JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 29, 1109-1120 (2013)

DOI:10.1688/JISE.2013.29.6.3

On the Min-Max 2-Cluster Editing Problem

LI-HSUAN CHEN[1], MAW-SHANG CHANG[2], CHUN-CHIEH WANG[1] AND BANG YE WU[1,*]

[1]*Department of Computer Science and Information Engineering*
National Chung Cheng University
Chiayi, 621 Taiwan
[2]*Department of Computer Science and Information Engineering*
Hungkung University
Taichung, 433 Taiwan

In this paper, we study the problem Min-Max 2-Cluster Editing which asks for a modification of a given graph into two maximal cliques by inserting or deleting edges such that the maximum number k of the editing edges incident to any vertex is minimized. We show the NP-hardness of the problem and present a polynomial-time algorithm when $k < n/4$, in which n is number of vertices. In addition, we design a 2-approximation algorithm and a branching algorithm for finding an optimal solution. By experiments on random graphs, we show that the exact algorithm is much more efficient than a trivial one.

Keywords: algorithm, clustering editing, graph modification problem, NP-hard, approximation algorithm

1. INTRODUCTION

Graph clustering is an important issue in computer science and finds many applications, especially in bioinformatics and machine learning. A cluster graph is an undirected graph consisting of disjoint maximal cliques. The maximal cliques in a cluster graph are also called clusters. Basically, the goal is to find a way to modify a given graph into a cluster graph. However, there are several versions of related problems, such as Consensus Clustering, Correlation Clustering and Cluster Editing. In the following, we shall only mention some of them.

Consensus Clustering: The problem is also known in the literature as the median partition problem. In [1], it was proposed to an application in bioinformatics. The minimization version is known to be Max-SNP-hard even with just three input clusterings, while the maximization version admits a PTAS [2]. A variant in which the number of clusters is fixed has also been studied [3].

Correlation Clustering: The problem (on complete signed graphs) was formulated and studied in [4], in which the authors presented a PTAS for the maximization version and a constant factor approximation algorithm for the minimization version. Ailon *et al*. [5] showed some approximation results for the minimization version, including the unweighted case and the weighted case with some interesting assumptions on weights. For

Received February 20, 2013; accepted April 19, 2013.
Communicated by Ruay-Shiung Chang and Sheng-Lung Peng.
* Corresponding author, E-mail: bangye@ccu.edu.tw.

$p \geq 2$, p-Correlation Clustering is a variant in which the number of clusters is required to be upper bounded by p. Giotis and Guruswami showed that both the minimization and the maximization versions of p-Correlation Clustering admit PTAS for the unweighted case [6]. The maximization 2-Correlation Clustering problem is also known as Balanced Subgraph which comes from the application in social network analysis [7, 8].

Cluster Editing: The problem looks for the minimum number of edge insertions and deletions to modify the input graph to a cluster graph [9]. Cluster Editing is closely related to graph clustering and arises in bioinformatics [10]. For an integer p, a cluster graph is a p-cluster graph if the number of clusters is at most p, and an important variant is p-Cluster Editing which modify the input graph to a p-cluster graph.

Balancing Sign Graph: A complete signed graph is a simple undirected graph with either a positive edge or a negative edge between each pair of vertices. On the social network analysis, sign graphs are often used to describe the 'like' or 'dislike' relationship amount a group of people [8]. As a representation of a social network, the sign graphs with a cycle of odd number of negative edges are considered unstable or unbalanced. Therefore, there arises the problem: balancing signed graph, which is to change the sign of the minimum number of edges such that there is no cycle with odd number of negative edges. Due to a theorem from Harary [7], it is known that balancing a complete signed graph is identical to solve the 2-Cluster Editing problem.

In this paper we study a natural variant, named Min-Max 2-Cluster Editing, of Cluster Editing. While Cluster Editing minimizes the total number of editing edges, for Min-Max Cluster Editing we hope that the number of inserting and deleting edges incident to any vertex is small. The decision version is called max-k-editable which determines if a graph can be modified into a 2-cluster graph in such a way that the number of editing edges incident to each vertex is at most k.

