# AUTOMATIC EXTRACTION OF TIME EXPRESSIONS AND REPRESENTATION OF TEMPORAL CONSTRAINTS

**Margus Treumuth**
University of Tartu (Estonia)

### Abstract

The paper describes a rule-based implementation of temporal information resolution in natural language. The temporal information resolution tool was implemented to work on Estonian texts and intended for integration with Estonian spoken language dialogue systems that act as interfaces to a database. Temporal information can often be a significant part of meaning communicated in dialogues. There are various kinds of dialogues where people negotiate dates and times. Therefore, the automatic extraction of temporal expressions in natural language is required in building dialogue systems where temporal constraints need to be enforced.

**Keywords**: Estonian natural language, dialogue systems, TimeML, TIMEX, deictic expressions, rule-based parser

## 1. Introduction

The paper describes a rule-based approach for processing temporal information in natural language. The extraction tool was implemented to work on Estonian texts and partially integrated into an Estonian spoken language dialogue system that is an interface to a theater information database (Treumuth, et al., 2006).

Temporal information can often be a significant part of meaning communicated in dialogues. There are various kinds of dialogues where people negotiate dates and times. Therefore, the automatic extraction of temporal expressions in natural language is required in building dialogue systems where temporal constraints need to be enforced. The time expression recognizer (e.g. as an annotation tool) could be a useful software tool among the other currently available Estonian language technology software tools.

## 2. Analysis

The analysis involved studying some dialogues that were held with a dialogue system. It turned out that the users of the dialogue system tend to query the database for intervals of time, rather than for a specific date. That is, instead of requesting information for a specific date as "January 11th", the users often tend to say "something in January". Therefore, much attention was paid to recognizing time intervals while implementing the tool.

The system can handle various constructions of time expressions where names of months and weekdays are used to represent an interval of time. Following are a few examples of some date expressions that are recognized by the temporal recognition system:

*pühapäeviti ja esmaspäeviti jaanuaris* - on Sundays and Mondays in January
*jaanuaris ja veebruaris* - in January and in February
*esmaspäeviti ja laupäeviti* - on Mondays and on Saturdays
*aprilli lõpus* - at the end of April
*juuni keskel* - in the middle of June
*oktoobri alguses* - in the beginning of October
*mais neljapäeviti* - on Thursdays in May

## 3. Implementation

The extraction tool was implemented as a standalone domain independent module, and was made available as a web-service, that can be plugged into dialogue systems with some minor adjustments.

### 3.1. Rules

The rule-based approach was chosen as in certain cases it can be worthwhile to spend some time and simply write down all the rules for a specific parsing task. The rules can be generated, thus simplifying the process, and kept in a text file outside of the program code. For a small enough domain the explicit rule-based grammar can be a much better choice than the statistical approach. I was not able to use the statistical approach as no TIMEX annotated corpus was available for Estonian language.

The grammar generates ca 1400 rules where regular expressions are mapped to corresponding SQL constraints as follows:

```
regular expression ==> SQL constraint
```

For example:

```
/ (oktoober|oktoobri)\S* laupäev\S* /U
==>
weekday(DATE) = 'laupäev' and DATE between 01.10.$YEAR and
last_day(01.10.$YEAR)
```

This rule would recognize expressions like *"oktoobris laupäeviti"*, *"oktoobri laupäevadel"* (in October on Saturdays). The formal representation given as the SQL constraint can be quite easily enforced on a relational database.

There are other ways to formally represent time expressions. The annotators often use TimeML and TIMEX standards. This could be useful to have this formal annotation as an intermediate step and transform to SQL from there on. Yet, going straight to SQL-like representation seemed to be more convenient at this point, considering the direct need to use the representation in SQL queries. Actually, the main idea and benefit of the current approach is the output of logical expressions that can be directly used in SQL queries.

The corresponding SQL constraints are integrated into a SQL query's template *WHERE clause* as follows:

```
<SELECT clause>
<FROM clause>
<WHERE clause
 [temporal constraints]
 [all other constraints]>
```

For example:

```
    SELECT title
    FROM performances
    WHERE
    weekday(DATE) = 'Saturday' and DATE between 01.10.2007
and last_day(01.10.2007)
    [all other constraints]
```

Upon execution of this query the dialogue system would return the performances that match the temporal constraint and the other domain specific constraints. These SQL-like constraints can easily be altered to suit the needs of a specific database engine (Oracle, MySQL, PostgreSQL, MS SQL). Also the functions *weekday* and *last_day* are available in most database engines or can easily be implemented.

