Unpacking Textuality

Cecilia FK Pun

co-tutelle PhD student, City University of Hong Kong/ University of Sydney

P/T SRA, the Halliday Centre for Intelligent Applications of Language Studies, CityU

14-Oct-2011 Friday Seminar, Sydney

Questions

- How do Periodicity, entities and commitment intersect?
- How are meanings committed in line with patterns of Theme and New at all levels?
- How can we better understand disciplinary knowledge through this kind of analysis?

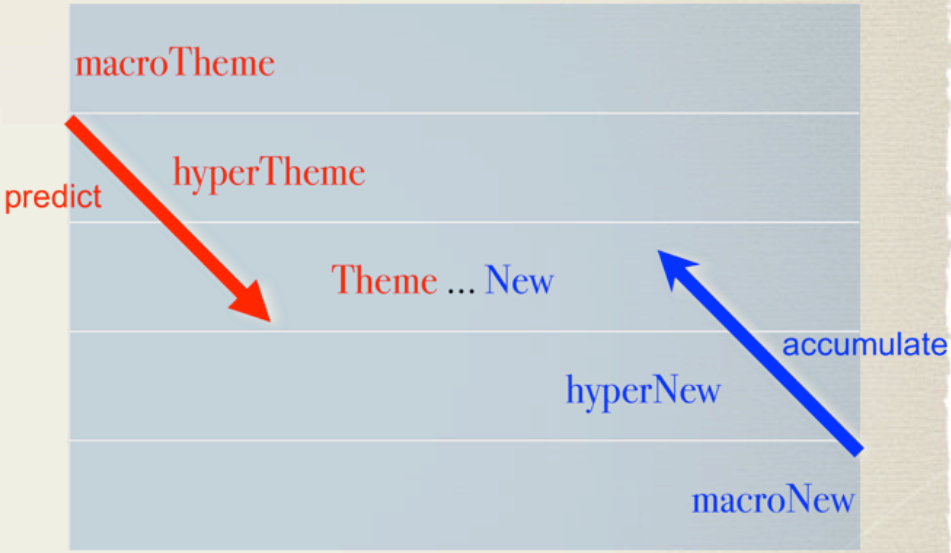
Overview

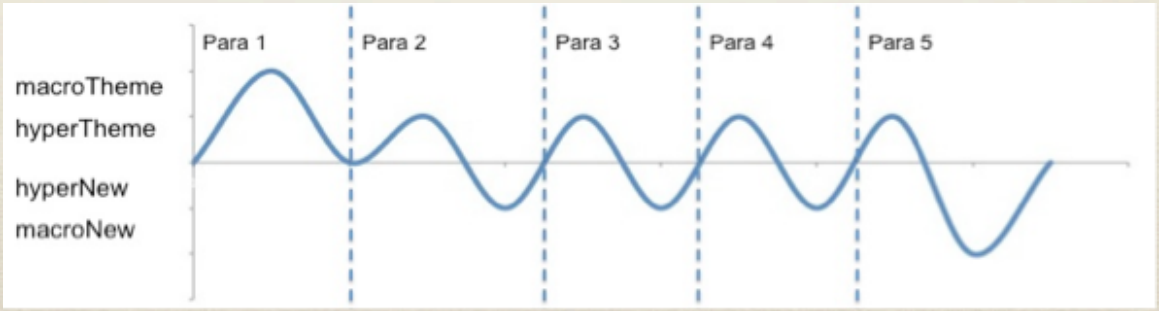
- Unfolding of textual continuity in text through
  - Periodicity
  - Abstract Entities
  - Commitment Theory

- On technical report (computational linguistics)
- On comparative interpretation (morphology)

Periodicity

- “is concerned with information flow- with the way in which meanings are packaged to make it easier for us to take them in” (Martin & Rose, 2007:187)
- Prediction- macroTheme, hyperTheme, Theme
- Accumulation- New, hyperNew, macroNew





Abstraction

- Different views on abstraction/ (abstract) nouns/ entities
  - General noun (Halliday & Hasan, 1976)
  - Vocabulary 3 (Winter, 1977)
  - Anaphoric nouns (Francis, 1986)
  - Simple Things (Halliday & Matthiessen, 1999)
  - Shell nouns (Schmid, 2000)
  - Signalling noun (Flowerdew, 2003)
  - Kinds of entities (Martin & Rose, 2007)
  - Cline of abstraction (Dreyfus & Jones, 2008)

