A method for GeoTime information retrieval based on question decomposition and question answering

Yokohama National University at NTCIR-8 GeoTime

Tatsunori Mori
Graduate School of Environment and Information Sciences, Yokohama National University
79-7 Tokiwadai, Hodogaya, Yokohama 240-8501, Japan
mori@forest.eis.ynu.ac.jp

ABSTRACT
In this paper, we report the evaluation results of our GeoTime information retrieval system at NTCIR-8 GeoTime. We participated in the Japanese mono-lingual task (JA-JA). Our proposed method for GeoTime information retrieval is based on question decomposition and question answering. We demonstrated that the proposed method is able to accept GeoTime questions and retrieve relevant documents to some extent. However, there is still room to improve the effectiveness of retrieval. In per-topic evaluation results, we can find there are some topics that cannot be appropriately handled by our method, and therefore the method lacks in robustness in terms of variety of GeoTime questions.

Categories and Subject Descriptors
I.2.7 [**Artificial Intelligence**]: Natural Language Processing

General Terms
Algorithms

Keywords
Question decomposition, question-answering

1. INTRODUCTION
In this paper, we will report the evaluation results of our GeoTime information retrieval system at NTCIR-8 GeoTime[2]. We participated in the Japanese mono-lingual task (JA-JA). Our proposed method for GeoTime information retrieval is based on question decomposition and question answering.

GeoTime information retrieval can be regarded as one special case of IR4QA, because q query submitted to a system is a natural language question in typical situations. We may straightforwardly consider documents that have good answer candidates for the question as documents relevant to the query. Therefore, we developed a system that utilize a question-answering system.

A GeoTime query, or a GeoTime question, usually has multiple interrogative, like when and where, to seek for geographical answers and temporal answers simultaneously. On the other hand, existing question-answering systems usually can answer simple questions, which have single interrogative. In order to cope with the issue, we introduced a method called *question decomposition*. It decomposes a GeoTime question into a set of simple factoid questions. Those simple factoid questions are submitted to an existing question-answering system.

The answer candidates for each simple factoid question have their own scores. The scores of answer candidates are integrated in a document-by-document manner to obtain document scores, which represent the relevance of documents to a given GeoTime question.

2. RELATED WORK
GeoTime information retrieval may be regarded as a special case of information retrieval for question answering (IR4QA) from the following viewpoints:

- the system is expected to retrieve documents that include answer candidates for a given query, or question,
- however, the user asks the system for geographical answers and temporal answers simultaneously by the query.

Although many approaches to IR4QA introduce some extensions to treat natural sentence questions or question types, their foundation are information retrieval systems[7].

However, there are some text processing method based on the result of question answering system. For example, Mori et al.[5] proposed a method for multi-answer-focused summarization using a question-answering engine. Importance of each sentence is calculated based on the scores of answer candidates appeared in the sentence. In this method multiple questions are take account of simultaneously.

Our GeoTime information retrieval takes the same kind of approach as the latter researches. While, in these researches, the scores of answer candidates are used to weight sentences, documents are weighted according to the score in our GeoTime information retrieval.

3. PROPOSED METHOD

Figure 1 shows the overview of the proposed method. It consists of the following three procedures, which correspond to Algorithms 3.1, 3.3, and 3.4 described later, respectively:

1. Decomposing a complex GeoTime question into a set of simple factoid questions,
2. Factoid question-answering for the simple questions,
3. Scoring documents according to the scores of answer candidates in each document.

These procedures are explained in detail in Sections 3.1, 3.2, and 3.3.

The rest of this section is organized as follow. In Section 3.1, we will introduce a method to decompose a GeoTime question into a set of simple factoid questions. Section 3.2 will present the overview of a factoid question-answering system we utilized. In Section 3.3, we will propose a method of GeoTime information retrieval

3.1 Question decomposition

GeoTime questions are usually complex questions, which have multiple interrogatives, like when, where, etc. We suppose that each GeoTime question is able to be decomposed into a set of simple factoid questions. The simple factoid questions obtained by the decomposition may be handled a factoid question-answering system.

