Nordic Journal of Computing (2000), ??–??.

APPROXIMATION ALGORITHMS FOR CLUSTERING TO MINIMIZE THE SUM OF DIAMETERS

SRINIVAS DODDI
Timbre Technologies, Freemont, CA.
srini@timbrecom.com

MADHAV V. MARATHE
Los Alamos National Laboratory, P.O. Box 1663
MS M997, Los Alamos, NM 87545.
marathe@lanl.gov

S. S. RAVI
Department of Computer Science
University at Albany - SUNY
Albany, NY 12222.
ravi@cs.albany.edu

DAVID SCOT TAYLOR
Department of Computer Science
University of California, Los Angeles, CA 90095-1596.
dstaylor@cs.ucla.edu

PETER WIDMAYER
Institute for Theoretical Computer Science
ETH, 8092 Zürich, Switzerland.
widmayer@inf.ethz.ch

Abstract. We consider the problem of partitioning the n nodes of a complete edge weighted graph into k clusters so as to minimize the sum of the diameters of the clusters. Since the problem is NP-complete, our focus is on the development of good approximation algorithms. When edge weights satisfy the triangle inequality, we present the first approximation algorithm for the problem. The approximation algorithm yields a solution which has no more than $O(k)$ clusters such that the sum of cluster diameters is within a factor $O(\ln(n/k))$ of the optimal value using exactly k clusters. Our approach also permits a tradeoff among the constant terms hidden by the two big-O terms and the running time. For any fixed k, we present an approximation algorithm that produces k clusters whose total diameter is at most twice the optimal value. When the distances are not required to satisfy the triangle inequality, we show that, unless P = NP, for any $\rho \geq 1$, there is no polynomial time approximation algorithm that can provide a performance guarantee of ρ even when the number of clusters is fixed at 3. We also present some results for the problem of minimizing the sum of cluster radii.

Received August 2000; revised October 2000; accepted xxx, 2000.

CR Classification: G.2.2

Key words: clustering, data mining, approximation algorithms, bicriteria problems, combinatorial algorithms

1. Introduction

1.1 Motivation

The main goal of clustering is to partition a set of objects into homogeneous and well separated subsets (clusters). Clustering techniques have been used in a wide variety of application areas including information retrieval, image processing, pattern recognition and database systems [12], [23], [31], [32]. Over the last three decades, several clustering methods have been developed for specific applications [19], [23]. Many of these methods define a distance (or a similarity measure) between each pair of objects, and partition the collection into clusters so as to optimize a suitable objective based on the distances. Some of the objectives that have been studied in the literature include minimizing the maximum diameter or radius, total pairwise distances in clusters, etc. The survey by Hansen and Jaumard [19] provides an extensive list of clustering objectives and applications for these objectives.

Clustering problems where the objective is to minimize the maximum cluster diameter have been well studied from an algorithmic point of view (see Section 1.4 for a summary). The focus of this paper is on clustering problems where the objective is to partition a given collection of objects into a specified number of clusters so as to minimize the sum of the diameters of individual clusters. The motivation for this objective is derived from the fact that in several applications, clustering algorithms that minimize the maximum diameter produce a "dissection effect" [19], [28]. This effect causes objects that should normally belong to the same cluster to be assigned to different clusters, as otherwise the diameter of a cluster becomes too large. In such applications, the sum of diameters objective is more useful as it reduces the dissection effect [19], [28].

1.2 Problem Formulation and Previous Work

To study the clustering problem in a general setting, we represent the objects to be clustered as nodes of a complete edge-weighted undirected graph $G(V,E)$ with $|V| = n$. The distance (or similarity measure) between any pair of objects can then be represented as the weight of the corresponding edge in E. For an edge $\{u,v\}$ in E, we use $\omega(u,v)$ to denote the weight of the edge. It is assumed that the edge weights are nonnegative. For any subset V' of V, the **diameter** of V', denoted by $\mathrm{DIA}(V')$, is the weight of a largest edge in the complete subgraph of G induced on V'. Note that when $|V'| = 1$, $\mathrm{DIA}(V') = 0$. A formal statement of the clustering problem considered in this paper is as follows.

Clustering to Minimize Sum of Diameters (CMSD)

Instance: A complete graph $G(V, E)$, a nonnegative weight (or distance) $\omega(u, v)$ for each edge $\{u, v\}$ in E and an integer $k \leq |V|$.

Requirement: Partition V into at most k subsets V_1, V_2, ..., V_k such that $\sum_{i=1}^{k} \mathrm{DIA}(V_i)$ is minimized.

In general, edge weights in instances of CMSD need not satisfy the triangle inequality. We use CMSD_Δ to denote instances of CMSD where edge weights satisfy the triangle inequality. Most of our results are for the CMSD_Δ problem. We assume without loss of generality that the optimal solution value to any given instance of CMSD_Δ is strictly greater than zero. We may do so since it is easy to determine whether a given instance of CMSD_Δ can be partitioned into a specified number of clusters each of which has a diameter of zero.

We now summarize the known results from the algorithmic literature for the CMSD problem. Brucker [5] showed that CMSD is NP-complete for any fixed $k \geq 3$. Hansen and Jaumard [18] studied the CMSD problem with $k = 2$ and presented an algorithm with a running time of $O(n^3 \log n)$. They also showed that for $k = 2$, the minimization problem for any given function of the two diameters can be solved in $O(n^5)$ time. When the input is specified as an undirected edge weighted graph with n nodes and m edges, Monma and Suri [28] showed that the CMSD problem for $k = 2$ can be solved in $O(nm \log n)$ time. This is an improvement over the algorithm of [18] for sparse graphs. Brucker [5] observed that the 1-dimensional version of CMSD_Δ can be solved efficiently for any value of k. For the Euclidean version of CMSD_Δ with $k = 2$, Monma and Suri [28] presented an algorithm which uses $O(n)$ space and runs in $O(n^2)$ time. Capoyleas et al. [6] also studied a generalized version of the CMSD_Δ problem for points in $\Re^2$. They showed that for any *fixed* k, the problem can be solved in polynomial time for any monotonic increasing function of cluster diameter or radius. Examples of such monotonic increasing functions include sum of diameters (or radii), maximum diameter (or radius), etc.

1.3 Summary of Results

We study the complexity and approximability of the CMSD problem. The main results of this paper can be summarized as follows:

(1) We show that unless P = NP, CMSD cannot be efficiently approximated to within any factor even when the number of clusters is fixed at 3. (In contrast, CMSD is known to be efficiently solvable when the number of clusters is equal to 2 [18], [28].)

(2) For CMSD_Δ, we show that if the constraint on the number of clusters must be met, then it is NP-hard to approximate the total diameter to within a factor $2 - \epsilon$, for any $\epsilon > 0$.

