Theoretical Computer Science 38 (1985) 293-306
North-Holland

CLUSTERING TO MINIMIZE THE MAXIMUM INTERCLUSTER DISTANCE*

Teofilo F. GONZALEZ**

Programs in Computer Science, University of Texas, Dallas, Richardson, TX 75080, U.S.A.

Communicated by J.D. Ullman
Received October 1983
Revised March 1984

Abstract. The problem of clustering a set of points so as to minimize the maximum intercluster distance is studied. An O(kn) approximation algorithm, where n is the number of points and k is the number of clusters, that guarantees solutions with an objective function value within two times the optimal solution value is presented. This approximation algorithm succeeds as long as the set of points satisfies the triangular inequality. We also show that our approximation algorithm is best possible, with respect to the approximation bound, if P$\neq$NP.

Key words. Algorithms, clustering, NP-completeness, approximation algorithms, minimizing the maximum intercluster distance.

1. Introduction

The problem of clustering a set of objects arises in many disciplines. Because of the wide range of applications, there are many variations of this problem. The main difference between these problems is in the objective function. Research in different fields of study during the past thirty years has produced a long list of clustering algorithms. However, very little is known about the merits of these algorithms. Even simple questions regarding to the computation complexity of most clustering problems have not yet been answered. In this paper we study the computational complexity of generating optimal and near optimal solutions to the problem of clustering a set of points so as to minimize the maximum intercluster distance.

Let $G=(V, E, W)$ be a weighted undirected graph with vertex set V, edge set E and a dissimilarity or weight function $W: E\rightarrow \mathbb{R}_0^+$ (the set of nonnegative reals). A partition of the set of vertices into k sets, $(B_1, B_2, \ldots, B_k)$, is called a *k-split*. The sets B_i in a k-split are called *clusters*. An *objective function*, $f: B_1, B_2, \ldots, B_k \rightarrow \mathbb{R}_0^+$, is defined for each k-split. In this paper we study clustering problems whose objective

* A preliminary version of this paper was presented at the 10th IFIP Conference on System Modeling and Optimization [8].

** Present affiliation: Department of Computer Science, University of California, Santa Barbara, CA 93106, U.S.A.

function is $\max\{M_1, M_2, \dots, M_k\}$, where M_i is the maximum weight of an edge whose endpoints are vertices in cluster B_i (we call this objective function MM).

Clustering problems usually have one of the following forms:

(P1) Given a graph G, an objective function f and an integer k, find a k-split with least objective function value, i.e. find a k-split $(B_1^*, B_2^*, \dots, B_k^*)$ such that $f(B_1^*, B_2^*, \dots, B_k^*) = \min\{f(B_1, B_2, \dots, B_k) | (B_1, B_2, \dots, B_k)$ is a k-split for $G\}$.

(P2) Given a graph G, an objective function f and a real number w, find for the least value of k a k-split with objective function value less than or equal to w, i.e. find a k-split $(B_1^*, B_2^*, \dots, B_k^*)$ such that $f(B_1^*, B_2^*, \dots, B_k^*) \leq w$ and $f(B_1, B_2, \dots, B_{k'}) > w$ for all k'-splits with $k' < k$.

(P3) Given a graph G, an objective function f, an integer k' and a real number w. Is there a k-split $(B_1, B_2, \dots, B_k)$ with objective function value $\leq w$ for some $k \leq k'$?

It can be easily shown that the decision problem (P3) is computationally not harder than (P1) and (P2), i.e. any algorithm that solves (P1) or (P2) can be used to solve (P3). This relation implies that if problem (P3) is NP-complete then both (P1) and (P2) are NP-hard. In what follows when we refer to optimization clustering problems, it is implied that we refer to problems of the form (P1). Whenever we wish to consider problems of the form (P2), we shall state it explicitly.

An *m-dimensional clustering problem* is one in which the vertices of G are points in the m-dimensional euclidean space, the set of edges is complete and the weight of each edge is given by the distance between the points it joins, i.e. $W(x_i, x_j) = \|x_i - x_j\|$, where $\|x_k\| = (\sum_l ((x_k)_l)^2)^{1/2}$. We assume that no two points have the same coordinates in all the dimensions.

We shall refer to our clustering problem as an α-βMM problem, where $\alpha \in \{2, 3, \dots, k\}$ is the number of clusters and β means that it is either a β-dimensional clustering problem ($\beta \in \{1, 2, \dots, m\}$), a problem defined over an arbitrarily weighted graph ($\beta = g$) or a problem defined over a graph whose set of weights satisfies the triangular inequality ($\beta = t$). As mention above, MM refers to the objective function 'max max'. In our notation we use k-2MM to refer to a 2-dimensional euclidean problem in which the number of clusters, k, is an input to the problem. Note that in the k-mMM problem both k and m are inputs to any algorithm that solves the problem.

A reader not familiar with NP-complete problems and approximate solutions is referred to [11, 10, 15]. Our notation is that of [11].

Using an approach similar to the one in [17], one can easily show that the k-gMM problem and its corresponding approximation problem are NP-hard for $k > 2$. The 2-gMM problem can be solved efficiently by reducing it to the problem of testing whether a graph is bipartite or not [4]. Brucker [4] also shows that the k-1MM problem can be solved in polynomial time. For optimization problems of the form (P2), one can easily show that the 2-approximation problem[1] is NP-hard. The proof

[1] By *2-approximation problem* we mean the problem of generating solutions with objective function value within two times the optimal solution value.

follows the same approach as the one in [17], but uses the fact that the 2-approximation problem for graph coloration is NP-hard [9]. Algorithms for other clustering problems appear in [1, 13, 18, 2, 3, 4, 10, 7, 19, 6, 5, 16, 20, 14].

