A constructive algorithm for realizing a distance matrix

Sacha C. Varone

LARIM - Department of Computer Engineering, École Polytechnique de Montréal, (Canada)

Abstract

The natural metric of a weighted graph is the length of the shortest paths between all pairs of vertices. The investigated problem consists in a representation of a given metric by a graph, such that the total length of the graph is minimized. For that purpose, we give a constructive algorithm based on a technique of reduction, fusion and deletion. We then show some results on a set of various distance matrices whose optimal realization is known.

Key words: metrics, reduction, heuristics

1991 MSC: 68R05, 05C10, 05C12, 05C50, 05C62

1 Introduction

The problem of realizing a distance matrix by a graph of minimal total length is a difficult problem. A lot of authors have considered the special case where the distance matrix is a tree metric, also called an additive metric (see for example [1–14]). Efficient algorithms that construct such trees were published in [15] or [16]. But real data describe merely a tree metric because they arise from a similarity measure that includes errors. Such a measure appears in various fields such as the study of evolution [17–20], the synthesis of certain electrical circuits [21], the seismic tomography, the traffic modelling [22,23] or the analysis of memory [10]. Therefore the interest was focus on approximating a metric by a tree metric [24–33] or finding the greatest sub-metric that is realizable by a tree [34].

Email address: sacha.varone@polymtl.ca (Sacha C. Varone).

1 This work has been partially funded by grant PA002-104974/1 from the Swiss National Science Foundation.

Preprint submitted to Elsevier Science 16 March 2005
In this paper we consider the problem of realizing a symmetric distance matrix by a weighted graph. We recall that a weighted graph $G = \langle V, E, w \rangle$ realizes the distance matrix $D$ of order $n$ if and only if $\{1, ..., n\} \subseteq V$ and $d^G_{ij} = d_{ij}$ for all $i, j = 1, ..., n$, where $d^G_{ij}$ is the length of a shortest path in $G$ between the vertices $i$ and $j$. The vertices in the set $\{1, ..., n\}$ are called external and the vertices in $V \setminus \{1, ..., n\}$ are called internal.

We use a heuristic to solve the problem since it has been proven to be NP-hard [35,36]. In the next section we describe the algorithm and in the last section we give some numerical experiments and make some remarks on the results.

2 Algorithm

We give below the algorithm that we have developed. Each step will be explained in the remaining of this section.

\begin{enumerate}
\item Compactify
\item Reduce
Until no compaction is possible
\item Greedy algorithm
\item Triangles’ reduction
\item Fusion
\item Remove useless edges/vertices
\item Decompaction/augmentation phase
\end{enumerate}

The complexity of this algorithm is polynomial in $O(n^4)$ (in the worst case) where $n$ is the number of entries in the distance matrix.

2.1 Compaction and reduction phase

This phase will be applied on the distance matrix itself. The aim is to simplify as much as possible the data. It is a preprocessing step.

Let $D$ be a distance matrix of order $n \times n$ and let $E_i$ be the matrix where

$$(E_i)_{jk} = \begin{cases} 
1 & : j = i \neq k \\
1 & : j \neq i = k \\
0 & : \text{elsewhere}
\end{cases}$$
Let $D_i(a) = D - a \ast E_i$ as in [1].

\[D_i(a) = \begin{pmatrix} d_{11} & \cdots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{n1} & \cdots & d_{nn} \end{pmatrix} - a \ast \begin{pmatrix} 0 & \vdots & 0 \\ \vdots & 1 \\ 0 & \vdots & 0 \end{pmatrix} \]

The question that arises then is: “How to choose the number $a$ such that $D_i(a)$ is still a distance matrix?” The answer is in the following theorem.

**Theorem 1** [1] $D_i(a)$ is a distance matrix $\iff a \leq \frac{1}{2}(d_{pi} + d_{ir} - d_{pr}) \forall p, r \neq i$.

The new metric $D_i(a)$ obtained from $D$ is called a compaction [37]. The compaction of an index $i$ of $D$ leads to a new matrix with a possible pair of equal rows (and by symmetry of equal columns). By deleting one of these equal rows and columns we obtain a new distance matrix whose order is one unit lower. This new matrix is called a reduction of $D$.

