21

MST CONSTRUCTION WITH METRIC MATRIX FOR CLUSTERING

Masahiro Ishikawa*, Yi Liu**,
Kazutaka Furuse*, Hanxiong Chen** and Nobuo Ohbo*
*Doctoral program in Engineering, University of Tsukuba, 1-1-1 Tennno-dai, Tsukuba 305-8573, Japan
mi@dblab.is.tsukuba.ac.jp
**Tokiwa System Engineering Inc.
liu_yi@tokiwa-sys.co.jp
**Department of Computer Science, Tsukuba International University, Manabe 6, Tsuchiurashi, Ibaraki 300-0051, Japan.
chx@dblab.is.tsukuba.ac.jp
*Instiute of Information Sciences and Electronics, University of Tsukuba, 1-1-1 Tennno-dai, Tsukuba 305-8573, Japan
{furuse, ohbo}@dblab.is.tsukuba.ac.jp

Abstract Clustering is one of the most important topics in the field of knowledge discovery from databases. Specifically, hierarchical clustering is useful because it can be used to interactively guide users in browsing a huge database. In many cases, database clustering can be modeled as a graph partitioning problem, because a database with a distance function defined on it can be regarded as an edge weighted graph. So process of MST(Minimal Spanning Tree) construction is a possible solution to this problem.

In this paper, we propose an efficient MST construction method for a database with an arbitrary distance function on it. Our method utilizes a metric index to reduce the number of distance calculations needed to construct an MST. For this purpose, we introduce a new metric index named *metric matrix*. Experimental results show that our method can reduce the number of distance calculations needed in comparison with the classical method.

Keywords: Clustering, MST, Metric Index, Data Mining Knowledge Discovery from Databases

H. Arisawa et al. (eds.), *Advances in Visual Information Management*

1. INTRODUCTION

Clustering is one of the most important topics in the field of knowledge discovery from databases. Clustering algorithms can be classified into two types, partitional and hierarchical ones. Especially, hierarchical clustering is useful because it can be used to guide users in exploring huge databases. Since distance functions are often defined on databases to measure dissimilarities between data objects, we can derive an edge-weighted complete graph from a database by mapping objects and distances to vertices and weights of edges, respectively. Then clustering of a database can be modeled as a partitioning problem of a graph derived from the database and a distance function defined on it.

MST(Minimal Spanning Tree) construction is a possible solution to the graph partitioning problem. Figure 1 illustrates why MST construction can be a solution to it. In the figure, an MST of a graph and its representation in a form of tree called *dendrogram* are depicted. Notice that a dendrogram can be viewed as a hierarchical clustering structure of the graph. Although well known classical algorithms for MST construc-

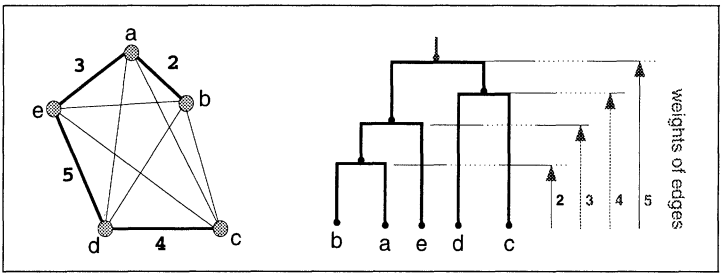

Figure 1 MST of a Graph and its Dendrogram

tion, such as Prim's and Kruskal's algorithms, are available, they are not efficient for many recent applications, such as image databases and video databases. Since a distance calculation tends to be expensive in such applications and graphs derived from databases are complete, the number of distance calculations needed for MST construction amounts to nearly $\frac{|V|(|V|-1)}{2}$, where $|V|$ is the number of vertices of the graph.

In order to reduce the number of expensive distance calculations, we propose an efficient MST construction method, which utilizes a metric index. This method can be applied to such a graph whose weights of edges satisfy the metric postulate. In our method, near neighbour searches centered at the same vertices are repeatedly performed by incrementally changing the radius using a metric index. Therefore, straight-

forward application of existing metric indices, such as MVP-tree (Tolga et al., 1997) and MI-tree (Ishikawa et al., 1999), to our method necessitates redundant distance calculations.

