IEICE TRANS. INF. & SYST., VOL.E97–D, NO.4 APRIL 2014

PAPER Special Section on Data Engineering and Information Management

Revision Graph Extraction in Wikipedia Based on Supergram Decomposition and Sliding Update*

Jianmin WU[†a)], ***Nonmember and*** **Mizuho IWAIHARA**[†b)], ***Member***

SUMMARY As one of the popular social media that many people turn to in recent years, collaborative encyclopedia Wikipedia provides information in a more "Neutral Point of View" way than others. Towards this core principle, plenty of efforts have been put into collaborative contribution and editing. The trajectories of how such collaboration appears by revisions are valuable for group dynamics and social media research, which suggest that we should extract the underlying derivation relationships among revisions from chronologically-sorted revision history in a precise way. In this paper, we propose a revision graph extraction method based on supergram decomposition in the document collection of near-duplicates. The plain text of revisions would be measured by its frequency distribution of supergram, which is the variable-length token sequence that keeps the same through revisions. We show that this method can effectively perform the task than existing methods.

key words: *Wikipedia, collaboration, revision history*

1. Introduction

In recent years, social media becomes more and more attractive to many people since it involves means of interactions among people in which they create, share, exchange and comment contents among themselves in virtual communities and networks [3]. As a collaborative project, online encyclopedia Wikipedia receives contribution from all over the world [13] and its content is well accepted by those who want reliable social news and knowledge.

Guiding by the fundamental principle of "Neutral Point of View", Wikipedia articles need plenty of extra editorial efforts other than simply content expanding and fact updating. Users can choose to edit on an existing revision and override the current one or revert to a previous revision. However, there is no explicit mechanism in Wikipedia to trace such derivation relationship among revisions, while the trajectories how such collaboration appears in Wikipedia articles in terms of revisions are valuable for group dynamics and social media research [12]. Also, research exploiting revision history for term weighting [2] requires clean history without astray, which can be accomplished by such trajectories.

Manuscript received July 11, 2013.
Manuscript revised October 30, 2013.

[†]The authors are with Waseda University, Kitakyushu-shi, 808–0135 Japan.

*This work is based on an earlier work: Revision graph extraction in Wikipedia based on supergram decomposition, In Proceeding of Joint International Symposium on Wikis and Open Collaboration, ACM, Hong Kong, 2013.

a) E-mail: Jianmin.wu@moegi.waseda.jp
b) E-mail: Iwaihara@waseda.jp

DOI: 10.1587/transinf.E97.D.770

Wikipedia now keeps all the versions' contents for each article and make the edit history publicly available. The meta-data of the edit history, such as timestamps, contributors, and edit comments is also recorded. Figure 1 shows a snapshot of typical Wikipedia edit history, which consists of article revision content and meta-data. Most existing research modeling Wikipedia's revision history choose tree [7], [18] or graphs [12] to represent the relationship, but few of them concern about the accuracy of their models.

We propose a method to model such trajectories as revision graphs, where revisions and their derivation relationships are well presented [19], from chronologically-sorted revision history. We extract these directed acyclic graphs based on supergram decomposition. The revision graph can be outputted in the XML-based "GraphML" format [24] and visualized by existing graph editor software, as shown in Fig. 2. This paper is extending from [20] by incorporating sliding update to improve efficiency, also detailed performance studies are carried out.

For a given revision r, at least one parent revision r_p should be identified in r's previous revisions, which involves comparison with those revisions. The best candidate should be decided by a certain similarity measure as well as the characteristics of Wikipedia editing. We consider the challenge of revision graph extraction (***RGE***) is twofold. The first part is the strategy that should be adopted to perform

Fig. 1 Typical edit history of Wikipedia.

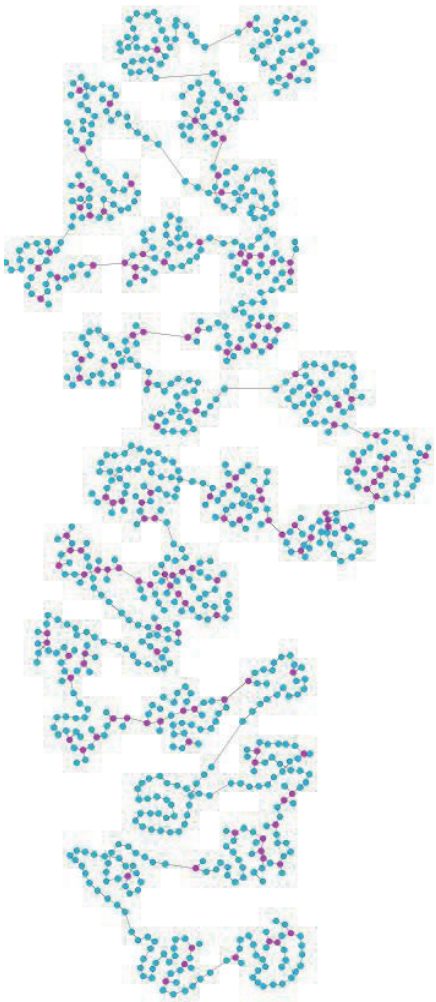

Fig. 2 Revision graph for first 1000 revisions of Wikipedia article "Edith Wharton," pink nodes indicate where branch happens.

comparisons over previous revisions. The second part is the existing differencing method and similarity metric for pairwise comparison.

