Formal Verification of Temporal Questions in the Context of Query-Answering Text Summarization

Nasrin Mostafazadeh[1], Omid Bakhshandeh Babarsad[2], and Gholamreza Ghassem-Sani[1]

[1] Sharif University of Technology, Tehran, Iran
[2] Mechatronics Research Laboratory, Qazvin, Iran
mostafazadeh@ce.sharif.edu, bakhshandeh@mrl.ir, sani@sharif.edu

Abstract. This paper presents a novel method for answering complex temporal ordering questions in the context of an event and query-based text summarization. This task is accomplished by precisely mapping the problem of "query-based summarization of temporal ordering questions" in the field of Natural Language Processing to "verifying a finite state model against a temporal formula" in the realm of Model Checking. This mapping requires specific definitions, structures, and procedures. The output of this new approach is promisingly a readable and informative summary satisfying the user's needs.

1 Introduction

With the significant growth of the Internet, users require to express their information needs as natural language questions and expect to get a natural language answer in a precise, comprehensible and coherent form. Also, they are not always looking for the exact answer of a single factoid question. There are often complex questions and some sort of reasoning and deduction is required for answering them. We propose that Question Answering (QA) systems can be improved by exploiting text summarization (TS) to extract more than just the exact answer from the document and provide the user with necessary background in the form of a coherent summary.

Addressing complex questions about temporal relation and ordering of events (such as after, before, simultaneous) is one of the demanding research topics in the field of QA. In this paper, we propose a novel event-and-query-based text summarization approach in order to tackle the mentioned problem. The overall idea is to consider the problem as verifying (i.e., proving to the user) a specification (i.e., query of the user) in the context of a logical Kripke model (i.e., a model of the input text document(s)) by means of Model Checking. Common Query-based Text Summarization (QTS) systems try to select sentences (or other desired components) based on degree of relevance to the query [2]. Such systems usually do not consider the query as a question to be answered, but

L. Kosseim and D. Inkpen (Eds.): Canadian AI 2012, LNAI 7310, pp. 350–355, 2012.

as a focus restriction on finding salient components. On the other hand, QA systems are based on finding the exact answer of the question (a name entity, sentence(s), etc.) or a paragraph containing the answer. Many related works believe that the answers to the complex questions are composed from the answers of the simple constituting factoid questions [8,7]. Some other works argue that temporal QA requires complex logical inferences [5,9]. Our approach does not tend to rank the text components according to the query, but it attempts at precisely answering the query in the form of a sequence of sentences at the length it decides (not a predefined summary ratio). Therefore, our approach introduces a new perspective which lies in the intersection between QTS and QA areas, which we characterize as "Query-answering Text Summarization".

Our algorithm encompasses four phases of Natural Language (NL) Pre-processing, Translation to Logic Form, Formal Model Verification, and NL Post-processing. The results of preliminary evaluation of the system show that our approach satisfies both 'readability' and 'informativeness' criteria. The rest of this paper is organized as follows: Section 2, steps through the Pre-processing and Logical Translation phases. We discuss the Formal Model Verification phase and the rationale of the entire approach in section 3. The evaluation of the system is discussed in section 4. Finally, section 5 concludes the paper's overall idea and focus.

2 Pre-processing and Logical Translation

In the NL-pre processing phase, we simultaneously annotate the query and input document with their corresponding temporal event relations tags. We need to extract TLINKs (temporal links) representing the relations between events of the given document and query based on TimeML [6] standard. From now on, we call the following three qualified TLINK relations as 'TLINK* relTypes' = {simultaneous, ibefore, before}. We consider the extraction of events and their relations in the TimeML format –an ongoing research field in NLP– a black-box in our approach.

Here, we define Temporal Acyclic Graph (TAG), having EVENT IDs as its node and TLINK* relations as the edges. A sample TAG is depicted in Fig. 1(A). If we define a 'process' as a set of correlated events, then the independent subgraphs implicitly indicate concurrent processes. We characterize each independent subgraph of TAG as a 'SubG' component. On the other hand, in Fig. 1, events S1 and S2 are taking place simultaneously along with each other so the subsequent events of S1 are implicitly synchronized with subsequent events of S2. Thus, we transform the simultaneous nodes into one super-node where two events will have identical relations with the rest of graph. After these manipulations, the resultant graph will be called TAG*. TAG* equivalent of Fig. 1(A) is shown in Fig. 1(B).

