Anchoring Temporal Expressions in Scheduling-related Emails

Benjamin Han, Donna Gates, and Lori Levin

Language Technologies Institute, Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
{benhdj|dmg|lsl}@cs.cmu.edu

Abstract. In this paper we adopt a constraint-based representation of time, *Time Calculus* (TC), for anchoring temporal expressions in a novel genre, *emails*. Email is sufficiently different from most studied genres - newswire texts, and its highly under-specified nature fits well with our representation. The evaluation of our anchoring system shows that it performs significantly better than the baseline, and the result compares favorably with some of the closest related work.

Keywords. Temporal information processing, computational semantics, knowledge representation, constraint solving

1 Introduction

Recent years have seen increasing research on extracting and using temporal information in natural language applications. Several works have addressed the problems of representing temporal information manifested in natural language [1,2,3], extracting and/or anchoring temporal and event related expressions [4,5,6,7,8], and ordering events [9]. The vast majority of these works (except for [4]), however, have focused on capturing temporal information in the *newswire* domain.

In this paper we set out to address a slightly different problem: one of anchoring temporal expressions in *scheduling-related emails*. This is motivated by the need of such information in building personal agents capable of scheduling meetings among different users. The idea is to have the agents read and understand scheduling-related emails, and engage in the followup negotiation processes (possibly with help from humans). An accurate understanding of temporal expressions therefore becomes one of the most important steps.

The difference in target genres (newswire vs. emails) is by no means a small one. In Sec. 2 we shall argue that the highly under-specified nature of temporal expressions in emails calls for a different approach for representing and anchoring them. In this work we adopted a constraint-based representation *Time Calculus* [10,11], to be described in Sec. 3, to give a compact representation for each temporal expression. The representation is capable of capturing the effect of granularity change in temporal expressions, and it can incorporate a

Dagstuhl Seminar Proceedings 05151
Annotating, Extracting and Reasoning about Time and Events
http://drops.dagstuhl.de/opus/volltexte/2005/316

temporal focus explicitly to facilitate a more modular approach in modeling the phenomenon of focus shifting.

We will then describe our architecture for accomplishing this task in Sec. 4. Two slightly different approaches were attempted: one uses temporal expressions and their ordering only, and the other exploits the information coming from verbal tenses. The experiments and the results are then reported in Sec. 5. Finally Sec. 6 concludes this paper and outlines the future work.

2 Temporal Expressions in Scheduling-related Emails

The extent of temporal expressions considered in this paper includes most of the expressions using temporal terms such as *2005*, *summer*, *evening*, *1:30pm*, *tomorrow*, etc. These expressions can be categorized into the following types:

- **Explicit**: these expressions can be uniquely *anchored*, i.e., positioned on a timeline. E.g., *June 2005*, *1998 Summer*, etc.
- **Deictic**: these expressions form a specific relation with the datestamp of an email. E.g., *tomorrow*, *last year*, *two weeks from today.*
- **Relative**: these include the other expressions that form a specific relation with a *temporal focus*, i.e., the implicit time central to the discussion. E.g., *over the summer*, *on Wendesday*, etc.

It should be noted that this does not represent an exhaustive list of all temporal expressions - notable omissions include duration expressions[1] (e.g., *for three hours*), recurrence expressions (e.g., *every Tuesday*) and rate expressions (e.g., *twice on Wednesday*). They cannot be straightforwardly placed on a timeline and therefore are less useful for our applications[2].

The email corpora used in our development and testing were collected from MBA students of Carnegie Mellon University over the year 1997 and 1998. The 277 students, organized in approximately 50 teams of 4 to 6 members, were participating in a 14-week course and running simulated companies in a variety of market scenarios [12]. The original dataset, the CSpace email corpus, contains approximately 15,000 emails. We manually picked 1,196 emails that are related to scheduling - these include scheduling meetings, presentations, or general planning for the groups. The emails are then divided into five sets - **email1** to **email5**, and only three of them are used in this work: **email1** and **email2** were used for development, and **email5** was used for testing. Table 1 shows some basic statistics of these three datasets[3], and an edited sample email is shown in Fig. 1 (names altered).

