Revision graph extraction in Wikipedia based on supergram decomposition

Jianmin Wu, Mizuho Iwaihara
Waseda University
Hibikino 2-7, Wakamatu, Kitakyushu, Fukuoka, 808-0135, Japan
jianmin.wu@moegi.waseda.jp, iwaihara@waseda.jp

ABSTRACT
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.

Categories and Subject Descriptors
K.4.3 [**Computers and Society**]: Organizational Impacts – *Computer-supported collaborative work.*

General Terms
Algorithms, Experimentation.

Keywords
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 [2]. As a collaborative project, online encyclopedia Wikipedia receives contribution from all over the world [5] 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 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 [3]. Also, research exploiting revision history for term weighting requires clean history without astray, which can be accomplished by such trajectories.

Wikipedia now keeps all the versions' contents for each article and make the edit history publicly available. Other useful information, such as timestamps, contributors, and edit comments is also recorded. Figure 1.1 shows a snapshot of typical Wikipedia edit history. Most existing research modeling Wikipedia's revision history choose trees or graphs to represent the relationship[3][7], but few of them concern about the accuracy of their models.

```
<page>
  <title>Square-free integer</title>
  <id>29525</id>
  <revision>
    <id>286083</id>
    <timestamp>2002-01-12T12:27:00Z</timestamp>
    <contributor>
      <ip>Georg Muntingh</ip>
    </contributor>
    <comment>*</comment>
    <text xml:space="preserve" bytes="123">An integer ''N'' is called
    squarefree if for every [[primedivisor]] ''p'' of ''N'', ''p'' does
    not divide N divided by p.

</text>
  </revision>
  <revision>
    <id>17754</id>
    <timestamp>2002-02-24T21:29:18Z</timestamp>
    <contributor>
      <ip>Conversion script</ip>
    </contributor>
    <minor/>
    <comment>Automated conversion</comment>
    <text xml:space="preserve" bytes="379">An [[integer]] ''N'' is
    called '''squarefree''' [[iff]] no [[perfect square]] except 1
    divides ''N''. Equivalently, ''N'' is squarefree iff in the
    [[fundamental theorem of arithmetic|prime factorization]] of ''N'',
    no [[prime number]] occurs more than once. Another way of stating
    the same is that for every [[primedivisor]] ''p'' of ''N'', ''p''
    does not divide ''N'' / ''p''.
</text>
  </revision>
  <revision>
  <revision>
  <revision>
```

Figure 1.1 Typical edit history of Wikipedia

In this paper, we propose a method to model such trajectories as revision graphs from chronologically-sorted revision history. We derive these directed acyclic graphs by extracting the underlying derivation relationships among revisions in a precise way, as shown in Figure 1.2. For a given revision *r*, it needs to be compared with previous revisions and decide a best candidate by a certain similarity measure. Based on the characteristics of Wikipedia editing, we assume that the best candidate is the one that takes least efforts to convert to *r*. 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.

WikiSym '13, Aug 05-07 2013, Hong Kong, China.

To find candidates that meet the above requirements is different from nearest neighbor search (NNS) in text mining. The conventional NNS deals with text corpus that is generally heterogeneous, while in our research the text content is mutually highly similar in the revision collection. Common text clustering

