak tak od_strony_fosy_tutaj no_tak_taka_malutka_no no_tak_taka_malutka_no tak tak

F	*Autom*	*Sequence*	*S*	*T*	*Transition quadruples*	*Output*
Figure 65	semi-coupled 4, spk1	turn infpr infpr time infpr infpr auto infsk time infpr infsk	q0	q8	q0, turn, tak, q1; q1, infpr, ale_tu_z_tylu_ktos_stoi, q2; q2, infpr, no_tam_ke_ja_mijam_kepe_tam_ktos_stoi_ale, q3; q3, time, ale, q2; q2, infpr, bo_jest_taki_domek_tam, q3; q3, infpr, na_prawo_od_tej_kepy, q3; q3, auto, niebieskiego, q4; q4, infsk, w_domku, q5; q5, time, filled_pause, q6; q6, infpr, takie_cos_tam_stoi, q7; q7, infsk, nie_wiem_czy_to_jest_facet_czy, q8	tak ale_tu_z_tylu_ktos_stoi bo_jest_taki_domek_tam ale bo_jest_taki_domek_tam na_prawo_od_tej_kepy niebieskiego w_domku filled_pause takie_cos_tam_stoi nie_wiem_czy_to_jest_facet_czy tak ale_tu_z_tylu_ktos_stoi bo_jest_taki_domek_tam ale no_tam_ke_ja_mijam_kepe_tam_ktos_stoi_ale na_prawo_od_tej_kepy niebieskiego w_domku filled_pause takie_cos_tam_stoi nie_wiem_czy_to_jest_facet_czy *tak ale_tu_z_tylu_ktos_stoi* *no_tam_ke_ja_mijam_kepe_tam_ktos_stoi_ale ale bo_jest_taki_domek_tam* *na_prawo_od_tej_kepy niebieskiego* *w_domku filled_pause* *takie_cos_tam_stoi* *nie_wiem_czy_to_jest_facet_czy* tak ale_tu_z_tylu_ktos_stoi no_tam_ke_ja_mijam_kepe_tam_ktos_stoi_ale ale no_tam_ke_ja_mijam_kepe_tam_ktos_stoi_ale na_prawo_od_tej_kepy niebieskiego w_domku filled_pause takie_cos_tam_stoi nie_wiem_czy_to_jest_facet_czy
	semi-coupled 4, spk2	infsk turn infpr cnt infpr infpr time time infpr infpr auto	q0	q7	q0, infsk, gdzie_smiech, q1; q1, turn, tak, q2; q2, infpr, i_tam_pause_pause_jest_cos_takiego_niebieskiego, q3; q3, cnt, tak, q4; q4, infpr, nie, q5; q5, infpr, przed_domkiem, q6; q6, time, filled_pause, q5; q5, time, filled_pause, q4; q4, infpr, pomiedzy_domkiem_a_kepa, q5; q5, infpr, na_naszej_sciezce, q6; q6, auto, dobrze, q7	gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie na_naszej_sciezce filled_pause filled_pause nie na_naszej_sciezce dobrze gdzie_smiech tak i_tam_pause_ _pause_jest_cos_takiego_niebieskiego tak nie na_naszej_sciezce filled_pause filled_pause nie przed_domkiem dobrze gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie na_naszej_sciezce filled_pause filled_pause pomiedzy_domkiem_a_kepa na_naszej_sciezce dobrze gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie na_naszej_sciezce filled_pause filled_pause pomiedzy_domkiem_a_kepa przed_domkiem dobrze gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie przed_domkiem filled_pause filled_pause nie na_naszej_sciezce dobrze

F	*Autom*	*Sequence*	*S*	*T*	*Transition quadruples*	*Output*
						gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie przed_domkiem filled_pause filled_pause nie przed_domkiem dobrze *gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie przed_domkiem filled_pause filled_pause pomiedzy_domkiem_a_kepa na_naszej_sciezce dobrze* gdzie_smiech tak i_tam_pause_pause_jest_cos_takiego_niebieskiego tak nie przed_domkiem filled_pause filled_pause pomiedzy_domkiem_a_kepa przed_domkiem dobrze … (8 other ouputs)

Appendix I Phonetically rich sentence extractor

```
#!/usr/bin/python
# Jolanta Bachan
# PhonRichDiphoneSentenceCollector_complex10.py
# 2010-10-13

#-------------------------------------------------
# Import modules
import re, operator

#-------------------------------------------------
# Definitions of global variables

#transcriptionfile = "allBOSStranscriptions.txt"
transcriptionfile = "ALL_BossJur_trans.txt"
textfile = "ALL_BossJur_text.txt"
#textfile = "allBOSStexts.txt"
#transcriptionfile = "czytaneJurisdictranscription.txt"
outputfile = "richsentences.txt"
outputdiphonefile = "diphonesinrichsentences.txt"
##out2 = "out2.txt"
outputfileorthography = "orthographicrichsentences.txt"
alldiphones = set()
diphone = ""
diphonearrays = []

#-------------------------------------------------
# Read in all transcriptions
def readintranscriptions(transcriptionfile):
    transcriptionfilehandle = open(transcriptionfile, 'r')
    alltranscriptions = transcriptionfilehandle.readlines()
    transcriptionfilehandle.close()
    return alltranscriptions

#-------------------------------------------------
# Read in all texts
def readintexts(textfile):
    textfilehandle = open(textfile, 'r')
```

```
    alltexts = textfilehandle.readlines()
    textfilehandle.close()
    return alltexts

#-------------------------------------------------
# Combine orthography with transcription
def orthtrans(allsentences, orthographicsentences):
    orthtranssentences = []
    i = 0
    for sentence in allsentences:
        sentencecombine = []
        sentencecombine.append(i)
        sentencecombine.append(sentence)
        sentencecombine.append(orthographicsentences[i])
        orthtranssentences.append(sentencecombine)
        i = i+1
    return orthtranssentences

#-------------------------------------------------
# Divide senteces into diphones
def divideintodiphones (orthtranssentences):
    sentencediphones = []
    allsentencediphones = []
    gettranscription = operator.itemgetter(1)
    transcribedsentences = map(gettranscription,
orthtranssentences)
    for sentence in transcribedsentences:
        sentence = re.sub('\r', '', sentence)
        sentence = re.sub('\n', '', sentence)
        sentence = re.sub('^', '_', sentence)
        sentence = re.sub('$', '_', sentence)
        sentence = re.sub('##', '_', sentence)
        sentence = re.sub('#', '', sentence)
        sentence = re.sub('t\^s\'', '7', sentence)
        sentence = re.sub('d\^z\'', '8', sentence)
        sentence = re.sub('t\^s', '3', sentence)
        sentence = re.sub('t\^S', '4', sentence)
        sentence = re.sub('d\^z', '5', sentence)
```

```
            sentence = re.sub('d\^Z', '6', sentence)
            sentence = re.sub('z\'', '9', sentence)
            sentence = re.sub('n\'', '1', sentence)
            sentence = re.sub('s\'', '2', sentence)
            sentence = re.sub('w\~', 'Q', sentence)
            sentence = re.sub('j\~', 'W', sentence)
            a = len(sentence)
            i = 1
            while i-1 < a:
                for phone in sentence:
                    diphone = sentence[i-2] + '-' + sentence[i-1]
                    sentencediphones.append(diphone)
                    i = i+1
            sentencediphones.pop(0)
            sentencediphones.append(sentence)
            allsentencediphones.append(sentencediphones)
            sentencediphones = []
        j = 0
        for items in orthtranssentences:
            items.append(allsentencediphones[j])
            j = j + 1
        return orthtranssentences

#-----------------------------------------------------
# Sort sentences according to the number of diphones they
contain
def sortdiphones(orthtranssentenceswithdiphones):
    allsorteddiphones = []
    orthtranssentences = []
    getdiphones = operator.itemgetter(3)
    repeateddiphones = map(getdiphones,
orthtranssentenceswithdiphones)
    i = 0
    for diphones in repeateddiphones:
    #    print diphones
        diphoneset = set(diphones)
        a = len(diphoneset)
        orthtranssentenceswithdiphones[i].append(diphoneset)
```

```
        i = i + 1
    getsorted = operator.itemgetter(4)
    sorteddiphones = map(getsorted,
orthtranssentenceswithdiphones)
    sorteddiphones = sorted(sorteddiphones, key=len)
    sorteddiphones.reverse()
    for diphoneset in sorteddiphones:
         for items in orthtranssentenceswithdiphones:
             if diphoneset == items[4]:
                 orthtranssentences.append(items)
    return orthtranssentences

#-------------------------------------------------
# Select diphone rich sentences:
# if the sencence cointains 5 or more new diphones which
#  have not yet appeared in the previous sentences then
#     add the sentence to the diphone rich sentences
# after adding the sentences with more than 5 new diphones,
# check if in the other sentences are 4 new diphones, 3, 2, 1
# add the sentences to the rich sentences
def selectdiphones(orthtranssentencessorted,alldiphones,
diphonearrays):
    # Define the number of new diphones in the selected
sentences
    j = 5
    diphonearray = []
    diphonearrays = []
    richsentences = []
    sentences = []
    newdiphoneset = []
    copyarray = []
    orthographicsentences = []
    sentence = ''
    getdiphones = operator.itemgetter(4)
    sorteddiphones = map(getdiphones, orthtranssentencessorted)
    for item in sorteddiphones:
        diphonearray = list(item)
        diphonearrays.append(diphonearray)
```

```
    # 0 means select sentences which contain new diphones from
j to 1 new diphone
    while j>0:
        i = 0
        for diphones in diphonearrays:

            old = len(alldiphones)
            oldlist = list(alldiphones)
            copydiphones = list(diphones)
            copyarray.append(copydiphones)
            for a in copydiphones:
                if not '-' in a:
                    sentence = a
                    diphones.remove(a)
                    sentences.append(a)

            newdiphoneset = compare(diphones, alldiphones)
            new = len(newdiphoneset)
            if new > old:
                difference = new - old
                if difference >= j:
                    richsentences.append(sentence)
                    getorthography = operator.itemgetter(2)
                    orth = map(getorthography,
orthtranssentencessorted)
                    orthographicsentences.append(orth[i])
                else:
                    oldset = set(oldlist)
                    newdiphoneset = oldset
                    alldiphones = newdiphoneset
            i = i + 1
        j=j-1
        diphonearrays = list(copyarray)
        copyarray = []
    return richsentences, orthographicsentences, newdiphoneset

#-------------------------------------------------
```

```
# Compare set of alldiphones with the diphones from the new
sentence
def compare(diphones, alldiphones):
    diphoneset = set(diphones)
    alldiphones.update(diphoneset)
    return alldiphones

#--------------------------------------------------
# Print diphone rich sentences to an output file
def printrichsentences (selectedsentences, diphone):
    outputfilehandle = open(outputfile, 'w')
    for diphone in selectedsentences:
        sentence = complexdiphones(diphone)
        line = sentence + '\n'
        outputfilehandle.write(line)
    outputfilehandle.close()

#--------------------------------------------------
# Print diphone rich sentences to an output file in orthography
def
printrichsentencesorthography(selectedsentencesorthography):
    outputfilehandleorthography = open(outputfileorthography,
'w')
    for sentence in selectedsentencesorthography:
      #  line = sentence + '\n'
        outputfilehandleorthography.write(sentence)
    outputfilehandleorthography.close()

#--------------------------------------------------
# Print diphones from rich sentences to an output file
def printdiphones(noduplicatediphones, diphone):
    noduplicatediphones = list(noduplicatediphones)
    noduplicatediphones = sorted(noduplicatediphones)
    outputdiphonefilehandle = open(outputdiphonefile, 'w')
    for diphone in noduplicatediphones:
        diphone = complexdiphones(diphone)
        line = diphone + '\n'
        outputdiphonefilehandle.write(line)
```

```
    outputdiphonefilehandle.close()

#-------------------------------------------------
# Convert simple characters back to complex SAMPA charachters
def complexdiphones(diphone):
        diphone = re.sub('7', 't^s\'', diphone)
        diphone = re.sub('8', 'd^z\'', diphone)
        diphone = re.sub('3', 't^s', diphone)
        diphone = re.sub('4', 't^S', diphone)
        diphone = re.sub('5', 'd^z', diphone)
        diphone = re.sub('6', 'd^Z', diphone)
        diphone = re.sub('9', 'z\'', diphone)
        diphone = re.sub('1', 'n\'', diphone)
        diphone = re.sub('2', 's\'', diphone)
        diphone = re.sub('Q', 'w~', diphone)
        diphone = re.sub('W', 'j~', diphone)
        return diphone

#-------------------------------------------------
# Call main modules
def main():
    allsentences = readintranscriptions(transcriptionfile)
    orthographicsentences = readintexts(textfile)
    orthtranssentences = orthtrans(allsentences,
orthographicsentences)
    orthtranssentenceswithdiphones =
divideintodiphones(orthtranssentences)
    orthtranssentencessorted =
sortdiphones(orthtranssentenceswithdiphones)
    selectedsentences, selectedsentencesorthography,
noduplicatediphones = selectdiphones(orthtranssentencessorted,
alldiphones, diphonearrays)
    printrichsentences(selectedsentences, diphone)
    printrichsentencesorthography(selectedsentencesorthography)
    printdiphones(noduplicatediphones, diphone)

#-------------------------------------------------
main()
```

