MASARYKOVA UNIVERSITA
FAKULTA INFORMATIKY

Anaphora Resolution

DIPLOMOVÁ PRÁCE

Václav Němčík

Brno, 2006

Declaration

I declare that this thesis is an original piece of work I produced myself. All the various sources I used while writing this work are properly cited (with a full reference) within.

Prohlášení

Prohlašuji, že tato práce je mým původním autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používal nebo z nich čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Acknowledgements

I would like to thank my supervisor, Dr. Ing. Ivana Kruijff-Korbayová, who has always provided me with expert advice and valuable hints and references. I am very grateful to my parents for their immense support, not only during the studies, but throughout my whole life. I would also like to thank doc. PhDr. Karel Pala, CSc. for his ongoing support and guidance. The work would be also hardly possible without Zdeněk Žabokrtský, who provided me with the evaluation data and the TrEd editor. I also greatly appreciate the help of James Thomas, M.A. and his hints concerning academic writing. Finally, I would like to express my thanks to all my friends for their help and tolerance throughout.

Abstract

One of the challenges in natural language understanding is to determine what entities are referred to in the discourse and how they relate to each other. This is a very complex task. But, as the first step, it is useful to determine coreference classes over the set of referring expressions. This thesis presents a system that perfoms automatic coreference resolution on syntactic basis. The system allows the realization of various AR algorithms in a modular way and can be used, in principle, with any natural language. The functionality of the system is shown on selected salience-based algorithms customized for Czech. Their performance is evaluated and a way how to combine them into a more successful meta-algorithm is proposed.

Keywords

anaphora, anaphora resolution, coreference, antecedent, discourse modeling

Contents

Chapter 1

Introduction

At present, resolution of anaphoric reference is one of the most challenging tasks in the field of Natural Language Processing (NLP). The ongoing research pursues mainly construction of automatic mechanisms for resolving pronouns. It is extremely difficult to give a complete, plausible and computable description of the resolution process, because we ourselves deal with it only subconsciously and are largely unaware of the particularities. This effortlessness contrasts with the fact that the underlying theory is very complex. The task of anaphora resolution is even frequently considered to be AI-complete.[1]

Nevertheless, anaphora resolution needs to be addressed in almost every application dealing with natural language (e.g. dialogue systems, systems for machine translation, or information extraction). Moreover, to keep the concerned application computationally feasible, it has to be addressed efficiently, which makes it impossible to use all resources and make all inferences known to be necessary for the proper treatment of certain cases. A trade-off between the plausibility of the model and its computational complexity needs to be reached. It is advantageous to consider the fact that many natural language systems operate over a restricted part of language and it is thus sufficient to handle instances common in the given domain.

The Goals of This Work

In this work, I decided to build a framework for performing anaphora resolution which is modular in many ways. As each application puts almost unique demands on priorities and adjustments of the anaphora resolution process, my framework aims at facilitating experimentation with various algorithms, their evaluation using various metrics, and allowing easy use of already implemented algorithms with other languages and data formats. This is achieved mainly by defining separate modules for individual phases of processing. The architecture of the framework is in accord with the principles formulated by Byron and Tetreault (1999) for their own system.

[1] This analogy (first used by Fanya S. Montalvo) to the term *NP-complete* in computer science expresses that solving the given computational problem is equivalent to constructing artificial intelligence, i.e. to making computers think.

Further, the framework is used to resolve textual pronominal anaphora in Czech. Although there is a number of AR algorithms formulated specifically for Czech, there has been little effort to implement, evaluate and compare them in a systematic way, mainly due to inavailability of annotated data. The existence of Prague Dependency TreeBank, now annotated for coreference, has made it possible to pursue these goals in this thesis.

However, space and time don't permit addressing AR in its full complexity and render it necessary to limit the scope of the work considerably. In the implementations, I decided not to resolve grammatical coreference. It requires elaborate investigation of the syntactic relations of the individual sentences and as it is highly formalism- and language- specific, it is of advantage to address it within the process of syntactic analysis. On the contrary, principles of textual anaphora concern phenomena widely applicable across languages.

This work concentrates mainly on traditional algorithms based on the concept of salience, and aims to evaluate their performance, and compare their strong and weak points. Finally, I will try to improve their performance by combining them into a certain form of meta-algorithm.

I chose to address salience-based algorithms for several reasons. The most prominent one is, that the algorithms of the Prague group, formulated specifically for Czech, all fall into this category. Further, their models are computationally feasible and transparent, which facilitates interpretation of resolution errors.

Further in this chapter, I summarize basic terminology related to AR, briefly resume relevant theoretical background and remark on the importance of AR in NLP applications. In the next chapter, I describe some of the most influential AR approaches and algorithms, which is followed by chapter 3, discussing the implementation of the proposed system and selected algorithms. Chapter 4 is concerned with selected methods for evaluating AR algorithms. Next, in chapter 5, I present and discuss performance of the implemented algorithms, and finally chapter 6 concludes the text with a brief recapitulation of the work and directions for further research.

1.1 Basic Definitions

This section provides defintions of basic concepts relevant for investigation of discourse and anaphora. Further, common types of anaphora are mentioned, followed by remarks on relevant theoretical background in the next section. Let's commence with an example:

(1) a. The thousand injuries of Fortunato I had borne as I best could;
b. but when he ventured upon insult, I vowed revenge.
c. You, who so well know the nature of my soul, will not suppose, however,
d. that I gave utterance to a threat.
e. At length, I would be avenged;
f. this was a point definitively settled ...

Without any doubt, this excerpt[2] from a story by Edgar Allan Poe is a text, or as linguists more generally term it, a **discourse**. Discourse can be understood as a unit of speech, or more generally, of communication[3] among possibly more participants (Čermák, 1997). An important question at this point is, whether any sequence of utterances forms a discourse.

(2) I wonder why she bought oranges in the end.

When we consider (2) as a possible continuation of (1), it can be clearly seen that the whole doesn't add up to a proper discourse. To word the intuitively obvious reasons for this, the sequence doesn't exhibit two qualities a discourse necessarily needs to have: coherence and cohesion. For instance, Bußmann (2002) defines them as follows:

- **coherence** – semantic and cognitive integrity of the meaning of the text
- **cohesion** – manifest properties of the text providing the reader with indications of the connectedness of the text parts

The above-mentioned sequence is not coherent, because it leaves the reader confused about how does a woman buying oranges relate to the speaker avenging himself. Unless very specific circumstances occur, the juxtaposition of these pieces of information is not interpretable. The sequence is also not cohesive, because linguistic cues in (2) do not connect with anything in (1), in particular, it is not clear who does "she" refer to and which events or circumstances relate to the phrase "in the end". However, (2) can be very well

[2] The excerpt is taken from the story "The Cask of Amontillado", published for instance in (Poe, 1994).

[3] For the sake of clarity and easiness of formulation, let's henceforth consider only communication between two communication participants and adopt the following assumption about them: the speaker will be referred to as to a female person (let's say, Susan) and the hearer will be referred to as to a male person (let's say, Henry). The terms *speaker* and *hearer* will be used regardless of whether the communication in question is spoken or written.

a part of a different discourse (e.g. where a narration about people obtaining food for a party precedes).

The means responsible for establishing cohesion and coherence relate closely to **reference** – the relation between a linguistic expression and the corresponding entity in the extra-linguistic reality, the so-called **referent** (Bußmann, 2002). Expressions having a referent are termed **referring expressions**. Modes of reference in its broader sense can be classified into:

- **exophora** (outer reference) – means of identifying referring expressions directly with the objects in the outer world; more specifically (Čermák, 1997) **deixis**, the use of gestures and linguistic expressions to refer to the elements of the communicative situation (Bußmann, 2002)
- **endophora** (inner reference) – means of relating referring expressions to other expressions within the discourse (Čermák, 1997)

Reference is a very complicated concept and it is beyond the scope of this work to present a complete fine-grained classification of its types.[4] For further subtleties and details on philosophical aspects of reference, I recommend referring to (Moore, 1993).

Anaphora (from the Greek $\alpha\nu\alpha\varphi o\rho\alpha$, "carrying back") is a special type of endophora relating an expression to another expression *preceding it* in the discourse (Čermák, 1997). In addition, the word or phrase "pointing back" is an **anaphor** and the expression it relates to is its **antecedent**. **Anaphora resolution** is the process of determining the antecedent of an anaphor (Mitkov, 2002).

When two referring expressions (esp. an anaphor and its antecedent) have the same referent, they are said to **corefer**, i.e. to be coreferential.[5] The corresponding relation (equivalence) is called **coreference**. Subsequent occurrences of coreferential referring expressions form a so-called **coreferential chain**. The following excerpt[6] contains an example of a coreferential chain (anaphors are in italics, their antecedents in bold; further, coreferring expressions have matching indices):

(3) a. It was about dusk, one evening during the supreme madness of the carnival season, that I encountered **my friend**$_i$.

b. ***He***$_i$ accosted me with excessive warmth, for ***he***$_i$ had been drinking much.

c. ***The man***$_i$ wore motley.

d. *He*$_i$ had on a tight-fitting, parti-striped dress . . .

[4] For instance, Halliday and Hasan (1976) define **homophora** – a specific kind of exophora requiring certain cultural or other knowledge to identify the object in the real world.

[5] It is necessary to distinguish between the anaphor's referent and antecedent. The antecedent is a linguistic expression it is related to, whereas the referent is the object in the real word the anaphor (and possibly, but not necessarily, also the antecedent) refers to.

[6] The excerpt is taken from (Poe, 1994).

It is possible to distinguish among many types of anaphora, based on various criteria. One basic disctinction is between **direct anaphora**, relating the anaphor to its antecedent by direct reference, and **indirect anaphora** (often termed **bridging** or **associative anaphora**), expressing a link to the antecedent involving a certain amount of knowledge or inference. Indirect anaphora very often relates the anaphor to the antecedent through hyponymy, hypernymy, meronymy, etc., but the association may be much looser and may require almost any world knowledge. The following example[7] contains two bridging links.

(4) a. From an iron staple depended **a short chain**, from another **a padlock**.
b. Throwing *the links* about Fortunato's waist, it was but the work of a few seconds to secure it.
c. Withdrawing *the key* I stepped back.

The first instance of bridging relates the expression "the links" to its antecedent ("a shorter chain") through meronymy. The second link associates "the key" in (4c) with the phrase "a padlock". In this case, the relationship is somewhat harder to define. A key and a padlock are merely two instruments conventionally used together to bring about a certain result.

Anaphora where the anaphor and the antecedent are coreferent is termed **identity-of-reference anaphora**. However, there are anaphors referring to a different instance of the same entity, or a very similar entity as the antecedent; analogically, anaphors referring to a subset, or an element of the set referred to by the antecedent. Such anaphora is called **identity-of-sense anaphora**. The most widely known example of this type of anaphora are the so-called paycheck sentences (one example is also in (4a)):

(5) The man who gave his **paycheck** to his wife was wiser than the man who gave *it* to his mistress.

Hajičová, Panevová, and Sgall (1985) mention an important distinction between **grammatical** and **textual anaphora**. Grammatical anaphora is conditioned by the rules of grammar (such as relative pronouns or control) and occurs in rather regular constructions. Textual anaphora is, from this point of view, indistinct. It is important to distinguish these types of anaphora types because it may be useful to adopt different strategies for their resolution. Grammatical anaphora can be resolved within the syntactic analysis of the sentence, whereas textual anaphora needs to be resolved based on a so-called **discourse model**. The discourse model is meant to correspond to the hearer's mental model of the ongoing discourse and is usually built incrementally.

Probably the most widely used classification of anaphora is based on the syntactic category of the anaphor:[8]

[7] The example is adapted from (Poe, 1994).

[8] The clasification is adopted from (Mitkov, 2002).

- **pronominal anaphora** covers cases where the anaphor is a pronoun. This is the most addressed anaphora type. Usually only third person pronouns are considered, because first and second person pronouns are understood to be deictic. Brief characterization of anaphoric pronouns can be found in subsection 1.2.1. An example of pronominal anaphora is for instance in (3).

- **definite descriptions** cover, apart from pronouns, also anaphors realized by definite NPs and proper nouns. These carry, compared to pronouns, more semantic content and are therefore often used to refer to the antecedent and at the same time elaborate on it. An example of a definite description is in (3c) and (6).

(6) a. From the beginning of June, a new airline will be operating to **Brno**$_i$.
b. It is expected to bring new tourists to *the second biggest Czech city*$_i$.

- **one anaphora** is a subcase of identity-of-sense anaphora where a non-lexical pro-form refers to the head or the first projection[9] of an NP.

(7) Petr koupil Janě pětadvacet **rudých růží**. *Jednu* bych také chtěla.
Petr bought to Jana twenty-five red roses. One I would too want.
"Petr bought Jana twenty-five red roses. I would like one too."

- **verb anaphora** relates a reduced VP to its full realization:

(8) Pavel **jel** **včera** **do Prahy**. Libor ∅ také.
Pavel went yesterday to Prague. Libor too.
"Pavel went to Prague yesterday. So did Libor."

- **zero anaphora (ellipsis)** is not a further autonomous class of anaphora but rather a set of special cases of the previously mentioned anaphor types. It comprises all instances referring through the absence of anaphor's surface form; for instance zero-subject in (9) or an empty verb in (8).[10]

(9) **Petr**$_i$ koupil **Janě**$_j$ pětadvacet rudých růží. $\emptyset_i$ miluje *ji*$_j$.
Petr bought to Jana twenty-five red roses. He loves her.
"Petr bought Jana twenty-five red roses. He loves her."

[9] The terminology of the X-bar theory is explained in (Jackendoff, 1977).

[10] In the given examples, the presence of a zero anaphor is indicated by a ∅ symbol.

The last classification to be mentioned here is the classification according to the position of the antecedent. When the antecedent is in the same sentence as the anaphor, we speak about **intrasentential anaphora**, otherwise about **extrasentential anaphora**. Intrasentential anaphora is more constrained by syntactic relations within the sentence and is often grammatical.

A very special case is **cataphora**, arising when reference to an entity mentioned subsequently in the text is made (Mitkov, 2002). Technically said, it is not anaphora, but the type of endophora converse to it, nevertheless, it is often handled as a very special type of anaphora. Cataphora is almost always intrasentential and is usually realized by a cataphoric pronoun, located in an embedded relative clause.

(10) Než $\emptyset_i$ stačil dorazit domů, byl **Petr**$_i$ úplně promoklý.
Before he managed to arrive home, was Petr entirely wet.
"Before he managed to get home, Petr was entirely wet."

1.2 AR and Linguistic Theory

This section offers several brief remarks on how theory of individual levels of language description relate to anaphora and anaphora resolution. Unfortunately, space doesn't permit mentioning all the relevant theory, only giving rather loose examples of some interesting interrelations.

1.2.1 Morphology and Syntax

Morphology and syntax provide us with various means of classifying anaphoric expressions. Let's state at least the basic types of anaphoric pronouns in Czech:

- **strong personal pronouns** (e.g. "jemu", "on", "ona")
  are realized by a standard, stressed, form of a personal pronoun.
- **weak personal pronouns** (e.g. "mu", "ho")
  are realized by an unstressed form of a personal pronoun – a **clitic**, which means it takes the Wackernagel position in the sentence (i.e. after the first constituent).
- **zero personal pronouns** ("$\emptyset$")
  usually realize the unvoiced subject of the clause – the morphology features are identifiable through the finite verb form endings.
- **demonstrative pronouns**[11] (e.g. "ten", "ta", "tomu")

[11] Demonstratives often appear as determiners in NPs like "tento vlak" ("this train"). In such cases, only the whole NP is anaphoric, but not the demonstrative *as such*.

refer rather unpredictably – they sometimes refer anaphorically to the last mentioned object, but often to an abstract entity, or are even deictic.

- **reflexive pronouns** (e.g. "se", "sebe", "svůj")
  are generally coreferent with the subject of the clause.[12]

- **possessive pronouns** (e.g. "jeho", "jejího")
  in Czech must agree in gender and number with the possessed entity (through the case ending), but also with the possessor (through lemma, or base form). However, this is very language-specific.[13]

- **relative pronouns** (e.g. "který", "jenž")

However, not all pronoun occurrences are anaphoric. Pronouns that are not used referentially are called **pleonastic** or **expletive** pronouns. In many languages, subject in every clause is required to be syntactically realized, even when it is semantically empty. This typically (e.g. in Germanic languages) concerns sentences about time and weather, and certain other constructions:

(11) **Es** ist eiskalt.
It is freezing cold

(12) **Es** ist zu spät.
It is too late.

(13) **Es** ist klar, dass er es getan hat.
It is clear, that he did it.

(14) Byl **to** Petr, kdo odešel.
Was **it** Petr, who left.
"It was Petr, who left."

Cleft constructions, like in sentence (14), make it possible to express a different focus[14] than the one resulting from the neutral word order. This is important in languages with a fixed word order, like English.

1.2.2 Semantics

Anaphoric pronouns are **synsematic** words, i.e. words that have no lexical meaning of their own. Indeed, at first glance, anaphoric pronouns don't seem to contain any meaning

[12] However, section 3.4.2 contains sentences that *can* be seen as counterexamples.

[13] In Frech, for instance, gender and number agreement links the possessive pronoun only to the possessed entity. There is no gender agreement between a possessive pronoun and the respective possessor.

[14] Refer to 1.2.4 for further details on Topic-Focus Articulation.

at all. However, their "meaning" can be perceived as *"I am to be identified with the most salient object in the previous discourse plausible in terms of grammar and semantics"*.

It is apparent that semantics plays a considerable role in anaphora resolution. Frequently, the choice of the correct antecedent relies heavily on its semantic plausibility – whether its semantic features comply or contradict with the anaphor's context. In many cases, the relevant semantic relations are rather intricate, as illustrated by the following example[15]:

(15) John hid Bill's keys. He was drunk.

The preferred interpretation is that *"He"* refers to Bill and that John hid his keys, because he didn't want him to drive while he was drunk. Establishing this link requires vast representation of world knowledge (for instance, that car keys make it possible to drive a car, or that drunk driving is dangerous), and desires of individual people (e.g. that John cares about Bill, doesn't want anything bad happening to him and is ready to act to prevent situations that are likely to end up bad for him).

Unfortunately, this explanation for John hiding Bill's keys is not the only one. It is possible to combine a number of other facts to reach the same conclusion (or possibly the inverse). Availability of a knowledge base raises the question where to start the inference and which direction it should be led. It is computationally unfeasible to try out all possibilities, and currently, no generally adequate strategy for constraining the search space is known. This remains to be the main obstacle of employing inference in AR.