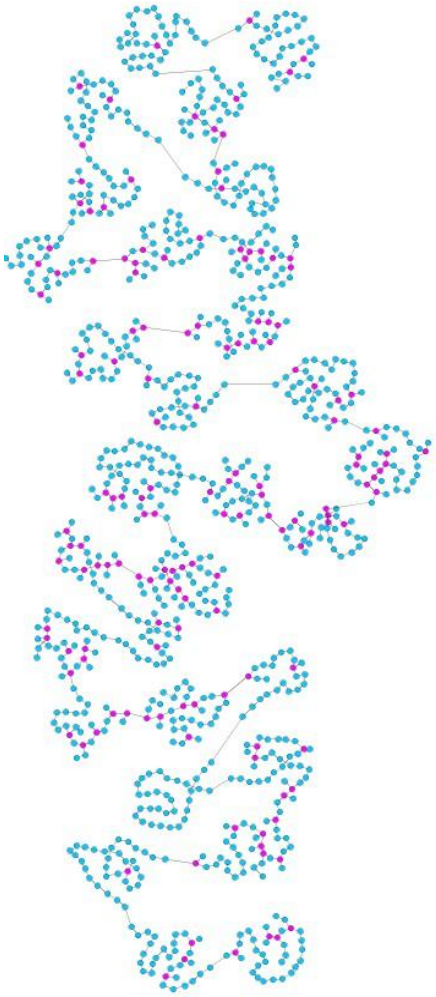

Figure 1.2 Revision graph for first 1000 revisions of Wikipedia article "Edith Wharton", pink nodes indicate where branch happens.

methods like kNN and SimHash [11] fail to distinguish such homogeneous texts. There is another issue we should notice. The overview of Wikipedia mining [4] 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.

In this paper, we first introduce existing work related to our research. In Section 3 we explain our motivation and basic process of supergram decomposition. We extend the model in Section 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. Finally we conclude our paper by summarizing findings and discussing several key issues.

2. RELATED WORK

Basically, a revision history modeling method should include three components: text differencing method, similarity measurement and comparison strategy. Most existing work focused on the first component. Fong et al. [3] 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 string 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 [3], 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 over-simplified and the correctness of the trajectories they build is not assured.

Cao et al. [2] 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 largest common subsequences are used for more precise comparison, which is closer to string matching problem.

Wu et al. [4] 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 history reconstruction problem on $\boldsymbol{R}$ is to find a directed acyclic graph (DAG) where nodes are revisions and edges $\langle r_i, r_j \rangle$ 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 revision history is a tree.

As the further research of [4], 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 nearest neighbor search[9][10]. 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 longest common subsequence-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 [12], 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* $s=t_1t_2..t_n$ in a revision subset $R' \subseteq \mathbf{R}$, where R' is called a *comparison scope*, is an n-gram ($n>=2$) such that s occurs in all the revision in R'.

Example 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.

Basically, for the revision set **R** of an article, we choose a comparison scope *R'* to restrict the candidates of revision parents, and we extract the *supergrams* by *path contraction* on the word transition graph on *R'*. Then we utilize *supergram diff* to compare revisions. More precisely, our method consists of the following steps:

1. **Pre-processing**. After text-cleansing and URL replacement, split all revisions into bigrams and construct a global inverted index *I* of bigrams on revisions.

2. **Word transition graph construction**. By scanning each revision, construct a word transition graph *G* for the revision scope. Compact *G* into a weighted multigraph *G'* by path contraction, extract the edges' weights in *G'* to construct the supergram list *S*.

3. **Supergram decomposition**. Decompose each revision based on S, and then construct an inverted index of *S* on revisions. Construct an inverted index of all the terms appearing in *S*.

3.1 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*, 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.

3.2 Word transition graph construction

Given an article *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, labeled with the collection frequency of bigram t_it_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** $G=(V, E)$ is a directed weighted graph such that each vertex $v_i \in V$ denotes a term $t_i \in D$. For two terms t *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_it_j has a frequency $f(t_it_j) > 0$ in R, and $f(t_it_j)$ is assigned as the edge weight.

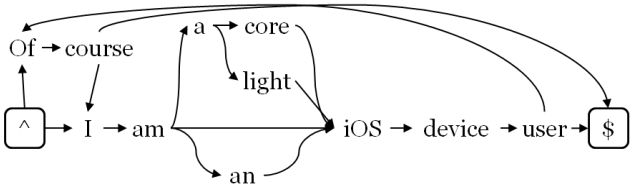

Figure 3.1 Word transition graph for Ex. 3.1

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 Figure 3.2. 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** $H=$ is a sequence of edges $<v_1,v_2>,<v_2,v_3>,..,<v_{n-1},v_n>$ ($n>=3$) 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 starting vertex v_1, called the *source*, satisfies $|out(v_1)| = 1$ and $|in(v_1)| \neq 1$. The *sink* v_n satisfies $|out(v_n)| \neq 1$ and $|in(v_n)|= 1$.