Kripke model (represented by a 4-tuple $K = (S, R, I, L)$) is a notion used to describe 'reactive systems'. For the logical translation phase, we model each SubG component of TAG* by a Kripke structure $SubGK = (S, R, I, L)$ where

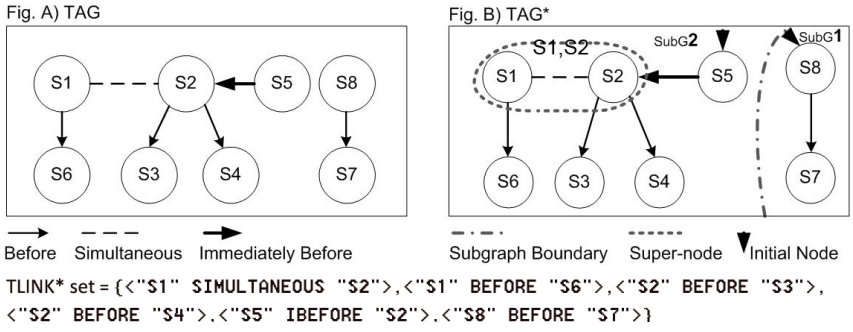

Fig. 1. Corresponding TAG of 'TLINK* set'

each $s_i \in S$ signifies one of the nodes of SubGK; each $(s_i, s_{i+1}) \in R$ denotes the TLINK* holding between s_i and s_{i+1}; and each $s_i \in I$ is an initial node (primary event). The set L contains all the primitive events labels together with Ø label, which conveys a dummy state. A state being labeled with event s_i indicates that s_i is occured and a state with lable Ø implies that no event happens. For unifying the $SubGK$s, we perform an asynchronous composition, where at each time instant only one independent component is selected to perform a transition. The set L associated with the resultant final Kripke model is extended by adding a couple of (s_i,s_j) to the previous L which signifies simultaneous occurrence of the two events s_i and s_j. The Kripke model resulted from asynchronous composition of SubG1 and SubG2 of Fig. 1(B) is depicted in Fig. 2.

After temporal modeling of the text, we should also translate the annotated query into a temporal logic formula. The term "Temporal Logic" has been widely used to refer to all approaches of representing temporal information within a

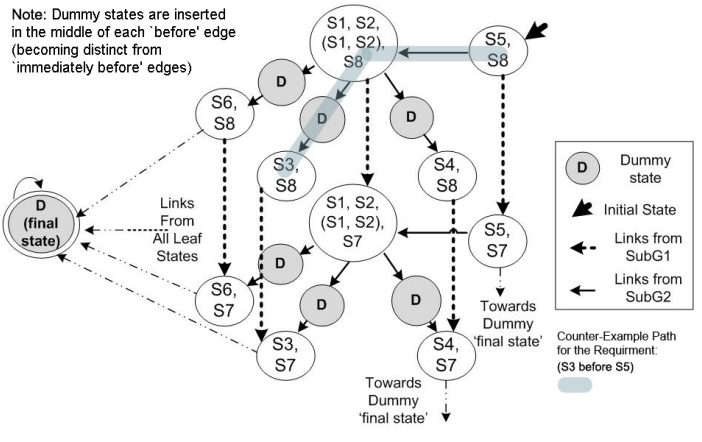

Fig. 2. Equivalent Kripke model of TAG* in Fig. 1(B)

