Master Thesis

Creating a Semantic Wiki using a Link Grammar-based Algorithm for Relation Extraction

Alan Akbik
akbik@mi.fu-berlin.de

Freie Universität Berlin
Institut für Informatik
AG Datenbanken

Betreuer : Prof. Dr. H. Schweppe

Berlin, March 31, 2009

Abstract

This thesis examines a linguistic approach to information extraction. It analyzes the theory that there are universally valid grammatical patterns for explicitly stated semantics in sentences in plain text. An algorithm aware of these patterns is in theory capable of extracting arbitrary semantics from grammatically correct sentences. In this thesis, the deep grammatical formalism used to describe the grammaticality of a sentence is link grammar, in which a total of 43 such grammatical patterns are identified of which the most common are illustrated. Semantic relations are sought in the form of subject-predicate-object triplets. An algorithm for information extraction is implemented and used with the plain text of the English Wikipedia with the goal of generating a semantically annotated wiki. Using this approach, a total of 2.64 million relation triplets are generated from the English Wikipedia corpus that use 312,744 distinct relation types. Difficulties encountered during the implementation of the approach are discussed. The generated results are analyzed and evaluated based on different considerations, yielding precision values of either 0.62, 0.82 or 0.91 and recall values of either 0.075 or 0.2. Two main problems are identified for the approach used in this thesis. First, there are difficulties in modeling the context of information within the relation triplet model which hinders the extraction of various types of semantics. Second, certain grammatical information is not reflected within link grammar, but is proven to be necessary in order to identify well working grammatical patterns. Information on the valency of verbs is identified as the missing grammatical information and discussed.

Zusammenfassung

Diese Arbeit befasst sich mit einem linguistischen Ansatz zur Informationsextraktion. Die Theorie universell gültiger grammatikalischer Muster für in natürlichsprachlichen Sätzen explizit ausgedrückte Semantik wird untersucht. Ein Algorithmus kann unter der Benutzung dieser Muster laut dieser Theorie jegliche Art von Semantik aus grammatikalisch korrekten Sätzen extrahieren. In dieser Arbeit wird der Link Grammar Formalismus verwendet, um die Grammatik eines Satzes darzustellen. Innerhalb des Formalismus werden 43 solcher grammatikalischer Muster gefunden, von denen die am häufigsten beobachteten Muster beschrieben werden. Semantik wird in der Form von Subjekt-Prädikat-Objekt Tripeln dargestellt. Ein Algorithmus für die Informationsextraktion wird auf dieser Basis implementiert und mit dem Text der Englischen Wikipedia verwendet. Das Ziel ist das automatische Generieren eines semantisch annotierten Wikis. Mit diesem Ansatz werden 2,64 Millionen Relationen aus der Englischen Wikipedia generiert, welche 312,744 verschiedene Relationstypen (Prädikate) verwenden. Schwierigkeiten bei der Implementierung des Ansatzes werden aufgezeigt. Die Ergebnisse werden analysiert und unter verschiedenen Berücksichtigungen bewertet, wodurch sich Precision-Werte von 0.62, 0.82 oder 0.91 und Recall-Werte von 0.075 oder 0.2 berechnen. Zwei Hauptprobleme des verwendeten Ansatzes werden ermittelt. Ein Hauptproblem sind Schwierigkeiten den Kontext von Informationen innerhalb von Relationstripeln zu modellieren, was die Extraktion verschiedener Arten von Semantik behindert. Das zweite Hauptproblem ist die Unvollständigkeit der grammatikalischen Information gegeben durch den Link Grammar Formalismus, was das Ermitteln gültiger grammatikalischer Muster beeinträchtigt. Das Fehlen und die Wichtigkeit der Information über die Verbvalenz wird aufgezeigt und diskutiert.

Eidesstattliche Erklärung

Anhand dieser eidesstattlichen Erklärung versichere ich die Master Thesis:

"Creating a Semantic Wiki using a Link Grammar-based Algorithm for Relation Extraction"

im Fachbereich Informatik der Freien Universität Berlin, Berlin 2009, vollständig eigens verfasst und keine anderen Hilfsmittel, als die angegebenen Quellen verwendet zu haben.

Berlin, 31. März, 2009

Alan Akbik

Acknowledgements

I would like to thank Jürgen Broß for his round-the-clock support and his valuable comments. Furthermore I would like to thank Marc Berendes, Adrian de Rivero, Madlen Jähnig and Dorothy Schaefer-Akbik.

But the boy clutched his father's sword, crying,

"So long as men live, there are crimes!"

The man's eyes filled with wonder.

"No, child," he said. "Only so long as men are deceived."

– From *The Darkness That Comes Before*
by Scott Bakker

Contents

Chapter 1

Introduction

This thesis examines a linguistic approach to information extraction from sentences in plain text. The main theory is that certain grammatical structures exist which are universally valid and therefore allow for the extraction of arbitrary semantics. The algorithm proposed and implemented in this thesis is therefore intended to be able to extract all explicitly stated facts within individual sentences in the form of *semantic relations*. Semantics are expressed in a subject-predicate-object triplet form, analogous to statements in RDF [23]. Subjects and objects represent concepts defined by the meaning of their term and are subsequently referred to as *entities* (resources in RDF). *Predicates* are used to describe the nature of the relationship between two entities with a sequence of words.

The method is implemented and applied to the entire corpus of the English Wikipedia in order to generate a knowledge base from its contents. Stated goal of the thesis is to use the extracted relation triplets to produce a fully usable semantic wiki. This serves as use case for the method and stands at the center of this thesis, allowing both the determination of quantitative results as well as qualitative insights about the theory of universally valid grammatical patterns. All challenges during the implementation of the use case are documented and discussed. The thesis concludes with producing a semantic wiki consisting of over 2.6 million relation triplets. The results are analyzed with regards to measures such as precision and recall, but also concerning usability and expressiveness of the generated knowledge base. Strengths and weaknesses of the algorithm and the original theory are discussed.

This chapter introduces the thesis and the method for relation extraction. In Section 1.1, both a general motivation for information extraction as well as motivation for the specific use case of converting Wikipedia into a semantic wiki are illustrated. In 1.2, the theory of semantifying Wikipedia as proposed in [22] is illustrated and the information extraction approach used in this thesis specified. Limitations and challenges are discussed. The overall goal of the thesis is stated in 1.3. In 1.4, an outline of this thesis is given.

1.1 Motivation

This section introduces the field of information extraction and illustrates how it can be used to enable a wealth of applications. As a specific example of a powerful application that can benefit from information extraction, the case for a semantic wiki is illustrated.

1.1.1 Information Extraction

While there exists an abundance of natural language text in digital form, the information it contains is primarily intended only for human readers and due to its unstructured form not immediately suitable for computerized automatic analysis. The field of information extraction is a form of text mining which concerns itself with making the information stated in natural language text available to machine-processing by extracting information such as entities, relationships between entities and attributes describing entities. It may therefore be seen as a restricted form of full natural language understanding attempted by a computer and is an important research area both in text mining as well as natural language processing [42, 33].

By making the wealth of information present in natural language text available for machine-processing, a wide variety of applications are enabled of which a non-exhaustive subset will be introduced here. In [33], applications are divided into several categories. One is **enterprise applications** such as *customer care* in which many forms of unstructured data are collected from customer interaction and integrated with the enterprise's databases. This information is intended to maximize the efficacy of customer-oriented enterprises. Extraction problems include the identification of product attribute value pairs from textual product descriptions [14], the acquisition of repair records from insurance claim forms [32] or even the identification of customer moods from conversation transcripts [16]. Another important category are **web oriented applications** which include the *intelligent placement of ads* on web-sites [7] or the automatic comparison of products and prices to enable *comparison shopping* [10]. A general example for the identification of entities in web oriented applications is the automatic hyperlinking of news articles (as well as blogs or websites) to background information on named entities.[1] There are also numerous **scientific applications**, such as combining and making available information from a multitude of resources like scientific papers. Examples for this can be found in the field of bio-informatics, in which proteins, genes and their interactions are extracted from paper repositories [30, 8]. Computerlinguistic challenges such as automatic translation or the resolution of coreferences are also aided by information extraction [29].

Information extraction can be used to automatically generate knowledge bases from corpora of plain text. Knowledge bases enable technologies such as *question-answering systems*, which allow structured ("semantic") queries to yield question-specific answers instead of the set of documents returned by regular keyword queries. The generation of a knowledge base from a large and diverse corpus, as well as a focus on the application of semantic querying, is the use case of information extraction as implemented in this thesis. The motivation

[1] www.linkedfacts.com

for the project is illustrated in the ensuing section.

1.1.2 Semantic Wikipedia

The Wikipedia Online Encyclopedia[2] is a rapidly growing and hugely successful example of a wiki. To date, there are more than 2 million articles in the English Wikipedia alone. Among main design principles contributing to the success of Wikipedia are *Openness* – giving any reader the possibility of changing a page (article) – and the possibility of pages to cite other pages [9]. These so called *page links* provide interconnectivity between the pages and enable the user to easily move about related pages in Wikipedia. According to Wikipedia's policy of being very simple to use, they can be effortlessly set by even inexperienced page authors. Above all, this simplicity is said to be one of the main factors for Wikipedia's strong growth[3] over the past years.

The growing scale of information in the Wikipedia database raises questions of how to organize this data efficiently as to allow more specific querying of its content. In the current version of Wikipedia a full-text keyword search over all articles is used as a means of information retrieval. To illustrate the shortcomings of purely keyword-based information retrieval, one may consider detailed search requests, such as 'compile a list of all German beers with a higher percentage of alcohol than 8%'. It is presumable that the information needed to adequately answer this question is to be found somewhere in the Wikipedia corpus. However, this information cannot be found in one single article, but rather is distributed among a range of articles. A user striving to get this information would have to make a number of keyword queries like 'strong German beer' or 'potent beer Germany' and then read all returned pages to manually compile such a list.

The problem here is that no underlying model of knowledge exists which formally describes the information stored in a regular wiki [29]. The content of Wikipedia's articles is designed to be human-readable and as such not easily accessible to machine-processing. One solution to overcome this lack is the introduction of a layer of machine-readable metadata, which describes contents of pages and their relations to other pages. Because the principle is borrowed from the Semantic Web project [4], a wiki with such a layer of information is called semantic wiki [29]. In a semantic wiki a formal language is typically used to annotate metadata, making it possible for machines to process it into a relational or an entity-relationship database. The more semantic metadata is annotated, the more complex the underlying model of knowledge becomes.

Because of its scale and its diverse content, a machine-readable model of the knowledge stored in the English Wikipedia would arguably be the largest ontology in existence, providing general "world knowledge" from which a multitude of technologies such as those mentioned in 1.1.1 can benefit. In particular with regard to the results of this thesis, applications such as semantic querying can be implemented over this knowledge base, allowing the wiki to function as a question-answering machine capable of providing exact answers to structured queries like the one mentioned above. Making the content of the English Wikipedia available for machine-processing is a major motivation of this thesis. The following section illustrates the approach used in this thesis to achieve this goal.

[2] www.wikipedia.org

[3] http://en.wikipedia.org/wiki/Wikipedia:Modelling_Wikipedia's_growth, March 2009

1.2 Problem Statement

This section introduces the process of generating a semantic wiki out of the English Wikipedia as proposed in [22] and illustrates which role in the process the algorithm implemented in this thesis plays. Limitations of the algorithm and the use-case are illustrated.

1.2.1 Semantifying Wikipedia

This section describes the method laid out in [22] by Krötzsch et al. of how to convert Wikipedia into a semantic wiki. The method builds upon the existing structure of Wikipedia and takes some inspiration from the Semantic Web project. A semantic relation as defined in RDF may be expressed as triplets of URIs (globally unique identifiers) [23], where the first URIs defines a *subject*, the second a *predicate* and the third an *object*. Subject and object are both representations of concepts (referred to as *entities*), while the predicate is used to describe the nature and logic of the semantic relation between two entities.

Wikipedia's existing structure may be seen in an analogous way. Each Wikipedia page describes a unique concept or theme. Each page has a unique page title, which can be used as its URI. Page titles that may refer to multiple concepts are disambiguated according to Wikipedia's conventions (discussed in 2.2.1), which ensures that each concept for which a Wikipedia page exists has an URI. Because of this, Wikipedia pages may be used as subjects or objects, meaning that 2 of the 3 URIs needed to form an RDF triplet are already present in Wikipedia. A direct equivalent of predicates does not exist, making it impossible to define the nature of the relationship of two entities in a machine-readable way. Wikipedia's page links however indicate that some form of relation exists between two linked pages. For example, the pages "Germany" and "Berlin" in the English Wikipedia are connected by numerous page links, which indicates that both pages have a semantic relationship. Without predicate information, the nature of this relationship cannot easily be determined (see Figure 1.1).

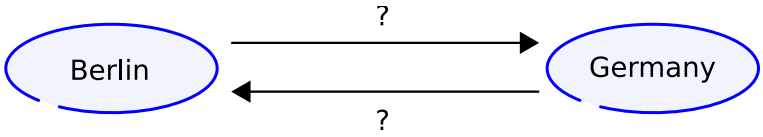

Figure 1.1: Page links between two entities. The nature of the relation is unknown.

Information to answer semantic search queries (such as *What is the capital of Germany?* or *In which city is Germany governed?*) cannot be extracted from this data structure, which is insufficient to serve as the formal model of logic mentioned in 1.1.2. In [22], a method is proposed which expands Wikipedia's data structure in order to make it usable for this purpose. In the paper, Krötzsch et al. propose to add predicate information to page links in order to complete the analogy to subject-predicate-object triplets in RDF.

To this end, they define *semantic page links* (referred to as typed links in [22]) as an extension of regular page links. Like regular page links they point to other pages, but additionally are annotated with a sequence of words describing the

nature of the relationship between the page in which the page link is set and the page to which the page link points. Therefore, just like page links they are unidirectional. By annotating existing page links in the Wikipedia corpus with predicate information, semantic page links are generated. These in turn are processed as subject-predicate-object triplets, thereby describing a formal model of knowledge.

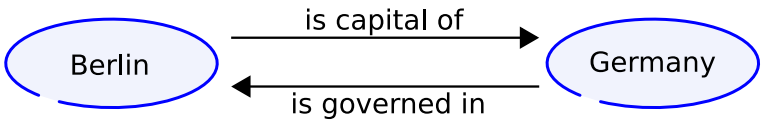

Figure 1.2: Semantic page links between articles.

In Figure 1.2 the analogy between two pages linked by semantic page links and subject-predicate-object triplets in RDF becomes apparent. The page in which the semantic page link is set is the subject of the relation, the annotation functions as the predicate and the page, to which the semantic page link points to is the object.

Technologies for the realization of Krötzsch et al.'s proposal are already in place. The Semantic MediaWiki project [21] has released an extension to the current Wikipedia software, extending its markup language to allow for the specification of semantic page links. According to the Wikipedia design principles of Openness and Simplicity the additional syntax is intended to be easy to use even for inexperienced authors.

However, manually annotating Wikipedia's page links may not be feasible due to the sheer vastness of their number. Also, lack of knowledge of the concepts of the Semantic Web and initial difficulties with the extended syntax may present hurdles for the average Wikipedia author. While semantic wikis offer the platform and functionality for the introduction of semantic page links, this possibility must be accepted and used by the Wikipedia user community. The problem is that the benefits of semantic annotation are only convincing if a very large number of semantic page links exist. If too many questions posed using semantic queries are insufficiently or not at all answered, it is doubtful that the online community will start the rigorous process of manually annotation tens of millions of semantic page links. At the same time, if the online community does not start using semantic page links, the critical mass needed to demonstrate semantic functionality in a convincing way cannot be generated.

In [22] Krötsch et al. state the following in regard to this problem:

> "Now in order to promote the new editing options in Wikipedia, it will be helpful to start an initiative for typing the links in some particular subdomain of Wikipedia. This task can best be solved in cooperation with a dedicated Wikiproject that has a good overview of some limited topic. The domain experts in this project can then develop a first hierarchy of link types and incorporate these types into part of the articles within this project. The generated OWL-output can then be used within offline tools to demonstrate the util-

> ity of the effort. Domains that already offer a rigidly used template, like countries, may be early adopters due to the efficiency gained by combining typed links and templates [...]"

The author of this thesis proposes a different approach to overcome this deadlock. Semantic page links can be automatically generated from plain text using the method presented in this thesis. By generating a critical mass of semantic page links, the author hopes to transform the English Wikipedia into a sufficiently annotated semantic wiki, which may serve to make the benefits and principles of a semantic wiki *visible* to the online community. Once such visibility is achieved, users may start adding semantic page links to their articles in order to make their articles more accessible for semantic queries. Since a high number of wrongly annotated semantic page links would greatly reduce the effectiveness of semantic queries, precision rather than recall shall present a priority in this work. The reasoning behind this priority is that incomplete answers are less likely to be badly perceived as wrong ones.

This thesis focuses on the problem of generating subject-predicate-object triplets from plain text in order to automatically annotate semantic page links. It is the opinion of the author that computerlinguistic analysis based on the link grammar formalism [37], a formalism for the representation of grammar in natural languages, can be used as a method for the extraction of semantics from text. The basic idea of the proposed algorithm is sketched in the ensuing section.

1.2.2 Extraction of Semantics

This section illustrates the theory behind the approach proposed and implemented in this thesis for relation extraction from plain text.

As previously noted, the main theory of this work is that certain grammatical structures exist which are universally valid and therefore allow for the extraction of arbitrary semantics. In order to find a set of grammatical patterns that express relations between entities, a deep linguistic analysis of sentences is performed using the information given by a dependency-style deep grammatical formalism called *link grammar* [37]. In this formalism, links are drawn above grammatically dependent terms within a sentence. These links are labeled according to the nature of the grammatical relationship of two terms. If not directly connected, one of the properties of the formalism (called *connectivity*) ensures that all terms of the sentence are at least indirectly connected via a number of intermediary terms. A path between two words of a sentence is called a *linkpath*. The source and target of such a link path are denoted as *start term* and *end term* respectively. The sum of all links describes the grammar of the entire sentence and is referred to as *linkage*.

Within this formalism it can be argued that if a direct relationship between two terms is expressed by linking them together, then a *chain of connected terms* describes the relationship between a start and stop term. This can be seen from two perspectives. From a grammatical point of view it can be argued that the sequence of link labels (the linkpath) from start to stop term describes the grammatical relationship between these terms. The theory of this thesis is that certain grammatical relationships between two terms also imply a se-

mantic relationship. Therefore, if a linkpath fulfills the necessary grammatical criteria, the sequence of words interlinked on that path (subsequently denoted as *wordpath*) can be seen as describing the semantic relationship between two terms.

The formalism will be illustrated with the example sentence "*Essen is a beautiful city in the Ruhr Area*". Because of the ambiguous nature of the English language (and most if not all natural languages), each sentence may have a number of different linkages. The linkage in Figure 1.3 is one possible representation of the example sentence. The example demonstrates how words are connected if a grammatical relationship exists. The determiners "a" and "the" in the sentence are connected to their nouns, the adjective "beautiful" is connected to the noun it modifies, the verb "is" is connected both to its subject and its object and the preposition "in" connects together two related nouns.

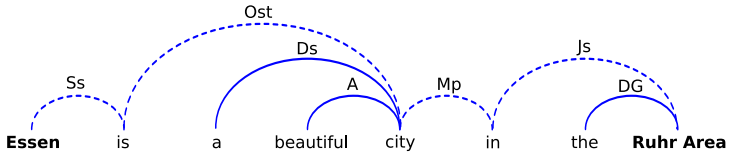

Figure 1.3: Linkage for an example sentence. Terms are linked according to their grammatical relationship. One path through the linkage is highlighted with dotted lines.

The first step of the approach is to identify terms in the sentence that can be assigned URIs, a challenge referred to as *identification of entities*. This is necessary to make the terms eligible to serve as subjects and objects in an RDF triplet. Assume that in this sentence some terms such as "Essen" and "Ruhr Area" can be identified as entities. To determine a predicate for a relation between the entities *Essen* and *Ruhr Area*, the information given in the linkage can be used. This is done by "tracing" the links connecting both terms. "Essen" is connected to the term "is" with the **Ss** link, which means that "Essen" is the subject (singular) of the verb "is". The verb in turn is connected to the term "city" via the **Ost** link, signifying that "city" is the direct object (singular) of the verb. "City" is connected to the preposition "in" with the **Mp** link, and the preposition is connected to the indirect object "Ruhr Area" with the label **Js**. This path is highlighted in Figure 1.4. A reference for all link labels used in this thesis is given in Appendix B.2.

For this example the *wordpath* is **IsCityIn** (see Figure 1.4). In this thesis, wordpaths are written by capitalizing the first letters of the individual words and joining them together to form one word. This notation is commonly found for relations in Semantic Web projects. The idea of the thesis is to use said wordpath as predicate for a semantic relation between start and end term of the path. In the case of the above example, a semantic relation between the entities *Essen* and *Ruhr Area* can be extracted simply by following the wordpath in the linkage, which yields the following relation:

IsCityIn(Essen, Ruhr Area)

A relation extracted in this manner has a certain grammaticality as given by the linkage in which it was found, described by its linkpath. Linkpaths are

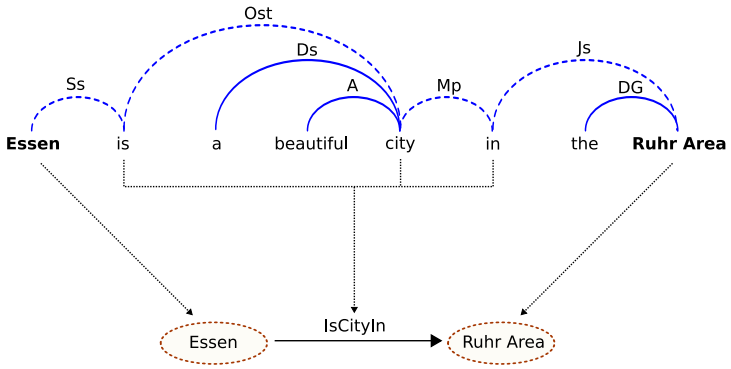

Figure 1.4: Relation extraction from an example sentence.

written in this thesis by separating each link label by a comma, and encompassing the labels by cursive brackets. For the above example, the linkpath is the following:

{Ss, Ost, Mp, Js}

Important to note is that a valid relation has been generated without considering the lexical meaning of terms in a sentence. The only information that is used in the relation extraction process is the linkpath, i.e. the grammatical relationships between words in a sentence. If a chain of terms with the same linkpath is found in the linkage of a different sentence, a valid semantic relation can be generated using the same method. To grasp the idea, consider exchanging one or multiple terms in a sentence with terms that behave equally within the formalism. E.g. replacing "city" with "place", yielding the sentence "*Essen is a beautiful place in the Ruhr Area*", from which using the same linkpath the valid relation IsPlaceIn(Essen, Ruhr Area) is extracted. Thanks to the application of a deep linguistic parser, sentences may also differ more strongly (i.e. inserted relative phrases, additional modifiers). The original theory states that as long as two terms are connected with a valid linkpath, the algorithm is able to find a suitable predicate to express their semantic relationship.

However, a great share of observable linkpaths is not useful with regard to the preceding considerations. To illustrate this, consider two arbitrary terms in the example sentence, such as "beautiful" and "Ruhr Area". The wordpath connecting "beautiful" to "Ruhr Area" is CityIn. Refer to Figure 1.5 for an illustration of this. A relation built using this information would be CityIn(beautiful, Ruhr Area), which is nonsensical and therefore false. The problem here is not that the original theory is incorrect, but rather that the sentence does not explicitly state any information which connects both terms in question. The adjective "beautiful" modifies another noun, but not "Ruhr Area". Note that because of this even a human annotator would have difficulties finding a predicate to connect "beautiful" to "Ruhr Area" in a way that expresses the semantics of the sentence.

This means that both valid and false relations are extracted from a given sentence using this method. An important part of the algorithm therefore is the identification of grammatical factors which enable the extraction of a valid

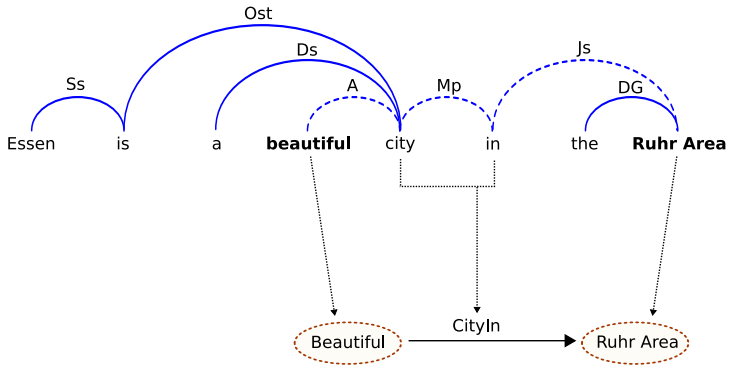

Figure 1.5: Example for the extraction of a false relation. The predicate for the relation which points from *beautiful* to *Ruhr Area* is CityIs which is false. The linkpath for the relation is {A, Mp, Js}.

relation and allow the algorithm to dismiss all others. These factors include the exact grammaticality as given by the linkpath, which due to its nature comprehensively describes the grammatical relation of two words in a given sentence. It must also be considered that additional criteria might be needed in order to maximize the ability of the algorithm to determine whether a relation is valid or false. Finding valid linkpaths and other criteria is a challenge lying at the heart of the successful implementation of the algorithm and necessary in order to prove the initial theory. Limitations to the algorithm and challenges for its realization are illustrated in the ensuing sections.

The approach introduced in this section has been dubbed *Wanderlust*, which is a German loanword designating an impulse or a strong desire to wander, because it is centered around following ("wandering along") paths of links and words.

1.2.3 Doppeldenk Limitations

Certain limitations exist regarding the quality of relation extraction with the proposed algorithm, which are derived from its methodology and properties of the Link Grammar Parser [37], the deep parser used in this thesis. The considerations presented here are a frame for the results that can be generated using Wanderlust. Since the algorithm analyzes text on a sentence-by-sentence level and no logical verification of extracted relations is performed, Wanderlust will add all knowledge explicitly stated in sentences to its model of knowledge, even if contradicting statements are found within the same text corpus. Because of the algorithm's signature ability to "simultaneously accept" two contradictory beliefs, its limitations have been dubbed *Doppeldenk*[4].

Context Sensitivity

One of the main limitations of Wanderlust is that all semantics extracted with the proposed algorithm and the Link Grammar Parser are vulnerable to errors made when a relation triplet is extracted without the context in which it is

[4]German for *doublethink*, see the novel *Nineteen Eighty-Four* by George Orwell

stated. A relation triplet may only be true in a certain context. Outside the context, such relations are false. This limitation has a multitude of causes. One basic problem is the subject-predicate-object triplet structure of semantic relations in RDF, which is generally not suited to include the context of statements, making is difficult to correctly express a range of semantics [18]. Another problem is that both Wanderlust and the Link Grammar Parser function at *sentence granularity*, meaning that no connection is made between multiple sentences of the same article.

As consequence, semantics can only be extracted if stated within *one* sentence. Any information expressed spanning several sentences cannot be extracted, leading to recall loss. This is especially problematic in cases where one sentence is put into context by another. Consider for example the following excerpt:

> "Independence Day (also known by its promotional abbreviation ID4) is a 1996 science fiction film about a hostile alien invasion of Earth. [...] Area 51 conceals a top secret facility housing a repaired attacker and three alien bodies recovered from Roswell in 1947. [...]"[5]

Because of the first sentence, a human reader will know that the article describes fictional content. The context for this excerpt is therefore "Storyline of the film 'Independence Day' (1996)". This information however is not present in all subsequent sentences. Relations extracted from the plot of the film are not recognized as dealing with fictional content and assumed to pertain to the entity defined by the Wikipedia page "Area_51", which covers a non-fictional military base. This poses a problem since a knowledge base built using such relations does not distinguish between fictional and non-fictional information, making it unsuitable to be used as a model of world knowledge.

This problem, however, is not limited to context information spanning sentences. Because of the inherent locality of the Link Grammar Parser (see 2.4), a lexical system, a relation may be extracted from one part of a sentence while dismissing information given in other parts of the sentence. An example for this can be found in the sentence "*In his dream, the sky was yellow*", in which a false relation pertaining to the color of the sky can be extracted if the overall context of the information given in the sentence is not identified. The context in this example is "In his dream". Even given such identification, the context cannot be included in the subject-predicate-object relation triplet structure used in this thesis.

This limitation has noticeable effects of the results of Wanderlust. Further discussion on context errors is provided in Section 4.2.2, their impact on the overall results of the thesis is analyzed in 5.1.3.

Predicate Diction

The second major limitation addresses the fact that all predicates generated with Wanderlust must be composed of words which are used in the sentence from which a relation is extracted. This has a number of consequences for the relation triplets the algorithm can find. The possible recall of the algorithm is

[5] taken from the English Wikipedia page http://en.wikipedia.org/wiki/Independence_Day-_(film), October 2008

reduced by this limitation, since it only allows for explicitly stated information to be found. Implicitly stated facts which a human reader can gather from a sentence cannot be extracted. Consider for example the sentence "*The Ruhr is a tributary of the Rhine*". Explicitly stated facts such as Is(Ruhr, Tributary) and IsTributaryOf(Ruhr, Rhine) can be found by Wanderlust, while implicit statements, such as Is(Rhine, River), cannot.

Other than recall, the limitation of predicate diction affects the usefulness of a knowledge base generated by Wanderlust. Because the algorithm will use terms within a sentence to generate predicates, a potentially large number of distinct relation types can be produced. This has both positive and negative aspects. On the plus side the use of many distinct relation types allows for the correct and unabstract modeling of knowledge, making it possible to fit relation triplets precisely to the information stated in a sentence. In a question-answering setting this means that a user can freely pose semantic queries without having to use a fixed set of predicates. On the other hand, processing this information in semantic applications becomes more difficult since it is more feasible to define a layer of logic for a knowledge base with a limited set of predicates. This gives rise to problems such as synonymous predicates (for example Killed(Apollo, Python) and Slew(Apollo, Python)) and predicates which imply one another (for example IsCityIn(Berlin, Germany) implying IsLocatedIn(Berlin, Germany)), which reduce the expressiveness of the generated knowledge base.

1.2.4 Challenges

This section lists the main challenges for the realization of this thesis pertaining to the implementation of Wanderlust, its application on the English Wikipedia corpus and the evaluation of the result.

Identification of Entities

As mentioned in 1.2.2, one of the prerequisites of Wanderlust is the necessity disambiguating and assigning URIs to terms in plain text. Such terms are also referred to as *entities* because they can be used as subjects and objects in relation triplets. Because Wanderlust is used with the English Wikipedia as corpus, the identification of entities as performed in this thesis uses Wikipedia pages as a means of disambiguation. The approach implemented to tackle this problem is specific to properties of Wikipedia and has been dubbed *Weitblick*[6]. It is introduced in 3.1.

Linkpath Validation

A central challenge to the thesis as illustrated in 1.2.2 is to enable Wanderlust to distinguish between valid and false relations (a problem referred to as *validation*) by identifying which linkpaths generally lead to valid relations and which do not. The challenge involved the creation of a large training corpus and exploratory data analysis in order to determine if and how linkpaths can be used for validation. Weaknesses in the approach needed to be identified and the use of additional features for validation discussed. Method and results of the data analysis are explained in Chapter 4.