Path contraction

By path contraction, each edge $<v_i, v_j>$ should satisfy both:

a) Correctness.

For any bigram t_it_j in revision set R, its frequency $f(t_it_j)$ is equal to the supergram frequency $f(s_k)$ in R, where s_k is the supergram that contains t_it_j.

b) Compactness.

If the source v_i has no in-degree ($|in(v_i,)| = 0$), the target v_j should have more than 1 out-degree ($|out(v_j)| > 1$). Otherwise the total degree of source and target should be more than 3.

Regarding such requirements, we describe the algorithm as follows:

Algorithm for path contraction:

Input: G = (V, E)
For each vertex $v_i \in V$ such that $|out(v_i)| > 0$,

for each $v_j \in out(v_i)$,

If v_j is a chain vertex with an outgoing edge $<v_j, v_k>$, create a new edge $<v_i, v_k>$ and label it with the concatenation of the label of $<v_i, v_j>$ and v_k's corresponding term t_k, and delete v_j from G.

Notice that each revision can be treated as a token sequence starting from the same source "^" and sink with the same

terminator "$". Thus there is no need to consider the cases of $|out(v_i)| = 0$, or $|in(v_i)| = 0$.

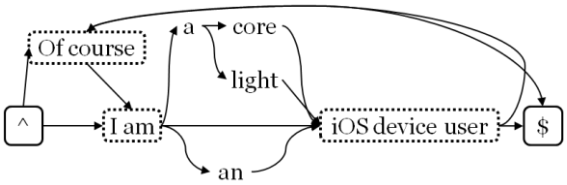

Figure 3.2 Word transition graph for E.g. 3.1, after path contraction

After path contraction, the original word transition graph is contracted to a multigraph such that each vertex v'_i's corresponding to term t_i has a frequency $f(t_i) > 0$ in at least one revision, and each edge $<t_i, t_j>$ representing a supergram s in either way:

a) If the edge label is a concatenation of (freq|terms), s is a new concatenation of t_i + *terms* $+t_j$.

b) If the edge label is an integer, $s = t_i\ t_j$.

4. COMPARISON SCOPE

In this section we extend supergram decomposition by introducing a sliding revision scope and finish the whole comparison on a revision set.

4.1 Comparison scope and supergram size

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. Another strong reason for scoping is scalability. In Wikipedia, articles pose various numbers of revisions from tens to tens of thousands. Without any heuristics, it takes $O(X^2)$ time to perform full pairwise comparisons for X revisions, which is too expensive especially for those popular articles with thousands of revisions. Regarding these issues, we extend the global decomposition by introducing a sliding comparison scope and finish the whole comparison on revision collection. The comparison stage consists of four stages:

1. Comparison scope determination

For each revision r_i, calculate r_i's comparison scope C_i based on r_i's timestamp.

2. Sliding decomposition

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.

3. Supergram diff score computing

Compare r_i with all revisions in C_i by a diff score defined on supergrams.

4. Candidate selection

Pick up the revisions with lowest k supergram diff score as the candidates for parents.

The following figure shows the basic process of decomposition based on sliding word transition graph. We describe major components in detail in the following subsections

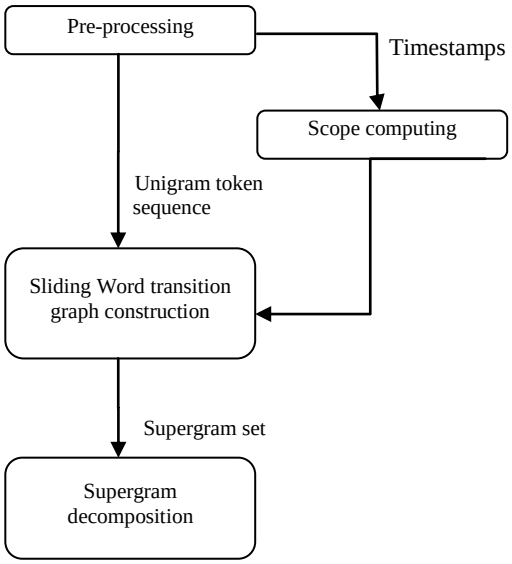

Figure 4.1 Extension of supergram decomposition

4.2 Comparison scope decision

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.