(3) In contrast to the non-approximability results above, we present a polynomial time bicriteria approximation algorithm [25] for CMSD_Δ. For fixed positive parameters x, ϵ and δ, this approximation algorithm outputs a solution with at most $((1+\frac{2}{x})\frac{e}{e-1}+\delta)k$ clusters, whose total diameter is within a factor of $(1+\epsilon)(x+2)(\ln(\frac{n}{k\delta}))$ of the minimum possible total diameter with k clusters. Here, the parameters x, ϵ and δ act to control a tradeoff between the cost of the solution, the number of clusters returned, and the running time of the algorithm. (Very recently, Charikar and Panigrahy [8] have developed approximation algorithms with better performance guarantees for CMSD_Δ. We elaborate in our conclusions.)

(4) When the number of clusters k is fixed, we present a simple approximation algorithm for CMSD_Δ which produces at most k clusters whose total diameter is within a factor of 2 of the optimal value. (It should be noted that for fixed $k \geq 3$, the complexity of CMSD_Δ is open.)

(5) When the number of clusters is fixed, we show that the problem of minimizing the sum of the cluster radii can be solved in polynomial time even for non-metric instances. This points out an interesting difference between minimizing the sum of cluster diameters (which is NP-hard in the non-metric case even for three clusters) and minimizing the sum of cluster radii.

1.4 Other Related Work

A number of researchers have addressed the clustering problem where the goal is to minimize the maximum diameter or radius of a cluster. In the location theory literature, the problem of minimizing the maximum radius is also known as the k-center problem. For the metric version of the problem of minimizing the maximum diameter, Gonzalez [16] presented a simple greedy heuristic that runs in $O(nk)$ time and provides a performance guarantee of 2. He also showed that, unless P = NP, the performance guarantee cannot be improved. Using a general technique for approximating bottleneck problems, Hochbaum and Shmoys [21] also presented a heuristic with a performance guarantee of 2 for the metric version of the k-center problem.

In [14], [27], it is shown that the problems of minimizing the maximum radius or diameter remain NP-hard even for points in $\Re^2$. For this geometric version, Feder and Greene [13] improved the running time of Gonzalez's heuristic to $O(n \log n)$. They also showed that it is NP-hard to achieve performance guarantees of 1.82 and 1.97 respectively for the diameter and radius problems in $\Re^2$. Recently, Agarwal and Procopiuc [1] have presented an exact algorithm with a running time of $n^{O(k^{1-1/d})}$ for the k-center problem for points in $\Re^d$. For any $\epsilon > 0$, they have also presented an $(1+\epsilon)$ approximation algorithm with a running time of $O(n \log k) + (k/\epsilon)^{O(k^{(1-1/d)})}$ for the problem.

Guttmann-Beck & Hassin [17] also study a clustering problem where the goal is to partition the vertex set of a complete edge weighted graph into k clusters. In their formulation, the measure for any cluster is the *total* weight of all the edges in the cluster, and the objective is to minimize the sum of the measures of the k clusters. (In contrast, our measure for any cluster is the *maximum* weight of an edge in the cluster.) Under the assumptions that the number of clusters k is *fixed* and that the edge weights satisfy the triangle inequality, they study two versions of the clustering problem. In one version, the size of each cluster is also specified as part of the problem instance, while in the other, the algorithm has the freedom to choose arbitrary cluster sizes. For both versions, they give approximation algorithms that are within a factor 2 of the optimal value. For the special case where $k = 2$, the graph $G(V, E)$ has an even number of nodes and the size of each cluster is $|V|/2$, they give an approximation algorithm with a performance guarantee of 1.7. For the case when $k = 2$ and the sizes of the two clusters may be arbitrary, Indyk [22] has presented a randomized polynomial approximation scheme (PTAS). An interesting aspect of this PTAS is that its running time is *sub-linear* in the size of the input.

Plesník [30] has addressed the problem of partitioning the *edges* of a given graph $G(V, E)$ into k subsets so that each subset forms a connected graph on the vertex set V, and a given function of the diameters of the resulting subgraphs is minimized. The objectives considered in [30] include minimizing the maximum diameter and minimizing the total diameter. It is shown that, unless $\mathsf{P} = \mathsf{NP}$, even for $k = 2$, these objectives cannot be efficiently approximated to within factors less than 3/2 and 5/4 respectively.

Several other types of clustering problems have also been studied in the literature. For example, Charikar et al. [7] study an incremental version of the clustering problem for minimizing the maximum radius. Pferschy et al. [29] study geometric versions of clustering problems using objectives such as minimizing the total perimeter. Agarwal and Procopiuc [2] study projective clustering problems where the goal is to cover a set of points in $\Re^d$ using hyper-strips, and the objective is to minimize the maximum width of the strips.

References where other types of clustering problems are studied include [3], [4], [9], [10], [26].

1.5 Organization

The remainder of this paper is organized as follows. In Section 2, we establish some preliminary results. In Section 3, we present our approximation results for $\textsc{Cmsd}_\Delta$. Section 4 presents lower bounds on achievable performance guarantees. Section 5 presents some results for the clustering problem where the goal is to minimize the sum of cluster radii. Some conclusions and open problems are presented in Section 6.

2. Preliminaries

2.1 Simple Bounds on the Optimal Solution Value

As mentioned earlier, we assume that the given instance of CMSD_Δ has a solution value greater than zero. For such an instance, it is obvious that the optimal solution value is at least as large as the smallest nonzero edge weight. Further, we can easily construct a feasible solution consisting of k clusters with total diameter at most the maximum edge weight: form one cluster consisting of $n - k + 1$ arbitrarily chosen vertices and make each of the remaining $k - 1$ vertices a singleton cluster. These observations are stated formally below.

OBSERVATION 1. Let I be an instance of CMSD_Δ such that the optimal solution value $\text{OPT}(I)$ is positive. Then

(1) $\text{OPT}(I)$ is at least as large as the smallest nonzero edge weight in I.

(2) $\text{OPT}(I)$ is at most the maximum edge weight in I. □

2.2 A Merging Lemma

The formulation of the CMSD problem requires that the clusters be pairwise disjoint. Clusters produced during the intermediate stages of our approximation algorithms may not satisfy the disjointness condition. The following lemma points out that for instances of CMSD_Δ, we can merge pairs of intersecting clusters without increasing the total diameter.