Define $a_0(i)$ as the maximal value such that $D_i(a_0(i))$ is a metric and let $a$ be smaller or equal to $a_0(i)$. $G_i(a)$ is a realization of $D_i(a)$ with its external vertices denoted by $v'_1, \ldots, v'_n$. To construct a realization $G$ of $D$ from $G_i(a)$ it is sufficient to add a vertex $v_i$ to $G_i(a)$ and connect $v_i$ to $v'_i$ with an edge whose weight is equal to $a$. This corresponds to step 6 of our algorithm.

**Theorem 2** [1] If $0 \leq a \leq a_0(i)$ and if $G_i(a)$ is an optimum realization of $D_i(a)$ then $G$ obtained from $G_i(a)$ is an optimum realization of $D$.

The first step towards searching for a realization of a distance matrix $D$ is to compact and to reduce it (if possible) iteratively so that the new matrix is still a metric. Note that, as mentioned in [37], if a distance matrix is tree-realizable then the iteration of the above process leads to a matrix of order 1.

This phase requires for each entry of the distance matrix to find two other entries $p$ and $r$ that minimize the quantity $d_{pi} + d_{ir} - d_{pr}$. The latter quantity is $a_0(i)$. This is done in a time complexity of $\Theta(n^2)$. Moreover the number of time that this compaction process has to be done is at most $\lceil \frac{n}{2} \rceil$ since it
is the number required by a simple path on \( n \) vertices. So the overall time complexity for this phase is \( \Theta(n^4) \)

From now on we will consider as a data a distance matrix of order \( N \times N \).

### 2.2 Greedy algorithm

A distance matrix \( D \) of order \( N \times N \) is trivially realized by the complete graph with for each edge a weight equals to the distance in \( D \). A first realization is obtained as follow:

**Greedy algorithm**

1. Start from a graph \( G = (V, \emptyset) \)
2. Sort the distances by a non-decreasing order in a list \( L \)
3. For each \( d_{i,j} \in L \)
   - if \( d^G_{i,j} > d_{i,j} \) add edge \((i, j)\) of weight \( d_{i,j} \)
   - if no path exists between \( i \) and \( j \), set \( d^G_{i,j} = \infty \)

In the case that the graph so far obtained is a polygon, it is also an optimal realization. This is formalized by the underlying theorem:

**Theorem 3** [38]Let \( D \) be realized by a polygon on at least four vertices denoted \( 1, 2, \cdots, n \). This realization is unique and optimal if and only if
\[
\forall i \mod(n) \ d(i - 1, i) + d(i, i + 1) = d(i + 1, i + 1)
\]

The greedy algorithm works in a time complexity of \( O(e \log(e)) \) where \( e \) is the number of distances, i.e. \( O(N^2 \log(N)) \)

We observe that if a distance matrix can be realized by an unicyclic graph with (pendant) trees attached to the vertices of the cycle, then the following algorithm leads to the optimal realization:

**Algorithm**

1. Repeat
   - Compactify
   - Reduct
   - Until no compaction is possible
2. Greedy algorithm
3. Decompaction/Augmentation phase

In this last algorithm step 1 ensures to get a reduced matrix whose realization is an unicyclic graph and step 2 constructs this graph.

**Example 4** The greedy algorithm applied to the below distance matrix gives
the complete graph.

\[ D = \begin{pmatrix}
0 & 4 & 4 & 2 \\
4 & 0 & 2 & 4 \\
4 & 2 & 0 & 4 \\
2 & 4 & 4 & 0 \\
\end{pmatrix} \]

\[ \begin{array}{c}
\text{a} \\
\text{b} \\
\text{c} \\
\end{array} \quad \begin{array}{c}
\text{X} \\
\text{Y} \\
\text{Z} \\
\end{array} \]

\[ \begin{array}{c}
4 \\
4 \\
4 \\
\end{array} \quad \begin{array}{c}
2 \\
4 \\
2 \\
\end{array} \]

2.3 Triangles’ reductions

The purpose of this step is to simplify all the triangles produced by the greedy algorithm. The general following transformation is applied.

\[ \begin{array}{c}
\text{a} \\
\text{b} \\
\text{c} \\
\end{array} \quad \begin{array}{c}
0.5(a+b-c) \\
0.5(b+c-a) \\
0.5(a+b-c) \\
\end{array} \]

We start from a realization \( G = (V, E, w) \) obtained with the greedy algorithm, with \( V \) as vertex set, \( E \) as edge set, and \( w \) a weight function on the edges. Since there is no optimal realization containing as a subgraph a complete graph on three vertices \([1]\), denoted by \( K_3 \) and called a triangle, we can look for existing \( K_3 \) in \( G \) and transform them into a 3-star (a star with 3 branches).

\[ K_3 \text{’reduction} \]

(1) Find the list \( L \) of all \( K_3 \) in \( G \)
(2) For each \( K_3 \in L \)
    If \( K_3 \) still exists in the current realization
    Transform it into a 3-star

This algorithm works in the worst case in \( O(N^3) \).