In Section 2 we present an $O(nk)$ algorithm for the k-tMM problem that guarantees a solution with objective function value within two times the optimal solution value. The k-2MM problem is shown to be NP-hard in Section 3, and in Section 4 it is shown that our approximation algorithm is best possible with respect to the approximation bound if $P \neq NP$.

2. Approximation algorithm for the k-tMM problem

In this section we present an approximation algorithm for the k-tMM problem. Our algorithm has worst case time complexity $O(nk)$ and it generates solutions with an objective function value within two times the true optimal solution value.

Let S be the set of points to be clustered and T be a subset of S. Assume that $|S| > k$ as otherwise the problem can be trivially solved. Set T is said to form a *$(k+1)$-clique of weight h* if the cardinality of set T is $k+1$ and every pair of distinct elements in T are at least h units apart. Let $\mathrm{OPT}(S)$ be the value of an optimal solution for the instance S of the k-tMM problem.

Lemma 2.1. *If there is a $(k+1)$-clique of weight h for S, then* $\mathrm{OPT}(S) \geq h$.

The proof of this lemma is obvious.

Our algorithm consists of an initialization phase and $k-1$ 'expanding' phases. In the initialization phase all the elements are assigned to set B_1, the first cluster. One of these elements is labeled the head of the cluster, ($head_1$). The selection of this element is arbitrary. During the jth expanding phase, some elements in clusters $B_1, \ldots, B_j$ are moved to cluster B_{j+1}. Also, one of these elements will be labeled the head of the new cluster ($head_{j+1}$). The construction of the new cluster is accomplished by first finding a node, v_i, in one of the first j clusters ($B_1, \ldots, B_j$) whose distance to the head of the cluster it belongs is maximal. Such a node will be moved to cluster B_{j+1} and called the head of the cluster. A node in any of the previously defined clusters will be moved to cluster B_{j+1} if its distance to v_i is not larger than the distance to the head of cluster it belongs to.

It is simple to show that our procedure constructs a k-split. Let $v_i \in B_j$, $1 \leq j \leq k$, be a node whose distance to $head_j$ is maximal. Let h be that distance. Since the set of weights satisfies the triangular inequality, we have that our solution has an objective function value $\leq 2 * h$. It is simple to show that at all time during the execution of the algorithm the distance from this point to the head of the cluster to which it belonged to was at least h. Since this node never became a head of a new cluster, we then have that every time a new cluster was defined the distance from its head to the head of any previously defined cluster was at least h, i.e.

$W(head_p, head_q) \geq h$ for $p \neq q$. Let $T = \{head_1, \ldots, head_k, v_i\}$. Clearly, T is a $(k+1)$-clique of weight h for S and by Lemma 2.1 we have that $\mathrm{OPT}(S) \geq h$. Hence, the solution generated by our algorithm has an objective function value $\leq 2 * \mathrm{OPT}(S)$.

The procedure just described is formally defined below.

```
Algorithm APPROX
  let $\{v_1, \ldots, v_n\}$ be the set of elements to be clustered;
  let $head_1$ represent $v_1$; let $B_1 \leftarrow \{v_1, \ldots, v_n\}$;
  for $l \leftarrow 1$ to $k-1$ do
    $h \leftarrow \max\{W(head_j, v_i) \mid v_i \in B_j$ and $1 \leq j \leq l\}$;
    let $v_i$ be one of the nodes whose distance to the head of the cluster it belongs
      to is $h$;
    move $v_i$ to $B_{l+1}$;
    let $head_{l+1}$ represent $v_i$;
    for each $v_t \in (B_1 \cup \cdots \cup B_l)$ do
      let $j$ be such that $v_t \in B_j$;
      if $W(v_t, head_j) \geq W(v_t, v_i)$ then move $v_t$ from $B_j$ to $B_l$;
      endif
    endfor
  endfor
  return $(B_1, \ldots, B_k)$;
end of algorithm;
```

In what follows we analyze the performance of our algorithm.

Theorem 2.2. *Algorithm* APPROX *generates a solution with an objective function value* $\leq 2 * \mathrm{OPT}(S)$.

The proof of this theorem follows from the above discussion.

Theorem 2.3. *The time complexity of Algorithm* APPROX *is* $O(kn)$.

The proof of this theorem is straightforward.

The bound of two given by Theorem 2.2 cannot be decreased because there are problem instances for which Algorithm APPROX generates solutions with an objective function value equal to $2 * \mathrm{OPT}(S) - \varepsilon$, for any $\varepsilon > 0$. One of these instances is given by the following example.

Example 2.4. k-1MM (all points lie on a straight line).

Case ($n = 4$ and $k = 2$):

$x_0 = 0$; $x_1 = \frac{1}{3}$; $x_2 = \frac{2}{3} - \varepsilon$ and $x_3 = 1$.

Step 1: $head_1$ is x_1 and $B_1 = \{x_0, x_1, x_2, x_3\}$.

Step 2: $head_1$ is x_1; $B_1 = \{x_0, x_1, x_2\}$; $head_2$ is x_3 and B_2 is $\{x_3\}$.

