Improving revision graph extraction in Wikipedia based on supergram decomposition

呉　建民[†]　　岩井原瑞穂[‡]

[†] 早稲田大学大学院情報生産システム研究科
〒808-0135 福岡県北九州市若松区ひびきの　２－７

E-mail: [†] 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. With this paper, we propose a revision graph extraction method based on supergram decomposition in the document collection of near-duplicates. We show that this method can effectively perform the task than existing methods.

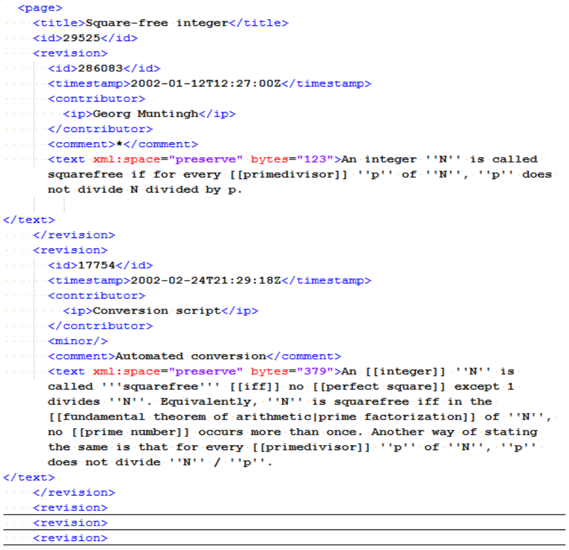
Keyword Social Media, Web Mining, Wikipedia

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, 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 research, 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. For a given revision *r*, it needs to be compared with some previous revisions and decide a best candidate by some 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 single long short edits.

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 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. P. Fong et al [3] proposed a detail 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. This research uses 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

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 searching [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 k-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 too much time to achieve integrity by longest common subsequence based diff algorithm.

We find that there are some token sequences that keep as a unit throughout the whole revision collection. For a small revision collection of several revisions, such token sequences is little but with long length. As the size of the revision collection 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 collection **R** is an *n*-gram (*n*>=2) such that *s* occurs in at least one revision in R, and if a token t_i (1<*i*<*n*) occurs in a revision *R*', then t_{i+1} always occurs just after t_i in *R*'.

Basically, for the revision collection **R** of an article, we extract the *supergrams* by *path contraction* on word transition graph, and utilize *supergram diff* to compare revisions. More concretely, 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 collection. 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, then construct an inverted index of *S* on revisions. Regarding all terms appearing in *S*, construct an inverted index of terms on *S*.

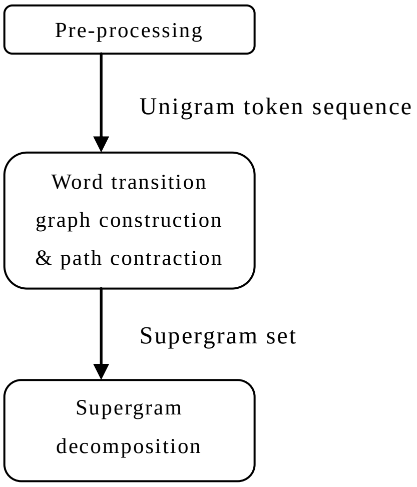


Figure 3.1 Process of supergram decomposition.

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 semantic 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 versions $r_1, r_2, \ldots, r_n$, each of them contains a sequence of tokens $D_i = [t_1, t_2, \ldots, t_l]$. In the following paragraphs, we denote

- v_i : vertex i labeled with t_i;
- $e_x(v_i, v_j)$: 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;
- $src(e)$: source vertex of edge e;
- $tar(e)$: target vertex of edge e

DEFINITION 3.2 Word transition graph

Given a document *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.

The word transition graph is allowed to contain cycles since the multiple appearance of frequent terms causes path that starts and ends at the same vertex but otherwise has no repeated vertices or edges. 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** $C=(V', E')$ is a subgraph of G containing only one connected component, with the property that every vertex $v'_i \in V'$, except the two ends, has only one incoming edge and one outgoing edge, i.e. $|out(v'_i)| = |in(v'_i)| = 1$, and v'_i is called a *chain vertex*. The starting vertex of C, namely the *source*, is defined as the vertex v_s such that $|out(v_s)| = 1$ and $|in(v_s)| \neq 1$. The *sink* is defined similarly.

