TimeML and Turkish Temporal Logic

Sadi Evren SEKER1 and Banu DIRI2
[1,2]Department of Computer Engineering, Yildiz Technical University, Istanbul , Turkey

Abstract *Turkish is one of the widely used and relatively difficult natural language for machine processing. One of the challenges in Turkish is the temporal logic and processing the time of events.*
For the Latin family natural languages, there are quite successful solutions like TimeML which is built on the Reichenbach tense analysis and Allen's temporal logic. Unfortunately, there is no previous work on Turkish languages up until now.
This paper covers the basic temporal models of Reichenbach and Allen and then proceeds to the improvement of these models to cover temporal logic behind Turkish natural language.
In order to test the success of this study, we have also created a corpus from child stories and tested the success of new implementation.

Keywords: TimeML, Reichenbach Temporal Logic, Allens Interval Logic, NLP, Event Ordering

1. Time Tagging

Time tagging which can be interpreted as extraction of temporal semantic from natural language texts is mainly used on almost all of the natural language research areas like question answering, text summarization or visualization of the texts[1][2].

A natural language text contains semantic value from many different categories. For example the location or personal information or a know-how can be carried through natural language sentences. Besides of all these information, all of the natural language sentences should contain a temporal logic since all the verbs are strongly connected to time. For example a sentence without a location information is possible but all sentences should contain a time being.

Time tagging is a technique for marking the temporal information of events, time expressions or the relations of events on the time line. Although the linearity of the time is an open discussion, the time tagging techniques are built over only linear temporal logics. For example the Allen's interval logic [3] or Reichenbach's temporal logic [4] are two samples for the representation of time by linear.

After the correct tagging of a natural language text, the implementation of some automat codes is possible to process the event ordering or question answering. The tagging can be done by two possible ways. For mature NLP languages which mostly solved the morphological and syntactic parsing problems, it is possible to implement an autonomous software to tag the temporal information on the NLP. On the other hand for the languages still under massively development of NLP which do not have a satisfactory success on morphological and syntactic levels, the only way of tagging the temporal information is manual.

Time tagging is still important for these languages to show a target for representation of semantic after the syntactic studies and prepare some tools after an achievement on these levels. For example in Turkish there is still no satisfactory syntactic and morphological tools to extract the semantic and also temporal logic in Turkish is different than Indo-European languages in some ways.

2. Reichenbach Temporal Logic

Reichenbach temporal logic is built on simple three temporal anchors.:

- Speech time (symbolized by S)
- Reference time (symbolized by R)
- Event time (symbolized by E)

Most of his study was focused on the natural languages. So he has formulated the order of these times.

For example a sentence like "I read the book" can be formalized as $R=E<S$ on the other hand a sentence like "I have read the book" can be formalized as $E<R=S$. Please note that on the former model, event is before the speech time and the speech is referring to the event time, so the event and reference times are equal and smaller than speech time on the model. For the latter example, again the event is before the speech time and but the speech is referring to the current time so the speech time and reference times are equal and greater than the event time.

Reichenbach has named these possibilities by using Anterior, Simple and Posterior aspects and Past, Present and Future tenses. So in English or in any natural language there can only be 9 possible meaningful time in the opinion of Reichenbach.

Below table covers these possibilities and samples for each of the case:

Table 1. All possible 13 permutation of Reichenbach temporal logic and their English tense/aspect and a sample for each case.

Permu-tation	Reichenbach Tense Name	English Tense	Sample
E<R<S	Anterior past	Past perfect	I had slept
E=R<S	Simple past	Simple past	I slept
R<E<S			
R<S=E	Posterior past		I would sleep
R<S<E			
E<S=R	Anterior present	Present perfect	I have slept
S= R=E	Simple present	Simple present	I sleep
S=R<E	Posterior present	Simple future	I will sleep
S<E<R			
S=E<R	Anterior future	Future perfect	I will have slept
E<S<R			
S<R=E	Simple future	Simple future	I will sleep
S<R<E	Posterior future		I shall be going to sleep

Please note that in the table 1, blank lines represents the meaningless cases of the permutation for English.