Algorithm 3.1 shows an algorithm of question decomposition we employed.

Algorithm 3.1: DECOMPOSEQUESTION(Qc)

comment: returns a set of tuples of $\langle Q, interrog\rangle$, where Q is a simple question with one interrogative $interrog$, which is obtained by the decomposition of an inputted complex GeoTime question Qc.

global $InterrogPats$
comment: $InterrogPats$ is a set of patterns that match with interrogatives in question sentences.

procedure PATTERNMATCH($Str, Pats$)
  comment: returns a set of tuples of position $\langle PosS, PosE\rangle$, where $PosS$ and $PosE$ are the start and end positions of a substring of Str matched with one of patterns $Pats$.
  return $(\{\langle PosS, PosE\rangle\})$

procedure SUBSTR($Str, \langle PosS, PosE\rangle$)
  comment: returns a sub-string $SubStr$ of Str that starts from position $PosS$ and ends at position $PosE$.
  return (SubStr)

procedure DELSUBSTRS($Str, Matches$)
  comment: returns a string $Str1$ that is obtained by deleting all substring expressed by $Matches$ from a string Str.
  return $(Str1)$

main
  $Ms \leftarrow \textsc{PatternMatch}(Qc, InterrogPats)$
  $Qs \leftarrow \bigcup_{M \in Ms}\{\langle \textsc{DelSubstrs}(Qc, Ms\backslash\{M\}), \textsc{Substr}(Qc, M)\rangle\}$
  return (Qs)

3.2 Factoid question-answering system

The factoid QA system used in this study is a real-time QA system based on [4]. It can answer Japanese factoid questions. As shown in Figure 2, the system comprises six processes — question analysis, interface to external search engine, passage extraction, sentence matching, answer generation, and pseudo voting. These processes have some parameters including those shown in Table 1.

Table 1: Description of system parameters

a:	Number of answers to be searched.
d:	Number of documents to be retrieved.
ppd:	Maximum number of passages retrieved from one document.
p:	Number of passages to be considered in the retrieved documents.
pwin:	Number of sentences in one passage.

The process of question analysis involves receiving a question from a user and extracting several types of information

