Partitioning of Web Graphs by Community Topology

Hidehiko Ino
Graduate School of
Information Science and
Technology
Hokkaido University
Sapporo, 060-0814 Japan
hino@main.ist.hokudai.ac.jp

Mineichi Kudo
Graduate School of
Information Science and
Technology
Hokkaido University
Sapporo, 060-0814 Japan
mine@main.ist.hokudai.ac.jp

Atsuyoshi Nakamura
Graduate School of
Information Science and
Technology
Hokkaido University
Sapporo, 060-0814 Japan
atsu@main.ist.hokudai.ac.jp

ABSTRACT

We introduce a stricter Web community definition to overcome boundary ambiguity of a Web community defined by Flake, Lawrence and Giles [2], and consider the problem of finding communities that satisfy our definition. We discuss how to find such communities and hardness of this problem.

We also propose Web page partitioning by equivalence relation defined using the class of communities of our definition. Though the problem of efficiently finding all communities of our definition is NP-complete, we propose an efficient method of finding a subclass of communities among the sets partitioned by each of $n-1$ cuts represented by a Gomory-Hu tree [10], and partitioning a Web graph by equivalence relation defined using the subclass.

According to our preliminary experiments, partitioning by our method divided the pages retrieved by keyword search into several different categories to some extent.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*data mining*; H.3.3 [**Information Systems**]: Information Storage and Retrieval—*clustering,information filtering*; G.2.2 [**Discrete Mathematics**]: Graph Theory—*network problems*

General Terms

Algorithms, Experimentation

Keywords

Web community, graph partitioning, maximum flow algorithm

1. INTRODUCTION

The World-Wide Web (WWW) is a huge network in which pages are connected by links. Inside this Web link structure, a lot of valuable information about the relationship between Web pages exists because Web links are created by People for the purpose of guidance to the related pages. Actually, this information has been already used for many application, for example, the ranking of pages retrieved by a search engine [6, 1].

WWW 2005, May 10-14, 2005, Chiba, Japan.
ACM 1-59593-046-9/05/0005.

One of the interesting substructures of the WWW network is a *community structure*, a structure of subgraphs with dense connections. A set of pages having such structure, a *Web community*, is conceivably created by people having the same interests. Therefore, discovered communities can be used in Web page search and recommendation.

There are mainly the following two detailed definitions of communities defined as dense subgraphs. One definition proposed by Kumar et al. [7] is that *a community is a dense directed bipartite subgraph which contains a complete bipartite subgraph of a certain size.* However, this definition is ambiguous because it still contains the word 'dense'. The definition would become clear if a community were defined as a complete bipartite subgraph itself, but then most communities would be too small and most pages would not belong to any communities. The other definition proposed by Flake, Lawrence, Giles [2] is that *a community is a vertex subset in which each member vertex has at least as many edges connecting to member vertices as it does to non-member vertices.* This definition is clear, and for every vertex subset, it is possible to decide whether it is a community or not. Besides, the possibility that large communities exist seems to be high, and most pages seem to belong to some communities that are not a whole set of vertices.

In order to find communities defined by Flake et al., which we call *FLG-communities* here, the following two methods were proposed so far, and both methods are trying not to find densely-connected vertex subsets but to find those sparsely connected to their outside vertices. One is a method based on *edge betweenness* proposed by Girvan and Newman [4]. The edge betweenness of an edge is defined as the number of shortest paths between pairs of vertices that run along it. Based on the idea that the edges connecting the inside and the outside of a community are expected to have high edge betweenness, Girvan and Newman proposed a method of removing an edge with the highest edge betweenness one by one. However, it is not guaranteed that FLG-communities are found by their method[1]. The other is a method using the *maximum flow algorithm* proposed by Flake et al. [2]. The maximum flow algorithm [9, 10] is an algorithm that calculates how much water must run through each edge in order to maximize the total amount of water running from a vertex (*source*) to another vertex (*sink*) on condition that the amount of water through each edge must be at most its given capacity. The method using the maxi-

[1] For example, in Figure 2, their method divides C_3 and $V - C_3$, and does not find the FLG-community C_4.

mum flow algorithm is based on the idea that sparse edges between the inside and the outside of an FLG-community become a bottleneck of the flow from an inside vertex to an outside vertex when the capacity of every edge is one. A subset of saturated edges, the edges through which the amount of water equal to its capacity is running, gives a *cut* that divides the set of all vertices into two sets, a set containing the source and a set containing the sink. Ford and Fulkerson's "max flow-min cut" theorem [9, 10] guarantees that this cut contains the minimum number of edges among all cuts dividing the source and the sink. Flake et al. claimed that, in order to be identified by the maximum flow algorithm, an FLG-community C must satisfy the condition that both $s^{\#}$ and $t^{\#}$ are larger than the number of edges in the cut $(C, V - C)$, where $s^{\#}$ is the number of edges between the source s and vertices in C, and $t^{\#}$ is the number of edges between the sink t and vertices in $V - C$.

There are two problems for the method proposed by Flake et al. First, the boundary of an FLG-community is ambiguous, that is, many slightly different subsets could be FLG-communities for one densely-connected part of a graph. Second, a vertex set found by a maximum flow algorithm might not be an FLG-community even if an FLG-community C satisfying the above condition exists.

To overcome the first problem, this paper introduces a stricter community definition than the FLG-community definition. In the FLG-community definition, only inside vertices of a community are conditioned. In our community definition, it must be also satisfied that each outside vertex has at least as many edges connecting to outside vertices as it does to inside vertices. (We call an FLG-community satisfying this condition an *IKN-weak-community*.) In addition to this condition, inside vertices must satisfy a stricter condition that each inside vertex has *more* edges connecting to inside vertices *than* it does to outside vertices. (We call an IKN-weak-community satisfying this condition an *IKN-community*.) The definition of IKN-community reduces boundary ambiguity of FLG-community to some extent because two distinct IKN-communities differ in at least two vertices.