[1] These are different from interval expressions, where an endpoint is explicitly specified; e.g., *for the next three hours.*

[2] We do include the duration expressions in a developmental version of our system, but it has not been evaluated. The application for now is also geared toward one-time meetings, hence the omission of recurrence expressions.

[3] The percentages in some rows do not add up to 100% because some expressions like coordination can be classified into more than one type.

Table 1. Basic statistics of **email1**, **email2** and **email5**

	# of emails	**# of timex**	**# of explicit timex**	**# of deictic timex**	**# of relative timex**
email1	253	300	3 (1%)	139 (46.33%)	158 (52.67%)
email2	266	297	6 (2.02%)	101 (34.01%)	192 (64.65%)
email5	126	184	5 (2.72%)	88 (47.83%)	92 (50%)

Date: **Thu, 28 Aug 1997 12:27:10 -0500**

Everyone,

Thanks for working hard **last night**. I appreciate everything you are doing (*omitted...*)

Sean & Mark please work together **this weekend** to increase the # of products in our library and conduct market research to understand consumer preferences better. Two heads are better & quicker than one.

I've looked at **last years** plans and we should try to extract ideas from those **tonight** rather than re invent the wheel. (*omitted...*)

John & Mark, I'll see you **at 6:00 -6:30** in the library....

Fig. 1. An edited email

The most apparent difference comparing these emails to newswire texts is in the percentage of explicit expressions occurring in the two different genres. In [9] it was reported that the proportion of such expressions is about 25% in the newswire corpus they used[4]. In contrast, explicit expressions only account for at most 3% in the three email datasets. This is not surprising given that people tend to type under-specified expressions in emails for economic reasons. Another thing to note is that of all of the non-explicit expressions, deictic expressions turn out to be fewer than relative expressions. Since deictic expressions can be anchored without tracking the temporal focus over a discourse and therefore can be dealt with in a fairly straightforward way, we may view the combined percentages of explicit and deictic expressions as a somewhat generous baseline performance of any anchoring system[5]. Based on Table 1 this baseline is at most around 50%.

Other differences between emails and newswire texts are that mispelling occurs more often, and people tend to be more creative when they compose short messages, e.g., using bullet lists, abbreviations, etc. Mispelling is not the only type of human errors, however. For example, in Fig. 2 the sender obviously associated a wrong day of week with the expression *tomorrow*. Also if a message

[4] Using the North American News Corpus.

[5] This is a bit generous since solving simple calendric arithmetics such as anchoring *last summer* requires a non-trivial modeling of human calendars; see Sec. 3.

Date: **Mon 15 Sep 1997 12:20:11 -0500**

(*omitted...*)As for the labor proposal, we should have it first thing **tomorrow** (**Monday**) **morning** to review.(*omitted...*)

Fig. 2. An email containing a human error

was sent around midnight, it is more often for the sender to use expressions such as *tomorrow* incorrectly[6]. Overall it is very difficult to recover from this type of errors.

3 Time Calculus for Natural Language

This section provides a concise review of Time Calculus (TC), a formal language we use to represent temporal expressions in this work. Readers are referred to [10,11,13] for more detail.

TC is a typed language consists of a set of temporal entities defined in a *calendar constraint system*, and a set of operators and relations. The *intensional* meaning of an expression is encoded as a formula in this language. TC has the following features:

1. The constraint-based modeling of human calendars is extensible, and can be used to derive missing information in an expression via constraint solving.
2. Granularity change is handled transparently via type coercion.
3. Temporal references can be explicitly introduced, and are useful for modeling deictic expressions (via variable `now`) and relative expressions (via temporal focus variable '_').
4. Many intuitions of temporal arithmetics are encapsulated in various operators of the formalism.

A calendar constraint system is basically a constraint graph with partial ordering. The nodes represent temporal *units* (e.g., `year`, `month`, `day`, etc) and can take on fully ordered sets of *values*. The edges (ordering) specify the *measurement relation* (e.g., `month` is measured by `day`), and can be assigned with constraints (e.g., February can't have more than 29 days). A *spine* of a calendar system is defined to be a representative path from a minimal unit to a maximal unit (e.g., the spine of the Gregorian calendar is $\texttt{year} \rightarrow \texttt{month} \rightarrow \ldots \rightarrow \texttt{sec}$). A temporal expression can be viewed as giving a partial set of assignments to the constraint system, and conventional methods such as AC-3 [14] can be used to solve for its consistency. Anchoring an expression thus can be achieved by solving the constraint satisfaction problem and reading off the assignments along the spine.

