An Introduction to Clique Minimal Separator decomposition

Anne Berry¹  Romain Pogorelcnik¹  
Geneviève Simonet²

Research Report LIMOS/RR-10-03

January 15 2010

¹LIMOS UMR CNRS 6158, Ensemble Scientifique des Cézeaux, F-63 173 Aubière, France, berry@isima.fr, pogorelc@isima.fr
²LIRMM, 161 Rue Ada, F-34392 Montpellier, France, simonet@lirmm.fr
Abstract

This paper is a review which presents and explains the decomposition by clique minimal separators, which is a promising tool for representing and decomposing experimental data. The pace is leisurely, we give many examples and figures. Easy algorithms are provided to implement this decomposition. The historical and theoretical background is given, as well as sketches of proofs of the structural results involved.

Keywords: graph decomposition; clique minimal separator; minimal triangulation
1 Introduction

This paper is designed as a presentation for the not so well-known graph decomposition which we call "clique minimal separator decomposition". In view of recent applications, this decomposition is a promising tool.

A separator is a set of vertices (of a connected graph), the removal of which disconnects the graph, and a clique is a set of pairwise adjacent vertices, and of course a clique separator is a separator which is a clique. Clique minimal separator decomposition is a process which breaks up an undirected graph into a set of subgraphs by copying the clique minimal separators into the connected components they defined, as will be explained in detail further.

Historically, clique separator decomposition was studied by graph theorists because some hard graph problems such as vertex coloring, maximum clique and testing for perfection, could be solved by a Divide-and-Conquer approach by first solving them on the subgraphs defined by a clique separator decomposition, and then merging the obtained results.

There are 2 main drawbacks to this decomposition:

- First, there is not necessarily a clique separator in an arbitrarily chosen graph.
- Second, the subgraphs obtained are not disjoint, which means that if the overlap is large, little is to be gained by a Divide-and-Conquer approach.

However, recent research has modelized data by graphs (such as text mining data [9], biochip data [14], or any data represented by a symmetric positive matrix such as distance data). In those graphs, a clique is often an important locus, since it reflects maximum interaction between its elements. These graphs almost always have clique separators. But in any case, when these graphs are obtained by choosing a threshold in a distance matrix, one can choose a threshold which defines a graph which contains a sufficient number of clique separators.

The fact that the subgraphs obtained overlap can be a desirable property in applications. For example, when searching for clusters of genes, it is nice to be able to allow a gene to interact with different gene groups, which is not the case with classical gene clustering.

The theory of clique decomposition and clique minimal separator decomposition has been studied by several authors. The mathematical baggage necessary to proving the results we present here is somewhat complicated, and the results are distributed between various papers, which is one of the
reasons for which this graph decomposition is not as well-known as it could be. We will describe the known results (while citing the corresponding reference papers when possible). We also devote a section to giving the proof (or at least a sketch of proof) of the main results we use.

Our aim in this paper is to explain this decomposition as simply as possible; the basic process is not difficult to understand, and it is not difficult to implement. However, it does involve some not so easy graph notions such as minimal separation. We have organized the results so that graph theorists will find the theoretical background, and so that scientists from other areas can use and implement this decomposition, by giving examples and explaining the implementations.

The paper is organized as follows:

- Section 2 gives some general graph notions, some particulars on minimal separation and minimal triangulation, and explains how a distance matrix defines a family of graphs.

- Section 3 provides a precise definition of clique minimal separator decomposition, as well as examples; we also discuss how to find the clique minimal separators of a graph efficiently, and give some properties of this decomposition.

- Section 4 gives an efficient process for computing the clique minimal separator decomposition of a graph.

- Section 5 provides a brief history of clique decomposition, with the bibliographic background as well as high-level proofs of the results we use, presented in this paper.

- We conclude in Section 6.

## 2 Graph Notions

### 2.1 General notions

All graphs in this work are undirected and finite. A graph $G = (V, E)$ is a set $V$ of vertices, $|V| = n$, and a set $E$ of edges which are pairs of vertices, $|E| = m$; an edge $\{x, y\}$ is denoted $xy$. The complement $\overline{G} = (V, \overline{E})$ of graph $G = (V, E)$ is the graph where, for any pair of vertices $\{x, y\}$, $x \neq y$, edge $xy \in \overline{E}$ if and only if $xy \notin E$. If $X \subset V$ is a set of vertices of $G = (V, E)$, $G(X)$ denotes the subgraph induced by $X$ (which is a graph whose vertices are the elements of $X$ and such that $xy$ is an edge of $G(X)$ if $x, y \in X$ and
$xy \in E$).

**Neighborhoods:** The **neighborhood** of a vertex $x$ in $G$ is $N_G(x) = \{y \neq x \mid xy \in E\}$; we omit subscript $G$ when it is clear from the context which graph we work on. The extended neighborhood of a vertex $x$ is $N_G[x] = N_G(x) \cup \{x\}$. We say that a vertex $x$ **sees** another vertex $y$ if and only if $xy \in E$, or that $x$ and $y$ are adjacent. The neighborhood of a set of vertices $A$ is $N_G(A) = \bigcup_{x \in A} N_G(x) - A$, and its extended neighborhood is $N_G[A] = N_G(A) \cup A$.