A clarification of what can be found by an *s*-*t* maximum flow algorithm approaches the second problem. We prove that what can be found by an *s*-*t* maximum flow algorithm is not an FLG-community but a *s*-*t* *quasi-IKN-community* in which all vertices but the two vertices s and t satisfy the conditions of IKN-community. In terms of FLG-communities, this means that a maximum flow algorithm always finds a vertex set whose members but the source s satisfy the condition of (strict[2]) FLG-community, which has been already proved by Flake, Tarjan and Tsioutsiouliklis [3]. So, an *s*-*t* maximum flow algorithm might seem to approximately find an FLG-community. However, the fact that a source might not satisfy the condition should not be neglected because a source is the most important vertex which must be contained in the community. On the other hand, the fact that all members but the source s (and the sink t) in a found set satisfies the conditions of FLG-community (IKN-community) ensures that only s (and t) should be checked to satisfy the conditions to know whether it is an FLG-community (IKN-community) or not.

An efficient algorithm to find not an *s*-*t* quasi-IKN-community but an *s*-*t* IKN-community (an IKN-community that includes s and excludes t) has not been known yet. As a difference between the problem of finding an *s*-*t* IKN-weak-community and the problem of finding an *s*-*t* quasi-IKN-community, which can be solved efficiently by an *s*-*t* maximum flow algorithm, we show the fact that each coordinate of every extreme point solution is an integer for the LP-relaxation of the integer programming in a formalization of the latter problem, but might not be an integer for that in a formalization of the former problem. This seems to support the hardness of the problem of finding an *s*-*t* IKN-community. Actually, very recently, the problem of deciding whether an *s*-*t* IKN-community (IKN-weak-community) exists or not for given s and t has been proved to be NP-complete even if weights are restricted to be 1 [8]. The problem of deciding whether an IKN-community (IKN-weak-community) exists or not in a given graph has been also proved to be NP-complete [3], which implies NP-completeness of the above *s*-*t* IKN-community (IKN-weak-community) problem without restriction on weights.

Partitioning Web pages into groups having similar properties is useful for information retrieval, and Web communities can be used for this purpose. We propose Web page partitioning by the equivalence relation defined using the class of IKN-communities, where two pages are equivalent if and only if the sets of IKN-communities including each page coincide. This equivalence relation can be also defined by using FLG-communities, but a partition obtained by the relation may not be useful for the boundary ambiguity problem described above. We also propose hierarchical partitioning by repeatedly applying this partitioning to the contracted graph in which all original vertices in the same partition are contracted into one vertex.

In order to partition a Web graph by the equivalence relation defined using IKN-communities, the existence of *s*-*t* IKN-communities should be checked for arbitrary vertices s and t, which is an NP-complete problem as described above. In this paper, we propose a coarser partitioning by the equivalence relation defined using a subclass of IKN-communities which are efficiently extracted by an *s*-*t* maximum flow algorithm. As mentioned above, sets found by an *s*-*t* maximum flow algorithm are *s*-*t* quasi-IKN-communities, so only a further check if s and t satisfy the requirements of being an IKN-community is needed. Our method find IKN-communities among all $2(n - 1)$ vertex subsets partitioned by one of $n-1$ *cuts* represented by a *Gomory-Hu tree* [10]. A Gomory-Hu tree of a connected graph $G(V, E)$ is a tree with a set of vertices V in which every cut $(C, V - C)$ obtained by removing one edge (s, t) is also a *s*-*t* minimum cut in G. It is known that a Gomory-Hu tree can be created efficiently by executing maximum flow algorithms $n - 1$ times.

According to our preliminary experiments, partitioning by our method divided the pages retrieved by keyword search into several different categories to some extent.

2. WEB COMMUNITIES

2.1 Definition

Let an undirected graph $G(V, E)$ be a graph in which each vertex represents a Web page and each edge represents a link between two distinct pages. Assume that a weight $w_{uv}(= w_{vu}) \geq 0$ is given to each pair of vertices u and v, and $w_{uv} = 0$ if there is no edge between u and v. A weight

[2] See the definition in Sec. 2.1

w_{uv} for a pair of vertices u and v can be any value when an edge exists between them, but we assume that it is 1 unless explicitly stated otherwise.

Flake, Lawrence and Giles [2] defined a web community, which we call an *FLG-community* here, in terms of undirected graphs as follows.

Definition 1 (Flake, Lawrence and Giles [2]). *An FLG-community is a vertex subset $C \subset V$ that satisfies the following Condition 1.*

Condition 1. $\sum_{v\in C} w_{uv} \geq \sum_{v\in V-C} w_{uv}$ *for all* $u \in C$.

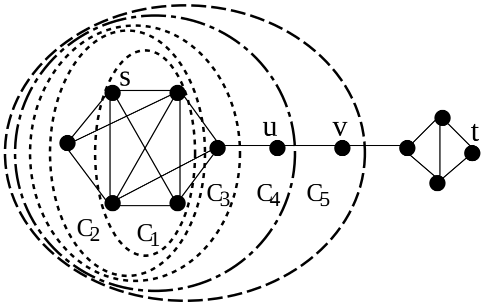

Figure 1: Example of FLG-Communities $C_1, C_2, ..., C_5$.

