A WIDE-RANGE EFFICIENT ALGORITHM FOR MINIMAL TRIANGULATION

SODA'99 Procceedings

Anne Berry*

Abstract

Traditionally, efficient algorithms for computing a minimal triangulation of a graph (i.e. embedding a graph into a triangulated graph by adding an inclusion-minimal set of edges) required first computing a special ordering on the vertices of the graph, called a minimal ordering. We give a new algorithm which efficiently computes a minimal triangulation using an arbitrary ordering on the vertices.

1 Introduction.

Computing a minimal triangulation consists in embedding a given graph into a triangulated graph by adding a set of edges (called a fill). If the set of edges added is inclusion-minimal, the fill is said to be minimal, and the corresponding triangulated graph is called a minimal triangulation. Finding a fill that is minimum is NP-complete ([10]).

Given a graph G and any ordering α on its vertices, an associated fill can be computed by repeatedly choosing the next vertex x in order α, adding the edges necessary to make the neighborhood of x into a clique (i.e. by making x simplicial), before deleting x; this process simulates the well-known simplicial elimination scheme defined by Fulkerson and Gross in [4] for the characterization of triangulated graphs.

Finding an ordering that produces a fill that is minimal is important in many areas of computer science, as database management, etc. (see [2]). The current best-time algorithm for producing a minimal ordering and the corresponding minimal fill is LEX M, from the celebrated paper of Rose, Tarjan and Lueker ([7]), with an $O(nm)$ worst-time. Despite its complexity merits, as yet unparalleled, LEX M has been shown to yield only a restricted family of minimal orderings, which moreover produce fills which are far from minimum ([2]).

As a result, costly heuristics have been proposed for finding orderings which produce low (non-minimal) fills, and then removing extra edges, in order to obtain a fill that is minimal.

We present a very simple algorithm which, given any ordering α on the vertices, produces a fill that is minimal by adding only the necessary edges at each step, instead of making the current vertex simplicial. This process can yield any minimal triangulation, (and actually characterizes minimal triangulation), and can be implemented to run in the same $O(nm)$ time as LEX M.

*LIRMM, 161, Rue Ada, 34 392, Montpellier, FRANCE, aberry@lirmm.fr

2 Notations.

The input graph is denoted $G = (V, E)$, with $\mid V \mid= n$, $\mid E \mid= m$. The transitory graph obtained at the end of each step of the algorithm is denoted $H = (V, E + F)$, $\mid E + F \mid= m'$. $N_G[x]$ is the set of vertices adjacent to x in graph G (and contains x); for $C \subseteq V, N_G(C) = (\cup_{x \in C} N_G[x]) - C$, and for $S \subseteq V$, $\mathcal{C}_G(S)$ is the set of connected components of $G(V - C)$. S is called a *separator* if $\mid \mathcal{C}_G(S) \geq 1 \mid$, a *minimal separator* if $\exists C_1, C_2 \in \mathcal{C}_G(S) \mid N(C_1) = N(C_2) = S$.

3 Algorithm.

Algorithm Lb-Triang

```
Input: A graph $G = (V, E)$, an ordering $\alpha$ on $V$.
Output: A minimal fill-in $F$ of $G$,
        A minimal triangulation $H = (V, E + F)$ of $G$.
Initialization: $H \leftarrow G$; $F \leftarrow \emptyset$;
For each vertex $x$ in $V$ taken in order $\alpha$ do
   For each connected component $C$ in $\mathcal{C}_G(N_H[x])$ do
      Make $N_G(C)$ into a clique by adding edge set $F'$;
      $F \leftarrow F + F'$; $H \leftarrow (V, E + F)$;
```

Example. Figure 1 shows the minimal triangulation obtained by processing vertices in order $(a, b, c, d, e, f, g, h, i)$ on input graph G represented by the non-dotted edges.

