Graph Clustering based on Structural Similarity of Fragments

Tetsuya Yoshida[1], Ryosuke Shoda[2], and Hiroshi Motoda[2]

[1] Graduate School of Information Science and Technology,
Hokkaido University
N-14 W-9, Sapporo 060-0814, Japan
yoshida@meme.hokudai.ac.jp
[2] Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka 567-0047, JAPAN
{shoda,motoda}@ar.sanken.osaka-u.ac.jp

Abstract. Resources available over the Web are often used in combination to meet a specific need of a user. Since resource combinations can be represented as graphs in terms of the relations among the resources, locating desirable resource combinations can be formulated as locating the corresponding graph. This paper describes a graph clustering method based on structural similarity of fragments (currently connected subgraphs are considered) in graph structured data. A fragment is characterized based on the connectivity (degree) of a node in the fragment. A fragment spectrum of a graph is created based on the frequency distribution of fragments. Thus, the representation of a graph is transformed into a fragment spectrum in terms of the properties of fragments in the graph. Graphs are then clustered with respect to the transformed spectra by applying a standard clustering method. We also devise a criterion to determine the number of clusters by defining a pseudo-entropy of cluster. Preliminary experiments with synthesized data were conducted and the results are reported.

1 Introduction

1.1 Motivation

Huge number of (computing) resources are available over the Web these days. Users may select some of the resources to perform their jobs by exploiting relations among the resources. For example, URLs are resources available over the Web, and they are connected to each other by hyperlinks, as shown in the left hand side of Fig. 1 (hyperlinks are depicted as dotted lines as directed edges). Suppose that the pattern of with URL {K, M, A, B} are frequently observed in a log file of web browsing. When a user follows or selects URL {K, M, A}, it is likely that he/she may select URL B. Thus, by discovering the pattern shown in the right hand side of Fig. 1, it will be possible to help users to select or locate the resource by recommending the resource in the pattern. As another example, web citation analysis is reported and compared with bibliographical citation

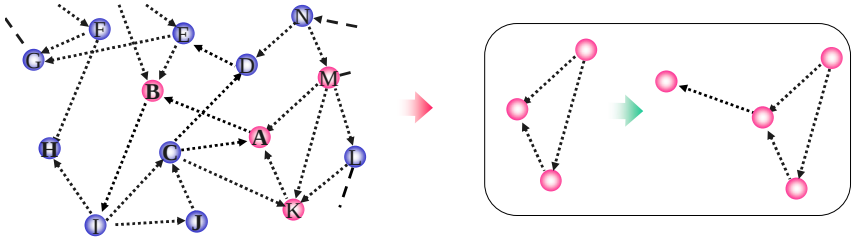

Fig. 1. Example of resource selection (web browsing pattern)

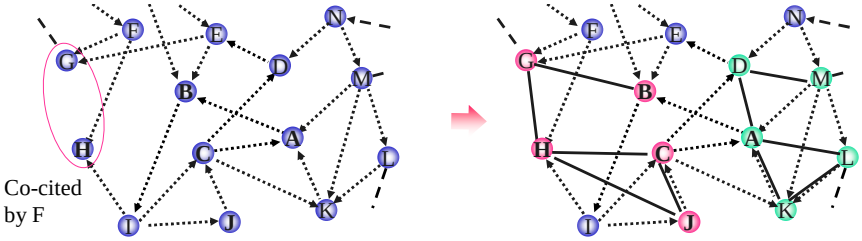

Fig. 2. Example of co-citation graph

analysis in [12]. Although URL G and H are not connected to each other directly, they are both pointed to (or, cited) by F. Thus, G and H are co-cited by F, as shown in the left hand side of Fig. 2. In terms of this kind of co-citation relation, URLs are (implicitly) connected to each other as shown in the right hand side of Fig. 2 (co-citation relations are depicted as thick lines as undirected edges). Once co-citation graphs are constructed, various analysis can be conducted on the graphs. For instance, URL H and A might be interesting because they have large number of co-citation relation.

As illustrated in the above examples, since resource combinations can be represented as graphs in terms of the relations among the resources, locating desirable resource combinations can be formulated as locating the corresponding graph. In our approach it is assumed that relations among resources are specified externally such as hyperlinks or co-citation relations, and how to define relations among resources to construct appropriate graph structured data is beyond the scope of this research. When a user tries to locate desirable resource combinations, ultimate goal of this research is to support the selection of resource combinations in terms of graph structures over the resources.

1.2 Mining Graph-Structured Data

Various researches have been conducted to extract knowledge from the vast body of unstructured Web data [2]. The majority of data mining methods widely used are for data that does not have structure and is represented by attribute-value

pairs. Decision tree [14, 15], and induction rules [10, 3] relate attribute values to target classes. Association rules often used in data mining also utilize this attribute-value pair representation. However, the attribute-value pair representation is not suitable for representing a more general data structure, and there are problems that need a more powerful representation. Most powerful representations that can handle relation and thus, structure, would be inductive logic programming (ILP) [11] which uses the first-order predicate logic. It can represent general relationships embedded in data, and has a merit that domain knowledge and acquired knowledge can be utilized as background knowledge. However, in exchange for its rich expressibility, the time complexity causes problem [5].

Since structure is represented by proper relations and a graph can easily represent relations, knowledge discovery from graph structured data poses a general problem for mining from structured data. Various researches such as AGM [6], FSG [7], Subdue [4], have been conducted on mining from graph structured data. Some examples amenable to a graph mining are finding typical web browsing pattern, identifying typical substructure of chemical compounds, finding typical subsequences of DNA and discovering diagnostic rules from patient history records.