[6]German for *farsightedness*, chosen because it is a precursor to Wanderlust

Scale

Deep linguistic analysis using the Link Grammar Parser is much more resource-consuming than using a shallow parser. A parse of the Wikipedia page "Berlin" which consists of 400 sentences on a dual-core 4300 with 1024 megabytes of RAM takes approximately 2 minutes if the maximum parse time per sentence is limited to one second. Considering that the English Wikipedia from October 2008 consists of more than 2.4 million pages, the issue of scale needed to be addressed by distributing the workload among a cluster of 50 computers. This was possible due to Wanderlust's nature of analyzing sentences independently. All extracted relations were saved in a single database.

Evaluation

Methods needed to be found which can be used to evaluate the results of the thesis. This is especially difficult because purely quantitative evaluation is not well suited for the evaluation of semantics. Qualitative elements such as how many extracted relations are valid or expedient are difficult to automate, making manual validation necessary. The evaluation and the overall results of the thesis are stated in Chapter 5.

1.3 Goal

This thesis has two principal goals: One is to examine the performance and usability of the Wanderlust algorithm, the other the generation of a semantic wiki. Both goals are interconnected. By attempting to produce a fully usable semantic wiki with a knowledge base generated from the English Wikipedia, Wanderlust is tested "in action" within a well defined use case scenario. By analyzing difficulties, observations and the results of the project, the strengths and weaknesses of Wanderlust with regard to relation extraction and the original theory can be determined. As a positive side effect, the semantic wiki knowledge base and other information generated in the course of the thesis may help to enable future work.

1.4 Thesis Outline

In Chapter 2, theoretical prerequisites such as Wikipedia, the Semantic MediaWiki platform, the link grammar formalism and related work are introduced. Chapter 3 explains in detail each step of the Wanderlust algorithm applied to the task of populating a semantic wiki, beginning with the method for entity detection. The problem of validation is discussed in Chapter 4, where a training corpus is annotated and used to derive a method for identifying valid relations. Sources of errors disrupting linkpath validation are named and discussed. Finally, in Chapter 5 the results are presented and analyzed. An overall conclusion with regards to the initial theory is reached and an outlook of future work given in Chapter 6.

Chapter 2

Background

This chapter introduces the theoretical foundations of different components used in this thesis and is therefore intended to illustrate the prerequisites needed to understand the Wanderlust algorithm and the generation of a semantic wiki. In the first section of this chapter, a number of related projects are introduced and their methodology compared to the approach used in this thesis. In Section 2.2, the English Wikipedia is discussed in its current form. Special mention of structural elements and conventions which aid the development of a semantic wiki on its basis are made. Section 2.3 draws upon this and introduces the Semantic MediaWiki extension, which adds elements of a semantic wiki to the current Wikipedia platform. The general idea as well as selected features are discussed. Finally, in 2.4, the link grammar formalism as the main computerlinguistic device used in this thesis is described and compared to other prevalent formalisms.

2.1 Related Work

This section discusses work related to this thesis. There are a variety of information extraction approaches that are used to acquire semantic relations from natural language text. A differentiation can be made between methods that address the whole Web as a corpus and techniques that focus on more restricted corpora such as Wikipedia. The former discourage deep grammatical analysis and instead focus on validating relations by mass assessment. The latter corpora typically provide a rich set of structured metadata which can be leveraged by extraction procedures. In the case of Wikipedia many articles are augmented with so called infoboxes, containing related attribute/value pairs. Articles may also be classified into the hierarchical category structure in Wikipedia. Using smaller corpora also allows for the use of a deep linguistic parser for the purpose of relation extraction as implemented in this thesis. In the following subsections, examples of related work for each of the above mentioned categories are given and compared to Wanderlust.

2.1.1 Statistical Assessment

A main challenge to extracting information from the Web is its inherent heterogeneity and large scale, thus hindering approaches utilizing deep linguistic analysis. Early systems are [1] and [5] which employ a pattern matching approach to derive arbitrary binary relations from web pages. Since both define patterns only over shallow linguistic information, they are exposed to even small variations in the linguistic expression of a relation. Like the majority of systems in that chain of work, they make use of the inherent information redundancy present in the Web which allows for statistical assessment of extracted relations.

KnowItAll [12], a more recent system, uses a small set of domain independent extraction patterns and assesses extractions by utilizing the counts provided by search engine hits to compute the relatedness of terms in a particular extraction. *TextRunner* [3] automatically trains extractors using a deep grammatical parser, which in turn depend only on POS-tag information and a noun phrase chunker. In both systems, no deep grammatical parser is applied during the relation acquisition process. Because no disambiguation of entities is performed, the result in both systems is a very large set of relation triplets with probability values for individual relations based on statistical assessment. The systems have been evaluated for individual relation types and different confidence measures, resulting in a number of different precision values. In [3], both systems are compared by randomly selecting 10 relation types that appear at least in 1,000 relations and comparing their precision. They found 11,476 correct relations and an average error rate of 12% for TextRunner against 11,631 correct relations and an average error rate of 18% for KnowItAll.

While both systems aim to extract arbitrary relations from plain text, main differences to Wanderlust are a focus on statistical methods. Most grammatical data on analyzed sentences is disregarded in order to allow for an analysis of greater amounts of text. Without identification and disambiguation of entities however, much of the information gathered runs a risk of redundancy and cannot be used as a knowledge base. Wanderlust by contrast aims to find the information extracted within single sentences and therefore does not rely on statistical verification of extracted relations. Because identification of entities is performed in the context of the use case, the result of this thesis can be used as a knowledge base.

2.1.2 Structured Metadata

Due to relatively broad coverage and advantageous characteristics (see Section 2.2), Wikipedia has been found to be an invaluable source for knowledge extraction. The majority of systems that utilize this corpus exclusively makes use of the structured metadata available in Wikipedia, but disregard the information hidden in the articles themselves. The approach of *DBPedia* [2] relies on the infoboxes available for many Wikipedia articles. These provide structured information which is consistently-formatted for articles describing instances of the same type. For a predefined subset of article types, handcrafted extractors have been written in *DBPedia*, yielding a multi-million set of RDF triplets. Further semantic annotations that are available in Wikipedia are categories which are arranged in a hierarchical structure. Wikipedia articles may be associated to one or more of these named categories. Ponzetto et al. [31] use a set of shallow

linguistic heuristics applied to the category tags in order to infer a taxonomy over the concepts described by Wikipedia articles. The *Yago* system [38] links Wikipedia category labels to WordNet entries. It then exploits the hypernym and hyponym relations available in WordNet to derive a highly accurate taxonomy. To derive further relations *Yago* depends on manually constructed rules applied to the Wikipedia category structure.

The use of the above mechanisms has the drawback of losing the wealth of information stored within the plain text of Wikipedia's pages and instead focusing only on the information that can be gained from structured metadata. [31] only allows to extract taxonomic relations and *Yago* is limited to a predefined set of relations. Wanderlust on the other hand, much like the approaches introduced in 2.1.1, is capable of extracting arbitrary relations from plain text.

2.1.3 Deep Linguistic Analysis

Recent work most similar to the approach used in this thesis are [20], [43] and [39]. While previously mentioned approaches either exclusively rely on the existence of structured data or on quantitative assessment, these systems allow to extract semantic relations from natural language text using information given by a deep parse. [20, 39] make use of the grammatical structure provided by a deep parse. Nakayama et al. [20] analyze the phrase structure trees produced by the Stanford NLP Parser [19]. Their extraction process is controlled by a small set of handcrafted patterns defined over the phrase structure trees, limiting the coverage of their approach. A heuristic based on link structure analysis of the Wikipedia corpus is used to classify sentences as important for an article. Only these sentences are considered in the extraction process. However, they do not examine the impact of this heuristic.

The *Kylin* system [43] exploits the correspondence between article text and structured data of infoboxes. Relations expressed in infoboxes are heuristically matched to sentences in the corresponding article. This way, sentences are labeled as containing specific relations. These labels are then used to learn relation specific extractors. Their approach achieves very high precision, but is limited to semantic relations that exist in infobox templates. Similar to Wanderlust the *Leila* information extraction system [39] uses the linkages produced by the Link Grammar NLP Parser [37]. The system combines deep linguistic analysis with machine learning methods. In the learning phase it relies on the existence of attribute/value pairs describing instances of a specific relation. As such the system is limited to relations for which instances are available in structured format.

The approaches introduced here share the property that only a limited set of relation types can be extracted using specifically crafted or trained extractors. Wanderlust by contrast seeks to identify universally applicable grammatical patterns which can be used to generate arbitrary relation types without needing to be specifically trained. The ability of finding arbitrary relations in plain unstructured text is the most important characteristic of Wanderlust.

2.2 English Wikipedia

This section discusses the English Wikipedia and its properties. As of March 2009, the Wikipedia Online Encyclopedia has more than 12 million pages in over 250 languages[1]. True to the design principle of Openness, a community of more than 16 million registered users has participated in authoring and revising Wikipedia content. As a result, Wikipedia is arguably the largest encyclopedia worldwide. As of March 2009, the English Wikipedia alone has more than 2.7 million pages. The German Wikipedia is the second largest, consisting of over 850,000 articles. Third largest is the French Wikipedia, which comprises over 750,000 articles.

This thesis operates on a full dump of the English Wikipedia, acquired at [2]. Because of its large and active community, the quality of writing (in terms of use of proper grammar and spelling) is high and factual. This greatly facilitates computerlinguistic analysis using Link Grammar NLP, which performs best on scientific, newspaper-like texts. Also, the quality-of-content is believed to be high. A study by Nature magazine in 2005 [15] which had experts compare 42 articles in the English Wikipedia and Encyclopedia Britannica discovered an equal amount of errors in both encyclopediae and a slightly greater amount of smaller factoid mistakes in Wikipedia (3.9 errors per article in Wikipedia, 2.9 errors per article in Britannica). The results of this study (which are not undisputed, see [6],[3]) indicate a very high quality-of-content for Wikipedia.

The English Wikipedia thus represents the most complete encyclopedia in the world with a very high quality-of-content. These properties support the decision to use the English Wikipedia as a basis corpus for this thesis. A semantic wiki is to be generated from this database by automatically extracting semantic relations from the data given by Wikipedia's pages and page links. These two elements are introduced in the following subsections where special mention of a number of properties that support the development of a semantic wiki as proposed in [22] and are used for the implementation of this thesis is made.

2.2.1 Pages

The *page* is one of Wikipedia's main elements. Each page represents an article in the sense of an encyclopedia and covers a topic as indicated by its *page title*. This title serves as an identifier and is unique, meaning that no two pages may have the same page title. In the following subsections, properties of pages are discussed and their possible use as entities in semantic relations illustrated.

Conventions

Pages adhere to a number of (non-binding, but community enforced) guidelines and conventions for page authors, which help in maintaining a common structure for pages. For page content, a manual-of-style is published on the Wikipedia web page, which somewhat belying the open source nature of Wikipedia helps ensure a mostly homogenous "look" across the pages. These uniform naming conventions and style guidelines increase order and therefore both readability

[1] http://meta.wikimedia.org/wiki/List_of_Wikipedias#Grand_Total, March 2009
[2] http://download.wikimedia.org/enwiki/, October 2008
[3] http://www.nature.com/nature/britannica/index.html, March 2009

and usability of the wiki. One example of such conventions are naming conventions[4], which cover among other things dates, numbers, people and places. For the page title, which both as an identifier and source of information is important to this thesis (see 3.1.1), a number of naming conventions exist. These include:

- Definite and indefinite articles at beginning of page names
- Naming conventions for people and places

Important here is that the naming of the page titles follows a certain regularity, for example when it comes to disambiguation as introduced in the ensuing subsection. Conventions exist as well for the overall style of the plain text of the page. The Wikipedia manual-of-style gives information among others things on when to capitalize or highlight words, when to use bold face or italics or how to write foreign (non-English) terms. For a full list of all conventions refer to [5]. For this work, the following conventions are important:

- Capital letters: Only proper nouns may be capitalized. Capitalization of whole words for emphasis is undesired. Acronyms may be capitalized, but letters in sources of acronyms should not (e.g. FOREX: FOReign EXchange, better written as *for*eign *ex*change). This convention is used to identify proper nouns.
- Boldface: In the first paragraph of a page, the topic discussed should be written in boldface, as well as all synonyms and acronyms. By convention, this should be the only use of boldface in pages. In the named entity recognition process of the algorithm, this convention is used to find synonyms. The process is described in 3.1.1.

While technically not obligatory for page authors, conventions are well-enforced by virtue of the large authoring community of the English Wikipedia. Pages which cover well known topics face more scrutiny than lesser known topics and are by observation usually better adapted to the manual-of-style and naming conventions.

Disambiguation

In most cases the page title reflects the topic of its page and serves as its unique identifier. This presents a problem for ambiguous terms which require different pages for their different meanings. In Wikipedia, there are mechanisms to handle the disambiguation of pages for ambiguous terms. If one meaning of an ambiguous term is more commonly used than any other of its meanings it is called the *primary topic*[6]. The page title for the primary topic is by convention the term itself, which means that all other meanings of the term must use a different page title. For these meanings, context information is added in brackets after the term, which is then used as page title.

Consider for example the page titled "Apollo" partly illustrated in Figure 2.1. The primary topic for the term as determined by the authoring community

[4] http://en.wikipedia.org/wiki/Category:Wikipedia_naming_conventions, March 2009
[5] http://en.wikipedia.org/wiki/Wikipedia:Manual_of_Style, March 2009
[6] http://en.wikipedia.org/wiki/Wikipedia:Disambiguation, March 2009

Figure 2.1: The page titled "Apollo". Primary topic is the Greek god. At the beginning of the page, a link to a disambiguation page is given.

is the Greek god named Apollo. All other meanings of the term use different page titles. A German automobile from the early 20th century which was also called "Apollo" for example has as page title "Apollo_(1910_automobile)". This page title is not straightforward for a user looking to find a page concerning the aforementioned automobile, because no conventions exist for the context information given in brackets aside from that it should short and reasonable. This means that guessing the title for the page a user is looking for may be difficult, which is why two mechanisms are applied to facilitate the search for pages not covering the primary topic of their page title: *Disambiguation links* and *disambiguation pages*.

Figure 2.2: Example disambiguation page for the term "Apollo".

Disambiguation links are used if a term has a very small number of meanings. They are listed at the top of the page covering the primary topic of a term and each link to a page covering another meaning. For each link, an explanation is given describing the topic of the target page. If a term has many possible

meanings, a list of disambiguation links at the beginning of a page can be too large, which is why in such cases only a single link is given which points to a disambiguation page. Such pages cover no topic of their own, but consist only of a list of disambiguation links. Disambiguation pages normally have the ambiguous term and the word "disambiguation" in brackets as page title. Refer to Figure 2.1, where a link to "Apollo_(disambiguation)" is given in the beginning of the page, leading to the page illustrated in 2.2. Some terms do not possess a clear primary topic. In such cases, the page title of the disambiguation page is the term itself (without the word "disambiguation" in brackets) and all meanings of the term have page titles with context information.

Pages as Entities

One of the main ideas of transforming Wikipedia into a semantic wiki is the possibility of viewing pages as entities. As discussed in the previous section, a term – and more importantly each meaning of a term – can have an individual page and identifier. Because these identifiers are unique in the entire system, pages may be used as URIs for their topics, which in the English Wikipedia are largely conceptual or thematic [34]. Both their unique identifiers and the fact that most pages contain an article covering a unique topic qualify Wikipedia pages for the use as entities. With over 2.4 million pages, the English Wikipedia covers a very large number of entities with which an extensive knowledge base can be built.

2.2.2 Page Links

Page links are the second of Wikipedia's elements important to this thesis. Page links provide interconnectivity between pages and facilitate browsing of related content for users. They consist of two elements: An anchor text and a target page. The anchor text is usually part of the plain text of the page, but is highlighted in order to distinguish it from other terms. The target page is the page which is opened when a user clicks on the highlighted term. In many cases, both anchor text and the page title of the target page are the same. In some cases, including disambiguated terms, they may be different. True to the design principle of Openness, Wikipedia's markup is intended to be easy to use for authors. In cases where both anchor text and the target page title are the same, annotation of a page link is done by simply surrounding a term by two pairs of rectangular brackets. This causes the term to be highlighted and serve as a link to the page with the term as page title. An example for a sentence with such page links is:

Python was killed by [[Apollo]].

Here, the term *Apollo* links to the identically named page titled "Apollo". In some cases, when a term is written in a form different than its main singular form for example to include plural endings, the original term is surrounded by rectangular brackets as above and the endings are attached to the link. An example for a sentence with such page links is:

San Francisco has [[bus]]es, [[taxicab]]s, and [[streetcar]]s.

In this case, the endings are part of the anchor text, but the link leads to the page as given by the term in rectangular brackets. The term *buses* would be highlighted but lead to the page "bus". The last of the most common cases is when anchor text and target page are different, making explicit mention of both necessary when defining a page link. Such page links are written by separating anchor text and target page title by a vertical bar ("|"). The entire construct is surrounded by two sets of rectangular brackets. Such page links must be used for example for ambiguous terms which link to pages other than their primary topic. An example of this is:

Apollo killed [[Python|Python_(mythology)]].

In this sentence, only the term "Python" is visible. By clicking on the term, the page covering the term as meant in the context of the sentence is opened, namely "Python_(mythology)". This means that a term can be disambiguated by virtue of its page link if it can be assumed that the greatest part of page links link a term to its meaning in the context of the page. This idea is a principal component of the identification of entities performed by Weitblick and is discussed in 3.1.

2.3 Semantic MediaWiki

This section introduces Semantic MediaWiki (SMW) [21], an open-source extension to the MediaWiki platform, which is the underlying software of Wikipedia. Founded and led by Markus Krötsch and Denny Vrandecic, the authors of [22], it implements the proposal laid out in the paper for the creation of a semantic wiki. It extends the Wikipedia markup language to allow for the annotation of semantic page links and adds new database tables for semantic relations to the system. Also, a number of powerful features are implemented, such as semantic queries and RDF-export which make use of the increased potential of a semantic wiki. Semantic page links and key features of SMW are introduced in the ensuing subsections.

2.3.1 Semantic Page Links

As illustrated in 1.2.1, semantic page links are an extension to regular page links which allow the author to specify the type of relationship the page in which the link is set and the page the link points to have. Annotation is performed by inserting predicate information into regular page links directly after the opening rectangular brackets and separated from the rest of the page link by two colons. In the following, an example of this is given for the page link [[Python|Python_(mythology)]]. The predicate is highlighted bold for better readability:

[[**killed::**Python|Python_(mythology)]]

The insertion of a semantic page link into a page causes the system to add a subject-predicate-object triplet into its database. The page in which the page link is set is the subject, while the page to which the page link points to is the object. If the above displayed semantic page link is written into the

page "Apollo", the relation Killed(Apollo, Python_(mythology)) is added to the knowledge base of the semantic wiki.

The system imposes no restriction for the choice of words of the predicate, meaning that users have the freedom to define any relation type. This makes SMW an ideal candidate for Wanderlust which is able to generate arbitrary predicates, but makes it vulnerable to problems stemming from the Doppeldenk limitation of predicate diction (see 1.2.3). One problem is **disambiguation** for predicates. Entities are disambiguated by creating distinct Wikipedia pages for each meaning of an ambiguous term. For a predicate, which is simply a sequence of words, this is not possible. Consider for example the predicate BornIn. This relation may link a person to both its *birthplace* or its *time of birth*. For the latter, a century, a year or an exact date is possible. This means that the predicate BornIn in itself is not disambiguated, which can obscure the meaning of a relation. In many cases, the missing disambiguation of predicates does not pose serious problems, because as in the case of BornIn, the relation can be disambiguated by virtue of the object. If the object of BornIn is a place, then BornIn refers to the place of birth. If it is a date, BornIn refers to the time of birth.

Another problem of predicate diction is that a certain type of relation may be indicated by a number of different predicates. Consider the previously mentioned example relation Killed(Apollo, Python), which may among other ways be written as Slew(Apollo, Python). As such, the system does not know that the relations Killed and Slew are **synonyms**, limiting the expressiveness of the model of knowledge. Similar to this is the problem of **implication**. The predicate CityIn may implicate the predicate LocatedIn, meaning that the relation CityIn(Essen, Ruhr Area) implicates the relation LocatedIn(Essen, Ruhr Area).

The Semantic MediaWiki platform offers a feature called *subproperties* to handle such implication. By defining CityIn as subproperty of LocatedIn for example, all queries for the former also query for the latter. This feature helps reduce difficulties in usability posed by large numbers of distinct relation types. Wanderlust uses heuristics in order to automatically generate as many of these subproperties as possible. The method used is illustrated in Section 3.5.

2.3.2 Features

The Semantic MediaWiki platform offers a number of features which make use of the information given by semantic page links. By browsing the properties of a page, all relations which point to and from the page are listed. This gives an overview of all relation triplets pertaining to the page in question that are known to the system and shows how it relates to other pages. Using the **browse view**, users can browse from page to page along the model of knowledge in the semantic wiki.

Another key feature of SMW are **semantic queries**, which give the semantic wiki the character of a question-answering system. Queries may include logical operators and comparators and are posed analogously to the annotation of semantic page links. Queries if successful are answered by a list of one or more page titles. A simple query may be posed by typing "[[killed::Python_(mythology)]]", which yields as answer the page which points to "Python_(mythology)" using the Killed predicate. If multiple pages fulfill this criterion, all are listed, as for example in the case of "[[tributary of::Ohio River]]". The result of this query

is a list of all tributaries of the Ohio River. An illustration of this is provided in Figure 2.3.

Figure 2.3: Example query for all tributaries of the Ohio River.

By including logical operators, queries can be formed to answer more complicated questions. The AND-operator is automatically assumed if two queries are written after one another. An example for this is "[[is::county]] [[located in::Iowa]]", which queries for all counties located in the US state of Iowa. By inserting the logical OR (written as "||") between two queries, all answers can be queried for which fulfill either one of both queries. An example of such a query would be "[[son of::Zeus]] || [[daughter of::Zeus]]", illustrated in Figure 2.4. By using all options provided by the semantic search option, powerful search queries may be posed to the system. In a strongly annotated version of the English Wikipedia, the semantic search may serve as a general world knowledge question-answering system.

Figure 2.4: Query using disjunction for sons and daughters of Zeus.

Semantic MediaWiki also offers the export of data in RDF, the resource description format used in Semantic Web applications. This feature makes data from the semantic wiki available to a possibly wide range of other applications. Possible uses include the generation of ontologies and semantic networks.

2.4 Link Grammar

This section gives an introduction to the basic principles of the link grammar formalism, which is used in this thesis for the purpose of relation extraction. Other prevalent formalisms are discussed in 2.4.2, where a comparison to link grammar is made and reasons for choosing the latter formalism as main instrument given. Section 2.4.3 discusses the Link Grammar Parser and its properties.

2.4.1 Formalism

Link Grammar [37] is a context-free formalism for the representation of grammar devised and realized by Daniel D. Sleator and David Temperly. It is based on the observation that in most natural languages if in a sentence arcs are drawn above words that relate to each other, these arcs will not cross [26]. Accordingly, a sentence is represented in link grammar format by connecting ("linking") all words which relate to each other in such a way as to fulfill the following meta properties:

- Planarity – links drawn above the words must not cross. This property is drawn from the above mentioned observation.
- Connectivity – pairwise links between all words will connect all words of the sentence together. This property ensures that there are no words or parts of a sentence which are not connected to the rest.
- Exclusion – the same pair of words may not be connected with more than one link. This somewhat trivial property means that a pair of words may have at most one form of grammatical relation.
- Satisfaction – in addition to the meta properties, links must satisfy linking requirements for each word in the sentence. These linking requirements differ between words and word types.

As previously noted, a sentence represented in link grammar formalism (also referred to as linkage) consists of a sequence of words and a set of links. Links are annotated with link labels, which indicate the nature of the relation between two connected words. Types of relations are named *link types*. Link grammar analysis is at the lexical level, meaning that each word in the lexicon has an individual definition describing of how it can be used in a sentence. In addition to the meta-properties, the individual properties of the words in the sentence need also be fulfilled. Consider in the following the determiners "a" and "the", the nouns "cat" and "snake", the name "Mary" and the verbs "ran" and "chased". According to word, word type and tense form each word has one or more *connectors*, which specify how it may relate to other words.

In Figure 2.5, connectors are shaped differently to illustrate that mismatching connectors cannot form connections. They are annotated with link types

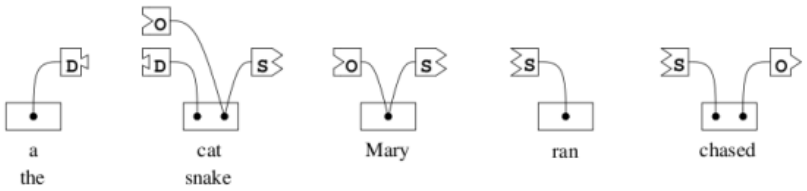

Figure 2.5: Example words in a link grammar lexicon and connectors. Figure taken from [37].

indicating how each word may relate to others. The word "cat" for example may either relate as subject (S-connector) or object (O-connector) to a verb which allows this, and/or relate to a determiner. While the verb "chased" allows for a direct object to be connected via the O-connector, this is not the case for the verb "ran". This means that a sentence such as "cat chased snake" is possible while "cat ran snake" is not, demonstrating how each word form has individual linking requirements.

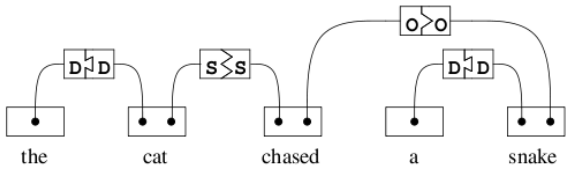

Figure 2.6: Words from Figure 2.5 linked to form a linkage. Figure taken from [37].

If a sequence of words can be connected in such a way as to fulfill the meta-properties, the sentence is considered grammatically valid. Figure 2.6 shows such an example constructed from the words mentioned above. The connector types form the link labels of the generated linkage. The resulting linkage for the example is illustrated in Figure 2.7

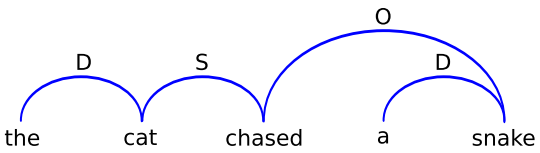

Figure 2.7: Simplified form of the linkage from Figure 2.6. The link types are written on the links connecting the words. Subtypes are not indicated.

Using this principle, sentences can be parsed into link grammar representation consisting of words, links and link labels. There are more than 100 link types, each with a number of *subtypes*, which may be used to connect words. Subtypes give additional information to further specify the link type and are written in lower case. The link type **S** for example which indicates a subject relationship to a verb, has among others the subtypes **p** (for plural) or **s** (for singular). Accordingly the link label (which comprises both the link type and

subtype) **Sp** stands for a plural subject relationship. An overview of all link labels used in examples in this thesis is given in Appendix B.2.

Link grammar may also be used to determine grammatically incorrect sentences. The sentence "the Mary chased cat" for example is rejected by link grammar, due to crossing links. Refer to Figure 2.8 for an illustration of this.

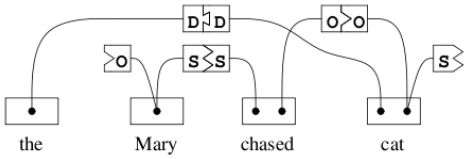

Figure 2.8: Example of a sentence which cannot be written into link grammar formalism, because the sentence is grammatically incorrect. Note the crossing links which is forbidden by the planarity criterion. Figure taken from [37].

This shows how link grammar displays the grammatical structure of a sentence. A parse is only possible if the sentence contains grammatically correct English. The following lists a number of advantageous properties of the formalism:

- Link grammar is a *lexical system*, meaning that grammar is distributed among the words of the dictionary. Since each word has a set of connectors, grammatical parsing can deal with the behavior of any words, including irregular verbs. This also facilitates the construction of a large grammar, as word definitions may be constructed and changed independently from each other.
- A parse using link grammar connects semantically and syntactically associated words together and gives information about the nature of their relation by assigning link labels. This important feature is used in this thesis to extract information on the relation between words.

Other grammatical formalisms are introduced and compared to link grammar in the following subsections.

2.4.2 Other Formalisms

Compared to other main grammatical formalisms, link grammar is perhaps the least known [35]. Arguably the most researched and accepted formalism is constituency theory. Another formalism related to link grammar is dependency theory. A short introduction to both formalisms is given in this section and a comparison made to link grammar. Finally, the reasons for choosing link grammar are illustrated.

The example sentence "Essen is a city in Germany" is used in this section. Its link grammar parse is given in Figure 2.4.2. In both the constituency as well as the dependency section the respective parses are illustrated.

Constituency

In constituency, the grammaticality of a sentence consists of elements which in turn consist either of elements or words. Each such element represents a

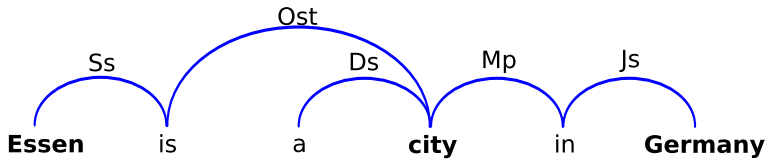

Figure 2.9: Linkage for the example sentence.

constituent, which in syntactic analysis is a word or a group of words which functions as a single unit within the sentence's syntactic structure. The grammaticality of a sentence is therefore represented in a hierarchical tree-like structure with each node being an element and each terminal node a word. The hierarchy of the elements and their types describe the sentence. [35]

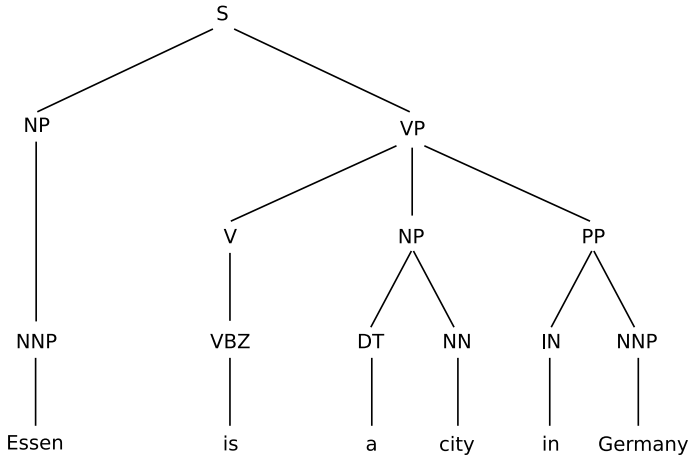

Figure 2.10: Constituent parse of the example sentence.

Consider the example sentence illustrated in Figure 2.10, generated by the Stanford Parser[7]. The root of the tree is the entire sentence (denoted **S** in the figure), which consists of a noun phrase (**NP**) and a verb phrase (**VP**). While the noun phrase consists only of the POS-tag **NNP** and thus the noun "Essen", the verb phrase consists of several elements. Note that all terminal nodes are words, while some nodes within the tree are grammatical elements. The POS-tags used in this thesis are from the Penn Treebank POS Tagset [25]. An overview over the POS-tags used in examples of this thesis is given in Appendix B.1.