A TC formula can represent a point (type *coordinate*), a set of points (type *enumeration*), or a duration (type *quantity*). Examples are $\{1987_{\texttt{year}}, \texttt{sep}, 9_{\texttt{day}}\}$

[6] E.g., using *tomorrow* to mean *today* when the message was sent at 1 AM.

for *September 9, 1987* (coordinate), [{wed},{fri}] for *Wednesday and Friday* (enumeration) and $|1_{\texttt{hour}},30_{\texttt{min}}|$ for *1 hour and 30 minutes* (quantity). More complex expressions can be represented by using various operators, relations and temporal references; e.g., {now+$|1_{\texttt{day}}|$} for *tomorrow*, {$|1_{\texttt{mon}}|$@{**bi**_}} for *the upcoming Monday* (or the first coming Monday in the future), $|< 1_{\texttt{hour}}|$ for *less than one hour*, and [{wed}: {fri}] for *Wednesday to Friday*.

In TC the semantics of the operators encapsulates many intuitions we have with respect to temporal expressions. An example is granularity change: if the current temporal focus is Friday, June 17, 2005 at 12:00pm, the formula {_+$|1_{\texttt{wed}}|$} (*next Monday*) is evaluated to be June 20, 2005 without anything specified for *hour* and *minute*. This is because in shifting a time point the quantity to be shifted usually encodes the target granularity. Granularity change is handled transparently via type coercion, which in turn is realized by constraint solving in the underlying calendar system.

4 Anchoring Emails

In this section we describe the system used in the experiments. We first conduct simple normalization procedure to mark various parts of a message (header, datestamp, body, etc.). The normalized emails are then sent to an information extraction system MinorThird [15] for *pre-tagging*; i.e., temporal expressions are marked up by the system via a set of simple rules. Since the focus of this paper is on investigating the effectiveness of our anchoring system, we manually go through the discovered expressions to fix any mistake.

Internally our anchoring system consists of three major components (see Fig. 3). Every temporal expression in an input email is first sent to a semantic CFG parser called SOUP [16] for parsing. The decision of using SOUP is a pragmatic one - SOUP has been used in many speech translation projects in CMU, and has a fairly well-developed set of time-related grammars. The resulting parse tree is then converted into a *feature structure* and sent to GenKit [17] for the purpose of building a corresponding TC formula. GenKit is a unification-based natural language generation system that has also been used in many machine translation projects in CMU. Although it is not developed for the purpose of producing a structured representation, the formalism allows a grammar developer to build a meaning representation in a fairly compositional way.

The last component in the anchoring system is the anchorer itself: it takes a datestamp, a list of TC formulae obtained from the corresponding temporal expressions, and optionally the verbal tenses associated with them, and produces a list of anchored time strings, which can take on two possible formats: for a coordinate it is `YYYY-MM-DD hh:mm:ss +/-zzzz` (Y: year, M: month, D: day, h: hour, m:minute, s: second, z: timezone offset); for an interval it is `[start : end]` where `start` and `end` are two coordinates. Question marks '`?`' may occur for any un-specified information. For example the anchored time string of {now+$|0_{\texttt{weekend}}|$} (*this weekend*) is shown in Fig. 3. The design of this format is