We have applied our graph mining method called GBI [8] to extract typical patterns from the hepatitis dataset provided by Chiba University Hospital in the Active Mining project [9, 20]. GBI extracts connected subgraphs from graph structured data by conducting greedy search without backtrack. Because of its greedy search, it can handle large scale graph structured data. One drawback of its search strategy is that search is incomplete in the sense that not all the subgraphs are enumerated. One of the problems we encountered in the project is that huge number of patterns (connected subgraphs in our approach) can be extracted from large scale graph structured data by applying our graph mining method. The number of extracted graphs will increase by applying other complete graph mining methods and the problem will get more severe. Thus, although patterns can be extracted from graph structured data by applying graph mining methods, it gets very difficult to evaluate all the extracted patterns.

1.3 Clustering Graph-Structured Data

Although the number of graphs to be considered can be huge, some components of graphs share structural properties. For example, chemical compound derived from benzene ring (aromatic compounds) have similar chemical properties because they share the benzene ring. This motivates a research branch called (quantitative) structure activity relationship in computational chemistry. In QSAR, the relationship between the property or activity of chemical compounds and their structure has been studied. As another example, starting from the pioneering work in [19], many researches have been conducted to reveal that a lot of many graphs or networks are categorized into a so called "small world" network in terms of their structure [1]. Small world networks share the properties that, although nodes are densely connected locally with their neighbors, the

average path length between two nodes in the network is relatively short as a whole due to some edges which connect distant nodes.

By assuming that graphs with similar structure share some properties, we aim at clustering graph-structured data based on their structural similarity in this research. When graphs are categorized into clusters, a small number of graphs can be selected from each cluster as representatives. As in the research on small world networks, we consider structural properties of graph in terms of connectivity (degree) of nodes in the graph, and transform the representation of graph into the corresponding spectrum. Transformation into the corresponding spectrum acts as a kind of hash function. Thus, if a user can specify the desirable resource combinations as a graph, our method can be utilized to discriminate the graphs of resource combinations into the similar and dissimilar ones in terms of the corresponding spectra. Graphs are then clustered with respect to the transformed spectra by applying a standard clustering method. We also propose a method based on our notion of pseudo-entropy of cluster to determine the number of clusters in clustering. Preliminary experiments on synthetic data were conducted and the results are reported in this paper.

As described in Section 1.2, this research is motivated by the application of GBI, which can handle general graph data with both directed and undirected edges. However, as a first step, only undirected graphs are considered in this paper, and labels of nodes and edges are not dealt with yet. The graph structure currently handled with corresponds to the example in Fig. 2. Since the labels are not yet dealt with, two graphs in the right hand side of Fig. 2 are considered as isomorphic, and are categorized into the same cluster. Thus, our current status is very preliminary since the content of resource (e.g., textual information in a URL) is not dealt with. However, filtering out graphs with respect to structure can be utilized as preprocessing for more fine-grained and detailed analysis of contents.

Various researches have been conducted on the similarity measure of graph structured data and clustering of graph structured data. Among related works, Topological Fragment Spectra (TFS) [17] has been proposed, which characterizes the properties of chemical compounds in terms of fragments (subgraphs) within the compounds. ANF [13] is proposed for the fast calculation of similarity for large scale graph structured data. Our method is motivated by TFS, however, it differs with respect to two aspects: 1) calculation of fragment spectra, and 2) extension to clustering of spectra.

Organization This paper is organized as follows. Section 2 describes a method for representing the properties of a graph as a spectrum of fragments (subgraphs). Experiments on the calibration of the proposed fragment spectrum are also reported. Section 3 describes the clustering of fragment spectra and a method for determining the number of clusters. Preliminary experiments with synthetic data are reported in Section 4. A short concluding remarks and future directions are described in Section 5.

2 Fragment Spectrum of Graph

2.1 Preliminaries

A simple graph is denoted by $G = (V, E)$ where V is a set of vertices and $E \subseteq V \times V$ a set of (undirected) edges in G. For any vertices v, $v' \in V$, if $(v, v') \in E$, v is said to be adjacent to v'. Let $G = (V, E)$ be a graph. For a vertex $v \in V$, the set of vertices adjacent to v is denoted by $N_G(v)$. $|N_G(v)|$, the size of $N_G(v)$, is called the degree of v with respect to G. $|N_G(v)|$ is also referred to as $degree_G(v)$. Let $G = (V, E)$ and $G' = (V', E')$ be two graphs. G is called a subgraph of G' when $V \subseteq V'$ and $E \subseteq E'$, and is denoted as $G \subseteq G'$.

2.2 Fragment Spectrum of Graph

Similarity measures of graph can be categorized into two approaches: direct comparison based approach and fingerprint based approach [16]. In the former approach, the similarity between two graphs is measured either by the maximum common subgraphs or the maximum common edge subgraphs. The maximum common subgraphs are identified in two graphs and the size (either number of nodes or edges) of the subgraphs are used to measure the similarity. Although the similarity can be measured directly on the graphs, exact identification of the maximum common subgraphs can be very expensive in practice. On the other hand, in the latter approach, a graph is represented as a bit string, each bit indicates the presence or absence of a predefined substructure (which acts as a key descriptor). The similarity between two graphs is measured by comparing their corresponding bit strings. Although key descriptor selection needs to be addressed in fingerprint based approach, we take this approach since it is rather simple and easy in practical use.