Clearly, the objective function value of the solution generated is $\frac{2}{3}-\varepsilon$. An optimal k-split is given by $\{\{x_0, x_1\}, \{x_2, x_3\}\}$ and its objective function value is $\frac{1}{3}+\varepsilon$.

General Case ($n = k+2$ and $k > 2$):

$x_0 = 0$; $x_1 = \frac{1}{3}$; $x_2 = \frac{2}{3}-\varepsilon$ and $x_i = i-2$ for $3 \leq i \leq n-1$.

Phase 1: *head*$_1$ is x_1 and $B_1 = \{x_0, \ldots, x_{n-1}\}$,

⋮

Phase k: *head*$_1$ is x_1; $B_1 = \{x_0, x_1, x_2\}$; each of $B_2, \ldots, B_k$ contains exactly one of $x_3, \ldots, x_{n-1}$; and the head of each of these clusters is the only element in it.

The value of the objective function for an optimal solution and the one for the solution generated by our algorithm are identical to the ones obtained in the previous case.

It Section 4 it is shown that the factor of two in the approximation bound is best possible if $P \neq NP$. Before proving such a result we need to show that some versions of our clustering problem are NP-hard.

3. The complexity of the k–2MM decision problem

In this section it is shown that the k-2MM decision problem is NP-complete. This result is obtained by reducing a restricted version of the exact cover by three sets problem to it.

The exact cover by three sets (XC3) problem was shown to be NP-complete in [10] and is defined as follows.

Exact Cover by Three Sets (XC3). Given a finite set of elements $X = \{x_1, x_2, \ldots, x_{3q}\}$ and a collection of 3-element subsets of X, $C = \{(x_{i_l}, x_{j_l}, x_{k_l}) \mid 1 \leq l \leq m\}$, in which no element in X appears in more than three subsets. The problem consists of determining whether C has an exact cover for X, i.e. a subcollection C' of C such that every element in X occurs in exactly one member of C'.

The restricted version of this problem, to be used in our reduction, is denoted RXC3. This problem is exactly like the XC3 problem, except that each element in X appears in exactly three subsets of C. Problem RXC3 is shown to be NP-complete in Appendix A.

In order to simplify the presentation of our result, we begin by showing that the k-gMM decision problem is NP-complete (Lemma 3.1). The construction used in this lemma is then modified to show that the k-gMM decision problem is NP-complete even when the input graph, after deleting all edges with weight different than one, is planar and no node is of degree greater than six (Lemma 3.2). We then show how this result can be used to prove that the k-2MM decision problem is NP-complete (Theorem 3.3). The reduction RXC3 $\propto$ k-gMM is identical to the one in [10], which was used to show that partition of a graph into triangles is NP-complete.

Lemma 3.1. *The decision problem k-gMM is NP-complete.*

Proof. It is simple to show that the decision problem k-gMM can be solved in nondeterministic polynomial time. We now show that RXC3 $\propto$ k-gMM.

Given an instance, (X, C), of the restricted exact cover by 3-sets problem, we construct an instance of the k-gMM decision problem which we denote KG. The instance $\mathrm{KG} = (G = (V, E, W), k, d)$ is defined as follows:

Vertex set: There is a vertext (v_i) for each element of the set X and nine vertices are introduced for each 3-element subset of X in C.

Edge set: The set of edges is complete, i.e. for every pair of vertices $i \neq j$ edge $\{i, j\}$ is in E.

Weights: For each 3-element subset of X in C, eighteen edges will get a wieght of one. The edges introduced for $(x_{i_l}, x_{j_l}, x_{k_l}) \in C$ are shown in Fig. 1. All other edges are given the weight of two.

k and d: The value for k is $3m + q$ and d is one.

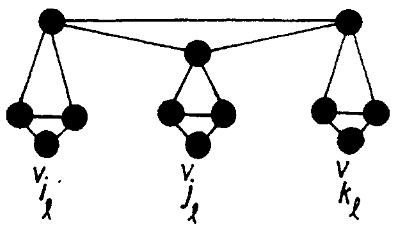

Fig. 1. Component representing a triplet in C.

In order to complete the proof of the lemma it is only required to show that KG has a k-split with objective function value $\leq d$ iff (X, C) has an exact cover, since the construction of KG can be carried out in polynomial time.

Claim. KG *has a k-split with objective function value* $\leq d$ *iff* (X, C) *has an exact cover.*

Proof. First it is shown that if (X, C) has an exact cover, then KG has a k-split $(B_1, B_2, \ldots, B_k)$ with objective function value $\leq d = 1$. Let C' be any exact cover for (X, C). Each subgraph representing a triplet in C' is grouped into the four clusters shown in Fig. 2(a). The subgraphs representing triples in $C - C'$ are grouped, as shown in Fig. 2(b), into three clusters. It is simple to show that such clustering forms a k-split with objective function value equal to d for KG.

In order to complete the proof of the claim it is only required to show that if KG has a k-split with objective function value $\leq d = 1$, then (X, C) has an exact cover. Let $B_1, B_2, \ldots, B_k$ be a k-split with objective function value $\leq d$ for KG. Since no four nodes are completely connected by edges with a weight of one and since the number of nodes in KG is $3 * k$, we have that each B_i must have exactly three nodes. This fact together with the construction rule (Fig. 1) can be used to show that each component (subgraph introduced for a triplet) is clustered as in Fig. 2(a) or (b).