3. Allen's Interval Logic

Allen's interval logic (AIL) or temporal logic (ATL) deals with orders of the events. The representation of event orders like "event A is before event B" or "event A is at the same time with event B" are the operators of this logic.

For example in an example sentence like "John ate an apple at the table after he has entered the room" we have events "eat" and "enter" also there are hidden event which John goes to the table and takes the apple in order to eat an apple from the table after he has entered the room. Figure 1 holds the order of events in this example.

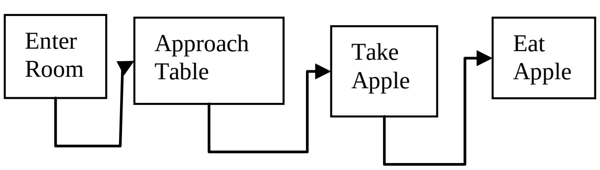

Figure 1. Sequence of events on sample sentence

If the states of the events are considered, we know John was outside of the room, before he has entered the room, also he was away from the table before he approaches table and he has no apples before taking apple.

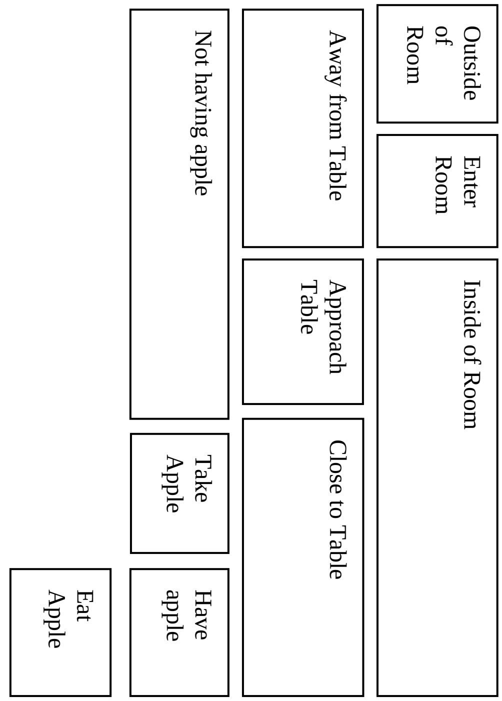

Figure 2. Event and states diagram

So from above sample sentence it is possible to conclude John had an apple when he is inside the room or John has had no apple while he was away from the table or while he was outside of the room.

Above sample can be modeled by using AIL. Let's say entering room (ER) requires to be outside of the room (OR) and after entering room the state is inside the room (IR) and similarly approaching table (AT) changes state of being away from table (SAT) to state of being close to table (SCT), taking apple (TA) is a transformation of state from not having apple (NHA) to having apple (HA). All these states are pre-requirements in the case of eating apple (EA).

Above sentence can be modeled in Allen Temporal Logic as below formulation:

Meets(OR,ER) ∧ Meets(ER,IR) ∧ During (ER,SAT) ∧ During (AT,IR) ∧ Meets(SAT,AT) ∧ Meets(AT,SCT) ∧ During(AT,NHA) ∧ During(TA,IR) ∧ During (TA,SCT) ∧ Meets(NHA,TA) ∧ Meets(TA,HA) ∧ During(EA,HA) ∧ During (EA,CT) ∧ During(EA,IR) ∧ Meets(TA,EA)

Above formulation demonstrates all the temporal states and events on the sample sentence. On the other hand a reader can interpret the above sentence and can add more states which can still be modeled by AIL. For example if John does the above order of events when he is hungry than this state can be added to the model of AIL. For this case the model would be:

Occurs (hungry, NHA) ∧ Holds (hungry, TA) ∧ Meets (hungry, EA)

Below list holds the possible operators of Allen Interval Logic:

- Before (x,y) or After (y,x)
- Overlaps (x,y) or Overlapped (y,x)
- Meets(x,y) or MetBy(y,x)
- Contains(x,y) or During (x,y)
- Starts(x,y) or StartedBy(y,x)
- Ends(x,y) or EndedBy(y,x)
- Equals(x,y)

4. Turkish Temporal Logic

Turkish [5] is a member of Ural-Altaic Language Family. This section analyses Turkish from the language perspective and shows important aspects of it. Turkish is characterized by certain morphophonemic, morphotactic, and syntactic features which are vowel harmony, agglutination of all-suffixing morphemes, free order of constituents, and head-final structure of phrases.

Turkish language uses Latin characters and the success of NLP studies on morphological and syntactic levels are still very low than the success rates of Indo-European languages such as English. Because of this fact, most of the researchers working on NLP are still far away of concentrating on formal semantic level studies. A part of this study is mainly focused on Turkish temporal logic, which is mainly concentrated on the semantic representation of Turkish natural language texts and will show the researchers an aim after achieving a satisfactory success on morphological and syntactic levels of Turkish.

Unfortunately Allen's temporal logic is not sufficient for representation of the Turkish temporal logic. Below cases states the temporal logic in Turkish where ATL is insufficient to represent.

4.1 Positive / Negative Terms in Turkish

These terms in Turkish represents a continuous event by using two verbs with opposite meanings. For example in English a single verb like "blink" is represented in Turkish with two separate verbs "yanıp sönmek" (to light and to fade) which term can also be represented by "to flash" or "to twinkle" in English which all are single verbs.

Another example is the translation of term "pacing up and down" or "pace back and forth" in Turkish. Again this term is represented by two separate verbs "gelip gitmek" (to come and to go). Or another example "restart" in Turkish is "kapatip acmak" (to close and to open).

Allen's temporal logic is a linear logic which is suitable for representing events in a linear manner. Unfortunately temporal logic behind Turkish natural language is not exactly linear. Although there are some studies of modeling time in a non-linear domain[6] , TimeML has been implemented linearly by using ATL. For example let's try to represent below Turkish sentence in ATL.

"The life signal on the safety buoy was blinking while the divers under water."

Above English sentence can easily be represented in ATL. The representation is in Figure 3.

Since we have no idea about the starting time of blinking of the life signal, either A-C or B-C or any time combination of "diving" and "start blinking" between these times is considered as correct from the above input sentence.

The ATL representation of the certain part of the above case would be as below:

Meets(SB,DUW) ∧ During (DUW,LSB) ∧ During (DOW,LSNB)

where , sb:"signal blinking", "duw: divers under water", "lsb: life signal blinking", "dow: divers out water", "lsnb: life signal is not blinking"

In the Turkish translation of above sentence, the representation would be as in Figure 4.