Path contraction

By path contraction, each edge $e' (v_i, v_j)$ should satisfy both:

a) Correctness.

For any bigram titj in revision collection R, its frequency f(titj) is equal to the supergram frequency f(sk) in R, where sk is the supergram that contains titj.

b) Compactness.

If the source of e' has no in-degree (|in(src(e'))| = 0), the target of e' should have more than 1 out-degree (|out(tar(e'))| > 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 $v_i \in V$, If $|out(v_i)| > 0$,
    for each $v_j \in out(v_i)$,
        If $v_j$ is a chain vertex with an outgoing edge $e(v_j, v_k)$, create a new edge $e'(v_i, v_k)$ and label it with the concatenation of the label of $e(v_i, v_j)$ 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 sinking with the same terminator "^", there is no need to consider the cases of $|out(v_i)| = 0$, or $|in(v_i)| = 0$.

After path contraction, the original word transition graph is contracted to a multigraph such that each vertex v'_i's corresponding term t_i has a frequency $f(t) > 0$ in at least one revision, and each edge $e(t_i, t_j)$ representing a supergram s in either ways:

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, $s = t_i\ t_j$

3.3 Supergram decomposition

Decompose each revision based on supergram set S, construct an inverted index of *S* on revisions. Regarding all terms appearing in *S*, construct an inverted index of terms on *S*. Then all revisions' supergram frequency distribution can be compared based on *S*.

4. Comparison

Recall the observation of supergram we mentioned before: a narrower scope will produce longer supergrams. This is because the number of edits is proportional to the scope length and fewer edit mean smaller chances, and supergrams tend to be undivided. Longer supergram is preferable in supergram decomposition because it reserve more integrity and reduce 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 $O(N^2)$ time to perform full pairwise comparison for an article posing *N* revisions, which is a unbearable expense of computing resource 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 4 parts:

1. Comparison scope determination

For each revision r_i, calculate r_i's comparison scope C_i based on r_i's timestamps and global bigram inverted index *I*.

2. Sliding decomposition

Construct a word transition graph G_i of all the revisions within C_i, and perform path contraction for G_i to get G_i'. Decomposed r_i' and all revisions in C_i into supergram frequency distribution based on the supergram set *S* extracted from G_i'.

3. Supergram diff score computing.

Compare r_i's supergram frequency distribution with all revisions' in C_i and compute their supergram diff scores.

4. Candidate selection.

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

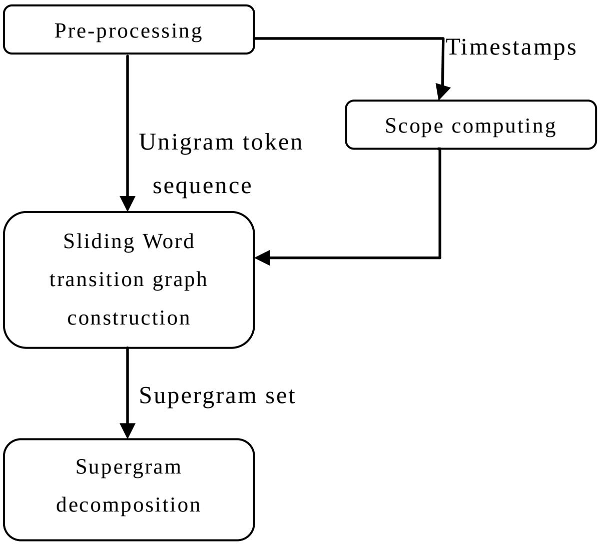
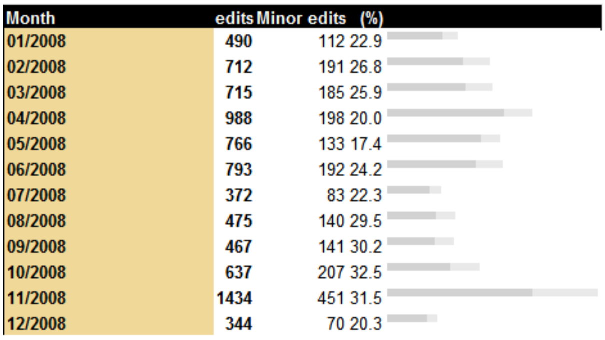


Figure 3.2 Sliding decomposition.