We aims at capturing structural properties of a graph based on its fragments in the graph and represents the graph as a fragment spectrum[1]. Currently a fragment of a graph is defined as a connected subgraph in the graph. Hereafter, we consider only connected subgraphs. Fig. 3 shows an example of the construction of fragment spectrum in our approach. The score of a fragment in a graph G is calculated by a function FScore, which is explained in Section 2.3. The graph in Fig. 3 has 1 fragment with score 1, 3 fragments with score 5, 3 fragments with score 6, etc. Based on the frequency of fragments with the same score, the fragment spectrum for the graph is represented as a vector $\boldsymbol{fs} = (0, 0, 1, 4, 3, \ldots)$, where $\boldsymbol{fs}[i]$ represents the frequency (count) of fragments with score i. Construction of fragment spectrum for a graph is shown in Fig. 4.

2.3 FScore function

A fragment F in a graph G is characterized as a score by a function called FScore based on its connectivity. Each node v in a fragment F is scored based on its degree and the score of F is calculated based on the scores of nodes in F. We consider the following two "degree" for a node $v \in F$:

[1] Following Topological Fragment Spectra (TFS) [17], we call a fragment spectrum.

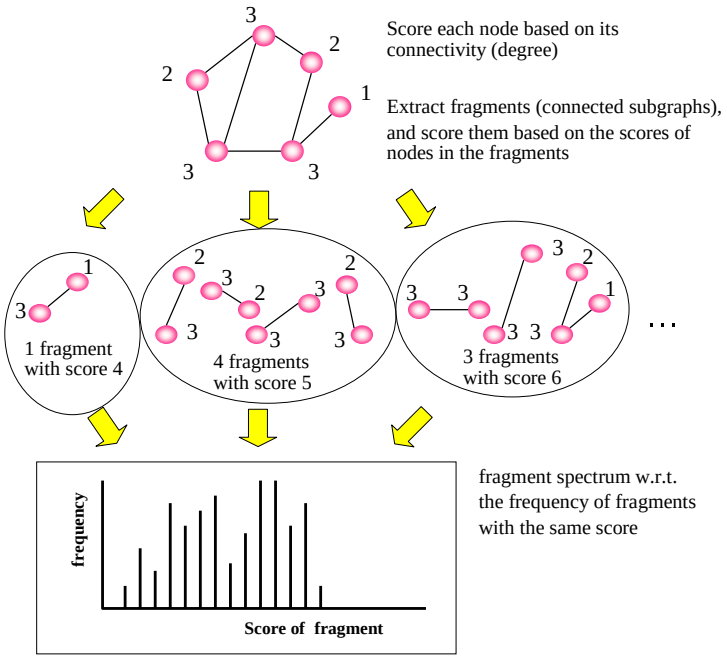

Fig. 3. Fragment spectrum of a graph

fragment_spectrum(graph G)
  $\boldsymbol{fs}$: fragment spectrum of G
  initialize $\boldsymbol{fs}$ to $\mathbf{0}$
  forall fragment (connected subgraph) $F \in G$
    $\boldsymbol{fs}[\mathbf{FScore}(F, G)] := \boldsymbol{fs}[\mathbf{FScore}(F, G)] + 1$
  return $\boldsymbol{fs}$

Fig. 4. Fragment spectrum construction

- degree sorely in the fragment
- degree within the original graph

The former follows the standard definition of degree in a graph, and focuses on the connectivity sorely in the fragment. However, the degree in a fragment is invariable to the graphs which contain the fragment. To reflect the difference of the original graph, the latter considers the degree within the original graph, hoping that the relationship of F to the remaining part of G works as a sort of context of F within G.

Table 1. FScore function

	degree of node	
	original graph	fragment
sum	FScore1	–
square sum	FScore3	FScore2

In addition, the difference between the scores of nodes can be magnified using polynomials of score. The score of a fragment is calculated based on the scores of nodes in the fragment:

- sum of the scores of nodes
- square sum of the scores of nodes

FScore functions to calculate the score of a fragment is summarized in Table 1. With the combination of "sum" and "fragment", the fragments with the same number of nodes and edges have the same score regardless their structure or connectivity. Thus, since it does not reflect structural properties of graph, it is not considered and thus is left blank (–) in Table 1.

The number of fragments in a graph G increases exponentially with respect to its size (the number of nodes and edges). Thus, it becomes difficult to compare spectra of graphs with different size since the overall shape of spectrum can become very different. To make it easy to compare fragment spectra of graphs with different size, although it is trivial, a normalized fragment spectrum is defined by dividing each value (frequency of fragments) in the fragment spectrum by the total number of fragments in the graph. Note that currently normalization is considered only for the frequency of fragments. Normalization of scores is not considered yet.

In summary, 6 FScore functions (3 variations in Table 1 and the corresponding normalized ones (NFScore1, NFScore2, NFScore3)) are currently used.

2.4 Calibration of Fragment Spectrum

This section reports the experiments on the calibration of the proposed fragment spectrum to verify that graphs with different structure can be differentiated with respect to the corresponding fragment spectra. 4 types of graph structured data are considered: line, ring, ring lattice and star. These types of graph structure are prepared such that the average path length between two nodes, which corresponds to the characteristic path length in small world networks [19], decreases in the order of line, ring, ring lattice and star structure. To simplify the calibration, relatively small scale graphs with these types are prepared, as shown in Figs. 5, 6, 7 and 8.