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

Figure 4.2 Edit count of Wikipedia article "Barack Obama" during 2008, the year of the U.S. presidential election1.

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 4.2 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 defined as below.

DEFINITION 4.1 Maximum comparison scope:

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

Given a revision history $H = \{(r_1, t_1), (r_2, t_2), ..., (r_m, t_m)\}$,where (r_i, t_i) denote a revision r_i with its timestamp t_i, the maximum comparison scope C for revision r_k is determined by either:

a) if $t_k - t_{k-T} > T$, $C = S_l$, or

b) if $\exists p > 0$ such that $t_k - t_{k-p} \leq T$ and $t_k - t_{k-p-1} > T$, $C = p$

where S_l denotes the least scope to ensure enough comparison for unpopular documents, T denotes the least time span for intense edits.

Notice that there could be a series of consecutive edits by the same contributor, we take the latest revision only 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 *rj* and *ri* but not in revisions between *rj* and *ri*.

4.3 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:

$$SD(r_a, r_b) = \{ s \in S \mid | f(s, r_a) - f(s, r_b) | > 0 \} \qquad (4.1)$$

DEFINITION 4.3. Supergram diff score

$$diffScore(r_a, r_b) = w_1 \cdot \sum_{s \in SD_{add}} |f(s, r_a) - f(s, r_b)| \cdot |s| + w_2 \cdot \sum_{s' \in SD_{del}} |f(s', r_a) - f(s', r_b)| \cdot log\,|s'| \qquad (4.2)$$

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.

5. EXPERIMENTAL EVALUATION

To evaluate the performance, we conduct two accuracy evaluation on the proposed method with 4 representative methods: sentence-level Jaccard distance [7], keyword clustering[10], n-gram cover[4] and the conventional token-level Edit distance. For each method, we compare its result revision graphs with manually constructed graphs on the existing ground truth [4], a collection of Wikipedia articles. As shown in Table 5.1, the ground truth data set contains 10 Wikipedia articles totaling 2000 revisions. In addition, we expand the revision set of article #4-"Edith Wharton" and #5-"Trailer (promotion)" to their all existing revisions, involving more than 1000 revisions of both articles. Such total revision sets contain the whole process of revision evolution and involve more remote copies, which demand more effectiveness for modeling methods.

Article #	Article Title	# of Branches
1	Racism	23
2	2006 Israel–Gaza conflict	12
3	PhpBB	37
4	Edith Wharton	53
5	Trailer (promotion)	42
6	Sarkar Raj	15
7	Grade inflation	24
8	Natal chart	11
9	Muhammad Naguib	8
10	Clarinet Concerto	12

Table 5.1 Ground Truth Statistics

All the revisions have been pre-processed according to Section 3.1 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 correct parent, which is shown in Figure 5.1.

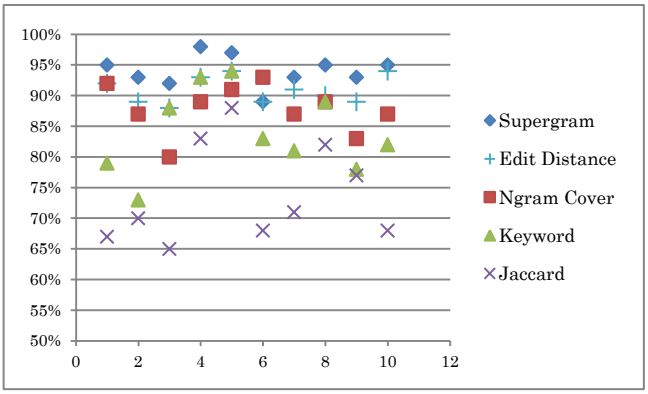

Figure 5.1 Parent accuracy

We evaluate branching errors that happen in different stages by reachability comparison. Given two revision graphs G_1, G_2 on the same revision set *D*, the *reachability accuracy* of G_2 on G_1 is defined as follows:

$$C(G_1, G_2) = \frac{2|G_1^+ \cap G_2^+|}{|D|^2} \qquad (5.1)$$

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 (5.1) 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.