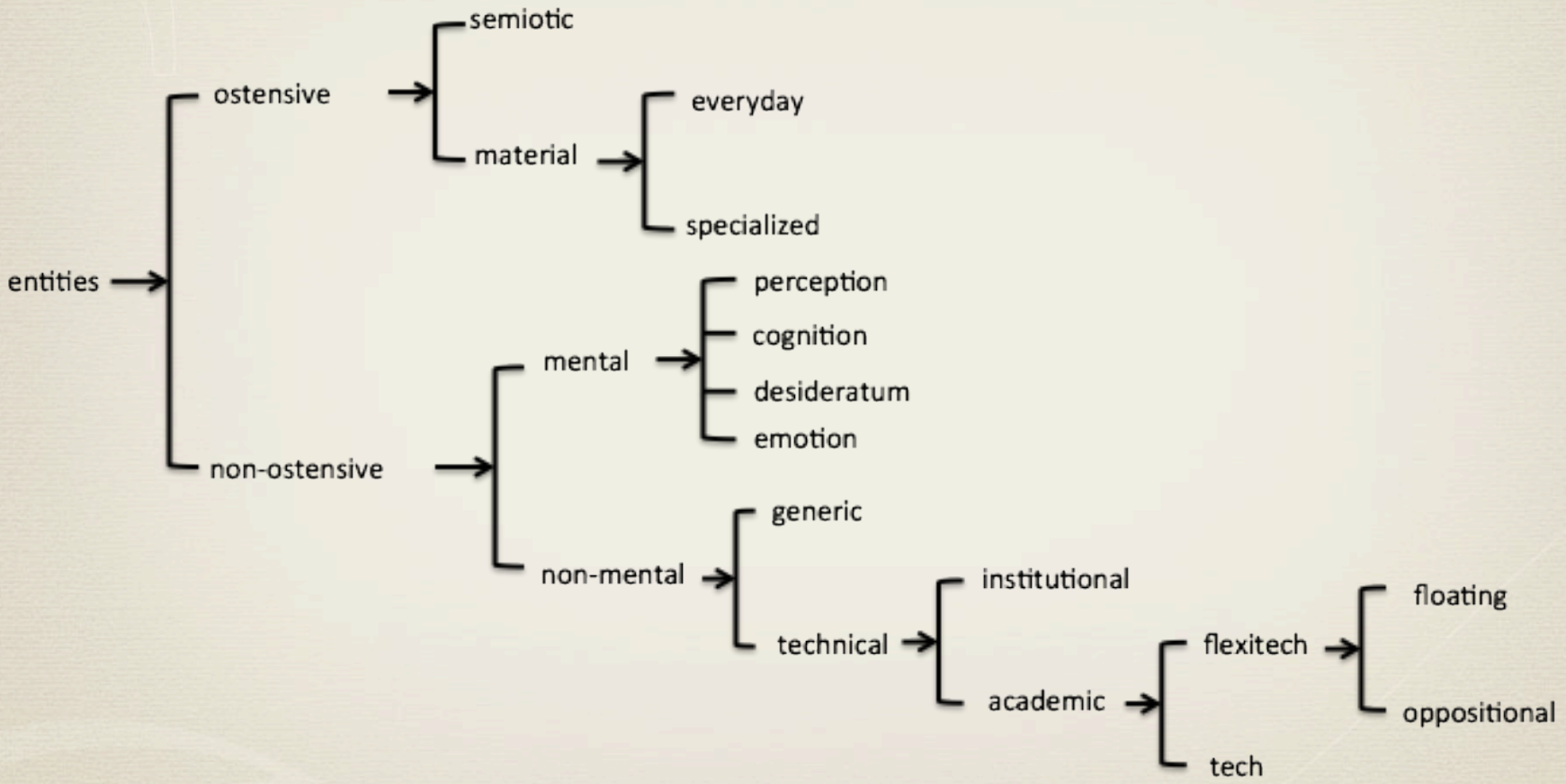
Semantic density & semantic gravity?
Knowledge code & knower code?