For the graphs with the same number of nodes, a graph with line structure and the one with ring structure differs only with one edge. Thus, these two structures can be considered as very similar. Ring lattice structure in Fig.7

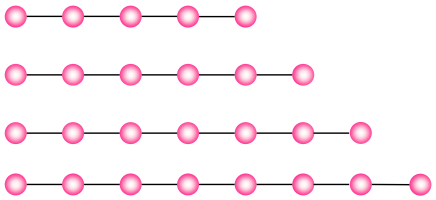

Fig. 5. Line

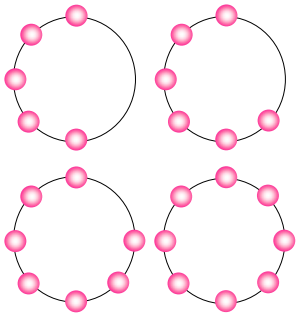

Fig. 6. Ring

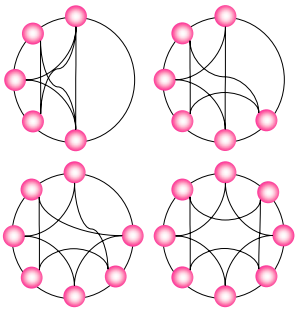

Fig. 7. Ring lattice

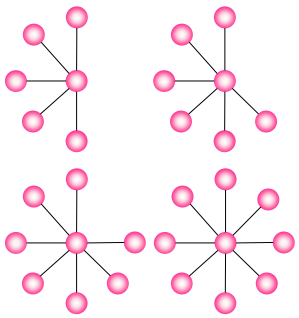

Fig. 8. Star

corresponds to the β model in [18]. It is similar to ring structure but each node is connected to k neighbors (here, k was set to 4). On the other hand, star structure is different from these in terms of the connectivity, since a graph with star structure has one "central" node which is connected to all the other nodes. When each node corresponds to a scientific paper and the relations among nodes are defined as co-citation relations (similar to the example in as in Fig. 2), the central node corresponds to a sort of seminal paper.

Examples of fragment spectra for the graphs are shown in the appendix. For instance, using NFScore 3 function in Section 2.3, spectra of graphs with star structure in Fig. 8 have peak band elements (normalized frequencies) at the tail of the spectra[2], due to its structure and scoring method, as shown in Fig. 21. Thus, if the maximal value of score is known for each graph or the spectrum is normalized with respect to the score, a high-pass filter for fragment spectrum might be useful to extract graphs with star structure among many graphs.

The graphs in Figs. 18, 19, 20 and 21 with 6 nodes are compared in Fig. 9. Fragment spectra were created using NFScore 3 (normalized score function was used to see the overall shape of spectra). The graph with ring structure has

[2] As described in the appendix, horizontal axes (score of fragment) are aligned to the maximal score in the rightmost graph in each figure.

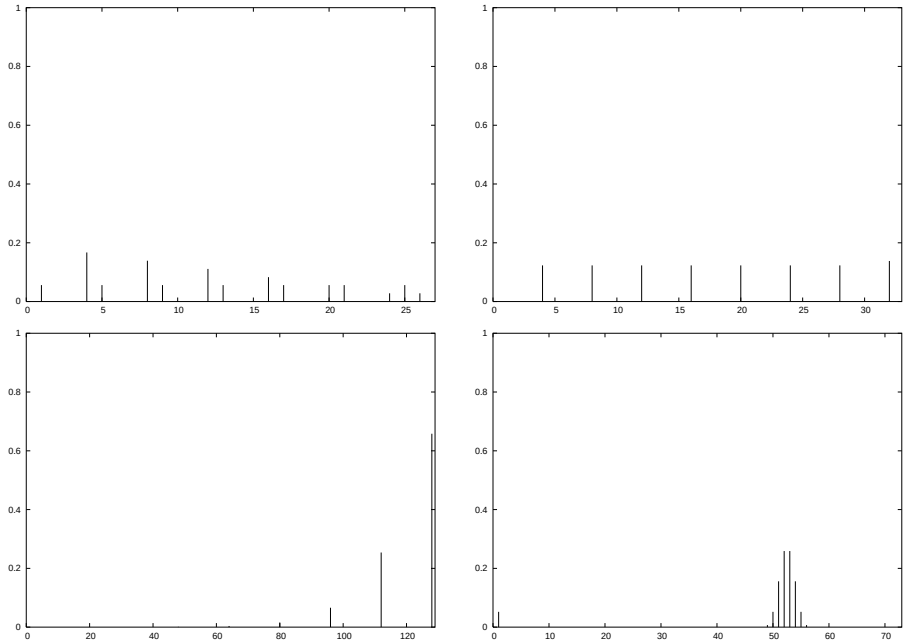

Fig. 9. Fragment spectra of graphs with 6 nodes using NFScore3 (upper left: line (Fig. 5), upper right: ring (Fig. 6), lower left: ring lattice (Fig. 7)) lower right: star (Fig. 8),

almost flat spectra (upper right in Fig. 9), because the frequency of fragments for each score is almost the same. The graph with line structure and the one with ring structure are quite similar in terms of the number of nodes or common subgraphs (upper left). The spectrum of the former also includes flat components, but it also includes another components with decreasing number of relative frequency. The graph with ring lattice has only several elements with growing frequencies (lower right) using NFScore3. The graph with star structure has a peak band (lower left), since the central node has the large score (degree) and the other nodes has a small score. Thus, since the scores of fragments differ only with respect to the number of non-central nodes, fragments in a graph with star structure tend to have almost similar scores.

From this result, it can be said that fragment spectra of graphs with different structure also tend to have rather different shape or pattern. Thus, the score functions can be used as a kind of hash functions to discriminate graphs with different structure. However, graphs with different spectra can be considered as different, but different graphs may come to the same spectrum when a hash collision occurs, as in many hash functions. Thus, we do not claim that fragment spectra of graphs are enough to differentiate graphs with different structures.