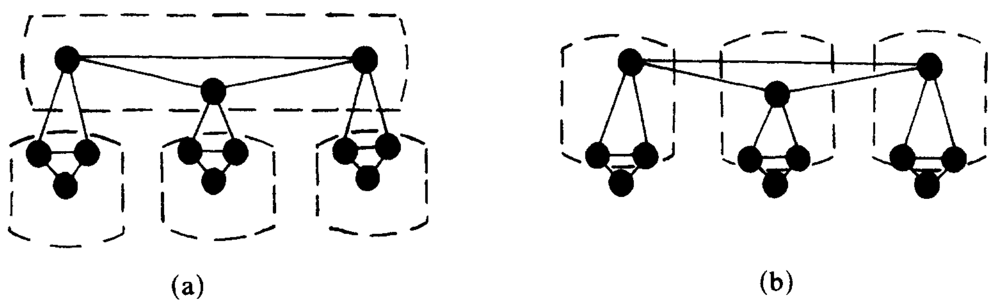

Fig. 2. Clusterings for the components representing triplets in C' (a) and the components representing triplets in $C - C'$ (b).

Since all nodes must be included in exactly one cluster, it must be that there are exactly q components clustered as in Fig. 2(a) and each of these components must include a different set of v-nodes. It is simple to show that the triplets in C represented by these q components form an exact cover for C. This completes the proof of the claim and therewith the lemma. □

Before proving our next result, we outline the construction to be used in it. First of all, the construction used in Lemma 3.1 (Fig. 1) is replaced by the one given in Fig. 3.

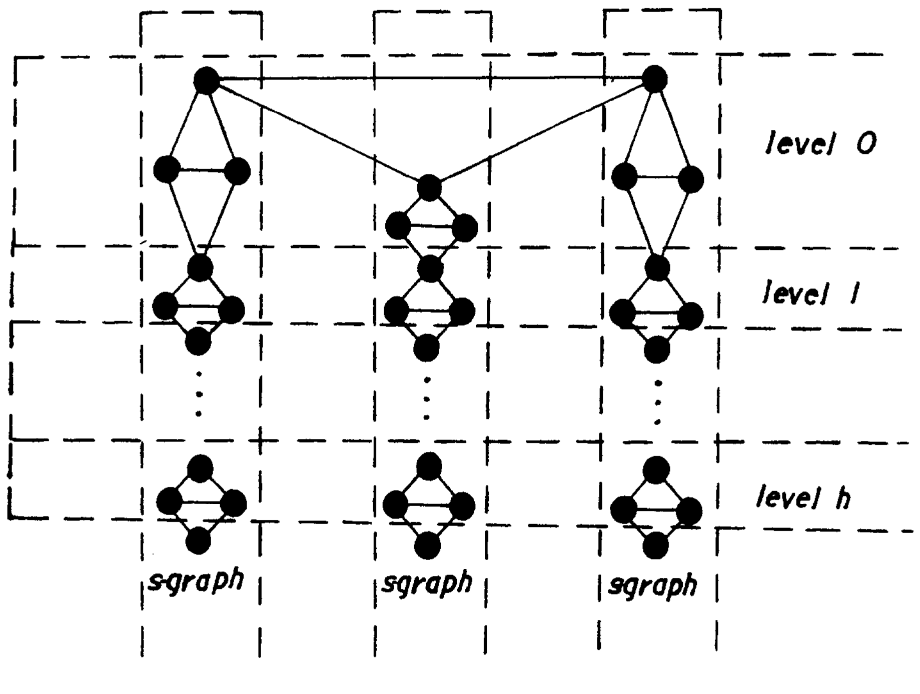

Fig. 3. Subgraph (component) representing each triplet in C.

Each vertical subgraph in Fig. 3 is called an s-graph. The topmost three nodes in each s-graph are said to be in *level* 0, the next set of three nodes belong to *level* 1 and so on, until *level h* is reached.

It is simple to show that not all the graphs constructed by using the above rule, starting with an instance of RXC3, are planar. In order to guarantee planarity, we shall modify our construction rule. The value for h is selected in such a way that at each level z ($z \geq 1$) only two adjacent s-graphs cross and at level h the three s-graphs that include node v_i are side by side without having edges cross. It is simple to show that a value for $h \leq (3m)^2$ can always be found. The crossing of two s-graphs at level z is handled by applying the transformation shown in Fig. 4.

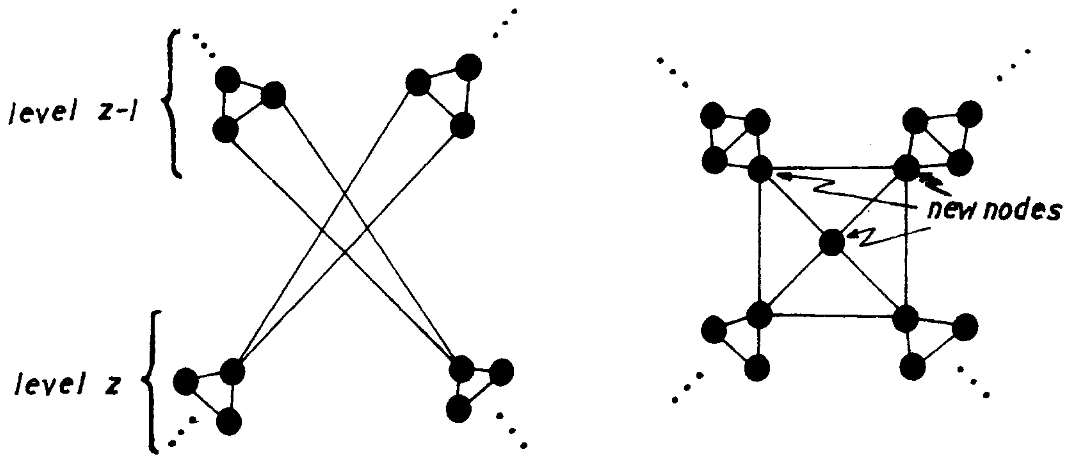

Fig. 4. Elimination of the crossing at level z between two s-graphs.