LEMMA 1. *Let I be an instance of* CMSD_Δ *given by the edge weighted complete graph $G(V, E)$ and integer k. Let $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ be a collection of subsets of V such that their union is V and the sum of the diameters of all the subsets in $\mathcal{C}$ is ψ. Further, suppose C_i and C_j $(i \neq j)$ are two sets in $\mathcal{C}$ such that $C_i \cap C_j \neq \emptyset$. Then the total diameter of the collection $\mathcal{C}'$ obtained by deleting C_i and C_j from $\mathcal{C}$ and adding the set $C_i \cup C_j$ is at most ψ.*

PROOF. The lemma would follow by showing that $\text{DIA}(C_i \cup C_j) \leq \text{DIA}(C_i) + \text{DIA}(C_j)$. To do this, let x be a node in $C_i \cap C_j$ and let u and v be two nodes in $C_i \cup C_j$ such that $\omega(u, v) = \text{DIA}(C_i \cup C_j)$. If u and v are both in C_i (or both in C_j), then $\omega(u, v) \leq \text{DIA}(C_i)$ $(\omega(u, v) \leq \text{DIA}(C_j))$, and the proof is trivial. So, assume that $u \in C_i$ and $v \in C_j$. By the triangle inequality, $\omega(u, v) \leq \omega(u, x) + \omega(v, x)$. Since u and x are both in C_i, $\omega(u, x) \leq \text{DIA}(C_i)$. Similarly, $\omega(v, x) \leq \text{DIA}(C_j)$. Therefore, $\text{DIA}(C_i \cup C_j) = \omega(u, v) \leq \text{DIA}(C_i) + \text{DIA}(C_j)$, and this completes the proof. □

In view of the above lemma, when considering instances of CMSD_Δ, we may repeatedly merge pairs of clusters with nonempty intersection until the clusters are pairwise disjoint. The merging process does not increase the total diameter of the clusters.

2.3 Transformation to Weighted Set Cover

TRANSFORMTOSETCOVER $(G(V,E), k, f)$
Input: A complete edge weighted graph $G(V,E)$, the number of clusters k and a nonnegative parameter f.

Output: An instance of weighted set cover problem with base set Q, and collection $\mathcal{W}$ of nonempty subsets of Q, each with a weight. The weight of a set $W \in \mathcal{W}$ is denoted by $c(W)$.

(1) $Q = V$. /* Note: $|Q| = n$. */

(2) $\mathcal{W} = \emptyset$.

(3) **for** each $v \in V$ **do**

 (a) Let $S_v = \{\omega(v,u) : u \in V\}$ and let $r_v = |S_v|$.
 (b) Sort S_v into increasing order. Let $\alpha_1 = 0 < \alpha_2 < \ldots < \alpha_{r_v}$ denote the sorted order.
 (c) **for** $i = 1$ **to** r_v **do**
  Let $W_v^i = \{u : \omega(u,v) \leq \alpha_i\}$.
  Let $c(W_v^i) = \mathrm{DIA}(W_v^i) + f/k$.
  Add (W_v^i) to $\mathcal{W}$.

(4) **return** $(c, Q, \mathcal{W})$

Our results rely on a transformation from instances of CMSD$_\Delta$ to instances of the weighted set cover problem. Given an instance of CMSD$_\Delta$ along with a nonnegative value f, the transformation TRANSFORMTOSETCOVER produces an instance of the weighted set cover problem. The value of f will be chosen suitably to limit the number of sets that can be chosen in a set cover. For each node $v \in V$, the transformation considers the nodes in V in increasing order of distances from v. For each distinct distance d, the transformation outputs a set consisting of all the nodes which are at a distance of at most d from v. The weight of each set W is $\mathrm{DIA}(W) + f/k$. It is clear that the transformation can be carried out in polynomial time. The following lemma points out an important property of the resulting set cover instance.

LEMMA 2. *Let I denote an instance of* CMSD$_\Delta$ *problem and let f be a nonnegative number. Let I' denote the instance of the weighted set cover problem produced by the transformation* TRANSFORMTOSETCOVER *when I and f are given as inputs. Let* $\mathrm{OPT}(I)$ *and* $\mathrm{OPT}(I')$ *denote the optimum solution values to I and I' respectively. Then,* $\mathrm{OPT}(I') \leq 2\,\mathrm{OPT}(I) + f$.

PROOF. Let $C_1, C_2, \ldots, C_k$ denote the clusters in an optimal solution to I. Thus, $\mathrm{OPT}(I) = \sum_{i=1}^{k} \mathrm{DIA}(C_i)$. We will show that there is a subcollection of k sets in I' such that the sets in the subcollection together cover the base set Q and the total weight of the sets in the subcollection is at most $2\,\mathrm{OPT}(I) + f$. The lemma would then follow immediately.

Consider each cluster C_i $(1 \le i \le k)$ in the optimal solution to I. If C_i contains two or more nodes, let v_i be a node in C_i such that v_i is one of the endpoints of an edge whose weight is equal to $\mathrm{DIA}(C_i)$. If C_i contains only one node (i.e., $\mathrm{DIA}(C_i) = 0$), let v_i be that node. Now, by the transformation TransformToSetCover, I' has a set, say W_i, that includes all the nodes which are at a distance of at most $\mathrm{DIA}(C_i)$ from v_i. By the triangle inequality, $\mathrm{DIA}(W_i) \le 2\,\mathrm{DIA}(C_i)$. So, $c(W_i) = \mathrm{DIA}(W_i) + f/k \le 2\,\mathrm{DIA}(C_i) + f/k$. Clearly, the subcollection $\{W_1, W_2, \ldots, W_k\}$ covers the base set Q. The weight of this cover is $\sum_{i=1}^{k} c(W_i)$, which is at most $\sum_{i=1}^{k}(2\,\mathrm{DIA}(C_i) + f/k) = 2\,\mathrm{OPT}(I) + f$. This completes the proof of the lemma. □

2.4 The Budgeted Maximum Coverage Problem

For obtaining our approximation result for CMSD_Δ (where the number of clusters k is a part of the problem instance), we use a known approximation result for the Budgeted Maximum Coverage Problem (Bmcp). Below, we provide a definition of the problem and state a known approximation result for the problem.

An instance of Bmcp consists of a base set $Q = \{q_1, q_2, \ldots, q_n\}$, a collection $\mathcal{W}$ of nonempty subsets of Q, a nonnegative weight $c(W)$ for each set $W \in \mathcal{W}$ and a nonnegative budget B. The goal is to choose a subcollection of sets from $\mathcal{W}$ so that the total cost of the chosen sets is at most B and the number of elements covered by the chosen sets is maximum. This problem is NP-hard since it is a restatement of the minimum cost set cover problem. The following approximation result for Bmcp is proved in [24].

Theorem 1. Bmcp *can be efficiently approximated to within a factor* $(1 - 1/e)$. □

In the sequel, the approximation algorithm referred to in Theorem 1 will be denoted by Approx-Bmcp. It is shown in [24] that Approx-Bmcp can also be used for the more general version of the Bmcp problem where there is a weight associated with each element of the base set, and the goal is to maximize the weight of the elements covered by the chosen sets. For our results, the unit weight version of Approx-Bmcp where the weight of each element of the base set is 1, suffices.

3. Approximating CMSD$_\Delta$

APPROX-CMSD$_\Delta$ $(G(V,E),k)$

Input: Complete edge weighted graph $G(V,E)$, the number k of clusters. (The algorithm also uses three fixed positive parameters x, ϵ and δ.)

Output: A set of no more than $((1+\frac{2}{x})\frac{e}{e-1}+\delta)k$ clusters, with sum of diameters at most $(1+\epsilon)(x+2)(\ln(\frac{n}{k\delta}))\,\mathrm{OPT}(I)$.

(1) $\mathcal{D} = \{\{v\} \ : \ v \in V\}$.

(2) OPTLowerBound = the smallest nonzero edge weight in $G(V,E)$.