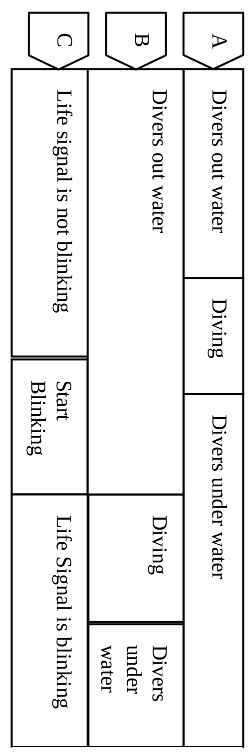

Figure 3. Event and states diagram

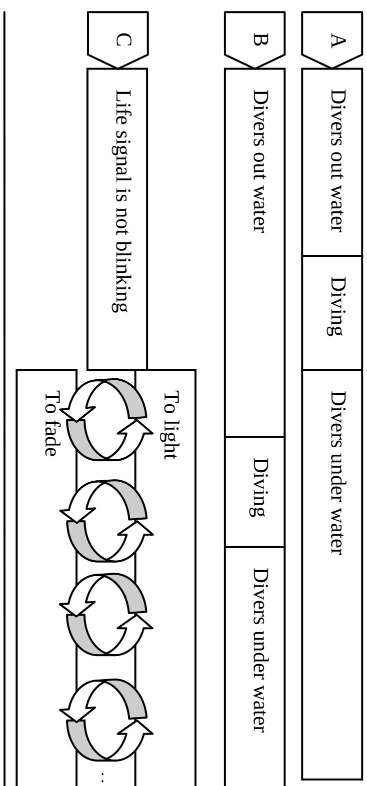

Figure 4. . Cyclic sequences in Turkic languages

In the temporal logic behind Turkish natural language states that even the event starts with lighting or fading, these two events follows each other and keeps going while the divers under the water.

This logic cannot be stated in ATL.

4.2 ATL and TimeML

The connection type of TimeML in the link layer is inspired from Allen's temporal logic [8]. And unfortunately does not support the full Turkish temporal logic explained in the previous section.

The link between events on TimeML are represented by using the TLink tags and the Backus-Naur Form (BNF) of TLink is quoted below:

```
relType ::= 'BEFORE' | 'AFTER' |
'INCLUDES' | 'IS_INCLUDED' |
'DURING' | 'DURING_INV' | 'SIMULTANEOUS'
| 'IAFTER' | 'IBEFORE' |
'IDENTITY' | 'BEGINS' | 'ENDS' |
'BEGUN_BY' | 'ENDED_BY'
```

Please note that the possibilities for "relType" symbol above are inspired from ATL and none of the above alternatives supports the Turkish temporal case explained in previous section. To clarify the above alternatives their explanations and samples are quoted below:

Table 2. ATL relation types.

relType	Comment	Sample
BEFORE	An event finishes before another starts	He came and seen the label.
AFTER	Reverse form of BEFORE and can substitute it.	
INCLUDES	A link to temporal expression or event which includes event.	John arrived in Boston last Thursday.
IS_INCLUDED	Reverse form of INCLUDES.	
DURING	A link to temporal expression or event while the event is on progress.	James was CTO for two years.
DURING_INV	Reverse form of DURING.	
SIMULTANEOUS	Two events at the same time.	
IAFTER	Immediately after	All passengers died when the plane crashed to the mountain.
IBEFORE	Reverse form of IAFTER.	
IDENTITY	Repeat of same verb.	He drove to Boston. While his drive he ate a donut.
BEGINS	A link to temporal expression or event which begins the event.	John was teaching since 1980.
ENDS	A link to temporal expression or event which ends the event.	John was in gym until 7.00pm.
BEGUN_BY	Reverse notation of BEGINS.	
ENDED_BY	Reverse notation of ENDS.	

As a solution, we suggest adding a new relation type named "CYCLES" to above table and the BNF which states that the cycle between two events continuously.

4.3 Reichenbach modeling and TimeML of Turkish temporal logic

Table 1 above covers the application of Reichenbach temporal logic to English and there are empty lines for meaningless cases in English. In Turkish temporal logic the same table can be modified as below:

Table 3. 13 permutation of Reichenbach temporal logic and their Turkish tense/aspect and a sample for each case.

Permutation	Reichenbach Tense Name	Turkish Tense	Sample
E<R<S	Anterior past	Geçmişin hikayesi	Uyumuştum
E=R<S	Simple past	Geçmiş	Uyudum
R<E<S		Gelecek hikayesi	
R<S=E	Posterior past		
R<S<E		Gelecek Rivayeti	Uyuyacakmış
E<S= R	Anterior present	Şimdiki Hikayesi	Uyumuşum
S= R= E	Simple present	Şimdiki	Uyuyorum
S= R<E	Posterior present	Gelecek	Uyuyacağım
S<E<R		Gelecek Hikayesi	Uyuyacaktı
S=E<R	Anterior future		Uyuyor olacağım
E<S<R			Uyumuş olacağım
S<R=E	Simple future	Gelecek Zaman	Uyuyacağım
S<R<E	Posterior future		Uyuyacak olacağım

Please note that Turkish has more alternatives for future tense. In fact in English there is no future tense but modals[9].

