

# Joint Cluster Analysis of Attribute Data and Relationship Data: the Connected $k$ -Center Problem

Martin Ester, Rong Ge, Byron J. Gao, Zengjian Hu, Boaz Ben-Moshe  
School of Computing Science, Simon Fraser University, Canada, V5A 1S6  
{ester, rge, bgao, zhu, benmoshe}@cs.sfu.ca

## Abstract

Attribute data and relationship data are two principle types of data, representing the intrinsic and extrinsic properties of entities. While attribute data has been the main source of data for cluster analysis, relationship data such as social networks or metabolic networks are becoming increasingly available. It is also common to observe both data types carry orthogonal information such as in market segmentation and community identification, which calls for a joint cluster analysis of both data types so as to achieve more accurate results. For this purpose, we introduce the novel Connected  $k$ -Center problem, taking into account attribute data as well as relationship data. We analyze the complexity of this problem and prove its NP-completeness. We also present a constant factor approximation algorithm, based on which we further design NetScan, a heuristic algorithm that is efficient for large, real databases. Our experimental evaluation demonstrates the meaningfulness and accuracy of the NetScan results.

## 1 Introduction

Entities can be described by two principal types of data: attribute data and relationship data. Attribute data describe intrinsic characteristics of entities whereas relationship data represent extrinsic influences among entities. By entities we mean corporeal or intangible objects of study interest that have distinctions, such as humans, organizations, products, or events. An entity holds endogenous properties and at the same time bears relations to other entities.

While attribute data continue to be the standard and dominant data source in data analysis applications, more and more relationship data are becoming available due to computerized automation of data probing, collection, and compiling procedures. Among them, to name a few, are acquaintance and collaboration networks as social networks, and ecological, neural and metabolic networks as biological networks. Consequently, network analysis [26, 23, 27] has been gaining popularity in the study of marketing, community identification, epidemi-

ology, molecular biology and so on.

The two types of data, attribute data and relationship data, can be more or less related. A certain relation between entity  $A$  and  $B$  may imply some common attributes they share; on the other hand, similar attributes of  $A$  and  $B$  may suggest a relation of some kind between them with high probability. If the dependency between attribute data and relationship data is high enough such that one can be soundly deduced from or closely approximated by the other, a separate analysis on either is sufficient. For example, the classical facility location problem [24] only manipulates locations (attributes) since the Euclidean distance between  $A$  and  $B$  can be used to well approximate their reachability via route connections (relationships).

On the other hand, often relationship data contains information that is independent from the attributes of entities. For example, two persons may share many characteristics in common but they never get to know each other; or in contrast, even with totally different demographics, they may happen to become good acquaintances. Due to rapid technological advances, the mobility and communication of humans have tremendously improved. As a consequence, the formation of social networks is slipping the leash of confining attributes. The small world phenomenon, as a consensus observation, has received voluminous cross-discipline attention [19].

The unprecedented availability of relationship data carrying important additional information beyond attribute data calls for joint analysis of both. Cluster analysis, one of the major tools in exploratory data analysis, has been investigated for decades in multiple disciplines such as statistics, machine learning, algorithm, and data mining. Varied clustering problems have been studied driven by numerous applications including pattern recognition, information retrieval, market segmentation and gene expression profile analysis, and emerging applications continue to inspire novel cluster models with new algorithmic challenges. The task of clustering is to group entities into clusters that exhibit internal co-

hesion and external isolation. Given both attribute data and relationship data, it is intuitive to require clusters to be cohesive (within clusters) and distinctive (between clusters) in both ways.

As a new clustering model taking into account attribute and relationship data, we introduce and study the Connected  $k$ -Center ( $CkC$ ) problem; i.e., the  $k$ -Center problem with the constraint of internal connectedness. The internal connectedness constraint requires that any two entities in a cluster are connected by an internal path, i.e., a path via entities only from the same cluster. The  $k$ -Center problem, as a classical clustering problem, has been intensively studied in the algorithms community from a theoretical perspective. The problem is to determine  $k$  cluster heads (centers) such that the maximum distance of any entity to its closest cluster head, the radius of the cluster, is minimized.

**Motivating applications:** The  $CkC$  problem can be motivated by market segmentation, community identification, and many other applications such as document clustering, epidemic control, and gene expression profile analysis. In the following, we further discuss the first two applications.