(3) **while** $(|\mathcal{D}|((1+\frac{2}{x})\frac{e}{e-1}+\delta)k)$ **do**

- (a) N = Set of vertices obtained by choosing one arbitrary vertex from each $D \in \mathcal{D}$.
- (b) E' = Edges in the complete subgraph induced by N.
- (c) $\mathcal{S}$ =TRANSFORMTOSETCOVER $(G(N,E'),k,x \cdot$ OPTLowerBound).
- (d) $\mathcal{C}$ = APPROX-BMCP$(\mathcal{S},(x+2)$OPTLowerBound).
- (e) **if** (Clusters in $\mathcal{C}$ cover $\geq (1-1/e)|N|$ nodes of N)
  SUCCESS: $\mathcal{D}$ = MERGE$(\mathcal{D} \cup C)$
  else
  FAILURE: OPTLowerBound = $(1+\epsilon)$OPTLowerBound

(4) **return**$(\mathcal{D})$

MERGE$(\mathcal{X})$

Input: A set of clusters $\mathcal{X}$. (Some pairs of clusters in $\mathcal{X}$ may have nonempty intersection.)

Output: A set of pairwise disjoint clusters such that they cover all vertices in any cluster of $\mathcal{X}$ and the sum of the diameters of the new clusters is no greater than that of the input clusters.

(1) **while** $(\exists Y, Z \in \mathcal{X}$ such that $Y \neq Z$ and $Y \cap Z \neq \emptyset)$ **do**

- (a) NEW = $Y \cup Z$.
- (b) Add NEW to $\mathcal{X}$.
- (c) Delete X and Y from $\mathcal{X}$.

(2) **return**$(\mathcal{X})$

3.1 Algorithm Overview

We give a brief description of our approximation algorithm APPROX-CMSD$_\Delta$, and introduce the terminology used in the analysis. The algorithm begins with every vertex of V in its own, singleton cluster, and proceeds through a series of rounds to merge them. In each round, the algorithm will either decrease the number of clusters stored, or it will increase an estimate of the optimal cost for solving the instance.

The algorithm has been simplified from the version outlined in [11], and has improved running time and performance guarantee. It has also been generalized to allow for performance tuning: there is a tradeoff between the number of clusters returned and the cost (total diameter) of those clusters.

In addition to the graph $G(V, E)$ and the number of clusters k, algorithm APPROX-CMSD$_\Delta$ uses three positive-valued parameters. One of these parameters, denoted by x, determines the desired tradeoff between the number of clusters (inversely proportional to x) and their cost (directly proportional to x). The other two parameters, denoted by ϵ and δ, play a significant role in determining the running time of the algorithm. These two parameters should be fixed as small constants.

Let OPT(I) be the optimal solution value to the problem instance under consideration. APPROX-CMSD$_\Delta$ maintains a relaxed lower bound, denoted by OPTLowerBound, of OPT(I). The algorithm ensures that the value of OPTLowerBound is always bounded by $(1+\epsilon)$OPT(I). Initially, OPTLowerBound is set to the smallest nonzero edge weight in the complete graph (see Part 1 of Observation 1). At all times, APPROX-CMSD$_\Delta$ also maintains a set $\mathcal{D}$ of clusters which cover all vertices V from the problem instance. For that reason, we refer to the clusters in $\mathcal{D}$ as *global clusters*. The algorithm begins with $\mathcal{D}$ consisting of $|V|$ singleton clusters, and progresses through a series of iterations. At the end of each iteration, either the clusters in $\mathcal{D}$ are combined to decrease the number of clusters or OPTLowerBound is increased to more accurately estimate OPT(I).

To do this, during each iteration, APPROX-CMSD$_\Delta$ constructs a vertex set N by selecting an arbitrary vertex from each of the global clusters in $\mathcal{D}$. It converts the complete subgraph induced on N into a set cover instance using TRANSFORMTOSETCOVER, and calls the Budgeted Maximum Coverage Approximation Algorithm (APPROX-BMCP) for a solution. (The inputs to APPROX-BMCP are a collection of sets along with a nonnegative weight for each set and a budget on the total weight of the sets to be chosen. The output of APPROX-BMCP is a collection of sets whose total weight is at most the given budget.) If the collection of sets returned by APPROX-BMCP covers less than $(1-1/e)|N|$ of the vertices in N, then OPTLowerBound estimate is provably less than OPT(I) (see Lemma 5), so OPTLowerBound is increased by the factor $(1+\epsilon)$.

As mentioned above, some calls to APPROX-BMCP return vertex sets (clusters) which contain at least $(1-1/e)|N|$ of the vertices in N. We call these *merging clusters*, because they are used to merge global clusters from $\mathcal{D}$, thereby reducing the total number of clusters. This entire process is repeated until the number of clusters in $\mathcal{D}$ is at most $((1+\frac{2}{x})\frac{e}{e-1}+\delta)k$. In subsequent sections, it will be shown that the algorithm runs in polynomial time and that it provides the specified performance bounds.

A complete description of the approximation algorithm is given in Procedure APPROX-CMSD$_\Delta$. This version facilitates the presentation of proofs in Section 3.2. Some ideas for restructuring the algorithm so as to improve its running time in practice are discussed in Section 3.3.

3.2 Correctness of the Algorithm

Clearly, APPROX-CMSD$_\Delta$ will return only when the number of global clusters is at most $((1+\frac{2}{x})\frac{e}{e-1}+\delta)k$. We need to prove that it does terminate in polynomial time and that the clusters it returns have a total diameter of no more than $(1+\epsilon)(x+2)(\ln(\frac{n}{k\delta}))\mathrm{OPT}(I)$. First, we argue that during each iteration, the call to APPROX-BMCP returns at most $(1+\frac{2}{x})k$ clusters whose cost is at most $(x+2)(1+\epsilon)\mathrm{OPT}(I)$.

LEMMA 3. *Let $G(V,E)$ be the graph of a* CMSD$_\Delta$ *problem instance I, with optimal k cluster sum of diameters equal to* $\mathrm{OPT}(I)$. *In each iteration of* APPROX-CMSD$_\Delta$, *the subgraph $G(N,E')$ has a k cluster solution whose total diameter is at most* $\mathrm{OPT}(I)$.