Also in Turkish some of the above cases have rare frequency for daily life. For example the cases with two words, which the second word in the samples starts with “ol-“ are almost out of Turkish temporal logic but still meaningful and understandable by any Turkish speaker.

Also the grammatical word ‘Hikaye’ can be translated into English like story telling and we will name this alternative as Story and word ‘Rivayet’ can be translated into English as reporting and we still name this alternative as ‘Learned’ to avoid mixture of reporting aspect in links of TimeML.

In fact, the story telling requires someone to live the event and reporting requires someone to learn in Turkish semantic so we have named these tenses as ‘Story’ and ‘Learned’.

We have modified the <MAKEINSTANCE> tag and added the below alternatives to make TimeML suitable with Turkish events:

```
tense ::= ’PAST’ | ’PRESENT’ | ’FUTURE’
| ’NONE’ | ’INFINITIVE’ | ’PRESPART’ |
’PASTPART’ | ‘STORY’ | ‘LEARNED’ |
‘BEING’
```

```
aspect ::= ’PROGRESSIVE’ | ’PERFECTIVE’
| ’PERFECTIVE_PROGRESSIVE’ | ’NONE’ | ‘
```

5. ATL and TimeML

TimeML is a time mark up language first developed in 2002 and recently very popular standard for stamping events [6], ordering events, reasoning about the persistence of events and modelling time expressions in natural language texts.

Obviously TimeML is developed on English natural language [7] and now some other languages like Ukrainian[8] or French[9] have been applied on TimeML with some modifications.

The latest version of TimeML which is modified in 2006 has three layers of semantic representation:

- Event Level
- Signal Level
- Link Level

In the event level, the events on the natural language text are stamped with the information of event like tense, aspect, modal, continuity or plurality. In the signal level those events are marked with the recurrence or durations. Finally on the up most level, link level those signalled events are connected with before, after, eventually and similar connections.[10]

The connection type of TimeML in the link layer is inspired from Allen’s temporal logic [8]. And unfortunately does not support the full Turkish temporal logic explained in the previous section.

The link between events on TimeML are represented by using the TLink tags and the Backus-Naur Form (BNF) of TLink is quoted below:

```
relType ::= ’BEFORE’ | ’AFTER’ |
’INCLUDES’ | ’IS_INCLUDED’ |
’DURING’ | ’DURING_INV’ | ’SIMULTANEOUS’
| ’IAFTER’ | ’IBEFORE’ |
’IDENTITY’ | ’BEGINS’ | ’ENDS’ |
’BEGUN_BY’ | ’ENDED_BY’
```

Please note that the possibilities for “relType” symbol above are inspired from ATL and none of the above alternatives supports the Turkish temporal case explained in previous section.

Table 2. ATL relation types.

relType	**Comment**	**Sample**
BEFORE	An event finishes before another starts	He came and seen the label.
AFTER	Reverse form of BEFORE and can substitute it.	
INCLUDES	A link to temporal expression or event which includes event.	John arrived in Boston last Thursday.
IS_INCLUDED	Reverse form of INCLUDES.	
DURING	A link to temporal expression or event while the event is on	James was CTO for two years.
DURING_INV	Reverse form of DURING.	
SIMULTANEOUS	Two events at the same time.	
IAFTER	Immediately after	All passengers died when the plane crashed to the mountain.
IBEFORE	Reverse form of IAFTER.	
IDENTITY	Repeat of same verb.	He drove to Boston. While his drive he ate a donut.
BEGINS	A link to temporal expression or event which begins the	John was teaching since 1980.
ENDS	A link to temporal expression or event which ends the event.	John was in gym until 7.00pm.
BEGUN_BY	Reverse notation of BEGINS.	
ENDED_BY	Reverse notation of ENDS.	