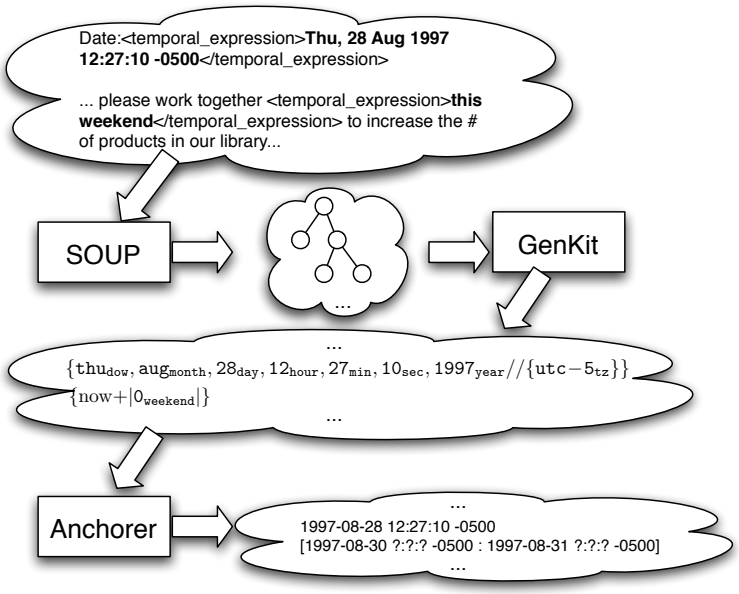

Fig. 3. Anchoring system

highly application oriented and can be altered without much of the change to the rest of our system.

As described in Sec. 2, relative expressions outnumber the other two types of temporal expressions, and anchoring them calls for the use of other information such as temporal focus and verbal tenses. The rest of this section describes these operations in detail.

4.1 Tracking temporal focus

We use a simple *recency-based* model to track temporal focus:

1. Set $f = f_0$, where f is the current temporal focus and f_0 is the anchored time of the datestamp.
2. For each temporal expression (in the order in which they occur in an email):
   (a) If the expression denotes an enumeration (e.g., an interval), take each element of the expression and start from 2 recursively.
   (b) Rewrite any occurrence of '_' (temporal focus) in the formula with f and solve for its consistency (anchoring).
   (c) If the result is still not anchored, try to merge f with it (see Sec. 4.2).
   (d) If the result is anchored, set f to it.

Table 2. Anchoring example for the email in Fig. 1

TC formula	Temporal focus (f)	Anchored time string
{now−|$1_{\texttt{night}}$|}	1997-08-28 12:27:10 -0500	[1997-08-27 18:?:? -0500 : 1997-08-27 23:?:? -0500]
{now+|$0_{\texttt{weekend}}$|}	1997-08-27 23:?:? -0500	[1997-08-30 ?:?:? -0500 : 1997-08-31 ?:?:? -0500]
{now−|$1_{\texttt{year}}$|}	1997-08-31 ?:?:? -0500	1996-?-? ?:?:? -0500
{now+|$0_{\texttt{night}}$|}	1996-?-? ?:?:? -0500	[1997-08-28 18:?:? -0500 : 1997-08-28 23:?:? -0500]
[{$18_{\texttt{hour}}$,$00_{\texttt{min}}$}: {$18_{\texttt{hour}}$,$30_{\texttt{min}}$}]	1997-08-28 23:?:? -0500	[1997-08-28 18:00:00 -0500 : 1997-08-28 18:30:00 -0500]

Note that we do not allow an unanchored expression to become a temporal focus. This is because in expressions such as *until tomorrow*, the starting point is effectively unanchorable and should not be used as a focus. Table 2 illustrates the process by listing the anchoring results for the email shown in Fig. 1[7].

4.2 Merging temporal focus

Many of the relative expressions in our email corpora do not explicitly signal a relationship with a temporal focus; e.g., in the expression *at 6:00 - 6:30* in Fig. 1, other than the fact that we do not know if the hours should be interpreted as AM or PM, the relationship with the temporal focus is not explicitly specified either (it could be a point before or after the temporal focus). For this kind of expressions, i.e., expressions that cannot be anchored, we use the following method to "merge" a temporal focus into them (step 2c in Sec. 4.1): for each temporal unit in the spine of our calendar system (from the minimal unit to the maximal unit), if its value is not specified at all, assign the value of the corresponding unit from a temporal focus to it. For example (Fig. 1), the date *August 28, 1998* is merged with the under-specified expression to give the last anchored time string.

It should be noted that the hour ambiguity (whether it's AM or PM) is not addressed in this operation. The example shown in Fig. 1 works because the system makes certain assumptions about the possible times for meetings. Clearly there will be cases where mistakes could occur. In the next sub-section we use verbal tenses to address this problem.