Appendix J Automatic diphone extractor – scripts

```
ReadMe file
1 – Run BLF2TextGrid.py converter
    input: BLF files
    output: TextGrid files with BLF phoneme notation
2 – Run extendedPL2PL1TextGrid.py converter
    input: TextGrid files with BLF phoneme notation
    output: TextGrid files with Polish SAMPA phoneme notation
3 – Run FindDiphonesInTextGrids.py
    input: TextGrid files with one tier on phone level
    output: DIPH files
4 – Run CutOutIndividualDiphones.py
    intput: DIPH files with corresponding WAV files
    output: diphone WAV files, MBROLAvoicedatabase.seg,
    notcutoutdiphones.txt
5 – For evaluation of diphones, run GenTextGrids4Diphones.py
    input: diphone WAV files, MBROLAvoicedatabase.seg
    output: TextGrids for diphone WAV files
6 – Concatenate diphones
    input: diphone WAV files
    output: concatenated diphone WAV files into WAV source
    files
```

Appendix J.1 BLF2TextGrid converter

```
#!/usr/bin/python
# Jolanta Bachan
# 2008-11-24
#-------------------------------------------------
# Import modules
import os, re, wave
#-------------------------------------------------
# Definitions of global variables
directory = "."
extension = "\.blf"
samplerate = 16000
#-------------------------------------------------
# List all the objects in a directory
def listfilenames(directory):
```

```
    filelist = os.listdir(directory)
    return filelist
#-------------------------------------------------
def listblffilenames(filelist):
    blflist = []
    for filename in filelist:
        if re.search(extension, filename):
            blflist.append(filename)
    return blflist
#-------------------------------------------------
def searchforwavfile(blffilename):
    extensionwav = '.wav'
    wav = re.sub(extension, extensionwav, blffilename)
    wavfile = wave.open(wav, 'r')
    length = (wavfile.getnframes())/float(samplerate)
    wavfile.close()
    return length
#-------------------------------------------------
def renameblf2textgrid(blflist):
    textgridlist = []
    extensiontextgrid = '.TextGrid'
    for blffile in blflist:
        textgrid = re.sub(extension, extensiontextgrid,
blffile)
        textgridlist.append(textgrid)
    return textgridlist
#-------------------------------------------------
def convertblffile(blflabels):
    labellist = []
    for label in blflabels:
        label = re.sub('\n','',label)
        labelstructure = label.split()
        samplenumber = 0
        sampletext = ''
        sampleintonation = ''
        if len(labelstructure) > 0:
            samplenumber = labelstructure[0]
        if len(labelstructure) > 1:
```

```
            sampletext = labelstructure[1]
        if len(labelstructure) > 2:
            sampleintonation = labelstructure[2]
        sampletriple = (samplenumber, sampletext,
sampleintonation)
        labellist.append(sampletriple)
    return labellist
#-----------------------------------------------------
def maketextgridfile(labelstructures, finalxmax):
    xmin = str(float(labelstructures[0][0])/samplerate)
    xmax = str(finalxmax) # modified by Jola
    intervalsize = str(len(labelstructures))
    textgridheader = """File type = "ooTextFile"
Object class = "TextGrid"
xmin = """ + xmin + """
xmax = """ + xmax + """
tiers? <exists>
size = 1
item []:
    item [1]:
        class = "IntervalTier"
        name = "phones"
        xmin = """ + xmin + """
        xmax = """ + xmax + """
        intervals: size = """ + intervalsize + '\n'
    textgriditemlist = ''
    for i in range(len(labelstructures)-1):
        labelstructuretext = re.sub('\"', '+',
labelstructures[i][1])
        textgriditem = '        intervals [' + str(i+1) + """]:
            xmin = """ + str(float(labelstructures[i]
[0])/samplerate) + """
            xmax = """ + str(float(labelstructures[i+1]
[0])/samplerate) + """
            text = """ + '"' + labelstructuretext + '"'
        textgriditemlist = textgriditemlist + textgriditem +
'\n'
    xmin = float(labelstructures[-1][0])/samplerate
```

```
    xmax = finalxmax # modified by Jola
    labelstructuretext = re.sub('\"', '+', labelstructures[-1]
[1])
    lasttextgriditem = '        intervals [' + intervalsize +
"""]:
            xmin = """ + str(xmin) + """
            xmax = """ + str(xmax) + """
            text = """ + '"' + labelstructuretext + '"'
    textgriditemlist = textgriditemlist + lasttextgriditem
    textgrid = textgridheader + textgriditemlist
    return textgrid
#----------------------------------------------------
def convertblf2textgrid(blffilenames,textgridfilenames):
    textgridlist = []
    for blffilename in blffilenames:
        finalxmax = searchforwavfile(blffilename) # by Jola
        blfhandle = open(blffilename,'r')
        blffile = blfhandle.readlines()
        blfhandle.close()
        labelstructures = convertblffile(blffile)
        textgrid = maketextgridfile(labelstructures, finalxmax)
        textgridlist.append(textgrid)
    for i in range(len(textgridfilenames)):
        textgridfilename = textgridfilenames[i]
        textgridhandle = open(textgridfilename,'w')
        textgridhandle.write(textgridlist[i])
        textgridhandle.close()
#----------------------------------------------------
def main():
    filenames = listfilenames(directory)
    blffilenames = listblffilenames(filenames)
    textgridfilenames = renameblf2textgrid(blffilenames)
    convertblf2textgrid(blffilenames,textgridfilenames)
    print "Finished!"
#----------------------------------------------------
main()
```

Appendix J.2 extendedPL2PL1 TextGrid converter

```
#!/usr/bin/python
# Jolanta Bachan
# 2008-11-24

# Import modules
import os, re
#-----------------------------------------------------
# Definitions of global variables
directory = "."
extension = "\.TextGrid"
#-----------------------------------------------------
# List all the objects in a directory
def listfilenames(directory):
    filelist = os.listdir(directory)
    return filelist
#-----------------------------------------------------
def listtextgridfilenames(filelist):
    textgridlist = []
    for filename in filelist:
        if re.search(extension, filename):
            textgridlist.append(filename)
    return textgridlist
#-----------------------------------------------------
def converttextgrid2textgrid_pl1(textgridfilenames):
    for textgridfilename in textgridfilenames:
      textgridhandle = open(textgridfilename, 'r')
        textgridfile = textgridhandle.readlines()
        textgridhandle.close()
        converttextgridcontent2textgrid_pl1(textgridfile,
textgridfilename)
#-----------------------------------------------------
def converttextgridcontent2textgrid_pl1(textgridfile,
textgridfilename):
    textgridline_pl1 = ''
    textgridfile_pl1 = []
    textgridfileheader_pl1 = []
    textgriridline_max = ''
```

```
    textgridline_interval = ''
    i = 0
    j = 0
    eovowel = ''
    textgridid = re.sub(extension, '_', textgridfilename)
    keytextgridline = 'intervals: size ='
    textgridlength = len(textgridfile)
    for textgridline in textgridfile:
        textgridfile[i] = re.sub('\n', '', textgridline)
        i = i + 1
        if re.search(keytextgridline, textgridline):
            keypos = i
# header
    i = 0
    while i<keypos:
        textgridfileheader_pl1.append(textgridfile[i])
        i = i + 1

    while keypos<textgridlength:
        if re.search('intervals',textgridfile[keypos]):
            j = j + 1
            textgridline_interval = textgridfile[keypos]
            textgridline_interval = re.sub('\[.*?\]', '[' +
str(j) + ']', textgridline_interval)
        elif re.search('xmin',textgridfile[keypos]):
            textgriridline_min = textgridfile[keypos]
            textgridfile[keypos] = re.sub('            xmin =
', '', textgridfile[keypos])
        elif re.search('xmax',textgridfile[keypos]):
            if keypos+4<textgridlength:
                if re.search('xmax', textgridfile[keypos+4]):
                    textgriridline_max = textgridfile[keypos]
                    textgriridline_max_pos4 =
textgridfile[keypos+4]
                    textgridfile_pos4 = textgridfile[keypos+4]
                    textgridfile[keypos] = re.sub('
xmax = ', '', textgridfile[keypos])
```

```
                    textgridfile_pos4 = re.sub('
xmax = ', '', textgridfile[keypos+4])
            else:
                textgriridline_max = textgridfile[keypos]
        elif re.search('text',textgridfile[keypos]):
            textgriridline_text = textgridfile[keypos]
            textgriridline_text = re.sub('\^', '',
textgriridline_text)
            textgriridline_text = re.sub('y', 'I',
textgriridline_text)
            textgriridline_text = re.sub('c', 'k',
textgriridline_text)
#            textgriridline_text = re.sub('\$j', '---',
textgriridline_text)
            textgriridline_text = re.sub('J', 'g',
textgriridline_text)
            textgriridline_text = re.sub('@', 'e',
textgriridline_text)
            checkvowel =  textgridfile[keypos]
            checkvowel = re.sub('            text = "', '',
checkvowel)
            checkvowel = re.sub('[#.`%*&\/|:<+\"]', '',
checkvowel)
            checkvowel = re.sub('\$p', '', checkvowel)
            checkvowel = re.sub('\$j', '', checkvowel)
            if keypos+4<textgridlength:
                if re.search('text',textgridfile[keypos+4]):
                    checkvowel_pos4 =  textgridfile[keypos+4]
                    checkvowel_pos4 = re.sub('            text
= "', '', checkvowel_pos4)
                    checkvowel_pos4 = re.sub('[#.`
%*&\/|:<+\"]', '', checkvowel_pos4)
                    checkvowel_pos4 = re.sub('\$p', '',
checkvowel_pos4)
                    checkvowel_pos4 = re.sub('\$j', '',
checkvowel_pos4)
            if re.search('[oe]',checkvowel):
                eovowel = checkvowel
```

```
                if eovowel != '':
                    if keypos+4<textgridlength:
                        if
re.search('text',textgridfile[keypos+4]) and
re.search('[wj]~',textgridfile[keypos+4]):
                            textgridfile_pos4 = re.sub('[jw]',
checkvowel, textgridfile[keypos+4])
                            textgridfile[keypos] = re.sub('
text = ', '', textgridfile[keypos])
                            textgridfile_pos4 = re.sub('
text = "', '', textgridfile_pos4)
                            textgridfile[keypos] = re.sub('"$',
'', textgridfile[keypos])
                            textgridfile[keypos] =
re.sub('[eo]', '', textgridfile[keypos])
                            textgridfile_pos4 = '
text = ' + textgridfile[keypos] + textgridfile_pos4
                            textgridline_pl1 =
textgridline_interval +'\n'+ textgriridline_min +'\n' +
textgriridline_max_pos4 +'\n' + textgridfile_pos4 +'\n'
                            textgridfile[keypos+4] = ''
                            eovowel = ''
                            j = j - 1
                            textgridfile_pl1.append(textgridlin
e_pl1)
                        else:
                            textgridline_pl1 =
textgridline_interval +'\n'+ textgriridline_min +'\n' +
textgriridline_max +'\n'+ textgriridline_text +'\n'
                            textgridfile_pl1.append(textgridlin
e_pl1)
                            eovowel = ''
                    else:
                        textgridline_pl1 =
textgridline_interval +'\n'+ textgriridline_min +'\n' +
textgriridline_max +'\n'+ textgriridline_text +'\n'
                        textgridfile_pl1.append(textgridline_pl
1)
```

```
            else:
                textgridline_pl1 = textgridline_interval +'\n'+
textgriridline_min +'\n' + textgriridline_max +'\n'+
textgriridline_text +'\n'
                textgridfile_pl1.append(textgridline_pl1)
        keypos = keypos + 1
    i = 0
    for textgridfileheader_pl1_line in textgridfileheader_pl1:
        textgridfileheader_pl1[i] = textgridfileheader_pl1_line
+ '\n'
        if re.search(keytextgridline,
textgridfileheader_pl1_line):
            textgridfileheader_pl1[i] = re.sub('.*', '
intervals: size = ' + str(j) + '\n',
textgridfileheader_pl1_line)
        i = i + 1

# concatenate header and body of TextGrid file
    textgridfile_pl1 = textgridfileheader_pl1 +
textgridfile_pl1
# create file (was '_pl1.TextGrid')
    extension_pl1 = '.TextGrid'
    textgrid_pl1 = re.sub(extension, extension_pl1,
textgridfilename)
    textgrid_pl1_filehandle = open(textgrid_pl1, 'w')

    for line in textgridfile_pl1:
        textgrid_pl1_filehandle.write(line)
    print 'Done ' + textgrid_pl1 + '!'
#--------------------------------------------------
def main():

    filenames = listfilenames(directory)
    textgridfilenames = listtextgridfilenames(filenames)
    converttextgrid2textgrid_pl1(textgridfilenames)
    print "Finished!"
#--------------------------------------------------
main()
```