By such description, RGE is close to the classic nearest neighbor search (NNS) problem in text mining: given a collection of n points, build a data structure which, given any query point, reports the data point that is closest to the query [1]. However, RGE differs from NNS in problem setting. First, the dissimilarity among a revision set is much lower than the text corpus that conventional NNS deals with [15]. The most popular algorithm for NNS, LSH-based algorithms [10], [11] would fail to distinguish among such near-duplicates texts. Moreover, the existence of the timestamps in the meta-data suggests that sequential relationship among revisions should be exploited.

There is another issue we should notice. The overview of Wikipedia mining [14] shows that the text amount of diff between two adjacent revisions is not proportional to the length of the article, that is, users would not contribute more text because of a longer article. With the relatively stable edit contribution amount, the longer an article grows, the less difference can be told by Jaccard distance, which suggests that we need absolute measure.

The remaining part of this paper is organized as follows: In Sect. 2 we introduce existing work related to our research. In Sect. 3 we explain the motivation and basic process of supergram decomposition. We extend the model in Sect. 4 by exploiting dependencies among revisions and narrowing down comparison scope for scalability. Section 5 evaluates the result generated by our method and compare with other representative methods by performance and precision evaluation. Finally we conclude our paper by summarizing findings and discussing several key issues.

2. Related Work

As mentioned before, a revision history modeling method should include two parts: comparison strategy and text differencing method with similarity metric. Most existing work focused on the second part. Fong et al. [9] proposed a detailed text differencing algorithm that finds all the different parts, including the case of phrase movement and sentence re-writing, between two given revisions based on hierarchical decomposition and the longest common subsequence (LCS) method, which is however way too computationally expensive in terms of large scale revision comparison.

In an investigation on structure and dynamics of Wikipedia's breaking news collaborations [11], Keegan et al. construct article trajectories of editor interactions as they coauthor an article. Examining a subset of this corpus, their analysis demonstrates that articles about current events exhibit structures and dynamics distinct from those observed among articles about non-breaking events. However, the similarity metric adopted in this research is oversimplified and the correctness of the trajectories they build is not assured.

In [8], Flöck et al. reconsider the algorithm for detecting reverts, which is an important role of revision modeling. This work refines the original definition of "revert" and proposes an algorithm based on the mixture model of paragraph alignment and comparison by LCS, slightly simplified than [9]. This model still cannot achieve a precise text differencing result due to the LCS.

Cao et al. [5] proposed a version tree reconstruction method for Wikipedia articles based on keyword clustering. This method uses tf-idf (term frequency and inverted document frequency) score to cluster similar revisions and then LCS would be used for more precise comparison, which is closer to string matching problem. The initial clustering suffers from a fixed number of signature values.

Wu et al. [19] proposed a revision graph extraction method for Wikipedia articles based on n-gram cover. An n-gram is a consecutive occurrence of n letters or words in a text. This research uses word-level n-gram distribution to denote revisions of the given articles with timestamps and find how a revision's n-gram distribution can be formed by

specific previous revisions'. But this method still suffers from error rate due to the plain model of *n*-gram diff score.

3. Supergram Decomposition

A *revision set* $\boldsymbol{R}$ is a set of revisions $\{r_1, r_2, \ldots, r_n\}$, where each revision has a timestamp. A *timestamp ordering* $r_i < r_j$ is a total ordering on their timestamps, meaning that r_i's timestamp is earlier than r_j's. The revision graph extraction problem on $\boldsymbol{R}$ is to find a directed acyclic graph (DAG) where nodes are revisions and edges $<r_i, r_j>$ are such that $r_i < r_j$ holds and r_j is created directly from r_i. We call r_i as r_j's *revision parent*. In general, a revision may have multiple parents due to merge of revisions and the revision graph is a DAG. But empirically such merges are rare, and in this paper we focus on the case where only branches happen.

Based on the characteristics of Wikipedia editing, we assume that the best candidate for r_j is the one that takes least efforts to convert to r_i. More specifically:

a) Adding takes more efforts than deleting.

b) Long edits take more efforts than short edits.

c) Multiple short edits take more effort than a single long edit

These assumptions require that the diff generated by certain method should reflect the integrity of the original text order as much as possible.

As the further research of [19], we carefully consider the model of *n*-gram cover. The *n*-gram frequency comparison method in *n*-gram cover model is from the shingling method, which has been a conventional method in NNS [4], [16]. In *n*-gram cover, only the different text among revisions has been noted and measured. Diff caused by edit behaviors will be detected as changes in *n*-gram frequency distribution. Although the positional information among tokens can be reserved partially by longer shingle (bigger *n*), the integrity of different edits cannot be recovered. On the other hand, it takes O(*MN*) time to achieve integrity by the LCS-based diff algorithm, where *M* and *N* are the total number of tokens in each revision. Although each article in Wikipedia English poses 136.7 revisions in average [21], the number of revisions often exceeds one thousand in popular articles. In such a situation, pairwise comparison on *X* revisions by certain measures requires $O(X^2)$ comparisons, which make full comparisons too expensive.

We find that there are some token sequences that keep appearing throughout the whole revision set. For a small revision set of several revisions, such token sequences is little but with long length. As the size of the revision set grows larger, long token sequences are split into shorter fragile due to modifications. Formally, we define such units as:

Definition 3.1 (Supergram)

A *supergram* $s = t_1 t_2 \ldots t_n$ in a revision subset $C \subseteq \mathbf{R}$, where *C* is called a *comparison scope*, is an *n*-gram ($n >= 2$) such that *s* occurs in all the revision in *C*.