Shamir *et al*. [9] studied the computational complexities of three edge modification problems. Cluster Editing asks for the minimum total number of edge insertions and deletions, while in Cluster Deletion (respectively, Cluster Completion), only edge deletions (respectively, insertions) are allowed. They showed that Cluster Editing is NP-hard, Cluster Deletion is Max SNP-hard, and Cluster Completion is polynomial-time solvable. For a constant integer $p \geq 2$, p-Cluster Deletion (similarly, p-Cluster Editing) is a variant of Cluster Deletion in which the desired solution must contain exactly p clusters. They also showed that p-Cluster Deletion is NP-hard for any $p > 2$ but polynomial-time solvable for $p = 2$, and p-Cluster Editing is NP-hard for any $p \geq 2$. In the literature, there are several results on the fixed-parameter time complexities for Cluster Editing and Cluster Deletion, for example [11-17], and the most recent result can be referred to [18]. A variant with vertex (rather than edge) deletions was considered in [19], and another variant in which overlapping clusters are allowed was studied in [20].

Although Min-Max 2-Cluster Editing is a natural variant, there seems no study in the literature. In this paper, we show the following results.

- Min-Max 2-Cluster Editing is NP-hard.
- If the input graph is k-editable with $k < n/4$, Min-Max 2-Cluster Editing can be solve in $O(n^2)$ time.

- We design a heuristic algorithm which always reports a 2-approximation solution.
- We also present a branching algorithm for solving the problem exactly. By designing good reduction rules, the algorithm performs much better than the trivial brute force method. We show the performance by experiments.

The rest of the paper is organized as follows. In section 2, we give some notation and definitions used in this paper. We show the NP-hardness in section 3 and the polynomial-time solvable case is shown in section 4. The approximation algorithm is presented in section 5. The branching algorithm and experimental results are in section 6. Finally concluding remarks are given in section 7.

2. PRELIMINARIES

The set-minus is denoted by "\", *i.e.*, for two sets S and S', $S \setminus S' = \{e \mid e \in S \text{ and } e \notin S'\}$. For a graph G, $V(G)$ and $E(G)$ denote the vertex and the edge sets, respectively. In this paper, a graph is always undirected and simple. Two vertices u and v are neighbors of each other if $(u, v) \in E$. For a vertex subset U, the subgraph of G induced by U is denoted by $G[U]$. For an edge subset F, the subgraph induced by F is the graph whose edge set is F and its vertex set contains all the endpoints of edges in F. Since there will be no confusion from the context, we also use $G[F]$ to denote the edge-induced subgraph.

The degree of a vertex v in a graph G, denoted by $d_G(v)$, is the number of its neighbors in G. For a vertex subset U, let $d(v, U)$ denote the number of neighbors of v in U, *i.e.*, $d(v, U) = |\{u \in U \mid (u, v) \in E\}|$. Let $d'(v, U) = |\{u \in U \mid (u, v) \notin E\}|$. For $F \subset V \times V$, we use $d_F(v)$ to denote $d_H(v)$ in which $H = (V, F)$. A clique is a complete subgraph. A k-clique is a clique of k vertices. A clique is maximal if it is not properly contained in another clique. A maximum clique of a graph is a clique of maximum number of vertices. For two sets S and T, let $S \oplus T = (S \setminus T) \cup (T \setminus S)$ denote the symmetric difference. For $G = (V, E)$, a set $F \subset V \times V$ is an editing set if $G' = (V, E \oplus F)$ is a 2-cluster graph.

PROBLEM: Max-k Editable
INSTANCE: A graph $G = (V, E)$ and a positive integer k.
QUESTION: Is there an editing set F such that $d_F(v) \leq k$ for each $v \in V$?

We say that a graph is max-k-editable, or simply k-editable, if such an editing set exists. The optimization version will be called Min-Max 2-Cluster Editing, which finds the minimized k such that the given graph G is k-editable.

A 2-partition $\pi = (V_1, V_2)$ of V is an unordered pair of vertex subsets V_1 and V_2 such that $V_1 \cup V_2 = V$ and $V_1 \cap V_2 = \emptyset$. Immediately, $G' = (V, E \oplus F)$ is a 2-cluster graph if and only if there exists a 2-partition $\pi = (V_1, V_2)$ of V such that both $G'[V_1]$ and $G'[V_2]$ are cliques, as well as $(u, v) \notin E \oplus F$ for any $u \in V_1$ and $v \in V_2$. We say that π is the feasible 2-partition if G is k-editable. At a glance, it seems that we should try to find the best editing set for Max-k Editable. But it is sometimes more convenience to find the feasible 2-partition.

Two vertices u and v are co-clustered in π if they are in the same cluster. For a 2-partition π, define the conflict between two vertices as follows.

$$C_\pi(u,v) = \begin{cases} 1 & \text{if } (u,v) \notin E \text{ and they are co - clustered; or} \\ & (u,v) \in E \text{ and they are not co - clustered} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The relation between conflicts and editing set comes from the definitions.