Market segmentation divides a market into distinct customer groups with homogeneous needs, such that firms can target groups effectively and allocate resources efficiently, as customers in the same segment are likely to respond similarly to a given marketing strategy. Traditional segmentation methods were based on attribute data only such as demographics (age, sex, ethnicity, income, education, religion and etc.) and psychographic profiles (lifestyle, personality, motives and etc.). Recently, social networks have become more and more important in marketing [16]. Ideas and behaviors are contagious. The relations in networks are channels and conduits through which resources flow [16]. Customers can hardly hear companies but they listen to their friends; customers are skeptical but they trust their friends [27]. By word-of-mouth propagation, a group of customers with similar attributes have much more chances to become like-minded. Depending on the nature of the market, social relations can even become vital in forming segments, and purchasing intentions or decisions may rely on customer-to-customer contacts to diffuse throughout a segment, for example, for cautious clients of risky cosmetic surgery or parsimonious purchasers of complicated scientific instruments. The  $CkC$  problem naturally models such scenarios: a customer is assigned to a market segment only if he has similar purchasing preferences (attributes) to the segment representative (cluster center) and can be reached by propagation from customers of similar interest in the segment.

Community identification is one of the major social network analysis tasks, and graph-based clustering methods have been the standard tool for this purpose [26]. In this application, clustering has generally been performed on relationship (network) data solely. Yet it is intuitive [23, 13] that attribute data can impact community formation in a significant manner. For example, given a scientific collaboration network, scientists can be separated into different research communities such that community members are not only connected (e.g., by co-author relationships) but also share similar research interests. Such information on research interests can be automatically extracted from homepages and used as attribute data for the  $CkC$  problem. As a natural assumption, a community should be at least internally connected with possibly more constraints on the intensity of connectivity. Note that most graph-based clustering methods used for community identification in network analysis also return some connected components.

**Contributions and outline:** This paper makes the following contributions:

1. We advocate joint cluster analysis of attribute data and relationship data, introduce the novel  $CkC$  clustering problem.
2. We analyze the complexity of the  $CkC$  clustering problem, prove its NP-completeness and present a corresponding 3-approximation algorithm.
3. Based on principles derived from the approximation algorithm, we propose a heuristic algorithm NetScan that efficiently computes a “good” clustering solution.
4. We report results of our experimental evaluation, demonstrating the meaningfulness of the clustering results and the scalability of the NetScan algorithm.

The rest of the paper is organized as follows. Related work is reviewed in Section 2. Section 3 introduces the  $CkC$  clustering problem and analyzes its complexity. In Section 4, we present an approximation algorithm for the proposed clustering problem. To provide more scalability, we also present an efficient heuristic algorithm in Section 5. We report experimental results in Section 6 and conclude the paper in Section 7.

## 2 Related Work

Theoretical approaches to cluster analysis usually formulate clustering as optimization problems, for which rigorous complexity studies are performed and polynomial approximation algorithms are provided. Depending on the optimization objective, many clustering problems and their variants have been investigated, such as the  $k$ -center and  $k$ -median problems for facility lo-

cation [24], the min-diameter problem [7], and so on. In the areas of statistics, machine learning, and data mining, clustering research emphasizes more on real life applications and development of efficient and scalable algorithms. Clustering algorithms can be roughly categorized [12] into partitioning methods such as  $k$ -means [20],  $k$ -medoids [19], and CLARANS [22], hierarchical methods such as AGNES and DIANA [19], and density-based methods such as DBSCAN [10] and OPTICS [1]. The above methods generally take into account only attribute data.

We also summarize some theoretical results on the  $k$ -center problem that are related to our theoretical analysis. It is well known that both the  $k$ -center and Euclidean  $k$ -center problem is NP-Complete for  $d \geq 2$  when  $k$  is part of the input [21]. Besides, in the case of  $d = 1$ , the Euclidean  $k$ -center problem is polynomially solvable using dynamic programming techniques. Meanwhile, when  $d \geq 2$ , if  $k$  is treated as a constant, the  $k$ -center problem can also be easily solved by enumerating all the  $k$  centers. However, as we will see in section 3, the  $CkC$  problem remains NP-Complete even for  $k = 2$  and  $d = 1$ . Hence in this sense, the  $CkC$  problem is harder than Euclidean  $k$ -center.

Recently, the increasing availability of relationship data stimulated research on network analysis [26, 23, 13]. Clustering methods for network analysis are mostly graph-based, separating sparsely connected dense sub-graphs from each other as seen in [6]. A good graph clustering should exhibit few between-cluster edges and many within-cluster edges. Graph clustering methods can be applied to data that is originally network data as well as to similarity graphs representing a similarity matrix, i.e., derived from the original attribute data. A similarity graph can be a complete graph as in agglomerative hierarchical clustering algorithms of single-link, complete link, or average link [17]; or incomplete retaining those edges whose corresponding similarity is above a threshold [11, 14]. CHAMELEON [18] generates edges between a vertex and its  $k$  nearest neighbors, which can be considered as relative thresholding. Note that none of the above methods simultaneously considers attribute and relationship data that represent independent information.