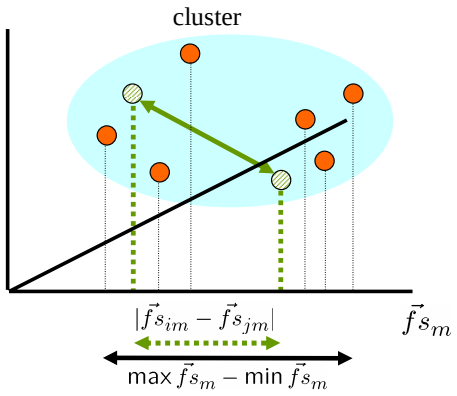

Fig. 10. Similarity of spectra

3 Clustering Fragment Spectrum

After transforming the representation of graphs into the corresponding fragment spectra by the method in Section 2, graphs are clustered with respect to the transformed spectra by applying a standard clustering method. Among various clustering methods, we simply utilize K-means method with respect to the similarity of fragment spectra. Our main contribution for clustering is to devise a criterion to determine the number of clusters, since the number of clusters need to be specified beforehand in many clustering methods. Inspired by the divide and conquer strategy in decision tree construction algorithms [14, 15], we view clustering of data as the division of the whole data into the specified number of clusters. By defining a pseudo-entropy of cluster, the quality of cluster is measured as in the information gain ration in [15] before and after the division of data into clusters.

3.1 Similarity of Fragment Spectra

A spectrum is conceived as a vector in a multi-dimensional space where each dimension corresponds to a score of fragment and the value represents the frequency or normalized frequency of fragments of the score. When comparing two fragment spectra of different dimensions (scores), the one with smaller dimension is padded with 0 for the larger dimensions. By regarding the spectra as vectors, it might be possible to utilize similarity measures for vectors such as cosine similarity. However, toward incorporating nominal attributes such as labels of nodes and edges in future work, currently similarity is measured for each dimension separately and averaged. Similarity in one dimension is calculated by projecting the vector onto the corresponding dimension, and measured as the relative range of the projection, as shown in Fig. 10. Our similarity measure of graphs G_i and G_j in terms of their corresponding spectra is defined as:

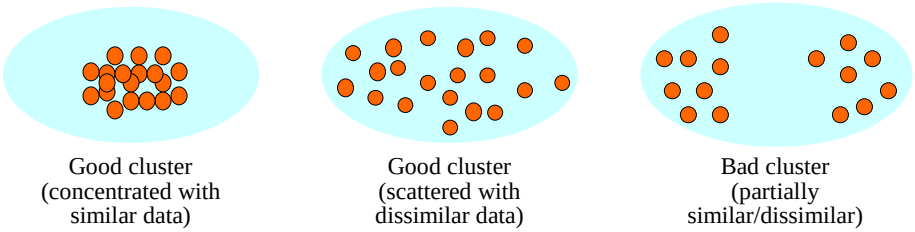

Fig. 11. Pseudo-entropy of cluster

$$S_{ij} = \frac{1}{M}\sum_{m=1}^{M}(1 - \frac{|\boldsymbol{fs}_{im} - \boldsymbol{fs}_{jm}|}{\max \boldsymbol{fs}_m - \min \boldsymbol{fs}_m}) \tag{1}$$

$\boldsymbol{fs}_{im}$: mth coordinate value of $\boldsymbol{fs}_i$ for graph G_i
$\max \boldsymbol{fs}_m$: maximal value for the mth coordinate for all spectra
$\min \boldsymbol{fs}_m$: minimal value for the mth coordinate for all spectra
M: maximal dimension for all the spectra

3.2 Pseudo-Entropy of Cluster

A machine learning method C4.5 [15], which constructs a decision tree, selects an attribute to divide the data based on the entropy of data before and after the division. We apply this approach to determine the number of clusters by defining a pseudo-entropy of data in clustering. Fig. 11 illustrates our notion of pseudo-entropy of cluster. In the standard concept of entropy, concentrated data has low entropy, and the scattered data has high entropy. However, we regard a cluster with concentrated data as good, since it certainly forms a cluster of data, but we also regard a cluster with scattered data as good, since it is difficult to further divide them into (meaningful) sub-clusters. Thus, we would like to give low pseudo-entropy to these clusters. On the other hand, if the data inside a cluster is rather separated as shown, we regard it as bad, since it is possible to further divide the cluster. Thus, we would like to give high pseudo-entropy to this kind of cluster.

We define a pseudo-entropy of cluster C_k as $PEnt(C_k)$ such that it has low value either when data is evenly distributed in a cluster or when all the data concentrate on a small portion in a cluster, as follows:

$$PEnt(C_k) = -\frac{1}{|C_k|^2}\sum_{i=1}^{|C_k|}\sum_{j=1}^{|C_k|}(S_{ij}\log_2 S_{ij} + (1 - S_{ij})\log_2(1 - S_{ij})) \tag{2}$$

$PEnt(C_k)$: pseudo-entropy of cluster C_k
$|C_k|$: size (number) of data in cluster C_k
$S_{ij} \in [0,1]$: similarity of fragment spectrum $\boldsymbol{fs_i}$ and $\boldsymbol{fs_j}$

$PEnt(C_k)$ is calculated based on the pair-wise comparison of data within the cluster. $S_{ij} \in [0,1]$ corresponds to the similarity of two data (spectra of graphs G_i and G_j) within the cluster, and $D_{ij} = 1 - S_{ij}$ corresponds to the dissimilarity.