There are few semantic ambiguities in Estonian temporal expressions. For example "august" (August) and "august" (from a hole). These are rare cases and can be handled without automatic sense disambiguation.

The anaphora resolution is not yet implemented in the extraction tool. The ambiguities and possible anaphora reference could be resolved in the dialogue management module by correction questions in correction sub-dialogues.

### 3.2. Input and output

The input to the extraction tool is a text in Estonian, e.g. an utterance from the user of a dialogue system.

The output of the system is:

1. the recognized time expression
2. the formal representation of the recognized time expression

For example:

**INPUT**: Ma tahaksin *veebruari teises pooles* teatrisse minna. (I would like to see a play in the second half of February)

**OUTPUT:**
RECOGNIZED: *veebruari lõpus* (in the end of February)

CONSTRAINT = DATE between 15.02.$YEAR and last_day(01.02.$YEAR)

The output can contain many sets of recognized expressions and constraints. The more specific ones are listed prior to less specific ones.

Also, notice that the input term and the output term differ slightly:

*"veebruari teises pooles"* - in the second half of February
vs.
*"veebruari lõpus"* - in the end of February

The output term is used by the dialogue system in generating the answer to the user and is a predefined term for each rule.

While the extraction tool is used to recognize time expressions and execute queries based on returned constraints, there is also need to provide input for the answer generation of the dialogue management module in a dialogue system, as the answer should also contain the recognized time expression in correct form (correct case in Estonian).

For that reason, the rules were extended by adding the recognized term into the rule as follows (in bold):

**oktoobris laupäeviti**

==>

/ (oktoober|oktoobri)\S* laupäev\S* /U

==>

weekday(DATE) = 'laupäev' and DATE between 01.10.$YEAR and last_day(01.10.$YEAR)

The recognized term, in correct form, can be used in generating an answer to the user by plugging it in a sentence. Assume a conversation:

<User>: Are there any performances in **October on Saturdays**?
<System>: Here are the plays that I found in **October on Saturdays** ...

The pattern can match multiple formats, yet the answer phrase can be fixed to one format, as the rules are built to support this approach.

### 3.3. Morphology

The Estonian Morphological Analyzer (Kaalep, 1997) was not used in generating the rules and was also not used in parsing the text. The inflections and agglutinations of Estonian date expressions are easily predictable and can be handled "manually". The morphological analyzer will be used as this work is continued. At this time the morphological analyzer is being used in the dialogue system that employs the temporal recognition module. The input to the temporal recognition module coming from a dialogue system is morphologically analyzed, providing lemmas or base forms, if no other forms yielded a recognition result.

### 3.4. Deictic Expressions

Temporal expressions in text vary from explicit references, e.g. *June 1, 1995*, to implicit references, e.g. *last summer*, to durations, e.g. *four years*, to sets, e.g. *every month*, and to event-anchored expressions, e.g. *a year after the earthquake*. (Hacioglu, et al., 2005)

Deictic expressions are expressions that refer to temporal aspect of an utterance and depends on the context in which they are used (Wiebe, et al., 1998). For example "*tomorrow*" depends on current date and is recognized as "*current date + 1 day*" (with respect to the conversation date).

The rules currently contain a non-terminal *$YEAR*, that is used to enforce dependencies to current date by avoiding looking in past dates. No other deictic expressions are represented in rules. The algorithm copes with deictic expressions in a separate parser. It can recognize patters like "*on weekends*", "*day after tomorrow*", "*today*", "*next Monday*" and so on.

### 3.5. Evaluation

Approximately 150 dialogues were analyzed and 10 user-tests were done to evaluate the parser, which resulted in a recall 53% and precision 79%.

This is just a preliminary evaluation, which shows that the parser still misses lot of expressions, yet the ones that are recognized are quite well normalized.

## 4. Future work

### 4.1. Constraint Relaxation

The constraint relaxation is implemented in a dialogue system that uses the temporal extraction tool, yet the rules for constraint relaxation are not defined in the temporal extraction tool but in the dialogue system. For example, the user might mention a date to a dialogue system that would result in *"not found"* response. Then it would be appropriate to relax this date constraint, as in the following dialogue.

<User>: Are there any performances on **Saturdays**?
<System>: No, yet I found one on this **Sunday** ...

Here we saw an example of a constraint relaxation where the original date constraint was relaxed by adding one day. This way the users of the system can receive some alternative choices, instead of plain *"not found"* responses.

The constraint relaxation properties can be held in the temporal extraction tool as long as they stay separate from the dialogue domain.