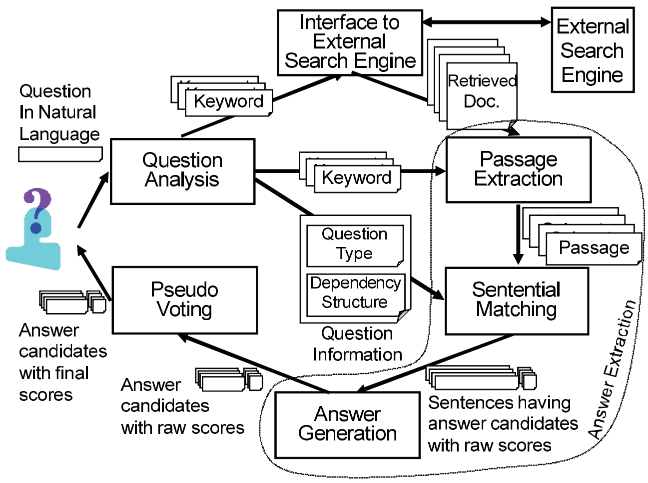

Figure 2: Factoid question-answering system

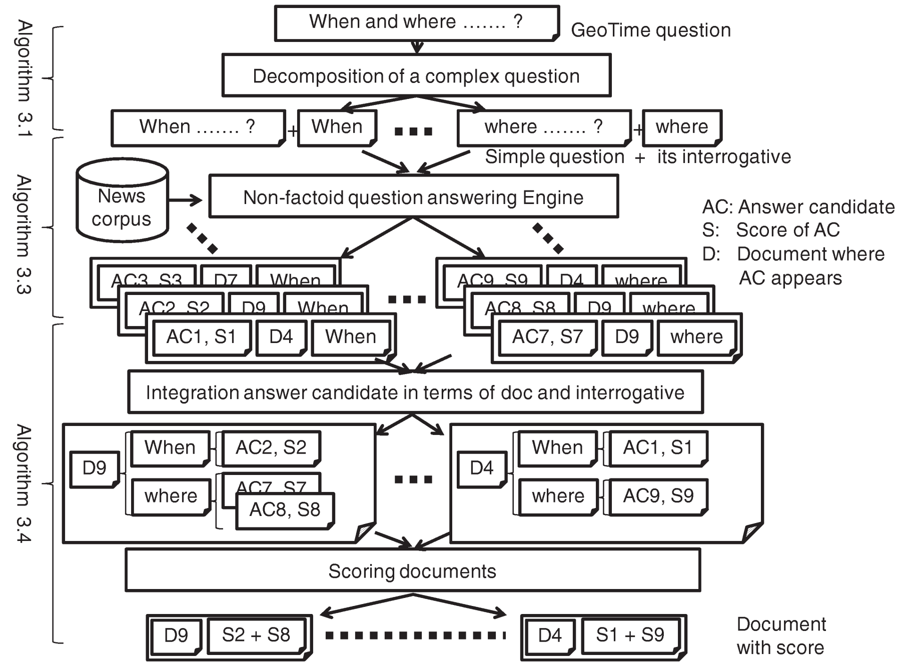

Figure 1: System overview

including a list of keywords and the question type. In this paper, we define the term *Keywords* as content words in a given question. The list of keywords is submitted to an external search engine for retrieving relevant documents. Although we may use any kind of external search engines with some wrapper programs to adjust protocols, we utilize our original search engine. It is based on a straightforward tf*idf method for term weighting and the vector space model for calculating similarity between a list of keywords and a document. We do not introduce any feedback methods to the engine.

The process of sentence matching involves receiving a set of sentences from the passage extractor. In this process, each morpheme is treated as an answer candidate and assigned a matching score as described below. It should be noted that a morpheme may be either a word or a part of a longer compound word. Therefore, in the latter case, the process of answer generation involves the extraction of a compound word including the answer candidate; and this compound word is then treated as a proper answer candidate.

3.2.1 Raw scores for answer candidates

In the factoid QA system, a composite matching score for an answer candidate is adopted as shown in Equation (1). We term this score *raw score* in this paper. It is a linear combination of the following subscores for the answer candidate AC in the i-th retrieved sentence L_i with respect to a question sentence L_q:

1. $Sb(AC, L_i, L_q)$ — matching score in terms of character 2-grams
2. $Sk(AC, L_i, L_q)$ — matching score in terms of the keywords
3. $Sd(AC, L_i, L_q)$ — matching score in terms of the dependency relations between an answer candidate and the keywords
4. $St(AC, L_i, L_q)$ — matching score in terms of the question type

In the calculation of $St(AC, L_i, L_q)$, we employ a named entity (NE) recognizer that identifies eight types of NEs defined in IREX-NE [3].

$$S(AC, L_i, L_q) = Sb(AC, L_i, L_q) + Sk(AC, L_i, L_q) + Sd(AC, L_i, L_q) + St(AC, L_i, L_q) \quad (1)$$

In order to reduce the computational cost, the A* search control is introduced in the sentence matching mechanism. With this control, the system can process the most promising candidate first, while delaying the processing of the other candidates, and perform the n-best search for the answer candidates.

3.2.2 Pseudo voting method in search scheme

Many existing QA systems exploit global information on answer candidates. In particular, redundancy is the most basic and important information. For example, there are previous studies that boost the score for answer candidates that occur multiple times in documents [1, 8]. This is known as the *voting method*.

