LEXICAL DATABASE ENRICHMENT THROUGH SEMI-AUTOMATED MORPHOLOGICAL ANALYSIS

Volume 1

THOMAS MARTIN RICHENS

Doctor of Philosophy

ASTON UNIVERSITY

January 2011

This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with the author and that no quotation from the thesis and no information derived from it may be published without proper acknowledgement.

Summary
Aston University
Lexical Database Enrichment through Semi-Automated Morphological Analysis
Thomas Martin Richens
Doctor of Philosophy
2011

Derivational morphology proposes meaningful connections between words and is largely unrepresented in lexical databases. This thesis presents a project to enrich a lexical database with morphological links and to evaluate their contribution to disambiguation.

A lexical database with sense distinctions was required. WordNet was chosen because of its free availability and widespread use. Its suitability was assessed through critical evaluation with respect to specifications and criticisms, using a transparent, extensible model. The identification of serious shortcomings suggested a portable enrichment methodology, applicable to alternative resources. Although 40% of the most frequent words are prepositions, they have been largely ignored by computational linguists, so addition of prepositions was also required.

The preferred approach to morphological enrichment was to infer relations from phenomena discovered algorithmically. Both existing databases and existing algorithms can capture regular morphological relations, but cannot capture exceptions correctly; neither of them provide any semantic information. Some morphological analysis algorithms are subject to the fallacy that morphological analysis can be performed simply by segmentation.

Morphological rules, grounded in observation and etymology, govern associations between and attachment of suffixes and contribute to defining the meaning of morphological relationships. Specifying character substitutions circumvents the segmentation fallacy. Morphological rules are prone to undergeneration, minimised through a variable lexical validity requirement, and overgeneration, minimised by rule reformulation and restricting monosyllabic output. Rules take into account the morphology of ancestor languages through co-occurrences of morphological patterns. Multiple rules applicable to an input suffix need their precedence established.

The resistance of prefixations to segmentation has been addressed by identifying linking vowel exceptions and irregular prefixes.

The automatic affix discovery algorithm applies heuristics to identify meaningful affixes and is combined with morphological rules into a hybrid model, fed only with empirical data, collected without supervision. Further algorithms apply the rules optimally to automatically pre-identified suffixes and break words into their component morphemes. To handle exceptions, stoplists were created in response to initial errors and fed back into the model through iterative development, leading to 100% precision, contestable only on lexicographic criteria. Stoplist length is minimised by special treatment of monosyllables and reformulation of rules. 96% of words and phrases are analysed.

218,802 directed derivational links have been encoded in the lexicon rather than the wordnet component of the model because the lexicon provides the optimal clustering of word senses. Both links and analyser are portable to an alternative lexicon.

The evaluation uses the extended gloss overlaps disambiguation algorithm. The enriched model outperformed WordNet in terms of recall without loss of precision. Failure of all experiments to outperform disambiguation by frequency reflects on WordNet sense distinctions.

Keywords: morphological rules; automatic affix discovery; derivational morphology; segmentation fallacy; derivational tree.

Acknowledgments

The research presented here was conducted under a full time EPSRC-funded research studentship. The project began under the supervision of Sylvia Wong, Lecturer in Computing Science and was concluded under the joint supervision of Ian Nabney, Professor of Computing Science and Ramesh Krishnamurthy, Lecturer in English. The author would also like to thank the following, all of whom have played a role in facilitating this research:

- The late Sharen Lloyd
- Steve Dalton
- Chris Buckingham
- Ken Litkowski
- Christian Boitet
- Mathieu Mangeot
- Nazaire Mbame

Tom Richens
www.rockhouse.me.uk
tom.working@rockhouse.me.uk

Contents

VOLUME 1

Attached CD

Tables in Main Text

Volume 1

Volume 2

Text Figures

Volume 1

Volume 2

Glossary

This glossary provides some definitions. Some more extended definitions can be found in §1.1. Where no definition is provided, one or more section numbers are indicated, where the term is defined, introduced or discussed. Names of Java classes are not included in this glossary but are generally self-explanatory or correspond to other concepts defined. For further information regarding the classes used in morphological analysis, the reader is referred to the Class Diagrams and Appendix 1. The usage of other classes, not found in Appendix 1, will be discussed where they are referred to. A fixed width font has been used when referring to Java classes and methods. Uppercase has been used for *relation types*, with underscores for separators. These are listed in Appendix 22.

The personal pronoun "I" has, by convention, been avoided in this thesis. "We" has also been avoided because this research was undertaken by a single individual. Consequently, extensive use has been made of the passive voice. Where "we" has been used, it refers to the author and the reader collectively.

Term	Definition or where explained
abstract HYPERNYM	§4.2.4.1
active participle	§1.1.4
affix frequency	§3.4
affixation	a prefixation or suffixation
affix stripping precedence	§3.5.1
allowable path	§6.1.1.2
alternation	a syntactic variation in the behaviour of words, especially verbs, usually conceptualised as forming pairs
Anglo-Norman	the dialect of French used by the ruling class in England (1066-1485), also used by the merchant class in the fifteenth century
antonym	§§1.1.1, 2.2.2.6, 4.3.5
antonymous	having an opposite meaning
argument	§1.1.3
atomic dictionary	§5.3.3.1
atomic stem dictionary	§5.3.17
automatic affix discovery	§3.4

automatic prefix discovery	§3.4
automatic suffix discovery	§3.4
B&P	Banerjee and Pedersen
baseline disambiguation	§6.3.6.4
BNC	British National Corpus
candidate affix / prefix / suffix	§3.4
candidate back	§5.2.1.2
candidate front	§5.2.1.2
CLASS_MEMBER relation	§4.3.1
clusterhead	§4.3.2
complement properties	§4.2.1.5
compound expression	§3.5.2
concatenation	§1.1.2
converse morphological rule	§3.2.2.1
converse relation	§1.3.2.2
corpus	digital collection of texts
corpus frequency	the number of occurrences of a word in a corpus
counter-exception	an exception to an exception
default heuristic	§3.4.1.2
derivational morphology	§1.1.2
derivational pointer	§3.2.1
derivational tree	§3.1.4
derivative	a word or morpheme derived from another word or morpheme (its root)
disambiguation	the process of identifying which meaning of a word applies in a context
disambiguation by frequency	§6.3.6.4
duplication criterion	§3.4
empirical	by observation (of data) rather than with reference to theory or by introspection
etymology	§1.1.2
Extended Gloss Overlaps	§6.1.1.4
footprint	§3.2.2.3
formal quale	§1.1.5
frame inheritance	§2.3.2
frameset	a set of frames
generic disambiguation algorithm	§6.3.6.1
gerund	§1.1.4
gloss	a definition of a word or phrase, sometimes (in WordNet) considered to include any usage examples
gloss overlaps	§6.1
granularity	the relationship between words and meanings conceptualised as texture such that a fine grain means many meanings

	per word and a coarse grain means few meanings per word
heuristic	a formula used for finding objects within a set, typically morphemes with specified occurrence data
homonym	a word spelt in the same way as another word
hybrid model	§3.5.4
HYPERNYM	§1.1.1
hyphenation	a word formed by linking two other words with a hyphen
hyponym	§1.1.1
ILI	interlingual index
inflectional morphology	§1.1.2
irregular prefix	§5.3.11.1
iterative development	software development methodology whereby there is a feedback loop from initial outputs into software refinement
lemma	§1.3.2.5
lemmatiser	§1.3.2.5
lexical database	a database containing information about words and their meanings
lexical relation	a morphological relation between two word forms
lexical restoration	§5.3.17.4.4
lexical validity requirement	§5.1.4
lexically valid	existing as an entry in the lexicon
lexicographic	pertaining to lexicography, hence in alphabetical order
lexicon	an alphabetic list of words which may or may not map to further information, in particular the lexicon derived from WordNet within this research project (a.k.a. the main dictionary) or the software object which encapsulates it.
linguistic	pertaining to language
linking vowel	§3.2.2.3
linking vowel exception	§5.3.11.9
main dictionary	that component of the lexicon software object whose entries correspond to all the words and compound expressions in the WordNet model
manual	by the exercise of human intelligence and knowledge, especially linguistic knowledge, as opposed to a computational process or algorithm

monosemous	having a single meaning
morpheme	§1.1.2
morpheme exception / counter-exception	§5.3.5.2
morphodynamic wordnet	§3.1.4
morphological analysis	the analysis of the morphological relationships between words
morphological awareness	§6.3.6
morphological enrichment	the addition of morphological relations to a lexical database
morphological relation	relation holding between two morphemes (typically words), which manifests as lexical similarity, whose semantic significance may or may not be defined
morphological rule	a rule specifying a morphological transformation between two affixes (one of which may be a NULL affix) and defining the relation that holds between affixations bearing those affixes, specifying the POSes of the affixations
morphologically related	having common lexical features indicating a derivational relationship
morphology	§1.1.2
morphosemantic	pertaining to both morphology and semantics
morphosyntactic	pertaining to both morphology and syntax
multilingual	with reference to more than one language
multilingually formulated rules	§5.1.2
Nearest Neighbours Algorithm	§6.3.6.3
negative lexical validity requirement	§5.3.11.4.1
NLP	natural language processing
NODE	New Oxford Dictionary of English
non-lexical stem	§5.1.5
ODE	Oxford Dictionary of English
OED1	Oxford English Dictionary
OED2	Online Etymology Dictionary
One by One Algorithm	§6.3.6.1.1
One by One with Fast Alternatives	§6.4.3.4
ontology	§2
optimal heuristic	§3.4.5
overgeneration	the generation of invalid data whether because an object referred to, most typically a word, does not exist or because it does not stand in a specified relation to another object
part of speech	§1.1.4

participle	§1.1.4
passive participle	§1.1.4
pertainym	a WordNet relation from an adjective to a noun such that the adjective can be defined as "pertaining to" the noun and, by extension, a WordNet relation from an adverb to an adjective of the kind where the adverb is formed by appending "-ly" to the adjective
phoneme	a phonetic unit of speech which corresponds to a written character in a phonetic script
polysemy	§2.1
POS	part of speech (§1.1.4)
POSes	parts of speech (§1.1.4)
prefix footprint	§3.2.2.3
prefix tree	§3.4
prefixation	a word comprising a prefix followed by a stem or the process by which such a word is formed
pre-identified suffix	§5.2.2
preposition taxonomy	§§4.2.1.1, 4.2.1.6, 4.2.4
Princeton WordNet	§1.1.1
proper case	having its first character in uppercase
proper case variation	§5.3.6
quale	§1.1.5
quasi-gerund	§1.1.4
RDF	Resource Description Framework
regular prefix	§5.3.11.1
regularised prefix	§3.2.2.3
relatedness measure	§6.1
relation	a connection between words or meanings
relation type	the kind of relationship between two objects specified by a relation between them
rhyming dictionary	§§3.4.2.1, 5.3.3.2
root	§1.1.2
Root Identification Algorithm	§5.2
sandhi	§3.2.2.3
satellite	§4.3.2
secondary prefix set	§5.3.11.6
secondary suffixation analysis	§5.3.14
segmentation fallacy	§3.3.2
semantic category	§2.2.2.2.5
semantic criterion	§3.4
semantic distance	§6.1.1.3

semantic field	§2.2.2.2.5
semantic relatedness	§6.1
semantic relation	a relation between meanings or between synsets representing meanings
semantic role	the role of a word within a context in conveying meaning relative to the remainder of the context
semantically valid	satisfying the semantic criterion
sense combination	§6.3.6.2
sentence frame	§1.1.3
sister	§2.1.2.3
source	the related word or meaning from which a relation maps to a target
stem	§1.1.2
stem dictionary	§5.3.10
stem dictionary pruning	§5.3.17.2
stem interpretation	§5.3.17
stem validity quotient	§3.4.1.1
stoplist	a list of words or morphemes to which an algorithm is not to be applied
successor count	§3.3.1
successor variety	§3.3.2
suffixation	a word comprising a stem followed by a suffix or the process by which such a word is formed
superordinate taxonomic categorizer	§4.2.2
synset	§1.1.1
syntactic	pertaining to syntax
syntax	the process by which words are combined into sentences
target	the related word or meaning to which a relation maps from a source; a word being disambiguated
telic quale	§1.1.5
topology	the disposition of arcs and nodes in part of a graph
TPP	The Preposition Project
tree	a fully connected conceptual or data structure comprising nodes and directed arcs, with a single root node, such that each node can have multiple arcs connecting it to nodes further from the root and, except for the root node, a single arc connecting it to a node nearer to the root
troponym	§2.2.2.1

undergeneration	the failure by an algorithm to generate valid data of the kind the algorithm is intended to generate
unique beginner	§2.2.2
unregularised prefix	§3.2.2.3
valency	§2.3.2.1
verb frame	§1.1.3
verb taxonomy	§2.2.2
verbal phrase	§§2.3.1.2, 3.2.3, 3.5.2
whole word exception / counter-exception	§5.3.5.2
window occupant	§6.3
word	§1.1.2
Word Analysis Algorithm	§5.2
word form	the combination of characters which corresponds to a word or compound expression
word formation	the historical process by which words come into existence
word segmentation	§3.3.2
word sense	§§1.1.1, 2.1
word sense disambiguation	the process of identifying which meaning of a word applies in a context
wordnet	§1.1.1
WordNet	§1.1.1
WordNet model	§1.3.2
WordNet relation	a relation encoded in WordNet
WordNet relative	object (synset or word sense) related to another object by a WordNet relation
WSD	word sense disambiguation

Lexical Database Enrichment through Semi-Automated Morphological Analysis

1 Introduction

1.1 Definitions

As this thesis contains much discussion of *wordnets*, in particular *Princeton WordNet*, and *derivational morphology* and some discussion of *verb frames*, *participles* and *gerunds,* it is worthwhile to clarify, at the outset, what is meant by these terms.

1.1.1 Wordnets

Wordnets are lexical databases consisting of *word senses*. In theory each word sense represents a unique sense for a word form. As such it is the intersection between a word form and a meaning. Word senses are grouped into sets of synonyms called *synsets*, such that each synset theoretically represents a unique meaning. The same word form can occur in many synsets. The synsets are connected to each other by a number of different types of semantic *relation*. The best known of these relations is the relation of HYPERNYM to HYPONYM, where, in the case of nouns, the HYPONYM *is a kind of* the HYPERNYM, as for instance a "robin" is a kind of "bird" (Miller, 1998). As there are many other kinds of birds, the single HYPERNYM "bird" will have many HYPONYMS, forming a *taxonomic tree*. There are also relations which are defined between word senses rather than between synsets. Most of the relations are non-reciprocal, such as between HYPERNYM and HYPONYM, but a few are reciprocal, such as the relation ANTONYM which is defined between word senses, where one ANTONYM is the opposite of the other, as with "left" and "right". Another important relation is MERONYM / HOLONYM or a part / whole relation, as between "wheel" and "car".

The original wordnet was Princeton WordNet (http://wordnet.princeton.edu/; Fellbaum, 1998; Miller, 1998), which has been re-released in successive versions up to version 3.0. Unless otherwise stated, in this thesis, the term *WordNet* will be used to refer to Princeton WordNet 3.0 and the term *wordnet* will be used generically. WordNet 3.0 contains 82115 noun synsets, 13767 verb synsets, 18156 adjective synsets and 3621 adverb synsets. Applications of WordNet are numerous and varied and include malapropism detection (Hirst & St-Onge, 1998), analogy processing (Veale, 2006) and various approaches to word sense disambiguation (Stetina & Nagao, 1997; Leacock & Chodorow, 1998; Banerjee & Pedersen, 2002; 2003; Sinha et al., 2006). Other wordnets in many languages have been modelled on Princeton WordNet, which has also been used as an interlingual index (*ILL*) to link wordnets in several languages in EuroWordNet (Vossen, 2002).

1.1.2 Derivational Morphology

In his dictionary, Crystal (1980) defines *morphology* as "the branch of grammar which studies the structure or forms of words, primarily through the use of the *morpheme* construct". A morpheme is the "smallest functioning unit in the composition of words" (Crystal, 1980), where *word* is used in the sense of a series of alphabetic characters delimited by spaces and/or punctuation marks (Crystal, 1980) which has *meaning potential* (Hanks, 2004). The morphology of a word is determined by *inflection* and *derivation* (Crystal, 1980). This distinction is to some extent arbitrary, but can be defined on the basis that in the case of inflectional morphology, only irregular forms are traditionally listed in a dictionary whereas in the case of derivational morphology all forms are listed. A *morpheme* is also a series of alphabetic characters and also has meaning potential. All words are therefore morphemes though not all morphemes are words. Morphological analysis comprises the analysis of words into their constituent morphemes.

Derivation, according to Crystal (1980), has 3 meanings in linguistics, of which 2 are relevant here:

- "one of the two main categories or processes of word formation" (as opposed to inflection) and
- "the origins or historical development . . . of a linguistic form" (*etymology*).

This thesis will demonstrate the inseparability of these 2 concepts[1].

Taking the uninflected form of a word, its internal morphology is entirely *derivational*. While words related by inflectional morphology generally belong to the same part of speech, those related by derivational morphology most often do not (Bosch et al., 2008). The above definition of "word" excludes hyphenated forms, which leaves three phenomena determining the morphology of a word, namely *concatenation, abbreviation* and *affixation*. Concatenation is where a word can be divided into two or more other words which occur in the lexicon. Abbreviation is where a word cannot be broken down into its derivational components since it is composed of a subset of the characters which make up the word of which it is an abbreviation. Concatenations and affixations however lend themselves to morphological analysis. An *affix*, according to Crystal (1980) is "the type of *formative* that can be used only when added to another morpheme" where *formative* is "a formally identifiable, irreducible grammatical element which enters into the construction of larger linguistic units. . .". An affix is a *bound morpheme,* which cannot occur as a separate word (Crystal, 1980). An a*ffixation* is a word which can be divided into two morphemes, a *stem*, which is generally the longer part and may or may not be a word in its own right, and an *affix,* which is a morpheme which occurs in the same position in more than one word. There are two kinds of affix, a *prefix*, which occurs at the beginning of a word and a *suffix* which occurs at the end of a word. A word may include more than one prefix and/or more than one suffix. Since the term *stem* is being used for the residue after removing a single affix, the term *root* can be used to indicate the residual morpheme after the removal of all affixes, "which cannot be further analysed without total loss of identity" (Crystal, 1980). Affix removal from several words can lead to the same root, which can then be considered as the root of a *morphological tree*

[1] de Melo & Weikum (2010) get into difficulties when they try to treat the two separately.

(§3.1.4), not to be confused with the *taxonomic trees* formed by HYPERNYM / HYPONYM relations in WordNet (§1.1.1), and whose *roots* are also discussed in this thesis (§2.2.2.2). The term *root* is also used for the immediate *morphological* antecedent of a suffixation, which is not necessarily the same as the stem obtained by word segmentation (§§3.2.3, 3.3). The immediate *root* of a suffixation (its *derivative*) is in most cases its *historical* antecedent, though *back formations*[2] are exceptions to this rule[3]. This analysis denies the existence in standard English of a third kind of affix, in the middle of the word, called an *infix*, though a prefix or suffix may occur in the middle of a word formed by concatenation.

1.1.3 Verb Frames

The semantics of verbs depends on the set(s) of *arguments* (words or phrases which must be present in order for a sentence to make sense) with which they co-occur. These sets can be defined in terms of syntax (*syntactic frames*) or semantics (*semantic frames*). We also find the terms *case frames* (Fillmore, 1968), *valency frames* (Pala & Smrž, 2004), *subcategorisation frames*, *verb frames* or *sentence frames*. The terms *verb frames* and *sentence frames* will be used interchangeably in this thesis for syntactic frames, though the term *verb frame* will be preferred, or *sentence frame* when referring to WordNet. A verb frame defines a number of arguments which are required by a verb in a context. It must be understood that all verbs tolerate additional prepositional phrases as *adjuncts*, particularly phrases specifying time, place and manner (Verspoor, 1997; Kingsbury et al., 2002; Amaro, 2006). We are concerned in this thesis only with frame elements which are semantically required by a verb, in one or more of its syntactic *alternations* (syntactic variations in verb behaviour).

[2] e. g. "sleazy" existed before "sleaze". I am grateful to Ramesh Krishnamurthy for this example.
[3] Back formations do not get any special treatment in this research exercise. The relation types encoded for suffixation phenomena (Appendix 22) do not specify the rare cases where the stem is derived from the suffixation. `LexicalRelation.SuperType.ROOT` (§5.3.6) should not be taken as evidence of a historical sequence.

1.1.4 Parts of Speech, Participles and Gerunds

The main classification of words used in this thesis is that of traditional grammar, which recognises 8 *parts of speech* (Marsh & Goodman, 1925).[4] Because of the continuing popularity of terms such as "POS-tagging", and the adequacy of the traditional categories as supertypes of the categories used in the CLAWS tagging system for the British National Corpus (subsequently referred to as the *BNC*; Appendix 64), the term *part of speech* is preferred to the more modern term *word class*, but *part of speech* will generally be abbreviated to *POS* (plural *POSes*). The terms *active participle* ("-ing") and *passive participle* ("-ed", "-en" etc.) are preferred to the traditional grammatical terms *present participle* and *past participle*, as more accurately expressing the semantic distinction between the two. A *gerund* is a participle used as a noun, usually but not always active in meaning. It is generally true to say that, in English, all participles can be used as adjectives and that all active participles can serve as *gerunds*. Many passive participles can also be used as gerunds which tend to be implicitly plural as in "the damned". The term *quasi-gerund* will be used in this thesis for a word ending in "-ion" and having the same meaning as an active or passive gerund.

1.1.5 Qualia

Pustejovsky (1991) introduces the concept of *qualia* roles which are different simultaneous properties of concepts which can be inherited by a HYPONYM from a HYPERNYM as follows:

- *Constitutive quale :* internal composition
- *Formal quale :* external form
- *Telic quale :* purpose
- *Agentive quale :* causation

[4] NOUN, VERB, ADJECTIVE, ADVERB, PREPOSITION, PRONOUN, CONJUNCTION. INTERJECTION also implemented in the WordNet model (§1.3.2) as an enumeration of `Wordnet.PartOfSpeech` even though Princeton WordNet only has 4 of them.

A concept may inherit different qualia from different concepts. This justifies multiple inheritance in wordnets.

Amaro (2006) and Amaro et al. (2006) illustrate this idea as follows: "gun" and "sword" are both HYPONYMS of "artifact" through the formal quale, but HYPONYMS of "weapon" through the telic quale. They point out that HYPONYMS of the same HYPERNYM may or may not be compatible: e. g. feline and canine are incompatible HYPONYMS of mammal through the constitutive quale, because the information about morphology is inconsistent between them. HYPONYMS are compatible when they extend the properties of their HYPERNYM in different dimensions e. g. from the HYPERNYM "dog", "Alsatian" and "poodle" extend the constitutive quale while "lap-dog" and "police dog" extend the telic quale. Different simultaneous physical properties along the same dimension are incompatible, but orthogonal ones can be consistent, for instance the pairs "long" and "short" or "thick" and "thin" are incompatible but either "thick" or "thin" is compatible with both "long" and "short". These rules are suspended for hypothetical contexts and metaphors.

1.2 Motivation

1.2.1 Fighting Arbitrariness

This research was motivated by several challenges posed by Dr. Sylvia Wong's paper (Wong, 2004), which asserts that the nature of the information contained in lexical databases such as WordNet is often arbitrary due to inconsistent hand-crafting and subjective judgments. As an example of inconsistencies resulting from arbitrary encoding, Wong cites the HYPERNYM / HYPONYM tree rooted at the concept "dog" in WordNet 1.5, which defines a "toy poodle" as a HYPONYM of "poodle, a "toy spaniel" as a HYPONYM of "toy dog", and a "spaniel" as a HYPONYM of "sporting dog". In the absence of any encoded multiple inheritance in this taxonomy, a "toy poodle" is not a kind of "toy dog" and a "toy spaniel" is not a kind of "spaniel". Amaro et al. (2006;

§§1.1.5, 1.2.1) demonstrate that simple tree structures are insufficient to capture the inheritance relationships between concepts, because one concept may inherit orthogonal properties from more than one other concept. Although there is multiple inheritance in WordNet, in this case it has not been applied, and so the orthogonal properties of breed, size and occupation are inherited inconsistently. This kind of inheritance is investigated in §2.2.2.2.

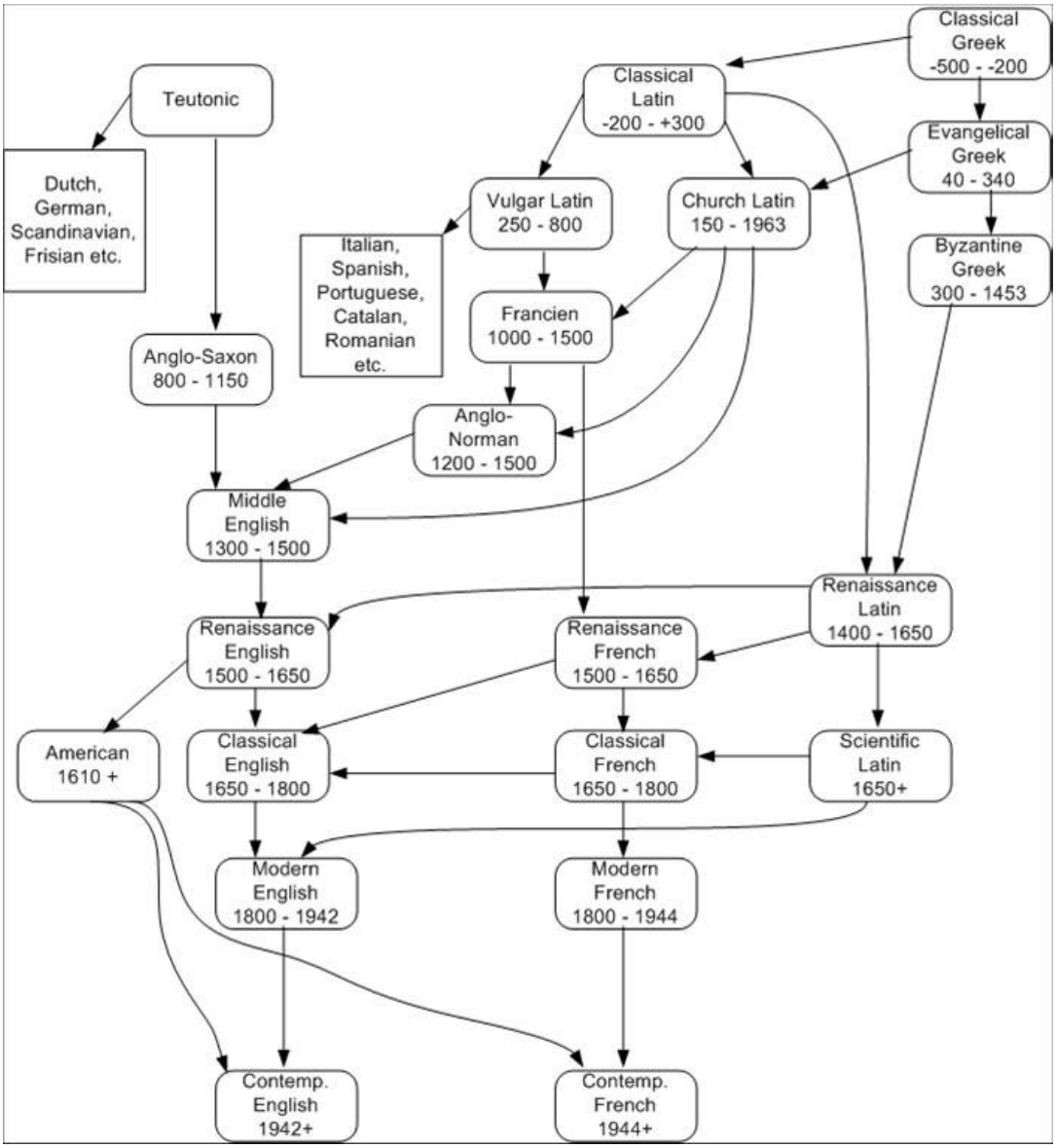
1.2.2 Derivational Morphology for Lexical Databases

Wong (2004) goes on to suggest (p. 236) that the system of "representation employed in a natural language . . . could aid the development of a lexical database", and observes that such a *system*, developed by the common consent of "millions of people over centuries . . . is *hidden* in most natural languages, especially those with phonetically driven orthography", but is explicit in Chinese, which is therefore more stable over time and facilitates the analysis of words into their component characters in a way which can be correlated easily with meaning. Wong also observes that the morphemic structure of words in one language might not be traceable without reference to other languages and concludes (p. 238) that "the set of relations observed in these languages is likely not to be sufficiently representative".

There was a time when Europe, like China, was politically and culturally united with a relatively static common language, Latin. While the use of Latin as the main written language outlived the political union of the Roman Empire by 1000 years, phonetic orthography did indeed mean that when written vernaculars emerged, they were not all mutually comprehensible. Within this dynamic context, the historical origins of the English language are extremely complex. To illustrate this complexity, a simplified diagram of its evolution is provided in Fig. 1[5]. The majority of words (as *tokens*) in any English corpus will be of Teutonic origin. However, the majority of words (as *types*) in the English lexicon are of Latin origin. Words (*types*) derived *directly* from Latin or

[5] The dates in the diagram represent dates between which there are written records and are mostly approximate.

Fig. 1: Evolution of English



derived from Latin *indirectly* through Anglo-Norman (Mediaeval French) display different spelling patterns. Because of these facts, knowledge of Latin and Anglo-Norman is advantageous for an understanding of English derivational morphology. The present author acquired an in-depth knowledge of the mechanics of *indirect* derivation from work on the corpus for the Anglo-Norman Dictionary[6] (http://www.anglo-norman.net), and of

[6] Prior to the commencement of this research project, the author's technical paper, *The Digital Representation of Contracted Script,* presented to the 8th. International Conference on Late and Vulgar

direct derivation through Classical Studies, and so was in an advantageous position from which to take up the challenge posed by Wong's remarks, of unveiling the *hidden system* which connects European languages across millennia from ancient Latin through to Modern English.

1.2.3 Project Aims

The main aims of this research project are, by largely automatic means,

- to discover relations between words based on derivational morphology,
- where possible to identify relation types corresponding to the semantic import of the morphological relations,
- to enrich a lexical database with these morphological or morphosemantic relations and
- to evaluate the contribution of the enrichment to word sense disambiguation (hereafter *WSD*).

Ample evidence will be presented (§3) that valid semantic relations can be discovered from derivational morphology and that these can be used to enrich a lexical database (§5), such that it performs demonstrably better at a task such as word sense disambiguation (§6), which is an essential task for many Natural language Processing (hereafter *NLP*) applications, including machine translation and information retrieval.

1.2.4 Fulfilment of Project Aims

In order to achieve the project aims, some kind of lexical database is required both as a starting point, an initial source of lexical data from which morphological relations can be inferred, and as a resource to be enriched with the relations discovered. The choice of WordNet was determined by its use in Wong's work, its free availability and its wide acceptance and widespread use in the NLP community. The ensuing investigation (§2)

Latin, St. Catherine's College, Oxford, September 2006 was not published in the proceedings but is available from http://www.rockhouse.me.uk/Anglo-Norman/index.html (referenced from the proceedings).

throws considerable doubt upon the wisdom of this choice. In retrospect, it might have been better to build a word list from an up to date corpus and use that as the primary data source. However, by the time the full extent of the faults and inconsistencies in WordNet had become apparent, it was too late to take this option within the project timetable, given that a lexical database, to be useful for applications involving WSD, needs to be more than simply a word list with morphological relations encoded between the words.

The two publicly available existing interfaces to WordNet are as a desktop application (available from http://wordnet.princeton.edu/wordnet/download/) and as a web resource (http://wordnetweb.princeton.edu/perl/webwn). Fulfilment of the project aim, and indeed even an assessment of the suitability of WordNet for the purpose, required a version of WordNet which could be interrogated in ways not possible with the existing interfaces, and which could be modified to incorporate the modifications from morphological enrichment. Thus the first requirement was to construct a model of WordNet which could be used as an experimental platform (§1.3.2). The next requirement was to critically evaluate the validity of the data contained (§2), with respect to specifications as to how wordnets should be structured (§§2.1.2.1, 2.2.2, 2.3.2.2) and criticisms directed at WordNet (§§1.2, 2.1, 2.2.2.2), to see to what extent it might be feasible to address its shortcomings, prior to attempting morphological enrichment.

Three possible approaches to the morphological enrichment of WordNet have been considered:

1. to identify morphosemantic relations from an existing database,
2. to infer morphosemantic relations from morphological rules derived from an existing database or
3. to infer morphosemantic relations from morphological phenomena empirically discovered from affix frequencies in the lexicon.

Of these approaches, the second two involve *morphological analysis*. Existing databases or algorithms may well capture regular morphological relations such as those between the following:

- compute
- computer: that which computes
- computation: computing
- computational: pertaining to computation
- computationally: by computation.

Simple morphological rules can easily be formulated to capture the syntax of such regular transformations, but no resources or algorithms (§3.3) have been found which capture exceptions to such relations and rules correctly, a shortcoming which this thesis sets out to rectify.

An investigation was conducted into the suitability of an existing data resource (CatVar: §3.1.2) as a basis for morphological enrichment. While this was found to be inadequate, it did serve as a basis for the identification of patterns of word formation which could be formulated as morphological rules (§3.2.2.1). However a systematic approach to morphological analysis (the identification of morphemes) requires the application of a morphological analysis algorithm or algorithms to empirical data. The primary algorithm developed and adopted in this thesis is the Automatic Affix Discovery Algorithm (§3.4), which identifies affixes to which morphological rules may be applicable or which may require translation from their languages of origin (§§3.2.3, 3.5.4, 5.3.11, 5.3.17). The Automatic Affix Discovery Algorithm was eventually combined with and a set of morphological rules, extended to accommodate the affixes discovered where applicable (§5.1), into a hybrid model which applies higher level algorithms to perform a complete morphological analysis of the words and compound expressions in the WordNet model and to enrich the model with morphosemantic relations. Finally the enriched lexical database or *morphosemantic wordnet* was evaluated by its performance at WSD using a known algorithm which employs the semantic relations already present in WordNet, adapted to employ the morphosemantic relations encoded (§6).

1.3 Experimental Platform

In order to investigate the soundness or otherwise of WordNet as a lexical database, and in order to enrich it with morphological data, a computational model was required, which could be interrogated in as many ways as possible and which could be modified (§1.2.4). Creating a model suggests an object-oriented approach because of the hierarchical nature of some of the concepts and the need for multiple interpretations or treatments of the data. The construction of an object-oriented model of WordNet allowed a large number of experiments to be conducted which involved interrogation (§§2.2-2.3), modification and enrichment (§§4-5) of the data. In this section, other object-oriented models will be reviewed, and the model adopted to achieve the project aims will be briefly described. As the model presented here has far more functionality than either WordNet or an online dictionary, and is extensible further, this approach to the analysis of language by computer can be considered to be an innovation.

1.3.1 Object-Oriented Approaches to Modelling Wordnet Data

1.3.1.1 RDF

Graves & Gutierrez (2006), in extolling the virtues of RDF (*Resource Description Framework*), cite very basic concepts such as data types and object-oriented features such as class inheritance and software extensibility. All these virtues are possessed, in at least equal measure by C++ and Java. The only relevant, specific characteristic of RDF is its suitability for use with directed graphs. However, a directed graph can be represented as a set of interlocking trees and a tree can be viewed as a set of interlocking linked lists. Therefore any language which has the explicit or implicit concept of a pointer (in the C++ sense), allows the modelling of any complex linked data structure, including a directed graph, as in the model used in this research project, though in the end it was implemented slightly differently (§1.3.2.2; Appendix 65).

Graves & Gutierrez reject the OWL Web Ontology Language on the grounds that it would introduce unnecessary complexity. The same could perhaps be said of RDF. The higher level the technology deployed, the more one becomes the prisoner of its formalisms. An object-oriented language gives the right level of abstraction for the rapid development of complex data structures and interrogation routines, without introducing formalisms which may not be suited to the data or applications.

Graves & Gutierrez describe some previous attempts to model WordNet using RDF. What is most striking is the length of time taken to achieve an inadequate model. It took 4 years for RDF developers to arrive at the notion of a word sense, which is the WordNet equivalent of an atom, and the very first class of object specified in the model used here, which was developed in a fraction of the time, without the need for the enormous amounts of double checking Graves & Gutierrez describe.

1.3.1.2 Python

Kahusk (2010) presents Python as a language of choice for modelling EuroWordNet data, because of its object-oriented features, but gives no reasons for the choice over better known object-oriented languages. The model presented has few classes and very few methods (all of which have equivalents in the model presented in §1.3.2), supporting only the limited functionality required for editing and managing EuroWordNet files, though it has been extended for other applications.

The conclusion here is that an object-oriented approach is desirable for modelling wordnet data, but specialised languages and technologies do not facilitate, but rather complicate, the development of such a model. For this thesis, the development of an object-oriented model of WordNet was only the first step. It needed to be done quickly and in a way that would allow complex queries and modifications. The difficulties reported by others using sophisticated but poorly adapted technologies confirm that a simple, extensible, portable and widely used language such as Java was the right choice.

1.3.2 The WordNet Model

1.3.2.1 Choice of Java

Some reasons for using Java have been given in §1.3.1. Portability between hardware platforms is another advantage. Another important consideration is the existence of suitable exception handling capabilities. Software development within the context of this project is very largely data-driven. For a project where one does not know, at the outset, what the data contains, while one may have an initial design idea, one must always expect that the data used will throw up unforeseen complications and one cannot assume that it will fit the design model. A number of `Exception` classes have been defined and exceptions are thrown in every conceivable circumstance where the data might not fit the design assumptions (Appendix 29). Much of the development time was taken up with adapting the model to fit unexpected data which provoked exceptions. The original design and subsequent modifications are shown in Class Diagrams 1-7. A detailed description of the model is available in Appendix 65. To facilitate cross-referencing to the code and documentation on the attached CD, names of methods implementing algorithms discussed in the following chapters have been provided in the footnotes. Names of input and output files have also been provided for anyone who wishes to examine them. The files referred to are also on the CD.

1.3.2.2 WordNet Relations (*Class Diagrams 4 & 5*)

The relations are encoded between the source and target objects, exactly as specified except that a converse relation is always encoded, so that all relations are navigable in both directions[7], whereas the WordNet documentation specifies only some relations as bidirectional. Converses of relations of types ANTONYM, VERB_GROUP_POINTER and DERIV are of the same type as the relation type of which they are converse. All other converses are of a different type, as specified in the documentation

[7] a decision without which some investigations would not have been possible.

(http://wordnet.princeton.edu/man/), or of a newly invented type, where no converse is recognised by the documentation (Appendix 22). The target of every `WordnetRelation` is represented as the corresponding Synset ID, and the target word of every `WordSenseRelation` (`WordnetRelation` holding between word senses) is held as the corresponding word number.[8]

1.3.2.3 Sentence Frames

Optionally, the 35 WordNet sentence frames (§1.1.3) are included, specifying their *valency* (§2.3.2.1) inferred from the description in the WordNet documentation (Kohl et al., 1998; §2.3; Appendix 2) and the assignations of sentence frames to verbs are read from file. For consistency, and to facilitate the interrogation of the frame information (§2.3), they are all assigned to an individual `Verb`. Where a `VerbSynset` is specified by the source data, the frame is assigned to every `Verb` within that `VerbSynset`.

1.3.2.4 The Lexicon (*Class Diagrams 2 & 7*)

A word sense represents the intersection of a word form with a meaning (§1.1.1). A wordnet is a way of organising word senses by meaning. A lexicon is a way of organising word senses by word form. Retrieval of a `Synset` from the `Wordnet` requires its synset ID to be known. Clearly it is desirable, and essential for most applications, to be able to retrieve all the word senses for a given word form, or all the synsets containing a `WordSense` with a specified word form. This functionality is provided by the Lexicon, at whose core is the main dictionary which provides mappings from every word or compound expression found in WordNet to a lexical record, corresponding to a single word form. In the original design, every lexical record held mappings from the identifiers

[8] In the original design, the target of every `Relation` was held as a reference to the target object. However, it proved impossible to de-serialise the serialised representation of the WordNet model from a serialised object file without a stack error, because of the bidirectional encoding of the relations. This was addressed by storing the targets as described. This slows down navigation of the relations, which became apparent during WSD tests (§6.4). In retrospect it would have been better to retain the storage of each target as a reference, to specify the corresponding identifiers during serialisation and then to retrieve the required references during de-serialisation. This will be corrected in future versions.

of every Synset containing the corresponding word form to the relevant sense number (for the specified word form), the word number (within the specified Synset) and the tag count (Brown Corpus frequency) for a single word sense. This design was subsequently modified to accommodate POS-specific queries (§3.5.3).

1.3.2.5 The Lemmatiser (*Class Diagram 6*)

The Lexicon contains entries of words and compound expressions found in WordNet. This does not include the lemmas (base forms) of inflected word forms. A Lemmatiser was needed to enable inflected words to be looked up in the Lexicon, so that the synsets or word senses corresponding to inflected words could be retrieved. This is essential for many applications including WSD. The lemmatiser requires two maps, one for regular inflections and one for exceptions (Class Diagram 6). The Lemmatiser also holds the constant array of inflectional suffixes which occur preceded by an apostrophe, namely {"d", "ll", "m", "re", "s", "ve"}. The Lemmatiser services lemmatisation queries which can be specified in a number of ways. The array of inflectional suffixes may also be consulted,[9] depending on how the query is specified, but if a modal verb is returned, it will not be found in the lexicon, as modal verbs are not represented in WordNet.

1.3.2.6 Applications of the Model and Related Publications

The experimental work discussed in §2 has been carried out by developing methods for interrogating the model, so as to derive embedded information which is not retrievable using standard WordNet interfaces, in order to expose the strengths and weaknesses of the database. Serial data has been output as text files and tabular data as *.csv* (*comma-separated values*) files which facilitate further analysis using a spreadsheet. Experimental work included an in-depth study of the relations between verbs (§2.2) culminating in a paper presented to the 22nd. International Conference on Computational Linguistics (Richens, 2008) which highlights ontology faults and the arbitrariness of the encoding, suggesting possible solutions.

[9] One or more hard-coded verbs will be returned.

Subsequent interrogatory experiments initially focussed on the representation of verb syntax (§2.3) and included a pilot study to assess the feasibility of enriching WordNet with data from derivational morphology (§3.2.2), leading to a paper presented to the 6th. International Workshop on Natural Language Processing and Cognitive Science (Richens, 2009a). This work prompted, and was facilitated by, the inclusion of the lexicon and lemmatiser. Additional functionality was added to the model to support experiments on Automatic Affix discovery (§3.4) presented to the 4th. Language & Technology Conference (Richens, 2009b).[10]

1.3.2.7 Subsequent Modifications

The model described here[11] is faithful to Princeton WordNet. The model has been subsequently modified by the addition of prepositions (§4.2) and *pruned* (§4.3) to remove superfluous synsets, word senses and relation types and to improve consistency in the encoding of the remaining relations[12]. Experiments in correcting the sentence frames by parsing the usage examples are briefly referred to in §2.4, but have not contributed to this thesis. The major modification to the model which is morphological enrichment is discussed in detail in §5.3.

[10] In addition to the author's papers cited above and presented at the respective conferences, two further papers *Automatic Affix Discovery for Wordnet Morphological Enrichment* and *Revising WordNet Sentence Frames to match Usage Examples* were accepted by the Global Wordnet Association for its 5th. conference in Mumbai, India, Jan.-Feb. 2010, but were subsequently withdrawn. The author also presented a seminar *La base WordNet, ses problemes et leur traitement éventuel* under the auspices of the Groupe d'Etude pour la Traduction Automatique et le Traitement Automatisé des Langues et de la Parole (GETALP), at the Laboratoire d'Informatique de Grenoble, Joseph Fourier University, Grenoble, 14th. May 2009.

[11] serialised to file *princeton.wnt*

[12] The preposition-enriched and pruned version is serialised as file *bearnet.wnt*. As far as the author is aware, there is no standardised file format for the representation of wordnets, unless the *Prolog* format (Appendix 65) be considered as such.

2 Investigation into WordNet

The first application of the WordNet model was a limited but rigorous investigation into certain properties of WordNet, which are hidden from the user of standard interfaces (§1.2.4), to see how far the criticisms (§§1.2, 2.1, 2.2) of it are justified. The WordNet documentation (Miller, 1998; Fellbaum, 1998; Kohl et al., 1998; http://wordnet.princeton.edu/) fails to mention or explain many of these properties or the inconsistencies discovered and discussed in this section. The discovery of inconsistencies was only possible through the exposure of hidden properties by the object-oriented model.

This chapter reviews criticisms, made or implied, of WordNet, additional to those of Wong (2004; §1.2.1, 1.2.2), The investigation into some of these criticisms through interrogation of the Java model is then described, along with the algorithms used for the interrogation. The purpose of this investigation was to assess the suitability of WordNet as a foundation for developing a morphologically enriched lexical database. Because most other WordNet-based research has concentrated on nouns, and because of the issues raised by Amaro and others (§§2.2.2.2, 2.3.2.2), this investigation has focussed mainly on verbs.

The review starts from a consideration of the validity of the atomic concept of a word sense, which is the fundamental building block of WordNet. The pitfalls of making sense distinctions are discussed (§2.1.1) along with their implications for granularity (§2.1.2.1). A brief investigation into the granularity of verb meanings is described (§2.1.2.2). This leads on to a consideration of the advantages and disadvantages of proposals for reducing the granularity by clustering word senses or synsets (§2.1.2.3).

Relations between word meanings are then considered, with particular reference to the organisation of concepts through hierarchical relations as an ontology (§2.2.1). Taking as a starting point Fellbaum's (1998) specification, a detailed investigation is described into

the verb taxonomy (§2.2.2), with reference to WordNet's *semantic categories*. This is cross referenced to other recent research in this area. This leads towards a consideration of ways in which the verb taxonomy could be improved and a review of the representation of verb syntax by the WordNet sentence frames (§2.3), to assess the possibility of using syntax as a guide to revising the taxonomy. The theoretical expectations of inheritance of verb properties are reviewed (§2.3.2.2) and the actual data is compared to those expectations (§2.3.2.3). These investigations will allow us to reach some conclusion as to the validity and consistency of WordNet (§2.4) and consider possibilities for addressing its deficiencies, prior to reaching any conclusion as to its suitability as a lexical database for morphological enrichment.

2.1 Word Senses

A *word sense* can be defined as the intersection between a word (or compound expression) and a meaning. The obvious implication is that a word can be *ambiguous*.

Pustejovsky (1991), following Apresjan (1973), distinguishes between two kinds of *ambiguity*: *homonymy* and *polysemy*: The two senses of bank as in "river bank" and "investment bank" are semantically unrelated: this is *homonymy;* on the other hand, within the second sense one can further distinguish between "bank" as a building and "bank" as an institution: this is *polysemy*. No such distinction is made in WordNet. The question remains open as to how many senses the word "bank", as a noun, has.

2.1.1 "I don't believe in word senses"[13]

Kilgarriff (1997) calls into question the very notion of a word sense. The historical perspective he presents is that the meanings of words have long been debated and that the

[13] attributed by Kilgarriff (1997) to Sue Atkins, former President of the European Association for Lexicography, Lexicographical Adviser to Oxford University Press and Editor of Collins-Robert English-French Dictionary, in a discussion at *The Future of the Dictionary* workshop, Uriage-les-Bains, France, October 1994.

advent of dictionaries was a response to that debate, subsequent to which dictionary definitions have come to be treated as facts, rather than as the opinions of lexicographers, despite the plethora of conflicting definitions and categorisations between different dictionaries.

The problem has been thrown into sharp relief with the advent of computer-based NLP, where most practitioners have simply accepted some or other supplied listing of senses for each word and attempted to disambiguate words in context into the supplied senses of which few have called into question the empirical validity.

Kilgarriff counters this naive acceptance by pointing out that there are different kinds and levels of sense distinctions: metaphor has been made prominent by Lakoff (1987) and regular polysemy by Apresjan (1973) and Pustejowsky (1991). Pustejowsky (1995) warns against the idea that a lexicon can enumerate the senses of a word. Along with Lakoff (1987), Pustejowsky rejects the idea of necessary and sufficient conditions completely, while developing the notion of preference rules (Jackendoff, 1983). At the same time there has been a growing interest in WSD and ways of evaluating it (§6.1). The lack of consensus on the boundaries between senses is a major inconvenience for computational linguistics.

2.1.1.1 Metaphor

Hanks (1997; 2004; 2006) distinguishes between *norms* and *exploitations*. Exploitations, or meaning extensions as Kilgarriff (1997) calls them, typically are metaphors[14]. Whether metaphorical or not, they employ *semantic coercion* (Pustejovsky, 1995), meaning that they force their syntactic dependents to take on exceptional *qualia* roles (§1.1.5). Hanks uses corpus pattern analysis to identify usages which do not conform to norms. In the case of the word "storm", he finds that metaphorical uses are more frequent than literal uses in a corpus. He identifies a *gradient of metaphoricity* for "storm", starting from its

[14] Kilgarriff's (1997) example of the use of "handbag" as a weapon is not metaphorical, because the basic definition of "handbag" still holds, but his further example "handbags at ten paces" clearly is metaphorical.

literal usages, associated with verbs such as "blow" and "abate", through expressions such as "get caught in a storm", where a verb is used metaphorically in relation to a literal storm, through usages where the word "storm" is itself metaphorical ("a storm of protest") to "a storm in a teacup", where neither "storm" not "teacup" are literal. Clues to metaphorical exploitations include abnormal governing verbs ("cause / spark a storm") and abnormal partitives ("storm of protest/controversy").

To complicate matters, metaphors, through time, become norms, as is the case with "to take by storm", which has been in use since the seventeenth century, and has been subject to further metaphorical exploitations in domains such as sport and fashion ("Diana took France by storm."). Again clues can be identified: "take the *world* by storm" will not be taken in a military sense, nor will "political storm".

Hanks (2006) cites corpus evidence to show that typical subjects of the verb "backfire" are "gamble", "plan", "car" or "truck", but not "rage" or "train ". He argues that "rage" cannot be a possible subject because, unlike a "plan", it is not intentional, but he provides no reason why a train should not backfire (assuming it is powered by an internal combustion engine). He goes on to state that we are dealing here with two meanings and then to present the hypothesis that when a child acquires the word "backfire", it is more likely to be in the "plan" sense, purely on the grounds of BNC evidence, which shows more instances of the "plan" meaning than of the "car" meaning.

This hypothesis is unconvincing for two reasons:

1. The BNC is not representative of contexts where children first acquire words.
2. The word "backfire" is a concatenation of "back" and "fire", which makes sense in the context of an internal combustion engine but not in the context of a plan.

Hanks himself questions the hypothesis, not on either of these grounds but from recollection of how he himself acquired the word as a child. A "plan backfiring" is then a metaphor, albeit an established one, derived from analogy probably to a firearm[15] rather

[15] Is this a third sense or the same sense as when the subject is an internal combustion engine?

than an internal combustion engine[16], but this example illustrates well why Hanks prefers to talk about norms and exploitations rather than literal and metaphorical meanings. An exploitation does in fact, over time, become a norm[17]. To say "the lunch backfired" would, Hanks suggests (p. 11) , be a further exploitation of the "plan" sense.

This brief excursion into the realm of metaphor confirms the difficulty of defining where one sense ends and another begins.

2.1.1.2 Translation Equivalents

Kilgarriff (1997) concludes that word senses are, at best, abstractions from clusters of usages (and that only in a specialised domain) and, at worst, the consequences of vested interests in dictionary publication. However he barely mentions the whole question of translation equivalents. Contexts which require two different words in language A imply two different senses of a word in language B. This suggests a possibly more objective way of distinguishing word senses. The issues involved have been explored in the development of EuroWordNet and BalkaNet and discussed in Vossen (2002; 2004) and EU (2004).

Sagot & Fišer (2008) use a subset of JRC-Acquis (http://langtech.jrc.it/JRC-Acquis.html), an untagged 8-language aligned corpus, to find translation equivalents, in order to derive a French wordnet automatically from Princeton WordNet plus other sources. Clearly translation equivalents could be found from an aligned bilingual corpus, but Sagot & Fišer use some of the other languages as a control to help maintain compatibility with EuroWordNet and BalkaNet.

They provide the example of the English word "law" and find 3 non-synonymous French translation equivalents: "droit", "loi" and "législation". We could say then that the English "law" has 3 word senses relative to French. They also find 3 Czech translation

[16] The meaning "premature ignition in an internal-combustion engine" is first recorded 1897; "affect the initiator rather than the intended object" (of schemes, plans, etc.) is attested from 1912 (OED2).
[17] Establishing norms is one of the great strengths of corpus linguistics.

equivalents: "právo", "zákon" and "předpis"; so we could also say that English "law" has 3 word senses relative to Czech, assuming that none of these are synonymous. However there is no one-to-one mapping between the French and Czech translation equivalents. In fact, looking at French and Czech together, there are 5 translation equivalent pairs: {"droit"; "právo"}, {"loi"; "právo"}, {"loi"; "zákon"}, {"législation"; "právo"} and {"législation"; "předpis"}, so we could say that relative to French and Czech, English "law" has 5 word senses, or fewer if any of the Czech words are synonymous. This is rather less than the 9 there could be in the worst case scenario. When we look at Bulgarian, we again find 3 translation equivalents: "законодателство", "право" and "закон" (and one lemmatisation error), but there is no one-to-one mapping between the Bulgarian and French or Czech translation equivalents except for Czech "zákon" to Bulgarian "закон" (if we ignore the lemmatisation error). English "law" has 9 or fewer word senses with respect to these 3 languages, considerably less than the 27 theoretically in the worst case scenario.

This approach tells us nothing about the relations between the senses identified except that they are not generally synonymous; the translation equivalence relations can only be synonymous where there is a one-to-one mapping. Huang et al. (2002) analyse the relations involved when there are two related pairs of translation equivalents, as part of the process of developing a Chinese wordnet from Princeton WordNet. Given two pairs of English-Chinese translation equivalents {*EW1*; *CW1*} and {*EW2*; *CW2*}, where there is a WordNet relation between *EW1* and *EW2*, if the semantic relations between the members of the two pairs of translation equivalents can be defined as some kind of wordnet relation then the relation between *CW1* and *CW2* can be defined in terms of the other relations, in particular the relation *CW1->CW2* can be defined as the combination of the relations *CW1->EW1*, *EW1->EW2* and *EW2->CW2*. Synonymies can be assigned a value of 0, so that if *EW2* and *CW2* are synonyms, then the relation *CW1->CW2* can be defined as the combination of the relations *CW1->EW1* and *EW1->EW2*, while if both translation equivalence relations are synonymous, the relation *CW1->CW2* can be defined as identical to the relation *EW1->EW2*. This gives satisfactory results, based on manual evaluation, in 88.5% of cases where both pairs of equivalents are synonymous nouns, but

in the non-synonymous cases it is not always clear what it means to combine two relations. In some cases this is relatively straightforward:

- ANTONYM + ANTONYM = SYNONYM ("little" -> "big" -> "small")
- HYPERNYM + HYPERNYM = HYPERNYM of HYPERNYM ("piston" -> "engine" -> "car")
- HYPONYM + HYPONYM = HYPONYM of HYPONYM ("car" -> "engine" ->"piston")

In the latter 2 cases, if no synonymous translation equivalent can be found, an abstract synset should be posited in wordnet construction. However where the two relations are not of the same type, relation *a* + relation *b* is not equivalent to relation *b* + relation *a*, as in the following cases:

- HYPONYM + ANTONYM = (another) HYPONYM ("move" -> "go" -> "come")
- ANTONYM + HYPONYM = HYPONYM of ANTONYM ("go" -> "come" -> "arrive")
- HYPERNYM + ANTONYM = ANTONYM of HYPERNYM ("arrive" -> "come" -> "go")

but in the following cases, if they occur, the result is indeterminate:

- ANTONYM + HYPERNYM = HYPERNYM OR another HYPERNYM of the ANTONYM (where there is multiple inheritance)
- HYPERNYM + HYPONYM = SYNONYM OR ANTONYM OR *sister term* (cf. Amaro et al., 2006; §2.2.2.3)
- HYPONYM + HYPERNYM = SYNONYM OR another HYPERNYM (where there is multiple inheritance)

HOLONYM and MERONYM relations behave in the same way as HYPERNYM and HYPONYM relations except that where an ANTONYM is involved the resultant relation is not reducible. These equations apply where one out of two pairs of translation equivalents is synonymous. Where neither pair is synonymous, the likelihood of an indeterminate outcome increases as three relations must be combined and Huang et al. do not attempt to infer the consequent relations.

The apparent paradoxes here arise from the phenomenon of dual inheritance which may be justified in that a word may have more than one HYPERNYM or ANTONYM with respect to different semantic dimensions such as qualia (§1.1.5; Amaro et al., 2006) or breed, size and occupation of dogs (Wong, 2004; §1.2.1), but in practice, in WordNet, multiple inheritance does not necessarily have any such justification (§2.2.2.2).

Huang et al. conclude that databases of translation equivalents should specify the semantic relation type (SYNONYM, HYPERNYM etc.) involved in the equivalence, which would be a major aid not only to wordnet construction but also to automatic translation. It would also be better if HYPERNYM/HYPONYM and ANTONYM relations in wordnets were labelled with respect to the semantic dimension to which they apply.