As a solution, we suggest adding a new relation type named “CYCLES” to above table and the BNF which states that the cycle between two events continuously.

6. Demonstration of Temporal Expressions

A more precise problem in temporal expressions is the design of data structures and optimization of the algorithms to demonstrate the temporal expressions.

In a text, multiple unlinked event chains can be written and the writer can link these multiple event chains at any time or furthermore can leave them unlinked.

For example both of below examples are possible for natural language texts:

1- “I was out on a boat with a few of my friends for fishing.”
2- “Six marine policemen searched PSS Pollu and discovered that it was carrying smuggled cigarettes.”

Above two quotations can be transformed to below event chains:

1- Go fishing → Being out on a boat
2- Being carrying smuggled cigarettes → Being searched by policeman → the cigarettes found

Above two chains have no link in the temporal domain. Writer can add a link and make these chains connected as below:

3.1- “Police didn’t permit our sailing because of investigation about PSS Pollu”

Or another possible link would be as below:

3.2- “On the return we have seen PSS Pollu is releasing oil into the sea and we have called police”

In the 3.1 linking, the order of event chains can be demonstrated as 2→1 and on the contrary the order of event chains in 3.2 linking is 1→2.

Above case is a demonstration of linking event chains. The possibility of linking events at any time or even multiple times in a natural language, makes it difficult to demonstrate on a time line.

Another problem is the demonstration of unlinked event chains. Putting two unlinked event chains into the same time line is impossible. On the other hand a demonstration algorithm should decide on which event chain to start from.

As a solution the events are demonstrated in a two dimensional domain instead of one dimensional time line. The first dimension of our demonstration is for the timeline as expected and the second dimension is for the unlinked events. So any number of unlinked events will get new units in second dimension.

Figure 5. Unlinked Events

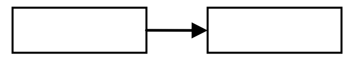

Figure 6. Linked Events

The unlinked events are demonstrated on y axis and the linked events are connected on x axis as demonstrated on figure 5 and 6 where the boxes indicates the events.

Above demonstration on figure 5 indicates that two events are unlinked, which can be interpreted as, the reader can not say anything about the time of the first event with respect to the second event.

7. Implementation and Testing

We have collected a corpus of 15 children stories with 1043 sentences and 1618 events. The distribution of events on tenses in corpus is listed below:

Table 4. Percentages of Turkish Reichenbach temporal model in our corpus.

Permutation	Turkish Tense	#	%
E<R<S	Geçmişin hikayesi	165	10%
E=R<S	Geçmiş	133	8%
R<E<S	Gelecek hikayesi	238	15%
R<S=E			0%
R<S<E	Gelecek Rivayeti	37	2%
E<S= R	Şimdiki Hikayesi	76	5%
S= R= E	Şimdiki	390	24%
S= R<E	Gelecek	47	3%
S<E<R		478	30%
S=E<R	*	1	0%
E<S<R		3	0%
S<R=E	Gelecek Zaman	48	3%
S<R<E	*	2	0%

Please note that the above corpus is mainly contains stories so it is an expected result to get higher percentages on the

* Does not exist in Turkish. Usually occurs because of mistranslation from a foreign language to Turkish.

past tenses and present tenses than the future tense. Also the future tenses with “ol-“ (Being) is almost zero because of their rarely usage in Turkish. In fact most of their usage are from translated stories in Turkish.

After adding the above relation type alternative to TLink BNF, we have re implemented the ttk-1.0 and Tango v1.5 which are two major software implemented for TimeML applications. Also we have created a corpus of child stories for test purposes in Turkish and tested the success of older and newer versions of TimeML, where in newer version the “CYCLES” relation type is implemented.