In contrast, we cannot exploit the voting method directly while searching answers because the system quits the searching after n-best answers are found. Therefore, an approximation of the voting method, termed *pseudo voting*, is introduced as follows: In the case that n-best answers are necessary, the system continues searching for answers until n *different* answer candidates are found. Therefore, the system may find other answer candidates that have the same surface expression as one of the answer candidates that has already been found. Consequently, we can partially use the frequency information of answer candidates by recording all the found answer candidates. In this paper, the pseudo voting score $S^v(AC, L_q)$ for the answer candidate AC is defined as follows:

$$S^v(AC, L_q) = (\log_{10}(freq(AC, AnsList)) + 1) \cdot \max_{L_i} S(AC, L_i, L_q) \quad (2)$$

where $AnsList$ is the list of answer candidates that have been found in the n-best search and $freq(x, L)$ is the frequency of x in L. In this paper, we call the pseudo voting score *the weighted score*. According to the experiments by Murata et al. [6], the above voting score is comparable with other good voting scores.

3.3 GeoTime information retrieval

By using the question decomposition method and the factoid question-answering system described in Section 3.1 and 3.2, respectively, we developed a GeoTime information retrieval method defined in Algorithm 3.2. It calls the following sub-procedures:

DecomposeQuestion() is defined as Algorithm 3.1 and decomposes a complex GeoTime question into a set of simple factoid questions,

GetAllAnsCands() is defined as Algorithm 3.3 and calls the factoid question-answering system to obtain answer candidates and their scores for all of the simple questions,

ScoreDocs() is defined as Algorithm 3.4 and calculates the score of each document according to the scores of answer candidates in the document.

In the procedure ScoreDocs(), all answer candidates are grouped by document, and then answer candidate in a document are grouped by interrogative of simple question, as shown in Figure 1. We define the sub-score of document in terms of an interrogative as the maximum score of answer candidates that associated with the interrogative, and finally define the score of document as the summation of the sub-scores over all interrogatives as shown in procedures ScoreDoc1() and ScoreDoc1() in Algorithm 3.4.

Since we have two types of scores of answer candidates, namely weighted scores and raw scores, two scoring strategies, Strategy 1 (weighted score) and Strategy 2 (raw score), are prepared, respectively.

Algorithm 3.2: GeoTime(Qc, $Strategy$)

comment: returns a set of tuples of $\langle D, S\rangle$, where D and S are a document and its score. The inputs Qc and $Strategy$ are the inputted GeoTime question and the ID of the scoring strategy, respectively.

$Qs \leftarrow$ DecomposeQuestion(Qc)
$ACs \leftarrow$ GetAllAnsCands(Qs)
$DSs \leftarrow$ ScoreDocs(ACs, $Strategy$)
return (DSs)

Algorithm 3.3: GetAllAnsCands(Qs)

comment: returns a set of tuples of $\langle D, interrog, AC, Sr, Sw\rangle$, where AC and D are an answer candidate and a document in which the answer candidate appears. $interrog$ is the interrogative asked in a decomposed question. Sr and Sw are the raw and weighted score of the answer candidate. The inputs Qs is a set of decomposed questions.

procedure QA(Q)
 comment: returns a set of tuples of $\langle AC, D, Sr, Sw\rangle$ for the question Q by using a factoid question-answering system.
 return ($\{\langle AC, D, Sr, Sw\rangle\}$)

main
 $ACs \leftarrow \{\}$
 for each $\langle Q, interrog\rangle \in Qs$
 do $\begin{cases} As \leftarrow \mathrm{QA}(Q) \\ \textbf{for each } \langle AC, D, Sr, Sw\rangle \in As \\ \quad \textbf{do } ACs \leftarrow ACs \bigcup \{\langle D, interrog, AC, Sr, Sw\rangle\} \end{cases}$
 return (ACs)

Algorithm 3.4: ScoreDocs(ACs, $Strategy$)

comment: returns a set of tuples of $\langle D, S\rangle$, where S is the score of document D.