This shows how in constituency the grammaticality is not necessarily centered around single terms, but rather expressed by hierarchies of groups of words. This stands in contrast to link grammar where grammaticality of the sentence is expressed through relations of the words.

Dependency

Dependency formalism is the second of the principal formalisms used to describe the grammaticality of sentences. Like constituency, a hierarchical tree-like struc-

[7] http://nlp.stanford.edu/software/lex-parser.shtml, March 2009

ture is applied, but here words may be both terminal *and* non-terminal nodes [35]. The representation of grammaticality is therefore not "detached" from the words but rather focused on which words of the sentence are dependent on each other. In this respect, dependency closely resembles link grammar. Consider the illustration of the example sentence in Figure 2.11, which was generated using Connexor[8]. The exact same words are connected as in Figure 2.4.2, demonstrating the similarity between the two formalisms.

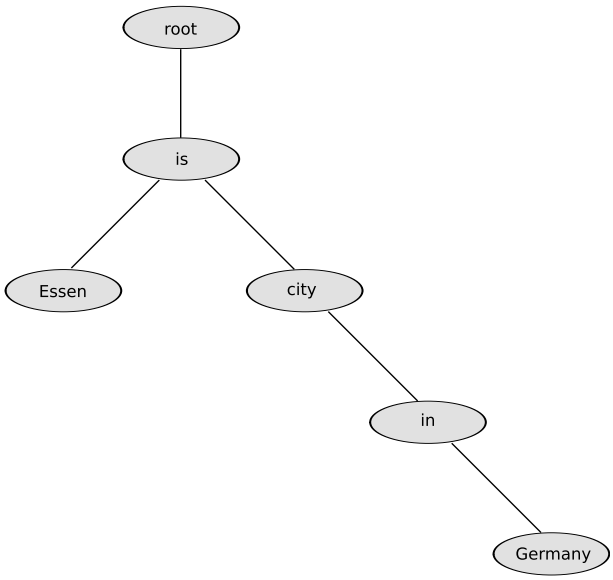

Figure 2.11: Example sentence in dependency formalism. The verb "is" is connected to the root of the tree. Dependent on the verb are the subject "Essen" and the object "city". The preposition "in" is dependent on "city", and "Germany" on "in".

Some differences exist between both formalisms. While in link grammar the grammatical relationship between two connected words is denoted by the link label, this need not be the case in dependency where unlabeled connections suffice. This is possible because in dependency connections are unidirectional and two words are connected if one word is grammatically "dependent" on the other. Dependency forms a tree-like structure where nodes are dependent on nodes which lie closer to the root.

Comparison

The main difference between both aforementioned formalisms is that in constituency grammaticality is expressed as a hierarchy of constituents to which words belong, while in dependency (much like link grammar) words are grammatically dependent on each other. Nevertheless, there is discussion on how different both formalisms actually are, given that algorithms exist which can transcribe a sentence parsed in one formalism to another [35]. In a dependency parse, for example, constituents could be seen as each node combined with all

[8] http://www.connexor.com/, March 2009

its underlying nodes. The Link Grammar Parser [37] comes with such an algorithm, making it possible to see parse results not only in link grammar, but also in constituent form.

As the purpose of this thesis is to find predicates for words (e.g. a sequence of words which "connect" a subject entity to an object entity), it was decided that a formalism which binds grammaticality to words would be used. For this purpose , the link grammar formalism was deemed to be the most useful because of numerous features. Firstly, there are a large number of link labels which classify the grammatical relationships between words of a sentence, enabling detailed and differentiated analysis. Secondly, the Link Grammar Parser is robust and performs well even on long sentences. And finally, the Link Grammar Parser is free, open source and actively maintained.

2.4.3 Parser

The Link Grammar Parser [37] is a parser for the English language implementing the link grammar formalism described above. Given a sentence it can produce a grammatical representation both in link grammar and in constituent form, covering a wide variety of grammatical phenomena. It has a dictionary of about 60.000 word forms, but can also handle words it doesn't recognize. The parser can skip over portions of a sentence it cannot interpret, making it robust. It is released under a license which allows unrestricted use in commercial applications and is available for download at [9]. Documentation including the Link Parser API and a detailed description of all link types can be found here as well. The parser is written in C, but bindings for other languages, such as Ruby[10] and Ocaml[11] also exist . Also, there are extensions of the Link Grammar Parser for the Russian and Persian languages.

Currently, the link grammar project is being maintained by the AbiWord team at [12], which use link grammar as a grammar checking feature in their open-source word processor AbiWord [40]. Stated goal is to improve the parser and make it robust for problematic sentences, a process during which the parser has been expanded by additional link types.

Problematic Sentence Types

This section lists some sentence types which the Link Grammar Parser struggles to parse correctly. As stated by the AbiWord team, the Link Grammar Parser has problems with

- Long and complex sentences
- Slang and ungrammatical sentences
- Bulleted lists and headlines, which are problematic because of irregular grammar

Most of these problematic sentence types do not affect this thesis, given that one of the reasons for choosing the English Wikipedia as corpus to operate on

[9] http://www.link.cs.cmu.edu/link/, March 2009
[10] http://www.deveiate.org/projects/Ruby-LinkParser, March 2009
[11] http://ramamurthy.ramu.googlepages.com/ocamllinkgrammar, March 2009
[12] http://www.abisource.org/projects/link-grammar/, March 2009

was its factual content and its mostly grammatical style of writing enforced by Wikipedia's active community. Sentences however can be long and complex, which can pose problems for the parser.

During implementation, other difficulties for the parser than those listed above have been noted. Specifically, these involve named entities which span several words. Because the parser operates on a lexical level, these entities are not recognized as a whole, but rather as a sequence of individual words. This proves problematic in some cases. One example of this found in the sentence "*Doom II: Hell on Earth is a first-person shooter video game created by id Software*" which contains the entity "Doom II: Hell on Earth". It is however not recognized as such and parsed as a sequence of words, disrupting the parse of the sentence. For such cases, a workaround has been applied in this thesis. It is described in Section 3.2.

Quality-of-Parse Variables

Each linkage produced by the Link Grammar Parser has three variables indicating the quality of the linkage as judged by the parser. The first of these variables is called *skipped words*. It indicates how many words of the sentence had to be skipped in order for the parse to be made. Skipped words are treated as if they are not part of the sentence, which has the potential of dramatically changing its semantics. The second variable is called *cost*. For some words, not all connectors have a cost of zero. Rather, some connections involving words are judged to be less likely than others. The higher the cost of a linkage, the less likely the chance of it being correct. Finally, the third variable is *linkage number*. It represents the ordering of linkages as returned by the parser. The linkage with the smallest number (e.g. 0) is believed by the parser to be the most likely to be correct. Therefore, this variable includes the variables of cost and skipped words. But even if multiple linkages have the same values of cost and skipped words, the linkage number orders these according to quality. Therefore, the variable carries information of its own, in addition to the information given by the other two variables.

The variables are employed in this thesis to insure a certain quality for the Wanderlust algorithm to operate on. An analysis of the impact of the variables is performed in Section 4.4.1.

2.5 Summary

This chapter has discussed and compared related work to the approach for information extraction used in this thesis. Theoretical prerequisites for the use case of generating a semantic wiki have been introduced. The English Wikipedia and the Semantic MediaWiki platform have been discussed with a focus on properties important to the use case of generating a semantic wiki. The ideas of using Wikipedia pages as entities and using Wanderlust to automatically annotate semantic page links have been elaborated on. The link grammar formalism as the main computerlinguistic device used in this thesis was described and compared to other prevalent formalisms. The reasons for choosing link grammar have been stated.

Chapter 3

Wanderlust Algorithm

This chapter introduces Wanderlust used for the specific task of generating a semantic wiki using the English Wikipedia corpus. For this application, the core method illustrated in Section 1.2.2 is extended by an entity tagging mechanism. An overview of the entire algorithm is given in Figure 3.1. The individual steps are described in detail in the following subsections.

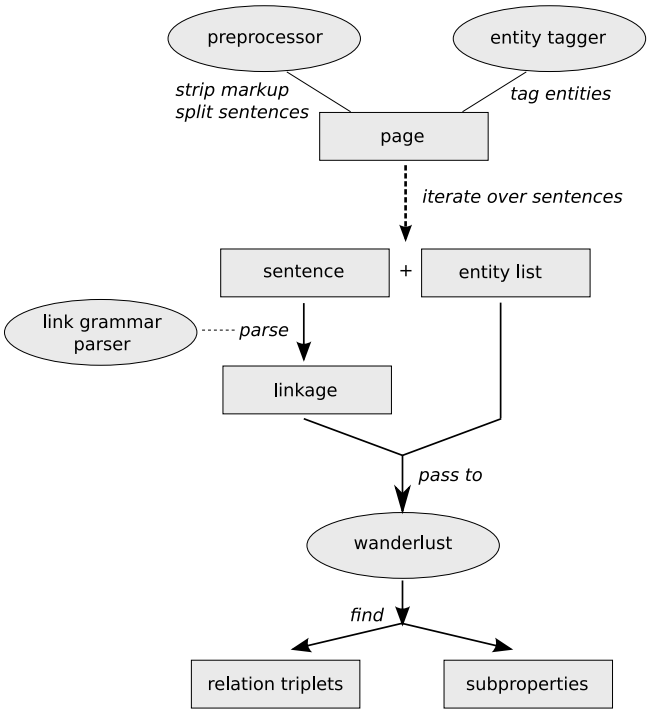

Figure 3.1: Outline for Wanderlust applied to the use case.

Wikipedia articles are analyzed one by one. A preprocessor removes all markup and meta information from the page, passing page links and terms written in bold face to an entity tagging procedure dubbed Weitblick. It finds named entities in an article and in a subsequent step disambiguates them. The

output of the procedure is a list of disambiguated entities (*entity list*) that are valid for the article. A detailed description of this method is given in Section 3.1.

The preprocessor proceeds to split the page into a set of sentences. Each sentence has a list of all entities found by Weitblick. All sentences that contain at least two entities are passed to the Link Grammar Parser, while all others are dismissed. To enhance the performance of the parser, sentences are rewritten beforehand as described in Section 3.2. The obtained linkages are input to Wanderlust which attempts to extract semantic relations for all pairs of entities. For each pair of entities within the sentence, all possible linkpaths are generated and analyzed. Those linkpaths that do not meet certain grammatical criteria are dismissed, a process illustrated in Section 3.3. The determination of these criteria is described and analyzed separately in Chapter 4. For all remaining linkpaths, wordpaths are generated which serve as predicates for the semantic relation between subject and object entities.

The output of Wanderlust are relation triplets which are stored in the SMW database. In addition to this, the algorithm also attempts to identify subproperties between extracted predicates with the procedure outlined in Section 3.5. For the purpose of this chapter, the algorithm is divided into five principal steps:

1. *Mark entities*: Find all entities in a given page using the Weitblick algorithm. This step is called "identification of entities" and is discussed in Section 3.1.

2. *Parse sentence*: Each sentence has a list of entities. In this step, parse sentence into link grammar formalism. The parsing step and the modifications made beforehand are discussed in Section 3.2. This step is referred to as "link grammar analysis".

3. *Make relations*: Make a list of all possible relations between all pairs of entities using wordpath information. The methodology is explained in detail in Section 3.3. This step is referred to as "linkpaths".

4. *Filter valid relations*: Eliminate all relations from the list which do not meet the necessary grammatical criteria to be considered valid. The filtering step is explained in Section 3.4. This step is called "validation".

5. *Normalize*: Normalize relations on the list. This is a post processing step which generates subproperties for SMW. It is discussed in 3.5 and is referred to as "normalization".

The first step is performed once for each page, while all others are performed for each sentence with more than two entities. In the following subsections, each of the five steps is discussed in detail.

3.1 Identification of Entities

As mentioned in the previous section, the first step of the algorithm is to identify words or sequences of words within a given page as entities. The algorithm constructed for this purpose is named *Weitblick* and utilizes features of Wikipedia, as described in this section.

A trivial approach to the problem of entity tagging might be to select all nouns and/or named entities from the text of the page and use the word or sequence of words itself as identifier. The problem here is that while a corresponding page may well exist in the English Wikipedia with such a page title, it is not determined whether this page is an accurate representation of the meaning of the word in the context where it was found. If for example each occurrence of the noun "Apollo" were to be identified with the URI of the Wikipedia page "Apollo" then all of the many possible meanings of the word would point to one page. The problem, thus, is one of disambiguation, a much discussed topic in computerlinguistics [28]. In order to assign URIs to nouns or named entities, their meaning within the context they are stated in needs to be determined.

Weitblick makes use of two properties of Wikipedia to solve this problem. The first property is Wikipedia's handling of disambiguation, which has been discussed in 2.2.1. Important here is that pages with different titles can be created for each distinct meaning of a term and that the most common meaning of a term has itself as page title. The second property is Wikipedia's page links. The underlying assumption of Weitblick is that the greatest part of page links will link a term to its disambiguated word sense. In the markup view of a page, all page links can therefore be seen as entities. This means that the identification of entities is trivial for all terms with page links within a page. However, the number of page links in respect to total number of nouns within a page is normally rather small which has a number of observed reasons. First, most page links are set only once for a given term within a page no matter how often a term is used. Second, some terms do not have page links, because the user community has not seen a reason to set a page link or neglected the term. While this may help the readability of pages (since articles with too many highlighted words may be more difficult to read), this also greatly reduces the number of sentences with two entities if only terms with page links are considered as such. Weitblick addresses these difficulties. For an overview of the algorithm refer to Figure 3.2.

The first step of Weitblick is to extract all terms with page links from a given page into an *entity list*. Operating under the premise that all occurrences of a term within a page usually refer to the same disambiguated word sense, the list represents all terms Weitblick can safely disambiguate in the page. If any term is listed twice because it is linked to two different pages by two page links, disambiguation for this term is no longer granted and it must be removed from the list. Since pages usually do not page link themselves, finding entities for the concept the page is about must be handled differently. This is described in 3.1.1.

The algorithm then starts to match terms in the entity list to the text of the page. Entities consisting of 3 or more words are blindly matched, while shorter entities are first compared to noun terms within the page. The exact method of entity matching is described in Section 3.1.2. Noun terms that could not be matched to entries in the entity list are classified as *attributes*. Using the heuristics described in 3.1.3 a portion of these attributes can be converted to entities.

In this section and all subsequent subsections, the following "example page" will be used to illustrate the functionality of Weitblick:

Essen is a city in the center of the **Ruhr Area** in North Rhine-

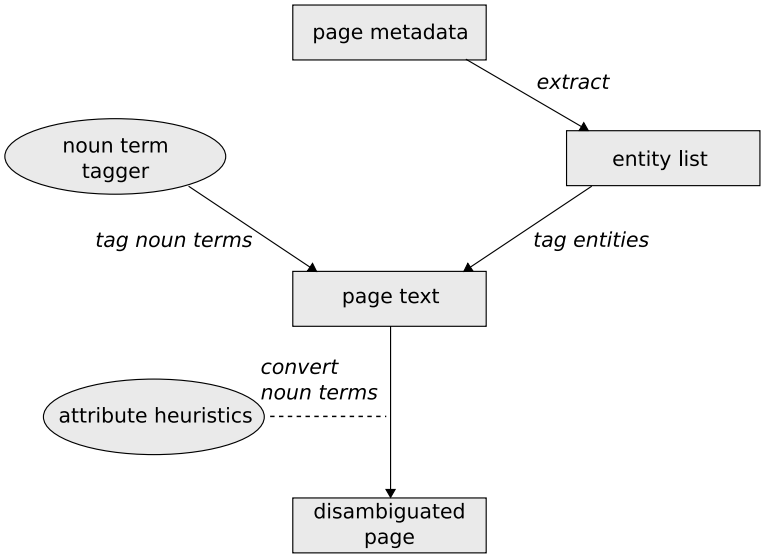

Figure 3.2: Overview of the Weitblick algorithm for identification of entities.

> Westphalia, **Germany**. The **city** was recently appointed **European Capital of Culture** for 2010 representing the whole Ruhr Area. Essen Abbey was founded there.

If the terms given in bold face have page links, the entity list for this text is: *Ruhr Area*, *Germany*, *city*, *European Capital of Culture*. The ensuing section shows how the title entity is added to this list.

3.1.1 Title Entity Extraction

The entity list is constructed using terms with page links. This method however cannot be used with the *title entity*, which is defined as the term(s) used for the concept the page is about. The problem is that pages in Wikipedia do not have page links to themselves, which means that it cannot easily be established which terms in the page refer to the page itself. This is especially obstructive, given that these terms typically are the most important of the page.

Because of this, the algorithm considers two additional features of a page aside from its page links. One is the page title: It usually equals the main term of the article and can thus be used as title entity. The most straightforward page titles are undisambiguated terms. The pages titles *Berlin*, *Onion* or *Cold War* for example can be used as title entities. Therefore, an entry in the entity list with term and page id can be easily made.

In other cases, refinements to the page title need to be made in order for it to be usable as title entity. For disambiguated terms, both the term and its context are part of the page title. The context is given in brackets after the term, such as *Python_(mythology)*, *Bank_(sea_floor)* or *Apollo_(crater)*. Because in a page dealing with a disambiguated sense of the term nonetheless the term itself will be used, the title entities can be generated by stripping away the context information in the page title. For the examples this yields *Python*,

Bank and *Apollo* each with their disambiguated page as identifier.

In certain cases, such as towns or boroughs, context information is separated from the main term by a comma. Examples for this are *New Berlin, Pennsylvania* or *New Berlin, Texas* to distinguish between multiple places with the same name by giving additional information. This can be seen as a form of disambiguation. Such pages are therefore treated analogously, the information after the comma is removed to extract the term to be used as title entity.

Another case are names of people, which usually contain a first and a last name, plus possibly a title, middle names or other information. Examples for such pages are *Carl Friedrich Gauss* or *Alexander von Humboldt*. However, the person whom the page it about is usually only referred to by its last name in the page text, making the full page title not the best term to be used as title entity. By using lists of common first names a simple heuristic has been employed to detect names and extract last names. In these cases, both the full name and the last name are saved as title entities into the entity list.

In order to find alternative terms for the page title (e.g. synonyms), a second property of Wikipedia is used. As noted in 2.2.1, synonyms of the page title are as per convention to be given in bold face during the first sentences of a page. This property can be used to extract page titles. Consider for example the initial sentences from the page called *onion* with page links removed:

> "**Onion** is a term used for many plants in the genus Allium. They are known by the common name "onion" but, used without qualifiers, it usually refers to **Allium cepa**. Allium cepa is also known as the '**garden onion**' or 'bulb' onion and 'shallot'."[1]

All three terms written in bold face are synonyms of the title as required by Wikipedia's guidelines and can therefore be used as title entities. The algorithm therefore extracts all terms written in bold face in the first paragraph of a given page and adds each term as a title entity to the entity list.

The use of the measures illustrated in this section ensures that aside from page linked terms also one or more title entities form part of the entity list. In the example page this means that by virtue of the page title the entity *Essen* is added to the entity list. The matching of these entities to the plain text of a page is described in the ensuing section.

3.1.2 Named Entity Matching

Once the entity list is extracted, Weitblick searches for terms in plain text which are equal to terms in the entity list. Weitblick begins with the longest entity (by number of words) and attempts to match it to terms in the page. All terms in the text that match entities are tagged and can subsequently not belong to any other entity. This procedure is repeated for all entities with a length greater than or equal to 3.

Entities of length 1 and 2 cannot be blindly matched this way. The problem is that shorter terms can form part of larger entities. This is illustrated on the example sentence "*The Berlin Wall became a symbol of the cold war*". Assume that only the term "Berlin", but not "Berlin Wall", is part of the entity list. Blindly matching the term "Berlin" into the example sentence would disrupt

[1] taken from http://en.wikipedia.org/wiki/Onion, October 2008

the named entity *Berlin Wall* and possibly lead to falsely extracted semantics. It must therefore be ensured that short entities are only matched with terms if by doing so sequences of nouns are not disrupted. The Stanford Log-linear Part-of-Speech Tagger [41] using the Penn Treebank tag set [25] was used to POS-tag sentences and identify sequences of nouns, hereafter referred to as *noun terms*. Consequently, entities of length 1 and 2 are only allowed to be tagged within the text if not disrupting a noun term.

The example page is therefore tagged as follows. The only entity consisting of more than three words is *European Capital of Culture* which is blindly matched into the text. All others are matched if no noun terms are disrupted. This means that the entity *Essen* is not matched into the noun term "Essen Abbey". Those noun terms that cannot be matched with entities are marked as attributes. The following shows the example page with entities highlighted bold and attributes in italics:

> **Essen** is a **city** in the *center* of the **Ruhr Area** in *North Rhine-Westphalia*, **Germany**. The **city** was recently appointed **European Capital of Culture** for *2010* representing the whole **Ruhr Area**. *Essen Abbey* was founded there.

Four noun terms are marked as attributes in the example page. Using the heuristic described in the ensuing section, a portion of these attributes is in a subsequent step converted to entities.

3.1.3 Attributes

Using the information given by page links and title entities, all named entities within the page that can be assigned page titles are marked as entities. All other named entities are marked as attributes, meaning that it is unknown which page titles can be used as identifiers for them. Heuristics are applied to convert attributes to entities, following some considerations.

The first consideration is the fact that pages without disambiguation information in their title are the most commonly thought of meaning of the title (e.g. the primary topic, see 2.2.1). Therefore if an attribute is blindly assigned the page which has the attribute term as page title, it may be assumed that more often than not this correctly disambiguates the term. A second consideration is based on the observation that page links are commonly not set for terms that are too obvious to need a page link to the page explaining its meaning. Because of this, terms with missing page links are less likely to be something other than their primary topic. This alone does not necessarily mean anything, since there may be numerous reasons for missing page links and more "unimportant" pages face less scrutiny (and therefore more neglect) by the Wikipedia community. However, both considerations taken together make a case for the possibility of converting attributes into entities by simply assigning them a page title equal to the attribute term, if such a page exists.

Tentative tests however had quickly shown that while by converting all attributes to entities recall is greatly strengthened, there is a high risk of precision loss. Therefore it was decided to only convert attributes into entities which consist of more than one word, such as *grape juice* or *capital city*. This approach takes into account a third consideration being that terms consisting of more than

one word are less likely to have many disambiguated senses, because they are already more specialized than the words they are made up of. This moderately raises recall while not greatly affecting precision.

Within the example page used in previous subsections, two of the four attributes are converted to entities, namely *Essen Abbey* and *North-Rhine Westphalia*. The final text of the example page after Weitblick entity tagging is therefore:

> **Essen** is a **city** in the *center* of the **Ruhr Area** in **North Rhine-Westphalia**, **Germany**. The **city** was recently appointed **European Capital of Culture** for *2010* representing the whole **Ruhr Area**. **Essen Abbey** was founded there.

After the completion of Weitblick, the text is split into sentences (each with a list of entities) and passed to the next step of Wanderlust.

3.2 Link Grammar Analysis

At this point in the algorithm, entities are known and all sentences with less than two dismissed. The remaining sentences each have a list of entities and are parsed into link grammar formalism for further analysis. In some cases, the sentence is modified beforehand in order to raise the chances of it being correctly understood by the parser. All entities which consist of more than one word are written together, so that the parser will treat the entire entity as one word. Because the written together form of the entities will in most cases not be part of the dictionary of the link parser, it will guess the word type of the unknown word to be a noun.

While in many cases the parser correctly identifies subsequent nouns to be part of the same entity (connected by the G or GN link in a linkage), it has problems with entities which include words other than nouns. Examples for this are *Federal Republic of Germany* or *Johann von Goethe*. Such entities are not treated as a noun but rather as two nouns connected by a preposition. In some cases, this incorrect treatment of entities has adverse effects on the quality of the parse. By writing all entities as one word, this problem is avoided.

See for example the sentence illustrated in Figure 3.3, which displays two linkages for the sentence "*Frederick I became the elector of the Margraviate of Brandenburg*". In the first linkage, the sentence is parsed in its original syntax, while in the second the two entities *Frederick I* and *Margraviate of Brandenburg* are written together. The first linkage is incomplete because the sentence could not be understood by the parser. The parser had to skip over two words (written cursive in the upper linkage) and parse the sentence "*I became the elector of the Margraviate Brandenburg*" instead, which changes the meaning of the sentence. Any relations extracted from this linkage are prone to error because of the high number of skipped words. The second linkage however is fully and correctly parsed, demonstrating the benefits of writing entities as one word.

Each sentence is then parsed by the Link Grammar Parser, yielding one or many linkages. The more complex a sentence, the more linkages are usually produced. Sentences which cannot be parsed by the parser must be dismissed at this point in the algorithm. Parsed sentences each have a list of entities and a number of linkages. Each linkage has values representing the score of the parse,

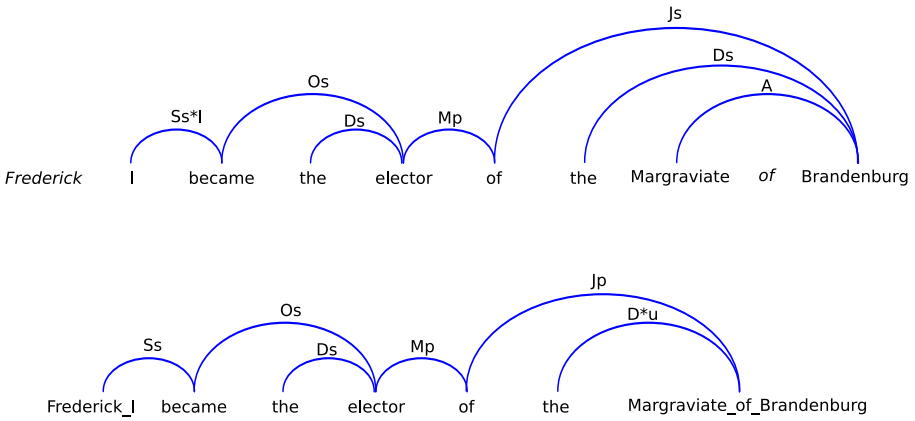

Figure 3.3: Two linkages for the sentence "*Frederick I became the elector of the Margraviate of Brandenburg*". The second linkage uses a modified form of the sentence with entities written as one word.

the number of skipped words and the parse ordering. This information is used in the ensuing step of Wanderlust.

3.3 Linkpaths

This section describes the third step of the Wanderlust algorithm. After the completion of the first two steps, each sentence has a list of entities and a number of linkages. Using this information the algorithm makes a list of unvalidated relations by building wordpaths for each pair of entities. During compilation of the list, some simple techniques are applied to filter out relations which are surely false (mentioned in 3.3.1) and relations which bear risks presented by the missing resolution of coreferences. The problem of coreferences is discussed in Section 3.3.3. In certain cases, the wordpath is modified to drop or contain additional words. These modifications are explained in 3.3.2. Upon completion of this step, the algorithm has produced a list of relations which are not yet validated, meaning that it includes both correct and false relations.

3.3.1 Relation List

Using the approach introduced in Section 1.2.2, the algorithm constructs linkpaths and wordpaths for each pair of entities in the sentence. The wordpaths which connect one entity to another are used as predicates in the relation triplet for both entities. Since wordpaths are constructed in all linkages for a given sentence, two entities may have more than one possible predicate. The example sentence "Essen is a city in Germany" will be used to give an illustration of this. Refer to Figure 3.4 for the two possible linkages of this sentence.

The example sentence has two possible linkages, the difference being the attachment of the "in Germany" part of the sentence. In one linkage it is attached as a modifier to the object "city", while in the other it is attached directly to the verb. By tracing linkpaths between all pairs of the three entities *Essen*, *city*

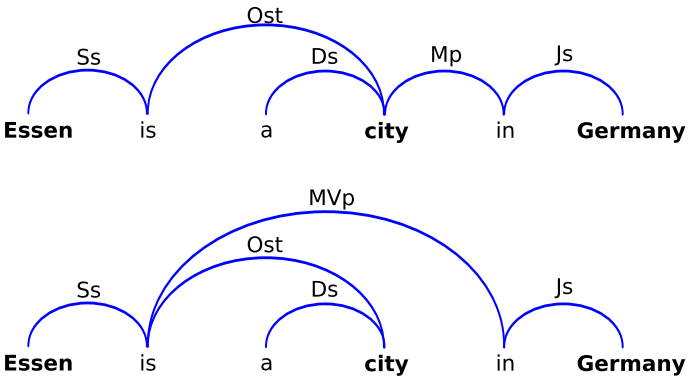

Figure 3.4: An example sentence with three entities (highlighted bold) and two possible linkages. Both linkages have an equal score.

and *Germany* marked in the sentence, a total of 8 different relations are found for this sentence. See Figure 3.5 for an overview of the relations.

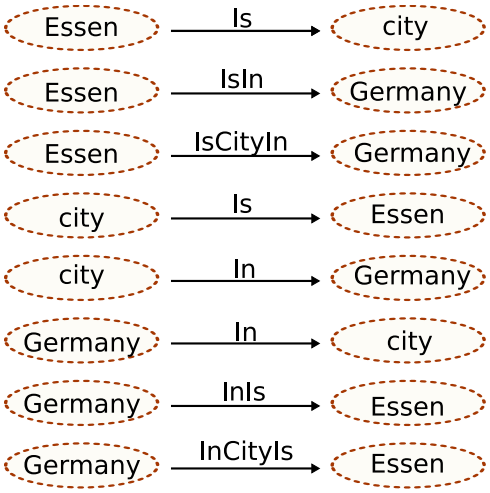

Figure 3.5: List of relations extracted from the linkages in Figure 3.4.

Of these three relations, only two are correct, namely Is(Essen, City) and IsCityIn(Essen, Germany). While intuitively the relation IsIn(Essen, Germany) should also be correct, this is not the case given that the predicate is constructed of an auxiliary verb (e.g. "is") and a preposition ("in"), which makes it semantically too weak to carry information. All incorrect relations must be dismissed in the next step of the algorithm. However, some relations may already be dismissed in pre-filtering by simply regarding the word types which make up the relation. Predicates for example always require at least one verb or noun in order to convey a relation between two noun terms. They must either have an initial noun (such as CapitalOf) or an initial verb (such as HasPopulationOf). All relations with predicates without a verb or noun on the first word position can be dismissed out of hand, which in this example are In(City, Germany),

In(Germany, City), InCityIs(Germany, Essen) and InIs(Germany, Essen). In these cases the predicate is initiated by a preposition, which is not possible.

In addition to this, some relations need to be eliminated in order to prevent false relations to be extracted because of coreferences. This problem is described in Section 3.3.3.

3.3.2 Wordpath Modification

This section deals with modifications that are made to the wordpath in certain cases. The most common modification is made when a wordpath incorporates a noun which in turn is modified by adjectives or number words. Terms that modify the meaning of the noun need to be included in the wordpath in order for it to more accurately mirror the semantics stated within a sentence. A noun with all modifiers is denoted as *expanded noun*. Wordpaths incorporating expanded nouns are called *expanded wordpaths*. A related modification are *particle groups*, which are non-noun groupings of words which fully appear in the wordpath if only one word in the group is touched by the linkpath. Another modification is that if so-called *skipwords* form part of the wordpath, they are omitted from predicates.

In the following subsections, the wordpath modifications are discussed and illustrated with example sentences.