Let \( K_3^1 \) and \( K_3^2 \) be two complete graphs with three vertices in the realization \( G \) such that \( K_3^1 \cap K_3^2 \neq \emptyset \). The transformation of \( K_3^1 \) into a 3-star deletes one edge of \( K_3^2 \) and we have two triangles less in the realization \( G \).

Nevertheless we can say that the deleted edge is a virtual edge in the transformed graph since there exists a shortest paths in \( G \) of length given by the distance matrix. If the transformation of this “virtual \( K_3 \)”, namely \( K_3^2 \), leads to a better realization, then we make it effective. We could therefore try to transform all paths formed on triplets of external vertices of \( G \). The time
complexity is again in $O(N^3)$.

The first triangle reduction applied to the previous example gives:

Example 5

Considering the “virtual $K_3$” on the vertices $x, y$ and $z$, we get:

2.4 Fusion

When applying the above mentioned transformation, a special topology can occur: two internal vertices can be linked to two common vertices and some other vertices. In Figure 1 below, the internal vertices $s_1$ and $s_2$ are linked to the external vertices $x_1$ and $x_2$. The vertices $v_1$ and $v_2$ are assumed to be external.

We assume that if the graph $G$ contains such a topology $T_1$, there exists another path linking $v_1$ to $v_2$ whose length $e$ is such that $e \leq c + (b - a) + d$. Let also assume (wlog) that $a \leq b$.

We consider the transformation of this topology $T_1$ into $T_2$ described below:

The result is a fusion between two edges into another one. In the case that
If \( a = b \) then \( s_1 \) and \( s_2 \) are reduced into one single vertex (fusion of vertices). We give a justification of that transformation:

**Lemma 6** If \( G \) is a realization of a metric \( D \) then \( G' \) obtained by the above transformation is also a realization of \( D \).

**Proof**

We have to show that the length of a shortest path in \( T_1 \) between two external vertices is not modified by this transformation.

We denote by \( P_{T_i}(x, y) \) the length of the shortest path between \( x \) and \( y \) in the topology \( T_i \). We have

- \( P_{T_2}(x_1, x_2) = a + (b - a) + a = a + b = P_{T_1}(x_1, x_2) \geq D(x_1, x_2) \).
- \( P_{T_2}(x_1, v_1) = a + c = P_{T_3}(x_1, v_1) \geq D(x_1, v_1) \).
- \( P_{T_2}(x_1, v_2) = a + (b - a) + d = b + d = P_{T_1}(x_1, v_2) \geq D(x_1, v_2) \).
- By symmetry, equalities hold when \( x_1 \) is replaced by \( x_2 \).
- \( P_{T_2}(v_1, v_2) = c + (b - a) + d \geq e = D(v_1, v_2) \) by our assumption. This means that between \( v_1 \) and \( v_2 \) there exists a path whose length is \( e = D(v_1, v_2) \).

**Remark 1** The transformation can be generalized to the case of internal vertices \( s_1 \) and \( s_2 \) of degree greater or equal to three. Again, the neighbors of \( s_1 \) and \( s_2 \) are assumed to be external. The condition under which this transformation remains consistent is:

\[
P_{G'}(u, v) \geq P_G(u, v) \quad \forall u \in N(s_1) \quad \text{and} \quad \forall v \in N(s_2)
\]

where \( N(s_i) \) denotes the set of neighbors of \( s_i \), \( i = 1, 2 \) and \( P_G(u, v) \) is the length of a shortest path in \( G \) between \( u \) and \( v \).