In order to overcome this difficulty, we introduce a new metric index for MST construction named *metric matrix*, which supports radius-incremental differential near neighbour searches and eliminates redundant distance calculations in the course of MST construction.

2. APPROACH

2.1. MST PROBLEM

Let $G = \langle V, E, \phi \rangle$ be a non-directed, connected, edge weighted graph, where V is a set of vertices, $E(\subseteq V \times V)$ is a set of edges, and $\phi : E \to \mathcal{R}$ is a real-valued weight function. If an edge exists for any pair of vertices, G is said to be *complete*. A *spanning tree* of G is a connected graph $G_{st} = \langle V, E_{st}, \phi \rangle$ such that $E_{st} \subseteq E$ and $|E_{st}| = |V| - 1$(this means that G_{st} has no cycle). An *MST* of a graph G is a spanning tree of G in which the sum of weights in the tree ($W = \sum_{e \in E_{st}} \phi(e)$) is minimal. The problem of constructing an MST of a given edge weighted connected graph is called the *MST problem*(Tarjan, 1983).

When the weight function ϕ of G is *metric*, i.e., ϕ satisfies the metric postulate (positivity, symmetricity and triangle inequality), G is called a *metric graph.*

2.2. GREEDY METHOD

The *greedy method* is a simple generic technique to solve the MST problem(Tarjan, 1983). We summarize this method from the view point of clustering here.

The method starts with the disjoint set of clusters, which contains each object in an individual cluster. Each cluster is associated with a set of edges, called its *front line*, and each edge in a front line connects a vertex inside the cluster with a vertex outside the cluster. A *safe edge* is one of the shortest edges(edges with minimal distance) in the front line. Then two clusters connected by a safe edge are successively merged until only one cluster remains. Exactly $|V| - 1$ edges are used to merge clusters and they construct an MST. Note that MST construction process can be viewed as a process of hierarchical agglomerative, single-linkage clustering. By specifying a procedure to choose two clusters to be merged in each step, a specific variation, such as Prim's and Kruskal's method, is derived.

Assuming that time complexity of a single distance calculation is $O(1)$, the gross time complexity of the greedy method is $O(|V|^2 \log |V|)$, when applied to a complete graph. $|V|^2$ part of this order is due to the fact that the number of edges in a complete graph with $|V|$ vertices is $\frac{|V|(|V|-1)}{2}$. Thus, the total cost of MST construction becomes prohibitive when a classical method is naively applied to a complete graph with an expensive distance function.

2.3. OUR APPROACH

To address the problem pointed out so far, our work aims to reduce the number of expensive distance calculations needed to construct an MST of a metric complete graph.
Our approach is founded upon the presumption that:

> *ideally, only edges contained in the constructed MST are needed for the MST construction.*

In other words, edges not contained in the constructed MST need not to be touched in the course of the MST construction. However, in reality, we have to examine other edges to ensure that edges at hand are really the edges to be contained in the final MST.

Since edges needed to construct an MST are *safe edges*, longer edges(edges with larger distances) are less probable to be selected from front lines in each step. Therefore, some longer edges could be *ignored* and distance calculations for them could be omitted in the course of MST construction. We expect that metric indices enable us to exclude longer edges from front lines and to reduce the number of distance calculations needed for MST construction.

Metric Indices and Near Neighbour Searches.

The point of our approach to reduction of the number of distance calculations is to reduce the size of front lines by excluding longer edges from them. This can be realized by utilizing a metric index. Several metric indices have been proposed for near neighbour searches in arbitrary metric spaces(Tolga et al., 1997, Brin, 1985, Ciaccia et al., 1997, Ishikawa et al., 1999) and they support *r-neighbour searches.*

Definition 1 (query region of an object q and radius r) *Given a query object q $(\in V)$ and radius $r(>0)$, r-neighbour of q is called the query region of q and r.*

Definition 2 (r-neighbour search)
Given a query object q $(\in V)$ and radius $r(>0)$, r-neighbour search of

q with radius r retrieves objects inside the query region of q and r, i.e., $\{v \mid v \in V \wedge \phi(q, v) \leq r\}$.