Intuitively, when comparing two graphs G_i and G_j, let's consider an event I where G_i and G_j are isomorphic. Also, let's consider an event N where G_i and G_j are not isomorphic. These two events are mutually exclusive. Since $S_{ij} \in [0,1]$, $D_{ij} \in [0,1]$, and $S_{ij} + D_{ij} = 1$, we treat S_{ij} as the probability of the event I and $D_{ij} = 1 - S_{ij}$ as the probability of the event N. The value $-(S_{ij} \log_2 S_{ij} + (1-S_{ij}) \log_2 (1-S_{ij})) = -(S_{ij} \log_2 S_{ij} + D_{ij} \log_2 D_{ij})$ corresponds to the binary entropy function for a binary random variable, whose value is I with probability S_{ij} and N with probability D_{ij}. Average of this value for all the pair-wise comparison of data in a cluster is calculated in (2).

When all the data in a cluster are the same and concentrate on a single point in multi-dimensional space, since $S_{ij} = 1$ for all the pairs of data in the cluster, the numerator of (2) becomes 0 and thus is minimized. Likewise, when all the data in a cluster are completely dissimilar ($S_{ij} = 0$) and scattered within the cluster, the numerator also becomes 0. On the other hand, when the data are partially similar and dissimilar each other ($S_{ij} = 0.5$), the numerator is maximized.

3.3 Information Gain Ratio for Cluster

Based on the difference of pseudo-entropy of cluster in (2) before (i.e., the whole unclustered data) and after clustering, we define an information gain ratio of cluster (IGRC) for the situation where the data are assigned to K clusters as:

$$IGRC = \frac{PEnt(C) - \sum_{k=1}^{K} \frac{|C_k|}{|C|} PEnt(C_k)}{-\sum_{k=1}^{K} \frac{|C_k|}{|C|} \log_2 \frac{|C_k|}{|C|}} \tag{3}$$

$PEnt(C_k)$: pseudo-entropy of cluster C_k
$PEnt(C)$: pseudo-entropy of the whole unclustered data

Note that when each data is assigned to a cluster with only the data, the numerator of (3) is maximized since the entropy of each cluster is 0. Thus, to penalize such an over-clustered situation, the difference is divided by the split gain of clustering as in C4.5 [15].

Our criterion for the number of clusters is to select the number of clusters which maximizes the value of IGRC. This criterion is used in the following experiments in Section 4.

4 Preliminary Experiment

Preliminary experiments were conducted to evaluate the proposed method over synthetic data. This section explains experimental settings and reports the results.

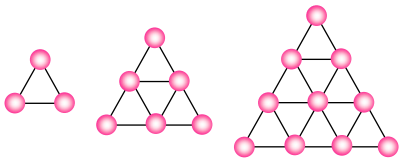

Fig. 13. Example of graphs in Exp.1

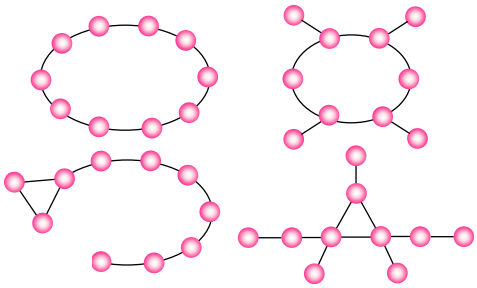

Fig. 14. Examples of graphs in Exp.2

4.1 Synthetic Data

In experiments, synthetic data (graphs) were created by preparing a predefined set of graphs (which are called base graphs in this paper)and appending the subgraphs in Fig. 12. The synthesized graphs are called derived graphs. The base graphs are prepared with respect to 1) the number of nodes and edges, and 2) configuration. Following two experiments were conducted:

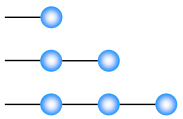

Fig. 12. appended subgraphs

Exp.1 number of nodes and edges: graphs with different number of nodes and edges but with similar configuration

Exp.2 configuration: graphs with dissimilar configuration but with the same number of nodes and edges

Base graphs used in Exp.1 and Exp.2 are shown in Figs. 13 and 14, respectively. The number of graphs (with the the base graphs and derived graphs) in Exp.1 was 18 and 61 in Exp.2. One drawback in the above setting is that, despite many researches have been conducted, "the correct" measure of similarity/dissimilarity of configuration of graphs is not known yet. Thus, when preparing graphs in Exp.2, it is based on our subjective assessment.

Table 2. k with maximal IGRC

FScore function	k with maximal IGRC	
	Exp.1	Exp.2
FScore1	9	2
FScore2	2,3	2
FScore3	8	17
NFScore1	3	2
NFScore2	2	2
NFScore3	3	2

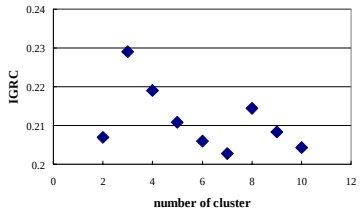

Fig. 15. IGRC in Exp.1 (with NFScore1)

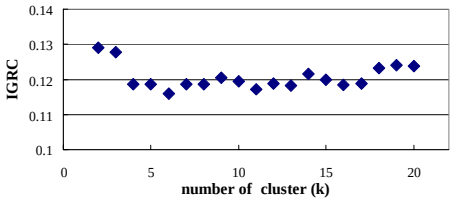

Fig. 16. IGRC in Exp.2 (with NFScore1)

4.2 Results

6 functions in Section 2 were used to create fragment spectra for the graphs in Exp.1 and Exp.2. The spectra were then clustered by K-means by setting K to the one with the maximal value of IGRC. If the structural properties of graphs are to be reflected on the number of nodes/edges and the configuration, we hypothesized that graphs should be categorized into clusters such that each cluster includes only the graphs which share the same base graph. Thus, the desirable value of K would be 3 and 4 for Exp.1 and Exp.2, respectively. The number of clusters K with the maximal value of IGRC in the experiments is summarized in Table 2. Changes of IGRC with NFScore1 for Exp.1 and Exp.2 are shown in Figs. 15 and 16, respectively.