Lemma 1: Suppose that G is k-editable and F is the editing set. Let $G' = (V, E \oplus F)$ and π be the corresponding feasible 2-partition. Then, $C_\pi(u, v) = 1$ if and only if $(u, v) \in F$.

For a vertex subset U, let $C_\pi(v, U) = \sum_{u \in U} C_\pi(v, u)$, and the conflict number of a vertex v is $C_\pi(v) = \sum_{u \in V} C_\pi(v, u)$. Max-$k$ Editable is equivalent to determining if there exists a 2-partition π such that $C_\pi(v) \leq k$ for each $v \in V$.

3. NP-COMPLETENESS

We shall show the NP-completeness of Max-k Editable by reducing from the 3-SAT problem. An instance of 3-SAT consists of n variables x_i, $1 \leq i \leq n$, and m clauses D_i, $1 \leq i \leq m$, in which each clause contains exactly three literals. We make the following two restrictions on the 3-SAT problem.

1. Assigning all variables false cannot satisfy all the clauses.
2. There is no true assignment making all literals of all clauses true.

It is not hard to see the restrictions do not affect the NP-completeness. For an instance of the 3-SAT problem, we add additional n variables x_i, $n + 1 \leq i \leq 2n$, and construct a graph $G = (V, E)$ as follows, in which V consists of pairwise disjoint subset M, U, and Q. Let $W > \max\{n, m\}$ be a sufficiently large integer and $J(i) = \{j \mid x_j \in D_i \vee \neg x_j \in D_i\}$ be the set of indexes for which variable x_j is involved in clause D_i.

- There is a vertex D_i for each clause. Let $M = \{D_i \mid 1 \leq i \leq m\}$ be a clique.
- Let U be the set of $2n$ disjoint cliques defined as follows. For $1 \leq i \leq 2n$, let X_i be a clique containing $2W$ vertices.
- Q is a clique of $4W$ vertices and there is no edge linking a vertex of Q to any vertex outside Q.
- For each clause D_i, if $x_j \in D_i$, then D_i is connected to all vertices of X_j. If $\neg x_j \in D_i$, then there is no edge between vertex D_i and any vertex in X_j. Otherwise, for $j \notin J(i)$, D_i is connected to W vertices in X_j. Note that D_i is connected to W vertices in X_j for any $n + 1 \leq j \leq 2n$ since neither x_j nor $\neg x_j$ is in D_i.
- $k = (2n + 1)W$.

Definition 1: For a 2-partition $\pi = (V_1, V_2)$, a vertex subset Y is not split by π if Y is entirely in V_1 or V_2. π is *regular* if it meets the following conditions: (1) $M \subset V_1$; (2) $Q \subset V_2$; and (3) X_j is not split by π for any j. For a 2-partition π not splitting any X_j, we define $T[\pi]$ as a truth assignment such that x_i is true if and only if $X_i \subset V_1$. Let $T'[\pi]$ be the reverse true assignment of $T[\pi]$.

Since a 2-partition is an unordered pair of vertex subsets, $(V_1, V_2) = (V_2, V_1)$ and we shall assume $M \subset V_1$ and $Q \subset V_2$ when (V_1, V_2) is regular.

Proposition 1: Suppose that $\pi = (V_1, V_2)$ is a regular 2-partition. The clause D_i is satisfied by $T[\pi]$ if and only if $C_\pi(D_i) \le (2n + 1)W$.

Proof: Since π is regular, $C_\pi(D_i, M) = C_\pi(D_i, Q) = 0$ and therefore $C_\pi(D_i) = C_\pi(D_i, U)$. For any literal $y_j = x_j$ or $\neg x_j$ in D_i, y_j is true if and only if $C_\pi(D_i, X_j) = 0$, and otherwise $C_\pi(D_i, X_j) = 2W$. For any variable x_j such that $j \notin J(i)$, $C_\pi(D_i, X_j) = W$ no matter which cluster X_j belongs to. There are exactly $2n - 3$ such variables since D_i contains three literals.

If D_i is satisfied by $T[\pi]$, at least one of its literals is true, and therefore $C_\pi(D_i, U) \le 4W + (2n - 3)W = (2n + 1)W$. Conversely if $C_\pi(D_i, U) \le (2n + 1)W$, at least one of its literals is true. ❑

Lemma 2: If there is a truth assignment satisfying all the clauses, then G is k-editable.