Finally, our research is also related to the emerging areas of constraint-based clustering and semi-supervised clustering. Early research in this direction allowed the user to guide the clustering algorithm by constraining cluster properties such as size or aggregate attribute values [25]. More recently, several frameworks have been introduced that represent available domain knowledge in the form of pairwise “must-links” and “cannot-links”. Objects connected by a must-link are supposed to be

long to the same cluster; those with a cannot-link should be assigned to different clusters. [3] proposes a probabilistic framework based on Hidden Markov Random Fields, incorporating supervision into  $k$ -clustering algorithms. [8] also considers additional minimum separation and minimum connectivity constraints, but they are ultimately translated into must-link and cannot-link constraints. A  $k$ -means like algorithm is presented using a novel distance function which penalizes violations of both kinds of constraints. The above two papers are similar to our research in the sense that they also adopt a  $k$ -clustering (partitioning) approach under the framework of constraint-based clustering. Nevertheless, in semi-supervised clustering, links represent specific constraints on attribute data. They are provided by the user to capture some background knowledge. In our study, links represent relationship data. They are not constraints themselves, but data on which different constraints can be enforced; e.g., “internally connected” as in this paper.

### 3 Problem Definition and Complexity Analysis

In this section, we introduce the Connected  $k$ -Center problem and analyze its complexity.

**3.1 Preliminaries and problem definition.** Attribute data can be represented as an  $n \times m$  entity-attribute matrix. Based on a chosen similarity measure, pairwise similarities can be calculated to obtain an  $n \times n$  entity-entity similarity matrix. Sometimes, similarity matrixes are transformed into graphic representations, called similarity graphs, with entities as vertices and pairwise similarities, possibly thresholded, as edge weights. Relationship data are usually modeled by networks comprised of nodes and links, which we call entity networks. In this paper, we concentrate on symmetric binary relations, thereby entity networks can be naturally represented as simple graphs with edges (links) as dichotomous variables indicating the presence or absence of a relation of interest such as acquaintance, collaboration, or transmission of information or diseases. Entity networks can as well be represented by adjacency matrices or incident matrices as graphs in general.

Nodes in an entity network do not have meaningful locations. With attribute data available, attributes for each entity can be represented as a coordinate vector and assigned to the corresponding node, resulting in what we call an “informative graph”. Informative graphs, with both attribute data and relationship data embedded, are used as input for our Connected  $k$ -Center problem to perform joint cluster analysis.

For low dimensional (attribute) data, informative graphs can be directly visualized. For higher dimen-

sional data, multidimensional scaling techniques can be invoked to provide such a graphical visualization. These visualizations can intuitively demonstrate the dependency and orthogonality of attribute data and relationship data. In the former case, the visualizations exhibit relatively short edges only, as shown in Figure 1 (a); in the latter case, the visualizations exhibit randomness in terms of edge length, as shown in Figure 1 (b).

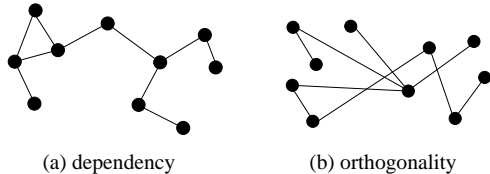


Figure 1: Attribute data and relationship data.

In this paper, the terms “vertex” and “node” are used interchangeably, so are “edge” and “link”. In the following sections, “graph” will refer to “informative graph” since we always consider two data types simultaneously.

Now we introduce the decision version of the  $CkC$  (Connected  $k$ -Center) problem, for constant  $k \geq 2$  and  $d \geq 1$ , where  $d$  is the dimensionality.

**DEFINITION 3.1. ( $CkC$  problem)** Given  $k$  as the number of centers, a radius constraint  $r \in \mathcal{R}^+$ , a distance function  $\|\cdot\|$  and a graph  $G = (V, E)$  where every node in  $V$  is associated with a coordinate vector  $w : V \rightarrow \mathcal{R}^d$ . Decide whether there exist disjoint partitions  $\{V_1, \dots, V_k\}$  of  $V$ , i. e.,  $V = V_1 \cup \dots \cup V_k$  and  $\forall 1 \leq i < j \leq k, V_i \cap V_j = \phi$ , which satisfy the following two conditions:

1. The induced subgraphs  $G[V_1], \dots, G[V_k]$  are connected. (internal connectedness constraint)
2.  $\forall 1 \leq i \leq k$ , there exists a center node  $c_i \in V_i$ , such that  $\forall v \in V_i, \|w(v) - w(c_i)\| \leq r$ . (radius constraint)

Intuitively, the problem is to check whether the input graph can be divided into  $k$  connected components, such that every component is a cluster with radius less than or equal to  $r$ , i. e., in each cluster, there exists a center node  $c$  and all the remaining nodes are within distance  $r$  to  $c$ .

We assume the given graph is connected which is reasonable for many application scenarios, such as social networks which are normally considered to be connected. Even if the entire graph is not connected, the problem can be applied to its different connected components.

**3.2 Complexity analysis.** Due to the similarity of the  $CkC$  problem to the traditional  $k$ -center problem, it is natural to ask the following question. *How much has the traditional  $k$ -center problem been changed in terms of hardness after attaching a constraint of internal connectedness?* To answer this question, we analyze the complexity of the  $CkC$  problem.