Nevertheless, it is possible to take advantage of more straightforward semantic knowledge. In some domains, certain settled procedures are of considerable importance and it is of advantage to formalize them by means of so-called **scenarios**. A scenario contains the individual actions of the event in question, their order and the relationships among them. This information can help us understand (and resolve) anaphors referring to objects salient solely through the situation, for instance, a bill in a restaurant. Although it hasn't been mentioned in the discourse, after the guest finishes their meal it instantly becomes relevant. Both the guest and the waiter know that paying the bill is the next step to be done and can possibly refer to it by a pronoun.

A further useful resource is also a list of **valency frames**. Each already disambiguated verb occurrence refers to a certain event or relation and the corresponding valency frame specifies which arguments the verb can take and the role each of them plays in the described event. This knowledge helps identifying the phrases containing the individual participants of the concerned event and assigning each to the proper position in the semantic representation. Further, the semantic knowledge about each valency slot usually constrains the type of entity which can play the corresponding role. For instance, the literal meaning of the verb *"to drink"* requires the agent to be animate and the patient to

[15] The example is adopted from (Jurafsky and Martin, 2000).

be a liquid. This can help to rule out antecedent candidates by considering their semantic plausibility.

The last, but not least, resource relevant for the AR process to mention here is **WordNet**, and ontologies in general. Ad hoc ontologies are worthwhile for example when constructing dialogue systems with a restricted domain. WordNet is a useful resource for resolving definite descriptions and dealing with bridging. A detailed account can be found for example in the work of Vieira and Poesio (2001), who performed extensive research about the application of WordNet on the resolution of definite descriptions.

1.2.3 Pragmatics

There is no precise definition of pragmatics, but generally speaking, it studies the use of language in communicative situations (Bußmann, 2002). The communicative situation has clearly a considerable impact on the process of AR. It describes the individual participants, their knowledge, beliefs and intentions, the spatial and temporal setting of the communication, cultural factors etc.[16]

The knowledge about the current communicative situation, as discussed in the previous subsection, is necessary to support certain kinds of inference. It is also a prerequisite for resolving deictic reference, brought about for instance by demonstrative pronouns, first and second person pronouns, or temporal and local adverbs. This is a crucial issue especially for dialogue systems or the interpretation of direct speech in texts.

However, there seem to be certain universal principles, which can be assumed to be valid in every communicative situation. The communication participants are expected to be cooperative and to intuitively help each other so that the communication is efficient and successful. Grice (1975) worded this as **the cooperative principle** and formulated it by means of guidelines for a successful communication, known as **conversational maxims**:

- **maxim of quality**
  Say the truth. Don't say anything you disbelieve or lack adequate evidence for.

- **maxim of quantity**
  Be as informative as required by the purpose of the current exchange, but don't be overly informative.

- **maxim of relevance**
  Be relevant.

[16] Čermák (1997) argues that the term **context** can be sometimes used as a synonym for a communicative situation. Nevertheless, it is usually meant to include only its linguistic factors, (i.e. the relevant part of the discourse only), whereas the extra-linguistic conditions are termed **situation**, or **situation context**. To make clear that both extra-lingustic factors and factors within the language are meant, the term **co-situation** can be used.

- **maxim of manner**
  Make your contribution clear, brief and orderly. Avoid opacity, ambiguity, and prolixity.

The maxims are assumed to be universally valid and when the speaker doesn't seem to be cooperative, the cooperation is assumed to take place at a deeper level.

The same principles apply for the use of referring expressions. Each entity can be referred to using a variety of expressions. But usually only the weakest referring expression allowing clear identification of the antecedent is appropriate. Use of stronger referential means than necessary sounds unnatural and makes the hearer speculate about a deeper explanation for it. This can be illustrated by the following example:

(16) Včera jsem potkal **Petra**$_i$. $\emptyset_i$ Byl nemocný.
Yesterday, I met Petr. He was ill.

(17) ? Včera jsem potkal **Petra**$_i$. *On*$_i$ byl nemocný.
Yesterday, I met Petr. He was ill.

(18) * Včera jsem potkal **Petra**$_i$. *Petr*$_i$ byl nemocný.
Yesterday, I met Petr. Petr was ill.

For a deeper insight into pragmatics, please refer to (Levinson, 2000).

1.2.4 Text Linguistics

Discourse cohesion is a very complex concept and space doesn't permit to elaborate on it at length here. But let's address at least one of its aspects and assert that for a discourse to be cohesive, it is necessary that each constituting utterance has a suitable **Topic-Focus Articulation** (TFA), also often termed **information structure**, or **functional sentence perspective**.[17]

The theory of TFA divides each utterance into two basic parts: **the topic** representing what the utterance is about (anchoring it to the previous discourse) and **the focus** expressing what is being said about the topic (advancing the discourse towards the communicative goal of the speaker).[18] Example (19) demonstrates these notions on Czech and Finnish.

[17] The first systematic account of TFA can be attributed to Vilém Mathesius, one of the founding members of the Prague Linguistic Circle – refer for instance to (Mathesius, 1966). A more recent and comprehensive study can be found in (Sgall, Buráňová, and Hajičová, 1980).

[18] This opposition is also often referred to as **theme – rheme/comment**, which in the terminology of Sgall, Buráňová, and Hajičová (1980) expresses a similiar distinction, based on communicative dynamism.

(19) a. Na stole je jídlo.
Pöydällä on ruokaa.
On the table is food.
"On the table, there is some food."

b. Jídlo je na stole.
Ruoka on pöydällä.
Food is on a/the table.
"The food is on a/the table."

(19a) speaks about a table which has presumably been recently mentioned and now the speaker adds new information about it, i.e. that there is some food on it. On the contrary, (19b) speaks about some specific food, uniquely identifiable to the hearer, and specifies its position.

Further, the example clearly reveals that there is a close relationship between the TFA and word order. In most languages there is a tendency for the topic to form the first part of the sentence, being followed by the focus.[19] This is the case also in Slavic and Finno-Ugric languages, which exhibit the so-called **"free" word order**[20]. This makes it possible to express that an item belongs to the topic (or focus) merely by moving the corresponding phrase to the desired position in the sentence. Czech displays this clearly in (19). In Finnish, the subject is additionally assigned case based on its definiteness (Lindroos and Čermák, 1982). English is a language with fixed word order and expresses this notion through articles and certain constructions, mainly **topicalization** and **clefts** (see example (19a) and (14), respectively).

The TFA of a sentence can be investigated through the so-called **question test**. It consists in considering the questions the current sentence is a plausible answer for. Generally speaking, the information given in the question corresponds to the topic and the things asked for to the focus.

(20) Petr jde do školy.
Petr goes to school.
"Petr is going to school."

(21) a. Kam jde Petr?
Where to goes Petr?
"Where is Petr going?"

b. Co dělá Petr?
What does Petr?
"What is Petr doing?"

[19] In this discussion, all sentences are assumed to have neutral intonation. In speech, the focus can be marked by uttering it with accent, regardless of the word order.

[20] This term is very unfortunate, as explained below.

c. Kdo jde do školy?
Who goes to school?
"Who is going to school?"

Clearly, (20) can be an answer for (21a) or (21b), but not for (21c), which would require a specific intonation of (20) to fit. (Sgall, Buráňová, and Hajičová, 1980) contains more details about the question test and also about the test through negation. Here I allude to the relation between the TFA and the scope of negation just by a self-explanatory example[21]:

(22) a. Rohlíčky prý jsou dneska zvláště vypečené. Je tomu tak?
Croissants are said to be today especially crispy. Is it so?
"Croissants are said to be especially crispy today. Is it so?"

b. Není tomu tak, Milosti. Vypečené rohlíčky zvlástě dnes nejsou.
It is not so, Grace. Crispy croissants especially today are not.
"No, it is not, Your Grace. Especially today, there are no crispy croissants."

This example leads us to one of the basic claims of the theory concerning TFA – that two variants of a sentence with different TFA have a different meaning and thus need to be regarded as two *distinct* sentences. Therefore it is rather misleading to use the term "free word order", which suggests that the choice of word order doesn't have any consequences. Flagrantly, this is not true.

(23) a. Mnoho lidí čte málo knih.
Many people read few books.
"Many people read few books."

b. Málo knih čte mnoho lidí.
Few books read many people.
"Few books are read by many people."

It is apparent that sentence (23a) refers to a certain group of people and says that they read few books (but each of them possibly different ones). However, (23b) postulates that there is a small number of books which are read by many people. This is a very special case of (23a) and thus it is very easy to think of a situation for which (23a) is true, but (23b) not.

Apart from the described semantic consequences, the concept of information structure is important for AR methods based on the concept of salience, such as centering, focusing, or the algorithms formulated by the Prague group. Description of some of them follows in the next chapter.

[21] The example is a slightly abridged dialogue excerpt taken from (Werich and Brdečka, 1951).

1.2.5 Psycholinguistics

It is obvious that the best methodology for processing language (therefore also for resolving anaphors) is to adopt the representations and strategies of humans themselves. Unfortunately, no thorough account of the underlying processes in the human brain is known and it is unlikely that this will change in the near future. These processes don't have to be studied directly, though. It is very advantageous to investigate them from the outside – to find out the results the brain produces for given inputs, which inputs it seems to deal with easily and which are apparently problematic. The resulting knowledge on how humans handle AR can be confronted with our hypotheses about the interpretation process or with the properties of whole computational models. The confrontation can shed light on the weak points of the construct in question, or yield certain evidence for its plausibility.

It is not straightforward to obtain relevant and accurate data characterizing the human performance on a given language interpretation problem. This usually involves having a large number of test persons carry out certain experiments. In this matter, it is useful to take advantage of the methodology common in cognitive psychology, which has rich experience with experiments of this kind. It also points out numerous pitfalls in designing such experiments and interpreting their results. Garnham (2001) mentions the following experiment types relevant to AR:

- **self-paced reading**: The test person is stepwise presented a text. To get the next part of the text, the person has to press a button (the previous parts of the text are usually deleted). The time between the keystrokes is measured and is assumed to correlate with the complexity of processing the respective text part.

- **eye-movement experiments**: The test person is supposed to read a given text while wearing a special helmet making it possible to track what they are looking at. This experiment type yields information about the time they needed to read each part of the text. Further it documents for instance cases when the test person had to return to the preceding context.

- **priming** – The test person is presented a text and at certain times, the presentation is interrupted and the test person is required to respond to a given stimulus. For instance, a question about the preceding text has to be answered. In the context of AR, the test person is often asked to name the referent of a pronoun in the text.

To present a simple example, let's imagine a linguist who wants to find out whether each anaphor is assigned its final interpretation at the time it is read. In order to find out, he carries out self-paced reading and priming experiments and collects the data.

Based on the results of the experiments (i.e. response times and antecedent choices), he draws the conclusion that humans initially adopt an antecedent candidate based on syntactic cues such as recency or parallelism, and if this choice doesn't comply with semantic information further in the sentence, the anaphor is re-interpreted.[22]

However, this conclusion *may potentially be* wrong – for instance because the figures supporting the conclusion could be strongly biased by other phenomena, which were not considered at all. This unwanted interference can be minimized by performing multiple experiments of various types. Although psycholinguistic experiments never provide *proofs* for hypotheses, they often yield convincing emprirical evidence. Many widely accepted linguistic hypotheses are based on psycholinguistic research.

For further details about psycholinguistic research concerning anaphora resolution, refer to (Garnham, 2001). Broader information about psycholinguistics as such can be found for instance in (Steinberg, 1993).

1.3 Importance of Anaphora Resolution in NLP Applications

Nowadays, anaphora resolution is addressed in many NLP applications. Proper treatment of anaphoric relations shapes the performance of applications such as machine translation, information extraction, text summarization, or dialogue systems.

Many early machine translation systems operated on a sentence-by-sentence basis. This didn't consider the ties between sentences and resulted in an incoherent text on output. When the treatment of anaphora is neglected, however, the text yielded by the system may be not only unnatural and incoherent, but possibly also factually incorrect.

The most striking problem lies in the fact that pronouns of many languages are required to match their antecedents in number and gender, which are unfortunately language-specific.[23] The antecedent of an anaphor in the source language can be translated by a phrase with a different gender. Therefore it is not appropriate to base the translation of an anaphor on its form in the source language, but rather on the translation of its antecedent. Assignment of inappropriate morphological features to the anaphor often leads to an undesirable change in the meaning of the sentence.

This problem arises for example when translating from German into Czech, as illustrated by example (24) and (25). (24) presents mismatch in gender, (25) mismatch in number.

(24)	a.	**Das Buch**	gefällt	Peter	sehr gut.	Er will	*es*	kaufen.
		The book	appeals to	Peter	very well.	He wants	it	to buy.
		(NEUT.SG.)					(NEUT.SG.)	

[22] Such phenomenona are often referred to as **garden-path effects**.

[23] Let's consider for instance that there are four grammatic genders in Czech, three in German, two in French, and none in Finnish.

b. Petrovi se **ta kniha** velmi líbí. Chce si *ji* koupit.
Petr the book very likes. He wants her to buy.
(FEM.SG.) (FEM.SG.)
"Petr likes the book very much. He wants to buy it."

(25) a. Ich suche **meine Uhr**. Ich kann *sie* nirgendwo finden.
I look for my watch. I can her nowhere find.
(FEM.SG.) (FEM.SG.)

b. Hledám **svoje hodinky**. Nemohu *je* nikde najít.
I look for my watch. I can not them nowhere find.
(FEM.PL.) (FEM.PL.)
"I am looking for my watch. I can't find it anywhere."

Systems for information extraction and text summarization contain mechanisms for obtaining information from relevant parts of the text. However, it is often the case that certain portion of the desired information is realized by pronouns, the antecedents of which are in otherwise irrelevant parts of the text. The pronouns need to be expanded with coreferent autosematic phrases, so that the acquired information is complete.

The importance of determining coreference in this field led the Message Understanding Conference (MUC-6 and MUC-7) to specify and pursue the so-called "coreference task". Please refer to the the respective proceedings for more information.

Information about individual systems taking advantage of anaphora resolution can be found for instance in (Mitkov, 1999).

Chapter 2

Anaphora Resolution Methods

This chapter provides a brief overview of selected notable anaphora resolution methods published so far. With regard to the scope of the work, this chapter is not meant to be exhaustive. Mainly the approaches relevant to the presented system will be described in more detail. Other techniques and research directions are dedicated only a rather superficial mention supplemented with references to relevant literature.

The first section of this chapter sketches certain common perspectives on the AR task and presents an overview of the important directions in AR research. The following sections (2.2 through 2.5) describe selected AR methods and their characteristics. Description of the system's architecture and the actual implementation of the algorithms follows in the next chapter.

2.1 Brief Overview of Outstanding Research Directions

There have been many various AR methods published over the last few decades. They all share certain priciples, and usually aim to account for the same phenomena and tendencies, however, they take different ways in modeling them. At any rate, even different ways usually have several meeting points. Therefore it is very difficult, if not impossible, to draw distinct lines between them.[1] I will try to group the methods roughly according to the type of knowledge they rely on, accepting that no method belongs distinctly to just one class, but rather every method belongs to each class to a certain extent. More detailed and fine-grained classifications supplemented with reviews on the characteristics and development of the individual methods can be found for example in (Hirst, 1981), (Mitkov, 1999) or (Mitkov, 2002).[2]

First, it is suitable to mention the early methods. These are methods based on quite simple ideas, a small set of rules, usually lacking any more sophisticated resources or linguistic background. Nevertheless, they are still able to exhibit considerable performance on common inputs. Let's term them **heuristic methods** and I would group them together

[1] Sometimes it is even difficult to decide, whether two approaches differ at all. Differences in terminology and application context may easily veil the fact that two models are in reality the same (or almost the same).

[2] (Mitkov, 2003), (Mitkov, 2002), and (Mitkov, Lappin, and Boguraev, 2001) provide an useful overview of the history of AR research.

with modern approaches based on the theory of probability or machine learning (see for instance (Aone and Bennett, 1995) or (Connolly, Burger, and Day, 1994) etc.). All these methods are somehow data-driven – they try to exploit regularities and patterns in data, not necessarily in a linguistic way.

Most older systems share the basic methodology principles. They define certain *positive constraints*, that is, specifications of what the possible antecedents of an anaphor are. After these are collected, *negative constraints* come to play and filter out all those antecedent candidates that don't match the anaphor for one reason or the other. If more than one candidate remains, they are ordered according to certain *preference rules* and the "most preferred" candidate is proposed as the antecedent.

This strategy is rather simple, nevertheless, quite plausible. Many modern sophisticated methods can be imagined as following this very scheme, only in a different, more complicated way.

One of the older systems exemplifying the heuristic approach to pronoun resolution is SHRDLU, the famous block-world system described in (Winograd, 1972). Another example is Hobbs' syntactic search, more on which can be found in the next section.[3]

Another, very broad family of AR methods can bear the label **semantic methods**. It contains methods working with semantic representations of utterances in some kind of logical calculus, usually predicate or intensional logic. Such representations are very advantageous for instance for handling quantifier-related phenomena. A complex theory of handling reference resolution is discussed for instance in (Webber, 1983). Some semantic methods take advantage of knowledge bases representing the basic principles and laws of this world and use them to reason about entities in the discourse. This reasoning may yield an explanation (a proof) for the plausibility of certain anaphor-antecedent pairs. Such pairs are favoured over pairs for which no such proof could be found. A system with these features was designed for example by Hobbs (1978).

Further concept useful in the AR process, and widely referred to in more theoretical approaches, is **discourse cohesion** or **rhetorical structure**. The methodology lies in identification of coherence relations between utterances in the discourse – e.g. recognizing that the first clause is elaborated by the second clause, the cause of which is given in the coordination of the third and fourth clause. The resulting discourse structure (usually having a tree-like form) systematizes the relationships to the broader context of each anaphor and thus reveals new ways of judging the relevance of the individual antecedent candidates.

There are several rather complex theories for handling discourse coherence and structure. One of them originates in the area of systemic linguistics and its still highly inspirative description can be found in (Halliday and Hasan, 1976). Further example is the Rhetorical

[3] In both of the mentioned systems, the AR methods are based on rules mostly motivated by syntax. Therefore it is as well legitimate to call them **syntactic methods**, together with other similar ones, especially the methods based on the traditional Chomskyan principles and binding theory. More information can be found for instance in (Lasnik, 1989).

Structure Theory (RST) published in (Mann and Thompson, 1988), or the intentionality-based theory formulated by Grosz and Sidner (1986). A nice review and comparison of the latter two can be found in (Moser and Moore, 1996).

The relation of rhetorical structure to interpretation of anaphoric expressions is discussed for example in (Asher, 1993), (Asher and Lascarides, 2003), or (Hobbs, 1979). It is also addressed by the Veins Theory formulated by Cristea, Ide, and Romary (1998). An AR system implementing a model based on Veins Theory is described in (Cristea et al., 2002).

The last group left to mention in this very rough classification are **methods based on the concept of prominence** of discourse objects. The central idea of these methods is the observation that, at each moment, the hearer perceives certain discourse objects to be brought closest to his attention by the flow of the discourse.

