Exploring automatic Theme identification: A rule-based approach
Lara Schwarz, Sabine Bartsch, Richard Eckart and Elke Teich

Abstract. Knowledge about Theme-Rheme serves the interpretation of a text in terms of its thematic progression and provides a window into the topicality of a text as well as text type (genre). This is potentially relevant for NLP tasks such as information extraction and text classification. To explore this potential, large corpora annotated for Theme-Rheme organization are needed. We report on a rule-based system for the automatic identification of Theme to be employed for corpus annotation. The rules are manually derived from a set of sentences parsed syntactically with the Stanford parser and analyzed in terms of Theme on the basis of Systemic Functional Grammar (SFG). We describe the development of the rule set and the automatic procedure of Theme identification and assess the validity of the approach by application to some authentic text data.

1 Introduction
Text data applications of NLP, such as information extraction (IE) or document classification (DC), require a new look at issues of discourse parsing. While the focus in discourse parsing has been on qualitative analyses of single texts – for instance identifying the meaningful, coherent parts of a text (generic structure, rhetorical structure, logical structure; see e.g., ??), interpreting reference relations (co-reference resolution) or analyzing information structure (e.g., ?) – the attention of NLP in IE/DC is on sets of texts and quantitative features. So far, the potential contribution of discourse knowledge for IE/DC applications has hardly been explored, since the predominant methods are string or word-based and even supervised data mining rarely employs information at higher levels of linguistic abstraction. Here, the bottleneck is often the lack of (large enough) corpora annotated in terms of discourse features.

The work reported on in this paper is placed in the context of enhancing corpora with linguistic features of discourse organization, an increasingly active research area (see e.g., ?, ? ?). We report on the derivation of rules for automatic Theme identification from a set of sample sentences instantiating the principal Theme options of English. Our approach combines automatic syntactic parsing with annotation of Theme (cf. the work by ? on enhancing the Penn Treebank in terms of features from Systemic Functional Grammar ?, or ? on annotating corpora in terms of Topic-Focus articulation). Even if automatic Theme identification may not achieve 100 % accuracy (and manual postediting might be needed), corpora annotated for Theme would
clearly be very useful as input for various machine learning applications as well as for linguists wishing to explore patterns of thematic progression in texts.

The paper is organized as follows. In Section 2 we present the underlying definition of Theme-Rheme employed in our work. Section 3 explains the experimental setting and presents the types of rules derived from the syntactic structure representation as well as the procedure for rule application. Section 4 discusses some results. We conclude with a summary and issues for future work (Section 5).

2 Definition of Theme in English

According to Systemic Functional Grammar (?: 38), the Theme is defined as the point of departure of the clause as message. It is the contextual anchor of the clause that is oriented towards the preceding discourse (?: 531). The rest of the clause is the Rheme. In English, the Theme occupies the first constituent position in the clause:

“As a general guide to start off with, we shall say that the Theme of a clause is the first group or phrase that has some function in the experiential structure of the clause.” (?: 66)

Themes can be either simple (consisting of one Thematic element) or complex (consisting of multiple Thematic elements). Figure ?? displays the major simple Theme options for English declarative clauses. For some examples illustrating the most common options see examples ??-?? below (Themes are underlined).

[1] The duke gave my aunt this teapot. [unmarked; nonpredicated, nonsubstitute]
[2] On Saturday night, I lost my wife. [marked, circumstantial; nonpredicated, nonsubstitute]
[3] What the duke gave to my aunt was a teapot. [unmarked; nonpredicated, substitute]
For clause moods other than declarative (i.e. interrogative, imperative), the options are partly overlapping and partly mood-specific (see examples ??-??).

Apart from these basic options with only one constituent filling the Theme position, there are also multiple Themes containing more than one constituent. For some examples, see ??-?? below.

In these examples, we have topical Themes, what; think, the constituents of which at the same time have textual and interpersonal function respectively: Aesthetically; Well; Jane.

3 Patterns, rules and rule interpretation

The SFG definition of Theme is based entirely on clause syntax, which means that the syntax can be used as the only criterion to identify the Theme. The basis for the identification of syntactic patterns matching Theme is provided by the Stanford PCFG Parser ?, a probabilistic parser with flexible output (parts of speech, phrase structure, dependency structure). For parsing results for examples ?? and ?? see Figures ?? and ??.. Below, we describe the syntactic patterns found matching the different Theme types (Section ??) and present the procedure implemented for Theme identification (Section ??).