Proof: We construct a regular 2-partition $\pi = (V_1, V_2)$ as follows. For each $1 \le j \le n$, $X_j \subset V_1$ if and only if x_j is true. The other cliques X_j are distributed such that V_1 contains exactly $n + 1$ cliques of U. Precisely speaking, let p be the number of variables which are assigned true, by the restriction 1 of the 3-SAT problem, $p > 0$. We set $\{X_i \mid n + 1 \le i \le 2n + 1 - p\} \subset V_1$ and all X_i are in V_2 for $i > 2n + 1 - p$. Note that V_1 contains exactly $n + 1$ cliques of U besides the clique M. Also V_2 contains $n - 1$ cliques of U and the clique Q. Therefore $|V_2| = (2n + 2)W$ and $|V_1| = (2n + 2)W + m$.

For any vertex $v \in V_2$, $d'(v, V_2) \le 2nW$ which is the number of vertices in V_2 but outside the clique containing v. Then $C_\pi(v) = d'(v, V_2) + d(v, M) \le 2nW + m \le (2n + 1)W$. Similarly, for $v \in U \cap V_1$, $C_\pi(v) = d'(v, U \cap V_1) + d'(v, M) \le 2nW + m \le (2n + 1)W$. Next consider any vertex $D_i \in M$. Since π is regular, by Proposition 1, $C_\pi(D_i) \le (2n + 1)W$. ❑

Next we show the other direction.

Lemma 3: If G is k-editable, then there is a truth assignment satisfying all the clauses.

Suppose that G is k-editable. By definition there exists a 2-partition $\pi = (V_1, V_2)$ such that $C_\pi(v) \le k$ for each $v \in V$. The key point is to show that π must be regular, and then Lemma 3 is immediately followed from Proposition 1.

Claim: For $1 \le i \le 2n$, clique X_i is not split by π.

Proof: Let $U^+ = U \cup Q$. Suppose by contradiction that X_i is split. For $y_1 \in X_i \cap V_1$,

$$C_\pi(y_1, U^+) = |X_i \cap V_2| + |V_1 \cap (U^+ \setminus X_i)|$$

and for $y_2 \in X_i \cap V_2$,

$$C_\pi(y_2, U^+) = |X_i \cap V_1| + |V_2 \cap (U^+ \setminus X_i)|.$$

Since the sum of the right-hand sides of the above two equations is $|U^+| = (4n + 4)W$, we have

$$\max_{y \in X_i} C_\pi(y) \geq (1/2)(C_\pi(y_1, U^+) + C_\pi(y_2, U^+)) = (2n + 2)W > k,$$

which is a contradiction and therefore X_i is not split. ❑

The next claim can be shown similarly.

Claim: Clique Q is not split by π.

By the above claim, we may assume $Q \in V_2$. Next we claim the vertices of U^+ are evenly distributed.

Claim: $|V_1 \cap U^+| = |V_2 \cap U^+| = (2n + 2)W$.

Proof: By the above claims, cliques in U^+ are not split. If U^+ is not evenly divided, one of them contains at least $(2n + 4)W$ vertices. For any vertex v in this set but not in Q, since at most $2W - 1$ of them are its neighbors, $C_\pi(v) \geq (2n + 2)W > k$, which is a contradiction. ❑

Claim: $M \subset V_1$.

Proof: Let $M_i = M \cap V_i$ for $i = 1, 2$. For $D_i \in M_2$,

$$C_\pi(D_i) \geq C_\pi(D_i, Q) + C_\pi(D_i, U) + |M_1| = 4W + (2n - 3)W + |M_1|, \tag{2}$$

and the equality holds only when $C_\pi(D_i, X_j) = 0$ for any $j \in J(i)$. That is, all the three literals are set true by $T'[\pi]$.

If $M_1 \neq \emptyset$, we have $C_\pi(D_i) > k$. Therefore M_1 or M_2 is empty. If $M_1 = \emptyset$ and the equality of Eq. (2) holds for each D_i, then $T'[\pi]$ satisfies all literals of all clauses. It is a contradiction to our restriction 2 of the 3-SAT problem, and therefore we obtain $M_2 = \emptyset$. ❑

Lemma 3 is shown by the above claims and the next theorem follows from Lemmas 2 and 3.

Theorem 1 Min-Max 2-Cluster Editing is NP-complete.