Ex. 3.1 Given the following revisions

R_1: I am iOS device user.

R_2: I am a core iOS device user.

R_3: I am a light iOS device user.

R_4: Of course I am an iOS device user.

R_5: I am an iOS device user of course.

"*I am*" and "*iOS device user*" are supergrams, since they both keep the same through R_1 to R_5 against other changes.

3.1 Word Transition Graph Construction

Given an article's revision set *R*with revisions $r_1, r_2, \ldots, r_n$, each of them consists of tokens from a vocabulary $D = \{t_1, t_2, \ldots, t_l\}$. In the following paragraphs, we denote

- v_i: vertex *i* labeled with t_i;
- $<v_i, v_j>$: edge *x* from v_i to v_j;
- $w(v_i, v_j)$: weight of $<v_i, v_j>$, labeled with the collection frequency of bigram $t_i t_j$;
- $out(v_i)$: set of all edges from v_i;
- $in(v_i)$: set of all edges to v_i.

Definition 3.2 (Word transition graph)

Given a revision set *R* on vocabulary *D*, a **word transition graph (WTG)** $G = (V, E)$ is a directed weighted graph such that each vertex $v_i \in V$ denotes a term $t_i \in D$. For two terms t_i *and* $t_j \in D$, a weighted directed edge $e(v_i, v_j) \in E$ exists between their corresponding vertices v_i and v_j if and only if the bigram $t_i t_j$ has a frequency $f(t_i t_j) > 0$ in *R*, and $f(t_i t_j)$ is assigned as the edge weight.

The word transition graph is allowed to contain cycles since a multiple appearance of frequent terms causes a path that starts and ends at the same vertex, as shown in Fig. 3. On the other hand, there exist chain-like subgraphs at which only one path exists, which correspond to Definition 3.1. Here we define such structure formally:

Definition 3.3 (Chain)

A *chain Q* is a sequence of edges $<v_1, v_2>$, $<v_2, v_3>$, .., $<v_{n-1}, v_n>$ ($n \geq 3$) in G such that $v_1, .., v_n$ are distinct, and each middle vertex, called a *chain vertex*, v_i ($1 < i < n$) has only one incoming edge and one outgoing edge, i.e. $|out(v_i)| = |in(v_i)| = 1$. The rest vertices in G are called *non-chain vertices*.

Given a revision r_i, and r_i's comparison scope C_i, a word transition graph *G = (V, E)* will be constructed for r_i by scanning all revisions within $C_i \cup \{r_i\}$.

```
FOR r' ∈ (C_i ∪ { r_i })
  FOR t_k ∈ r'
    IF t_k ∈ V
      w(v_{k-1}, v_k) ← w(v_{k-1}, v_k) +1
    ELSE
      add <v_{k-1}, v_k> in E
      w(v_{k-1}, v_k) ← 1
```

Since each token t_k is scanned only once, the construction time is proportional to the total token number *L* in the revisions within $C_i \cup \{r_i\}$, i.e. O(L) time. After G is constructed, the indegree and outdegree of each vertex is clear

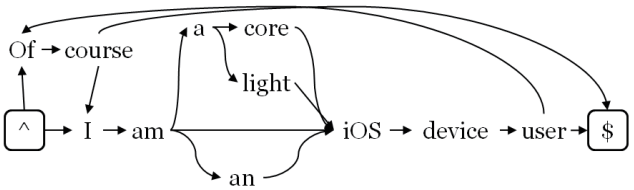

Fig. 3 Word transition graph for Ex. 3.1.

so that the chain vertex set V_{chain} and non-chain vertex set V_{non} can be identified. Another scan will be executed to extract supergram that between chain vertices. Then each revision within $C_i \cup \{r_i\}$ can be decomposed according to the extracted supergrams and get the supergram frequency distribution.

4. Sliding Update and Diff Evaluation

We start this section by the full process of revision graph extraction based on supergram decomposition with sliding update on revision collection *R*. The comparison stage consists of 5 stages:

1. Pre-processing
   For each revision $r_i \in R$, generate its unigram token sequence.
2. Comparison scope computing
   Calculate r_i's *comparison scope* C_i based on r_i's timestamp.
3. Sliding decomposition
   a. If i = 0, construct a word transition graph G_i of all the revisions within C_i,decompose r_i and all revisions in C_i based on the supergram set S extracted from G_i.
   b. If i > 0, perform *sliding update* for $G_{i-1}$and the supergram frequency distribution.
4. Supergram diff score computing
   Compare r_i with all revisions in C_i by a diff score defined on supergrams.
5. Candidate selection
   For r_i, Pick up the revisions with lowest k supergram diff score as the candidates for parents.

The following figure shows the work flow of decomposition based on sliding word transition graph.

4.1 Pre-Processing

We first split the original revision text into a unigram token sequence. The text content in the original revision files contains plenty of *Wiki Markups* [23], which give specific metadata tags on plain text. While splitting the text, such markups are extracted by regular expression and will be reserved as single tokens in the following steps. The second task is replacing the URLs appearing in the text. No matter how many terms a URL involves, it has no more contribution to add a new URL than to add a single word. We replace each URL with a 16-byte string generated by MD5 for consistency.