This notion can be compared to a *spotlight* the speaker directs at things she needs the hearer to see in order to deliver him the intended message. She takes advantage of the fact that she can assume the hearer to see the spotlighted object, moves the spotlight to an object somehow related to it and by doing this, pushes the discourse further towards her communicative goal.

There are several theoretical frameworks modeling this view, and this plurality has unfortunately caused some terminological confusion. For the exact terminology and descriptions of the individual approaches, please refer to sections 2.2 through 2.5. Unfortunately, space doesn't permit to give an account of *Focusing*, a comprehensive description of which can be found for instance in (Sidner, 1979) or (Sidner, 1983).

As already mentioned, the presented classification is very hard-grained and the individual groups are not clear-cut. Most systems combine several of the outlined aspects – e.g. hardly any system can do completely without syntactic and semantic mechanisms.

2.2 Hobbs' Syntactic Search

This section describes one of the earlier AR algorithms. It was presented in (Hobbs, 1978) as "the syntactic approach" or "the naive algorithm". Despite its relative simplicity, on common texts, it exhibits performance which is still comparable to most state-of-the-art techniques.

The heart of the algorithm relies on a small number of straightforward rules motivated by preceding research in the field of generative transformational grammar. These rules specify what should be done when a pronominal anaphor is found – how to "look for the antecedent". They are purely procedural and, unlike all other algorithms described in this chapter, Hobbs' syntactic search doesn't build any successive discourse model[4].

The algorithm is meant to be a part of a more complex left-to-right interpretation process and it is assumed that it can already use an unambiguous surface parse tree with

[4] The model is so to say the data itself, together with the algorithm's rules, which specify how to use it.

all syntactically recoverable ommisions expanded within. Hobbs further assumes that this tree is phrasal and contains $\bar{X}$ nodes as postulated by the X-bar theory[5]. The presence of $\bar{X}$ nodes is necessary to distinguish the following cases:

(26) **Mr. Smith**$_i$ saw a **driver**$_j$ in *his*$_{i/j}$ truck.

(27) **Mr. Smith**$_i$ saw a driver$_j$ of *his*$_i$ truck.

The fact that the lexical nodes *"driver"* and *"his"* are not under the same $\bar{N}$ node in (26), makes it possible for them to co-refer. In contrast, in (27), where the whole PP *"of his truck"* is under the $\bar{N}$ node of *"driver"*, such coreference is syntactically not possible.

This constraint is, among others, in-built in the following rules[6] for searching the antecedent of pronouns. They describe how to traverse the surface parse tree and, on the way, recognize antecedent candidates, determine which of them are to be rejected and which of them proposed as the correct antecedent. Implicitly, when an NP to be returned by the algorithm doesn't match the gender and number of the anaphor, it is rejected and the algorithm proceeds with the next step.

1. Begin at the NP node immediately dominating the pronoun.

2. Go up the tree to the first NP or S node encountered. Call this node X, and the path passed to reach it p.

3. *Search*[7] in the subtree of X to the left of p. Propose as the antecedent any NP node that is encountered which has an NP or S node between it and X.

4. If node X is the highest S node in the sentence, *search* the surface parse trees of the previous sentences, starting with the most recent one – when an NP is encountered, it is proposed as the antecedent.

5. From node X, go up the tree to the first NP or S node encountered. Call this new node X, and the path traversed p.

6. If X is an NP and if p did not pass through the $\bar{N}$ node that X immediately dominates, propose X as the antecedent.

7. *Search* in the subtree of X to the left of p. Propose any NP node encountered as the antecedent.

8. If X is an S node, *search* in the subtree of X to the right of p, but do not go below any NP or S nodes. Propose any NP node encountered as the antecedent.

[5] See for example (Chomsky, 1970) or (Jackendoff, 1977)

[6] The given formulation is adopted from (Hobbs, 1978), and for the sake of brevity, occasionally abridged.

[7] All searches stated in the rules are performed in the left-to-right, breadth-first fashion.

9. Go to step 4.

To briefly summarize this detailed wording, steps 2–3 deal with the part of the tree, the antecedent candidates in which can usually be referred to only by a reflexive pronoun. Steps 5–9 gradually climb up the tree, stop at each NP or S node, and search for the antecedent from there. Step 4 extends the search to the previous sentences and applies only when no antecedent is found in the current one.

The algorithm considers only NP antecedents. The author himself argues that allowing S nodes as antecedents would cause serious problems.

On the other hand, the algorithm can be improved by introducing *semantic selectional constraints*. These constraints would specify which anaphor and antecedent pair types (in addition to the morphologically non-agreeing ones) do not match. The complexity and effectivity of such constraints depends on the resources available. One straightforward possibility may be for instance checking whether the semantic features of the antecedent are compatible with the semantic constraints put on the valency slot of the anaphor. This could prevent errors, for example, if words like *"house"* or *"love"* were proposed as antecedents for *"it'* in "He forgot to pack it with him twice".

Generally, the more such semantic constraints we employ, the less semantics-based errors we make. Nevertheless, the utility of such constraints is limited. For example, they are almost of no use for pronouns like *"he"*.

The subsequent chapter of (Hobbs, 1978) describes a system performing complex semantic analysis with details on handling pronoun resolution. As this extension is not relevant for this work, further information can be found in the original article mentioned above.

2.3 Centering

This chapter describes centering theory, a complexer theory modeling various phenomena related to the prominence of discourse objects. It pursues which objects are brought to the closest attention of the hearer by the flow of the discourse, how this happens, and what are the implications. Especially the relation to the choice of the proper types of referring expressions, and to the local coherence of the discourse, are of imminent interest.[8]

The initial proposals of the theory were published in (Joshi and Kuhn, 1979) and (Joshi and Weinstein, 1981), but their ideas were largely inspired by Sidner's work on immediate focusing (Sidner, 1979). They introduced the centering terminology to prevent confusion with theories (for example the Sidner's) using similar terms for similar, but still slightly different concepts.

[8] Centering theory is formulated in accordance with the above-mentioned theory of discourse structure formulated by Grosz and Sidner (1986). In the terms of this theory, it can be said that centering models the local-level component of attentional state (Grosz, Joshi, and Weinstein, 1995).

Centering proposes a number of constraints, and claims that if the discourse in question meets them, the hearer perceives it as coherent and is able to interpret it straightforwardly, without having to spend any unnecessary mental effort. This can be illustrated by the following example (Grosz, Joshi, and Weinstein, 1995):

(28) a. John went to his favorite music store to buy a piano.
b. He had frequented the store for many years.
c. He was excited that he could finally buy a piano.
d. He arrived just as the store was closing for the day.

(29) a. John went to his favorite music store to buy a piano.
b. It was a store John had frequented for many years.
c. He was excited that he could finally buy a piano.
d. It was closing just as John arrived.

Intuitively, discourse (28) is more coherent than (29). It is continuously centered around a single individual (John), whereas each utterance in (29) is about something else than the previous one (John–store–John–store). This sounds much more clumsy and ponderous, and the hearer is not able to intepret such sequence of utterances as smoothly as sequence (28). It is apparent that the two examples convey the same information, only the information is packaged in a different way. (29) is somehow more difficult to unwrap, in other words, *puts higher inference load on the hearer*. As explained below, centering accounts for this phenomenon.

Another argument for the plausibility of centering theory is that it explains certain garden path effects in the interpretation of pronouns. The following example taken from (Grosz, Joshi, and Weinstein, 1995) shows that choosing an unfortunate referring expression type may mislead the hearer considerably. It is assumed that he resolves pronouns immediately, before processing the rest of the utterance, that is, when there's still no semantic information available. The semantic information made available at a later point can rule out this choice and force the hearer to backtrack.

(30) a. Terry really goofs sometimes.
b. Yesterday was a beautiful day and he was excited about trying out his new sailboat.
c. He wanted Tony to join him on a sailing expedition.
d. He called him at 6AM.
e. He was sick and furious at being woken up so early.

The use of a pronoun to refer to Tony in (30e) is clearly confusing. In the whole of (30a–30d), the center of attention is Terry. Therefore, when interpreting the pronoun in

(30e), he is the obvious first choice and is more likely to be understood as the referent. The erroneousness of this decision can be revealed only after realizing that Terry being sick and woken up doesn't fit the context. Referring to Tony with a full noun results in a much more natural and coherent text.

Let's proceed with the exposition of the definitions and claims of centering theory. Let's assume that a discourse consists of a number of discourse segments, each of which has the form of a sequence of utterances $U_1 \ldots U_m$. Then, for each utterance U_i holds:

- It has a set of **forward-looking centers**, $C_f(U_i)$.
  They represent entitities **realized**[9] in the given utterance.
- It has a *single* **backward-looking center**[10], $C_b(U_i)$, and it holds $C_b(U_i) \in C_f(U_i)$.
  $C_b(U_i)$ represents the center of attention after U_i has been processed. That $C_b(U_i)$ is a single entity is one of the fundamental claims of centering theory.
- $C_b(U_i)$ connects with one member of $C_f(U_{i-1})$.
- The forward-looking centers are partially ordered[11] according to their prominence. The higher the rank of a particular center, the more likely it is to become $C_b(U_{i+1})$.
- The most highly ranked element of $C_f(U_i)$ is the so-called **preferred center** $C_p(U_i)$.

Using the just introduced definitions, we can formulate one of the claims of centering theory, also known as "Rule 1".

> **RULE1**: If any element of $C_f(U_n)$ is realized by a pronoun in U_{n+1}, then the $C_b(U_{n+1})$ must also be realized by a pronoun.[12]

The rule defines a clear relationship between the choice of a referring expression type and local (in)coherence of discourse. It reflects the tendency to realize C_b by a pronoun in English, or by corresponding referential means in other languages (e.g. zero-pronouns in Czech).

To gain terminology useful to word further characteristics of local coherence, the following **transition relations** between two neighboring utterances (U_n and U_{n+1}) have been defined:

[9] The exact definition of *"entity x is realized in U"* depends on the particular semantic theory adopted. The definition is supposed to combine syntactic, semantic, discourse and intentional factors. With a slight simplification, it may correspond to "U contains an expression referring to x". A proper definition, not overlooking implicitly mentioned entities, is beyond the scope of this work.

[10] The only exception is the first utterance in the segment, which has no backward-looking center.

[11] The most widely used ordering for English is grammatical obliqueness.

[12] The constrapositive form of this implication offers another interesting view: if $C_b(U_{n+1})$ is not realized by a pronoun, no other $C_f(U_{n+1})$ element can be.

- **Center continuation**: $C_b(U_{n+1}) = C_b(U_n)$ and $C_b(U_{n+1}) = C_p(U_{n+1})$.
  That is, $C_b(U_n)$ remains to be the backward-looking center also in U_{n+1} and is likely to keep this role in U_{n+2}.
- **Center retaining**: $C_b(U_{n+1}) = C_b(U_n)$ but $C_b(U_{n+1}) \neq C_p(U_{n+1})$.
  In other words, $C_b(U_n)$ is the backward-looking center in both U_n and U_{n+1}, but in U_{n+1} it loses its prominent position and it is unlikely to become $C_b(U_{n+2})$.
- **Center shifting**: $C_b(U_{n+1}) \neq C_b(U_n)$.
  Brennan, Friedman, and Pollard (1987) introduce a further distinction:
  - **Smooth shift** (shifting-1): $C_b(U_{n+1}) = C_p(U_{n+1})$
  - **Rough shift** (shifting): $C_b(U_{n+1}) \neq C_p(U_{n+1})$

Now it is possible to pronounce the so-called "Rule 2", which suggests which transitions produce higher inference load.

> RULE2: Sequences of continuation are preferred over sequences of retaining; and sequences of retaining are preferred over sequences of shifting (smooth over rough shift).

This is a significant constraint on individuals and systems generating texts. They should plan and construct their discourse in a way which minimizes the undesirable transitions. Consequently, texts like (28) come into existence, instead of texts like (29).

The most notable anaphora resolution procedure utilizing centering theory was presented by Brennan, Friedman, and Pollard (1987) and therefore it also known as **the BFP-algorithm**.

The algorithm computes **the anchors** of the individual utterances, i.e. pairs of the form $< C_b, C_f >$, and proceeds in three steps, *constructing the anchors*, *filtering out implausible anchors*, and *ranking the anchors*.[13]

1. CONSTRUCTION OF THE ANCHORS FOR U_n
   - Create a list of referring expressions and sort them by grammatical relation.
   - Create a list of possible forward-looking centers, expanding each referentially ambiguous element (pronouns, NPs) into a set containing all their possible referents.
   - Create a list of all possible backward-looking centers – elements of $C_f(U_{n-1})$ and NIL for the possibility there is none to be found.

[13] Note that this resembles the strategy sketched in section 2.1.

- Create a set of anchors by combining the sets from the previous two steps. This yields a set containing $< C_b(U_n), C_f(U_n) >$ elements.

2. FILTERING THE PROPOSED ANCHORS

- Filter out anchors ruled out by binding constraints.
- Eliminate all anchors where the proposed $C_b(U_n)$ is not the highest ranked element of $C_f(U_{n-1})$ realized in $C_f(U_n)$.
- Eliminate all anchors violating the Rule 1, i.e. exclude all anchors where there are some elements of C_f realized by pronouns, but C_b isn't equal to any of them.

3. RANKING THE REMAINING ANCHORS

- Classify each anchor according to the transition it induces.
- Rank the anchors using the Rule 2, and adopt the most highly ranked one.

A more exhaustive description of centering, its motivations and features is presented in (Grosz, Joshi, and Weinstein, 1995). A number of commented examples and a confrontation of centering with the Praguian salience-based approach can be found in (Kruijff-Korbayová and Hajičová, 1997). Further extension of the theory was proposed e.g. by Strube (1998).

2.4 Activation-based Methods

This section describes methods proposed by the linguists of the Prague group. The methods to be described are based on the concept of activation (or salience) and are usually formulated in the terms of the local framework, **the Functional Generative Description** (FGD) of language.

The main ideas are very close to the ideas of centering, however, the formal grounds differ considerably. Communication is seen as a sort of game between the hearer and the speaker. Each of them has a certain image of the world, an important component of each being the so-called **Stock of Shared Knowledge** (SSK), a representation of objects that are spoken about in the discourse or occur in the communicative situation, as well as of their properties and mutual relations (Hajičová, Hoskovec, and Sgall, 1995).

The SSK is known to have hiearchical structure, reflecting that some items are more *activated* than others, that is, are closer to the attention of the hearer, and thus accessible by weaker referential means. Throughout the discourse, the speaker tries to take advantage of the current state of the SSK, especially its most activated items, and successively change it in a way corresponding to her communicative goal.

Using the terminology introduced in section 1.X, the speaker builds the *topic* of an utterance[14] of entities highly ranked in the SSK, and relates them to *focus*, consisting of previously unmentioned entities (or relating already mentioned entities to the topic in a new way). From a different perspective, the topic locates an item in the SSK and the focus describes how it should be changed in the hearer's image of the world (i.e. also in SSK).

These notions make it apparent that the TFA of the individual utterances should play a considerable role in discourse modeling, and consequently, also in the methodology of anaphora resolution. Next, I will mention two AR algorithms building on the just presented ideas. The first one is taken from (Hajičová, 1987), and the other one from (Hajičová, Hoskovec, and Sgall, 1995).

2.4.1 Algorithm 1

The heart of the algorithm presented in (Hajičová, 1987) is modeling of the SSK. The model is slightly simplified – it considers only items introduced in the discourse by nominal expressions.

At each point of the discourse, the model tracks the activation of each item and does it incrementally, utterance-by-utterance, always computing the values on the basis of the previous ones. The individual activation values are represented by non-negative integer numbers. The lower the number, the higher the activation, the highest possible activation thus being 0.

In order to reveal how activations of objects usually change in discourse and what the circumstances that bring about these changes are, the author of the method performed empirical research. The most important tendencies and regularities discovered are:

- The items referred to in the focus of the immediately preceding utterance are the most activated ones at the particular point of discourse.
- In the topic of an utterance, reference with a full NP strenghtens the activation of the referred item more than pronominal reference.
- The activation of items referred to in topic is more steady (fades away less quickly) than the activation of items in focus.
- If the activation of an item changes, then also the activations of all associated items should change (considering the "closeness" of the relation).
- Certain expressions can function as so-called *thematizers*, e.g. "concering ..." in English or, "ohledně ..." and "pokud se jedná o ..." in Czech. The item they introduce gains a higher activation than it would get when mentioned on its own.

[14] "Utterance" is a unit of parole and it is meant to correspond to a simple sentence, a complex sentence, or a clause of a compound sentence.

Condition: *Effect:*	x has an activation of a and is referred to by a pronoun the activation of x remains the same (that is, a)
Condition: *Effect:*	x has an activation of a and is referred to by an NP in the focus the activation of x is 0
Condition: *Effect:*	x has an activation of a and is referred to by an NP in the topic the activation of x is 1
Condition: *Effect:*	if an object gets an activation of m through reference all objects associated to it get an activation of $m + 2$
Condition: *Effect:*	if x is mentioned in a thematizer expression as the left-most expression of the clause the activation of x is changed to 1
Condition: *Effect:*	if none of the former rules apply (for all x which is neither referred to or associated to objects referred to in the utterance) the activation of x is increased by 2

Table 2.1: Rules for incremental computation of activation values

- If an SSK item is neither referred to, nor associated to an item mentioned in the current utterance, its activation drops.

The algorithm employs these principles in a number of rules formulated through preconditions and effects, see table 2.1. The discourse is processed utterance after utterance and the rules are applied to each referrential expression in the current utterance. Clearly enough, for each such expression, at least one of the rules applies. If an expression meets more of the specified preconditions, the rule which assigns the item the highest activation is used.

To enable the application of the first rule, it remains to be specified, how to resolve pronouns – that is, how to match a pronoun in the current utterance to the corresponding item in the SSK.[15] The pronoun is identified with the highest activated item in the SSK meeting all necessary constraints, above all, matching the pronoun considered in gender, number and person. This model can be extended by considering further, for instance semantic, constraints.

2.4.2 Algorithm 2

A further algorithm was proposed by Hajičová, Hoskovec, and Sgall (1995). It is derived from the same ideas as the previous algorithm and is also similar in the way it models

[15] For the sake of brevity, I will not go into detail on how to match definite descriptions in the second and third rule.

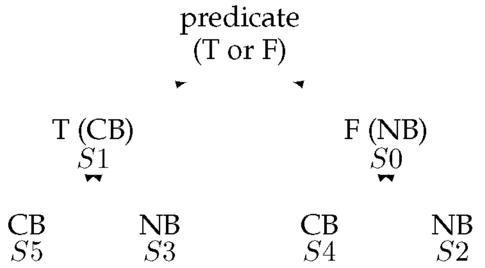


Figure 2.1: Partition of dependency tree nodes

them. At any rate, it is slightly more specific, and enables more delicate parametrization.