4. POLYNOMIAL-TIME SOLVABLE CASE

In this section we show how to solve the problem when $k < n/4$. Here and after we shall let $G = (V, E)$ be the input graph and $n = |V|$.

We define an operation "moving vertices" on 2-partitions as follows. Let $\pi = (V_1, V_2)$ be a 2-partition of V. Another 2-partition π' is obtained by moving a vertex u in π if $\pi' = (V_1 \oplus \{u\}, V_2 \oplus \{u\})$. Recall that $\oplus$ is the symmetric difference, and therefore

$$V_i \oplus \{u\} = \begin{cases} V_i \setminus \{u\} \text{ if } u \in V_i \\ V_i \cup \{u\} \text{ if } u \notin V_i \end{cases}. \tag{3}$$

In other words, u is moved from its original part to the other. The operation of moving a

set of vertices is defined similarly, and we denote by $\pi' = \Delta_U(\pi)$ that π' is obtained by moving a set U of vertices in π. The next relation follows from definitions directly.

$$C_{\pi'}(u,v) = \begin{cases} 1 - C_\pi(u,v) & \text{if exactly one of } u \text{ and } v \text{ is in } U \\ C_\pi(u,v) & \text{otherwise} \end{cases} \quad (4)$$

Consequently, if $\pi' = \Delta_{\{u\}}(\pi)$, then $C_{\pi'}(u) = n - 1 - C_\pi(u)$.

For any vertex v, the v-partition is a 2-partition $\pi = (V_1, V - V_1)$, in which $V_1 = \{v\} \cup \{u \mid (u, v) \in E\}$.

Lemma 4: If G is k-editable, for any $v \in V$ there exists $U \in V$ with $|U| \le k$ such that $\pi^* = \Delta_U(\pi)$ is feasible, in which π is the v-partition.

Proof: By the assumption, there exists a vertex 2-partition π^* such that $C_{\pi^*}(u) \le k$ for each $u \in V$. Therefore for any v, the size of the set U defined by $U = \{u \mid C_{\pi^*}(v, u) = 1\}$ is less than or equal to k. By definition, moving U in π^* results in the v-partition π, *i.e.*, $\pi = \Delta_U(\pi^*)$. ❑

We shall call the set U in Lemma 4 a feasible moving set.

Lemma 5: Suppose that π is the v-partition for arbitrary v and G is k-editable with $k < n/4$. For any vertex u and any feasible moving set U, if $C_\pi(u) \ge n/2$, then $u \in U$.

Proof: By Lemma 4, there exists a feasible partition $\pi^* = \Delta_U(\pi)$ with $|U| \le k$. Since $k < n/4$, if $C_\pi(u) \ge n/2$, we have $C_\pi(u) > 2k$. If $u \notin U$, moving U will reduce the conflict number of u by at most k and $C_{\pi^*}(u) > (n/2) - (n/4) > k$, which is a contradiction. Therefore, $u \in U$. ❑

Lemma 6: Suppose that π is the v-partition for arbitrary v and G is k-editable with $k < n/4$. For any vertex u and any feasible moving set U, if $C_\pi(u) < n/2$, $u \notin U$.

Proof: By Lemma 4, there exists a feasible partition $\pi^* = \Delta_U(\pi)$ with $|U| \le k$. By the assumption $k < n/4$, if $u \in U$ and $\pi' = \Delta_{\{u\}}(\pi)$, $C_{\pi'}(u) = n - 1 - C_\pi(u) \ge n/2$. Moving the other $|U| - 1$ vertices will still have $C_{\pi^*}(u) \ge (n/2) - |U| + 1 > k$, which is a contradiction, therefore, $u \notin U$. ❑

Theorem 2 If G is k-editable and $k < n/4$, the problem Min-Max 2-Cluster Editing can be solved in $O(n^2)$ time.

Proof: Starting from the v-partition π for an arbitrary vertex v, we compute $U = \{u \mid C_\pi(u) \ge n/2\}$. Then we move U to obtain $\pi' = \Delta_U(\pi)$, and compute $k' = \max_{u \in V} C_{\pi'}(u)$. It is trivial that all the steps can be done in $O(n^2)$ time.

If G is k-editable with $k < n/4$, by the above two lemmas, each vertex either must be moved or cannot be moved. Therefore π' is the only possible feasible 2-partition. If $k' < n/4$, we obtain the minimized k; and otherwise there is no feasible solution with $k < n/4$. ❑

5. AN APPROXIMATION ALGORITHM

Consider the greedy algorithm which beginning from an arbitrary partition π, repeatedly moves the vertex u with the maximum conflict number until all the vertices have conflict number at most $n/2$.