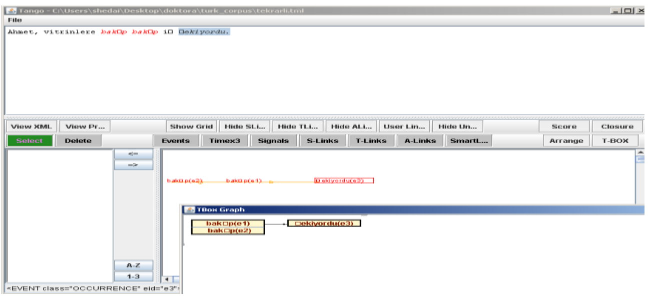

Fig5. Sample screenshot of ATL implementation for Turkish

- Success of TimeML in Turkish Corpus without CYCLES relation: 53%
- Success of TimeML in Turkish Corpus with CYCLES relation: 55%

There is a 2% of increase after the implementation of the new relation type to TimeML.

By this study, we have enhanced TimeML a step beyond to cover Turkish temporal logic. After the above modifications TimeML can be used in both Turkish and English and can model more events successfully. Also this enhancement depends on the Reichenbach temporal logic and already discovered by him as a permutational manner.

Also the study is applied on a corpus and the numerical results have been demonstrated above. The success of TimeML before modification would be 55% percent and after the modifications we have suggested the success covers all possible event tenses which is 91%.

8. Conclusion

During this study, we have stated the Allen’s temporal logic and its applications on natural language processing. The comparison between ATL and temporal logic behind Turkish also criticized and an additional relation type to ATL is suggested to cover Turkic languages. Also this suggestion in logical level is carried on to the implementation layer and an application is modified using this theoretical logic. The tests carried on a corpus composed by simple Turkish child stories have showed the importance of this addition and increased the success of representation of Turkish natural language sentences in both ATL and TimeML.

This study is based on TimeML in order to keep connection with the previous studies on. Since TimeML is built on ATL and Reichenbach temporal logics, the modeling of Turkish is enforced to fit those logics during this study. On the other hand, there are some temporal logics which can handle the Turkish temporal semantic already. Unfortunately there is not any machine computable implementation standards for those logics. Creating a wider markup language to substitute TimeML would be a future work.

9. References

[1] Automatic Time Expression Labeling for English and Chinese Text, Kadri Hacioglu, Ying Chen and Benjamin Douglas Springer LNCS, Computational Linguistics and Intelligent Text Processing, ISSN 0302-9743 , pages 548-559 , 2005

[2] Robust temporal processing of news, Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, Inderjeet Mani , George Wilson Pages: 69 - 76 , 2000

[3] Allen Linear (Interval) Temporal Logic – Translation to LTL and Monitor Synthesis– Grigore Ro¸su1 _ and Saddek Bensalem, CAV'06, LNCS 4144, pp 263-277, 2006

[4] Reichenbach, H., Elements of Symbolic Logic, New York: Macmillan (1947)

[5] Ozkirimli A, Türk Dili: Dil ve Anlatım, İstanbul Bilgi University Press, 2001.

[6]Frank D. Anger, Debasis Mitra, Rita V. Rodriguez, “Satisfiability in Nonlinear Time: Algorithms and Complexity“ , Proceedings of the Twelfth International FLAIRS conference, AAAI, 1999

[7] TimeML, TERQAS, (2002), TimeML has been developed in the context of three workshops starting from 2002.

[8] “TimeML Annotation Guidelines Version 1.2.1” Roser Saur´ı, Jessica Littman, Bob Knippen, Robert Gaizauskas, Andrea Setzer, and James Pustejovsky (January 31, 2006)

[9] Natalia Kotsyba Using Petri nets for temporal information visualization Études Cognitives/Studia Kognitywne, CEEOL (2006)

[10] Andr´e Bittar , Annotation of Events and Temporal Expressions in French Texts Proceedings of the Third Linguistic Annotation Workshop, ACL-IJCNLP 2009, pages 48–51, Suntec, Singapore, 6-7 August 2009. c 2009 ACL and AFNLP

[11] Christian Kissig and Laura Rimell , Closing TLink-Relations (Reasoning with Intervals) June 23, 2005