Kinds of Entities

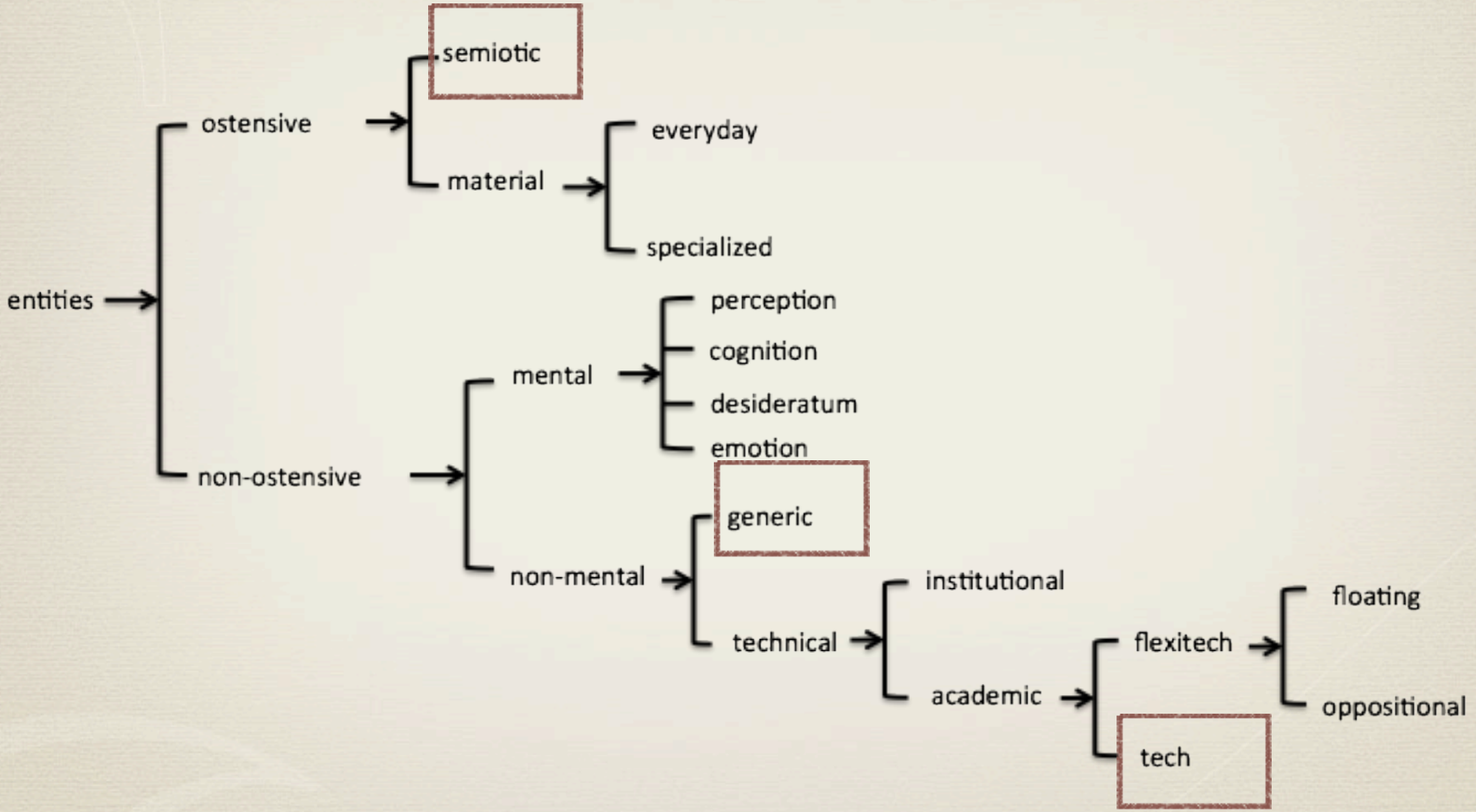
Kinds of Entities (Martin & Rose 2007:114)

Kinds of entities	Sub-kind	Examples
Concrete	everyday	*man, girlfriend, face, hands, apple, house, hill*
	specialized	*mattock, lathe, gearbox*
Abstract	technical	*inflation, metafunction, gene*
	institutional	*offence, hearing, applications, violation, amnesty*
	semiotic	*question, issue, letter, extract*
	generic	*colour, time, manner, way, kind, class, part, cause*
Metaphoric	process	*relationship, marriage, exposure, humiliation*
	quality	*justice, truth, integrity, bitterness, security*

Entities


entities
ostensive
semiotic
material
everyday
specialized
non-ostensive
mental
perception
cognition
desideratum
emotion
non-mental
generic
technical
institutional
academic
flexitech
floating
oppositional
tech