This definition has the problem of boundary ambiguity. For example, in the graph of Figure 1, C_1, C_2, C_3, C_4 and C_5 are all FLG-communities, though they are essentially the same densely connected part of the graph. This raises the problem of boundary ambiguity when we want to extract one community for one densely connected part. Therefore, we propose a stricter definition as follows.

Definition 2. *An **IKN-weak-community** is a vertex subset $C \subset V$ that satisfies Condition 1 and the following Condition 2.*

Condition 2. $\sum_{v\in C} w_{uv} \leq \sum_{v\in V-C} w_{uv}$ *for all* $u \in V - C$.

Definition 3. *An **IKN-community** is a vertex subset $C \subset V$ that satisfies the following Condition $1'$ and Condition 2.*

Condition $1'$. $\sum_{v\in C} w_{uv} > \sum_{v\in V-C} w_{uv}$ *for all* $u \in C$.

Flake, Tarjan and Tsioutsiouliklis [3] consider a community definition using Condition $1'$ only. Here, we call such communities *strict FLG-communities.*

Note that

$$\begin{aligned} & \{C : C \text{ is an IKN-community}\} \\ \subseteq\ & \{C : C \text{ is an IKN-weak-community}\} \\ \subseteq\ & \{C : C \text{ is an FLG-community}\}. \end{aligned}$$

The whole set V and the empty set $\emptyset$ are trivial IKN-communities. Vertex subsets C_3, C_4 and C_5 are IKN-weak-communities among 5 FLG-communities $C_1, C_2, ..., C_5$ in Figure 1, and only C_3 is an IKN-community. Note that C_1, C_2 and C_3 are strict FLG-communities. The following proposition ensures that if C is an IKN-community, $C \cup \{v\}$ for any $v \notin C$ and $C - \{v\}$ for any $v \in C$ is not an IKN-community, which means that boundary ambiguity is reduced.

Proposition 1. *For any two distinct IKN-communities C_1 and C_2, their symmetric difference contains at least two vertices.*

Proof. Let $u \notin C_1$ and $C_2 = C_1 \cup \{u\}$. Since C_2 satisfies Condition $1'$, $\sum_{v\in C_1} w_{uv} = \sum_{v\in C_2} w_{uv} > \sum_{v\in V-C_2} w_{uv} = \sum_{v\in V-C_1} w_{uv}$, which contradicts the assumption that C_1 satisfies Condition 2. □

2.2 How to Find Web Communities

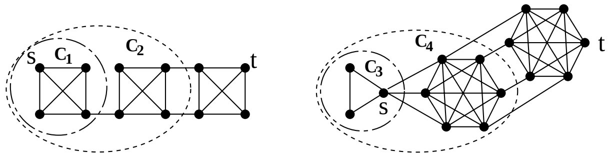

Figure 2: C_2 and C_4 are communities that are not found by an s-t maximum flow algorithm.

Flake, Lawrence and Giles proposed a method that uses a maximum flow algorithm to find an FLG-community. As shown by them, it is a clear fact that an s-t maximum flow algorithm fails to identify an FLG-community C which includes a vertex s and excludes a vertex t, when the number of edges between C and $V - C$ is larger than the number of edges $s^{\#}$ between s and $C - \{s\}$, or the number of edges $t^{\#}$ between t and $V - C - \{t\}$. Then, when the number of edges between C and $V - C$ is smaller than both $s^{\#}$ and $t^{\#}$, can C identified by an s-t maximum flow algorithm? The answer is no. In the case of the left graph in Figure 2, C_1 and C_2 satisfy the condition but the FLG-community found by an s-t maximum flow algorithm is C_1 only. In the case of the right graph in Figure 2, C_4 satisfies the condition but the found set is C_3, which is not even an FLG-community.

These examples indicate the following two facts for communities C that include s, exclude t, and have the number of edges between C and $V - C$ which is smaller than both $s^{\#}$ and $t^{\#}$.

- Not all such communities C can be found by an s-t maximum flow algorithm.
- Non-FLG-communities can be found by an s-t maximum flow algorithm even if such communities C exist.

The second fact make us reluctant to use the algorithm for finding FLG-communities.

In the followings, we make clear what can be found by a maximum flow algorithm.

Let $G' = (V, E')$ be a directed graph generated from G by replacing each undirected edge (u, v) to two directed edges (u, v) and (v, u). Given two vertices s and t, let $f_{s,t}$ denote a *maximum flow* [9] from s to t when the capacity of each edge (u, v) is w_{uv}. We define $S(f_{s,t})$ as the subset of vertices that are reachable from s in the residual graph [9] of G' for $f_{s,t}$. Note that a residual graph of $G' = (V, E')$ for f is defined as the graph that is composed of all vertices and the edges (u, v) with positive residual capacity $r_{uv} = w_{uv} - f(u, v) + f(v, u)$.

We call an IKN-community that includes s and excludes t an *s-t IKN-community*. If a subset of vertices that includes s and excludes t satisfies Condition $1'$ and Condition 2 except s and t, then we call the subset an *s-t quasi-IKN-community*. Then, the following proposition holds.