Expanded Wordpaths

In most cases, the wordpath consist only of the words the linkpath touches. If however a word in the linkpath is part of a noun, the full noun core and all adjective and number word modifiers are included for this position of the wordpath. This is done because if a word in the predicate forms part of a larger term, then only including the word but not the entire term may disrupt the semantics of the relation. An example of this can be found in the sentence

"Krypton is a fictional planet in the DC_Comics_Universe."

The sentence is modified according to its entities as described in 3.2, which is why the term *DC_Comics_Universe* is written together. A linkage for this sentence is illustrated in Figure 3.6. From this linkage, the relation IsPlanetIn(Krypton_(Comics), DC_Comics_Universe) is extracted, which contains the noun "planet".

The expanded noun for "planet" in this sentence is "fictional planet". If the expanded noun is incorporated into the wordpath, the extracted relation is IsFictionalPlanetIn(Krypton_(Comics), DC_Comics_Universe) which more closely matches the semantics conveyed in the sentence than IsPlanetIn(Krypton-_(Comics), DC_Comics_Universe). This means that using the expanded noun helps the algorithm to more closely model the semantics of a sentence.

A negative side effect of this practice however is that the total number of predicates generated by Wanderlust is greatly increased, since all predicates containing nouns can additionally contain a number of adjectives. The distinct predicates **IsBigCityIn**, **IsSmallCityIn** and **IsLargestCityIn** for example are all extended forms of the predicate **IsCityIn**, which demonstrates how the use of extended wordpaths results in greater predicate dispersion. A solution to this problem is to automatically define predicates with expanded nouns to

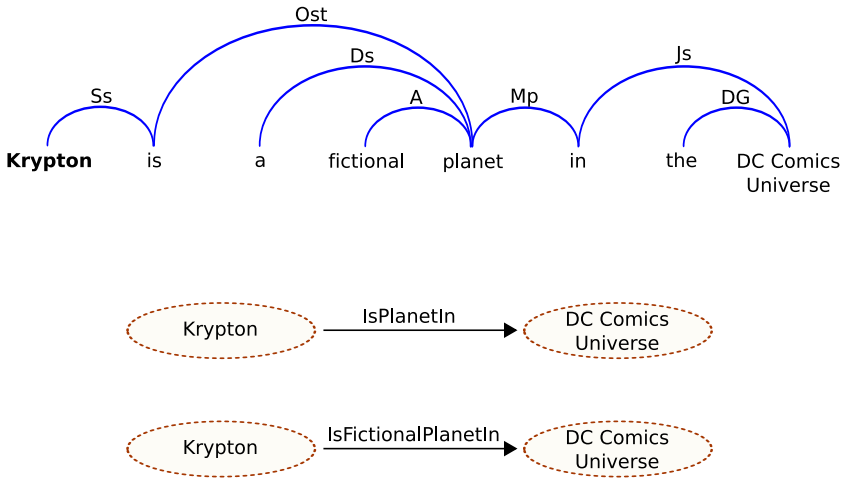

Figure 3.6: Linkage for the example sentence. The upper relation is extracted if expanded noun information is disregarded. The lower relation is extracted using the expanded noun.

be subproperties of predicates using normal nouns. The three aforementioned predicates can be defined as subproperties of IsCityIn, thereby reducing the negative effects of the greater predicate dispersion. The automatic definition of such subproperties uses the information given by expanded nouns and forms part of the normalization step of the algorithm. It is discussed in detail in 3.5.3.

Skipwords

As implemented in other information extraction systems such as TextRunner [3], Wanderlust drops relative pronouns from predicates. This enables the algorithm to correctly handle relative clauses. An example of this can be seen in the sentence:

> "The man went for a walk with Fluffy, who was his cat."

A linkage for the final part of the sentence is displayed in Figure 3.7. Using the linkpath {Mxsr, S**r, Ost} the relation WhoWas(Fluffy, Cat) is extracted. Because "who" is a skipword it is omitted from the path, leading to the relation Was(Fluffy, Cat).

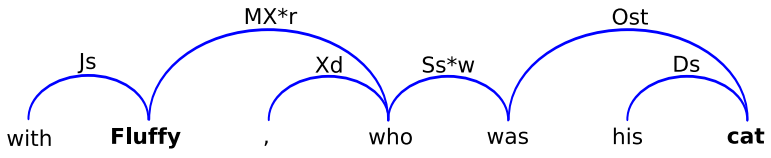

Figure 3.7: Portion of the example sentence highlighting how the linkpath connecting the entities "Fluffy" and "cat" touches a relative pronoun.

Particle Groups

Particle groups are combinations of terms which if touched by the linkpath must be completely included into the predicate. One example of such occurrences is in sentences using terms of negation, such as "not", as for example in the sentence:

"Switzerland is not a member of the EU."

A linkage for this sentence is displayed in Figure 3.8. The particle group consists of the words "is" and "not". The linkpath connecting Switzerland to EU is {Ss, Ost, Mp, Js} but passes through the particle group. Therefore, the valid relation IsNotMemberOf(Switzerland, EU) is extracted. Without the particle group the false relation IsMemberOf(Switzerland, EU) would be found.

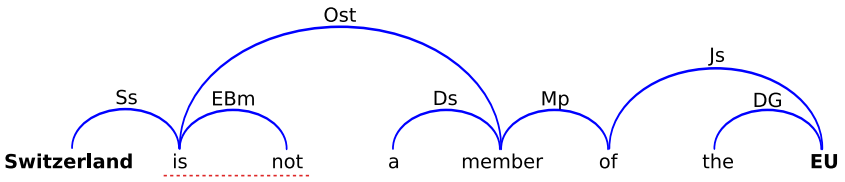

Figure 3.8: Linkage for the example sentence. The particle group is underlined.

Another example of particle groups are fixed expressions handled by the Link Grammar Parser. These expressions have individual link types which begin with "**ID**". An example of such an expression can be found in the sentence:

"Psychiatrists are referred to as Shrinks."

In this sentence, as can be seen in Figure 3.9, the terms "referred" and "to" form a particle group. Because of this, the relation AreReferredToAs(Psychiatrists, Shrinks) instead of the false relation AreReferredAs(Psychiatrists, Shrinks) is found.

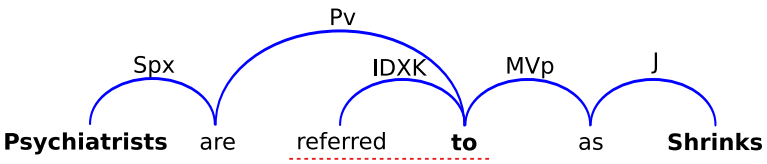

Figure 3.9: Linkage for the example sentence. The particle group is underlined.

Particle groups are identified using link types. If two terms are connected with certain link types, they are identified as particle groups. All particle groups identified within this thesis are accordingly marked in the overview of link types in Appendix B.2.

3.3.3 Coreferences

Coreferences in linguistics are multiple references to the same referent (or entity as in the sense of this thesis) using different terms. Coreferences may be synonyms, but also personal pronouns ("he", "it"), demonstrative pronouns ("this",

"that") or more general terms for a specific entity. In the following, the concept is illustrated on an example text.

> "Berlin lies at the Spree. The city is known for its numerous beach bars along the river."

The term *city* references the entity *Berlin* from the previous sentence. Likewise, the term *river* references the entity *Spree*. It can be seen from this example that coreferences may span over sentences. Because Wanderlust analyzes each sentence independently, such information is lost. The second sentence alone for example is empty of information if the references of the words "city" and "river" are unknown. Resolving these references is called coreference resolution [11], which would transform the text into the following:

> "Berlin lies at the Spree. Berlin is known for its numerous beach bars along the Spree."

In this case both sentences can be viewed individually and correctly analyzed by Wanderlust. The resolution of coreferences however is a separate topic in computerlinguistics which to date has not been sufficiently solved [11]. Main challenges for this problem include a lack of a knowledge base. A machine cannot replace the terms "city" and "river" in the example without having some sort of knowledge base which asserts that Berlin is a city and that the Spree is a river. Such a knowledge base is the intended goal of this thesis, meaning that somewhat ironically the result of this thesis might help solve a problem for something it needs itself. Without well functioning coreference resolution however, the algorithm faces challenges which are overcome by heuristic measures severely restricting the extraction of relations for terms which might be coreferences.

To illustrate the effects of unhandled coreferences consider the semantics expressed in the second sentence alone. The terms "city" and "river" have page links pointing to the identically named pages and are in this context false entities. The term "beach bars" however is not because it does not refer to a different referent. The sentence in Wikipedia markup thus looks as follows:

> The [[city]] is known for its numerous [[beach bars]] along the [[river]].

Unhandled coreferences cause Wanderlust to find IsKnownFor(City, Beach Bars), a relation falsely connecting the entity *city* (which describes the general concept of a city) to the entity *beach bars* using the predicate IsKnownFor. The sentence actually states IsKnownFor(Berlin, Beach Bars). As a consequence, the algorithm does not permit terms which may be coreferences to be used as subjects for relations. To identify such terms, a heuristic is applied which classifies all entities as either *concept entity* or *unique entity*. Their definition is as follows:

- Concept entity: Describe general concepts which may be instantiated by proper nouns. Examples of concept entities include *city*, *river*, *language* or *book*. All nouns which start with a lower case letter are classified as concept entities.
- Unique entity: Represent instances of concept entities and cannot be instantiated. Examples of unique entities may include *Berlin* (instantiation of city), *Spree* (river), *Latin* (language) or *Die Vermessung der Welt*

(book). All nouns which start with an upper case letter are classified as unique entities.

Relations may only be formed between two unique entities or from a unique entity to a concept entity. A concept entity may not form a relation with neither a unique nor a concept entity. This is illustrated in Figure 3.10.

Figure 3.10: Concept entities are symbolized by round, unique entities by rectangular boxes. Blue arrows with arrowheads show possible relations. Red arrows with flat heads show unallowed relations.

This measure serves to reduce the error of coreferences, while at the same time greatly reducing recall. While the latter is somewhat unsatisfactory, this solution to the problem was found to be the only possibility within the scope of the thesis and the current quality of coreference resolution.

Because of coreferences, the following two relations are dismissed from the relations list found in the previous section (see Figure 3.5): Is(City, Essen) and In(City, Germany), the latter already being dismissed in 3.4.1 because of the predicate being initiated by a preposition.

3.4 Validation

This section deals with the fourth step of the Wanderlust algorithm. In the preceding step, a list of relations is generated for a given sentence with little guarantee to their validity. This step performs a grammatical analysis in order to distinguish between valid and false relations and filter out the latter from the list. The grammatical features analyzed are discussed in this section. As introduced in 1.2.2, the linkpath is used to characterize the nature of the grammatical relationship between subject and object. Validation of relations is mainly based on this feature. In addition, the *POS-path* is considered, which is defined as the POS-tags of the words in the wordpath. This adds data on the words to the information given by the linkpath. While POS-tag information is used by Weitblick, it is not used by Wanderlust validation. The already present information is nonetheless stored to enable future analysis.

The quality of parse information provided by the Link Grammar Parser is also considered. This is important because it represents the confidence the algorithm may have in the output of the parser. If the number of skipped words is too high for example, the meaning of the sentence may change which raises the risk of falsely extracted semantics. If the score or the ordering of the parse is not low enough, it means that the parser itself is not confident about the linkage it produced. Quality of parse information must be evaluated independently from the linkpath, as even a perfect linkpath may produce false relations if it is extracted from a low quality linkage.

The feature vector of each relation thus includes the linkpath and the quality-of-parse information. This information has been determined as the best features

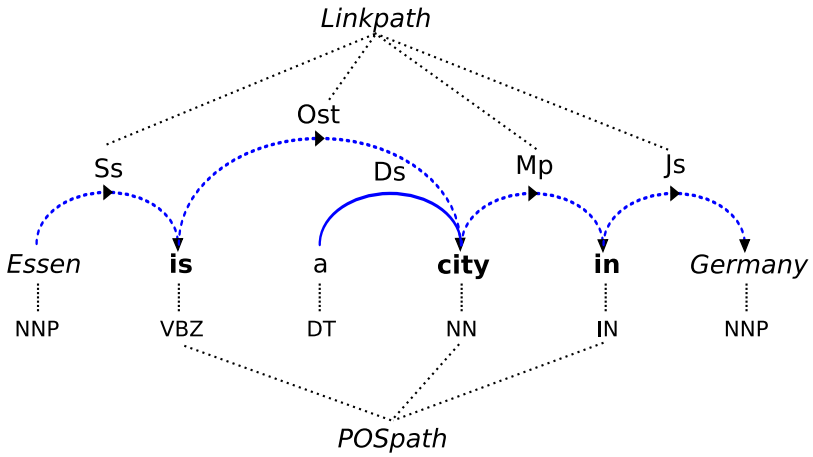

Figure 3.11: Example analysis of the relation IsCityIn(Essen, Germany). The linkpath is {Ss, Ost, Mp, Js} and the POSpath is {VBZ, NN, IN}.

to be used to distinguish valid from false relations. They have been selected after exploratory data analysis on a manually generated large training corpus of valid and false relations has shown positive results for these features. Details on the generation of the training data, the analysis and its results are given in Chapter 4. Important is that relations are filtered out according to two criteria: First, the quality-of-parse information must indicate a high level of quality. Second, the linkpath must represent valid relations in the highest number of cases. For this purpose a *coefficient* is computed from the training data representing the algorithm's confidence in the linkpath. For the use case of generating a semantic wiki from the English Wikipedia corpus, Wanderlust was run with the following parameters for validation:

- *Skipped words*: No more than one word in the linkage may be skipped.
- *Ordering*: The linkage number must be lower than 2.
- *Linkpath*: The linkpath must have a coefficient of 0.5 or higher.

The linkages for the example sentence considered in Figures 3.4 and 3.11 both have perfect scores, meaning that no relations are dismissed because of quality-of-parse information. The minimum linkpath coefficient is only met by two relations, namely Is(Essen, City) and IsCityIn(Essen, Germany) as illustrated in 3.12. All other relations are dismissed.

All relations which pass the criteria are considered valid by Wanderlust and entered into the SMW database. In the following step, the algorithm attempts to generate subproperties in order to add a layer of logic to the model of knowledge.

3.5 Normalization

This section deals with the final step in the Wanderlust algorithm. All false relations have been dismissed, leaving a list of valid relations. The problem of predicate diction has so far not been addressed. Wanderlust can generate

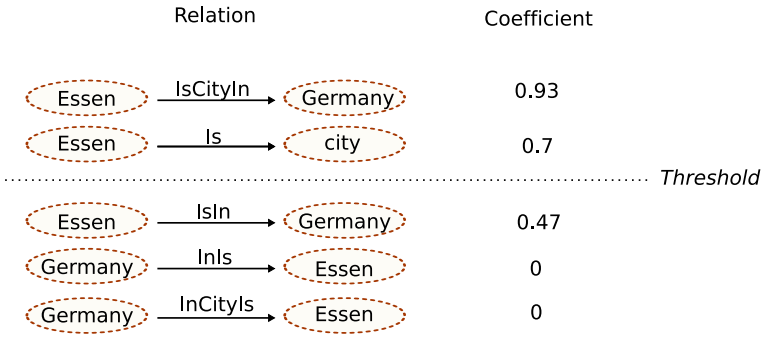

Figure 3.12: This figure shows five relations of the example sentence and their coefficients. Only the top three are passed to this step of the algorithm, the lower two actually already being filtered out because of the predicate being initiated by a preposition. Only two of the coefficients lie higher than the required threshold, which means that all relations below the dotted line are erased from the list.

a large number of distinct relation types, leading to problems noted in Section 1.2.3. In order to minimize the dispersion of predicates an attempt is made to group predicates into hierarchies, thereby facilitating standardization for purposes such as semantic querying. While there may exist great possibilities of normalizing predicates according to synonyms and semantics, normalization performed by Wanderlust is limited to heuristics. State-of-the-art rewriting of predicates presents a challenge to itself and lies beyond the scope of this work.

Using subproperties, the algorithm automatically defines certain predicates to be subtypes of others according to heuristics introduced in this section. A distinction is made between three types of normalization.

3.5.1 Temporal Groups

Predicates in the temporal group are normalized by rewriting and dropping verbs and auxiliary verbs which contain nothing but information on the verb tense. Consider for example the verb "to be" in the English language, which is present in a large number of predicates, such as IsCapitalOf, WasMadeFrom or HasBeenLocatedIn. In fact, many relations start with this verb in one of its tense forms, which carries little information but the fact whether the relation refers to something which still exists, or something which used to be. This information shall be referred to as the *continuity* of a relation. Consider for example the predicates WasCapitalOf and HadBeenCapitalOf. Both essentially carry the same information and continuity (being that the relation IsCapitalOf no longer exists), but are written differently. Normalization can therefore be performed by defining the second predicate to be a subproperty of the first.

There are many verb forms and auxiliary verbs which can be automatically written into subproperties this way. Refer to Figure 3.13 for an illustration of this. The first step is reducing all temporal information to either "Is", "Are", "Was" or "Were". The original predicates are defined as subproperties of the reduced ones. In a second normalization step, the continuity information is dropped altogether. All previous predicates are set as subproperties of this predicate. This enables a user to choose between posing queries with or without

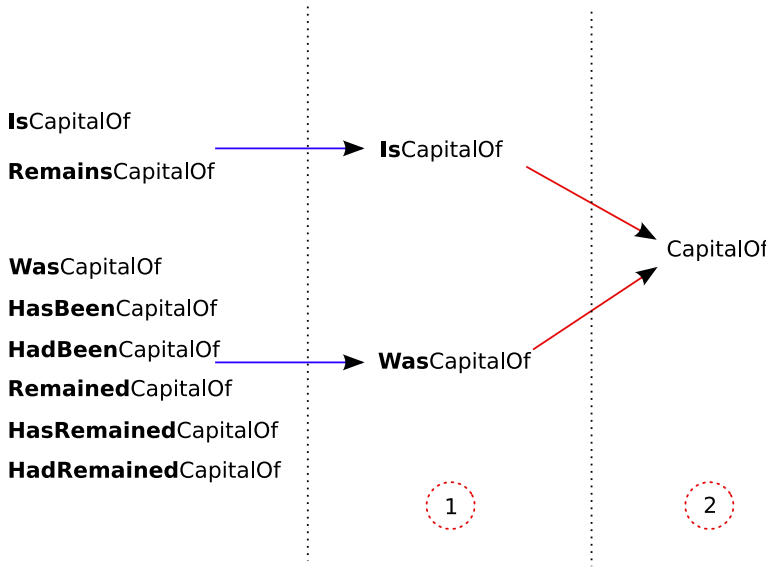

Figure 3.13: Example of temporal normalization for the CapitalOf predicates. The parts of the predicate which give continuity information are highlighted bold. In the first normalization step, continuity information is reduced to IsCapitalOf or WasCapitalOf. In the second step, all continuity information is dropped to form the relation CapitalOf.

continuity information. As can be seen in the figure, queries for CapitalOf will therefore also automatically query for IsCapitalOf and WasCapitalOf, as well as their subproperties. With one query a large number of possible phrasing of predicates are therefore automatically covered.

Temporal normalization is performed by following a manually composed template which is applied to verbs. An overview for the first step of temporal normalization is given in 3.1. In the second step "Is", "Are", "Was" and "Were" are also dropped.

Table 3.1: Temporal normalization.

Replacement	Removed
Was	HasBeen HadBeen HadBecome Became Remained HasRemained HadRemained
Were	HaveBeen HaveRemained
Is	Remains Becomes HasBecome
Are	HaveBecome

3.5.2 Inheritance Groups

Another form of normalization are inheritance groups which are derived from predicates using expanded nouns. By dropping the adjective and number word modifiers one by one, a chain of subproperties can be generated. This shall be demonstrated with the example sentence "*Dirk is a very good friend of Fred*". Its

linkage and expanded noun are given in Figure 3.14. The relation IsVeryGood-FriendOf(Dirk, Fred), using the expanded noun "very good friend", is found.

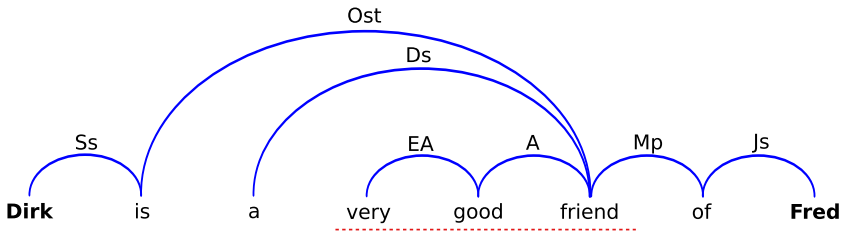

Figure 3.14: Linkage for the example sentence. The expanded noun is underlined.

The algorithm proceeds to shorten the expanded noun one word at a time until the noun is reached. For each shorter version of the expanded noun, a predicate is constructed to which the longer version is a subproperty. For the example this means that the predicate IsGoodFriendOf is constructed, to which IsVeryGoodFriendOf is subproperty. In a second step the predicate IsFriendOf to which IsGoodFriendOf is subproperty is constructed. Figure 3.5.2 shows an example of both inheritance and temporal normalization.

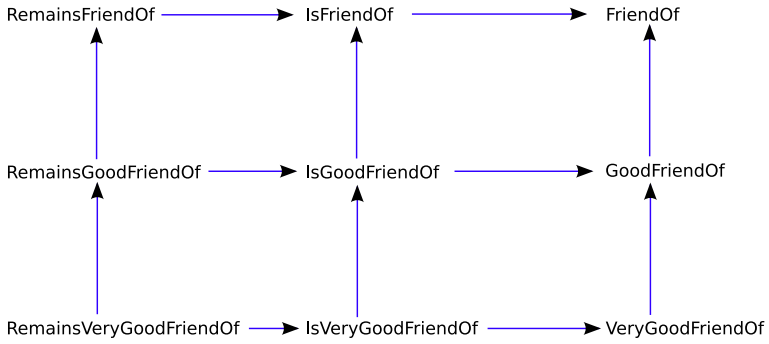

Figure 3.15: Figure representing normalization of the predicate RemainsVery-GoodFriendOf. Arrows upwards are temporal, arrows rightwards inheritance normalization. A total of 8 subproperties are generated for the predicate.

These two forms of normalization are the extent of Wanderlust's automatic generation of subproperties. The following section indicates additional possibilities.

3.5.3 Semantic Groups

Semantic groups are more complicated than temporal groups, because rather than the continuity of the predicate its semantics are considered. As previously mentioned, far reaching normalization of semantics is not performed by Wanderlust. One example of semantic normalization is the extraction of the LocatedIn predicate from predicates which imply that this relation exists. Consider for example the relation CityIn(Essen, Germany). This relation implies the relation LocatedIn(Essen, Germany). Because of this, a list of predicates which

imply the LocatedIn relation has been hand-crafted and used for normalization purposes.

The problem with the normalization of semantic groups is that no methodology is applied. The few groups which are normalized by Wanderlust have been identified by hand meaning that of the very large number of semantic groups only few are normalized. Two example semantic groups are listed in Appendices B.5 and B.6.

3.6 Summary

This chapter discussed the implementation of Wanderlust for the use case of generating a semantic wiki. Five principal steps of the algorithm were identified and illustrated in detail. The first step is the identification of entities in plain text, a process performed by Weitblick, an algorithm created specifically for this purpose. It has been shown how Weitblick uses the information given by Wikipedia page links to achieve this goal and how heuristics are applied for terms without page links.

After identification of entities, sentences are passed to Wanderlust, which performs a link grammar analysis (step two). It has been shown how sentences are modified beforehand to increase the chances of correct parsing by the link grammar parser. Using the parse, a list of relations is constructed using wordpaths between entities as predicates (step three). A number of wordpath modifications have been illustrated which help the algorithm to correctly model semantics.

In step four, validation using linkpath coefficients is performed to filter out all false relations. The validation procedure was illustrated. The normalization procedure used in the thesis, step five of the algorithm, was explained. It was shown how heuristics are applied to automatically generate subproperties for predicates, thereby addressing problems caused because of predicate diction.

In the next chapter, the determination of the parameters for validation is discussed.

Chapter 4

Validation

This chapter covers one of the main challenges of the thesis, namely the distinction between valid and false relations extracted by Wanderlust. This problem is referred to as *validation* and is encountered by the algorithm in step four (see 3.4). At the onset of the thesis, the initial theory was that the grammatical information given by the linkpath indicates whether a relation may be extracted from a linkage or not. Support for this theory was sought by manually analyzing linkage diagrams and constructing a small set of hand-written validation rules based on linkpath information. The hand-crafted validation procedure performed reasonably well to support the theory that some linkpaths are generally valid (referred to as "good" linkpaths) while others (referred to as "bad" linkpaths) are not. See Figure 4.1 for an illustration of this principle.

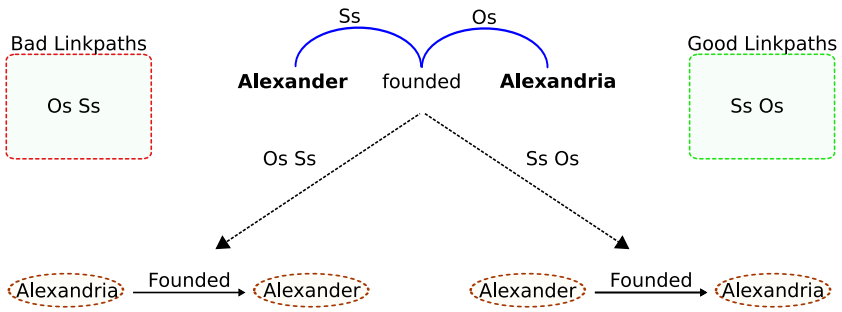

Figure 4.1: Illustration of the good and bad linkpath principle. Two relations are extracted. Founded(Alexandria, Alexander) is extracted with the linkpath {Os Ss} which is in the list of bad linkpaths (top left). Therefore it is classified as false. The relation Founded(Alexander, Alexandria) is classified as valid because it was extracted with the linkpath {Ss Os} which is in the list of good linkpaths (top right).

In order to find a more complete list of valid linkpaths it was decided to annotate a corpus of Wikipedia pages with semantic relations for training and testing purposes. Using this corpus an analysis was performed as to which linkpaths might be considered "good" and which "bad". Using the annotated

pages as a training set, coefficients were generated for all linkpaths. This was done by analyzing the results found by each distinct linkpath; the number of true positives was divided by the number of total positives found for each linkpath. The resulting *linkpath coefficient* states the proportion of true positives among all positives returned; a linkpath coefficient of 1 states that all relations found with this linkpath are true positives, while a coefficient of 0 states that all relations are false positives.

Based on preliminary observations it was hoped that these coefficients would be unequivocal, meaning either very high or very low, since this would allow for linkpaths to be classified as either "good" or "bad". If, on the other hand, too many linkpath coefficients are in a medium range (around 0.5) such a classification becomes more difficult, undermining the theory that the linkpath alone as a feature is enough to determine whether a relation is valid or false.

The outline of this chapter is as follows: In 4.1, the annotation procedure used to generate the annotated corpus is explained. Some measures taken to ease the workload are introduced there. In 4.2, the linkpath coefficient and its use for validation is analyzed. A number of problems are observed, which put into question the theory that the linkpath alone may be used for validation and instead suggest the need for additional features. This leads to the test set analysis in 4.3, where linkpath coefficients are trained on a training set and used with a test set of annotated data. The results are analyzed with regards to the difficulties noted in 4.2. Based on these results, additional features and alternative validation mechanisms are discussed in 4.4. Taking into account the considerations of Sections 4.1 to 4.4, the exact parameters and technique for validation are chosen and stated in Section 4.5.

4.1 Annotation

This section discusses the generation of a corpus of Wikipedia pages annotated with all relations that can be found using Wanderlust. Because such a corpus needs to be sufficiently large, a number of existing semantic corpora and knowledge bases were considered. None however have been found to meet the criteria to be usable for the purpose of this thesis. Wanderlust is intended to find all information that is explicitly stated within a sentence. This means that the training corpus must consist of sentences that are annotated with arbitrary relation types conveying all explicitly stated information. This is not given with the results of related work (see Section 2.1 for an overview of related work) or existing knowledge bases such as ConceptNet [24], which consist only of knowledge, but not annotated sentences. Furthermore, [2, 38, 24] only have a limited set of relation types. While [12, 3] generate arbitrary relations, no disambiguation of entities is performed and precision levels of the results are judged not high enough for the use as training data.

In light of the linguistic approach chosen in this thesis it was therefore decided to manually generate a corpus of annotated sentences. A batch of Wikipedia pages were annotated with all subject-predicate-object triplets that can be found using Wanderlust, thereby producing a project specific gold standard. The corpus consists of a number of Wikipedia pages and a knowledge base reflecting all information stated within them in the form of *facts* (relations known to be true) and *antifacts* (relations known to be false). In order to ease the

workload and facilitate the generation of a reasonably large annotated corpus, the annotation technique used for this purpose makes use of project specific requirements and employs a bootstrapping approach. These two aspects of the annotation technique are illustrated in 4.1.1 and 4.1.2 respectively.

Using this method a corpus containing 10,585 distinct semantic relations was generated. The corpus is used to generate linkpath coefficients as well as for the analysis of additional features. By splitting the corpus into a training and test subset, the efficacy of a validation technique can be automatically evaluated.

4.1.1 Relation Confirmation

Since the annotated corpus is specific to the thesis, tools were written which perform the first three steps of the Wanderlust algorithm for a given sentence or an entire page. In each sentence, entities are identified, predicates by virtue of the link grammar parse extracted and a list of all pairs of entities and their predicates presented. The entire unvalidated relation list found in this step (covered in 3.3) is presented to a user who then must confirm all relations in the list which are valid. All non-confirmed relations are automatically assumed to be false. This reduces the task of finding and annotating semantics in sentences to confirming relations. The tools written for this purpose are illustrated in Appendix A.

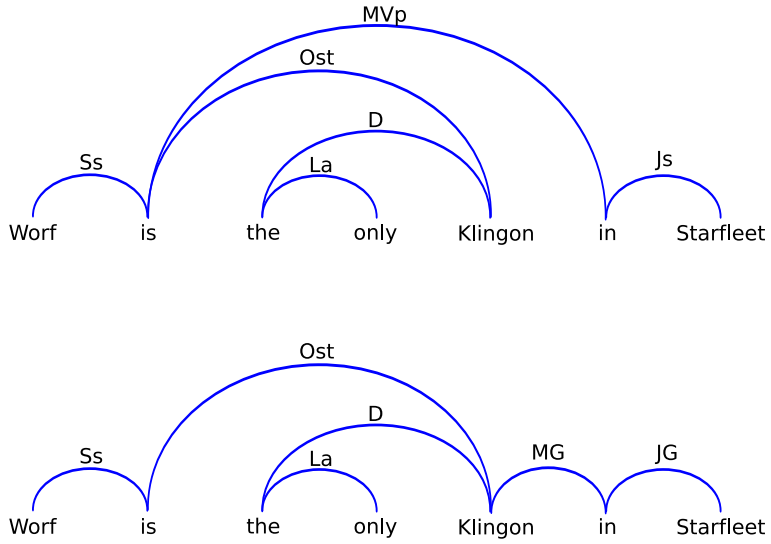

Figure 4.2: Linkages for an example sentence.