Lemma 7: The greedy algorithm stops within n^2 iterations and takes $O(n^3)$ time.

Proof: Consider the total conflict number over all vertices. The result follows from two facts. First the total conflict number strictly decreases after each iteration. Second, the initial total conflict number is at most $n(n-1)$. The time complexity follows from each iteration takes $O(n)$ time to update the conflict numbers and find the vertex to move. ❑

Corollary 1 Any undirected simple graph is $n/2$-editable.

By Theorem 2, we can check in $O(n^2)$ time if a graph is k-editable with $k < n/4$, as well as find the optimal k if the answer is yes. If the answer is no, we can find a feasible solution for $k \leq n/2$ by the greedy algorithm. Since in this case the optimal k is at least $n/4$, we find a 2-approximation. Therefore we have the next result.

Theorem 3 A 2-approximation solution of Min-Max 2-Cluster Editing can be found in $O(n^3)$ time.

6. A BRANCHING ALGORITHM AND EXPERIMENTS

In this section we present an exact algorithm for Min-Max 2-Cluster Editing. In the following we focus on the decision version. That is, the algorithm determines if the input graph is k-editable. To find the minimized k, we first check if there is a solution with $k < n/4$. If not, we start from $k = n/4$ and iteratively increase k one by one to find the minimum k such that G is k-editable. Since Max-k Editable looks for a feasible 2-partition, a straightforward algorithm is to check the existence of a feasible 2-partition. But our branch-and-bound algorithm focuses on the feasible moving set instead of the 2-partition. In the branching algorithm, all vertices are divided into three subsets: M, M', and U, which contain the vertices must in the moving set, must not in the moving set, and undetermined, respectively. Starting from a v-partition for an arbitrarily chosen vertex v and an initial moving set $M = \emptyset$, the algorithm chooses an undetermined vertex u and recursively searches the two branches: adding u into M or into M'.

Similar to many branch-and-bound algorithms, the theoretical efficiency is hard to analyze. The theoretical worst-case time complexity of our branch algorithm is $O(n^2B(n, k))$, in which $B(n, k)$ is the binomial coefficient of choosing k in n objects. The algorithm takes the advantage that the size of a feasible moving set is at most k while there are 2^{n-1} possible 2-partitions. However, since $B(n, k)$ goes up to 2^n when k is large, the time complexity is exponential. But we may improve the practical performance by designing some reduction rules to avoid exhaustive search. The efficiency will be shown by experimental results.

In section 4, we have derived two rules. When $k < n/4$, every vertex can be determined if it needs to move to obtain a feasible partition. However, when $k \geq n/4$, not all the vertices can be determined. But we can generalize it to design reduction rules. For a partial solution (M, U) and an initial v-partition π, we can have the following two rules. Let $\pi' = \Delta_M(\pi)$.

Lemma 8: For any $u \in U$, if $C_{\pi'}(u) > 2k - |M|$, then any feasible moving set $S \supset M$ must contain u.

Proof: By definition $|S| \leq k$. Since $C_{\pi'}(u) > 2k - |M|$, if $u \notin S$, moving $S \setminus M$ from π' will reduce the conflict number of u by at most $k - |M|$ and result in a conflict number at least $C_{\pi'}(u) - (k - |M|) > k$, which is a contradiction. ❑

Lemma 9: For any $u \in U$, if $C_{\pi'}(u) < n - 2k + |M|$, then any feasible moving set $S \supset M$ cannot contain u.

Proof: Since $C_{\pi'}(u) < n - 2k + |M|$, if $u \in S$, then $C_{\pi''}(u) > n - 1 - (n - 2k + |M|) = 2k - 1 - |M|$, in which $\pi'' = \Delta_{\{u\}}(\pi')$. Then moving $S \setminus (M \cup \{u\})$ in π'' will reduce the conflict number of u by at most $k - |M| - 1$ and result in a conflict number larger than $2k - 1 - |M| - (k - |M| - 1) = k$, which is a contradiction. ❑

Table 1. Average running times on 20 Type-1 random graphs.

p	$n = 30$	$n = 35$	$n = 40$	$n = 50$	$n = 300$
0.2	1.59(3.46)	13.1(48.6)	> 100	> 100	> 100
0.5	1.07(1.94)	10.7(21.2)	> 100	> 100	> 100
0.8	0.005(0.055)	0.0008(0.195)	0.0013(1.42)	< 1(> 100)	< 1(> 100)