**Paths and cycles:** A **path** is a sequence $(x_1, x_2, \ldots, x_k)$ of different vertices such that for $i < k$, $x_i$ sees $x_{i+1}$. A **chord** in a path is an edge $x_i x_j$ where $j > i$ and $j \neq i + 1$. A path is **chordless** if it has no chord. A **cycle** is a 'closed path', i.e. a sequence $x_1, x_2, \ldots, x_k, x_1$ of vertices, with $x_1, x_2, \ldots, x_k$ all different, such that for $i < k$, $x_i$ sees $x_{i+1}$. A **chordless cycle** is a cycle with no chord. A chordless cycle with 5 or more vertices is called a **hole**, the complement of such a cycle is called an **antihole**.

**Connexity:** A graph is **connected** if there is a path connecting any pair of vertices, disconnected otherwise; when the graph is disconnected, the maximal connected subgraphs are called the **connected components** of the graph.

**Cliques:** A **clique** is a set of vertices which are pairwise adjacent. An **independent set** is a set of vertices with no edges.

**Perfection:** A graph is **perfect** if it contains hole or antihole with an odd number of vertices.

**Example 2.1** In Figure 1, $(a, b, c, k)$ is a chordless cycle of length 4, \{c, d, j, k\} is a clique, $N(c) = \{b, d, j, k\}$, $N(\{a, b\} = \{c, k\}$.

![Figure 1: Graph $G$, our running example.](image-url)
2.2 Minimal separation, chordal graphs, and minimal triangulation

Definition 2.2

- A subset $S$ of vertices of a connected graph $G$ is called a separator (or sometimes a cutset) if $G(V - S)$ is not connected.

- A separator $S$ is called an $ab$-separator if $a$ and $b$ lie in different connected components of $G(V - S)$, a minimal $ab$-separator if $S$ is an $ab$-separator and no proper subset of $S$ is an $ab$-separator.

- A separator $S$ is a minimal separator, if there is some pair $\{a, b\}$ such that $S$ is a minimal $ab$-separator.

Alternately, $S$ is an $xy$-minimal separator if and only if $G(V - S)$ has at least 2 connected components $C_1$ and $C_2$ such that $N(C_1) = N(C_2) = S$; such components are called full components of $S$ in $G$. $S$ is a minimal separator for any $\{x, y\}$ with $x \in C_1$ and $y \in C_2$.

A connected graph with no minimal separator is a clique.

When the graph is not connected, the set of minimal separators is the union of the sets of minimal separators of the connected components.

Example 2.3 Let us illustrate these notions on a tree: a tree has two kinds of vertices: leaves and articulation nodes (which are its minimal separators). Figure 2 shows how removing articulation node $\{a\}$ disconnects the graph into several connected components; $\{a\}$ is a $df$-separator, as $d$ and $f$ lie in two different components.

![Image](image_url)

Figure 2: $\{a\}$ is a $df$-separator.

Example 2.4 On the more complex graph $G$ of Figure 1: $\{c, d, j, k\}$ is a separator, but it is not minimal. $\{d, j, k\}$ is a minimal separator, with full components $\{a, b, c\}$ and $\{e, f, g\}$. $\{j, k\}$ is also a minimal separator, with full components $\{h, i\}$ and $\{a, b, c, d, e, f, g\}$. Note that minimal separator $\{j, k\}$ is included in minimal separator $\{d, j, k\}$.
Note that in an arbitrary graph, there can be an exponential number of minimal separators ($O(2^n)$). Figure 3 shows an example of this.

Figure 3: This graph has an exponential number of minimal separators, which are the combinations from the Cartesian product $\{a_1, b_1\} \times \{a_2, b_2\} \times \ldots \times \{a_n, b_n\}$.

**Chordal graphs:** A graph is chordal if it has no chordless cycle of length more than 3; chordal graphs are extensions of trees and have many similar properties. Dirac [10] originally defined the concept of a minimal separator in order to characterize chordal graphs, by showing that a graph $G$ is chordal if and only if every minimal separator of $G$ is a clique. In a chordal graph, there are less than $n$ minimal separators [18]. Figure 4 shows a chordal graph $H$.

Figure 4: Chordal graph $H$, with minimal separators: $\{b, k\}, \{c, k\}, \{d, j, k\}, \{j, k\}, \{i, j\}, \{d, g\}$.

**Perfect elimination orderings:** Another interesting characterization of chordal graphs is that they have a simplicial ordering on the vertices: a vertex is said to be simplicial if its neighborhood is a clique. A graph is chordal if and only if one can repeatedly find a simplicial vertex, and remove it, until no vertex is left. Since at each step a vertex is removed, this defines a series of transitory graphs: at step $i + 1$, the transitory graph is input graph...
G from which the first i vertices have been removed; of course, the simplicial vertex is found in this transitory graph. This process is called a simplicial elimination scheme, and defines an ordering on the vertices called a perfect elimination ordering (peo).

Each minimal separator of a chordal graph H is the neighborhood of a vertex in the transitory graph in the simplicial elimination scheme [18]. We will say that such a vertex generates a minimal separator of H (w.r.t. a given peo of H):

**Definition 2.5** Let H = (V, E) be a chordal graph, let α be a perfect elimination ordering of H. We say that vertex x generates minimal separator S w.r.t. α if the set of neighbors of x (in the transitory graph induced by α) which have a number higher than that of x is equal to S.

Thus, to compute the set of minimal separators of H, it is sufficient to compute a peo of H and to select the vertices that generate a minimal separator of H w.r.t. α. This is easy to do using a special kind of peo defined by algorithms such as MCS [6]. We will use such a peo for our implementation in Section 4.