Entities

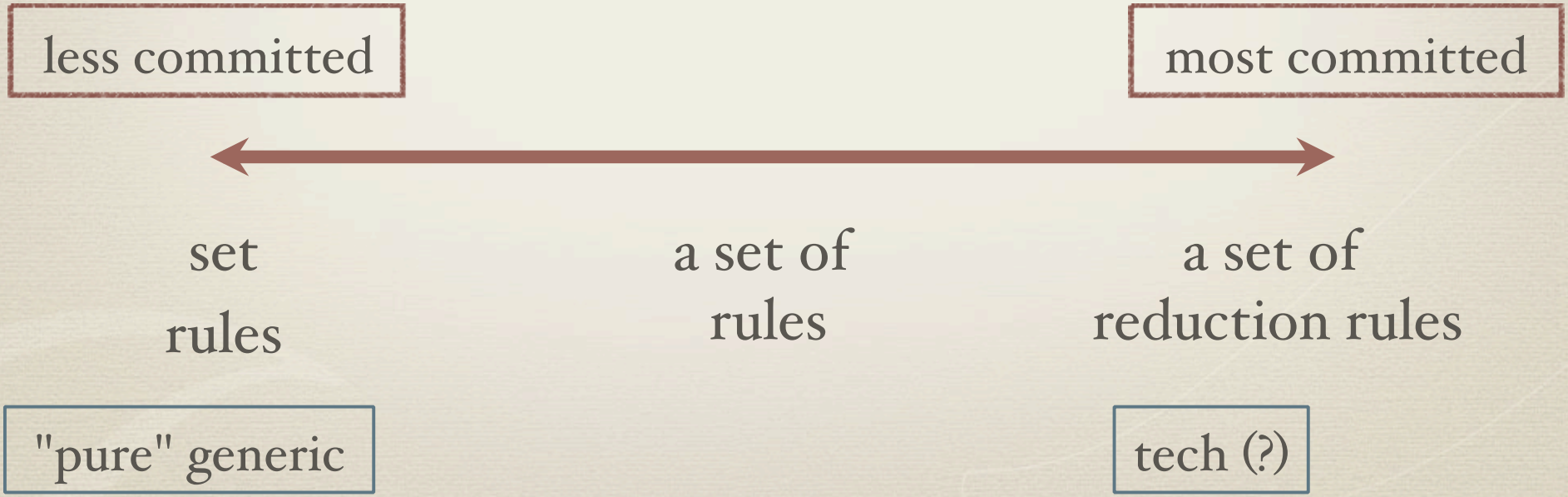

entities
ostensive
semiotic
material
everyday
specialized
non-ostensive
mental
perception
cognition
desideratum
emotion
non-mental
generic
technical
institutional
academic
flexitech
floating
oppositional
tech

(Abstract) Entities

- **ostensive**
  - **semiotic** (meaning realized in a visible way; non-discipline-specific)
    - eg. text, paragraph, segment, …
- **non-ostensive**
  - **generic** (naming kinds of things; non-discipline-specific)
    - eg. fragments, order, rules, factors, kinds, ...
  - **tech** (learned through institutional settings; discipline-specific)
    - eg. lemmaization, information retrieval, inflected verb, ...

Commitment Theory

- Provides explanatory power for findings
- Provides another dimension for examining meaning in text
- Mapping the (abstract) entities and commitment
  - meaning potential



Texts

Technical report (computational linguistics)

Comparative interpretation (morphology)

Genre

- Technical report (computational linguistics)
- Stages: (Abstract) ^ Orientation ^ Description & Discussion ^ Coda

- Comparative interpretation (morphology)
- Stages: (Abstract) ^ Orientation ^ Description ^ Discussion ^ Coda

Technical Report

computational linguistics

Model Text

- Discussion on observation and findings
  - on lemmatisation

- Periodicity structure
  - macroTheme, hyperThemes, macroNew
    - abstract entities & commitment
- 2 paragraphs
  - abstract entities & commitment

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

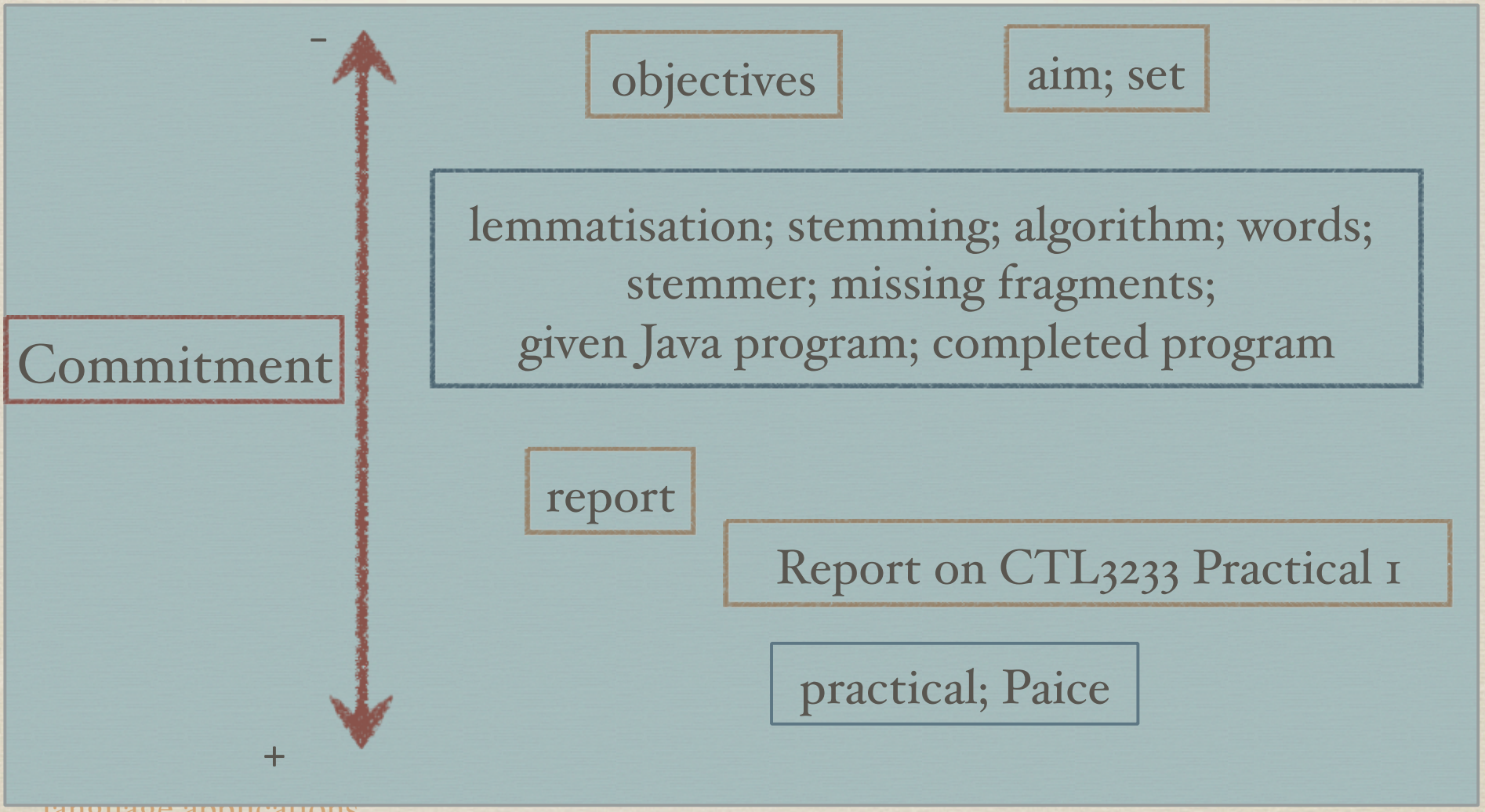
Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.