Take for example the sentence "*Worf is the only Klingon in Starfleet*", which has two linkages as shown in Figure 4.2. Given that the terms *Worf*, *Klingon* and *Starfleet* are entities, a total of 5 relations are extracted, which are displayed in Figure 4.3. Note that all relations with the entity *Starfleet* as subject were dismissed in Wanderlust pre-filtering because their predicates are initiated with a preposition, which is not allowed (for example InIs(Starfleet, Klingon)). The annotator must mark those relations considered valid. All others are automatically dismissed as false.

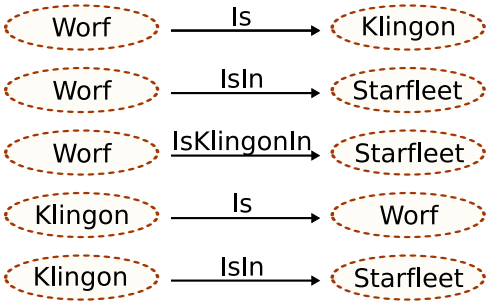

Figure 4.3: All unvalidated relations as generated by Wanderlust.

Upon selection of valid relations, the tools perform two steps. First, valid relations are entered as facts into the factbase and false relations are entered as antifacts into the antifactbase. Second, the grammatical features of each relation are extracted into a feature vector which is tagged either as positive or negative example and saved in a database of feature vectors which is used for the bootstrapping technique described in 4.1.2.

4.1.2 Bootstrapping

Bootstrapping in machine learning is the use of a classifier using its own predictions to train itself, a commonly used technique for semi-supervised learning [44]. Principles from this approach are incorporated into the annotation technique used in this thesis in order to ease the workload of manual annotation. The annotated corpus is periodically analyzed and the results incorporated as pre-filters into the annotation tools. If for instance a certain type of linkpath yields a false result in a high number of cases, and the total number of observed instances of this linkpath is reasonably high, then all subsequent relations extracted using this linkpath can be safely dismissed. This means that as more and more relations are annotated, a rising number of relations can be safely dismissed without asking the annotator for confirmation. The idea of bootstrapping as used in this thesis is to filter out all false relation types which lie below a certain confidence threshold, thereby greatly reducing the workload for the annotator. As annotation progresses, the threshold is slowly raised. Valid linkpaths will stay above this threshold to the end of the analysis.

The danger of bootstrapping is that local phenomena encountered only in the first batch of the training corpus could be mistaken as universally valid and therefore lead to wrongful pre-filtering. Once any linkpath falls below the threshold, further occurrences of relations with this linkpath are no longer offered to the annotator, meaning that this linkpath is dismissed for good. Yet another difficulty is the fact that during annotation it is unclear whether the linkpath coefficient as sole feature for validation would be sufficient. A bootstrapping approach based only on the linkpath could be distorting given the possibility that other features might be needed to be taken into account.

Because of the above considerations, bootstrapping was used "lightly". The first linkpath coefficient threshold was set very low at 0.1 and only applied if this relation type had been encountered more than 50 times. Gradually it was raised to 0.2, but not higher because of uncertainties concerning the validation tech-

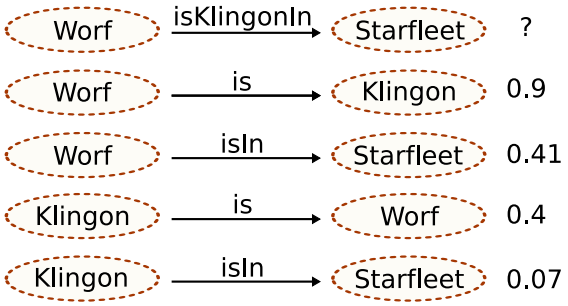

Figure 4.4: The relations are rated according to the linkpath coefficient introduced in 3.4. The uppermost relation has a linkpath which has not been observed often enough in order to ensure representativity. This means that this relation cannot be automatically dismissed by the bootstrapping mechanism but rather must be confirmed or rejected by the annotator. Below, all other relations are evaluated by the algorithm. For the topmost relation, the algorithm is most certain of validity, for the lowermost the least. With a threshold of 0.1, the lowermost relation is automatically rejected. By raising the threshold, more relations are affected.

nique. Nevertheless, the filter helped ease the workload of manual annotation by eliminating those relations which were very certain to be false.

The approach is demonstrated on the example sentence discussed in 4.1.1. See Figure 4.4 for the evaluated relations and how many are automatically dismissed due to bootstrapping.

4.2 Linkpath Coefficient

This section analyzes and discusses the feasibility of using the linkpath coefficient as validation technique. In Section 4.2.1, statistical data from the training set is analyzed. While the results of the analysis do encourage the use of the linkpath coefficient for validation, the numbers suggest that additional features may be needed to achieve reasonable precision. In 4.2.2, sources of errors which interfere with validation via linkpaths are listed and examined.

4.2.1 Analysis

This section analyzes the usability of the linkpath for validation. The distribution of linkpaths and their coefficients, as well as precision and recall values are computed. Two main issues are of interest regarding the linkpath coefficient. First, a very high number of distinct linkpaths is possible considering that each linkpath consists of up to 6 link labels which in turn may be one of over 100 link types. Even if not all sequences of link labels in terms of grammaticality are possible, this still amounts to a very high number of linkpaths. If the total number of distinct linkpaths is too high, a massive annotation effort would be required to compute all coefficients, which would lie outside the scope of this thesis. A second issue that is analyzed in this section is the theory that linkpaths can be classified as either “good” or “bad”, which requires a large part of

coefficients to be either very high or very low.

The analysis is performed on a corpus of 48 large pages. In total there are 10,585 relations, of which 2,350 are marked as valid and 8,235 as false. Note that the relatively high ratio of valid relations is caused by the hard-coded pre-filtering steps laid out in 3.3 which eliminate a large number of false relations already in step 3 of the Wanderlust algorithm. A very high number of automatically dismissed relations are therefore not part of this figure.

Distribution of Linkpaths

In this section, the distribution of linkpaths in occurrences per linkpath is analyzed. A total of 3,331 distinct linkpaths are found in this set of data. The topmost common linkpaths ordered by number of occurrence are illustrated in Figure 4.5. A small number of linkpaths occur very frequently, while a very large number is rare. The topmost common ten linkpaths, for example, encompass 2,191 and therefore more than one fifth of all 10,585 relations. The top 20 and 30 linkpaths make up 2,815 and 3,147 of all relations respectively.

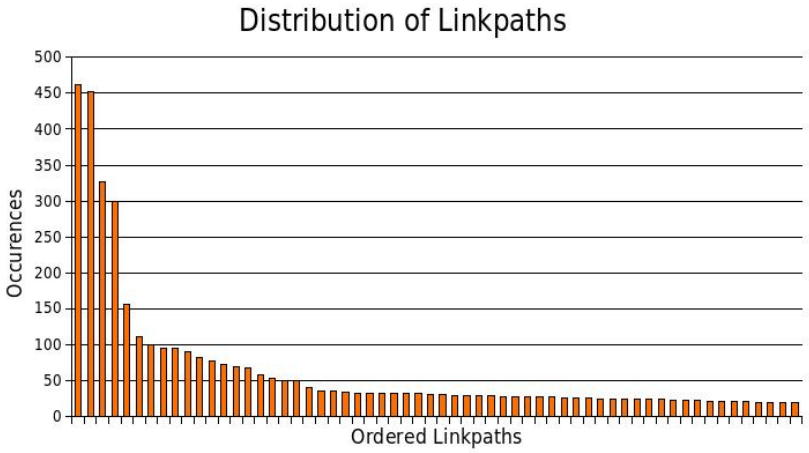

Figure 4.5: Distribution of linkpaths, ordered along the x-axis by number of occurrence. The topmost common 60 linkpaths are displayed in this graph, which slowly approaches zero along the x-axis. This graph shows that a small group of linkpaths occur very frequently, while a large number are very rare.

A number of observations should be noted when regarding the high number of distinct linkpaths and the percentage of valid relations found with the topmost 10 to 30 linkpaths. First, the very high number of linkpaths is significantly diminished when limiting the maximum linkpath size to 5, meaning that a predicate may only consist of a maximum of 4 terms. This reduces the number of linkpaths to 2,571, while having very little effect on the number of valid relations found. Refer to Table 4.1 for an overview of the effects of dismissing all linkpaths of length 6. When including such linkpaths, 2,350 valid and 8,235 false relations are found, yielding a ratio of 0.22 of valid relations within all found relations. Excluding such relations yields 2,215 valid and 7,079 false relations, thereby improving the ratio to 0.24. The ratio of relations of size 6 is listed under “Difference” in Table 4.1 with 0.1, well below the ratio average.

Table 4.1: Linkpaths of size 6

Case	#Valid	#Nonsensical	Ratio (good / total)
Include	2350	8235	0.22
Exclude	2215	7079	0.24
Difference	135	1156	0.1

This number supports the decision to dismiss or neglect relations of such length. Another factor supporting this decision is the observation that relations of size 6 are of dubious usability, given that semantic relations are preferably short and succinctly put. In Figure 4.6, some examples for valid relations of length 6 from the training corpus are given in order to illustrate this.

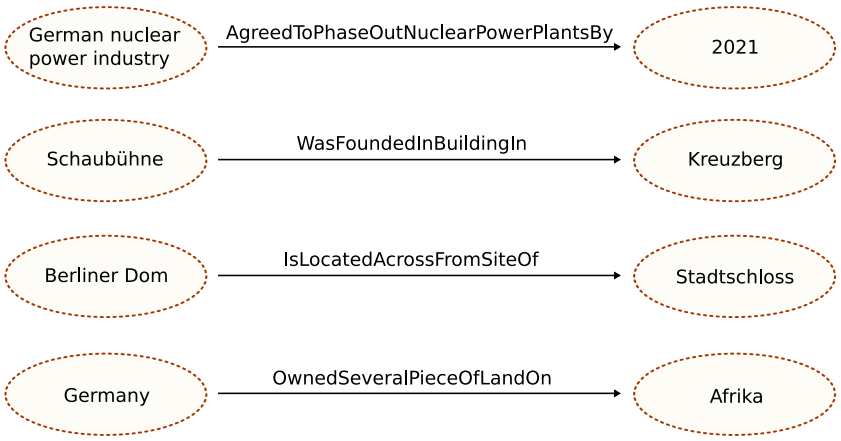

Figure 4.6: Example relations with linkpaths of length 6. Note that some relations contain more than 5 words because of expanded wordpaths (see 3.3.2) and particle groups (see 3.3.2).

From these examples it can be gathered that predicates of this size are too long to be useful in terms of semantic querying or ontology building. Combined with the observations of Table 4.1 this makes a strong case for neglecting linkpaths of length 6. It should also be noted that in the top 60 linkpaths by occurrence as displayed in Figure 4.5 only one linkpath is of length 6, while a large number of such linkpaths are very rare in occurrence. The distribution with or without such linkpaths is therefore essentially the same.

Still, at 2,571 the number of distinct linkpaths is very high. If each coefficient would require a minimum of 50 examples to be computed, a minimum of 128,550 relations would have to be annotated. The corpus used here consists of 10,585. However, the distribution of linkpaths is such that in top 140 linkpaths a total of 4,942 and therefore nearly half of all relations are included. If the topmost common linkpaths are also those which yield a substantial portion of all *valid* relations, then concentrating only on them would suffice. Whether this is the case is analyzed in the next subsection.

Possible Recall

This section analyzes the possible recall of valid relations from the training corpus using linkpaths. In this section, only linkpaths with at least 10 example relations are considered, since all other linkpaths have too few occurrences in order to be considered representative. Given that linkpaths with few examples are a large majority of all distinct linkpaths found in the training corpus, the question which must be answered at this point is whether they can be dismissed from the analysis. This is important because the alternative of annotating enough pages in order to find a representative amount of examples for each of the over 2,500 linkpaths is hardly feasible.

In Table 4.2, an overview is given of how many linkpaths from the training corpus have a minimum number of occurrences, and how many of the valid linkpaths are encompassed within these linkpaths. The first column lists the results for the topmost occurring linkpaths, which have been observed in the annotated corpus a minimum of times as indicated by the forth column. As can be seen in this table, there are 20 linkpaths with at least 40 examples with which over half of all valid linkpaths are found, meaning that the top 20 linkpaths have a maximum possible recall of 0.52. By decreasing the number of minimum occurrences, the top 32, 46, 60, 84 and 140 are regarded respectively, raising recall. With the top 140 linkpaths, 74% of all valid relations are found.

Table 4.2: Number of valid relations for the topmost occurring linkpaths.

Topmost linkpaths	#Valid	Recall	Min #
Top 20	1218	0.52	40
Top 32	1301	0.55	30
Top 46	1435	0.61	25
Top 60	1483	0.63	20
Top 84	1595	0.68	15
Top 140	1749	0.74	10

Therefore, with the top 140 linkpaths Wanderlust has the potential of achieving a recall of 0.74. This number is considered acceptable, even if a higher number is desired. However, some observations must be pointed out in regard to this result. First, it should be noted that of the 551 relations which are not found using the top 140 linkpaths, a total of 88 are found with linkpaths of length 6 and therefore of dubious use anyway. A portion of those relations that are not found using the topmost common linkpaths may be explained through errors in the Link Grammar Parser, the errors being the cause for the "rarity" of the linkpath. This is believed to account for some part of the 551 relations not found using the topmost 140 linkpaths. The recall in terms of relations that *can* be extracted using Wanderlust is therefore believed to be higher than the number given in Table 4.2.

Based on these observations it had been decided that focusing on a small subgroup of linkpaths and developing heuristics for the large quantity of rare linkpaths would be sufficient for relation extraction in terms of possible recall. The use of coefficients built on the topmost occurring linkpaths and first predictions of recall and precision values are given in the next subsection.

Coefficient Potential

This section discusses the linkpath coefficients and their possible use for relation validation. The idea is to neglect the large majority of rare and instead focus on the topmost common linkpaths. In Table 4.3, a joined coefficient is calculated for the most common 20 to 140 linkpaths.

Table 4.3: Combined coefficients for topmost linkpaths.

Topmost linkpaths	#Valid	Total	Coefficient
Top 20	1218	2815	0.43
Top 32	1279	3178	0.4
Top 46	1434	3510	0.41
Top 60	1478	3812	0.39
Top 84	1574	4188	0.38
Top 140	1704	4772	0.36

As can be seen in the table, the joined coefficient of the topmost linkpaths is always around 0.4. With the top 20 linkpaths a total of 2,815 relations are extracted of which only 1,218 are valid. The joined coefficient of 0.43 is well below the desired high precision level. When looking at the individual coefficients of the top 60 linkpaths it can be observed that the whole range from 0 to 1 is represented (see Figure 4.7). This stands in contrast to the initial hope of either very high or very low coefficients.

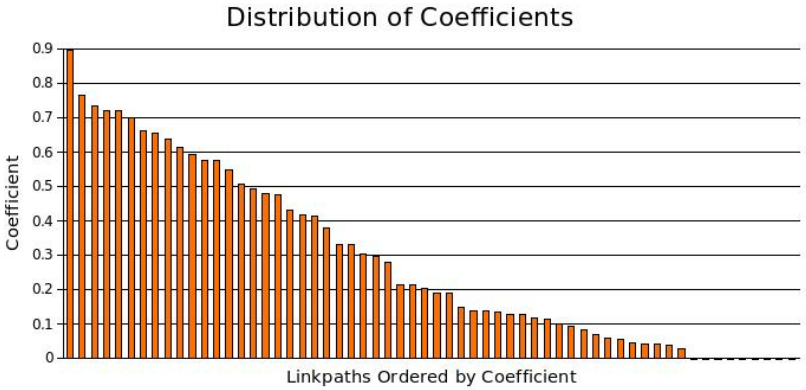

Figure 4.7: Coefficients in the top 60 linkpaths.

Note that the last 9 of the most common linkpaths have a coefficient of 0 and account for 244 relations. A total of 20 linkpaths lie below the threshold of 0.1 accounting for 653 relations. By dismissing all linkpaths with low coefficients the overall precision can be raised, which however comes at a cost for recall. The idea is to classify linkpaths above a certain coefficient as "good" and all others as "bad". Only good linkpaths will be used for validation. Table 4.4 shows recall and precision levels for the top 60 linkpaths and different coefficient thresholds.

The recall level is measured against all 2,350 valid relations in the entire corpus, not just against the valid relations which can be found with the top

Table 4.4: Precision and recall levels for the top 60 linkpaths.

Threshold	#Positives	#True positives	Precision	Recall
0.7	754	561	0.74	0.24
0.6	1345	949	0.71	0.40
0.5	1539	1055	0.69	0.45
0.4	1885	1212	0.64	0.52

60 linkpaths. As can be seen, the tradeoff is not satisfying in either respect. Raising the threshold to 0.7 raises precision to a barely acceptable 0.74, but leaves recall at 0.24. Lowering the threshold to 0.4 leaves recall at a passable 0.52 but lowers precision to an unacceptable 0.64.

These numbers *do not* support the theory that the linkpath coefficient alone is sufficient for validation. The hope that linkpath coefficients would be either very high or very low has not fully come true. While good linkpaths have failed to appear in coefficients, many bad linkpaths can clearly be named. Of the 3,331 linkpaths, a total of 2,881 have coefficients lower than 0.1. The question therefore is why generally good linkpaths fail sometimes in the validation task. This is analyzed in 4.3.

Conclusions

The analysis brings forth arguments for and against the use of linkpath coefficients for validation. On the positive side, the analysis of linkpath distribution by number of occurrence indicates that a small group of linkpaths is very common, while a large majority is very rare, meaning that with the topmost common linkpaths a reasonable recall level can be achieved. These facts support the decision to focus on a set of common linkpaths, while developing heuristics for the rest, allowing linkpath coefficients to be generated with a feasible annotation effort.

However, a clear classification of linkpaths as either "good" or "bad" cannot be done, given that while there are many linkpaths that can be clearly dismissed, too few linkpaths have very high coefficients. This indicates that additional features may be needed. The following section identifies sources of errors, i.e. situations, when the linkpath alone is insufficient to decide whether a relation is valid or false.

4.2.2 Problems

Results from the previous section put into question the idea of using the linkpath coefficient for validation. While there are some linkpaths with reasonably high coefficients and a large number of linkpaths with coefficients close to zero, the portion of common linkpaths with coefficients in a medium range is very large. This suggests that the information given by the linkpath is not enough, but rather that additional information is required. This section lists reasons for generally good linkpaths finding false positives and generally bad linkpaths finding false negatives. Each of the following causes of errors has been observed during the annotation process:

- Parse errors
- Entity errors
- Context errors
- Incomplete object errors

Each of the error types may be seen as a "layer of distortion", because for the most part they cannot be detected with information given by the Link Grammar Parser, but interfere with the accuracy of the linkpath coefficient. This means that they can cause good linkpaths to produce false positives, thereby lowering linkpath coefficients for good linkpaths. Similarly, they can cause bad linkpaths to have higher coefficients. In the following subsections, each error is discussed separately. It is important to note that most of the error sources do not challenge the initial theory of the thesis. In Section 4.3, a tentative analysis on the distribution of error classes is performed on a test set of annotated sentences, providing statistical data.

Parse Errors

One problem is the degree of distortion caused by errors in the Link Grammar Parser. A good linkpath may produce false positives in a sentence that has not been correctly understood by the Link Grammar Parser. Similarly, a bad linkpath may produce false negatives if applied to an incorrect linkage. Parse errors therefore affect both the accuracy and performance of linkpath coefficients generated from an annotated corpus.

The Link Grammar Parser has three quality-of-parse variables which it uses to "gauge" the accuracy of a linkage, as introduced in 2.4. During annotation, the variables *skip* and *linkage number* were tentatively kept low in order to limit the impact of parser errors and the annotation workload. All linkages in which more than two words of the sentence were skipped were dismissed. The same was done for linkages with a linkage number higher than 2.

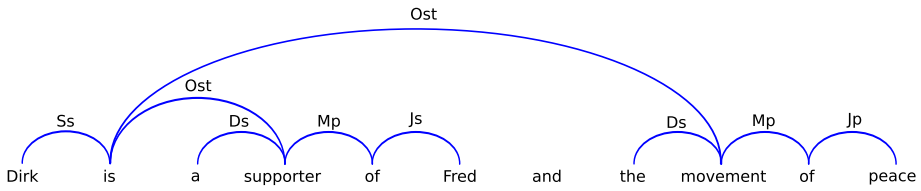

Figure 4.8: Example sentence which has not been correctly understood by the Link Grammar Parser (but has perfect quality-of-parse values). The link between "is" and "movement" is false. The term "movement" should correctly be linked to the first "of". Because of the error, the false positive Is(Dirk, Movement) is found using the good linkpath {Ss, Ost}. This parse has nonetheless perfect quality-of-parse variables.

However, a high number of parse errors were observed nonetheless. These include cases where the parser gave perfect scores for false linkages. An example of such a case is illustrated in Figure 4.8. The use of more restrictive values concerning the three quality-of-parse variables to limit distortion in linkpaths

may therefore be questioned, because it relies on the ability of the parser to correctly gauge the accuracy of a produced linkpath. An analysis of the effects of using more restrictive quality-of-parse values is examined in 4.4.1.

Entity Errors

This class of errors is the result of faulty entity detection in step one of the Wanderlust algorithm. It can be the result of one of the premises of Weitblick being wrong, namely that a term with a page link is always linked to the meaning of the term as meant in the current context. The following sentence (with Wikipedia markup) is an example where this is not correct:

> Physical modeling synthesis is the [[synthesizer|synthesis]] of [[sound]] by using a set of[[equation]]s and [[algorithm]]s to simulate a physical source of sound.

The terms "sound", "equation" and "algorithm" are correctly linked. The term "synthesis" however links to the page "synthesizer", which describes a machine rather than the concept of "synthesis". In this sentence, the false positive Is(Physical_modeling_synthesis, Synthesizer) is found using a good linkpath. Another problem is Weitblick's handling of attributes, which it attempts to convert to entities (see 3.1.3). This procedure can also result in false identification of entities. Entity errors are another layer of distortion which interferes with validation using linkpath coefficients.

Context Errors

This is a class of errors anticipated within Doppeldenk and occurs when information is stated in plain text which is put into context by another part of the text. Extracting this information without its context can produce incorrect relation triplets. The RDF model is generally not suited to include the context of statements, making is difficult to correctly express a range of semantics [18]. Since the subject-predicate-object relation triplets used in SMW are modeled on RDF, this gives rise to a multitude of problems. The following gives a non-exhaustive list of sources for context errors. Generally problematic are pages or paragraphs which

- describe fictional content, such as the storyline of books or movies
- express supposition, such as unproven theories
- express opinions and beliefs

Wanderlust does not determine that a relation is true only in a certain context. One problem is that sentences are analyzed independently of each other, meaning that all information spanning across sentences is lost. An article might start with a sentence like "*This page is about a fictional alternate universe*". Within this page, any number of untrue facts can be stated. Wanderlust will find the stated facts as positives, even though they are just true within the context of the first sentence. Since the context is not stated within the relation triplets this distorts the resulting knowledge base.

Another class of context errors are those which are found in "factual" sentences with none of the above listed qualities. This includes facts which are only true in a certain context. An example of this can be found in the sentence "*At night, all cats are gray*". The fact Are(Cats, Gray) stated in this sentence is true only in the context as given by the first two words of the sentence. Outside the context this fact is a false positive, even though extracted using a good linkpath. Even if a fact is put into context within the sentence itself (like in the above example), the information given by the Link Grammar Parser is insufficient to identify the context. This will be demonstrated with the following two example sentences:

"It was believed that the earth was flat."

and

"It is known that the earth is round."

Both sentences have identical link grammar parses (see 4.9). Using the good linkpath {Ss, Pa} however yields the false positive Was(Earth, Flat) as well as the true positive Is(Earth, Round). This example shows that the information whether a fact is true universally or only within a certain context is not found within a linkage.

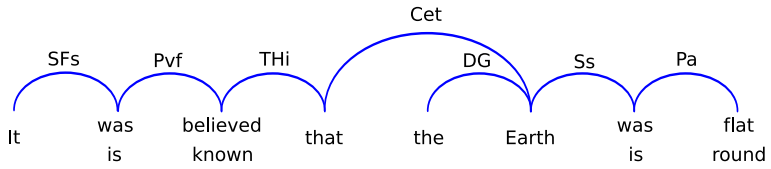

Figure 4.9: Linkage for the two example sentences. Where the words in the sentences differ, one choice is written below the other. In any case, each sentence has the same linkage.

Context errors are difficult to address. The context for all information found using Wanderlust needs to be identified and included in the semantic relations, which may require an extension to the subject-predicate-object model [18]. The challenge may include identifying pages or paragraphs which convey fictional information, supposition or opinion. But also factual information true only in a certain context needs to be identified. The identification and correct handling of context has been judged to lie outside the scope of this thesis. Context errors are not handled by Wanderlust and will therefore distort the calculation of linkpath coefficients (or any other learning algorithm applied) as well as disrupt the knowledge base generated by the algorithm.

Incomplete Object Errors

The initial theory of the proposed method is that certain linkpaths are generally usable to express semantic relations between entities. While many reasons for errors are listed in this section, none have yet challenged this theory. Parse errors provide faulty data and entity tagging is a precursor to, but not part of,

Wanderlust. Context errors can be traced to an incompleteness in the subject-predicate-object relation triplet model and the need of identifying the context of information.

The incomplete object error on the other hand compromises the initial theory. The error occurs when a rule is applied to a verb or an expression which needs more objects than the rule provides in order to be helpful. In many ways it is linked to the theory of *verb valency*. Valency in linguistics is a term which is used to describe how many arguments (i.e. subjects and objects) a verb requires or "binds" to itself [17]. Depending on the verb and its disambiguated word sense, a verb typically requires between 0 and 2 arguments. Important here is that each verb has a minimum number of objects which are required for it to make sense. In addition to this, verbs may have any number of optional objects.

Avalent verbs take no arguments, not even a subject. Examples for this include "*it rains*" or "*it snows*". The pronoun "it" in this context is an expletive pronoun, which means that it is nonexistent, unknown or irrelevant and therefore does not count as a verb argument.

Monovalent verbs take only one argument, typically a subject. Examples for this are "*He sleeps*" and "*She sings*".

Bivalent verbs take two arguments, typically a subject and an object. Verbs of this type are the minimum requirement needed in order to build a subject-predicate-object triplet. Examples of such verbs include "to lose something" as in "*I lost my cell phone*" or "to fight against something" as in "*Bill fought against Ted*".

Trivalent verbs take three arguments, typically a subject and two objects. Examples of such verbs are "to give something to someone" as in "*Zeus gave flowers to Hera*" or "to introduce someone to someone" as in "*He introduced him to her*". Using verbs which take more than two arguments poses difficulties for extracting knowledge, because subject-predicate-object triplets make connections only between one subject and one object. This problem is illustrated in a number of examples to follow.

The verb "to give something to someone" for example is trivalent, but may also have additional objects for time and place, as in "to give something to someone at some place at some time". A necessary object cannot be dropped and replaced by one of the optional objects, such as in "to give something at some time", which makes little sense.

The problem is that the information on which objects are necessary and which are not is not found within a linkage. Verbs are linked to optional and necessary objects using the same link labels. This will be demonstrated with the example sentences "*Zeus gave flowers to Hera*" and "*Alexander_the_Great founded Alexandria in Egypt*". Both sentences each have two linkages with identical link types, see Figures 4.10 and 4.11, but use verbs of differing valency. While "to found something" is divalent, the verb "to give something to someone" is trivalent. This shows how information on verb valency is not reflected in a linkage.

Using the valid linkpaths (without subtypes) {S, O} and {S, O, M, J} a total of four relations are extracted from these two sentences. The first two that are

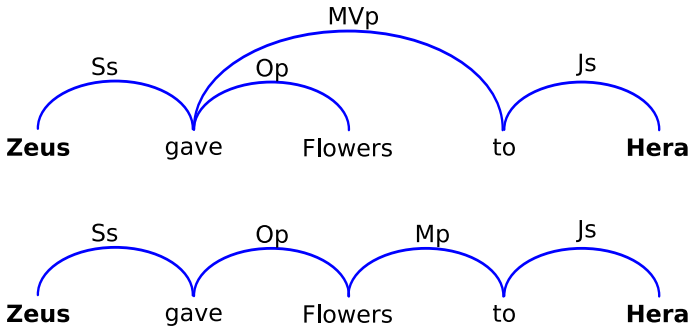

Figure 4.10: Linkages for the sentence "*Zeus gave flowers to Hera*". Verb is trivalent.

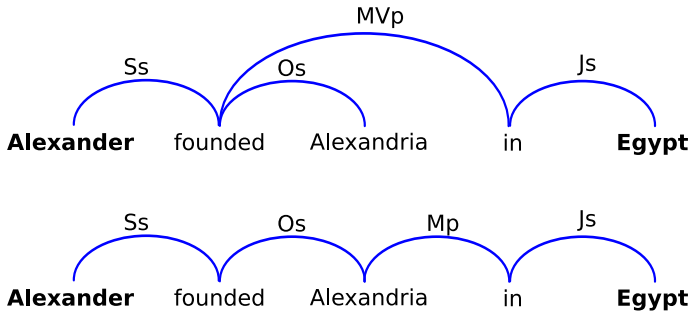

Figure 4.11: Linkages for the sentence "*Alexander founded Alexandria in Egypt*". Verb is divalent.

considered are the two relations found using {S, O}, namely Founded(Alexander, Alexandria) and Gave(Zeus, Flowers). While the former relation relates a valid fact, the latter is missing a second object in order to convey meaningful semantics. The relation Gave(Zeus, Flowers) is therefore false. This is a general problem for linkpaths of length one, which connect together two entities via one verb. If this verb is trivalent, a third argument is missing, causing the relation to be false.

Linkpaths of length three on the other hand are capable of handling trivalent verbs. Consider one of the two relations found in the example sentences using {S, O, M, J}, namely GaveFlowersTo(Zeus, Hera). In this example, the first object "flowers" is made part of the predicate connecting the subject "Zeus" with the second object "Hera". Therefore, all three necessary nouns are incorporated into the relation, making it valid. Using the same rule on the other example sentence yields the valid FoundedAlexandriaIn(Alexander, Egypt). The verb "to found something" is divalent, but may also bind additional non-necessary objects to itself, relating information such as place and time (e.g. "to found something somewhere sometime"). Because of this, the relation is valid even if a linkpath of length three is applied to a divalent verb.

These examples show that verb valency, especially in the cases of trivalent verbs, poses a problem for the algorithm. The first problem is that linkpaths of

lengths one and two which generally find valid relations fail to do so in sentences using trivalent verbs. Linkpaths of length three can also produce false positives using trivalent verbs if one optional object is part of the relation instead of a necessary one.

This problem cannot easily be resolved. Since the information on which objects are necessary and which optional is not reflected in a linkage, the grammatical patterns identified in this thesis within the link grammar formalism are not complete. One possible solution to this is to add this information to the analysis by obtaining lists of verbs and verb phrases with their valency from WordNet [27] or a similar resource. However, this requires a disambiguation step for verbs, since many verbs have ambiguities of differing valency. The verb "to give", for example, exists both in trivalent *and* divalent forms. An example of a trivalent form is "to give something to someone" as used in example sentence 4.10. An example of a divalent form is the verb in the context of the sentence "*Zeus gave the marching order*", where no second object is needed.