**Minimal triangulations:** A triangulation is a chordal completion of a graph: a set F of edges (called the fill edges) is added to the graph in order to obtain a chordal graph.

**Definition 2.6** Let G = (V, E) be a graph. A chordal graph H = (V, E + F) is called a triangulation of G, and F is called the fill. The triangulation is said to be:

- minimal if for no proper subset F' of F, H' = (V, E + F') is chordal.
- minimum if no other minimal triangulation has less fill edges.

**Example 2.7** Graph H from Figure 4 is a minimal triangulation of Graph G from Figure 1 (see Figure 5).

Computing a minimum triangulation is NP complete [23], but computing a minimal triangulation can be done in \(O(nm)\) time (or a little less for dense graphs [15, 13]). This problem has given rise to several recent papers, so there are now many minimal triangulation algorithms (see e.g. [2], [3], [4], and [12]). We will use one of them (MCS-M) in Section 4 where we describe an implementation.
Minimal elimination orderings: One way to compute a triangulation is to force the graph into having a perfect elimination ordering: repeatedly pick a vertex and add any edges missing in its neighborhood (in the transitory graph) and remove it, until no vertex is left. This will yield a triangulation whose fill is the set of all edges added in this process.

A minimal triangulation is obtained when using a special kind of ordering on the vertices for this, called a minimal elimination ordering (meo). An efficient algorithm has been devised for doing this: LEX M [19]. In Section 4, we will use a simpler version of LEX M, MCS-M [2]. Both these algorithms yield a minimal triangulation \( H \) of the input graph \( G \) and an ordering \( \alpha \) that is both a meo of \( G \) and a peo of \( H \), and makes it easy to compute the set of vertices generating the minimal separators of \( H \).

2.3 Defining a family of graphs using a distance matrix

If we are given a distance matrix \( M \) on a set \( \mathcal{A} \) of objects, which is often the case in applications, we can define a graph by choosing a threshold \( \theta \): \( G_\theta = (V, E_\theta) \), with \( V = \mathcal{A} \), and for \( x, y \in \mathcal{A}, xy \in E_\theta \) if \( d(x, y) \leq \theta \). If we scan the entire set of different values \( v_1 < \ldots < v_k \) of \( M \) in increasing order, this defines a family of graphs \( G_{v_1} \subset \ldots \subset G_{v_k} \), which we call the threshold family of graphs associated with \( M \).

A simple example below gives a matrix, as well as the graphs defined by the different thresholds: 6, 8, 10 and 12.

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0</td>
<td>6</td>
<td>12</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>b</td>
<td>6</td>
<td>0</td>
<td>12</td>
<td>12</td>
<td>16</td>
</tr>
<tr>
<td>c</td>
<td>12</td>
<td>12</td>
<td>0</td>
<td>6</td>
<td>10</td>
</tr>
<tr>
<td>d</td>
<td>12</td>
<td>12</td>
<td>6</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>e</td>
<td>16</td>
<td>16</td>
<td>10</td>
<td>8</td>
<td>0</td>
</tr>
</tbody>
</table>
Figure 6: Graphs $G_6 \subset G_8 \subset G_{10} \subset G_{12} \subset G_{16}$. $G_0$ is an independent set and $G_{16}$ is a clique.

The example below shows graphs obtained for various thresholds of a distance matrix obtained from biochip data on yeast *Saccharomyces cerevisiae* [14].

![Figure 7: Graphs at various thresholds of a distance matrix from biochip data [14].](image)

---

3 Defining clique minimal separator decomposition

3.1 Definitions and examples

The decomposition by clique minimal separators can be defined by the following algorithmic process on input graph $G$: define a collection of subgraphs (which at the beginning contains only $G$) by repeatedly applying the following decomposition step on a subgraph which has a clique minimal separator, until none of the subgraphs has a clique minimal separator.
Decomposition Step 3.1 Given a graph $G'$, find a clique minimal separator $S$ of $G'$ and a full component $C$ of $S$; replace $G'$ with the following 2 subgraphs: $G_1 = G'(S \cup C)$, $G_2 = G'(V - C)$.

The decomposition is the set of subgraphs obtained in the end, which are called atoms.

The decomposition is unique, the resulting set of atoms does not depend on the order in which the decomposition steps are executed. This is because of the following property of clique minimal separator decomposition:

Property 3.2 Let $\mathcal{S}$ be the set of minimal separators of a graph $G'$, let $S$ be a clique minimal separator of $G'$; after application of one decomposition step 3.1, each other minimal separator is a minimal separator of either $G'_1$ or of $G'_2$.

Thus in particular, any clique minimal separator will be a clique minimal separator of $G'_1$ or of $G'_2$. Note that the clique minimal separator which was used, $S$, may or may not still be a minimal separator of $G'_2$, depending on whether $S$ had more than two or only two full components.

The atoms obtained are characterized as follows:

Characterization 3.3 The clique minimal separator decomposition of a graph is the set of subgraphs which are connected, containing no clique minimal separator, and are maximal for both these properties.

Example 3.4

- Figure 8 gives the decomposition of a tree.

- On any chordal graph, this decomposition yields the set of maximal cliques of the input graph.

- In a non-chordal graph, some of the atoms will not be cliques. Let us use our graph $G$ from Figure 1 as an example: The clique minimal separators of $G$ are: $\{ c, k \}$, $\{ j, k \}$, $\{ d, j, k \}$. Figure 9 shows a decomposition tree and the set of atoms.