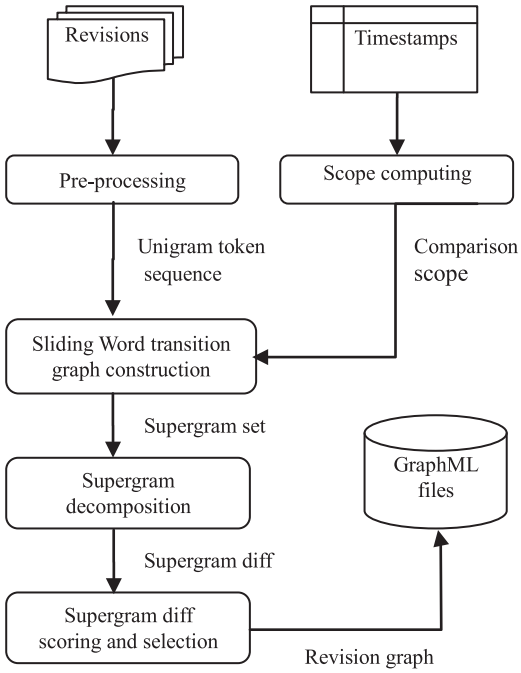

Fig. 4 Full process of supergram decomposition by sliding update.

4.2 Comparison Scope Computing

Recall the observation of supergrams we mentioned before: a narrower scope will produce longer supergrams. This is because the number of edits is proportional to the scope size and fewer edits mean smaller chances, and supergrams tend to be undivided. Longer supergrams are preferable in supergram decomposition because it reserves more integrity and reduces the total number of supergrams.

We can draw the assumption based on the characteristics of Wikipedia editing: The further one revision is from the current revision, the less possible that the current one is derived from that revision.

But before we limit the comparison scope to a fixed number of previous revisions, we consider the frequent edit behavior within a certain period of time as another important factor according to the timestamps in the edit history's meta information. Intense editing activity could be caused by edit wars, increasing popularity of the article, or immediate updates after related events happen, and the total number of edits in a week could easily exceed any preset number. Figure 5 shows the edit count of Wikipedia article "Barack Obama" during 2008, the year of the U.S. presidential election, and significant peak can be found in November, when the election was held. Thus, all the previous revisions within certain time span should be examined, regarding the fact that contributors' attention can last for a period of time.

A fixed scope would not be able to capture the whole process of the intense edit activity, while fixed time span can cover only little revisions. Considering such trade-off, we employ *maximum comparison scope* to denote the largest number of previous revisions to be compared, which is de-

Month	edits	Minor edits	(%)
01/2008	490	112	22.9
02/2008	712	191	26.8
03/2008	715	185	25.9
04/2008	988	198	20.0
05/2008	766	133	17.4
06/2008	793	192	24.2
07/2008	372	83	22.3
08/2008	475	140	29.5
09/2008	467	141	30.2
10/2008	637	207	32.5
11/2008	1434	451	31.5
12/2008	344	70	20.3

Fig. 5 Edit count of Wikipedia article "Barack Obama" during 2008, the year of the U.S. presidential election[†].

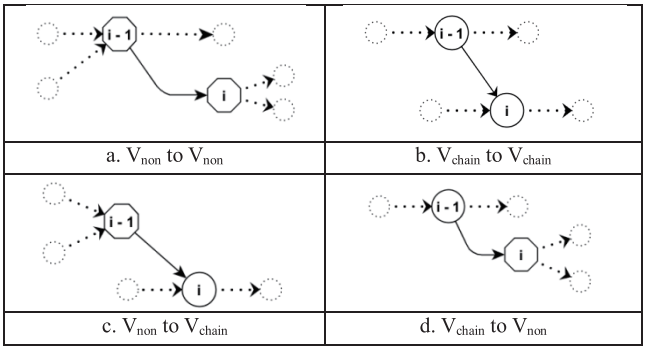

Fig. 6 Examples of chain breaking situations.

fined as below.

Definition 4.1 (Maximum comparison scope)
Given a revision history $H = \{(r_1, ts_1), (r_2, ts_2), \ldots, (r_m, ts_m)\}$, where (r_i, ts_i) denote a revision r_i with its timestamp ts_i, the maximum comparison scope C for revision r_k is the collection of revisions determined by either:
a) $C = \{r_{k-S_L}, r_{k-S_L+1}, \ldots, r_{k-1}\}$, if $t_{S_{k-S_L}} - t_{S_k} > T_l$, other wise
b) All revisions within T_l.

where S_L denotes the minimum revision number to ensure enough comparison for unpopular documents, T_l denotes the minimum time span to avoid the limit of fixed number of revision during the periods of intense edits. For example, for a popular article, the minimum revision number S_L will not be applied because we might loss important revert or rollback by stopping at the middle.

Notice that there could be a series of consecutive edits by the same contributor. In this case, we take only the latest revision and omit the others, since we focus on the collaborative authoring and editing process rather than individual perspective.

Another issue we should notice is the phenomenon of *remote copy*, which is the behavior that copying a piece of text from an ancient revision such that there is no appearance of such text within the scope of *Maximum comparison scope*. Simply expanding the scope to that ancient revision includes unnecessary revisions and lowers the efficiency. We choose to include this kind of ancient revision as individual revision alone. Formally, an ancient revision is identified as follows:

A revision r_j is a potential remote ancestor of r_i if and only if there is a bigram b_k that appears in r_j and r_i but not in revisions between r_j and r_i.