procedure Docs(ACs)
 comment: returns a set of all documents appeared in ACs.
 return ($\{D\}$)

procedure Interrogs(ACs)
 comment: returns a set of all interrogatives appeared in ACs.
 return ($\{Interrogative\}$)

procedure ScoreDoc1(D, ACs)
 return ($\sum_{i \in \mathrm{INTERROGS}(ACs)} \max_{\langle D,i,AC,Sr,Sw\rangle \in ACs} Sw$)

procedure ScoreDoc2(D, ACs)
 return ($\sum_{i \in \mathrm{INTERROGS}(ACs)} \max_{\langle D,i,AC,Sr,Sw\rangle \in ACs} Sr$)

main
 $DSs \leftarrow \{\}$
 for each $D \in$ Docs(ACs)
 do $\begin{cases} \textbf{if } Strategy == 1 \\ \quad \textbf{then } DSs \leftarrow DSs \bigcup \{\langle D, \mathrm{ScoreDoc1}(D, ACs)\rangle\} \\ \quad \textbf{else if } Strategy == 2 \\ \quad \textbf{then } DSs \leftarrow DSs \bigcup \{\langle D, \mathrm{ScoreDoc2}(D, ACs)\rangle\} \end{cases}$

4. EXPERIMENTAL RESULT

We conducted four runs as shown in Table 3. The difference among the runs is due to the scoring strategy and the parameter settings of the question-answering system. The setting of common parameters of the question-answering system, which are described in Table 1, is shown in Table 2. It should be noted that the value of parameter 'a' represents the number of answers to be searched and it is almost same as the number of document to be scored.Therefore, we have much smaller number of retrieved documents than usual experiments of information retrieval.

The overall evaluation result is summarized in Table 4. Per-topic evaluation results are shown in Figures 3, 4, and 5.

Table 2: Common parameter settings of the question-answering system

d	pwin	ppdoc
250	3	3

Table 3: Submitted runs

Run ID	Strategy	a	p
FORST-JA-JA-01-D	1 (weighted score)	10	30
FORST-JA-JA-02-D	2 (raw score)	10	30
FORST-JA-JA-03-D	1 (weighted score)	20	60
FORST-JA-JA-04-D	2 (raw score)	20	60

Table 4: Mean of each evaluation metrics

Run ID	mean AP	mean Q	mean nDCG
FORST-JA-JA-01-D	0.233	0.259	0.332
FORST-JA-JA-02-D	0.286	0.284	0.372
FORST-JA-JA-03-D	0.206	0.238	0.324
FORST-JA-JA-04-D	0.276	0.287	0.377

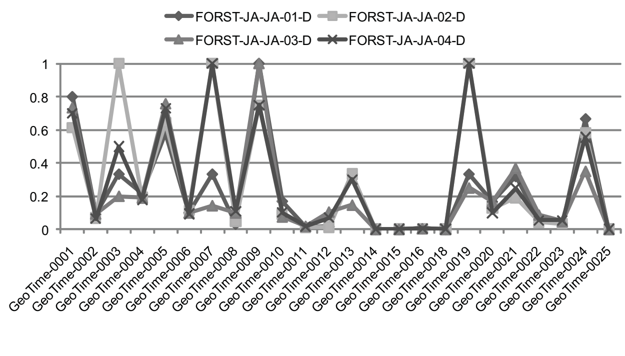

Figure 3: Per-topic Average precision (AP)