- Figure 10 shows an example from [14], where biochip data is represented by a graph by choosing a threshold; the graph is then decomposed into its atoms.
Figure 8: (a) A decomposition step using minimal separator \( \{a\} \). The minimal separators (except for \( \{a\} \)) are partitioned into the two subtrees obtained. (b) Total decomposition.

Figure 9: A decomposition tree for graph \( G \) from Figure 1 and the set of atoms of \( G \).

When the atoms are defined by thresholds from a distance matrix, as shown in Figure 7, one has to choose a threshold. If the graph is insufficiently connected, as for thresholds .5 and .75, there are too many isolated vertices (vertices with no neighbor), which are not very informative. On the other hand, if the graph is too highly connected, the clique minimal separators tend to be very large and to have an overlap which is too large. Threshold 1.25 was chosen, along with other criteria, as a good compromise. Figure 10 below shows the atoms obtained for this threshold.
3.2 Properties of the atoms

- The number of atoms is at most $n$.

- The intersection between atoms is always a clique (which may be empty), and it is not necessarily a minimal separator (even when it is not empty).

**Example 3.5** In our example from Figure 9, atom $\{a, b, c, k\} \cap atom \{c, d, j, k\} = \{c, k\}$, which is not a minimal separator; atom $\{a, b, c, k\} \cap atom \{i, j, k, h\} = \{j, k\}$, a minimal separator.

3.3 An equivalent process

Another way of doing the decomposition is using a clique minimal separator $S$ and copying it in all the components it defines. For a component $C$ which is not a full component, we need to copy only the part of $S$ which sees $C$. This corresponds to a clique minimal separator which is properly included in $S$.

The property we use is the following:

**Property 3.6** Let $S$ be a minimal separator, let $(C_i)$ be connected components of $G(V - S)$. Then $\forall i$, $N_G(C_i)$ is a minimal separator of $G$.

This leads to the following decomposition step:

**Decomposition Step 3.7** Given a graph $G'$ with a clique minimal separator $S$, let $C_1,...C_k$ be the connected components of $G'(V - S)$. Replace $G'$ with the subgraphs $G'(C_i \cup N_{G'}(C_i))$.

After such a decomposition step using clique minimal separator $S$, $S$ and the clique minimal separators subsets of $S$ can no longer be a minimal separator in any of the subgraphs obtained, and all the other minimal separators of $G$ are partitioned into the subgraphs obtained.
Example 3.8 Figure 11 shows a decomposition step in graph $G$ from Figure 1 using Decomposition Step 3.7.

![Diagram showing decomposition steps]

Figure 11: A decomposition of $G$ using Decomposition Step 3.7

An application of Decomposition Step 3.7 corresponds to several applications of Decomposition Step 3.1.

3.4 How to compute the clique minimal separators

The only efficient known way to compute the clique minimal separators of a graph is to extract them from a minimal triangulation.

This is because of the following property:

Property 3.9 Let $G = (V, E)$ be a graph, let $H = (V, E + F)$ be a minimal triangulation of $G$. The clique minimal separators of $G$ are exactly the minimal separators of $H$ that are cliques in $G$.

Thus in order to find the clique minimal separators of a graph, the general process is:

- compute a minimal triangulation $H$ of $G$.
- compute the minimal separators of $H$.
- check each minimal separator of $H$ to see whether it is a clique in $G$.

Computing the minimal separators of a minimal triangulation of a graph can be done efficiently using an ordering provided by a search algorithm such as MCS-M, and Steps 1 and 2 can be merged, as we will see in Section 4.

Example 3.10 Graph $H$ from Figure 4 is a minimal triangulation of Graph $G$ from Figure 1. The minimal separators of $H$ are: 
\{b, k\}, \{c, k\}, \{d, j, k\}, \{j, k\}, \{i, j\}, \{d, g\}. Three of them are cliques in $G$: \{c, k\}, \{d, j, k\}, \{j, k\}, so they are clique minimal separators of $G$. 

13
3.5 Some problems which can be solved using the atoms

- Minimum triangulation: if a minimal triangulation is computed for each of the atoms of the clique minimal separator decomposition of a graph, then the union of the set of fill edges obtained defines a minimal triangulation for $G$. Thus if for each atom a minimum-sized fill is computed, the resulting fill will be a minimum triangulation of $G$, since the sets of fill edges in the atoms are pairwise disjoint.

- Perfection: Any chordless cycle of length 4 or more is preserved by a decomposition step, as well as any antihole, so this decomposition preserves holes and antiholes. Thus a graph is perfect if and only if all its atoms are perfect.

- Coloring: An optimal coloring is obtained by merging optimal colorings of the atoms.

- Maximum clique: Any maximal clique of the graph is preserved after a decomposition step, so finding the size of the largest clique of each atom will yield the size of the largest clique of the original graph.

4 Algorithms and implementations

There are several ways of computing the decomposition of a graph into atoms. The simplest is to implement Decomposition Step 3.1. We will use Algorithm MCS-M [2], a search algorithm which computes a minimal triangulation $H = (V, E + F)$ of the input graph $G = (V, E)$, represented by a set $F$ of fill edges, as well as an ordering $\alpha$ of the vertices. MCS-M numbers the vertices from $n$ to 1, and the ordering $\alpha$ obtained is a meo of the input graph $G$ and a peo of the minimal triangulation $H$ which is computed [5].

Once this ordering and this triangulation are computed, we will use them in a second algorithm, (Algorithm Atoms), which will generate the atoms by scanning the vertices from 1 to $n$.