Thus, in order to handle the incomplete object error, the algorithm must identify necessary and optional objects for a verb, which in turn requires a method of verb sense disambiguation. It has been judged to lie outside the scope of the thesis to address this problem. The initial theory is insofar compromised as that in addition to the linkpath the valency of verbs must be considered.

4.2.3 Conclusions

A diverse number of reasons for why good linkpaths sometimes find false positives have been found. Interestingly, most reasons do not challenge the initial theory that good and bad linkpaths exist. In the cases of context, parse or entity errors, the source of the error lies outside the information given by linkpaths. So while good linkpaths do exist, to combat each error class additional information must be taken into consideration, such as a sentence's "context".

Because the handling of each of these error classes presents a challenge to itself which lies outside the scope of this thesis, a tentative analysis on the distribution and impact of these errors must be made. It must be decided whether the algorithm might perform "reasonably well" in spite of the noted problems. This analysis is performed in 4.3.

4.3 Test Set Analysis

The results and observations of the previous section lead to two important questions, which at this point need to be answered: First, statistical information must be generated which shows how often errors from each of the causes of errors named in 4.2.2 appear. This must be done in order to gauge the actual effect each cause of errors has on validation via linkpath coefficients. Second, it must be determined how distorting the *semantics* of false positives are to the overall knowledge base Wanderlust generates. During annotation all relations containing either unhelpful, nonsensical or untrue information were labeled as false. The precision and recall values in Section 4.2.1 were computed accordingly. However, in order to make a more accurate prediction of the algorithm in terms of produced semantics a more differentiated version of precision variables, which takes the "graveness" of the errors into account, is expedient.

For this analysis, a test set of 6 annotated pages was used consisting of a total of 705 relations, of which 183 were labeled as valid and 522 labeled as nonsensical. A list of linkpath coefficients was generated using the training set, with which Wanderlust extracted relations from the test set. The minimum linkpath coefficient was 0.5, both maximum linkage number and maximum number of skipped words was 2.

Using these settings, Wanderlust found 142 relations, of which a total of 39 relations were false positives. This puts precision at 0.73 and recall at 0.56, both of which were expected values based on the analysis of Section 4.2.1. The results were analyzed in order to answer the two above questions. In Section 4.3.1, the graveness of the errors is analyzed. In Section 4.3.2, the errors are assigned to error cause classes. Examples for each error cause class found in the test set are given and distribution of errors among the classes analyzed.

4.3.1 Impact of False Positives

This section addresses the second of the two questions mentioned in 4.3. Specifically, it must be analyzed how grave the impact of false positives is on the knowledge base which Wanderlust generates. False positives may contain unhelpful, nonsensical or untrue information. The 39 false positives found in the test set were split into these three categories which represent the "graveness of the semantic error". The distribution among the categories is given in Table 4.5.

Table 4.5: Categories of false positives.

Category	# False positives
Unhelpful	20
Nonsensical	8
Untrue	11

The categories are introduced in the following subsections, where examples for false positives are given and their impact on the overall semantic validity of the knowledge base generated by Wanderlust discussed. Note that the graveness of a semantic error cannot always clearly be established and may therefore be debatable. In the first two subsections, alternative values for precision are computed.

Unhelpful

The first category, called "unhelpful", consists of relations which have been labeled false because they pertain incomplete or unhelpful information. Examples of such relations are:

- Is(Kármán_line, Definition) - found in the sentence "*The Kármán line, defined as 100 km above the Earths surface, is a working definition for the boundary between atmosphere and space*".
- Has(Moon, Density) - found in "*The Moon has a mean density of 3,346.4 kg, making it the second densest moon in the Solar System after Io*".

- IsReferredToIn(Solar_System, Science_fiction) - found in "*By extension, the Solar System is often referred to in science fiction as the Sol System*".
- Is(Venus, Setting) - found in "*A terraformed Venus is the setting for a number of diverse works of fiction that have included Star Trek, Exosquad and the manga Venus Wars*".

In all cases, the information given by the relations is not false, but simply too general or unnecessary to be of much help. However, false positives in this group do not convey wrongful semantics and therefore do not distort the model of knowledge which Wanderlust is designed to construct from the English Wikipedia. This means that it is maintainable to leave out false positives from this group when computing precision for the algorithm. If the 20 false positives in this group are counted as true instead of false positives, the precision of the algorithm is 0.87 on the test set and therefore very near the desired value.

Nonsensical

This is the second category of false positives. Relations in this category are nonsensical, meaning that they contain neither true nor false information. Examples of such relations are:

- Reaches(Jupiter, Opposition) - found in "*As a result, each time Jupiter reaches opposition it has advanced eastward by about the width of a zodiac constellation*".
- IsSpacecraftTo(Galileo_orbiter, Jupiter) - found in "*So far the only spacecraft to orbit Jupiter is the Galileo orbiter, which went into orbit around Jupiter on December 7, 1995*".
- IsPlanetFrom(Jupiter, Planet) - found in "*Jupiter is the fifth planet from the Sun and the largest planet within the solar system*".
- WasObservedTo(Venus, Outer_planets) - found in "*Venus was observed by the Galileo and Cassini spacecraft during flybys on their respective missions to the outer planets, but Magellan would otherwise be the last dedicated mission to Venus for over a decade*".

False positives in this category distort semantics by giving information without meaning. However they do not contain false facts, which are the strongest form of semantic distortion. Some features of the semantic wiki, for example the core feature of semantic querying, are not impaired by nonsensical information. A semantic query which would yield any of the above example relations would have to explicitly ask a nonsensical question, which typically will not be the case.

Because of this, it is possible if debatable to leave out nonsensical false positives when computing precision and recall values for the algorithm. If leaving out nonsensical false positives and counting unhelpful relations as true positives, the algorithm's precision is 0.92 on the test set, a value well within desired levels.

Untrue

This is the last category of false positives, which includes all relations which relate untrue information. Examples are:

- WasMotherOf(Jord, Annar) - extracted from "*In Norse mythology, the Earth goddess Jord was the mother of Thor and the daughter of Annar.*"
- Is(Venus, Unidentified_flying_object) - found in "*As the brightest point-like object in the sky, Venus is a commonly misreported unidentified flying object*".
- Is(Saturn, Hydrogen) - found in "*The planet Saturn is composed of hydrogen, with small proportions of helium and trace elements*".
- Orbited(Venus, Earth) - found in "*Pythagoras is usually credited with recognizing in the sixth century BC that the morning and evening stars were a single body, though he thought that Venus orbited the Earth*".

These relations distort the semantics of the model of knowledge generated by Wanderlust and make up approximately 10% of all relations found. Finding causes for untrue relations being false positives is therefore of the highest priority in the error-analysis of the following section.

4.3.2 Error Classes

This section analyzes the causes of false positives in the test set. All 39 false positives can be traced to one of the error classes introduced in 4.2.2. Table 4.6 gives the distribution of false positives among the error classes.

Table 4.6: Classes of error causes.

Class	# False positives
Incomplete object errors	19
Parse errors	12
Context errors	8
Entity errors	0

The distribution shows that almost half of all false positives in the test set are caused by incomplete object errors. The next largest group is parse errors, which account for 12 errors, followed by context errors accounting for 8. In the test set no entity error was found. In order to determine which error class causes the most distortion to the knowledge base refer to Table 4.7 where false positives are classified according to semantic graveness.

As can be seen in this table, parse errors account for the gravest errors, i.e. the most nonsensical and untrue false positives. Context errors are equally split among all classes, while incomplete object errors overwhelmingly cause unhelpful false positives. Based on the considerations in Section 4.2.2 this distribution is not surprising. A more detailed analysis of the error classes as observed in the test set is given in the following subsections.

Table 4.7: Error classes and graveness.

Class	Unhelpful	Nonsensical	Untrue
Parse errors	0	4	8
Context errors	3	3	2
Incomplete object errors	17	1	1
Entity errors	0	0	0

Parse Errors

This section discusses false positives in the test set that are extracted because of incorrect linkages produced by the Link Grammar Parser. A large fraction of untrue false positives (i.e. the gravest semantic error) comes from this class. The problem with false parses is that linkpaths which normally are valid fail, because they are applied to a sentence that has not been correctly understood by the parser. The link grammar quality-of-parse variables may be but not necessarily are an indication of this. One example of a parse with skipped words for instance is the sentence:

> "In 1868, Norman Lockyer hypothesized [that] these absorption lines [were] because of a new element which he dubbed helium, after the Greek Sun god Helios."

The Link Grammar Parser could only parse the sentence by skipping the two words in rectangular brackets, yielding a semantically different sentence. In this sentence, the false positive Hypothesized(Norman_Lockyer, Absorption_lines) is found. This relation is a valid match for the semantics of the sentence if not considering the skipped words. Since the skipped words however are part of the sentence, the extracted relation is a nonsensical false positive.

Skipped words however are not the singular cause of incorrect linkages. Of all 12 parse errors, only 5 can be traced to skipped words. An analysis of restricting quality-of-parse variables more strongly is performed in 4.4.1 where it is found that limiting the algorithm to consider only linkages without skipped words increases precision only slightly while more strongly reducing recall. This means that many linkages with skipped words still allow for the extraction of true positives. Therefore lowering the maximum number of skipped words to 1 or 0 is debatable.

Another indicator for false linkages is the linkage number. One example sentence with a good first linkage and a bad second linkage is the following:

> "In Norse_mythology, the Earth goddess Jord was the mother of Thor and the daughter of Annar."

The first linkage as produced by the Link Grammar Parser is displayed in Figure 4.12, the second in Figure 4.13. In both figures, the first words of the sentence have been omitted for a better display of the important part of the sentence. As can be seen, the first linkage correctly relates the grammaticality of the sentence. The relations WasMotherOf(Jord, Thor) and WasDaughterOf(Jord, Annar) are extracted from this sentence both using the valid linkpath {Ss, Ost, Mp, Js}. The second linkage however is incorrectly parsed. The problem lies

with the word "and" which the parser fails to correctly put in grammatical context. Using the same rule as above, the untrue relation WasMotherOf(Jord, Annar) is found.

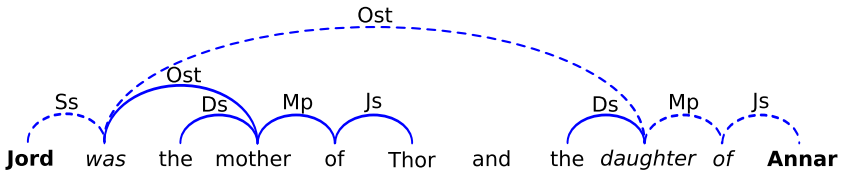

Figure 4.12: First linkage of the example sentence. The correct relation WasDaughterOf(Jord, Annar) is found. The according linkpath is highlighted.

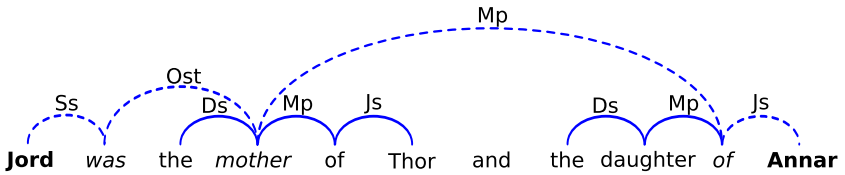

Figure 4.13: Second linkage of the example sentence. The false relation WasMotherOf(Jord, Annar) is found. The according linkpath is highlighted.

Note that both linkages have the same score and number of skipped words. The only difference is linkage numbering which in both cases is very low. Such errors can only be averted if only relations found in linkages with a linkage number of 0 are permitted. This however comes at a price of recall loss, as analyzed in Section 4.4.1. Of the 13 errors made because of false linkages, a total of 7 come from parses greater than 0, and 5 come from parses greater than 1.

A final problem are sentences without skipped words where even the first returned linkage is a false parse. An example of this is the sentence:

> "The Moon is a differentiated body, being composed of a geochemically distinct crust, mantle, and core."

This sentence is parsed erroneously tagging the word "being" as a noun, which changes its semantics. Instead of the gerund "being" initiating a subordinate clause, the sentence is parsed as an enumeration. It therefore falsely states that the moon is a differentiated body, a being composed of a geochemically distinct crust, a mantle and a core. From this false parse the two untrue relations Is(Moon, Mantle) and Is(Moon, Core) are extracted. A total of 3 false positives are extracted from linkages with perfect quality-of-parse variables.

Parse errors make up for the highest part (8 of 11) of untrue false positives in the test set and therefore are the most distorting error class. They can be reduced by further restricting the link parser quality-of-parse variables, which however comes at the price of recall. An analysis of the effects of further reducing the quality-of-parse variables is performed in 4.4.1.

Context Errors

This section gives some examples for false positives in the test set, extracted because of context errors. Information is stated in one part of a sentence, but put into a different context in another. The loss of the context causes the extracted information to be false. An example of such a sentence is:

> "Pythagoras is usually credited with recognizing in the sixth century BC that the morning and evening stars were a single body, though he thought that Venus orbited the Earth."

Using the valid linkpath {Ss, Os}, the false positive Orbited(Venus, Earth) is extracted from the sentence. The linkpath is a basic subject-verb-object rule applied to a local part of the sentence, while neglecting the overall context. In the example sentence the context needed in order for Orbited(Venus, Earth) to be a true positive is "Pythagoras thought that". This context however is neither found nor can it be included in a subject-predicate-object triplet.

Other examples of false positives in the test set are:

- Has(Earth, Tolerance), found in "*Hence compared to a perfect ellipsoid, the Earth has a tolerance of about one part in about 584, or 0.17%, which is less than the 0.22% tolerance allowed in billiard balls.*"
- BecomesOpaqueTo(Sun, Visible_light), found in "*The visible surface of the Sun, the photosphere, is the layer below which the Sun becomes opaque to visible_light.*"

This class of errors makes up of 8 total errors in the test set, of which 2 are untrue, 3 nonsensical and 3 unhelpful.

Incomplete Object Errors

This section gives examples of false positives in the test set caused by incomplete object errors. They are caused by missing objects for verbs or verb phrases in the predicates. Examples of such errors are:

- Has(Moon, Apparent_magnitude), found in "*During its brightest phase, at full moon, the Moon has an apparent magnitude of about –12.6.*".
- Believed(Ancient_Egyptians, Venus), found in "*The Ancient Egyptians believed Venus to be two separate bodies and knew the morning star as Tioumoutiri and the evening star as Ouaiti.*"
- Is(Mount_Chimborazo, Feature), found in "*Because of the bulge, the feature farthest from the center of the Earth is actually Mount Chimborazo in Ecuador.*"
- Has(Saturn, Large_number), found in "*Saturn has a large number of moons.*"

In these examples, objects are missing to make the information stated by the relations useful. The relation Believed(Ancient_Egyptians, Venus) for example uses the verb phrase "to believe something to be something". The second object is missing, causing the information to be an unhelpful false positive.

The relation Has(Saturn, Large_number) uses the verb phrase "to have a number of something". The object "of something" is not part of the relation, causing it to be an unhelpful false positive.

As was shown in Table 4.7, the highest part of false positives caused by this error class is of error graveness unhelpful, which is the lowest error graveness in terms of semantic distortion. This means that while a large number of errors are caused by the incomplete object error, their impact on the knowledge base is limited.

4.3.3 Conclusions

The analysis of the test set has revealed that parse errors are responsible for the gravest errors in validation using linkpath coefficients. A case for further restricting the link grammar quality-of-parse variables is made in order to limit the impact of parse errors.

Context errors and incomplete object errors are as previously noted not addressed in the thesis. While incomplete object errors account for the largest number of false positives in the test set, their impact on the knowledge base is limited due to most errors being of low error graveness. This is not surprising, given that these errors are mostly caused by missing objects, which renders verbs and verb phrases incomplete, but not false.

Alternative validation techniques and features are tentatively discussed in the ensuing section.

4.4 Alternatives

This section discusses validation using additional features and different learning algorithms. As previously established, most of the causes of errors each require more attention than can be given within the scope of this thesis. An exception to this are parse errors, which can be reduced by restricting the algorithm to consider only linkages with certain quality-of-parse variables. An analysis of this is performed in 4.4.1. The reasons for using linkpath coefficients as opposed to more advanced learning algorithms for purposes of validation are discussed in 4.4.2.

4.4.1 Additional Features

In order to increase the precision of the algorithm it was considered to use additional features aside from the linkpath coefficient for the purpose of validation. A number of easily measurable features have been tentatively discussed to be part of the feature vector, such as

- POS-tag information; because the Stanford POS-tagger is already part of the implementation, the POS-tags of subject, predicate and object of each relation were included in the feature vector
- the Link Grammar Parser quality-of-parse variables *skip*, *score* and *linkage number*
- the *distance* (number of words) between subject and object

- the total number of words in the sentence

The problem however is that most of these features do not address the error causes identified in 4.2.2, with the exception of parse errors. The quality-of-parse variables are arguably important, but as previously demonstrated, the Link Grammar Parser sometimes fails to correctly estimate the quality-of-parse of a returned linkage. Even linkages with perfect quality-of-parse variables have been found to be false. A thorough analysis was conducted of the effects of restricting each of the quality-of-parse variables, performed on the same training set as in Section 4.2.1. Linkpaths of length 6 are not considered. The training set thus consists of 9,294 relations, of which 2,215 are marked as valid and 7,079 as false. The results of the analysis are given in the following subsections.

Skipped Words

Early results with skipped words had shown that skipping words can be very disruptive to the parse of a sentence. This is because the sentence is parsed as if the skipped words are not part of the sentence, which can completely change its semantics. Because of this, the number of skipped words was limited to a maximum of 2 words per sentence early on. This had the side effect of reducing the annotation workload as all parses of a given sentence with more than 2 skipped words were dismissed out of hand.

In the following the effects of lowering the maximum number of skipped words to 1 and 0 are analyzed with regards to possible recall and precision. Table 4.8 lists an overview of all valid relations found in linkages with a maximum of 0, 1 and 2 skipped words respectively within the training set. The column "# Valid" lists how many of the 2,215 valid relations in the training corpus are found in linkages with a maximum number of skipped words as indicated by the first column. The possible recall values computed represent the maximum recall possible with a perfect validation process.

Table 4.8: Possible recall levels with respect to maximum skipped words.

Max skip	# Valid	Possible recall
2	2215	1
1	2078	0.94
0	1678	0.76

The table shows that in terms of recall it is safe to lower the maximum number of skipped words to 1, if not 0. The impact on precision levels must now be analyzed. The Table 4.9 lists the impact of reducing the maximum number of skipped words on precision. The overall precision level is listed for different linkpath coefficient threshold and max skip values. Not that all relations of error graveness classes unhelpful, nonsensical and untrue count as false positives in this calculation.

With a minimum linkpath coefficient of 0.5, precision of 0.69. By limiting the maximum number of skipped words to 1, precision is raised to 0.71. Limiting it to 0 further raises precision to 0.72. Overall precision increase is modest.

Table 4.9: Precision levels with respect to max skipped words and linkpath coefficient threshold.

Coefficient threshold	0.7	0.6	0.5	0.4
Skip 2	0.74	0.71	0.69	0.64
Skip 1	0.75	0.71	0.71	0.65
Skip 0	0.75	0.73	0.72	0.67

Linkage Number

Linkages for a given sentence are ordered by a linkage number. The linkage with the number 0 is by estimation of the parser the most likely linkage for the sentence and therefore the most trustworthy. It was discovered early on that higher linkage numbers usually represent bad parses. This observation is mirrored in [35] where it is noted that "Link Grammar manages to parse very complex verb phrases correctly, even if many incorrect ambiguities are reported."

As a consequence of early observations, the maximum number of linkages per sentence was limited to 2. This greatly reduced the effort of annotation, as for complex sentences linkages numbering in the hundreds are possible. Table 4.10 lists how many valid relations are found in linkages with a maximum linkage number as given by the first column.

Table 4.10: Possible recall levels with respect to maximum linkage number.

Max linkage	# Valid	Possible recall
2	2215	1
1	2063	0.93
0	1873	0.85

The table shows that effects of limiting the maximum linkage number to 1 on possible recall are minimal. Limiting the maximum linkage number to 0 lowers possible recall further, though not as low as limiting the number of skipped words to 0.

In Table 4.11, precision levels for different thresholds and maximum linkage numbers are given.

Table 4.11: Precision levels with respect to max linkage number and threshold.

Coefficient threshold	0.7	0.6	0.5	0.4
Skip 2	0.74	0.71	0.69	0.64
Skip 1	0.76	0.72	0.7	0.67
Skip 0	0.77	0.72	0.71	0.68

As when limiting the number of skipped words, effects on precision are modest. The best precision level is achieved when using a coefficient threshold of 0.7 and a maximum linkage number of 0.

Linkage Cost

The last Link Grammar Parser variable, cost of a linkage, and its impact on precision and recall is discussed in this section. By limiting the variable to values lower than 10, recall is lowered as listed in Table 4.12.

Table 4.12: Possible recall levels with respect to maximum linkage cost.

Max cost	# Valid	Possible recall
10	2211	1
9	2204	1
8	2196	0.99
7	2188	0.99
6	2147	0.97
5	2088	0.94
4	2039	0.92
3	1912	0.86
2	1742	0.79
1	1365	0.62
0	939	0.42

As can be seen in this table, limiting the maximum cost of linkages down to 4 or 3 only has little impact on recall. Lower levels more greatly affect recall. At 2, recall is down to 0.79 and at 1 to 0.62. Limiting linkages to a cost of 0 reduces recall to a very low 0.42.

In Table 4.13, effects on precision are listed.

Table 4.13: Precision levels with respect to max cost and threshold.

Coefficient threshold	0.7	0.6	0.5	0.4
Cost 10	0.74	0.71	0.69	0.64
Cost 6	0.75	0.7	0.69	0.65
Cost 4	0.76	0.72	0.71	0.68
Cost 2	0.77	0.73	0.71	0.68
Cost 1	0.76	0.74	0.72	0.69
Cost 0	0.78	0.75	0.73	0.73

This table shows moderate increases in precision as the maximum cost of linkages is lowered. Precision levels of up to 0.78 are achieved allowing only linkages with a cost of 0. However, as evident from Table 4.12 this comes at a very high price of recall. Because of this, cost is a variable which is only of limited usability for enhancing precision.

Conclusions

The link grammar quality-of-parse variables may be used to enhance precision levels of grammatical validation using linkpaths. Precision gains however are moderate for the most part, indicating that falsely parsed sentences are not the dominant cause of wrongful relation extraction in Wanderlust. As has been observed in the test set analysis, parse errors are potentially responsible for

a large part of untrue false positives, making even moderate precision gains important.

While linkage cost comes at a high price of recall, both linkage number and skipped words can be further limited to increase overall precision of the validation procedure.

4.4.2 Alternative Learning Algorithms

Given that an annotated training set exists and that a number of additional features have been suggested, the possibility of training a classifier with a well known data-mining algorithm such as decision trees or support-vector-machines must be discussed. To this end, the Weka DataMining suite [13] has been employed. The suite features a wide range of classifiers which may be trained with annotated data. The resulting classifiers can be saved as model-files and integrated into a java program.

A number of models were tentatively tested and integrated into the Wanderlust algorithm to perform validation in place of using the linkpath coefficient. Performance was found to be generally low. The problem here is that as previously noted most of the causes of errors cannot be solved using the features listed in 4.4.1; Entity errors are caused by a separate process, context errors require identification of context and possibly an extension to the relation triplet model, and incomplete object errors require verb sense disambiguation and valency information. These errors cause good linkpaths to fail. A learning algorithm will try to explain this using features which are unrelated to the error causes.

Because a large number of features are observed, but compared to the total number of possible linkpaths the training data set is relatively small, a learning algorithm is vulnerable to overfitting. An illustration of this problem can be found in Figure 4.14 which shows a segment of a decision tree generated with Weka and the training data set.

```
P1 = VBZ
|   L2 = Ost
|   |   ObjPOS = NN
|   |   |   SentenceLength <= 30: 1 (120.0)
|   |   |   SentenceLength > 30
|   |   |   |   LGSkip <= 0: 1 (5.0)
|   |   |   |   LGSkip > 0: 0 (11.0)
|   |   ObjPOS = JJ NN NN
|   |   |   SubPOS = NNP
|   |   |   |   Distance <= 3: 1 (2.0)
|   |   |   |   Distance > 3: 0 (16.0/1.0)
|   L2 = Os
|   |   ObjPOS = JJ NN
|   |   |   LGParse <= 6: 1 (25.0)
|   |   |   LGParse > 6
|   |   |   |   SubPOS = NNP
|   |   |   |   |   Distance <= 7: 1 (3.0)
|   |   |   |   |   Distance > 7: 0 (3.0)
```

Figure 4.14: Excerpt of a decision tree for the training set.

Due to a large number of distinct link types and possible linkpaths, the model is very complex, allowing it to precisely fit the training data. However, the rules in this decision tree cannot be used as the universally valid grammatical patterns sought in this thesis. Distinctions are made for example according to the POS-tags of the object (**ObjPOS**) which works in the training set, but not generally.

Even if disregarding all information but link labels and quality-of-parse variables, the models generated with various learning algorithms have been found to be too specific to the training data. The problem is that in order for the training set to have enough representative examples of linkpaths it needs to be much larger than the set used in this thesis.

4.4.3 Conclusions

The use of additional features aside from the linkpath has been found to be of limited use. Further restricting the link grammar quality-of-parse variables provides limited increases in precision, which because of the impact of false positives due to parse errors are nonetheless estimated to be significant.

The use of additional features and more sophisticated learning algorithms has been dismissed due to problems stemming from missing features, a limited size of the training set and the resulting problems of overfitting. In light of the linguistic approach used in this thesis and the universally valid grammatical patterns that are sought, the use of linkpaths has been chosen as a simple and potentially effective method for validation.

4.5 Summary / Conclusions

This section summarizes the chapter and makes conclusions based on the analysis of the feature vectors and observations made from the annotated corpus.

First and foremost, the decision to use linkpath coefficients as primary validator has been deemed feasible. Even though a very high number of different linkpaths exist, most grammatical phenomena are covered by a small batch of the most common linkpaths. This means that basing validation on the topmost common linkpaths is possible and can even be beneficial because an analysis can focus on a small set of rules. A total of 43 linkpaths have been identified which fulfill the requirements of a minimum coefficient of 0.5 and have been observed more than 50 times in the annotated corpus. The linkpaths are listed in Appendix B.4, each with the number of relations extracted from the English Wikipedia corpus.

For uncommon linkpaths, a heuristic will be used to generate coefficients by disregarding subtype information. Linkpaths without subtype information are referred to as *abstract linkpaths*. Since abstract linkpaths are less specialized than regular linkpaths, a larger number of examples will be found in the annotated corpus from which coefficients can be generated. Because the disregarding of subtype information also reduces the algorithm's confidence in the coefficient, all relations extracted using abstract linkpaths are stored in a separate database for further analysis.

It has also been determined to limit the maximum number of skipped words to 1 and the maximum linkage number to 2. This will lead to an expected

precision of 0.71, which is increased further by special treatment rules stated in the next chapter. As indicated by the tentative analysis of the test set, most false positives are either of error graveness unhelpful or nonsensical, meaning that the semantic distortion caused by false positives is believed to be much lower than a precision level of 0.71 implies. For the same reason, linkpaths with a coefficient of 0.5 are believed to be of higher accuracy than the coefficient implies. Other features such as POS-tagging of subjects, objects and the predicate have been deemed to be of negligible importance.

The initial theory that linkpaths can be classified as either good or bad has been found to be true with certain reservations. The algorithm is vulnerable to linguistic problems such as coreferences and context sensitivity. However, the theory that some linkpaths are generally valid is not greatly impaired by these observations since the information required to handle these problems lies outside the grammatical information of the sentence itself. The incomplete object error on the other hand must be seen as a crucial flaw in the algorithm. In order for the theory of good and bad linkpaths to be correct, the algorithm must be aware of the valency of verbs and verb phrases and be able to distinguish between necessary and optional objects. This error however has been found to be mostly responsible for false positives that are not semantically distorting, limiting its impact on the knowledge base generated by Wanderlust.

The ensuing section analyzes the results gathered from using Wanderlust with the English Wikipedia corpus.

Chapter 5

Results

This chapter lists and evaluates the results of the thesis. Wanderlust was applied to the English Wikipedia corpus dated from October 2008 which contains slightly more than 2.4 million entries. Due to the CPU intensive natural language parsing task, the algorithm was run in parallel on a cluster of 50 commodity hardware machines. Parsing results are stored in a separate database. The parameters for Wanderlust were as described in the previous chapter, leading to the extraction of **2.64 million relations** for **1,38 million entities** with a total of **312,744 distinct predicates**. The normalization procedure yielded a total of **749,703 subproperties**. This data is hereafter referred to as the *result set*.

A range of data was generated and observed. Each relation was stored in a database along with a large number of observed features, such as POS-tag information on subject, object and predicate, the distance (number of words) between subject and object, the length of the sentence and, if part of the predicate, the extended entity and its POS-tags. A full overview of all observed features is given in Appendix B.3. Although these features are not used in the validation process, they are stored nonetheless in order to enable analysis on the result set as well as future work. A quantitative analysis pertaining to the values recall and precision is performed in 5.1. Causes for recall and precision loss are listed and weighted.

In Section 5.2, the result set is analyzed qualitatively. The most common relation types are discussed and placed into categories in order to gain insights about the knowledge stored in the result set. Predicates are placed into groups according to their POS-tags and the most common linkpaths for each group of predicates analyzed and illustrated. This serves to give an overview over the most common grammatical patterns Wanderlust is able to interpret. Also in this section, those relations dismissed by the algorithm are analyzed. Using special treatment rules, a portion of these dismissed relations can be recovered in order to raise recall and by extension precision of the result set. Examples of this are given in 5.2.2 and 5.2.3. A total of 410,566 relations are recovered with these rules, raising the result to over **3 million relations**. The dataset containing dismissed relations is hereafter referred to as *secondary dataset*.

5.1 Quantitative Result Analysis

This section performs an analysis on the result set in order to determine precision and recall values for the algorithm. First, a number of random Wikipedia pages were manually annotated with subject-predicate-object triplets, hereafter known as the *evaluation set*. All data a human reader could find within the pages was annotated, regardless of the algorithm's restrictions (coreferences, Doppeldenk limitations). This was done to enable a differentiated analysis of the algorithm's recall computed both against all information contained in text samples as well as all information that can be found within the limitations of the approach. Some difficulties encountered during the annotation process are illustrated in 5.1.1.

Wanderlust was then applied to the annotated pages. 155 true positives were found in the evaluation set, which contains of 2,077 positives. Compared to what a human reader might find, the raw recall of Wanderlust is thus 7.5%. All unfound positives were assigned into classes representing the reason for recall loss. Section 5.1.2 gives an overview of these classes and analyzes the distribution of unfound positives in order to identify weaknesses in the algorithm and priorities for future work. When excluding recall loss suffered because of one of the algorithm's limitations, an overall recall of 20% is calculated.

The total number of positives found by Wanderlust is 251, of which 155 are true positives classified as *useful*. For an overview of the classification of positives according to the usefulness for the knowledge base refer to 5.1.3. In this section, a number of precision values based on different considerations are calculated, as was done in the test set analysis. When counting only true positives of class useful as true positives, precision is 62%.