Appendix J.3 Find diphones

```
#!/usr/bin/python
# Jolanta Bachan
# FindDiphonesInTextGrids.py
# 2008-11-24
# Revised on 2009-03-18
#-------------------------------------------------
# Import modules
import os, re
#-------------------------------------------------
# Definitions of global variables
directory = "."
extension = "\.TextGrid"
#-------------------------------------------------
# List all the objects in a directory

def listfilenames(directory):
    filelist = os.listdir(directory)
    return filelist
#-------------------------------------------------
# List only TextGrid files
def listtextgridfilenames(filelist):
    textgridlist = []
    for filename in filelist:
        if re.search(extension, filename):
            textgridlist.append(filename)
    return textgridlist
#-------------------------------------------------
# Read in TextGrid files
def finddiphones(textgridfilenames):
    for textgridfilename in textgridfilenames:
      textgridhandle = open(textgridfilename, 'r')
        textgridfile = textgridhandle.readlines()
        textgridhandle.close()
        converttextgrid2diph(textgridfile, textgridfilename)
#-------------------------------------------------
# Convert TextGrid files to the DIPH file format.
# DIPH format is composed of 8 columns:
```

```
# 1) diphone ID
# 2) diphone start time (in seconds)
# 3) diphone middle boundary time (in seconds)
# 4) diphone end time (in seconds)
# 5) first phone label
# 6) second phone label
# 7) diphone label
# 8) normalised diphone label for diphone filename

def converttextgrid2diph(textgridfile, textgridfilename):
    diphfile = []
    i = 0
    j = 0
    textgridid = re.sub(extension, '_', textgridfilename)
    keytextgridline = 'intervals: size ='
    textgridlength = len(textgridfile)
    diphfileline = ''
    for textgridline in textgridfile:
        textgridfile[i] = re.sub('\n', '', textgridline)
        i = i + 1
        if re.search(keytextgridline, textgridline):
            keypos = i
    while keypos<textgridlength:
        if re.search('intervals',textgridfile[keypos]):
            j = j + 1
            diphfileline = textgridid + str(j)
        elif re.search('xmin',textgridfile[keypos]):
            textgridfile[keypos] = re.sub('            xmin =
', '', textgridfile[keypos])
            textgridfile[keypos] = re.sub(' ', '',
textgridfile[keypos])
            diphfileline = diphfileline + '\t'
+textgridfile[keypos]
        elif re.search('xmax',textgridfile[keypos]):
            if keypos+4<textgridlength:
                if re.search('xmax', textgridfile[keypos+4]):
                    textgridfile_pos4 = textgridfile[keypos+4]
```

```
                    textgridfile[keypos] = re.sub('
xmax = ', '', textgridfile[keypos])
                    textgridfile_pos4 = re.sub('
xmax = ', '', textgridfile_pos4)
                    textgridfile[keypos] = re.sub(' ', '',
textgridfile[keypos])
                    textgridfile_pos4 = re.sub(' ', '',
textgridfile_pos4)
                    diphfileline = diphfileline + '\t' +
textgridfile[keypos] + '\t' +  textgridfile_pos4
        elif re.search('text',textgridfile[keypos]):
            if keypos+4<textgridlength:
                if re.search('text',textgridfile[keypos+4]):
                    textgridfile[keypos] = re.sub('
text = ', '', textgridfile[keypos])
                    textgridfile_pos4 = re.sub('
text = ', '', textgridfile[keypos+4])
                    textgridfile[keypos] = re.sub('_#', '',
textgridfile[keypos])
                    textgridfile[keypos] = re.sub('[#.`
%*&\/|:<\"\+]', '', textgridfile[keypos])
                    textgridfile[keypos] = re.sub('SIL', '_',
textgridfile[keypos])
                    textgridfile[keypos] = re.sub('\$p', '_',
textgridfile[keypos])
                    textgridfile[keypos] = re.sub('\$j',
'junk', textgridfile[keypos])
                    textgridfile[keypos] = re.sub(' ', '',
textgridfile[keypos])
                    textgridfile_pos4 = re.sub('_#', '',
textgridfile_pos4)
                    textgridfile_pos4 = re.sub('[#.`%*&\/|:<\"\
+]', '', textgridfile_pos4)
                    textgridfile_pos4 = re.sub('SIL', '_',
textgridfile_pos4)
                    textgridfile_pos4 = re.sub('\$p', '_',
textgridfile_pos4)
```

```
                    textgridfile_pos4 = re.sub('\$j', 'junk',
textgridfile_pos4)
                    textgridfile_pos4 = re.sub(' ', '',
textgridfile_pos4)
                    diphonelabel = textgridfile[keypos] + '-' +
textgridfile_pos4
                    diphonelabel = normaliselabel(diphonelabel)
                    # Create diphone file line with 8 columns
                    diphfileline = diphfileline + '\t' +
textgridfile[keypos] + '\t' +  textgridfile_pos4 + '\t' +
textgridfile[keypos] + '-' +  textgridfile_pos4 + '\t' +
diphonelabel+'\n'
                    diphfile.append(diphfileline)
        keypos = keypos + 1

# Create DIPH file
    extension_diph = '.diph'
    diph = re.sub(extension, extension_diph,  textgridfilename)
    diphfilehandle = open(diph, 'w')
    for line in diphfile:
        diphfilehandle.write(line)

#-----------------------------------------------------
# Normalise diphone labels so that they can be used for the
diphone filename
# Replacement of:
# - special characters
# - capital letters
# - underscore '_'

def normaliselabel(diphonelabel):
    diphonelabel = re.sub('\~', 'n', diphonelabel)
    diphonelabel = re.sub('\'', 'i', diphonelabel)
    diphonelabel = re.sub('\^', '', diphonelabel)
    diphonelabel = re.sub('\?', 'q', diphonelabel)
    diphonelabel = re.sub('S', 'ss', diphonelabel)
    diphonelabel = re.sub('Z', 'zz', diphonelabel)
    diphonelabel = re.sub('N', 'nn', diphonelabel)
```

```
    diphonelabel = re.sub('B', 'bb', diphonelabel)
    diphonelabel = re.sub('G', 'gg', diphonelabel)
    diphonelabel = re.sub('C', 'cc', diphonelabel)
    diphonelabel = re.sub('J', 'jj', diphonelabel)
    diphonelabel = re.sub('I', 'ii', diphonelabel)
    diphonelabel = re.sub('U', 'uu', diphonelabel)
    diphonelabel = re.sub('O', 'oo', diphonelabel)
    diphonelabel = re.sub('E', 'ee', diphonelabel)
    diphonelabel = re.sub('@', 'ea', diphonelabel)
    diphonelabel = re.sub('_', 'SIL', diphonelabel)
    return diphonelabel
#-------------------------------------------------
def main():

    filenames = listfilenames(directory)
    textgridfilenames = listtextgridfilenames(filenames)
    finddiphones(textgridfilenames)
#-------------------------------------------------
main()
```

Appendix J.4 Cut out individual diphones

```
#!/usr/bin/python
# Jolanta Bachan
# 2008-12-02
# Modified on 2009-03-19

# Import modules
import os, re, commands, wave, operator
#-------------------------------------------------
# Definitions of global variables
directory = '.'
extension = '\.diph'
extensionwav = '.wav'
tab = '\t'
alldiphoneset = set([])
alldiphones = set([])
diphfilelines = []
alldiphfilelines = []
samplerate = 16000
```

```
pause = '_'
i = 0
#-----------------------------------------------------
# List all the objects in a directory
def listfilenames(directory):
    filelist = os.listdir(directory)
    return filelist
#-----------------------------------------------------
# List only DIPH files
def listdiphfilenames(filelist):
    diphlist = []
    for filename in filelist:
        if re.search(extension, filename):
            diphlist.append(filename)
    return diphlist
#-----------------------------------------------------
# Read in DIPH files
def finddiphones(diphfilenames, alldiphones, diphfilelines):
    for diphfilename in diphfilenames:
      diphhandle = open(diphfilename, 'r')
        diphfile = diphhandle.readlines()
        diphhandle.close()
        diphoneset, alldiphfilelines =
finddiphonesset(diphfile, alldiphoneset)
        alldiphones = alldiphones.union(diphoneset)
        sortedalldiphfilelines = sorted(alldiphfilelines,
key=operator.itemgetter(6))
    return alldiphones, sortedalldiphfilelines
#-----------------------------------------------------
def finddiphonesset(diphfile, alldiphoneset):
    diphones = set([])
    for diphline in diphfile:
        diphitems = re.split('\s', diphline)
        diphonelabel = diphitems[6]
        diphones.add(diphonelabel)
        diphfilelines.append(diphitems)
        alldiphoneset = alldiphoneset.union(diphones)
    return alldiphoneset, diphfilelines
```

```
#-------------------------------------------------
# THE TRICKY MODULE
def selectdiphones(diphfilenames, alldiphoneset,
sortedalldiphfilelines, i):
    label = ''
    j = 0
    instances = 1
    # Create MBROLA database SEG file
    segfile = 'MBROLAvoicedatabase.seg'
    segfilehandle = open(segfile, 'a')
    # Create error TXT file
    errorfile = 'notcutoutdiphones.txt'
    errorfilehandle = open(errorfile, 'a')
    for diphline in sortedalldiphfilelines:
        if label != '':
            if diphline[6] == label and j < instances:
                i = cutoutdiphone(diphline, segfilehandle,
errorfilehandle, i)
                j = j + i
            elif diphline[6] != label:
                label = diphline[6]
                j = cutoutdiphone(diphline, segfilehandle,
errorfilehandle, i)
            else:
                label = ''
                j = 0
                alldiphoneset.remove(diphline[6])
        elif label == '' and diphline[6] in alldiphoneset:
                label = diphline[6]
                j = cutoutdiphone(diphline, segfilehandle,
errorfilehandle, i)
#-------------------------------------------------
# Cut out diphone if it meets the conditions
def cutoutdiphone(diphline, segfilehandle, errorfilehandle, i):
    wavitems = re.split('_', diphline[0])
    wavfilename = wavitems[0] + extensionwav
    wavfilehandle = wave.open(wavfilename, 'r')
```

```
    wavlength = (wavfilehandle.getnframes())/float(samplerate)
    wavfilehandle.close()
    starttime = float(diphline[1]) + ((float(diphline[2]) -
float(diphline[1]))/2) - 0.05
    lengthtime = (((float(diphline[2]) - float(diphline[1]))/2)
+ ((float(diphline[3]) - float(diphline[2]))/2)) + 0.1
    checklimit = (lengthtime * 16000)
#    diphonefilename = diphline[7] + '_' + diphline[0] +
extensionwav
    diphonefilename = diphline[7] + extensionwav
    checklength_lefthalfphone = (float(diphline[2]) -
float(diphline[1]))/2
    checklength_righthalfphone = (float(diphline[3]) -
float(diphline[2]))/2
    diphonestart = int(0.05*16000)
    diphonemiddle = int((0.05 + (float(diphline[2]) -
float(diphline[1]))/2)*16000)
    diphoneend =  int(((float(diphline[3]) -
float(diphline[2]))/2*16000) + diphonemiddle)
    if re.search(pause, diphline[4]) and
checklength_lefthalfphone > 0.1:
        starttime = float(diphline[2]) - 0.15
        lengthtime = (float(diphline[3]) -
float(diphline[2]))/2 + 0.2
        checklimit = lengthtime * 16000
        diphonemiddle = int(0.15*16000)
        diphoneend =  int(((float(diphline[3]) -
float(diphline[2]))/2*16000) + diphonemiddle)
    elif re.search(pause, diphline[5]) and
checklength_righthalfphone > 0.1:
        lengthtime = (float(diphline[2]) -
float(diphline[1]))/2 + 0.2
        checklimit = lengthtime * 16000
        diphoneend =  int(0.1*16000 + diphonemiddle)
    # Create the command for SOX
    soxcommand = 'sox '+ wavfilename + ' ' + diphonefilename +
' trim ' + str(starttime) + ' ' + str(lengthtime)
    # Create the line for the SEG file
```

```
    segfileline = diphonefilename + tab + diphline[4] + ' ' +
diphline[5] + tab + str(diphonestart) + tab + str(diphoneend) +
tab + str(diphonemiddle) + '\n'
    checktime = (float(diphline[3]) - float(diphline[2]))/2 +
float(diphline[2]) + 0.05
    if starttime < 0:
        segfileline =  'error 1: ' + segfileline
        errorfilehandle.write(segfileline)
        i = 0
    elif checktime > wavlength:
        segfileline =  'error 2: ' + segfileline
        errorfilehandle.write(segfileline)
        i = 0
    elif checklimit > 10000:
        segfileline =  'error 3: ' + segfileline
        errorfilehandle.write(segfileline)
        i = 0
    else:
        commands.getstatusoutput(soxcommand)
        segfilehandle.write(segfileline)
        i = 1
    return i
#-------------------------------------------------
def main():
    filenames = listfilenames(directory)
    diphfilenames = listdiphfilenames(filenames)
    alldiphoneset, sortedalldiphfilelines =
finddiphones(diphfilenames, alldiphones, diphfilelines)
    selectdiphones(diphfilenames, alldiphoneset,
sortedalldiphfilelines, i)
#-------------------------------------------------
main()
```