We will start by giving the general MCS-M algorithm, then give an expanded version, which we call MCS-M+, as will be explained afterwards, then go on to give the algorithm which yields the atoms.

$G'$ will be the transitory elimination graph obtained from a copy of $G$ by eliminating the vertices which have been numbered.
Algorithm MCS-M

**input**: An undirected graph \( G = (V, E) \).

**output**: A minimal elimination ordering \( \alpha \) on \( V \) and a minimal triangulation \( H = (V, E + F) \) of \( G \).

**init**: \( F \leftarrow \emptyset; G' \leftarrow G \); Initialize the labels of all vertices as 0 ;

for \( i = n \) downto 1 do

Choose a vertex \( x \) of \( G' \) of maximal label ; \( Y \leftarrow N_{G'}(x) \);

foreach vertex \( y \) of \( G' \) not belonging to \( N_{G'}[x] \) do

if there is a path from \( x \) to \( y \) in \( G' \) such that every internal vertex on the path has a label strictly smaller than \( label(y) \) then

\[ Y \leftarrow Y + \{y\} \]

foreach \( y \) in \( Y \) do

\[ F \leftarrow F + \{xy\}; label(y) \leftarrow label(y) + 1 \];

\( \alpha(i) \leftarrow x \); Remove \( x \) from \( G' \);

\( H \leftarrow (V, E + F) \);

We will now expand this algorithm in two ways.

First, we will implement "if there is a path from \( x \) to \( y \) in \( G' \) such that every internal vertex on the path has a label strictly smaller than \( label(y) \)".

We do this in a similar way as in the implementation of LEX M given in [19] by a single search in \( G' \). For each label value \( j \), set \( \text{reach}(j) \) contains the reached vertices having label \( j \), as well as the vertices having a label strictly smaller than \( j \) reached from a vertex having label \( j \).

Second, we will compute the set \( X \) of vertices that generate the minimal separators of the triangulation \( H \), as described in [6]. The idea behind this is that as long as the labels of the chosen vertices keep getting larger, we are inside a clique of \( H \); when suddenly the label of the chosen vertex \( x \) stops getting larger, \( x \) generates a minimal separator of \( H \). Each such vertex \( x \) is added to set \( X \).
Algorithm MCS-M+

input: An undirected graph $G = (V, E)$.

output: A minimal elimination ordering $\alpha$ on $V$, a minimal triangulation $H = (V, E + F)$ of $G$, and the set $X$ of vertices which generate a minimal separator of $H$.

init: $F \leftarrow \emptyset$; $G' \leftarrow G$ ;
Initialize the labels of all vertices as 0 ; $s \leftarrow -1$ ; $X \leftarrow \emptyset$ ;

for $i = n$ downto 1 do
  Choose a vertex $x$ of $G'$ of maximal label ; $Y \leftarrow N_{G'}(x)$ ;
  if $\text{label}(x) \leq s$ then
    $X \leftarrow X + \{x\}$ ;
    $s \leftarrow \text{label}(x)$ ;
  Mark $x$ reached and mark all other vertices of $G'$ unreached ;
  for $j = 0$ to $n - 1$ do
    $\text{reach}(j) \leftarrow \emptyset$
    foreach $y$ in $N_{G'}(x)$ do
      Mark $y$ reached;
      Add $y$ to $\text{reach}(\text{label}(y))$;
  for $j = 0$ to $n - 1$ do
    while $\text{reach}(j) \neq \emptyset$ do
      Remove a vertex $y$ from $\text{reach}(j)$ ;
      foreach $z$ in $N_{G'}(y)$ do
        if $z$ is unreached then
          Mark $z$ reached;
          if $\text{label}(z) > j$ then
            $Y \leftarrow Y + \{z\}$;
            Add $z$ to $\text{reach}(\text{label}(z))$;
          else
            Add $z$ to $\text{reach}(j)$;
      endforeach
    endforeach
    foreach $y$ in $Y$ do
      $F \leftarrow F + \{xy\}$; $\text{label}(y) \leftarrow \text{label}(y) + 1$ ;
      $\alpha(i) \leftarrow x$ ; Remove $x$ from $G'$;
    $H \leftarrow (V, E + F)$.

Once we have obtained an MCS-M ordering $\alpha$, as well as the corresponding minimal triangulation $H$ of $G$ and the set $X$ of generators of the minimal
separators of $H$, we run through the vertices from 1 to $n$. We use the transitory graphs $G'$ and $H'$, initialized as $G$ and $H$ respectively. At each step $i$ processing vertex $x = \alpha(i)$, we check whether $x$ is in the set $X$. If it is, the neighborhood $N_{H'}(x)$ of $x$ in $H'$ is a minimal separator of $H$; we check whether it is a clique in $G$. If it is, then $S$ is a clique minimal separator of $G$. In that case we compute the connected component $C$ of $G(V - S)$ which contains $x$; $G'(S \cup C)$ is an atom, and is stored as such; $C$ is then removed from $G'$. In any case, we then remove $x$ from $H'$.

Algorithm Atoms

**input**: A graph $G = (V, E)$, a meo $\alpha$ of $G$, the corresponding minimal triangulation $H = (V, E + F)$ of $G$, the set $X$ of vertices that generate a minimal separator of $H$.