The algorithm adopts the numerical representation of activation from the algorithm described above, nevertheless, suggests a more elaborate partition of the utterance based on the TFA. The partition is defined by the position of nodes in the dependency tree of the utterance and is sketched in figure 2.1.

The groups $S1$ and $S0$ consist of daughters of the predicate node (i.e. nodes of depth 1). $S1$ contains the ones in the topic, and $S0$ the ones in focus. All other nodes are either direct, or indirect descendants of nodes in $S0$ and $S1$. The groups with odd numbers contain nodes in the $S1$'s subtree. $S5$ the contextually bound ones, $S3$ the contextually non-bound ones. The groups with even numbers, containing nodes in the $S0$'s subtree are defined analogically. Further, let's define the following sets:

$$P(0) = S0$$

$$P(1) = S1 \cup S4 \cup S5$$

$$P(2) = S2$$

$$P(3) = S3$$

Nodes in the same $P(x)$ contain nodes equally activated in SSK and thus equally accessible by extrasentential anaphors. Let's assume that the discourse is a sequence of utterances $(V_1, \dots, V_n)$ and O is the set of all discourse objects. This makes it possible to formalize the assignment of activation to SSK elements by introducing function $B : \{1, \dots, n\} \times O \longrightarrow \mathbb{N}_0 \cup \{\perp\}$, assigning a number (or undefined value $\perp$) to each utterance index i and object t. This number encodes:

- index of the utterance t has been mentioned in for the last time before V_i
- the position of t in this utterance (i.e. number x such that $t \in P(x)$ in the respective utterance)
- the degree of t's salience after uttering V_i

Analogically to the previous algorithm, for each $i \in \{1, \ldots, n\}$ a $t \in O$, the value of $B(i,t)$ is computed recursively (based on values $B(i_0, t)$ for $i_0 < i$, the base of recursion being $\forall t \in O : B(0,t) = \perp$):

- If t is not mentioned in V_i, and $B(i-1,t) = \perp$, then $B(i,t) = \perp$
- If t is not mentioned in V_i, and $B(i-1,t) \neq \perp$, then $B(i,t) = B(i-1,t) + 4$
- If t is mentioned in V_i in position $P(j)$, then $B(i,t) = j$

As explained (and illustrated by several examples) in (Hajičová, Hoskovec, and Sgall, 1995), the following claims can be made about the SSK items, their $B(i,t)$ salience values, and their accessibility by extrasentential pronominal anaphora in V_{i+1}:

- if $B(i,t) \leq 2$, and also if $B(i,t) = 4p$, or $B(i,t) = 4p + 1$ for $p > 0$, it is possible to refer to t using weak anaphoric means (e.g. unstressed pronominal forms like *"mu"*)
- if $B(i,t) = 4p + 2$ for $p > 0$, or $B(i,t) = 4q + 3$ for $q \geq 0$, pronominal reference to t in V_{i+1} would be far-fetched
- if $B(i,t) = 0$, stronger referential means (like stressed personal pronoun forms, or demonstrative pronouns) are preferred when referring to t
- ellipsis can be used to refer only to objects with $B(i,j) \in \{0, 1\}$

The precise figures this model defines are suited for Czech. It is assumed to perform well also for other languages, mainly the related ones. Moreover, it offers many straightforward opportunities for re-parametrization.

2.5 Methods Combining Salience Factors

This section describes the RAP system presented by Lappin and Leass (1994). Like the systems described in the previous section, it is based on the concept of salience. However, the way it deals with salience is different, and worthwhile.

So far, only few algorithms for pronoun resolution have been able to account for all preferences and tendencies of anaphorical references. This system offers a way of formulating and combining several so-called "**salience factors**", and demonstrates how to perform AR considering all of them.

RAP (Resolution of Anaphora Procedure) has the following components:

- **an intrasentential syntactic filter** specifying constraints on NP-pronoun coreference within a sentence
- **a morphological filter** determining non-agreement in gender, number and person

- **a procedure for identification of pleonastic pronouns**, i.e. eliminating non-referential pronoun occurences from the discourse model
- **an anaphor binding algorithm** for resolving reciprocal and reflexive pronouns to antecedents in the same sentence
- **a procedure for computing salience parameters**, i.e. assigning each referring expression a salience value based on the respective salience factors
- **a procedure for keeping track of equivalence classes**
- **a procedure specifying preferences** for selecting the antecedent from a list of candidates

The processing[16] of each utterance in the discourse in question is performed in the following way:

Before starting the actual processing of the utterance, salience values of all items mentioned in the discourse so far (if any) are cut in half to account for the preference for recent antecedents. All items the salience value of which sank below 1, are removed from the model (they are assumed to have faded out completely).

Next, all NPs in the utterance are listed and classified (as definite NPs, pleonastic pronouns, reflexive and reciprocal pronouns, etc.). Based on this, it is determined which expressions introduce a new entity, which of them are non-referential, and which of them are left to be resolved.

Then, salience factors are applied to the individual referring expressions found in the previous step. Table 2.2 lists selected salience factors and the corresponding salience values. Each expression is assigned the sum of all factors it complies with.

Points for **sentence recency** are given to any NP in the current sentence (it favours more recent antecedents, together with the measure described above). **Subject emphasis** gives credit to expressions in subject position, **existential emphasis** to NPs in existential constructions. **Head noun emphasis** increases salience of each NP that is not embedded in another NP,[17] etc. The factors formulated in RAP are based mainly on syntactic concepts. Semantic features and real-world knowledge are not considered.

Next, the syntactic filter is used to rule out incorrect pronoun-NP pairs. Then, the binding algorithm binds reflexive and reciprocal pronouns (when it yields an ambiguous result, the candidate with the highest salience value is chosen).

Each remaining third person pronoun is resolved in the following way:

- A list of all possible antecedent candidates and their salience weights is made. Within this actual resolution process, the weights are adjusted to the particular pronoun

[16] For the sake of brevity, the description provided here is in some points slightly simplified.

[17] As salience values can be only positive, this is the only way how to penalize certain types of NPs – to increase the salience of every other NP.

Factor type	*Initial weight*
Sentence recency	100
Subject emphasis	80
Existential emphasis	70
Accusative emphasis	50
Indirect object and oblique complement emphasis	40
Head noun emphasis	80
Non-adverbial emphasis	50

Table 2.2: Some salience factor types and their initial weights

occurence – antecedents following the pronoun are penalized (this reflects implausibility of cataphora), and antecedents filling the same valency slot as the pronoun are given extra credit (this measure favours parallelism).

- A salience threshold is applied and all candidates with a lower salience value are not considered any further.
- All appropriate agreement features are determined, allowing ambiguity.
- The best candidate is selected (separately a singular and a plural one, if number ambiguity exists in the language in question). First, morphological filtering is performed, then pronoun-antecedent combinations previously found out to be wrong by the syntactic filter are eliminated, and the most salient remaining item is declared to be the antecedent of the pronoun (when there is both a plural and singular candidate, the more salient one is chosen).

This methodology and, above all the precise salience factor weights, have been reached after extensive experimentation and numerous re-adjustments of the individual weights. This is a very powerful strategy and makes it possible to re-fit the system for a particular language and genre. On top of that, a further section of (Lappin and Leass, 1994) suggests an additional improvement of the system. It is achieved by introducing certain lexical preferences obtained through statistics. This eliminates errors in cases where several candidates hardly differ in salience weights, and the lexical content plays the decisive role. However, this doesn't completely make up for the absence of semantic features. This remains to be RAP's biggest weakness.

Chapter 3

System Implementation

This chapter describes the functionality of the presented framework, and shows how it instantiates the algorithms presented in the previous chapter. The first section reveals the framework by explaining the individual data structures and interfaces, their meaning and interplay. Next, section 3.2 contains information about modules for loading data and saving results, section 3.3 focuses on mechanisms parametrizing the algorithm run. Finally, section 3.4 describes the implementation of the selected algorithms.

3.1 Overall System Architecture

The presented framework for modular AR is implemented in Java[1] and this section describes it in two steps. First, I provide a rather technical description of the structures used to represent data, and consequently, I explain how they participate in the resolution process and comment on the modularity of the framework.

The fundamental data structures are defined by the classes of the `anaph.data` package. To start with, `anaph.data.Text` represents the text to be processed. It consists mainly of a list of sentences (`Sentence` objects), each containing a tree structure. One possible tree representation of a sentence is a phrase structure tree, which has a long tradition, especially for configurational languages such as English. A sentence can be also represented by a dependency tree, which is much more suitable for languages with "free" word order, such as Czech.

The mentioned tree representations are supported by the framework through the class `PhrNode` and `DepNode`, respectively. They stand for the nodes of the trees and differ only in their assumptions about subtrees. Each sentence is required to have either a dependency or a phrasal tree representation (or both).

For many purposes, it is useful to process texts in units slightly smaller than sentences, such as clauses. These can be represented by `Clause` objects and can be yielded through a `ClauseSplitter` (see section 3.3.3 for more details).

[1] The framework implementation follows the Java 2 Platform, Standard Edition API Specification in version 1.4.0. For more details about the API provided by the framework, please refer to appendix A or directly to the documentation on the enclosed disc.

Not all nodes of the tree representation are relevant to the AR process. It concerns mainly phrases that are themselves anaphors, or can be their antecedents. A so-called **markable** is created for each such phrase. In the presented framework, a `Markable` is understood as a set of nodes corresponding to the relevant phrase. Markables are usually obtained through appropriate detectors (refer to section 3.3.1 for further details).

In many languages, information about the morphology features of markables is essential in determining the correct antecedent for an anaphor. Such information is meant to be stored in a `Morphology` object and agreement between two such objects can be determined using the `Agreement` interface (see 3.3.2 for further details).

Adopting the presented terminology, the goal of AR is to find anaphoric relationships between markables. These can be represented by means of `Link` objects. There are two predefined kinds of links: directed and undirected (`DirectedLink` and `UndirectedLink` resp.). Undirected links are suitable for linking coreferential markables (coreference is an equivalence, therefore it is also symmetric) and directed links are useful for expressing unsymmetric relations (such as bridging). The implementations of both assume that the orientation from the anaphor to its antecedent is given, however, directed links can be traversed only in one direction, whereas undirected ones in both.

Links of any kind can be grouped into a `LinkSet` object, which defines various methods for traversing the links as if they formed a graph. Above all, this allows partitioning the markables in question into connected components, i.e. coreference classes (represented by `EquivalenceClass` objects). A set of such classess represents the coreference over the given markables (`Equivalence`). A set of links and the corresponding equivalence over markables are a part of the `Text` object.

A further part of an `Text` instance is a `Constraints` object, representing relevant constraints on coreference. It consists of two set of links, positive and negative. Positive links represent all known anaphoric links among the markables and negative ones represent relations which are not allowed to occur (e.g. for syntactic or semantic reasons). Apart from collecting these data, this object allows checking a potential anaphoric link for consistency with all constraints on coreference known so far. Constraints are usually generated by a suitable `ConstraintDetector` (see section 3.3.5 for more information).

Now it is possible to depict the AR process in the framework. It is realized through an implementation of the `anaph.algorithm.Algorithm` interface and plainly said, it enriches a given text with the computed anaphoric links and coreference. This computation can be suited to the current needs using appropriate implementations of relevant interfaces (`anaph.algorithm.options.*`). These are passed to the `Algorithm.performAlg()` method[2] for performing AR and specify:

[2] The alternative of this method, `Algorithm.processClause()`, which performs just one step of the computation (and is iteratively called by the former method to process the whole text) can be parametrized in an analogical way. Please refer to the documentation for further details.

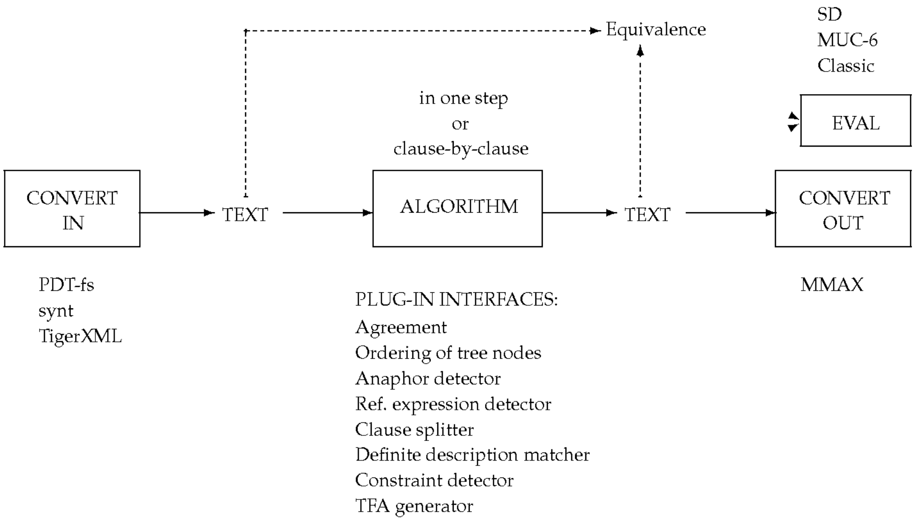


Figure 3.1: Overview of the system's architecture

- which nodes should be considered as markables (see 3.3.1)
- how to determine agreement in morphology (see 3.3.2)
- how to split sentences into smaller processing units (see 3.3.3)
- how to match definite descriptions (see 3.3.4)
- how to determine constraints on coreference (see 3.3.5)
- how to compute TFA values (see 3.3.6)
- the parametrization of the particular algorithm used (`AlgDepOptions` – see the respective subsection of 3.4)

A sketch of the AR process can be found in figure 3.1. Details on loading and saving `Text` representations can be found in section 3.2, mechanisms for evaluation are discussed in chapter 4.

The existence of the above-listed interfaces makes it possible not only to suit the AR process to the current conditions (such as language, formalism, availability of annotation, etc.), but also gives a lot of freedom for exprerimenting. It is advisable that the individual algorithms are made as abstract as possible, so that they could be used with all implementations of the interfaces, which are application-specific by nature. This allows not only running an algorithm for instance with different markable detectors, but also using the same detectors for different algorithms. In order to achieve this, it suffices to implement markable detectors and a clause splitter for each formalism and define morphological agreement for each language or tagset. This makes the system easily extensible.

Regardless of all this, the framework can be used to implement algorithms compatible only with certain interface implementations, and vice versa. In certain cases, this may be of advantage and can save considerable amount of work. However, the free hand to do this gives the programmer enough rope to hang himself. The possibility to freely interchange the individual interface implementations is too tempting to be abandoned without a reason.

In addition to the method for processing a whole text, the `Algorithm` interface also provides a method for processing it step-by-step. This offers the possibility to combine several algorithms into a meta-algorithm, yielding antecedents based on their results. For further details on meta-algorithms, refer to 3.4.6.

I would like to at least briefly mention two already existing frameworks with a similar architecture. The first one was developed at the University of Rochester by Byron and Tetreault (1999). The authors emphasize the advantages of modularity and encapsulation of the system modules into layers. For their system, they define three. The AR layer (roughly corresponding to the `Algorithm` interface), the translation layer for creating data structures, and the supervisor layer for controlling the previous layers (in my system, this would correspond to writing small programs that only "select" the desired algorithm and interface instances and merely call them).

Another system was produced by Cristea et al. (2002) and defines layers from a different perspective. The text layer contains referring expressions and their attributes are projected to the projection layer. The projection layer contains feature structures that can be related to discourse entities on the semantic layer. The architecture is used to implement four models of different kinds. More details can be found in (Cristea et al., 2002).

The following two sections proceed with the description of the framework by mentioning the data and by elaborating on the interfaces parametrizing the AR process. Section 3.4 describes the implementation of selected AR algorithms. Particularly, 3.4.1 demonstrates the AR process on a sample algorithm.

3.2 Input/Output Modules

One of the most important issues in every AR system is probably the data. Methods for loading the data into the internal representation and subsequent saving the results need to be defined.

In the presented framework, this is addressed by the `TextReader` and `TextWriter` interfaces, respectively. The reader is required to implement a `readInData` method for filling a given `Text` object with the content of a specified file. Analogically, the writer is required to implement a `writeData` method for creating files based on the internal representations.

Each implementation deals with a specific data format, however, usually contains an option interface for handling different tagsets, requirements on preprocessing etc.

3.2.1 MMAX

MMAX, developed at the EML in Heidelberg, is a tool for multiModal annotation in XML. It employs the concept of a *markable* to represent linguistic data at various levels of language description. Further, it is possible to define relations between markables and thereby account for diverse phenomena such as valency structure, prosody, or coreference.

To exploit the advantages of the tool for visualization of AR results, I implemented the `MMAXWriter` class for projecting `Text` objects into MMAX data structures (a list of words and sentences). Further, the collection of markables is saved as an annotation level and is enriched with attributes expressing coreference classes. Based on this data, the MMAX tool visualizes the text and provides a sensible insight into the AR results by making it possible to interactively highlight coreferent markables.

For the purposes of this work, it was not necessary to export information about syntactic structure into MMAX. However, it is possible to define a separate annotation level for representing syntactic trees, should this turn out to be rewarding in the future.

Further information about MMAX can be found for instance in (Müller and Strube, 2003).

3.2.2 Prague Dependency TreeBank

The Prague Depependency TreeBank, created by Hajič et al. (2005), contains about 50,000 manually annotated sentences, represented by dependency trees. At present it is the only large Czech corpus annotated for coreference.[3] Thanks to the generosity of Jan Hajič and Zdeněk Žabokrtský, in March 2005, I was provided with the current preliminary version of PDT 2.0. I received the data in the so-called fs-format, for which I implemented an input module – the `PragueDepTreeBankFS` class.

The implementation of the parser separates the interpretation of node attribute values and morphology features from the rest of the parsing process. This makes it possible to straightforwardly re-use the parser for other data stored in the same format. By default, the set of links and the equivalence of the newly created `Text` object is induced by the annotation in the input file.

A disadvantage of this corpus representation is, that it is stored in files not corresponding to documents. Thus, it is necessary to load a whole set of files and re-group the sentences according to their identifiers.

Initially, I intended to implement an output module for PDT, however, producing a proper fs-file requires retaining all attribute values given in the input file. For each node, there are

[3] The details about the coreference annotation can be found in (Kučová et al., 2003). Information about the corpus as such can be found for instance in (Hajičová, Panevová, and Sgall, 1999).

more than hundred attributes (with potentially ambiguous values), and since only less than ten of them are relevant for the AR process, it is very unreasonable to include all of them into the `Text` representation. However, in spite of the inexistence of an output module, it is still possible to save the results of the AR process in the MMAX format. In addition to that, evaluation outputs can be generated.

3.2.3 Synt

Synt is a syntactical analyzer developed at the Faculty of Informatics in Brno. Description of the parsing and analysis methodology and further details about synt can be found for instance in (Horák, 2001).