Appendix J.5 Generate TextGrids for diphones

```
#!/usr/bin/python
# Jolanta Bachan
# GenTextGrids4Diphones.py
# 2008-12-08
# Revised on 2009-03-18
```

```
#-------------------------------------------------
# Import modules
import os, re, wave
#-------------------------------------------------
# Definitions of global variables
directory = "."
extension = "\.seg"
samplerate = 16000
#-------------------------------------------------
# List all the objects in a directory
def listfilenames(directory):
    filelist = os.listdir(directory)
    return filelist
#-------------------------------------------------
# Find the SEG file
def findsegfilename(filelist):
    segfilename = ''
    for filename in filelist:
        if re.search(extension, filename):
            segfilename = filename
    return segfilename
#-------------------------------------------------
# Read in the SEG file
def readsegfile(segfilename):
    seghandle = open(segfilename, 'r')
    segfile = seghandle.readlines()
    seghandle.close()
    return segfile
#-------------------------------------------------
# Base on the data in the SEG file and the measurement of
# the length of the diphone WAV files, create TextGrid files
# for each diphone WAV file.
def generatetextgrid4diphones(segfile):
    for segline in segfile:
        segcontent = re.split('\s', segline)
        wavfilename = segcontent[0]
        wavlength = getlength(wavfilename)
        extensionwav = '\.wav'
```

```
        extensiontextgrid = '.TextGrid'
        textgridfilename = re.sub(extensionwav,
extensiontextgrid, wavfilename)
        textgridfile = open(textgridfilename,'w')
        xmin = '0'
        xmax = str(wavlength)
        diphonestart = str(float(segcontent[3])/samplerate)
        diphonemiddle = str(float(segcontent[5])/samplerate)
        diphoneend = str(float(segcontent[4])/samplerate)
        intervalsize = '4'
        textgrid = """File type = "ooTextFile"
Object class = "TextGrid"
xmin = """ + xmin + """
xmax = """ + xmax + """
tiers? <exists>
size = 1
item []:
    item [1]:
        class = "IntervalTier"
        name = "halfphones"
        xmin = """ + xmin + """
        xmax = """ + xmax + """
        intervals: size = """ + intervalsize + """
        intervals [1]:
            xmin = """ + xmin + """
            xmax = """ + diphonestart + """
            text = ""
        intervals [2]:
            xmin = """ + diphonestart + """
            xmax = """ +  diphonemiddle +"""
            text = \""""" + segcontent[1] + """\"
        intervals [3]:
            xmin = """ + diphonemiddle + """
            xmax = """ +  diphoneend +"""
            text = \""""" + segcontent[2] + """\"
        intervals [4]:
            xmin = """+ diphoneend + """
            xmax = """ + xmax +"""
```

```
            text = \"\"""
        textgridfile.write(textgrid)
#----------------------------------------------------
# Measure the length of the diphone WAV file
def getlength(wavfilename):
    wavfile = wave.open(wavfilename, 'r')
    length = (wavfile.getnframes())/float(samplerate)
    wavfile.close()
    return length
#----------------------------------------------------
def main():
    filenames = listfilenames(directory)
    segfilename = findsegfilename(filenames)
    segfile = readsegfile(segfilename)
    generatetextgrid4diphones(segfile)
    print "Finished!"
#----------------------------------------------------
main()
```

Appendix J.6 Concatenate diphones

```
#!/usr/bin/python
# Jolanta Bachan
# ConcatenateDiphones.py
# 2008-12-08
#----------------------------------------------------
# Import modules
import os, re, operator, commands, wave
#----------------------------------------------------
# Definitions of global variables
directory = "."
extension = ".wav"
samplerate = 16000
#----------------------------------------------------
# List all the objects in a directory
def listfilenames(directory):
    filelist = os.listdir(directory)
    return filelist
#----------------------------------------------------
# List the (diphone) WAV files
```

```
def listwavfilenames(filelist):
    wavlist = []
    for filename in filelist:
        if re.search(extension, filename):
            wavlist.append(filename)
    return wavlist
#-------------------------------------------------
# Create an unsorted embeded list of diphone filenames, e.g.
# [['d-j', 'A0002', '5'], ['e-b', 'A0009', '24'], ['b-r',
'A0010', '2'],
# ['v-a', 'A0009', '10'], ['zi-i', 'A0010', 10], ['x-p',
'A0001', '31']]
def structurewavfilenames(wavfilenames):
    unsortedwavfiles = []
    for wavfile in wavfilenames:
        wavfile = re.sub(extension, '', wavfile)
        # Create a list from a diphone filename structure with
3 elements:
        # 1 - diphone label
        # 2 - diphone source filename
        # 3 - diphone place of occurrence in the source file
        wavfilestructure = wavfile.split('_')
        # Append the 3-element list of the diphone filename
structure
        # to the unsorted list of diphone filenames. The 3-
element list
        # is embeded into the unsorted list of diphone
filenames.
        unsortedwavfiles.append(wavfilestructure)
    return unsortedwavfiles
#-------------------------------------------------
# Sort the unsorted diphone filenames according to the diphone
source filename, e.g.
# [['u-f', 'A0001', '38'], ['n-t', 'A0001', '36'], ['t-u',
'A0001', '37'], ...,
# ['s-p', 'A0002', '29'], ['o-n', 'A0002', '9'], ['b-a',
'A0002', '24'], ...,
```

```
# ['e-s', 'A0009', '5'], ['r-ni', 'A0009', '26'], ['e-g',
'A0009', '19'], ...,
#  ['b-r', 'A0010', '2'], ['SIL-q', 'A0010', '5'], ['e-zi',
'A0010', '9']]
def sortwavefiles(unsortedwavfilenames):
    sortedwavfiles = sorted(unsortedwavfilenames,
key=operator.itemgetter(1))
    return sortedwavfiles
#-------------------------------------------------
# List the source filenames, e.g.
# ['A0001', 'A0002', 'A0009', 'A0010']

def findbasicwavfiles(sortedwavfilenames):
    i = 0
    basicwavfiles = []
    for sortedwavfilename in sortedwavfilenames:
        if sortedwavfilenames[i][1] != sortedwavfilenames[i-1]
[1]:
            basicwavfile = sortedwavfilenames[i][1]
            basicwavfiles.append(basicwavfile)
        i = i + 1
    return basicwavfiles
#-------------------------------------------------
# Sort the sorted list of diphone filenames according to the
place of diphone occurrence
# e.g. [['b-r', 'A0010', 2], ['r-r', 'A0010', 3], ['r-SIL',
'A0010', 4]]
# and call the concatenation function
def sortdiphones(basicwavfilenames, sortedwavfilenames):
    unsorteddiphones = []
    i = 0
    length = len(sortedwavfilenames)

    for sortedwavfilename in sortedwavfilenames:
        while i<length:
            sortedwavfilenames[i][2] =
int(sortedwavfilenames[i][2])
```

```
            if  sortedwavfilenames[i][1] ==
sortedwavfilenames[i-1][1]:
                unsorteddiphones.append(sortedwavfilenames[i-
1])
            else:
                unsorteddiphones.append(sortedwavfilenames[i-
1])
                sorteddiphones = sorted(unsorteddiphones,
key=operator.itemgetter(2))
                unsorteddiphones = []
                concatenatediphones(sorteddiphones)
            i = i + 1
    if sortedwavfilename[1] == sortedwavfilenames[i-1][1]:
        sortedwavfilename[2] = int(sortedwavfilename[2])
        unsorteddiphones.append(sortedwavfilenames[i-1])
        sorteddiphones = sorted(unsorteddiphones,
key=operator.itemgetter(2))
        concatenatediphones(sorteddiphones)
#--------------------------------------------------
# Concatenate sorted diphones:
# 1 - Extract pure diphones without the context
#     of 50ms on the left and on the right of the diphone
# 2 - Write the pure diphones into the WAV files with '_diph'
extension
# 3 - Concatenate the pure diphone files
def concatenatediphones(sorteddiphones):
    soxcommandcontent = ''
    soxcommandcontentmore = ''
    concatwavfilename = sorteddiphones[0][1] + '_concat'+
extension
    diphonefilename = sorteddiphones[0][0] + '_' +
sorteddiphones[0][1] + '_' + str(sorteddiphones[0][2]) +
extension
    cpcommand = 'cp ' +   diphonefilename + ' ' +
concatwavfilename
    commands.getstatusoutput(cpcommand)
    i = 1
    for sorteddiphone in sorteddiphones[1:]:
```

```
        i = i + 1
        diphonefilename = sorteddiphone[0] + '_' +
sorteddiphone[1] + '_' + str(sorteddiphone[2]) + extension
        onlydiphonefile = sorteddiphone[0] + '_' +
sorteddiphone[1] + '_' + str(sorteddiphone[2]) + '_diphone' +
extension
        diph = sorteddiphone[0] + '_' + sorteddiphone[1] + '_'
+ str(sorteddiphone[2])
        diphonefile = wave.open( diphonefilename, 'r')
        diphonefilelength =
(diphonefile.getnframes())/float(samplerate)
        diphonefile.close()
        diphonelength = float(diphonefilelength - 0.1)
        soxtrimcommand = 'sox ' + diphonefilename + ' ' +
onlydiphonefile + ' trim 0.05 ' + str(diphonelength)
        commands.getstatusoutput(soxtrimcommand)
        soxcommand = 'sox '+ concatwavfilename +  ' ' +
onlydiphonefile + ' new.wav'
        commands.getstatusoutput(soxcommand)
        mvcommand = 'mv new.wav ' + concatwavfilename
        commands.getstatusoutput(mvcommand)
#-------------------------------------------------
# Call main functions
def main():
    filenames = listfilenames(directory)
    wavfilenames = listwavfilenames(filenames)
    unsortedwavfilenames = structurewavfilenames(wavfilenames)
    sortedwavfilenames = sortwavefiles(unsortedwavfilenames)
    basicwavfilenames = findbasicwavfiles(sortedwavfilenames)
    sortdiphones(basicwavfilenames, sortedwavfilenames)
    print "Finished!"
#-------------------------------------------------
main()
```

Appendix K Text material used for the Polish MBROLA voice creation

Appendix K.1 Phonetically rich sentences

Phonetically rich sentences were extracted automatically using the phonetically rich sentence extractor and come from BOSS (Demenko et al. 2007) and JURISDIC (Demenko 2008) databases.

W opartej na II wojnie światowej strategicznej grze komputerowej "Blitzkrieg" musisz wykazać się zdolnościami taktycznymi i błyskawicznie reagować na wydarzenia.
Układ dokrewny czyli wewnątrzwydzielniczy zbudowany jest z narządów wewnątrzwydzielniczych regulujących różnorodne funkcje organizmu dzięki wydzielanym do układu krążenia hormonom.
John Rambo występował jak zawsze w wytartych dżinsach i przepoconym podkoszulku i jak zawsze wygrał wszystkie pojedynki w przeróżnych stylach walki: od boksu po dżudo.
W raporcie kontrolerzy zarzucają urzędnikom złamanie prawa o zamówieniach publicznych i wydanie miliona złotych w sposób nieoszczędny lub niegospodarny.
W tym krótkim opowiadanku Stasiuka o awanturze na wiejskiej zabawie jest przezabawna scena gdzie na salę za kuśtykającym żebrakiem wbiegają głodne prosiaki.
Muszę nałożyć na pana grzywnę za posługiwanie się podrobionymi dokumentami oraz próbę przekupienia funkcjonariusza polskiej Straży Granicznej.
Cła wywozowe mają charakter dodatkowego podatku ciążącego na towarach sprzedawanych ze względów pozaekonomicznych po niższych cenach w kraju niż za granicą.
W wielu rankingach spółka Warwick uchodzi za światowego lidera w dziedzinie produkcji pieców do obróbki cieplnej metali i stopów.
Ogólnopolski Festiwal Piosenki Żeglarskiej Zęza to coroczna impreza szantowa organizowana na początku maja w Łaziskach Górnych.
Ministerstwo zdrowia wdrożyło program profilaktyki raka piersi i umożliwiło darmowe badania mammograficzne kobietom w fazie menopauzy.
który był bardzo odpowiedzialnym i odważnym człowiekiem należał do wielu ważnych organizacji wojskowych założonych w czasie drugiej wojny swiatowej.
Podczas wojny firma Daimler-Benz znana przede wszystkim z samochodów marki