The cost in time of such a transformation is a constant. But the search for such a topology could be very expansive if there exists a lot of internal vertices to be checked. The triangles’ reductions gives at most \( \binom{N}{3} \) internal vertices. So the number of occurrences of topology \( T_1 \) is bounded by \( \binom{N}{3}^2 \) and hence the fusion step is polynomial in \( O(N^6) \). Since this could be too much time consuming in practice, this phase has been applied a number of time proportional to the number of internal vertices. Therefore this fusion step runs in \( O(N^3) \).

**Example 7** We go on with our example and apply the fusion step. The realization obtained so far is:

![Diagram](attachment:image.png)
2.5 Deletion

Once the fusion step has been applied the current realization can contain useless edges and/or vertices. These edges have to be deleted in order to reduce the total length of the realization. This could be done by an updating rule which runs in the worst case in $O(N^3)$. As each edge has to be checked, the total running time for this step would be in $O(N^9)$. Since some experiments of our algorithm on random distance matrices have shown that the deleted edges $(u, v)$ are (almost) always such that $u$ belongs to the neighbors of $s_1$ and $v$ belongs to the neighbors of $s_2$, we have restricted the deletion phase to the set of edges $\{(u, v) \mid u \in N(s_1), \ v \in N(s_2)\}$. Hence the deletion phase runs in $O(N^2)$.

Example 8 Finally, in our example, the edge linking $z$ and $t$ could be deleted. We get:

![Diagram](image)

2.6 Decompactication/Augmentation

The last step of our algorithm is to get a realization of the original distance matrix. This can easily be done in at most $O(N)$ iterations with the method described in 2.1.

3 Numerical experiments

There is a lack of numerical results in the literature for the problem we have solved. Therefore we have produced our own data. We tested our algorithm on several distance matrices induced by a given topology. The appendix contains optimal realizations whose induced metrics were applied to our algorithm.

We indicate in the first column the name of the data file. Columns 2 to 4 shows the results without the compaction phase. Column 2 indicates the weight of the realization constructed with the greedy algorithm. Then we applied the $K_3$ reduction (column 3) and the “virtual $K_3$” reduction (column 4). We present in column 5 the result obtained with the compaction phase. The last column
shows the weight of an optimal realization.

<table>
<thead>
<tr>
<th>Topology</th>
<th>Greedy</th>
<th>$K_3$</th>
<th>“Virtual $K_3$”</th>
<th>Compact</th>
<th>Optimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>triangle</td>
<td>9</td>
<td>4.5</td>
<td>4.5</td>
<td>4.5</td>
<td>4.5</td>
</tr>
<tr>
<td>star4</td>
<td>12</td>
<td>8</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>star5</td>
<td>20</td>
<td>12</td>
<td>10</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>star6</td>
<td>30</td>
<td>20</td>
<td>14</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>H</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4triangles</td>
<td>93</td>
<td>46.5</td>
<td>46.5</td>
<td>46.5</td>
<td>46.5</td>
</tr>
<tr>
<td>4nontree</td>
<td>20</td>
<td>13</td>
<td>10</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>PolygonN</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
<td>n</td>
</tr>
<tr>
<td>2pentagons</td>
<td>22</td>
<td>16</td>
<td>11</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>2squares</td>
<td>16</td>
<td>16</td>
<td>9</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>Cube2</td>
<td>16</td>
<td>16</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>Hakimi64</td>
<td>18</td>
<td>13</td>
<td>13</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>Imrich84</td>
<td>250</td>
<td>192</td>
<td>156</td>
<td>130</td>
<td>128</td>
</tr>
</tbody>
</table>

Data used in these experiments are not extensive but focus on different topologies. We can observed that the constructive algorithm we have presented in this paper works very well with various kinds of topologies. Our algorithm can be improved, especially at step 4 where one could find other types of topologies.

Our goal was to propose a method to tackle the problem of realizing a distance matrix by a weighted graph. As far as we know, this method belongs to the very few ones which solve the problem of realizing a (non-tree) distance matrix by a graph.

A Appendix

Unless explicitly given, all edges are assumed to have a unit length. The metric is induced by the black circles.
triangle

starN (N=4,5,6)

H

PolygonN : polygon on N vertices

2pentagons

2squares : an example presented by Simoões-Pereira in [16]

4triangles

cube2

Hak64 : an example presented by Hakimi and Yau in [1]

Imr64 : an example presented by Imrich, Simoões-Pereira and Zamfirescu in [39]

---

2 In [1] the graph is slightly different since it is not optimal
References