I implemented an input module for synt, the `Synt_ambig_morph` class, which takes account of the fact that synt produces ambiguous output – a potentially high number of phrase structure trees. To remove the ambiguity, an interface is defined for the parser, which allows defining a method for combining the trees available for each sentence into a single tree representation.

All necessary interfaces are implemented for synt, however, the implementation doesn't go into such depth as implementations for PDT. It is meant to provide solid base for future extension.

3.2.4 TiGerXML

Initially, a preliminary version of this system was used with the TiGer corpus. This corpus was developed within the TiGeR project, a cooperation of the Saarland University and the University of Stuttgart. It contains about 35.000 syntactically annotated sentences taken from German newspaper articles represented by means of phrasal structure trees.

The corpus data is stored in the TigerXML format, for which I implemented an input module, the `TigerXML` class. It separates the parsing process itself from the interpretation of morphology. More information about the TiGer corpus can be found for instance in (Brants et al., 2002).

3.3 Interfaces Customizing the Algorithm Run

This section describes the interfaces which make it possible to suit the AR process to the needs of the current application. Their implementations are passed as parameters to the methods for performing AR. The definitions of the individual interfaces can be found in the `anaph.algorithm.options` package.

In addition to the described interfaces, the AR algorithms are also passed a specification how to order nodes (and indirectly also markables) in the current tree structure. This ordering is usually used as the last criterion for the antecedent choice. Apart from linear ordering, it is possible to sort nodes based on their grammatical role, semantic role, the depth of the node in the tree etc. The implemented comparators can be found in `anaph.algorithm.options.depord` (`phrord` resp.) and will not be further discussed here.

3.3.1 Detectors of Anaphoric and Non-anaphoric Markables

There are two types of markables, anaphoric and non-anaphoric. Accordingly, there are also two detector interfaces, `AnaphDetector` and `RefExprDetector`. They detect different markables, however, they do it in the same way.

The interfaces provide two methods. One determines whether a given node represents a markable or not, and when yes, it returns it. The other is given a whole clause and returns a list of all markables in the clause, sorted as specified by a given comparator. Further, these methods generate so-called exclusions. These can be understood as "near misses". Usually, all markables belong to a certain syntactic category, but not all its nodes are meant to be markables (such as pleonastic pronouns in the case of detecting pronominal anaphors). The excluded markables are collected and can be used to illustrate the relation of the detected markables to the respective broader category, esp. by means of a standard disclosure (see section 4.4 for more details).

First, let's address the instantiations of the `AnaphDetector` interface. For PDT, there are two of them. The first one mirrors the way anaphors are detected in (Kučová and Žabokrtský, 2005). It is implemented by the `AnaphDetector_Dep_pdtZZ_cz` class and returns all nodes representing personal pronouns (according to the tectogrammatical lemma) annotated to refer textually. In the context of markable detection, this is a perfect cheat and it is implemented here mainly for purposes of comparison.

The main anaphor detector for PDT is stored in the `AnaphDetector_Dep_pdt_cz1` class and detects anaphors based on their POS tag. All nodes tagged as third person full or weak personal pronouns are detected, pronouns of first and second person are excluded. The same applies to possessive pronouns. Demonstrative pronouns are also returned, but the variety of exclusions is richer. All demonstratives that modify a noun, or link an embedded clause are exluded (e.g. "tento vlak", "to, že přišel"). Occurrences of demonstratives in a number of specific constructions (e.g. "čím … tím" or "a to") are also excluded. The detection of zero personal pronouns is even more complex and is done by a special method during the pre-processing of the text. The nodes representing zero pronouns in subject position seem not to differ from unvoiced participants of certain nominal and technical constructions. The distinction is made based on the position of the node in the clause and the presence of nodes with individual grammatical roles.

The `AnaphDetector_Phr_synt_cz1` class contains an anaphor detector for synt. It detects anaphors based on the syntactic category (i.e. non-terminal type) and returns all personal and possessive pronouns that are not reflexive, also all demonstrative pronouns that do not modify an NP.

The anaphor detector for TiGer (implemented by AnaphDetector_Phr_TiGer_de1) detects markables based on the POS tag. It returns all attributive possessive pronouns and all other substitute pronouns (except for interrogative and relative) unless they are a part of a coordinated noun phrase.

The RefExprDetector interface is designed to detect non-anaphoric markables, and for PDT, it is implemented by RefExprDetector_Dep_pdt_cz3. It detects referring expression based on their POS tag. All nouns and nominal numerals are considered to be markables. It also detects deadjective nouns (e.g. "raněný", "každý"), however, these are annotated as ordinary adjectives in PDT and are very difficult to distinguish from them. The distinction is done heuristically based on the position in the clause and the dependency type. Certain occurrences of nodes annotated as "foreign phrases" are also regarded as markables. Finally, all conjunctions, disjunctions and appositions consisting of already detected markables are also returned.

The RefExprDetector_Phr_synt_cz1 class detects non-anaphoric markables for synt based on the syntactic category. All NP non-terminals are declared to be markables.

The detector of non-anaphoric markables in TiGer is implemented by the RefExprDetector_Phr_TiGer_de1 class and detects the markables based on the syntactic category. Nonterminals tagged as a noun phrase, coordinated noun phrase, multi-word proper noun, or a prepositional phrase are returned. Terminal nodes represent markables only in certain syntactic positions and these are mainly proper nouns.

3.3.2 Agreement in Morphology

Morphology plays a significant role in AR. The antecedent is required to match the anaphor in certain morphology features.[4] For the studied languages (Czech and German), agreement in gender, number and person is required. Therefore it is necessary for each markable to carry information about its morphology, in the form of a Morphology object. As morphology features of a phrase can be ambiguous, the object consists of a set of variants. Each defines values of relevant attributes. Unfortunately, different corpus formats use different tagsets and therefore it is necessary either to normalize them while interpreting the input data, or to specify a separate way of determining agreement for each of them, by means of an Agreement interface implementation. The interface defines a single method agree(), which is to return a truth value given a pair of morphology objects.

In the case of Czech, PDT uses a different tagset than synt, but the difference is only in notation. The individual attributes have the same sets of values. It is assumed that even within a single variant, one attribute can have multiple values. Agreement in morphology

[4] However, there are certain rare cases, where anaphoric relation takes place in spite of a morphology mismatch.

is specified by the `Agreement_sms_cz` class. Two morphology objects are determined to agree when there exists a variant in one having a non-empty intersection of values for gender, number and person with some variant of the second object.

The morphology information for TiGer is for historical reasons always represented by a single variant. Nevertheless, the definition of agreement is analogic. It only needs to be remarked, that in contrast to Czech, German doesn't distinguish masculine animate and inanimate, and gender in plular.

3.3.3 Splitting Sentences into Smaller Processing Units

As mentioned in 3.1, within this framework, discourses are represented by means of `Text` objects consisting of sentences. However, for anaphora resolution, smaller processing units than sentences are more suitable. These can correspond for instance to clauses. However, this is not the only possibility – in certain cases it may be of advantage to process embedded subordinate clauses together with the main clause. The sentences within the system can be split into smaller processing units (`Clause` objects) using the implementations of the `ClauseSplitter` interface. Analogically to the interface for markable detection, it provides two methods. One determines whether the given node is a root of the desired processing unit. The second is given a sentence and a node comparator, and returns a sorted list of `Clause` objects.

To start with, the `Splitter_Uni_dummy` class defines a dummy implementation of this interface. It can be used when sentences already correspond to the desired processing units, regardless of the formalism.

For PDT, the implementation in the `Splitter_Dep_pdt_cz4` class is used. It splits the dependency trees into clauses, that is, into parts each of which has a finite verb form. Several exceptions to this need to be considered. Above all, the detector ignores all verb forms not realized on the surface (e.g. verb forms subject to ellipsis). Therefore, conjunctions like "chtěl zastavit a vystoupit" are considered to be a single clause. Further, certain trees contain nodes (esp. technical ones) that do not belong to any subtree corresponding to the detected clauses. Such nodes are assigned to all top-level clauses. Sentences lacking a finite verb form are assumed to form a single big clause.

The `Splitter_Phr_synt_cz1` class defines splitting of sentences in the synt output format. One of the grammar non-terminals corresponds to a clause root and all its occurrences are detected as clause roots.

Sentences of the TiGer corpus can be split using the `Splitter_Phr_TiGer_del` class. Clause roots are considered to be all non-terminals annotated as sentences, coordinated sentences or discourse level constituents. In cases of incomplete or erroneously annotated sentences, the virtual root created by the parser is also regarded as a clause root.

3.3.4 Matching of Definite Descriptions

This work addresses mainly pronominal anaphora, however, as mentioned in chapter 1, there are also other types of anaphorical relations. It is advantageous to interrelate the resolution of pronouns with the resolution of other phenomena. To provide the pronominal implementations of the `Algorithm` with access to mechanisms for resolving definite descriptions, the framework defines the `DDMatcher` interface.

The interface specifies four methods. The `isDD()` method determines whether a given markable is a definite description[5]. The `matchesDD()` method determines, whether a given definite description and antecedent candidate match, the `matchDD()` method extends this to a whole list of candidates and returns the first matching one. Finally, in situations, where no candidates in form of markables are available, the `searchDDMatch()` method can be used to search for the antecedent by means of tree traversal.

The resolution of definite descriptions is not the main objective of this work and should only assisst the resolution of pronouns, therefore I concentrated mainly on cheap heuristics. From these I implemented:

- `stringMatch` – matches expressions with the same surface form
- `lemmaMatch` – matches expressions with the same sequence of lemmas
- `headStringMatch` – matches expressions with the same heads
- `headLemmaMatch` – matches expressions with the same head lemmas
- `stringMED` – matches expressions the minimum edit distance of which (represented as a word sequence) is under a given threshold
- `lemmaMED` – matches expressions the minimum edit distance of which (represented as a sequence of lemmas) is under a given threshold

For all these measures, I implemented a PDT-specific class with a `isDD()` method detecting noun phrases modified by a demonstrative pronoun.

The importance of the last two measures for reference resolution is postulated by Strube, Rapp, and Müller (2002), who used the concept of **minimum edit distance**[6] within their system based on machine learning.

I performed a brief manual analysis of the results and found out, that the minimum edit distance is frequently low only owing to the presence of a demonstrative in the candidate.

[5] Plainly, this method is format-specific.

[6] This metric was initially formulated by Vladimir Levenshtein and is therefore often termed Levenshtein distance. Broadly speaking, it defines distance of two sequences through the number of substitutions, insertions and deletions necessary to transform one into the other.

def. description		antecedent candidate		distance
tato	kočka	bílá	kočka	1
this	*cat*	*white*	*cat*	
tato	kočka	tato	tramvaj	1
this	*cat*	*this*	*tram*	
XXXXX	kočka	bílá	kočka	1
this	*cat*	*white*	*cat*	
XXXXX	kočka	tato	tramvaj	2
this	*cat*	*this*	*tram*	

Table 3.1: Illustration of minimum edit distance values

This is illustrated by table 3.1, and motivated me to slighly modify this measure by substituting the demonstrative in the definite description by a non-word. Manual investigation of the data hints that the resulting measure is much more plausible.

3.3.5 Detectors of Coreference Constraints

Anaphora is a very complex phenomenon and it is of advantage to take account of the fact that other aspects of language put various constraints upon it. As mentioned above, these can be represented by means of a `Constraints` object. However, the constraints are usually not known prior to the resolution process and it may be of advantage to generate them dynamically. This functionality is provided by the `ConstraintDetector` interface. It is meant to provide relevant constraints for each given clause and lists of markables.

I implemented two constraint detectors for PDT. The `CsDet_Subj_pdt1` class yields constraints securing disjoint reference between a markable in subject position and every other non-reflexive phrase markable. The `CsDet_GrammAndSubj_pdt1` detector provides the same constraints. On top of that, it examines the information about grammatical coreference in the annotation to relate the markables in the current resolution process. In other words, this accesses coreference resolved within the syntactic analysis.

3.3.6 Generating TFA Information

Many algorithms described in chapter 2 function based on the TFA of sentences. If this information is not available in the text, it can be at least heuristically determined through an instance of the `TFABuilder` interface. It defines a single method, which is given a clause and enriches the nodes in the sentence representation with values of attributes specifying TFA.

The Prague Dependency Treebank readily contains information about TFA of all relevant nodes and it is not necessary to implement a TFA builder.

Time didn't permit implementing an automatic generator of TFA information for synt output format.

The TFA_Phr_TiGer_VFIN class contains a TFA builder for TiGer. It is based on the assumption that the border between topic and focus is often the finite verb form. Therefore, all nodes to the left of the finite verb form are considered to form the topic, the rest of the clause is considered to be the focus.

3.4 Algorithm Modules

This section describes the way how the algorithms depicted in chapter 2 are instantiated within the presented system. First, subection 3.4.1 demonstrates the algorithm module functionality in the framework on a plain algorithm based on recency. Next, subsection 3.4.2 describes the implementation of the Hobbs' Syntactic Search, subsection 3.4.3 is on the implementation of centering, and 3.4.4–3.4.5 pursue the salience-based algorithm modules. Finally, subsection 3.4.6 discusses the possibility to improve performance by combining several algorithms into a meta-algorithm.

3.4.1 Implementing Plain Recency Algorithm

The Plain Recency Algorithm implemented by the Alg_PlainRecency_* classes is based on the intuitive assumption, that antecedents closer to the anaphor should be preferred. The implementation is rather straightforward, which renders it suitable to demonstrate the functionality of the Algorithm interface in a bit more detail.

The initModel() method re-initializes the discourse model used by the algorithm. In this case, it is a simple list of markables (new markables are added to the front), which is cleared.

The performAlg() method is used to perform AR on a whole Text object. However, in reality, it is just a wrapper for the processClause() method. It initializes the discourse model and iterates over sentences. In case the sentence is not divided into clauses yet, the given clause splitter is used for this. For each clause, the current TFA builder is run, all markables are detected and coreferential constraints computed. This work is done by the implementations of the individual interfaces passed to this method as parameters. And finally, the AR process is passed on to the processClause() method, which yields a set of links. These are subsequently added to the Text object. This is done for every clause in the text.

The actual AR is performed by the processClause() method. Its main duty is to integrate the markables of this clause to the discourse model, and to compute the anaphoric links. The discourse model of this algorithm is rather uncomplicated. The anaphoric and non-anaphoric markables are combined and inserted to the beginning of the global list. Anaphors are resolved simply by iterating this list and considering each element. First element agreeing in morphology (according to the current Agreement implementation) and complying with the coreference constraints (according to the Constraints in the current Text object) is declared the antecedent. The corresponding UndirectedLink is created and all computed links are returned. Further, each non-anaphoric markable in the clause is tested for being a definite description, according to the current DDMatcher implementation. The potential resolution process is passed to the DDMatcher object itself and the resulting link is added to links found in this clause.

3.4.2 Implementing Hobbs' Syntactic Search

The implementation of the Hobbs' syntactic search algorithm (cf. section 2.2) can be found in classes anaph.algorithm.Alg_HobbsNaive_* and can be parametrized by classes in package anaph.algorithm.algDepOptions.hobbs (implementing the AlgDepOptions_Hobbs interface).

The algorithm itself is procedural, which leaves the implementation no other possibility than to strictly follow the algorithm's individual steps. Nevertheless, to open space for using the algorithm with multiple languages and for experimenting, it can be parametrized in a number of ways.

Firstly, based on the format of the syntactic trees used, it can be determined which nodes in the trees correspond to the terms in the formulation of the algorithm – i.e. which nodes represent nominal phrases, root nodes of clauses or whole sentences.

Secondly, in case the language in consideration puts different constraints on where the anaphor's antecedent should be looked for than stipulated for English, it is possible to set the algorithm to skip some of its steps where antecedent candidates are considered (i.e. steps 3, 6, 7 and 8).

Next, it is apparent that the condition put on antecedent candidates in step 6 is formalism- but unfortunately also language- specific. Therefore, it is possible to re-fit this condition to the particular language and formalism.

For Czech, it is rather unclear what this condition should look like. Plainly, the issue is not as easy as drawing the line between (31a) and (31b), as Hobbs (1978) claims it is the case for English.

(31) a. **Mr. Smith**$_i$ saw a **driver**$_j$ in *his*$_{i/j}$ truck.

b. **Mr. Smith**$_i$ saw a driver$_j$ of *his*$_i$ truck.

Similar cases are discussed at length in (Toman, 1991) where it is strictly stated that a reflexive can be bound by a clause subject only. On the other hand, it is also assumed that certain infinitive constructions, NPs together with an adjacent PP, etc. can form so-called "small clauses". These seem to be able to bind reflexives within themselves, to a phrase which can be understood as the "subject" of this small clause. Unfortunately, it seems to be very vague which phrases have the power to form a small clause and which not. Let's consider the following sentences (assumed small clauses are indicated by square brackets):

(32) **Jan**$_i$ pomáhal Karlovi$_j$ ve *svém*$_{i/*j}$ bytě.
Jan helped Karel in (REFL.POSS.) flat.
"Jan helped Karel in his flat."

(32′) Jan$_i$ pomáhal **Karlovi**$_j$ v *jeho*$_{*i/j}$ bytě.
Jan helped Karel in (PERS.POSS.) flat.
"Jan helped Karel in his flat."

(33) **Úřady**$_i$ zbavily novináře$_j$ *svých*$_{i/*j}$ nepřátel.
Authorities rid journalists (REFL.POSS.) enemies
"The authorities rid the journalists of their enemies."

(34) **Karel**$_i$ viděl [**Petrovu**$_j$ kopii *svého*$_{i/j}$ obrazu].
Karel saw Petr's copy (REFL.POSS.) picture
"Karel saw Petr's copy of his picture."

(35) **Karel**$_i$ nesnášel [**Petrovy**$_j$ ódy na *svého*$_{i/j}$ učitele].
Karel hated Petr's odes about (REFL.POSS.) teacher
"Karel hated Petr's odes about his teacher."

(36) **Vláda**$_i$ učinila [**komisi**$_j$ nezávislou na *svém*$_{i/j}$ programu].
Government made board independent of (REFL.POSS.) program
"The government made the board independent of its program."

(36′) **Vláda**$_i$ učinila [**komisi**$_j$ nezávislou na *jejím*$_{i/j}$ programu].
Government made board independent of (PERS.POSS.) program
"The government made the board independent of its program."

Clearly enough, in (32) and (33), the reflexive can refer to the subject of the whole sentence only. In (34) both co-indexations seem to be possible, even though there is a strong preference for the picture to belong to Karel, whereas in (35),(36), both co-indexations seem to be equally likely. It is very interesting to note that the complementary distribution of personal and reflexive possessives (asserted by Chomsky's principle A and B), undoubtedly valid for (32)/(32′), breaks down in (36)/(36′).

A proper treatment of such cases has been a big issue in Czech linguistics for more than a hundred years and is clearly beyond the scope of this work. The implemented algorithm was tested with several simple rules of the thumb and the results were compared (please refer to chapter 5 for details).