Mercedes-Benz produkowała serie silników do maszyn wojennych.
W najmroczniejszych zakątkach tajgi i tundry nawet najpłytsze jeziora i rzeki pozostają przez większą część roku zamarznięte.
W przypadku ciąży pozamacicznej z uwagi na niemożność prawidłowego rozwoju zarodka poza jamą macicy najczęściej występuje poronienie.
Jedna z największych firm handlowych w centralnej Polsce Hea oferuje wyroby hutnicze dla potrzeb budownictwa i konstrukcji.
Najnowsze badania genetyczne dowodzą niezwykle bliskiego pokrewieństwa między żbikiem i kotem nubijskim sugerując.
Ksiądz Rydzyk zaprosił Leppera do telewizji TRWAM i na antenie Lepper rozmawiał z Rydzykiem o koalicji i pomocy dla sadowników.
Papież zawsze był orędownikiem ekumenizmu i starał się współkształtować płaszczyznę porozumienia z religią półksiężyca.
Pani profesor Przecławska zajmuje się pedagogiką oraz teorią upowszechniania kultury i literatury dla dzieci i młodzieży.
Nawet najlepsi stratedzy nic by tu nie pomogli bo Koreanki rozpierzchają się jak ptaki ile razy próbuję do nich podejść.
Wypowiedź przedstawiciela Kremla o „historycznym ekstremiźmie" Polski zabrzmiała jak reprymenda dla nieposłusznego podwładnego.
Jej łakomstwo przekroczyło wczoraj wszelkie granice a ilość zjedzonej przez nią wędliny jest wprost niewyobrażalna.
Dzięki subskrypcji możesz otrzymywać na swój email informacje dotyczące nowości oferowanych w naszym serwisie
Ta eksperymentalna architektura powstała w okresie między upadkiem dynastii seldżuckiej a początkiem dominacji Osmanów.
Rzeczpospolita jako pierwsza opisała najświeższe informacje dotyczące skandalu łóżkowego polityków Samoobrony.
W godzinach południowych ksiądz arcybiskup Stanisław Gądecki spotkał się z księżmi seniorami zamieszkałymi w Archidiecezji.
Spośród uprawiających sztukę poetycką trubadurów albo spośród dworskich grajków i śpiewaków wywodzili się minstrele.
W tamtych czasach na londyńskich przedmieściach grasowali bandyci i nie istniało nic

takiego jak nadzór policyjny.
Na zakończenie roku szklonego zorganizowano młodzieży wycieczkę do Muzeum Etnograficznego Wdzydze Kiszewskie.
Cyrkowcy i hipnotyzerzy muszą lubić wieczną łazęgę w poszukiwaniu rozochoconej i rozwrzeszczanej hałastry.
Obecnie coraz szersze zastosowanie w lecznictwie zyskują sulfonamidy o przedłużonym działaniu bakteriostatycznym.
Trzecim co do wielkości miastem Iranu jest Isfahan położony około pięćset kilometrów na południe od Teheranu.
Opcjonalnie dołączamy do tego aparatu uchwyt montażu sufitowego i superkrótkoogniskowy obiektyw stały.
Adepci i adeptki sztuki pisania poezji powinni wpierw zaznajomić się z twórczością Cypriana Kamila Norwida.
Jedynie przewoźnicy z UE z uprawnieniem do świadczenia międzynarodowych usług przewozowych będą mieli prawo świadczenia wewnątrzkrajowych usług.
Językiem gudżarati posługuje się ponad czterdzieści sześć milionów użytkowników w samych tylko Indiach.
Znowu palnąłem głupstwo przy księdzu i teraz żadne pochlebstwa na temat arcybiskupstwa nie załagodzą sprawy.
Z iłołupków karbońskich wyrabia się głównie cegłę pełną oraz około dziesięć procent cegły dziurawki oraz cegły kratówki.
Nieuchronnym skutkiem nowej restrykcyjnej ustawy będzie rosnące zniechęcenie i nieuctwo wśród młodych ludzi.
Na Festiwalu Pieśni Patriotycznej w Biłgoraju przypomniano postać marszałka Józefa Piłsudskiego.
Lekarz podejrzewa że u najwrażliwszych mogło dojść do zapalenia w obrębie stawu skroniowo-żuchwowego.
Trudno powiedzieć jednoznacznie w czyjej głowie zrodził się pomysł powołania pułku szwoleżerów jastarniańskich.
Czołowy francuski siatkarz Stephane Antiga od przyszłego sezonu będzie zawodnikiem Skry Bełchatów.
W najmroźniejszym miesiącu roku zwiedzaliśmy Nową Gwineę porośniętą bonsajami i

innymi ciekawymi roślinami.
przy czym już podstawowy kurs Silvy obejmuje nauczenie się samoczynnego wprowadzania mózgu w stan alfa.
Cała biżuteria została ręcznie wykonana ze srebra oraz z kamieni półszlachetnych takich jak: agat.
Tomasz Kręcielewski zdobył w Brunszwiku brązowy medal otwartych mistrzostw Niemiec w judo.
Kucharka wstawała najrychlej z całej służby i wybierała co najtłustsze kawałki mięsa na rosołek dla dziedziczki.
Najciekawszą propozycją festiwalu filmów azerbejdżańskich był magiczny „Ujeżdżacz smoków".
w Deklaracji Stuttgardzkiej Rady Europy wyraźnie zadeklarowano chęć stymulacji procesów integracyjnych.
Wzruszającą nowelę o ciężkim losie uchodźcy zza wschodniej granicy stworzył Siergiejewicz.
Spróbuj policzyć te wszystkie zadania a końcowe liczby zapisz w tabelce na ostatniej stronie.
Kilka odcinków serialu „Bonanza" w latach sześćdziesiątych wyreżyserował Robert Altman.
Do roku dwa tysiące pierwszego sucre było oficjalną walutą państwową Ekwadoru.
Rodzina Zdzisława Krzyżanowskiego mieszkała kiedyś w żydowskiej dzielnicy miasta na ulicy Wrocławskiej
Podróżnik arabski Ibrahim ibn Jakub był członkiem poselstwa dyplomatycznego.
Pobiegnij do sklepu tytoniowego na Grunwaldzką i kup miętowe Irisy albo Zefiry.
W ogrodzie posadził mnóstwo brzóz z białawą korą i miniaturowych drzewek bonsai.
Zauważcie jak Gotfryd swobodnie się czuje w todze dziekańskiej – jakby się w niej urodził.
W instytucie Towarzystwa Fraunhofera w Erlangen stworzono algorytm MP3.
Agroturystyka jest głównym źródłem utrzymania w Gierzwałdzie.
Bóg dopilnowuje aby nirwany nie uzyskał żaden niemoralny tchórz.
Na juwenalia międzyuczelniane każdy miał przyjść w cudzych ciuchach.
Ciasto nawilżone olejkiem anyżowym smakuje tak samo jak kiedyś.

Mam szczerze dosyć kmiotków pyszniących się swoimi wyczynami.
Wybrałem się do kina złakniony mocnych wrażeń.
więc skończ już grać na tej harmonijce.
Olzą płyńmy wskroś Zaolzia
Naddruk książki też się sprzedał i wydawca zlecił dodrukowanie kilkuset egzemplarzy książki.
Menedżerzy naszej firmy obdzwaniają agencje w poszukiwaniu nowej twarzy do reklamy.
Konflikt wokół tarcz antyrakietowych narasta i nie widać szansy na kompromis.
Najwyżsi włodarze postanowili nie walczyć o zrzucenie jarzma najeźdźcy.
Bądźcie tak mili i podzielcie się z Ziemkiewiczem waszymi rewelacjami.
Gdy bartnik toruński wszedł do karczmy słychać było pomruki.
Przestań krzyczeć i nie syczże już na biednego nicponia.
Chłopak z ulicy Staszica nie szturchnął cię nienaumyślnie.
Äśbereinstimmungen in BinÄ¤rdatei (Standardeingabe).
Koń warczy jak ćma w noc szarą
Oddziel dziś go Oldze
Na łomżyńskim rynku żebrze młoda Rumunka z czworgiem dzieci w okularach słonecznych i kolorowych chustach.
Nieproszeni goście chytrym sposobem wkręcili się na wieczorek zapoznawczy w ciechocińskim sanatorium.
W wyniku roszad personalnych funkcjonariusz Interpolu znalazł się nieoczekiwanie w Lillehammer.
kukiełki i postacie wyświetlane na ekranie okazało się najciekawszym przedsięwzięciem festiwalu.
W tej dziwnej grze w tenis backhand kickboxera okazał się lepszy od backhandu paleoantropologa.
W informacji o czynszach nie uwzględniono lokali zastępczych i pomieszczeń socjalnych.
Ich wyzwaniem stało się uczynić z clubbingu aktywność przyjazną dla środowiska.
Przed nałożeniem tej odzieży przeciwchemicznej na ćwiczeniach radzę ci wsmarować w twarz jakiś krem.
Przy tak dżdżystej pogodzie kładzenie posadzek innych niż drewniane mija się z celem.
Bogusław Mec nagrał nostalgiczną płytę zatytułowaną „Recepta na życie”.

Doceniam umiejętność niewdawania się w konflikty i efektywność działania.
a późniejsze losy Łobza też łączą się z losami rodu Borków.
W Dzień Babci zjechało się wiele babć z różnych stron kraju.
Siedmioro siedmioletnich dzieci potrafiło odmieniać rzeczowniki
ja zrzeknę się swojej części na rzecz Matyldy.
Zarząd Do Spraw Cudzoziemców: ZdsC lub ZCU.
mieliby dziś z tego obrazu niezły szmal.
To jest ochraniacz naszego jednorożca.
Pan dziedzic zbeształ zaś giermka.
Ale i on miał coś do powiedzenia.
W bryczce cmokają się i gwiżdżą
Księżyc lśnił mocno noc całą
Wesele Figara i Don Giovanni są jednymi z najczęściej grywanych oper Wolfganga Amadeusza Mozarta.
Kawalkada dwuśladów przejechała rynek obwieszczając wszystkim wieść o triumfie reprezentacji.
Żeby ogrzać kilogram wody o jeden stopień potrzeba prawie cztery tysiące dwieście dżuli.
Management szkoły wypłacił sobie tysiączłotowe dodatki z okazji rocznicy.
Na lato doskonały będzie kapelusik z cieńszej tkaniny bawełnianej podobnej do jeansu.
Ten sztumski zawodnik o wielkiej sile bierze udział w biatlonie po raz pierwszy.
Elektroakupunktura to najlepszy sposób na depresję o podłożu psychogennym.
Jednym z bardziej uczęszczanych klubów muzycznych Kielc jest klub koncertowy „Mefisto.”.
Najnowsze modele przenośnych odtwarzaczy CD posiadają system przeciwwstrząsowy.
Przechwycenie kuriera wroga było punktem zwrotnym dla wojsk Jagiełły.
skończże wreszcie wygibasy z rowerkiem i chodź lepiej tutaj do piaskownicy.
Beata wyidealizowała swój związek z Krzyśkiem do tego stopnia.
Na Białostocczyźnie rosną buki i olchy oraz krzewy różnego typu.
A ten leń znowuż wybiera się na nadmorskie wywczasy z małżonką.
Ten gazociąg jest wynikiem polityki proamerykańskiej.
Wewnątrzsieciowe i wewnątrzszkolne połączenie nie było zabezpieczone cieplnie.
Przed deszczem schroniliśmy się u jakichś ludzi dobrych.

kto zdobędzie większość głosów: partia Centrum.
Kuomintang uwięził bośniackich oportunistów.
Połóż ręce na kierownicy i zostań w tej pozycji.
Zaoszczędź na wstęp do rezydencji Habsburgów.
Oskarżyli dwóch karłów o kradzież portmonetki.
Łotysz stanął w szranki z postawnym Kaszubem.
Ta z którą mnie widziałaś jest zbzikowana.
rozmiękczę ten owoc i dam wam po ćwiartce.
przepisów BHP oraz kwalifikacji załogi.
W dalszym ciągu noszono welon z przepaską.
broń się wodą święconą i czosnkiem
Nazbierałem garść ładnych jagód.
Dostał w mięsień dwugłowy ramienia.
Podłącz zasilacz do gniazdka.
Sprawcy PGWP z Wietnamu.
Znalazłeś szal Rajmundy.
On żył zacniej niż żonka
Slajd powieś na ścianie.
Wolał pozostać uczciwym robotnikiem bez kasy niż wspinać się po szczeblach kariery oszukując ludzi.
Zapraszam wszystkich zgromadzonych: koniecznie przyjedźcie na recital Alicji Dudziak już za dwa tygodnie.
Część z satelit opuściła przestrzeń wokółziemską i stała się statkami – sondami międzyplanetarnymi.
Wewnątrzwspólnotowe reguły są według niego nie wiele znaczącym mętlikiem sprzecznych zasad.
Rańtuchem nazywano kobiece wełniane nakrycie głowy używane w Polsce między szesnastym a osiemnastym wiekiem.
Półzbroja husarska należy do typu wybitnie użytkowego uzbrojenia ochronnego końca XVII wieku.
Kapelusze rydzy przed smażeniem należy obgotować w ciągu kilku minut we wrzącej wodzie.