The formal analysis is rather technical, we precede it with an intuitive explanation. We say a solution (or partitioning) is *legal*, if all the  $k$  partitions (or clusters) are disjoint and the corresponding induced subgraphs are connected. Since  $k$  is fixed as a constant, a naive algorithm would enumerate all combinations of  $k$  centers, and for each combination, assign remaining nodes to the centers such that both the radius constraint and the internal connectedness constraint are satisfied. However, we note that there may exist some “articulation” node  $v$  which itself can connect to two or more centers within radius  $r$ , and  $v$  is critical (the only choice) to connect some other nodes to their corresponding centers. In a legal partitioning, every articulation node must be assigned to a unique center. If there are many such articulation nodes, it is difficult to assign each of them to the right center in order to maintain the linkages for others. Therefore the naive algorithm may fail to determine the specified clustering. Hence, intuitively the  $CkC$  problem is hard even for a constant  $k$ . In the following, we prove a hardness result for the  $CkC$  problem, by a reduction from 3CNF-SAT. For completeness, we define the 3CNF-SAT problem as follows:

**DEFINITION 3.2. (3CNF-SAT)** Given a set  $U = \{u_1, \dots, u_n\}$  of variables, let  $C = C_1 \wedge C_2 \wedge \dots \wedge C_m$  be an instance of 3CNF-SAT. Each  $C_i$  contains three literals

$$C_i = l_i^1 \vee l_i^2 \vee l_i^3,$$

where each literal  $l_i^x, x = 1, 2, 3$ , is a variable or negated variable. Decide whether there is a truth assignment of  $U$  which satisfies every clauses of  $C$ .

**THEOREM 3.1.** For any  $k \geq 2$  and  $d \geq 1$ , the  $CkC$  problem is NP-Complete.

*Proof.* We only show the case for  $k = 2$  and  $d = 1$ , the same proof can be easily extended to larger  $k$  and  $d$ . First we prove  $C2C$  is in NP. We can nondeterministically guess a partitioning of a graph  $G$  and pick a node as center from each partition. For each partition, we can deterministically check if it is valid by traversing the corresponding subgraph to verify both the connectedness and radius constraints.

Next, we perform a reduction from 3CNF-SAT to show the NP-hardness. Denote  $L = \{u_1, \bar{u}_1, \dots, u_n, \bar{u}_n\}$  to be the set of literals. For any 3CNF-SAT instance

$C = C_1 \wedge C_2 \wedge \dots \wedge C_m$ , we construct an instance of  $C2C$   $f(I) = (G, w, r)$ , where  $G = (V, E)$  is a graph,  $w : V \rightarrow \mathcal{R}$  is a function which assigns a coordinate vector to each node,  $r \in \mathcal{R}^+$  is the radius constraint, by the following procedure:

1. First, we create a set of nodes  $V = PULUCUAUB$ .  $P = \{p_0, p_1\}$  where  $p_0, p_1$  are two center nodes;  $L$ ,  $C$  are the sets of literals and clauses respectively;  $A = \{a_1, \dots, a_n\}$  and  $B = \{b_1, \dots, b_n\}$  are two sets of nodes introduced only for the purpose of reduction. In the following, we shall refer to the cluster including node  $p_0$  (or  $p_1$ ) as  $V_0$  (or  $V_1$ , resp.).
2. Next, we link the nodes created in step 1. First, we connect  $p_0$  and  $p_1$  with every node in  $L$ . Second, for every literal  $l \in L$  and clause  $C_i \in C$ , we link  $l$  and  $C_i$  if  $l \in C_i$ . Finally,  $\forall i \in \{1, 2, \dots, n\}$ , we link each of  $a_i$  and  $b_i$  with each of  $u_i$  and  $\bar{u}_i$ . Refer to Figure 2 for a visualization of graph  $G$ . Note that every node in  $A, B, C$  can only connect to the center nodes  $p_0$  and  $p_1$  via some node in  $L$ , hence the nodes in  $L$  are *articulation* nodes.

Now, we assign each node in  $V$  a carefully chosen coordinate such that each node of  $A, B, C$  is within distance  $r$  to one unique center node  $p_0$  or  $p_1$ . Note in order to have a legal partitioning, every articulation node in  $L$  must be assigned to an appropriate center (cluster). For the reduction, we associate a truth value (true or false) to each cluster; accordingly, the allocations of these articulation nodes can then be transferred back to a truth assignment for the input 3CNF-SAT instance  $I$ . Besides, we need to guarantee that the truth assignment we get for  $I$  is proper, i.e.,  $\forall i \in \{1, 2, \dots, n\}$ , node  $u_i$  and  $\bar{u}_i$  belong to different clusters. Node sets  $A$  and  $B$  are two gadgets introduced for this purpose.