In Exp.1, the desired value of K (K=3) was obtained with NFScore1 and NFScore3. In addition, the graphs were clustered as intended, in the sense that all the graphs which share the same base graph were categorized into the same cluster. On the other hand, except for FScore3, K=2 gave the maximal value of IGRC in Exp.2, and the hypothesized value (K=4) was not attained.

4.3 Discussions

From Table 2, it can be said that normalized score functions gave better clustering. The number of subgraphs in a graph G increases exponentially with respect to its size (the number of nodes and edges). Thus, as the size of graph increases,

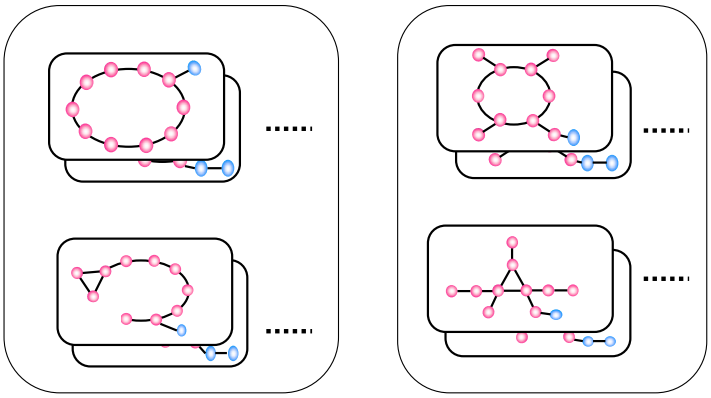

Fig. 17. Clustered graphs in Exp.2

the similarity of graphs which share the same base graph tends to decrease since the difference in the frequency of fragments with the same score gets larger in (1). This results in categorizing the graphs into different clusters. Normalization contributed to reducing this effect. On the other hand, with FScore2 and NFScore2 which calculate the degree of node within a (extracted) fragment, the maximal value of IGRC was obtained at K=2 for both Exp.1 and Exp.2. Thus, these functions were not effective for clustering graphs with respect to configuration. This indicated that considering the degree of node in the context of the original graph would be effective to reflect structural properties of fragment.

As described above, the value of IGRC was maximized at K=2 except for FScore3 in Exp.2. Fig. 17 illustrates the clustered graphs with NFScore1. As shown in Fig. 17, all the graphs which share the same base graph were assigned into the same cluster and they were not mixed up between different clusters. However, our criterion (maximization of IGRC) could not divide the clusters into smaller ones. The graphs in the left hand side in Fig. 14 were categorized into the same cluster. One possible conjecture is that a graph with ring structure can be rewritten into the other corresponding graph by removing one edge and adding it as another edge. If these graphs are considered as similar in terms of rewriting operation of graph[3], the desirable number of clusters can be considered as 3. In the previous result, the value at K=3 has the similar value at K=2 with NFScore1, albeit it was not the maximal. Still, the result of Exp.2 indicate that much work need to be done to improve our clustering method with respect to configuration.

[3] A kind of edit distance measure is used in Subdue [4] for inexact graph matching.

5 Concluding Remarks

This paper has described a graph clustering method based on structural similarity of fragments (currently connected subgraphs are considered) in graph structured data. The representation of a graph is transformed into a fragment spectrum, which represents the frequency distribution of fragments, in term of the the connectivity of a node in the fragment. The graphs are then clustered by applying a standard clustering method (K-means method) with respect to the transformed fragment spectra. The quality of cluster is estimated based on a pseudo-entropy of cluster in order to determine the number of clusters. Preliminary experiments with synthesized graphs were conducted and the results are reported. The results indicate that our method can cluster graph structure data with respect to the number of nodes and edges, but much work need to be done with respect to the configuration. Especially, the number of clusters tends to be under-estimated by our criterion. Currently our method only deals with superficial similarities in structure. We would like to extend our method to incorporate the label of node and edge so that resource combinations over the Web can be considered in terms of their contents.

Acknowledgments

The authors are grateful to the editors for fruitful discussions to refine this paper. They also gratefully acknowledge the enormous help by Takashi Matsuda for his advise on the implementation and experiments. This work was partially supported by the grant-in-aid for scientific research (No. 60002309, No. 13131206) funded by the Japanese Ministry of Education, Culture, Sport, Science and Technology.

References

1. L.A.N. Amaral, A. Scala, M. Barthélémy, and H. E. Stanley. Classes of small-world networks. *Proceedings of the National Academy of Sciences*, 97(21):11149–11152, 2000.
2. S. Chakrabarti. *Mining the Web: Discovering Knowledge from Hypertext Data*. Morgan Kaufmann, 2002.
3. P. Clark and Niblett T. The cn2 induction algorithm. *Machine Learning*, 3:261–283, 1989.
4. D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.
5. L. Dehaspe, H. Toivonen, and R. D. King. Finding frequent substructures in chemical compound. In *Proc. the 4th International conference on Knowledge Discovery and Data Mining*, pages 30–36, 1998.
6. A. Inokuchi, T. Washio, and H. Motoda. Complete mining of frequent patterns from graphs: Mining graph data. *Machine Learning*, 50(3):321–354, 2003.
7. M. Kuramochi and G.Karypis. Frequent subgraph discovery. In *Proc. of the 1st IEEE ICDM*, pages 313–320, 2001.