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

4.1 Comparison scope decision

In a near-duplicate collection without any auxiliary information, it is hard to apply aggregation technique to divide the collection into several non-overlapping subsets. Fortunately, Wikipedia edit history is naturally sequential. 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 4.1 Edit count of Wikipedia article "Barack Obama" during 2008, the year of the U.S. presidential election[1].

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 4.1 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.

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 biggest number of previous revisions to be compared, which is defined as below.

DEFINITION 4.1 Maximum comparison scope:

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 ∃p > 0 such that t_k- t_{k-p} ≤ 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.

4.2 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

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

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. As heuristics, the logarithms are to the base of 10, since the deleting operations is less effort-taking job.

4.3 Candidate selection

It is possible that with the lowest supergram diff score there come multiple revisions with the same score. For example, if a revision *r* just revert to one previous revision *r'*, the child of *r* has at least 2 revisions that have the same supergram diff score. To avoid ambiguity, we define that the parent revision should be the one with the latest timestamp in such cases.

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

5. Experimental evaluation

We conduct experimental evaluation on our method against several existing revision relationship modeling methods on a collection of Wikipedia articles.

Article #	Article Title	# of Branches
1	Racism	23
2	2006 Israel–Gaza conflict	12
3	PhpBB	37
4	Edith Wharton	16
5	Federal republic	33
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

To evaluate the performance of the proposed method, we conduct a series of accuracy evaluation with 3 representative methods: sentence-level Jaccard distance, keyword clustering, n-gram cover. For each method, we compare its result revision graph with manually constructed graphs on the existing ground truth [4]. As shown in Table 5.1, the data set contains 10 Wikipedia articles totaling 2000 revisions. The branch counts indicate the total revert events and diversity, potentially suggesting the degree of popularity and controversy. All the revisions have been pre-processed according to Section 3.1 so that all methods start with the same token sequence.

The parent accuracy is evaluated by the percentage of the revisions that has the correct parent.

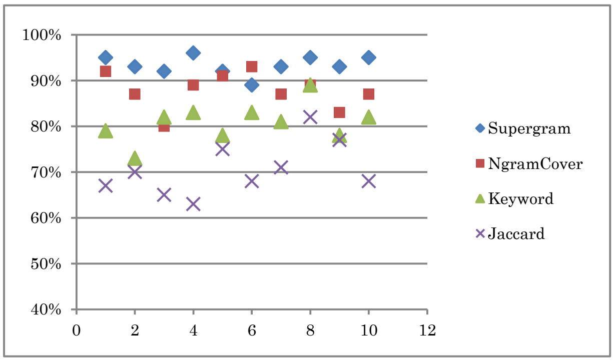


Figure 5.1 Parent accuracy

We evaluate branching errors that happen in different stages by reachability comparison. Given two revision graphs $G_{1,}$ G_2 on the same revision collection 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.

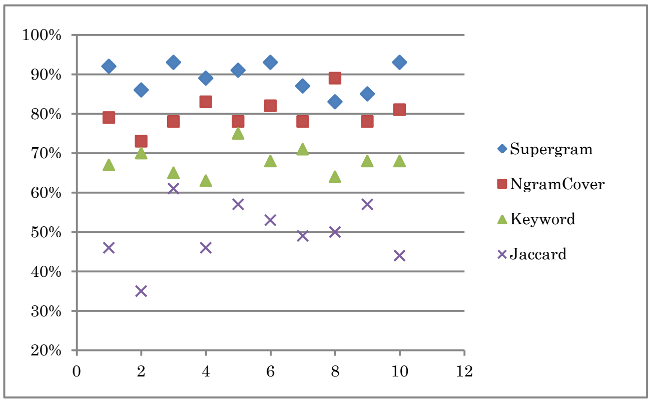


Figure 5.1 Reachability accuracy

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 previous approach by the n-gram cover.

References

[1] Wilkinson, D. M. and Huberman, B. A. Cooperation and quality in wikipedia. In Proc. WikiSym'07 (Montreal, Quebec, Canada 2007). ACM.

[2] Ahlqvist, Toni; B äck, 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] Sabel, M.. Structuring wiki revision history. WikiSym pp. 125-130, 2007

[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 WWW '07. ACM, New York, NY, USA,2007