4.3 Sliding Update

So far for each revision, a word transition graph will be constructed and the supergram frequency distribution according to the graph can be calculated. Notice that the adjacent comparison scopes are mostly overlapped, i.e. most of their revisions are the same, it is much efficient to update the different revisions upon an existing transition graph than to construct a new transition graph for all revisions in the later comparison scope. Moreover, by update the changed supergram frequency of the updated transition graph, we do not have to perform supergram decomposition for overlapped revisions again. Such updates on transition graph and supergram distribution is called *sliding update*.

The sliding update considers two operations: adding the new revision, and deleting the revisions that are out of the new scope. The adding operation on WTG is basically the same with the construction of WTG but with more concerns on the update for supergram distribution. For each token t_i that is read from the new revision, if edge $<t_{i-1}, t_i>$ exists in the WTG, we should simply update $w(t_{i-1},t_i)$ and the corresponding supergram frequency. But if not, a new edge is created and a new bigram of $(t_{i-1}t_i)$ will be added into supergram distribution. The new edge $<t_{i-1},t_i>$ connects vertices in the following situations:

a. Non-chain vertex to Non-chain vertex chain vertex.
b. Chain vertex to chain vertex.
c. Non-chain vertex to chain vertex.
d. Chain vertex to non-chain vertex.

Figure 6 shows examples of the four situations, where a circle indicates a chain vertex and an octagon indicates a non-chain vertex. Also, since in situation b, c and d the chain property of existing chain vertex has been violated by the new edge, the involved chain has to be broken and an update on the corresponding supergram in the distribution is needed. For a chain $H = <v_1,v_2>, <v_2,v_3>, \ldots, <v_i,v_{i+1}> .., <v_{n-1},v_n>$ $(1 < i < n, n \geq 3)$ that breaks at v_i, the existing entry of the corresponding supergram $s = (t_1 \ldots t_i \ldots t_n)$ in the supergram distribution will be replaced by the entry of $(t_1 ... t_i)$ and $(t_i \ldots t_n)$ with the same frequency.

To delete a revision from the transition graph is the inverse operation of adding. The edge weights will be subtracted according to the bigram frequency in the deleted revision. When the weight of an edge turns to 0, that edge should be deleted. Notice that the involved vertices might regain the chain property likewise, so an update of merging entry in the supergram distribution should be needed.

After the sliding update, the word transition graph is ready for further updates and the supergram frequency distribution can be used for pairwise comparison.

[†]http://en.wikipedia.org/wiki/Barack_Obama

4.4 Supergram Diff Score Computing

For pairwise revision comparison, we first create the *supergram diff* for two revisions, and then calculate the *supergram diff score* to measure their difference.

Definition 4.2 (Supergram diff)
Given a supergram set S, we denote the supergram frequency distribution of revision r_a as $f(s_i, r_a)$ ($s_i \in S$). For two revisions r_a and r_b, the supergram diff SD is the set of supergrams with a non-zero residual frequency between r_a and r_b:

$$SD(r_a, r_b) = \{s \in S \,|\, |f(s, r_a) - f(s, r_b)| > 0\} \quad (1)$$

Definition 4.3 (Supergram diff score)

$$diffScore(r_a, r_b) = w_1 \cdot \sum_{s \in SD_{add}} |f(s, r_a) - fs, \\ rb_{\cdot|S|} + w2 \cdot s' \in SDdelfs', \ ra - fs', \ rb.log|s'| \quad (2)$$

where SD_{add} is the set of all supergrams such that $f(s, r_a) - f(s, r_b) > 0$, and SD_{del} is defined similarly, w_i is the weight for discrimination between adding and deleting operations.

We set $w_1 = 0.65$ and $w_2 = 0.35$ empirically to maximize the difference. As heuristics, the logarithms are to the base of 10, since the deleting operation is a less effort-taking job.

4.5 Candidate Selection and Output

After the calculation of supergram diff score has been done, revisions will be ranked by their scores. Basically, the one with the lowest score will be considered as the parent revision. However, sometimes there could be multiple candidates with the same score. In that case, we select the latest one, since.

Once every revision's parent revision has been found, the revision graph is finished. We choose to serialize the revision graph in a common format for graph, "GraphML". Then the graph can be utilized by other software or programs.

5. Experimental Evaluation

To evaluate the precision and performance, we conduct two accuracy evaluations and an execution time test on the proposed method with 4 representative methods: sentence-level Jaccard distance [18], keyword clustering [5], n-gram cover [19] and the conventional token-level Edit distance. For each method, we compare its result revision graphs with manually extracted revision graphs on the dataset of selected Wikipedia articles.

5.1 Dataset

Long article with many revisions are good for distinguishing the performance and effectiveness of methods in the RGE problem because they involve more edit behavior. We enlist 50 articles in Wikipedia that satisfy the following basic criteria:

- In English Wikipedia.
- Has at least 300 tokens in the latest revision.
- Has at least 200 revisions.