Finally, to complete the listing of possible adjustments, the algorithm can be used with the above-described mechanisms for handling referential constraints. It seems to be redundant to apply grammatical constraints, because all of them should be actually already reflected by the way the tree traversal is made. However, as already mentioned on page 21, the possibility to employ semantic constraints (would they be available) is quite promising.

3.4.3 Implementing Centering

There are numerous possibilities how to employ the notion of centering in an AR algorithm, from which I have chosen to implement the BFP-algorithm described on page 24. The implementing classes are `anaph.algorithm.Alg_CenteringBFP_*` and are required to be parametrized by an implementation of the `AlgDepOptions_CenteringBFP` interface. This interface allows custom specification of two important steps of the algorithm – the construction of the list of forward-looking centers for the current utterance, and the final choice of the combination of links between the current and previous utterance. The concept of utterance is formalized by the `CenteringUtterance` class, which also defines the individual transition types and how they are to be determined.

The algorithm starts off by combining the lists of anaphoric and non-anaphoric markables in the current clause into the list of forward-looking centers. The actual ordering is specified by the options object. The ranking of C_f-elements in the original formulation of centering was in terms of thematic roles, however, as argued in (Grosz, Joshi, and Weinstein, 1995), many authors and ongoing psycholinguistic research suggest that it should be based on grammatical roles and surface position.

Next, the algorithm computes the set of all possible (in terms of agreement in morphology) antecedents in $C_f(U_{i-1})$ for each anaphor and definite description in the constructed $C_f(U_i)$. If there are no antecedent possibilities, the anaphor is treated as unresolvable.

Consequently, a set of utterances is constructed, each of which stands for a different combination of anaphor antecedents and $C_b(U_i)$ (it is a cartesian product of $C_f(U_i)$ and the antecedent sets for all resolvable anaphors). Each element of this set represents a possible linking of this utterance to the previous one.

This set, however, may yield linkings which are not plausible. The algorithm proceeds by filtering it in the following way:

- rule out all combinations violating the constraints passed to the algorithm
- rule out all combinations for which it doesn't hold that C_b is the $C_f(U_{i-2})$ element with the highest ranking of those realized in U_i
- rule out all combinations violating the **RULE1**

Finally, from the remaining combinations, one is chosen based on the algorithm options. Generally, combinations with a smoother transition type to the preceding utterance are to be preferred. Further criteria are subject to experimenting.

3.4.4 Implementing Approaches of the Prague Group

The system contains implementation of algorithms referred to in section 2.4 as "Algorithm 1" (classes `Alg_Hajicova1987_*`) and "Algorithm 2" (classes `Alg_HHS1995_*`).

Both algorithms use the same representation of the SSK, embodied in the framework by the `anaph.data.model.orl.ObjRegistry` interface. It is a collection of entries, each of which stands for a discourse object together with its activation, the markable representing it, the set of markables found to be coreferent with it, morphology feature values and an identification number.

Additionally, the SSK provides various methods, most importantly methods for introducing a new entity into the SSK, and for updating an entry when it gets re-mentioned, that is, referred to by a different markable later in the discourse. To determine the antecedent, further method is defined, returning the most activated entry in the SSK, or the most activated entry matching the respective anaphoric markable in morphology features.

The default ordering of the SSK entries is based on the activation value and the order of its registering into the SSK. However, an arbitrary ordering can be specified when creating a new SSK object.

The strategy of "Algorithm 1" is rather straightforward. First, it considers all anaphoric markables in the current utterance and resolves it using the above-mentioned functionality of the SSK object. If no morphologically appropriate entry is found, the anaphor is treated as a non-anaphoric markable, i.e. it is introduced into the SSK with the activation corresponding to its membership in T or F of the utterance (as prescribed by the table 2.1). When a proper antecedent is found, the corresponding entry in SSK is updated accordingly.

However, these changes are not made directly to the activation values of the individual entries. This would have bad impact on the resolution of subsequent anaphors in the clause (and might require extra re-sorting of the SSK entries). All changes are made to an extra variable containing the postulated activation of this entity in the next utterance or the value meaning "not referred to in this utterance".

Next, the non-anaphoric markables are treated, and matching of definite description is made.

Finally, all entries in the SSK are updated – either their activation is set to the desired value, or (if the entry has not been manipulated within the processing of this utterance), it is incremented according to the last rule of table 2.1.

The algorithm can be parametrized, allowing for change of any figure occurring in the formulation of table 2.1.

The functionality of the implementation of "Algorithm 2" is very analogical. The only difference is, that it allows for much more extensive parametrization.[7] It is specified through the `AlgDepOptions_HHS` interface.

[7] Theoretically, even the exact "Algorithm 1" can be obtained by an appropriate parametrization.

The previous algorithm reflected only one possible distinction of markable position in the utterance, that is, according to the TFA. The options object of "Algorithm 2" allows an arbitrary partition of markables into any groups, for example to $P(0)$ – $P(3)$ as defined in section 2.4. It is also possible to define a custom assignment of activation when introducing and re-accessing SSK entries (not necessarily based solely on the group membership of the markable concerned). Finally, the method searching the SSK for the most activated item can be re-specified – mainly to enable ruling out antecedent candidates based on specific activation values (for instance, to reflect the claims on page 29). The most plausible combination of settings can be reached through experimentation.

3.4.5 Implementing Salience Factors

The `Alg_LappinLeass_*` classes contain the implementation of an algorithm inspired by the Lappin and Leass' RAP system.

The algorithm takes advantage of a data structure very similar to the SSK mentioned in the previous subsection. The noteworthy differences are, that the salience values are represented by real numbers (not integers) and that the measuring is inverted – that is, the higher the number, the higher the corresponding salience.

The central point of this algorithm is the notion of a salience factor, represented by the `SalienceFactor` interface. Each individual instantiation reflects specific circumstances influencing the salience of a discourse object and defines the corresponding salience gain. There are two basic types of salience factors: factors that apply to single markables (`SF1_*` classes), and factors concerning anaphor-antecedent pairs (`SF2_*` classes).

Any combination of factors can be passed to the algorithm. It resolves the anaphors by considering all candidates above a certain salience threshold, at the same time matching the anaphor in all necessary morphology features and complying with all presently known coreference constraints. Single-markable factors are applied to the anaphor and for each of its antecedent candidates, rewards of all the markable-pair factors. The candidate with the highest resulting salience score is chosen to be the antecedent and is assigned the sum of the salience values concerning it and the anaphor.

Non-anaphoric referential expressions are assigned the sum of single-markable factor values and introduced into the set of discourse objects.

The framework allows formulating arbitrary factors and rewarding them with custom weights. Starting with the original values mentioned in table 2.2, the values yielding optimal performance can be reached through experimentation.

3.4.6 Meta-algorithms

As already discussed in chapter 1 and 2, despite the discussed AR algorithms aim to model tendencies and regularities apparent in NL texts, they can not be expected to perform flawlessly. Their discourse models keep track only of a limited number of aspects needed for the proper resolution of all coreferential links in texts. However, different algorithms choose the antecedent in different ways and therefore where one fails, other ones may succeed.

This observation can be used to build a more successful AR algorithm. The framework makes it possible to have more algorithms running in parallel, each providing us with the antecedents it has computed. If we have any knowledge about the performance of the algorithms, or we even know one of them has proved to be successful with similar instances, we can use this knowledge to prevent making unnecessary errors.

The system contains ready-made classes (`Alg_Meta_*`) for experimenting with meta-algorithms. It is a standard implementation of the `Algorithm` interface and is required to be parametrized by an object with options. These include the specification of the individual algorithms to be used, their names and options. Most importantly, the options object has to define how the choice of the antecedent for each anaphor should be made, having access to the anaphoric markable in question and all anaphoric links the individual algortihms found for this utterance.

The antecedent choice can be based for example on the performance of the algorithms on the individual anaphor types. The proposal of the algorithm exhibiting the highest performance on the actual anaphor type is accepted. Another possible strategy is to select the antecedent found by the most algorithms (possibly using suitable weighting).

In case there is an algorithm with just minor flaws, it may be of advantage to invert the view of the situation and to concentrate rather on where the algorithms make errors (than on seeking their strong points). The results of the most successful algorithm are used, except for instances it is known to perform poorly on. To resolve these, the next best algorithm is used etc.

The most effective strategy strongly depends on the actual algorithms and the distribution of their results.

Chapter 4

Evaluation

This chapter describes the principles of evaluating anaphora resolution algorithms and systems[1], presents selected state-of-the-art scoring techniques and discusses their qualities and problems.

Scoring mechanisms play a very important role, because they show us, whether our algorithms are applicable in given conditions. A good score for a particular construct hints us that we can count with its high performance. On the other hand, bad score reveals certain flaws and more or less directly points to sections we should improve. It can also be an impulse to choose a different AR algorithm, or to alter its parametrization.

In other words, the evaluation methods used and the resulting figures directly determine whether, and when, how will the algorithm (or system) be used. To facilitate deducing the quality of the algorithm from the evaluation figures and its comparison to other algorithms, it is desirable to take advantage of well-known, standard evaluation methods. This chapter describes the most important ones and their implementation in the presented system.

The first section of this chapter contains general notes on the role of evaluation in the field of anaphora resolution and related outstanding issues. Next, sections 4.2 through 4.4 provide an overview of the most influential anaphora resolution evaluation techniques, followed by the details of their implementation in section 4.5.

4.1 Overview

Proper evaluation of algorithm's performance may be even more important than the algorithm itself, at least in the field of computational linguistics.

As we have seen in the previous chapters, there is a whole variety of anaphora resolution approaches and underlying models. Very often, differences in their fundamental ideas make it impossible to compare them transparently. As pointed out by Carletta (1996), a lot of work in computational linguistics depends on subjective judgements, and in the

[1] Let's understand the term "anaphora resolution system" as any complexer linguistic application accepting raw (unannotated) input and performing anaphora resolution. Whether, and how the (internal) AR results are subsequently used (e.g. in a dialogue model, in machine translation) is, from our point of view, irrelevant.

past, research was frequently judged according to the plausibility of author's explanations. Plainly enough, this approach to evaluation lacks solid grounds[2] and a well-defined, quantitative approach should be used instead.

Unfortunately, even if we adopt a plausible scientific scoring mechanism and feed it with very costly human-annotated data, there is still a number of issues which can have an undesirable effect on the result figures.

Firstly, as explained in detail in (Mitkov, 2001), we have to realize there is a big difference between evaluating performance of an AR algorithm and a whole AR system. When assessing an AR algorithm, we usually feed it with disambiguated, human-annotated input. The task of an AR system, which has to do the whole linguistic analysis on its own, is much more difficult. Any mistake made in the pre-processing such as POS tagging, named entity recognition, NP extraction, identification of pleonastic pronouns, etc. may influence the subsequent processing and the mistake can get projected up to the AR level.

A further point raised by Mitkov (2001) concerns pre- and post-processing tools as such. Even if we strictly distinguish evaluating AR algorithms and AR systems, the way of obtaining the linguistic information still needn't be uniform. There are various parsers and corpora providing the required information in different format, detail and at different reliability rate.

The next very important factor is the data. When comparing AR algorithms, for various technical reasons it is rarely possible to run all the concerned algorithms on the same data. Unfortunately, besides the fact that any evaluation is influenced by the amount of the testing data, performance of many algorithms varies considerably across languages and genres. A model adjusted to a certain domain may perform fairly poorly when faced with a text of a different domain – for instance different referential expressions may be common there, or they may be generally used in a different way. Even within a given domain, a style of an individual author may be referentially "rather complicated" or "rather smooth". Although several measures for quantifying resolution complexity[3] have been formulated, they are either model-dependent or too complex to be obtained automatically.

For all these reasons, expressing AR algorithm's performance by the means of a single number may be too simplistic. A much more suitable approach is to use quantitative scorers not to *express* the algorithm's performance, but just to *compare* it with other algorithms.[4] This is, in addition, a good way of exposing the algorithm's strong points and weaknesses, an overview of which should be part of every AR evaluation.

[2] Human intuition is often right, however, not less often it happens to overlook important "details" and may be dangerously misleading. Let's think for example of the well-known paradox of Achilles and the turtle formulated by Zeno of Elea.

[3] See (Mitkov, 2001) for further details.

[4] Hobbs' naive syntactic approach is commonly chosen as a baseline, mainly because it is simple (to re-implement) and despite it's simplicity and year of publication its performance is still a challenge for the state-of-the-art algorithms.

Following sections describe specific anaphora resolution scorers, starting with the simplest and, at the same time, most important ones. Next section pursues scoring methods based on the traditional notion of precision and recall, section 4.3 sketches the formulation of the scorer proposed at the 6th MUC conference, and section 4.4 discusses the Standard Disclosure report format suggested by Donna Byron. Finally, the whole section 4.5 gives details of the implementation of these evaluation methods in the presented system.

4.2 Precision and Recall

The simplest way of evaluating AR results is based on ideas and terminology common in machine learning and information retrieval and was formulated by Aone and Bennett (1995).

To be able to adopt this evaluation method, we have to accept viewing anaphora resolution as a classification task, that is, assignment of one of a fixed set of potential antecedents to each anaphor.[5] The result to evaluate is thus a set of binary "links" (anaphor–antecedent pairs). By confronting it with a "golden standard" (a set of *correct* links), we can compute the following measures:

$$R = \frac{\textit{number of correct resolutions}}{\textit{number of anaphors identified by the system}} \tag{4.1}$$

$$P = \frac{\textit{number of correct resolutions}}{\textit{number of attempted resolutions}} \tag{4.2}$$

$$F = \frac{(\beta^2 + 1).P.R}{\beta^2.P + R} \tag{4.3}$$

The first measure[6] is termed **recall** and characterizes *the coverage* of the evaluated algorithm – the proportion of anaphors it can handle. The second measure is known as **precision** and describes *how successful* the resolution is. The given formulae produce real numbers from 0 to 1, however, the results are traditionally given in percent.

As it is not difficult to build an algorithm with $R = 100\%$ (by assigning every potential antecedent to every anaphor) or $P = 100\%$ (by resolving only anaphors, the antecedents of which are evident, regardless of how rare this is), recall and precision figures are always stated together.[7] Additionally, these figures are sometimes supplemented with a measure

[5] Accepting this view is, when confronted with the theory of reference, rather misleading. There is no direct link between an anaphor and its antecedent – they only both have a referential link to the same object. It is more transparent to think about coreference information in terms of equivalence classes (sets).

[6] By the *anaphors identified by the system* we understand the anaphors the system "knows about" (and "knows" it is supposed to resolve them) – even if it may leave some of them unresolved in the result.

[7] By simple modifications to the algorithm it is usually possible to increase R at the cost of decreasing P and vice versa. Different trade-offs in this sense may be suitable for different applications.

(4.3) called the **F-measure**. It combines and reflects both recall and precision, therefore it also characterizes both coverage and success. The β parameter in formula (4.3)[8], makes it possible to adjust the importance of recall in the resulting figure. When we put $\beta = 1$, precision and recall are treated as equally important, increasing β increases the importance of recall and, analogically, decreasing β decreases the importance of recall.

Unfortunately, there is a slight variance among authors in the definition of recall. Some follow the above-stated definition of Aone and Bennett (1995), but others, e.g. Baldwin (1997), tend to claim the recall definition shouldn't consider only the anaphors identified by the system, but *all* anaphors as *in the golden stadard*. In his struggle to clarify the terminological ambiguity, Mitkov (2001) named this measure **success rate** and formulated it as follows:

$$SR = \frac{\textit{number of correct resolutions}}{\textit{number of all anaphors}} \tag{4.4}$$

All of the stated measures seem to be rather vague in explaining what exactly should be treated as *"a correct resolution"*. This may vary slightly across authors, depending on the framework and theory they use, and should always be clarified by the author in the accompanying text.

Another useful, and closely related, way of expressing evaluation figures widely used in machine learning and classification is the so-called **confusion matrix**. It is a matrix M, where $m_{i,j}$ represents the number of items belonging to the (coreference) class i classfified by the system as members of the class j (the diagonal thus reveals the number of correct resolutions). Unfortunately, this original notion is not very suitable for reporting AR results, as there is no predefined set of classes. Nevertheless, it may still be of great advantage to use this concept in a slightly adapted way. One of the many possibilities is to confront individual anaphors (as rows) with the individual antecedents (as columns), stating `true/false` for every resolution performed by the system. If there is a more general classification of anaphors available, a confusion matrix comparing these classes may also yield an interesting insight into the algorithm's performance.

Evaluation measures based on the above-stated definitions of recall and precision were not tailored for the needs of computational linguistics and their use may thus lack elegance and have certain undesired qualities. On the other hand, they are simple and as classification tasks are common in many fields, also widely understood.

4.3 MUC-6 Scoring

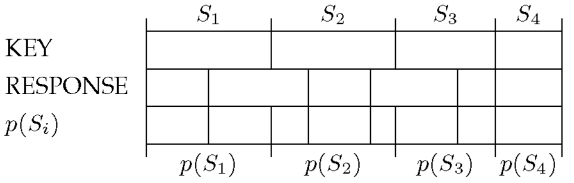
The following scorer was proposed by Vilain et al. (1995) within the *Coreference task* of the sixth Message Understanding Conference (MUC-6). In contrast to the scorer from the previous section, the MUC-6 scorer doesn't operate with "links", but rather with equivalence

[8] The parameter is more precisely termed *relative importance given to recall over precision*.

classes obtained from the respective coreference chains. This perspective is apparently more plausible, because it doesn't compare the individual links that add up to coreference classes, but these classes as such, regardless of the actual way the corresponding links induce them. It is thus not necessary to make, sometimes very uneasy, decisions whether a particular link realization is correct, on the other hand, this may also give credit to blatantly wrong link combinations, which by chance, or pure luck, lead to the same coreference classes.

Vilain et al. (1995) name the manually annotated coreference chains the **key**, and the coreference chains obtained as system output the **response**.

Let T be the entire set of expressions, S_i the equivalence classes as generated by the key and R_j the equivalence classes as generated by the response. For each S_i we define $p(S_i)$ as the partition of S_i relative to the response. In other words, let $p(S_i)$ consist of classes, each of which contains only elements that belong to S_i and to the same R_j for some j. This construction is illustrated by the following figure:



Let us define $c(S_i)$ as the minimal number of "correct" links necessary to generate the equivalence class S_i. For $c(S_i)$ clearly holds

$$c(S_i) = |S_i| - 1 \tag{4.5}$$

Let us further define $m(S_i)$ as the number of "missing" links in the response relative to the key class S_i. This is the number of links necessary to reunite the components of the $p(S_i)$ partition, therefore

$$m(S_i) = |p(S_i)| - 1 \tag{4.6}$$

Then, **recall** can be defined on S_i as

$$R = \frac{c(S_i) - m(S_i)}{c(S_i)} \tag{4.7}$$

which can be in turn simplified as follows

$$R = \frac{c(S_i) - m(S_i)}{c(S_i)} = \frac{(|S_i| - 1) - (p(S_i) - 1)}{|S_i| - 1} = \frac{|S_i| - p(S_i)}{|S_i| - 1} \tag{4.8}$$

By extending the definition of recall from one equivalence class to the whole set T, we get the form

$$R = \frac{\sum(|S_i| - p(S_i))}{\sum(|S_i| - 1)} \tag{4.9}$$

Precision is computed using the same formula, only with the roles of key and response switched.