2.1.1.3 Conclusions on Word Senses

The translation equivalence approach to word sense identification no doubt has its problems (multiword expressions being the most obvious), but aligned parallel corpora do provide an empirical method of enumerating word senses to satisfy the requirements of automatic translation; indeed this approach (extended to multiword expressions) lies at the heart of statistical machine translation. If it were possible to extend this procedure to every language, then it would theoretically be possible to compute a finite maximal[18] number of word senses required for every English word. On these grounds, and these grounds alone, the theoretical position that there is no such thing as a word sense, or that it can, at best, only be a lexicographer's abstraction from a cluster of usages, is to be rejected. We are left with an enormous variety of dictionaries and wordnets which have non-empirical sense distinctions, among which at one extreme we have corpus-based dictionaries, which at least use empirical corpus data as a starting point to WordNet at the other, where the sense distinctions appear to arise from undocumented and apparently arbitrary decisions arising from conflicting theoretical models ranging from

[18] because some may be synonyms.

psycholinguistics to frame semantics[19]. Some further discussion on the relative merits of WordNet and other sense distinctions will be found in §6.2, but we will now look at the specific issue of whether WordNet sense distinctions are too fine.

2.1.2 Granularity

In the absence of any consensus as to how many senses any word has, in encoding lexical databases, the number of senses of any word should perhaps be decided on pragmatic rather than theoretical grounds. It is not always possible to tell the difference between closely related WordNet senses, nor is there any evidence that they are based on usage patterns or collocations, let alone translation equivalents. In the absence of any distinction in WordNet between homonymy and polysemy (Apresjan, 1973; Pustejovsky, 1991), the multiplicity of senses poses a problem for the encoding of relations based on morphology (§§3.2.1, 3.5.3). This section will review some other problems which arise from this fine granularity and consider some proposed solutions.

2.1.2.1 Implications of WordNet Granularity for Multilingual Wordnet Development

EuroWordNet (Vossen, 2002) comprises wordnets in several European languages, linked by an interlingual index (*ILI*) modelled on WordNet 1.5, to which composite records have been added by clustering word senses, to provide better translation equivalents. It is preferable, for this application of WordNet, if sense distinctions are not too fine-grained, as this makes it more difficult to establish equivalences across languages. Senses need to be grouped according to regular polysemy into composite ILI records comparable to Pustejovsky's (1991) complex types. Polysemy is not simply a characteristic of a particular language, since a subset of polysemous meanings of a word can map to a subset of polysemous meanings of another word in another language. For instance, in many European languages, words such as "embassy" and "university", or their

[19] There is a lack of documentation concerning these decisions either in the book (Miller, 1998; Fellbaum, 1998; Kohl et al., 1998) or on the website (http://wordnet.princeton.edu/).

equivalents, can mean either institution or building (Vossen, 2004). These meanings, though distinguishable, are clearly related by a common underlying concept, which can define members of a composite ILI record in EuroWordNet, which is, in fact, a *cluster* of synsets.

Attempts to convert the WordNet-based ILI into a "universal index of meaning" require either maximisation of the number of concepts, so that the ILI is always either the superset of concepts in the other wordnets, or minimisation to a set of essential concepts (Vossen, 2002). The overhead of the former approach is prohibitive; the latter is equivalent to clustering.

The BalkaNet project (EU, 2004) uses the same ILI as EuroWordNet. Within this project, the developers of the Serbian wordnet complained that it was difficult to grasp the differences between similar synsets, especially with misleading examples. They cite the following sets of words with WordNet sense numbers, which they would consider to be synonyms, but which are not synonyms in WordNet:

> *{fluid 1; fluid 2}, {depart 1; go 15; go away 2; travel away; go away 3; go forth 1; leave 10}, {conveyance 3; vehicle 1}*

2.1.2.2 Investigation into WordNet Granularity

In order to assess the granularity of verbs in WordNet, the number of senses for each verb was counted, along with the proportion of the synsets involved which contain no other words or compound expressions. Table 1 shows the 20 verbs with most senses encoded. The encoded polysemy seems excessive; no human subject not trained in lexicography is likely to identify so many senses.

At the start of the research project, a subjective evaluation was conducted of the sense distinctions among some polysemous verbs. This evaluation was done using WordNet 2.1, unlike the subsequent experiments which used WordNet 3.0. One problem found was an inconsistent approach to the composition of glosses, which frequently fail clearly to

Table 1: 20 most polysemous verbs

Verb	No. of senses	% where this word is the only member of the synset
break	59	52.54%
make	49	46.94%
give	44	50.00%
take	42	26.19%
cut	41	63.41%
run	41	36.59%
carry	40	62.50%
get	36	19.44%
draw	36	44.44%
hold	36	30.56%
play	35	62.86%
fall	32	65.63%
go	30	26.67%
catch	29	44.83%
call	28	64.29%
work	27	40.74%
raise	27	40.74%
turn	26	53.85%
cover	26	46.15%
set	25	24.00%

define the verb sense in such a way that it can be distinguished from others. It is striking that within this proliferation of poorly distinguishable verb senses, some basic meanings are still not represented, such as "bear" in the sense of "support weight", "get" in the sense of "go" and "find" as "take without being given or stealing". The most usual usage of "do", as an auxiliary verb followed by an infinitive without "to", is not mentioned. Many different verb "senses" in WordNet represent slightly different usages. The differences are between the verb frames rather than the verbs themselves. If a common gloss can be applied to several "senses", then this suggests that the senses could be merged as long as a correct and complete list of frames is supplied.

2.1.2.3 Clustering of Word Senses and Synsets

Peters et al. (1998) note that the high level of ambiguity in WordNet results in poor performance for WSD (cf. §§6.4.4, 7.3). For EuroWordNet, word senses have been clustered into coarser-grained groups, appropriate for representing translation equivalents (Vossen, 2002; 2004; §2.1.2.1). The clustering is based on the principles of generalisation, regular polysemy (Apresjan, 1973; Pustejovsky, 1995) and sense extension based on *denotational* alternations such as between "lamb" as an animal and "lamb" as a food and *diathesis* alternations as between transitive and intransitive usages of the same verb ("I broke the window"; "The window broke").

Peters et al. (1998) advocate the deployment of the following similarity rules to identify candidates for clustering:

1. *Sisters* defined as senses of the same word having a common HYPERNYM.
2. *Autohyponymy*, where 2 senses of the same word stand in a HYPERNYM-HYPONYM relation to each other.
3. *Twins* defined as synsets with at least 3 words in common.
4. *Cousins*, defined as patterns of regular polysemy manifested where 2 synsets with related meanings have common sets of words as HYPONYMS.

Mihalcea & Moldovan (2001) propose the following conditions for pairs of synsets to be merged:

1. if the synsets are verbs linked by a VERB_GROUP_POINTER.
2. if the set of words in each synset is identical and the number of words in each is greater than 1.
3. if each synset contains at least 1 common word and they have a common HYPERNYM.
4. if the number of common words between the synsets >= a threshold value *K*.
5. if the 2 synsets have at least 1 word in common, and share an ANTONYM.
6. if they have at least 1 word in common and share a PERTAINYM.

This approach effectively addresses the issue of granularity through a clearly defined set of rules. However, all these rules are likely to have the effect of merging verbal synsets, the difference between which represents a verb alternation (Levin, 1993). While there are examples (Lee et al., 2006) of verb alternations already occupying the same synset, this obscures verb syntax and should be avoided. An alternative solution is proposed in §3.5.3 (see also §2.4).

2.2 Taxonomy

2.2.1 Ontology

2.2.1.1 Shortcomings of WordNet-like Ontologies

Poesio et al. (2003) find three main problems with using WordNet as an information source for semantic relations:

1. Some words are not in WordNet.
2. Some sets of words used as synonyms, e. g. {"slump"; "crash"; "bust"} are not encoded as synonyms in WordNet.
3. The HOLONYM/MERONYM hierarchy is incomplete: thus "room", in WordNet is a MERONYM of "building" but not of "house".

Guarino (1998) finds serious problems with various ontologies, with particular reference to the way they handle instances of regular polysemy (Apresjan, 1973; Pustejovsky, 1991; 1995). His critique includes the WordNet ontology where it should be true to say that the relation between a HYPONYM *A* and its HYPERNYM *B* corresponds to saying that *A* "is a" *B*. The problem here is that a relation between words does not necessarily correspond to a logical relation between classes of real-world entities. Guarino considers that the "is a" relation is poorly understood so as to be frequently "overloaded" in various ways in WordNet, as follows:

- *Confusion of senses:*

  *A window **is a**n opening.*

  *A window **is a** panel.*
- *Sense reduction:*

  *An association **is a** group.*
- *Overgeneralisation:*

  *A place **is a** physical object.*

  *An amount of matter **is a** physical object.*
- *Suspect type-to-role link:*

  *A person **is a** living thing.*

  *A person **is a** causal agent.*

  *An apple **is a** fruit.*

  *An apple **is a** food.*

Most of these examples could be addressed by encoding more cases of multiple inheritance. The issue of roles and types is taken up by Trautwein & Grenon (2004), who consider the advantages of having a completely separate taxonomy for roles. They point out that the WordNet ontology tends to encode those roles with high real-world occurrence in the cultural environment which gave rise to WordNet, such that while many animals are found categorised as foods (Pustejovsky, 1991; 1995; Amaro et al., 2006), insects generally are not. Whether it is possible to capture all such complexities in an ontology is unclear, but certainly it is not possible in a mostly mono-hierarchical structure with underdefined relations such as the WordNet HYPERNYM/HYPONYM taxonomy.

Guarino (1998) concludes that most ontologies result from "a mixture of ad-hoc creativity and naive introspection". An analysis of WordNet's verb taxonomy (§2.2.2) confirms this. He proposes a much more formal approach to ontology construction.

Guarino classifies objects as concrete or abstract (e. g. Pythagoras' theorem), and concrete objects as continuants (e. g. an apple) and occurrents (e. g. the fall of an apple).

He asserts that that occurrents are generated by continuants, but does not say what the continuant is which generates the fall of the apple. He further asserts, as does Vossen (2002), that abstract objects do not have a location in space or in time. This assertion is incapable of being proved or disproved. Did Pythagoras' theorem exist before Pythagoras?[20] Abstractions are concepts. They exist in human minds. If abstractions exist independently of human minds, then they must exist in the mind of *God*, which is inconsistent with Guarino's otherwise *atheistic* ontology (see next paragraph). Otherwise the abstractions themselves are elevated to a divine status, which demands a *pantheistic* ontology.

These observations serve to demonstrate how tricky ontology construction is, pointing towards underlying philosophical assumptions in Guarino's work, which are inherent in his proposed ontological levels. He states that an animal as an intentional agent is dependent on an animal as a biological organism which in turn depends on an animal as a piece of matter. While this view may have widespread scientific support and may be fashionable, there is also a view that the dependence is in the opposite direction, as in Hindu philosophy, while during the mediaeval period, when modern European languages took shape, the fashionable view was that all three depend on God. It is not easy, perhaps impossible, to construct an ontology without any philosophical assumptions, and different philosophical assumptions are likely to generate different ontologies. In a lexical database the best ontology must be the one which best fits the language, which may not be the same for all languages and which may be culturally dependent with regards to philosophical fashion.

One must conclude that while a more formal approach to ontology is undoubtedly an improvement on an ad-hoc approach, Guarino's formalism is unconvincing. A formalism is required which is free of philosophical assumptions. The question remains as to whether this is possible.

[20] presumably so, as it was known to the ancient Babylonians and Egyptians.

2.2.1.2 Is a Correct Ontology Possible?

Brewster et al. (2005), take account of recent developments such as the Semantic Web, but argue that, irrespective of formalisms, it is impossible to build an ontology which is either free of philosophical assumptions or capable of fulfilling all likely requirements. Citing the highly scientific example of the Gene Ontology, they point out that an ontology is always out of date by the time it has been constructed, because knowledge is in a constant state of flux. In fact the real world also is in a constant state of flux[21]. They argue convincingly that in order to be finite, an ontology must necessarily lie.

Unlike Guarino (1998; §2.2.1.1), Brewster et al. show an awareness of the dependence of an ontology on a philosophical view, contrasting the traditional positivist view with more modern theories of knowledge, some of which acknowledge the need for change in knowledge representations and question whether knowledge from different theoretical concepts is ever comparable, given the dependence of the use of words and concepts on theory. Surprisingly views from cognitive science, as represented by Lakoff (1987), are not brought into their review of theories of knowledge. Lakoff systematically lays to rest the positivist view with its stable hierarchies such as those which dominate the WordNet taxonomy despite the theoretical basis of WordNet in psycholinguistics (Fellbaum, 1998; Miller, 1998).

Brewster et al. argue that any attempt to arrive at a set of precise and unambiguous concepts is doomed to failure, because any knowledge representation is necessarily a human expression and the development of knowledge itself depends on people discovering nuances in their forerunners' atomic concepts. Brewster et al. consider but reject the usefulness of corpora as sources for ontology construction on the grounds that text always has underlying assumptions, a body of assumed knowledge common to the writer and reader. While a text may challenge or modify these collective assumptions, it cannot avoid them; otherwise a university level book on a specialised aspect of a more

[21] The Gene Ontology is nevertheless useful.

general subject would have to begin with a full exposition of the more general subject from elementary first principles.

A novel approach to the discovery of semantic relations between words has been developed by LIRMM[22]. A set of internet games (*jeux de mots*; http://www.lirmm.fr/jeuxdemots) has been created which require the players to say which words in a set are related, and, at a more advanced level, to select, from a set of semantic relation types, which best fits the relationship between a pair of words. Players are rewarded when their answers agree with those of most other users. The game has been made available in several languages. Up to 29th. August 2010, 1,025,178 semantic relations (for French) had been identified in this way. The results are used by LIRMM and by GETALP[23]. This empirically produced data (available from http://www.lirmm.fr/~lafourcade/JDM-LEXICALNET-FR/) is suitable for the encoding of the kinds of relations found in WordNet.

2.2.1.3 Compatibility of Existing Ontologies

Returning to a more pragmatic level at which lexical databases can be constructed and used for machine translation, given an awareness of the pitfalls of existing ontologies, it is surprising to note the relative ease with which Knight & Luk (1994) manage to merge three ontologies (PENMAN, ONTOS and WordNet) and two dictionaries (Longman's Dictionary of Contemporary English and Harper-Collins Spanish-English Bilingual Dictionary) into the single PANGLOSS ontology for use in rule-based machine translation. This is achieved with the aid of the following algorithms:

- a definition match algorithm which matches definitions of different meanings of homonyms in different resources using the common words in the definitions,
- a hierarchy match algorithm which matches definitions of different meanings of homonyms using common subsumers in different ontologies and

[22] Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier. http://www.lirmm.fr
[23] Groupe d'Etude pour la Traduction Automatique et le Traitement Automatisé des Langues et de la Parole, Laboratoire d'Informatique de Grenoble; http://getalp.imag.fr/

- a bilingual match algorithm which matches sets of translation equivalents to WordNet synsets containing the same items.

The success of this approach perhaps depends on underlying similarities in the resources used, which in turn could suggest that the underlying philosophies of the various ontologies were similar from the outset.

Less straightforward was the integration of *Le Dictionnaire Integral* (LDI) with WordNet to create the Alexandria online translator (Dutoit & Papadima, 2006). Leaving aside the language difference, WordNet is mainly mono-hierarchical, whereas in LDI multiple inheritance is the norm. In LDI, the word "yen" is in the monetary unit *class* but also in the Japan *domain*; "warrior", "nobleman" and "Japanese" are all LDI HYPERNYMS of "samurai" while in WordNet, only "warrior" is a HYPERNYM. Dutoit & Papadima say that the LDI approach makes glosses like "money of Japan" for "yen" redundant[24]: the meaning of a word is defined by the topology of that part of the graph which links it to the relevant concept. The model has no need of synsets, because synonymy is discovered when two words share the same local topology. While in WordNet several word senses map to a single Synset, in LDI a relatively small number of concepts and combinations of concepts map to word senses. Treating the two resources as graphs, Dutoit & Papadima consider that the two cannot be merged, as there is no formal redundancy. To integrate the two effectively means importing the contents of WordNet into LDI, introducing the notion of synsets, mapping the French EuroWordNet synsets to the relevant word senses and adding glosses to the synsets.

2.2.1.4 Conclusions on Ontology

- WordNet fails to capture many instances of synonymy and MERONYMY.
- The *is a* (HYPERNYM/HYPONYM) and *has a* (HOLONYM/MERONYM) hierarchies in WordNet are flawed.

[24] The WordNet gloss for *yen* is in fact: "the basic unit of money in Japan; equal to 100 sen ". Dutoit & Papadima (2006) do not state whether or how the implied MERONYM is handled in LDI.

- An ontology based on formal principles is likely to be better than an ad-hoc one like that of WordNet.
- Any ontology will necessarily have underlying philosophical assumptions; it would be better in all cases if these were explicit.
- A perfect ontology is unlikely ever to be possible.
- Despite diverse formalisms and philosophies, it is sometimes possible to map between different ontologies.
- LIRMM's *jeux de mots* has the potential to offer a more empirical way of discovering semantic relations.

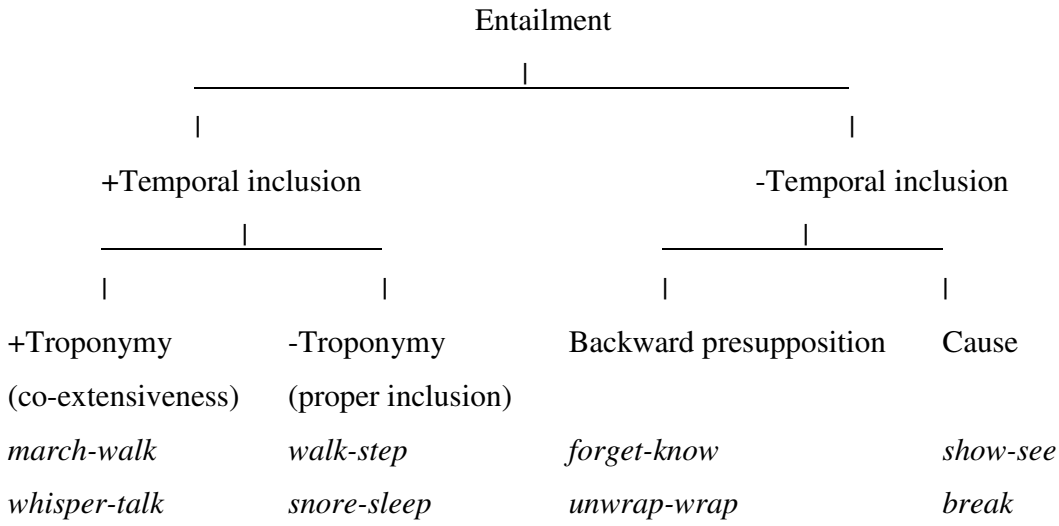
2.2.2 Investigation into the Verb Taxonomy

2.2.2.1 Introduction

Most studies on WordNet have focussed on nouns. The study presented in this section focuses mainly on verbs, for which ontological principles are even less clearly established. The HYPERNYM / TROPONYM and ANTONYM relations in WordNet involving verbs are to be examined. In the case of verbs, a HYPONYM is also called a TROPONYM. To "march" is the TROPONYM of to "walk" because to "march" is to "walk" *in a particular way* (Fellbaum, 1998). Because it seems intuitively likely for anomalies to be concentrated where the relational structure is more complex, the phenomenon of multiple inheritance in the hierarchical data structures formed by the HYPERNYM / TROPONYM relation is of particular interest. This has been analysed rigorously using the algorithm described in §2.2.2.2.1.

The only document which specifies what the WordNet verbal relations mean is Fellbaum (1998), who defines and specifies the various relations encoded between verbal synsets and considers troponymy and causation to be special cases of entailment (Fig. 2). Note that "proper inclusion" and "backward presupposition" are not encoded as separate relations but are subsumed by the general *entailment* relation.

Fig. 2: Specification of verbal relations (after Fellbaum, 1998)



Smrž (2004; p. 211) proposes a number of tests for validating wordnets. These include the following inconsistency checks:

- "dangling links (dangling uplinks[25])"
- "cycles in uplinks"
- "cycles in other relations"
- "topmost synset not from the defined set (unique beginners)"
- "non-compatible links to the same synset"

In fact, in the absence of a defined set of unique beginners, it is impossible to distinguish a "dangling uplink" from "topmost synset not from the defined set ".

Also listed are "queries retrieving 'suspicious' synsets or cases that could indicate mistakes of lexicographers" including:

- "multi-parent relations"
- "near antonyms differing in their hypernyms" (Huang et al., 2002; Vossen, 2002; §2.2.2.3.2)

[25] In the context of the verb taxonomy, an "uplink" means one or more HYPERNYM relations, so a "dangling uplink" occurs when a verb has one or more TROPONYMS but no HYPERNYM.

These tests have been applied in the development of BalkaNet. The following investigation seeks instances of the listed faults or potential faults within WordNet 3.0.

2.2.2.2 Hypernyms and Troponyms

In theory (Fellbaum, 1998), WordNet noun and verb synsets form a set of taxonomic trees, each with a unique beginner or root, excluding the possibility of multiple inheritance; in practice multiple inheritance is allowed where two HYPERNYMS of a synset are in different semantic categories (§2.2.2.2.5). Liu et al. (2004) accept that multiple inheritance across category boundaries is legitimate, but have found thousands of cases of *rings* (Appendix 3) within supposed trees, which arise when a synset has two HYPERNYMS within the same category, which themselves must, according to the specification, have a common HYPERNYM they have also found *isolators*, trees isolated within their own category whose only HYPERNYM lies in another category. The existence of the latter is acknowledged by Fellbaum (1998).

There are two other possible anomalies: one is a *cycle* (Appendix 3(c)), a special case of a ring where following the HYPERNYM relation in one direction leads back to where one started; the other is another kind of isolator, where a synset has no HYPERNYM at all. Liu et al. (2004) consider this possibility legitimate on the grounds that it applies to the unique beginners of each semantic category in WordNet. Although Fellbaum (1998) allows for more than one unique beginner per verb category, such cases are worthy of examination to see whether they correspond to her specification.

2.2.2.2.1 Algorithm for Identifying Topological Anomalies in Hierarchical Relations

An algorithm was developed to discover occurrences of these kinds of anomaly in WordNet 3.0, in the course of a more general investigation into multiple inheritance. The algorithm recursively models the direct and indirect HYPERNYMS of every synset as *an upside-down tree* (where the synset is the root and its most remote indirect

HYPERNYMS are the leaves). Where a cycle occurs, a stack error eventually results[26]; an *isolator* occurs where all the HYPERNYMS are in a different category to the synset under investigation; a ring is identified wherever a synset is found more than once in the same upside-down tree. This approach, unlike that of Liu et al. (2004), does not assume any correlation between semantic categories and HYPERNYMS and so can identify rings which straddle category boundaries. A simplified representation of the algorithm follows:

```
for each Synset
{
      hypernymCount = number of hypernyms
      if (hypernymCount == 0)
      {
            ROOT FOUND
      }
      else
      {
            categoryMismatches = 0;
            for each hypernym
            {
                  if current Synset.category != hypernym, category
            {
                        categoryMismatches++;
                  }
            }
            if (categoryMismatches == hypernymCount)
            {
                  ISOLATOR FOUND
            }
            upside-downTree = findIndirectRelations(currentSynset);
            if (hypernymCount > 1)
            {
                  nodeList = preorderEnumeration of tree;
                  while (tree has more nodes)
```

[26] In the final implementation, the stack error is pre-empted as soon as the root of any upside-down tree or sub-tree recurs elsewhere in the tree.

```
                {
                        currentSynset = nodeList.nextElement();
                        if (synsetList.contains(currentSynset))
                        {
                                RING FOUND
                        }
                }
        }
}

findIndirectRelations(Synset)
{
        upside-downTree = new upsideDownTreeNode(currentSynset);
        for each hypernym
        {
                try
                {
                        nextUpside-downTree
                        = findIndirectRelations(thisHypernym);
                        upside-downTree.add(nextUpside-downTree);
                }
                catch (StackOverflowError)
                {
                        CYCLE FOUND;
                }
        }
        return upside-downTree;
}
```

2.2.2.2.2 Cycle

The original implementation of this algorithm generated a stack error when applied to a number of verbal synsets: on investigation it was discovered that in each case the same

cycle was encountered, which is the only one in WordNet 3.0. It comprises 2 synsets, each of which is encoded as HYPERNYM of the other.[27]

2.2.2.2.3 Rings

Liu et al. (2004; p. 348) define a ring as being formed where a synset "has at least 2 fathers in its own category", which must necessarily, according to the specification, have a common ancestor also within that category. The algorithm presented here (§2.2.2.2.1) uses a broader definition of ring as any case where a synset has two HYPERNYMS such that these HYPERNYMS themselves have a common HYPERNYM or one of them is the immediate HYPERNYM of the other. However a distinction has been made between the different cases of ring with respect to membership of semantic categories. The same tests were applied to nouns for comparison (Table 2)[28]. Out of the 8 rings in the verb hierarchies, 4 belong to each of 2 topologies (Appendix 3, Tables 3-4).

Table 2: Rings in the WordNet taxonomy

Case with respect to semantic categories	Verbs	Nouns
Single category	5	1
Ancestry crosses categories but direct relations are in same category as headword	2	1984
Ancestry crosses categories and direct relations cross categories	1	379
TOTAL	**8**	**2364**
TOTAL using definition from Liu et al. (2004)	**7**	**1985**
Results using WordNet 2.0 obtained by Liu et al. (2004)	**17**	**1839**

Table 3: Verb rings with asymmetric topology (Appendix 3(a))

Initial Synset	Simple Hypernym	Compound Hypernym
warm up	exercise, work	work, put to work
reflate	inflate	change, alter
eat (transitive)	eat (intransitive)	consume, ingest
procrastinate	procrastinate, stall	delay

[27] synsets 202422663 {"restrain"; "keep"; "keep back"; "hold back"} glossed as "keep under control; keep in check" and 202423762 {"inhibit"; "bottle up"; "suppress"} glossed as "control and refrain from showing; of emotions, desires, impulses, or behavior".
[28] Total numbers of noun and verb synsets are given in §1.1.1.

Table 4: Verb rings with symmetric topology (Appendix 3(b))

Initial Synset	Hypernym 1	Hypernym 2	Grandparent
turn	turn, grow	discolour	change
inspan	yoke	harness, tackle	attach
outspan	unyoke	unharness	unhitch
smuggle	export	import	trade, merchandise

With the asymmetric topology (Appendix 3(a)), assuming that the relations are otherwise correct, it would be a simple matter to remove the link between the initial synset and the compound HYPERNYM, thus removing the dual inheritance and the ring. With the symmetric topology (Appendix 3(b)), no such simple remedy exists. Liu et al. assert that a ring implies a paradox because they assume that two HYPONYMS of a single HYPERNYM must have opposite properties in some dimension and therefore cannot have a common HYPONYM, as a HYPONYM must inherit all the properties of its HYPERNYM. In fact, two HYPONYMS can modify properties of their HYPERNYMS in two different dimensions (for a discussion, with particular reference to *qualia* properties see Amaro et al., 2006; §§1.1.5, 2.3.2.2), so there need not be any paradox. The symmetric ring starting from the word "turn" in the sense "the leaves turn in Autumn" involves different properties (Table 4): "turn, grow" is distinguished from "change" by specifying that the timescale is gradual, while "discolour" specifies which attribute is to change; "turn" in the above sense inherits both properties of gradual timescale and colour attribute. In the remaining three cases of symmetric rings, the gloss for the initial synset contains the word "or", to convey not a syntactic alternation but an ambiguity. The two HYPERNYMS in each case are in fact HYPERNYMS or synonyms of the respective two meanings, and the grandparent is indeed a common ancestor. The remedy here would be to split the ambiguous synsets into two, thereby removing the dual inheritance and the ring. We can conclude then that out of the eight *rings* among verbs, in seven cases a correction can be made and in one case the ring and the multiple inheritance are valid.

2.2.2.2.4 Dual Inheritance Without Rings

There are 31 verbs in WordNet which have two HYPERNYMS. None have more than two HYPERNYMS. The word "or" occurs in the glosses of nine of these verbs. There are four (possibly five) examples where dual inheritance can be justified in terms of inheritance of two different *qualia* (Amaro et al., 2006; §§1.1.5, 2.3.2.2; Table 5). The *formal* quale is concerned with what is physically done, while the *telic* quale is concerned with the purpose or end result of the action.

Table 5: Legitimate dual inheritance

Word form(s)	Formal quale	Telic quale
date, date stamp	stamp	date
assemble, piece	join, bring together	make, create
execute, put to death	kill	punish, penalize
carve	cut	shape, form

The fifth example (not in Table 5) is where "sing" (intransitive) is given as a HYPERNYM of "sing" (transitive). The other HYPERNYM of "sing" (transitive) is given as a "interpret, render" (necessarily transitive). The HYPERNYM of "sing" (intransitive) is given as "talk, speak", which is really a sister term whose common HYPERNYM would be "utter" (Miller & Johnson-Laird, 1976), which represents the *formal quale*, while "interpret, render" represents the *telic quale*. So, in this case, there is an *underlying* dual inheritance of different qualia properties.

2.2.2.2.5 Isolators

1593 examples were found of isolators among verbs and 2527 among nouns. These results approximate to those of Liu et al. (2004), who found 1551 verb isolators and 2654 noun isolators in WordNet 2.0. Since the concept of isolator is dependent on WordNet semantic categories, the 15 verb categories are tabulated in Appendix 4. Among 41 sample pairs of TROPONYM and HYPERNYM in different categories (Table 6), in 17 cases (rows 2 & 3) one verb's category can be considered a subset of the other's category e. g. *motion* and *creation* are subsets of *change,* and *competition* is a subset of *social*. By

manual evaluation, some 14 verb synsets (rows 4 & 5) were judged to be in the wrong category: examples among the HYPERNYMS are "form, take form", categorised as *stative* and "season, flavour" as *perception*. Examples among the TROPONYMS are "conspire, collude" as *cognition*, "live out, sleep out" as *consumption* and "air-condition" as *possession*. In 15 cases (row 7), the TROPNYM relation does not appear to match Fellbaum's (1998) definition (Fig. 2).

Table 6: Isolating relations

Row	Relation encoded as hypernymy across category boundaries	Instances
0	Categories mutually exclusive	1
1	Categories not mutually exclusive *of which:*	40
2	(Hypernym also belongs to troponym category)	(5)
3	(Troponym also belongs to hypernym category)	(12)
4	Invalid hypernym category	4
5	Invalid troponym category	10
6	Hypernym / troponym relation correct	26
7	Hypernym / troponym relation incorrect *of which:*	15
8	Troponym is troponym of one alternation of hypernym	1
9	Hypernym is cause of troponym	2
10	Troponym is troponym of cause of hypernym	2
11	Hypernym temporally includes troponym	1
12	Hypernym is precondition of troponym	1
13	Synonymous	5
14	Metaphor	1
15	No near relation	2

In 26 out of 41 cases (row 6), the HYPERNYM relation was judged to be correct, but the HYPERNYM category differs from the TROPONYM category. This arises because the WordNet verb categories are, for the most part, not mutually exclusive. The majority of these categories represent overlapping *semantic fields*. It is not therefore surprising that the *isolator* phenomenon occurs and that this does not necessarily imply an error. The only categories which could be considered not to overlap are *stative* with *change* and *creation* and the much smaller semantic field *weather* with most of the other semantic fields. The *stative* category belongs to the *Aktionsart* categorisation of verbs which distinguishes it from verbs of *activity*, *achievement* and *accomplishment* and is orthogonal to the categorisation of verbs into semantic fields (Vendler, 1967; Moens &

Steedman, 1988; Amaro, 2006). Moreover, a verb can belong to more than one *Aktionsart* category, as these categories apply to verbs *in contexts*.

The level of arbitrariness and incorrectness of the WordNet verbal semantic categories is greater than is the case for WordNet relations. Whereas the theoretical basis for WordNet relations is at least consistent within itself (whether one agrees with it or not) and the errors are of failure to conform to the specification, in the case of the semantic categories, the theoretical basis is itself inconsistent, being, as it is, a compromise between orthogonal systems of verb categorisation, dominated by a system of overlapping semantic fields.

The semantic categories in WordNet are based, according to Fellbaum (1998), on a standard work on psycholinguistics (Miller & Johnson-Laird, 1976). The latter discusses, in detail, verbs of motion, possession, vision and communication, which are the bases of the WordNet categories *motion*, *possession*, *perception* and *communication,* and identifies subclasses of these. Other semantic fields mentioned are contact (*contact*), bodily activity (*body*), thought (*cognition*) and affect (*emotion*). Miller & Johnson-Laird acknowledge that these categories overlap, but WordNet does not allow a verb to belong to more than one semantic category. Fellbaum (1998) and her team have added the remaining categories without providing any clear theoretical basis. Of these *competition* is subsumed by *social*, while *consumption* is subsumed by *body*. *Weather* would seem to be a fairly coherent and self-contained field, but the remaining categories *change*, *creation* and *stative* are not semantic fields at all but, if anything, are part of an orthogonal classification which is poorly adhered to.

2.2.2.2.6 Roots of the Verbal Taxonomy

There are 559 verb synsets in WordNet 3.0 which have no HYPERNYM, spread over all verb categories. Of these, 225 have no TROPONYMS either, meaning that they are completely disconnected from any hierarchical structure, leaving 334 which have TROPONYMS but no HYPERNYM. Of these, 96 have a single direct TROPONYM and

of these 80 have no indirect TROPONYMS. Excluding these 80, we are left with 254 verb synsets which have no HYPERNYM and more than 1 direct or indirect TROPONYM. This is very different from the theoretical position that each verb category has at most a handful of unique beginners (Fellbaum, 1998).

In the case of nouns, we find a different situation: of all the 7726 noun synsets without a HYPERNYM, 7714 have no HYPONYMS either; 7 have a single HYPONYM, leaving only 5 candidates for unique beginners of taxonomic trees. Of these only 1 has a depth > 1, which is synset number 100001740, "entity", the intended root of the entire taxonomy (Miller 1998). Many of the 7714 noun synsets with no HYPERNYMS or TROPONYMS have no other relations either and many are proper nouns. It is debatable whether proper nouns have any place in a lexical database (§4.3.4): where they are connected by any relation, then the connections are based on judgments such as "Albert Einstein was a genius", which, though one may agree, is of the nature of an opinion, impossible to verify and hence arbitrary. WordNet is supposed to be a lexical database, not an encyclopaedia. The following noun categories have no roots within them: 1, 2, 7, 8, 12, 13, 16, 19, 20, 22, 23, 24, 25, and 27.

To determine which verb roots are intended to be the unique beginners, an examination was made of all the 254 candidates. More than one candidate unique beginner was found in every verb category, the minimum being 5 for category 34 *consumption*. According to Fellbaum, category 38 *motion* should have two unique beginners "expressing translational movement" and "movement without displacement" respectively. These two meanings can be found among the 19 candidates in this category. Similarly category 40, *possession* should have 3 unique beginners, representing the basic concepts "give", "take" and "have", whereas in fact there are 15 candidates including these 3.

According to Fellbaum (p. 72), "communication verbs are headed by the verb *communicate* but immediately divide into two independent trees expressing verbal and nonverbal (gestural) communication". She continues: "these are not lexicalized in English." In fact WordNet 3.0 gives 7 senses of "communicate" all of which have

HYPERNYMS. Fellbaum identifies a further subdivision between spoken and written language, but the only reference to "write" among these 254 verbal synsets occurs in category 36: *creation*. Category 32 *communication* has 18 candidates. These include basic concepts like "utter" and "mean" at one extreme and very specific concepts such as "cheer up", "guarantee" and "designate" at the other. There appears to be no connection between the theory and the practice here.

It is always possible to define a verb in terms of another verb with one or more arguments. This is a method of identifying HYPERNYMS, which appears to have been used extensively, though inconsistently, in the construction of WordNet, using the glosses for semi-automatic HYPERNYM generation. Full automation of such a technique would lead inevitably to a *cycle* (§2.2.2.2.2). There have to be unique beginners in order to avoid this (Blondin-Massé et al., 2008).

On a dataset of this size (254 synsets), it is also feasible to manually assign HYPERNYMS for most of the verbal synsets. There is clearly more than one possible solution in many cases. In some cases, it is sufficient to provide a more generic verb or verbal phrase as a HYPERNYM; in other cases, a combination of a verb and one or more arguments (usually involving an additional verb) is required to define the verb. In these cases the first or *auxiliary* verb can be considered as the HYPERNYM, for instance *to learn* could be defined as *to start to know: learn* is then a TROPONYM of *start*, not of *know*, because learning is *a kind of* starting, but not *a kind of* knowing; the *learning* process is *temporally co-extensive* (Fig. 2) with the process of *starting to know* but not with the state of *knowing*. The same applies to *"forget"* defined as *stop remembering*. A similar approach has been applied to the development of a top level preposition taxonomy (§4.2.4.3).

2.2.2.3 Antonyms

ANTONYMS differs in two ways from the other relations we have been examining: first, it is a symmetric or reciprocal relation: the relation traversed in one direction being of the

same type as the relation traversed in the other; second, ANTONYMS are defined between word senses and not between synsets. The reasons for this are rooted in psycholinguistics (Fellbaum, 1998; but see §4.3.5).

Table 7: Multiple ANTONYM scenarios

Phenomenon	**Freq.**
Spelling variation of which:	7
(-ise / -ize)	(6)
Single correct antonym	10
Ambiguity	2
Two antonyms in same synset	2
No valid antonyms	5
TOTAL	**26**

2.2.2.3.1 Multiple Antonyms

As with the HYPERNYM/HYPERNYM relations, ANTONYMS has been investigated by finding verbs which have more than one ANTONYM and manually evaluating the validity of the ANTONYM relations. There are 26 such cases among the verbs in WordNet. Table 7 categorises the instances of multiple ANTONYMS. Of the 10 cases in Table 7 where only one of the ANTONYMS was judged correct, two are cases of confusion over the causative/inchoative alternations of "lock" and "unlock", one confuses transitive and reflexive uses of "dress", one confuses transitive and intransitive uses of "begin" and one confuses *event* and *state* meanings of "clasp". "Profit" and "lose" are correctly encoded as ANTONYMS of each other while "break even" is encoded as a second ANTONYM of both. This suggests an ambiguity in the concept of ANTONYM. "Lose" means *negative* profit while "break even" means *zero* profit (and *zero* loss). So there is a scale from "profit" (+*ve*.) through "break even" (*zero*) to "lose" (-*ve*.) The concept ANTONYM is being used in WordNet both for the relation between +*ve*. and -*ve*. and for the relation between +*ve*. (or -*ve*.) and *zero*. Postulating a new relation of SEMI-ANTONYM could resolve this, eliminating the need for multiple ANTONYMS for a single concept. Vincze et al. (2008) propose an orthogonal subdivision of encoded ANTONYMS into true ANTONYMS and *converses*, like "buy" and "sell" or "profit" and

"lose", where both members of the pair refer to the same event from an opposite point of view.

2.2.2.3.2 Antonyms Without a Common Hypernym

A pair of ANTONYMS should have a common HYPERNYM (Huang et al., 2002; Vossen, 2002; Smrž, 2004). Excluding 11 pairs of verb ANTONYMS which either have multiple inheritance or include one or more TROPONYMS of the *cycle* referred to in §2.2.2.2.2, there are 316 pairs of verb ANTONYMS in WordNet which do not have any direct or indirect common HYPERNYM, as against 222 which do.

Table 8: ANTONYMS *with no common HYPERNYM*

Phenomenon	**Freq.**
Missing common hypernym	16
Common hypernym in one ancestry	5
False antonymy	6
Other	1
TOTAL	**28**

Table 8 categorises instances of ANTONYM pairs with no common HYPERNYM. The case of "disembark" : "embark" is of special interest, because the head of the ancestry for "disembark" is "arrive" and the head of the ancestry for "embark" is "enter", which can be construed as a TROPONYM of "arrive". This paradox arises because the ancestry of "disembark" is defined with reference to the *journey* while the ancestry of "embark" is defined with reference to the *vehicle*. Both frames of reference are valid and so "disembark" can be considered as a TROPONYM of "arrive" with reference to the *journey* and of "leave" with reference to the *vehicle*, while "embark" can be considered as a TROPONYM of "leave" with reference to the *journey* and of "arrive" with reference to the *vehicle*. This could be regarded as legitimate dual inheritance, based on dimensions orthogonal to all *qualia*.

2.2.2.4 Conclusion

Any application of WordNet which measures semantic distance employs WordNet relations to do so (§6.1). Banerjee & Pedersen's (2003) WSD results (§6.1.1.4) are noticeably poorer for verbs than for nouns. Moreover, while the most useful relations for nouns were HYPONYM and MERONYM, in the case of verbs, the example sentences proved more useful than either. Their best results for verbs were obtained by using all WordNet relations indiscriminately. This finding may reflect the poor quality of the verbal relations and suggests that the limited success achieved by algorithms which measure lexical distance using WordNet relations depends on the fact that when a relation is encoded, some relation does in fact exist, even though the type of relation encoded is not necessarily correct. Algorithms which employ specific relations seem to be succeed better with the more clearly defined relations, namely HYPERNYM and ANTONYM (Huang et al., 2002). These observations drive us towards the conclusion that improvements to the WordNet relations might well be useful for improving on the performance of WordNet as a tool for interlingual tasks and WSD.

Ignoring the absence of some valid semantic relations, which is difficult to quantify, in the course of this investigation, many shortcomings have been discovered in the encoding of relations in WordNet, where the implementation does not conform to the theory in a high proportion of instances. It would seem appropriate at this point to recall the list of consistency checks proposed by Smrž (2004; §2.2.2.1).

Over 500 cases have been found among verbs alone of "topmost synset not from the defined set (unique beginners)" or "dangling uplinks". One instance has been found of "cycles in uplinks". A number of "multi-parent relations" have also been found. In studying antonyms, we have also found instances of "non-compatible links to the same synset" and abundant instances of "antonyms differing in their hypernyms".

Given that Smrž's tests have been applied in the development of BalkaNet, it is clear that the standard of quality control for WordNet is not as high as it is for BalkaNet, a

discovery which is shocking, given the reliance of the construction of BalkaNet on WordNet.

This investigation culminated in the presentation of some of the findings at the COLING 2008 conference (Richens, 2008). The main conclusions can be summarised as follows:

- The implementation of verbal relations in WordNet does not conform to the specification in a high proportion of instances.
- In their present state, the verbal relations in WordNet serve only to indicate where a relation exists between two verbs, often not defining correctly what type of relation exists.
- Topological anomalies can be corrected.
- The only valid cases of dual inheritance are where different but compatible properties are inherited. Many more such relations could be encoded.
- WordNet semantic categories for verbs are, for the most part, not mutually exclusive and lack a consistent theoretical basis. The level of arbitrariness and incorrectness of the categories is greater than that of the relations. It is not possible to encode semantic fields correctly on the basis of one category per verb.
- A new proposed relation, SEMI-ANTONYM is defined.
- The ANTONYM relation should be redefined as holding between synsets rather than word senses (§4.3.5).
- ANTONYM ancestries can be made symmetric by correcting HYPERNYM errors.

Because this investigation into errors originally highlighted by Smrž (2004) and Liu et al. (2004) has revealed serious anomalies among verbs, and others (Wong, 2004) have found similar anomalies among nouns, it is worth giving consideration to any methodology which can assist in the automatic detection of valid HYPERNYM / HYPONYM relations for any POS.

One approach to automatically generating HYPERNYM / HYPONYM relations is by selecting the main terms from the glosses and using the synsets containing the senses for these terms as HYPERNYMS for the synsets containing the glosses. The high proportion of HYPERNYM word forms in the glosses suggests that the taxonomy has, at least in part, been encoded in this way, so that the taxonomy generated mirrors that obtained by digraph analysis of the glosses (Blondin-Massé et al., 2008). The difficulty with this approach is determining which sense of the proposed HYPERNYM word is intended. This problem has been addressed by the WordNet Gloss Disambiguation Project, culminating in the release in XML format of the Princeton WordNet Gloss Corpus (http://wordnet.princeton.edu/glosstag) in January 2008. This development opens up the possibility of rebuilding the entire taxonomy automatically on the basis of the disambiguated glosses. While the results of implementing such a procedure can only be as good as the glosses themselves, it would at least result in a consistent encoding of the hierarchical relations. An alternative basis for reorganising the verb taxonomy might be to infer it from the syntactic properties of the verbs (§2.3.2). Before this possibility can be seriously considered, we need to look at how verb syntax is represented in WordNet.

2.3 Syntax

Syntax is the first requirement on the road from computer representation of lexical data to computer representation of semantics (Hanks, 1997; Jackendoff, 1983). Verb syntax in WordNet is represented mainly by the WordNet sentence frames (§1.1.3), which are here investigated in detail.

WordNet provides a set of 35 generic sentence frames in the file *frames.vrb*, available with WordNet and listed in Appendix 2. The frames are referenced by number from each verb synset, in an attempt to define the arguments the verbs in the synset can take. Unfortunately, although a few possible prepositions are indicated, the global wildcard "PP" is extensively used without going into more detail. The only explicit selectional restrictions on the arguments are animate or inanimate roles as *somebody* or *something*.

2.3.1 WordNet Sentence frames

WordNet sentence frames (Appendix 2) are allocated sometimes to a synset and sometimes to an individual word sense. In encoding them in the Java model (§1.3.2.3), each frame was instantiated as an object of class `WordnetVerbFrame` with its frame number as an identifier. For the sake of structural consistency, each verb sense has been given its own set of frame numbers, even where these are the same for every verb in the synset. This made it easier to calculate how many different sets of frames (hereafter *framesets*) are present in each synset (Table 9).

Table 9: Distribution of framesets among verb synsets

Frameset count	Number of verb synsets
0	0
1	13550
2	212
3	4
4	1
> 4	0

2.3.1.1 Synsets with More than 2 Framesets

The 5 synsets which have more than 2 framesets were examined in detail in order to evaluate the correctness of the frame assignments. Each frame assignment was manually marked as correct or incorrect, based on native speaker familiarity, or as unknown in the case of unfamiliar verbs from American dialect or slang. None was found to be correct. Examples of incorrect frames are transitive frames for "get word" and "refer" (inconsistently glossed as "make reference *to*") which are intransitive and require the prepositions "of" and "to" respectively. Missing frame assignments include frame 22 for "get word" as in "somebody gets word of something" and frames 8 and 24 for "need" glossed as "require as useful, just, or proper" as in "somebody needs something" and "somebody needs somebody to do something".

2.3.1.2 Synsets with 2 Framesets

The same procedure was carried out with a sample of 33 verb synsets with two framesets. Only 3% were found to be correct and complete. Within this data, the synset {"confront", "face", "present"}, is ambiguous. It is glossed "present somebody with something, usually to accuse or criticize" with examples:

1. "We confronted him with the evidence"
2. "He was faced with all the evidence and could no longer deny his actions"
3. "An enormous dilemma faces us"

The gloss is consistent with examples (1) and (2), but inconsistent with (3) which represents an alternation of the verb "face".

Synset {"show", "usher"} is glossed "take (someone) to their seats, as in theaters or auditoriums". Here there is a missing frame, which does not occur in the list of 35 frames recognised by WordNet: ("Somebody ----s somebody to something") is not in the list, but only the generic equivalent ("Somebody ----s somebody PP").

There is an inconsistency in how WordNet handles verbal phrases of the form verb + *w*, where *w* is a word which can be used as either adverb or preposition[29], depending on whether it has a nominal argument in the context, although the presence or absence of such an argument does not change the meaning of the phrase. Sometimes the phrase is encoded as a word form within a synset, with transitive and intransitive frames, and sometimes only the verbal component is encoded, with one or more of frames 20, 21 and 22 which take a prepositional phrase as an argument.

Synset {"partake", "share", "partake in"} displays this problem: the gloss is: "have, give, or receive a share of". For no obvious reason "share in" is not listed. The frames provided are no. 8 (transitive) for all three verbs and 2 (intransitive) for "partake" only. This is incorrect because "partake" cannot be used transitively, though "partake in", treated as a verb in itself, clearly can. No frames carrying prepositional phrase arguments are listed.

[29] frequently termed a particle, a term avoided in this thesis (§1.1.4).

While encoding "partake in" as a verb covers the prepositional phrase governed by "in" for the verb "partake" it does not cover the prepositional phrase governed by "in" for the verb "share", nor does it cover the phrases "partake of" and "share with".

2.3.1.3 Synsets with 1 Frameset

The same procedure was carried out on a sample of 239 verbs in 136 synsets with a single frameset. 38% were found to be correct and complete. In many cases, the examples provided show a verb in a frame which is not within its frameset, although perfectly correct (Table 10). Where no frame number is shown, the frame from the example has not been encoded because there is no such frame within WordNet. These frames are not unusual. In the remaining cases, the frames have been encoded without reference to the examples.

Table 10: Frames missing from single frameset sample

Synset ID	**Example**	**Word forms**	**Missing frame**	
			No.	Syntax
200756649	She pretends to be an expert on wine	profess, pretend	28	Somebody ..s to INFINITIVE
200870577	She warned him to be quiet	warn	28	Somebody ..s to INFINITIVE
200977689	His wife declared at once for moving to the West Coast	declare	n/a	Somebody ..s for Ving something
201373718	brush the bread with melted butter	brush	31	Somebody ..s something with something
201392080	The birds preened	preen, plume	2	Somebody ..s
201569896	The mansion was retrofitted with modern plumbing	retrofit	31	Somebody ..s something with something
201605404	The ivy mantles the building	mantle	11	Something ..s something
201668421	illustrate a book with drawings	illustrate	31	Somebody ..s something with something
201768630	The event engraved itself into her memory	engrave	n/a	Something ..s something PP
201969601	the earth's movement uplifted this part of town	uplift	11	Something ..s something
202348057	It was recommitted into her custody	recommit	21	Somebody ..s something PP
202384940	I invited them to a restaurant	invite	20	Somebody ..s somebody PP

Table 11: Additional frames required

Synset ID	Word forms	Additional frames	Example
202000547	show, usher	Somebody ..s somebody to something	The usher showed us to our seats
202680814	discontinue, stop, cease, quit, lay off	Somebody ..s from V-ing something	*He ceased from smoking tobacco*
200870577	warn	Somebody ..s somebody against Ving something	*He warned him against smoking tobacco*
	discourage	Somebody ..s somebody from Ving something	*He discouraged him from smoking tobacco*
	admonish	Somebody ..s somebody against Ving something	*He admonished him against smoking tobacco*
200977689	declare	Somebody ..s for Ving something	His wife declared at once for moving to the West Coast
201373718	brush	Somebody ..s something with something	brush the bread with melted butter
		Something ..s something with something	*The car-wash brushed the car with soap*
201410223	strike	Somebody ..s somebody adj./n.	The boxer struck the attacker dead
		Something ..s somebody adj./n.	*The collision struck the passenger dead*
201490958	yoke	Somebody ..s somebody adv.	Yoke the draft horses together
201768630	engrave	Something ..s something PP	The event engraved itself into her memory
201894520	breeze	Somebody ..s adv.	*She breezed in*
202205272	take	Somebody ..s something from something	*He took the jar from the shelf*
		Somebody ..s somebody from somebody	*He took her child from her*
		Somebody ..s somebody from something	*He took her from the school*
		Something ..s something from somebody	*The wind took my hat from me*
		Something ..s something from something	*The storm took the roof from the house*
		Something ..s somebody from somebody	*Death took his parents from him*
		Something ..s somebody from something	*His new job took him from home*

2.3.1.4 Additional Frames

We are concerned here only with frame elements which are semantically required by a verb, in one or more of its syntactic alternations. Table 11 lists all the additional frames identified as being required by the data so far, in addition to the 35 defined. The examples

illustrate the missing frames. Those in italics are concocted from imagination; the others are in WordNet.

2.3.2 Frame Inheritance

2.3.2.1 Valency

Valency is a concept borrowed originally from chemistry. In linguistics it is generally applied to verbs to represent the number of mandatory nominal arguments they require (Crystal, 1980; Verspoor, 1997; Pala, & Smrž, 2004), ranging from zero for "rain" ("it" in "It is raining" carries no semantic content and is redundant in some languages e. g. Spanish "Llueve") through to at least 3 for "put" as in "I put the book on the table." which requires subject, object and a prepositional phrase of destination.

2.3.2.2 Theory of Frame Inheritance

Amaro (2006) found verbs "mover" ("move" transitive) and "tirar" ("take") with valencies 2 and 3 respectively in a HYPERNYM / TROPONYM relation in a Portuguese wordnet. He also found verbs "mover-se" ("move" intransitive) and "andar" ("walk"), with equal valency in the same relation. In the latter case the TROPONYM is specialised from the HYPERNYM by an implicit specification of *manner* of movement. He identifies other specialisations of TROPONYMS with respect to their HYPERNYMS as corresponding to thematic roles such as *goal*.

Amaro et al. (2006) use English examples to show that the number of arguments can be greater or smaller for a TROPONYM than it is for its HYPERNYM: for instance "put" is a TROPONYM of "move" (transitive) because to put something is to move it in a particular way, but while "move" only requires two arguments, subject and object, and expression of the *goal* (destination) is optional, for its TROPONYM, "put", the goal argument is compulsory, such that the HYPERNYM has valency 2 and the TROPONYM

has valency 3. "Box" (verb) is a TROPONYM of "put" (to "box" is to "put" in a particular way), but *incorporates* the goal, thereby reducing the number of arguments required to 2. Thus some arguments are inherited from HYPERNYM to TROPONYM and others become *shadow arguments*. The development of these concepts leads to the formulation of rules for *frame inheritance*.

2.3.2.3 Investigation into Frame Inheritance

It is reasonable to expect that some verb arguments be inherited through the HYPERNYM / TROPNYM taxonomy (Pustejovsky, 1991; Amaro, 2006; Amaro et al., 2006), while some arguments may be added or deleted by a TROPONYM. Although the WordNet set of sentence frames is incomplete, and the frames using prepositional phrases are underdefined with respect to the choice of preposition, it should still be possible to identify which frames inherit from which others through the simple mechanism of adding one argument to the existing set. The table in Appendix 5, with frames arranged in order of valency, defines the natural inheritance from one frame to another. Note that frame 23 has been ascribed a valency of 1.5 because the genitive is semantically, though not syntactically, an argument of the verb; it *semantically* inherits from frame 8 which has a valency of 2.

Appendix 5 encapsulates frame inheritance according to the following rules, based on Amaro et al. (2006; §2.3.2.2):

- A TROPONYM can inherit a frameset from its HYPERNYM without adding any external arguments.
- A TROPONYM can inherit a frameset and add an argument thereby instantiating another frame.
- A TROPONYM cannot have any frame whose valency exceeds that of its HYPERNYM by more than one.
- A TROPONYM cannot drop an argument at the same time as adding one.

- The valency of a TROPONYM can only be less than that of its HYPERNYM where an inherited argument becomes a shadow argument, incorporated into the meaning of the verb.

Where the frameset of either HYPERNYM or TROPONYM or both contains multiple frames, a distinction can be drawn between the TROPONYM *inheriting* correctly, meaning that each of the TROPONYM's frames inherits correctly from at least one of the HYPERNYM's frames, and the HYPERNYM *bequeathing* correctly, meaning that each of the HYPERNYM's frames is correctly inherited by at least one of the TROPONYM's frames.

2.3.2.3.1 Algorithm for Validating Frame Inheritance

Appendix 5 was used to associate a list of inheritable frames with each `WordnetVerbFrame` object in the model. An algorithm was devised to determine whether the frame inheritance is correct for each HYPERNYM / TROPNYM relation, allowing inheritance according to the table in Appendix 5, but also inheritance by deleting an argument, which is the *reverse* of normal inheritance which adds an argument, to allow for shadow arguments. The algorithm models the HYPERNYM / TROPONYM hierarchies as trees, where the HYPERNYM is the parent and the TROPONYM is child.

```
investigate inheritance of verb frames
{
      for each synset
      {
            if (hypernym_count == 0)
            {
                  tree = find indirect relations(thisSynset,
                  HYPONYM);
                  if ((hyponym_count > 1) OR (tree.depth() > 1))
                  {
                        report WN3 Verb Frame
                        Inheritance(thisSynset);
```

```
                }
            }
}

find indirect relations(thisSynset, RELATION)
{
      tree = new tree_node(thisSynset);
      for each RELATION
      {
                next_tree = find indirect relations(RELATION);
                tree.add(next_tree);
      }
      return tree;
}

report WN3 Verb Frame Inheritance(this_synset )
{
      if (child_count > 0)
      {
            while (more_children)
            {
                  check valid inheritance(this_synset, nextChild);
                  report WN3 Verb Frame Inheritance(nextChild);
            }
      }
}

check valid inheritance(parent, child)
{
      if (parent has multiple framesets) OR (child has multiple
      framesets))
      {
            return false;
      }
      matches = table of Boolean values;
      for (each child Frame)
      {
```

```
            child_inherits_correctly = false;
            for (each parent frame)
            {
                  match = ((child_frame == parent_frame)
                  OR (child_frame inherits parent_frame )
                  OR (parent_frame inherits child_frame ));
                  child_inherits_correctly = child_inherits_correctly
                  OR match;
            }
      }
      parent_bequeaths_correctly = false;
      for (each parent frame)
      {
            for (each child Frame)
            {
                  parent_bequeaths_correctly =
                  parent_bequeaths_correctly OR match;
            }
      }
      return (child_inherits_correctly AND
      parent_bequeaths_correctly);
}
```

The algorithm was applied to the WordNet data, excluding 744 HYPERNYM / TROPONYM relations involving multiple framesets. Some 8937 relations were found to conform to the requirements for frame inheritance, while 3486 failed to meet these requirements.