PROOF. Note that $N \subseteq V$. Thus, any optimal clustering for $G(V,E)$ naturally induces a clustering on N with cost no more than $\mathrm{OPT}(I)$. If this produces fewer than k clusters, singleton clusters can be added. $\square$

LEMMA 4. *In* APPROX-CMSD$_\Delta$, *the call to* APPROX-BMCP *never returns more than $(1+\frac{2}{x})k$ clusters; that is, $|\mathcal{C}| \le (1+\frac{2}{x})k$ in every iteration.*

PROOF. The additive term $x \cdot$ OPTLowerBound passed to TRANSFORMTOSETCOVER ensures that each vertex set created has cost at least $x \cdot$ OPTLowerBound$/k$. Any more than $(1+\frac{2}{x})k$ of these sets will cost more than $(x+2)$OPTLowerBound, the budget given to APPROX-BMCP. Since APPROX-BMCP never returns clusters whose cost exceeds the allotted budget, the lemma follows. $\square$

LEMMA 5. *In* APPROX-CMSD$_\Delta$, *if* OPTLowerBound $\ge \mathrm{OPT}(I)$, *then* APPROX-BMCP *will return sets which cover at least $(1-1/e)|N|$ of the vertices in N.*

PROOF. By Lemma 2, the call to TRANSFORMTOSETCOVER returns a set cover instance whose optimal solution value is no more than $2\,\mathrm{OPT}(I)+x\cdot$ OPTLowerBound. When OPTLowerBound $\ge \mathrm{OPT}(I)$, the last expression is bounded by $(2+x)$OPTLowerBound. Therefore, by Theorem 1, the call to APPROX-BMCP with budget $(2+x)$OPTLowerBound will return sets which cover at least $(1-1/e)|N|$ of the vertices. $\square$

Thus, for any value of OPTLowerBound $\ge \mathrm{OPT}(I)$, the call to APPROX-BMCP with budget value set to $(2+x)$OPTLowerBound will return at most $(1+\frac{2}{x})k$ clusters. Further, these clusters will contain nodes from at least $(1-1/e)|\mathcal{D}|$ different global clusters of $\mathcal{D}$, because every node in N is from a separate cluster in $\mathcal{D}$.

Next, we show that the two primary variables of APPROX-CMSD$_\Delta$ behave as they should. First, OPTLowerBound must always be bounded by $(1+\epsilon)\mathrm{OPT}(I)$. Second, the clusters of $\mathcal{D}$ must always contain all points from V.

LEMMA 6. *In* APPROX-CMSD$_\Delta$*, the value of* OPTLowerBound *is always less than* $(1+\epsilon)\mathrm{OPT}(I)$.

PROOF. For a problem instance with a nonzero optimal solution, $\mathrm{OPT}(I)$ is at least the minimum nonzero edge weight in the graph (Part 1 of Observation 1). As the approximation algorithm initializes OPTLowerBound to the minimum nonzero edge weight, the lemma holds at the beginning of the algorithm. Subsequently, the only place where OPTLowerBound is increased is the line marked FAILURE, which is reached only when the call to APPROX-BMCP fails to cover enough vertices from N. By Lemma 5, this occurs only when OPTLowerBound $< \mathrm{OPT}(I)$. Thus, increasing OPTLowerBound by a factor of $(1+\epsilon)$ will not violate the lemma. □

LEMMA 7. *The clusters returned by the function* MERGE *contain all vertices from the clusters passed to the function.*

PROOF. Within MERGE, when a pair of clusters Y and Z is removed from the input set $\mathcal{X}$, it is replaced by the cluster $Y \cup Z$. Since the latter cluster contains all of the vertices in Y and Z, the lemma follows. □

LEMMA 8. *Within* APPROX-CMSD$_\Delta$*, the clusters of* $\mathcal{D}$ *always contain all vertices from* V.

PROOF. Initially, $\mathcal{D}$ contains each vertex of V as a singleton cluster. $\mathcal{D}$ only changes value in the line marked SUCCESS. At that point, clearly all of the clusters in $\mathcal{D}$ are passed to MERGE. By Lemma 7, MERGE will always return clusters containing all the nodes in the clusters of $\mathcal{D}$. □

Next, we need to bound the number of clusters returned by MERGE as well as the cost of these clusters. We have already shown that during any iteration within APPROX-CMSD$_\Delta$, the call to APPROX-BMCP returns at most $(1+\frac{2}{x})k$ clusters in $\mathcal{C}$, with cost no more than $(2+x)$OPTLowerBound. By Lemma 6, the last expression is at most $(2+x)(1+\epsilon)\mathrm{OPT}(I)$. Next, we show that when MERGE is called by the program, the number of clusters in $\mathcal{D}$ is reduced.

LEMMA 9. *When called from* APPROX-CMSD$_\Delta$, MERGE$(\mathcal{D} \cup C)$ *returns at most* $(1+\frac{2}{x})k + |\mathcal{D}|/e$ *clusters. Further, the sum of diameters of the returned clusters is no more than the sum of the diameters of the clusters in* $\mathcal{D}$ *and* $\mathcal{C}$.

PROOF. MERGE repeatedly combines a pair of intersecting clusters into a single cluster. By Lemma 1, this process will not increase the total diameter of the clusters. This proves the second part of the lemma.

To prove the first part, note that by Lemma 4, $\mathcal{C}$ has at most $(1+\frac{2}{x})k$ clusters. Since MERGE is called in this iteration of APPROX-CMSD$_\Delta$, the clusters in $\mathcal{C}$ must intersect all but at most $|\mathcal{D}|/e$ of the clusters from $\mathcal{D}$. (Otherwise, this iteration of Approx-Cmsd$_\Delta$ will reach the statement labeled FAILURE rather than the one labeled SUCCESS, and the Merge function would not have been called.) Any cluster $Y \in \mathcal{D}$ which intersects a cluster $Z \in \mathcal{C}$ will eventually be merged with Z or with a superset cluster which replaced Z. Thus, at least $(1 - 1/e)|\mathcal{D}|$ times, two clusters from $\mathcal{C} \cup D$ will be replaced by one, leaving at most $|C| + |D| - (1 - 1/e)|\mathcal{D}| \leq (1 + \frac{2}{x})k + |\mathcal{D}|/e$ clusters. □

In Approx-Cmsd$_\Delta$, just after initialization, the clusters of $\mathcal{D}$ have cost zero. Their cost may increase only during each call to Merge. So, the final sum of diameters of the clusters in $\mathcal{D}$ is bounded by the sum of the diameter costs of clusters $\mathcal{C}$ generated by the calls to Merge. For every iteration, we have already shown that Approx-Bmcp returns merging clusters $\mathcal{C}$ with sum of diameters bounded by $(2+x)(1+\epsilon)\mathrm{OPT}(I)$. To bound the total cost of the global clusters returned by Approx-Cmsd$_\Delta$, we need only to bound the number of times the statement labeled SUCCESS is reached during the execution of the algorithm.