3. Finally, we simply set an arbitrary positive value to  $r$  and assign each node  $v \in V$  a coordinate as follows:

$$w(v) = \begin{cases} 0, & \text{if } v \in B; \\ r, & \text{if } v = p_0; \\ 2r, & \text{if } v \in L; \\ 3r, & \text{if } v = p_1; \\ 4r, & \text{if } v \in A \cup C. \end{cases}$$

Figure 3 illustrates the deployment of nodes on the line.

Clearly the above reduction is polynomial. Next, we show that  $I$  is satisfiable if and only if  $f(I) = (G, w, r)$  has a legal partitioning.

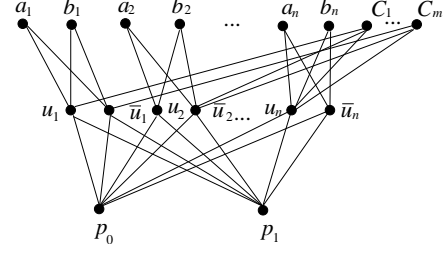


Figure 2: The constructed graph  $G$ .

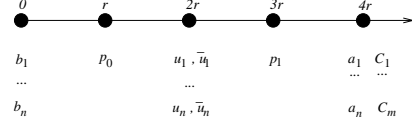


Figure 3: Deployment of nodes on the line.

“ $\Leftarrow$ ”: If  $f(I) = (G, w, r)$  has a legal partitioning, we can have the following simple observations:

OBSERVATION 3.1.

1. Both  $p_0$  and  $p_1$  must be selected as centers, otherwise some node can not be reached within  $r$ .
2. For the same reason, each node in  $A$  and  $C$  must be assigned to cluster  $V_1$  and each node in  $B$  must be assigned to  $V_0$ .
3. For any  $i \in \{1, \dots, n\}$ , node  $u_i$  and  $\bar{u}_i$  can not be in the same cluster. If node  $u_i$  and  $\bar{u}_i$  are both assigned to cluster  $V_0$  (or  $V_1$ ), some node in  $A$  (or  $B$ ) is not able to connect to  $p_1$  (or  $p_0$ ).
4. For each clause  $C_i \in C$ , there must be at least one literal assigned to cluster  $V_1$ ; otherwise  $C_i$  will be disconnected from  $p_1$ .

We construct a satisfying assignment for  $I$  as follows. For each variable  $u_i \in U$ , if  $u_i$  is assigned to  $V_1$ , set  $u_i$  to be true, otherwise false. Note by Observation 3.1.3,  $u_i$  and  $\bar{u}_i$  are always assigned with different values, hence the assignment is proper. Moreover, the assignment satisfies  $I$  since, by Observation 3.1.4, all the clauses are satisfied.

“ $\Rightarrow$ ”: If  $I$  is satisfiable, we construct a partitioning  $\{V_0, V_1\}$  as follows.

$$V_0 = B \cup \{p_0\} \cup \{l_i \in L \mid l_i = \text{false}\}, V_1 = V \setminus V_0.$$

It is easy to verify that the above solution is valid since every node in  $V$  is within distance  $r$  from one center node,  $p_0$  or  $p_1$ .

Finally, we note that the above proof can be easily extended to bigger  $k$  and  $d$ . When  $k > 2$ , one can always

add  $k - 2$  isolated nodes (hence each of them must be a center) to graph  $G$  and apply the same reduction; when  $d > 1$ , one can simply add  $d - 1$  dimensions with identical value to the coordinate vector  $w$ . ■

The internal connectedness constraint poses new challenges to the traditional  $k$ -center problem. Table 1 compares the hardness of these two problems in different settings.

	Traditional $k$ Center	$CkC$
$k$ is a constant	Polynomial Solvable	NP-complete
$k$ is an input, $d = 1$	Polynomial Solvable	NP-complete
$k$ is an input, $d > 1$	NP-complete	NP-complete

Table 1: Complexity results.

*Remarks:*

1. Theorem 3.1 also implies that the following sub-problem is NP-hard: Given a set of  $k$  centers, assign each node to one particular center to obtain  $k$  clusters which are all internally connected and the maximum cluster radius is minimized.
2. Similar to the  $CkC$  problem, one can define the connected  $k$ -median and  $k$ -means problems. In fact, the proof of Theorem 3.1 can be extended to these problems to show their NP-Completeness.

#### 4 Approximation Algorithm

In this section we consider the min- $CkC$  problem, the corresponding optimization version of  $CkC$  and present an approximation algorithm.

**DEFINITION 4.1.** (*min- $CkC$  problem*) Given  $k$  as the number of centers, a distance function  $\|\cdot\|$ , and a graph  $G = (V, E)$  where every node in  $V$  is associated with a coordinate vector  $w : V \rightarrow \mathcal{R}^d$ . Find the minimum radius  $r \in \mathcal{R}^+$ , such that there exist disjoint partitions  $\{V_1, \dots, V_k\}$  of  $V$ , i. e. ,  $V = V_1 \cup \dots \cup V_k$  and  $\forall 1 \leq i < j \leq k, V_i \cap V_j = \phi$ , which satisfy the internal connectedness constraint and radius constraint defined in Definition 3.1.