2.3.2.3.2 Extended Definition of Valid Frame Inheritance

The analysis showed many cases where inheritance took place by imposing tighter selectional restrictions, where one argument changed from "something" to "somebody". Such inheritance can be considered legitimate as it does not violate the rules. This kind of inheritance is only valid unidirectionally since the TROPONYM must be more specific than the HYPERNYM (Appendix 6). In each case the valency of the TROPONYM's

frame must be the same as that of the HYPERNYM, except in the case of frame 23 inheriting from frame 1, where the genitive is added.

There are also HYPERNYMS which accept either "something" or "somebody" for an argument, with TROPONYMS which only accept "something", very often something quite specific. For instance "mail" can be considered as a TROPONYM of "send", but whereas one may "send" *somebody* or *something*, one may only mail *something*. In this case, assuming that the destination or recipient is not expressed, frame 8 inherits from the frame pair (8, 9).

Some frames specify arguments which are incompletely defined, for instance frame 10 specifies the *Adjective/Noun* in frame 6 is to be *somebody*, while frame 11 specifies the *Adjective/Noun* in frame 6 is to be *something*. Frame 17 specifies the preposition "with" and the preposition's argument as *something* and so inherits from frame 20, which merely specifies a prepositional phrase. These are cases of unidirectional inheritance. Frames 4 and 6 have bidirectional inheritance on the grounds that a prepositional phrase can substitute for an adjective and vice versa.

2.3.2.3.3 Adapted Algorithm to Incorporate Broader Definition of Valid Frame Inheritance

The algorithm was adapted slightly to distinguish between bidirectionally and unidirectionally valid inheritance:

```
check valid inheritance(parent, child)
{
      if (parent has multiple framesets) OR (child has multiple
      framesets))
      {
            return false;
      }
      matches = new table of Boolean values;
      for (each child Frame)
```

```
        {
              child_ inherits_correctly = false;
              for (each parent frame)
              {
                    match = ((child_frame == parent_frame)
                    OR (child_ frame unidirectionally inherits
                    parent_frame )
                    OR (child_frame bidirectionally inherits parent_
                    frame )
                    OR (parent_frame bidirectionally inherits child_
                    frame ))
                    OR child_frame unidirectionally inherits (parent_
                    frame AND self);
                    child_inherits_correctly = child_inherits_correctly
                    OR match;
              }
        }
        parent_bequeaths_correctly = false;
        for (each parent frame)
        {
              for (each child Frame)
              {
                    parent_bequeaths_correctly =
                    parent_bequeaths_correctly OR match;
              }
        }
        return (child_inherits_correctly AND
        parent_bequeaths_correctly);
}
```

With this revised algorithm, the number of relations with valid inheritance was 10281 while the number failing was 2142.

2.3.2.3.4 Final Evaluation of Frame Inheritance

In order to gauge the extent to which the relations or the framesets were incorrect among cases of invalid inheritance, a sample of 53 relations (involving 106 synsets) violating the relaxed rules for frame inheritance was taken from the data generated by the revised algorithm. There were no multiple framesets within the sample. The correctness of both framesets and relations was manually evaluated. Ignoring 7 synsets with animals as arguments[30], 30 out of 99 synsets had incorrect frames and 48 had missing frames, out of which 5 require frames which are not listed in WordNet. 37 synsets (34.91%) were considered correct, as having no incorrect or missing frames. 8 synsets with a single framesets were found to require multiple framesets in order for all the verbs in them to be encoded with the correct frames. Appendix 7 evaluates the correctness of the HYPERNYM / TROPONYM relations within this dataset.

Appendix 7 evaluates some relations as "reversed", where the inheritance of framesets was correct in the opposite direction to that of the encoded relation. Others are evaluated as "indirect" where the TROPONYM cannot inherit validly from the HYPERNYM but can inherit from an *abstract* synset interposed between the two which in turn inherits from the HYPERNYM. To put this in another way, *remote* inheritance should be allowed, meaning that if frame *a* does not validly inherit from frame *b*, but there are abstract verbal concepts $c_1...c_n$, which would inherit validly from *b*, and would be inherited from validly by *a*, then the inheritance from *b* to *a* should be allowed.

It is clear from the results obtained, that if verbs were correctly allocated to synsets, and sentence frames and relations correctly encoded, there would be a strong correlation between *semantic inheritance* of *verb meaning* and *syntactic inheritance* of *sentence frames*, to such an extent that a correct encoding of sentence frames could be used to guide a less arbitrary encoding of hierarchical semantic relations between verb meanings.

[30] Animals are inconsistently treated as "somebody" or "something".

We can conclude from this study of WordNet sentence frames that they are not a suitable vehicle for the representation of verb syntax for the following reasons:

1. Many encoded sentence frames are not appropriate for the verbs to which they are assigned.
2. Many valid frames are not encoded.
3. Many possible frames are not included in the list of 35.
4. Many synsets contain verbs which have different syntax but have not been provided with multiple framesets.
5. Mis-encoded relations and frames obscure the relationship between semantic and syntactic inheritance.

Experiments have been undertaken to replace the WordNet sentence frames with an alternative set empirically derived by parsing the usage examples[31]. Although a version incorporating alternative frames was successfully produced[32], it is not discussed in this thesis because of reservations about possible flaws in the algorithm which evaluates the parses and also because attempts to validate it against parsed sentences from the BNC produced results which were incomplete, inconsistent and inconclusive. It is hoped that this line of research will reach a satisfactory conclusion in the future and a forthcoming publication on this subject can be expected. This would allow the verb taxonomy to be reorganised in such a way as to conform to principles of frame inheritance. To do this properly however would probably require a reduction of the excessive verb polysemy and a review of the allocation of verbs to synsets.

2.4 Conclusions on WordNet

The research presented above has confirmed the following shortcomings of WordNet, some identified by previous researchers and others discovered in the course of the investigation:

[31] by integrating the Stanford Parser, available as Java classes, into the WordNet model, from http://nlp.stanford.edu/software/lex-parser.shtml#Download.
[32] serialised as *cubnet.wnt*.

- Encoding is arbitrary (whether manual or automatic) leading to incorrect semantic relations (Wong, 2004; §2.2.2).
- Some semantic relations are incorrect or absent (§2.2).
- The granularity is too fine, some synsets not being semantically distinguishable from each other (Vossen, 2002; 2004; EU, 2004; §2.1.2).
- The structure has not been validated (Liu et al., 2004; Smrž, 2004; §2.2.2).
- The verb categories are arbitrary (§2.2.2.2.5).
- The set of sentence frames is insufficient, being explicit only for selected prepositions in selected frames.
- The representation of selectional restrictions is crude (§2.3).
- The encoding of sentence frames is inconsistent with the examples given (§2.3).
- Some parts of speech are missing, in particular prepositions (addressed in §4.2).
- Arbitrary encyclopaedic information is found in synsets without HYPERNYMS but connected by INSTANCE or HOLONYM relations (§§2.2.2.2.6; addressed in §4.3.4).

Although it would be desirable to correct all the erroneous relations in WordNet, the manual overhead of doing so would be too great to be feasible within the context of this project. The manual reassignment of words to synsets and re-evaluation of individual relations between synsets would require many person-years of lexicographic effort.

The overhead of correcting the relations between verbs in WordNet could be reduced by using the glosses as a guide to redesigning the taxonomy (§2.2.2.4). The internet game approach (§2.2.11.2) also could contribute to the correction of semantic relations. An alternative approach is to use the principles of frame inheritance (Amaro, 2006; Amaro et al., 2006; §2.3.2). As sentence frames are inheritable, they could be used to inform a further correction of the verb taxonomy. However the quality of the existing sentence frames is not sufficient to support such an operation (§2.3.1). Correction of the sentence frames could be achieved by parsing of the usage examples (§2.3.2.3.4). Frame inheritance and gloss analysis could then be used in tandem for correction of the

taxonomy. Such an approach would highlight any inconsistencies between the glosses and the usage examples, which would be useful in its own right.

This proposal for correction of the sentence frames and the verb taxonomy has to wait for another research project. Instead, what is proposed for this project is a computational approach to those corrections and enhancements which can for the greater part be automated, though the need for manual intervention cannot be ruled out.

The immediate remedies proposed are the encoding of prepositions, limited correction of some types of semantic relation and some pre-cleaning of data, to reduce the amount of arbitrary encyclopaedic information. Many incorrect semantic relations will remain: it will be interesting to observe whether their negative impact on a WSD algorithm (*Extended Gloss Overlaps*; Banerjee & Pedersen, 2002; 2003; §6.1.1.4) which uses WordNet relations can be diluted by supplementing them with morphological and morphosemantic relations, empirically discovered through morphological analysis, in an enriched lexical database or morphosemantic wordnet. It also will be interesting to compare the performance of such a WSD algorithm when WordNet semantic relations are excluded and only empirically discovered morphological and morphosemantic relations are used (§6).

3 Investigation into Morphology

Derivationally related words, as distinct from words which have a co-incidental morphological resemblance, are necessarily also semantically related in some way. The assignation of semantic relation types to relations based on derivational morphology is challenging (§3.1.3), but because of the semantic significance of many morphological relations, any lexical database, including WordNet, which is deficient in such information, could benefit enormously from enrichment with such relations.

The aim of this section is to find the best methods of morphological analysis for the purpose of morphological enrichment of a lexical database. A review of other work in this field starts with the Porter (1980; §3.1.1) stemmer which implements *generalised spelling rules*. This stemmer was used in the development of the CatVar database (§3.1.2). The possibility of using CatVar data as an alternative to morphological analysis is considered, but rejected, though it is found to be a useful starting point for the formulation of morphological rules (§3.2.2.1). Various proposals for the morphological enrichment of wordnets and the creation of morphological wordnets are reviewed (§§3.1.3-3.1.5), some of which suggest a rule-based approach. The concept of a *derivational tree* is found to be particularly useful as it specifies the direction of derivation. The requirements for morphological enrichment and the limitations of WordNet derivational pointers are considered and the possibilities of the rule-based approach, beyond simple generalised spelling rules, are explored experimentally in §3.2, being applied to both suffixation and suffix stripping, and offering the potential for the discovery of morphosemantic relations.

An alternative to the rule-based approach is the deployment of morphological analysis algorithms for the automatic identification of morphemes. The best existing word segmentation algorithms are reviewed (§3.3), but are found all to be subject to the same *segmentation fallacy*, the naive assumption that a satisfactory morphological analysis of a word can always be obtained by segmentation. An entirely new algorithm for automatic

affix discovery through the creation of *affix trees* applying a *duplication criterion* is presented in §3.4. Heuristics using *affix frequencies, parent frequencies* and *stem validity quotients* for sorting character combinations in accordance with a *semantic criterion* are described and evaluated, and an optimal heuristic is identified. This leads towards the conclusion that the best morphological analysis will be obtained by adopting a *hybrid model*, making use of both the Automatic Affix Discovery Algorithm and morphological rules in such a way as to support each other (§3.5.4) and safeguard against the segmentation fallacy. Numerous problems and pitfalls will be discussed along the way, with particular reference to the necessity and difficulties of implementing multilingually formulated morphological rules, so that by the end of this section, a clear way forward to sound morphological analysis for lexical database enrichment (§5) will have been presented and an affix stripping precedence rule established (§3.5.1). Consideration is also given to the best way to encode morphological relations (§3.5.3) and the conclusion is reached that lexical relations between words should be encoded in the lexicon, separately from the semantic relations between meanings encoded in the wordnet component of the model. These lexical relations can be considered as morphosemantic in so far as morphological rules can identify the relation types.

3.1 Background

3.1.1 Some Simple Stemmers

Porter (1980) proposes a suffix stripping methodology for use in information retrieval. In a system containing a set of documents indexed by the words in their titles or abstracts, greater efficiency and economy can be attained by conflating derivationally related words carrying related meanings. The approach adopted assumes the absence of a stem dictionary but the presence of a suffix list (as in §5.2.2).

Rather than trying to discover morphological relations wherever possible, Porter is at pains to avoid conflating words which, although morphologically related, may be

semantically distant within a given domain, such as "relate" and "relativity" in physics. Porter claims that, beyond a certain point, proliferation of rules will be counterproductive, because overgeneration will outweigh valid applications of the rules (cf. §§3.2.2.2). The remainder of the article is taken up with describing how the algorithm applies generalised rules for suffix stripping. The algorithm requires considerably less code than previous attempts at the task, which it outperforms. Porter also points out that suffix stripping rules should not be applied if the stem is too short, a conclusion arrived at pragmatically, without any known linguistic basis (cf. §§3.2.2, 5.1.1).

Minnen et al. (2001) describe the development of a lemmatiser and morphological generator to handle English *inflectional* morphology. The lemmatisation task undertaken is trivial because English is so poor in inflectional morphology, but their work is analogous on a small scale to the analysis for *derivational* morphology undertaken in this thesis. Comparatives and superlatives of adjectives, which are among the few examples of inflectional morphology in English, are excluded. Their project is implemented in Flex (Levine et al., 1992), which is a high level interface for expressing rules implemented in C. Their analyser (lemmatiser) required 1400 POS-tag dependent Flex rules. The development required the incorporation of data from numerous sources including the previous GATE morphological analyzer (Cunningham al., 1996), which itself borrows from the WordNet 1.5 exception lists, which are sufficient on their own for constructing a lemmatiser (§1.3.2.5). This module in WordNet is robust and reliable and widely used as an English lemmatiser by non-native speakers who otherwise have no use for WordNet[33]. The proliferation of rules was required in order to reduce the size of the exception list to 25%, by defining rules such as "-ves" -> "-f" for noun singularisation. The generator is essentially an inversion of the analyzer. This research represents little advance on Porter (1980).

[33] feedback at the present author's seminar *La base WordNet, ses problemes et leur traitement éventuel* at the Laboratoire d'Informatique de Grenoble, Joseph Fourier University, Grenoble, 14th. May 2009.

3.1.2 A State of the Art Morphological Database?

Habash & Dorr (2003) introduce their *categorial variation* database, CatVar (http://clipdemos.umiacs.umd.edu/catvar/), which is examined in detail below (§3.1.2.1). They define a categorial variation of a word as "a derivationally related word with possibly a different part of speech" (p. 17). They assert that 98% of all divergences in the structuring of meaning between languages involve categorial variation, such that their database should be a useful tool for Machine Translation. They classify previous approaches as either *reductionist* or *analytical*, such as Porter (1980; §3.1.1) or *expansionist* or *generative*. The former approach finds root forms from complex words and the latter generates complex words from roots. The main problem of the latter approach is *overgeneration*. Previous work is criticised for overgeneration, although CatVar also overgenerates (§3.1.2.1). Habash & Dorr say almost nothing about how CatVar was created: the description is insufficient to reproduce their work, or to discover why CatVar overgenerates in some cases and undergenerates in others.

The authors describe the evaluation process, which employed not an authoritative lexicographic resource but 8 native speaker annotators, who were asked to classify the cluster members into these categories:

1. definitely belonging,
2. belonging except for POS error,
3. belonging except for spelling error,
4. uncertain,
5. wrong.

Inter-annotator agreement was 80.75%. By conflating (1), (2) and (3), 98.35% inter-annotator agreement was achieved. The results reported after combining the annotations were 68% definitely belonging, 0.01% belonging except for POS error, 0% belonging except for spelling error, < 3% uncertain and <1% wrong. This leaves at least 28% unaccounted for. There was 26% undergeneration measured by related words which the annotators could think of. The authors discount 61% of the undergeneration on the grounds that the words in question occur elsewhere in the database. It is unclear how they

conclude that they achieved 91.82% precision (cf. 90.78% calculated in §3.1.2.1; first 2 columns of Table 12). They excuse the poor performance, saying that many of the morphological connections missed could be found by the Porter (1980) stemmer (§3.1.1).

Habash & Dorr (no date) say almost nothing about the CatVar database to add to Habash & Dorr (2003), to which they refer for "a more detailed discussion and evaluation of CatVar". In neither paper is there a sufficient explanation of how CatVar was created. Again they criticise previous systems, among which they single out the Porter (1980) stemmer, for their "crude approximating" nature, a criticism more appropriately addressed to their own system, given the limited remit and relative antiquity of the Porter stemmer. They do however rightly point out the utility and importance of accurate morphosemantic data for language generation, despite their inaccurate morphology and the complete absence of semantics from their database.

3.1.2.1 Analysis of CatVar Sample Dataset

The CatVar database (http://clipdemos.umiacs.umd.edu/catvar/) is a lexical database organised as 51972 clusters of words. Each word is represented as a {word form : POS} pair, so that the same word form may occur more than once in the same cluster as a different POS. The words in each cluster are supposed to be morphologically related.

From the CatVar database a random sample was taken of 521 clusters containing at least 3 pairs each, comprising 2417 pairs altogether.

The first observation made about this dataset was that it contained unfamiliar word forms. The entire dataset was checked against the lexicon in the WordNet model. 251 word forms were not in the lexicon as the given POS. This list was compared against the Cambridge Advanced Learner's Dictionary online (http://dictionary.cambridge.org/), which also failed to find any of these words as the specified POS except for proper case forms "Buddhist", "Catholic" and "Satan". Some of the unattested word forms were active participles used as adjectives or nouns and passive participles used as adjectives.

These uses of participles are grammatically legitimate irrespective of their attestation by any lexicon. Excluding these participles there remain 174 unattested forms.

The absence of a word from any particular lexicon can never prove that a word does not exist. However, the lexicon coverage of WordNet is comprehensive compared to other lexical resources examined. Given that the objective is to find morphological relations between words already in WordNet, the extension of the lexicon with unattested word forms is outside the scope of this research project. So especially in the context of the undergeneration discussed below, from the standpoint of WordNet, the unattested words in the sample can be considered to represent an overgeneration of 7.20%. In addition some 49 words (2.02%) in the dataset are morphologically unrelated to the headwords (Appendix 8), despite superficial resemblances. This brings the total overgeneration up to 9.22% (first 2 columns of Table 12). This gives a precision of 90.78%, compared to Habash & Dorr's (2003) figure of 91.82%.

Table 12: Comparison of autogenerated Results with CatVar data
(see also §3.2.2.2.1)

Dataset	**CatVar sample dataset**	**Autogeneration from CatVar sample dataset**		**CatVar sample dataset only**	**Auto-generation only**	**Common to both**
Ruleset	**n/a**	**Full**	**Restricted**	**Full**	**Full**	**Full**
Not in lexicon	174	0	0	174	0	0
In lexicon but unrelated	49	70	0	44	65	5
In lexicon and related	2194	2432	2151	183	421	2011
Overgeneration	9.22%	2.88%	0%	n/a	n/a	n/a
Coverage	Baseline	+3.52%	-11.01%	n/a	n/a	n/a
Precision	90.78%	97.20%	100%	n/a	n/a	n/a
TOTAL	2417	2502	2151	401	486	2016

Undergeneration in CatVar is impossible to quantify, in the absence of any comparable resource, prior to the complete morphological analysis of the lexicon. Table 13 shows some related words identified but not found in the appropriate cluster. This has been compiled simply by thinking up words related to the headwords which are not found in the corresponding clusters. As such it should be considered as the minimal

undergeneration. Numerous other examples have been found through the experiments described in §3.2.2. Given the observed undergeneration in the sample data and the subsequent experimentally demonstrated undergeneration, recall can be demonstrably improved (Table 12). So we must conclude that the CatVar database is seriously incomplete.

Table 13: Undergeneration in the CatVar dataset

CatVar headword	**Missing morphological relatives**
activist	active
agreeable	agree
ammoniate	ammonia
artist	art
behaviour	behave
biologic	biology
charitable	charity
collectivise	collective, collect
cosmology	cosmologist, cosmos
demographer	demography
easterly	east
ethnographer	ethnography
facial	face
felony	felon
geology	geologist
heavy	heave
ideology	ideologue, ideologist
incidental	incident, incidence
motile	motion, move
mystify	mystery, mysterious
numeral	number
pally	pal
pantheist	pantheism
passive	pass
phonology	phonologist, phonetic, phone
quarterly	quarter
radial	radius
religious	religion
ripen	ripe

CatVar headword	Missing morphological relatives
scholastic	scholar, school
script	scribe
sensible	sense
skyward	sky
soften	soft
swim	swimmer
taxonomic	taxonomy, taxonomist
theologise	theology, theologian
traditionalism	traditional, traditionalist, tradition
vertebral	vertebra
worsen	worse

Given the overgeneration and undergeneration, the CatVar database does not appear to be a reliable or complete resource for information about morphological relations between words. It will be shown that clusters of derivationally related words have an internal structure (§3.1.4; Fig. 4, §3.2.2.2.2; Fig. 5, §3.2.2.4) which indicates which words are derived from which others. This is not elucidated by the CatVar clusters. The encoding of directionless derivational links between words which are members of CatVar clusters has already been achieved to some extent in WordNet 3.0 (§3.2.2.4). This is not the best way to represent morphological data in a lexical database. Overall, we must conclude that CatVar does not represent the best approach to morphological enrichment of a lexical database. Alternative approaches will be proposed and evaluated (§§3.2-3.4), creating confidence that a better morphologically enriched database can be produced, which will then be presented and evaluated (§§5-6).

3.1.3 Previous Work on the Morphological Enrichment of WordNet

Fellbaum & Miller (2003)[34] describe how the directionless derivational pointers which they call "morphosemantic links", the WordNet DERIV relations, came to be encoded between word senses in WordNet 2.0. This work covers only suffixations and homonyms. No attempt has been made to capture the morphological relations of prefixations, concatenations or compound expressions, except where a concatenation also exists as a corresponding compound expression punctuated by a space.

The starting point was a list of 16 derivational suffixes for nouns derived from verbs[35] and 3 for verbs derived from nouns[36]. These were obtained from literature, contrasting with the empirical approach to suffix identification adopted in this thesis (§3.4.2). There is no discussion as to whether these suffixes can simply be appended or removed or whether substitution is required (§3.2.2), and so it is unclear whether this work is limited by the segmentation fallacy (§3.3). Only a short list of exceptions was compiled.

The nouns and verbs ending with the listed suffixes were then extracted from WordNet. A list of noun-verb homonym pairs was also extracted. The resultant lists were subjected to a manual process of removing homonym pairs which the team did not consider to be related, and nouns which, in their opinion, were not derived, as expected, from verbs. In the absence of a set of morphological rules governing the behaviour of the suffixes (§3.2), it was necessary also manually to go through the lists of words exhibiting the suffixes, pairing nouns and verbs.

[34] A copy of this article was finally obtained when this thesis was almost ready to submit, and so has been reviewed retrospectively and played no part in the development of the rest of the thesis. The article makes it clear that the DERIV relations between word senses in WordNet are not based on CatVar, as it had previously appeared in the light of available circumstantial evidence.
[35] "-acy", "-age", "-al", "-ance", "-ancy", "-ant", "-ard", "-ary", "-ate", "-ation", "-ee", "-er", "-ery", "-ing", "-ion", "-ure"
[36] "-ate", "-ify", "-ize"

Much of the discussion in Fellbaum & Miller's paper concerns the problems of choosing the relevant word senses for linking, where there are multiple senses of one or both of the morphologically related words. Some reliance was placed on semantic fields encoded as WordNet semantic categories (§2.2.2.2.5), but this operation also was conducted manually by the team, a task made far more difficult and arbitrary by the fine granularity of WordNet (§2.1.2), especially in the case of verbs with abundant nominal derivatives. Just how arbitrary this process was is revealed by the examples "mothball" whose noun and verb senses were judged to be related and "shoehorn" whose senses were judged to be unrelated. The level of inter-annotator agreement is not discussed. Fellbaum & Miller take the view that this assignation of derivational links to word senses is necessary, that it cannot be achieved by a rule-based approach and that the manual procedure described can make "all and only the appropriate sense distinctions" (p. 77). Avoiding this kind of arbitrary approach was a major reason for the decision made for the purposes of this thesis, to encode derivational morphology as holding between words in the lexicon, rather than between word senses in WordNet (§3.5.3).

It is not surprising that the WordNet set of derivational pointers is incomplete, given the limited number of suffixes considered and the failure to tackle concatenations and prefixations. Fellbaum & Miller conclude that their work is a step towards addressing the problems which morphosemantic relations pose for automatic systems. It is difficult to concur, when their work has been conducted almost entirely by a manual approach, involving a large number of undocumented, arbitrary decisions, consistent with those made in the original design of WordNet, in as far as it has been possible to elucidate these (§2).

No attempt has been made to encode the direction of derivation. Although one must acknowledge that establishing the direction of derivation between homonyms is difficult (WordNet's own frequency data can be used for this; §5.3.6), it should still be possible to encode the direction of derivation from roots to suffixations. Despite the use of the term "morphosemantic links", no attempt has been made to identify the semantic relation types of the relations encoded.

Fellbaum et al. (2007) acknowledge that the derivational pointers are not semantic but purely morphological. They state, questionably, in their introduction, that "English derivationally (*sic*) morphology is highly regular", and acknowledge that they assumed, at the time when the morphological relations were introduced, that there was "a one-to-one mapping between affix forms and their meanings", an assumption which they take to be widespread. However they have undertaken some laborious research to discover the falsity of the assumption, which is largely what their paper describes.

In particular, with reference to the derivation of nouns from verbs by appending the suffixes "-er" and "-or", they "assumed that, with rare exceptions, the nouns denote the *agents* of the event referred to by the verb". They provide a table of their findings, which is incorporated into the first two columns of Table 14, which show that less than two thirds of *their* examples are of *agents*. It is notable that of the few examples for which they actually provide details, many are American usages, especially those categorised as *undergoer, cause, result* and *purpose*.

Table 14: Semantic and syntactic roles of the "-er" suffix

Semantic role according to Fellbaum et al. (2007)	**Occurrences found by Fellbaum et al. (2007)**	**Equivalent Syntactic role**	**Subject instances**
Agent	2584	Subject	2584
Instrument	482	Subject	482
Inanimate agent / Cause	302	Subject	302
Event	224	Gerund	
Result	97	*No valid example*	
Undergoer	62	Subject	62
Body part	49	Subject	49
Purpose	57	Locative	
Vehicle	36	Subject	36
Location	36	Locative	
TOTAL	**3929**		**3515**
Agent/TOTAL	**65.77%**		
Remainder/TOTAL	34.23%		
Subject/TOTAL			**89.46%**
Remainder/TOTAL			10.54%

Vincze et al. (2008) observe that derivational relations encoded in WordNet can often translate as syntactic functions, typically involving a part of speech transformation. Almost 9/10 of the categories to which Fellbaum et al. (2007) assign their examples conform to the syntactic role of subject (Table 14) in traditional grammar. The "-er" suffix, then, represents not a *semantic* relation (as understood in *Frame Semantics* (Fillmore, 1968; Ruppenhofer et al., 2006) but a *syntactic* one, which does, outside the conceptual constraints of Frame Semantics, have some semantic import. It is true to say that a *printer prints*, irrespective of whether the printer is a person or a tool. This *syntactic* role subsumes most of the different *thematic* roles identified for the suffix. In the morphological ruleset introduced in §3.2.2, it is simply assigned SUBJECT as its relation type (Appendix 10).

Bosch et al. (2008) seek to enrich WordNet with morphological relations on the grounds that wordnets are more useful when the network is dense. They propose the formulation of morphological rules to allow the automatic encoding of such relations (§3.2) but do not describe any implementation. They acknowledge the overgeneration risk where morphological rules generate words which do not occur but not the risk of identifying false derivational relations (§3.2.2.2). They observe that overgeneration can be addressed by automatic cross reference to a lexical resource such as a dictionary or corpus, but that manual checking is needed to detect undergeneration. They suggest that overgeneration may require the reformulation of the rules in such a way as not to overgenerate (§§3.2.3, 5.1), and realise that there is no 1-to-1 mapping from morphology to semantics as Fellbaum et al. (2007) had hoped, but that in some cases the same word form is polysemous with respect to different semantic roles. Likewise a single semantic relation can be represented by more than one affix.

The main conclusions to be drawn here, beyond the insufficiency of the existing WordNet derivational pointers, are that the imposition of linguistic theories, even theories as widely accepted as frame semantics, is not necessarily helpful to the understanding of morphological relations, and that theory is no substitute for empirical

evidence, especially in the linguistic domain where no theory has yet comprehensively explained observable phenomena. It is a mistake to attempt to map directly from morphology to semantics without passing by the more rigorously and robustly defined domain of syntax, which will be represented in this thesis by the frequent adoption of syntactic relation types for relations between suffixations and their morphological roots (§3.2; Appendix 22).

3.1.4 Derivational Trees

Mbame (2008) proposes a *Morphodynamic Wordnet*, which connects morphologically related words and multiword expressions in a way which captures extensions to meaning, inclusive of metaphors. He defines the morphogenesis of semantic forms as the generation of senses from a semantic nucleus represented by a lexical root. This is illustrated with numerous derivatives of the root "trench" in a number of different semantic domains. These can be mapped into a *derivational tree* structure rooted at "trench"[37].

This representation is superior to the *cluster* representation (§3.1.2), in that it shows clearly that there is always a root form among a set of morphologically related forms (a set *all* of whose members are morphologically related to *all* other members), and that there is always a derivational hierarchy, with each form being derived from one parent (within the tree). This hierarchy corresponds to the historic evolution of forms from each other which is a progressive enrichment of language through time. This clearly does not rule out dual inheritance of concatenations: the word "trenchcoat" is derived from "trench" and from "coat" and thus is a member of 2 of the interlocking derivational trees of which a morphodynamic wordnet would be composed.

[37] In discussions with Nazaire Mbame (Clermont-Ferrand, May 2009), agreement was reached that the structure might not always be a tree, but might be a bush. This is equivalent to an acyclic directed graph.

To produce detailed derivational trees of the kind illustrated by Mbame requires a great deal of painstaking lexicographic and historical research[38] which is outside the scope of a computational project, but the tree structure is an informative and computationally tractable way to represent sets of morphologically related words. CatVar clusters would be better represented in such a way. The corresponding derivational tree representations of the clusters could be determined by identifying the morphological rules governing the derivation within the clusters.

A morphodynamic wordnet does not require any underlying semantic wordnet. It can be constructed using only a lexicon as a starting point. This construction can be achieved by a combination of the application of morphological rules (§3.2) and algorithms to discover morphological phenomena (§3.4) in the same way as the morphologically enriched lexicon whose development is described in §5. The only structural difference between the morphosemantic wordnet as produced by this project and the morphodynamic wordnet proposed by Mbame is the inclusion of the underlying semantic wordnet from which the lexicon was derived.

3.1.5 Morphological Enrichment across Languages

Bilgin et al. (2004) take the view that enriching wordnets with morphosemantic links will enhance their functionality. They assert that the use of morphology to discover semantic relations is the best way to create a wordnet or to enrich an existing wordnet. They make the further innovative suggestion that *morphosemantic* relations discovered in one language can be exported as *semantic* relations into another language. For example, the Turkish verbs "yikmak" and "yikilmak" are related by a regular morphological rule which represents a causative relation between them. Their English equivalents are "tear down" and "collapse", which are clearly not morphologically related, but the same causative relation holds between them. Thus the Turkish morphological relation could be used to enrich an English wordnet. The authors point out however that morphological relations hold between word *forms* and not word *senses.* It is a lexicographic task to identify the

[38] an enormous task with a lexical database the size of WordNet.

correct synset in the target wordnet, for each of the related words, whether or not it is in the same language as the morphological relation. They also point out that the same affix can be used to represent more than one semantic relation on its stem (cf. §3.1.3). Experiments with the Turkish causal affix were highly productive in generating causal relations missing from WordNet. An adequate morphologically enriched lexical database for the source language is a prerequisite for the systematic application of this interesting approach.

Koeva et al. (2008) suggest that Slavic languages are much richer in such regular morphological relations than English, and as such are a suitable source for exporting discovered semantic relations, as suggested by Bilgin et al. (2004). They see a need for more theoretical investigation in order to classify the mapping from derivational to semantic relations. Although Slavic languages are rich in regular morphological variants, they say that the regularity is limited, and too much automation risks overgeneration of non-existent word forms (cf. §3.2.2.2). Moreover a word form derived by a regular morphological transformation from its root, corresponding to a regular semantic transformation, may subsequently acquire meaning extensions or exploitations (§2.1.1) which are not paralleled by other words derived according to the same rule.

3.1.6 Inference of Morphological Relations from a Dictionary

Hathout (2008) seeks to discover the morphological structure of the lexicon from morphological similarities between words and analogies derived from morphological analysis of the words in the glosses of the online dictionary *Trésor de la Langue Française* (http://atilf.atilf.fr/). The methodology is strictly graph-based. This approach to morphology dispenses with the concepts of morpheme and affix and considers every possible *n*-gram of characters >= 3-gram which can be extracted from each word. It allows not only the discovery of morphologically related word pairs, but also the calculation of morphological resemblance as the reciprocal of the graph distance between them. It is thus a fully empirical approach, not influenced by linguistic theory: no special status is conferred upon any of the *n*-grams. Complex relationships between sets of words

as well as individual words are drawn out from the dictionary definitions. The success of his approach suggests that the definitions in the Trésor de la Langue Française are more consistent than those in WordNet. Hathout provides evidence that formal features are more reliable than semantic ones in predicting meaningful morphological relations.

Hathout infers morphological relations partly from semantic relations, the reverse of what is attempted with morphological rules in this thesis (§§3.2, 5.1). But it is similar to automatic affix generation (§3.4) in that the *n*-grams used are entirely automatically generated.

3.2 A Rule-based Approach

After summarising the requirements for the morphological enrichment of a lexical database by a rule-based approach, and the limitations of the morphological data already encoded in WordNet and in CatVar, this section describes a pilot study which formulates morphological rules from a sample of the CatVar data, applies the rules, as far as possible, algorithmically, and evaluates their performance at suffixation and suffix stripping tasks. The formulation of some of the rules required to capture the morphological relationships exhibited by the sample data involves the morphology of ancestor languages of English. Some such *multilingually formulated rules* cannot be applied within a monolingual database, while others can be applied without reference to the ancestor languages. In either case, their non-application or monolingual application has a decisive and detrimental effect on the results, by way of undergeneration and overgeneration respectively.

3.2.1 Requirements for the Morphological Enrichment of WordNet

There are several prerequisites for the enrichment of a lexical database with relations based on derivational morphology. First of all the morphological relations need to be

identified. Any automated process risks *overgeneration* and *undergeneration*. Both will be illustrated by examples from the CatVar database (Habash & Dorr, 2003). To avoid these pitfalls requires more rigour than has been applied in the creation of that database (§3.1.2). The necessary rigour can be applied by formulating well informed morphological rules (§§3.2.2.1, 5.1.2). If affixed and non-affixed forms, either of which can be generated from the other by the application of a well informed rule, both occur in the lexicon, then a morphological relation is more likely to exist between them, but if the rule is ill informed, then the resemblance between the two forms is more likely to be co-incidental (§3.2.2.2). Having generated possible affixed or de-affixed word forms from an input word form, it is a simple matter to identify which of the word forms generated exist within a lexicon. Morphological relations discovered can then be encoded between related words, subject to verification of their validity.

Morphological relations have already been encoded, to a limited extent, in WordNet, as derivational pointers. There is no doubt that far more of these could be encoded. Unfortunately WordNet derivational pointers do not provide information about which of the two words they connect is derived from the other (§3.1.3) and so cannot be used to construct derivational trees (§3.1.4), nor do they provide any information about the semantic or syntactic import of the derivational relationship: they serve only to indicate that a relation exists but say nothing about what that relation means. More information is required before any kind of semantic inference can be made from the existence of such a relation. It would clearly be advantageous if morphological relations could be translated as semantic relations (Bilgin et al., 2004; Koeva et al., 2008). A morphological rule can be formulated as a transformation from one set of word forms to another. In order to employ it as a *semantic* tool it needs to be more fully formulated so as to define a transformation of *meaning*, which is a *semantic relation* (Bilgin et al., 2004; Bosch et al., 2008). While some morphological transformations may represent a single semantic relation, others may represent more than one (§3.1.5).

Because WordNet frequently assigns the same word form to multiple synsets, representing multiple meanings, it is not straightforward to decide where to position

pointers representing newly discovered derivational relations. It is widely agreed (Peters et al., 1998; Vossen, 2000; EU, 2004) that the hair-splitting distinctions between WordNet senses is excessive (§2.1.2). Moreover WordNet does not distinguish between homonymy and polysemy (Apresjan, 1973; Pustejovsky, 1991). The vast choice of positions for semantic pointers stands as an impediment to the automation of the enrichment process.

One approach, which would make this problem more tractable, would be to coarsen the grain, reducing the number of synsets by clustering them (Peters et al., 1998; Vossen, 2000; §2.1.2.3). This would reduce the number of choices in where to place the derivational pointers. Even within a clustered wordnet, there will still be choices to be made about where to position new pointers, but the fewer the number of synsets, the more often those pointers will have a unique candidate position and so the more the encoding of them can be automated. An alternative approach, which circumvents the problem of polysemy, is to encode derivational pointers within the lexicon rather than within the WordNet model itself. This issue is taken up in §3.5.3.

Once a morphological rule has been validated *lexically,* through examination of the output it generates, establishing that the word forms it connects are indeed related, it ideally needs also to be validated *semantically,* to establish that the relations between word forms generated by the rule match the semantic relation defined for the rule, where a unique semantic relation can be defined for all applications of the rule. For practical purposes it may need to be inferred that, where the semantic relation matches in a sufficiently large sample, it can be applied universally. However if the instances where the morphological transformation encapsulated in the rule is applicable represent more than one semantic relation, the possible *semantic* relations will need to be generalised as a single *syntactic* relation (§3.1.3), or, failing that, as a generic *morphological relation*, specifying only the direction of the derivation (§3.1.4).

3.2.2 Pilot Study on the Formulation and Application of Morphological Rules

This section discusses a pilot study to formulate rules from a limited sample from the CatVar database, after detailed examination and removal of the overgenerations. The study proceeds to the algorithmic application of the rules discovered and *lexical* validation of their performance[39] when applied to two datasets. The problems associated with multilingually formulated rules are highlighted.

3.2.2.1 Formulation of Morphological Rules from the CatVar Dataset

The CatVar sample dataset reviewed in §3.1.2.1, was revised by removing the overgenerated word forms. From painstaking linguistic analysis of the revised dataset, a set of morphological rules was manually formulated to encapsulate the morphological and semantic transformations involved (Appendix 9). The morphological transformations exhibited by the dataset were almost entirely examples of suffixation. There were only 2 examples of prefixation, namely "bespectacled" and "embranchment" and a few examples of abbreviation. There were sufficient examples of suffixation, and of identical word forms being used as different POSes, for rules to be formulated.

Many of the suffixed forms found in the CatVar dataset are in fact active and passive participles used as adjectives and gerunds. Because passive participles are frequently irregular in English, the use of an exception map is required. The exception map encapsulated in the lemmatiser (§1.3.2.5) is suitable for suffix stripping, but for applying suffixes to roots a reversed exception map is generated from it, in which the keys are irregular verbs and the values are their passive participles. Active participles are always regular in English, subject to general suffixation rules. Given the exceptions, the rules for participle formation (which is really *inflectional* rather than *derivational* morphology)

[39] Semantic validation will be left for future research.

have to be considered as *conditional* rules, while the remainder of the suffixation rules have been treated as *unconditional* (see also §5.1.1).

The verbosity of many of the rules (Appendix 9) is an indicator of the level of precision needed to ensure that the rules are as well-informed as possible. The rules have generally been formulated using the verb "may", indicating that they apply in some but not all cases. Any assumption to the contrary would result in gross overgeneration. In applying the rules, the lexicon derived from WordNet has been employed to validate all word forms generated.

To correctly determine the rules governing suffixation in English, it is essential to understand the hybrid nature of the language, which means that different rules apply depending on the etymological history of the words. This is further complicated by the fact that some words of Latin origin[40] have come into the English language directly while others have come indirectly through Anglo-Norman. For simplicity, in the course of this study and within the rules themselves, the Anglo-Norman dialect has been referred to simply as "French". Many English words are derived from Latin participles, especially passive participles, which are frequently irregular in Latin. Consequently the morphological rules for the formation of these words cannot be specified without reference to Latin grammar. The same principle applies to words derived from the genitive case of Latin nouns. Where English words are derived from the active participles of verbs of Latin origin, there is the further complication, that whereas Latin active participles have a nominative ending "-ans" or "-ens" (genitive "-antis" or "-entis") from which we get English adjectives in "-ant" or "-ent", French active participles always end in "-ant", resulting in English adjectives in "-ant" even when one would expect "-ent" from the Latin origin.

Some of the rules which refer to languages other than English have been formulated in such a way that a transformation from one English word form to another can be applied

[40] Suffixations of Anglo-Saxon origin, unlike those of Latin origin, are generally formed by simply appending a suffix to a stem, as with adjectival suffixes "-some", "-ful" and "-less", nominal suffixes "-er", "-ness" and "-ship", verbal suffix "-en" and adverbial suffix "-ly" (Appendix 10).

(the reliability of this procedure is investigated in §3.2.2.2), while others cannot be applied without reference to lexical resources pertaining to the other languages (italicised in Appendix 9).

The morphological rules as presented in Appendix 9 are preceded by some generalised spelling rules for the application of suffixes to and removal of suffixes from words to generate other words. The spelling rules apply to those morphological rules which involve the addition or removal of suffixes, but are redundant for those morphological rules which specify substitutions of one suffix for another.

A few morphological rules have been formulated to govern POS transformations between identical word forms, but particularly in the case of nouns and verbs, the semantic relations involved are too diverse to be specified. In these cases, automatic generation may be possible and automatic identification of morphological relations may also be possible, but automatic semantic interpretation of these morphological relations is not realistic. The greater bulk of the ruleset comprises rules governing morphological transformations associated with POS transformations, usually with discernable semantic significance, but there are some rules which govern transformations where the POS remains the same, but which still possess semantic significance.

In order to use the morphological rules computationally, they clearly need to be represented in a computationally tractable form. In Appendix 10, each rule is tabulated in such a way that it can be applied to automatic generation of suffixes, suffix stripping or semantic relation identification, from the morphological relations expressed by the rules. The first four fields were defined initially as for suffixation, where the *source* fields apply to the input word form and the *target* fields apply to the output. The first source field *morpheme to remove* will be empty where a suffix can simply be appended according to the generalised spelling rules, otherwise a substitution rule will apply. The first target field *morpheme to append* contains the applicable suffix. For a suffixation, each rule will be applied only to a word which ends with the character combination in the *morpheme to remove* field, unless that field is empty. There are also source and target POS fields. A

rule will only be applied where the source POS matches the input. The target POS will be associated with the output. A suffix stripping application[41] needs to swap the source and target fields to create *converse* morphological rules (§3.2.2.2.2).

In order to capture the semantics associated with the rules, a *relation* field represents the semantic or syntactic transformation associated with each morphological transformation, expressing the type of relation which applies from source to target. Long but transparent names have been chosen for the relation types (Appendix 22) in preference to coining an entirely new terminology. Where the corresponding relation type exists in WordNet, the WordNet name has been used. The new relation types proposed are tentative and further research is required to confirm the extent of their applicability. In the analysis described in §5, they are implemented as a field of class `MorphologicalRule` (§5.1.1) specifying the `Relation.Type` of the relations discovered through the application of morphological rules. Because the types are tentative, they played no part in the implementation discussed in §3.2.2.2 and are not used for WSD in the evaluation presented in §6. A suffix stripping application needs also to specify the converses of the semantic relation types (Appendix 22), for the *converse* morphological rules (§3.2.2.2.2).

The following examples illustrate the transformations involved (cf. Table 15).

Original formulation 1 (*substitution; generalised spelling rules not applicable*):

> If a verb ends in "-ate", there may be a corresponding adjective ending in "-ative", whose meaning corresponds to the adjectival use of the active participle. (*monolingual rule; example:* "accumulate" : "accumulative")

Original formulation 2 (*no substitution: generalised spelling rules applicable*):

> If a verb is derived from French, then there may be an adjective formed by appending the suffix "-ant". The meaning of the adjective corresponds to the adjectival use of the active participle. (*multilingual rule applied monolingually; example:* "depend" : " dependant")

[41] as in suffixation analysis by the morphological analyser (§5.3.7).

Table 15: Computational representation of morphological rules

Rule				**Relation**
Source		**Target**		
Morpheme to remove	**POS**	**Morpheme to append**	**POS**	
ate	VERB	ative	ADJECTIVE	Participle[42]
	VERB	ant	ADJECTIVE	Participle

The majority of the semantic relations exhibited by the meanings of the morphological transformations have no equivalent in WordNet. WordNet could be enormously enriched by the addition of the semantic relation types proposed in Appendix 10, and their encoding where they are morphologically indicated. Table 16 shows which relation types exist in WordNet and how many rules[43] indicate each relation type, for those types shared by 2 or more rules.

The most important new relation type discovered holds between a verb and its gerund or a word with the same meaning as its gerund (§1.1.4). The extensive set of nouns ending in "-ion" generally carry the same meaning as an active gerund though sometimes they carry the same meaning as a passive gerund. In this thesis, such words are termed *quasi-gerunds*. From the data from automatic suffix discovery (§3.4.2), we know that some 84.72% of these words end in "-tion", and of those, 78.18% end in "-ation" (for possible applications see §7.4.1). Despite their usually active meaning these quasi-gerunds are derived from the Latin passive participle, where a corresponding Latin verb exists. Where no Latin verb exists, they are most usually generated by appending the suffix "-ation". Because Latin passive participles are frequently irregular, the morphological relationships between the English quasi-gerunds and their corresponding verbs are even more irregular. The formulation of morphological rules to govern their formation in English was too complex to be undertaken within the pilot study. A large number of morphological rules are required to govern their formation in English, without reference to Latin (§5.1.2)..

[42] meaning that the target is used as an adjective with the same meaning as the active participle, the suffix "-ant" being derived from a Latin or French active participle.
[43] in the original ruleset.

Table 16: Rules per relation (original ruleset)

Relation	No. of rules	WordNet relation
Pertainym	23	Pertainym
Gerund	18	None
Participle	18	Participle
ChacterisedBy	16	None
Indeterminate	11	n/a
StateOfBeing	12	None
Believer/practioner	9	None
Synonym	8	Synonym
Make	7	Cause
NearSynonym	7	None
Qualified	6	None
Result	6	None
Subject	5	None
Belief/practice	4	None
Having	4	None
Potential	4	None
Object	3	None

3.2.2.2 Application of Morphological Rules

3.2.2.2.1 Autogeneration of Suffixed Forms

The morphological rules are implemented using class `POSTaggedMorpheme` and its subclasses `POSTaggedSuffix`, and `POSTaggedWord` (which requires lexicon validation[44]; Appendix 1; Class Diagram 8)[45]. Each rule is defined in terms of a transformation between one `POSTaggedSuffix` (the *source*) and another (the *target*). In order to apply the rules and test their performance, a Suffixation Algorithm was developed to apply any morphological rule to any word to which it is applicable. The Suffixation Algorithm inputs a `POSTaggedWord` and the source and target of a rule, and outputs a `POSTaggedWord` array comprising 0, 1 or 2 elements. No output is generated unless the

[44] `CatVarTuple` is a subclass of `POSTaggedWord` which carries information about its WordNet relations.
[45] later adaptation in Class Diagram 11.

POS of the input `POSTaggedWord` matches that of the source. Where the suffix form fields of each `POSTaggedSuffix` are empty, no morphological change applies but only a part of speech change; where the suffix form field of the source is empty and that of the target is non-empty, the target suffix form is appended to the input `POSTaggedWord`, subject to general spelling rules, to generate a maximum of 2 alternative output words; where both suffix form fields are non-empty, the rule only applies to an input whose word form ends with suffix form of the source, which is replaced with that of the target, without reference to general rules.

The algorithm exploits the lexicon in the WordNet model (§1.3.2.4) for validation[46]; the irregular inflection data derived from the WordNet exception files (§1.3.2.5; Fig. 3) is also checked in the case of conditional rules. As the WordNet model does not have access to non-English data, those rules whose formulation refers to other languages[47] could not be applied (§§3.2.2.1, 5.1.2). Where rules which refer to non-English data could be rephrased without reference to that data, the rules were applied accordingly, though consequent false generations were anticipated.

Suffixation Algorithm[48]

NB:

1. *"y" is treated as a vowel;*
2. `apply morphological rule` *outputs 0, 1 or 2 suffixations from the input word;*
3. *Parameter* `word` *is a* `POSTaggedWord` *representing the input word;*
4. *Parameter* `source` *is a* `POSTaggedSuffix`*;*
5. *Parameter* `target` *is a* `POSTaggedSuffix`*.*

```
apply morphological rule(word, source, target, lexicon, output)
{
      if (source.POS == word.POS)
```

[46] The `POSTaggedWord` constructor invokes the required lookup and sets or clears a Boolean validity field.
[47] wholly in Italics in Appendices 17-18.
[48] private methods of class `Suffixer`.

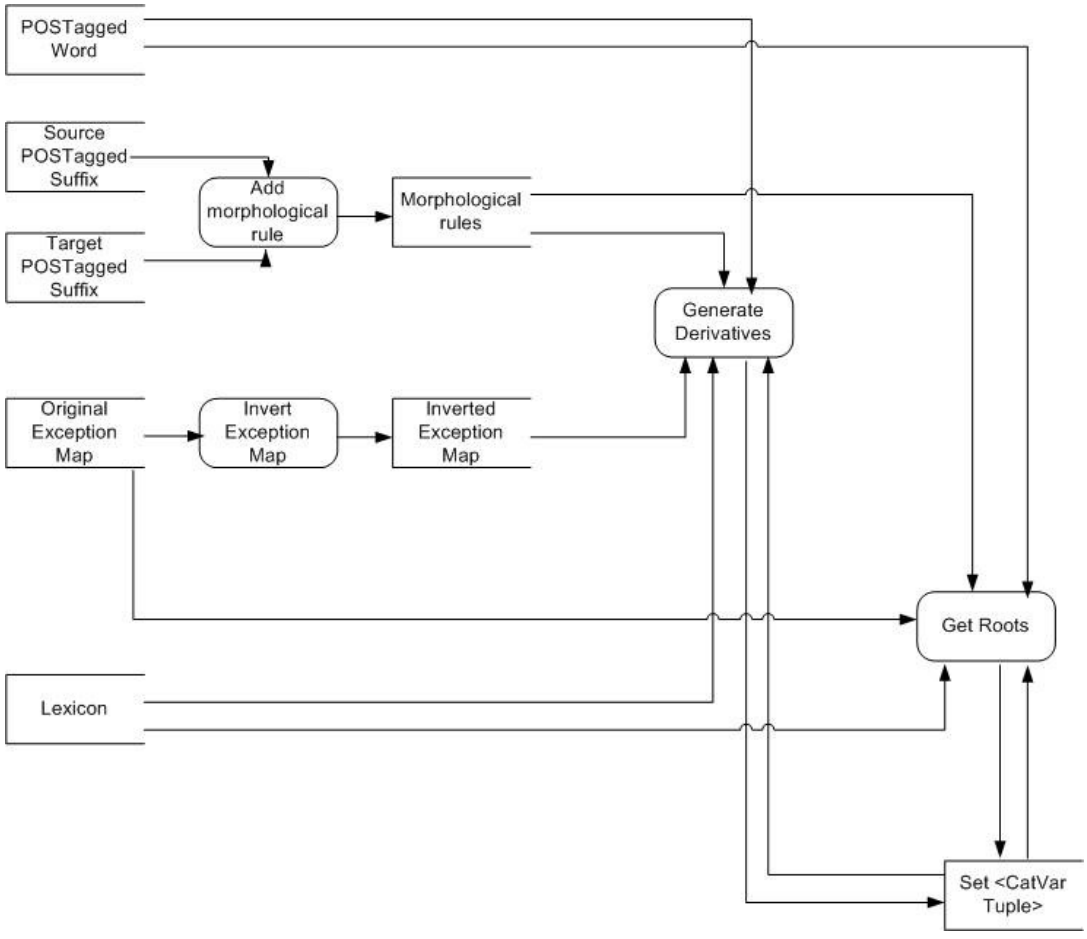
```
	{
		if (source.wordForm equals(""))
		{
			new_wordForms = append
			(word.wordForm, target.wordForm);
			for each wordForm in new_wordForms)
			{
				new_Word = new POSTaggedWord
				(new_wordForm, target.POS, lexicon);
				if (new_Word valid)
				{
					add new_Word to output;
				}
			}
		}
		else
		{
			new_wordForm = substitute
			(word.wordForm, source.wordForm, target.wordForm);
			new_Word = new POSTaggedWord
			(new_wordForm, target.POS, lexicon);
			if (new_Word valid)
			{
				add new_Word to output;
			}
		}
	}
}

append(stem, suffix)
{
	if (suffix.length > 0)
	{
		if (first letter of suffix is a vowel)
		{
			if
			(penultimate letter of stem is a vowel)
			AND
```

```
(stem does not end with "w", x" "er" "or" or "om"))
AND
(last letter of stem is a consonant)
AND
      ((stem.length == 2)
      OR
      (letter preceding penultimate letter of stem
      is a consonant)
      OR
            ((stem.length >= 4)
            AND
            (letter preceding penultimate letter of
            stem is "u" preceded by "q")
{
      if (stem is monosyllabic)
      {
            double the terminal consonant of the
            stem;
      }
      else
      {
            output[0] = stem with terminal
            consonant doubled + suffix;
            output[1] = stem + suffix;
            return output;
      }
}
else if (suffix starts with("i"))
{
      if (stem ends with "ie")
      {
            replace terminal "ie" of stem with "y";
      }
      else if
      ((stem ends with "e")
      AND
      (penultimate letter of stem is a consonant or
      "u"))
```

```
            {
                  remove terminal "e" from stem;
            }
      }
      else if
      ((stem ends with "y" )
      AND
      (penultimate letter of stem is a consonant))
      {
            replace terminal "y" of stem with "i";
      }
      else if
      ((stem ends with "e")
      AND
            ((suffix starts with("e"))
            OR
            (penultimate letter of stem is a consonant or
            "u")
      {
            remove terminal "e" from stem;
      }
}
else
{
      if (stem ends with "e")
      {
                  output[0] = stem with terminal "e"
                  removed + suffix;
                  output[1] = stem + suffix;
                  return output;
      }
      if
      ((stem ends with "y" )
      AND
      (stem is not monosyllabic)
      AND
      (penultimate letter of stem is a consonant))
      {
```

```
                        replace terminal "y" of stem with "i";
                    }
                }
        }
        output = stem + suffix;
        return output;
}
```

Fig. 3: Process diagram for morphological rule application



Comparison of Autogenerated Results from Suffixation Generation with CatVar data

In order to produce a dataset which could be compared with the CatVar dataset, the Suffixation Algorithm was applied with every rule in turn to one or more seed words

from each CatVar cluster in the sample dataset. The suffixations generated were recycled as input until no more lexically valid suffixations were generated. Since the headwords of the CatVar clusters are sometimes not the root forms, the shortest word in each cluster was used as a seed. Where there is more than one shortest word (or the same word form as different POSes), all of these shortest words have been used as seeds.

The autogenerated dataset resulting from applying the rules comprised 2502 words, compared to 2417 in the CatVar dataset. (Both datasets include the same seed words.) However the performance of the autogeneration was clearly better when overgeneration is taken into account, since all the words in the latter were validated against the lexicon.

While the CatVar dataset includes 174 words other than participles which are not attested in WordNet and a further 49 morphologically unrelated words, the autogenerated set contained no unattested words but 70 unrelated words (Table 12, §3.1.2.1). The autogenerated set contained 2432 valid morphologically related words compared to 2194 in the CatVar dataset. A complete list of unrelated words in the autogenerated set is in Appendix 11. Altogether 486 words were generated which were not in the CatVar dataset, of which 421 were morphologically related to the seed word, leaving 65 unrelated[49]. A further 5 unrelated words are found in both datasets.

Among the autogenerated set, most of the words unrelated to their seed word were generated from another unrelated word, so that within any cluster, one error could cause further consequential errors, for instance "moral" was incorrectly generated from "more" and led to 10 consequent overgenerations such as "moralise" and "morality". Altogether 25 initial errors led to a further 45 consequential errors. 21 rules overgenerated of which 15 overgenerated more than once.

183 related words found in the CatVar dataset were not autogenerated. Table 17 explains the causes of this undergeneration: 28 plurals in "-s" were outside the scope of the rules;

[49] These were generated correctly, inasmuch as they conform to the rules, but incorrectly, in that the morphological resemblance is coincidental.

20 undergenerations arose from non-implementation of rules requiring reference to Latin passive participles: implementing these rules is the most important single improvement that could be made to the ruleset (§5.1.2).

Table 17: Main causes of undergeneration

Cause	Clusters affected
Plural	28
Latin passive participle	20
No consistent rule for suffix	15
POS incompatible with rule	6
Root not in CatVar	5
Unidentified cause	4
Requires de-prefixation	4
Irregularity of Latin origin	3
Irregular spelling	3
Latin genitive	2
Latin active participle	2
Derivative not in lexicon	2

11 forms were not generated because no consistent rule could be found for the application of the "-e" suffix[50]; suffixes "-ure" and "-arian", were also not implemented because insufficient data had been collected to establish consistent rules for their application; 6 words were not generated because the rule required a different POS for either source or target; 5 root forms including "biology" and "vertebra" are missing from the CatVar dataset and consequently their derivatives were not generated.