Lemma 10. *In* Approx-Cmsd$_\Delta$, *the statement labeled SUCCESS is reached at most* $\ln(\frac{|V|}{k\delta})$ *times before the algorithm terminates.*

Proof. Let d_i be the number of global clusters in $\mathcal{D}$ when the statement labeled SUCCESS is reached for the i^{th} time. We begin with $d_0 = |V|$. Each time SUCCESS is reached, the global clusters $\mathcal{D}$ are replaced by at most $(1 + \frac{2}{x})k + |\mathcal{D}|/e$ new global clusters by Merge (Lemma 9). Thus, we have the recurrence $d_i \leq (1 + \frac{2}{x})k + d_{i-1}/e$ for $i \geq 1$. Unrolling this recurrence, we get $d_i \leq (1 + \frac{2}{x})(\sum_{j=0}^{i-1} e^{-j})k + d_0/e^i$. Letting $i = \ln(|V|/k)$, we get $d_{\ln(|V|/k)} \leq (1 + \frac{2}{x})(\sum_{j=0}^{\infty} e^{-j})k + kd_0/|V| = (1 + \frac{2}{x})\frac{e}{e-1}k + k$. After i *additional* iterations, $d_{\ln(|V|/k)+i} \leq (1 + \frac{2}{x})\frac{e}{e-1}k + k/e^i$, so $\ln(1/\delta)$ additional iterations in which SUCCESS is reached will lower the number of global clusters to the required number, namely $((1 + \frac{2}{x})\frac{e}{e-1} + \delta)k$. The lemma follows. □

In view of Lemma 10, it follows that the total cost of the solution returned by Approx-Cmsd$_\Delta$ is at most $(1 + \epsilon)(x + 2)(\ln(\frac{n}{k\delta}))\mathrm{OPT}(I)$ for n total vertices.

To get an upper bound on the running time of the algorithm, we also need to bound the number of times the statement labeled FAILURE is reached in the execution of Approx-Cmsd$_\Delta$.

Lemma 11. *During the execution of* Approx-Cmsd$_\Delta$, *the number of times the statement labeled FAILURE can be reached is* $O(\log_{1+\epsilon} \frac{\max}{\min})$, *where* $\min$ *is the smallest nonzero edge weight and* $\max$ *is the largest edge weight in the graph.*

PROOF. By inspection, the algorithm never decreases the value of OPTLowerBound. Each time the statement labeled FAILURE is reached, OPTLowerBound is increased by the factor $1+\epsilon$. Thus, after reaching the FAILURE statement $\Theta(\log_{1+\epsilon} \frac{\mathrm{OPT}(I)}{\min})$ times, OPTLowerBound $\geq$ OPT(I). By Lemma 5, once OPTLowerBound $\geq$ OPT(I), each call to APPROX-BMCP will result in reaching the statement labeled SUCCESS rather than the one labeled FAILURE. Since OPT$(I) \leq \max$ (Part 2 of Observation 1), the number of times FAILURE is reached is at most $O(\log_{1+\epsilon} \frac{\max}{\min})$. □

We are now ready to argue that APPROX-CMSD$_\Delta$ runs in polynomial time. Let $T(r)$ denote the running time of APPROX-BMCP. From Lemmas 10 and 11, the number of iterations of APPROX-CMSD$_\Delta$ is $O(\log_{1+\epsilon} \frac{\max}{\min} + \ln(\frac{n}{k\delta}))$ with no large hidden constants. For each iteration, the running time can be seen to be $O(n^2 \log n + T(n^2))$, where the two terms represent the running times of TRANSFORMTOSETCOVER and APPROX-BMCP respectively. Because MERGE is called with at most $O(n)$ clusters of nodes from the set of n, it can be implemented to run in $O(n^2)$ time. So, the running time of MERGE is dominated by that of TRANSFORMTOSETCOVER. Since $T(r)$ is polynomial by [24], so is our algorithm.

THEOREM 2. *For any fixed positive parameters x, ϵ and δ, there is a polynomial time approximation algorithm for the* CMSD$_\Delta$ *problem which returns at most $((1+\frac{2}{x})\frac{e}{e-1}+\delta)k$ clusters whose total diameter is at most $(1+\epsilon)(x+2)(\ln(\frac{n}{k\delta}))$ times the optimal solution value with k clusters.* □

3.3 Additional Observations Concerning the Algorithm

By setting different values of x, we tradeoff cost for number of clusters. For instance, we can limit the number of clusters to at most $2k$ by setting $x > 2e/(e-2)$. For example, choosing $\epsilon = \delta = .01$ and $x = 7.9$ $[= 8 - 10\epsilon] > 2e/(e-2)$, we get:

COROLLARY 1. *There is a polynomial time approximation algorithm for the* CMSD$_\Delta$ *problem which returns at most $2k$ clusters whose total diameter is at most $[O(1) + 10\ln(n/k)]$ times the optimal solution value with k clusters.* □

For very large values of x, the number of clusters returned by our method approaches $k(1+\delta)e/(e-1) \approx 1.6k$. Our method does not seem to allow a reduction below the factor of 1.6 for the number of clusters while guaranteeing a performance bound on the total diameter. We note that there is an easy way to find k clusters with a total diameter of at most $2k \cdot \mathrm{OPT}(I)$: Use the heuristic due to Gonzalez [16] (or the one due to Hochbaum and Shmoys [21]) to obtain a 2-approximation for the problem of minimizing the maximum cluster diameter. It is easy to see that this gives a factor $2k$ cost approximation for the problem with exactly k clusters. For $k < \ln(n/k)$ this approach may provide a better solution than APPROX-CMSD$_\Delta$.

The outline of APPROX-CMSD$_\Delta$ can be restructured to reduce its running time in practice. First, when the statement labeled FAILURE is reached, there is no need to execute Steps 3(a) and 3(b) of the algorithm since the vertex set N does not change. We need to execute only Steps 3(c), 3(d) and 3(e). Further, in Step 3(c), the sets produced by the transformation do not change; the only change is that the cost of each set is increased by the amount (ϵxOPTLowerBound$/k$). It is straightforward to incorporate these modifications into the outline of Procedure APPROX-CMSD$_\Delta$.

3.4 An Approximation Algorithm for CMSD$_\Delta$ *with Fixed k*

When k is fixed, it is possible to obtain a simple 2-approximation algorithm for the CMSD$_\Delta$ problem using the transformation TRANSFORMTOSETCOVER. We present this result below.

THEOREM 3. *When the number of clusters k is* fixed, *there is a 2-approximation algorithm for* CMSD$_\Delta$.

PROOF. The steps of the approximation algorithm are as follows.

(1) Using the transformation TRANSFORMTOSETCOVER, construct an instance of the minimum cost set cover problem from the given instance of CMSD$_\Delta$ with the parameter f set to zero.

(2) Find a minimum cost set cover consisting of at most k sets. Since k is fixed, this step can be done in polynomial time by exhaustive search.

(3) If the collection of sets obtained in Step 2 are not pairwise disjoint, then repeatedly merge pairs of sets with nonempty intersection until the collection is pairwise disjoint.

(4) Output the collection of sets found in Step 3 as the solution to the CMSD$_\Delta$ instance.

Clearly, the approximation algorithm runs in polynomial time. Applying Lemma 2 with $f = 0$, the cost of an optimal set cover is at most twice the optimal solution value of the CMSD$_\Delta$ instance. Step 2 finds an optimal solution to the set cover problem, and by Lemma 1, the merging operations in Step 3 do not increase the total diameter of the clusters. Thus, the total diameter of the clusters produced is at most twice the optimal value. □

4. Non-Approximability Results

4.1 Non-Approximability Without Triangle Inequality

We show that, unless $\mathsf{P} = \mathsf{NP}$, CMSD cannot be efficiently approximated to within any factor even when the number of clusters is fixed at 3. We establish this result through a simple modification to the reduction from

Graph 3-colorability (3-COLOR) given in [5] to establish the NP-hardness of CMSD.