8. T. Matsuda, H. Motoda, T. Yoshida, and T. Washio. Mining patterns from structured data by beam-wise graph-based induction. In *Proc. of The Fifth International Conference on Discovery Science*, pages 422–429, 2002.
9. T. Matsuda, T. Yoshida, H. Motoda, and T. Washio. Beam-wise graph-based induction for structured data mining. In *International Workshop on Active Mining (AM-2002): working notes*, pages 23–30, 2002.
10. R. S. Michalski. Learning flexible concepts: Fundamental ideas and a method based on two-tiered representaion. *Machine Learning: An Artificial Intelligence Approach*, 3:63–102, 1990.
11. S. Muggleton and L. de Raedt. Inductive logic programming: Theory and methods. *Journal of Logic Programming*, 19(20):629–679, 1994.
12. S. Nomura, T. Miki, and T. Ishida. Comparative Study of Web Citation Analysis and Bibliographical Citation Analysis in Community Mining. *IEICE Transaction*, J87-D-I(3):382–389, 2004. (in Japanese).
13. C.R. Palmer, P.B. Gibbons, and C. Faloutsos. ANF: A fast and scalable tool for data mining in massive graphs. In *Proc. of the KDD-2002*, 2002.
14. J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, 1986.
15. J. R. Quinlan. *C4.5:Programs For Machine Learning*. Morgan Kaufmann Publishers, 1993.
16. J.W. Raymond, C.J. Blankley, and P. Willett. Comparison of chemical clustering methods using graph- and fingerprint-based similarity measures. *Molecular Graphics and Modelling*, 21(5):421–433, 2003.
17. Y. Takahashi, H. Ohoka, and Y. Ishiyama. Structural similarity analysis based on topological fragement spectra. *Adavances in Molecular Similarity*, 2:93–104, 1998.
18. D.J. Watts. *Small Worlds: The Dynamics of Networks Between Order and Randomness*. Princeton University Press, 2004.
19. D.J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, 1998.
20. T. Yoshida, G. Warodom, A. Mogi, K. Ohara, H. Motoda, T. Washio, H. Yokoi, and K. Takabayashi. Preliminary analysis of interferon therapy by graph-based induction. In *Working note of International Workshop on Active Mining (AM-2004)*, pages 31–40, 2004.

Appendix: Examples of Fragment Spectrum

Fragment spectra of graphs shown in Figs. 5, 6, 7 and 8. are shown in Figs. 18, 19, 20 and 21, respectively. For each type of graphs, horizontal axes (score of fragment) are aligned to the maximal score in the rightmost graph in each figure. To see the overall shape of spectra, normalized score functions are used to create fragment spectra and thus the maximal value of vertical axis is set to 1.0. Since the normalized frequency in NFScore1 is the same with the one in NFScore3 (except that x axis is rather stretched out since the score of each node is square summed), NFScore2 and NFScore3 are used to create the spectra. Figs. 18, 19, 20 and 21 indicate that fragment spectra of graphs with the same structure have similar shape or pattern and that fragment spectra of graphs with different structure also tend to have rather different patterns in terms of the corresponding spectra.

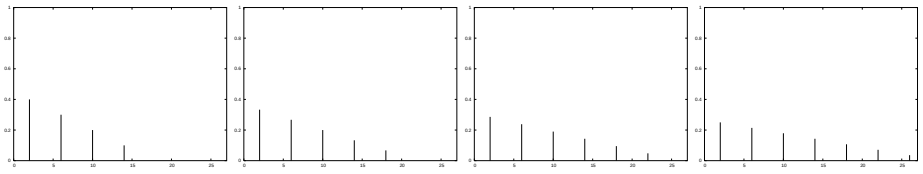

Fig. 18. Spectra for graphs with line structure (with NFScore2)

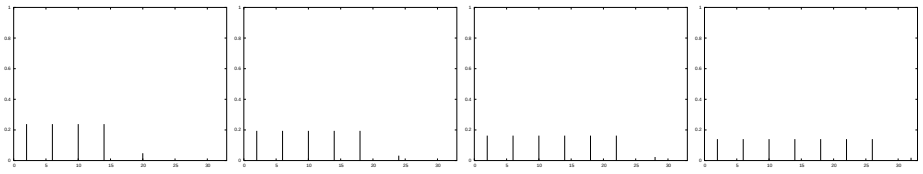

Fig. 19. Spectra for graphs with ring structure (with NFScore2)

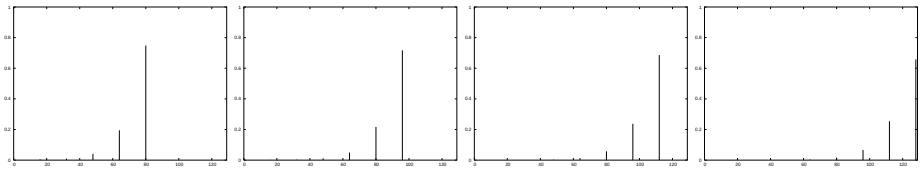

Fig. 20. Spectra for graphs with ring lattice structure (with NFScore3)

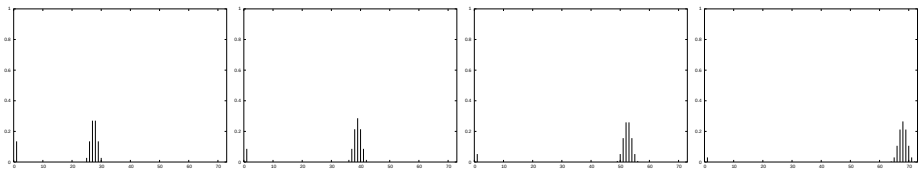

Fig. 21. Spectra for graphs with star structure (with NFScore3)