Restricted ruleset application

In order to eliminate all overgeneration, the 21 rules which overgenerated were removed from the ruleset and the experiment was repeated. As expected, the effect was the complete elimination of morphologically unrelated words. However, the removal of the overgenerating rules resulted in 190 words in the CatVar dataset were no longer represented. Of these only 3 were morphologically unrelated. The number of words generated was reduced from 2502 to 2151 (Table 12).

[50] most typically, an Anglo-American spelling divergence, e. g. "iodin" : "iodine".

Productivity of morphological rules

The productivity of the rules was measured by counting rule executions, where execution produces lexically valid, but not necessarily morphologically related output. Appendix 12 shows the productivity of all the rules. Some of the most productive rules are prone to overgeneration. With the restricted ruleset, because the outputs from the rules which had been suppressed were not available for recycling, there were some changes to the relative productivity of the rules.

Where the ratio of overgeneration to productivity is greater than 0.5, the rule is generating more wrong data than right data. Of 7 such rules, 3 were formulated multilingually but applied monolingually (§3.2.2.1). Monolingual applications of multilingually formulated rules are 6 times more likely to generate more wrong than right data than rules which are formulated monolingually. Correct multilingual application of these rules would yield a significant improvement in performance (for the solution see §5.1.2).

Application of morphological rules to a random word list

In order apply a more objective test for the validity of the morphological rules, they were applied to a sample of words in the lexicon. Because the applicability of the ruleset might vary according to word length, random word lists were generated of each word length from 4 to 14 characters. The lists were then concatenated to form a word list comprising 1012 word forms. The complete ruleset was applied to all of these words. A further 787 words were generated of which 19 (Table 18) were unrelated to the seed word as follows:

brae: braless (adj.)
comb: combative (adj.), combatively (adv.), combativeness (n.)
hack: hackee (n.)
made: made (n.) madly (adv.), madness (n.)
mint: mince (n.)

past: pasted (adj.)

ware: warily (adv.), wariness (n.), warship (n.), wary (adj.)

parch: parchment (n.)

decree: decrement (n.)

supply: suppliant (n.), suppliant (adj.)

literal: literate (adj.)[51]

Table 18: Performance on suffixation and suffix stripping with word list

	Word list	**Suffixation**	**Suffix stripping**	
Ruleset	**n/a**	**Full**	**Full**	**Restricted**
In lexicon but unrelated	n/a	19	39	14
In lexicon and related	n/a	768	887	729
Wordforms generated	1012	787	926	743
Coverage	Baseline	+77.77%	+91.50%	+73.41%
Precision	n/a	97.59%	95.78%	98.11%
Overgeneration	n/a	2.41%	4.21%	1.88%
TOTAL	1012	1799	1938	1755

Table 19: Worst overgenerating rules with word list dataset

Source		Target		Overgenerations per rule execution
Wordform	POS	Wordform	POS	
	VERB	ative	ADJECTIVE	3.00
	VERB	ed	NOUN	1.00
al	ADJECTIVE	ate	ADJECTIVE	1.00
e	NOUN	y	ADJECTIVE	0.75
	VERB	ant	ADJECTIVE	0.67
	VERB	ee	NOUN	0.50
	VERB	ment	NOUN	0.29
nt	ADJECTIVE	nce	NOUN	0.25

The rules arranged by productivity on this dataset will be found in Appendix 13. Table 19 shows the rules which most seriously overgenerated with this dataset, with the ratio of overgeneration to productivity. Of the rules which produced a ratio >= 0.5, only 1 was formulated monolingually ("-ed" suffix in Table 19; cf. italicisations in Appendix 9).

[51] not related in OED1.

3.2.2.2.2 Suffix Stripping

Because the word list dataset contains words of up to 14 characters, it is suitable for experimenting with suffix stripping. The general suffixation rules were adapted as suffix stripping rules, similar to Porter (1980; §3.1.1), though derived independently. The Suffix Stripping Algorithm employed was essentially the inverse of the Suffixation Algorithm in §3.2.2.2.1 and is a slightly more primitive version of the algorithm described in detail in §5.2.2.3 and Appendix 14.

Suffix Stripping Algorithm[52]

NB:

1. *"y" is treated as a vowel;*
2. `apply converse morphological rule` *outputs 0, 1 or 2 words from the input suffixation;*
3. *Parameter* `suffixation` *is a* `POSTaggedWord` *representing the input word;*
4. *Parameter* `source` *is a* `POSTaggedSuffix`*;*
5. *Parameter* `target` *is a* `POSTaggedSuffix`*.*

```
apply converse morphological rule(suffixation, source, target, lexicon,
output)
{
      if (source.POS == word.POS)
      {
            if (target.wordForm equals(""))
            {
                  new_wordForms = remove
                  suffixation.wordForm, source.wordForm);
                  for each wordForm in new_wordForms
                  {
                        new_Word = new POSTaggedWord
```

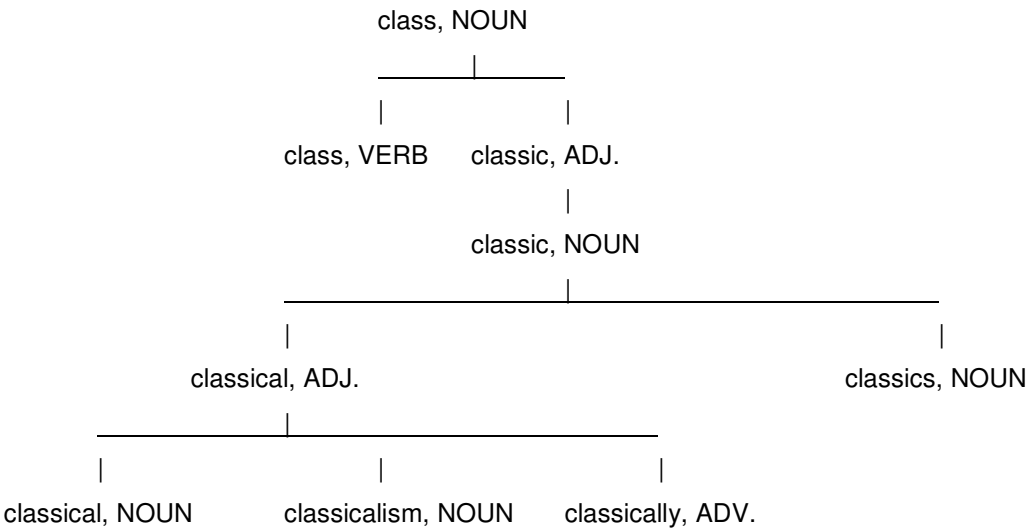
[52] private methods of class `Suffixer`.

```
                        (new_wordForm, target.POS, lexicon);
                        if (new_Word valid)
                        {
                              add new_Word to output;
                        }
                  }
            }
            else
            {
                  new_wordForm = substitute
                  (suffixation.wordForm, source.wordForm,
                  target.wordForm);
                  new_Word = new POSTaggedWord
                  (new_wordForm, target.POS, lexicon);
                  if (new_Word valid)
                  {
                        add new_Word to output;
                  }
            }
      }
}

remove(full_word, suffix)
{
      stem_length = full_word_length - suffix_length;
      stem = full_word substring(0, stem_length);
      if (suffix_length > 0)
      {
            if (first letter of suffix is a vowel)
            {
                  if
                  ((stem does not end with "w", "x", "err", "orr" or
                  "omm")
                  AND
                  (stem ends with two identical consonants))
            `     {
                        output[0] = stem;
                        output[1] = stem without terminal letter;
```

```
            return output;
        }
        else if ((suffix starts with "i" ) AND (stem ends
        with "y"))
        {
            output[0] = stem;
            output[1] = stem + "ie";
            return output;
        }
        else if ((stem ends with("i"))
        AND (penultimate letter of stem is a consonant))
        {
            output[0] = stem + "e";
            output[1] = stem with terminal "i" replaced
            by "y";
            return output;
        }
        else if
        ((stem ends with "u")
        OR
            ((stem ends with a consonant)
            AND
            (penultimate letter of stem is a vowel))
        OR
        (penultimate letter of stem is a vowel))
        {
            output[0] = stem;
            output[1] = stem + "e";
            return output;
        }
    }
        else
    {
        if
        ((stem ends with("i"))
        AND
        (stem is not monosyllabic)
        AND
```

```
                    (penultimate letter of stem is a consonant))
                    {
                          replace terminal "i" of stem with "y";
                    }
                    else
                    {
                          output[0] = stem;
                          output[1] = stem + "e";
                          return output;
                    }
              }
        }
        output = stem;
        return output;
}
```

Fig. 4: Derivational tree containing "classical"



Results from Suffix stripping

The result of applying the Suffix Stripping Algorithm to the word list data was to generate a further 926 words of which 39 were morphologically unrelated (Table 18).

Application of suffix stripping can be productive for some words for which suffixation is also productive as shown for "classical" in Fig. 4.

69 cases of undergeneration in this experiment were identified plus 6 cases of consequent undergeneration. The causes of the observed undergeneration are tabulated in Appendix 15, summarised in Table 20. 12 out of 69 undergenerations (17.39%) arose because of an unimplemented rule involving Latin passive participles. Cases marked "Asynchronous French imports", mean that both words have a Medieval French derivation, but the spellings do not correspond because they were imported probably at different times from a language whose spelling was not yet standardised. In a further 3 cases both words are imported from Medieval French and the relation between them corresponds to a morphological transformation wholly within the French language. In all 28 out of 69 undergenerations (40.58%) involve the morphology of languages other than English (addressed in §5.1.2). Rules of inflectional morphology (apart from participle and gerund formation) had not been formulated. The data suggests the need for additional rules involving the suffixes "-ish", "-en", "-ure" and "-eous".

Table 20: Main causes of undergeneration in suffix stripping

Reason for undergeneration	**Instances**
Latin passive participle	12
POS	6
Asynchronous French imports	5
Plural	5
French morphological rule	3
Latin genitive	3
Missing morphological rules	20

Table 21 shows the rules which overgenerated in suffix stripping and the ratios of productivity to overgeneration. All these rules involve removing a suffix and none involve substitution.

Table 21: Worst overgeneration in suffix stripping

Source		Target		Langs.	Total overgeneration	Overgenerations per rule execution
Wordform	POS	Wordform	POS			
age	NOUN		VERB	1	4	1.33
ed	NOUN		VERB	1	2	1.00
en	VERB		NOUN	1	2	1.00
al	NOUN		VERB	1	4	0.57
eer	NOUN		NOUN	1	1	0.50
man	NOUN		NOUN	1	2	0.50
age	NOUN		NOUN	>1	1	0.33
ise	VERB		NOUN	1	4	0.25

Table 22: Rules generating more wrong than right data on word list dataset

	Source		**Target**		**Over-generations per rule execution**	**Languages in formulation**
	Word form	**POS**	**Word form**	**POS**		
		V	ative	Adj.	3	1
		V	ed	N	1	1
	al	Adj.	ate	Adj.	1	1
	e	N	y	Adj.	0.75	1
		V	ant	Adj.	0.67	> 1
Suffixation		V	ee	N	0.5	1
	age	N		V	1.33	> 1
	ed	N		V	1	1
	en	V		N	1	1
	al	N		V	0.57	1
Suffix stripping	eer	N		N	0.5	1
	man	N		N	0.5	1

3.2.2.2.3 Overgeneration of Suffix Generation and Suffix Stripping Compared

Table 22 shows those rules which generated more wrong data than right data in the two word list experiments. The last column in the table indicates where overgeneration was caused by monolingual application of a multilingually formulated rule, including the worst overgenerating rule for suffix stripping. Correct multilingual application of such rules could yield an improvement in performance. Certain rules overgenerate below a threshold word length (Porter, 1980), producing false associations such as between "fin" and "fine"; "read" and "ready", and between unrelated homonyms.

Table 23 shows all the rules which overgenerated in more than one experiment. All these rules involve appending or removing a suffix and none involve substitution; none of them were multilingually-formulated. Of these rules, appending "-ed" to a verb to form a noun has produced *only* overgeneration. Further investigation into the circumstances in which these worse performing rules overgenerate might enable these rules to be reformulated. Shorter words tend to be morphologically irregular. It would be useful to look at threshold word lengths, below which certain rules overgenerate. These issues are taken up in §5.1.

Table 23: Persistently overgenerating rules

Unsuffixed POS	Suffix	Suffixed POS	Langs.	Output overgeneration / rule productivity		
					Word list	
				CatVar	Suffixation	Suffix stripping
NOUN	y	ADJECTIVE	1	0.13	0.14	0.09
VERB	al	NOUN	1	0.38	0	0.57
NOUN	man	NOUN	1	0.09	0	0.5
NOUN	age	NOUN	>1	0.67	0	0.33
NOUN	ate	VERB	1	0.67	0	0.2
VERB	er	NOUN	1	0.03	0	0.02
VERB		NOUN	1	0.005	0	0.01
NOUN		VERB	1	0.02	0	0.003
VERB	ed	NOUN	1	0	1.00	1.00
VERB	ed	ADJECTIVE	1	0	0.02	0.11
ADJECTIVE	ly	ADVERB	1	0	0.01	0.03

3.2.2.3 Prefixations in the Random Word List

So far all the experiments with affix generation and affix stripping have been applied to suffixes. Because only 2 cases of prefixation occurred in the CatVar dataset, no conclusions could be drawn about prefixations. However an examination was made of prefixations in the random word list (§3.2.2.2.1) to see if any rules could be deduced.

Irregular forms of prefixes can be identified by a *footprint*, which is a combination of characters not necessarily the same as the base form of the prefix, but which result from the process of prefixation. An *unregularised prefix* is either a *standard* prefix (a prefix in

its original morphological form) or the modified prefix component of a prefix *footprint* (§3.4.1), with morphological differences from the standard form of the prefix. A *regularised prefix* is an unregularised prefix regularised to its original morphological form. Each regularised prefix is semantically identical in origin, though its meaning in context may vary with the stem to which it is attached, but such semantic variations bear no relation to the morphological variations of the unregularised prefix or its footprint. The transformations involved in prefix regularisation are called *sandhi*.

To illustrate these concepts, take the word "imperil": here the stem is "peril" and the unregularised prefix is "im-", which corresponds to the regularised prefix "in-" but since, according to the identified rules (for further details see §§5.3.11.4.2, 5.3.11.5), "in-" only changes to "im-" under certain conditions, the footprint is "imp-". Conducting a lexicon search on this footprint will discover only those instances of the unregularised prefix "im-" which are modifications of "in-" before "p". For another example take the word "acquiescence": here the stem is "quiescence" and the unregularised prefix is "ac-", the footprint is "acqu-" and the regularised prefix is "ad-".

Some prefixes occur in two different forms, one ending with a consonant, which is the form which precedes a vowel at the beginning of the stem ("mon-" in "monaural"), and the other with a linking vowel, which is the form which precedes a consonant at the beginning of the stem ("mono-" in "monochrome"). Since it is not always clear whether the linking vowel is part of the prefix or not, and it may be debatable whether the form without a linking vowel is an abbreviation of the form with a linking vowel or the form with a linking vowel is an extension of the form without a linking vowel, this phenomenon has been treated separately from the regularisation of prefixes as described above. This issue is taken up in §5.3.11.9.

Table 24 shows the 20 most frequently occurring prefixes in the random word list in their regularised form. The occurrence counts include the modified forms which have been regularised as well as occurrences of the regular form. It is noticeable that a high proportion of these prefixes have a Latin or Greek origin, often as prepositions. The

Table 24: Most frequent prefixes

Regularised prefix	Occurrences	Original language(s)	Meaning1	Meaning2	Meaning3
in	34	Latin/English	in	not	ANTONYM
un	34	English	ANTONYM	not	
con	21	Latin	with	together	
de	20	Latin	from	down	ANTONYM
re	18	Latin	back	again	
ex	16	Latin	out(of)		
dis	13	French	ANTONYM		
sub	9	Latin	under		
ad	8	Latin	to		
non	8	Latin	not		
pre	8	Greek	before		
a	6	Greek	without	not	ANTONYM
per	6	Latin	through	thorough	
pro	6	Latin	for		
en	5	French	in		

English translations of some of these prepositions also occur themselves as prefixes[53]. It is also worth noting that the same prefix is likely to have more than one meaning (§5.3.11.3), and that several common prefixes convey antonymy (§§5.3.5).

3.2.2.4 Application to the Enrichment of WordNet

In order to investigate whether WordNet could be usefully enriched by encoding more morphological relations between word senses and whether it could be further usefully enriched by interpreting morphological relations between word senses as semantic relations (Bilgin et al., 2004; Koeva et al., 2008; §3.1.3), the first step is to discover what proportion of morphological relations are already encoded in WordNet, either as derivational pointers or as other types of relation.

[53] See Appendix 50 for the paucity of prefixes of Anglo-Saxon origin: only "hind-", "mid-", "under-", "be-", "deed-", "die-", "kin-", "none-", "off-", "un-" and "with-" occur, though "a-" (non-antonymous) and "in-" (non-antonymous) are sometimes Anglo-Saxon. These amount to 2% of the valid prefixes identified in §5. In most words beginning with an English preposition, including all prefixations derived from English prepositions not listed here, the rest of the word is also a word in its own right. Such cases can be considered as *concatenations*.

WordNet Relations between members of CatVar Clusters

Inasmuch as the CatVar sample is representative of morphologically related word clusters, it is pertinent to ask how many of the morphological relations between members of the sample clusters are already encoded in WordNet. Class `CatVarTuple` stores the relations in which the WordNet senses of the word form it represents, or the synsets to which these senses belong, participate[54]. All the words in the sample dataset were implemented as instances of `CatVarTuple` and each cluster was implemented as a `CatVarCluster`[55]. The Suffixation and Suffix Stripping Algorithms were adapted to output `CatVarTuple` arrays instead of `POSTaggedWord` arrays, which were similarly grouped into clusters for each seed word. It was then a simple matter to count the number of WordNet relations between the members of each `CatVarCluster`. WordNet derivational pointers were counted separately. For the CatVar sample dataset, 2366 Wordnet relations were found between pairs of synsets or word senses containing one or more words from within the same CatVar cluster. Of these 1963, or 82.97% are derivational pointers, making an average of 4.54 WordNet relations (3.77 derivational pointers) per cluster.

Since it is possible for more than one WordNet relation to exist between the same two synsets, or for one relation to exist between two synsets and another to exist between two word senses each of which belongs to one of the two synsets, the number of duplicate relations was also calculated, totalling 86. The maximum possible number of relational pairings for each cluster (excluding duplicates) was calculated as

$$\frac{n^2 - n}{2}$$

where n = the number of members of the cluster. This would be the number of relations if there was a relation between each member of the cluster and every other member.

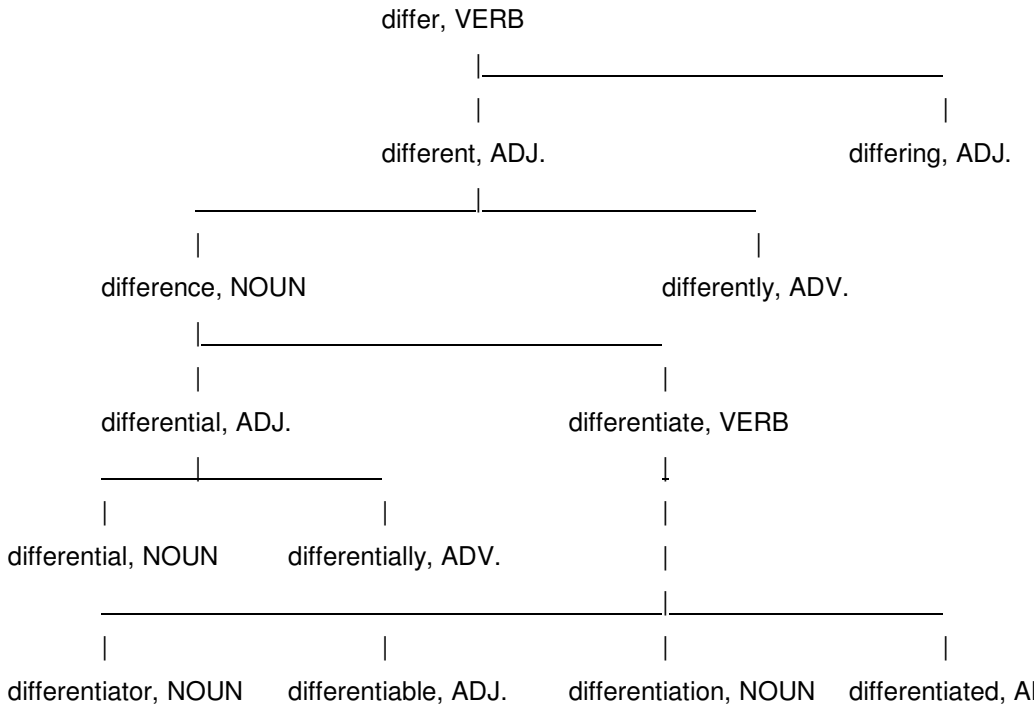
[54] The `CatVarTuple` constructor searches the WordnNet model for all the relations of all the senses of the word represented, whether betweensynsets or word senses.
[55] Class Diagram 8.

Since derivation is a directional phenomenon, each member of a cluster can be considered to be directly derived from 1 and only 1 other member. However all correct members are related directly or indirectly and every member is directly or indirectly derived from a common root, so that the entire cluster forms a derivational tree (§3.1.4; Fig. 5). The ideal or optimal number of relations per cluster is then equivalent to the number of links between nodes in a tree which is

$$n-1$$

where n = the number of nodes.

Fig. 5: Derivational tree for a CatVar cluster



The representation of derivational relationships within a cluster as a derivational tree, implying the directionality of morphological relations, might be useful for detecting false morphological relations generated algorithmically. For instance the CatVar dataset links the word "student" to the word "stud". A morphological rule might be formulated to represent the transformation from a noun to another noun by appending "-ent"; another rule might represent the transformation from a noun with suffix "-y" to another noun by

substituting "-ent", then the word "student" would be treated as simultaneously derived from "stud" and from "study"[56]. This dual inheritance would violate the tree structure so that an exception could be detected by the algorithm. This would highlight the fact that only one of the proposed roots of "student" can be correct, at which point human intervention could quickly establish that only "study" and not "stud" is the root of "student".

Using the above definitions of maximum possible and ideal or optimal, it was discovered that over the entire CatVar sample dataset, only 6.17% of the maximum possible relations were realised in WordNet while 54.64% of the optimal number were realised. This means that almost half these morphological relations are not encoded, confirming the potential for further enrichment of WordNet with morphological relations.

With the dataset generated from the word list (§3.2.2.2.1) by suffixation, there were an average of 0.60 WordNet relations per cluster of which 80.29% were derivational pointers. The WordNet relations represented 3.9% of the maximum possible and 34.14% of the optimum. With the dataset generated from the word list by suffix stripping, there were an average of 0.91 WordNet relations per cluster of which 78.87% were derivational pointers. The WordNet relations represented 4.02% of the maximum possible and 34.00% of the optimum.

Comparison of WordNet relation occurrence between members of clusters of derivationally related words for each experiment.

Table 25 shows little variance between experiments in the proportion of the WordNet relations which are derivational pointers. However, using CatVar data as a starting point yields a significantly higher relation count. This discovery suggested that CatVar data had already been used for WordNet enrichment, as planned (Habash & Dorr, 2003). However this is refuted by Fellbaum and Miller (2007; §3.1.3). It would appear then that the

[56] This proposal applies only to suffixations, which constitute the greater part of the CatVar data. It clearly does not apply to concatenations such as “trenchcoat” (§3.1.4), nor does it apply to prefixations.

undocumented methodology used for the creation of CatVar was similar to that adopted by Fellbaum and Miller, and it seems likely that some derivational pointers have been subsequently re-encoded as other WordNet relations. It is also abundantly clear that there is plenty of scope for further enrichment.

Table 25: WordNet relations between members of clusters of derivationally related words

	CatVar dataset		Word list suffixation		Word list suffix stripping	
	TOTAL	**AVERAGE**	**TOTAL**	**AVERAGE**	**TOTAL**	**AVERAGE**
WN DERIV relations within cluster	1963	3.77	664	0.60	1008	0.91
WN relations within cluster	2366	4.54	827	0.75	1278	1.15
DERIV as proportion of WN relations	82.97%		80.29%		78.87%	
Duplicate relations	86	0.17	26	0.02	34	0.03
Total synsets / cluster		9.01		3.12		4.30
MAX possible relations / cluster excl. duplicates		70.98		18.54		27.95
Proportion of possible relations in WN	6.17%		3.90%		4.02%	
Optimal relation count / cluster		8.01		2.12		3.30
Proportion of optimal relation count realised in WN	54.64%		34.14%		34.00%	

3.2.2.5 Conclusions from the Pilot Study

The provisional conclusions about the rule-based approach which can be drawn at this stage, presented at the NLPCS 2009 Workshop (Richens, 2009a) may be summarised as follows:

- CatVar is not reliable for identifying morphological relations.
- There is scope for improving WordNet by enrichment with morphosemantic relations.
- Morphological rules are not reliable below a threshold word length.
- Deployment of multilingual resources to apply multilingually formulated morphological rules would improve recall and precision.

- Morphological rules could better be formulated from empirical data such as the frequencies of affix occurrences in the lexicon.

3.2.3 Conclusions on Morphological Rules

Suffixes are better served than prefixes by morphological rules. It seems impossible and unnecessary to formulate a set of rules for prefixation as for suffixation. Only generalised spelling rules are required. The reasons for this lie in the essential differences between prefixation and suffixation in English. Prefixes do not perform part of speech transformations. While meanings have been identified for the prefixes investigated (Appendix 50; §5.3.11.3), these meanings do not generally correspond to syntactic transformations as is the case for suffixes, the notable exception being prefixes which express antonymy (§§3.5.1, 5.3.5). Many prefixes correspond to words used as prepositions. These frequently occur in antonymous pairs such as between prefixes "ana-" and "cata-". While WordNet can be enriched with morphological relations between prefixations and their stems, much more research needs to be undertaken before any semantic relations, apart from antonymy, can be established. If prepositions were added to WordNet, then prefixes could be associated with them and relations could be encoded between the prepositions and the corresponding prefixations. This would be a first step towards representing the semantics of prepositions and their corresponding prefixes. Insufficient data has so far been gathered on prefix meanings. Many prefixations correlate with verbal phrases of the verb + *particle* type discussed in §§4.1.1, 4.2.1.2 (see also §3.5.2).

Further investigation is needed to establish whether all or most instances of common prefix footprints are semantic instances of the prefix and not simply co-incidences of character combinations, without the corresponding etymology or meaning. Occurrences of each footprint will need manual evaluation.

The representation of sets of morphological relations between members of clusters of morphologically related words as trees with a single root (§3.1.4) applies to suffixation

but not generally to prefixation. This is because the meaning of suffixes (in all the cases examined with the exception of "-man") is always grammatical or relational. To put this another way, suffixes are not words in their own right; they convey meaning only by defining a relation upon their stems. Prefixes on the other hand (with the exception of those which convey antonymy) have meaning in their own right: they may exist as words in their own right; if not, they correspond to a single and translatable word in another language. Consequently prefixations have *dual inheritance*: they are morphologically derived from both prefix and stem, each of which contribute an element, however obscure, to the meaning of the prefixation. In this respect prefixations are more akin to concatenations than they are to suffixations, whose singular inheritance is encapsulated in the morphological rules (§3.2.2.1, Appendix 10). Prefixations where the prefix conveys antonymy can be added to the clusters of words morphologically related by suffixation and represented as derivational trees.

Overgeneration is a consequence of attempting to encode derivational morphology without reference to etymology. Etymology avoids making false connections such as between "moth" and "mother" (Bilgin et al., 2004). Correctly encoding morphological data requires correctly decoding derivational history. This involves unravelling language back through its evolution. This evolution has taken place, in Europe (Fig. 1, §1.2.2), with no respect for the boundaries between languages, which have only been defined relatively recently in the course of that evolution, mainly on political rather than linguistic criteria, while Latin remained the only standardised language. In the course of this evolution, ancient morphemes have acquired layers of affixes, while words have accumulated new layers of meaning which sometimes efface previous meanings. For instance the word "catholic", itself a prefixation derived from a Greek word for "whole", used to mean "universal", but has come to have an sectarian meaning[57]. However, premature encoding of semantic relations corresponding to the morphological transformations performed by prefixation, from delving too deeply into etymology, runs

[57] While the original meaning has not completely disappeared from use, the implicitly contradictory sectarian meaning has become dominant.

the risk of identifying semantic relations which belong to history but which are unlikely to be helpful, when applied to NLP tasks involving entirely modern texts.

Experiments with affix generation and removal have demonstrated some possible pitfalls in identifying morphological relations. There is a risk that overgeneration by morphological rules may outweigh the discovery of relations (Porter, 1980; §3.1.1). Some morphological rules have been shown to be unreliable as applied, and need more rigorous formulations (§5.1). It appears that certain rules overgenerate beyond a threshold word length, which is best measured in syllables. From observations of false associations such as between "fin" and "fine" and "read" and "ready", and between monosyllabic homonyms, it is suggested that the threshold lies between 1 and 2 syllables, so that the applicability of a suffix to a word is significantly less probable if that word is monosyllabic and, conversely, that to produce a monosyllabic output from suffix stripping is much less likely to be correct than when the output is polysyllabic. Restrictions on the application of morphological rules to generate monosyllables (§5.1.1) would allow the automatic processing of more regular longer words while avoiding overgeneration from shorter words. Undergeneration consequent upon this approach is addressed in §5.3.14.2.

Some of the most important morphological rules have not been applied, for lack of multilingual resources. Some others have been applied monolingually, often with unsatisfactory results. Erroneous connections as between "carry" and "carrion"; "bully" and "bullion", are the result of applying the "-ion" suffix indiscriminately, without reference to the Latin passive participles to whose stems they are generally applicable. The most important cause of undergeneration observed has been non-application of rules requiring reference to these participles. Applying such rules is the most important single improvement that could be made. This will be taken up in §5.1.2. Possible approaches are the harnessing of appropriate multilingual resources or inference from co-occurrences of morphological patterns in the lexicon. Latin passive participles could be identified from quasi-gerunds, assisted by the morphology of stems from prefix stripping, exploiting

common patterns such as between {"conceive" : "conception"} and {"perceive" : "perception"} and between {"permit" : "permission"} and {"commit" : "commission"}.

3.3 Review of Existing Morphological Analysis Algorithms

This section will review, from a linguistic point of view, three algorithms which apply numeric methods for morphological analysis. The authors who present these algorithms each acknowledge the contribution of their predecessor and all use some kind of corpus data as input for their experiments. The adequacy of the corpora for the purpose will also be examined. The first algorithm uses a phonetic representation of language; the sufficiency of the other algorithms will be judged partly by their ability to handle spelling irregularities. Particular emphasis will be placed on questioning their common initial assumption that morphological analysis can be achieved by segmentation, an assumption upon which considerable doubt is thrown by the results obtained, but which is only belatedly called into question by the last of the three authors.

3.3.1 From Phoneme to Morpheme

Harris (1955) attempts to identify word and morpheme boundaries within utterances, treated as sequences of phonemes, by counting the number of possible *successors* and *predecessors* of each phoneme, which tend to peak at such boundaries. The successor of a phoneme *n* is the next phoneme in the sequence and its predecessor is the previous phoneme. The possible successors and predecessors are identified from a corpus of elicited utterances, transcribed, without word segmentation, using phonetic characters.

Given a test utterance as a sequence of phonemes and a collection of control utterances in the same format, the basic algorithm can be represented as follows:

```
successor counts is an array of integers whose size = test utterance
length - 1
for each value of n from 0 to test utterance length - 1
{
      successors = empty collection of phonemes
      sequence = test utterance up to and including the phoneme at
      position n
      for each control utterance
      {
            if (control utterance starts with sequence)
            {
                  successor = phoneme at position n + 1 of control
            utterance
                  if (successors does not contain successor)
                  {
                        add successor to successors
                  }
            }
      }
      successor count = size of successors;
      successor counts[n] = successor count;
}
segment initial position = 0;
for each value of n from 0 to test utterance length - 1
{
      if (
            (successor counts[n] > successor counts[n - 1])
            AND
            (successor counts[n] > successor counts[n + 1]))
      {
            place segment boundary after n
      }
}
```

Harris proposes various variations on this basic algorithm, of which the most important is to use predecessor counts to increase the level of confidence in the segmentation derived from successor counts.

Implicit in this work is the assumption that it is always possible to segment words into morphemes, an assumption regarded as fallacious in this thesis (§§3.3.2, 3.3.3). The preference for using phonetics is not intrinsic to the methodology which can equally well be applied, using standard characters, to written text. A comprehensive lexicon is more informative about patterns of successor and predecessor possibilities among alphabetical characters than an elicited set of utterances is about such patterns among phonemes.

Automatic affix discovery (§3.4) uses the relative frequencies of initial and terminal character sequences and also takes into consideration the frequencies of their parent and child character sequences where the child is the combination of the parent and its successor, in the case of suffix discovery, or the combination of the parent and its predecessor in the case of prefix discovery. To this extent, automatic affix discovery can be considered to be an extension of Harris's approach.

3.3.2 Word Segmentation

Hafer & Weiss (1974) build on the work of Harris (1955; §3.3.1) in an exercise in word segmentation motivated by the requirements of information retrieval (cf. Porter, 1980; §3.1.1). As such they are satisfied with an imperfect identification of stems, as long as it will enable queries to be handled correctly.

Their basic algorithm is exactly the same as that of Harris except they use text with normal alphabetical characters instead of a phonetic representation. As such, segmentation into words is not required, only segmentation of words into morphemes. They use a corpus of words, which is the equivalent of a limited lexicon, to replace the control utterances used by Harris. Like Harris, they employ predecessor variety counts as well as successor variety counts, because successor variety counts always decrease towards the end of a long word, skewing the results. For computational efficiency, they use a *reverse corpus* for rapid determination of predecessor counts, a technique similar to the deployment of a *rhyming dictionary* in the methodology of automatic suffix discovery

(§§3.4.2.1, 5.3.3.2). Their first major innovation is to take into consideration instances where the beginning or end of a test word exactly matches a word in their corpus. They represent this scenario by making the successor count negative, where the match occurs at the beginning of the word, or the predecessor count negative, where the match occurs at the end of the word. They differ from Harris in preferring to set cutoff values for predecessor and successor variety counts and placing a segment break where such cutoff values are reached, rather than using peaks.

One major innovation of Hafer & Weiss is the use of measures of entropy to weight the possible successors or predecessors according to their probability. However among the 15 different experiments they describe, at no point does the deployment of entropy measures result in an improvement to the results.

Since the purpose of their endeavour is to identify stems for information retrieval purposes, a stem identification algorithm is required, to be applied to the segmented words. The stem identification algorithm is very loosely described: by default, where a word consists of two segments, the first is treated as the stem, but if the first segment "occurs in many different words, it is a probably a prefix" (p. 375), but just how many, they do not say. In cases where there are two segments both of which are words in their own right, a phenomenon referred in this thesis as a *concatenation* (§§3.5.2, 5.3.4), both are treated as stems.

They refer to the use of three corpora, but results are given only for 2. All words of less than 3 letters were excluded on the grounds that to include "be" and "an" would result in a false segmentation of "bean". It is unclear why they do not consider using such words for the control words, particularly as "be-" is a recognised prefix. One of the corpora also had words in a given list of function words removed and the other had all words with less than 5 letters removed. While removal of function words is a standard procedure in NLP, no convincing justification is given for the removals.

Cutoff values were set at 5 for successor variety counts and 17 for predecessors. In experiments where the variety counts were added together, the cutoff was set to 23. Negative values, encoded where whole words were identified, were treated as if they exceeded the cutoff values so as always to trigger a break. This is an error, as the initial experiments in concatenation analysis described in this thesis demonstrate. One can only surmise that the word "ion" was not in any of their corpora (§5.3.4.2).

Precision was measured as the number of correct cuts divided by the total number of cuts, but how correctness was judged is not stated. Recall was measured as the number of correct cuts divided by the total number of true boundaries, but how the true boundaries were determined is also not stated. The assumption that there is always one correct way to segment a word into morphemes is implicit in this work. This assumption is contradicted by many instances of prefixation and suffixation which are not simply a matter of putting a morpheme before or after another but frequently involve the disappearance or appearance of letters, as is amply illustrated by the spelling rules and morphological rules presented in this thesis (§3.2.2; Appendices 9, 10, 14, 36).

Of the 15 experiments described, 2 are rejected as so unsuccessful that it was not deemed worthwhile to record the results, namely using only successor variety count cutoffs, and segmentation before a suffix which is a complete word in itself. The description of the results of the other experiments reflects the authors' unambitious criteria, which may be justified by the stated motivation: a recall of 51% is described as "fair" (where both successor and predecessor variety counts are required to reach a cutoff at the same point); when the results from stem identification are discussed, a precision of 74% on one corpus and 61% on another is described as "quite good". Better results are attainable by more linguistically informed methods (§5).

In general, with various combinations of variety counts using both peaks and cutoffs, wherever the recall is good, the precision is poor and vice versa. In the case of successor variety peaks, it is acknowledged that less than half the cuts are correct. The examples given include "diffusion" segmented into "di", "ff" and "usion". This illustrates the

inadequacy of segmentation as a tool for morphological analysis: "dif-" is a recurrent modification of the irregular prefix "dis-" before "f", occurring also in "different" and "difficult"[58] (verified by OED2; §§5.3.11.2, 5.3.11.5). It is fallacious to assume that once an affix is identified, the true stem is by default simply the residue after removing the affix from the word (§3.2.2; Appendices 9, 10, 36). This will be referred to as the *segmentation fallacy*.

The best results are obtained by a hybrid method, which places a cut where it identifies a whole word to the left confirmed by a predecessor count of at least 5 or where a predecessor count of at least 17 is confirmed by a successor count of at least 2.[59] This gives 91% precision and 61% recall. The equivalent method using entropy performs less well, though it was subsequently modified to give the next best results.

Errors in stem identification illustrate the need to take spelling rules into account (e. g. "wives" not associated with "wife"). Hafer & Weiss conclude from false stems such as "elect" for "electron" that it is better to use a high precision method than a high recall method and so abandon all the other methods, including all those which use entropy, in favour of the hybrid method detailed above for their final experiments with information retrieval. Detailed results for stem identification are given for this method: these results are classified according to whether the computed stem is deemed to be *"correct", "too long", "too short"* or *"wrong",* but no criteria are given for these classifications.

Examples where the stem identified is *too long* include "hopefully" where the stem extracted is "hopeful"[60], and two examples of words derived from Latin irregular passive participles: "descriptively" not associated with "described" and "transmissions" not associated with "transmitted". Such examples demonstrate the inadequacy of a methodology which ignores the historical evolution of languages in favour of purely numeric criteria for the purpose of morphological analysis.

[58] The prefix footprint is "diff-".
[59] It is not stated how these thresholds were arrived at.
[60] The suffix "-ly" is one of the easiest to identify (from its frequency), but the suffix "-ful" appears to be too difficult for this methodology.

The authors consider the case of stems which are *too short* to be more serious. Here they cite two cases of terminal whole word identification: "ring" in "appearing" and "red" in "cleared" and "compared". They cite these cases as reasons to eliminate short words from the corpus, but this would undoubtedly have a detrimental impact on recall.

Examples of stems which are *wrong* include "trans" for "transplant", where the prefix "trans-" has not occurred with sufficient frequency in the corpus, though it is an easy prefix to identify in that it is not prone to spelling modifications. Another example is "care" for "career", where application of simple spelling rules would address the problem, such that "carer" but not "career" could be considered a derivative of "care". Another example, "ear" for "early" involves a violation of the required POSes encapsulated in the morphological rule which allows removal of "-ly" from an adverb to obtain an adjective[61] (Appendices 9-10).

The authors seem happy with their results for information retrieval, which outperform a lexicon for their limited purposes. However their conclusion (p. 385) that "accurate word segmentation is achieved" is indefensible, even given their limited objectives, as evidenced by the examples they give from their own results.

3.3.3 Minimum Description Length

Goldsmith (2001) sets out to acquire the morphology of any language from any corpus with no dictionary and no morphological rules. His underlying model uses the principles of the information-theoretic *Minimum Description Length* (*MDL*) framework, which seeks to find "the most compact representation of the data and the most compact means of extracting that compression" (p. 154), which, he argues will correspond to the best morphology. In this context, the "representation" is through the means of stems and suffixes (there is no a priori reason why the method should not be extended to prefixes).

[61] "Early" can be an adjective or adverb but "ear" can only be a noun.

Acknowledging the contribution of Harris (1955), he assesses that the heuristic is good, but is not capable of further refinement.

Goldsmith's approach involves the extraction, from a corpus, of a list of suffixes, a list of stems and a list of signatures, each of which comprises a mapping from a minimum of two stems to a minimum of two suffixes. To achieve the most compact representation, the stems and suffixes must themselves be encoded in such a way that the most frequent characters require the fewest number of bits, while the most frequent stems and suffixes are similarly represented by the fewest bits. That analysis of the words in the corpus into stems and suffixes which occupies the fewest bits (allowing for the additional bits to store the lengths of the structures) is deemed to be the best morphology. The basic model is complicated by the fact that a stem may itself be a word which itself can be subdivided into stem and affix. Allowing for this, the minimum description length can be calculated as a *figure of merit* against which any analysis can be assessed. Thus the Minimum Description Length framework evaluates the quality of a morphological analysis and can be used to direct the search for an optimal analysis; it is not a tool for morphological analysis itself.

The actual morphological analysis is performed by a heuristic, which applies cuts to split words into stem and suffix. Three approaches are described. However the first approach (*expectation-maximisation*) is dismissed on the grounds that it will always prefer to make a cut either after the first letter or before the last letter. The next approach (*Boltzmann distribution*) prefers relatively long suffixes and stems and cuts every word, which is clearly not optimal as not all words carry suffixes. The final heuristic counts all *n*-grams of 2 to 6 letters which appear at the end of each word, including an end of word symbol. Using a measure of *weighted mutual information*, the likelihood that an *n*-gram is a suffix is calculated. The top 100 then become the *set of candidate suffixes*. All the words which contain one of these suffixes are then split. Since some words end with more than one of the candidate suffixes, the *figure of merit is* used to choose among them. The initial results, using Twain's *Tom Sawyer* as the corpus, were produced by this approach.

This methodology is similar to automatic affix discovery (§3.4), in so far as a list of candidate suffixes is generated by numeric means. However automatic affix discovery does not need any end of word symbol, since all suffixes by definition occur at the end of words and all prefixes at the beginning of words. Goldsmith limits the *n*-grams to 6-grams (5-grams in reality since there is always an end of word symbol) on the grounds that "no grammatical morphemes require more than five letters in the languages we are dealing with" (p. 172). This statement is incorrect, since he does deal with French, which has grammatical suffixes "-issons" (6+1) and "-issions" (7+1) and Latin which has "-averitis" and "-averatis" (8+1), "-avissemus" and "-avissetis" (9+1). Automatic affix discovery as described in this thesis allows up to 10-grams (§3.4.1.1), a limit which was set only when it was discovered that 11-grams produced no candidate prefixes (defined in the broadest possible way as any combination of letters which occurs at the beginning of more than one word). Also setting a limit of 100 to the set of candidate suffixes seems somewhat restrictive: no justification is given for it. Automatic affix discovery generates candidate affix sets comprising tens of thousands of members and the heuristics adopted (which do not include weighted mutual information) are used to sort the set, not to limit it; the criteria for choosing a heuristic are linguistic. The most important difference in approach however is that in this thesis it is not assumed that the stem is by default the residue from affix removal (§3.3.2). Goldsmith, unlike Harris (1955) and Hafer & Weiss (1974) at least shows that he is aware that this is not always the case, but does not go far enough in exploring the implications of the segmentation fallacy (but see also below).

Goldsmith's initial results include all the main inflectional suffixes for English, the irregular inflectional suffix "-en", the abbreviated terminations "-'ll", "-n't" and "-'s" (but not "-'d") and various common derivational suffixes including "-tion" (but not "-ion" or "-ation"). The author does not acknowledge these omissions. One problem which is acknowledged is the over-application of various short suffixes. In particular many words ending in "-s" have been treated as suffixations when they are not. There are a few false suffixes such as configurations of lowercase roman numerals (not acknowledged) and the spurious suffixes "-n", "-p" "-red" "-st" and "-t", all applied to the spurious stem "ca-" (acknowledged). Such errors arise from the segmentation fallacy which is implicit in this

version of the software. The same fallacy gives rise to failure to associate "abbreviates" and "abbreviated" with "abbreviating" and "wins" with "winning". Spelling variations of this kind are well known, and the problem is acknowledged but not resolved. Double suffixes "-ings" and "-ments" are not recognised as such. This particular problem can be addressed by MDL being applied to attempts to split suffixes. Inflectional suffixes preceded by "t" are also generated. Goldsmith proposes to address this by applying MDL while temporarily disallowing single letter suffixes, and the remaining problems by introducing a post-analysis *triage* phase (below). He is aware of, but has not yet got to grips with, other problems which illustrate the segmentation fallacy. These arise in particular from irregular Latin passive participles, of which he acknowledges only the "d"/"s" alternation as in "intrude"/"intrusion" etc. He brackets this with the "i"/"y" alternation, which has a completely different origin. Reference is made to words with identical stems but unrelated meanings, but no solution to this is offered, nor indeed is likely ever to be possible by application of semantically ignorant numeric methods.

Without having addressed the acknowledged shortcomings of his approach, Goldsmith goes on to present results for various languages using corpora ranging in size from 100,000 to 1,000,000 words (tokens). Unfortunately he provides only a handful of the first alphabetically ordered examples for each of only the top 10 signatures for each, which casts relatively little light on the morphology of the other languages, all of which are much more highly inflected than English. The results for a 500,000-word corpus of English (part of the Brown Corpus) do not differ significantly from the results for Tom Sawyer. For French, 9 of the top 10 signatures are for groups of adjectives. The stem lists given for these signatures are limited to the first 9 or 10 alphabetically. Only one of these signatures has the adverbial suffix "-ment" and all the examples given for it have stems ending in "-e". None of the other signatures include the adverbial suffix "-ement". Another signature has the feminine singular and plural suffixes "-e" and "-es" but not the masculine plural "-s", even though 2/10 of the examples can carry that suffix. Another signature has both plural suffixes but no feminine singular suffix even though all the examples given can carry it. These results are to be expected. A very large corpus would be required to find all the possible inflections of all the adjectives. The only non-

adjectival signature given applies to a group of verbs with a set of 12 common regular verbal inflections, but there are only 4 verb stems in the group, which encompass a full alphabetic range, indicating that it is the complete list of stems. As verbal inflections are numerous, a very large corpus, undoubtedly larger than any existing corpus, would be required in order to find all the possible inflections of any regular verbs. Goldsmith acknowledges that he needs to find a way to merge signatures where not all possible suffixes are represented into groups where they are all represented. This problem is addressed by the *paradigm* structure (see below).

The top signature for Latin[62] is the co-ordinating conjunctive suffix "-que" which can occur with any word. The remaining 9 signatures in the top 10 comprise 6 groups of nouns, 2 groups of adjectives and 1 mixture of nouns and adjectives. Most of these signatures are subsets of regular declensions, one is a small group of 3rd. declension nouns whose regularity only arises from the non-occurrence of their nominative singular forms in the corpus and one is a group drawn from all declensions which occur in the corpus, but in accusative singular and plural forms only, so that the suffixes are "-m" and "-s". Thus the classification bears very little relation to the common properties of groups of nouns and adjectives which have been recognised since antiquity. These results do have one merit however, in that they suggest that there is a simpler way of defining Latin grammar than the way it is traditionally taught, in other words that MDL would have the potential to derive a grammar that is simpler by virtue of being shorter. However, given the lacunae, this potential could probably never be achieved without a corpus larger than the entire corpus of known Latin texts.

For Italian, two corpora were used, one of 100,000 words and one of 1,000,000 words. The results neatly demonstrate that corpus size is a critical factor. With the 100,000-word corpus, there are no verbal signatures, and most of the signatures are composed entirely of single vowels (the stems not being provided for Italian). With the 1,000,000-word corpus one signature appears comprising (at least in part) common regular verbal inflections.

[62] clearly mainly ecclesiastical Latin, judging from the range of words

Goldsmith goes on to evaluate his own results, categorising them as "good", "wrong" (incorrect analysis) "failed" (no analysis) or "spurious" (atomic word split) and awards himself around 83% "good" for both English and French. His criteria for "good" clearly do not include completeness (all inflections represented). His criterion for calculating recall at 85% to 91% does not account for incompleteness either; it is simply based on how much of the corpus has been analysed. The evaluation is an assessment of whether each compound consists of the specified stem and suffix but does not consider whether each possible suffix is given for each word.

Goldsmith says that he is "surprised" how often "it was difficult to say what the correct analysis was" (p. 182), giving examples for most of which there is no correct segmentation (illustrating the segmentation fallacy). In most of these cases, he has marked the results as "good". His criteria for this include one reasonable criterion, that it is better to have an analysis which groups related words together, even though it is debatable what the stem is, than to group them separately with different stems. The other criterion is unclearly stated, but the example is "alumnus" and "alumni", where the stem is clearly "alumn-", and there are enough examples of this regular Latin inflection in English to justify its inclusion in a morphological analysis. He implies that the system should be given credit for discovering such phenomena, but not penalised when it fails to do so. When it comes to proper nouns, his criteria become even more arbitrary. Assessing results from a version which has not adequately come to terms with multiple suffixes, he is at a loss when confronted with a French verb such as "écrire", for which a grammar book will say that the stem is "écr-", even though all its forms start with "écri-", but which also has a longer stem "écriv-" to which various regular inflections can be applied. This phenomenon is commonplace among French verbs and is not confined to French.

After presenting this evaluation, Goldsmith takes up the issue of triage, which clearly had not been fully implemented at the time of writing. He cites the example of the signature *NULL;ine;ly* applicable only to the stem "just" and suggests that *ine* should be removed leaving the much more widespread signature *NULL;ly* and creating a new signature

comprising only *ine* to which other stems could be added. This approach could be systematically applied to signatures with only 1 (or perhaps 2) stems, but would mean allowing the same stem to occur in more than one signature, which is a major departure from the original approach. Applying this approach has impacts which increase the description length in some areas while decreasing it in others: the overall impact is not stated.

When it comes to the issue of incomplete subsets of inflectional signatures, relating signatures to each other has an adverse effect on the description length, calling into question the underlying thesis that the shortest description is necessarily the best. He proposes to introduce a new structure into the model, which he calls a *paradigm*, which is essentially a set of related signatures. This solution would be an improvement but does not address the underlying issue where a signature is incomplete not because of omissions in the corpus, but because of unimplemented spelling rules as in the case of *NULL;s* for "occur", where the doubling of the "r" in "occurring" has not been allowed for.

In summarising the outstanding issues, Goldsmith is non-committal about the desirability of handling multiple suffixes of the type implicit in French verbs such as "écrire" discussed above, and seems still to have no solution for "-ings" and "-ments". He does however finally come to terms with the segmentation fallacy, suggesting the implementation of an operator which can delete the last character of the stem, as for instance to connect "loving" to "love". A similar operator could remove the second "r" in "occurring", and other operators could handle many of the issues relating to the segmentation fallacy. The incorporation of such operators would allow his system to handle the basic spelling rules governing affixation in English, which the far simpler approach of Porter (1980; §3.1.1) achieved 20 years earlier.

Another issue raised rather belatedly is the precedence which has been assumed of suffix stripping over prefix stripping. It will be shown in this thesis that, while this is a good rule of thumb, it is vital to distinguish between antonymous and non-antonymous

prefixation in this regard. Removal of antonymous prefixes such as "un-" should take precedence (§3.5.1).

One must conclude that, although MDL has very interesting potential, there will come a point where results cannot be improved further because large enough corpora are not available and may never be available. It appears to be necessary to violate the principles of MDL to some extent in order to get the best results. The results presented, insofar as they are good, depend less on MDL than on the segmentation algorithm. The major pitfall is the segmentation fallacy. Without coming to terms with this, it is impossible to get a satisfactory association between related words.

Nothing that Goldsmith says has any bearing whatever on meaning. In this he perhaps emulates Chomsky, though Goldsmith is very modest in his conclusion when he talks about the goals Chomsky (1957) considered unachievable of producing a grammar automatically from a corpus, and being able to determine which grammar is the best with respect to a corpus. Goldsmith comes nearer to achieving these goals than anyone previously. However, more attention to the actual properties of each language is required before such goals become attainable.

One application which Goldsmith's methodology would undoubtedly be very good at, though one that he is not setting out to achieve, is language identification. It should easily be possible to associate sets of signatures from different corpora to generate signatures for languages. This would undoubtedly be very useful for organisations dealing with documents in multiple languages, and whose staff do not have any knowledge of those languages. Another possibly useful application would be as an aid to deciphering text in a forgotten language. However, for the purpose of morphological analysis, it still has a long way to go.

3.3.4 Conclusions on Word Segmentation

The main problem with all three algorithms reviewed here is their naive assumption that one can always obtain morphemes simply by segmenting a word, without inserting or deleting anything. This assumption has been referred to as the segmentation fallacy. Its falsity is amply demonstrated by the morphological rules already presented and by the observed properties of prefixations (§3.2.2). Hafer & Weiss (1974) fail to see the fallacy even when confronted with it, while Goldsmith (2001) realises the implications but fails to follow them up. Both ignore elementary spelling rules. The results obtained are disappointing from the point of view of a linguist: while Hafer & Weiss clearly build on the work of Harris (1955), Goldsmith himself sees no way to build on that of Hafer & Weiss; to get any significant improvement on Goldsmith's results would require impossibly large corpora.

In the rest of this thesis, an approach to the morphological analysis of words will be presented which avoids the segmentation fallacy, by first identifying affixes primarily by occurrence frequencies, but aided by other heuristics, and then applying rules, grounded in observation and etymology, governing the associations between affixes and the way they attach themselves to morphemes. While some work on the latter task has already been presented (§3.2.2), an algorithm to accomplish the primary task will now be introduced (§3.4), which will be used to feed into the rule-based approach and into other algorithms, to perform the complete morphological analysis presented in §5, using the lexicon as the sole data source.

3.4 Automatic Affix Discovery

This section describes an algorithm originally developed for the automatic identification of prefixes and then adapted for the identification of suffixes. The algorithm involves extracting initial and terminal character sequences of words from the lexicon and arranging them in trees where each level of the tree contains character sequences with

one more character than the at previous level, so that not only the frequencies of the character combinations (*affix frequencies*) but the ratios of those frequencies to the frequencies of their parent combinations (*parent frequencies*) can be used as an indicators of semantic relevance. The lexically valid proportion of the stems obtained by removing each character combination from the words in which it occurs (*stem validity quotient)* is a further indicator of semantic relevance. These indicators are combined for use as heuristics for sorting the data in the tree so as to bring to the fore the most semantically relevant combinations. Results are evaluated with reference to morphological rules and the performance of various heuristics are discussed with a view to establishing an *optimal heuristic*.

To qualify as an *affix*, a character sequence must satisfy the *duplication criterion*, that it occurs at the beginning (*prefix*) or end (*suffix*) of more than one word. It must also satisfy the *semantic criterion*, that it carries some *meaning potential* (Hanks, 2004), or at least defines a relation upon its stem. Any initial or terminal character sequence which satisfies the duplication criterion can be considered as a *candidate* affix, to be accepted or rejected as a *valid* affix according to the semantic criterion. The set of all prefixes in any language is then that subset of the set of all initial character sequences whose members satisfy these two criteria, and the set of all suffixes is that subset of the set of all terminal character sequences whose members satisfy the same criteria. That subset of the set of all prefixes whose members satisfy the duplication criterion can be considered as the set of all *candidate* prefixes to be accepted or rejected as a prefixes according to the semantic criterion; similarly the set of all *candidate* suffixes is that subset of the set of all suffixes whose members satisfy the duplication criterion. These sets can be computed from a digital lexicon. Given a lexicon derived from WordNet, it was clearly possible to compute the set of candidate prefixes from the alphabetical list of words which is the `keyset`[63] for that lexicon.

In order to distinguish between valid affixes (those which satisfy the semantic criterion) and coincidental character combinations, it is relevant to record the number of lexicon

[63] set of keywords.

occurrences of each affix (*affix frequency*) and to compare this with the frequency of its *parent* affix (*parent frequency*). By this it is meant, for instance, that the meaningless candidate prefix "su-" is parent of any prefix comprising "su-" plus one successor (in the sense used by Harris, 1955; §3.3.1), of which the most productive in terms of further successor frequencies are "sub-" and "sup-", as shown in Fig. 6. Where all the words starting or ending with a character sequence of length *n* also start or end with a character sequence of length *n* + 1, then the character sequence of length *n* need not be considered as a candidate affix as long as the character sequence of length *n* + 1 is considered as such. For instance "-fication" in English need not to be considered as a candidate suffix, since all its instances in the lexicon are also instances of "-ification".

To facilitate the identification of parent-child relationships between candidate affixes, the preferred data structure for modelling the set of candidate prefixes or suffixes is an *affix tree*[64], whose nodes are candidate affixes, associated with their lexicon occurrence counts. Within the prefix tree branch presented in Fig. 6, "sub-" and "super-" have the most obvious semantic significance and are an antonymous pair of Latin prepositions. This semantic significance coincides with a greater number of successors, and so a greater number of child prefixes. This correlation provides a first clue as to how to elucidate the semantic criterion (§3.4.1).