Lemma 3.2. *The decision problem k-gMM is NP-complete even when the input graph, after deleting all the edges with a weight different than one, is planar and no node is of degree greater than six (we call this problem 'restricted k-gMM').*

Proof. Clearly this problem is in NP. Let us now show that RXC3 α restricted k-gMM. Given any instance (X, C) of the exact cover by 3-sets, we construct an instance $KG = (G = (V, E, W), k, d)$ of the restricted k-gMM decision problem. The construction follows the rules described immediately preceding this lemma. All the edges introduced by the rules implied by Figs. 3 and 4 will have the weight of one (note that every time we use the rule given by Fig. 4 four of the edges introduced by Fig. 3 are deleted). Since the graph is complete, the remaining edges will receive a weight of two. It is simple to show that the number of nodes is $3(q + 3m(h+1) + h)$. The value for d is one and the number of clusters, k, is $q + 3m(h+1) + h$. Clearly, KG is an instance of the restricted k-gMM problem and it can be constructed in polynomial time (with respect to the size of the RXC3 problem). In order to complete the proof of the lemma it is only required to show that KG has a k-split with objective function value $\leq d$ iff (X, C) has an exact cover.

Claim. KG *has a k-split with objective function value $\leq d$ iff (X, C) has an exact cover.*

Proof. Our proof follows the same lines as the one for Lemma 3.1. First we show that if (X, C) has an exact cover, then KG has a k-split, $(B_1, B_2, \ldots, B_k)$, with objective function value $\leq d = 1$. Let C' be any exact cover for (X, C). If we consider the graph before introducing the crossings (Fig. 4), the grouping of nodes into clusters is similar to the one used in the previous lemma for the components representing elements in $C - C'$ and C'. The way one deals with the crossings is shown in Fig. 5. It is simple to verify that the clustering obtained by following the above procedure is a k-split for KG with objective function value $\leq d = 1$.

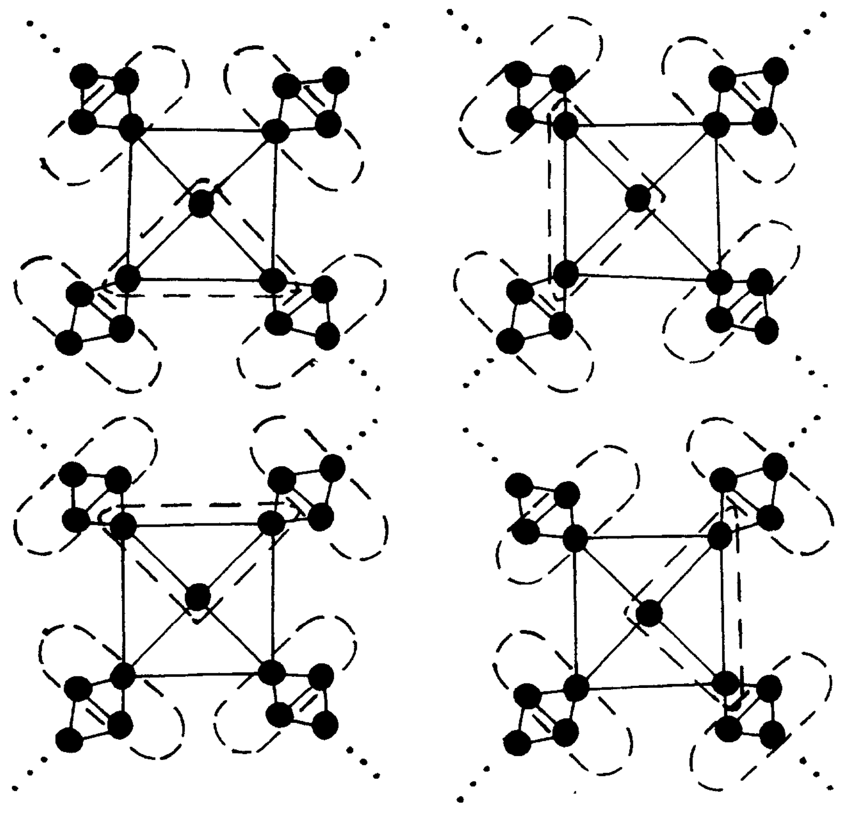
Fig. 5. Clustering at the crossings.

We now show that if KG has a k-split, $(B_1, B_2, \ldots, B_k)$, with objective function value $\leq d = 1$, then (X, C) has an exact cover. Let $B_1, B_2, \ldots, B_k$ be a k-split with objective function value $\leq d$ for KG. Each B_i has exactly three nodes, since there is no 4-clique in which all edges have weight one and the number of nodes in KG is $3k$. Because of this and the construction rules, the top part of each component is clustered as the top part of the component in Fig. 2(a) or (b). If we now show that the clustering in the bottom part, where the v-nodes appear, is preserved as in Fig. 2(a) and (b), then the same arguments as the ones in the previous lemma can be used to complete the proof of the lemma.