5.1.1 Method

It was decided to manually annotate a number of random Wikipedia pages with all data a human reader could find. A total of 4,005 sentences were annotated with 2,077 relations. This is a ratio of approximately one relation every two sentences which might appear scant. The reason for this is that data was only annotated if it could be put in the form of a subject-predicate-object triplet with the restriction of using "reasonable" predicates and only **existing** pages in the corpus. These three restrictions severely limit the manner of information that can be expressed as relation-triplets.

One important problem is that some relation triplets are true only in a certain context. In the following, this problem is illustrated with the example sentence "*A small amount of black eumelanin in the absence of other pigments causes grey hair*". Outside the context "absence of other pigments" the relation Causes(Black_eumelanin, Grey_hair) is false. Including this context into the relation triplet yields unreasonably long relation types such as CausesInAbsenceOfOtherPigments(Black_eumelanin, Grey_hair) or even CausesInAbsenceOfOtherPigmentsTheApparitionOf(Black_eumelanin, Grey_hair). This demonstrates how certain information cannot be stated within a subject-predicate-object triplet as used in this thesis. Accordingly, such relations are not annotated.

Another important problem is that while there are a large number of Wikipedia pages that may serve as entities, there are many concepts for which

no page exists. The above mentioned sentence for example states information about a type of melanin for which no Wikipedia page exists within the corpus. Depending on the page, there may be a large number of terms for which no individual pages exist. Examples include individual characters of books or movies, making it difficult to annotate a page recounting the storyline of fictional work. A related problem is that most numbers and percentages do not have pages in the corpus, making it impossible to model such information within subject-predicate-object triplets. This affects among others pages stating statistical information or mathematical formulae.

Taken together these problems explain the ratio of annotated relations per sentence in the evaluation set.

5.1.2 Recall

Of all 2,077 annotated relations in the evaluation set, a total of 155 were found by Wanderlust. This means that the recall for Wanderlust is 7.5% compared to what a human reader might find in a page. The 1,922 relations not found by Wanderlust were split into several classes. Each class represents a reason for recall loss. Table 5.1 lists all classes and the absolute and relative amount of relations in each.

Table 5.1: Quantification of recall loss.

	# Relations	Percent
All relations	2077	100%
Doppeldenk	799	38.5%
Coreference	518	24.9%
Entity	242	11.7 %
Parse error	121	5.8%
Other	37	1.7%
Insufficient linkpath	205	9.9%
True positives	155	7.5 %

The classes are listed according to the order of incidence. In the following, each of the classes in the table is discussed. By far the greatest is **Doppeldenk**, accounting for 799 or 38.5% of all unfound relations. This class is listed first because relations which lie outside the Doppeldenk limitations can by definition not be found by the algorithm. The second is **coreference** which is all recall loss suffered because of missing coreference resolution. It accounts for 518 or 24.9% of all unfound relations. Missing coreference resolution is a stated limitation of the algorithm's implementation, but is not part of Doppeldenk since theoretically this problem can be addressed separately.

If a relation lies within the limitations of the algorithm the first error class is **entity**, which arises even before a sentence is parsed into link grammar. In order for a relation to be found the necessary terms in the sentence must be disambiguated, a process performed by Weitblick entity tagging. Insufficient tagging accounts for 242 or 11.7% of all unfound relations. The next precondition for Wanderlust is the correct parsing of a sentence by the Link Grammar Parser. Recall loss suffered because of **parse errors** account for 121 or 5.8%

of relations. **Other** is a set of reasons which occur very rarely, such as problems with the sentence splitter. Finally, **insufficient linkpath** means that all prerequisites for a relation to be extracted are met (entities found, parse successfully generated), but a relation is not found because the linkpath is either unknown or has too low confidence. Insufficient linkpath accounts for 205 or 9.9% of relations.

At 7.5% recall is very low when compared to all annotated relations. The first two classes (namely Doppeldenk and coreference) however lie outside of the algorithm's limitations and were therefore never expected to be found by Wanderlust. Taken together, both classes account for a total of 1,317 and therefore 73% of all annotated relations. Precision values may therefore also be calculated without these relations. Table 5.2 shows percentage values for all relations that can be found and lie within the algorithm's limitations.

Table 5.2: Quantification of recall loss within the limitations of Wanderlust.

	# Relations	Percent
All relations	760	100%
Entity	242	31.8 %
Parse error	121	15.9%
Other	37	4.9%
Insufficient linkpath	205	27%
True positives	155	20.4 %

In this perspective, the portion of true positives is 20.4%. Entity accounts for 31.8%, parse error for 15.9% and insufficient linkpath for 27%. So depending on the perspective, the recall for Wanderlust is either 7.5% or 20.4%. Both values are rather small, raising the question of why in Section 4.3 a much better recall value of 56% was computed. The difference lies in the annotation approach illustrated in Section 4.1 in which the first three steps of the Wanderlust algorithm were performed and the user had to decide whether the relations offered by Wanderlust were valid or false. This process eliminates all sources of errors, including entity and parse errors and focuses instead only on linkpaths. If all errors but insufficient linkpaths are discounted in the evaluation set, a total of 360 positives are found, of which 205 are false negatives and 155 are true positives. This puts precision at 43%, a value near the tentative 56% found in the test set.

In conclusion, areas of future work in order to raise recall of the algorithm are the inclusion of coreference resolution and amelioration of the entity tagger. Most of the relations dismissed because of insufficient linkpaths are stored in the secondary dataset. In 5.2, some powerful examples are given of how to retrieve this data and therefore further raise recall of the algorithm. Most are directly related to the incomplete object error, which as demonstrated in 4.2.2 is the cause for certain low coefficients. By addressing the error in future work, the overall recall can be raised.

5.1.3 Precision

Wanderlust finds 251 positives in the evaluation set. In order to calculate differentiated precision values for the algorithm, the positives are distributed among

classes of correctness. Most of these classes were already introduced in Section 4.3.1. The most common class of relations found is *useful* true positives which positively contribute to the model of knowledge built by Wanderlust. Less useful, but not false, are relations of class *unhelpful*. These relations, such as Has(Moon, Density) or Is(Kármán_line, Definition) are semantically too weak in order to be useful for most application areas. Relations of class *nonsensical* are meaningless statements such as IsPlanetFrom(Jupiter, Planet) or WasObservedTo(Venus, Planet). This information is neither true nor false. The last correctness class are *untrue* false positives which distort the model of knowledge Wanderlust generates. The distribution of the false positives among these classes is given in Table 5.3.

Table 5.3: Correctness of relations.

Group	#
Useful	155
Unhelpful	51
Nonsensical	24
Untrue	21

As was done in the test set analysis, a number of precision values can be calculated for the algorithm based on this distinction. While useful relations are clearly true and untrue relation clearly false positives, the other two correctness classes might be seen either way. Refer to Table 5.4 for different precision values. If only useful relations count as true positives, precision is 0.62. Because unhelpful relations relate incomplete or unnecessary facts, but no false semantics, they can be seen as true positives, putting precision at 0.82. As previously argued, some application domains such as semantic search are not negatively affected by nonsensical relations. When counting nonsensical relations as neither true nor false, a precision value of 0.91 is calculated.

Table 5.4: Precision values.

True positives	False positives	Precision
Useful	Unhelpful + Nonsensical + Untrue	0.62
Useful + Unhelpful	Nonsensical + Untrue	0.82
Useful + Unhelpful	Untrue	0.91

The distribution of errors among correctness classes is therefore as expected based on the analysis of the test set in Section 4.3. A difference to the aforementioned analysis has however been revealed in the distribution among error cause classes. While in the test set incomplete object errors were the dominant cause for precision loss, this has not been found to be true in the evaluation set. Refer to Table 5.5 for error causes and their effect on precision. The first four error causes have already been introduced in Section 4.3.2. The fifth, *core errors*, had not been observed in the test set and arises in cases where the title entity is a concept entity. Because it has been argued that title entities are not references to other referents within their pages, they have been allowed to be used as subjects for relations. This has not been found to be universally true, causing errors.

Table 5.5: Quantification of precision loss.

Class	# False positives	Percent
Context errors	32	33.3%
Parse errors	22	22.9%
Coref errors	17	17.7%
Incomplete object errors	13	13.5%
Entity errors	12	12.5%

Compared to the analysis of the test set, a strong discrepancy in the proportion of context errors is observed. At 33.3%, context errors make up a much higher proportion than the 21% observed in the test set. The reason for this is that the pages in the test set were not particularly vulnerable to context errors, while the evaluation set contained some that were. One such case is the page "January_31" which, though consisting of only 41 sentences (about 1% of all sentences in the evaluation set), accounts for 11 context errors (11.5% of all false positives). In this page, a number of events are described. The fact that these events took place on a January 31th is only given by the context in the page, not mentioned in every single sentence. Without this context however, the information gathered is false in many cases. This demonstrates the disrupting effects context errors can have on the overall performance of Wanderlust.

In conclusion, all causes for precision loss are areas for future work. Many go hand in hand with causes for recall loss. Amelioration of the entity tagger or resolution of coreferences, for instance, both have positive effects both on precision and recall. In the estimation of the author, context errors as the most common cause for precision loss as well as incomplete object errors as a crucial incompleteness in the algorithm represent priorities for further development.

5.2 Qualitative Result Analysis

Unlike Section 5.1 in which recall and precision values are measured, this section attempts to make a qualitative analysis of the relations found by Wanderlust. One intention is to establish what kind of information is generally found within the Wikipedia corpus by Wanderlust in order to better understand the model of knowledge that is generated. To this end, the topmost common 30 predicates are listed and analyzed in 5.2.1. Another intention is to analyze the most commonly used linkpaths in order to portray what kind of grammaticality the algorithm is able to understand and what kind of predicates are generated from it. A number of linkpaths are analyzed in 5.2.2 according to what relations they produce and what problems may arise. In both sections, example special treatment rules are defined which can be used to improve both precision and recall of the result set. Some of these rules make use of the secondary dataset.

5.2.1 Relation Types

This section analyzes the types of relations that make up the generated knowledge base in order to determine its expressiveness. The topmost common 30 predicates of the result set are listed in Table 5.6. The ensuing subsections

qualitatively discuss the predicates according to their information content, the problem of ambiguity and the potential of identifying generally false predicates in order to raise overall precision.

Table 5.6: List of most common predicates in the result set.

Predicate	# Relations	Category
Is	627357	Taxonomic
Was	203237	Taxonomic
WasBornIn	50234	People
Has	25082	
Became	17479	Taxonomic
IsLocatedIn	14264	Location
IsVillageIn	13854	Location
IsTownIn	12464	Location
WasFoundedIn	11184	Institution
Won	10192	Institution / People
Attended	10046	People
Had	9281	
IsCityIn	8899	Location
Joined	8719	Institution / People
IsSpeciesOf	8702	Taxonomy / Biology
IsTributaryOf	7038	Location
IsGenusOf	6900	Taxonomy / Biology
IsCommuneIn	6350	Location
Defeated	6347	People / Institution
Played	6226	People
IsNameOf	5910	
IsTributaryIn	5771	Location
WasEstablishedIn	5594	Institution
IsVillageOf	5526	Location
WasElectedTo	5507	People
IsMemberOf	5403	Institution / People
Received	5371	
WasElectedIn	5126	People
Released	5068	

Information Content

The topmost common 30 predicates listed in Table 5.6 are placed into one or several categories if possible, depending on the type of information they pertain to. The category is indicated in the table. All relations that carry **taxonomic** information are placed in the accordingly named category. Some relation types carry taxonomic information in the domain of biology and are accordingly marked. A very dominant category is **location**. All predicates in this category state where the subject of a relation is located. Other dominant classes of relation types are those which pertain to **institutions** or **people**. This indicates that a large number of entities in the result set are either institutions, locations or people.

This information enables the manual generation of super-properties for some of the categories. One example is the definition of a super-property for all taxonomic relations, namely **IsA**. It encompasses the top two predicates, which account for over a third of all total relations, as well as the 5th most common predicate Became (which already is a subtype of Was). The taxonomies in the domain of biology, namely IsSpeciesOf and IsGenusOf are also part of this super-property. Another example is the location category. A wide variety of relation types exist in the entire corpus that are of this type. By defining one super-property for all these relations, queries for **LocatedIn** are significantly strengthened. A full list of all properties covered by these super-properties is given in Appendices B.5 and B.6.

Ambiguous Predicates

Ambiguity in predicates in general is a commonly observed phenomenon in the result set. Since predicates as generated by Wanderlust are simply a sequence of words, many of which use ambiguous verbs, it is often impossible to assign a single meaning to a relation type. One prominent group of ambiguity are predicates that can have both a time and a place as object. In the top 30 predicates examples are WasBornIn, WasFoundedIn, WasEstablishedIn and WasElectedIn. WasBornIn for example can be used with a location, as in WasBornIn(Albert_Einstein, Ulm), or with a year, as in WasBornIn(Albert_Einstein, 1879).

Disambiguation of these relation types can be performed by virtue of the subjects and objects of relations. The above mentioned group of predicates is disambiguated by virtue of their objects. A semantic query such as WasBornIn(?, Ulm) is an example of the predicate being disambiguated by the object being a place. A query for WasBornIn(Albert_Einstein, ?) however is not, meaning that it can be interpreted as either "*Where* was Albert Einstein born?" or "*When* was Albert Einstein born?".

This gravely affects the problem of defining a layer of logic for the relation types in the generated knowledge base. The result set consists of a very large amount of distinct relation types, many of which are ambiguous. To illustrate the difficulties of defining logic for ambiguous predicates consider the examples Attended and WasEducatedAt, two relation types which most commonly take a university or school as object. For purposes of semantic querying it is expedient to define one to be a subproperty of the other, since both relation types mostly convey synonymous information. This however is not possible because of the ambiguity of both predicates. The predicate Attended for example is also used in the context of "attending a meeting", such as in Attended(Abraham_Lincoln, Revival_Meetings). Since WasEducatedAt cannot be used in such a context, the two example predicates cannot be defined to be synonyms.

The conclusion of these observations is that because of predicate ambiguity it is very difficult to define a layer of logic for the result set. Without such a layer, the knowledge base cannot reach its full potential in terms of expressiveness.

False Predicates

While the previous section noted that most relation types are ambiguous, there are some that are always false. This can be the case in false positives of er-

ror graveness nonsensical, which often are a sequence of words that is without meaning. By deleting all relations containing nonsensical predicates, the overall precision of the result set can be raised.

As previously noted, one and two word predicates are unsuited for relations containing trivalent verbs. Predicates consisting of more than two words are vulnerable to incomplete object errors when using an optional instead of a necessary object in trivalent verbs, which can produce nonsensical predicates. In order to gauge the effects the identification and removal of false predicates can have on the result of Wanderlust, the topmost common predicates of length one were scanned. Problematic predicates can be identified and removed. Tentative removal of false predicates led to the dismissal of 19,589 relations from the result set. A detailed description of the procedure is given in 5.2.2.

5.2.2 Selected Linkpaths

This section introduces the most common linkpaths in order to illustrate the different types of grammatical patterns Wanderlust interprets. Linkpaths are grouped according to the word types of the predicates they generate. For each group, the most common linkpaths and their coefficients are given and example linkages illustrated. In most cases an analysis of the topmost common predicates is performed, either in order to manually remove relations using false predicates or to retrieve relations using valid predicates from the secondary dataset.

Subject-Verb-Object

The most common grammatical relationship for linkpaths is a simple subject-verb-object relation, in which two terms serve as subject and object to a verb and the verb is used as predicate in the relation triplet for the terms. As previously mentioned, the **S** link type denotes a subject-verb relationship, where the **s** and **p** subtypes indicate whether the subject is singular or plural (e.g. the *numerus*). Analogously, the **O** link type denotes an object-verb relationship, with the subtypes again indicating the numerus of the object. One important subtype here is **t**, which is used if the verb is a form of "to be". In the following, the most common linkpaths for this class of relation are listed with linkpath coefficients:

- {Ss, Os}, coefficient: 0.76
- {Ss, Ost}, coefficient: 0.89
- {Ss, Op}, coefficient: 0.76

Note that the linkpath {Ss, Ost} has a much better coefficient than the other linkpaths. The reason for this is that while {Ss, Os} and {Ss, Op} are vulnerable to incomplete object errors in cases where the verb is trivalent, {Ss, Ost} is not since it always employs a form of the divalent "to be", thus being mostly responsible for the relation types Is and Was. The discrepancy between these coefficients indicates the impact of the incomplete object error.

The 20 most common predicates found using this grammatical relationship are listed in Table 5.7. Note that most predicates are primarily divalent, which indicates that the impact of the incomplete object error is mostly centered

Table 5.7: List of most common predicates length one.

Predicate	# Relations	Category
Is	673129	Taxonomic
Was	218572	Taxonomic
Has	26923	
Became	19497	Taxonomic
Won	12028	Institution / People
Attended	10674	People
Had	10086	
Joined	9537	People
Defeated	7794	People / Institution
Played	6789	People
Received	5806	
Released	5540	
Left	5327	Institution / People
Made	5213	People
Are	4768	
Used	4010	People
Wrote	3542	People / Institution
Uses	3318	People
Entered	3302	
Founded	3232	

around more rare relation types. Manual analysis of the top 200 predicates of this class yielded 12 that were judged to be very vulnerable to incomplete object errors, such as Gave, Introduced and Asked. A total of 19,539 relations are formed using these predicates. A random probe of 200 relation triplets in this class revealed only 35 true positives and therefore a precision of 0.175, which encourages the deletion of these relations from the result set in order to raise overall precision.

Object-Verb-Subject

This section discusses the inversion of the linkpath {Ss, Ost}, namely {Ost, Ss}. A relation constructed using this linkpath connects the object of a verb to its subject by using the verb as predicate. This, intuitively, should not be possible given that verbs are usually unidirectional. It has been found however to produce valid relations in a significant number of times. With 18,268 relations in the result set it is the 15th most commonly used linkpath and the fifth most commonly used group of predicates by word types. It has a coefficient of 0.52, meaning that more than half of all relations found with this linkpath are valid. The high number of positive examples can be traced to a common sentence construction in is-phrases, when the concept entity is stated before the unique entity. An example of this is the sentence:

> "The national currency is the Norwegian krone."[1]

[1] Sentence from http://en.wikipedia.org/wiki/Norway, October 2008

See Figure 5.1 for a linkage of the sentence. In this sentence, the false relation Is(National_currency, Norwegian_krone) is extracted using the commonly valid linkpath {Ss, Ost}. However, because a concept entity may not serve as subject, the relation is rejected by Wanderlust. The relation Is(Norwegian_krone, National_currency) on the other hand uses a unique entity as subject and is therefore permitted.

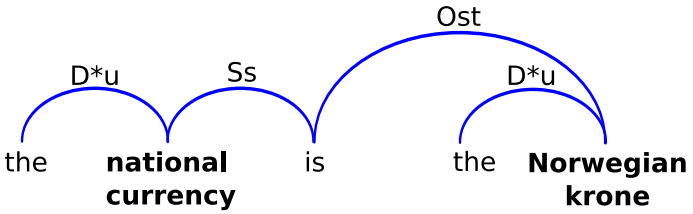

Figure 5.1: Example sentence with concept entity as subject.

This type of sentence construction is seen exclusively with a form of the verb "to be" as predicate, meaning that if the subject of the relation is a unique entity and the object a concept entity, this linkpath produces valid relations. If, on the other hand, both subject and object are unique entities this is not the case. Consider a sentence like

> "Julius Caesar was a Dictator Perpetuus, which was a highly irregular form of dictator, an official position in the Roman Republic."[2]

In this example, both the relations Was(Julius_Caesar, Dicator_Perpetuus) (using {Ss, Ost}) and Was(Dictator_Perpetuus, Julius_Caesar) (using {Ost, Ss} are found. Of the two, only the former is valid. Based on these observations, a special treatment rule can be defined to make {Ost, Ss} valid exclusively if the object of the relation is a concept entity. Using this rule, a total of 18,268 relations are found. A random probe of 200 relations has revealed a precision of 0.88 for the linkpath using the special treatment rule, opposed to 0.52 without.

Subject-Verb-Preposition-Object

Linkpaths in this class denote a subject-verb-preposition-object relationship, which is the second most common type of relationship found using Wanderlust. It occurs if a subject is connected to a verb which in turn is connected via a preposition to its object. The verb and the preposition form the predicate used to connect the subject to the object. This class of linkpaths has been found to have generally very low coefficients. The most common examples are:

- {Ss, MVp, Js}, coefficient: 0.18
- {Ss, MVp, Jp}, coefficient: 0.19
- {Ss, MVp, J}, coefficient: 0.16

[2]Sentence from http://en.wikipedia.org/wiki/Roman_Empire, October 2008

Because this class of relations is very common it merits a closer look in spite of linkpath coefficients below 20%. Consider two sentences: "Austria and Prussia fought together against Denmark" and "Zeus transformed Chelone into a tortoise". See Figures 5.2 and 5.3 for their linkages. When applying the rule {Ss, MVp, Js} to both link grammar parses, three relations are extracted. The two relations from the first sentence are FoughtAgainst(Austria, Denmark) and FoughtAgainst(Prussia, Denmark), both of which are valid. The relation from the second sentence is TranformedInto(Zeus, Tortoise), which is false.

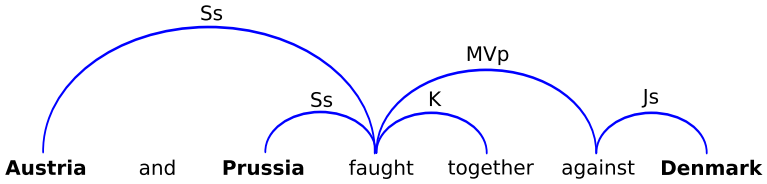

Figure 5.2: Example sentence. Entities are highlighted bold.

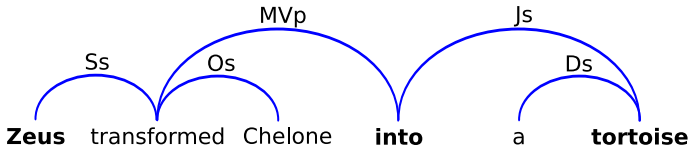

Figure 5.3: Example sentence. Entities are highlighted bold.

Once again the problem is one of verb valency. The verb "to fight against someone" is divalent using a predicate, which is a match for this relation type. The verb "to transform" on the other hand, as meant in the context of the second sentence, namely "transform someone into something", is a trivalent verb. The relation type connects the subject to the indirect object, yielding the false relation TranformedInto(Zeus, Tortoise). But while in subject-verb-object the highest number of predicates use divalent verbs, this class of relations is most often found using trivalent verbs or instances of divalent verbs connected to an optional instead of a necessary argument, resulting in a low overall coefficient. Consider the topmost common predicates of length 2 listed in Table 5.8. Only 3 of the top 20 predicates can be used in this class of relations, while all others are incomplete.

While generally incorrect, this class of relations can have a high accuracy when using only relations with divalent verbs that are connected to a necessary object via a preposition. Because of this observation, all relations in this class were saved into a database table of the secondary dataset. When manually scanning the topmost common 500 predicates, a total of 120 predicates are found which are unlikely to be incomplete. They include the three predicates marked in Table 5.8 and are used in **175,703 relations**. An analysis of a set of 200 random relations using these predicates have found 171 to be correct, indicating an accuracy high enough to justify transferring these relations to the SMW Database.

Table 5.8: List of most common predicates length two.

Predicate	# Relations	Usable
IsIn	337558	
IsBy	64972	
WasIn	57758	
IsFor	38847	
IsFrom	30847	
WasFrom	28241	
IsOn	22096	
IsTo	21531	
WasTo	18463	
WasFor	16544	
IsAt	13430	
IsWith	12858	
WasBy	11312	
MovedTo	9958	yes
WasAt	8836	
WasDuring	8480	
IsBetween	8464	
DiedIn	8371	yes
ReturnedTo	7954	yes
AreIn	7824	

Subject-Verb-Noun-Preposition-Object

This is the third most common class of relation and has already been used in examples of previous chapters. It occurs in cases when a verb takes three arguments, namely a subject and two objects. Note that this must not mean that the verb is trivalent, since a divalent verb can also bind an optional argument to itself. The first object is made part of the linkpath connecting subject to second object. The following lists the most common linkpaths for this class of relation as found in the test set:

- {Ss Ost Mp Jp}, coefficient: 0.64
- {Ss Ost Mp Js}, coefficient: 0.71
- {Ss Os Mp Jp}, coefficient: 0.6

The linkpaths {Ss Ost Mp Js} and {Ss Ost Mp Jp} have the highest linkpath coefficients, which can be attributed to the link label **Ost**. It indicates that a form of the usually divalent verb "to be" is used in the linkpath, therefore limiting errors due to verb valency. Other linkpaths in this class are vulnerable to incomplete object errors in cases where a trivalent verb is used in the linkpath using an optional instead of a necessary object. However, such mistakes are limited mostly to rare relation types. Consider an overview of the 20 topmost common relation types of this class listed in Table 5.9, of which not a single predicate is subject to the incomplete object error and use a form of the verb "to be". Within the topmost common 400 predicates of this class only 14 use a different verb. This can be explained by the fact that other linkpaths in this

class use arbitrary verbs, causing predicate dispersion, while {Ss Ost Mp Js} and {Ss Ost Mp Jp} concentrate only on the verb "to be". Incomplete object errors are therefore more likely to affect rare relation types.

Table 5.9: List of most common predicates length three with noun.

predicate	# relations
IsVillageIn	13854
IsTownIn	12464
IsCityIn	8899
IsSpeciesOf	8702
IsTributaryOf	7038
IsGenusOf	6900
IsCommuneIn	6350
IsNameOf	5910
IsTributaryIn	5771
IsVillageOf	5526
IsMemberOf	5403
IsMunicipalityIn	4512
IsAlbumBy	4291
IsTownOf	3944
WasMemberOf	3790
IsTownshipIn	3341
IsDistrictOf	3138
IsCensus-designatedPlaceIn	2924
WasSonOf	2659
IsMunicipalityOf	2554

This class of relations is also vulnerable to an error which affects all predicates that carry a noun. Problems arise if the noun in the predicate is a coreference, making the predicate incorrect or less useful. An example of this can be found in the sentence "*East Germany renamed its national airline to Interflug, which ceased operations in 1991*" where the relation RenamedAirlineTo(East_Germany, Interflug) is found, in which the predicate RenamedAirlineTo carries a coreference to a specific airline. This information is lost, reducing the usefulness of the relation.

Subject-AuxiliaryVerb-Participle-Preposition-Object

This is the fourth most commonly occurring class of relations. It occurs when a subject is linked via a participle and its auxiliary verb to an object. The most common linkpaths use the link type **P**, which links forms of the verb "to be" to words which can be used as complements, such as prepositions, adjectives and participles. The first term of the predicate is therefore a form of the verb "to be" used as auxiliary verb which is dropped from the predicate in temporal normalization. The most common linkpaths of this class are listed with their coefficients in the following:

- {Ss Pv MVp Jp}, coefficient: 0.691
- {Ss Pv MVp Js}, coefficient: 0.7

- {Ss Pvf MVp Jp}, coefficient: 0.72

An example of this class of relations is given with the sentence "*Traditionally, sauerkraut is prepared in a stoneware crock*". A linkage for this sentence is illustrated in Figure 5.4. The link label **Pv** connects forms of the verb "to be" to passive participles. **CO** is used to connect "openers" such as "traditionally" to nouns such as "sauerkraut", as used in the sentence. The link label **Xc** connects words to commas to its right. Using the linkpath {Ss Pv MVp Jp} the relation PreparedIn(Sauerkraut, Stoneware_crock) is found in the linkage (note that the first term of the predicate IsPreparedIn is dropped due to normalization).

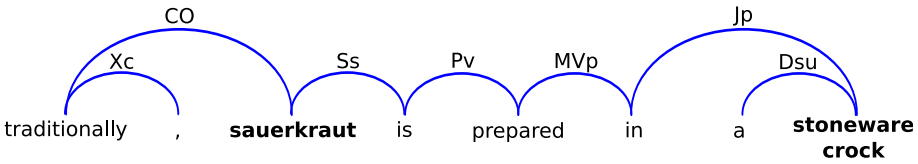

Figure 5.4: Example sentence for the linkpath {Ss Pv MVp Jp} with entities highlighted bold.

For an overview of the topmost common predicates found in this class refer to Table 5.10. The table is a good example of the diversity of relation types Wanderlust finds.

Table 5.10: List of most common predicates length three with participle.

Predicate	# Relations
WasBornIn	50234
IsLocatedIn	14246
WasFoundedIn	11183
WasEstablishedIn	5593
WasElectedTo	5504
WasElectedIn	5120
WasEducatedAt	4568
WasFormedIn	3538
WasReleasedIn	3489
WasElectedAs	3326
WasBuiltIn	3058
WasFoundedBy	2956
WasCreatedIn	2744
WasBornAt	2684
WasReplacedBy	2506
WasRaisedIn	2485
WasBornOn	2348
WasNamedIn	2300
IsServedBy	2262
WasNominatedFor	2230

Subject-AuxiliaryVerb-Adjective-Preposition-Object

This class of relations is similar to the class introduced in the previous subsection. Instead of a participle, a predicative adjective is used in the linkpath. An example of this can be observed in the example sentence "*Austria is also famous for its Apfelstrudel*". Its linkage is illustrated in Figure 5.5. The link label **Pa** connects a small number of verbs to predicative adjectives. The link label **EBm** is for connecting adverbs to forms of "to be" before object, adjective or prepositional phrases. Using the linkpath {Ss Pa MVp Jp} the (normalized) relation FamousFor(Austria, Apfelstrudel) is extracted.

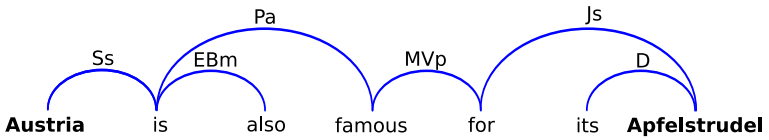

Figure 5.5: Example sentence for the linkpath {Ss Pa MVp Jp} with entities highlighted bold.

Subject-Verb-Preposition-Object-Preposition-Object

This class of relations is most common for relations of length 4. It models a relationship in which a verb is directly connected to a subject and indirectly to two objects via a preposition. An example of such a verb (or verb phrase) is "to fall in love with someone". Because no connection to an object without a preposition is made, analogously to subject-verb-preposition-object, this class of relations is vulnerable to incomplete object errors. The following linkpaths are common occurrences in the test set:

- Ss MVp Jp Mp Jp, coefficient: 0.12
- Ss MVp Jp Mp Js, coefficient: 0.32
- Ss MVp Js Mp Jp, coefficient: 0.23
- Ss MVp Js Mp Js, coefficient: 0.33

All linkpaths have coefficients ranging from 0.12 to 0.33, meaning that all relations of this class are dismissed. Because there are numerous occurrences however, they merit a closer look. An example of a relation extracted using one of the above linkpaths can be found in "*Coronis fell in love with Ischys, the son of Elatus*". A linkage for the important part of the sentence is given in Figure 5.6. The valid relation FellInLoveWith(Coronis, Ischys) is found.

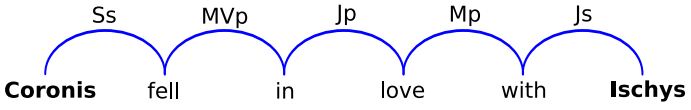

Figure 5.6: Example sentence for the linkpath {Ss MVp Jp Mp Jp}.