PROPOSITION 2. *$S(f_{s,t})$ is an s-t quasi-IKN-community.*[3]

PROOF. Let $u \in S(f_{s,t})$ and $u \neq s$. Since there is no edge from u to any vertices in $V - S(f_{s,t})$ in the residual graph for $f_{s,t}$, $f_{s,t}(u, v) = w_{uv}$ and $f_{s,t}(v, u) = 0$ hold for all $v \in V - S(f_{s,t})$. Thus,

$$\begin{aligned}\sum_{v\in V-S(f_{s,t})} w_{uv} &= \sum_{v\in V-S(f_{s,t})} f_{s,t}(u,v)\\ &\leq \sum_{v\in V} f_{s,t}(u,v) = \sum_{v\in V} f_{s,t}(v,u)\\ &= \sum_{v\in S(f_{s,t})} f_{s,t}(v,u)\\ &\leq \sum_{v\in S(f_{s,t})} w_{vu} = \sum_{v\in S(f_{s,t})} w_{uv}\end{aligned}$$

holds. If equality $\sum_{v\in V-S(f_{s,t})} w_{uv} = \sum_{v\in S(f_{s,t})} w_{uv}$ holds, $f_{s,t}(u, v) = 0$ and $f_{s,t}(v, u) = w_{vu}$ hold for all $v \in S(f_{s,t})$, which means that there is no edge from any vertices in $S(f_{s,t})$ to u in the residual graph. This contradict the fact that u belongs to $S(f_{s,t})$. Thus, all vertices in $S(f_{s,t}) - \{s\}$ satisfies Condition $1'$.

Similarly, it can be proved that all vertices in $V - S(f_{s,t}) - \{t\}$ satisfies Condition 2. □

REMARK 1. *For $u = s, t$, $\sum_{v\in V} f_{s,t}(s, v) = \sum_{v\in V} f_{s,t}(v, s) + |f_{s,t}|$ and $\sum_{v\in V} f_{s,t}(t, v) + |f_{s,t}| = \sum_{v\in V} f_{s,t}(v, t)$ hold instead of $\sum_{v\in V} f_{s,t}(u, v) = \sum_{v\in V} f_{s,t}(v, u)$, where $|f_{s,t}|$ is the value of flow $f_{s,t}$. Therefore, $S(f_{s,t})$ might not be an s-t IKN-community.*

REMARK 2. *Cut $(S(f_{s,t}), V - S(f_{s,t}))$ is one of the s-t minimum cuts [9], but sets C and $V - C$ for an s-t minimum cut $(C, V - C)$ might not be s-t and t-s quasi-IKN-communities. For example, cut $C_4, V - C_4$ in Figure 1 is an s-t minimum cut but neither of them is an s-t or t-s quasi-IKN-community because u for C_4 and v for $V - C_4$ do not satisfy Condition $1'$. But for all s-t minimum cut $(C, V - C)$, both C and $V - C$ are s-t* quasi-IKN-weak-communities, *where an s-t quasi-IKN-weak-community is a vertex subset that includes s, excludes t and satisfies Condition 1 and Condition 2.*

Note that Proposition 2 indicates that we only have to check that s satisfies Condition $1'$ and t satisfies Condition 2 in order to know whether $S(f_{s,t})$ is an IKN-community or not.

As shown above, an *s-t* maximum flow algorithm only guarantees that a found set is an *s-t* quasi-IKN-community, and an efficient algorithm that finds *s-t* IKN-community has not been known yet. Note that *s-t* quasi-IKN-communities always exist but *s-t* IKN-communities might not exist.

In the followings, we show one evidence that the problem of finding an *s-t* IKN-community looks computationally hard. The problem can be formalized as the following integer program that is obtained by adding conditions to an integer program [10] in a formalization of the minimum cut problem, of which LP-relaxation is the LP-dual program of a linear program in a formalization of the maximum flow problem.

PROBLEM 1. *Minimize* $\sum_{(u,v)\in E'} w_{uv} d_{uv}$

subject to

$$d_{uv} - p_u + p_v \geq 0 \text{ for } (u, v) \in E' \quad (1)$$

$$p_s - p_t \geq 1 \quad (2)$$

$$-\sum_{v:(u,v)\in E'} w_{uv} d_{uv} \geq -\frac{1}{2} \sum_{v:(u,v)\in E'} w_{uv} \text{ for } u \in V \quad (3)$$

$$-\sum_{u:(u,v)\in E'} w_{uv} d_{uv} \geq -\frac{1}{2} \sum_{u:(u,v)\in E'} w_{uv} \text{ for } v \in V \quad (4)$$

$$d_{uv} \in \{0, 1\} \text{ for } (u, v) \in E' \quad (5)$$

$$p_u \in \{0, 1\} \text{ for } u \in V \quad (6)$$

The following proposition holds.

PROPOSITION 3. *Problem 1 has a feasible solution. ⇔ An s-t IKN-weak-community exists.*

PROOF. (⇒) Let $\{p^*_u, d^*_{u,v} : u \in V, (u, v) \in E'\}$ be an optimal solution. let $C = \{u : p^*_u = 1\}$. Note that $s \in C$ and $t \notin C$ because $p^*_s - p^*_t \geq 1$. Then, the optimality leads that

$$d^*_{uv} = \begin{cases} 1 & \text{if } p^*_u = 1 \text{ and } p^*_v = 0 \\ 0 & \text{otherwise.} \end{cases}$$

For $u \in C$,

$$\sum_{v\in V-C} w_{uv} = \sum_{v\in V} w_{uv} d^*_{uv} \leq \frac{1}{2} \sum_{v\in V} w_{uv}$$

holds by Inequality (3). Thus, C satisfies Condition 1. For $v \in V - C$,

$$\sum_{u\in C} w_{uv} = \sum_{u\in V} w_{uv} d^*_{uv} \leq \frac{1}{2} \sum_{u\in V} w_{uv}$$

holds by Inequality (4). Thus, C satisfies Condition 2.
(⇐) For an *s-t* IKN-community C, set p_u for $u \in V$ and d_{uv} for $(u, v) \in E'$ as follows.

$$p_u = \begin{cases} 1 & \text{if } u \in C \\ 0 & \text{otherwise} \end{cases}$$

$$d_{uv} = \begin{cases} 1 & \text{if } p_u = 1 \text{ and } p_v = 0 \\ 0 & \text{otherwise.} \end{cases}$$

Then, p_u for $u \in V$ and d_{uv} for $(u, v) \in E'$ are a feasible solution of Problem 1. □

The following is the LP-relaxation of Problem 1.