3.4.1 Automatic Prefix Discovery

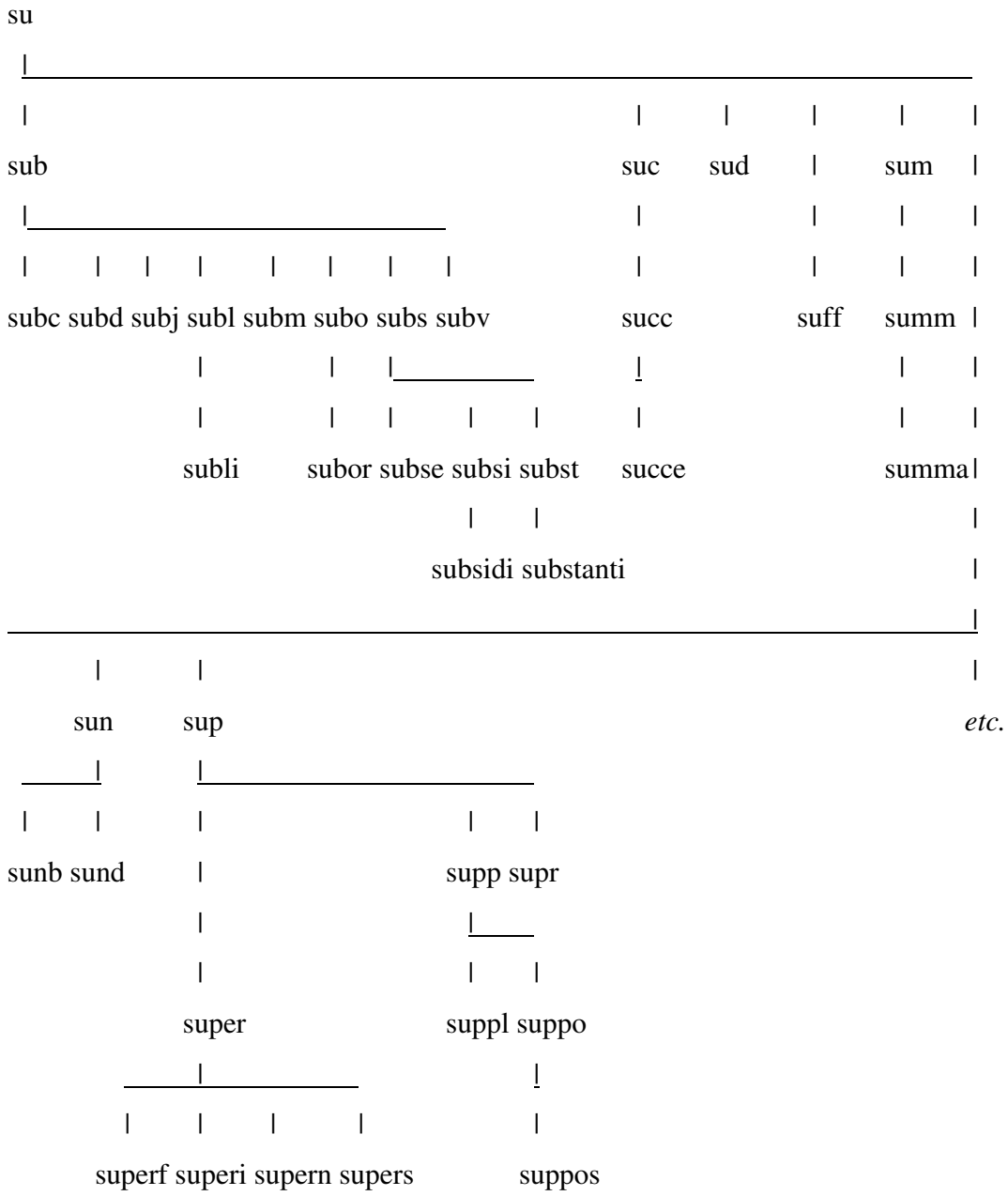
3.4.1.1 Prefix Tree Construction

At each level, a *prefix tree* is populated with candidate prefixes with one more character than at the previous level. Every possible combination of alphabetic characters at each level is looked up in the lexicon to see whether it occurs at the start of more than one word. If so then a `Prefix` object is created with that character combination. The number

[64] not to be confused with a derivational tree.

Fig. 6: Part of prefix tree rooted at "su-"

(prefix candidates with occurrence count < 10 have been omitted)



of levels was limited to 10 since at the last level no character sequences were found which occurred more than once at the beginning of a word.

The first attempt at constructing a prefix tree, branch by branch, took about 24 hours to run, because of the large number of lexicon traversals required. In order to improve efficiency the algorithm was optimised to construct each level of the prefix tree in succession, so as to minimise the number of lexicon traversals required. This added complexity but reduced runtime to about 5 seconds. A single lexicon traversal is performed for each level of the tree and the number of characters is increased at each level. At each level, all the possible character combinations are generated in the same order as they appear in the lexicon, which accounts for the improved performance. Because of the duplication criterion, candidate prefixes with only one occurrence are excluded from the tree. Candidates with only one child are deleted after constructing the tree, since their status as parents of a single child cannot be established when they are instantiated, but only on instantiation of the child.

The algorithm needs not only to find candidate prefixes but also to store information which may be relevant to determining which candidates satisfy the semantic criterion. The frequency of lexicon occurrence (as a prefix) f_c (*affix frequency*) of a candidate is obviously related to the probability of its being a valid prefix and is calculated by the prefix constructor. Also, the higher the proportion of the occurrences of its parent f_p (*parent frequency*) which is represented by a candidate, the more likely it is that it is a valid prefix.

Prefix Tree Construction Algorithm (*see also Class Diagrams 9 & 10*)

```
discoverPrefixes
{
      prefixTree = new PrefixTree();
      look up stems in lexicon;
      for (each prefix in prefixTree)
      {
            if (prefix has more than one child)
            {
                  calculate prefix. q_s ;
```

```
            }
            else
            {
                  delete prefix as irrelevant;
            }

      }
      create prefix set ordered according to a heuristic;
}

prefixTree ()
{
      root = new Prefix("");
      for each level
      {
            addLevel(root);
            while (newRoot does not exist)
            {
                  if root has child
                  {
                        newRoot = first child of root;
                  }
                  else
                  {
                        root = changeBranch(root);
                  }
            }
            root = newRoot;
      }
}

addLevel(parent)
{
      reset lexicon iterator;
      form = parent.form + "a";
      currentPrefix = new Prefix(form);
      current_prefix. f_p = parent. f_c ;
```

```
        while ((currentPrefix is not in lexicon) && (form does not end
        with "z"))
        {
              form = next possible lexical form with same number of
              characters;
              currentPrefix = new Prefix(form);
              current_prefix. f_p = parent. f_c;
        }
        if (currentPrefix is not in lexicon)
        {
              navigationalPrefix = currentPrefix; //mark for removal
        }
        make currentPrefix child of parent;
        while (currentPrefix exists)
        {
              currentPrefix = nextPrefix(currentPrefix);
        }
        if (navigationalPrefix exists)
        {
              remove navigationalPrefix
        }
}

nextPrefix(previousPrefix)
{
        valid = false;
        currentForm = previousPrefix.form;
        parentPrefix = parent of parentPrefix;
        while (not valid)
        {
              if (currentForm ends with "z")
              {
                    parentPrefix = changeBranch(parentPrefix);
                    newForm = parentPrefix.form;
                    newForm = newForm+ "a";
              }
              else
```

```
            {
                  newForm = currentForm with last letter increased;
            }
            newPrefix = new Prefix(newForm);
            newPrefix. f_p = parentPrefix. f_c;
            if (newPrefix occurs more than once)
            {
                  valid = true;
            }
            else
            {
                  currentForm = newForm;
            }
      }
      make newPrefix child of parentPrefix;
      return newPrefix;
}

changeBranch(currentPrefix)
{
      generationCounter = 0;
      rightPlace = false;
      while (not rightPlace)
      {
            nextPrefix = next sibling of currentPrefix;
            while (nextPrefix does not exist)
            {
                  currentPrefix = parent of currentPrefix;
                  increment generationCounter;
                  nextPrefix = next sibling of currentPrefix;
            }
            currentPrefix = nextPrefix;
            while (generationCounter > 0)
            {
                  currentPrefix = first child of currentPrefix;
                  decrement generationCounter;
            }
```

```
            rightPlace = true;
        }
        return currentPrefix;
}
```

Recording Stem Information

Every word beginning with a candidate prefix can be segmented into a prefix and a residue, which can provisionally[65] be considered as the stem. It might be relevant to examine whether the stem obtained by such a segmentation exists as a word in the lexicon (Hafer & Weiss, 1974; §3.3.2). To achieve this, the prefix constructor stores all the stems that occur with each prefix, and the prefix tree maintains a global alphabetic list of stems, each associated with a list of the prefixes with which it occurs. After the construction of the tree is complete, one final traversal of the lexicon is performed, to identify which of the stems exist as words in their own right within the lexicon. The proportion of the stems occurring with each prefix which are also words is then calculated and stored with the prefix as its *stem validity quotient* q_s. The data concerning stems was not analysed or evaluated initially, but proved to be a productive research direction (§3.4.4).

3.4.1.2 Heuristics to Elucidate the Semantic Criterion

Once the prefix tree has been constructed, a complete set of candidate prefixes can be obtained from it, sorted according to a heuristic intended to prioritise prefixes which satisfy the semantic criterion. Candidate prefixes can be manually evaluated, by linguistic criteria, as to whether they have meaning potential (*semantic validity*); the performance of a heuristic at prioritising candidates which satisfy the semantic criterion can be evaluated by counting the number of semantically valid prefixes occurring within the first

[65] Because of the segmentation fallacy (§3.3), such an automatic segmentation must be regarded as provisional.

n prefixes[66] returned. The affix frequency f_c is one possible heuristic. Affix frequency can also be expressed as a proportion of parent frequency f_p: the higher the proportion of f_p represented by f_c, the more likely it is that the prefix is semantically valid. So

$$\frac{f_c}{f_p}$$

is another possible heuristic. Arguably the weighting of f_c should be greater than that of f_p. So

$$\frac{f_c^2}{f_p}$$

was also tried. The stem validity quotient q_s was used in heuristics at a later stage in the research program (§3.4.4).

Applying each of the three heuristics

$$f_c, \frac{f_c}{f_p} \text{ and } \frac{f_c^2}{f_p}$$

in succession produces progressively better results in prioritising candidates which satisfy the semantic criterion. Because of this, the *default* heuristic adopted was

$$\frac{f_c^2}{f_p}.$$

This heuristic was confirmed as the best of the three by the initial results (§§3.4.1.3, 3.4.2.2) but was eventually surpassed by the others (§3.4.4)[67].

3.4.1.3 Results from Automatic Prefix Discovery

Irregular forms of prefixes can be identified by their *footprint* (§3.2.2.3). These footprints are an aid to identifying prefixes in the lexicon. The footprint is either the base form of

[66] It is not being suggested here that a threshold can be set above which any heuristic provides only valid results or below which it produces only invalid results.

[67] The fields of each prefix in a prefix set ordered by one heuristic can be written to a file in *.csv* format, with one row per prefix. This can then be re-sorted on any other heuristic in a spreadsheet application, without any need for re-construction. This facilitates comparisons of heuristic performance.

the prefix, or begins with an abbreviated or otherwise modified form of the prefix, followed by one or more characters which belong to the morpheme to which the prefix is applied. All standard modifications of prefixes can be traced back to classical Greek and Latin.

The prefix tree generated comprised 32434 candidate prefixes: the first 100, sorted on default heuristic

$$\frac{f_c^{\,2}}{f_p}$$

are listed in Appendix 16, summarised in Table 26. Candidate prefixes have been manually assessed as to whether they satisfy the semantic criterion. Appendix 16 includes the prefix footprints "imp-" for "in-" + "p", "comp-" for con-" + "p" and "app-" for "ad-" + "p". There is one clear case of a double prefix: "unre-" (= "un-" + "re-").

Table 26: Top 100 candidate prefixes

Status	Freq.
Valid	32
Invalid	59
Footprint	3
Abbreviated	5
Double	1
TOTAL	100

3.4.2 Automatic Suffix Discovery

3.4.2.1 Extension of the Algorithm to Suffix Discovery

The object-oriented approach adopted greatly facilitated the adaptation of automatic prefix discovery to suffix discovery, since `Prefix` and `Suffix` could be encoded as subclasses of the abstract superclass `Affix`, and `PrefixTree` and `SuffixTree` could be encoded as subclasses of `AffixTree` (Class Diagrams 9 & 10). The greater part of the code required is implemented as methods of classes `Affix` and `AffixTree`. In this

context, the suffix "-ation" is to be considered as a child of the suffix "-tion" whose parent is in turn "-ion".

The main challenge in adapting the algorithm to suffix discovery was that the lexicon was ordered alphabetically in normal lexicographic order, whereas what was required for suffix identification was an ordering in alphabetical order of the last letter of each word, with a secondary ordering in alphabetical order of the penultimate letter of each word and so on. This corresponds to the concept of a *rhyming dictionary*, as used by amateur poets. This needed to be generated from the lexicon.

It proved easier to generate a dictionary of reversed word forms in parallel with the generation of the lexicon, rather than deriving a rhyming dictionary from the lexicon. The lexicon is generated by collecting all the word forms from all the synsets in WordNet, adding each new word form encountered as a key associated with a pointer to its first occurrence in WordNet, and then associating an additional pointer with the key each time the same word form is encountered (§1.3.2.4). The `keyset` is automatically arranged in alphabetical order. By reversing the order of the characters within each new word form and using the reversed word form as a key within a separate data structure, it is possible to generate the dictionary of reversed word forms in parallel with lexicon generation (Class Diagram 2). Lookups in the dictionary of reversed word forms are performed simply by reversing the order of the characters of the morpheme to be looked up as part of the lookup process. This does not impact significantly on execution time of lexicon traversals. Although the dictionary of reversed forms is not identical to a poet's rhyming dictionary it is referred to henceforth, for brevity, as *the rhyming dictionary* (see §5.3.3.2 for a variation on this idea).

3.4.2.2 Results from Automatic Suffix Discovery

32817 candidate suffixes were generated: the first 100, sorted on default heuristic

$$\frac{f_c^{\,2}}{f_p}$$

are listed in Appendix 17. Any attempt to evaluate the performance of heuristics when applied to candidate suffixes by manual assessment of their semantic validity runs the risk of arbitrariness: consider the suffixes "-on", "-ion", "-tion" and "-ation": "-on" can occur as the singular inflection of words of Greek origin (plural "-a"), but in 72% of cases is part of "-ion", of which 84.72% are instances of "-tion", and of those, 78.18% are instances of "-ation" (§§3.2.2.1, 7.4.1). The rules determining the application of "-ion", "-tion" and "-ation" to form quasi-gerunds by appending them to the end of words or substituting them for one or more terminal letters are complex and require reference to Latin grammar (see italicised sections in Appendix 9; §3.2.2.1 and solution in §5.1.2).

3.4.3 Comparison of Results from Automatic Affix Discovery with Results from the Pilot Study on Morphological Rules

In order to make a less arbitrary assessment of the performance of heuristics when applied to candidate suffixes, the suffixes generated were compared to the suffixes generated by morphological rules (§3.2.2).

3.4.3.1 Undergeneration by Automatic Suffix Discovery

Table 27 shows the only suffixes listed in the rules (Appendix 10) but which were not generated by automatic suffix discovery. The data from automatic suffix discovery does not include suffixes all instances of which are also instances of the same child suffix. For instance "-fication" is not included because all the instances discovered were also instances of "-ification".

In all cases where a non-unique suffix listed in the rules is not generated by automatic suffix discovery, the child suffix is generated. Automatic suffix discovery therefore has the potential to inform the formulation of morphological rules. Deployment of heuristics will allow a systematic approach to rule formulation starting from the most important suffixes (§5.2.2.4).

Table 27: Undergeneration by automatic suffix discovery

Rule-based suffixes not generated by automatic suffix discovery	Child suffix generated by automatic suffix discovery
-fication	-ification
-ysate	*unique*
-yze	-lyze

3.4.3.2 Heuristics Tested against Morphological Rules

The suffixes generated by the full original morphological ruleset were marked in the output from automatic suffix discovery as "applied" (rules cover all instances), "partly applied" (rules cover some instances) or "not applied" (no instances covered by existing rules). The output was then sorted by each heuristic in turn and the number of suffixes applied by the rules occurring within the top 20 according to the heuristic was counted (Table 28). Adopting the morphological ruleset as a provisional benchmark for candidate suffix evaluation, these results confirmed the default heuristic

$$\frac{f_c^{\,2}}{f_p}$$

as the best of these three heuristics for discovering suffixes which conform to the semantic criterion.

Table 28: Suffixes applied by the rules occurring within the top 20 by each heuristic

Heuristic	**Applied**	**Partly applied**	**Not applied**	**Invalid**	**TOTAL**
f_c	6	0	2	12	20
$\frac{f_c}{f_p}$	2	0	0	18	20
$\frac{f_c^{\,2}}{f_p}$	9	3	2	6	20

Table 29: First 100 prefixes by 3 heuristics

Heuristic	$\frac{f_c^2}{f_p}$	$\frac{f_c^2 q_s}{f_p}$	$\frac{f_c^2 q_s^2}{f_p}$
Valid	32	60	47
Invalid	59	5	1
Footprint	3	1	0
Abbreviated	5	1	1
Double	1	1	0
Concatenation	0	31	50
Irregular	0	1	1
TOTAL	100	100	100

Table 30: Top 20 candidate prefixes sorted on $\frac{f_c^2 q_s}{f_p}$

Prefix	$\frac{f_c^2}{f_p}$	$\frac{f_c^2 q_s}{f_p}$	$\frac{f_c^2 q_s^2}{f_p}$	Validity
un	1936.56	1514.81	1184.91	Valid
in	1084.73	413.96	157.98	Valid
re	836.27	320.31	122.68	Valid
over	269.09	253.38	238.58	Valid
non	218.55	205.80	193.80	Valid
dis	361.59	204.83	116.03	Valid
de	486.61	154.70	49.18	Valid
out	136.64	107.63	84.78	Valid
inter	170.28	93.81	51.68	Valid
under	105.26	92.83	81.87	Valid
super	123.01	77.38	48.67	Valid
counter	81.10	77.24	73.56	Valid
anti	98.56	63.67	41.13	Valid
micro	83.01	61.27	45.22	Valid
semi	66.67	60.00	54.00	Valid
pre	136.45	56.80	23.64	Valid
trans	152.91	53.07	18.42	Valid
con	282.04	52.17	9.65	Valid
s	601.53	48.87	3.97	Invalid
photo	56.15	48.53	41.95	Valid

3.4.4 Additional Heuristics

In an attempt to improve the results from automatic affix discovery, the stem validity quotient was introduced into new heuristics on the principle that the greater the stem

validity quotient (q_s), the more likely the affix is to satisfy the semantic criterion. With no known theoretical precedent and no preconception regarding the weighting of q_s, heuristics

$$f_c\, q_s,\ f_c^{\,2} q_s,\ \frac{f_c\, q_s}{f_p},\ \frac{f_c^{\,2} q_s}{f_p} \text{ and } \frac{f_c^{\,2} q_s^{\,2}}{f_p}$$

were all experimentally applied. Of these,

$$\frac{f_c^{\,2} q_s}{f_p} \text{ and } \frac{f_c^{\,2} q_s^{\,2}}{f_p}$$

produced results (Table 29) significantly better at prioritising semantically valid prefixes than those previously achieved. Invalid prefixes and footprints were almost eliminated from the top 20, but a large number of concatenations appeared. The three best performing heuristics illustrated in Table 29 show advantages for each:

- $\frac{f_c^{\,2} q_s}{f_p}$ performs best for finding valid prefixes;
- $\frac{f_c^{\,2}}{f_p}$ performs best at distinguishing between prefixes and concatenations;
- $\frac{f_c^{\,2} q_s^{\,2}}{f_p}$ gives fewest semantically invalid results.

The top 20 prefixes according to heuristic $\frac{f_c^{\,2} q_s}{f_p}$ are listed in Table 30.

Table 31: Top 20 candidate suffixes by 3 heuristics

Heuristic	Rule applied	No rule identified	Rule applies to child	Invalid	TOTAL
$\frac{f_c^{\,2}}{f_p}$	12	3	5	0	20
$\frac{f_c^{\,2} q_s}{f_p}$	13	4	3	0	20
$\frac{f_c^{\,2} q_s^{\,2}}{f_p}$	0	1	0	19	20

Table 32: Top 20 candidate suffixes sorted on $\frac{f_c^2 q_s}{f_p}$ [68]

Suffix	$\frac{f_c^2}{f_p}$	$\frac{f_c^2 q_s}{f_p}$	$\frac{f_c^2 q_s^2}{f_p}$	**Morph. rule**
ing	2498.66	69.67	1.94	Yes
er	2958.42	63.56	1.37	Yes
e	2607.03	36.63	0.51	No
ed	2054.22	29.82	0.43	Yes
ate	809.39	23.50	0.68	Yes
ation	1260.21	21.89	0.38	Yes
al	1252.90	21.13	0.36	Yes
able	693.53	20.92	0.63	Yes
ic	1988.63	19.63	0.19	Yes
ion	1748.11	19.39	0.22	Child
on	1625.66	19.19	0.23	Grand-child
ine	353.63	18.10	0.93	No
ight	108.00	18.00	3.00	No
ent	574.72	16.76	0.49	Yes
ble	593.96	16.46	0.46	Child
ive	584.49	16.28	0.45	Yes
age	164.15	16.25	1.61	Yes
ism	732.70	14.31	0.28	Yes
like	190.02	14.21	1.06	No
ly	1285.72	14.09	0.15	Yes

The morphological ruleset was again adopted as a provisional benchmark for candidate suffix evaluation (§3.4.2.2). The performance of heuristic

$$\frac{f_c^2 q_s^2}{f_p}$$

deteriorated dramatically when applied to suffixes, while

$\frac{f_c^2 q_s}{f_p}$ remained competitive, outperforming $\frac{f_c^2}{f_p}$ (Table 31).

This indicates that the optimal weighting of the stem validity quotient is less for suffixes than for prefixes, which is consistent with the view that suffixations cannot be as readily segmented as prefixations (see §3.3 on the problems of segmentation and §3.2.3 for the

[68] The use of the original morphological ruleset as a benchmark for heuristic evaluation gave these results. This does not imply that the suffixes missing from that ruleset are invalid. For subsequent extensions to the ruleset see §5.1.

sufficiency of general spelling rules for prefix stripping; see also Appendix 9 for many cases where the root of a suffixation cannot be found by segmentation). The top 20 suffixes according to heuristic

$$\frac{f_c^{\,2} q_s}{f_p}$$

are listed in Table 32. These results were presented to the LTC 2009 Conference (Richens, 2009b).

3.4.5 Conclusions on Automatic Affix Discovery

An automatic approach to affix discovery has been demonstrated. The best heuristics for prioritising candidate suffixes according to the semantic criterion have been identified as

$\frac{f_c^{\,2}}{f_p}$ (the default heuristic) and $\frac{f_c^{\,2} q_s}{f_p}$.

The results from automatic prefix discovery show advantages for each of the heuristics

$\frac{f_c^{\,2}}{f_p}$, $\frac{f_c^{\,2} q_s}{f_p}$ and $\frac{f_c^{\,2} q_s^{\,2}}{f_p}$.

The main advantage of the default heuristic

$$\frac{f_c^{\,2}}{f_p}$$

is that it performs best at distinguishing between prefixations and concatenations. It was expected to be relatively straightforward to develop an algorithm to filter out concatenations from the input data prior to running the Automatic Prefix Discovery Algorithm (but see §5.3.4.2). Assuming that this is feasible in practice, it would appear that the *optimal* heuristic for application to both prefix and suffix stripping is

$$\frac{f_c^{\,2} q_s}{f_p}.$$

This will be the heuristic used in primary affixation analysis (§§5.3.7, 5.3.11) though the default heuristic will also be used in secondary affixation analysis (§§5.3.14, 5.3.16).

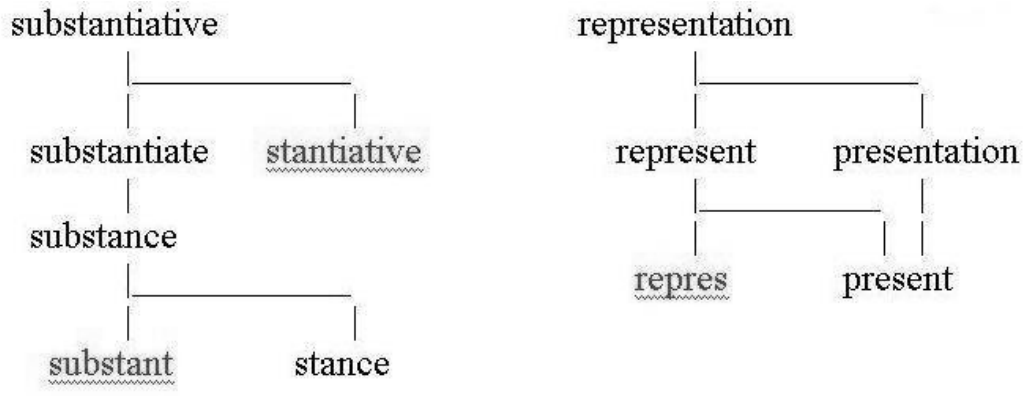
3.5 Final Considerations Prior to Morphological Analysis and Enrichment

3.5.1 Affix Stripping Precedence

One consequence of the difference between typical prefixation and typical suffixation (§3.2.3) is that it provides a guide to the affix stripping precedence rules to be applied when analysing the derivation of a word which has both prefix and suffix. Suffix stripping needs to be conducted first, so that the prefixed residue of the de-suffixed word can be posited as the root of the corresponding derivational tree, each member of which will have the same prefix. Only from that root can dual inheritance be allowed in further tracing the dual derivation of the root, which is common to the entire tree (§3.2.3).

To illustrate this principle (Fig. 7) take the word "substantiative". By removing the suffix "-ive", we get "substantiate". Substituting "-ce" for its derivative "-tiate" we get "substance", the parent of "substantiate" in the derivational tree. Substituting "-nt" for its

Fig. 7: Derivational trees illustrating affix stripping precedence



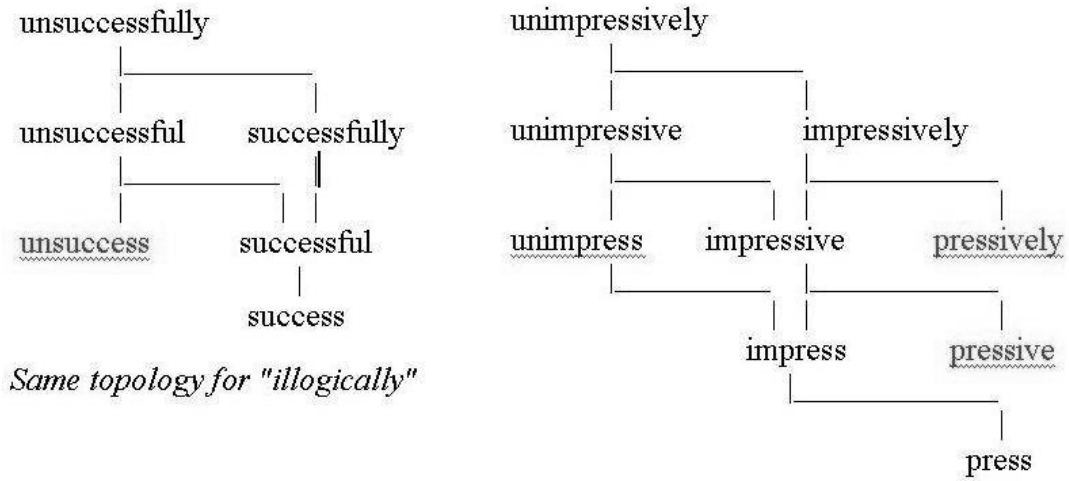
derivative "-nce" we get "substant", which is not lexically valid, so "substance" is the root of the tree. Then the prefix "sub-" may be separated from the stem "stance" which is a

morpheme conveying a meaning related to but not identical to the word "stance". However if we attempt prefix stripping first, we get "sub-" and "stantiative", which is not lexically valid and we miss the morphosemantically related terms "substantiate" and "substance" altogether.

Similarly with the word "representation" (Fig. 7), if one removes the prefix "re-" first, one will get the word "presentation". If suffix "pre-" is then removed we get "sentation" which is not lexically valid. Moreover "presentation" is semantically more remote from "representation" than the word "represent" which will be generated by giving precedence to suffix stripping. The word "present" would then be generated. It also would be generated by giving precedence only to the first prefix followed by the first suffix.

When we look at antonymous prefixations, we find a different scenario (Fig. 8). With the word "unsuccessfully", if suffix stripping takes precedence we get "unsuccessful" and then the lexically invalid word "unsuccess", and we miss the related words "successfully", "successful" and "success". If, on the other hand, antonymous prefix

Fig. 8: Derivational trees illustrating affix stripping precedence with antonymous prefixes



removal takes precedence, we get "successfully". Giving priority to suffix stripping over non-antonymous prefix stripping, we then get "successful" and "success". We miss the valid term "unsuccessful", but we arrive at the root word. Similarly with "unimpressively", if suffix stripping takes precedence we get "unimpressive", then "unimpress", which is only ever used as the participle "unimpressed" and we miss four related words, but if antonymous prefix stripping takes precedence we get "impressively" and, again prioritising suffix stripping over non-antonymous prefix stripping, we then get "impressive" and "impress". Finally non-antonymous prefix stripping may occur to give the root word "press", missing the valid term "unimpressive". The loss of the connections between "unsuccessfully" and "unsuccessful" and between "unimpressively" and "unimpressive" is unfortunate[69], but giving precedence to suffix stripping in this context would result in more connections being lost. So the precedence rule will be adopted that removal of antonymous prefixes should have the highest precedence, followed by suffixes, followed by non-antonymous prefixes. When finding morphological relations by synthesis (as in §3.2.2.2.1) rather than analysis (as in §3.2.2.2.2), the precedence rules will obviously be reversed.

3.5.2 Compound Expressions and Concatenations

Little attention has been given in this study so far to the morphological relations between multiword expressions and hyphenations (together referred to as *compound expressions*; §5.3.2) and concatenations and their components. Because of their regular lexical properties, in theory it should be much easier to identify these than the relations implied by affixation (but see §5.3.4.2). Their derivation from their components is self-evident and neither conforms to, nor requires, the application of morphological rules. There is, however, scope for the integration of their morphological relationships within a lexical database. Concatenations whose constituents are all nouns are likely to be HYPONYMS or MERONYMS of the last of the nouns.

[69] but it will still be possible to navigate the indirect connection through the derivational tree.

Table 33: Prefixations corresponding to verbal phrases

(Suffixes are shown in italics.)

Word form	**Verbal phrase**
ex-it	go out
in-come	come in
in-vade	go in
out-set	set out
sur-vive	live on
up-heave	heave up
pre-vis-*ion*	see before
com-pute-*r-ise*	think with
de-scrip-*tion*	write down
ex-tract-*able*	drag out
im-port-*ation*	carry in
ex-tort-*ion-ist*	twist out
over-estimate	estimate over
trans-miss-*ion*	send across
com-memor-*ative*	remember with
pre-determine-*d*	determine before
trans-ship-*ment*	ship across

A particularly important kind of multiword expression is a verbal phrase, whose constituents are a verb and a preposition or adverb (§2.3.1.2 & note). Provided that prepositions are first added to WordNet, there is also scope for enrichment by establishing relations between verbal phrases and their constituents. Many prefixations comprise a prepositional prefix and a verbal stem (§3.2.3). These correspond to verbal phrases. The examples in Table 33 occur among the prefixed forms in the random word list (§3.2.2.2.1). They include examples of English, French and Latin preposition-verb combinations. The last example is a verb, not derived from Latin, but prefixed by a Latin preposition. The Latin preposition-verb combinations were in many cases already combined in classical Latin, but the processes of Latin and Greek prefixation, obeying the same spelling rules (§§3.2.2.3, 3.4.1.3), still occur today in coining scientific vocabulary.

No precedence rules have yet been established with regard to de-concatenation. It is tentatively assumed that de-concatenation should take precedence over affix stripping (but see §5.3.4.2) since the products of de-concatenation, by definition are always words in their own right which may themselves include affixes, whereas affixes are atomic, unless one considers concatenations of affixes to be affixes in their own right.

3.5.3 Implications of WordNet Granularity for Lexical Database Enrichment

There is plenty of scope for enriching WordNet with data relating to derivational morphology. The Java model of WordNet (§1.3.2) is a firm foundation for implementing and demonstrating this enrichment. However the structure of WordNet raises questions about how best to do this. As it stands, existing morphological data is encoded as derivational pointers, whose directionality does not necessarily reflect the directionality of derivation. These pointers link word senses rather than the words themselves.

The ambiguity of words presents an obstacle to the correct automatic encoding of morphological relations (§3.2.1), but the fine grain of WordNet aggravates the problem by exaggerating the extent of ambiguity (Peters et al., 1998; Vossen, 2000; §2.1.2). Much manual intervention would be required, unless exaggerated ambiguity is reduced by an optimal pre-clustering.

A review of clustering algorithms (§2.1.2.3) raises the question of which clustering criterion would be optimal for the task in hand. The optimal clustering for the encoding of morphological relations is necessarily a *lexical* clustering, which merges different senses of the same word which have the same POS. In the vast majority of cases in WordNet, such senses are derivationally identical. The results from the pilot study suggest that most semantically unrelated homonyms are *monosyllables* (§3.2.2.2.3), which can be treated with extra caution (§3.2.3); the ambiguities of *polysyllabic* words are usually cases of polysemy (Apresjan, 1973; Pustejovsky, 1991; §2.1). Lexical clusters, just like synsets, are sets of word senses, but they are grouped by word form instead of meaning (§1.3.2.4). Just as a word sense can only ever belong to a single synset, so it can only ever belong to a single lexical cluster. Lexical clusters cannot overlap with each other and nor can synsets. Lexical clusters and synsets can and do however frequently overlap with each other.

A lexicon, by definition, exhibits a lexical clustering of word senses. Although the WordNet model has been adapted to accommodate synset clusters (Class Diagram 3), it is vastly more economical, in terms of both computer memory and human time to optimise the lexical clustering by modifying the original model (Class Diagram 2) to create a new model (Class Diagram 7; Appendix 1) where a distinction is made between a `GeneralLexicalRecord` and a `POSSpecificLexicalRecord`, with the `GeneralLexicalRecord` for each word encapsulating a separate `POSSpecificLexicalRecord` for each POS of that word. This achieves the optimal clustering, without the need to implement synset clusters.

As the revised lexicon design (Class Diagram 7) represents the optimal clustering of word senses for morphological analysis and enrichment, relations discovered through morphological analysis are to be encoded as *lexical* relations in the lexicon component rather than as semantic relations in the wordnet component of the model. So *morphological* relations will be referred to henceforth as *lexical* relations. Since each `WordSense` in the model specifies a word form and POS and since each `LexicalInformationTuple` (now encapsulated within a `POSSpecificLexicalRecord`) specifies the corresponding synset identifiers and word numbers, it is possible to navigate any combination of WordNet relations between synsets and lexical relations between `POSSpecificLexicalRecord`s, given that all relations are encoded bidirectionally (§1.3.2.2). Such an approach does not preclude the specification of semantic types for the morphological relations. Moreover, it will provide another decisive advantage: neither morphological analysis nor enrichment with morphological relations need refer directly to WordNet, but only to the lexicon; either the morphological analyser itself or the relations discovered will then be portable, with a minimum of modifications, to entirely independent digital lexica (§5) without the identified shortcomings of WordNet (§2).

3.5.4 Conclusion: A Hybrid Model

The rule-based approach to morphological analysis, subject to the considerations expressed in §3.2.3, has the potential to identify the relation types of many morphosemantic relations between suffixations and between suffixations and their roots, without succumbing to the segmentation fallacy. Any set of morphologically related suffixations with a common root, together with the morphosemantic relations between them, forms a derivational tree in which both the direction of derivation and the semantic or syntactic type of each relation can be determined.

However, in order to be applied in a non-arbitrary manner, the rule-based approach needs to apply converse morphological rules to suffixes pre-identified by automatic suffix discovery. The rule-based approach is not applicable to prefixations, other than antonymous prefixations. Automatic prefix discovery will identify prefixes, but a methodology for its application in prefixation analysis still needs to be established (§5.3.11). Automatic affix discovery with suitable heuristics can ensure that morphological analysis reflects empirical data rather than being governed by theory.

The deployment of effective heuristics for candidate affix selection according to the semantic criterion will maximise the *unsupervised* automatic component of morphological analysis, while minimising the *supervised* manual refinement component. The heuristic-driven prioritisation of candidate suffixes from automatic suffix discovery can be used to inform the formulation of morphological rules applying to suffixations (§5.2.2.4). This will lay the foundation for a *hybrid* model, fed only with empirical data, collected in an unsupervised manner, but interpreted syntactically and semantically. The interpretation must be sufficiently supervised to capture exceptions, in order to ensure a high quality outcome. More generalised spelling rules for prefixation can be extrapolated from the data from automatic prefix discovery. The affix stripping precedence rule established in §3.5.1 can be applied by conducting antonymous prefixation analysis first, followed by suffixation analysis, followed by non-antonymous prefixation analysis. The

assumed precedence of concatenation analysis over all these (§3.5.2) is tentative and needs to be exercised with extreme caution (§5.3.4).

Within a hybrid model, relations based on derivational morphology can be identified by analysing words in the lexicon iteratively into their components. Care needs to be taken to ensure that no affix is removed before establishing that it is not in fact part of a longer affix. This can be achieved by examining child affixes within the affix tree before removing the parent affix. The reverse approach, of attempting to construct longer words from components would generate a much greater number of non-existent words, and in any case is not feasible, because while lists of candidate affixes have been produced, a list of stems cannot be produced without first undertaking the analytical approach. Enrichment of the lexicon component of any lexical database with the morphological relations identified from within it can be accomplished through the encoding of lexical relations between words in the lexicon as indicated in §3.5.3. The enrichment of the lexicon component of the WordNet model will create a morphosemantic wordnet.

4 Adaptations of the WordNet Model Prior to Morphological Enrichment

This chapter takes up the conclusions at the end of §2.4, regarding limited improvements to the WordNet model to be implemented prior to morphological analysis and enrichment. Although extensive possible improvements have been identified, only those which can be achieved by a largely automated process are to be adopted. In order to be complete, a lexical database should include all eight parts of speech (§1.1.4), of which WordNet contains only four[70]. Because *prepositions* are the most numerous part of speech after these four, and because of their relevance to the morphology of many concatenations and prefixations, the addition of prepositions to WordNet and the creation of a preposition taxonomy were priorities. The remaining improvements proposed are modifications to the relations and the elimination, by automatic methods as far as possible, of disconnected proper nouns.

4.1 Proposed Modifications

4.1.1 Encoding of Prepositions

Prepositions are "the set of items which typically precede noun phrases . . . to form a single constituent of structure" (Crystal, 1980). There are no prepositions in WordNet. Jackendoff (1983) uses the concept of *intransitive* preposition for words like "forward" and for adverbial homographs of prepositions which others prefer to call *particles*[71]. The term *intransitive preposition* conflicts with the morphology of the word preposition and is not mentioned by Crystal (1980). Such words are considered by traditional grammar, and will be considered here as *adverbs*. Many prepositions double as adverbs (or have transitive and intransitive uses) and so some are found in WordNet as adverbs.

[70] nouns, verbs, adjectives and adverbs.
[71] Both terms are avoided in this thesis, the set of 8 traditional parts of speech being preferred (§1.1.4).

Prepositions play an important part in the formation of *prefixes*, which are one of the major constituents of morphology (§3.2.3) and a key role in the identification of sentence frames (§2.3.1) and in the derivational morphology of verbal phrases (§3.5.2). Consequently the completion of the project depends on encoding prepositions, which will fulfil the most immediate need for enriching WordNet.

4.1.2 Pre-cleaning of Data

The next most immediate task is to clean out irrelevant and erroneous data, as far as this can be done quickly and automatically. A lexical database is not an encyclopaedia, and it is not helpful to include arbitrary and subjective encyclopaedic information in it in an attempt to answer questions like "Who is a genius?" (§2.2.2.2.6). Proper nouns are to be excluded, except where they are connected to other nouns by valid[72] semantic relations. A secondary, pragmatic reason for giving priority to this task was to limit the memory requirements of the model, so as to avoid memory shortage during morphological enrichment.

4.2 Enrichment of the WordNet Model with Prepositions

This section starts by reviewing some theoretical discussions and research concerning prepositions, especially The Preposition Project (Litkowski & Hargraves, 2005; http://www.clres.com/prepositions.html; hereafter *TPP*). Attention is focussed on the relations between prepositions, a consideration relevant to constructing a preposition taxonomy. The enrichment of the WordNet model with prepositions, using data from TPP, is then described in detail. For consistency with WordNet, synonymous prepositions are grouped into synsets. Identification of preposition synonyms is governed by TPP data, except for a few ambiguities. The construction of the preposition taxonomy was initially based on the TPP taxonomy of semantic role types, but at a higher level, a lexically

[72] for the criteria see §4.3.4.

driven taxonomy, implied by Jackendoff (1983) and reflecting more subtle relationships between preposition meanings, has been superimposed on the taxonomy implicit in the data.

4.2.1 Background

4.2.1.1 The Syntactic Role of Prepositions

Jackendoff (1983) argues that temporal ordering is mentally represented in spatial terms. He goes on to demonstrate that the same polysemous verbs are frequently used in the same syntactic frames to refer to several of the semantic fields place, time, possession, identification, circumstance and existence. He also makes an important distinction between different types of *path* expression:

1. Bounded paths: where a source or a goal is expressed by "from" or "to" such that the reference object is an endpoint of the path.
2. Directions: where a source or a goal is expressed by "away from" or "towards") such that the reference object is *not* an endpoint of the path.
3. Routes: where the path is expressed by a preposition such as "via", "along" or "through" and no endpoint is expressed.

A direction is less specific than a bounded path: if one goes "to" a place, one also goes "towards" it, but not vice versa. This means that "to" is a HYPONYM of "towards" and "from" is a HYPONYM of "away from".

These observations are relevant to the creation of a preposition taxonomy (§§4.2.1.6, 4.2.4). Such a taxonomy needs to capture the relationships between the uses of prepositions such as "from" and "to" as expressions of space and of time (§4.2.4.2). While the spatial sense may well be the original sense, as Jackendoff argues, neither is in fact a generalisation of the other. A lexical taxonomy is required where abstract, generic meanings of such prepositions are the HYPERNYMS, of which spatial, temporal and other uses are HYPONYMS and where bounded paths are HYPONYMS of directions (§4.2.4.3; Appendix 26).

4.2.1.2 Summary of Recent Research

Baldwin et al. (2009) summarise recent research into the computational handling of prepositions. They note that different approaches to NLP have widely divergent attitudes towards prepositions ranging from the extreme of treating them as *stop words* to be ignored to a full semantic treatment. They point out that 4 of the 10 most frequent words in the BNC are prepositions.

They follow Jackendoff's (1983; §4.2.1.1) distinction between transitive and intransitive prepositions, categorising intransitive prepositions as either *particles* usually forming the non-verbal component of a verbal phrase (considered in this thesis as adverbs), copular predicates as in "the doctor is *in*" and prenominal modifiers as in "an *off* day". These latter 2 usages are considered here as adjectives.

They go on to summarise 25 years of research into *attachment ambiguity*, the problem of whether a prepositional phrase is governed by a verb or by one of its nominal arguments, which is a major cause of parser error. Selectional restrictions on the object of the preposition may provide a clue to resolving such ambiguities. The most promising results seem to be achieved by post-processing of parser output. The intractable nature of this problem has been a factor motivating the classification of verbs according to the frames which they share (Kipper et al., 2004). Noting that WordNet and its derivatives (EuroWordNet, BalkaNet, HowNet etc.) focus on *content words*, they conclude (p.137) that the "time seems right to develop preposition sense inventories for more languages". The challenge for English has already taken up by Litkowski & Hargraves (2005; 2006, §4.2.1.4), but the present project is the first attempt to include prepositions in a version of WordNet.

4.2.1.3 Identification of Preposition Hypernyms

Litkowski (2002) examines the definitions of prepositions, including prepositional multiword expressions, in NODE (1998). These are mainly of two types: non-substitutable definitions which describe the usage of a sense of a preposition and substitutable definitions which in turn subdivide into those comprising participles (e. g. "overlooking" for a sense of "above") and those which end with a preposition (e. g. "on every side of" for "around"; "on the subject of" for "about"). The final preposition in these cases is considered as the HYPERNYM of the preposition being defined. He then performs digraph analysis on the dictionary, as described by Blondin-Massé et al. (2008)[73], treating the verbs corresponding to the participles, or the final prepositions in the definitions, as the HYPERNYMS of the preposition senses being defined. A single round of digraph analysis on NODE eliminated 309 out of 373 entries. The remaining 64 are classified into 25 groups, regarded as "strong components", used in the definitions of other prepositions, reducible by iterative digraph analysis to a grounding kernel of 8 "primitives", which are not defined in terms of other prepositions or participles (Appendix 23).

Table 34: Disambiguation of preposition definitions (after Litkowski, 2002)

Preposition defined	**Definition**	**Final preposition**	**Final preposition sense**
after	in imitation of	of	deverbal
on behalf of	as a representative of	of	partitive
like	characteristic of	of	predicative deverbal

An analysis which identifies the senses of the final prepositions being used and not just their word forms requires disambiguation of the final prepositions, of which "of" is the most frequent (175 instances in NODE) and also the one with most senses in any dictionary (60 in OED1 (1971-80), not including subsenses). Table 34 shows some of Litkowski's disambiguations, in terms of the 9 senses of "of" in NODE. "In imitation of" is *deverbal* because the object of the preposition (both original and HYPERNYM) is the

[73] The methodology described by Blondin-Massé et al. is possibly more sophisticated.

object of the verb "imitate". The assignation of *partitive* to "as a representative of" is an unfamiliar extension of the concepts of whole and part. Litkowski suggests that a verb taxonomy can be used to find the indirect HYPERNYMS of prepositions defined by participles. The WordNet verb taxonomy is unfortunately not consistent enough for this task (§2.2.2.2).

4.2.1.4 The Preposition Project (TPP)

The Preposition Project (Litkowski & Hargraves, 2005; http://www.clres.com/prepositions.html) finds prepositions in the FrameNet corpus (Ruppenhofer et al., 2006) using FrameNet Explorer (http://www.clres.com/FNExplorer.html). The prepositions are then disambiguated into their senses in ODE (2003), later replaced (Litkowski & Hargraves, 2006) by NODE (1998). The syntactic functions of the prepositions are identified and intuitively assigned to semantic roles, independently of linguistic theories, with the intention of creating a resource useful for NLP[74]. The dictionaries were chosen for their organisational clarity and because of their reliance on corpus evidence. The main other resource used is Quirk et al. (1985), principally for identifying other prepositions which are used in similar ways to a given preposition. The authors consider that all 3 resources are incomplete in their coverage of prepositions but that by combining them in this way they can arrive at a comprehensive resource.

Different verbs prefer different prepositions but the same preposition may occur as a dependent of the same verb with a different *frame element* being assigned to its object (e. g. "arrive by" may be followed by a *Mode_of_transportation* or a *path* element) and with different synonyms ("in" and "via" respectively). Litkowski & Hargraves have used FrameNet Explorer to discover other such alternative syntactic realisations (e. g. "enter through"). The number of such alternative realisations which are not recorded in any dictionary was found to be unexpectedly great. The granularity of FrameNet frame

[74] While this approach appears quite different to that previously adopted (§4.2.1.3), the resultant taxonomy is similar (§4.2.1.5). Hence digraph analysis was not required for developing the preposition taxonomy described in §4.2.4.

element names is much finer than traditional thematic roles (Fillmore, 1968) and these names have often been preferred in assigning names to the semantic role types.

Because TPP is the most systematic computational resource available on prepositions, the data from TPP (http://www.clres.com/prepositions.html) has been chosen for use in this project as the basis for adding prepositions to the WordNet model (§4.2.2).

4.2.1.5 Inheritance of Preposition Senses

Litkowski & Hargraves (2006) discuss the coverage of TPP and the semantic inheritance of particular preposition senses from more general senses. As regards coverage, the semantic roles assigned are found to cover several established introspectively derived lists of semantic roles, though TPP roles are finer-grained and many of these are absent from Quirk et al. (1985).

The initial analysis of inheritance started from considering the final preposition in the definition of another preposition as candidate HYPERNYM for the preposition defined (Litkowski, 2002; §4.2.1.3). This resembles the approach to identifying HYPERNYMS from glosses widely employed in the construction of WordNet (§2.2.2.2.6), and presupposes some definition of HYPERNYM other than "is a", which is clearly inapplicable to prepositions. Litkowski & Hargraves (2006) propose a definition (p. 41) taking the form of the *hypothesis*: "the semantic relation name and the complement properties of an inherited sense are more general than those of the inheriting sense". Most of the inherited senses could be disambiguated; of those which could not, it is notable that some were regional variations such as Scots "*frae*" for "*from*". Such cases will be treated here as synonymous, so that "frae" is a synonym of *every* sense of "from" (§4.2.3.1).

The high level of consistency found, where treating the disambiguated sense of the final preposition as the HYPERNYM yielded a sense where the semantic relation type and complement properties of the HYPERNYM were generalisations of those of the HYPONYM corroborates the digraph analysis methodology.

4.2.1.6 Other Considerations for a Preposition Taxonomy

Jackendoff (1983; 1990; §4.2.1.1) demonstrates clear parallelisms between the usages of identical prepositions in different semantic roles, which suggests that, in the case of prepositions, lexical distinctions are more fundamental than distinctions between semantic roles. This strong evidence of common properties of all senses of most prepositions motivated the more lexically driven approach to preposition taxonomy adopted here (§4.2.4).

Litkowski & Hargraves (2006) advocate the implementation of a WordNet-like network for prepositions. The development of such a resource, integrated with the WordNet model used in this research project, takes the TPP file[75] as a starting point (§4.2.2). The initial criterion adopted here for identifying preposition HYPERNYMS is based on the classification of semantic roles into *superordinate taxonomic categories* encoded in the TPP taxonomy files. If the superordinate taxonomic categorizer of a preposition sense *a* is the semantic role type of a preposition sense *b*, then *b* is the HYPERNYM of *a* if the synset representing *b* contains all the word forms in the synset representing *a*. However an overriding priority is given to *lexical* inheritance.

One of the main purposes for encoding prepositions was to enable automatic mapping from prefixes to the prepositions representing their meanings (§§4.2.4, 5.3.11). This meant that a generalisation of all the senses of each preposition was considered at the outset to be a requirement. To do this automatically would require a generic representation of the preposition, as choosing the correct semantic role type would require manual intervention. This was an additional reason for giving priority to lexical inheritance. In the end, the decision to encode morphological relations in the lexicon rather than in the wordnet (§3.5.3) meant that this requirement for a generic representation was fulfilled by the `POSSpecificLexicalRecord` (Appendix 1) for the preposition rather than by any `PrepositionalSynset`.

[75] *tpp.xml* (latest version by courtesy of Ken Litkowski).

4.2.2 Loading the Preposition Data[76]

The `PrepositionLoader`[77] encapsulates a main preposition map[78], each entry in which maps from a preposition word form to a `PrepositionRecord` list in which each `PrepositionRecord` represents a sense of that preposition word form. Within each `<entry>` element in the TPP file, there is a single `<hw>` (headword) element indicating a preposition word form and one or more `<S>` (sense) elements representing its senses. For each `<S>` element within each entry, the `PrepositionLoader` creates a `PrepositionRecord` assigning values to its fields from xml elements (Appendix 24). The `PrepositionRecord` is added to the main preposition map, indexed by its headword as a key.

The `PrepositionLoader` encapsulates sets of possible values for certain corresponding fields of any `PrepositionRecord`, which are determined by the text content of the corresponding XML element. These sets have been written to the files indicated in Table 35. The term *superordinate taxonomic categorizer* refers to a taxonomic category of *semantic role types*.

Table 35: `PrepositionLoader` fields, XML elements and files

`PrepositionRecord` field	**XML element**	**Output file**
semanticRoleType	`<srtype>`	*semanticRoleTypes.txt*
superOrdinateTaxonomicCategorizer	`<sup>`	*superOrdinateTaxonomicCategorisers .txt* (Appendix 25)
relationToCoreSense	`<srel>`	*relationToCoreSenses.txt*

[76] The ensuing description of the encoding of prepositions has been meticulously annotated here in the belief that wordnet construction should be thoroughly documented and that the documentation should be accessible to the research community.

[77] A new instance of `PrepositionLoader` is created, which parses file *tpp.xml* (the latest version obtained from Ken Litkowski) and outputs the copyright message. A new instance of `PrepositionalTaxonomyBuilder` is created, sharing the main preposition map of the `PrepositionLoader`.

[78] `Map<String, List<PrepositionRecord>>`

4.2.3 Prepositional Synonym Identification

4.2.3.1 Spelling Variants

Some monosemous preposition headwords are spelling variants of other polysemous preposition headwords[79], where the full range of senses is not listed but there is a single `<S>` (sense) element.[80]. Every `PrepositionRecord` corresponding to one of these monosemous headwords is removed from the main preposition map and a `PrepositionRecord` list is obtained from its synonym[81]. Each `PrepositionRecord` listed is cloned and the clone's word form is changed to that of the monosemous preposition. The clone is added to the valid synonyms field of the `PrepositionRecord` cloned and the `PrepositionRecord` cloned is added to its clone's valid synonyms.[82].

4.2.3.2 Encoded Synonyms

The TPP file specifies which synonym headwords are synonyms of each preposition sense, but does not specify which sense of a synonym is the synonymous sense. As synonyms must necessarily have a common semantic role type, synonym identification can be performed by comparing the semantic role types of each `PrepositionRecord` representing the sense of one preposition with those of each `PrepositionRecord`

[79] as for instance "frae" is synonymous with "from" (§4.2.1.5).
[80] In these cases, typically the text content of either the `<cprop>` (*complement properties*) element or the `<srtype>` (*semantic role type*; §4.2.1) element refers to the other preposition, the text content of element `<sup>` (*superordinate taxonomic categorizer*) is "Tributary" and the content of the `<srel>` (*relation to core sense*) element either is "informal sound spelling." or starts with "core: " (file *uniquePrepositionSenses.txt).*
[81] In such cases, because of some inconsistencies in the encoding, two separate `PrepositionRecord` lists are made for the polysemous headword: one list comprises every `PrepositionRecord` mapped to from the headword contained in the complement properties field of the monosemous preposition's `PrepositionRecord`, with the prefix "SEE " removed; the other list comprises every `PrepositionRecord` mapped to from the headword contained in the semantic role type field of the monosemous preposition's `PrepositionRecord`, with the prefix "ALL_" removed. These fields have been converted to uppercase to mask inconsistencies. If the word forms obtained from the two fields of the monosemous preposition's `PrepositionRecord` are the same, then only one list is used; if one list is empty then the other is used; otherwise the intersection of the two lists is used.
[82] The modified clones are written to the variant spellings field of the `PrepositionLoader`. Summaries of the fields of all the monosemous prepositions to which this procedure is applied have been written to file *uniquePrepositionSenses.txt.*

representing its synonym. This leaves fewer ambiguities than comparing superordinate taxonomic categorizer fields, and can be confirmed by comparing synonym fields to ensure that the word form of each is listed as a synonym of the sense of the other.

Each sense of each synonym of each sense of each preposition[83] is examined to see if the semantic role types of the two senses are identical. If a single synonym sense is found for any preposition sense with an identical semantic role type and each headword is listed as a synonym of the other sense, then the `PrepositionRecord` representing that synonym sense is added to the valid synonyms field of the `PrepositionRecord` representing the preposition sense of which it is a synonym.

During development, the 18 sets of multiple matching senses of synonymous prepositions were written to a file[84]. These were manually reviewed and the multiple synonymous senses were re-categorised as synonym, hypernym or hyponym[85]. The status of each `PrepositionRecord` which represents a member of such a set is read from this file[86] as one of these three relation types.

4.2.3.3 Creating Prepositional Synsets

For each sense of each preposition word form, a new object is created of class `Preposition`, which inherits from class `WordSense`[87]. Each time a `Preposition` object

[83] excluding those with variant spellings removed from the main preposition map
[84] *Triple matched synonyms.csv* comprising multi-line records specifying the fields of a `PrepositionRecord` grouped in such a way that the first record in each of the 18 groups represents a sense of a preposition headword, and the remaining records in the group represent the multiple synonymous senses of its synonymous headword.
[85] in another column.
[86] *Triple matched synonyms.csv* is read in the same order as it was written, such that when multiple senses of a synonym of a sense are found, the next group of records from the file will correspond to the same sense followed by its multiple synonym senses (all of which necessarily have the same headwords). The `PrepositionRecord` is added to the valid synonyms, valid hypernyms or valid hyponyms field as appropriate, within the `PrepositionRecord` representing the preposition sense of which it is a synonym. Each `PrepositionRecord` listed in the variant spellings field of the `PrepositionLoader` is then restored to the main preposition map.
[87] The *word form* and *relation to core sense* fields are assigned from the data held in the `PrepositionRecord` in the main preposition map corresponding to the preposition sense. Each new

is created, the PrepositionalTaxonomyBuilder creates or finds the corresponding PrepositionalSynset[88]. If no synonymous ID is found, a new PrepositionalSynset is created[89] and added to the global synset map[90]. The newly created Preposition is added to the PrepositionalSynset[91]. Once a Preposition has been created from every PrepositionRecord, and assigned to a PrepositionalSynset, the lexicon is updated with the new data. 800 prepositional synsets are created, containing 1111 prepositions representing 312 word forms.