Table 2. Average running times on 20 Type-2 random graphs with $n = 40$ and $m = 40+kl$.

l	$k = 10$	$k = 12$	$k = 14$	$k = 16$	$k = 18$
2	0.0005(0.001)	0.0009(0.002)	0.0010(0.002)	0.0013(0.003)	0.3922(3.796)
5	0.0005(0.001)	0.0007(0.002)	0.0010(0.002)	0.0016(0.004)	16.0351(54.998)
10	0.0007(0.001)	0.0007(0.002)	0.0009(0.002)	0.0243(30.498)	102.0826(422.9)

In the remaining paragraphs of this section, we show the experimental results. Two types of random graphs are used in the experiments. In a random graph of the first type, an edge exists with an independent identical probability p. For the second type, a graph is generated from a 2-cluster graph and an editing set F such that $|F| = m$ and there are l vertices with degree k in F. Two kinds of experiments were performed: efficiency of the exact algorithm, and practical performance of the approximation algorithm.

In Table 1, we compare the execution times (in seconds) of the exact algorithm with and without reduction rules for Type-1 data. The numbers in the parentheses are running times without reduction rules. It is shown that using the reduction rules greatly reduces the execution time, especially on the graphs with higher density. In the density cases $p = 0.8$, the optimal solutions on graphs with 300 vertices can be found within one second. As shown in Table 2 we also performed tests on Type-2 data. Again, the numbers in the

parentheses are running times without reduction rules. The experimental results show that when $l \leq 10$, the algorithm with reduction runs very fast for $n/4 < k \approx n/3$ but the running time gets larger when k is close to $n/2$. It should be noticed that the running times are smaller in the cases with smaller l.

Table 3. Average approximation ratio times on 20 Type-1 random graphs.

p	$n = 30$	$n = 35$
0.2	1.0541(1.1538)	1.0562(1.0625)
0.5	1.0813(1.1538)	1.0531(1.0625)
0.8	1(1)	1(1)

Table 4. Average (worst) approximation ratio on 20 Type-2 random graphs with $n = 40$ and $m = 40 + kl$.

p	$k = 15$	$k = 16$	$k = 17$	$k = 18$
0.2	1	1	1	1.0056(1.1111)
0.5	1	1	1.0147(1.1765)	1.0611(1.1111)
0.8	1	1.0125(1.2500)	1.1235(1.1765)	1.1056(1.1111)

In the second experiment, as Tables 3 and 4, the approximation ratios are tested on both types of graphs with vertices less than 100 and $k \leq n/3$. In all the tests we performed the approximation ratios are all within 1.3.

7. CONCLUDING REMARKS

In addition to good approximation algorithms and exact algorithms for Min-Max 2-Cluster Editing, interesting future works also include how to generalize the results of this paper to the version that the number of cluster is more than two or the number of clusters is not specified.

ACKNOWLEDGMENT

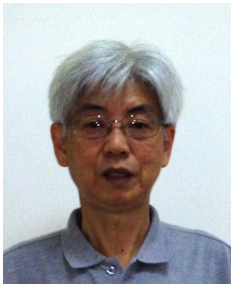
L.-H. Chen and M.-S. Chang were supported in part by NSC 99-2221-E-241-015-MY3, and B.Y. Wu and C.-C. Wang were supported in part by NSC 100-2221-E-194-036-MY3 and NSC 101-2221-E-194-025-MY3 from the National Science Council, Taiwan.

REFERENCES

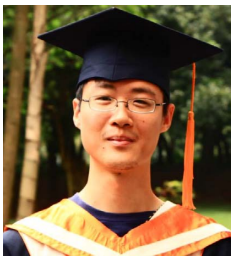
1. V. Filkov and S. Skiena, "Integrating microarray data by consensus clustering," *International Journal on Artificial Intelligence Tools*, Vol. 13, 2004, pp. 863-880.
2. P. Bonizzoni, G. D. Vedova, R. Dondi, and T. Jiang, "On the approximation of correlation clustering and consensus clustering," *Journal of Computer and System Sciences*, Vol. 74, 2008, pp. 671-696.
3. P. Bonizzoni and G. D. Vedova, R. Dondi, ''A ptas for the minimum consensus