All existing metric indices utilize the triangle inequality to filter out objects outside the query region of q and r. However, such filtering is not sufficient because some objects outside the query region can not be filtered out. To remove such objects, distances between the query object and objects passing through the filter have to be calculated. Almost all distance calculations in a metric index are performed for this refinement purpose.

3. MST CONSTRUCTION WITH METRIC INDEX

In the classical greedy method described in 2.2, each cluster has a front line and the front line of each cluster contains every edge connecting a vertex inside the cluster and a vertex outside the cluster. However, all edges except the shortest one in each front line can be omitted, since only the shortest edge is needed to merge clusters in each step.

We aim to exclude such edges from front lines by using a metric index. Figure 2 shows the framework of our method obtained from the classical framework by modifying the following points:

1 initially, front lines contain no edge (line 5), and

2 front lines are updated after cluster selection (line 12).

In the method `C.updateFrontLine()` invoked at line 12, near neighbour searches by a metric index are performed to retrieve short edges to ensure that the shortest edges are contained in the front line. This is, so to speak, *search on demand* strategy. Notice that, as MST construction process proceed, near neighbour searches centered at the same vertices may be performed repeatedly by increasing the radii stepwise for updating front lines.

Performance.

Two factors play important roles in the performance analysis. One is the number of edges to be contained in front lines, and the other is the number of near neighbour searches performed to update front lines. The former affects the number of distance calculations and it is strongly dependent on the performance of a metric index used, while the latter is principally dependent on the algorithm of the method `P.selectCluster()`.

Thus, `P.selectCluster()` determines the characteristics of each variations. More precisely, the algorithm of `P.selectCluster()` determines

V	-	Vertices of the graph
P	-	Disjoint set of clusters (partition)
C	-	A cluster (pair of a set of vertices C_V and an associated front line C_F)
C_V	-	Vertices in a cluster
C_F	-	Front line of a cluster (edges between C_V and $V \backslash C_V$)
e	-	An edge (triplet of two vertices u, v and ϕ(u,v))
MST	-	Set of edges constructing an MST

```
1: // Initialization of the partition
2: P ← ∅
3: for all u ∈ V do
4:    C_V ← {u}
5:    C_F ← ∅ // initially, front line is empty
6:    P.append(⟨C_V, C_F⟩)
7: end for
8: // MST construction by merging clusters
9: MST ← ∅
10: while |MST| < |V| − 1 do
11:    ⟨C_V, C_F⟩ ← C ← P.selectCluster()
12:    C.updateFrontLine() // front line update
13:    ⟨u, v, dist⟩ ← e ← C_F.findSafeEdge()
14:    MST.append(e)
15:    P.mergeClusters(e)
16: end while
17: return MST
```

Figure 2 **Framework for Our Method**

the sequence of clusters which governs the principal characteristics of each variations.

In the process of MST construction, the following steps are repeated exactly $|V| - 1$ times:

1 select a cluster C_i from the partition P (by `P.selectCluster()`),

2 find a safe edge e_i from the front line of C_i (by $\mathtt{C_F}$`.findSafeEdge()`),

3 merge two clusters connected by e_i (by `P.mergeClusters(`$\mathtt{e_i}$`)`).

We get a sequence of selected clusters

$$C_0, \quad C_1, \quad C_2, \quad \cdots, \quad C_{|V|-2}.$$

Since near neighbour searches are possibly performed for every vertex in the selected cluster in the method `C.updateFrontLine()`, the total number of near neighbour searches performed (let it be S_N) is proportional to the sum of numbers of vertices in the selected clusters, i.e., $S_N \simeq \sum_{i=0}^{|V|-2} |C_i|$.