In the following we prove an inapproximability result for the min- $CkC$  problem, which can be viewed as a corollary of Theorem 3.1.

**THEOREM 4.1.** For any  $k \geq 2$ ,  $\epsilon > 0$ , the min- $CkC$  problem is not approximable within  $2 - \epsilon$  unless  $P = NP$ .

*Proof.* We only study the case for  $k = 2$ . We show, if there is a polynomial algorithm  $\mathcal{A}$  which is guaranteed to find a feasible solution within  $(2 - \epsilon)opt$ , it can

actually be used to solve the 3CNF-SAT problem. The reduction is similar to the proof of Theorem 3.1. First, for a given 3CNF-SAT instance  $I$ , we construct a  $C2C$  instance  $f(I) = (G, w, r)$  by the same procedure as in the proof of Theorem 3.1. Later, we invoke Algorithm  $\mathcal{A}$  on the input  $(G, w, r)$ .

Since the coordinates of all nodes are multiples of  $r$ , the optimal radius must also be a multiple of  $r$ . Hence, if Algorithm  $\mathcal{A}$  returns a solution smaller than  $2r$ , the optimal radius must be  $r$ . By the same argument as in the proof of Theorem 3.1,  $I$  is satisfiable. Otherwise if Algorithm  $\mathcal{A}$  returns a solution bigger than or equal to  $2r$ , since Algorithm  $\mathcal{A}$  is guaranteed to find a solution within  $(2 - \epsilon)r$ , the optimal radius is at least  $2r$  and consequently  $I$  is not satisfiable. Hence, unless  $P = NP$ , the min- $CkC$  problem can not be approximated within  $2 - \epsilon$ . ■

Let  $opt$  be the optimal radius of the above min- $CkC$  problem. In the following we propose a polynomial algorithm which is guaranteed to find a solution within  $3opt$ . To prove this result, first we remove the constraint of min- $CkC$  that each point must belong to exactly one of the clusters, and define a relaxed problem which we will refer to as min- $CkC'$ :

**DEFINITION 4.2.** (*min- $CkC'$  problem*) Given  $k$  as the number of clusters, a distance function  $\|\cdot\|$ , and a graph  $G = (V, E)$  where every node in  $V$  is associated with a coordinate vector  $w : V \rightarrow \mathcal{R}^d$ . Find the minimum radius constraint  $r \in \mathcal{R}^+$ , such that there exist node sets  $V_1, \dots, V_k \subseteq V$ ,  $V = V_1 \cup \dots \cup V_k$ , which satisfy the internal connectedness constraint and radius constraint defined in Definition 3.1.

We first propose a polynomial procedure to solve the min- $CkC'$  problem. Let  $opt'$  be the optimal radius of min- $CkC'$ . Clearly  $opt' \leq opt$  since  $opt$  is also a feasible solution of min- $CkC'$ . Next we present a procedure to transfer the optimal solution of min- $CkC'$  to a solution of min- $CkC$  with radius at most  $3opt'$ . Combining the two procedures we obtain an algorithm for min- $CkC$  with approximation ratio 3.

**4.1 Solving the min- $CkC'$  problem.** To solve the min- $CkC'$  problem, we need the following notion of *reachability*:

**DEFINITION 4.3.** Given a graph  $G = (V, E)$ , for  $u, v \in V$ ,  $v$  is reachable from  $u$  (w.r.t.  $r$ ) if there exists a path  $u \rightarrow s_1 \rightarrow \dots \rightarrow s_k \rightarrow v$ , such that  $s_1, \dots, s_k \in V$ ,  $(s_i, s_{i+1}) \in E$  and  $\forall 1 \leq i \leq k, \|w(u) - w(s_i)\| \leq r$ .

Intuitively,  $v$  is reachable from  $u$  w.r.t.  $r$ , if and only if  $v$  can be included in the cluster with center  $u$

---

**Algorithm 1** A polynomial algorithm for the min- $CkC'$  problem.

---

- 1: Calculate all pairwise distances of  $V$  and store those distance values in set  $D$ ;
  - 2: Sort  $D$  in decreasing order;
  - 3: **for** every value  $d \in D$  **do**
  - 4:   Enumerate all possible  $k$  centers;
  - 5:   **for** every set of  $k$  centers  $\{c_1, \dots, c_k\} \subseteq V$  **do**
  - 6:     Perform BFS from each center  $c_i$  and mark all nodes which are reachable from  $c_i$  w.r.t.  $d$ ;
  - 7:     **if** all vertices are marked **then**
  - 8:       Return  $d$  and  $k$  clusters;
  - 9:     **end if**
  - 10:   **end for**
  - 11: **end for**
- 

and radius  $r$ . Clearly it can be decided in polynomial time by performing a breadth first search (BFS) for node  $v$  from node  $u$ . Based on this, we propose a simple algorithm, Algorithm 1, to solve the min- $CkC'$  problem.