logical framework[1]. For our problem, the appropriate form of Temporal Logic is the Computational Tree Logic(CTL) –which is interpreted over the computation tree and paths of Kripke structure. Common logical quantifiers $\{\forall, \exists\}$ and boolean connectives $\{\neg, \rightarrow, \wedge, \vee\}$ are available in CTL. CTL also provides four temporal operators (i.e., F, G, X, and U) as follows. For 'p' $\in$ L and s[t] $\in$ S: **(F) Finally or future** : 'Fp' is true in s[t] iff p is true in some s[t'] with t' $\geq$ t; **(G) Globally**: 'Gp' is true in s[t] iff p is true in all s[t'] with t $\leq$ t'; **(X) Next**: 'Xp' is true in s[t] iff p is true in s[t+1]; **(U) Until**: 'pUq' is true in s[t] iff q is true in some state s[t'] with t $\leq$ t' and p is true in all states s[t"] with t $\leq$ t" $\lneq$ t'. For example, we can formalize the concept "there exists some paths in which birth occurs and death never occurs in future" by formula EF(Birth $\rightarrow$ AG($\neg$ Death)), where 'E' is the symbol for *'$\exists$ a path'* and 'A' is the symbol for *'$\forall$ existing paths'*. After investigating the spectrum of various user queries, we distinguished three major query models. TAG specification and equivalent CTL formula of each model is listed in Table 1.

Table 1. The variety of possible user queries: All possible relations within an input TAG are considered. In each sentence, (e_i) denotes an extracted event.

Query Model#1: Did Einstein embarked($e1$) on the construction($e2$) of unified field theories After World War II($e3$)? **TAG Spec.1**: ($e3$ before $e1$) & ($e1$ simultaneous $e2$) **CTL1**: AG(($e3 \rightarrow$ AF(e1)) $\wedge$ ($\neg$ e1)) $\wedge$ EF($e1 \wedge e2 \wedge (e1, e2)$)
Query Model#2: Did Einstein take ($e1$) the position of Professor of Theoretical Physics at Princeton while renounced ($e2$) his German citizenship? **TAG Spec.2**: ($e1$ simultaneous $e2$), **CTL2**: EF($e1 \wedge e2 \wedge (e1, e2)$)
Query Model#3: Did Einstein married ($e1$) his cousin, Elsa, immediately after he divorced ($e2$) Mileva? **TAG Spec.3**: ($e1$ ibefore $e2$), **CTL3**: AG((e1 $\rightarrow$ AX(e2)) $\wedge$ ($\neg$ e1))

3 Formal Model Verification

"Formal Verification" is the act of proving or disproving the correctness of a system's functionality with regards to a certain specification or requirement, by using mathematical techniques. In "Model checking" (an automatic technique for verifying 'concurrent'finite state systems) [3], the target system is modeled as a finite state transition system (Kripke model, M), and the specifications are expressed in a propositional temporal logic (CTL formula, ϕ). Then by efficiently exploring the state space of the system model, model checker automatically verifies whether the specification is satisfied (i.e., $M \models \phi$). One of the most significant features of model checking highly appropriate for our problem is that when a specification is not held, a 'counterexample' (i.e., a proof of the contrary behavior of the system) is produced. Model Checker produces our desired output (path of states) only in the case that the specification does not hold. What if the

[1] As defined in the Stanford Encyclopedia of Philosophy.

specification is verified to be true, in which case there is no counter example? The solution is to modify the primary specification (and produce a secondary specification) such that it does not hold anymore. The resultant counter example by using the secondary specification will be a proof that the primary specification holds. All models of primary CTL formulas of Table 1 and their corresponding secondary queries are shown in Table 2. Afterwards we input the primary specification together with the SMV representation of the input text (automatically extracted from the Kripke model) to the NuSMV [4] Bounded Model Checker to obtain the shortest possible counterexample. In NL Post-processing phase, the final summary can be simply constructed by selecting sentences corresponding to the events of the counter example. In this phase various complex refinement techniques can be used for constructing the summary which is left as a future work.