3.1 Syntactic patterns and Theme types

Manual Theme identification comprises two steps:

- determining the boundaries of the Theme in terms of syntactic phrases;
- assigning a Theme type to the unit identified.
In the simplest case, the Theme is the left-most constituent immediately dominated by the $S$ node in the parse tree (see Figures ?? and ??). This is, in fact, also the most frequent case in English (Subject-Theme, Circumstance-Theme; see again examples ?? and ?? above). However, there are many syntactically more complex Themes, as shown in examples ??-?? above. Also, in order to assign a Theme type, we must take into consideration the mood of the clause (declarative, imperative or interrogative, see examples ??-??). For each mood type, several different syntactic structures are possible and for each of the possible structures, we can have an unmarked or marked Theme variety. Each combination of syntactic structure, mood and markedness/unmarkedness of Theme creates its own pattern (or set of patterns) to which a Theme type can be assigned. Identification of the Theme then amounts to specifying the possible patterns in terms of partial syntactic trees that map onto the different Theme types. On the basis of these patterns, a set of rules can be specified.
Exploring automatic Theme identification: A rule-based approach

<table>
<thead>
<tr>
<th>#</th>
<th>Theme Type</th>
<th>Example</th>
<th>Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unmarked Subject</td>
<td>The duke has given my aunt that teapot.</td>
<td>(S(NP(VP())))</td>
</tr>
<tr>
<td>2</td>
<td>Existential There</td>
<td>There were three jovial Welshmen.</td>
<td>(S(NP(VP())))</td>
</tr>
<tr>
<td>3</td>
<td>Unmarked Nominalization</td>
<td>What the duke gave to my aunt was that teapot.</td>
<td>(S(SBAR(VP())))</td>
</tr>
<tr>
<td>4</td>
<td>Marked Adjunct: Adverbial Phrase</td>
<td>Merrily we roll along.</td>
<td>(S(ADVP(VP())))</td>
</tr>
<tr>
<td>5</td>
<td>Marked Adjunct: PP</td>
<td>From house to house I went my way.</td>
<td>(S(PP(VP())))</td>
</tr>
<tr>
<td>6</td>
<td>Marked Object as Theme</td>
<td>The teapot the duke has given my aunt.</td>
<td>(S(NP(VP())))</td>
</tr>
<tr>
<td>7</td>
<td>Marked Nominalization</td>
<td>What they could not eat that night the Queen next morning fried.</td>
<td>(S(SBAR(NP(VP))))</td>
</tr>
<tr>
<td>8</td>
<td>Exclamative</td>
<td>How dreadful she sounds.</td>
<td>(S(ADJP(VP())))</td>
</tr>
<tr>
<td>9</td>
<td>Unmarked WH-Question</td>
<td>Who wants a glass of wine?</td>
<td>(SBARQ(VP()))</td>
</tr>
<tr>
<td>10</td>
<td>Marked Yes/No Interrogative</td>
<td>Is there a glass of wine?</td>
<td>(SQ(VP()))</td>
</tr>
<tr>
<td>11</td>
<td>Marked Inverted Clause</td>
<td>The right is it?</td>
<td>(SINV(VP()))</td>
</tr>
<tr>
<td>12</td>
<td>Unmarked Imperative</td>
<td>You roll.</td>
<td>(S(VP()))</td>
</tr>
<tr>
<td>13</td>
<td>Unmarked Thematic Equative</td>
<td>The one who gave my aunt that teapot was the duke.</td>
<td>(S(NP(SBAR(VP))))</td>
</tr>
<tr>
<td>14</td>
<td>Marked Thematic Equative</td>
<td>That is the one I like.</td>
<td>(S(NP(DT)(VP(SBAR(VP))))(VP())))</td>
</tr>
</tbody>
</table>

Table 1. Simple Theme patterns and examples

that match the partial trees (see Section ??). For a start, we focus on simple Themes.

The patterns and corresponding rules were developed using a set of 85 sentences taken from (?: 65–81) which are considered representative of the different Theme types in English, but exhibit some syntactic variation. The examples were run through the parser and annotated manually for Theme according to Halliday’s model. Clause mood, one of the factors in Theme identification, can be read off the parse tree so that the corresponding mood-specific Theme types can be accounted for. For example, we observe that a declarative sentence has a top node S, whereas an interrogative has either SBARQ or SQ as top node (see patterns in Table ??). Additionally, there are more general patterns that hold for all mood types. This includes, for instance, the options of marked Theme. This can be seen in example ??, the interrogative version of example ?? above, with an identical Theme (conflated with the Adjunct):

[10] **On Saturday night, did you lose your wife?**

Altogether, we derived 14 patterns for simple Themes from the examples, which correspond to 14 rules (see Section ??). These are displayed in Table ?? together with examples.

3.2 Rules and the tree-rule processor

We map the parsing output to an XML representation as commonly used in work on corpora ?. We are thus building on modified output from the Stanford parser for our analysis. Thus, the formal basis for rule specification is an XML representation.

The tree-rule processor is a simple pattern matcher implemented in Java which applies rules defined in a rule file to a parse read from a parse file. Figure ?? shows a trimmed down version of a parse file. A parse file can contain any number of
sentences. Even though the parser generates more information, we currently only use the category (cat) attribute in the rules.