language applications.

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

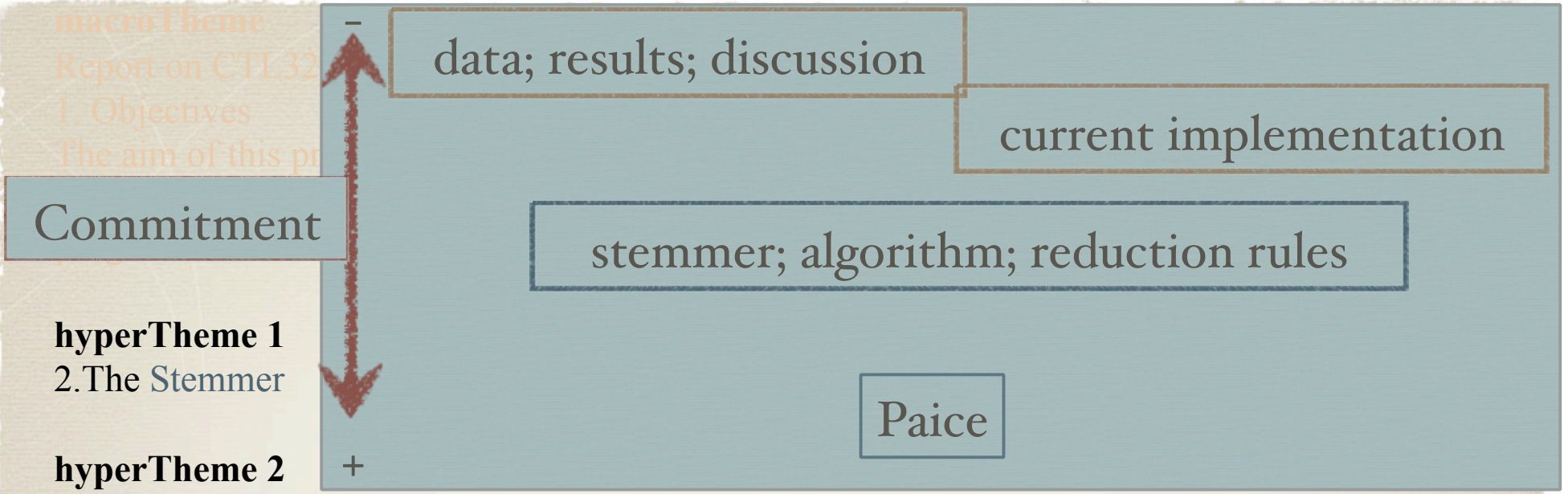
hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.