**output**: The set $A$ of atoms of $G$, the set $\mathcal{S}_H$ of minimal separators of $H$, the set $\mathcal{S}_c$ of clique minimal separators of $G$.

\[
G' \leftarrow G; \quad H' \leftarrow H; \quad A \leftarrow \emptyset; \quad \mathcal{S}_H \leftarrow \emptyset; \quad \mathcal{S}_c \leftarrow \emptyset;
\]

for $i = 1$ to $n$ do

\[
x \leftarrow \alpha(i);
\]

if $x \in X$ then

\[
S \leftarrow N_{H'}(x); \quad \mathcal{S}_H \leftarrow \mathcal{S}_H \cup \{S\};
\]

if $S$ is a clique in $G$ then

\[
\mathcal{S}_c \leftarrow \mathcal{S}_c \cup \{S\};
\]

$C$ $\leftarrow$ the connected component of $G' - S$ containing $x$;

$A \leftarrow A + \{G'(S \cup C)\}$; $G' \leftarrow G' - C$;

Remove $x$ from $H'$;

$A \leftarrow A + \{G'\}$.

5 Theoretical and bibliographical background

5.1 A brief history of clique minimal separator decomposition


1980. The problem of finding a clique separator in an arbitrary graph was addressed by Whitesides [22], who presented an $O(n^3)$ algorithm to find
one clique separator.

1983. Tarjan [20] addressed the problem of finding a decomposition of an arbitrary graph using its clique separators. He noted that using Whitesides’ algorithm, this would require $O(nm^3)$ time. He proposed an $O(nm)$ time algorithm to do this, by showing that no fill edge of a minimal triangulation can join two connected components defined by a clique separator. Computing an meo required $O(nm)$ time, as he showed in [19] (1976), and he proved that using an meo, a clique separator decomposition can be computed in $O(nm)$ time. Tarjan left open the question of defining a unique clique separator decomposition.

1990. Dahlhaus, Karpinsky and Novick [8] proposed a parallel algorithm for clique separator decomposition, and mentioned that they ”assume that each clique separator is a minimal clique separator, because this results in smaller separators and fewer atoms”, but this point is not proved.

1990. Concurrently, Leimer [16], in a paper sent out for publication in 1986, describes in every mathematical detail how to obtain an optimal and unique decomposition using the clique minimal separators of the graph. The paper is very thorough, but unfortunately the notations are complicated and the paper is difficult to read. Leimer also describes how to use LEX M (also called the RTL algorithm) to find a clique minimal separator decomposition in $O(nm)$ time.

5.2 Computing the clique minimal separators of a graph

We first prove that for any graph $G$ and any minimal triangulation $H$ of $G$, the clique minimal separators of $G$ are exactly the minimal separators of $H$ that are cliques in $G$ (property 3.9).

For this, it is sufficient to show that any clique of $G$ has the same number of full components in $G$ as in $H$. This follows from the following property, which is an extension of Lemma 1 from [20].

**Property 5.1** Let $G = (V, E)$ be a graph, let $H = (V, E + F)$ be a minimal triangulation of $G$ and let $S$ be a clique of $G$. Then $G(V - S)$ and $H(V - S)$ have the same connected components, with the same neighborhoods.

**Proof:** Let $H'$ be the graph defined by $H' = (V, E + F')$, where $F'$ is the set of edges in $F$ that are contained in $N_G[C]$ for some connected component $C$ of $G(V - S)$. By definition of $H'$, $G(V - S)$ and $H'(V - S)$ have the same connected components, with the same neighborhoods. Let us show that $H'$ is chordal. Let $\mu$ be a cycle in $H'$ of length more than 3, and let us show that $\mu$ has a chord in $H'$. As $H$ is chordal, $\mu$ has a chord, say $xy$, in $H$. If $xy$ is in
$H'$ then it is a chord of $\mu$ in $H'$. Otherwise there is a connected component $C$ of $G(V - S)$ such that $x$ is in $C$ and $y$ is in $V - N_G[C]$, or conversely. Hence there are two non-consecutive vertices $w$ and $z$ of $\mu$ that are both in $N_G(C)$, and therefore both in clique $S$ of $H'$. It follows that $wz$ is a chord of $\mu$ in $H'$, which completes the proof that $H'$ is chordal. As $G \subseteq H' \subseteq H$, with $H'$ chordal and $H$ a minimal triangulation of $G$, $H' = H$, and therefore $G(V - S)$ and $H(V - S)$ have the same connected components, with the same neighborhoods.

Corollary 5.2 (Property 3.9) Let $G = (V, E)$ be a graph, let $H = (V, E + F)$ be a minimal triangulation of $G$. The clique minimal separators of $G$ are exactly the minimal separators of $H$ that are cliques in $G$.

The fact that each minimal separator of a chordal graph $H$ is generated by a vertex of $H$ w.r.t. a peo of $H$ [18] is well-known. A way of selecting the exact vertices that generate a minimal separator is discussed in [6], and the following algorithm is presented, based on the expanded version of MCS presented in [7] to generated the maximal cliques and a clique tree.