Even though the above-described scorer gives hardly any room for subjective interpretation of data correctness and it was specifically tailored for scoring coreference tasks, it still exhibits certain shortcomings. Bagga and Baldwin (1998) remark for instance, that the scorer doesn't give any credit for separating out a singleton entity incorrectly assigned to a bigger coreference chain.[9] Additionally, they argue that the MUC-6 scorer regards all errors to be equal, whereas in reality some may cause much greater damage than others. [10]

4.4 Byron's Standard Disclosure

The previous sections were pursuing the exact methodology of quantitative evaluation, that is, the ways of expressing AR result quality *in numbers*. Numbers play a central role in every evaluation, but mere numbers do not suffice. The more exact numbers are, the more they, unfortunately, tend to distract from the substance they actually measure. As a result of this, some authors describe in the minutest detail *how successful* their systems are, but almost leave the reader in the dark in *what* do their systems *exactly do*.

There are only few systems performing entirely robust anaphora resolution. The exact goals of all the various systems are in reality quite different – some attempt to resolve all anaphoric expressions inclusive of definite descriptions, some limit themselves to pronominal anaphors, but the most systems, as a matter of fact, resolve only *certain types* of anaphoric pronouns. The anaphor types covered do not only differ across systems, the respective authors describe them also in varying detail, sometimes using incompatible terminology. All this makes AR system evaluation much more complex than computing a couple of numbers.

Well aware of this difficult situation, Byron (2001) proposed a more sophisticated way of reporting resolution results, so that it accounts for all the above-mentioned issues. The original proposal was formulated for reporting results of pronominal resolution, however, the author herself explains that is can be straightforwardly used for any resolution task.

The main idea of the proposal lies in taking account of the entire task in its full complexity, not only of its subtask addressed by the system. The proposed report format, named **standard disclosure**, simultaneously reflects the actual decomposition of the evaluation data into several parts.

The first part of the data to be reported on consists of *exceptions*, that is, cases which actually do not belong to the task, but confusingly look very much as if they would. Byron

[9] From the other perspective, it doesn't penalize incorrect assignment of singleton entities to other coreference chains.

[10] In most applications, a link wrongly connecting two distinct coreference classes causes the more damage, the larger, or the more important these classes are. The exact importance of an individual error may be very system-specific though.

(2001) names them **non-referential exclusions**. In pronominal resolution these are mainly pleonastic pronouns, and in more general nominal resolution these would be definite noun phrases that do not refer[11].

Further section of the standard disclosure format describes the *uncovered phenomena*, referred to by Byron (2001) as **referential exclusions**. Even though they are in general a part of the task, the system is on purpose designed not to resolve them. Either because they are too complex (e.g. abstract reference), or because they are not interesting (e.g. first/second person pronouns, demonstratives, etc.).

The last section of the proposed report format refers to the actual domain of the system, the so-called **evaluation set**, that is, the anaphors the system is actually designed to resolve. This is the suitable place for stating quantitative evaluation measures. Byron (2001) also proposes a new metric characterizing the proportion of the general resolution task the system performs correctly, the **resolution rate**:

$$RR = \frac{|correct\ resolutions|}{|evaluation\ set| + |referential\ exclusions|} \tag{4.10}$$

A sample report in the proposed format is given as Table 4.1.[12] By reading it top-down, it can be seen how transparently the table reveals the structure of the task – what does the particular task encompass, in which extent it is addressed by the system, and how successful the system is in doing this. In addition, the columns of the table allow structuring the report with regard to the individual lexical items, or, more generally, to the individual *categories* in scope.

Further features of the format are mostly self-explanatory and their detailed description goes beyond the scope of the present work. Discussion offering a deeper insight into the format can be found in (Byron, 2001).

4.5 Implementation

This section discusses the implementation of the above-described scorers in the presented framework.

Firstly, subsection 4.5.1 contains general information on what role evaluation modules play in the whole framework and how they interact with other modules. Further subsections give more detail about the implementation of the individual scorers.

[11] Such expressions may occur for instance in idioms, like *the dogs* in the sentence *"The company went to the dogs during the recession in the thirties."*

[12] The given example is a slightly altered version of the sample presented in (Byron, 2001).

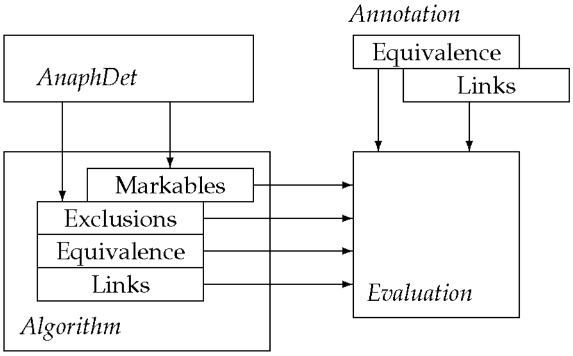
Evaluation Corpus Name: Peanut dialogues
Genre: Two-party problem-solving dialogues
Size: 15 dialogues, 937 turns, 31 minutes total speaking time

Anaphor Types:	*Her*	*She*	*He*	*Him*	*His*	*It*	*Its*	Other	Total
A: Raw Word Count	22	25	89	44	7	94	12	186	479
Non-Referential Exclusions									
Pleonastic	0	0	0	0	0	6	0	2	8
Abandoned Utterance	0	0	1	0	1	0	0	2	4
B: Sum Non-Referential	0	0	1	0	1	6	0	4	12
C: Referential (A-B)	22	25	88	44	6	88	12	182	467
Referential Exclusions									
Plural	0	0	0	0	0	0	0	120	120
Demonstrative	0	0	0	0	0	0	0	36	36
1st/2nd Person	0	0	0	0	0	0	0	24	24
Reported Speech	0	0	1	0	0	0	0	2	3
Event Anaphora	0	0	0	0	0	15	0	0	15
D: Sum Ref Exclusions	0	0	1	0	0	15	0	182	198
E: Evaluation Set (C-D)	22	25	87	44	6	73	12	0	269
Results									
Technique Alpha									
F:#Correct: Ante (Inter)	7/7	16/17	35/45	20/21	2/3	30/41	2/3	0	112 (72%)
F:#Correct: Ante (Intra)	15/15	7/8	35/42	20/23	3/3	24/32	9/9	0	113 (86%)
Errors: Cataphora	0	0	7/7	0	0	3/3	0	0	10
Errors: Long Distance	0	2/2	4/4	0	0	4/4	0	0	10
G:#Correct: Refs	21	22	67	38	5	52	11	0	216 (75%)
Errors: Chaining	1	0	0	1	0	0	0	0	2
Resolution Rate (G/C)	95%	88%	76%	86%	83%	59%	92%	0%	45%
New Technique Beta									
H:#Correct: Ante (Inter)	5/7	17/17	45/45	15/21	2/3	34/41	3/3	0	121 (88%)
H:#Correct: Ante (Intra)	15/15	7/8	31/42	23/23	3/3	27/32	6/9	0	112 (85%)
Errors: Cataphora	0	0	7/7	0	0	1/3	0	0	8
Errors: Long Distance	0	2/2	4/4	0	0	4/4	0	0	10
I:#Correct: Refs	20	23	76	38	5	61	8	0	231 (86%)
Errors: Chaining	0	0	1	2	0	0	0	0	3
Resolution Rate (I/C)	91%	92%	87%	86%	83%	69%	67%	0%	49%

Table 4.1: Sample standard disclosure for a fictional system

4.5.1 Evaluation modules in the framework

The implementation of the individual AR scorers and related matters can be found in the `anaph.eval` package. A sketch of an evaluation module's interaction with other modules of the framework is shown on the following figure:



Each evaluation module gets two kinds of information – information about the so-called "golden standard", i.e. the annotation we want to compare our results against, and the various information resulting from the run of the resolution algorithm we want to evaluate.

It is important that both these information components are compatible with each other, that is, they consist of objects based on the same structure in the same text. In the ideal case, the annotation and the system's output is over markables obtained through the same anaphor detector. Otherwise, the differences in markable composition may influence the evaluation results considerably.

The first part of the information is the annotation. It consists of a set of co-referential links and the equivalence they generate. This represents the true division of markables into co-reference classes and the links that induce them. This information can be obtained either through an input module fed with annotated data, or by taking the results of a baseline algorithm.

The second part comprises the algorithm output and is slightly more complex. It also contains a set of co-referential links and an equivalence, like the previous part. On top of that, it also contains the outputs of the anaphor-detector module. Firstly, these are markables detected during the algorithm run in the respective text. And secondly, these are the so-called exclusions[13]. These are produced at each anaphor detector call and, together with their types, saved to a set given as a parameter. What exactly is determined to be a markable and what is considered as an exclusion, depends on each anaphor detector.

[13] For a more detailed description of exclusions and their properties, see section 4.4.

Generating of exlusions can be suppressed by passing NULL to the anaphor detector as the respective parameter.

4.5.2 Precision and Recall

The measures based on the traditional notions of precision and recall are implemented by the anaph.eval.Classic class. It allows computing measures described in section 4.2 according to the formulae 4.1 through 4.4. It is possible to compute each measure separately, or to save a report containing all figures.

Correct resolutions, are considered to be not only links linking the same anaphor to the same antecedent as in the annotation, but also links linking the anaphor to any markable in the same co-reference class according to the annotation.

The implementation itself takes advantage only of the numbers appearing in formulae 4.1 – 4.4 and is very straightforward.

4.5.3 MUC-6 Scoring

The implementation of a scorer based on the MUC-6 model theoretic notions can be found in the anaph.eval.Muc6 class. It provides methods for separate computation of precision and recall (as described in section 4.3) and a possibility to save an overall report.

The computation is based on partitioning the annotation's equivalence classes with regard to the system output and vice versa. This is modeled by a structure assigning each golden standard's class and system output class a set of markables that belong to the intersection of these classes. This structure directly yields all figures necessary to compute recall (or precision, respectively).

4.5.4 Byron's Standard Disclosure

The anaph.eval.StandardDisclosure class contains a generator of reports in the style of Byron's stadard disclosure. Unlike the previous scorers, this scorer processes and assesses the information about exclusions. The report is written in the LaTeX source format.

The implementation is rather technical and is based on building several auxiliary structures. Firstly, all markable types occurring in the data are gathered and are used as column labels for the resulting table. This is a straightforward extension to the Byron's version using lexical items. It is possible to get diverse views of the data set by using different anaphor detectors varying in the types they assign to markables.

Secondly, all exclusion types (these can vary across anaphor detectors as well) are summed up and used to build a structure assigning each markable and exclusion type the set of all corresponding (excluded) markables. This structure is used to generate the whole part of the report pursuing exclusions.

Next, another structure is built, yielding a markable set for each combination of a markable type and link type (as in the system's output). Analogically to the Classic module, each of these links is checked for correctness and all the performance measures are generated accordingly. Each cell expresses *how many links out of how many are correct* which is then the basis for the line totals stating the number of correct links and their proportion in percent among all links of the particular type. The column totals contain the resolution rate of the individual markable types.

The fact that the annotation and the system results can contain different markables of different types treated it necessary to include information relating them within the report. For each algorithm result, there is a line giving information about how many markables of the given type were detected by the system, and how many of them correspond to markables in the annotation (the *common/system*-line). Further, to link the correctly resolved anaphors not only to the annotation, but just to its part common with the markables yielded by the system, the generated table provides a *correct/common* line. This reduces the influence of the detector performance on the presented figures.

In contrast to the sample report presented by Byron, the report generated by this module doesn't contain any information about errors. Consistent summarization of errors requires human intervention and is left to the post-processing phase. As soon as clarity and transparency of information is maintained, addition of any information during the post-processing phase is highly encouraged.

Chapter 5

Results

This chapter discusses the performance of the individual algorithms implemented within the framework. The first section contains a few notes on the performance of the algorithms in their original implementations and discusses an already existing AR system for Czech. Finally, section 5.2 shortly presents the data used, and gives account of the AR results, including notes on the underlying experimentation. The next chapter concludes the work by summarizing the results and sketching directions for further research.

5.1 Performance of the Original Implementations

This section provides a brief account of the performance of the presented algorithms, as implemented by their authors.

Starting with the oldest algorithm, Hobbs' implementation of his syntactic search was evaluated on a collection of newspaper, novel and historical texts (a total of approximately 60 pages). After removing expletive "it" occurrences from the data, the algorithm worked successfully in 88.3% of cases. This was further enhanced by employing semantic information.

Lappin and Leass' RAP system was reported by its authors to succeed in 86% of cases. However, the evaluation was performed on a rather small data set, containing just 360 pronouns. It can be assumed that the given figure results from a genre-specific combination of factors and on generic texts, the system's performance would drop slightly.

The BFP-algorithm was presented without evaluation figures. It was employed in an HPSG-based interface to a database query application. However, Jurafsky and Martin (2000) argue that BFP was evaluated by others and mention accuracy of 77.6%.

To my knowledge, no quantitative evaluation of the Prague group approaches has been published so far, except for a superficial note in (Hajičová, Kuboň, and Kuboň, 1990).

The only AR system implemented for Czech known to me was presented by Kučová and Žabokrtský (2005). It is based on a set of rules filtering out implausible antecedent candidates. After all filters have been applied, it selects the closest of the remaining candidates as the antecedent. The system was evaluated on PDT 2.0 and is said to have a success rate of 60.4%.

However, it could be argued that this measure is biased by several issues. Firstly, it uses the annotation data to detect anaphors. Secondly, it treats nodes in certain artificially generated constructions as textual anaphora. These constructions are very regular and are therefore predestined for successful resolution by a rule-based system. These cases include for instance the identification of unvoiced actants of nominal phrases, or more plainly, consturctions expressing comparisons. Each of these contains two trees, one for the original event and another for the event it is compared to. The second tree is a copy of the first one and differs only in the participants compared by the phrase. The participants present in both trees are connected by links of the "textual" type, however, in my opinion, they are clearly technical. I discovered 1353 occurrences of the two phrase types mentioned, which is a considerable part of the annotated data.

In spite of these details, the only notable flaw of this work is, in my opinion, that it is PDT-specific and would have to be re-implemented from scratch to apply to other data.

5.2 Performance of the Implemented Algorithms

This section provides information about the performance of the algorithm implementations within this system. First, I briefly mention the data used for the evaluation, then I address the individual algorithms.

As already mentioned in 3.2.2, the corpus used for evaluation of AR within this system is PDT 2.0 in its preliminary draft from March 2005 (Hajič and others, 2005). It contains dependency tree representations of about 50.000 sentences (grouped into 23 document sets) with approx. 45.000 coreference links[1]. I used trees of the tectogrammatical level.[2] The evaluation is performed by the classes of the `anaph.eval` package and only links annotated as textual are considered. Anaphors in constructions mentioned in the previous section are excluded. The standard disclosure given in table 5.1[3] represents results for the whole corpus, the figures in table 5.2 are averages of numbers for the individual data sets.

The implementation of "Algorithm 1" provides various flavours of SSK. They differ at the point of integrating a resolved anaphor into the model. I initially assumed that it is plausible to include the anaphor in the SSK entry of its antecedent, just updating its morphology features in a certain way. However, an implementation providing a brand new SSK entry for each anaphor outperformed all the other models. Further, I experimented with the specification of the activation rules. The original formulation is to assign the activation of 1 to phrases in the topic and 0 to the ones in focus. However, the exactly opposite assignment yielded slightly better results. Time didn't allow a thorough analysis of this

[1] Approximately 23.000 of the annotated links are grammatical, and approx. 22.000 textual.

[2] PDT 2.0 contains two more description levels, morphological and analytical (Kučová and Žabokrtský, 2005).

[3] Many columns of the table, containing rather marginal data, had to be removed to make the table dimensions fit the page.

fact, however, I assume this paradox could be credited to interplay with syntactic phenomena.

The "Algorithm 2" exhibited the best performance for the default parametrization described in section 2.4. I experimented with dividing the utterance according to grammatical roles, however, the resulting performance was rather unstable. It was slightly more successful on few instances, but noticeably poorer on the most of the data.

The parametrization of Hobbs is quite straightforward. All efforts to define a plausible condition for step 6 have failed. The trivial strategy to always prevent the node in step 6 from being the antecedent yielded the best results.

The BFP algorithm exhibited interesting behaviour. First, I experimented with the ordering of C_f items. Orderings based on surface position, anaphors first, other markables afterwards, performed very comparably to orderings based on grammatical or thematic roles. However, after re-specifying the sorting of remaining link combinations to reflect grammatical roles as well, the performance has risen considerably. The ordering based on grammatical roles has already been used by many researchers. Although the best ordering of C_f elements is still searched for, grammatical roles can be rewardingly used for most languages.

The factors in the algorithm inspired by the Lappin and Leass' RAP system make it possible to extensively experiment with their weights. However, the weights presented by the authors guarantee fair performance which is very hard to improve. I was unable to find a combination of factor weights that would perform better. With the default parametrization, this algorithm exhibits the best performance of the algorithms implemented within the system. Table 5.1 and 5.2 contain further details about the results.

The visualization of the results through a standard disclosure shows that the factor-based algorithm is not the most successful on all anaphor types. This suggests the possibility to achieve a better performance by constructing a meta-algorithm. I used the factor-based algorithm and the centering algorithm to build a meta-algorithm, where centering addresses the anaphor types it is overall more successful on. The results of this composition is stated as the last line of table 5.1. It can be seen that it improved the results by 2-3%.

The fact that centering outperforms the factor-based approach on demonstrative and zero pronouns is not a coincidence. Both of these pronoun types are typically used to refer within the preceding clause, which is the main scope of the centering model. On the other hand, it ignores farther antecedents and fails in resolving them. Salience-based models are more suitable for such instances, because they are more general and not locally restricted.

Further, the results clearly demonstrate that among pronouns, demonstratives are the most difficult to resolve. This is not a surprise. They often refer to abstract entities, whole discourse segments, are used as syntactic markers, or are even deictic. However, the algorithms perform rather poorly also on possessive pronouns, which would be expected to have similar referential properties as other pronouns.