Since this linkpath does not work with the largest part of verbs and verb phrases, a similar special treatment rule as for subject-verb-preposition-object

is used. Relations of this class are stored in the secondary dataset and the most common predicates are manually scanned. Refer to Table 5.11 for the topmost common 20 predicates of this class. One interesting observation in these predicates is that each can be defined as subproperty of LocatedIn.

Table 5.11: List of most common predicates length four.

Predicate	# Relations
IsInDistrictOf	9216
IsInStateOf	4589
IsInRegionOf	4380
IsInDepartmentIn	4073
IsInProvinceOf	3472
IsInIndianStateOf	2860
IsInAdministrativeDistrictOf	2105
IsInDepartementIn	2013
IsInCantonOf	1932
IsInCantonIn	1688
IsInDepartementOf	1637
IsInCityOf	1555
IsInDutchProvinceOf	1389
IsInMunicipalityOf	1322
IsInStateIn	1207
IsInDepartmentOf	1002
IsInCountyOf	965
IsInAreaOf	927
IsInTownOf	784
IsInDistrictIn	727

After scanning the topmost 300 predicates, a total of 217 were deemed as not vulnerable to incomplete objects. Together, they are used in 76,071 relations. A tentative manual evaluation of 200 random relations showed 169 to be valid, indicating an accuracy high enough to justify inserting these relations into SMW. Other examples of using the secondary dataset to increase overall recall are given in the ensuing section.

5.2.3 Secondary Dataset

The secondary dataset stores, as previously noted, a part of all dismissed relations. While no full analysis of this set is performed, this section demonstrates how using some simple methods suffices to retrieve a portion of these relations. In total, **410,567 relations** are retrieved from the secondary data set this way. The dataset is divided into three categories.

The **incomplete object set** contains all relations of classes subject-verb-preposition-object and subject-verb-preposition-object-preposition-object due to their vulnerability to incomplete object errors as discussed in 5.2.2. As was illustrated, manual identification of the most common predicates in these classes that are unlikely to be affected by incomplete object errors has yielded **251,775** formerly dismissed relations that can be added to the result set.

The **abstract linkpath set** contains all relations that needed to be validated using abstract linkpaths because of a lack of examples in the training set for regular linkpath validation. A portion of all dismissed relations from precision loss class rule are arguably in this set. One possible way of retrieving this data is by manually analyzing the topmost common abstract linkpaths and gauge their precision. Relations that are validated using an abstract linkpath with sufficient precision can be added to the result set. Tentative analysis of the topmost common abstract linkpaths found four with a reasonable precision: {S O M J}, {S O M MV IN}, {S PP O M J} and {PF SI M J}. Together, they encompass **158,792** relations that can be added to the result set.

The **low confidence set** contains all relations validated using linkpaths with confidence values lower than 0.5. This set exists for the purpose of further analysis and identification of features that enable the retrieval of lost relations. One possible way of raising precision of the low confidence set might be to extract only relation types from this set that are also found in the result set.

5.3 Summary

This chapter has performed a quantitative and qualitative analysis on the results produced by Wanderlust applied to the English Wikipedia. Precision and recall values based on different considerations have been calculated. Depending on the definition, recall was found to be 7,5% or 20,4%. Precision was put at 62%, 82% or 91% depending on the definition of true and false positives. The generated knowledge base was found to contain information centered around people, places and institutions. A high number of predicates were found, many of which are ambiguous, making the definition of a layer of logic very problematic for the result set.

The most commonly found grammatical patterns that Wanderlust manages to interpret were illustrated. The secondary dataset has been tentatively used to manually raise precision and recall values, indicating a direction of future work. The next chapter gives an overall summarization of this thesis as well as an outlook on future work.

Chapter 6

Conclusions

This section summarizes the thesis and makes overall conclusions based on the analysis of the result set and observations of challenges to the validation procedure. An outlook into future work is given.

6.1 Original Theory

This thesis analyzed the theory that there are universally valid grammatical patterns for explicitly stated semantics in sentences in plain text. An algorithm aware of these patterns can according to this theory extract arbitrary semantics from grammatically correct sentences. The deep grammatical formalism used in this thesis is link grammar, in which such grammatical patterns in the form of linkpaths were sought. Coefficients were calculated for linkpaths using an annotated corpus with the intent of identifying generally valid linkpaths. While a total of 43 linkpaths have been identified with acceptable coefficients, no linkpaths with coefficients high enough to suggest that they are generally valid have been found.

An analysis performed on a test corpus revealed a number of reasons for precision loss, explaining the coefficients. Most of the problems are computerlinguistic issues which, while affecting the calculation of coefficients, do not challenge the initial theory. Examples of this are the disambiguation of entities and the resolution of coreferences. Both issues are areas of ongoing research and are addressed in this thesis by using heuristics.

A difficult problem encountered in this thesis are context errors, which occur when information is extracted from a sentence without its context. Outside the context given by the text in which relations are found they may be untrue, which severely affects the precision of the algorithm. As shown, context errors are observed in many forms, some of which are a result of viewing text at sentence granularity.

Most importantly, the analysis on the test set revealed a principal incompleteness to the method as implemented in this thesis. The problems stems from searching for grammatical patterns within the link grammar formalism, which makes no distinction between necessary and optional objects of verbs. This information is therefore not reflected in the grammatical patterns identified in this thesis. It was shown how this error disrupts the calculation of linkpath co-

efficients and lowers the performance of relation extraction using linkpaths. In order to address this problem, a verb sense disambiguation must be performed which enables the algorithm to distinguish between necessary and optional objects of a verb. This has been judged to lie outside the scope of the thesis, but represents a priority in future work.

In conclusion, the initial theory is supported with the results presented in this thesis. While the information on how words within a sentence relate to each other as provided by the link grammar formalism has been shown to be incomplete, the patterns identified in this thesis nevertheless have resulted in the generation of a large knowledge base with acceptable precision values.

6.2 Semantic Wiki

In this thesis, an algorithm dubbed Wanderlust has been created that uses the 43 linkpaths identified as representing valid grammatical patterns to extract subject-predicate-object triplets from sentences in plain text. Wanderlust was applied to the corpus of the English Wikipedia with the stated goal of generating a knowledge base for the creation of a semantic wiki. This was to serve as a specific use case for the approach, enabling a detailed analysis of challenges and results. The Semantic MediaWiki platform was used to store the subject-predicate-object triplets generated by Wanderlust, which enables the use of features such as semantic querying, allowing the result of the thesis to be used as a question-answering system. A total of 2,64 million relation triplets are generated from the corpus using 1,38 million distinct entities.

The knowledge base consists of a large number of distinct relation types, which has both positive and negative effects on usability. On the positive side, users can pose semantic queries or define semantic page links intuitively, i.e. without first learning a fixed set of predicates. The concept of free word choice in predicates allows relation triplets to precisely match the semantics stated within a sentence. On the negative side, a large number of predicates makes the task of adding a layer of logic to the knowledge base more complicated and in many cases arguably impossible. Problems arise because the predicates generated using Wanderlust are vulnerable to ambiguity. As illustrated in this thesis, it is difficult to define logic for a predicate which can have many different meanings. This inhibits the semantic wiki from reaching higher levels of expressiveness.

To address the difficulties caused by predicate diction, a normalization procedure was written which generated 749,703 subproperties that order predicates along a chain of implication. The core feature of semantic queries is aided by these subproperties, since a query for one relation type will query for all its subproperties as well. The relations and subproperties generated in this thesis have resulted in the automatic creation of a large semantic wiki.

6.3 Future Work

This section concludes the thesis with an outlook on future work. A number of reasons for recall and precision loss have been identified and discussed. In order to ameliorate the performance of Wanderlust, these issues need to be addressed.

In terms of recall, priorities in future work include adding a method for the

resolution of coreferences and improving the entity tagger. The former accounts for a very large part of recall loss since terms that may be coreferences are not used as subjects in relation triplets by Wanderlust. By including a method for coreference resolution, this heuristic can be abandoned. With regards to the entity tagger, future work must focus on adding alternative methods of disambiguating terms without page links.

In terms of precision, methods for addressing context errors need to be developed. This may include extending the subject-predicate-object relation triplet model to allow for a relation to specify the context within which it is valid. Because context errors have been observed in a multitude of forms some of which cannot be detected at sentence granularity, the detection of context errors may be an extensive area of future work.

The information conveyed by the link grammar formalism has been found to be incomplete for the purpose of general validation. Verbs and verb phrases need to be identified and disambiguated in order to ascertain which objects are necessary in terms of verb valency. Addressing this incompleteness in the algorithm is an important priority for future work with regards to the initial theory of this thesis.

Appendix A

Tools

This appendix covers the two tools created for the purpose of assisted annotation.

A.1 Sentence Annotator Tool

The Sentence Annotator Tool is used as an analyzing tool for sentence by sentence annotation of semantics. A user can enter the name of a page and click "look up" which causes the Wikipedia page to be loaded into the tool. Weitblick entity tagging is performed and all Wikipedia markup removed. The user can iterate through the sentences of the page by clicking "next" and "previous" (see Figure A.1).

Each sentence is analyzed by Wanderlust. Linkages and entities are displayed. Information on entities and attributes such as its start position (word number in sentence), its length (number of words the entity comprises of) and whether it is a unique or a concept entity is displayed as illustrated in Figure A.2. In the linkage box illustrated in Figure A.3, the linkages are displayed with their quality-of-parse information, as well as a POS-tag parse of the sentence. In the relations box illustrated in Figure A.4, all unvalidated relations found by Wanderlust are listed. A user must confirm those that are valid, all others are automatically dismissed. The annotation of the sentence is stored in a database.

This tool may be used not only for annotation, but also for observation of behavior of Wanderlust and link grammar towards certain types of sentences for purposes of debugging and general analysis.

A.2 Stiefeletten Bootstrapping Suite

The Stiefeletten Bootstrapping Suite is designed for the second phase of annotation, after a reasonable level of confidence in Wanderlust has been established using SAT annotation. The tool performs a full Wanderlust analysis of an entire page. The results of the analysis are listed in several tabs.

1. The tab **relations** (see Figure A.6) lists all positives returned by Wanderlust. The coefficient threshold and other validation variables including the quality-of-parse variables, the maximum distance between subject and

object and the maximum number of words in a sentence can be set in Figure A.7. Distinct values for linkpaths of different lengths are possible. A change in these variables affects the list of returned relations.

2. The tab **known** lists all relations that are already known by virtue of the fact- and antifactbase. Accordingly, the tab **unknown** lists all relations that are not yet part of the knowledge base.

3. The tab **suggested** is the main tab for the annotation purpose. All relations that meet the thresholds set in Figure A.5 are listed here. A number of low confidence relations are by virtue of the thresholds automatically dismissed. The user must confirm those relations in the list that are valid in order to annotate the page.

4. The tabs **false correct** and **false nonsensical** are used if the page is already annotated. The relation found by Wanderlust are automatically tested against the annotation and all false positives and false negatives listed along with details on the linkpath used for extraction. See Figure A.8 for an illustration of this.

A number of additional functions have been added to the suite. The tab **title entity** lists all synonyms of the title entity as found by Weitblick. In the tab **Wiki Tags**, semantic page links as formed in SMW are generated and can be added to pages. The tab **autofind** can be used to enter a list of page names in order to bootstrap or analyze a group of pages at once.

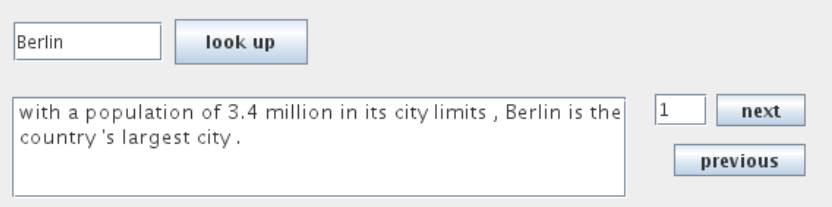

Figure A.1: SAT: Screenshot of the look up and sentence scroll area.

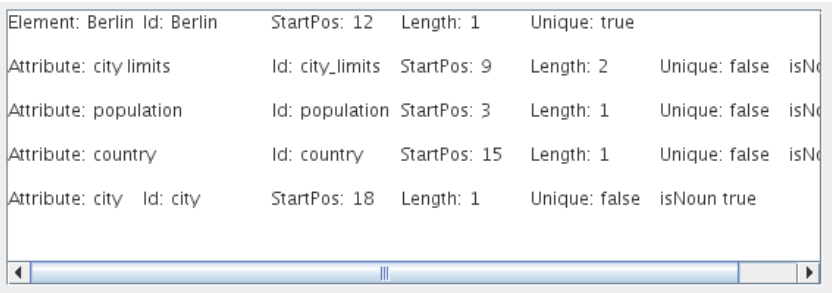

Figure A.2: SAT: Screenshot of the entity box, showing a list of all entities and attributes found for the sentence looked up in Figure A.1.

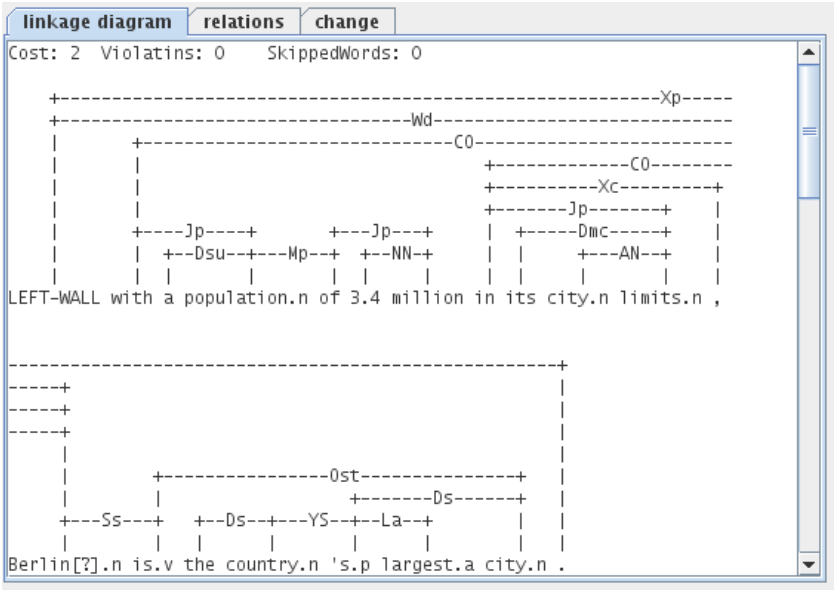

Figure A.3: SAT: Screenshot of the linkage box showing linkages and quality-of-parse information for the sentence.

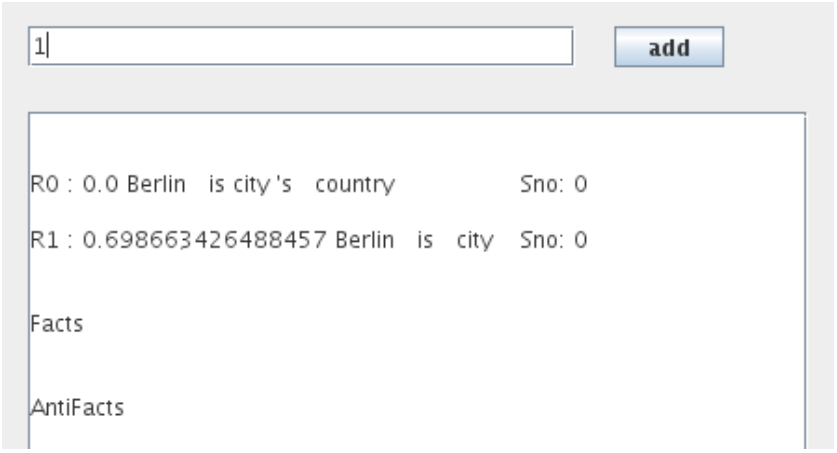

Figure A.4: SAT: Screenshot of the relation box showing all relations found preceded by its linkpath coefficient.

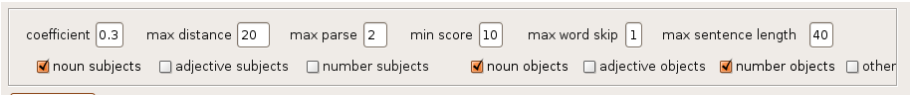

Figure A.5: Bootstrapper: Main threshold box for Wanderlust. Note that the main threshold variable *coefficient* is set very low to ensure that only relations with very low confidence values are dismissed out of hand.

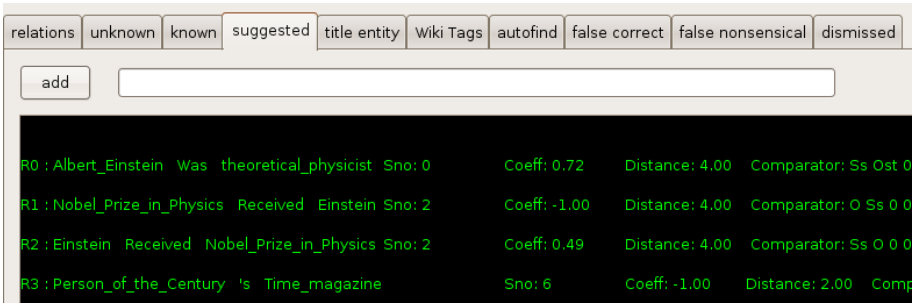

Figure A.6: Bootstrapper: List of all suggested relations and their coefficients. Unknown linkpaths have a coefficient of -1 and are not automatically dismissed. Relations found using low coefficient linkpaths are filtered out beforehand. A user must confirm the relations on this tab in order to annotate the page.

Figure A.7: Bootstrapper: Tabbed pane where thresholds can be set for relations of a specific length.

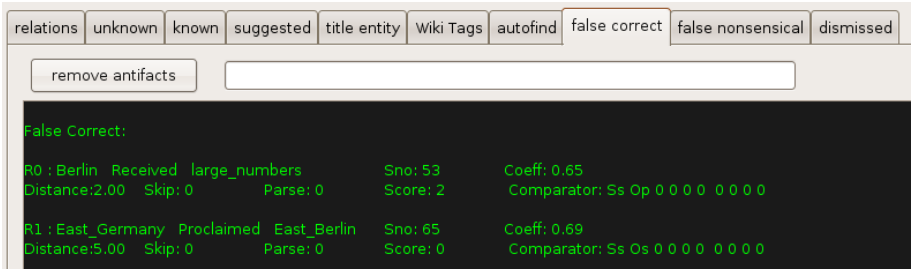

Figure A.8: Bootstrapper: List of all false positives found by Wanderlust. In case of false annotation, the annotation can be removed using the 'remove antifacts' button.

Appendix B

Tables

This appendix covers a number of tables.

A list of all POS-tags used in examples in this thesis and a short description is given in B.1. Similarly a list of all link types used in the thesis is given in B.2 along with a very short description of their meaning. Those links that indicate particle groups are accordingly marked. For a detailed explanation of all link types refer to [36].

Table B.3 gives an overview over all observed features for a relation. Table B.4 lists all 43 valid linkpaths identified in this thesis, along with the number of relations that were found with each. The two semantic normalization super-properties IsA and LocatedIn and their subproperties are given in B.5 and B.6.

Table B.1: List of POS-tags in the Penn Treebank POS Tagset.

POS-tag	Description
NNP	Proper noun
VBZ	Verb, 3rd person singular present
DT	Determiner
NN	Noun, singular or mass
IN	Preposition or subordinating conjunction
JJ	Adjectives

Table B.2: List of link types. Particle groups are marked.

Link label	Description	Particle group
A	Pre-noun adjective–noun relationship	
Cet	Verb–clausal complement relationship,	
CO	Opener–subject relationship	
D	Determiner–noun relationship	
Ds	Determiner–noun, noun singular	
DG	Determiner–noun, noun proper noun	
D*u	Determiner–noun, noun uncountable singular	
Dsu	Determiner–noun, noun singular and uncountable	
Dmc	Determiner–noun, noun countable plural	yes
Dmu	Determiner–noun, noun uncountable plural	yes
EA	Adverb–adjective relationship	
EBm	Adverb–Verb "to be" relationship	
ID	Special class of link types to connect idiomatic strings	yes
IDXK	Idiomatic string "referred to"	yes
J	Preposition–object relationship	
Js	Preposition–object, object singular	
Jp	Preposition–object, object plural	
K	Verb–particle relationship	yes
La	Determiner–superlative adjective relationship	
Mp	Noun–prepositional phrases modifying nouns relationship, observed mainly as noun–preposition relationship	
MVp	Verb–modifying phrase relationship, observed mainly as verb–preposition relationship	
MX*r	Noun–relative pronoun after comma relationship	
N	"Not"–auxiliary verb relationship	yes
ND	Connects numbers with certain expressions	yes
NR	Fraction word–superlative relationship	yes
O	Object–verb relationship	
Os	Object–verb, object singular	
Op	Object–verb, object plural	
Ost	Object–verb, object singular, verb form of "to be"	
Pa	Verb–predicative adjective relationship,	
Pv	Verb "to be"–passive participle relationship	
Pvf	Verb "to be"–passive participle, "filler it" as subject	
S	Subject–verb relationship	
Ss	Subject–verb, subject singular	
Ss*I	Subject–verb, subject first person singular	
Ss*w	Subject–verb, subject relative pronoun	
Sp	Subject–verb, subject plural	
Spx	Subject–verb, subject plural, verb form of "to be"	
SFs	Special subject link-type used for certain "filler" subjects like "it" and "there"	
THi	Word–"that" clause complement relationship	
Xc	Word–comma to a right relationship	
Xd	Word–comma to a left relationship	

Table B.3: All observed features for a relation triplet.

Feature	Description
Subject ID	Internal identifier for subject
Subject namespace	Wikipedia namespace, all pages are namespace 0
Subject title	Page title of the subject
Relation title	Title of the predicate
Object ID	Internal identifier for object
Object namespace	Wikipedia namespace, all pages are namespace 0
Object title	Page title of the object
Extraction rule	Rule (linkpath) used to extract relation
Found in	Title of page in which the relation was found
Skip	Number of skipped words in the linkage
Score	Score of the linkage
Linkage	Linkage number
Extended relation	Predicate with placeholder for extended noun if exists
Extended entity text	Text of extended noun if exists
Extended entity POS-tags	POS-tags of extended noun if exists
Links of linkpath	Fields for each link label in the linkpath
POSTags of wordpath	Fields for each POS-tag in the wordpath

Table B.4: List of valid linkpaths with number of relations found in the result set.

linkpath	# relations
Ss Ost	801,140
Ss Ost Mp Js	436,346
Ss Os	322,657
Ss Pv MVp Js	240,055
Ss Ost Mv MVp Js	124,211
Ss Ost Mp Jp	114,825
Ss Os Mp Js	86,402
Ss Pv MVp Jp	72,264
Ss Pv MVp IN	69,465
Ss Pv MVp Js MX	40,448
Ss Pv MVp J	37,727
Ss Os Mp Jp	35,951
Ss Op	34,548
Ss Op Mp Js	26,864
S Os	25,629
Ost Ss	18,268
Ss Pa MVp Js	17,241
Ss Pa MVp Jp	15,686
Ss Os MXs	14,857
Ss Pvf MVp Js	14,725
Ss PP Os	14,535
Ss Pv Os	12,456
Ss TO I Os	12,301
Ss Op Mp Jp	10,786
Ss Pvf MVp Jp	8,455
Ss Op MXp	6,507
Ss Pv MVp Js Mp Jp	6,490
Ss OF Jp	4,606
Ss Pa	3,946
Ss O Mp Js	3,603
Ss PPf Ost	2,989
Ss O Mp Jp	2,528
PF SIs	1,642
Ss PPf Os	1,535
MXsp MVp Jp	1,388
Ss Pvf Os	1,188
Ss PPf Ost Mp Jp	1,010
Bs Os	798
R RS Os	245
MXsr S**w Os	219
MXsr S**w MVp Jp	166
MXsr Ss*w Pv MVp Jp	130
Ss K	9
Total	2,646,841

Table B.5: Subproperties of IsA.

Is
Was
KindOf
TypeOf
OneOf
GenusOf
SpeciesOf

Table B.6: Subproperties of LocatedIn.

LocatedIn
CityIn
TownIn
VillageIn
BridgeIn
ValleyIn
TributaryIn
InDistrictOf
InStateOf
InRegionOf
InDepartmentIn
InProvinceOf
InDepartementIn
InCantonOf
InCantonIn
InDepartementOf
InCityOf
InMunicipalityOf
InStateIn
InDepartmentOf
InCountyOf
InAreaOf
InTownOf
InDistrictIn

List of Figures

List of Tables

Bibliography

[1] E. Agichtein and L. Gravano. Snowball: extracting relations from large plain-text collections. pages 85–94, 2000.

[2] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. *Lecture Notes in Computer Science*, 4825:722, 2007.

[3] M. Banko, M. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction from the web. *Proceedings of IJCAI*, 2007.

[4] T. Berners-Lee, J. Hendler, O. Lassila, et al. The semantic web. *Scientific American*, 284(5):28–37, 2001.

[5] S. Brin. Extracting patterns and relations from the world wide web. *Lecture Notes in Computer Science*, pages 172–183, 1999.

[6] E. Britannica. Inc. Fatally flawed: Refuting the recent study on encyclopedic accuracy by the journal Nature, 2006.

[7] A. Broder, M. Fontoura, V. Josifovski, and L. Riedel. A semantic approach to contextual advertising. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 559–566. ACM New York, NY, USA, 2007.

[8] R. Bunescu, R. Ge, R. Kate, E. Marcotte, R. Mooney, A. Ramani, and Y. Wong. Comparative experiments on learning information extractors for proteins and their interactions. *Artificial Intelligence in Medicine*, 33(2):139–155, 2005.

[9] W. Cunningham. Wiki design principles. *Available from Internet:< http://c2.com/cgi/wiki?WikiDesignPrinciples>. Access 02/28/09*, 2008.

[10] R. Doorenbos, O. Etzioni, and D. Weld. A scalable comparison-shopping agent for the World-Wide Web. In *In Proceedings of the First International Conference on Autonomous Agents*, 1997.

[11] P. Elango. Coreference resolution: A survey. *Project report of the course Ädvanced natural language processingïn Computer Science Departments, University of Wisconsin Madison*, 2006.

[12] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. Web-scale information extraction

in knowitall:(preliminary results). In *Proceedings of the 13th international conference on World Wide Web*, pages 100–110. ACM New York, NY, USA, 2004.

[13] S. Garner. Weka: The waikato environment for knowledge analysis. In *Proc. of the New Zealand Computer Science Research Students Conference*, pages 57–64, 1995.

[14] R. Ghani, K. Probst, Y. Liu, M. Krema, and A. Fano. Text mining for product attribute extraction. *ACM SIGKDD Explorations Newsletter*, 8(1):41–48, 2006.

[15] J. Giles. Special Report–Internet encyclopaedias go head to head. *Nature*, 438(15):900–901, 2005.

[16] M. Jansche and S. Abney. Information extraction from voicemail transcripts. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 320–327. Association for Computational Linguistics Morristown, NJ, USA, 2002.

[17] K. Kessel. Reimann, Sandra (2005): Basiswissen Deutsche Gegenwartssprache. *Tübingen und Basel: Francke, S*, pages 91–125.

[18] O. Khriyenko and V. Terziyan. Context description framework for the semantic web. In *Proceedings Context 2005 Context representation and reasoning workshop, Paris (FR)*, 2005.

[19] D. Klein and C. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics*, pages 423–430. Association for Computational Linguistics Morristown, NJ, USA, 2003.

[20] T. H. Kotaro Nakayama and S. Nishio. Wikipedia link structure and text mining for semantic relation extraction. In *Proceedings of the Workshop on Semantic Search (SemSearch 2008) at ESCW 2008*, volume CEUR Workshop Proceedings, pages 59–73, Tenerife, Spain, June 2008.

[21] M. Krötzsch, D. Vrandecic, and M. Völkel. Semantic mediawiki. In *Proc. 5th International Semantic Web Conference (ISWC06)*, pages 935–942. Springer.

[22] M. Krötzsch, D. Vrandecic, and M. Völkel. Wikipedia and the semantic web-the missing links. In *Proceedings of WIKIMANIA*, volume 2005. Institute AIFB, University of Karlsruhe, Germany, 2005.

[23] O. Lassila and R. Swick. Resource description framework (rdf) model and syntax specification. w3c recommendation, 1999. *Available from Internet:< http://www. w3. org/TR/REC-rdf-syntax/>. Access 02/28/09*, 4, 2001.

[24] H. Liu and P. Singh. ConceptNet - a practical commonsense reasoning tool-kit. *BT Technology Jsournal*, 22(4):211–226, 2004.

[25] M. Marcus, M. Marcinkiewicz, and B. Santorini. Building a large annotated corpus of English: The Penn Treebank. *Computational linguistics*, 19(2):313–330, 1993.

[26] I. Mel'čuk. *Dependency syntax: theory and practice*. State University of New York Press, 1988.

[27] G. Miller. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

[28] R. Navigli. Word sense disambiguation: a survey. 2009.

[29] V. Ng. Shallow semantics for coreference resolution. In *Proc. IJCAI*, 2007.

[30] C. Plake, T. Schiemann, M. Pankalla, J. Hakenberg, and U. Leser. AliBaba: PubMed as a graph. *Bioinformatics*, 22(19):2444, 2006.

[31] S. Ponzetto and M. Strube. Deriving a large scale taxonomy from wikipedia. In *Proceedings of the national conference on artificial intelligence*, volume 22, page 1440. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[32] F. Popowich. Using text mining and natural language processing for health care claims processing. *ACM SIGKDD Explorations Newsletter*, 7(1):59–66, 2005.

[33] S. Sarawagi. *Information Extraction*. Now Publishers Inc., Indian Institute of Technology, SCE, Mumbai 400076, India, 2008.

[34] R. Schenkel, F. Suchanek, and G. Kasneci. YAWN: A semantically annotated Wikipedia XML corpus. *Kemper, A., Schoning, H., Rose, T., Jarke, M., Seidl, T., Quix, C., and Brochhaus, C., editors*, 12:277–291, 2007.

[35] G. Schneider. A linguistic comparison of constituency, dependency and link grammar. *Zurich, Switzerland: Licentiate Thesis, University of Zurich*, 1998.

[36] D. Sleator. Link grammar guide-to-links, available from internet:< http://www.link.cs.cmu.edu/link/dict/index.html>. last access: 02/26/2009.

[37] D. Sleator and D. Temperley. Parsing english with a link grammar. In *Third International Workshop on Parsing Technologies*, 1993.

[38] F. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. pages 697–706, 2007.

[39] F. M. Suchanek, G. Ifrim, and G. Weikum. Combining linguistic and statistical analysis to extract relations from web documents. pages 712–717, 2006.

[40] D. Temperley, D. Sleator, and J. Lafferty. Abiword-word processor for everyone.

[41] K. Toutanova and C. Manning. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics-Volume 13*, pages 63–70. Association for Computational Linguistics Morristown, NJ, USA, 2000.

[42] S. Weiss, N. Indurkhya, T. Zhang, and F. Damerau. *Text Mining: Predictive Methods for Analyzing Unstructured Information*. Springer, 2004.

[43] F. Wu and D. Weld. Autonomously semantifying wikipedia. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 41–50. ACM New York, NY, USA, 2007.

[44] X. Zhu. Semi-supervised learning literature survey. *world*, page 11, 2008.