PROBLEM 2. *Minimize* $\sum_{(u,v)\in E'} w_{uv} d_{uv}$

[3] A part of this proposition, namely, a claim that all vertices in $S(f_{s,t}) - \{s\}$ satisfy Condition $1'$, has been already proved by Flake, Tarjan and Tsioutsiouliklis [3].

subject to

$$d_{uv} - p_u + p_v \geq 0 \; for \; (u,v) \in E' \quad (7)$$

$$p_s - p_t \geq 1 \quad (8)$$

$$-\sum_{v:(u,v)\in E'} w_{uv} d_{uv} \geq -\frac{1}{2} \sum_{v:(u,v)\in E'} w_{uv} \; for \; u \in V \quad (9)$$

$$-\sum_{u:(u,v)\in E'} w_{uv} d_{uv} \geq -\frac{1}{2} \sum_{u:(u,v)\in E'} w_{uv} \; for \; v \in V \quad (10)$$

$$d_{uv} \geq 0 \; for \; (u,v) \in E' \quad (11)$$

$$p_u \geq 0 \; for \; u \in V \quad (12)$$

For the corresponding LP-relaxation of the minimum cut problem, the conditions represented by Inequalities (9) and (10) are not needed, and it has been proved that each coordinate of every extreme point solution is 0 or 1 [10], which is not true for the Problem 2. Actually, the optimal solutions of Problem 1 do not coincide with those of Problem 2 as shown in Figure 3.

As hardness results on finding an *s*-*t* IKN-community, stronger results have been proved very recently. In [8], NP-completeness was proved for the problem of deciding whether an *s*-*t* IKN-community (IKN-weak-community) exists or not for given s and t even if weights are restricted to be 1. Flake, Tarjan and Tsioutsiouliklis [3] also proved NP-completeness for the problem of deciding whether an IKN-community (IKN-weak-community) exists or not in a given graph, which implies NP-completeness of the above *s*-*t* IKN-community (IKN-weak-community) problem without restriction on weights.

The following proposition, which claims that an IKN-community can be constructed from a set satisfying only Condition $1'$ or Condition 2, may help the task of Web community discovery. The constructed set might be a whole set or an empty set.

PROPOSITION 4. *(1) For any subset $C \subseteq V$ that satisfies Condition $1'$, the minimum IKN-community including C can be constructed.*
(2) For any subset $C \subseteq V$ that satisfies Condition 2, the maximum IKN-community included in C can be constructed.

PROOF. (1) Let $B_{out}(C)$ denote the set of vertices in $V - C$ that does not satisfy Condition 2. Let $C_0 = C$, and define C_{i+1} as $C_i \cup B_{out}(C_i)$. Then, C_{i+1} also satisfies Condition $1'$ when C_i satisfies that condition. Since C_0 satisfies the condition, all C_i satisfy it. The sequence $C_0, C_1, ...$ is monotonically increasing and $|V| < \infty$, so there exists n_0 such that $C_n = C_{n_0}$ for all $n \geq n_0$. Then, C_{n_0} is an IKN-community because $B_{out}(C_{n_0}) = \emptyset$. Since the minimum community including C_i trivially contain $B_{out}(C_i)$, C_{n_0} is the minimum IKN-community including C.
(2) This can be proved similarly. □

If both of two sets C_1 and C_2 satisfy Condition $1'$, then $C_1 \cup C_2$ also satisfies that condition, thus we can also construct the minimum IKN-community containing both of them. Similarly, we can find the maximum IKN-community contained in both of two sets that satisfy Condition 2. Therefore, different IKN-communities can be found by using a set of IKN-communities. (See Figure 4.)

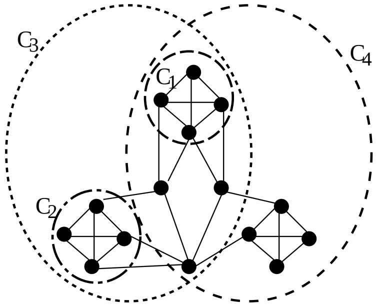

Figure 4: C_3 is the minimum IKN-community containing $C_1 \cup C_2$. C_1 is the maximum IKN-community contained in $C_3 \cap C_4$.

3. GRAPH PARTITIONING BY COMMUNITY TOPOLOGY

3.1 Definition

Let $\mathcal{C}$ be a class of subsets of V in a graph $G = (V, E)$. For each vertex u, define the class $\mathcal{U}(u)$ of neighborhoods of u as the set of subsets in $\mathcal{C}$ that contain u, that is, $\mathcal{U}(u) = \{C : C \in \mathcal{C}, u \in C\}$. Consider the following equivalence relation R:

$$uRv \overset{def}{\Leftrightarrow} \mathcal{U}(u) = \mathcal{U}(v).$$

We call partitioning by this equivalence relation *partitioning by $\mathcal{C}$*. In this paper, we consider partitioning by the class of IKN-communities.

By regarding each equivalence class as one vertex, we can generate a contracted graph. We can obtain a higher level partitioning by the class of IKN-communities in the contracted graph. This procedure can be repeated until every equivalence class becomes composed of only one vertex.

In this paper, we propose a hierarchical partitioning by the class of IKN-communities through repeating partitioning and contraction.

See Figure 5 for an example of partitioning by the class of IKN-communities and a contracted graph for a partitioning.