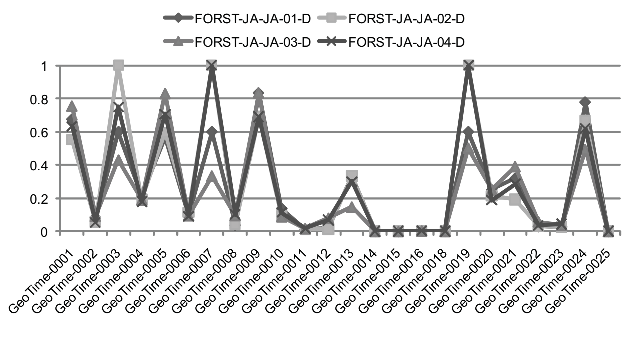

Figure 4: Per-topic Q

5. DISCUSSION

According to Table 4, Strategy 2 (raw score) is superior to Strategy 1 (weighted score). On the other hand, the parameter settings of question answering do not seriously affect to the effectiveness in GeoTime retrieval. There are no statistically significant difference among runs in terms of any evaluation metrics according to the Wilcoxon matched pairs signed rank sum test.

In per-topic evaluation results shown in Figures 3, 4, and 5, we can find there are some topics that cannot be appropriately handled by our method, and therefore the method lacks in robustness in terms of variety of queries.

Especially, the question decomposition module we implemented failed to decompose GeoTime questions into sets of simple questions in following cases.

- Failures because of lack of patterns.
  For example, GeoTime-0010, GeoTime-0018.

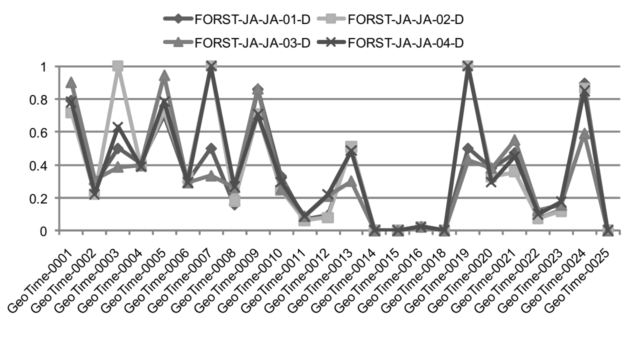

Figure 5: Per-topic nDCG

- Failures because the given questions consist of two separate questions, which cannot be handled by our question-answering systems.
  For example, GeoTime-0015, GeoTime-0020, GeoTime-0023.

6. CONCLUSIONS

In this paper, we proposed a method of GeoTime information retrieval based on based on question decomposition and question answering. We demonstrated that the proposed method is able to accept GeoTime questions and retrieve relevant documents to some extent. However, there is still room to improve the effectiveness of retrieval.

7. REFERENCES

[1] C. L. Clarke, G. V. Cormack, and T. R. Lynam. Exploiting redundancy in question answering. In *Proceedings of SIGIR '01: the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 358–365, 2001.

[2] F. Gey, R. Larson, N. Kando, J. Machado, and T. Sakai. NTCIR-GeoTime Overview: Evaluating Geographic and Temporal Search. In *Proceedings of the Eighth NTCIR Workshop Meeting*, June 2010.

[3] IREX Committee, editor. *Proceedings of IREX workshop*. IREX Committee, 1999. (in Japanese).

[4] T. Mori. Japanese question-answering system using A* search and its improvement. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(3):280–304, Sept. 2005.

[5] T. Mori, M. Nozawa, and Y. Asada. Multi-answer-focused multi-document summarization using a question-answering engine. *ACM Transactions on Asian Language Information Processing (TALIP)*, 4(3):305–320, Sept. 2005.

[6] M. Murata, M. Utiyama, and H. Isahara. Use of multiple documents as evidence with decreased adding in a japanese question-answering system. *Journal of Natural Language Processing*, 12(2):209–247, Mar. 2005.

[7] T. Sakai, N. Kando, C.-J. Lin, T. Mitamura, H. Shima, D. J. K.-H. Chen, and E. Nyberg. Overview of the ntcir-7 aclia ir4qa task. In *Proceedings of the Seventh NTCIR Workshop Meeting*, Dec. 2008.

[8] J. Xu, A. Licuanan, and R. Weischedel. TREC2003 QA at BBN: Answering definitional questions. In *Proceedings of the twelfth Text Retrieval Conferene (TREC 2003)*, 2003.