A rule consists of a label and a pattern. The label denotes the Theme type to be assigned and is encoded in the label attribute of the rule element. The pattern is expressed as an XML fragment specifying those elements and attributes which must be present in the parse for the rule to match. It is encoded in the children of the rule element. Attributes not specified in the rule, but present in the parse, are ignored. If a particular branch of the parse is of no importance, this branch is expressed as a wildcard (relax) in the rule. Figure ?? shows the Marked Adjunct: PP rule that matches the parse in Figure ??.

Generally, the rules apply in the order of more specific to more general. This is done by matching the rules against each other. If a rule matches another, the rule that matched is less specific.

```
<Constituent cat="S">
  <Constituent cat="PP">
    <Constituent cat="IN">On</Constituent>
    <Constituent cat="NP">
      <Constituent cat="NNP">Saturday</Constituent>
    </Constituent>
  </Constituent>
  <Constituent cat="NP">
    <Constituent cat="PRP">I</Constituent>
  </Constituent>
  <Constituent cat="VP">
    <Constituent cat="VBD">lost</Constituent>
    <Constituent cat="NP">
      <Constituent cat="PRP">my</Constituent>
      <Constituent cat="NN">wife</Constituent>
    </Constituent>
  </Constituent>
</Constituent>
```

Figure 4. Example XML parse output

```
<rule id="rule 5" label="Marked Adjunct: PP">
  <Constituent cat="S">
    <!-- mark-theme -->
    <Constituent cat="PP">Sat</Constituent>
    <!-- mark-theme -->
    <Constituent cat="VP">lost</Constituent>
    <!-- mark-theme -->
  </Constituent>
</rule>
```

Figure 5. Rule: Marked Adjunct: PP
4 Application and results

To test our approach, we carried out three experiments applying the rules to (a) the set of sample sentences used as base data to derive the patterns, (b) to a small set of texts and (c) to a larger corpus of 209 abstracts.

In order to measure the quality of the rules, the tree-rule processor may be run in a training mode. In the training mode, the input parse file may be annotated at any opening tag with an expect attribute naming the rule which is expected to match at this point. At most, one rule can be expected. Whenever a rule matches at an element that does not yet expect a rule, the user is prompted with a selection of all rules that matched. The user may then choose one of the rules as the “correct” one or choose none at all. The choices are saved back to the parse file. Using these “annotated” parse files, we can generate some statistics on the performance of the rules. The rest of this section will discuss the performance of the rules on the three sets of test data.

4.1 Set of sample sentences

Table ?? shows the performance statistics of the rules on the first data set. Here (and in subsequent tables), sentences found is the total number of full sentences provided to the tree-rule processor by the parser, classified sentences is the percentage of the total number of sentences for which a match was found and total matches represents the number of times a rule matched a sentence, clause or sentence fragment. Matches MET, or true positives, is the number of times an expected rule was matched and matches UNEXPECTED are instances when no rule was expected, yet a rule matched. This also includes incorrectly parsed sentences, clauses and sentence fragments, which the tree-rule processor attempts to match to a rule. Either these fragments have no Theme and therefore no rule is expected, or the sentence is incorrectly parsed (see Section ??) and no rule can be expected to match. Matches MISMATCHed are all rules that matched, though another rule was expected (false positives). The overall precision is the percentage of correct matches in the total number of matches (including all UNEXPECTEDs).

For this first test we focused on simple Themes. Also, we removed all incorrectly parsed sentences (see Section ?? for a discussion of limitations on our approach). This left a set of 68 sentences of the original 85. For the results, see again Table ??.

In every case where a rule was expected, the correct rule was found. The UNEXPECTED matches are sentences or sentence fragments that matched a rule that was not expected, because no Theme could be identified for the fragments in the training mode.
4.2 Set of sample texts

The second data set on which we tested our approach is composed of 48 sentences from four small texts. These texts are a mixture of academic abstracts and general interest texts. They were chosen for their variety, in order to expose the tree-rule processor to a wide variety of sentence types. In this test, we did not remove incorrect parse trees or complex Themes.

For all four texts, over 83% of the sentences could be classified. A precision of 100.00% was achieved in all cases where a rule was expected. The sentences with no match are cases of complex Themes or incorrect parse trees.

4.3 Test corpus

The third set of data is a corpus composed of 700 sentences in 209 academic abstracts from the fields of mechanical engineering, linguistics, and computer science. Here, we were able to classify 89% of the sentences with a precision of 81.74%. Across the whole corpus, we had only 6 Matches MISMATCHed in close to 1000 total matches.