Our proof has been reduced to showing that the crossings behave as in Fig. 5. We prove this by contradiction. Suppose that in some crossing we do not preserve the desired clusterings. Then it must be one of the two cases shown in Fig. 6 or the mirror images of these cases. We ignore the latter two cases since their proofs are similar to the one for the two cases given by Fig. 6.

For the case given by Fig. 6(a) we have that the center node has to be cluster with two other elements, but there are no two elements not yet been clustered that form a clique of size three in which all edges have a weight of one with the center node. Hence, either there is a cluster with less than three nodes or a cluster has an

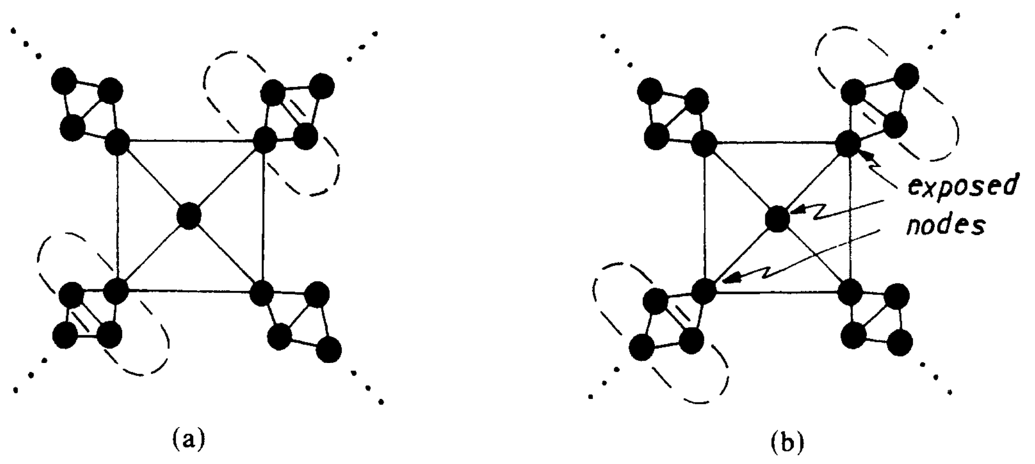

Fig. 6. Invalid clusterings.

edge with a weight greater than d. A contradiction. For the case in Fig. 6(b) we have that the center node can only cluster with one of the exposed nodes. The other exposed node does not have two other nodes not yet been clustered with whom it forms a clique of size three in which all edges have a weight of one. Therefore, either there is a cluster with less than three nodes or there is an edge in a cluster with weight greater than d. In both cases we have a contradiction. This completes the proof of the claim and therewith the lemma. □

In Fig. 1 the subgraphs consisting of two triangles placed side by side will be called a *diamond*. The *ends* are the two nodes of degree two in it. It should be clear that two diamonds joined one after the other by their ends can replace any diamond in the construction used in Lemma 3.2 and the resulting construction can also be used to prove our result as long as the number of diamonds replaced is bounded by a polynomial on n (the number of nodes in the graph).

In the final transformation we replace the constructions implied by Figs. 2 and 3 by the one in Fig. 7. Note that the replacement indicated in the bottom left part of Fig. 7 is only applied to the v-nodes.

After taking care of some simple details one can show that two points are at a distance $\leq d+\varepsilon$ iff these two points had an edge between them with a weight of one in the construction used in Lemma 3.2 (after having added several diamonds as implied by Fig. 7).

Theorem 3.3. *The* k-2MM *decision problem is* NP-*complete.*

Proof. The construction used to prove this result follows the rules shown in Fig. 7 and the proof is similar to Lemma 3.2. □

4. The complexity of the approximation problem

In this section we examine the computational complexity of generating approximate solutions to our clustering problem. It will be shown that the approximation