Algorithm MCS-Minseps

**input** : A chordal graph $H = (V, E)$.

**output** : The set $S$ of minimal separators of $H$.

**init** :

$V_{NUM} \leftarrow \emptyset$; $H_{NUM} \leftarrow H(V_{NUM})$; $H_{ELIM} \leftarrow H$; $S \leftarrow \emptyset$;

Initialize the labels of all vertices as $0$; $s \leftarrow -1$;

for $i = n$ downto 1 do

Choose a vertex $x$ of $H_{ELIM}$ of maximum label;

$V_{NUM} \leftarrow V_{NUM} + \{x\}$; $H_{NUM} \leftarrow H(V_{NUM})$;

if $\text{label}(x) \leq s$ then

$S \leftarrow S \cup \{N_{H_{NUM}}(x)\}$;

$s \leftarrow \text{label}(x)$;

foreach $y \in N_{H_{ELIM}}(x)$ do

$\text{label}(y) \leftarrow \text{label}(y) + 1$;

Remove $x$ from $H_{ELIM}$;

This process is integrated in the implementation given in Section 4.
5.3 Decomposing a graph into atoms

Decomposition Step 3.1 and Algorithm Atoms of Section 4 are largely inspired from the decomposition of a graph by clique separators described by Tarjan [20]. Tarjan considers clique separators instead of clique minimal separators, with the drawback that the set of atoms obtained is not unique (see Subsection 5.4). He also computes a meo of \( G \) and the corresponding minimal triangulation of \( G \), but he selects the vertices that generate a clique separator instead of a clique minimal separator of \( G \). The proof of the clique separator decomposition algorithm given in [20] can easily be modified into a proof of Algorithm Atoms. We will give an idea of this proof.

**Theorem 5.3** Algorithm Atoms computes a clique minimal separator decomposition of input graph \( G \).

**Proof:** (idea of the proof) The proof works by induction on \( k = |X'| \), where \( X' \) is the set of vertices of \( X \) that generate a clique of \( G \) w.r.t. \( \alpha \), i.e. the set of vertices of \( G \) that generate a clique minimal separator of \( G \) w.r.t. \( \alpha \). If \( k = 0 \) then the computed decomposition is \( \{G\} \), which is correct since \( G \) has no clique minimal separator. We suppose that the decomposition is correct if \( |X'| \leq k \). Let us show that it is still correct if \( |X'| = k + 1 \). Let \( x \) be the first vertex of \( X' \) in ordering \( \alpha \), let \( S \) be the clique minimal separator of \( G \) generated by \( x \) w.r.t. \( \alpha \), let \( C \) be the connected component of \( G(V - S) \) containing \( x \), let \( G_1 = G(S \cup C) \) and \( G_2 = G(V - C) \). For \( j \) in \( \{1, 2\} \), let \( V_j \) be the vertex set of \( G_j \), let \( \alpha_j \) be the restriction of \( \alpha \) to \( V_j \), let \( H_j = H(V_j) \) and \( X'_j = X' \cap V_j \). It can be shown that for \( j \) in \( \{1, 2\} \), \( \alpha_j \) is a meo of \( G_j \), with \( H_j \) as corresponding minimal triangulation of \( G_j \). Moreover, each vertex of \( V_2 \) generates the same set in \( G_2 \) w.r.t. \( \alpha_2 \) as in \( G \) w.r.t. \( \alpha \), which is a minimal separator of \( H_2 \) if and only if it is a minimal separator of \( H \). Hence \( X'_2 \) is exactly the set of vertices of \( G_2 \) that generate a a clique minimal separator of \( G_2 \) w.r.t. \( \alpha_2 \), with \( |X'_2| \leq k \) since \( X'_2 \subseteq X' - \{x\} \). It follows by induction hypothesis that \( G_2 \) is correctly decomposed. It remains to show that \( G_1 \) is an atom of \( G \). It can be shown that each vertex of \( C - \{x\} \) precedes \( x \) in ordering \( \alpha \), and generates the same set in \( G_1 \) w.r.t. \( \alpha_1 \) as in \( G \) w.r.t. \( \alpha \), which is a minimal separator of \( H_1 \) if and only if it is a minimal separator of \( H \). As no vertex of \( C - \{x\} \) generates a clique minimal separator of \( G \) w.r.t. \( \alpha \) (since \( x \) is the first vertex that does), no vertex of \( C - \{x\} \) generates a clique minimal separator of \( G_1 \) w.r.t. \( \alpha_1 \) either. Now each vertex of \( S \cup \{x\} \) generates a subset of \( S \), which is not a separator of \( G_1 \), w.r.t. \( \alpha_1 \). Hence \( G_1 \) has no clique minimal separator, and is therefore an atom of \( G \). \( \square \)
5.4 The unicity of clique minimal separator decomposition

Tarjan [20] originally defined the decomposition step of a graph by a clique separator as follows:

**Decomposition Step 5.4** Let $S$ be a clique separator of $G' = (V', E')$, let $(S, A, B)$ be a partition of $V'$ such that no vertex in $A$ is adjacent to a vertex in $B$. Replace $G'$ by $G_1' = G'(S \cup A)$ and $G_2' = G'(S \cup B)$.

He remarked that the set of atoms obtained by repeatedly applying this decomposition step is not unique, giving a counterexample that is recalled in Figure 12, and left the problem of the unicity of the decomposition open.

Leimer [16] solved this problem, showing that using clique minimal separators instead of clique separators ensures unicity of the decomposition (with the additional condition that each one of $A$ and $B$ contains a full component of $S$ in $G'$ in the case of Decomposition Step 5.4). We present some elements of the proof of this result, which is associated in [16] with proofs of more complex results which make it difficult for non-specialists to follow. We first recall some definitions given in [16].

**Definition 5.5** A graph is prime if it is connected and has no clique separator (or equivalently, no clique minimal separator).

A mp-subgraph of a graph $G$ is a maximal prime subgraph of $G$.