4.3 Using verbal tenses

In the email shown in Fig. 1 we know the expression *at 6:00 - 6:30* means 18:00 - 18:30 because the expression is associated with a future tense (*I'll see you*

[7] Several assumptions have been made for temporal units like `night`: e.g., `night` is assumed to range from 6pm to 11pm.

Table 3. Results over the development and the testing datasets

email1	50% (eval)		74% (dev)	85% (dev)	
email2		61% (eval)	66% (dev)		
email5					77.3% (without tense) 77.83% (with tense)

...), and the datestamp is already at noon. This motivates the following simple method to disambiguate hours:

1. If the tense associated with a coordinate `c` is in the past, we rewrite it into $\{-|1_{\texttt{c}}|@\{\textbf{b\_}\}\}$; i.e., the nearest coordinate `c` *before* the temporal focus.
2. Otherwise we rewrite it into $\{|1_{\texttt{c}}|@\{\textbf{bi\_}\}\}$; i.e., the nearest coordinate `c` *after* the temporal focus.

For example, the coordinate for the expression *6:00* in Fig. 1 should be represented as $\{6|18_{\texttt{hour}},00_{\texttt{min}}\}$, and it can be rewritten into $\{|1_{\{6|18_{\texttt{hour}},00_{\texttt{min}}\}}|@\{\textbf{bi\_}\}\}$ to give the correct interpretation 18:00.

In the experiments reported here we only used verbal tenses to disambiguate hours. Although it should be equally applicable to expressions such as *on Wednesday*, from our observation this kind of expressions (days of week) almost always refer to an upcoming date. Hence we decided to always rewrite them into formulae like $\{|1_{\texttt{wed}}|@\{\textbf{bi\_}\}\}$.

The verbal tenses used here are obtained from the output of a part-of-speech tagger. Since we only parse temporal expressions, the verb associated with an expression is discovered by scanning its local context: the first verb found in the left/right context is considered to be the right one.

5 Experiments and Results

We developed and tested our system over a time period of approximately three months, and the results are reported in Table 3. The percentages shown in the table are *accuracies*, i.e., the number of correctly anchored expressions over the total number of temporal expressions over a dataset. In the beginning we did not implement any focus tracking mechanism (i.e., always use a datestamp as the focus), and did not use any tense information. The baseline was established over the **email1** dataset (50%). The result confirms our estimate given in Sec. 2. Next we developed over **email1**, added the recency-based focus model, and tested on **email2** (61%). We then iteratively developed and tested over these two datasets, and finally reached 85% accuracy over the **email1** dataset. The resulted system was then tested on the unseen dataset **email5**: at this point we introduced tense information and therefore obtained two performance numbers (77.3% and 77.83%).

Table 4. Error categories of the last two experiments over **email1** and **email5**

	Accuracy	**Syntax-semantics errors**	**Human errors**	**Anchoring errors**
email1	85%	11%	1.67%	2.33%
email5 (without tense)	77.3%	13.51%	2.16%	7.03%
email5 (with tense)	77.83%	13.51%	2.16%	6.5%

Table 4 reports the error categories of the two most recent experiments over **email1** and **email5**. The syntax-semantics errors are mistakes made at parsing temporal expressions using SOUP and building TC formulae using GenKit, the human errors are described in Sec. 2, and the rest are the anchoring errors. The accuracy numbers are all compared favorably to the baseline (50%), and if we ignore the human errors, the accuracy of the anchoring system over the unseen dataset is about 80%. To put this performance in perspective, in [4] a similar task was performed over transcribed scheduling-related phone conversations. They reported an average accuracy 80.9% over the CMU test set and 68.9% over the NMSU test set. Although strictly speaking the two works cannot be compared due to differences in the nature of the corpora (transcription vs. typing), it represents a closer match compared to the other works done on newswire genre. It should be noted that [4] also adopted a recency-based focus model.

Another thing to note is that the big jump in the anchoring error rate between **email1** (dev set) and **email5** (test set) is due to the accumulated errors introduced by false temporal foci. The improvement brought by the use of verbal tenses is small but noticeable. Had we used the tense information on **email1**, we could gain another 1.3% in the anchoring accuracy.