clustering problem with a fixed number of clusters,'' in *Proceedings of the 11th Italian Conference on Theoretical Computer Science*, 2009, pp. 55-58.
4. N. Bansal, A. Blum, and S. Chawla, "Correlation clustering," *Machine Learning*, *Special Issue on Clustering*, Vol. 56, 2004, pp. 89-113.
5. N. Ailon, M. Charikar, and A. Newman, "Aggregating inconsistent information: Ranking and clustering," *Journal of ACM*, Vol. 55, 2008, pp. 1-27.
6. I. Giotis and V. Guruswami, "Correlation clustering with a fixed number of clusters," *Theory Computing*, Vol. 2, 2006, pp. 249-266.
7. F. Harary, "On the notion of balance of a signed graph," *Michigan Mathematical Journal*, Vol. 2, 1953, pp. 143-146.
8. S. Wasserman and K. Faust, *Social Network Analysis*, Cambridge University Press, Cambridge, 1994.
9. R. Shamir, R. Sharan, and D. Tsur, "Cluster graph modification problems," *Discrete Applied Mathematics*, Vol. 144, 2004, pp. 173-182.
10. T. Wittkop, J. Baumbach, F. Lobo, and S. Rahmann, "Large scale clustering of protein sequences with FORCE-a layout based heuristic for weighted cluster editing," *BMC Bioinformatics*, Vol. 8, 2007, pp. 396.
11. S. Böcker, S. Briesemeister, Q. B. A. Bui, and A. Truss, "Going weighted: Parameterized algorithms for cluster editing," *Theoretical Computer Science*, Vol. 410, 2009, pp. 5467-5480.
12. J. Chen and J. Meng, "A 2*k* kernel for the cluster editing problem," in *Proceedings of International Conference on Computing and Combinatorics Conference*, 2010, Vol. 6196, 2010, pp. 459-468.
13. P. Damaschke, "Bounded-degree techniques accelerate some parameterized graph algorithms," in *Proceedings of Workshop on Parameterized and Exact Computation*, LNCS, Vol. 5917, 2009, pp. 98-109.
14. P. Damaschke, "Fixed-parameter enumerability of cluster editing and related problems," *Theory of Computing Systems*, Vol. 46, 2010, pp. 261-283.
15. J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier, "Graph-modeled data clustering: Fixed-parameter algorithms for clique generation," *Theory of Computing Systems*, Vol. 38, 2005, pp. 373-392.
16. J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier, "Automated generation of search tree algorithms for hard graph modification problems," *Algorithmica*, Vol. 39, 2004, pp. 321-347.
17. J. Guo, "A more effective linear kernelization for cluster editing," *Theoretical Computer Science*, Vol. 410, 2009, pp. 718-726.
18. S. Böcker and P. Damaschke, "Even faster parameterized cluster deletion and cluster editing," *Information Processing Letters*, Vol. 111, 2011, pp. 717-721.
19. F. Hüffner, C. Komusiewicz, H. Moser, and R. Niedermeier, "Fixed-parameter algorithms for cluster vertex deletion," *Theory of Computing Systems*, Vol. 47, 2010, pp. 196-217.
20. M. R. Fellows, J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann, "Graph-based data clustering with overlaps," *Discrete Optimization*, Vol. 8, 2011, pp. 2-17.

Li-Hsuan Chen (陳立軒) is a Ph.D. student of Computer Science and Information Engineering in National Chung Cheng University. He received his Master degree in physics from National Tsing Hua University in 2009. His research focuses on computer algorithms, graph theory, and social network analysis.



Maw-Shang Chang (張貿翔) received his B.S. degree in electric engineering from National Cheng Kung University in 1975, M.S. degree in information management from National Tsing Hua University in 1983, and Ph.D. degree in computer science from National Tsing Hua University in 1990. In 1990, he joined the Department of Computer Science and Information Engineering, National Chung Cheng University, where he became Professor in 1996. From 1996 to 2001, he was the director of library at National Chung Cheng University. He is currently the Chairman of Computer Science and Information Engineering at Hungkuang University. His current research interests include computer algorithms, graph theory, computational molecular biology, and library information systems.



Chun-Chieh Wang (王俊杰) received his MS degrees in Computer Science and Information Engineering at National Chung Cheng University in 2013.

Bang Ye Wu (吳邦一) received the B.S. degree in 1986 from the Department of Electrical Engineering at Chung Cheng Institute of Technology. He received the M.S. and the Ph.D. degrees in Computer Science from National Tsing Hua University in 1991 and 1999, respectively. In 2008, he joined the faculty of National Chung Cheng University and is presently a Professor at Computer Science and Information Engineering. His research

interests include algorithms, social network analysis, and graph theory.