Leimer proved that the clique minimal separator decomposition of a graph $G$ is the set of mp-subgraphs of $G$ (Characterization 3.3). The proof of this characterization relies on the following property.
Property 5.6 Let $G = (V, E)$ be a graph, let $S$ be a clique of $G$ and let $C$ be a full component of $S$ in $G$. Then there is a subset $A$ of $V$ such that $S \subseteq A$ and $G(A)$ is prime.

Proof: (idea of the proof) The property holds if $G$ is chordal since in that case, as $S$ is a clique of $G$ and $C$ is a full component of $S$ in $G$, there is a vertex $x$ of $C$ such that $S \subseteq N_G(x)$ (this follows from the results in [18]), and therefore $S \cup \{x\}$ is a clique and induces a prime graph.

Let us show that it holds for any graph $G$. Let $G^*$ be the graph obtained from $G$ by making each mp-subgraph of $G$ into a clique. $G^*$ is chordal and has the same mp-subgraphs as $G$ [16]. There is a subset $A$ of $V$ such that $S \subseteq A$ and $G^*(A)$ is prime. Let $B$ be a subset of $V$ such that $A \subseteq B$ and $G^*(B)$ is a mp-subgraph of $G^*$. Then $S \subseteq B$ and $G(B)$ is prime. □

He also showed the following characterization, from which we can deduce Characterization 3.3:

Characterization 5.7 The clique minimal separator decomposition of graph $G$ is the set of mp-subgraphs of $G$.

Proof: We suppose that the decomposition step which is used is Decomposition Step 5.4 with the additional condition that each one of $A$ and $B$ contains a full component of $S$ in $G'$, which is more general than Decomposition Step 3.1.

The proof goes by induction on the number $n$ of vertices of the graph. The property trivially holds if $n = 1$. Suppose that it holds for any graph with at most $n$ vertices. Let us show that it holds for a graph $G = (V, E)$ with $n + 1$ vertices.

The property trivially holds if $G$ has no clique minimal separator. We suppose that $G$ has a clique minimal separator $S$. Let $(S, A, B)$ be a partition of $V$ such that no vertex in $A$ is adjacent to a vertex in $B$ and each one of $A$ and $B$ contains a full component of $S$ in $G$. Let $G_1 = G(S \cup A)$ and $G_2 = G(S \cup B)$. Let us show that the set $\text{atoms}(G)$ of atoms obtained by replacing $G$ by $G_1$ and $G_2$ and further decomposing $G_1$ and $G_2$ is equal to the set $\text{mp-subgraphs}(G)$ of mp-subgraphs of $G$. Clearly $\text{atom}(G) = \text{atoms}(G_1) \cup \text{atoms}(G_2)$. As by induction hypothesis $\text{atoms}(G_1) = \text{mp-subgraphs}(G_1)$ and $\text{atoms}(G_2) = \text{mp-subgraphs}(G_2)$, it is sufficient to show that $\text{mp-subgraphs}(G) = \text{mp-subgraphs}(G_1) \cup \text{mp-subgraphs}(G_2)$.

Let us first show that $\text{mp-subgraphs}(G) \subseteq \text{mp-subgraphs}(G_1) \cup \text{mp-subgraphs}(G_2)$. Let $G(M)$ be in $\text{mp-subgraphs}(G)$. Then $M \subseteq S \cup A$ or $M \subseteq S \cup B$ (otherwise $M \cap S$ would be a clique separator of $G(M)$) and therefore $G(M) \in \text{mp-subgraphs}(G_1) \cup \text{mp-subgraphs}(G_2)$.
Let us show now that $mp$-subgraphs($G_1$) $\cup$ $mp$-subgraphs($G_2$) $\subseteq$ $mp$-subgraphs($G$).
Let $G(M) \in mp$-subgraphs($G_1$) $\cup$ $mp$-subgraphs($G_2$), say $G(M) \in mp$-subgraphs($G_1$). By Property 5.6, as $A$ contains a full component of $S$ in $G$, no subset of $S$ induces a $mp$-subgraph of $G_1$. Hence $M \cap A \neq \emptyset$. Let $G(M')$ be a $mp$-subgraph of $G$ with $M \subseteq M'$. As $mp$-subgraphs($G_1$) $\cup$ $mp$-subgraphs($G_2$), $G(M') \in mp$-subgraphs($G_1$) $\cup$ $mp$-subgraphs($G_2$), and as $M' \cap A \neq \emptyset$ (since $M \cap A \neq \emptyset$ and $M \subseteq M'$), $G(M') \in mp$-subgraphs($G_1$). As $G(M)$ and $G(M')$ are both $mp$-subgraphs of $G_1$ with $M \subseteq M'$, $M = M'$. Hence $G(M) \in mp$-subgraphs($G$). It follows that $mp$-subgraphs($G$) = $mp$-subgraphs($G_1$) $\cup$ $mp$-subgraphs($G_2$), which completes the proof by induction. $\square$

**Corollary 5.8** The decomposition by clique minimal separators is unique.

### 6 Conclusion

Clique minimal separator decomposition is simple to implement and particularly well-suited to applications involving graphs chosen from a distance matrix. Though the theoretical background is somewhat complicated, it is absolutely not necessary to master it in order to use this decomposition.

One of the promising aspects of this decomposition is that once a threshold is chosen, the "clusters" which are formed as atoms of the corresponding graph are uniquely defined, depending solely on the structure of the data.

Regarding graph visualization, the set of atoms can be organized into a meta-graph where the vertices are the atoms. For instance [14], there can be an edge between two atoms when their intersection is a clique minimal separator. This yields a more global view of the graph, while displaying some of its structural properties.

### References