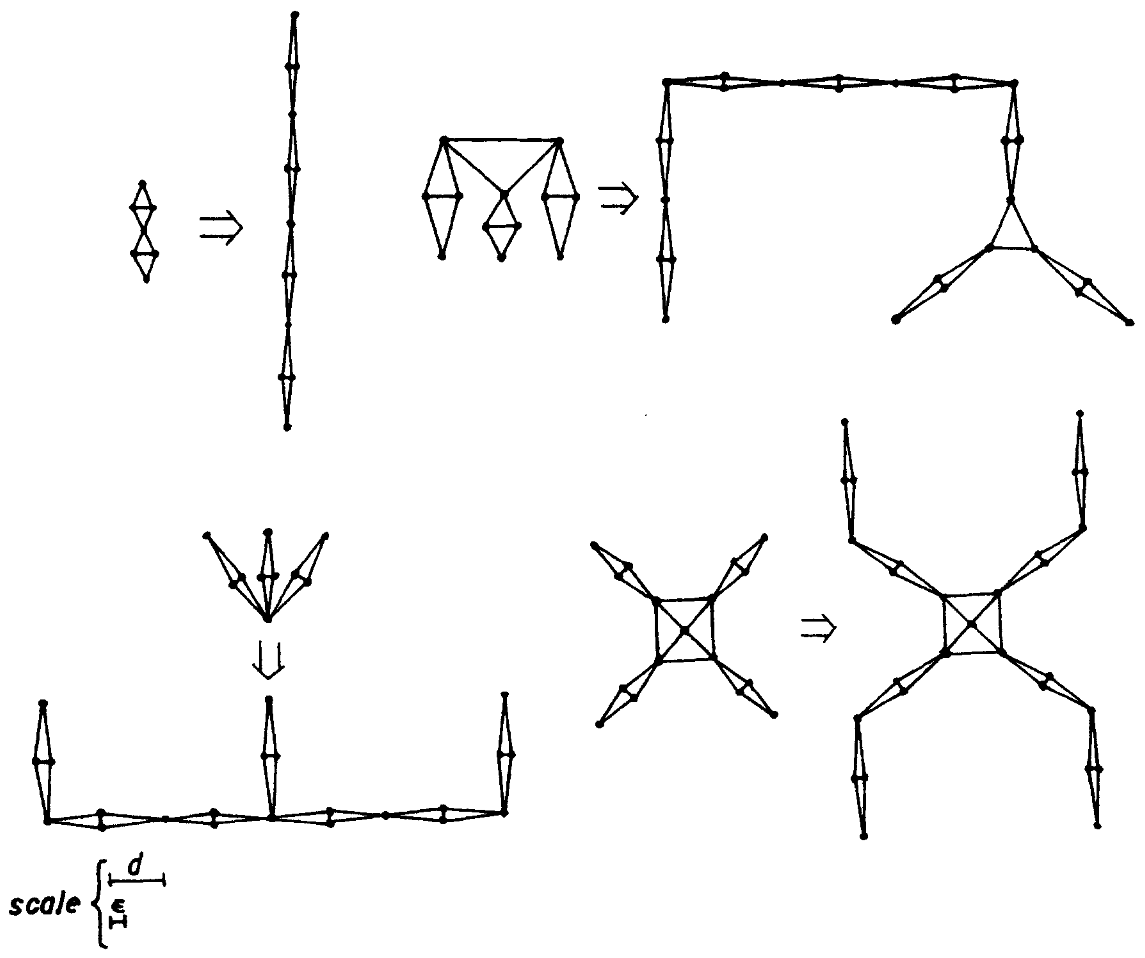

Fig. 7. Final transformations.

algorithm presented in Section 2 is best possible, with respect to the approximation bound, if $P \neq NP$.

After a careful examination of the construction rules shown in Fig. 7, one can easily prove that the closest three points not at a distance $\leqslant d+\varepsilon$ of each other in the construction for Theorem 3.3 is at least $\sqrt{2} * d$. Using this fact together with the techniques used in [17], one can prove the following result.

Lemma 4.1. *The k-2MM $(\sqrt{2}-\varepsilon)$-approximation problem is* NP*-hard for all* $\varepsilon > 0$.

Proof. The proof follows from the above discusion. □

This result can be strengthened by replacing the construction used for the crossing of two s-graphs by the one shown in Fig. 8.

Theorem 4.2. *The k-2MM $(2\cos(\frac{1}{6}\pi)-\varepsilon)$-approximation problem is* NP*-hard for all* $\varepsilon > 0$.

Proof. The construction used in this proof is similar to the one used in Theorem 3.3. The main difference is in the way one deals with the crossing of two s-graphs. In this construction, the crossing of the two s-graphs is avoided by using the

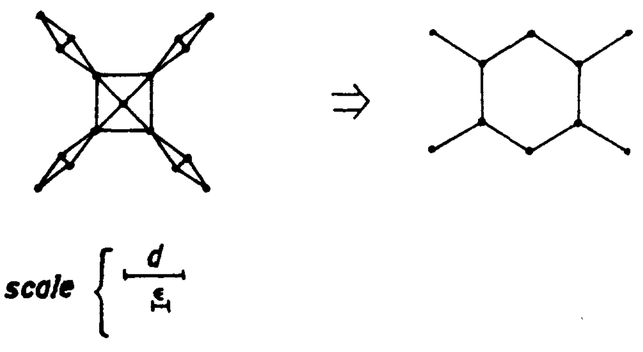

Fig. 8.

construction shown in Fig. 8. Also, the diamonds must be rearranged in such a way that the angle between any two adjacent diamonds is at least 120°. The proof of this theorem is similar to Lemma 4.1. The main difference is that one has to show that solutions with the desired objective function value have exactly 2 clusters in each hexagon (Fig. 8). □

Note that if instead of using the hexagon (Fig. 8) one uses a figure with more than six sides, then one cannot improve on the bound given by Theorem 4.2. The reason for this is that the smallest angle between the two sides (adjacent to a 'leg') in the new figure and the 'leg' would be less than 120°. This will allow invalid clustering with objective function value $<2\cos(\frac{1}{6}\pi)$. A stronger result can be obtained for three or more dimensions because crossings can be avoided by using the third dimension to avoid the edge overlap.

Theorem 4.3. *The k-3MM $(2-\varepsilon)$-approximation problem is* NP*-hard for all $\varepsilon > 0$.*

Proof. The proof follows by the above discusion. □

Clearly, the result for the k-3MM problem also holds for the k-tMM problem. However, a much simpler proof can be obtained by using the techniques in [17] and the reduction in Lemma 2.1. That instance KG used in Lemma 2.1 satisfies the triangular inequality because all the weights are 1 or 2 and the graph is completely connected.