PROPOSITION 1. *Unless* $\mathsf{P} = \mathsf{NP}$, *for any* $\rho \geq 1$, *no polynomial time algorithm for the* CMSD *problem can provide a performance guarantee of* ρ, *even when the number of clusters is fixed at 3.*

PROOF. Suppose for the sake of contradiction that for some $\rho \geq 1$, there is a polynomial time approximation algorithm $\mathcal{A}$ that provides a performance guarantee of ρ for CMSD. We will show that $\mathcal{A}$ can be used to solve the 3-COLOR problem in polynomial time, contradicting the assumption that $\mathsf{P} \neq \mathsf{NP}$.

Let $G(V, E)$ denote the undirected graph which represents an arbitrary instance of 3-COLOR. We construct an instance of the CMSD problem (without triangle inequality), consisting of a complete edge weighted graph G' on the vertex set V as follows. For any pair of vertices u and v, the weight of $\{u, v\}$ is set to $3\rho + 1$ if $\{u, v\}$ is an edge in E and to 1 otherwise. The number of clusters is set to 3. It is easy to see that if G is 3-colorable, then the optimal solution value to the CMSD instance is at most 3. Using this fact, it straightforward to verify that when $\mathcal{A}$ is executed on G', the total diameter of the 3 clusters returned by $\mathcal{A}$ is at most 3ρ if and only if G is 3-colorable. $\square$

This non-approximability result should be contrasted with the known result that the CMSD problem is solvable in polynomial time for 2 clusters [18], [28].

4.2 A Non-Approximability Result for CMSD_Δ

PROPOSITION 2. *Unless* $\mathsf{P} = \mathsf{NP}$, *for any* $\epsilon > 0$, *no polynomial time algorithm for the* CMSD_Δ *problem can provide a solution which satisfies the bound on the number of clusters and whose total diameter is within a factor* $2 - \epsilon$ *of the optimal value.*

PROOF. We use a reduction from the CLIQUE problem [15]. Let the undirected graph $G(V, E)$ and integer $J \leq |V|$ denote an arbitrary instance of the CLIQUE problem. We construct an instance of the CMSD_Δ problem consisting of a complete edge weighted graph G' on the vertex set V as follows. For any pair of vertices u and v, the weight of $\{u, v\}$ is set to 1 if $\{u, v\}$ is an edge in E and to 2 otherwise. Obviously, the resulting edge weights satisfy the triangle inequality. The number of clusters k is set to $|V| - J + 1$. Now, it is straightforward to see that if G has a clique with J or more vertices, then G' can be partitioned into at most k clusters with a total diameter of 1: the vertices of the clique form one cluster of diameter 1 and each of the remaining $|V| - J$ vertices forms a separate cluster with a diameter of zero. Further, if G does not have a clique with J or more vertices, then any solution with at most k clusters must have a total diameter of at least 2. The proposition follows. $\square$

5. Minimizing the Sum of Cluster Radii

In this section, we consider a closely related clustering problem where the goal is to minimize the sum of the radii of the clusters (rather than the sum of the diameters of clusters). To discuss these results, we first recall the definition of cluster radius.

DEFINITION 1. *Let C be a cluster. For any node v in C, let d_v denote the maximum distance between v and any other node in C. The radius of C is given by* $\min\{d_v : v \in C\}$. *A node v for which d_v is equal to the radius of C is a* **center** *of C.*

We use CMSR to denote the clustering problem where the objective is to minimize the sum of the cluster radii. Instances of CMSR where the edge weights satisfy the triangle inequality are denoted by CMSR_Δ.

When edge weights satisfy the triangle inequality, it is easy to see that the diameter of a cluster is at most twice the radius. Therefore, our approximation result for CMSD_Δ (Section 3) carries over to CMSR_Δ with different constant terms. (In the bounds for both the number of clusters and their cost, changing the 2 to a 1 gives the bounds for CMSR_Δ.) The non-metric instances of CMSD and CMSR exhibit an interesting difference when the number of clusters k is fixed. As shown in Section 4, CMSD is NP-hard even for $k = 3$. However, modifying the algorithm of Subsection 3.4, we can give an optimal solution to CMSR in polynomial time. The only differences are the costs assigned to each set in the transformation to a set cover problem, and our method for insuring that the clusters are disjoint.

The steps are as follows

(1) Use TRANSFORMTOSETCOVER to change the problem into a set cover problem, but within TRANSFORMTOSETCOVER, assign costs to each set by the radius from the vertex v currently being considered, instead of the diameter of the cluster. Variable $f = 0$. Thus, from each vertex v, there are $|V|$ possible sets, of the different radius values possible from v, and $|V|^2$ total possible sets.

(2) Find a minimal cost set cover consisting of at most k sets by exhaustive search.

(3) For any vertex occuring in more than one set, arbitrarily delete it from one of the sets (unless it is the center of one of the sets, in which case delete it from the *other* set). Repeat this until all sets are disjoint.

(4) Output the final sets as clusters.

Every center and radius value of a cluster in the optimal solution will be selected as a center and radius value in the first step, and these will clearly give a solution to the set cover problem. Deleting the duplicate vertices will not violate that the solution is still a set cover, nor increase the cost of any set. (In an optimal solution to the set cover problem created above, no vertex will be the center of more than one non-singleton set. In the case of a singleton set being chosen, and that point also being the center of another set from the solution, eliminate the singleton set.) So for fixed values of k, without triangle inequality, CMSR is easier than CMSD (unless $\mathsf{P} \neq \mathsf{NP}$).

When edge weights satisfy the triangle inequality, it is easy to see that the diameter of a cluster is at most twice the radius. Using this observation, it follows that for fixed k, an optimal solution to the CMSR problem gives 2-approximation for the CMSD_Δ problem. On the other hand, the result implied by Theorem 3 yields an algorithm with a slightly better running time for $k \geq 3$.

6. Concluding Remarks

We considered the clustering problem where the goal is to minimize the sum of the diameters of the clusters. We presented approximation algorithms and non-approximability results. In particular, our approximation algorithm for CMSD_Δ allows a tradeoff between the number of clusters returned and the sum of the diameters of the clusters.

Charikar and Panigrahy [8] have informed us that they have recently developed algorithms with greatly improved approximation bounds for the CMSD_Δ problem: they produce exactly k clusters while only allowing a constant factor increase in cost. They also have a simple greedy algorithm with logarithmic cost increase, and some incremental clustering results.

A promising research direction is to consider the geometric version of CMSD_Δ. As indicated in Section 1.2, this problem is known to be solvable in polynomial time for *fixed k* [6]. However, the complexity of this problem for variable k is open. If the problem is NP-hard, it is also of interest to investigate whether geometric techniques can be used to devise heuristics that provide better performance guarantees.