hyperTheme 1
2.The Stemmer

hyperTheme 2
The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3
3. Data

hyperTheme 4
4. Results and Discussion

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

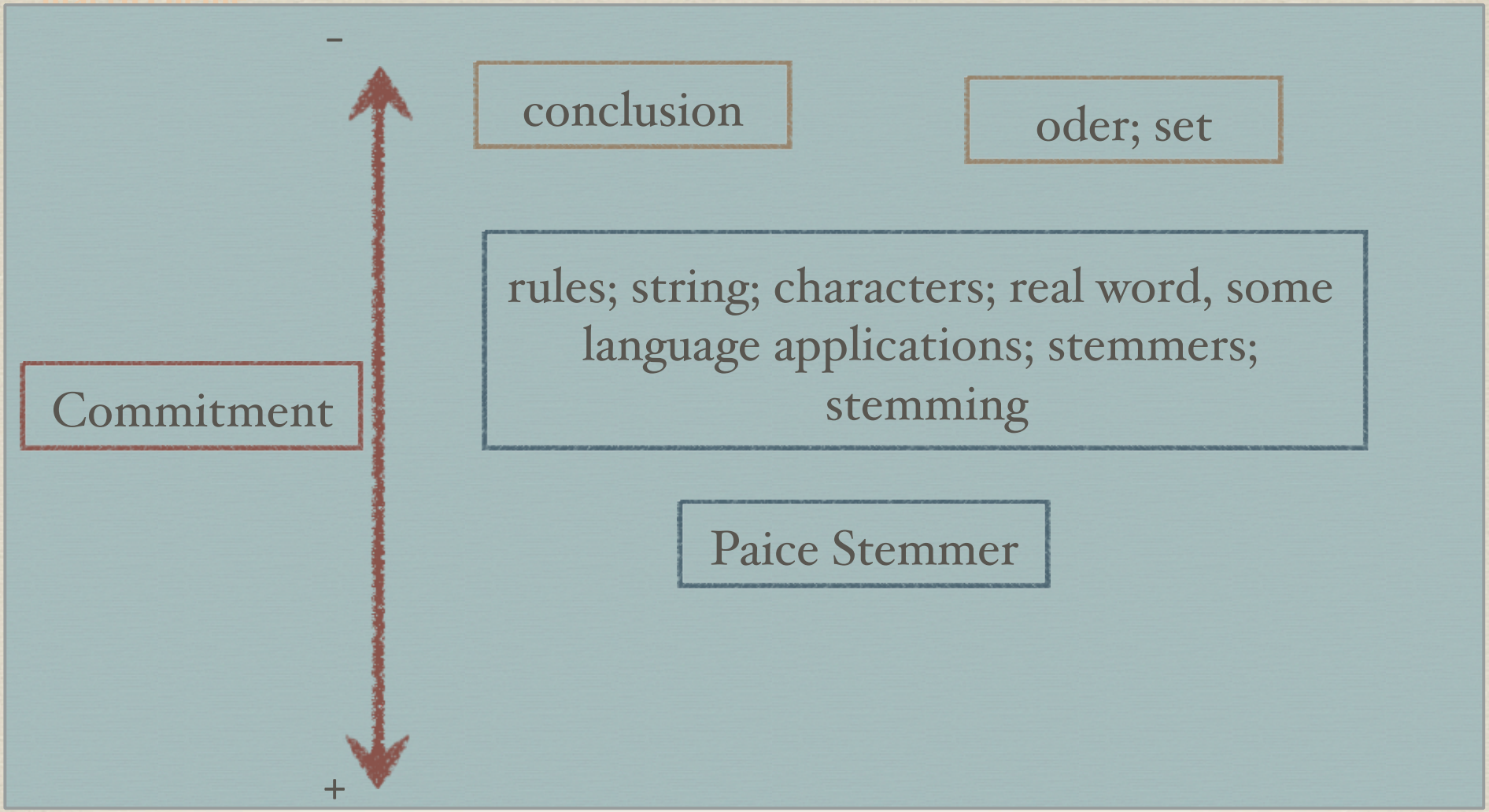
hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.



macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

So far...

- tech & generic
- moving from more committed to less

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

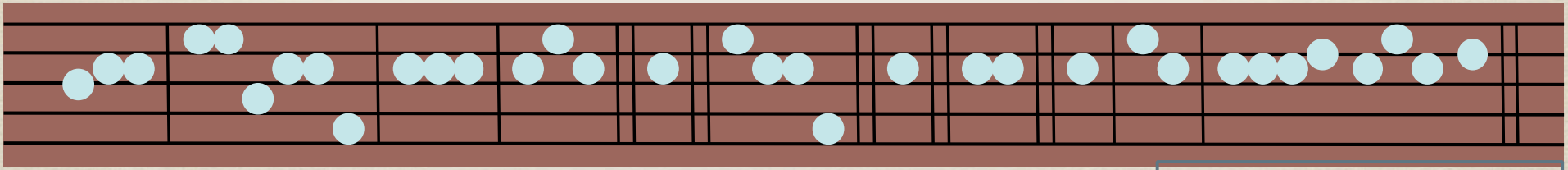
hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.



semantic wave?

macroTheme

Report on CTL3233 Practical 1: Lemmatisation and Stemming

1. Objectives

The aim of this practical is to implement and test a stemmer based on the algorithm by Paice (1977). The algorithm was thoroughly studied and the missing fragments in the given Java program were filled in. The completed program was tested on a set of words.

hyperTheme 1

2.The Stemmer

hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

hyperTheme 3

3. Data

hyperTheme 4

4. Results and Discussion

macroNew

5. Conclusion

Hence the Paice Stemmer was implemented and tested. The order of the rules is important. Although the string of characters resulting from stemming might not always form a real word, stemmers only need a set of rules and run fast, and are therefore widely used in some language applications.