**Runtime complexity:** Note  $|D| = \binom{n}{2}$  and we enumerate all possible  $k$  centers, hence the search space is polynomial, i.e.,  $n^{k+2}$ . Besides, BFS takes time at most  $O(n^2)$ . Thus Algorithm 1 must terminate in  $O(n^{k+4})$  steps.

**4.2 Back to the min- $CkC$  problem.** Let  $sol' = \{V'_1, \dots, V'_k\}$  be the clustering found by Algorithm 1 where  $V'_i \subseteq V$ . Note  $V'_1, \dots, V'_k$  may not be disjoint. We propose the following procedure (Algorithm 2) to determine  $k$  disjoint clusters for the min- $CkC$  problem, denoted by  $sol = \{V_1, \dots, V_k\}$ . Let  $c_1, \dots, c_k$  be the centers of  $V_1, \dots, V_k$ . Note that  $c_1, \dots, c_k$  are also the centers of  $V'_1, \dots, V'_k$ .

---

**Algorithm 2** 3-approximation algorithm for the min- $CkC$  problem.

---

- 1: **for**  $i$  from 1 to  $k$  **do**
  - 2:    $V_i = \phi, c_i \leftarrow c'_i$ ;
  - 3:   Add all nodes reachable w.r.t.  $r$  from  $c_i$  in  $G[V'_i \setminus \cup_{j=1}^{i-1} V_j]$  to  $V_i$  (by performing a BFS from  $c_i$  in  $G[V'_i \setminus \cup_{j=1}^{i-1} V_j]$ );
  - 4:   **for** every node  $v \in (\cup_{j=1}^{i-1} V_j) \cap V'_i$  **do**
  - 5:     Add all nodes reachable w.r.t.  $r$  from  $v$  in  $G[V'_i]$  to the cluster of  $v$  (by performing a breadth first search from  $v$  in  $G[V'_i]$ );
  - 6:   **end for**
  - 7: **end for**
  - 8: Output clusters  $V_1, \dots, V_k$ ;
- 

Algorithm 2 assigns every node in  $V$  to a unique

cluster  $V_i$  for  $1 \leq i \leq k$ . For each iteration  $1 \leq i \leq k$ , line 3 allocates nodes in  $V'_i$  which have not been assigned to any previous clusters  $V_1, \dots, V_{i-1}$  to  $V_i$ . Afterwards, there may still be some unassigned nodes in  $V'_i$ . In line 5, we allocate the unassigned nodes to one of the clusters  $V_1, \dots, V_{i-1}$  from which they are reachable w.r.t.  $r$ .

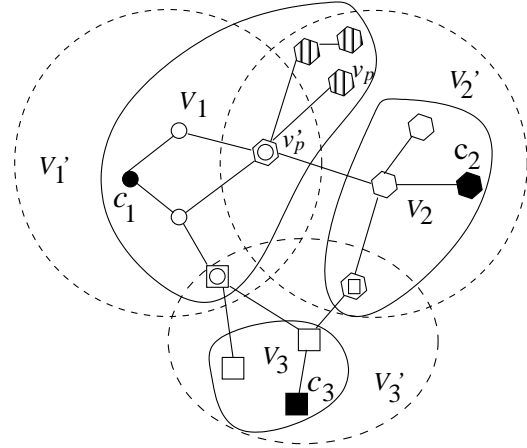


Figure 4: Demonstration of Algorithm 2.

Algorithm 2 is illustrated in Figure 4. The circles with dashed lines represent three initial clusters (with overlapping)  $V'_1, V'_2$  and  $V'_3$  generated by Algorithm 1. Applying Algorithm 2, we obtain three new clusters  $V_1, V_2$  and  $V_3$  that are disjoint (The center nodes do not move).

**LEMMA 4.1.** *Algorithm 2 is guaranteed to find a solution of the min- $CkC$  problem with maximum radius at most  $3opt'$ .*

*Proof.* First we show that Algorithm 2 assigns every node  $u \in V$  to a unique cluster. There are two cases. In case 1,  $u$  can be reached via a path from the center node  $c_i$  without having any node previously assigned to  $V_1, \dots, V_{i-1}$  on the path; then,  $u$  is assigned to  $V_i$  in line 3 of Algorithm 2. In case 2,  $u$  is connected to  $c_i$  via some node  $v \in \cup_{j=1}^{i-1} V_j$ ; then, in line 5 of Algorithm 2,  $u$  is assigned to the cluster that  $v$  belongs to.