Anarchiści w rejonie bydgoskim musieli zejść do podziemi i klepią teraz straszną biedę.
Opowiedzcie jak to staroświeccy doradcy hetmańscy próbowali zaprowadzić inny porządek w tym powiecie.
W telewizyjnym quizie wzięli udział reprezentanci pięciu sołectw województwa mazowieckiego.
Przez obfitą mżawkę nie udało się zidentyfikować numeru rejestracyjnego pojazdu.
Koreańskie władze mamią zwykłych ludzi ideami i nielegalnie przygotowują broń nuklearną.
Na wniosek Ministerstwa Skarbu Państwa Ernst Jansen został odwołany z rady nadzorczej PZU.
Zaczną dopłacać do interesu jeśli nie przywrócą poprzedniego systemu i nie zaczną działać ostrożniej.
Wkrótce przybywa tam także Kanadyjczyk Caravaggio i młody sikhijski saper Kip.
Szef z okazji swoich czterdziestych urodzin stawiał dziś wszystkim pracownikom piwa.
Ze względu na swoją dietę muszę przestać jeść chleb i jakiekolwiek inne pieczywo.
W stanach anemii i osłabienia psychicznego i fizycznego zaleca się spożywanie pierzgi.
ale jachty kojarzą mi się z mdłościami i rozpaczliwą chęcią powrotu na brzeg.
Po drodze zahaczyli o Utrecht zadowalając się zaledwie godzinnym zwiedzaniem.
Nie zrezygnujemy z wyjazdu niechby miało być zimno i deszczowo przez cały tydzień.
Jezioro Czchowskie to sztuczny zbiornik dlatego wymaga solidnego oczyszczacza.
W tajdze syberyjskiej samce tego gatunku żyją maksymalnie do pięciu lat.
które wynalazłszy suchy kawałek ziemi ogradzały ją patykami niby płotem.
Smerfetka otworzyła pudło i westchnęła z zachwytem: „Cóż to za smerfne cudeńka”.
W orszaku pluszaków pierwszy kroczył Miś uszatek wraz z prosiaczkiem i króliczkami.
Jacek zabrał się z jakimś hipisem do Maroko i spędził tam dwa miesiące.
Jad żmii zygzakowatej jest niebezpieczną mieszaniną kilku toksyn.
Rozlana na podwórko woda zamarzła więc dzieci mogły jeździć na łyżwach.
Co wieczór zdumiewało mnie dudnienie bębnów dochodzące gdzieś z dżungli.
w której tłumaczą na czym polega alzheimer i inne podobne schorzenia.
Najlepsze przyjaciółki Zośka i Gośka uprawiają jogging wspólnie od lat.
Czy promieniowanie radioaktywne ma z energią cieplną jakiś związek?
Póki mamy siły kopmy a potem będziemy mogli sobie pouzywać do woli.

Na pierwszą komunię świętą dostał psa rottweilera od rodziców.
Obecnie najpowszechniejsze formaty graficzne w komputerach to: gif.
Maleńkie bursztynki przywiózł z obozu turystycznego w Monako.
Wokalista zespołu Big Cyc dzwonił wczoraj do swojego agenta.
Nie mogłem odskrobać resztek przypalonej zupy od dna garnka
wynędzniała twarz i zapadnięte policzki pokryte szarą skórą.
Twoja ulubiona książka o damach i huzarach leży przed drzwiami.
Wizy z ograniczoną ważnością terytorialną (typ ”LTV B” or “LTV C”).
Felicyty Hufman w serialu jest mamą trójki małych potworów.
twierdząc że siniaki nabili mu Białorusini bądź Rosjanie.
Włóż duplikat do opakowania z plastiku lub owiń go folią.
Przewozili jakichś członków prezydium do Frankfurtu.
Umazał się jakąś zielonkawą mazią i wyglądał jak zombie.
Najtrafniej udało się to zrobić Magdzie i nie ma się co dziwić.
których żadne państwo nie uznaje za swoich obywateli.
Co za dziwna kobieta siedzi na tym zdjęciu w wołdze Bohdana?
że Jezioro Ełckie jest zniszczone przez działalność człowieka.
a ten stary odtwarzacz Grundiga wyrzuć na śmietnik.
Ten łysy taksiarz był naprawdę bardzo nieprzyjemny.
Markotny kapitan śledził kurs swojego statku.
Kynologia nie jest tylko nauką o tresurze psów.
gdy wybieraliśmy się grać w softball na plażę.
Czołgi w Kalkucie przerwały religijne rytuały.
Napiszcie jak nasz odźwierny poszedł do Mekki.
Dżentelmeński mężczyzna podszedł bliżej.
Mu w hollywodzkiej karierze nie pomogłam.
Bezmięsny dzień rozpoczął beztłuszczowy post.
ale ten ksiądz często takie wygłasza.
Z książki wystawała odgięta strona
Sopot i Gdańsk na pewno są miastami.
Włączyłam światła przeciwmgielne.
a wewnątrz niego znaleziono broń z brązu.

pod wnęką stał sprzęt dzwonnika
Prosiłam aby wyszedł i aby zamknął drzwi.
Mi i mężowi wasz zięć się podobał.
Przestańże szyć z zamszu te buciki.
W sen zimowy to zwierzę nie zapada.
Nie uprawiam joggingu w Miedzeszynie.
Na Półwiejskiej jest klub jazzowy.
zamiast tańszej typu „hatchback”.
Nastąpił skurcz lewej nogi.
Bank chłopski tą ceną dzwoni
Nie znam założeń szintoizmu.
Słyszałeś ten ciężki łomot?
Czy uprawiałeś kiedyś dżudo?
Ból łokcia u dżokeja to klęska
Masz dzisiaj różyczkę zieloną
Woź żyto bo zżęty mój łan
Dzwoniły dzwony w kościołach.
częstując ziemniakami.
takie już są castingi.
Nie usłyszałem dzwonka.
Przestań chrzanić.
Nie pieprzmy tego.
Widziałeś łunę?
JPG i BMP.

Appendix K.2 Word list

In the word list, on the left there is the target diphone and on the right there are the word or sequence of words in which the target diphone is present in the SAMPA notation.

p x kup chatę	ts r pomóc radą	u o to mu odświeży pamięć
p e~ ma kupę kasy	dz d nie ma władz domu	u u to mu ukryje fałdy
p o~ za pąkami	dz g Miradz Górny	e~ b ja tę bułę zjem
b d subdjakon	dz Z nie ma władz życiowych	e~ k ja tę kartę mam
b v subwarta	dz x Miradz chłodny	e~ x ja tę chatę
b x subchala	dz dz nie ma władz dzwonu	e~ dz ja tę dzwnonię
b dz' subdzień	dz dZ nie ma władz dżoany	e~ dZ ja tę dżoanę
b o~ za bąkiem	dz i Miradz i okolice	e~ dz' ja tę dźwignię
b m submonitor	dz l Miradz leśny	e~ i ja tę ikonę mam
g g zygzag głowy	dz w nie ma władz Łasku	e~ e ja tę esterę
g dz mag dzwonny	dz j Miradz jest tam.	e~ a ja tę annę
d x pod chatą	dz _ Miradz	e~ l ja tę luizę
g x tag chaty	tS tS ja Ciebie czczę zawsze	e~ r ja tę rację
g I jem gyrosa	tS r smycz radka	e~ w ja tę ławę
g m mag manny	dZ b Modż barwna	o~ z' za tą zimą
f e~ za szafę po to	dZ z Kadż z okna	o~ l za tą lampą
f o~ za szafą to jest	dZ Z Kadż że strachu	n p pan paweł
f n traf nasz	dZ z' Madż ziemna	n r pan ryszard
f w traf ławkę	dZ x Tadż chlebowa	n j pan janek
v e~ on węszy tu	dZ dz Kadż dzwonaa	n' p nie ma pań panino
s S nas sztucznie bawi	dZ dz' Kadż dziwna	n' z' nie ma pań ziobro
s s' nas się lubi	dZ e~ zmiażdżę cię	n' dz nie ma pań dzannych
s ts' nas cieszy	dZ m Kadż mała	n' dZ nie ma pań dżaro
s e~ niosę to wam	dZ n Kadż nagła	n' l nie ma pań lubianych
s n' nas nie kocha	dZ n' Kadż nie ta	n' w nie ma pań ładnych
z x maz Haliny	dZ l Kadż leniwa	l I za Lytantem
z dz to z dzbanem	dZ r Kadż rogu	l o~ na lądowniku
Z i to toż inny	dZ w Kadż łamie się	l l pal lupy
Z o~ zażądam od was	dZ _ Miradż	r d kar dawnych
Z r nie żryj tyle	ts' n' mać nie dla mnie	r z' kar zielonych
Z w toż łotra to	dz' g radź górnemu	r e~ za rękę
Z j toż ja go znam	dz' Z radź żabie	w z' był ziomem
Z _ kolarz	dz' x radź chacie	w o~ na łąkie
z' b maź błotna	dz' dz radź dzbanowi	w w był ładny
z' d maź dobra	dz' dZ radź dżoanie	j i maj inny
z' z' maź źrebaka	dz' dz' radź dziwnie	_ x chłodno tu

z' x maź chłodna	dz' o jest dzionek	_ I yyyyyyy
z' dz maź dzbana	dz' e~ jest dzięcioł	_ e~ ęęę
z' u Kaziu to zna	dz' o~ moje dziąsło	_ o~ ąąąą
z' e~ Kazię to lubię	dz' m radź mojej mamie	e-e to je eto
z' n paź na mnie	dz' n radź nam	f-a ta fatamorgana
z' l paź lubi mnie	dz' n' radź nie mnie	m-x mam chatę
z' w paź łowi	dz' l radź lamie	o-t to jest ot co nico
z' j paź jest tu	dz' w radź ładnie	o-ts na koc masz iść
z' _ to paź	dz' j radź jestem	o~-u tu są ujścia
x z' zapach ziarna	dz' _ to radź	t-j kot jest tam
x ts zapach cebuli	i dz ta mi dzwoni	u-m zakumać coś
x dz zapach dzbanu	i dZ Kamer Imidż jest dziś	v-dz coś w dzbanie
x e~ jestem chętna	I dZ to ty dżoano	w-dz był dzban
x o~ idę z zochą na spacer	I I yyy-yyy	w-i był inny
x m aa hmmm	I a to ty Adamie	z'-z zaź zamarzł
ts e~ mam pracę tą	u e te mu estera niesie	Z-z'zaż zimno