4.2.4 Constructing the Preposition Taxonomy

The TPP data and the associated taxonomy files released with it imply a taxonomy of prepositional semantic roles (Litkowski & Hargraves, 2006), which is an advance on the

---

Preposition is assigned to the *instance* field of the corresponding PrepositionRecord. Sense numbers are assigned to each Preposition object restarting from 1 for each preposition word form.

[88] A PrepositionalSynset is found if the PrepositionRecord corresponding to the preposition sense has a valid *ID* field (> 0), which will be equal to the *ID* of the PrepositionalSynset. Otherwise, its synonyms are searched for a valid *ID*. If every synonym *ID* found is valid and equal, then the corresponding PrepositionalSynset with that *ID* is retrieved from the global synset map encapsulated in the wordnet.

[89] When a new PrepositionalSynset is created, it is assigned the next available *ID*, starting from 500000000, such that each *ID* is unique in the wordnet. The value of the *ID* has no significance apart from indicating the order of creation. The fields of a PrepositionalSynset include a set of *superordinate taxonomic categorizers*, a single *semantic role type* and a set of *complement properties*, none of which are initialised with any data by the constructor.

[90] If unequal *IDs* are found, any PrepositionRecord representing a synonym with a *superordinate taxonomic categorizer* different from that of the PrepositionRecord corresponding to the preposition sense is removed from the synonym list and the search for a unique valid *ID* is repeated. If unequal *IDs* are still found a fatal exception is thrown.

[91] When a Preposition is added to a PrepositionalSynset, the *ID* of the PrepositionalSynset is copied to the Preposition and to the corresponding PrepositionRecord. The gloss and examples from the PrepositionRecord are added to the PrepositionalSynset. The *superordinate taxonomic categorizer* of the PrepositionRecord is added to the set held by the PrepositionalSynset. The semantic role type of the PrepositionRecord is assigned to the PrepositionalSynset but a fatal error occurs if it already has a different one. The *complement properties* of the PrepositionRecord are added to those of the PrepositionalSynset. In all cases, every Preposition representing a synonym of the current PrepositionRecord is added to the new PrepositionalSynset unless it already has a valid *ID*, indicating that it has already been added. If it does have a valid *ID*, but this differs from the *ID* of the new PrepositionalSynset, indicating that the synonym has been added to another synset, then the *superordinate taxonomic categorizer* of the synonym is compared with that of the current PrepositionRecord. If it differs, then the synonym is removed from the synonym list. If the *superordinate taxonomic categorizer* is the same as that of the current PrepositionRecord, then the *semantic role type* of the synonym is compared with that of the current PrepositionRecord. If this also differs, then the current PrepositionRecord is cloned but without its synonyms, a new Preposition is created from the clone and the new Preposition is added to the new PrepositionalSynset. If the *semantic role type* is the same, while the *superordinate taxonomic categorizer* differs, a fatal exception occurs.

taxonomy based on digraph analysis presented by Litkowski (2002), though largely consistent with it (§4.2.1.5). Since prepositions with diverse meanings can share semantic role types, the semantic role taxonomy is treated as applicable to senses of the same or synonymous prepositions. Because of the parallelisms between the usages of the same preposition in different roles (Jackendoff, 1983; §4.2.1.6), lexical distinctions between one `PrepositionalSynset` and another (with different lexical content) override this taxonomy (§4.2.4.2).

4.2.4.1 Building the Implicit Taxonomy

A taxonomy map[92] is created and populated with taxonomy records mapping from parents to lists of children, where each child is a semantic role type and each parent is either a semantic role type or a superordinate taxonomic categorizer. This information is read from taxonomy files, one for each semantic role type[93]. The taxonomy file for each semantic role type gives one or more parent types for that semantic role type.

A `PrepositionalSynset` list is created for each semantic role type which does not also occur as a superordinate taxonomic categorizer, comprising every `PrepositionalSynset` found in the global synset map with that type. A HYPERNYM search is conducted for each `PrepositionalSynset` in the list: for each word form in each `PrepositionalSynset`, a list is obtained from the lexicon of every `PrepositionalSynset` which includes that word form. Any `PrepositionalSynset` which includes the word form and whose semantic role type, according to the taxonomy map, is the taxonomic parent of the semantic role type of the current `PrepositionalSynset`, is added its the set of candidate HYPERNYMS[94].

If there is only one candidate HYPERNYM for a `PrepositionalSynset`, then it is assigned as its HYPERNYM; if there are multiple candidate HYPERNYMS and any of

[92] `Map<String, List<String>>`
[93] The taxonomy files must be found in a subdirectory of the default directory called *taxonomy*.
[94] Any empty semantic role type is excluded from this operation.

them are non-abstract (have one or more glosses or examples), then a fatal error occurs; if there are 2 candidate abstract HYPERNYMS for a `PrepositionalSynset`, one of which has the same superordinate taxonomic categorizer, then that candidate is assigned as its HYPERNYM; otherwise all the candidates are assigned as HYPERNYMS.

When a `PrepositionalSynset` is assigned as HYPERNYM of another `PrepositionalSynset` (its HYPONYM):

- a new `Preposition` is created for every word form of the HYPONYM not represented in the HYPERNYM;
- the relation to core sense field of each `Preposition` is defined as "CORE: " + the semantic role type of the HYPERNYM;
- each new `Preposition` is added to the HYPERNYM;
- an entry for the HYPERNYM is added to the lexicon;
- a `WordnetRelation` of `Relation.Type.HYPERNYM` is encoded from each HYPONYM to the HYPERNYM and its converse `WordnetRelation` of `Relation.Type.HYPONYM` is encoded from the HYPERNYM to each HYPONYM.

4.2.4.2 High Level Abstract Taxonomy

Once the implicit taxonomy is complete, a new abstract HYPERNYM is created for each set of `PrepositionalSynset`s (its HYPONYMS), which share the same set of word forms and the same semantic role type and have, as yet, no HYPERNYM. The semantic role type of the abstract HYPERNYM is the parent semantic role type of the semantic role type of the HYPONYMS, as read from the taxonomy map[95]. Each abstract HYPERNYM has a `Preposition` encoded in it for each of the same set of word forms as are possessed by its HYPONYMS. The abstract HYPERNYM is then added to the global synset map. Relations are encoded between the HYPERNYM and its HYPONYMS in the

[95] This semantic role type, which is always also a superordinate taxonomic categorizer, is also encoded as a superordinate taxonomic categorizer of the HYPERNYM.

way described in §4.2.4.1. This procedure ensures that every non-abstract `PrepositionalSynset` belongs to a taxonomic tree. Each of the top HYPERNYMS of these trees represents the intersection between a combination of word forms and a superordinate taxonomic category corresponding to a semantic role type taxonomy.

In order to provide a high level abstract HYPERNYM for each combination of word forms possessed by any `PrepositionalSynset` which has no HYPERNYM, the same operation is now repeated, ignoring semantic role types. The HYPONYMS of each high level abstract HYPERNYM are the abstract HYPERNYMS for each superordinate taxonomic category with the same set of word forms[96]. Thus the resultant taxonomy comprises a high level lexical categorisation by combinations of word forms and a secondary classification corresponding to the classification of semantic role types into superordinate taxonomic categories.

4.2.4.3 Top Level Abstract Taxonomy

The properties of the preposition taxonomy so far constructed automatically were analysed using the method proposed for verbs (§2.2.2.2.1). Each `PrepositionalSynset` without a HYPERNYM was defined mentally so that HYPERNYMS could be assigned manually, using an existing combination of word forms where possible, and assigning more than one where appropriate (Appendix 26). The following additional word form combinations, representing very high level abstractions, were found to be required:

- *away from; not at*
- *among; between*
- *as not*
- *near; with*
- *caused by*
- *not caused by*
- *as why*

[96] A high level abstract HYPERNYM has an empty semantic role type and superordinate taxonomic categoriser field and its relation to core sense equals "CORE:".

- *as not why;*

A high level abstract `PrepositionalSynset` is created to represent each of these additional word form combinations and is added to the global synset map; the lexicon is updated accordingly. Records are then read from file[97], each of which comprises 2 fields which represent the word forms of the HYPONYM and the word forms of the HYPERNYM. The highest level synsets with each of the 2 combinations of word forms are found and relations are encoded between them with the first synset as HYPONYM and the second as HYPERNYM, as described in §4.2.4.1.

The resultant taxonomy has 6 top HYPERNYMS namely:

- *as*
- *as not*
- *at*
- *near; with*
- *not at*
- *with reference to*

This can be contrasted with Litkowski's (2002) original taxonomy (§4.2.1; Appendix 23). The differences are due to non-differentiation of preposition senses in Litkowski's presentation of his digraph analysis and the high priority given to synonym identification and lexical distinctions in the development of the taxonomy presented here.

4.2.4.4 Prepositional Antonyms

The top level HYPERNYMS in the second column of Appendix 26 were arranged alphabetically without duplicates and, wherever possible, each member of the resultant set was manually assigned an ANTONYM from the same set, with a common HYPERNYM (Smrž, 2003; Huang et al., 2002; Vossen, 2002; §2.2.2.3) in all cases except where one or both ANTONYMS are top HYPERNYMS (Appendix 27). The

[97] *Top ontology.csv* (Appendix 26)

ANTONYM data[98] is read and processed in the same way as the top level ontology[99], except that relations of `Relation.Type.ANTONYM` are encoded in both directions between the pairs.

After each pair of top level ANTONYMS is encoded, ANTONYM relations are also encoded between those pairs of HYPONYMS of the top level ANTONYMS which have the same lexical content as the top level ANTONYMS, and the same superordinate taxonomic categorizer as each other. This operation is performed recursively so that ANTONYM pairings are cascaded down the taxonomy as far as the shared lexical content and superordinate taxonomic categorizer requirements hold without interruption. This creates symmetrical ANTONYM ancestries with a common HYPERNYM (§2.2.2.3). The resultant preposition taxonomy is headed by three pairs of ANTONYMS: {"as"} paired with {"as not"}, {"at"} paired with {"not at"} and {"near"; "with"} paired with {"sans"; "without"}; {"with reference to"} has no ANTONYM.

Encoding of ANTONYMS is the final phase of enrichment of the WordNet model with prepositions. No claim is made regarding the originality or completeness of the information regarding prepositions. Simply a major gap in the coverage of WordNet has been filled, to the minimal extent necessary, with data discovered by the latest research. The assignation of prepositions to synsets and the encoding of relations between them has been documented and, as far as possible, data-driven.

4.3 Pruning the WordNet Model

The interrogation of the WordNet model has revealed many faults and inconsistencies in the relations (§2.2.2). While correction of all of these is highly desirable, the scope of such an operation is extremely broad and would require a great deal of manual lexicographic effort which would clearly not be possible within the project timeline. While correction of the WordNet sentence frames has been attempted, and this could be a

[98] file *Antonyms.csv* (Appendix 27)
[99] file *Top ontology.csv* (Appendix 26)

step towards the correction of the verb taxonomy (§§1.3.2.7, 2.3.2, 2.4), bringing this line of research to a satisfactory conclusion falls outside the scope of this project. Consequently, correction prior to morphological enrichment has been confined to the removal of disconnected proper nouns and limited rationalisation of relations where the process can be automated. The changes made are briefly discussed here in the order in which they are executed[100]. The phases involved are elimination of CLASS_MEMBER relations, replacement of adjectival SIMILAR-CLUSTERHEAD relations with HYPERNYM-HYPONYM relations, elimination of PERTAINYM relations between adjectives, a reduction of the number of disconnected proper nouns and the replacement of PERTAINYM and ANTONYM relations between word senses with the same type of relations between the corresponding synsets.

4.3.1 The CLASS_MEMBER Relation

The CLASS_MEMBER relation is used in WordNet to categorise how words are used as distinct from what they mean. It is the only relation type with subtypes: TOPICAL, REGIONAL and USAGE.

- TOPICAL class-membership relationships hold between noun synsets representing narrow categories and adjectives which apply to them, e. g. "chirpy" is a member of class "bird". The synset {"vegetation "; "flora"; "botany"} has TOPICAL members {"mown"; "cut"; " unmown"; "uncut"; "sprouted"; "dried-up"; "sere"; "sear"; "shriveled"; "shrivelled"; "withered"}.
- REGIONAL class-membership has been used to associate word senses with their countries of currency. Some British terms not used in America are associated with the synset representing Great Britain; much smaller sets are given for Scotland, Canada and the United States.
- The main USAGE classes are all categories of words and phrases, such as "plural", "disparagement", "ethnic slur", "slang", "trademark", "trade name" and

[100] `NaturalLanguageProcessor.pruneWordnet()`

"colloquialism". "Ping-Pong" and "carborundum" are both encoded as trademarks. USAGE has also been used extensively in error for REGIONAL (e. g. "baking tray", "zebra crossing" and "sandpit" are encoded as USAGE members of the REGIONAL class representing Great Britain).

The sets of class members are incomplete, the range of classes is arbitrary and the encoding is erratic. It would be possible to add fields to the `WordSense` class to indicate its status with respect to each subtype, but there is not enough information provided to make this a worthwhile exercise. For these reasons, all CLASS_MEMBER relations and their converses have been deleted[101].

4.3.2 SIMILAR and CLUSTERHEAD Relations

Adjectives in WordNet are organised in a completely different way from nouns and verbs, in that no HYPERNYM-HYPONYM relations are encoded. These are replaced by SIMILAR-CLUSTERHEAD relations, where an adjective *clusterhead* maps by a SIMILAR relation to several adjective *satellites*, but no adjective can be at one and the same time a clusterhead and a satellite. A sample was taken of 106 SIMILAR relations, which were then classified manually (Table 36).

In 70% of cases the clusterhead is the HYPERNYM of the satellite. Every SIMILAR relation has been replaced with a HYPONYM relation and every CLUSTERHEAD relation with a HYPERNYM relation[102], for the following reasons:

- the level of accuracy (70%: Table 36) is as good as that found in the verb taxonomy (§2.2.2);
- having the same kind of taxonomy for adjectives as for nouns will facilitate the application of any WSD algorithm which uses HYPONYM and HYPERNYM relations (§6.1);

[101] `Secator.abolishClassMembership()`
[102] `Secator.changeclusterHeadToHypernyms()`

- because HYPERNYM/ HYPONYM relations have not been allowed between adjectives, PERTAINYM relations have been used, inconsistently, to link adjectives, (§4.3.3).

Table 36: Classification of SIMILAR-CLUSTERHEAD relations

Category	Instances
Clusterhead is hypernym of satellite	74
Satellite is hypernym of clusterhead	8
Clusterhead is synonym of satellite	15
Clusterhead is sister of satellite	3
Clusterhead is unrelated to satellite	6
TOTAL	106

Table 37: Reclassification of PERTAINYM relations between adjectives

New Relation	**Instances**
SIMILAR	25
DERIV	12
ANTONYM	1
Total	38

4.3.3 Adjective to Adjective PERTAINYM Relations

The PERTAINYM relation is used typically to indicate the noun from which an adjective is derived or the adjective from which an adverb is derived, and clearly expresses a semantic and not merely a lexical relationship. In preparation for the re-encoding of these relations between synsets, representing meanings, instead of between word senses (§4.3.5), a few cases were unexpectedly discovered of PERTAINYM relations between two adjectives. The semantic import of these relations cannot be the same as in the other cases. Examination of the adjective to adjective PERTAINYMS[103] (Appendix 28) showed that they could all be reclassified as SIMILAR, DERIV or ANTONYM. The number of instances of each reclassification is shown in Table 37. Reclassification as SIMILAR would violate the rule that an adjective must be a CLUSTERHEAD or a SATELLITE but not both (§4.3.2, Appendix 65). This was an additional reason for

[103] *Pertainyms to Derivs.csv*

replacing SIMILAR relations with HYPONYM relations (§4.3.2). Therefore the relations reclassified as SIMILAR in Appendix 28 have been re-encoded as HYPONYM[104] and the remainder have been re-encoded as they were reclassified.

4.3.4 Proper Nouns

WordNet 3.0 contains many proper nouns, often connected to the rest of the graph only by CLASS-MEMBER, INSTANCE-INSTANTIATED or MERONYM-HOLONYM relations. CLASS-MEMBER relations have already been removed (§4.3.1); INSTANCE relations encode mainly proper names as instances (in the opinion of the encoders) of various concepts encapsulated by synsets, including such niceties as "Einstein was a genius", and provide incomplete lists for such categories as "physicist" and "king". The selection is narrow and intrinsically arbitrary. It is hard to see the reason for including this kind of encyclopaedic information in a lexical database; MERONYM-HOLONYM relations are used to identify the geographical locations of towns, rivers etc. This *world knowledge* again belongs in an encyclopaedia rather than a lexical database. While there may have been some justification for including this kind of information in the past, there is none since the advent of easily accessible encyclopaedic resources such as Wikipedia.

On the other hand, proper names such as names of countries may be relevant when they are linked to adjectives referring to nationality. It is useful to retain PERTAINYM relations such as between "French" and "France". Accordingly an algorithm[105] was developed to delete those proper nouns which have only CLASS-MEMBER, INSTANCE-INSTANTIATED or MERONYM-HOLONYM relations.

---

[104] `Secator.abolishAdjectiveToAdjectivePertainyms`

[105] `Secator.removeProperNouns` was the first algorithm developed for the purpose of modifying the data content of the WordNet model. It required a method for synset deletion which gave rise to a consideration of how safely to delete synsets in this or any other circumstance. Synset deletion must ensure:

- that all relations targeted on the synset to be deleted are also deleted;
- that a concurrent modification error is avoided if iterating through the Synset map;
- that the lexicon is marked as inconsistent until it can be revised.

The definition of proper noun is not as clear-cut as it might seem. The main criterion obviously is that a proper noun is a noun in proper case (starting with a capital letter). The most obvious exception to this rule is the word "I". WordNet includes foreign names, many of which are prefixed by a lowercase word, e. g. "de" in French; some others start with an apostrophe. Acronyms such as NATO can be considered as proper nouns, but compounds like "NATO base" are not. Proper noun identification is further complicated by initials and hyphenations.

In the light of these considerations, the algorithm for removing proper nouns treats a noun as a proper noun *unless*:

- it has only 1 character, or starts with a numeral, punctuation mark or lowercase letter, unless it starts with "de ", "da ", "von " or "van ";
- the second character is " ", "-" or "'" and the third character is a punctuation mark, numeral or in lowercase;
- it consists of more than one word of which the first is all in uppercase (an acronym);
- it contains any word of more than 3 letters which does not start with an upper case character, unless that word ends with a hyphen or contains a hyphen followed by an uppercase letter.

The removal of proper noun synsets reduces the number of noun synsets from 82115 to 75455. No other synsets have been deleted during pruning.

4.3.5 Transfer of Semantic Relations between Word Senses to the Synsets which Contain them

Some relations in WordNet, in particular PERTAINYM and ANTONYM relations, are encoded between word senses rather than between synsets. The application of algorithms which measure semantic distance, or otherwise use WordNet relations for WSD (§6.1.1) would be facilitated if all semantic relations were encoded between synsets rather than

between word senses. Since all members of a synset purportedly have the same meaning, semantic relations logically hold between synsets rather than word senses, despite the psycholinguistic view (Miller, 1998) that ANTONYMS hold between individual words.

Of the relations between word senses:

- the CLASS-MEMBER relation had already been eliminated (§4.3.1);
- the ANTONYM relation has been transferred to synsets[106];
- the PERTAINYM relation has been transferred to synsets[107], except when encoded between 2 adjectives (§4.3.3);
- the DERIV relation is really a lexical relation so it can remain encoded between word senses;[108]
- the SEE-ALSO relation has been used as a "catch-all" where the nature of a relation has not been determined and has been applied mostly to adjectives; it is to be retained because it has been used successfully by WSD algorithms (Banerjee & Pedersen, 2003; §6.1.1.4);
- there is no specification for the meaning of the VERB_GROUP_POINTER relation; it is a poor indicator of syntactic similarity between verb synsets and has been ignored[109].

4.4 Conclusions from Preliminary Modifications

The modifications made to the WordNet model, while complete in themselves, fall far short of addressing all the errors and inconsistencies discovered (§§2.2, 2.3). Further desirable modifications, as outlined in §2.4, could not have been brought to a satisfactory

[106] `Secator.applyAntonymsToSynsets()`
[107] `Secator.applyPertainymsToSynsets()`
[108] Ideally this directionless derivational relation type should be given directionality, but systematic morphological enrichment (§5.3) will make it redundant.
[109] 1748 pairs of verb synsets are linked by VERB_GROUP_POINTERS. None of these are connected either to each other or to other synsets by cause or entailment relations although some correspond to causal relationships. Since Levin (1993) defines verb groups as having common behaviour with respect to their arguments, an investigation was made to see whether the synsets linked by verb group pointers had the same framesets (§2.3.1). Only 342 out of the 1748 pairs had identical framesets. Of the 1406 pairs with different framesets, the framesets of 446 pairs had the same set of valencies, leaving 960 pairs with differing valency sets.

conclusion within the project timescale, given that the main objective was morphological analysis and enrichment.

The presence of prepositions allows relations to be encoded between morphemes, particularly prefixes which derive from or translate prepositions, and the relevant prepositions. It would also allow the encoding of mappings between sentence frames and the prepositions they specify, once a satisfactory set of sentence frames has been obtained (§§1.3.2.7, 2.4).

The lexical database we are left with is still far from perfect. However, the extensive coverage of the English language, although not entirely up to date and somewhat partial to American usages, is nevertheless one of WordNet's main strengths. This has been improved by the addition of prepositions, though pronouns and modal verbs are still missing.

Given that a decision has been taken to apply morphological enrichment as lexical relations within the lexicon component of the model (§§3.5.3), rather than applying it to the wordnet component, the morphologically enriched lexicon will have a validity independent of the relational errors in WordNet (§2.2). The methodology for enriching the lexicon is equally applicable to any other lexicon, provided that it respects the distinctions between the minimal set of eight parts of speech (§1.1.4), and (preferably) has some corpus frequency data.

5 Morphological Analysis and Enrichment of the Lexicon

This section will describe the development of a morphological analyser, which although constructed with the aid of the lexicon derived from WordNet, is independent of that lexicon and portable to any other English lexicon (§3.5.3) which conforms to the basic specifications in §4.4. The morphological analysis of words in a hybrid model (§3.5.4), combining unsupervised automatic affix discovery with the supervised application of morphological rules, requires first that the morphological ruleset should be sufficiently comprehensive to capture all the regular transformations which occur between suffixations, as well as between suffixations and their non-suffix-bearing constituent morphemes, referred to as their roots. So this chapter will begin by presenting the enhancements made to the morphological rules (§5.1) to address the problems identified during the pilot study (§3.2.2), in particular the problems relating to the impossibility of applying multilingually formulated rules correctly within a monolingual lexical database. Such rules will be supplanted by more specific monolingually formulated rules.

The hybrid morphological analyser also requires algorithms to apply these rules optimally and to break words into their components in different ways for different morphological phenomena (particularly concatenation and affixation analysis), without falling into the trap of the segmentation fallacy (§3.3). Word segmentation will in many cases be performed, but it is never assumed that the results of such a segmentation represent the morphological roots of the word so segmented: generalised spelling rules must be applied and the morphological rules, for the most part, apply suffix substitutions, which could only be applied through a segmentation-based approach in those cases where the longer suffix of the derivative is fully inclusive of the shorter suffix of the root. The resistance of some prefixations to meaningful segmentation is addressed by the recognition of linking vowel exceptions (§5.3.11.9) and of irregular prefixations, involving a finite set of irregular prefixes (§5.3.11.2). In this chapter the terms *de-concatenation, affix stripping, prefix stripping* and *suffix stripping* will be used only for processes which involve

segmentation; higher level processes which take account of the pitfalls of segmentation will be termed *concatenation analysis, affixation analysis, prefixation analysis* and *suffixation analysis*. The section will proceed to present the two main new algorithms required for conducting morphological analysis (§5.2) while avoiding the segmentation fallacy, the *Word Analysis Algorithm* and the *Root Identification Algorithm*.

The entire process of morphological analysis performed by the hybrid model (§3.5.4) and the morphological enrichment of the database with lexical relations based on derivational morphology, derived by that analysis, will then be presented sequentially from compound expression analysis through iterations of concatenation and affixation analysis (§5.3). The sequence of affixation analysis operations is primarily determined by the affix stripping precedence of antonymous prefixations over suffixations over non-antonymous prefixations (§3.5.1). The iterative development process by which the morphological analyser was created will be presented in parallel with its functionality. During the earlier phases of the analysis, a positive *lexical validity requirement* is imposed on the output, meaning that all identified morphological roots must be words found in the lexicon, morphologically related to the input. This requirement is progressively relaxed during the course of affixation analysis, so that first the affixes themselves are exempted from this requirement while the stems are still subject to it, and then, at later stages, the stems also are exempted, so that a stem dictionary can be made to include all such *non-lexical stems*. These stems are themselves subjected to morphological analysis in the final stages. Morphological enrichment comprises the encoding of lexical relations between morphological relatives, namely the compound expressions, words and stems which are the inputs to the analysis and their identified, morphologically related components as output by the analysis, either words in their own right or the translations of components which are not lexically valid. Where the analysis has found morphological rules to be applicable, these lexical relations correspond to the links in the derivational trees to which the input and output words belong; their relation types are determined by the morphological rules. The outcome of morphological enrichment of the WordNet model is a morphosemantic wordnet; the outcome of encoding lexical relations, derived by the

same portable morphological analyser, in any other lexicon, would be a morphologically enriched lexical database.

5.1 Extensions to Morphological Rules

The pilot study (§3.2.2) revealed many instances of overgeneration and undergeneration by morphological rules, making it clear that the rules needed to be reviewed, in particular:

1. most overgenerations occurred when morphological rules were applied to suffix removal to generate monosyllabic roots (addressed in §5.1.1);
2. other overgenerations arose from attempts to apply multilingually formulated rules monolingually (addressed in §5.1.2);
3. most undergenerations arose from the failure to apply multilingually formulated rules which cannot be applied monolingually (addressed in §5.1.2);
4. other undergenerations arose because the morphological ruleset was not complete (addressed in §5.1.3).

Since more than one rule can be applied to the same input suffix, some way of establishing the precedence of rules was called for (§5.1.4), and finally some provision needed to be made for suffixations which resist analysis as long as there is a requirement that the output word be lexically valid (§5.1.5).

A compact, computationally tractable format having been established (§3.2.2.2, Appendix 10), it was not necessary for new rules to be formulated linguistically like the original set (§3.2.2.1; Appendix 9). Simply the requisite fields were defined and added to the tables of rules (§5.1.1, Appendices 10 & 36).

5.1.1 Additional Fields

Many overgenerations which occurred during the pilot study (§3.2.2.2.2) arose from the application of morphological rules in such a way as to generate monosyllabic roots; suppression of these rules would result in undergeneration. To address this problem, a Boolean field `applicableToMonosyllabicRoot` was added to the specification for a morphological rule, to determine whether or not the rule is to be applied when the result is a monosyllabic root. If `applicableToMonosyllabicRoot` is true then there is a risk of overgeneration of monosyllabic roots, but if it is false then there is a risk of undergeneration, suppressing valid monosyllabic roots. An overgeneration tolerance threshold needed to be set above which monosyllabic roots should be suppressed and below which they should be tolerated for the sake of avoiding undergeneration. Setting the threshold too high would require more manual effort by way of creating stoplists (§§5.2.2.5, 5.3). With these considerations in mind, a 10% threshold was adopted so that `applicableToMonosyllabicRoot` was set to false for those rules whose monosyllabic outputs were incorrect in more than 10% of cases of suffixation analysis or homonym analysis during the pilot study or during subsequent iterative development (§5.2.2.4, 5.3). Where already-implemented rules were re-specified, the specification applied to the original rule was inherited unless contra-indicatory evidence was acquired (§5.1.2). The re-specified multilingually formulated rules which had not previously been applied in any form were generally set initially to reject monosyllabic roots by default, though this setting was modified where evidence justified such a modification. For the implementation of these restrictions see §§5.2.2.5, 5.3.7.4.

The specification of additional fields, namely the `Relation.Type` field introduced in §3.2.2.1 but not implemented in the experiments in §3.2.2.2 and the Boolean field described in the previous paragraph, meant that morphological rules could no longer be stored as simple mappings between a source `POSTaggedSuffix` and a target `POSTaggedSuffix` as they had been for the original experiments described in §3.2.2. Instead, a Java class `MorphologicalRule` was introduced, with the additional fields, and

the rules thereafter were stored in tables[110] in which each key is a source `POSTaggedSuffix` mapping to all the rules for which it is the source. The rules used for suffix stripping are termed *converse* morphological rules, because the morphological rules were originally formulated for adding suffixes to roots (§3.2.2.2.1). The converse rules are stored in separate tables. The *conditional* rules (§3.2.2.1) are also stored separately.

5.1.2 Re-specification of Multilingually Formulated Rules

The priority for extending the morphological ruleset was to find an adequate computationally tractable formulation of those rules which had only a linguistic formulation because they require reference to languages other than English (those wholly in italics in Appendix 9). Of these, by far the most important group are those which concern quasi-gerunds, where the suffix "-ion" is not also an instance of its grandchild suffix "-ation" (§3.2.2.1).

The stem to which "-ion" attaches (in almost all cases which are not instances of "-ation" as well as many cases which are instances of "-ation") is the stem of a Latin passive participle with "-us" removed, which is equivalent to the *supine* of a Latin verb with "-um" removed. Irregular supines of Latin verbs are listed in a Latin dictionary. The original plan was to acquire the infinitives of these verbs from a Latin lexical resource, Perseus (http://www.perseus.tufts.edu/). However, given a knowledge of Latin, the overhead of obtaining these infinitives automatically and then identifying the related English verbs manually would have been greater than the manual effort of identifying the English verbs directly from the English quasi-gerunds.

Other frequently occurring suffixes whose usage is specified by multilingually formulated morphological rules are "-al", "-ant", "-eal", "-ent", "-ic" and "-itis". In order to obtain the stems carrying these suffixes, a suffix tree was constructed (§3.4.2), and all

[110] `Map<POSTaggedSuffix, List<MorphologicalRule>>`

the stems with which these suffixes occur were extracted, in addition to the stems for "-ion". The stem counts for these suffixes are shown in Table 38.

Table 38: Stem counts for suffixes specified by multilingually formulated rules

	Suffix	**Stem count**
	ion	2434
of which	ation	1612
	others	822
	al	2194
of which	eal	102
	others	2092
	ic	545
	itis	174
	ant	390
	ent	928

Table 38 shows that there are 822 stems for suffix "-ion" where it is not an instance of "-ation". The resultant list is short enough to be amenable to the manual identification of new morphological rules from co-occurrences of morphological patterns (§3.2.3). The 54 new rules identified, most, but not all, of which involve Latin passive participle derivations, are listed in Appendix 30.

The suffix "-al" likewise needs to be treated differently when it is not also an instance of "-eal". Those rules applicable to the suffix "-al" which had been applied in the pilot study showed a strong tendency to overgenerate while its applicability to the genitive stem of a Latin noun had been specified in the formulation (Appendix 9), but not applied. Suffix "-eal" is applied to the genitive stem of Greek nouns (medical terms) representing bodyparts. The stems found for "-al" included some Latin genitive stems along with other instances which could be grouped to form rules. 55 new rules were identified to specify suffix "-al" (Appendix 31), of which only 2 apply to "-eal".

17 new rules were identified for the irregular suffix "-ic" (Appendix 34), which, like "-al", caused a lot of overgeneration in the pilot study, but shows little of the expected

preference for Latin genitive stems, and 7 new rules were identified for "-itis" (Appendix 35), which again applies to the genitive stem of Greek words representing bodyparts.

Suffix "-ent" is generally derived from the active participle of a Latin verb with an infinitive in "-ere"; suffix "-ant" is sometimes derived from the active participle of a Latin verb with an infinitive in "-are", but is often an indicator of a derivation from Latin through French, where the active participle always ends with this suffix (§3.2.2.1). The irregularities encapsulated in the 35 new rules identified for "-ant" (Appendix 32) and the 45 for "ent" (Appendix 33) reflect these complexities. It might appear that some of these rules are over-specified, as many of the source morphemes could be reduced to an empty morpheme or just "-e" and many target morphemes could be reduced to "-ent". The detailed specification is justified on the following criteria:

- some preceding consonants seem to prefer "-ant" while others prefer "-ent" (Appendices 32-33);
- specifying specific rules for individual preceding consonants allows their applicability to monosyllables to be individually specified (§5.1.1).

No attempt was made to re-specify the remaining multilingually formulated rules. With the possible exception of the suffix "-ible", automatic suffix analysis did not yield a sufficient number of valid stems for this approach to be viable. However instances of "-ible" and other suffixes specified by the remaining multilingually formulated rules were trapped by the procedures described in §5.1.3.

5.1.3 Additional Rules

Undergeneration and overgeneration were observed in the output from suffixation and homonym analysis (§§5.3.6-5.3.8) during iterative development of the morphological analyser in the same way as during the pilot study (§3.2.2). Additional rules were formulated as a result of these observations as follows:

- **Undergeneration:** Throughout the implementation of suffixation and homonym analysis, unidentified roots files are generated (§§5.3.6.1, 5.3.7.4, 5.3.8, 5.3.14.2).

The instances of failed morphological analyses in these files arising from the absence of rules for some automatically discovered suffixes were examined with a view to identifying additional morphological rules. Most of the additional rules were identified in this way (§5.3.7).

- **Overgeneration:** At the same time, where erroneous analyses were discovered in the output (§§5.3.7.3, 5.3.14.2), instead of making an addition to a stoplist or applying a monosyllabic restriction (§5.1.1), it was sometimes possible to re-specify the morphological rule which overgenerated in such a way that it would no longer cause the same overgeneration, typically by specifying longer source and target morphemes.

The final ruleset can be found in Appendix 36.

5.1.4 Rule Precedence

Since the same input suffix can be the target of more than one morphological rule (the source of the converse morphological rule applied when removing or replacing it) there needs to be some way of choosing which rule to apply. In the majority of cases, only one rule will produce lexically valid output (an output word which occurs in the lexicon) and that rule must be chosen, but there are cases where more than one analysis can produce lexically valid output, so rules applicable to the same input suffix are ordered within the list to which each input suffix maps in such a way as to give precedence to the most likely analysis where more than one analysis is possible. The optimum ordering of the rules applying to the removal of any suffix is that which requires the least deployment of stoplists.

The output from the application of a morphological rules is considered to be lexically valid if it occurs in the lexicon. As long as a lexical validity is required of the output (as long as a positive *lexical validity requirement* is imposed), precedence generally needs to be given to more unusual rules so that a rule which applies only in exceptional cases will be passed over in the majority of cases but applied where it does generate lexically valid output. Generally, but not necessarily, the rule which generates lexically valid output

words when applied to the greatest number of input words is the most widely applied but has the lowest precedence, so that the number of lexically valid outputs can be a guide to ordering the rules, though the ordering has been subsequently revised where results demonstrated that this was necessary (§5.2.2.4). In the case of a handful of rules, the relative recorded frequencies[111] of the possible output words turn out to be the best guide to the correct analysis, irrespective of the precedence of the rules (§5.2.2.6).

5.1.5 Non-lexical Rules

Many suffixations comprise a suffix preceded by a *non-lexical stem* (a stem which is not lexically valid as the POS specified by the rule which generated it). In some cases, not only is the stem not lexically valid, but neither is any suffixation generated by replacing the original suffix according to any rule. Where no rule produces lexically valid output when applied to a word with a valid suffix, during secondary suffixation analysis (§5.3.14), there needs to be a default rule, for which the requirement for lexically valid output can be waived. This will generally be the rule which generates lexically valid output when applied to the greatest number of other inputs. So the single default non-lexical rule applicable to the removal of each input suffix is usually, though not necessarily, the rule with lowest precedence. The non-lexical rules are stored independently of the main ruleset (for implementation see §5.2.2.5).

5.2 New Algorithms for Morphological Analysis

In addition to the unsupervised Automatic Affix Discovery Algorithm already presented (§3.4), morphological analysis requires a *Word Analysis Algorithm* which can break words into their components in the simplest case of concatenation analysis but also in more complex cases, without falling into the trap of the segmentation fallacy (§3.3). Also required is a *Root Identification Algorithm* which applies morphological rules in such a way as to identify morphological relationships correctly, where more than one rule is

[111] Brown Corpus frequencies in the case of the WordNet-based lexicon.

applicable, and to avoid applying any rule erroneously. The two new algorithms are presented in this section.

5.2.1 Word Analysis Algorithm

5.2.1.1 Purpose

The need to give precedence to concatenation analysis over affixation analysis has already been postulated (§3.5.2). In theory it should be a simple matter to separate concatenations (words which comprise a sequence of other shorter words) into their component words. It is however clear that some words can be broken down into smaller words in more than one way, none of which is necessarily correct, for example "assassin" could be broken down into "as" + "sass" + "in" or "ass" + "ass" + "in" or "ass" + "as" + "sin", none of which have anything to do with the word's etymology. An algorithm was therefore required which would output a list of alternative arrays[112], each of which represents a breakdown of an input word into shorter words, so as to include all such possible breakdowns. In devising such an algorithm, it is worth considering whether a generic algorithm could be devised which could also be used in affixation analysis. The primary difference between the tasks of concatenation analysis and affixation analysis is that with concatenation analysis, it is a requirement that the components output all be lexically valid words, whereas with affixation analysis there is no such requirement, but there is a requirement that the affix or affixes be valid, which can be tested against the results from automatic affix discovery. A common algorithm then requires to be supplied with lists of acceptable output morphemes for particular positions within the input word, whether these morphemes be words or affixes: in the case of concatenation analysis, each position must be occupied by a word found in the lexicon, or rather in its single word subset, the *atomic dictionary* (§5.3.3.1); in the case of affixation analysis, only the initial or terminal position must be occupied by a valid affix, depending on whether prefixation or suffixation analysis is being performed. There is no such requirement on the stems

[112] `List<String[]>`

from affixation analysis as the stem dictionary is an output from, not an input to, the process of morphological analysis, otherwise the analysis would be bound to some particular linguistic theory rather than being empirical.

5.2.1.2 Requirements

It is clearly pointless and inefficient to supply the algorithm with words or affixes which the word being analysed does not contain, and so a method is required of creating the relevant lists of valid components to supply to the algorithm. The algorithm can be supplied with lists of candidate morphemes for the beginning and end of the word to be analysed (*candidate fronts* and *candidate backs*), but supplying lists for the middle would be extremely complex and inefficient as we do not know at the outset how many components there may be, but in the majority of cases there are only two. If removal of a combination of a candidate front and a candidate back leaves no residue, then a 2-element array will be added to the output; if there is an acceptable morpheme in the middle, then a 3-element array will be added to the output; otherwise recursion will be required after deriving new lists of candidate fronts and candidate backs applicable to the residue in the middle.[113]

5.2.1.3 Generating Candidate Lists

Given the existence of a *rhyming dictionary* (§3.4.2.1), although it was not originally designed for this purpose, and given that the rhyming dictionary used at this stage contains exactly the same information as the atomic dictionary, except that the word forms are reversed (§5.3.3.2), it is practical to use the rhyming dictionary for generating candidate back lists. This allows exactly the same method to be used to generate each

[113] In practice, candidate lists for all the words to be analysed (the contents of the atomic dictionary in the case of initial de-concatenation) are generated first and stored temporarily in two tables (`Map<String, List<Morpheme>>`) `candidatesWithFronts` and `candidatesWithBacks`, whose keysets are both the same as that of the atomic dictionary. Each key maps to the corresponding list of candidate fronts or candidate backs. The analysis algorithm is then applied to each word in the atomic dictionary, using the corresponding lists of candidate fronts and candidate backs.

candidate list. Simply the spelling of each item in each candidate back list will have to be re-reversed before the list can be used.

In its simplest form the algorithm which generates a list of candidates is as follows:

```
List<String> makeCandidate(short minStemLength, short frontWindowSize,
String word, Set<String> vocabulary)
{
  candidateFronts = empty List of Strings;
  if (length of word >= minStemLength)
  {
    while (frontWindowSize <= length of word - minStemLength)
    {
      String candidateFront = initial substring of word
        whose length =  frontWindowSize;
      if (vocabulary.contains(candidateFront))
      {
        add candidateFront to candidateFronts;
      }
      increment frontWindowSize by 1;
    }
  }
  return candidateFronts;
}
```

Here `frontWindowSize` is initially the minimum acceptable length for the first component, `minStemLength` is the minimum acceptable length for the rest of the word and `vocabulary` (for initial concatenation analysis) is the keyset of the main dictionary.[114]

[114] The actual implementation is more complicated in that each candidate is represented as a `Morpheme` and if `candidateFront` is not contained in `vocabulary`, it is written to a list of rejected components and two Boolean parameters `frequencyCorroboration` and `backwards` are passed. If `frequencyCorroboration` is true then `candidateFront` will be rejected if its frequency, as recorded in the main dictionary is zero (if `backwards` is false) or if the frequency of its reversed form is zero (if `backwards` is true).

In practice, for initial concatenation analysis, `minStemLength` and `frontWindowSize` are both set to 2 and an empty list is returned if any word starts with a numeral, punctuation mark or uppercase letter.

5.2.1.4 The Main Algorithm

In its original and simplest recursive form the Word Analysis Algorithm can be represented as follows:[115]

```
List<String[]> analyse(String wholeWord, List<String> candidateFronts,
List<String> candidateBacks)
{
  breakdowns = empty list of String arrays;
  for each candidate front in candidateFronts
  {
    for each candidate back in candidateBacks
    {
      core = wholeWord;
      delete candidate_back.length characters from the end of core;
      if (the length of core >= the length of candidate front)
      {
        a number of characters equal to the length of candidate front
          are deleted from the beginning of core;
        if (core is an empty String)
        {
          breakdown is a 2-element String array;
          breakdown[0] = candidate front;
          breakdown[1] = candidate back;
          breakdown is added to breakdowns;
        }
        else if (the length of core >= 2)
```

[115] In the actual implementation (§§5.3.4.1, 5.3.4.4; method `MorphologicalAnalyser.connect`), a `StringBuilder` is created from `wholeWord` and the deletions are performed on the `StringBuilder`, from which `core` is then extracted.
The final, considerably more complex multi-purpose version of this algorithm is implemented as `MorphologicalAnalyser.connect`. For discussion of variants using a `WordBreaker` see §§5.3.11.4, 5.3.17.4).

```
{
  if (dictionary contains core)
  {
    breakdown is a 3-element String array;
    breakdown[0] = candidate front;
    breakdown[1] = core;
    breakdown[2] = candidate back;
    breakdown is added to breakdowns;
  }
  else if (core.length() >= 4)
  {
    coreFronts is a candidate front List made from core;
    if (there are any candidates in coreFronts)
    {
      coreBacks is a candidate back List made from core
        backwards;
      if (there are any candidates in coreBacks)
      {
        the contents of coreBacks are reversed;
        String array coreBreakdown = analyse
          (core, coreFronts, coreBacks);
        if (coreBreakdown is not null)
        {
          breakdown is a String array
            with the number of elements in coreBreakdown + 2;
          index = 0;
          breakdown[index] = candidate front;
          index is incremented by 1;
          for (each element in coreBreakdown)
          {
            breakdown[index] = element ;
            index is incremented by 1;
          }
          breakdown[index] = candidate back;
        }
      }
    }
    if (breakdown is not null)
```

```
            {
              breakdown is added to breakdowns;
            }
          }
        }
      }
    }
  }
  return breakdowns;
}
```

5.2.2 Root Identification Algorithm

The purpose of the Root Identification Algorithm is to find the morphological root of an original word, using a pre-identified suffix from automatic suffix discovery (§5.3.7.3), with which the word ends. This task is complicated by the following uncertainties:

- the pre-identified suffix may be part of a longer suffix or contain a shorter suffix;
- there may be more than one morphological rule which could be applied;
- the original word may not be a suffixation.

5.2.2.1 Input and Output Classes

The Root Identification Algorithm returns a `POSTaggedSuffixation` (Class Diagram 11) representing the morphological root of an original word passed as a `POSTaggedWord` parameter. This may seem paradoxical but is a requirement because:

- a `POSTaggedSuffixation` stores both the original suffix of the word from which it is derived and the current suffix, which may be an empty `String` (a null suffix);
- a `POSTaggedSuffixation` also stores the `Relation.Type` of the `LexicalRelation` to be encoded between the original word (the derivative) and the `POSTaggedSuffixation` (the root).

The next subsection describes how the original algorithm determined the `POSTaggedSuffixation` to be returned.

5.2.2.2 Original Root Identification Algorithm

An initial check is made to see if the original word is a participle (adjective) or gerund (noun equivalent of participle). If so, the lemmatiser's exception map is interrogated to see if the original word has any irregular participle stems. If any is found, it is represented as a verb `POSTaggedSuffixation` (without any encapsulated morphological rule) of `Relation.Type.VERBSOURCE_OF_GERUND` (if the original word is a noun) or `Relation.Type.VERB_SOURCE` (if the original word is an adjective). The `POSTaggedSuffixation` generated is added to a `POSTaggedSuffixation` list.

If the original word is not a noun or adjective or if the above procedure adds nothing to the `POSTaggedSuffixation` list, and the pre-identified suffix with the original word's POS maps to any converse conditional morphological rule in the converse conditional morphological rule map (§5.1.1), then any such rules are executed (§5.2.2.3), adding 0 or more items to the `POSTaggedSuffixation` list.

If there is, by now at least 1 `POSTaggedSuffixation` in the list, each `POSTaggedSuffixation` is checked for the following validity criteria:

1. it has at least 2 letters;
2. it has a different word form from the original word (otherwise it will be handled separately by homonym analysis).

If any `POSTaggedSuffixation` fails this validity check, then the `POSTaggedSuffixation` is removed from the list.

If the `POSTaggedSuffixation` list is empty, and for as long as it remains empty, each converse morphological rule is considered in turn. If the original word ends with the suffix to be removed as specified by the rule, which in turn ends with the pre-identified suffix from automatic suffix discovery, and the POS specified by the rule for the suffix to

be removed is the same as that of the original word, then the rule is executed. For instance, if the pre-identified suffix is "-ion", the original word is "consumption" (noun) and the converse morphological rule maps from "-umption" (noun) to "-ume" (verb), then the rule will be executed and the `POSTaggedSuffixation` "consume" (verb) will be generated, encapsulating the original suffix "-umption" (noun) and the new suffix "-ume" (verb).

The same validity check is applied as described above, with the same consequences if it fails.

Once a morphological rule has generated at least one `POSTaggedSuffixation`, the first `POSTaggedSuffixation` in the list is always returned because it is deemed correct through the prioritising order of morphological rules (§5.1.4) and of the suffixes generated by the generalised spelling rules. If no `POSTaggedSuffixation` is generated then `null` is returned.

5.2.2.3 Morphological Rule Execution

The *Rule Execution Algorithm* was developed from the Suffix Stripping Algorithm employed during the pilot study (§3.2.2.2.2). The version presented here is a refinement of that Suffix Stripping Algorithm.

`Suffixer.executeReverseMorphologicalRule` executes a MorphologicalRule applying it to an original word with an original suffix, adding 0 or more `POSTaggedSuffixation`s to a List, each of which encapsulates a word form generated by replacing the original suffix of an original word with the rule's target.

If the original word is proper case it is changed to lowercase before the rule is executed unless the original suffix is "-er" as noun and the rule's target holds an empty `String` tagged as noun or the original suffix is "-ic" as adjective and the rule's target is tagged as

a noun. These exceptions are required to capture derivations for words such as "Londoner" and "Vedic".

If the rule's target is an empty `String`, a default stem is obtained by removing the original suffix from the end of the original word and placing the truncated word in an array of new word forms by default, subject to generalised spelling rules (Appendix 14), which generate alternative array elements overriding the default. If the rule's target is a non-empty `String`, a single new word form is generated by replacing the original suffix with the rule's target at the end of the word to which suffix stripping is to be applied. Reference to generalised spelling rules is not required for this operation as the rules themselves specify exactly which new character sequence is to replace which original character sequence.

However many new word forms there are, each is represented as a `POSTaggedSuffixation` encapsulating the `MorphologicalRule`, its `Relation.Type` and the `Wordnet.PartOfSpeech` specified by the rule's target.

Originally there was an automatic requirement that the output must be lexically valid. However, in secondary suffixation analysis (§5.3.14), this requirement does not apply, so `Suffixer.executeReverseMorphologicalRule` (morphological rule execution) has been modified to take a Boolean parameter specifying whether the output must be lexically valid.

5.2.2.4 Iterative Development of the Root Identification Algorithm

The straightforward procedure described above (§5.2.2.2) was applied in initial suffixation analysis (§5.3.7.3) with pre-identified suffixes, from successive suffix sets drawn from successive `SuffixTree` (§5.3.7.1) constructions from successive versions of the rhyming dictionary and the underlying atomic dictionary. Modifications to the procedure were developed iteratively in response to observed patterns of overgeneration and undergeneration in the output from suffixation analysis (§5.3.7.4) and subsequently

in response to the requirement to apply the procedure in circumstances where lexically valid output was not required, as in secondary suffixation analysis (§5.3.14). This iterative development also involved the specification of additional morphological rules to handle new suffixes drawn from successive of `SuffixTree` constructions (§5.1.3). Iterative development of the morphological analyser as a whole is discussed at the start of §5.3.

5.2.2.5 Final Version of the Root Identification Algorithm

The final version of the algorithm, the outcome of several iterative development cycles has the following modifications:

- Prepositions as well as adjectives are checked to see if they are irregular participle stems.

- In addition to checking for irregular participle stems, if the original word is an adjective or adverb then the lemmatiser's exception map (Appendix 65) is interrogated to see if the original word has any irregular stems of which the original word is the comparative or superlative form or irregular adjective stems of which the original word is the derived adverb. If any of either of these kinds of irregular stem are found, it is represented as a `POSTaggedSuffixation` of `Relation.Type.ADJECTIVE_SOURCE` (without any morphological rule) and added to the `POSTaggedSuffixation` list.

- Morphological rules are executed, with a Boolean lexical validity requirement (§§5.1.4) passed as a parameter to the Root Identification Algorithm.

- After each conditional rule is executed, the last `POSTaggedSuffixation` added to the list is checked to see whether it is monosyllabic. If the `POSTaggedSuffixation` is monosyllabic, and either the rule is inapplicable to

monosyllables (§5.1.1) or the lexical validity requirement parameter is false (§5.3.14.1), then the `POSTaggedSuffixation` is removed from the list.

- The validity check has a third criterion, that the original word does not map to the `POSTaggedWord` equivalent of the `POSTaggedSuffixation` in the suffix stripping *stoplist* supplied to the procedure and developed in response to observed instances where rules do not apply (§§5.3.7.4, 5.3.14.2).

- If a `POSTaggedSuffixation` fails the validity check, and the lexical validity parameter is false, then it is not deleted but marked as *unsuitable,* so that it can subsequently be reviewed by other criteria, prior to encoding any relation between the original word and the `POSTaggedSuffixation` (§5.3.14).

- If the `Relation.Type` of the `POSTaggedSuffixation` returned, passed to it by the rule which generated it, is `Relation.Type.DERIV`, representing a non-directional morphological relationship (this `Relation.Type` is inherited from WordNet, where it does not specify the direction of derivation), then this is changed to `Relation.Type.DERIVATIVE` if the POS-specific Brown Corpus frequency of the original word is greater than that of the `POSTaggedSuffixation`, or to `Relation.Type.ROOT` if the POS-specific Brown Corpus frequency of the original word is less than that of the `POSTaggedSuffixation`.

- Each converse morphological rule is tried in turn in the following specific manner designed to catch omissions by earlier versions:
  - A current list of rules is defined as all those to which the suffix to be removed as specified by the rule maps in the converse morphological rules map. These are pre-arranged in order of precedence (§5.1.4).
  - If there is more than one morphological rule in the current list and the lexical validity parameter is false, then the unique morphological rule, to which the suffix maps in the converse non-lexical morphological rules map (§5.1.5) is added to the current list of rules.

- The rules in the current list of rules are executed in turn, with the Boolean lexical validity requirement passed as a parameter to the Root Identification Algorithm overridden by true, except for the final rule, which, if it was added from the converse non-lexical morphological rules, will be executed with the Boolean lexical validity requirement passed as a parameter to the Root Identification Algorithm.
- Exceptionally, for a few suffixes for which optimal ordering of the rules cannot be relied upon to give satisfactory results, a *frequency-based modification* is employed (§5.2.2.6, Appendix 37).

5.2.2.6 The Frequency-based Modification

Optimal ordering of the applicable rules gives unsatisfactory results for suffixes "-ical" as an adjective, "-ician" as an noun, "-able" as an adjective, and "construction" as a noun. This is addressed by applying the *frequency-based modification*[116]. This creates a shortlist from the current list of rules and executes the rules in the shortlist, but only that `POSTaggedSuffixation` which has the greatest Brown Corpus frequency out of the those generated is added to the `POSTaggedSuffixation` list. Numeric parameter *last resort count* (`underrideAtEnd`) is passed to the frequency-based algorithm. The last resort count parameter specifies the number of rules at the end of the current list which are to be excluded from the shortlist. If execution of the shortlisted rules does not produce any `POSTaggedSuffixation`, then the excluded rules at the end of the current list are executed and the results are added to the `POSTaggedSuffixation` list. The last resort count was individually tuned for each suffix. It is set to 0 for "-ical" as an adjective and "construction" as a noun, 1 for "-ician" as an noun and 2 for "-able" as an adjective. This gives satisfactory results except for the suffix "-ical" as an adjective, to which a further modification has been applied where an initial attempt is made to execute the first morphological rule in the current list: if this is successful then the other rules are ignored.

[116]implemented as `Suffixer.selectDesuffixationByFrequency`.

5.3 Implementation of Morphological Analysis and Enrichment of the Lexicon

A complete morphological analysis of the words and phrases in the lexicon requires the analysis of compound expressions (multiword expressions and hyphenations) and concatenations into their constituent words and the analysis of affixations into their constituent morphemes, which may or may not also be words. The morphological enrichment of the lexicon requires the encoding of relations between compound expressions (§5.3.2) and concatenations (§5.3.4) and their constituent words, and between affixations and the words and the meanings of the morphemes from which they are derived (§§5.3.5.3, 5.3.7.3, 5.3.11.7).

Fundamental differences between non-antonymous prefixations on the one hand and suffixations and antonymous prefixations on the other have already been observed (§§3.2.3, 3.5.1). these differences are summarised in Table 39.

Table 39: Affixation properties

Property	**Non-antonymous Prefixations**	**Suffixations and Antonymous Prefixations**
Rules required	Only generalised spelling rules	Complex application rules
Semantic contribution	Independent meaning component	Define relation upon stem
Inheritance	Dual	Single
Word class	Preserve	Modify
Affix class	Preposition or noun	None
Affix-stripping precedence	Secondary	Primary

Because of these differences, the way in which relations are encoded in each case will differ. In the case of suffixations (§5.3.7.3) and antonymous prefixations (§5.3.5.3), a single relation can be encoded between each affixation and the word or stem from which

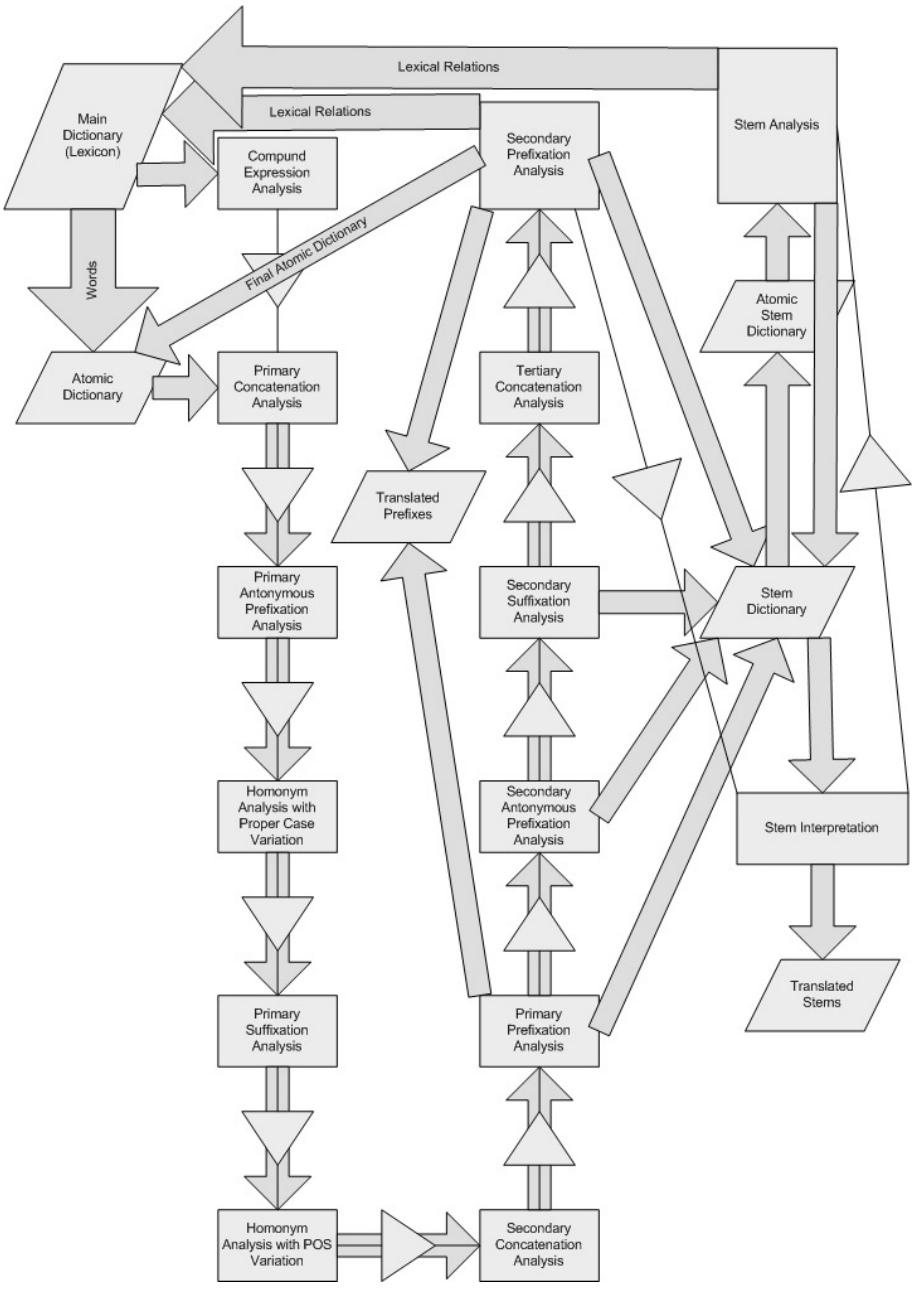
it is derived, as determined, in the case of a suffixation, by the relevant morphological rule and, in the case of an antonymous prefixation, by the application of general spelling rules. The type of relation encoded will be ANTONYM in the case of antonymous prefixations and in the case of suffixations it will be specified by the morphological rule. In the case of non-antonymous prefixations, two relations can be encoded, one between the prefixation and its stem, which may or may not also be a word and one between the prefixation and the meaning of the prefix (§5.3.11.7). Relations can also be encoded between stems and their meanings (§5.3.17.3.2), thereby reconnecting those stems which are not words to the lexicon.