More precisely, we select the articles by four categories:

- Random articles (RA): We use the official "Random article" function on Wikipedia page and pick up the qualified articles in order to be non-bias sampling. RA represents the majority of Wikipedia article and all methods are supposed to work with them.
- Featured (FA) and non-featured articles (NFA): An article with a "Feature" label in Wikipedia is the symbol of high quality content, which requires much more effort than original articles [17]. We adopt the article list from WPQAC (Wikipedia Quality Assessment Corpus) [6] as the second part. WPQAC is constructed for evaluating relationship between the writing process and article quality. It includes 10 pairs of comparable FA and NFA that cover a broad quality spectrum, each with nearly identical text volume (in characters) and edit frequency.
- Top-edited articles: We select the top 15 articles with the most revisions in English Wikipedia, according to the official statistics [22]. Such articles include all kinds of edit behaviors.

The titles of the articles in the dataset are listed in Table 1, with the general statistics including their total number of revision (**Rev. #**), total number of unique contributors (**Contributor #**, unregistered users' number is shown in the parentless), the average token number of each revision (**Avg. Token #**) and the number of branches (**Br. #**) in the manual graph. The articles in the dataset cover topics that vary from political, historical issues to bibliography. The right part of the table will be addressed in Sect. 5.3.

For each article, we dump the first 200 revisions from the official export site, with an exception that the article of "1941 Atlantic hurricane season" has only 149 revisions. We perform the manual extraction by a 3-researcher team and build up a ground truth dataset of 9949 revisions involving more than 14 million tokens

5.2 Precision Evaluation

In this evaluation, all the revisions have been pre-processed according to Sect. 4.1 so that all methods start with the same token sequence. Each compared method adopts the default parameter and initial setting, and the comparison scope for each revision is all of its previous revisions.

The parent accuracy is evaluated as the percentage of the revisions that has the same parent revision as in the ground truth revision graph, which is shown in Fig. 7.

We evaluate branching errors that happen in different stages by reachability comparison. Given two revision

Table 1 Article list and Execution time test.

Article Category	ID	Article Title	Rev. #	Contributor #	Avg. Token #	Br #	Time		
							n-gram cover	Supergram	Edit distance
Random Articles	1	Racism	11152	4773 (2651)	1465.2	23	13.9	35.6	502.5
	2	Hello	2886	1654 (743)	155.3	40	2.1	5.6	12.5
	3	PhpBB	1364	652 (330)	1420.7	37	10.9	27.8	316.6
	4	Edith Wharton	1274	672 (331)	1914.6	23	13.9	35.2	495.5
	5	Trailer (promotion)	1178	714 (364)	1723.6	42	13.9	34.7	507.6
	6	Federal republic	842	531 (311)	312.4	43	1.9	5.9	10.6
	7	Martin St. Louis	758	433 (218)	813.1	32	5.0	12.8	101.3
	8	French campaign in Egypt and Syria	655	329 (132)	1241.1	22	5.5	14.4	127.3
	9	Sarkar Raj	627	315 (199)	340.4	15	5.6	13.9	104.3
	10	Grade inflation	576	298 (177)	1554.4	24	13	33.2	469.5
	11	2006 Israel–Gaza conflict	542	193 (62)	612.6	12	7.7	19.9	177.4
	12	Altai Mountains	479	303 (113)	1338.9	15	6.7	17.6	169.1
	13	Natal chart	386	191 (87)	556.5	11	6.3	16.2	120.9
	14	Muhammad Naguib	332	226 (80)	2046.7	8	13.2	33.7	607.3
	15	Clarinet Concerto	291	156 (54)	1222.8	12	9.8	25.0	297.1
Featured Articles	16	Dactylic hexameter	227	143 (57)	1606.9	21	7.4	19.3	185.3
	17	William de Corbeil	417	74 (13)	1805.3	6	9.1	22.8	302.4
	18	Victoria Cross (Canada)	513	176 (60)	834.6	19	7.5	19.9	154.5
	19	Deinosuchus	737	293 (109)	636.0	21	4.2	11.1	98.8
	20	Winfield Scott Hancock	773	382 (166)	1943.5	32	4.8	12.0	108.5
	21	Laplace-Runge-Lenz vector	578	135 (38)	2006.9	6	12.3	31.4	416.8
	22	Introduction to general relativity	871	293 (108)	1697.7	14	10.1	26.2	342.6
	23	U.S. Academic Decathlon	1464	393 (230)	1455.9	22	9.5	24.4	311.9
	24	Song Dynasty	3996	1142 (614)	1612.1	32	11.0	28.1	424.5
	25	Euclidean algorithm	1749	524 (237)	824.2	24	12.5	31.8	462.3
Non-featured Articles	26	1941 Atlantic hurricane season	145	49 (2)	1494.6	2	5.7	15.1	108.7
	27	European Liberal Democrat and Reform Party	661	315 (128)	921.7	9	6.3	16.5	139.2
	28	Erlang (programming language)	938	572 (301)	668.4	16	4.5	11.7	109.8
	29	Intel 8086	953	570 (309)	707.5	33	4.3	11.9	67.7
	30	Dhole	972	536 (242)	1334.3	35	7.4	19.0	162.6
	31	United Nations Relief and Works Agency	871	291 (84)	1675.2	28	8.6	21.4	232.3
	32	Subwoofer	1271	596 (339)	1686.7	13	8.8	22.7	206.0
	33	John Cage	2064	1061 (529)	2255.2	20	13.1	33.3	510.9
	34	Haile Selassie I	4111	1541 (776)	967.1	12	13.5	33.7	605.1
	35	United Methodist Church	2047	732 (358)	1354.2	21	10.4	27.0	366.1
Top-edited Articles	36	George W. Bush	45268	14268 (6382)	1470.0	22	8.0	20.7	229.0
	37	List of WWE personnel	38152	4386 (2647)	1057.1	17	5.4	13.8	76.0
	38	United States	32504	9296 (3596)	1261.2	29	7.4	19.3	149.4
	39	Wikipedia	32071	12809 (6468)	1786.7	28	7.6	19.6	205.3
	40	Michael Jackson	27048	6394 (2076)	1639.4	15	8.5	21.5	245.1
	41	Jesus	26589	6565 (2508)	1304.0	17	7.4	18.9	160.1
	42	List of Total_Drama_characters	25448	5667 (4622)	3568.6	19	16.2	40.9	786.5
	43	Catholic Church	23891	5511 (3160)	3314.3	26	16.8	42.7	891.5
	44	Adolf Hitler	23501	8007 (3213)	1335.4	32	6.0	15.1	149.3
	45	Barack_Obama	23505	6264 (1048)	1222.8	16	6.1	16.0	100.4
	46	Britney Spears	23048	9456(4465)	2006.9	42	12.3	31.3	406.6
	47	World War II	22654	7153(3621)	1697.7	43	6.3	15.7	179.6
	48	List of programs broadcast by ABS-CBN	21724	6103(1392)	1455.9	32	4.3	11.8	99.5
	49	The Beatles	21543	8435(5672)	1612.1	24	5.4	13.7	134.5
	50	Wii	21521	5983(2671)	1338.9	12	7.1	18.3	186.5