Appendix L Perception test sentences

Appendix L.1 Test 1

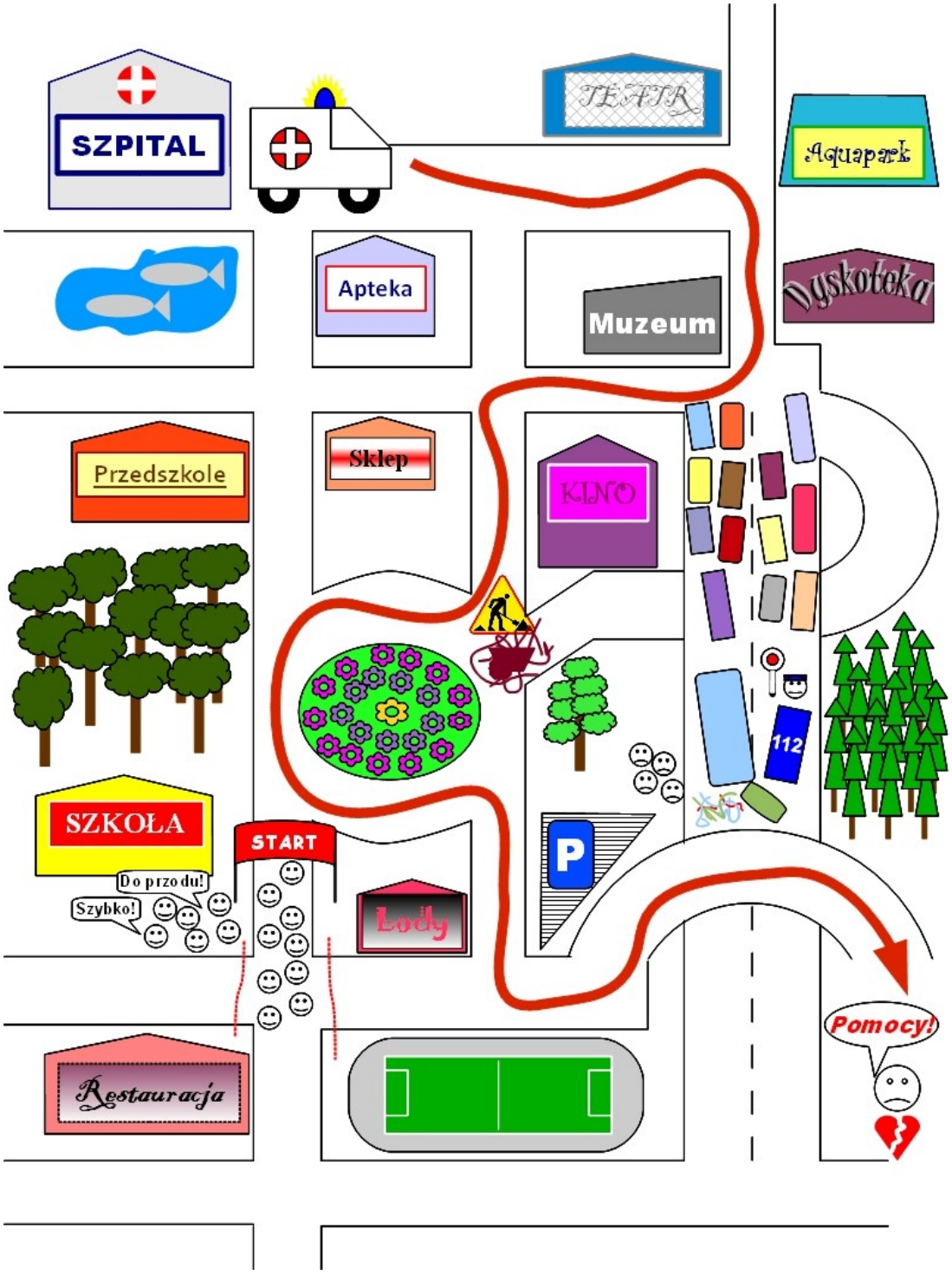
	Filename	*Sentence*	*Words*
1	B0055	Lecz są gzymsy albo gzik.	5
2	A0150	To najsprzeczniejsze zeznanie, jakie kiedykolwiek słyszałem.	6
3	B0080	Widzą chrzan biały na rzęsach.	5
4	A0280	Wszystkie dzieci kochają wakacje.	4
5	A0210	Jego dzisiejszy występ niewątpliwie potwierdził jego ogromny talent.	8
6	A0033	Odgnieciony ślad głowy był wyraźnie na poduszce.	7
7	A0060	Wszedłszy do biura, spostrzegłam, że ktoś grzebał w moich dokumentach.	10
8	B0030	Książę daj pół ziemi i siostrę.	6
9	B0113	Najjaśniej gada z tą lalką dziś.	6
10	B0105	Tęsknił żigolak pod żlebem.	4
11	A0310	Wejście do budynku jest wzbronione.	5
12	A0250	W nocy spadła świeża warstwa śniegu i poranny krajobraz wyglądał olśniewająco.	11
13	B0060	Lecz późną nocką idą raźniej.	5
14	A0240	Dziewka umyła gliniane garnczki w strumyku i położyła na zielonej trawie do wyschnięcia.	13
15	B0075	Wal po tym czymś stopą.	5
16	B0015	Boś cały w wiśniowym soku.	5
17	A0360	Nie znam się na literaturoznawstwie.	5
18	B0045	Obcy ptak co drzemał na pniu.	6
19	A0341	Wyszłam na spacer z psem półzmierzchem.	6
20	B0005	Móc czuć każdy odczynnik.	4
		Total	***126***

Appendix L.2 Test 2

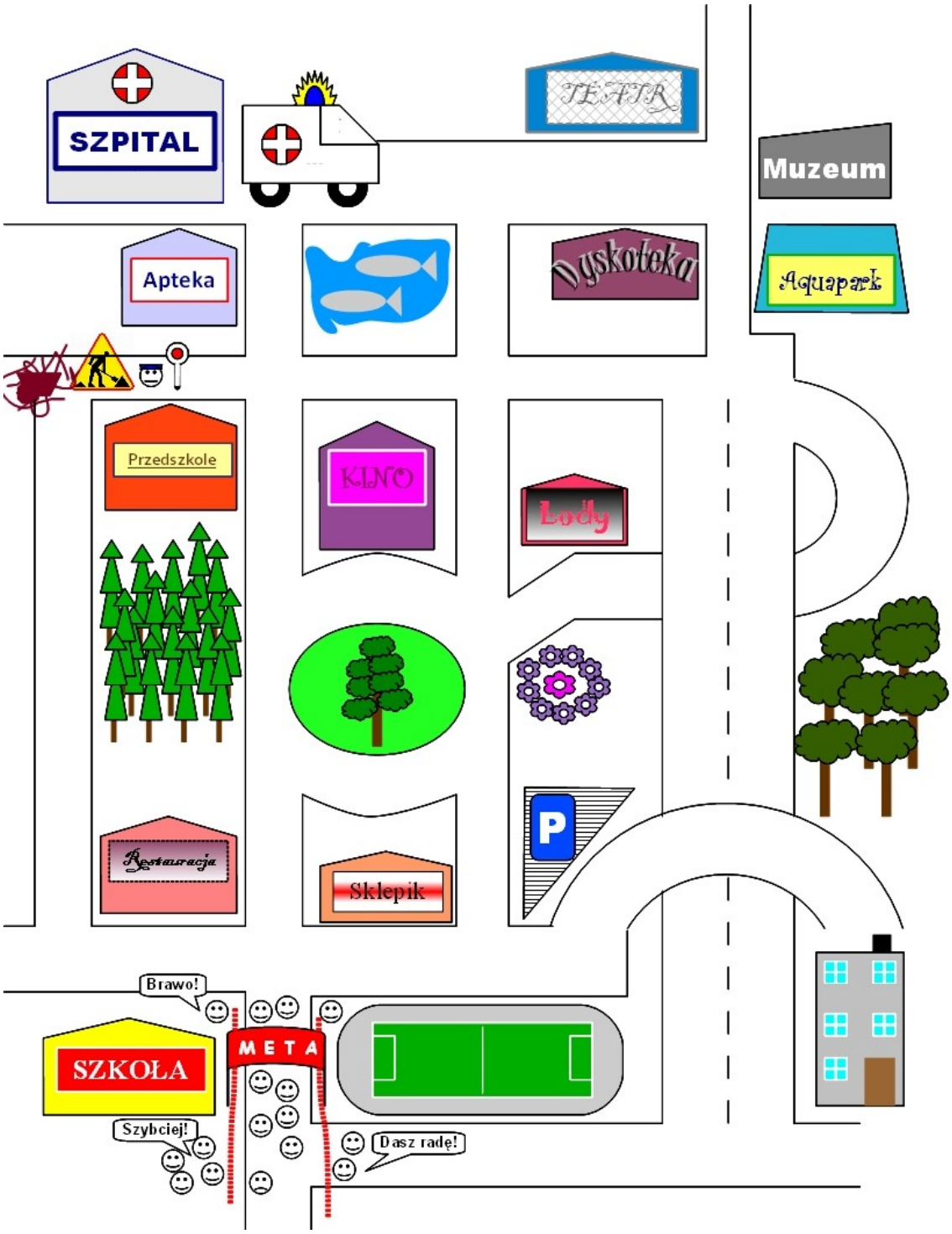
	Filename	***Sentence***	***Voice***
1	E0500	To małe, martwe zwierzę, które mama znalazła na chodniku było prawdopodobnie ofiarą tegorocznego mrozu.	female PL1
2	E0470	Męczy mnie, kiedy moja współlokatorka całymi dniami narzeka na wszystko wokół niej.	original
3	E0459	Wolał pozostać uczciwym robotnikiem bez kasy niż wspinać się po szczeblach kariery oszukując ludzi.	female PL1
4	E0440	Herbata to ulubiony napój na śniadanie wśród wielu Polaków.	male PL2
5	E0481	Dermatolog nie dał mi gwarancji, że ten nowy krem nie spowoduje wysypki na mojej skórze.	male PL2
6	E0459	Wolał pozostać uczciwym robotnikiem bez kasy niż wspinać się po szczeblach kariery oszukując ludzi.	original
7	E0500	To małe, martwe zwierzę, które mama znalazła na chodniku było prawdopodobnie ofiarą tegorocznego mrozu.	original
8	E0459	Wolał pozostać uczciwym robotnikiem bez kasy niż wspinać się po szczeblach kariery oszukując ludzi.	male PL2
9	E0481	Dermatolog nie dał mi gwarancji, że ten nowy krem nie spowoduje wysypki na mojej skórze.	female PL1
10	E0440	Herbata to ulubiony napój na śniadanie wśród wielu Polaków.	original
11	E0470	Męczy mnie, kiedy moja współlokatorka całymi dniami narzeka na wszystko wokół niej.	male PL2
12	E0440	Herbata to ulubiony napój na śniadanie wśród wielu Polaków.	female PL1
13	E0470	Męczy mnie, kiedy moja współlokatorka całymi dniami narzeka na wszystko wokół niej.	female PL1
14	E0481	Dermatolog nie dał mi gwarancji, że ten nowy krem nie spowoduje wysypki na mojej skórze.	original
15	E0500	To małe, martwe zwierzę, które mama znalazła na chodniku było prawdopodobnie ofiarą tegorocznego mrozu.	male PL2