5. Discussion

We have shown that the k-2MM $(2\cos(\frac{1}{6}\pi)-\varepsilon)$-approximation and the k-3MM $(2-\varepsilon)$-approximation problems are NP-hard. Both of these results also hold for the k-tMM problem. For all of these problems we showed that the 2-approximation problem can be solved in $O(nk)$ time. Clearly, for three or more dimensions our approximation algorithm is best possible, with respect to the approximation bound, if $P \neq NP$. Our results can also be used to show that the k-2MΣ and its corresponding approximation problem are NP-hard. The objective function in the k-gMΣ problem is the max$\{S_l \mid S_l$ represents the sum of the weight of the edges whose endpoints are in cluster $l\}$. The construction used in Section 3 can be easily adapted to show that partition of a graph into triangles is NP-hard even when the graphs to be partitioned are planar and no node is of degree greater than six. Recently, Hochbaum and Shmoys [12] obtained similar results for the k-tMM problem. However, their results do not apply for the k-mMM problem, where m is any integer greater than one.

Appendix A

In this appendix we show that the RXC3 problem is NP-complete. This is accomplished by reducing the XC3 problem to the RXC3 problem.

Theorem A.1. *The* RXC3 *problem is* NP-*complete.*

Proof. Since it is simple to show that RXC3 is in NP, we only show that XC3 $\propto$ RXC3. Given an instance (X, C) of the XC3 problem, the following procedure returns an instance of the RXC3 problem:

```
Procedure BUILD ($X$, $C$);
  begin
    if there is some element in $X$ that appears in no subset in $C$ then
        return $(\{x_1, x_2, \ldots, x_6\}, \{(x_1, x_2, x_3), (x_1, x_2, x_4),$
                $(x_1, x_5, x_6), (x_2, x_5, x_6), (x_3, x_4, x_5), (x_3, x_4, x_6)\})$;
    endif
    if the number of elements in $X$ that are included in exactly one triplet in $C$ is
        not a multiple of three then
          Let $(X, C)$ be three copies of the old $(X, C)$
    endif
    while there is an element in $X$ that does not appear in exactly three triplets in
        $C$ do
        let $x_i$, $x_j$ and $x_k$ be three distinct elements in $X$ such that all of them appear
          in either one or two triplets in $C$;
        let $y_1$, $y_2$ and $y_3$ be three elements not in $X$;
        add $y_1$, $y_2$ and $y_3$ to $X$;
        add $(x_i, y_1, y_2)$, $(x_j, y_2, y_3)$, $(x_k, y_3, y_1)$, $(y_1, y_2, y_3)$ to $C$;
    endwhile
    return $(X, C)$;
end of procedure;
```

It is simple to show that:

(1) the above procedure can be carried out in polynomial time,

(2) the output produced by the algorithm is an instance of the RXC3 problem, and

(3) the instance of the RXC3 constructed by the algorithm has an exact cover iff there is an exact cover for (X, C).

One can easily use the facts just mentioned above to complete the proof of the theorem. □

References

[1] J.G. Augustson and J. Minker, An analysis of some graph theoretical cluster techniques, *J. ACM* **17** (1970) 571-588.

[2] H.H. Bock, *Automatische Klassifikation* (Vandenhoek und Ruprecht, Gottingen, 1974).

[3] L.D. Bodin, A graph theoretic approach to the grouping of ordering data, *Networks* **2** (1972) 307-310.

[4] P. Brucker, On the complexity of clustering problems, in: R. Henn, B. Korte and W. Oletti, eds., *Optimizing and Operations Research*, Lecture Notes in Economics and Mathematical Systems (Springer, Berlin, 1977).

[5] R. Duda and P. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973).

[6] L. Fisher and J. Van Ness, Admissible clustering procedures, *Biometrica* **58** (1971) 91-104.

[7] W.D. Fisher, On grouping for maximum homogeneity, *JASA* **53** (1958) 789-798.

[8] T.F. Gonzalez, On the computational complexity of clustering and related problems, in: *Proc. 10th IFIP Conf. on System Modeling and Optimization* (1981) 174-182.

[9] M.R. Garey and D.S. Johnson, The complexity of near-optimal graph coloring, *J. ACM* **23**(1) (1976) 43-69.

[10] M.R. Garey and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman, San Francisco, 1980).

[11] E. Horowitz and S. Sahni, *Fundamentals of Computer Algorithms* (Computer Science Press, Potomac, 1978).

[12] D.S. Hochbaum and D.B. Shmoys, Powers of graphs: A powerful approximation technique for bottleneck problems, in: *Proc. 1984 STOC Conf.*, 1984.

[13] D.B. Johnson and J.M. Lafuente, Controlled single pass classification algorithm with applications to multilevel clustering, Scientific Rept. #ISR-18, Information Science and Retrieval, Cornell University, 1970.

[14] F.J. Rohlf, Single link clustering algorithms, RC 8569 (#37332) Res. Rept., IBM Th.J. Watson Research Center, 1980.

[15] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller and J.W. Thatcher, eds., *Complexity of Computer Computations* (Plenum Press, New York, 1972) 85-104.

[16] W.S. Meisel, *Computer-Oriented Approaches to Pattern Recognition* (Academic Press, New York, 1972).

[17] S. Sahni and T.F. Gonzalez, P-complete approximation problems, *J. ACM* **23** (1976) 555-565.

[18] G. Salton, *The Smart Retrieval System, Experiments in Automatic Document Processing* (Prentice-Hall, Englewood Cliffs, NJ, 1971).

[19] G. Salton, *Dynamic Information and Library Processing* (Prentice-Hall, Englewood Cliffs, NJ, 1975).

[20] M.I. Shamos, Geometry and statistics: Problems at the interface, in: J.F. Traub, ed., *Algorithms and Complexity: New Directions and Recent Results* (Academic Press, New York, 1976) 251-280.