The application of the rules and algorithms described in §5.1 and §5.2 needs to be supervised in such a way as to avoid the encoding of false derivational relations where exceptions apply. This can be achieved by the deployment of lists of exceptions (*stoplists*), which need to be created in response to the errors discovered from the output of each phase of the analysis of the English language. This requires iterative development of the model, where the stoplists created in response to errors are fed back into the model before proceeding onto the next phase of development. This approach leads to consistent precision estimates of 100% on the final output from each phase of morphological analysis, wherever the initial output has been fully reviewed. This 100% precision can be contested on linguistic grounds of disagreement with the manual evaluation of results, where there is room for individual interpretation. Apart from compound expressions analysis, the morphological analysis is itself iterative (§§5.3.4-5.3.16), partly because the stems from affixation analysis may themselves be affixations, but mainly because the assumed precedence of concatenation analysis over affixation analysis (§3.5.2) frequently does not apply, largely because many affixes comprise character sequences identical to unrelated words (§5.3.4.2). The assumed precedence of concatenation analysis has been retained in the interests of minimising manual intervention through the compilation of stoplists, thereby maximising automation.

The sequence of morphological analysis phases (Fig. 9) was primarily determined by precedence considerations (§3.5), corroborated by a review of the contents of the atomic

Fig. 9: Dataflows and sequence of morphological analysis phases

(Wide arrows represent dataflows; lines carrying triangles represent the sequence of execution; rectangles represent analysis phases; parallelograms represent data stores. The dataflows shown are simplified for clarity: lexical relations are generated from every phase of the analysis; the dataflow from each phase to the next is held in the atomic dictionary[117], which is modified at the end of each phase by removal of the words analysed..)



[117] The rhyming dictionary (not shown) is maintained in a state consistent with the atomic dictionary.

dictionary (§5.3.3.1) on completion of development of each phase. Further details of considerations impacting on sequencing decisions are discussed at the beginning of each subsection describing a phase in the analysis. Although the model has been developed iteratively, the analysis, combining unsupervised automatic affix discovery with the supervised application of the rules and algorithms developed, can be described sequentially, because the order in which the requisite iteratively developed analysis phases are executed corresponds to the order in which they were developed. The major iterations in the analysis itself will be presented sequentially as primary, secondary and tertiary phases of processes which are fundamentally the same but subject to some modifications. To avoid confusion, the present tense will be preferred for the description of software behaviour in the course of the *execution* process of *successful* experiments, while the past tense will be preferred for the discussion of development decisions, particularly where manual intervention was involved, and for the description of software behaviour in the course of the *development* process, including *unsuccessful* experiments.

5.3.1 Software Design for Morphological Analysis

The morphological analysis described here uses some classes developed for the earlier experiments with automatic affix recognition (§3.4) and morphological rule implementation (§3.2.2.2), some of which have been modified or extended as subclasses[118] (Appendix 1; Class Diagrams 10 & 11).

Morphological analysis is performed on a lexicon, with the modified design (§3.5.3; Class Diagram 7), based on the pruned WordNet model, enriched with prepositions (§4) but without any sentence frames[119]. The same lexicon is enriched with lexical relations connecting entries with their morphological roots at the end of each analysis phase.

---

[118] These classes are held in three packages `Morphology` (containing general utilities), `Morphology.automaticAffixDiscovery` and `Morphology.ruleBased`. An interface hierarchy provides an orthogonal grouping of component classes: interface `AffixRepresentation` groups classes which represent affixes (`Affix, AffixString, AntonymousPrefix, POSTaggedAffix, POSTaggedSuffix, Prefix, PrefixString, Suffix, SuffixString, TranslatedPrefix`); interface `Root` groups classes which represent stems (`POSTaggedStem, Stem, TranslatedStem`).

[119] loaded from file *bearnet.wnt.*

5.3.2 Compound Expression Analysis

The term compound expression refers to multiword expressions or phrases and hyphenated word combinations. These are both amenable to morphological analysis, being derived from their component words. Compound expression analysis is logically the first phase of morphological analysis, since all other entries in the lexicon are single words, into which compound expression analysis divides the compound expressions. Since multiword expressions can contain hyphenations, but hyphenations cannot contain multiword expressions, it is logical to start with multiword expression analysis and then proceed to hyphenation analysis. Morphological enrichment involves encoding lexical relations between each compound expression and its component words. The POS of each compound expression is given by WordNet, but the POSes of the component words are not. The relations encoded will be more precise if the POSes of the component words can be determined.

5.3.2.1 Multiword Expression Analysis

A *possibility map* is generated comprising mappings from multiword expressions to `LexicalPossibilityRecord` lists. Each `LexicalPossibilityRecord` represents the lemma of a component word of the multiword expression as all its possible POSes as found in the lexicon.

A customised, logic-based algorithm[120] was developed to find the correct POS for each component of every multiword expression, taking account of the number of components, the POS of the multiword expression as defined in WordNet and of those other components of the same multiword expression which have only one possible POS and of the possible POSes of the others, rejecting various sequences of POSes as implausible, given the POS of the multiword expression. Expressions are analysed starting by default

[120] Confidence in off-the shelf products was at a low level after experiments with the Stanford Parser (http://nlp.stanford.edu/software/lex-parser.shtml; §2.4); it seemed likely to be both easier and more effective to write an algorithm customised to the specific requirements. The precision achieved vindicates this decision.

from the last word and proceeding towards the first word. The algorithm was developed in the integrated development environment, without any preconception or initial design. Development began from manual parsing of sample multiword expressions, finding the most frequently occurring patterns and assuming that these patterns applied to all the multiword expressions whose components had the same sequence of sets of possible POSes. The algorithm was developed further through an iterative interactive process of sampling the results, observing the common properties of the incorrect results and inserting additional logic to handle them, until an overall accuracy of 96.5% was achieved. The complexity of the algorithm does not lend itself to a straightforward description and anyone interested is referred to the code where it was originally formulated, in Java[121].

Because of its complexity and the relatively insignificant impact it has on the encoding of lexical relations, the POS-tagging algorithm will not be discussed further. It has been retained because of its high precision, but multiword expression analysis can easily be modified to ignore it, the only consequent difference being that relations between multiword expressions and their components would be encoded as non-POS-specific. Where the POSes of the components of a multiword expression cannot be determined by the algorithm, the whole multiword expression is written, as a `POSTaggedMorpheme`, to a set of failures. Where the POSes of the components can be determined, an entry is added to a *compound expression map*, mapping from each multiword expression to a list of `POSTaggedMorpheme` components.

The multiword expression encapsulated in each `POSTaggedMorpheme` in the set of POS identification failures is split into its components and each component is checked against the `LexicalPossibilityRecord` to which the `POSTaggedMorpheme` maps in the possibility map. Components which match the word form in a `LexicalPossibilityRecord` and which do not start with a non-alphabetic character are added to a component list. A mapping is then created from the `POSTaggedMorpheme`

[121] `MorphoSemanticWordnetBuilder.analyseMultiwordExpressionComponents`

representing the multiword expression to its component list and added to an unidentified components map.

Relations are encoded between each multiword expression in the compound expression map and each of its components, specifying the POS of the component and between each multiword expression in the unidentified components map to each of its components, without specifying the POS of the component (Appendix 18).

5.3.2.2 Hyphenation Analysis

Hyphenations are analysed in the exactly same way as multiword expressions except that no attempt is made to identify the component POSes[122]. Although an attempt has been made to find the POSes of the components of hyphenations using the same algorithm as for multiword expressions, the results are only 91.4% correct and this is not considered sufficiently precise to justify encoding relations between hyphenations and their components as POS-specific. This failure reflects the fact that the components of a hyphenation are not required to fit into the overall syntax of their sentential contexts in the same way as the components of multiword expressions. The identification of a set of words in a context as a multiword expression is arbitrary and lexicographers will differ as to which word sequences they consider to merit dictionary entries, though *n*-gram counts in a corpus provide an empirical guide. A hyphenation on the other hand manifests itself physically in a context and lexicographers can use frequency evidence directly to determine when to incorporate them into dictionaries.[123].

[122] Methods `MorphoSemanticWordnetBuilder.processMultiWordExpressions()` and `MorphoSemanticWordnetBuilder.processHyphenations()` are identical, except that Boolean parameter `pOSSpecific` of method `lexicon.encodeLexicalRelationsFromMorphemelists` is set to true in `processMultiWordExpressions()` and false in `processHyphenations()` so that POSes are ignored.
[123] It was naively assumed that all hyphenation components would occur in the lexicon. Were this not been the case, a fatal exception would be thrown. In retrospect, it is questionable whether all hyphenation components truly correspond to the matching lexicon entries; this thesis, for instance, contains hyphenations whose first element is a prefix. This realisation calls for further research.

5.3.3 Construction of the Atomic and Rhyming Dictionaries

5.3.3.1 Atomic Dictionary

All subsequent morphological analysis operations apply to single words which are analysed into their constituent parts, namely other words, morphemes or non-lexical stems. These stems may themselves be combinations of morphemes, which are in turn analysed into their constituents (§5.3.17.4). In order to exclude multiword expressions and hyphenations from these analyses but include words until they have been analysed but exclude them thereafter, a separate data structure is required, containing all those words which have not yet been analysed, giving their possible POSes. This is called the atomic dictionary, because in theory, at the end of the analysis it should contain only atomic words, which cannot be broken down into meaningful constituents.[124]

The atomic dictionary does not require the same complex structure as the main dictionary, as there is no need to duplicate the information which connects entries to the wordnet nor any need to encode relations between the items contained in the atomic dictionary. The only information needed in the atomic dictionary is the set of possible POSes for each word form as recorded in the main dictionary. Consequently it is implemented as a `Map<String, Set<Wordnet.PartOfSpeech>>`. The atomic dictionary is initially created so as to contain all those keys to entries in the main dictionary which comprise a single unhyphenated word, mapping to their possible POSes. When a word has been analysed into at least two components, the word is removed from the atomic dictionary. Components which are words in their own right will already be in the atomic dictionary; those which are not words in their own right will be handled in a number of ways detailed in §§5.3.5-5.3.17.

The atomic dictionary is temporary and mutable. It progressively decreases in size until it contains only words which cannot be analysed, which will be either morphological roots

[124] For how far this is achieved in practice, see §§5.3.17.1, 5.3.18.

which cannot be further analysed or foreign loan-words which obey different morphological rules proper to their languages of origin or to the precursors of those languages. Many words of foreign origin can however be successfully subjected to morphological analysis as many morphological phenomena are common to multiple European languages, (Appendix 9).

5.3.3.2 Rhyming Dictionary

The concept of a rhyming dictionary has already been introduced (§3.4.2.1) as a tool for automatic suffix recognition. In the context of a complete morphological analysis of a language, however, it is not required during compound expression analysis. The rhyming dictionary used for subsequent operations is derived from the atomic dictionary. It must be updated after any operation which removes an analysed word from the atomic dictionary, before it is accessed again. Some operations remove the entry for the reversed word form from the rhyming dictionary immediately after removing the entry for the normal word form from the atomic dictionary, but in many cases it is sufficient, and easier, to rebuild the rhyming dictionary after the completion of a particular phase of morphological analysis. Analysis is facilitated by including part of speech information in the rhyming dictionary and so it too is implemented as a `Map<String, Set<Wordnet.PartOfSpeech>>`, identical to the atomic dictionary except that the word forms which are its keys are reversed.

5.3.4 Primary Concatenation Analysis

A concatenation is a word which wholly consists of a sequence of 2 or more other words, from which it is derived both etymologically and semantically. A precedence of concatenation analysis over affixation analysis has been assumed (§3.5.2) because the words into which concatenation analysis divides concatenations can themselves be affixations, whereas no instance of an affixation, among whose components there is a concatenation, readily comes to mind. In theory, it should be straightforward to analyse each concatenation into its component words, using the Word Analysis Algorithm, in its

simplest form (§5.2.1). In practice however the Word Analysis Algorithm tends to overgenerate, because many affixes are lexically identical to words to which they are etymologically and semantically unrelated (§5.3.4.2), so that a correct segmentation of the word is frequently not a correct concatenation analysis because the word is an affixation, not a concatenation. The remainder of this section is concerned with the correction of this overgeneration and selection of the optimal analysis when more than one analysis is possible.

5.3.4.1 Original Concatenation Analysis Procedure

Two maps `candidatesWithFronts` and `candidatesWithBacks` are created mapping from each word in the atomic dictionary to its candidate lists as described in §5.2.1.3. The Word Analysis Algorithm is then applied to each word in the atomic dictionary and the results are stored in a concatenations map[125], comprising mappings from concatenations to lists of components, each list representing a possible analysis of the word. The contents of the concatenations map are written to file[126] (for output file formats see Appendix 19).

The analysis procedure limits the number of possible analyses of a concatenation to one. To achieve this, a selection procedure takes place. The selection procedure works on the following assumptions:

1. there are never more than 2 alternative analyses;
2. the number of components in the first analysis is unequal to the number of components in the second analysis unless that number is 2;
3. where both analyses have 2 components, then either the first component of one array will end with "s" or the combined *Brown Corpus frequency* of the components of each analysis will differ.

If any of these assumptions are violated, then all analyses are rejected.

[125] `Map<String, Morpheme[]>`
[126] *Concatenations with components.csv*

The selection procedure works as follows: since further analysis is possible, where the analyses have different numbers of components, the analysis with the fewest components is accepted and the other is rejected. If 2 alternative analyses have 2 components each, then if the first component of only one of the analyses ends with "s", that analysis is selected, otherwise the analysis is selected whose components have the highest combined Brown Corpus frequency.

5.3.4.2 Initial Results from Primary Concatenation Analysis

11115 words were analysed by the first attempt at applying the above procedure. The maximum number of components discovered was 5. At a glance (Table 40), it was immediately apparent that the procedure produced more incorrect results than correct.

Table 40: First 20 initial results from concatenation analysis

Whole word	First component	Middle component	Last component	Evaluation
abhorrent	abhor		rent	Incorrect
abjection	abject		ion	Incorrect
ableism	able		ism	Incorrect
abolishable	abolish		able	Incorrect
abolitionism	abolition		ism	Incorrect
aboveboard	above		board	Correct
aboveground	above		ground	Correct
abruption	abrupt		ion	Incorrect
absentminded	absent		minded	Correct
absorbable	absorb		able	Incorrect
abstraction	abstract		ion	Incorrect
abstractionism	abstract	ion	ism	Incorrect
abstractionism	abstraction		ism	Incorrect
academically	academic		ally	Incorrect
academicism	academic		ism	Incorrect
acceptability	accept		ability	Incorrect
acceptable	accept		able	Incorrect
acceptably	accept		ably	Incorrect
acceptant	accept		ant	Incorrect
acceptation	accept	at	ion	Incorrect

Of the 20 results in Table 40, only 3 are correct, namely "above-board"," above-ground" and "absent-minded". The first component is correct in every case, but all remaining 17

last components are wrong and the two middle components are also wrong. Suffixes "-ion", "-ism", "-able", "-ally", and "-ability" have been treated as whole words. Of these, "ion" and "ally" as whole words bear no relation to the suffixes. The words "able" and "ability" are obviously closely related to the corresponding suffixes and the word "ism" was coined from the suffix, but these connections do not make these outputs acceptable: suffixations require processing in a different way to concatenations (§5.3.7). In "abhorrent", "-rent" has been treated as a whole word, when it is of course suffix "-ent" preceded by a reduplicated "r". The 2 instances where a word has been divided into 3 are cases of double suffixation. These kinds of errors occurred throughout the data.

Out of 79 words beginning with "ad-", 57 were treated as having the word "ad" (abbreviation for "advertisement") as their first component (Appendix 39). In none of these cases is this analysis correct; most of them are instances of prefix "ad-". The results where recursion had occurred (Tables 41-42) were again unacceptable:

Table 41: First 10 initial results from recursive concatenation analysis

Whole word	First component	Second component	Penultimate component	Last component	Evaluation
amphiprostyle	amp	hi	pro	style	Incorrect
arthroscope	art	hr	os	cope	Incorrect
arthroscopy	art	hr	os	copy	Incorrect
arthrospore	art	hr	os	pore	Incorrect
arthrosporous	art	hr	os	porous	Incorrect
asseveration	ass	eve	rat	ion	Incorrect
autofluorescent	auto	flu	ore	scent	Incorrect
automatonlike	auto	ma	ton	like	Incorrect
automatonlike	auto	mat	on	like	Incorrect
bagassosis	bag	as	so	sis	Incorrect

Table 42: Complete initial results from 5-component recursive concatenation analysis

Whole word	First component	Second component	Middle component	Penultimate component	Last component
enterostenosis	enter	os	te	no	sis
inconsideration	in	con	side	rat	ion
instrumentation	in	strum	en	tat	ion
intentionally	in	ten	ti	on	ally
lackadaisically	lack	ad	ai	sic	ally
reduplication	red	up	li	cat	ion

5.3.4.3 Candidate Component Filtration

It was clear however that these erroneous results did not signify that affixation analysis should take precedence over concatenation analysis. Such an approach would produce even more erroneous results (§3.5.2). What was required was to create *stoplists* containing known prefixes and suffixes where they occurred as words in these initial results (as well as any other words which were wrong), so as not to generate these false analyses, on the understanding that concatenation analysis would be repeated (without the same stoplists) after initial affixation analysis. In order to limit the size of the stoplists required, *frequency corroboration* was introduced into the creation of candidate lists (§5.2.1.3), so that words with a recorded Brown Corpus frequency < 1 were excluded from the candidate lists.

A *first component stoplist* was created, comprising 312 words (Appendix 40) but it turned out that a *last component stoplist* would contain more than half the words which appeared as last components and so it would be more economical to use a *startlist* of words from which any last component must be selected. This comprises 986 words (Appendix 41).

The erroneous last components from the initial results from primary concatenation analysis, which would have formed the last component stoplist, were employed to populate the *false lexical stem set*, (Appendix 38), used for filtering out non-lexical stems (§5.3.11.7) prior to encoding relations between prefixations and their stems. This set was subsequently modified to specify the POSes of the stems as discovered through prefixation analysis.

It is debatable, when the first component of a word is an English preposition (e. g. "after") and the remainder of the word is a whole English word, whether we are dealing with a prefixation or a concatenation. Decision on this question, which would determine how such words are analysed, was deferred (see §5.3.11.3), by including such prepositions in the first component stoplist.

5.3.4.4 Revised Procedure for Primary Concatenation Analysis

In the revised procedure, each candidate front which matches a word in the first component stoplist[127], is removed from `candidatesWithFronts` and each candidate back which does not match a word in the last component stoplist[128] is removed from `candidatesWithBacks` before the analysis.

Since the results from recursion (§§5.2.1) showed no sign of being helpful and filtration is applied only to the first and last component, recursion is suppressed in the revised procedure, and the number of morphemes in the `Morpheme` array generated for each word is limited to two. This still allows for further analysis of the components at a later stage.

If an analysis is produced comprising a valid initial word and a valid final word separated by an "s", then, exceptionally, the "s" is dropped as it is regarded as an inflectional suffix (e. g. "woodsman" is analysed into "wood" and "man".

5.3.4.5 Encoding of Lexical Relations between Concatenations and their Components

After writing to the output files, each concatenation in the concatenations map is looked up in the main dictionary to discover all its possible POSes. A `POSTaggedMorpheme` is then created for each of these POSes. A mapping from each `POSTaggedMorpheme` to a list of its components, read from the concatenations map is added to a second concatenations map[129]. The concatenation is removed from the atomic dictionary and its reversed form is removed from the rhyming dictionary.

The second concatenations map, in which each mapping maps from a `POSTaggedMorpheme` representing the concatenations to a list of its components, is used

[127] file *Concatenation first component stoplist.txt*
[128] file *Concatenation last component startlist.txt*
[129] `Map<POSTaggedMorpheme, List<String>>`

for encoding relations between each concatenation and its components. (Appendix 18). The analysed concatenations are removed from the atomic dictionary.

4116 concatenations are analysed with the stoplists in place. The stoplists ensure 100% precision. Recall of 65% can be inferred from the number of concatenations which remained unanalysed until subsequent phases of concatenation analysis.

5.3.5 Primary Antonymous Prefixation Analysis

While the atomic dictionary may still contain some valid concatenations, these will all contain exceptional morphemes which could be affixes. It is therefore necessary to embark upon affixation analysis, with the awareness that some apparent affixations may in fact really be concatenations. Affixation analysis starts with the precedence rules established that antonymous prefix stripping takes precedence over suffix stripping which in turn takes precedence over non-antonymous prefix stripping (§3.5.1).

5.3.5.1 Hazards of Antonymous Prefixation Identification

The precondition for antonymous prefix stripping is to identify which prefixes are antonymous. A provisional list compiled from footprints from the original automatic prefix discovery (§3.4.1) agreed with Kwon (1997). The best known antonymous prefixes are "non-" and "un-", which are always antonymous except when they are really parts of longer prefixes (Appendix 42). The irregular prefix "in-" is sometimes antonymous and sometimes not. It is referred to as irregular because it has various footprints (§§3.2.2.3, 3.4.1.3) corresponding to *sandhi* spelling modifications as follows:

"in-" + "b" = "imb-"

"in-" + "l" = "ill-"

"in-" + "m" = "imm-"

"in-" + "n" = "ign-"

"in-" + "p" = "imp-"

"in-" + "r" = "irr-".

Prefix "a-" is generally antonymous but modifies to "an-" before a vowel. Obviously not all words beginning with "a-" have an antonymous prefix. Prefix "anti-" is antonymous and can be abbreviated to "ant-" as in "antacid" but must not be confused with non-antonymous prefix "ante-". Prefixes "dis-", "de-" may sometimes be antonymous, "dis-" being an Anglo-Norman modification of "de-". Both can have a meaning of "away from" and the boundary between this meaning and antonymy is fuzzy. The same goes for "contra-", with a primary meaning of "against", its abbreviation to "contr-" before a vowel and its Anglo-Norman variant "counter-". Kwon (1997) considers "anti-", "counter-" and "de-" to be extras, rather than true antonymous prefixations. All these prefixes are stored in a constant `String` array of antonymous prefixes[130], but words which begin with them are not automatically treated as antonymous prefixations, the task of identifying which is hampered by the aforementioned complications which can be summarised as follows:

1. Some antonymous prefixes have spelling variants;
2. Some prefixes are only sometimes antonymous;
3. In some cases the boundary between antonymy and non-antonymy is fuzzy;
4. An apparent prefix can be part of a longer prefix or word.

The issue of spelling variants was addressed by including all of these in the antonymous prefixes array (but see also §5.3.5.3).

5.3.5.2 Morpheme and Whole Word Exceptions and Counter-Exceptions

The issue of prefixes being parts of longer prefixes was addressed by introducing, in addition to the obvious concept of a *whole word exception*, the concepts of *morpheme exception*, *whole word counter-exception* and *morpheme counter-exception*. Thus although "a-" is an antonymous prefix, "ab-" is a non-antonymous prefix in its own right,

[130] {"un", "in", "imb", "ign", "ill", "imm", "imp", "irr", "dis", "de", "counter", "contra", "contr", "non", "anti", "ant", "an", "a"}

so "ab-" is a morpheme exception. However some words beginning with "ab-" do not begin with prefix "ab-", but with antonymous prefix "a-" followed by "b", as in "abiogenesis" and "abasic". These are whole word counter-exceptions. Moreover antonymous prefix "a-" can modify to "ab-" before "n" as in "abnormal", so "abn-" is a morpheme counter-exception. Some words beginning with "ab-" have a non-antonymous "a-" prefix as in "aback" and "ablaze". These can be ignored (for now but see §§5.3.11.2, 5.3.11.5) as they are covered by the general "ab-" morpheme exception.

Now take the case of words beginning with "an-", which is a spelling modification of antonymous prefix "a-" before a vowel, but can also represent antonymous prefix "a-" followed by "n". Non-antonymous prefix "ana-" is a morpheme exception, but there are whole word counter-exceptions where antonymous prefix "an-" occurs before "a" as in "anaemia" and "anarchic". Non-antonymous prefix "ante-" is another morpheme exception, but "anti-" is another antonymous prefix in its own right, with morpheme exception "antiqu-" as in "antiquarian" and "antiquity".

In practice it is not necessary to list all these exceptions and counter-exceptions, because antonymous prefixation, at this stage, is only considered as a possibility if a valid word can be discovered by removing the prefix.

Whole word exception lists can also handle the problem of sometimes antonymous prefixes, such as "in-" and its spelling modifications. To deal with these required a manual review of every word in the atomic dictionary beginning with "ign-", "ill-", "imb-", "imm-", "imp-", "in-" and "irr-" and classify them as antonymous or non-antonymous. This work was necessary in any case to deal with irregular non-antonymous prefixation (§5.3.11) Uncertain cases were referred to the OED2, backed up by OED1 and Burchfield (1972).

All words beginning with "un-" were examined likewise (Appendix 42). Morpheme exceptions identified included "uni-", with numerous whole word counter-exceptions and "under-", with morpheme counter-exception "underiv-".

Having established the concepts of four different kinds of exception and built incomplete lists of each, to avoid having to perform a similar analysis on every word beginning with "a-" it was easier to proceed experimentally by encoding an algorithm for identifying antonymous prefixations and then to extend the exception lists on reviewing the resultant file[131], comprising pairs of antonymous prefixations and their non-prefixed equivalents (their candidate antonyms). All incorrect pairings were dealt with by adding an entry to the whole word exception list, or to the morpheme exception list with any further required entries added to the counter-exception lists[132]. All uncertainties were again checked against OED2, OED1 or Burchfield (1972). This procedure was repeated until satisfactory results were obtained. (Appendix 43).

5.3.5.3 Antonymous Prefix Identification Procedure

The antonymous prefix stripping procedure iterates through the constant `String` array of antonymous prefixes {"un", "in", "imb", "ign", "ill", "imm", "imp", "irr", "dis", "de", "counter", "contra", "contr", "non", "anti", "ant", "an", "a"}, and for each antonymous prefix it iterates through the atomic dictionary looking for words beginning with that antonymous prefix. When such a word is encountered, it is checked against the exception lists. If the word is in the whole word exception list, then an exception holds and nothing is done. If it starts with a morpheme listed in the morpheme exception list, then an exception holds and nothing is done unless it is listed in the whole word counter-exception lists or starts with a morpheme listed in the morpheme counter-exception list.

---

[131] *WordsWithAntonymousPrefixes.csv* (format in Appendix 19).

[132] The exception lists are held in the following files:

- *Antonymous prefix whole word exceptions.txt*;
- *Antonymous prefix morpheme exceptions.txt*;
- *Antonymous prefix whole word counter-exceptions.txt*;
- *Antonymous prefix morpheme counter-exceptions.txt*.

The ordering of the exception list files reflects the order in which the exceptions were discovered. The lists are re-ordered alphabetically when they are read from file and implemented as sets to eliminate any possible duplicates.

If no exception holds, either because the word is not in the whole word exception list, or because it does not start with a morpheme listed in the morpheme exception list, or because it is covered by a counter-exception, then the prefix is stripped off and the resulting word is looked up in the main dictionary. If it is found, a mapping from the prefixed word to its non-prefixed equivalent, considered as a candidate antonym, is written to an *antonymous prefixation map*, subject to a minimum length of 2 letters including at least 1 vowel. Prefix stripping is a simple matter of deleting the specified antonymous prefix, unless the antonymous prefix starts with "i" but is not "in-", in which case the last letter of the prefix replaces the first letter of the result. No other spelling rules are required for this operation. The contents of the antonymous prefixation map are written to file[133].

3444 antonymous prefixations are identified. Measures of precision and recall are inappropriate because of the fuzziness of the boundary between antonymous and non-antonymous prefixations (§5.3.5.1). The antonymous prefixations identified are removed from the atomic dictionary. Non-translating `ANTONYM` relations are encoded between each antonymous prefixation in the antonymous prefixation map to its unprefixed equivalent (Appendix 18).

5.3.6 Analysis of Homonyms with Proper Case[134] Variation

Because of the fuzziness of the distinction between antonymous and non-antonymous prefixations, and because of the problems caused by possible antonymous prefixes being sometimes identical to the first part of non-antonymous prefixes, completion of antonymous prefixation analysis needs to be deferred until after at least an initial phase of non-antonymous prefixation analysis. Given the precedence rule adopted (§3.5.1), the next phase should be suffixation analysis. However, it will simplify the rest of morphological analysis if as many proper case words as possible can be analysed first.

[133] *WordsWithAntonymousPrefixes.csv* (format in Appendix 19)
[134] first character in uppercase.

Since this analysis is applied to word forms and not to word senses, homonymy only arises in one of two scenarios:

1. where there is a case difference (in particular where one word is proper case, usually but not always a proper noun);
2. where the same word occurs as more than one POS.

In general, from observation of the data, polysyllabic proper case words with non-proper case homonyms of the same POS can be considered as derived from their non-proper case counterparts (Table 43), but non-proper case homonyms of monosyllabic proper case words are largely unrelated ("bill", "Bill"; "welsh", "Welsh"). Where a polysyllabic proper case word has no non-proper case homonym of the same POS, but has a proper case homonym of a different POS, then the homonyms can be treated in the same way as pairs of non-proper case homonyms with different POSes, which is as if the pair of homonyms was a pair of suffixations, both with null suffixes (meaning the suffixes are empty strings), the relationship between which is defined by a morphological rule. The lexical relation to be encoded between the homonyms has the relation type specified by the morphological rule. Such homonym pairs can be treated as special cases of suffixations. It is therefore appropriate that homonym analysis should take place in juxtaposition with suffixation analysis. On the basis of these observations, analysis of homonyms with proper case variation is now performed as described in this section.

5.3.6.1 Methodology for Homonyms with Proper Case Variation

The root of each possible POS of each proper case word in the atomic dictionary which has more than 2 letters is represented as a `POSTaggedMorpheme`, and a `POSTaggedSuffixation` is generated to represent its root[135] in one of three ways as follows.

1. If the third character of the word form is a capital, a null `POSTaggedSuffixation` is generated on suspicion that it is an acronym or abbreviation (the third character

[135] For the handling of back-formations please refer to §1.1.2 and notes.

is chosen to cover abbreviations comprising period-separated capitals such as "A.D.") .

2. Otherwise, if the lowercase form is in the main dictionary with the same POS as the original word,, a `POSTaggedSuffixation` is generated representing its lowercase form, `Relation.Type.ROOT` and no morphological rule.

3. If the lowercase form is not in the lexicon, then the `POSTaggedSuffixation` is generated by executing, with a positive lexical validity requirement, the first converse morphological rule which is applicable to a null suffix (whose target will always also be a null suffix) and to the POS of the original word such that the `POSTaggedSuffixation` will necessarily encapsulate a homonym of the original word if that word has any homonyms, otherwise a null `POSTaggedSuffixation` will be generated. The application of rules applying to null suffixes never generates more than one `POSTaggedSuffixation`.

The `Relation.Type` and `LexicalRelation.SuperType`[136] of the `LexicalRelation` encapsulated in the `POSTaggedSuffixation` determine whether the `POSTaggedSuffixation` is indeed the root of the original word or whether it is its derivative. However, if the `Relation.Type` is `Relation.Type.DERIV` indicating a directionless morphological relationship, this means that the rule cannot determine whether its source or its target is the root and the root is deemed to be the more frequent homonym. In technical terms this means:

- if the Brown Corpus frequency of the original word is greater than that of the `POSTaggedSuffixation` then the `Relation.Type` of the `POSTaggedSuffixation` is redefined as `Relation.Type.DERIVATIVE`;

[136] Every `LexicalRelation` has a `SuperType` to indicate the direction of derivation (either `ROOT` or `DERIV`). The `LexicalRelation.SuperType` must be consistent with the `Relation.Type`; see Appendix 1 under `LexicalRelation`).

- if the Brown Corpus frequency of the original word is less than that of the `POSTaggedSuffixation` then the `Relation.Type` of the `POSTaggedSuffixation` is redefined as `Relation.Type.ROOT`.

Since frequency information is not available for prepositions, if the original word is a preposition then the `POSTaggedSuffixation`'s `Relation.Type` remains unchanged and the direction of derivation remains indeterminate. The same applies if the 2 frequencies are equal.

If the `POSTaggedSuffixation` is monosyllabic then the `POSTaggedSuffixation` is replaced by a null `POSTaggedSuffixation`, because the application of homonym analysis to monosyllabic proper case words produces mostly false derivations.

A homonym map is created for each word analysed in which each `POSTaggedMorpheme` representing a particular POS of the proper case word maps to the morphologically related homonymous `POSTaggedSuffixation` generated by the above procedure. No mapping is created if the `POSTaggedSuffixation` is null (as for abbreviations and acronyms and monosyllables). No mapping is created from "Attic" to "attic" (the only morphologically unrelated pair found in the original results).

The POSes of any `POSTaggedSuffixation` in the homonym map whose encapsulated `Relation.Type` is not `Relation.Type.DERIV` or `Relation.Type.DERIVATIVE` are removed from the word's entry in the atomic dictionary as a homonymous derivational root has been found for it. If no `POSTaggedSuffixation` values in the map have `Relation.Type.DERIV` or `Relation.Type.DERIVATIVE`, then the entire entry for word is removed from the atomic dictionary, as homonymous derivational roots have been found for them all. For each entry in the homonym map, a row is written to file[137] (samples in Table 43). Manual review of the results showed that correct ordering of the morphological rules (§5.1.4) allows this method to reliably output the single best candidate for the homonymous root (or derivative) of the original word. 1386 homonym pairs are identified.

[137] *Primary Identical words Results.csv* (format in Appendix 19)

Table 43: Primary homonym result samples

POSTagged Morpheme		POSTagged Suffixation		Relation.Type	Morphological Rule	
Wordform	**POS**	**Wordform**	**POS**		**Source POS**	**Target POS**
Abecedarian	N.	abecedarian	N.	ROOT	n/a	n/a
Aramean	N.	Aramean	ADJ.	DERIV	N.	ADJ.
Bhutanese	N.	Bhutanese	ADJ.	DERIV	N.	ADJ.
Celtic	N.	Celtic	ADJ.	ROOT	N.	ADJ.
Deliverer	N.	deliverer	N.	ROOT	n/a	n/a
Frisian	N.	Frisian	ADJ.	DERIV	N.	ADJ.
Hunter	N.	hunter	N.	ROOT	n/a	n/a
Korean	ADJ.	Korean	N.	DERIV	ADJ.	N.
Marine	N.	marine	N.	ROOT	n/a	n/a
Negro	N.	negro	ADJ.	DERIVATIVE	N.	ADJ.
Phallus	N.	phallus	N.	ROOT	n/a	n/a
Rumanian	ADJ.	Rumanian	N.	DERIV	ADJ.	N.
Skinner	N.	skinner	N.	ROOT	n/a	n/a
Tudor	N.	Tudor	ADJ.	DERIVATIVE	N.	ADJ.

5.3.6.2 Encoding of Lexical Relations between Homonyms

If the `Relation.Type` of the `POSTaggedSuffixation` is `DERIVATIVE` or `ROOT`, a `LexicalRelation.SuperType` is defined to be `the same as that type`. If the `Relation.Type` is neither `DERIVATIVE` nor `ROOT`, then the `LexicalRelation.SuperType` is defined to be `ROOT` unless either the `POSTaggedMorpheme` is a verb or preposition or the `POSTaggedSuffixation` is a noun or adverb, in which case the `LexicalRelation.SuperType` is defined to be `DERIVATIVE`. This rule, defined from observation of the preliminary results, defines the direction of derivation, where this has not been determined from the morphological rules. Non-translating relations of the specified type and supertype are encoded between each `POSTaggedMorpheme` in the homonym map and the corresponding `POSTaggedSuffixation` (Appendix 18).

5.3.6.3 Rhyming Dictionary Revision

At this point, since the atomic dictionary has been modified without corresponding modifications to the rhyming dictionary, the rhyming dictionary is replaced with a new

one comprising the reversed word forms of the words currently held in the atomic dictionary, mapping to their POSes as recorded in the atomic dictionary. This procedure is repeated at intervals throughout the rest of the morphological analysis, whenever the atomic dictionary has been modified without corresponding modifications to the rhyming dictionary.

5.3.7 Primary Suffixation Analysis

Proper case words having been analysed, as far as possible, as being derived from their non-proper case counterparts, it is now possible to proceed to suffixation analysis, as having a lower precedence than antonymous prefixation analysis, but a higher precedence than non-antonymous prefixation analysis (§3.5.1). Suffixation analysis requires some kind of definition of what is and what is not a suffix. An empirical methodology for suffix identification has already been elaborated in §3.4.2.

5.3.7.1 Suffix Tree Construction

As compound expressions, concatenations, antonymous prefixations and proper case homonyms have already been analysed, the `SuffixTree` used here is constructed from the rhyming dictionary rebuilt from the atomic dictionary which excludes these, and not from a rhyming dictionary built from the main dictionary as described in §3.4.2. It is therefore not identical to the `SuffixTree` described there.

5.3.7.2 Primary Suffix Set

A primary suffix set[138] is created, comprising all the suffixes in the `SuffixTree`, ordered by a `Comparator<Affix>` which imposes a primary ordering by the optimal heuristic.

$$\frac{f_c^2 q_s}{f_p}$$

[138] `Set<Affix>`

where f_c = affix frequency, f_p = parent frequency and q_s = stem validity quotient (§3.4.5). A secondary ordering is imposed by affix frequency and a tertiary lexicographic ordering. The purpose of the primary suffix set is to prioritise those candidate suffixes which are most likely to satisfy the semantic criterion

A table is generated from the suffix set, each row of which represents a candidate suffix which has at least one child in the underlying `SuffixTree`. The columns in the table represent the following fields:

- orthographic form;
- f_c;
- $\frac{f_c}{f_p}$;
- $\frac{f_c^2}{f_p}$ (default heuristic);
- q_s;
- d = number of child Suffixes;
- f_p;
- $f_c - f_d$ (number of occurrences of child Suffixes in Lexicon).

The rows in the table are ordered in descending order according to the optimal heuristic. The table of suffixes comprises 26940 entries and is written to file[139].

5.3.7.3 Suffixation Analysis with Reference to Automatically Discovered Suffixes

Since the purpose of the primary suffix set is to prioritise those candidate suffixes which are most likely to satisfy the semantic criterion (§3.4) according to the optimal heuristic, a secondary suffix set is required which includes the semantically valid suffixes

[139] *Suffixes.csv* (format in Appendix 19)

prioritised while discarding the rest. This is achieved by selecting the first 100 suffixes. This decision is justified on the following grounds:

- the density of semantically valid suffixes in the primary suffix set trails off rapidly after the first 100;
- the outstanding semantically valid suffixes will be handled during secondary suffixation analysis;
- the 98% recall achieved (§5.3.7.4) confirms that 100 is a suitable threshold.

The secondary suffix set (Appendix 44) is arranged in descending order of suffix length with a secondary lexicographic ordering. Ordering by suffix length is essential to ensuring that child suffixes have priority over their parents, so that the suffix "-ion", for example will not be treated as an instance of the suffix "-on". A more code-like representation of the Suffixation Analysis Algorithm described here is in Appendix 21.

An outer loop iterates through the atomic dictionary, processing every word in turn. For each word, a `Map<POSTaggedMorpheme, POSTaggedSuffixation>` is created. A middle loop iterates through the possible POSes of the current word. For each POS the word is represented as a `LexiconLinkedPOSTaggedWord` with that POS. An inner loop iterates through the secondary suffix set, each member of which is considered as a pre-identified suffix. If any word ends with the pre-identified suffix then a `POSTaggedSuffixation` is generated representing the morphological root of the current `LexiconLinkedPOSTaggedWord` obtained through the Root Identification Algorithm using the pre-identified suffix with a positive lexical validity requirement (§5.2.2). The inner loop continues to iterate as long as no `POSTaggedSuffixation` has been generated and there remain untried suffixes in the set. When a `POSTaggedSuffixation` is generated representing the root of the `LexiconLinkedPOSTaggedWord`, then an entry is added to the map comprising the `LexiconLinkedPOSTaggedWord` as a `POSTaggedMorpheme` representing the original word and the `POSTaggedSuffixation` representing its root. When the inner loop terminates without any `POSTaggedSuffixation` being generated,

then nothing is added to the map, but a record is written[140] (for output file formats see Appendix 19).

Once the middle loop has finished iterating through the current word's POSes, another loop iterates through the map created, processing each entry. In this process, two further validity tests are applied:

1. any monosyllabic `POSTaggedSuffixation` generated by a rule inapplicable to monosyllables is rejected;

2. the `Relation.Type` of each `POSTaggedSuffixation` is checked. If its `Relation.Type` is `Relation.Type.DERIV` (indicating a directionless morphological relationship), then the `POSTaggedSuffixation` is deemed NOT to be the root of the `POSTaggedMorpheme` which maps to it and is rejected.

If the `POSTaggedSuffixation` is rejected, the POS of the `POSTaggedMorpheme` is retained in the entry in the atomic dictionary for the current word and no lexical relations are encoded, otherwise a row representing the result is written to file[141], the POS of the `POSTaggedMorpheme` is removed from the entry in the atomic dictionary and lexical relations are encoded. If the root `POSTaggedSuffixation` is monosyllabic, the same data is written to another file[142], preceded by the reversed word form of the original word, to facilitate reordering by original suffix.

Relations of the type specified by the morphological rule which generated the `POSTaggedSuffixation` are encoded between each derivative `POSTaggedMorpheme` and the corresponding root `POSTaggedSuffixation` (Appendix 18).

[140] to file *X1 unidentified roots.csv*
[141] *X1 Suffix stripping Results.csv* (format in Appendix 19)
[142] *X1 monosyllabic roots.csv*

If all POSes have been removed from the entry for the current word in the atomic dictionary, then the entire entry for the current word is deleted from the atomic dictionary.

5.3.7.4 Results from Primary Suffixation Analysis

The implementation of suffixation analysis, applying the Root Identification Algorithm to the words in the atomic dictionary using automatically pre-identified suffixes was first attempted using a set of morphological rules little changed since the pilot study (§3.2.2.1). As expected, there was massive undergeneration because rules involving languages other than English had not been applied. The data in the original unidentified roots file (§5.3.7.3) was used to inform the formulation of additional morphological rules (§5.1.3).

The original implementation had no stoplist, but overgeneration in the results, through successive cycles of iterative development, quickly demonstrated the need for one. False analyses informed the creation of the stoplist and the following modifications to the morphological rules:

- the specifying of some rules as inapplicable to monosyllabic roots (§5.1.1),
- the revision of some rules to specify longer source and target suffixes (§5.1.2) and
- the ordering of rules with a common source to apply precedence (§5.1.4)

The suffix stripping stoplist[143] passed to the Root Identification Algorithm (§5.2.2.5) is populated with data from file[144]. Each key in the stoplist comprises a `POSTaggedWord` encapsulating the false derivative word form as the false derivative POS; each value comprises a `List<POSTaggedWord>` containing the false roots of the key.

The process of primary suffixation analysis remains substantially the same as described in §5.3.7.3 except for modifications to the Root Identification Algorithm (§5.2.2.5). After

[143] `Map<POSTaggedWord, List<POSTaggedWord>>`
[144] *Suffix stripping stoplist.csv* (format in Appendix 20)

implementation of the changes to the ruleset and the Root Identification Algorithm and the implementation of the stoplist, the final results of this phase comprise analyses of 24534 suffixations written to file[145]. Of these 5117 have monosyllabic roots[146]. A precision of 100% may be contested as there is room for lexicographic interpretation as to exactly what is and is not a suffixation. Subject to the same caveat, recall is inferred from the results of subsequent phases to be 98%.

5.3.8 Analysis of Homonyms with POS Variation

As mentioned in §5.3.6, in an analysis applied to word forms and not to word senses, homonymy without proper case variation only arises where the same word occurs as more than one POS. The relationships between homonyms with POS variation are defined by morphological rules so that each pair of homonyms can be treated as a pair of suffixations both with null suffixes. It is therefore logical to proceed to the analysis of homonyms with POS variation immediately after suffixation analysis. The lexical relation to be encoded between the homonyms is the lexical relation specified by the applicable rule. This allows homonyms without proper case variation to be processed in the same way as homonyms with proper case variation (§5.3.6), with the following variations:

1. Every possible POS of every word in the atomic dictionary which has more than 2 letters and more than 1 POS is analysed.
2. Every `POSTaggedSuffixation`s is generated by applying morphological rules.
3. If any 2 entries exist in any `Map<POSTaggedMorpheme, POSTaggedSuffixation>` such that the `Relation.Type` encapsulated in the `POSTaggedSuffixation` of the one is the converse of the `Relation.Type` of the other and the POS of the `POSTaggedMorpheme` in each of the two entries is the same as that of the `POSTaggedSuffixation` in the other, which together would imply that each is derived from the other, then the `Relation.Type` of each `POSTaggedSuffixation` is redefined as `Relation.Type.DERIV`, representing a directionless morphological relationship between 2 POSes of the same word,

[145] *X1 Suffix stripping Results.csv* (format in Appendix 19)
[146] *X1 monosyllabic roots.csv*

where the direction of derivation cannot be determined from the morphological rules.

4. The data generated is written to separate files[147]

9782 pairs of homonyms are linked, of which 4720 are monosyllabic. The samples in Appendix 45 show 4 false connections ("frank", "net", "sallow" and "spar") and one complex case involving multiple senses ("hatch"). This represents an estimated precision of 95.4% (92.6% for monosyllables; 98.0% for polysyllables). The monosyllabic results contain errors such as linking "still" as a noun from "still" as a verb. The optimal solution would be to construct a stoplist, which would be a lengthy manual task for which the time has not yet been found. The alternative would be to suppress all the monosyllabic roots, which would eliminate too much correct data.

The rhyming dictionary is revised again, as previously, before proceeding to the rest of the analysis.

5.3.9 Secondary Concatenation Analysis

Now that the 100 most frequent suffixes have been fed into the suffixation analysis process (§5.3.7.3) and the vast majority of suffixations have been removed from the atomic dictionary, it would appear that concatenation analysis can now usefully be repeated with relaxed restrictions, but with the awareness that there will still be apparent concatenations which really are prefixations.

[147] table *Secondary Identical words Results.csv*: one time out of 100, the same data is written to *Secondary Identical words Result Samples.csv*; if the `POSTaggedSuffixation` is monosyllabic, the data is written to *Secondary Monosyllabic Identical words.csv*.

5.3.9.1 Requirements for Secondary Concatenation Analysis

It is obvious, as no prefixation analysis has yet taken place, that the same first component stoplist is still required, and so concatenation analysis was repeated, exactly as before, except with a null last component startlist, so that `candidatesWithBacks` would not be filtered.

Table 44: First 20 initial results from secondary concatenation analysis

Whole word	First component	Middle component	Last component
abhorrent	abhor		rent
abruption	abrupt		ion
accordion	accord		ion
addax	add		ax
addend	add		end
aircrew	air		crew
airfare	air		fare
airscrew	air		crew
albumin	album		in
allotrope	allot		rope
alphabet	alpha		bet
anymore	any		more
argonon	argon		on
argumentation	argument	at	ion
armlet	arm		let
armrest	arm		rest
babyhood	baby		hood
bachelorhood	bachelor		hood
ballad	ball		ad
ballpen	ball		pen

5.3.9.2 Results from Secondary Concatenation Analysis

The results in Table 44 show similar errors to the very first concatenation analysis results, indeed the first two rows of this table can be found in Table 40 (§5.3.4.2). There were still unidentified suffixes partly because of the limited suffix set applied to suffixation analysis and partly because the morphological ruleset was not yet complete at this stage of development so that irregular applications of common suffixes had not been captured. Rather than attempting to execute more refined suffixation analyses while the atomic

dictionary was still full of concatenations, it appeared that it would be more economical on stoplists to process as many concatenations as possible at this stage, which means that it is still necessary to impose restrictions on `candidatesWithBacks`, so a new last component startlist was developed iteratively from observations of errors in the results, with the awareness that yet another concatenation analysis round would be required at a later stage. (Appendix 46).

It became clear during the process of iterative development that almost all analyses with 3 components were wrong (e. g. "anticlockwise" analysed into "antic"; "lock"; "wise" and "codefendant" as "code"; "fend"; "ant". To address this, a new Boolean parameter was added to the Word Analysis Algorithm (§5.2.1.4), to specify, if true, that a limit of 2 was to be set on the number of components for a valid analysis. This parameter is set to false for primary concatenation analysis (to preserve its existing behaviour thereby avoiding the need for repeating the results analysis) and true for secondary concatenation analysis.

Also during the process of iterative development some erroneous first components occurred which had not occurred during primary concatenation analysis, so the filtration procedure (§5.3.4.3) for candidate fronts was revised to use a complementary first component stoplist (Appendix 47). In all other respects the procedure for secondary concatenation analysis is identical to that for primary concatenation analysis.

After finalisation of the new last component startlist and the supplementary first component stoplist, only 225 concatenations are analysed by secondary concatenation analysis (Appendix 48), the startlists and stoplists still being very restrictive, ensuring 100% precision but a recall of only 10%. Further less restricted concatenation analysis is deferred until after prefixation analysis and several iterations of suffixation analysis. The poor recall achieved during this phase suggests that it could safely be omitted with suitable amendments to the stoplists used during the phases up to tertiary concatenation analysis. Such an omission would not however contribute to any improvement in the final results.

5.3.10 Stem Dictionary

Up to this point, it has been a requirement for all morphological analyses that all discovered morphological components apart from affixes must be words in their own right. While this requirement is not always applicable to suffixations, and subsequent phases of suffixation analysis will allow for this (§5.3.14.1), it is more often than not inapplicable to prefixation analysis. Most English prefixes are not English words, and, when they are, the word often has nothing to do with the prefix. Where a stem from prefixation analysis exists as a word, that word is usually *not* the true stem. The reasons for this are historical: many English prefixations are derived from Latin and Greek prefixations, the prefix having become agglutinated to the stem in the pre-classical period and remained stuck there ever since, even when the prefixed word has become subsequently modified. To complicate matters further, scientists coining technical vocabulary for phenomena discovered or invented have, for centuries, adopted the same pre-classical word formation practices, using the same spelling rules as in classical Latin and Greek, including traditional Latin transliteration spelling rules for words of Greek origin. It is only in the mid-twentieth century, with American ascendancy in scientific research that these centuries-old practices started to change.

In pre-classical agglutinations, the semantics which determined the choice of prefix may well be lost in the mists of time such that the meaning of the prefix says little about the meaning of the word, though this is by no means always the case. However the meanings of prefixes are likely to be more relevant in scientific vocabulary than in pre-classical agglutinations. For these reasons, prefixation analysis is to be considered a useful exercise.

It is essential then, from this point, to allow analyses whose components are not words, and the first such components will be prefixes and stems from prefixation analysis. Since most prefixes are not English words, they are not in the lexicon. However, most prefixes are Latin or Greek words whose translations are in the lexicon. Relations can therefore be encoded between prefixations and the prefix meanings directly without any need to store

the prefixes. Stems however may be subject to further analysis, particularly in cases of double prefixation, and so need to be stored. For this purpose a stem dictionary[148] is created at this point, encapsulated, like all the other dictionaries within the `Lexicon`.

5.3.11 Primary Prefixation Analysis

Concatenations, antonymous prefixations and suffixations all having been analysed as far as is possible without non-antonymous prefixation analysis. It is now time according to the precedence rule (§3.5.1), for the analysis of non-antonymous prefixes to commence.

5.3.11.1 Prefix Categories

Successful analysis of prefixations into their prefixes and stems depends on making a distinction between regular prefixes, where the stem may be obtained by removing the prefix *footprint*, subject to *linking vowel exceptions* (§5.3.11.9) and irregular prefixes, which have multiple footprints associated with the same meanings. All prefix footprints can be found by automatic prefix discovery, but while regular prefixes so discovered can be separated from their stems with reference to no other information apart from linking vowel information, this is not true of irregular prefixes. To complicate matters further, many regular prefixes begin with one or more characters which also constitute an irregular prefix, so it is necessary to establish a set of irregular prefix footprints and add to it all the regular prefixes which begin with these footprints and list the instances of each prefix. This suggests that irregular prefixation analysis should precede regular prefixation analysis. The alternative would be to use the methodology applied to antonymous prefixation analysis, but it proved more straightforward to implement a common procedure for regular and irregular non-antonymous prefixations than a common procedure for antonymous and irregular non-antonymous prefixations.

[148] `Set<POSTaggedStem>`

5.3.11.2 Irregular Prefixes

The irregular prefix map houses mappings from prefix footprints which begin with an irregular prefix footprint, and which henceforth will be regarded as irregular prefix footprints, to `IrregularPrefixRecord` lists containing every `IrregularPrefixRecord` which shares that footprint. Each `IrregularPrefixRecord` specifies the footprint, a character sequence to be deleted in order to obtain the stem (usually but not always the same as the footprint), a character sequence to be inserted to obtain the stem (usually empty), the corresponding `TranslatedPrefix`, and a list of instances of words which begin with that prefix. The irregular prefix map is populated from file[149] (as Appendix 49 but with more instances), with the aid of the irregular prefix translations (§5.3.11.3). The initial set of irregular prefix footprints was extracted from the results from the original automatic prefix discovery experiments (§3.4.1; Appendix 16), excluding those footprints which are always antonymous. All instances of words beginning with these footprints were extracted from the lexicon and manually allocated to the corresponding irregular prefix or to a regular prefix whose footprint (beginning with an irregular footprint) was added to the irregular prefix footprint set. Doubtful allocations were confirmed or corrected with reference to OED1, Burchfield (1972) and OED2. Subsequently further additions were made from erroneous results from later cycles of prefixation analysis (§5.3.16.1).

5.3.11.3 Prefix Translations

Since prefixes do not occur in the main dictionary, lexical relations must be encoded between prefixations and the lexically valid meanings of their prefixes. These meanings are stored in the regular and irregular prefix translations maps[150], in which the entries map from the name of a `TranslatedPrefix` to the `TranslatedPrefix` itself. The map is

[149] *Irregular prefixes.csv*; file format in Appendix 20.
[150] each implemented as a `Map<String, TranslatedPrefix>`.

populated from files[151] (Appendix 50). The name of a `TranslatedPrefix` is, by default but not necessarily, the same as the prefix footprint; the name of an irregular prefix is, by default, the same as the regularised form of the irregular prefix footprint prefix (§3.2.2.3). A unique name is given to a `TranslatedPrefix`, whose etymology and meanings are unrelated to those of another prefix with an identical footprint, by appending a digit to the default name(Table 45).

Table 45: Differentiation of prefixes by name

Footprint	Name	Translation	Instances			
coll	con	with	collaborate	collapse	collate	etc.
coll	col	glue	collage	collagen	colloid	etc.
coll	coll	neck	collar	collet	etc.	
coll	coll1	cabbage	collard	etc.		
coll	coll2	coal	collier	colliery		
coll	coll3	colic	collywobbles			

Each `TranslatedPrefix` encapsulates a morpheme array[152], each element of which represents a lexically valid meaning of the prefix as its specified POS. The translations were provided from a knowledge of the Greek, Latin and Anglo-Norman origins of most of the prefixes, supplemented and corroborated, where necessary, by OED1 and OED2. In selecting the most appropriate translations, the actual uses of the prefix were taken into consideration and the principle of utility was allowed to override that of etymological fidelity, with the most useful rather than the most accurate translation being placed first.

The irregular prefix translations are the translations of the prefixes in the irregular prefix map (§5.3.11.5); the regular prefix translations are the translations of the valid prefixes in successive secondary prefix sets (§5.3.11.6).