6 Conclusion and Future Work

In this paper we have adopted a constraint-based representation of time, *Time Calculus* (TC), to accomplish the task of anchoring temporal expressions in a novel genre, *emails*. We believe that the genre is sufficiently different from newswire texts, and its highly under-specified nature fits well with a constraint-based modeling of human calendars. TC also allows for an explicit representation of *temporal focus*, and many of our intuitions about *granularity change* and *temporal arithematics* are encapsulated in its type system and operators. The performance of our anchoring system is significantly better than the baseline, and compares favorably with some of the closest related work.

In the future we will re-examine our focus tracking mechanism (being the most significant source of errors), and possibly treat it as a classification problem (similar to [9]). The focus merging and the use of tense also needs to be investigated more fully and possibly will become part of a separate discourse

module. We would also like to expand our coverage of temporal expressions to include other types of expressions such as recurrence expressions. Finally we would like to take advantage of the additional information provided by email threading and quoting.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Defense Advanced Research Projects Agency (DARPA), or the Department of Interior-National Business Center (DOI-NBC).

References

1. Setzer, A.: Temporal Information in Newswire Articles: an Annotation Scheme and Corpus Study. PhD thesis, University of Sheffield (2001)
2. Saurí, R., Littman, J., Gaizauskas, R., Setzer, A., Pustejovsky, J.: TimeML Annotation Guidelines, Version 1.1. (2004)
3. Hobbs, J.R., Pan, F.: An ontology of time for the semantic web. TALIP Special Issue on Spatial and Temporal Information Processing **3** (2004) 66–85
4. Wiebe, J.M., O'Hara, T.P., Ohrstrom-Sandgren, T., McKeever, K.J.: An Empirical Approach to Temporal Reference Resolution. Journal of Artificial Intelligence Research **9** (1998) 247–293
5. Mani, I., Wilson, G.: Robust Temporal Processing of News. In: Proceedings of ACL-2000. (2000)
6. Schilder, F., Habel, C.: From Temporal Expressions To Temporal Information: Semantic Tagging Of News Messages. In: Proceedings of ACL-2001: Workshop on Temporal and Spatial Information Processing, Toulouse, France (2001)
7. Vazov, N.: A System for Extraction of Temporal Expressions from French Texts Based on Syntactic and Semantic Constraints. In: Proceedings of ACL-2001: Workshop on Temporal and Spatial Information Processing, Toulouse, France (2001)
8. Filatova, E., Hovy, E.: Assigning Time-Stamps To Event-Clauses. In: Proceedings of ACL-2001: Workshop on Temporal and Spatial Information Processing, Toulouse, France (2001)
9. Mani, I., Schiffman, B., Zhang, J.: Inferring Temporal Ordering of Events in News. In: Proceedings of the Human Language Technology Conference (HLT-NAACL'03). (2003)
10. Han, B., Kohlhase, M.: A Time Calculus for Natural Language. In: The 4th Workshop on Inference in Computational Semantics, Nancy, France (2003)
11. Han, B., Lavie, A.: A Framework for Resolution of Time in Natural Language. TALIP Special Issue on Spatial and Temporal Information Processing **3** (2004) 11–32
12. Kraut, R.E., Fussell, S.R., Lerch, F.J., Espinosa, A.: Coordination in teams: Evidence from a simulated management game. Journal of Organizational Behavior **(Under review)** (2005)

13. Han, B.: Time Calculus for Natural Language - Tagging Guidelines. Unpublished draft, Language Technologies Institute, Carnegie Mellon University (2005)
14. Mackworth, A.K.: Consistency in networks of relations. Artificial Intelligence **8** (1977) 99–118
15. Cohen, W.: Methods for identifying names and ontological relations in text using heuristics for inducing regularities from data, http://minorthird.sourceforge.net (2004)
16. Gavaldà, M.: Soup: A Parser for Real-world Spontaneous Speech. In: Proceedings of the 6th International Workshop on Parsing Technologies (IWPT-2000), Trento, Italy (2000)
17. Han, B., Lavie, A.: UKernel: A Unification Kernel. Technical Report CMU-LTI-03-177, Language Technologies Institute, Carnegie Mellon University (2004)