4. THE METRIC MATRIX

Existing metric indices cannot avoid duplication of distance calculations for the same edges in the course of MST construction. Hence the total number of distance calculations could not be reduced in spite of the reduction of the number of edges for which distances should be calculated.

What we really want is to invoke successive near neighbour range searches centered at the same query objects (see Figure 3). In order to

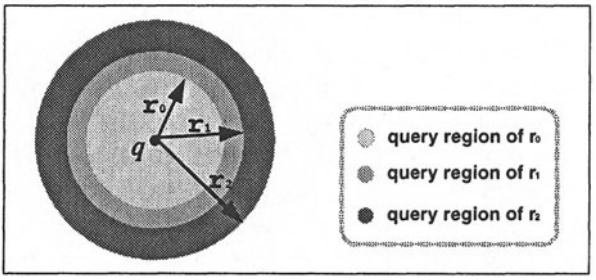

Figure 3 Radius-Incremental Near Neighbour Range Searches

avoid duplication of distance calculations in successive near neighbour range searches, we introduce a new metric index named *metric matrix.* The metric matrix supports the following searches.

Definition 3 (r-nearby search)
Given a query object q ($\in V$) and radius $r(>0)$, r-nearby search of q with radius r retrieves a set of objects, which includes the objects inside the query region of q and r. In other words, objects retrieved by an r-nearby search includes the objects retrieved by an r-neighbour search with the same parameters.

Definition 4 (differential r-nearby search)
Given a query object q ($\in V$) and radius $r_{i+1}(>0)$, a differential r-nearby search retrieves a set of objects, which includes the objects inside the query region of q and r_{i+1}. If a differential r-nearby search of q with radius r_{i+1} is preceded by a r-nearby search of q with radius r_i ($< r_{i+1}$), it never retrieve the objects already retrieved by the preceded search with radius r_i.

By using differential r-nearby searches for successive near neighbour searches in the course of MST construction, we can completely eliminate duplications of distance calculations.

Every existing metric index performs the following steps in the process of r-neighbour searches:

1 candidate selection (pruning or filtering) and

2 candidate refinement (remove objects outside the query region).

The first step is principally accomplished by using the triangle inequality. The second step eliminates objects residing outside the query region. In the second step, distances between the query object and candidate objects must be calculated. On the other hand, the metric matrix skips the second step, thus it needs distance calculations only in the first step. This is the reason why the set of object retrieved by an r-nearby search can contain objects outside the query region.

A metric matrix consists of several *metric vector*s. We call the number of metric vectors in a metric matrix the *degree* of it. Every metric vector is associated with its reference object. A metric vector is a vector of distances from the reference object to the other objects, and it is sorted in the ascending order. Several information related to each object, such as object IDs or radii specified so far, are also associated with a metric vector.

Let vp be the reference object of a metric vector, q be a query object, and R be the distance between them. Then if an object u is inside the query region of q and radius r, the distance between u and vp should hold the following inequalities:

$$R - r \quad \leq \quad \phi(vp, u) \quad \leq \quad R + r.$$

We call this range as a *candidate range*. In a metric vector a continuous region corresponds to a candidate range, and it is called a *candidate region*. A candidate region in a metric vector serves as a filter for the candidate selection, because objects inside a query region must be located in it. A metric matrix can be viewed as a collection of such filters. A set of objects retrieved by an r-nearby search is obtained by taking intersection of objects located in candidate regions in metric vectors.

5. EXPERIMENTAL RESULTS

Experiments are based on synthetic data sets. Data sets are prepared by the procedure described in (Jain et al., 1988) which generates normally distributed clusters in a d-dimensional vector space. In our experiments, data space is two-dimensional, clusters are equally sized and variance is $\sigma^2 = 0.1$, and clusters' centers are uniformly distributed. Distance is evaluated using the L_2 metric, i.e. $\phi(u, v) = L_2(u, v) =$

$(\sum_{i=1}^{d} |u[i] - v[i]|^2)^{\frac{1}{2}}$ where d is the dimension of the data space. The degree of metric matrix is $\log_2 |V|$ for a data set of size $|V|$.