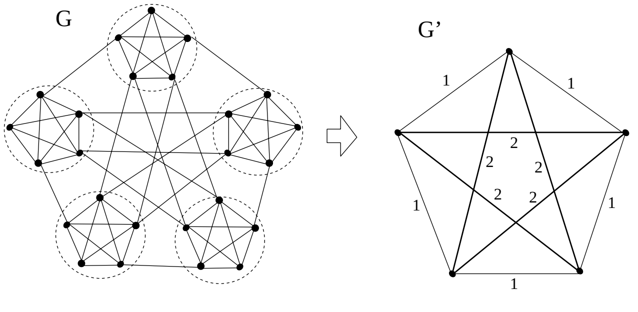

Figure 5: Left: Partitioning by the class of IKN-communities (Every edge weight is one.), Right: Contracted graph of the left graph.

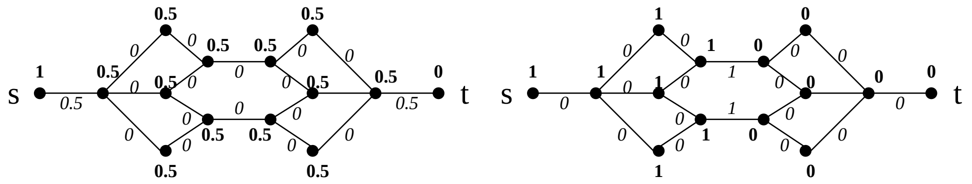

Figure 3: Right: Optimal solution of Problem 2 (value of objective function:1), Left: Optimal solution of Problem 1 (value of objective function:2). Bold numbers represent p_u, and italic numbers represent d_{uv}. Note that d_{uv} for each edge (u, v) directed to left, which is 0, is omitted.

3.2 Efficient Algorithm for Partitioning

Considering NP-completeness for the problem of deciding whether an s-t IKN-community exists or not for arbitrary two vertices s and t, partitioning by the whole class of IKN-communities is not practical. Here, we propose a method that finds a subclass of IKN-communities efficiently and thus generates a coarse partitioning by the subclass.

The maximum flow algorithms are efficient but IKN-communities might not be found by them as commented in Remark 1. However, the sets found by them are always s-t quasi-IKN-communities C, so we only have to check that the source s satisfies Condition $1'$ and the sink t satisfies Condition 2 to decide whether C is an IKN-community or not. Here, we consider the problem of finding the subclass of IKN-communities that can be found by an s-t maximum flow algorithm for all pairs of vertices s and t.

For an undirected graph $G(V, E)$ with n vertices, existence of a set $\mathcal{D}$ of $n - 1$ cuts that satisfies the following property is known [10].

Property 1 For arbitrary two vertices s and t, there exists an s-t minimum cut in $\mathcal{D}$.

Furthermore, there exists a set $\mathcal{D}$ that has Property 1 and is represented by a tree T called a *Gomory-Hu tree* [10] which is composed of vertices in V and $n - 1$ edges representing $n-1$ cuts in $\mathcal{D}$. A Gomory-Hu tree can be efficiently created by repeatedly using a maximum flow algorithm $n - 1$ times. We propose a method that finds IKN-communities among the sets partitioned by the $n - 1$ minimum cuts represented by a Gomory-Hu tree.

Note that there might be minimum cuts which are not represented by one Gomory-Hu tree, and there might exist other Gomory-Hu trees representing different set of minimum cuts for the same graph. Thus, our proposed method does not check whether $S(f_{s,t})$ is an IKN-community for all $s, t \in V$. In order to raise the possibility of finding IKN-communities, we adopt the following heuristics.

Define $T(f_{s,t})$ as the set of vertices from which t is reachable in the residual graph for a flow $f_{s,t}$. Then, $T(f_{s,t})$ can be also proved to be a t-s quasi-IKN-community. Note that $T(f_{s,t})$ can coincide with $V - S(f_{s,t})$ but is different generally. In the process of constructing a Gomory-Hu tree, we check both sets $S(f_{s,t})$ and $T(f_{s,t})$ after every finding maximum flow $f_{s,t}$, and adopt the one that is an IKN-community as a minimum cut, if either $S(f_{s,t})$ or $T(f_{s,t})$ is an IKN-community.

For example, the leftmost graph in Figure 6 has Gomory-Hu trees expressed by the bottom graphs in the figure that represent different sets of minimum cuts, which are also drawn by broken lines in the corresponding top graphs of the figure. Note that the middle Gomory-Hu tree does not have the minimum cut $\{0, 1\}, \{2, 3, 4, 5\}$, but the rightmost one does, where set $\{2, 3, 4, 5\}$ is an IKN-community. Note that, for this graph, the rightmost Gomory-Hu tree is constructed by the above heuristics.

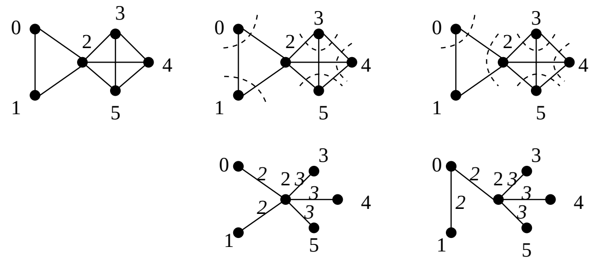

Figure 6: Example of Gomory-Hu trees representing different set of minimum cuts.