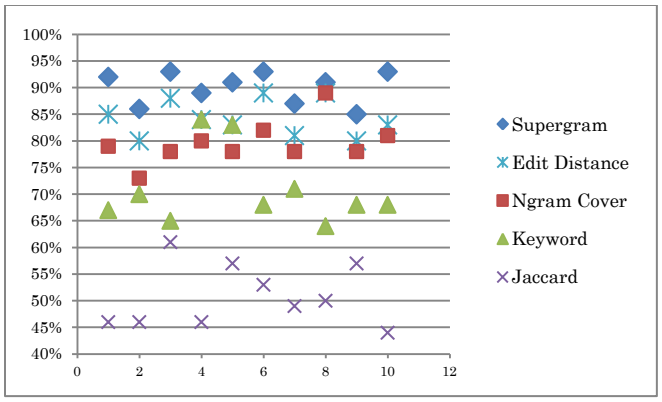

Figure 5.2 Reachability accuracy

In both evaluations, the proposed method prevails on all test articles. 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 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. However, it can achieve the level of Edit distance in #4 and #5 because it fails to cluster by keyword but performs the same procedure of Edit distance. 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.

6. CONCLUSION

In this paper, we proposed supergram technique for accurate reconstruction of Wikipeida revision history. Supergrams are extracted from a word transition graph by path contraction. Our proposed method outperforms existing text comparison/clustering methods. In the future, we will investigate further optimization of comparison scopes, and develop applications utilizing extracted revision graphs, such as visualizations.

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

7. REFERENCES

[1] Dennis M. Wilkinson and Bernardo A. Huberman. 2007. Cooperation and quality in wikipedia. In Proceedings of the 2007 international symposium on Wikis (WikiSym '07). ACM, New York, NY, USA, 157-164.

[2] Ahlqvist, Toni; Bäck, A., Halonen, M., Heinonen, S. "Social media roadmaps exploring the futures triggered by social media". VTT Tiedotteita - Valtion Teknillinen Tutkimuskeskus (2454): 13.,2008

[3] Brian Keegan, Darren Gergle, Noshir Contractor, Staying in the Loop: Structure and Dynamics of Wikipedia's Breaking News Collaborations in Proc. WikiSym'12. ACM.

[4] Jianmin Wu, Mizuho Iwaihara, "Wikipedia revision graph extraction based on n-gram cover", Proc. Int. Workshop on Graph Data Management and Mining, WAIM 2012 , Lecture Note in Computer Science 7419, pp. 29–38, 2012

[5] A. Lih. Wikipedia as participatory journalism: Reliable sources: Metrics for evaluating collaborative media as a news resource. Proc. Int. Symp. Online Journalism 2004

[6] Myers, E, An O(ND) Difference Algorithm and Its Variations. Algorithmica, 1(2): 251–266 , 1986

[7] Mikalai Sabel. Structuring wiki revision history. In Proceedings of the 2007 international symposium on Wikis (WikiSym '07). ACM, New York, NY, USA, 125-130.

[8] U. Manber, "Finding similar files in a large file system," Proc. USENIX Conference, pp. 1-10, 1994.

[9] A.Z. Broder, "On the resemblance and containment of documents," Proc. Compression and Complexity of Sequences, pp. 21-29, Positano Italy, 1997.

[10] Cao, Z., Iwaihara, M., Wikipedia version tree reconstruction by clustering revisions through keywords, IEICE Technical Report DE2011-32, 2011

[11] Gurmeet Singh Manku, Arvind Jain, and Anish Das Sarma. Detecting near-duplicates for web crawling. In Proceedings of the 16th international conference on World Wide Web (WWW '07). ACM, New York, NY, USA, 141-150.

[12] Taha Yasseri, János Kertész, Value Production in a Collaborative Environment, Journal of Statistical Physics, pp. 414-439, Springer US, 2013.