Experiment had been done by two variations derived from our mehtod. They are named nnPrim and nnSmallestFirst, respectively. Figure 4 shows the result. Notice that a complete graph with 20,000 vertices has 199,990,000 edges, hence our method needs less number of distance calculations than 0.2% of distance calculations needed by the classical method.

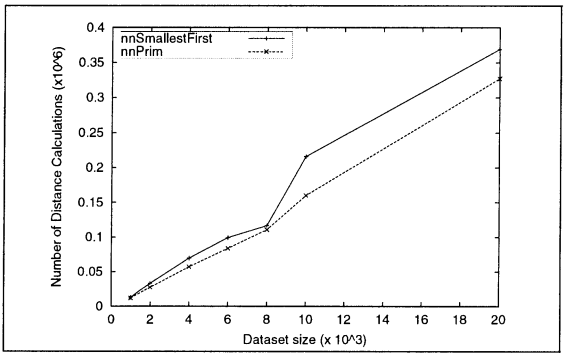

Figure 4 Experimental Results

6. SUMMARY

We proposed an approach for MST construction for large complete metric graphs utilizing a metric index. Using a metric index, we aimed to reduce the number of expensive distance calculations needed to construct an MST. To fulfill our purpose, we also invented a new metric index named *metric matrix* which supports differential r-nearby searches.

Applying the approach to the classical generic greedy method, we obtained a generic method for metric graphs. The experiments show that the proposed method reduced the number of distance calculations significantly.

It is worth mentioning that density information can be gathered in the process of our method, since near neighbour searches for vertices are performed. Remind that the process of MST construction can be viewed as a process of single-linkage clustering. Hence the clusters obtained from an MST may incur the defects of the single-linkage clusterings. By applying the criteria of density based clustering(Ester et al., 1996), the resultant clusters might be enhanced.

References

Ester, M. Kriegel, H.-P. Sander, J. and Xu, X. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise, *In Proc. of the 2nd Int'l Conference on Knowledge Discovery in Databases and Data Mining, Portland, Oregon.*

Tolga B. and Ozsoyoglu, M. (1997) Distance-based indexing for high-dimensional metric spaces, *Proceedings of the ACM SIGMOD Conference on Management of Data, Tucson, Arizona, May 1997.*

Ishikawa, M. Notoya, J. Chen, H. and Ohbo, N. (1999) A metric index MI-tree, *IPSJ Transations on Databases, Vol.40, No.SIG6, 1999.*

Tarjan, R. E.(1983) Data Structure and Network Algorithm, *Society for Industrial and Applied Mathematics.*

Brin, S. (1995) Near neighbour search in large metric spaces, *Proceedings of the 21st VLDB Conference, Zurich, Switzerland, 1995.*

Ciaccia, P. Patella, M. and ZezulaBib, P. (1997) M-tree: An efficient access method for similarity search in metric spaces, *Proceedings of the 23rd VLDB Conference, Athenes, Greece, 1997.*

Jain, A. K. and DubesBib R. C. (1988) Algorithm for Clustering Data, *Printice Hall.*

Biographies

Masahiro Ishikawa is a graduate student in a doctoral course of engineering of University of Tsukuba Japan. He received his Master of Eng. degree from the University of Tsukuba in 1995 and 1997, respectively.

Yi Liu is an engineer of Tokiwa Engineering Systems Company. She received her Master of Eng. degree from the University of Tsukuba in 1998.

Kazutaka Furuse is a research associate of the Institute of Information Sciences and Electronics, University of Tsukuba. He received his B.E., M.E., and Dr.Eng. degrees from the University of Tsukuba in 1988, 1990, and 1993, respectively.

Hanxiong Chen is an assistant professor of Department of Computer Science, Tsukuba International University. He received his PhD from the University of Tsukuba in 1993.

Nobuo Ohbo is a professor of the Institute of Information Sciences and Electronics, University of Tsukuba, Japan. He received his B.S., M.S., and Dr.Sci. degrees from the University of Tokyo in 1968, 1970, and 1979, respectively.