Table 2. Secondary CTL formula and path expectation for initial yes/no output in response of each primary CTL formula

Y/N	Primary CTL formula	Secondary CTL Formula	Path Expectation
yes	EF($e1 \wedge e2 \wedge (e1, e2)$)	$\neg$EF($e1 \wedge e2 \wedge (e1, e2)$)	Containing the simultaneous events.
yes	AG(e1 $\rightarrow$ AX(e2))	check all possibilities	The proof of any relation which holds.
yes	AG(e3 $\rightarrow$ AF(e1))	same as above.	same as above.
no	EF($e1 \wedge e2 \wedge (e1, e2)$)	same as above	same as above.
no	AG(e1 $\rightarrow$ AX(e2))	No Nead.	The proof why the answer was no.
no	AG(e3 $\rightarrow$ AF(e1))	No Nead	same as above.

4 Evaluation and Results

As discussed earlier, the problem we have addressed lies in the intersection between QA and QTS systems. Thus, the evaluation metrics, methodologies, peer and gold standard documents which are normally being used in these two fields can not be applied to our system. As a result, we should define new evaluation metrics and methods based on the observed user satisfiability factors. The evaluation of our system can be regarded as a separate essential research. As a preliminary evaluation, we evaluate the system manually. We use document sets created for the biography evaluation of the task 5, Document Understanding Conference (DUC), 2004. For each document a set of questions covering all query models of Table 2 were designed. For the 'readability factor', our evaluation was based on the DUC five readability assessment criteria: Grammaticality, Non-redundancy, Referential clarity, Focus and Structure and Coherence. For the 'informativeness factor', the following criteria were designated for assessing system's query answering and summarization performance: Correct Y/N answer (verification), Correct Relation Identification, Exact Accompanying Proof,

Accompanying Background Information. The evaluation was accomplished by human judges who had read the guidelines for the assessment task. The preliminary results showed that the accuracy of temporal ordering verification is 100%. Also, the system had satisfied all the user expectations for providing the accompanying text as a background information and proof of the verification. The current system has very simple pre and post processing phases; by employing complex summary generation techniques, we can produce more focused and cohesive summaries in future.

5 Conclusion

In this paper, we introduced a novel approach for temporal query-answering text summarization. We addressed temporal ordering questions by mapping the problem into the formal verification of the query against the finite state Kripke model of the input text. The obtained results of preliminary system evaluation show that our approach is capable of producing coherent and informative summaries.

References

1. Biere, A., Cimatti, A., Clarke, E., Strichman, O., Zhu., Y.: Bounded model checking. In: Advances in Computers, vol. 58. Academic Press (2003)
2. Chali, Y., Joty, S.: Unsupervised Approach for Selecting Sentences in Query-based Summarization. In: FLAIRS (2008)
3. Doron, O.G., Clarke, E.M., Orna Grumberg, J.: Model Checking. MIT Press (1999)
4. Cimatti, A., Clarke, E.M., Giunchiglia, F., Pistore, M., Roveri, M., Sebastiani, R., Tacchella, A.: NuSMV: a new symbolic model checker. It'l. J. on STTT 2(4), 410–425 (2000)
5. Harabagiu, S., Bejan, C.A.: Question Answering Based on Temporal Inference. In: AAAI Workshop on Inference for Textual QA (2005)
6. Pustejovsky, J., Castaño, J.M., Ingria, R., Sauri, R., Gaizauskas, R.J., Setzer, A., Katz, G., Radev, D.R.: TimeML: Robust specification of event and temporal expressions in text. In: AAAI Symposium on New Directions in QA, pp. 28–34 (2003)
7. Sequete, E., Martinez-Barco, P., Munoz, R., Vicedo, J.: Splitting Complex Temporal Questions for QA Systems. ACL (2004)
8. Sequete, E., Vicedo, J.L., Martinez-Barco, P., Munoz, R., Llorens, H.: Enhancing QA systems with complex temporal question processing capabilities. JAIR 35, 755–811 (2009)
9. Schockaert, S., Ahn, D., De Cock, M., Kerre, E.E.: Question Answering with Imperfect Temporal Information. In: Larsen, H.L., Pasi, G., Ortiz-Arroyo, D., Andreasen, T., Christiansen, H. (eds.) FQAS 2006. LNCS (LNAI), vol. 4027, pp. 647–658. Springer, Heidelberg (2006)