Acknowledgements

It is our pleasure to thank Professor R. Sarnath (Department of Computer Science, St. Cloud State University, St. Cloud, MN) for his valuable comments. We also thank Gabriel Istrate and Don Hush for several useful conversations during the course of writing the paper. The research was supported by Department of Energy under Contract W-7405-ENG-36 and the National Science Foundation under Grant CCR-97-34936. This work was carried out when the last three authors were visiting the Los Alamos National Laboratory.

References

[1] P. K. Agarwal and C. M. Procopiuc, Exact and Approximate Algorithms for Clustering. *Proc. 9th ACM-SIAM Symposium on Discrete Algorithms* (SODA'98), San Francisco, CA, Jan. 1998, pp. 658–667.

[2] P. K. Agarwal and C. M. Procopiuc, Approximation Algorithms for Projective Clustering. *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms* (SODA'2000), San Francisco, CA, Jan. 2000, pp. 538–547.

[3] B. Awerbuch, B. Berger, L. Cowen and D. Peleg, Near-Linear Time Construction of Sparse Neighborhood Covers. *SIAM J. Computing*, Vol. 28, No. 1, 1998, pp. 263–277.

[4] V. Batagelj, S. Korenjak-Cerne and S. Klavzar, Dynamic Programming and Convex Clustering. *Algorithmica*, Vol. 11, No. 2, Feb. 1994, pp. 93–103.

[5] P. Brucker, On the Complexity of Clustering Problems, in *Optimization and Operations Research*, Lecture Notes in Economics and Mathematical Systems, Vol. 157, Edited by M. Beckmann and H. Kunzi, Springer-Verlag, Heidelberg, 1978, pp. 45–54.

[6] V. Capoyleas, G. Rote and G. Woeginger, Geometric Clusterings. *J. Algorithms*, Vol. 12, No. 2, Jun. 1991, pp. 341–356.

[7] M. Charikar, C. Chekuri, T. Feder and R. Motwani, Incremental Clustering and Dynamic Information Retrieval. *Proc. 29th Annual ACM Symposium on Theory of Computing* (STOC'97), El Paso, TX, May 1997, pp. 626–634.

[8] M. Charikar and R. Panigrahy, Personal Communication, Aug. 2000.

[9] A. Datta, Efficient Parallel Algorithms for Geometric k-Clustering Problems. *Proc. 11th Annual Symposium on Theoretical Aspects of Computer Science* (STACS'94), Caen, France, Feb. 1994, Springer-Verlag Lecture Notes in Computer Science, Vol. 775, pp. 475–486.

[10] J. S. Deogan, D. Kratsch and G. Steiner, An Approximation Algorithm for Clustering Graphs with a Dominating Diametral Path. *Information Processing Letters* , Vol. 61, No. 3, Feb. 1997, pp. 121–127.

[11] S. R. Doddi, M. V. Marathe, S. S. Ravi, D. S. Taylor and P. Widmayer, Approximation Algorithms for Clustering to Minimize the Sum of Diameters. *Proc. 7th Scandinavian Workshop on Algorithm Theory* (SWAT 2000), Bergen, Norway, July 2000, pp. 237–250.

[12] R. Duda and P. Hart, *Pattern Classification and Scene Analysis* , Wiley-Interscience, New York, NY, 1973.

[13] T. Feder and D. H. Greene, Optimal Algorithms for Approximate Clustering. *Proc. 20th Annual ACM Symposium on Theory of Computing* (STOC'88), Chicago, IL, May 1988, pp. 434–444.

[14] R. Fowler, M. Paterson and S. Tanimoto, Optimal Packing and Covering in the Plane. *Information Processing Letters* , Vol. 12, 1981, pp. 133–137.

[15] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness* , W. H. Freeman and Co., San Francisco, CA, 1979.

[16] T. F. Gonzalez, Clustering to Minimize the Maximum Intercluster Distance. *Theoretical Computer Science* , Vol. 38, No. 2-3, Jun. 1985, pp. 293–306.

[17] N. Guttmann-Beck and R. Hassin, Approximation Algorithms for Min-sum p-Clustering. *Discrete Applied Mathematics* , Vol. 89, 1998, pp. 125–142.

[18] P. Hansen and B. Jaumard, Minimum Sum of Diameters Clustering. *Journal of Classification* , Vol. 4, 1987, pp. 215–226.

[19] P. Hansen and B. Jaumard, Cluster Analysis and Mathematical Programming. *Mathematical Programming* , Vol. 79, Aug. 1997, pp. 191–215.

[20] D. S. Hochbaum (Editor), *Approximation Algorithms for NP-Hard Problems* , PWS Publishing Company, Boston, MA, 1997.

[21] D. S. Hochbaum and D. B. Shmoys, A Unified Approach to Approximation Algorithms for Bottleneck Problems. *J. ACM* , Vol. 33, No. 3, July 1986, pp. 533–550.

[22] P. Indyk, A sublinear time approximation scheme for Clustering in Metric Spaces. *Proc. 40th IEEE Symposium Foundations of Computer Science,* (FOCS'99), New York, NY, Oct. 1999, pp. 154–159.

[23] A. Jain and R. Dubes, *Algorithms for Clustering Data* , Prentice-Hall, Inc., Englewood Cliffs, NJ, 1988.

[24] S. Khuller, A. Moss and J. Naor, The Budgeted Maximum Coverage Problem. *Information Processing Letters*, Vol. 70, 1999, pp. 39–45.

[25] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz and H. B. Hunt III, Bicriteria Network Design Problems. *J. Algorithms* , Vol. 28, No. 1, July 1998, pp. 142–171.

[26] J. Matousek, On Approximate Geometric k-Clustering. Manuscript, Department of Applied Mathematics, Charles University, Prague, Czech Republic, 1999.

[27] N. Megiddo and K. J. Supowit, On the complexity of some common geometric location problems. *SIAM J. Computing* , Vol. 13, 1984, pp. 182–196.

[28] C. L. Monma and S. Suri, Partitioning Points and Graphs to Minimize the Maximum or the Sum of Diameters. *Proc. 6th International Conference on Theory and Applications of Graphs*, Vol. 2, (Wiley) 1991, pp. 899–912.

[29] U. Pferschy, R. Rudolf and G. J. Woeginger, Some Geometric Clustering Problems. *Nordic J. Computing*, Vol. 1, No. 2, Summer 1994, pp. 246–263.

[30] J. Plesník, Complexity of Decomposing Graphs into Factors with Given Diameters or Radii. *Math. Slovaca*, Vol. 32, No. 4, 1982, pp. 379–388.

[31] P. Raghavan, Information Retrieval Algorithms: A Survey. *Proc. 8th ACM-SIAM Symposium on Discrete Algorithms* (SODA'97), New Orleans, LA, Jan. 1997, pp. 11–18.

[32] T. Zhang, R. Ramakrishnan and M. Livny, Birch: An Efficient Data Clustering Method for Very Large Databases. *Proc. ACM-SIGMOD International Conference on Management of Data* (SIGMOD'96), Aug. 1996, pp. 103–114.