A brief analysis of the data hints that there may be several reasons. Firstly, under the

Anaphor Types:	*DEMO*	*FULL*	*WEAK*	*ZERO*	*POSS*	*TEXTUAL*	Total
A: Raw Word Count	6596	1561	445	5075	1458	21600	36735
Non-Referential Exclusions							
DEFDESC_DET	5295	0	0	0	0	0	5295
B: Sum Non-Referential	5295	0	0	0	0	0	5295
C: Referential (A–B)	1301	1561	445	5075	1458	21600	31440
Referential Exclusions							
PERS_12	0	1561	445	5075	1458	0	8539
SUBORD_DET	1301	0	0	0	0	0	1301
D: Sum Ref Exclusions	1301	1561	445	5075	1458	0	9840
E: Evaluation Set (C–D)	0	0	0	0	0	21600	21600
Results							
Hajicova87							
common/system	3461/4493	3255/3302	691/694	8183/10922	3129/3135	0/0	18719/22546
ANAPH	465/4493	1391/3302	284/694	3359/10922	929/3135	0/0	6428 (28.51%)
correct/common	13.43%	42.73%	41.1%	41.05%	29.69%	0%	34.34%
Resolution Rate	8.02%	28.6%	24.93%	21.0%	20.23%	0%	19.85%
HHS95							
common/system	3461/4493	3255/3302	691/694	8183/10922	3129/3135	0/0	18719/22546
ANAPH	542/4493	1493/3302	242/694	3490/10924	932/3135	0/0	6699 (29.71%)
correct/common	15.66%	45.86%	35.02%	42.65%	29.78%	0%	35.79%
Resolution Rate	9.35%	30.7%	21.24%	21.82%	20.29%	0%	20.68%
Hobbs							
common/system	3461/4493	3255/3302	691/694	8183/10922	3129/3135	0/0	18719/22546
STEP_3	10/201	97/279	19/73	140/1072	0/3	0/0	266 (16.2%)
STEP_4	243/3775	586/2175	104/438	1541/6653	292/1182	0/0	2766 (19.45%)
STEP_7	28/234	326/676	49/148	1214/2549	669/1480	0/0	2286 (44.93%)
STEP_8	2/96	0/148	1/31	23/428	5/461	0/0	31 (2.66%)
UNRESOLVED	0/187	0/24	0/4	0/220	0/9	0/0	0 (0%)
correct/common	8.18%	31.0%	25.04%	35.66%	30.87%	0%	28.57%
Resolution Rate	4.88%	20.75%	15.19%	18.24%	21.03%	0%	16.52%
BFP							
common/system	3461/4493	3255/3302	691/694	8183/10922	3129/3135	0/0	18719/22546
ANAPH	930/4493	1369/3302	303/694	4738/10922	1008/3135	0/0	8348 (37.03%)
correct/common	26.87%	42.06%	43.85%	57.9%	32.21%	0%	44.6%
Resolution Rate	16.05%	28.15%	26.6%	29.6%	21.95%	0%	25.78%
LappinLeass							
common/system	3461/4493	3255/3302	691/694	8183/10922	3129/3135	0/0	18719/22546
ANAPH	554/4493	1654/3302	353/694	4666/10922	1144/3135	0/0	8371 (37.13%)
correct/common	16.0%	50.81%	51.08%	57.02%	36.56%	0%	44.72%
Resolution Rate	9.56%	34.01%	30.99%	29.17%	24.90%	0%	25.85%
MetaAlg							
common/system	3461/4493	3255/3302	691/694	8183/10922	3129/3135	0/0	18719/22546
BFP	931/4493	0/0	0/0	4740/10924	0/0	0/0	5671 (36.78%)
LL	0/0	1667/3302	352/694	0/0	1143/3135	0/0	3162 (44.34%)
correct/common	26.90%	51.21%	50.94%	57.92%	36.52%	0%	47.18%
Resolution Rate	16.07%	34.27%	30.90%	29.63%	24.88%	0%	27.27%

Table 5.1: Standard disclosure for the presented system

	Haj87	HHS95	Hobbs	BFP	L&L	Meta
`Classic`						
Precision	31.25	32.43	27.11	50.48	40.63	48.93
Recall	30.75	32.34	26.62	36.97	39.68	39.24
F-measure	30.99	32.17	26.86	42.67	40.15	43.54
Success rate	33.46	34.73	28.96	40.24	43.18	42.70
`MUC-6`						
Precision	38.46	39.74	38.68	49.30	47.24	48.51
Recall	33.81	35.12	33.85	36.03	43.47	38.96

Table 5.2: Performance of the system in traditional measures

influence of Germanic languages in media, people increasingly tend to misuse possessive personal pronouns in situations where reflexive forms would be adequate. Secondly, the models used in the implemented algorithms model extra-sentential anaphora. However, phrases like (37) are quite common, but clearly intrasentential.

(37)	Karel	a	jeho	kamarádi
	Karel	and	his	friends

An ordinary personal pronoun couldn't be used in such syntactic position, which hints that possessive pronouns have a more intricate referential domain and should be preferrably addressed separately.

Chapter 6

Conclusions

This work presented a modular framework for anaphora resolution and disscussed the advantages of modularity in AR systems. Further, it described how this framework instantiates selected, mainly salience-based algorithms. The algorithms were consistently evaluated using a broad variety of measures and it was shown, that the modularity of the system allows improving their performance by combining them into meta-algorithms.

Plainly, there is still a very long way to go towards an adequate treatment of anaphora for Czech and this work is just the first step. Necessarily, more complicated models and precious resources need to be employed in this process. Especially, it is desirable to utilize the currently pursued semantic resources and the results of the fast-progressing field of psycholinguistics.

Appendix A

System API Overview

This appendix contains a list of the most important packages. A more detailed insight into the system API can be obtained by referring to the Javadoc documentation on the enclosed disc.

- `anaph.algorithm`
  contains the classes with the implemented algorithms

- `anaph.algorithm.algDepOptions`
  contains options objects for individual algorithm types

- `anaph.algorithm.options`
  contains specifications of interfaces to the algorithm

- `anaph.algorithm.options.agr`
  contains specifications of agreement

- `anaph.algorithm.options.anaphdet`
  contains classes with detectors of anaphoric markables

- `anaph.algorithm.options.clausesplit`
  contains classes for splitting sentences into clauses

- `anaph.algorithm.options.cstrdet`
  contains detectors of coreference constraints

- `anaph.algorithm.options.ddmatcher`
  contains methods for matching definite descriptions

- `anaph.algorithm.options.refexprdet`
  contains detectors of non-anaphoric markables

- `anaph.algorithm.options.tfabuild`
  contains methods for heuristic annoatation of TFA information

- `anaph.converters.in`
  contains input modules (for loading data)

- `anaph.converters.out`
  contains outpus modules (for saving data)

- `anaph.data`
  contains definitions of besic object types

- `anaph.data.model.centering`
  contains structures used by the centering algorithm implementation

- `anaph.data.model.or1`
  contains definitions of discourse models for salience-based methods

- `anaph.data.model.or2`
  contains definitions of discourse moduels for factor-based methods

- `anaph.eval`
  contains methods for evaluation of AR

- `anaph.tools`
  contains various useful, often format-dependent, methods

- `anaph.run`
  contains various pre-defined applications

Bibliography

Aone, Chinatsu and Scott William Bennett. 1995. Evaluating automated and manual acquisition of anaphora resolution strategies. In *Proceedings of the Association of the Canadian Linguistics (ACL'95)*, pages 122–129.

Asher, Nicholas. 1993. *Referring to Abstract Objects in Discourse*, volume 50 of *Studies in Linguistics and Philosophy*. Kluwer Academic Publishers, Dordrecht.

Asher, Nicholas and Alex Lascarides. 2003. *Logics of Conversation*. Cambridge University Press.

Bagga, Amit and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Workshop on Linguistic Coreference at 1st International Conference on Language Resources and Evaluation (COLING-ACL'98)*.

Baldwin, Breck. 1997. Cogniac: High precision coreference with limited knowledge and linguistic resources. In *Proceedings of the ACL'97/EACL'97 workshop on Operational factors in practical, robust anaphora resolution*.

Barbu, Catalina and Ruslan Mitkov. 2001. Evaluation tool for rule-based anaphora resolution methods. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*.

Brants, Sabine, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The tiger treebank. In *Proceedings of the Workshop on Treebanks and Linguistic Theories*, Sozopol.

Brennan, Susan E., Marilyn W. Friedman, and Carl J. Pollard. 1987. A centering approach to pronouns. In *Proceedings of the 25th Annual Meeting of the ACL*, pages 155–162, Standford.

Bußmann, Hadumod. 2002. *Lexikon der Sprachwissenschaft*. Alfred Kröner Verlag, Stuttgart.

Byron, Donna K. 2001. The uncommon denominator: A proposal for consistent reporting of pronoun resolution results. *Computational Linguistics Special Issue on Computational Anaphora Resolution*, 27(4):569–577.

Byron, Donna K. and Joel R. Tetreault. 1999. A flexible architecture for reference resolution. In *Proceedings of the Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL-99)*.

Carletta, Jean. 1996. Squibs and discussions – assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.

Čermák, František. 1997. *Jazyk a jazykověda*. Pražská imaginace, Praha.

Černý, Jiří. 1996. *Dějiny lingvistiky*. Votobia, Olomouc.

Chomsky, Noam. 1970. Remarks on nominalization. In Roderick A. Jacobs and Peter S. Rosenbaum, editors, *Readings in English Transformational Grammar*. Ginn and Company, Waltham, Massachusetts, pages 184–221.

Connolly, Dennis, John D. Burger, and David S. Day. 1994. A machine learning approach to anaphoric reference. In *Proceedings of the International Conference on New Methods in Language Processing (NeMLaP)*. ACL.

Cristea, Dan, Nancy Ide, and Laurent Romary. 1998. Veins theory: A model of global discourse cohesion and coherence. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 281–285, Montreal, Canada. Association for Computational Linguistics.

Cristea, Dan, Oana-Diana Postolache, Gabriela-Eugenia Dima, and Catalina Barbu. 2002. AR-engine – a framework for unrestricted co-reference resolution. In *Proceedings of The Third International Conference on Language Resources and Evaluation, LREC-2002*, Las Palmas, Spain.

Garnham, Alan. 2001. *Mental Models and the Interpretation of Anaphora*. Essays in Cognitive Psychology. Psychology Press, Hove, East Sussex.

Grice, Herbert Paul. 1975. Logic and conversation. In Peter Cole and J.L. Morgan, editors, *Syntax and semantics: Speech acts*. Academic, New York, pages 41–58.

Grosz, Barbara and Candace Sidner. 1986. Attention, intention, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

Grosz, Barbara J., Aravind K. Joshi, and Scott Weinstein. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.

Hajič, Jan et al. 2005. The prague dependency treebank 2.0, `http://ufal.mff.cuni.cz/pdt2.0/`. Developed at the Institute of Formal and Applied Linguistics, Charles University in Prague. To be released by Linguistic Data Consortium in 2006.

Hajičová, Eva. 1987. Focusing – a meeting point of linguistics and artificial intelligence. In P. Jorrand and V. Sgurev, editors, *Artificial Intelligence Vol. II: Methodology, Systems, Applications*. Elsevier Science Publishers, Amsterdam, pages 311–321.

Hajičová, Eva, Tomáš Hoskovec, and Petr Sgall. 1995. Discourse modelling based on hierarchy of salience. *The Prague Bulletin of Mathematical Linguistics*, (64):5–24.

Hajičová, Eva, Petr Kuboň, and Vladislav Kuboň. 1990. Hierarchy of salience and discourse analysis and production. In *Proceedings of Coling'90*, Helsinki.

Hajičová, Eva, Jarmila Panevová, and Petr Sgall. 1985. Coreference in the grammar and in the text. *The Prague Bulletin of Mathematical Linguistics*, (44):3–22.

Hajičová, Eva, Jarmila Panevová, and Petr Sgall. 1999. *Manuál pro tektogramatické značkování*. Number TR-1999-07. Prague, Czech Republic.

Halliday, M.A.K. and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.

Hirst, Graeme. 1981. Discourse-oriented anaphora resolution in natural language understanding: A review. *American Journal of Computational Linguistics*, 7(2):85–98.

Hobbs, Jerry R. 1978. Resolving pronoun references. In Barbara J. Grosz, Karen Spärck-Jones, and Bonnie Lynn Webber, editors, *Readings in Natural Language Processing*. Morgan Kaufmann Publishers, Los Altos, pages 339–352.

Hobbs, Jerry R. 1979. Coherence and coreference. *Cognitive Science*, 3(1):67–90.

Horák, Aleš. 2001. *The Normal Translation Algorithm in Transparent Intensional Logic for Czech*. Ph.D. thesis, Masaryk University Brno, Faculty of Informatics.

Jackendoff, Ray. 1977. *X-Bar Syntax: A Study of Phrase Structure*. MIT Press, Cambridge, Massachusetts.

Joshi, Aravind K. and Steve Kuhn. 1979. Centered logic: The role of entity centered sentence representation in natural language inferencing. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 435–439, Tokyo.

Joshi, Aravind K. and Scott Weinstein. 1981. Control of inference: Role of some aspects of discourse structure – centering. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 385–387, Vancouver, B.C.

Jurafsky, Daniel and James H. Martin. 2000. *Speech and Language Processing (An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition)*. Prentice Hall, New Jersey.

Kehler, Andrew. 1997. Current theories of centering for pronoun interpretation: A critical evaluation. *Computational Linguistics*, 23(3):467–475.

Kruijff-Korbayová, Ivana and Eva Hajičová. 1997. Topics and centers: A comparison of the salience-based approach and the centering theory. *Prague Bulletin of Mathematical Linguistics*, 67:25–50.

Kučová, Lucie, Veronika Kolářová, Zdeněk Žabokrtský, Petr Pajas, and Oliver Čulo. 2003. Anotování koreference v pražském závislostním korpusu. Technical report, Charles University, Prague.

Kučová, Lucie and Zdenek Žabokrtský. 2005. Anaphora in czech: Large data and experiments with automatic anaphora resolution. In Václav Matoušek, Pavel Mautner, and Tomáš Pavelka, editors, *Text, Speech and Dialogue, 8th International Conference, TSD 2005, Karlovy Vary, Czech Republic, September 12-15, 2005, Proceedings*, Lecture Notes in Computer Science, pages 93–98. Springer.

Lappin, Shalom and Herbert J. Leass. 1994. An algorithm for pronominal anaphora resolution. *Computatinal Linguistics*, 20(4):535–561.

Lasnik, Howard, editor. 1989. *Essays on Anaphora, Studies in Natural Language and Linguistic Theory*. Kluwer Academic Publishers, Dordrecht.

Levinson, Stephen C. 2000. *Pragmatik*. Max Niemeyer Verlag, Tübingen, third edition. Neu übersetzt von Martina Wiese.

Lindroos, Hilkka and František Čermák. 1982. *Stručná mluvnice finštiny*. Univerzita Karlova, Praha.

Mann, William C. and Sandra A. Thompson. 1988. Rhetorical structure theory: Towards a functional theory of text organization. *Text*, 8(3):243–281.

Mathesius, Vilém. 1966. *Řeč a sloh*. Československý spisovatel, Praha.

Mitchell, Tom. 1997. *Machine Learning*. McGraw-Hill, New York.

Mitkov, Ruslan. 1999. Anaphora resolution: The state of the art. Technical report, University of Wolverhampton.

Mitkov, Ruslan. 2001. Towards a more consistent and comprehensive evaluation of anaphora resolution algorithms and systems. *Applied Artificial Intelligence*, 15(3):253–276.

Mitkov, Ruslan. 2002. *Anaphora Resolution*. Longman, London.

Mitkov, Ruslan. 2003. Anaphora resolution. In Ruslan Mitkov, editor, *The Oxford Handbook of Computational Linguistics*. Oxford University Press, Oxford, pages 266–283.

Mitkov, Ruslan, Shalom Lappin, and Branimir Boguraev. 2001. Introduction to the special issue on computational anaphora resolution. *Computational Linguistics Special Issue on Computational Anaphora Resolution*, 27(4):473–477.

Moore, Adrian William. 1993. *Meaning and Reference*. Oxford University Press, New York.

Moser, Megan and Johanna D. Moore. 1996. Toward a synthesis of two accounts of discourse structure. *Computational Linguistics*, 22(3):409–420.

1996. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, San Mateo, CA. Morgan Kaufmann.

1998. *Proceedings of the Seventh Message Understanding Conference (MUC-7)*, San Mateo, CA. Morgan Kaufmann.

Müller, Christoph and Michael Strube. 2002. An API for discourse-level access to xml-encoded corpora. *Proceedings of the 3rd International Conference on Language Resources and Evaluation*.

Müller, Christoph and Michael Strube. 2003. Multi-level annotation in mmax. *Proceedings of the 4th SIGdial Workshop on Discourse and Dialogue*.

Poe, Edgar Allan. 1994. The cask of amontillado. In *The Complete Illustrated Stories and Poems*. Chancellor Press, Auckland, pages 115–121.

Russell, Stuart J. and Peter Norvig. 2003. *Artificial Intelligence, A Modern Approach*. Prentice Hall, New Jersey.

Sgall, Petr, Eva Buráňová, and Eva Hajičová. 1980. *Aktuální členění věty v češtině*. Academia, Praha.

Sidner, Candace L. 1979. *Towards a Computational Theory of Definite Anaphora Comprehension in English Discourse*. Ph.D. thesis, Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

Sidner, Candace L. 1983. Focusing in the comprehension of definite anaphora. In Michael Brady, editor, *Computational Models of Discourse*. MIT Press, Cambridge, pages 267–330.

Steinberg, Danny D. 1993. *An Introduction to Psycholinguistics*. Learning About Language. Longman, London.

Strube, Michael. 1998. Never look back: An alternative to centering. In *Proceedings of the 17th international conference on Computational Computational Linguistics, Volume 2*, pages 1251–1257, Montreal, Quebec, Canada.

Strube, Michael, Stefan Rapp, and Christoph Müller. 2002. The influence of minimum edit distance on reference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 312–319, Philadelphia. Association for Computational Linguistics.

2004. The TiGer Project, `http://www.ims.uni-stuttgart.de/projekte/TIGER/`.

Toman, Jindřich. 1991. Anaphors in binary trees: an analysis of czech reflexives. In Jan Koster and Eric Reuland, editors, *Long-distance anaphora*. Cambridge University Press, Cambridge, pages 151–170.

Vieira, Renata and Massimo Poesio. 2001. An empirically based system for processing definite descriptions. *Computational Linguistics*, 26(4):539–593.

Vilain, Marc, John Burger, John Aberdeen, Dennis Connoly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of the Sixth Message Understanding Conference (MUC-6)*.

Webber, Bonnie Lynn. 1983. So what can we talk about now? In Michael Brady, editor, *Computational Models of Discourse*. MIT Press, Cambridge, pages 331–372.

Werich, Jan and Jiří Brdečka. 1951. Císařův pekař a pekařův císař. Československý státní film.

Winograd, Terry. 1972. *Understanding Natural Language*. Academic Press, New York.