graphs G_1, G_2 on the same revision set D, the *reachability accuracy* of G_2 on G_1 is defined as follows:

$$C(G_1, G_2) = \frac{2\left|G_1^+ \cap G_2^+\right|}{|D|^2} \quad (3)$$

where G_1^+, G_2^+ are the transitive closures of G_1, G_2, $|D|^2/2$ is half the number of all the node pairs. By formula (3) we focus on how far (in terms of number of total descant revisions) an error can reach, so errors that happen in the early stage or those that involve more succeeding revisions have greater loss in accuracy.

In both evaluations, the proposed method prevails on most of the test articles, as we can see from the scatter plot and the average accuracy. The Edit distance method achieve the second best position because it can separate more intact diff, but it fails to deal with the case of text movement and expansion. The n-gram cover method results in more false-positive branches because it treats the deletion content

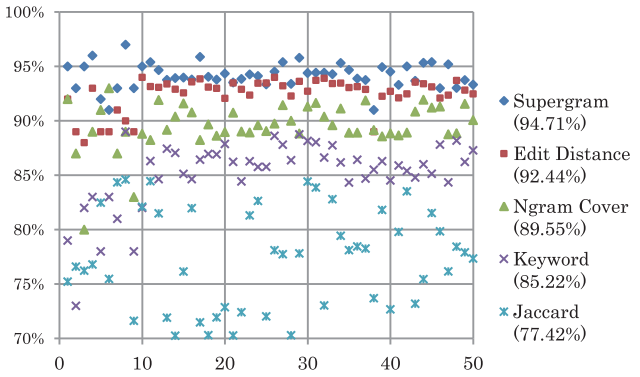

Fig. 7 Parent accuracy result.

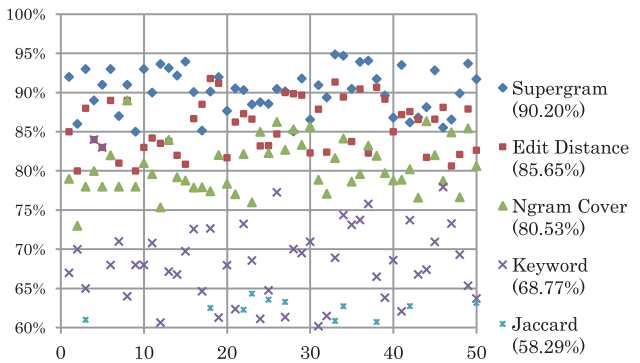

Fig. 8 Reachability accuracy.

as the same as the adding content and fails to handle those revisions that both addition and deletion happen. The keyword clustering method performs worse than n-gram cover on most articles with more false-negative branches. The Jaccard distance method has the most false-negative branches in the later stage of the revision set, since the relative difference is too small to distinguish a branch. All methods fail to choose the nearest revision as the parent for those severe vandalism cases of heavily deletion or even full text deletion.

Regarding the difference among article categories, the only finding is that the proposed method has slightly bigger advantage to Edit distance method on RA and TA categories than on FA and NFA.

5.3 Execution Time Test

We also run an execution time test for 3 methods with the top precision, Edit distance, n-gram cover and the proposed method.

Each method is implemented in Java with default parameter setting and executed in the same environment. Revisions are with the same pre-processing equipped with comparison scope based on timestamps so that the execution time will be recorded by the same criteria.

In Table 1, we list up the articles' ID and their average revision token numbers. In each cell under the name of method, the average execution time (by millisecond) that a method takes to determine the parent revision for each revision is shown. The n-gram cover method achieves best efficiency because it treats and evaluates revisions in the simplest way so that the pairwise comparison is performed in a linear time. The proposed method takes about 2.5 times more than n-gram model due to the extra cost of word transition graph construction and supergram decomposition, which is the trade-off between precision and efficiency. The edit distance method's result is much slower than others, given that the time complexity is square time.