<table>
<thead>
<tr>
<th>Sentences found</th>
<th>700</th>
<th>Sentences found</th>
<th>48</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentences with no match</td>
<td>75</td>
<td>Sentences with no match</td>
<td>8</td>
</tr>
<tr>
<td>Classified sentences</td>
<td>623</td>
<td>Classified sentences</td>
<td>44</td>
</tr>
<tr>
<td>Avg matches per sentence</td>
<td>1.39</td>
<td>Avg matches per sentence</td>
<td>1.60</td>
</tr>
<tr>
<td>Total matches MET (true positives)</td>
<td>73</td>
<td>Total matches MET (true positives)</td>
<td>46</td>
</tr>
<tr>
<td>Total matches MISMATCHed (false positives)</td>
<td>0</td>
<td>Total matches MISMATCHed (false positives)</td>
<td>0</td>
</tr>
<tr>
<td>Total matches UNEXPECTED</td>
<td>29</td>
<td>Total matches UNEXPECTED</td>
<td>31</td>
</tr>
<tr>
<td>Overall precision (with unexpected)</td>
<td>25.57%</td>
<td>Overall precision (with unexpected)</td>
<td>50.75%</td>
</tr>
</tbody>
</table>

Table 2. Performance on sample sentences

Table 3. Performance on Test Set

4.4 Assessment of results

The results of our tests are a good indication that our approach is viable for automatically identifying Themes, and that the rules we have established so far are a solid
Exploring automatic Theme identification: A rule-based approach

foundation on which we can expand our approach. However, there are some problems concerning the syntactic parses and some limitations concerning the Theme identification procedure.

Clearly, the performance of the tree-rule processor depends on the performance of the parser. The more complex a sentence, the more likely it is that the input from the parser is “unclean”. For instance, in example ??, our tree-rule processor will attempt to match a rule for each clause, but the incorrect parse (see Figure ??) produced by the PCFG version of the Stanford parser prevents it from correctly identifying the rule for the inverted clause.

[11] Beyond the main complex is a lovely stream that bubbles under a wooden bridge, and further on are steep stone steps leading to another complex.

Figure 6. Stanford PCFG parse of a sentence containing inversion and coordination

The Factored model of the parser returns better results in cases of complex sentence structure, as can be seen from the parse of example ?? shown in Figure ?? below. Yet for the majority of simple sentences, the PCFG model provides more consistent results. One solution to this problem would be to adopt a layered approach, where complex sentences are re-parsed using the Factored model.

Another issue is the occurrence of rules matching sentence fragments. This inflates the number of unexpected results, and has a negative effect on the precision of the tree-rule processor. To counteract this, negative rules can be developed that are more specific to the problematic fragments than the Theme rules. This would label the sentence fragments for later removal from the number of matches and prevent them from being erroneously labelled as a Theme. To test this theory, one negative rule has been introduced to eliminate rules matching “to” + verb constructions, for example in We decided to go to the movies. In our set of sample sentences from Halliday, this negative rule reduced the number of unexpected matches from 29 to 22.
In the test corpus, the number of unexpected matches drops with the inclusion of the negative rule from 185 to 93. This increases the precision of the tree-rule processor on the test corpus by 4.87%. The goal is to continue identifying patterns that could be eliminated using a negative rule in order to increase the precision and reliability of the program.

5 Summary and conclusion

Automatic Theme identification has long been on the agenda for desirable annotation types that have the potential of sparking progress in studies of discourse organization. Recent research efforts in this area clearly show that in order to seriously push corpus-based discourse studies and scalable NLP applications beyond the level of the sentence, annotations at the discourse level are indispensable and preferably attained with minimal manual intervention.

This paper proposes an implementation for Theme identification on the basis of a relatively simple pattern matching algorithm that matches a set of well-defined linguistic rules against a syntactically parsed text corpus. The approach adopted is able to deal with different types of Theme in clauses with all possible mood configurations as well as with all standard simple Theme types described in the literature. Even though we do not achieve 100% accuracy on free text, the approach adopted already delivers good performance in the identification of simple Theme types.

The current limitations of the approach lie in two areas: firstly, multiple Themes are not covered yet; and secondly, complex sentences produce erroneous parses with the PCFG model of the Stanford parser. We will thus need to expand the rule set for Theme identification and find a way of working around the parser’s problems.
The observation of the discrepancies between the performance of the PCFG and the Factored parsers on clauses with different levels of complexity requires further testing in order to evaluate the possibilities of a combined application of the two parsers in the hope that this will deliver the most optimal parsing results for our Theme rule interpreter. Finally, an expanded rule set is going to be applied to more text from additional genres and more diverse registers and again evaluated in terms of performance. In terms of applications, we want to analyze patterns of thematic progression on the basis of a Theme-annotated corpus (see e.g., ?) as well as explore the possibilities of data mining for detecting differences and commonalities between register-diversified corpora on the basis of Theme information.

References