It is almost always true that when a word begins with an English preposition, the rest of the word is also lexically valid and so it was decided at this stage, that when the first

[151] *Detailed Prefix meanings.csv* & *Detailed Irregular prefix meanings.csv*; file format in Appendix 20. The POS of each translation is given as either a word or a special code comprising the initial letters of 2 POSes separated by '/'; the initial 'A' represents ADVERB before '/' or ADJECTIVE after '/'.
[152] `POSTaggedMorpheme[]`

component of a word is an English preposition (e. g. "after"; §5.3.4.3) that the word should not be treated as a prefixation but as a concatenation. Prefixation analysis can then proceed on the basis that a translation is always required. Such concatenations are processed during tertiary concatenation analysis (§5.3.15).

5.3.11.4 Adaptation of the Word Analysis Algorithm for Prefixation Analysis

Prefixation analysis is performed using the same Word Analysis Algorithm as is used for concatenation analysis (§5.2.1), but with null `candidateBacks` and with the `StringBuilder` upon which deletions are performed replaced by a `WordBreaker`.

5.3.11.4.1 Prefix Stripping using a Word Breaker (*Class Diagrams 12 & 13*)

The original idea for the `WordBreaker` class was to extend Class `StringBuilder`, but this is not possible since `StringBuilder` is declared `final` in Java. Instead, `WordBreaker` implements interface `CharSequence`, which `StringBuilder` also implements, and encapsulates a `StringBuilder` in which the word undergoing modifications is stored. All the operations specified by `CharSequence` are implemented by passing them on to the encapsulated `StringBuilder`. The delete operation is not specified by the interface but is the single operation which differs from that of a `StringBuilder`, returning a `Morpheme`. This solution results in additional complexity in the Word Analysis Algorithm (§5.2.1.4). A subclass `IrregularWordBreaker` is applied for the analysis of irregular prefixations. The following description applies to a regular `WordBreaker` as applied to regular prefix stripping.

The deletion performed by a `WordBreaker` can handle the removal from its *embedded word* (the word represented by its encapsulated `StringBuilder`) of either a prefix (when the value of parameter `start` = 0) or a suffix (when the value of `end` equals the length of

the embedded word)[153]. As we are currently concerned with prefix stripping, only the prefix stripping functionality will be described here. The prefix footprint equivalent to the substring of the embedded word specified by `start` and `end` is looked up in the regular prefix translations map (§5.3.11.3), to find the single corresponding `TranslatedPrefix`. If there is no entry in the regular prefix translations map for the specified footprint, then an error message is output and a `LemmaMismatchException` is thrown. This is non-fatal, merely indicating that the embedded word does not start with a known regular prefix. The stem formed by simple deletion of the prefix footprint from the word embedded in the `WordBreaker` is represented as a `POSTaggedWord` with a *negative lexical validity requirement* (meaning that it need not be lexically valid). A `Prefixation`[154] is created encapsulating the `TranslatedPrefix` and the stem with only that POS specified. The `TranslatedPrefix` is returned, while the embedded word is replaced with the stem.

5.3.11.4.2 Irregular Word Breaker

The deletion performed by an `IrregularWordBreaker` is more complex, though it handles only prefixations[155]. The irregular prefix footprint equivalent to the substring of the embedded word specified by `start` and `end` is looked up in the irregular prefix map, to find the corresponding list of irregular prefix records (§5.3.11.5). The `IrregularPrefixRecord` in the list which holds the word embedded in the `IrregularWordBreaker` as one of its instances is selected. If no such `IrregularPrefixRecord` is found then a non-fatal `LemmaMismatchException` is thrown. The `TranslatedPrefix` encapsulated in the `IrregularPrefixRecord` is extracted. The stem is formed by deleting from the embedded word the character sequence to be deleted as specified by the `IrregularPrefixRecord` and replacing it with the character sequence to be inserted (if any). A `Prefixation` is created as in the case of

[153] If both these conditions are true or neither is, then a `StringIndexOutOfBoundsException` is thrown (for consistency with `StringBuilder`); if `start` is equal to `end`, then `null` is returned.
[154] Class used for passing information between the `Prefixer` and a `WordBreaker`.
[155] A `StringIndexOutOfBoundsException` is thrown in the same circumstances as for a regular `WordBreaker` or if an attempt is made to apply it to suffix stripping.

a regular `WordBreaker`, and the `TranslatedPrefix` is returned, while the embedded word is likewise replaced with the stem.

5.3.11.4.3 Usage of Word Breakers by the Word Analysis Algorithm

When the Word Analysis Algorithm is passed a `WordBreaker` instead of a `StringBuilder`, the outer loop iterating through candidate fronts (§5.2.1.4) is only allowed to execute until a single morpheme array has been generated, representing the analysis of the prefixation into prefix and stem. The delete method of the `WordBreaker` is invoked with `start` equal to 0 and `end` equal to the length of the candidate front, which either returns a `TranslatedPrefix` or throws a `LemmaMismatchException`. In the latter case execution continues with the next candidate front (if any). If there are no more candidate fronts, the algorithm terminates. The `TranslatedPrefix` replaces the candidate front and the stem becomes the core. A 2-element morpheme array is generated comprising the `TranslatedPrefix` and the stem.

5.3.11.5 Irregular Prefixation Analysis

Irregular prefixations are handled before regular prefixations, on the basis that the set of irregular prefix footprints is known and finite as the keyset of the irregular prefix map, while the set of regular prefix footprints is indeterminate, being limited only by the duplication criterion of automatic prefix discovery (§3.4). Although automatic prefix discovery can discover irregular prefix footprints, it is not applied to the atomic dictionary until irregular prefixations have been removed, thereby preventing irregular prefixations from being handled as if they were regular.

Every word in the atomic dictionary is treated as a potential prefixation. The footprints which are the keys to the irregular prefix map[156] (Appendix 49) are used as an initial prefix set. Candidate front lists are generated for each word (§5.2.1) using this set as `vocabulary` without frequency corroboration (§5.3.4.3); so `candidatesWithFronts`

[156] `Map<String, List<IrregularPrefixRecord>>`

(§5.3.4.1) will comprise mappings from the words in the atomic dictionary to lists of any irregular prefix footprints with which they begin. Candidate front lists are reordered so that the longest irregular prefixes are always tried first. Candidate back lists are generated using a null vocabulary, such that each list contains only an empty character string. Each word in the atomic dictionary in turn is embedded in an `IrregularWordBreaker`, which is passed to the Word Analysis Algorithm. If a `LemmaMismatchException` is thrown, the word is placed in a rejected components map, mapping to an empty array, otherwise a mapping from the word to the morpheme array returned by the Word Analysis Algorithm is added to a primary prefixations map. The contents of the rejected components map and the primary prefixations map are both written to file[157].

The words which are keys in the primary prefixations map are removed from the atomic dictionary and their reversed forms from the rhyming dictionary. They are looked up in the main dictionary to identify their possible POSes. Each word as each of its possible POSes is represented as a `POSTaggedMorpheme`. Each stem (the second element in the morpheme array to which the word maps in the primary prefixations map), as each of the word's possible POSes is also represented as a `POSTaggedMorpheme`. A secondary prefixations map is generated comprising mappings from each `POSTaggedMorpheme` representing a word to a 2-item list of morphemes of which the first is the `TranslatedPrefix` (the first element in the morpheme array to which the word maps in the primary prefixations map) and the second is the `POSTaggedMorpheme` representing the stem.

5.3.11.6 Regular Prefixation Analysis

After removal of the irregular prefixations from the atomic dictionary, a `PrefixTree` is constructed from the atomic dictionary (§5.3.3.1) and a primary prefix set[158] is generated

[157] *Irregular rejected prefixation components.csv* & *Irregular prefixations with components.csv* (format in Appendix 19).
[158] *Prefixes.csv* (format in Appendix 19); implemented as `Set<Affix>`.

from it in the same way as the primary suffix set is generated from the atomic-dictionary-based `SuffixTree` (§5.3.7.2), using the same optimal heuristic

$$\frac{f_c^2 q_s}{f_p}.$$

Although this heuristic was not proven optimal for prefix stripping (§3.4.4), it was among the best contenders and performs well on the `PrefixTree` constructed from the atomic dictionary, from which most concatenations have already been removed. It has therefore been chosen as the optimal heuristic for prefixation analysis also, though the default heuristic

$$\frac{f_c^2}{f_p}$$ (§3.4.1.2)

is also used in iterative prefixation analysis (§5.3.16.1). The purpose of the primary prefix set is to prioritise those candidate prefixes which are most likely to satisfy the semantic criterion. A secondary prefix set (Appendix 51) is created in the same way and for the same reasons as the secondary suffix set (§5.3.7.3), again arranged in descending order of affix length with a secondary lexicographic ordering. There being far more semantically valid prefixes than suffixes, its size is set to 500. The secondary prefix set is used as vocabulary for generating candidate front lists without frequency corroboration (§5.3.4.3).

Prior to first applying the same procedure using the Word Analysis Algorithm as for irregular prefixes, it was necessary to populate the regular prefix translations map with the prefixes in the secondary prefix set and their translations (§5.3.11.3). This process needed to be repeated for each subsequent prefixation analysis using a fresh `PrefixTree` (§5.3.16.1).

Every remaining word in the atomic dictionary is again treated as a potential prefixation in the same way as for irregular prefixation, except that a regular `WordBreaker` is passed to the Word Analysis Algorithm[159] and the mappings from each `POSTaggedMorpheme`

[159] results written to *X1Rejected prefixation components.csv* & *X1Prefixations with components.csv* (Appendix 19).

representing a word to a 2-item list are written to the same secondary prefixations map which already contains the irregular prefixation analyses.

5.3.11.7 Encoding of Lexical Relations between Prefixations and their Components

Each entry in the secondary prefixations map now comprises a derivative prefixation mapping to a 2-item list containing a prefix as a `TranslatedPrefix` and a stem as a `POSTaggedMorpheme`.

The stem is represented as a `POSTaggedStem`, which is looked up in the stem dictionary. If a corresponding entry is found (a `POSTaggedStem` with the same word form and POS), then the `POSTaggedStem` which was looked up is overwritten by the corresponding entry, which is necessarily the same except that it will already have a list of affixes associated with it and lexical relations encoded from its `POSSpecificLexicalRecord` to corresponding affixations.

The set of *false lexical stems*, each represented as a `POSTaggedMorpheme`, has already been populated from file[160]. It comprises morphemes which occur as the stems of prefixations and whose word forms and POSes are identical to, but whose meanings differ from, words in the lexicon (Appendix 38). If the stem is found in the main dictionary as its specified POS, and is not included in the false lexical stem set, relations are encoded between the prefixation and the stem in the main dictionary (Appendix 18). If the stem is not found in the main dictionary as its specified POS, or is included in the false lexical stem set, then relations are encoded between the prefixation and the `POSSpecificLexicalRecord` encapsulated in the `POSTaggedStem`, the `TranslatedPrefix` is added to the list of affixes associated with the `POSTaggedStem` and the `POSTaggedStem` is added to the stem dictionary, overwriting any existing `POSTaggedStem`, so that the `POSTaggedStem` in the stem dictionary will include the

[160] *Prefixation stem stoplist.csv* (format in Appendix 20)

prefix in its affix list. Irrespective of the lexical status of the stem, translating relations are encoded between the prefixation and each meaning of the `TranslatedPrefix` (Appendix 18)[161].

5.3.11.8 Initial Results from Regular Prefixation Analysis

The first results from regular prefixation analysis comprised 6224 analyses all of which were reviewed, leading to the manual creation of a stoplist from the 2070 incorrect analyses, an initial precision of 67%. The analysis procedure was modified to read this stoplist into a `Map<String, Set<String>>` comprising mappings from prefixes to the stems paired with those prefixes in the incorrect analyses and to reject the incorrect analyses by consulting the stoplist.

5.3.11.9 Linking Vowels

The only spelling irregularities that need to be taken into consideration with regular prefixes are variations with regard to the presence or absence of a linking vowel (most usually 'o'), generally, but not invariably, determined by whether the stem begins with a vowel or a consonant. This issue was raised during development of automatic prefix discovery (§3.2.2.3), but any decision as to how to handle it was deferred. In a `PrefixTree`, a prefix with a linking vowel occurs as the child of the prefix without a linking vowel, but in the primary prefix set obtained from the `PrefixTree`, the order in which such a pair occurs is determined by the optimal heuristic and is not predictable from orthography. Consequently, the finite secondary prefix set may include a prefix with a linking vowel or the same prefix without the linking vowel or both. No objective criterion being known to establish whether the linking vowel is part of the prefix or not,

[161] The following fatal exceptions can be thrown by this procedure:

- a `DuplicateRelationException` if either any meaning of any prefix (as its specific POS) or any prefixation (ignoring its POS) is not in the main dictionary;
- a `DataFormatException` if the number of components in the analysis is not equal to 2;
- an `UnexpectedPOSException` if the first listed component morpheme is not a `TranslatedPrefix` or if the second listed component morpheme is not a `POSTaggedMorpheme`.

the prefix translations map includes any form which occurs in the secondary prefix set, or any subsequent secondary prefix set during iterative prefixation analysis (§5.3.16.1). This guarantees that the prefixation will be linked to the correct prefix meanings, but the stem needs correction where either a stem with a missing initial vowel is associated with a prefix with a linking vowel (a linking vowel exception) or an erroneous vowel occurs agglutinated to a stem and the prefix has no linking vowel (a reverse vowel linking exception).

Although the secondary prefix set includes both "hydr-", as in "hydrate" and "hydro-", as in "hydroxide", "hydro-" occurs first because the secondary prefix set is ordered in descending order of word length. Consequently "hydroxide" will be analysed as "hydro-" + "-xide". This is a linking vowel exception where the stem needs to be corrected to "-oxide". The prefix does not need to be corrected as "hydr-" and "hydro-" both occur in the regular prefix translations map, mapping to the same meanings. The prefix "man-" occurs in the secondary prefix but "manu-" does not. Consequently "manufacture" is analysed as "man-" + "-ufacture". This is a reverse linking vowel exception where the stem needs to be corrected to "-facture". The prefix does not need to be corrected as "man-" occurs in the prefix translations map.

The initial results were screened for linking vowel errors and all instances were collected into files[162] (Appendix 52). The analysis procedure was revised to read these files into maps of the same format as the stoplist and to consult both maps to apply the necessary correction, namely, in the case of a linking vowel exception, to copy the last letter of the prefix to the beginning of the stem, and in the case of a reverse linking vowel exception, to remove the first letter of the stem. Only the stem is corrected; the prefix is never modified as it is always identifiable in the translations map.

The final results, after corrections to the irregular prefix map, the irregular prefix translations map and the regular prefix translations map, comprise 5197 analysed

[162] *Linking vowel exceptions.csv* and *Reverse linking vowel exceptions.csv*; file format in Appendix 20.

prefixations[163]. These results are necessarily incomplete because only 500 prefixes are allowed, and subsequent cycles of prefixation analysis are therefore required (§5.3.16), but with reference to the results from secondary prefixation analysis, recall is 96%, with precision improved to 100% by stoplist deployment. These figures may be contested on lexicographic criteria, particularly with regard to the categorisation of words which start with English prepositions as concatenations (§5.3.11.3).

5.3.12 Secondary Antonymous Prefixation Analysis

Because primary antonymous prefixation analysis is subject to the requirement that the antonyms discovered by removing antonymous prefixes must be lexically valid words, a second cycle of antonymous prefixation analysis is required in order to capture instances of antonymous prefixation where the stem is not a word. This analysis has the highest precedence and can now be conducted excluding prefixes beginning with "a" and prefixes "dis-", "de-", "counter-", "contra-", "contr-", which are semi-antonymous prefixes already handled by non-antonymous prefixation analysis and assigned semi-antonymous meanings, leaving a reduced set of antonymous prefixes: {"un", "in", "imb", "ign", "ill", "imm", "imp", "irr", "non"}. The same procedure as for primary antonymous prefixation analysis is applied to the remaining words in the atomic dictionary using this smaller set, but with the same exception lists, though with a negative lexical validity requirement.

The resultant antonymous prefixations map[164] is reorganised in the same format[165] as the primary prefixations map in non-antonymous prefixation analysis (§5.3.11), though each morpheme array only contains a single element housing the stem. The contents of this map are written to file[166]. The prefixations are removed from the atomic dictionary and a secondary prefixations map is generated in the same way as for non-antonymous prefixation analysis, where each entry maps from a `POSTaggedMorpheme` representing a

[163] *X1Prefixations with components.csv* (Appendix 19)
[164] `Map<POSTaggedWord, POSTaggedWord>`
[165] `Map<String, Morpheme[]>`
[166] *Residual antonymous prefixes.csv* (format in Appendix 19)

word as a particular POS to a 1-item list of morphemes whose sole element is the `POSTaggedMorpheme` representing the stem.

Relations between the prefixations and their antonymous stems are encoded in the same way as during non-antonymous prefixation analysis (Appendix 18), except that the prefix itself is discarded and the relations encoded are of type `ANTONYM`, and "NOT_" is added to the affixes of the `POSTaggedStem`. 260 antonymous prefixations are analysed.

5.3.13 Pruning the Atomic Dictionary

As relations have been encoded between homonyms with proper case difference, and no further analysis of proper case words is intended, all uppercase entries and entries starting with numerals or punctuation marks are now removed from the atomic dictionary.

The atomic dictionary is also checked for homonym pairs with POS variation, where only one of the POSes is in the atomic dictionary entry for the word and whose members are linked, in the main dictionary by a `POSSpecificLexicalRelation` of `Relation.Type.DERIV`, implying that each is derived from the other. This could occur as a consequence of homonym analysis (§5.3.8). If any such instance is found, the POS which is in the atomic dictionary entry is removed, and, if that leaves the entry with no POSes, then the entire entry is removed.

After the atomic dictionary has been pruned, the rhyming dictionary is again revised as previously.

5.3.14 Secondary Suffixation Analysis

Antonymous prefixation analysis now being complete and the remaining concatenations still being subject to confusion with suffixations, suffixation analysis now has the highest precedence. Since primary suffixation analysis operates with a positive lexical validity

requirement, there is clearly still scope for identifying more suffixations where the stem is not a word.

5.3.14.1 Differences from Primary Suffixation Analysis

Secondary suffixation analysis initially operates in the same way as primary suffixation analysis (§5.3.7), except with a negative lexical validity requirement and with a supplementary stoplist[167] (§5.3.14.2). The negative lexical validity requirement triggers modified behaviour of the Root Identification Algorithm (§5.2.2.5) as follows.

- Any monosyllabic `POSTaggedSuffixation` generated by inflectional morphology or by conditional morphological rules is systematically rejected irrespective of the applicability of the rule to monosyllables.

- Any `POSTaggedSuffixation` which fails the validity check (against the stoplists) is not deleted, but is marked as *unsuitable*, meaning that it is unsuitable for encoding of a lexical relation in the main dictionary.

- The frequency-based modification (§5.2.2.6) is not applied.

- If there is more than one morphological rule in the current list, then the unique default non-lexical morphological rule applicable to the suffix (§5.1.5) is added to the current list of rules. This rule represents the most probable analysis of the derivative word into stem and suffix.

- The rules in the current list of rules are applied in turn with an overriding positive lexical validity requirement, except for the final rule, which is applied, if it is a non-lexical rule, with a negative lexical validity requirement, so that when no analysis discovers a lexically valid stem, the most probable analysis involving a non-lexical stem is returned.

[167] *Secondary suffix stripping stoplist.csv* (format in Appendix 20)

Once the middle loop (§5.3.7.3; Appendix 21), iterating through the derivative word's POSes, has terminated, during execution of the loop which iterates through the map created, any monosyllabic `POSTaggedSuffixation` generated by a rule inapplicable to monosyllables is not automatically rejected, but if it is lexically valid, it also is marked as *unsuitable*. Any `POSTaggedSuffixation` which is not lexically valid or which is marked as unsuitable is not written to the results and no relations are encoded in the main dictionary using it.

If any `POSTaggedSuffixation` is not lexically valid or is valid but is marked as unsuitable, then it is treated as a stem but not a word. The POS of the derivative word is removed from the derivative word's entry in the atomic dictionary. A `POSTaggedStem` is created from the `POSTaggedSuffixation`. If the `POSTaggedStem` is already in the stem dictionary, it is overwritten by the entry in the stem dictionary, for the reasons given in §5.3.11.7, otherwise it is added to the stem dictionary. The original suffix component of the `POSTaggedSuffixation` is added to the stem's suffix list encapsulated in the `POSTaggedStem`. A relation is then encoded between the derivative word and the `POSSpecificLexicalRecord` encapsulated in the `POSTaggedStem` in the stem dictionary (Appendix 18).[168]

5.3.14.2 Initial Results from Secondary Suffixation Analysis

The results from secondary suffixation analysis are written to files[169], in the same way as the results from primary suffixation analysis are written to files prefixed with "X1" (§5.3.7.3).

Overgeneration of lexically valid words in the initial results from secondary suffixation analysis was addressed by supplementing the stoplist retained from primary suffixation analysis and applied to secondary suffixation analysis with a secondary stoplist

[168] When the inner loop terminates without any `POSTaggedSuffixation` being generated, then nothing is added to the map, but a record is written to file *X2 unidentified roots.csv* (format in Appendix 20).
[169] *X2 Suffix stripping Results.csv, X2 Suffix stripping Result Samples.csv* & *X2 monosyllabic roots.csv* (Appendix 19)

comprising the false derivative-root pairs[170] (Appendix 53). The application of the stoplists does not preclude the identification of the same roots as stems (§5.3.14.2). The secondary stoplist remains in force through the subsequent cycles of iterative suffixation analysis (§5.3.14.3), and records were added to the secondary stoplist, iteratively, through observation of overgenerations in the results from those cycles.

Undergeneration was addressed by allowing a `POSTaggedSuffixation` marked as unsuitable to be *reprieved* if it is found, with its original suffix, in a *reprieves map*[171] (Appendix 54), a concept similar to that of counter-exceptions as in antonymous prefixation analysis (§5.3.5.2). Each key in the reprieves map encapsulates the word form and POS of the `POSTaggedSuffixation` to be reprieved and each value is the set of original suffixes one of which the `POSTaggedSuffixation` must possess in order to be reprieved. The words to be reprieved are often monosyllabic and marked as unsuitable because a rule is encoded as inapplicable to monosyllables. The entries in the reprieves map are read from a file[172], manually created by examination of each `POSTaggedSuffixation` marked as unsuitable. Any reprieved `POSTaggedSuffixation` is treated as lexically valid and suitable, is written to the results and is used for encoding a lexical relation within the main dictionary. The reprieves map remains in force through the subsequent cycles of iterative suffixation analysis, and its contents were augmented iteratively through observation of undergenerations in the results from those cycles.

After addressing overgeneration and undergeneration, the encoding of relations between derivative words and stems in the stem dictionary was manually monitored for unrelated roots and derivatives. The unique error found was the encoding of "event" as the root of "eventide"[173]. The uniqueness of this exception confirms the reliability of the methodology. The revised procedure for secondary suffixation analysis achieves 54% recall, subject to lexicographic interpretation.

---

[170] contained in file *Secondary suffix stripping stoplist.csv*.
[171] `Map<POSTaggedWord, Set<String>>`
[172] *Final suffixation reprieves.csv*; format in Appendix 20.
[173] subsequently been hard-coded as an exception.

5.3.14.3 Iterative Suffixation Analysis

Secondary suffixation analysis is followed immediately by a series of iterations of SuffixTree construction and suffixation analysis. Each iteration comprises the following operations.

- The rhyming dictionary is revised as previously (§ 5.3.6.3).

- A new `SuffixTree` is constructed from the rhyming dictionary as previously (§5.3.7.1).

- A primary suffix set is obtained from the new `SuffixTree`, ordered by a `Comparator<Affix>` which imposes a primary ordering by the optimal heuristic $\frac{f_c^2 q_s}{f_p}$.

- Suffixation analysis is performed in the same way as in secondary suffixation analysis as described in §5.3.14.1, except with a larger secondary suffix set (§5.3.7.3; Appendix 55), comprising the first *200* suffixes returned by the primary suffix set's `Iterator`, to include unusual suffixes.

- Because manual inspection of the primary suffix set generated using the optimal heuristic showed that the remaining semantically valid suffixes were scattered throughout the set (see also §5.3.16.2), an alternative primary suffix set is obtained from the same new `SuffixTree`, with a primary ordering[174] by the default heuristic

  $\frac{f_c^2}{f_p}$ (§3.4.1.2)

[174] imposed by method `public int Affix.compareTo(Object o)`

- Suffixation analysis is repeated in the same way[175] with a secondary suffix set (Appendix 55) comprising the first 200 suffixes returned by the alternative primary suffix set's `Iterator`.

Any productive suffixation analysis operation reduces the size of the atomic dictionary. Iterative suffixation analysis therefore continues until the size of the atomic dictionary, measured at the beginning of each iteration, has not decreased during the course of the iteration. This occurs after the second iteration with the WordNet-based lexicon.

The Morphological ruleset, the secondary stoplist and the reprieves file continued to be updated iteratively with semantically valid suffixes obtained from new secondary suffix sets throughout the course of the implementation of secondary and iterative suffixation analysis.

Iterative analysis discovers 176 further suffixations. The full results are in Appendix 55. Meaningful quantification of precision and recall is not realistic as there is too much room for interpretation where unusual suffixes are concerned.

After secondary suffixation analysis, the atomic dictionary is again pruned and the rhyming dictionary is again revised as previously.

5.3.15 Tertiary Concatenation Analysis

Tertiary concatenation analysis proceeds initially as secondary concatenation analysis (§5.3.9), except without any stoplists or startlists and without frequency corroboration (§5.3.4.3) in the creation of candidate lists. These changes effectively lift the restrictions imposed on concatenation analysis (though the number of components is still limited to 2), which should now be unnecessary insofar as suffixation analysis is now complete, though there is still a likelihood of prefixes being mistaken for words participating in

[175] The file prefix for output files from each suffixation analysis operation changes at each such operation from *X2* through *X3*, *X4* etc.

concatenations as their first component. To deal with these and any other anomalies, the secondary concatenations map is filtered using a fresh stoplist (Appendix 57), which comprises whole words which are not to be treated as concatenations. Any entry in the secondary concatenations map whose key (the word analysed) is in this stoplist is removed from the secondary concatenations map prior to encoding of relations between the concatenations and their components as during secondary concatenation analysis. Words beginning with an English preposition (§§5.3.4.3, 5.3.11.3) are analysed at this stage. 1956 concatenations are analysed[176]. In a sample set sampled at a rate of 1 in 20, 35 errors were found, suggesting an estimated precision of 64.3%, with 100% recall if possible 3-grams are ignored. This poor result arises because the initial output was not fully reviewed for the compilation of the stoplist.

5.3.16 Secondary Prefixation Analysis

Having been applied with as few restrictions as possible, at this stage concatenation analysis and suffixation analysis can be considered complete. Therefore, for a complete analysis of all the words in the lexicon, there remains only the task of secondary prefixation analysis.

5.3.16.1 Iterative Prefixation Analysis

Secondary prefixation analysis is iterative from the start, in a way comparable to iterative suffixation analysis (§5.3.14.3). The procedure comprises a series of iterations of PrefixTree construction and prefixation analysis as previously described (§5.3.11.6) [177]. Each iteration comprises the following operations.

- A new `PrefixTree` is constructed.

[176] *X3Concatenations with components.csv* (format in Appendix 19)
[177] The file prefix for output files from each prefixation analysis operation changes at each such operation starting at *X2* through *X3*, *X4* etc.

- A primary prefix set is obtained from the new `PrefixTree`, ordered using the optimal heuristic

  $\frac{f_c^2 q_s}{f_p}$.

- Prefixation analysis is performed with a secondary prefix set (Appendix 56) of 500 prefixes.

- Relations are encoded between the prefixations and their stems and prefix meanings using the data in the prefixations map returned by the analysis.

Iterative prefixation analysis continues until the size of the atomic dictionary, measured at the beginning of each iteration has not decreased during the course of the iteration. The whole iterative procedure is then repeated in the same way as before except that the primary prefix set is obtained from the each new `PrefixTree`, ordered using the default heuristic

$\frac{f_c^2}{f_p}$ (§3.4.1.2).

A total of 7 iterations of `PrefixTree` construction and prefixation analysis are executed, 3 with the optimal heuristic and 4 with the default heuristic.

The regular prefix translations map (§5.3.11.3) and the lists of linking vowel exceptions and reverse linking vowel exceptions (§5.3.11.9) continued to be updated iteratively with throughout the course of the implementation of iterative prefixation analysis.

The full results from iterative prefixation analysis are in Appendix 56. Precision and recall are subject to interpretation: the word segmentation achieved is questionable[178], but the prefix meanings mapped to are all correct, apart from the spurious instances of prefix "mer-", translated as "part", in the results from the 6th. secondary prefix set[179].

[178] Segmentation is not the objective (§3.3.4).
[179] accidentally overlooked but easily corrected by additions to the stoplist.

5.3.16.2 Differences between Iterative Analysis of Prefixations and Suffixations

The procedure described in §5.3.16.1 differs somewhat from the procedure for iterative suffixation analysis (§5.3.14.3). These differences arise from the fact that there are far more semantically valid prefixes than semantically valid suffixes. The reasons for the variation have to do with the contents of the primary and secondary suffix and prefix sets. These were inspected after the first execution of the first analysis operation in each iterative analysis. Inspection of the primary and secondary prefix set showed that the next prefixes following the cutoff after the 500th. prefix had a high proportion of valid prefixes, whereas, in the case of suffixation analysis, this was not the case, but there were semantically valid suffixes scattered throughout the primary set. Consequently, priority was given, in iterative suffixation analysis, to changing the heuristic, while for prefixation analysis, a change of heuristic was not called for as long as a fresh `PrefixTree` would provide a fresh supply of valid prefixes.

After secondary prefixation analysis, the atomic dictionary is again pruned as previously.

5.3.17 Stem Processing

Samples (1/50 entries) were taken of the atomic dictionary after completion of the implementation of each analysis procedure described in this section These samples were used to confirm the most immediate requirements for further analysis, suggested by precedence considerations (§3.5). A sample taken of the atomic dictionary after secondary prefixation analysis (Appendix 58) reveals that it is dominated by genuinely atomic words which cannot be further broken down, spelling variants, abbreviations and words whose morphology arises from inflectional and derivational phenomena belonging to other languages (Table 46). A few concatenations remain such as "anywhere", whose components are not in the lexicon ("where" is not in WordNet) and affixations with unique affixes rejected by automatic affix discovery or affixes insufficiently frequent to

arise even during iterative affixation analysis. With these few exceptions, the analysis of words as concatenations and affixations at this stage is complete. The only remaining task in a complete morphological analysis is the analysis of the stems themselves, which may well include secondary affixes or even valid words.

Table 46: Analysis of atomic dictionary samples

Reason for inclusion	**Instances**	**%**
Atomic	26	22.22%
Foreign	21	17.95%
Spelling variant	11	9.40%
Abbreviation	10	8.55%
Unidentified affix	9	7.69%
Obscure	8	6.84%
Irregular multilingual derivation	7	5.98%
Irregular Anglo-Norman spelling transformation	5	4.27%
Onomatapoeic	5	4.27%
Irregular quasi-gerund	4	3.42%
Back formation	2	1.71%
Concatenation component not in WordNet	2	1.71%
Invention	2	1.71%
Erroneous stoplist entry	1	0.85%
Missing from Irregular prefix instances	1	0.85%
Old Norse Gerund	1	0.85%
U.S. college student slang	1	0.85%
Unhandled inflectional suffix	1	0.85%
TOTAL	**117**	**100.00%**

Stem processing is the process of converting the stem dictionary from a repository for unidentified morphemes into a useful adjunct to the lexicon. The three main phases of stem processing are pruning, interpretation and analysis. Pruning involves the investigation of redundancy in the stem dictionary, the removal of which involves some correction of the lexical relations in the main dictionary. Stem interpretation involves the assignation of meanings to as many stems as possible and the encoding of relations between those stems and their meanings. Stem analysis is similar to the morphological analysis of words, without the expectation of finding many components in the lexicon. It involves the simultaneous identification of prefixes and suffixes at the beginnings and ends of stems originally derived from words with multiple affixes.

5.3.17.1 Creation of the Atomic Stem Dictionary

Just as morphological analysis of the contents of the lexicon requires (§5.3.3.1) an atomic dictionary, so the morphological analysis of the contents of the stem dictionary requires an atomic stem dictionary. This is now created, in the same format as the main atomic dictionary and is populated with mappings from the word forms of the stems in the stem dictionary to their recorded POSes.

5.3.17.2 Pruning the Stem Dictionary

Up to this point the contents of the stem dictionary had not been subject to any kind of checking. Examination of the stem dictionary revealed unnecessary entries such as "sexual" as a noun, which is not lexically valid and appeared in the stem dictionary because the direction of derivation of lexically valid words such as "bisexual" as a noun from "bisexual" as an adjective could not be determined automatically during homonym analysis. So "bisexual" as a noun remained in the atomic dictionary to be treated, during prefixation analysis, as derived from prefix "bi-" and "sexual" as a noun. In fact, "bisexual" as a noun is derived from "bisexual" as an adjective, which in turn is correctly derived through prefixation analysis from prefix "bi-" and "sexual" as an adjective. Thus the stem "sexual" as a noun is redundant, even though as a non-lexical stem it has a negative lexical validity requirement. To correct such anomalies, the derivations of such prefixations are revised and the lexical relations representing the false derivation are deleted and re-encoded by the following algorithm (a more code-like description is available in Appendix 59).

An outer loop iterates through the stems in the stem dictionary. An alternative POS is sought in the main dictionary for each non-lexical stem. If there are multiple alternatives, the one with most relations of `Relation.Type.DERIVATIVE` is selected. If an alternative POS exists, then a set is created comprising every `POSSpecificLexicalRelation` of `Relation.Type.DERIVATIVE` from the original stem in the stem dictionary. The targets of these relations are one or more prefixations with potentially false derivations. An inner

loop iterates through this set. Each of these prefixations is examined to see if its POS is the same as that of the original stem in the stem dictionary. If so then it is treated as falsely derived. Every `POSSourcedLexicalRelation` of `Relation.Type.ROOT` and every `POSSpecificLexicalRelation` of `Relation.Type.DERIV` from that prefixation is then deleted. The prefix component of the prefixation is deleted from the original stem's prefix list.

When the inner loop has terminated, if the stem has no relations left of `Relation.Type.DERIVATIVE`, then any relations of `Relation.Type.ROOT` from the stem are also deleted[180]. If the stem still has any other relations of `LexicalRelation.SuperType.DERIVATIVE`, then relations are encoded between the stem and its alternative POS[181] and written to file[182]. The stem's POS is then removed from its entry in the atomic stem dictionary. If the stem now has no relations at all, it is removed from the stem dictionary.

A unique exception, the stem "ax", is exempted from stem dictionary pruning, as this would create a false derivational relation between "coax" as a noun and "coax" as a verb, while the derivation of "coax" as a noun from non-lexical stem "ax" is correct.

Stem dictionary pruning leaves the stem dictionary with 16456 entries, which are written to file[183].

5.3.17.3 Stem Interpretation

Despite stem dictionary pruning, the analyses which feed into the stem dictionary are not necessarily valid with respect to those stems. In particular, since iterative suffixation is relatively unrestricted, the stems discovered and the relations encoded between them and

[180] All deletions of relations imply the deletion of the converse relation also.
[181] The primary relation is encoded in the `POSSpecificLexicalRecord` encapsulated in the stem and the converse relation is encoded in the `POSSpecificLexicalRecord` in the main dictionary corresponding to the alternative POS (format in Appendix 18).
[182] *Stem relations from stem dictionary pruning.csv* (format in Appendix 19)
[183] *Affixation stems1.csv;* format in Appendix 19.

the words from which they were treated as derived are not necessarily valid and as such are unsuitable for use by any application. Unlike the main dictionary, the stem dictionary contains no references to the wordnet component of the model, and its lexically invalid entries do not occur in the wordnet. Only where a common meaning can be assigned to a stem where it occurs with every one of its associated affixes can the information in the stem dictionary be considered reliable or useful.

Of 16070 stems (from an earlier version of the stem dictionary), 14196 occurred only with a single affix. These are necessarily both the least reliable and the least useful. A further 1197 occurred only with one of two affixes, leaving a manageable 677 with three or more affixes to be manually validated and interpreted, so that relations could be encoded between the stems and their meanings, turning the stem dictionary into a useful and reliable resource for applications.

Table 47: Identical stems with unrelated meanings

Original words	Stem	Stem POS	Translation	Translation POS	Associated Prefixes		
acrobat	bat	NOUN	goer	NOUN	acro	#	
combat	bat	NOUN	hitting	NOUN	con	#	
megabat, microbat	bat	NOUN	bat	NOUN	mega	micro	#

5.3.17.3.1 Stem Translations File[184] (Appendix 60)

Stem translations were arrived at in the same way, and with reference to the same resources, as prefix translations (§5.3.11.3). Again the principle of utility was allowed to override that of etymological fidelity. Where instances of the same stem as the same POS had unrelated meanings, they were treated as separate stems and separate entries were made in the stem translations file (Table 47). Some stems turned out to be meaningless character combinations and were excluded. Up to three translations (related meanings) were encoded per stem. The POSes of the translations are not necessarily the same as those of the stems, since the POS of a `POSTaggedStem` from prefixation analysis is the

[184] file *Stem meanings.csv*; file format in Appendix 20.

same as that of the prefixation, while the POS of a `POSTaggedStem` from suffixation analysis is determined by the morphological rule which generated the `POSTaggedSuffixation` from which it was created.

5.3.17.3.2 Stem Interpretation Procedure

A `TranslatedStem` is created from each record in the stem translations file and is added to a stem translations map[185], in which each key is a stem word form and each value is a set of corresponding translated stems. Once every `TranslatedStem` has been read into the stem translations map, the word form of each `POSTaggedStem` in the stem dictionary is looked up in the stem translations map. If a matching entry is found then the `TranslatedStem` set carrying the stem's meanings is read from the map.

Each affix listed as a possible affix for the `POSTaggedStem` is then checked against every `TranslatedStem` in the set whose POS matches that of the `POSTaggedStem`. If the affix is not listed as an affix for any `TranslatedStem`, then the original affixation is recovered by searching through the targets of the relations of `Relation.Type.DERIVATIVE` from the stem, which are the derivatives of the stem. The original affixation is identified depending on whether the affix is a suffix or a prefix as follows:

- for a suffix, the original suffixation is the derivative which ends with the suffix, and whose POS matches that of the suffix;
- for a prefix, the original prefixation is the derivative which has a set of relations of `Relation.Type.ROOT` whose targets match the meanings of the prefix, which is stored in the prefix list of the `POSTaggedStem` as a `TranslatedPrefix`.

Once the original affixation has been recovered, the relation of `Relation.Type.DERIVATIVE` from the `POSSpecificLexicalRecord` of the `POSTaggedStem` to the original affixation is deleted, the affix is removed from the `POSTaggedStem` and the affixation is restored to the atomic dictionary.

[185] `Map<String, Set<TranslatedStem>>`

Once all the affixes of the `POSTaggedStem` have been checked in this way, translating relations are encoded between the `POSTaggedStem` and every meaning[186] of each `TranslatedStem` in the set with a matching POS (Appendix 18)[187].

5.3.17.4 Stem Analysis

A complete morphological analysis of the contents of the stem dictionary has not been attempted within the project scope because stem morphology largely comprises the morphology of languages other than English, from which most of the stems originate. Stem analysis as described here is conducted to the extent possible with the aid of existing morphological rules and existing algorithms with minor modifications. It is performed using the Word Analysis Algorithm (§5.2.1) and a `FlexibleWordBreaker`, a new subclass of `WordBreaker` (§5.3.11.4) which has a POS field and an embedded stem instead of an embedded word. Its `delete` method (`FlexibleWordBreaker.delete(int start, int end)`) can perform either prefix stripping or suffix stripping, by replacing the embedded stem with a morpheme which is either a `Prefixation` (if `start` is equal to 0) or a `POSTaggedSuffixation` (if `end` is equal to the length of the embedded word). The method returns a `TranslatedPrefix` (if `start` is equal to 0) or the `POSTaggedSuffixation` (if `end` is equal to the length of the embedded word). The next 2 subsections describe the functionality of `FlexibleWordBreaker.delete(int start, int end)` for prefix stripping and for suffix stripping.

5.3.17.4.1 Prefix Stripping for Stem Analysis

Unless the prefix specified by `start` and `end` is listed as an irregular prefix footprint in the irregular prefix map, a `Prefixation` and a new stem are generated in the same way[188]

[186] A fatal error occurs if any meaning of any `TranslatedStem` in the stem translations map is not in the main dictionary or if the same `Relation` is already encoded as a different subclass of `LexicalRelation`.
[187] This does not address the ambiguity illustrated in table 47. To address this would require the creation of a separate POSTaggedStem for the distinct meanings and reassignation of the affixes accordingly. This in turn would require the redefinition of class POSTaggedStem.
[188] by `WordBreaker.delete(int start, int end)`.

as described in §5.3.11.4.1. The new stem replaces the old stem as the embedded stem. The `TranslatedPrefix` component of the `Prefixation` is returned.

If the prefix specified is listed as an irregular prefix footprint, a list is made of every `IrregularPrefixRecord` to which the prefix footprint maps in the irregular prefix map. That `IrregularPrefixRecord` in the list which has the most instances is selected for the purpose of stem identification and a new stem is formed using that `IrregularPrefixRecord` in the same way as by an `IrregularWordBreaker` (§5.3.11.4.2). A `ComplexPrefixation` (Class Diagram 13) is then generated encapsulating the new stem and a `TranslatedPrefix` list. This list includes the `TranslatedPrefix` from every listed `IrregularPrefixRecord` which yields the same new stem when stripped from the old stem in the same way. A new `TranslatedPrefix` is returned with all the meanings of every `TranslatedPrefix` in the `ComplexPrefixation`.

5.3.17.4.2 Suffix Stripping for Stem Analysis

A variant of the Root Identification Algorithm (§5.2.2) is applied to the stem embedded in `FlexibleWordBreaker` (the original stem) with the POS specified by the `FlexibleWordBreaker`, without any validity checking and without any frequency-based modification. Unless a root is found from irregular inflectional morphology or a conditional rule is successfully applied, which represents regular inflectional morphology, only the unique non-lexical morphological rule is applied from any current list of rules (§5.2.2.5), since there is no expectation of or preference for lexically valid output from the analysis of non-lexical stems. The word form of the `POSTaggedSuffixation` generated becomes the new stem and the POS encapsulated in the `FlexibleWordBreaker` (Class Diagram 12) is replaced by that of the `POSTaggedSuffixation`, which is then returned.

5.3.17.4.3 Adaptation of the Word Analysis Algorithm to Stem Analysis

Candidate lists are created, without frequency corroboration (§5.3.4.3), of candidate fronts and candidate backs for all the stems in the atomic stem dictionary. Candidate fronts are generated using, as `vocabulary`, a prefix set created from the prefix footprints held in the keysets of the regular and irregular prefix maps plus the elements of the constant array of antonymous prefixes. This includes all semantically valid prefixes found in previous rounds of automatic prefix discovery, subject to the cutoffs imposed in the creation of secondary prefix sets (§§5.3.11.6, 5.3.16.1). Candidate backs are generated using a suffix set which is a copy of the keyset of the converse morphological rules map, comprising all the suffixes for whose analysis morphological rules have been created. This includes all semantically valid suffixes found in previous rounds of automatic suffix discovery, subject to the cutoffs imposed in the creation of secondary suffix sets (§§5.3.7.3, 5.3.14.3)[189].

A single loop iterates through the stems contained in the combined keysets of `candidatesWithFronts` and `candidatesWithBacks`. If any stem has no candidate fronts then a single empty candidate front is created; if any stem has no candidate backs then a single empty candidate back is created. Each candidate list is reordered to prioritise the longest candidates. The Word Analysis Algorithm (§5.2.1.4) is then applied without recursion and with a `FlexibleWordBreaker` which triggers the following variations in the behaviour of the algorithm to handle suffix stripping and prefix stripping simultaneously[190]:

- A copy of the original POS of the `FlexibleWordBreaker` is kept and the POS of the `FlexibleWordBreaker` is restored from this copy for each new candidate front or candidate back.

[189] Rejected components are not saved. Candidate backs are reversed (§5.2.1.3) but there is no requirement for the keysets to `candidatesWithFronts` and `candidatesWithBacks` to be identical.
[190] Since the allowable combinations are prefix + stem, stem + suffix and prefix + stem + suffix, the morpheme array returned must have either 2 or 3 elements, otherwise a fatal `LemmaMismatchException` is thrown.

- An attempt is made to obtain a `POSTaggedSuffixation` from each candidate back by invoking the delete method of the `FlexibleWordBreaker` as in §5.3.11.4.2.

- An attempt is made to obtain a `TranslatedPrefix` from each candidate front by invoking the delete method of the `FlexibleWordBreaker` as in §5.3.11.4.1.

- If both a valid `POSTaggedSuffixation` and a valid `TranslatedPrefix` have been obtained, a new `POSTaggedSuffixation` is created with the word form of the `TranslatedPrefix` deleted from the beginning of the existing `POSTaggedSuffixation`, but with its other fields identical to those of the existing `POSTaggedSuffixation`.

- A core POS is defined as being the same as the current POS of the `FlexibleWordBreaker` and the core is defined to be the stem currently held in the `FlexibleWordBreaker`.

- If the core is empty and there is a valid `TranslatedPrefix` and a valid `POSTaggedSuffixation`, then the morpheme array returned comprises the `TranslatedPrefix` and the `POSTaggedSuffixation`.

- If the core is empty and there is a valid `TranslatedPrefix` but no valid `POSTaggedSuffixation`, a `POSTaggedStem` is created from the candidate back, with the `TranslatedPrefix` as its unique affix, and the morpheme array returned comprises the `TranslatedPrefix` and the `POSTaggedStem`.

- If the core is not empty and there is a valid `TranslatedPrefix` but no valid `POSTaggedSuffixation`, then a `POSTaggedStem` is created from the core, with the `TranslatedPrefix`, as its unique affix, in which case the morpheme array returned comprises the `TranslatedPrefix` and the `POSTaggedStem`.

- If the core is not empty and there is a valid `TranslatedPrefix` and a valid `POSTaggedSuffixation`, then a `POSTaggedStem` is created from the core with the `POSTaggedSuffix` representation of the original suffix component of the `POSTaggedSuffixation` as its unique affix and the morpheme array returned comprises the `TranslatedPrefix`, the `POSTaggedStem` and the `POSTaggedSuffixation`.

- In any other circumstance, a non-fatal `LemmaMismatchException` is thrown, the POS of the `FlexibleWordBreaker` is restored from the copy and execution continues with the next candidate front.

Multiple affixes are addressed by iterative stem analysis (§5.3.17.5). A mapping between the `POSTaggedStem` from the stem dictionary corresponding to the stem being analysed, and a morpheme list corresponding to the morpheme array output by the Word analysis Algorithm is added to a stem affixations map[191] .

5.3.17.4.4 Lexical Restorations

Before encoding any relation between a stem and its components, it is necessary to consider the possibility that some of the components may be words in their own right. It was assumed as probable that any *monosyllabic* component of a stem which exists as a word with the specified POS *does not carry* the same meaning as that word, but that any otherwise similar *polysyllabic* component *does carry* the same meaning. The assumption with respect to monosyllables was corroborated by analysis of result samples, but no complete check was made for valid monosyllabic components as their omission cannot cause overgeneration but only undergeneration[192]. The procedure for encoding relations between stems and their components (§5.3.17.4.5) writes to a lexical restorations file[193] any derivative-component pair where the component is polysyllabic and is found in the

[191] as a `Map<POSTaggedStem, List<Morpheme>>`.
[192] Undergeneration is relatively unimportant at this stage, given that a complete morphological analysis of the stems would require multilingual resources.
[193] *Lexical restorations.csv* (now empty)

Table 48: Stems with lexically valid polysyllabic components

Existing stem	Existing POS	Lexically valid component	Component POS
alfilerium	***NOUN***	***filer***	***NOUN***
ambidexter	ADJECTIVE	dexter	ADJECTIVE
anoperinea	NOUN	perineum	NOUN
areflexium	NOUN	reflex	NOUN
chrysanthem	***NOUN***	***anthem***	***NOUN***
cryptanalyse	VERB	analyse	VERB
cystoparalyse	VERB	paralyse	VERB
distomatos	***NOUN***	***tomato***	***NOUN***
elater	***ADJECTIVE***	***later***	***ADJECTIVE***
helianthem	***NOUN***	***anthem***	***NOUN***
hemiparas	***NOUN***	***para***	***NOUN***
hydrocannabinol	NOUN	cannabin	NOUN
indehisce	VERB	dehisce	VERB
infrigidate	VERB	frigid	ADJECTIVE
malabsorb	VERB	absorb	VERB
maladjust	VERB	adjust	VERB
malocclude	VERB	occlude	VERB
mandata	***NOUN***	***datum***	***NOUN***
metropia	***NOUN***	***opium***	***NOUN***
neocolonial	NOUN	colonial	NOUN
neoexpression	NOUN	express	VERB
neoromantic	NOUN	romantic	NOUN
oxymethyl	NOUN	methyl	NOUN
parathyroidism	NOUN	thyroid	NOUN
pedagog	***ADJECTIVE***	***agog***	***ADJECTIVE***
pedimenta	***NOUN***	***mentum***	***NOUN***
pretending	***ADJECTIVE***	***tending***	***ADJECTIVE***
sideropenium	***NOUN***	***open***	***NOUN***
subdivided	ADJECTIVE	divide	VERB
suprainfect	VERB	infect	VERB
supraorbit	NOUN	orbit	NOUN
uranalyse	VERB	analyse	VERB
xeranthem	***NOUN***	***anthem***	***NOUN***

main dictionary. Initial results are shown Table 48, where incorrect analyses, which defy the assumption with respect to polysyllables, are in bold italics. To correct these results a

lexical restorations stoplist[194] (Table 49) is required, comprising all the invalid components[195].

Table 49: Lexical restoration stoplist

Morpheme	POS
agog	ADJECTIVE
anthem	NOUN
datum	NOUN
filer	NOUN
later	ADJECTIVE
mentum	NOUN
open	NOUN
opium	NOUN
para	NOUN
tending	ADJECTIVE
tomato	NOUN

5.3.17.4.5 Encoding of Relations between Stems and their Components

(*a more code-like representation of this subsection is available in Appendix 61).*

An outer loop iterates through each entry in the stem affixations map, where each key is a derivative `POSTaggedStem` and each value is a list of component morphemes. Stems which have already been interpreted (§5.3.17.3) are excluded from relation encoding. If the derivative has not already been interpreted, then a middle loop iterates through its components.

All the relations described here are encoded between a `POSSpecificLexicalRecord` encapsulated in the derivative stem (Appendix 18) and, except where otherwise stated, a `POSSpecificLexicalRecord` within the lexicon. The relations encoded depend on the class and the lexical validity of each component as follows:[196]

- If the component is a polysyllabic lexically valid `POSTaggedStem` not in the lexical restorations stoplist (Table 49), then relations are encoded between the

[194] `Set<POSTaggedMorpheme>`
[195] created from file *Lexical restoration stoplist.csv* (format in Appendix 20).
[196] A fatal `DuplicateRelationException` is thrown if any derivative is not a `POSTaggedWord` or is not in the main dictionary.

derivative stem and the component word. The derivative and the component are written to the lexical restorations file[197].

- If the component is a `POSTaggedStem` and is monosyllabic or lexically invalid or in the lexical restorations stoplist, then relations are encoded between the derivative stem and the component stem. The stem dictionary and atomic stem dictionary are updated with the component, its affix list and its POS.
- If the component is a `TranslatedPrefix`, then an inner loop iterates through its meanings, and, for each meaning, translating relations are encoded between the derivative `POSTaggedStem` and the meanings.
- If the component is a polysyllabic lexically valid `POSTaggedSuffixation`, not in the lexical restorations stoplist, then relations are encoded between the derivative and the component, with the type encapsulated in the `POSTaggedSuffixation`. The derivative and its POS, followed by the component and its POS are written to the lexical restorations file[198].
- If the component is a `POSTaggedSuffixation` and is monosyllabic or lexically invalid or in the lexical restorations stoplist, then a `POSTaggedStem` is created from the `POSTaggedSuffixation` and added to the stem dictionary. Its word form is added to the atomic stem dictionary (if not already present) and its POS is added to the POSes mapped to in the atomic stem dictionary by its word form. Relations are encoded between the derivative and its component, with the type encapsulated in the `POSTaggedSuffixation`.

5.3.17.5 Iterative Stem Analysis and Final Results

Stem analysis is performed iteratively with the same prefix and suffix sets, so as to recycle every new `POSTaggedStem` created through the analysis, allowing the discovery of multiple affixes. The net effect of stem analysis is to reduce the size of the atomic stem dictionary, which is measured at the start of each iteration. Iterative analysis continues

[197] *Lexical restorations.csv* (now empty)
[198] *Lexical restorations.csv* (now empty)

until the atomic stem dictionary ceases to decrease in size (after the fifth iteration). At each iteration, the contents of the contents of the stem affixations map are written to file[199]. The lexical restorations are also written to file[200]. The contents of this last file are as in the non-italicised rows in Table 48. No lexical restorations occur after the first iteration with the lexical restorations stoplist applied.

The fields of the stems in the stem dictionary are finally written to file[201]. Stem interpretation is then repeated, in case any of the interpreted stems have acquired additional affixes, but no further translations were supplied at this stage.

5.3.18 Final Result of Morphological Analysis and Enrichment

The morphological analysis of the lexicon is now complete, apart from the interpretation of stems which occur with less than 3 affixes. The lexicon has been morphologically enriched by encoding lexical relations between words, stems and compound expressions, replicating the links in the derivational trees to which these belong and showing the direction of derivation from morphological roots to their derivatives. The roots of those trees whose nodes are prefixations are extended to translations of prefixes and stems, forming an interlocking set of acyclic directed graphs which, together with the modified original model of WordNet, constitute a morphosemantic wordnet. The relation types of lexical relations defined by morphological rules convey the *semantic* relationships between the morphological relatives which are their participants, as far as can be determined automatically: such relations can be regarded as *morphosemantic*. Where semantic relationships could not be defined, *syntactic* relationships are defined by the relation types of rule-based relations: these relations are *morphosyntactic*. The hybrid methodology combining automatic affix discovery with morphological rules avoids the

[199] *StemsX0components.csv* through *StemsX1components.csv*, *StemsX2components.csv* etc.
[200] *StemsX0 Lexical restorations.csv* etc.
[201] *Affixation stems2.csv*

segmentation fallacy and requires minimal adaptation to be applied to the morphological analysis and enrichment of the lexicon component of any other lexical database.

The final results comprise 437604 lexical relations (Table 50), all based on derivational morphology. As relations are always double-encoded (§1.3.2.2), this corresponds to 218802 links or arcs between lexical records, of which 80.6% are links between words or between compound expressions and words and 19.4% are links between a word and a stem. 21.0% of the links are between a prefixation or a stem and the translation of a prefix or stem. 89.5% of the links make connections between specific parts of speech, 7.2% are specific at one end and only 3.3% specify a part of speech at neither end. The main dictionary and stem dictionary are serialised and written to a serialised object file[202]. Of 145224 words and phrases in the main dictionary at the start of the morphological analysis, only 5917 remain in the atomic dictionary at the end. This means that 95.9% of the words and phrases in the WordNet model have been analysed.

Table 50: Lexical relations encoded from morphological analysis

	Relations	Links
Lexical relations	437604	218802
Lexical relations where source is stem	42394	42394
Lexical relations where target is stem	42394	
Word-to-word lexical relations	352816	176408
Translating lexical relations	91778	45889
Non-translating lexical relations	345826	172913
POS-specific lexical relations	391492	195746
POS-sourced lexical relations	15745	15745
POS-targeted lexical relations	15745	
POS-less lexical relations	14662	7311

Table 51 shows that the mean number of lexical relations per synset is much higher for prepositions than for any other POS. This reflects the preponderance of prepositions among prefix translations. The relatively high figure for adverbs can be accounted for

[202] *morphlex.wnt*. The morphosemantic wordnet can be reassembled for use by applications from files *bearnet.wnt* (the pruned wordnet enriched with prepositions which was the starting point of the morphological analysis) and *morphlex.wnt.* Clearly, it would be desirable for this data to be made available in a more widely recognised format, but there is no standard for the representation of wordnets, unless the *Prolog* format (Appendix 65) be considered as such.

partly by adverbs which are homonyms of prepositions and partly by the high number of adverbs regularly derived from adjectives by appending the "-ly" suffix.

Table 51: Lexical relation densities for each POS

POS	No. of lexical relations	Synset count after pruning	Mean relations per synset
NOUN	258863	75455	3.43
VERB	46636	13767	3.39
ADJECTIVE	65351	18156	3.60
ADVERB	19607	3621	5.41
PREPOSITION	16780	800	20.98
All POSes	**407237**	**111799**	**3.64**

The successful enrichment of the WordNet-based lexicon fulfils the project objective. The precision and recall of each phases have been provided at the end of the description of the phase, wherever it is possible to quantify these. As some results are open to lexicographic interpretation and all are open to lexicographic evaluation, sample results have been provided in the Appendices and the filenames of the full analysis results have been provided in the footnotes. The usefulness of the morphological enrichment however remains to be evaluated. This will be assessed in the next chapter, which will investigate what impact morphological enrichment has on the performance of an established, WordNet-based disambiguation algorithm.