6. Conclusion

In this paper, we proposed supergram technique for accurate revision graph extraction from Wikipedia edit history. Revision are decomposed and compared by supergrams, which are extracted from a word transition graph. Our proposed method outperforms existing text comparison methods. In the future, we will investigate further optimization of comparison scopes, and develop applications utilizing extracted revision graphs, such as visualizations.

Acknowledgments

This research was in part supported by "Ambient SoC Global Program of Waseda University" of the Ministry of Education, Culture, Sports, cience and Technology, Japan nd JSPS KAKENHI Grant Number 25330367.

References

[1] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," Commun. ACM, vol.51, no.1, pp.117–122, Jan. 2008.
[2] A. Aji, Y. Wang, E. Agichtein, and E. Gabrilovich, "Using the past to score the present: Extending term weighting models through revision history analysis," CIKM '10, pp.629–638, ACM, New York, NY, USA, 2010.
[3] T. Ahlqvist, A. Bäck, M. Halonen, and S. Heinonen, "Social media roadmaps exploring the futures triggered by social media," VTT Tiedotteita, vol.2454, p.13, 2008.
[4] A.Z. Broder, "On the resemblance and containment of documents," Proc. Compression and Complexity of Sequences, pp.21–29, Positano, Italy, 1997.
[5] Z. Cao and M. Iwaihara, "Wikipedia version tree reconstruction by clustering revisions through keywords," IEICE Technical Report, DE2011-32, 2011.
[6] J. Daxenberger and I. Gurevych, "A corpus-based study of edit categories in featured and non-featured Wikipedia articles," Proc. 24th COLING, pp.711–726, Mumbai, India, 2012.
[7] M. Ekstrand and J.T. Riedl, "rv you're dumb: Identifying discarded work in Wiki article history," WikiSym '09, ACM, New York, NY, USA, 2009.
[8] F. Flöck, D. Vrandečić, and E. Simperl, "Revisiting reverts: Accurate revert detection in Wikipedia," Proc. Hypertext and social media, pp.3–12, ACM, New York, USA, 2012.
[9] P.K. Fong and R.P. Biuk-Aghai, "What did they do? Deriving high-level edit histories in Wikis," WikiSym '10, ACM, New York, NY, USA, 2010.
[10] P. Indyk and R. Motwani, "Approximate nearest neighbor: Towards removing the curse of dimensionality," Proc. STOC '98, pp.614–623, Dallas, USA, 1998.

[11] P. Indyk, "Nearest neighbors in high-dimensional spaces," in Handbook of Discrete and Computational Geometry, pp.877–892, CRC Press, 2003.
[12] B. Keegan, D. Gergle, and N. Contractor, "Staying in the Loop: Structure and dynamics of Wikipedia's breaking news collaborations," WikiSym '12, ACM, New York, NY, USA, 2012.
[13] A. Lih, "Wikipedia as participatory journalism: Reliable sources: Metrics for evaluating collaborative media as a news resource," Proc. Int. Symp. Online Journalism, 2004.
[14] O. Medelyan, D. Milne, C. Legg, and I.H. Witten, "Mining meaning from Wikipedia," Int. J. Hum.-Comput. Stud., vol.67, no.9, pp.716–754, Sept. 2009.
[15] G.S. Manku, A. Jain, and A.D. Sarma, "Detecting near-duplicates for web crawling," WWW '07, pp.141–150, ACM, New York, NY, USA, 2007.
[16] U. Manber, "Finding similar files in a large file system," Proc. USENIX Conference, pp.1–10, 1994.
[17] B. Stvilia, M.B. Twidale, L.C. Smith, and L. Gasser, "Information quality work organization in Wikipedia," Journal of the American Society for Information Science and Technology, vol.59, no.6, pp.983–1001. 2008.
[18] M. Sabel, "Structuring wiki revision history," WikiSym '07, ACM, New York, NY, USA, pp.125–130, 2007.
[19] J. Wu and M. Iwaihara, "Wikipedia revision graph extraction based on n-gram cover," Proc. Int. Workshop on Graph Data Management and Mining, WAIM 2012, pp.29–38, 2012.
[20] J. Wu and M. Iwaihara, "Revision graph extraction in Wikipedia based on supergram decomposition," Proc. Int. Joint International Symposium on Wikis and Open Collaboration, ACM, Hong Kong, 2013.
[21] T. Yasseri and J. Kertész, "Value production in a collaborative environment," Journal of Statistical Physics, vol.151, pp.414–439, Springer US, 2013.
[22] http://en.wikipedia.org/wiki/Wikipedia:Database_reports/Pages_with_the_most_revisions
[23] http://en.wikipedia.org/wiki/Help:Wiki_markup
[24] http://graphml.graphdrawing.org/specification.html

Jianmin Wu received his B.S. degrees in Software Engineering from Nanjing University in 2010. He received his D.Eng. degree from Graduate School of IPS, Waseda University in 2012. He is now a Ph.D candidate in Waseda University.

Mizuho Iwaihara received his B.Eng, M.Eng. and D.Eng. degrees all from Kyushu University, in 1988, 1990, and 1993 respectively. He was a research associate and then associate professor in Kyushu University, from 1993 to 2001. From 2001 to 2009, he was an associate professor at Department of Social Informatics, Kyoto University. Since 2009, he is a professor at Graduate School of IPS, Waseda University. He is a member of IEICE, IPSJ, ACM and IEEE CS.