4. PRELIMINARY EXPERIMENTS

Keyword	learning theory		jaguar	
Level	0	1	0	1
#Vertex	2919	366	2834	559
#Edge	6015	147	19226	169
#(Isolated vertex)	240	258	377	420
#(Connected component)	22	4	52	9
#IKN-community	117	12	147	20
#(Equivalence class)	126	12	182	27

Table 1: The number of vertices, edges, isolated vertices, and connected components (that are not isolated vertices) in two graphs created by the procedure Subgraph, and the number of found IKN-communities (that are not connected components themselves), and equivalence classes (that are not isolated vertices) partitioned by the class of the found IKN-communities.

We conducted experiments on graph partitioning by a subclass of IKN-communities, using subgraphs of the WWW that are composed of the pages related to given keywords. For given keywords, we construct a subgraph by using the *Subgraph* procedure proposed by Kleinberg [6], which retrieves t pages by using a search engine and adds all pages that are linked from or linking to at least one of them, though the number of pages linking to is restricted within

```
(#pages:1590)
┬┬─ 0  tip.psychology.org (Theory Into Practice (TIP))
│├─ 2  www.usask.ca/education/coursework/802papers/mergel/brenda.htm (Learning Theories of Instructional Design)
│├─ 4  www.funderstanding.com/about_learning.cfm (Funderstanding - About Learning)
│└ other 928 pages(45 pages)
├┬─ 1  tip.psychology.org/theories.html ( TIP: The Theories )
│└ other 10 pages(0 pages)
├┬─ 3  www.ozline.com/learning/theory.html (ozline - Working the Web for Education)
│└ other 58 pages(0 pages)
├┬─ 7  www.exploratorium.edu/IFI/resources/research/constructivistlearning.html (Constructivist Learning Theory)
│└ other 2 pages(0 pages)
├┬─ 8  www.educationau.edu.au/archives/cp/04.htm (Learning Theories)
│└ other 7 pages(0 pages)
└ other 53 partitions

(#pages:656)
┬┬─ 10  www.learningtheory.org (COLT: Computational Learning Theory)
│├─ 59  www.crm.es/Activities/Act2003-04/LearningTheory/LearningTheoryhome.htm (Mathematical Foundation of Learning Theory)
│├─ 61  theory.lcs.mit.edu/COLT-98
│└ other 462 pages(9 pages)
├┬─ 19  plato.stanford.edu/entries/learning-formal (Formal Learning Theory)
│└ other 2 pages(0 pages)
├┬─ 43  www.cis.udel.edu/ case/colt.html ( John Case's COLT Page)
│└ other 19 pages(0 pages)
├┬─ 51  liinwww.ira.uka.de/bibliography/Ai/colt.html (Bibliography on Computational and Algorithmic Learv...)
│└ other 8 pages(0 pages)
├┬─ 52  www.esat.kuleuven.ac.be/sista/natoasi/ltp2002.html (NATO-ASI LTP2002)
│└ other 15 pages(0 pages)
└ other 15 partitions

(#pages:179)
┬┬─ 35  w4.evectors.it/itEntDirectory/topic?topic=learning_theory (w4)
│├─106 www.unimelb.edu.au/HB/subjects/468-110.html (468-110 Advanced Learning Theory)
│└ other 125 pages(0 pages)
└ other 6 partitions

(#pages:1)
──── 40  psych.fullerton.edu/jmearns/book3.htm (Applications of a Social Learning Theory of Personv...)

(#pages:6)
┬┬─ 42  www.acm.org/sigchi/chi96/proceedings/papers/Soloway/es_txt.htm (Learning Theory in Practice: Case Studies of Learv...)
│└ other 4 pages(0 pages)
└ other 1 partitions
─ other 265 partitions
```

Figure 7: Result for keywords "learning theory."

d pages. In our experiment, we used search engine Google (www.google.co.jp), and set *t* and *d* to 200 and 50, respectively. Note that we removed all *intrinsic* links [6], namely, links to pages of the same domain, as Kleinberg did. By the procedure Subgraph, graphs with the number of vertices, edges, isolated vertices and connected components shown in Table 1 are obtained for two keywords "learning theory" and "jaguar".

We hierarchically partitioned the obtained graphs by using our proposed graph partitioning algorithm two times. Namely, in the level-1 partitioning, we used a contracted graph that is created by regarding each equivalence class obtained in the level-0 partition as one vertex. As shown in Table 1, 117 and 147 IKN-communities are found in the level-0 partitioning, and 12 and 20 IKN-communities are found in the level-1 partitioning.

The results of hierarchical partitioning for the two graphs are shown in Figure 7 and Figure 8. We ranked each partition by the highest Google rank of the pages included in the partition. In terms of this ranking, the figures show the top 5 level-1 partitions and the top 5 level-0 partitions for each of those level-1 partitions. The pages accompanied with their URLs and titles are ones of which Google rank is within 200 and is within the top 3 among the members of each level-0 equivalence class. The number of pages whose URLs and titles are not described is shown after the word "other", and the number of those pages whose Google rank is within 200 is also shown in the following parentheses.