Appendix M Map task: emergency scenario

Appendix M.1 The map for the leading person

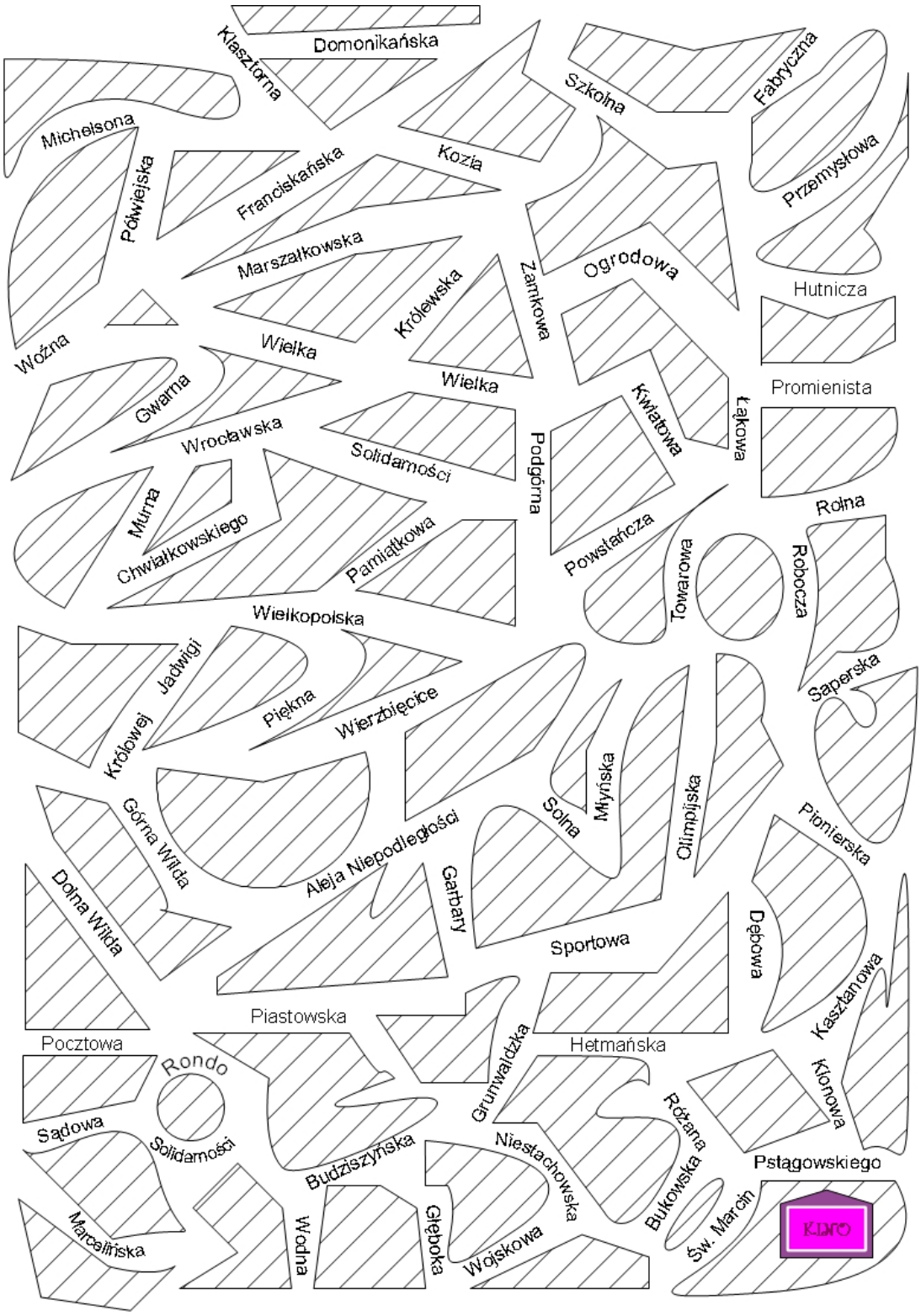


Appendix M.2 The map for the following person

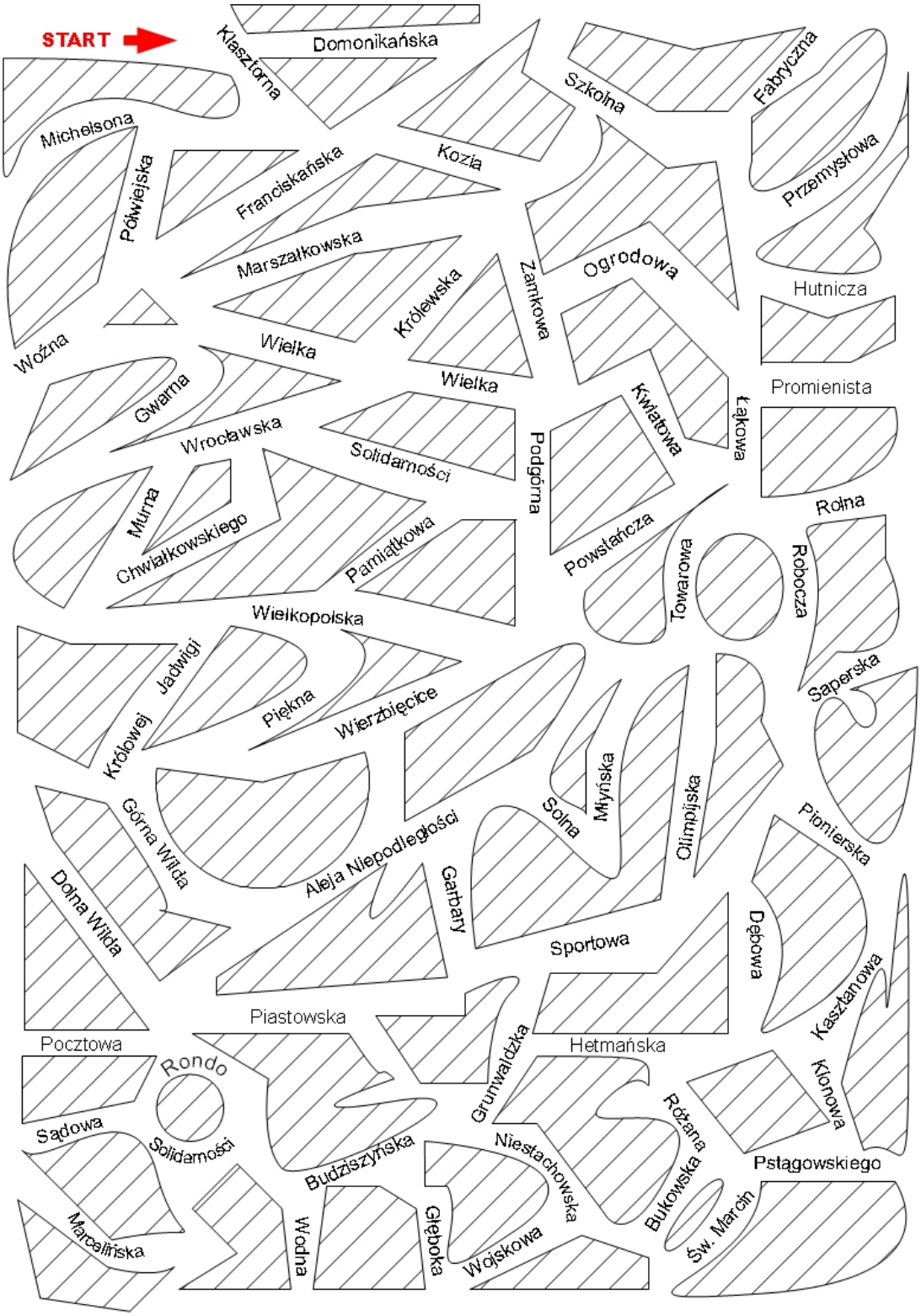


Appendix N Map task: neutral scenario

Appendix N.1 The map for the leading person



Appendix N.2 The map for the following person



Appendix O Draw wavform, pitch and annotation for stereo sounds – Praat script

```
## By David W. at Interspeech 2009
## Modified to stereo by Jolanta Bachan, 2011-03-12
## select a Sound, Textgrid & Pitch together
form Draw
  real Xmin 0
  real Xmax 0
  real Pitchmin 0
  real Pitchmax 500
endform

s = selected ("Sound")
p = selected ("Pitch", 1)
p2 = selected ("Pitch", 2)
t = selected ("TextGrid")

Erase all
vpx1 = 0.5
vpx2 = 10
vpy1 = 0.5
vpy2 = vpy1 + 2
Select inner viewport... vpx1 vpx2 vpy1 vpy2
select s
Draw... xmin xmax -1 1 n Curve
Draw inner box
vpy1 = vpy2
vpy2 = vpy1 + 1
Select inner viewport... vpx1 vpx2 vpy1 vpy2
select p
Draw... xmin xmax pitchmin pitchmax n
Draw inner box
vpy1 = vpy2
vpy2 = vpy1 + 1
Select inner viewport... vpx1 vpx2 vpy1 vpy2
select p2
Draw... xmin xmax pitchmin pitchmax n
Draw inner box
```

```
vpy1 = vpy2
vpy2 = vpy1 + 2
Select inner viewport... vpx1 vpx2 vpy1 vpy2
select t
Draw... xmin xmax n y y
#Marks bottom every... 1 0.5 y y n

Select inner viewport... vpx1 vpx2 0.5 vpy2
Marks bottom every... 1 0.5 y y y
```

Appendix P Demonstration dialogue system script

```
#!/usr/bin/python
# demoDialogueSystem-informal-notext.py
# Jolanta Bachan 2011-04-14

import sys, os, random
from datetime import datetime, date, time
#-----------------------------
states=[['q0','a','q1'],['q1','a','q0'],['q0','f','q5'],
['q5','f','q0'],['q1','b','q2'],['q2','b','q1'],['q1','e','q6'],
['q6','e','q1'],['q2','c','q3'],['q3','c','q2'],['q2','d','q7'],
['q7','d','q2'],['q4','g','q5'],['q5','g','q4'],['q5','h','q6'],
['q6','h','q5'],['q5','k','q8'],['q8','k','q5'],['q6','i','q7'],
['q7','i','q6'],['q6','l','q9'],['q9','l','q6'],['q8','n','q9'],
['q9','n','q8'],['q8','p','q10'],['q10','p','q8'],['q10','s','q11'],
['q11','s','q10'],['q11','u','q12'],['q12','u','q11'],
['q12','w','q13']]
streets =
['a','b','c','d','e','f','g','h','i','j','k','l','m','n','o','p','q'
,'r','s','t','u','v','w','x','y','z']
streets = set(streets)
posreplies = ['a-potem.wav','acha-acha-dobrze.wav','dobrze-
tak.wav','tak-nastepnie.wav','acha-acha.wav','tak-okej.wav','acha-
jazda-prosto.wav','i-wtedy.wav','acha.wav','no-dobrze.wav','tak-
rozumiem.wav','tak-tak-rozumiem.wav','dobra-
okej.wav','tak.wav','dobra.wav','okej.wav','yyy-dobrze-
dalej.wav','dobrze-dobrze-oczywiscie.wav','prosto.wav','yyy-
tak.wav','dobrze-dobrze-tak-rozumiem.wav','rozumiem-yyy-prosto.wav']
negreplyNoStreet = ['czyli-nie-rozumiem.wav','yyy-problem-nie-ma-
ulicy.wav','nie-nie-nie-ma-ulicy.wav','jeszcze-raz.wav','yyy-
jeszcze-raz-prosze.wav','yyy-no-nie-ma-ulicy.wav','yyy-nie-znam-
ulicy.wav']
negreplyWrongWay = ['no-chyba-zmiana-nie-moge.wav', 'no-nie-
moge.wav','ojej-nie-moge.wav','e-e-nie-ma-wejscia.wav']
playcommand = ''
introreply = ''
allmoves = []
#-----------------------------
def instructions():
    print 'INSTRUKCJE:'
    print 'Wyobraz sobie, ze rozmawiasz z osoba z centrali pogotowia
ratunkowego. Twoim zadaniem jest poprowadzenie karetki pogotowia ze
szpitala do osoby z zawalem serca wzdluz ulic zaznaczonych na mapie.
Wpisz nazwe ulicy, aby przesunac karetke od skrzyzowania do
skrzyzowania. UWAGA: Nie wszystkie drogi sa przejezdne! System
bedzie gotowy na Twoja odpowiedz, jesli na ekranie pojawi sie:
-----------------------------'
    print 'Dodatkowe opcje:'
    print 'co - powtorzenie ostatniej wypowiedzi'
    print '? - wyswietlenie informacji, w ktorym miejscu jest
karetka'
    print 'exit - wyjscie z programu'
    raw_input()
#-----------------------------
def opening():
    d = datetime.now()
```

```
    introreplies = []
    introreplies.append(str(d))
    playcommand = 'play -q Other/dzien-dobry-witam.wav'
    os.system(playcommand)
    playcommand = 'play -q Other/jak-sie-nazywasz.wav'
    os.system(playcommand)
#    print "Jak sie nazywasz?"
    print '------------------------------'
    imie = raw_input()
    introreplies.append(imie)
    playcommand = 'play -q Other/kobieta-mezczyzna.wav'
    os.system(playcommand)
#    print "Kobieta czy mezczyzna?"
    print '------------------------------'
    i = 0
    n = 1
    while i<n:
        print "k/m"
        km = raw_input()
        if km == 'm':
            playcommand = 'play -q Other/szpital-male.wav'
            i = 2
        elif km == 'k':
            playcommand = 'play -q Other/szpital-female.wav'
            i = 2
        elif km == 'exit':
            exit()
        introreplies.append(km)
        os.system(playcommand)
        print '------------------------------'
    i = 0
    n = 1
    while i < n:
        instring = raw_input()
        introreplies.append(instring)
        if instring == 'tak':
            i = 2
        elif instring == 'exit':
            exit()
        else:
             os.system(playcommand)
        print '------------------------------'
    if km == 'm':
        playcommand = 'play -q Other/prosba-male.wav'
    elif km == 'k':
        playcommand = 'play -q Other/prosba-female.wav'
    os.system(playcommand)
#    print 'Gdzie chcesz isc?'
    print '------------------------------'
    return playcommand,  introreplies
#------------------------------
def dialogue(states, streets, posreplies, negreplyNoStreet,
negreplyWrongWay, introreply):
    playcommand = introreply
    inputstrings = []
    i = 0
    n = 1
```

```
    path = []
    state = 'q0'
    finalstate = 'q13'
    while i<n:
        instring = raw_input()
        inputstrings.append(instring)
        if instring == 'co':
            os.system(playcommand)
#            print "A teraz gdzie?"
            print '------------------------------'
        elif instring == 'exit':
#            print path
            break
        elif instring == '?':
            print path
            print 'Karetka jest tu: '+ state + '!'
            os.system('display Maps/maptask-states-small.jpg')
        elif instring not in streets:
#            print 'Nie znam.'
            replyindex = random.randint(0,6)
            playcommand = 'play -q NegFeedback/' +
negreplyNoStreet[replyindex]
            os.system(playcommand)
            print '------------------------------'
        else:
            done = False
            for currstate in states:
                if currstate[0]==state and currstate[1]==instring:
                    path.append(instring)
                    state = currstate[2]
                    replyindex = random.randint(0,21)
                    playcommand = 'play -q PosFeedback/' +
posreplies[replyindex]
                    os.system(playcommand)
                    done = True
                    if state == finalstate:
                        i=2
                    break
            if done == False:
#                print 'Sprobuj jeszcze raz.'
                replyindex = random.randint(0,3)
                playcommand = 'play -q NegFeedback/' +
negreplyWrongWay[replyindex]
                os.system(playcommand)
       #         print '------------------------------'
#            print "Gdzie teraz?"
            print '------------------------------'
    return path, inputstrings

#------------------------------
def closing(initreplies, yourpath, allmoves):
    os.system('play -q Other/dziekuje.wav')
    a = len(initreplies)
    b = len(allmoves)
    print 'Twoja sciezka: ', yourpath
    print 'Ilosc Twoich odpowiedzi: ', a+b-1
    print 'Odpowiedzi na wstepie: ' , initreplies
```

```
    print 'Odpowiedzi dotyczace drogi: ' ,allmoves
    d = datetime.now()
    outputfile = open('answers.txt', 'a')
    allanswers = []
    pathstring = ''
    allanswers.append('Duration: ')
    for p in yourpath:
        pathstring = pathstring + p + ' '
    allanswers.append(pathstring)
    allanswers.append('--------------------------')
    initstring = ''
    for init in initreplies:
        initstring = initstring + init + ' '
    allanswers.append(initstring)
    allanswers.append('--------------------------')
    allstring = ''
    for al in allmoves:
        allstring = allstring + al + ' '
    allanswers.append(allstring)
    allanswers.append('--------------------------')
    allanswers.append(str(a+b-1))
    allanswers.append(str(d))
    allanswers.append('##########################')
    for item in allanswers:
        item = item + '\n'
        outputfile.write(item)
    outputfile.close()

#------------------------------
def main():
#    instructions()
    introreply, initreplies = opening()
    yourpath, allmoves = dialogue(states,streets, posreplies,
negreplyNoStreet, negreplyWrongWay, introreply)
    closing(initreplies, yourpath, allmoves)

#------------------------------
main()
```