hyperTheme 1

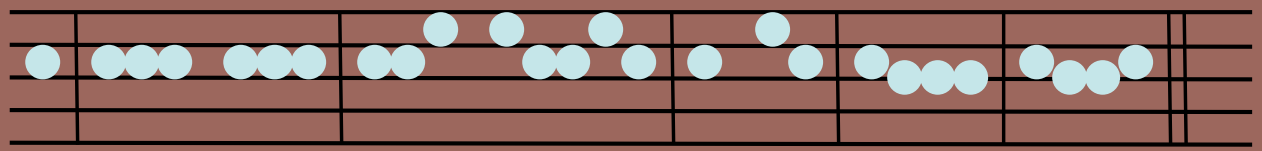
2.The Stemmer

A stemmer is used to reduce a given word form to its root word, like a lemmatiser reducing a word to its lemma. However, a lemmatiser is expected to always give a linguistically valid word as the result, and it therefore consists of a set of reduction rules with a dictionary to check whether the result is a proper word. This is not necessary for a stemmer, which only requires a set of reduction rules. For example, a rule like “ATIONAL --> ATE” would replace the word-ending“ational” with “ate”. Thus a word like “relational” would be reduced to“relate” by a stemmer.

hyperTheme 1

2.The Stemmer

A stemmer is used to reduce a given word form to its root word, like a lemmatiser reducing a word to its lemma. However, a lemmatiser is expected to always give a linguistically valid word as the result, and it therefore consists of a set of reduction rules with a dictionary to check whether the result is a proper word. This is not necessary for a stemmer, which only requires a set of reduction rules. For example, a rule like "ATIONAL --> ATE" would replace the word-ending"ational" with "ate". Thus a word like "relational" would be reduced to"relate" by a stemmer.



hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

There were 26 rules. The rules were listed in a plain text file, as in the following examples:

```
-       ies     y   ARY
ARY     ary     -   finish
```

The first rule above deals with the ending “ies”, which would be removed and then replaced with “y”, and the program would jump to the rule with label “ARY” and continue the process. The second rule has label “ARY” and deals with the ending “ary”, which would simply be removed. The main program would thus traverse the rules one by one, try to match with the ending of the input word, and perform the necessary action accordingly. Hence, if the input word is “dictionaries”, it will first be reduced to “dictionary” by replacing “ies” with “y”, and then to “diction” by removing “ary”. The completed program fragment is shown below.

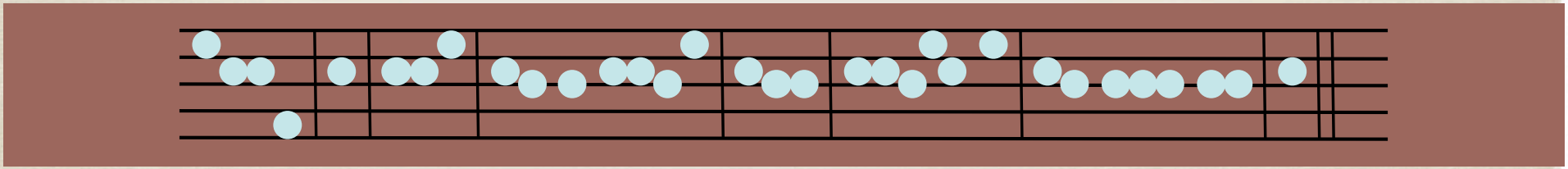
hyperTheme 2

The current implementation was based on the algorithm and reduction rules described in Paice (1977).

There were 26 rules. The rules were listed in a plain text file, as in the following examples:

```
-       ies     y   ARY
ARY     ary     -   finish
```

The first rule above deals with the ending “ies”, which would be removed and then replaced with “y”, and the program would jump to the rule with label “ARY” and continue the process. The second rule has label “ARY” and deals with the ending “ary”, which would simply be removed. The main program would thus traverse the rules one by one, try to match with the ending of the input word, and perform the necessary action accordingly. Hence, if the input word is “dictionaries”, it will first be reduced to “dictionary” by replacing “ies” with “y”, and then to “diction” by removing “ary”. The completed program fragment is shown below.



Comparative interpretation

Morphology

Model Text

- Explain the differences of the two morphological processes
- Periodicity structure
  - macroTheme, hyperThemes, macroNew
    - abstract entities & commitment

macroTheme

According to Fromkin (2000), morphology is the study of word formation and word structure. Two methods of word formation are inflection and derivation. In inflection, morphemes are added to words to signal grammatical relations (Stump 2001). Inflectional morphemes are required by syntactic rules. Unlike inflection, derivation forms words with new lexical meanings (Beard 2001). Further, derivational morphemes are not required by syntactic rules. This essay will compare inflectional morphology with derivational morphology by examining their functions and natures. Examples from the article "The Australian Accent" are used to illustrate the differences.

hyperTheme 1

Inflectional morphology serves grammatical functions whereby morphemes are added to words to make a sentence grammatical.

hyperTheme 2

Derivational morphology refers to the formation of new words by adding derivational morphemes to the existing words (base).

hyperTheme 3

There are a number of differences between inflectional morphology and derivational morphology.

macroNew

In conclusion, inflectional morphology and derivational morphology differ in their function, nature and the effect imposed on the formed words. Inflectional morphology serves grammatical functions whereas derivational morphology forms words with new lexical meanings. Moreover, inflection does not change the part of speech of the inflected words but derivation usually changes the part of speech of the derived words. Finally, inflection is more regular than derivation.