### 4.2. Correction Questions

When integrating the parser with a dialogue system, it would be useful to accept some additional input from the dialogue system in addition to the current utterance. For instance, knowing the dates that were recognized earlier in the current conversation would provide a way to accept corrections from a user, in case the user would like to clarify prior temporal expressions.

There are some problems with deictic expressions that can be solved by correction questions. For example, if user mentions the word "*weekend*" on Sunday evening, does the user mean next weekend or the current weekend.

The correction questions are not implemented in the rules, as they tend to be domain specific. The rules could be extended by adding correction questions and choices for corresponding answers, also as long as they stay separate from the dialogue domain.

## 5. Comparison with other approaches

The automatic time expression labeling for English and Chinese Text by Hacioglu (Hacioglu et al. 2005) uses a statistical time expression tagger. Yet, they have trained the system on a corpus that has been tagged by a rule-based time expression tagger.

Under my supervision a bachelor thesis was completed where a rule-based time expression tagger was built (precision of 93% and recall 71%). The tagger does not provide normalized output or any kind of formal representation of the date expression. It just places the recognized time expressions between <TIMEX> tags. This tagger could be used as a preprocessing step in generating the rules for the extraction tool to improve its set of rules. I have not considered using this tagger to build a tagged corpus and switching to statistical approach as the rule-based approach is easier to control and I don't have a corpus that would provide a considerable amount of temporal negotiation dialogues.

Wiebe (Wiebe et al. 1998) has addressed the anaphora resolution and has used an empirical approach to temporal reference resolution. They have adopted a recency-based focus model. That is, the most recent antecedent receives the highest score.

The anaphora resolution is not done in my system. I have decided to solve the ambiguities and possible anaphora reference in the dialogue management module by handling correction questions in correction sub-dialogues.

Berglund (Berglund 2004) has used a common annotation scheme - TimeML (Time Mark-up Language) and has built a rule-based custom tagger to annotate text with TimeML. He has used the tagger in a system that is used to detect and order time expressions. I have decided to skip the common annotation schemes in my output as an intermediate step, as it seemed more convenient for the QA task to produce output in logical expressions that are compatible with SQL.

Saquete (Saquete et al. 2006) presents a rule-based approach for the recognition and normalization of temporal expressions and their main point is that this way it was easy to port the system for different languages. They have used a Spanish system to create the same system for English and Italian through the automatic translation of the temporal expressions in rules. I agree with this advantage and can confirm that the porting from Estonian to English and vice versa would mostly consist of automatically translating the rules.

## 6. Conclusion

The paper has described an implementation of an automated extraction tool for processing temporal information in Estonian natural language. This rule-based approach can be used for other languages (English). The main idea and benefit of current approach is the output of logical expressions that can be used in SQL queries.

The rules can be improved in using constraint relaxation options and predefined question-answer sets for correction sub-dialogues.

Four other approaches to extracting temporal information are discussed in the paper to provide some comparison.

## 7. References

Anders Berglund. Extracting Temporal Information and Ordering Events for Swedish. Master’s thesis report. 2004.

Kadri Hacioglu, Ying Chen, and Ben Douglas. Automatic Time Expression Labeling for English and Chinese Text. *In Proceedings of CICLing-2005*, pages 348-359; Springer-Verlag, Lecture Notes in Computer Science, Vol. 3406. 2005.

Heiki-Jaan Kaalep. An Estonian Morphological Analyser and the Impact of a Corpus on Its Devel-opment. *Computers and the Humanities* 31: 115-133. 1997.

E. Saquete, P. Marinez-Barco, R. Munoz. Multilingual Extension of a Temporal Expression Normalizer using Annotated Corpora, Cross-Language Knowledge Induction Workshop, 2006.

Margus Treumuth, Tanel Alumäe, Einar Meister. A Natural Language Interface to a Theater Information Database. *Proceedings of the 5th Slovenian and 1st International Language Technologies Conference 2006 (IS-LTC 2006)*, 27-30. 2006.

J. M. Wiebe, T. P. O'Hara, T. Ohrstrom-Sandgren, and K. J. McKeever. (1998). An Empirical Approach to Temporal Reference Resolution. *Journal of Artificial Intelligence Research*, 9, 247-293. 1998.

MARGUS TREUMUTH is a PhD student at the University of Tartu. He received his MSc (Computer Science) at the University of Tartu, dealing with dialogue systems interacting with a user in Estonian. His doctoral study also focuses on dialogue systems. E-mail: treumuth@ut.ee.