Next, we bound the maximum radius of a node  $u$  to the corresponding center node. In case 1, since  $u$  is assigned to  $V_i$ , the distance between  $u$  and  $c_i$  is at most  $opt'$ . In case 2, observe that the maximum distance between  $u$  and  $v$  is at most  $2opt'$  due to the triangle inequality and the fact that  $u$  and  $v$  were in the same set  $V'_i$ . Besides, we observe that the distance between  $v$  and its corresponding center node  $c_j$  is at most  $opt'$ . Therefore, again by the triangle inequality, the distance between  $u$  and its corresponding center node is at most  $3opt'$ . ■

**THEOREM 4.2.** *Combining algorithm 1 and 2 gives a polynomial 3-approximation algorithm for the min- $CkC$  problem.*

*Remark:* Note the approximation results rely on the triangle inequality; thus, to make the approximation results valid, the distance function for min- $CkC$  has to be metric. However, for the NP-completeness proof, the distance function can be non-metric.

## 5 Heuristic Algorithm

The complexity analysis has demonstrated the hardness of the  $CkC$  problem. Moreover, Theorem 3.1 implies that even the assignment step alone, i.e., given  $k$  centers finding the optimal assignment of the remaining nodes to minimize the radius, is NP-hard. While providing an algorithm with guaranteed performance is important theoretically, the expensive enumeration operation prevents it from being practical in large datasets. In this section, we propose **NetScan**, an efficient heuristic algorithm.

NetScan follows a three-step approach, which starts by picking  $k$  centers randomly, then assigns nodes to the best center and refines the clustering results iteratively.

- Step I: Randomly pick  $k$  initial cluster centers.
- Step II: Assign all nodes to clusters by traversing the input graph.
- Step III: Recalculate cluster centers.

The algorithm repeats steps II and III until no change of the cluster centers occurs or a certain number of iterations have been performed. In step III, finding the optimal center from a group of  $n$  nodes requires  $O(n^2)$  time. To make it more scalable for large datasets, we select the node closest to the mean of the cluster as the new center. Typically, the mean provides a reasonably good approximation for the center.

The three steps look similar to the  $k$ -means algorithm. However, the complexity analysis tells us that given  $k$  centers, finding an optimal assignment requires a search through an exponential space, which is unacceptable in practice. Thus, the major challenge of NetScan is finding a good membership assignment, i.e., step II.

From the design principle of the approximation algorithm, we observe that the BFS-based approach provides an efficient way to generate clusters without violating the internal connectedness constraint. Inspired by this observation, we start the membership assignment from the center. Neighboring nodes (directly connected by an edge of the graph) of already assigned nodes are gradually absorbed to the cluster. The search

starts from cluster centers with respect to an initial radius threshold  $R_0$ . Nodes are tested and assigned to the first cluster where their distances to the center are no larger than the current radius threshold  $R_i$ . If all the centers have been processed with  $R_i$  and not all nodes have been assigned, the assignment is resumed in the next *radius increment round* with a larger radius threshold  $R_{i+1}$ . The pseudo code of step II is provided in Algorithm 3, and more detailed aspects of NetScan are discussed later. A running example is illustrated in Figure 5 with the assignment sequence given in Table 2.

---

### Algorithm 3 Step II of NetScan.

---

```

1: Empty working queue  $Q$ ;
2: for every center  $c_j$  of cluster  $C_j$  do
3:   Append all unassigned neighbors of  $c_j$  to  $Q$ ;
4:   while  $Q$  is not empty do
5:     Pop the first element  $q$  from  $Q$ ;
6:     if  $\|q - c_j\| \leq R_i$  then
7:       if  $q$  is a potential articulation node then
8:         Invoke the look-ahead routine to decide the
           membership for  $q$ . If  $q$  should be assigned
           to  $C_j$ , append  $q$ 's unassigned neighbors to
            $Q$ ; otherwise, only assign  $q$  to the right
           cluster without appending  $q$ 's neighbors to
            $Q$ ;
9:       else
10:        Assign  $q$  to  $C_j$  and append  $q$ 's unassigned
           neighbors to  $Q$ ;
11:      end if
12:    end if
13:  end while
14: end for
15: if all nodes are assigned to some  $C_j$  then
16:   Stop;
17: else
18:   Increase  $R_i$  and goto 1;
19: end if

```

---

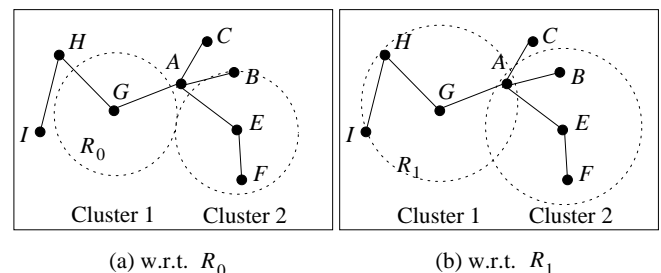


Figure 5: NetScan demonstration.

**Why gradually increment  $R_i$ ?** The radius threshold  $R_i$  plays an important role in minimizing the maximum