macroTheme

According to Fromkin (2000), morphology is the study of word formation and word structure. Two methods of word formation are inflection and derivation. In inflection, morphemes are added to words to signal grammatical relations (Stump 2001). Inflectional morphemes are required by syntactic rules. Unlike inflection, derivation forms words with new lexical meanings (Beard 2001). Further, derivational morphemes are not required by syntactic rules. This essay will compare inflectional morphology with derivational morphology by examining their functions and natures. Examples from the article "The Australian Accent" are used to illustrate the differences.

hyperTheme 1

Inflectional morphology serves grammatical functions whereby morphemes are added to words to make a sentence grammatical.

hyperTheme 2

Derivational morphology refers to the formation of new words by adding derivational morphemes to the existing words (base).

hyperTheme 3

There are a number of differences between inflectional morphology and derivational morphology.

macroNew

In conclusion, inflectional morphology and derivational morphology differ in their function, nature and the effect imposed on the formed words. Inflectional morphology serves grammatical functions whereas derivational morphology forms words with new lexical meanings. Moreover, inflection does not change the part of speech of the inflected words but derivation usually changes the part of speech of the derived words. Finally, inflection is more regular than derivation.

macroTheme

According to Fromkin (2000), morphology is the study of word formation and word structure. Two methods of word formation are inflection and derivation. In inflection, morphemes are added to words to signal grammatical relations (Stump 2001). Inflectional morphemes are required by syntactic rules. Unlike inflection, derivation forms words with new lexical meanings (Beard 2001). Further, derivational morphemes are not required by syntactic rules. This essay will compare inflectional morphology with derivational morphology by examining their functions and natures. Examples from the article "The Australian Accent" are used to illustrate the differences.

hyperTheme 1

Inflectional morphology serves grammatical functions whereby morphemes are added to words to make a sentence grammatical.

hyperTheme 2

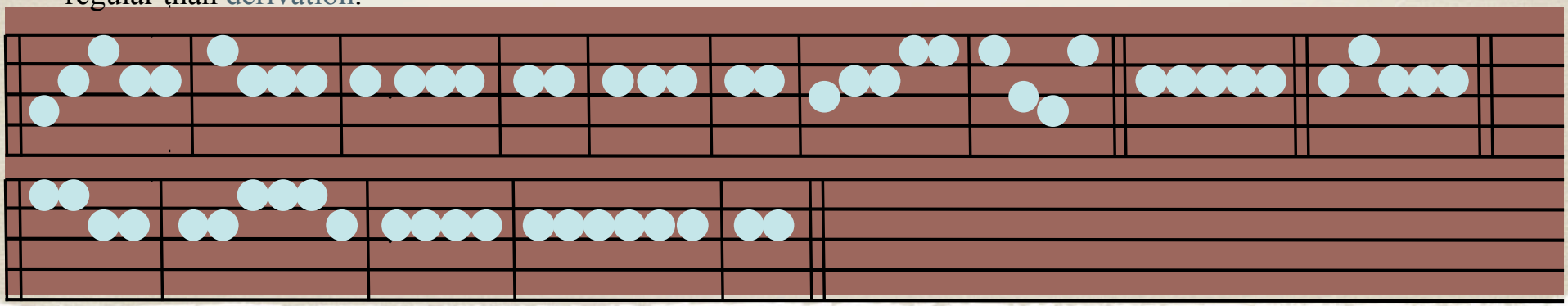
Derivational morphology refers to the formation of new words by adding derivational morphemes to the existing words (base).

hyperTheme 3

There are a number of differences between inflectional morphology and derivational morphology.

macroNew

In conclusion, inflectional morphology and derivational morphology differ in their function, nature and the effect imposed on the formed words. Inflectional morphology serves grammatical functions whereas derivational morphology forms words with new lexical meanings. Moreover, inflection does not change the part of speech of the inflected words but derivation usually changes the part of speech of the derived words. Finally, inflection is more regular than derivation.



Periodicity

- Technical report (computational linguistics)

macroTheme

Objective

hyperTheme 1

Description on program concerned

hyperTheme 2

Test Data

hyperTheme 3

Results and Discussion

macroNew

Summary

- Comparative interpretation (morphology)

macroTheme

Objective

hyperTheme 1

Introduction

hyperTheme 2

Derivational morphology

hyperTheme 3

Differences of the two morphological processes

macroNew

Summary

Entities & Commitment

- abstract entities
  - highly technical
- commitment theory
  - moving from more committed to less committed

Next steps...

- mapping of entities with commitment theory
- LCT cosmologies?
  - Semantic wave
  - Knowledge code and knower code?
  - Cosmologies?

- "Dative power" of generic, semiotic and tech within NG?
  - - focus
  - Epithet/Classifier^Thing
  - Thing^Qualifier

- students' texts
  - macroTheme, hyperThemes, macroNew
    - abstract entities & commitment

Comment? Feedback? Question?