Figure 7 is the result for keywords "learning theory". There are mainly two learning theories, one is in the area of education and psychology, the other is computational learning theory. The second level-1 partition consists of the pages related to computational learning theory, and the other level-1 partitions consist of the pages related to edu-

```
(#pages:613)
┬┬─  0  www.jaguar.com/global/default.htm (Jaguar Cars)
│├─120  jaguar.anort.com
│├─152  chulkov.com/jaguar/jaguar.htm (&#160;JAGUAR X-Type Best in the world the automobiv...)
│└─ other 66 pages(0 pages)
├┬─  2  www.jaguar-racing.com
│└─ other 34 pages(0 pages)
├┬─  3  www.jaguar-racing.com/uk/flash
│└─ other 6 pages(0 pages)
├┬─  6  www.jaguarcars.com (Jaguar Cars)
│└─ other 44 pages(0 pages)
├┬─  7  www.jaguarcars.com/jp
│├─  8  www.jag-lovers.org (Jag-lovers - here to provide everything for the Jav...)
│├─  9  www.jag-lovers.org/brochures (Jag-lovers unique original Brochures and Adverts cv...)
│└─ other 207 pages(30 pages)
└─ other 33 partitions

(#pages:1)
───  1  www.jaguar.com/de/de/home.htm (home)

(#pages:984)
┬┬─  4  www.apple.com/macosx (Apple - Mac OS X)
│├─  5  www.apple.com/macosx/overview (Apple - Mac OS X - Overview)
│├─ 44  www.macattorney.com/tutorial.html (OS X 10.2 Jaguar Troubleshooting)
│└─ other 522 pages(7 pages)
├┬─ 11  www.schrodinger.com (Schr&ouml;dinger)
│└─ other 12 pages(0 pages)
├┬─ 12  www.schrodinger.com/Products/jaguar.html (Schr&ouml;dinger: Jaguar Program)
│└─ other 10 pages(0 pages)
├┬─ 16  www.jaguarmodels.com (Jaguar Models - Main Page (resin model kits))
│└─ other 4 pages(0 pages)
├┬─ 35  www.oneworldjourneys.com/expeditions/jaguar (One World Journeys — Jaguar: Lord of the Mayan Junv...)
│└─ other 5 pages(0 pages)
└─ other 25 partitions

(#pages:7)
┬┬─ 19  dspace.dial.pipex.com/agarman/jaguar.htm (Jaguar)
│└─ other 1 pages(0 pages)
├┬─ 37  www.bluelion.org/jaguar.htm (Panthera onca)
│└─ other 2 pages(0 pages)
└─ other 1 partitions

(#pages:1)
─── 20  dspace.dial.pipex.com/agarman/bco/jaguar.htm (Jaguar)
─ other 443 partitions
```

Figure 8: Result for keyword "jaguar."

cational learning theory, though those partitions are small except the first one. The number preceding each URL shows its Google rank, and you can see that the top 10 pages of Google search result is biased toward educational learning theory. Our result indicates that partitioning by IKN-communities can be used to produce balanced search results.

Figure 8 is the result for keyword "jaguar". The top 2 level-1 partitions are related to the automobile, the third one is mainly composed of pages related to the computer and the fourth and fifth ones are related to the animal. In the computer partition, the first level-0 partition, whose members are in majority, is related to the Mac OS, the second and third level-0 partitions consist of the pages of a software company producing a package called Jaguar, and the fourth and fifth level-0 small partitions are composed of pages not related to the computer.

5. CONCLUDING REMARKS

The method finding IKN-communities on the way to constructing a Gomory-Hu tree is computationally efficient. It runs in $O(mn^2 \log n)$ time[4] by using a maximum flow algorithm developed by Sleator and Tarjan [9], where m is the number of edges and n is the number of vertices. However, there might be many IKN-communities that cannot be found by this method. As a result, partitions obtained by the method are possibly too coarse. Therefore, the algorithm that can efficiently find more IKN-communities should be developed. One candidate of such algorithms is a method that solves Problem 1 using some optimization method.

The efficient method based on edge betweenness devel-

[4] This computational time can be reduced by using a faster algorithm [5], though its time bound is more complicated.

oped by Girvan and Newman [4] may be used to find IKN-communities. They conducted experiments for computer-generated graphs, in which 128 vertices are partitioned into 4 groups with 32 vertices, and each vertex is connected to 16 other vertices by randomly-generated edges, k of them are vertices in other groups and $16 - k$ of them in the same group. According to the reported result, their algorithm extracted 4 groups completely when $k \leq 6$. Note that 4 groups in their graphs are IKN-communities when $k \leq 7$. Their method does not guarantee that any vertex satisfies the conditions of IKN-communities for any vertex subset obtained by the method, and the method can not find overlapping IKN-communities. But their method runs in $O(mn^2)$ time and has possibility that larger IKN-communities can be found, so we think that further study on using their method to find IKN-communities should be done.

6. ACKNOWLEDGEMENTS

We thank two doctor-course students in Tokyo Institute of Technology, Mr. Shigezumi and Mr. Yamamoto, and their supervisor Prof. Watanabe for having interest in our proposed problems and proving hardness results on them. We also thank anonymous reviewers for their helpful comments.

7. REFERENCES

[1] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks*, 30(1-7):107–117, 1998.

[2] G. Flake, S. Lawrence, and C. Giles. Efficient identification of web communities. In *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 150–160, 2000.

[3] G. Flake, R. Tarjan, and K. Tsioutsiouliklis. Graph clustering and mining cut trees. *Internet Mathematics*, 1(3):355–378, 2004.

[4] M. Girvan and M. Newman. Community structure in social and biological networks. *Proc. Natl. Acad. Sci. USA*, 99:7821–7826, 2002.

[5] V. King, S. Rao, and R. Tarjan. A faster deterministic maximum flow algorithm. *Journal of Algorithms*, 17:447–474, 1994.

[6] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[7] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins. Trawling the web for emerging cyber-communities. *Computer Networks*, 31(11-16):1481–1493, 1999.

[8] A. Nakamura, T. Shigezumi, and M. Yamamoto. On NK-community problem. In *Proceedings of the Winter LA Symposium 2005*, pages 12.1–12.8, 2005.

[9] R. Tarjan. *Data Structure and Network Algorithm.* Society for Industrial and Applied Mathematics, 1983.

[10] V. Vazirani. *Approximation Algorithms.* Springer-Verlag, 2001.