Step 1: $N_H[a] = \{a, b, c, d, e\}$, $\mathcal{C}_G(N_H[a]) = \{\{f, g\}, \{h, i\}\}$, $N_G(\{f, g\}) = \{b, c\}$; fill-in edge bc is added; $N_G(\{h, i\}) = \{b, d, e\}$; fill-in edges bd, be and de are added.

Step 2: $N_H[b] = \{a, b, c, d, e, f, h\}$, $\mathcal{C}_G(N_H[b]) = \{\{g\}, \{i\}\}$, $N_G(\{g\}) = \{c, f\}$; fill-in edge cf is added; $N_G(\{i\}) = \{e, h\}$; fill-in edge eh is added. At this stage, the graph is already triangulated, and no further edge will be added during the rest of the execution.

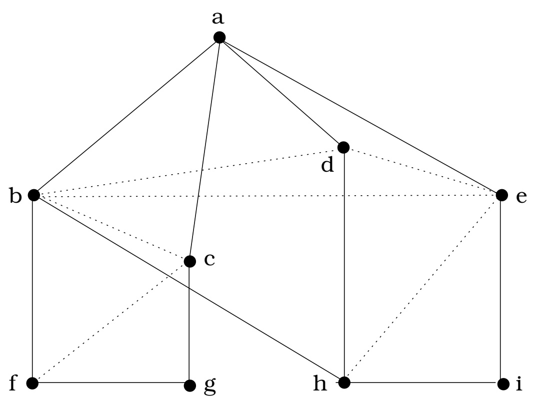

Figure 1: Graph H; fill-in edges are dotted.

4 Mechanism and complexity.

This algorithm forces an arbitrary graph into respecting a not widely known characterization for triangulated graphs given by the otherwise famous (see[3]) paper of Lekkerkerker and Boland ([6]), which implicitly checks whether every minimal separator contained in the current vertex neighborhood is a clique. Unlike the characterization given in [4], upon which algorithm LEX M is based, the vertices can be processed in any order.

Our algorithm repeatedly makes into a clique every minimal separator (expressed as some $N_G(C)$) contained in the neighborhood of the vertex being processed, instead of making the whole neighborhood into a clique as with a traditional fill, and can use any ordering on the vertices.

The algorithm is simple (and can be easily implemented by an undergraduate student), but the proof uses some non-trivial modern results on minimal separation (see [5]). Our interesting complexity is due to the fact that the connected components of $\mathcal{C}(N[x])$ are the same in the transitory graph H as in G; the graph search which computes these connected components can thus be run on the input graph in $O(m)$ time, instead of the transitory graph in $O(m')$ time. Making $N_G(C)$ into a clique can be done globally in $O(m')$ time using a technique such as that described in [9] for computing a fill, which makes our overall complexity $O(nm + m') = O(nm)$.

5 Recursive version using clique minimal separator decomposition.

Clique minimal separator decomposition consists in copying some clique minimal separator S at each step into each component C of $\mathcal{C}(S)$. (More precisely, $N(C) \subseteq S$ is copied into component C (see [1])). A weaker version of this decomposition was studied by Tarjan ([8]); the decomposition turns out to be unique when only clique minimal separators are used, which is precisely what an improvement suggested by Tarjan himself in [8] does, although the author did not know it.

At the end of each step of our algorithm, the minimal separators contained in the vertex neighborhood are cliques and can be used as clique separators to decompose the graph; the process is then recursively repeated on each of the subgraphs obtained. This recursive version is similar to the minimal triangulation algorithm presented in [5], the complexity of which was however estimated as $O(n^3)$.

Strong invariants of clique minimal separator decomposition ([1]) ensure that at each step of such a decomposition, the remaining minimal separators are partitioned into the subgraphs obtained; thus when a minimal separator is chosen in a vertex neighborhood of some subgraph in the course of the triangulating process, it is a minimal separator in its own right for the global graph, just as if the graph had not been decomposed. Since the graph searches are run on subgraphs, it is reasonable to conjecture that the mean-time complexity of this recursive version is very interesting and possibly linear.

References

[1] A.Berry, J-P.Bordat, *Decomposition by clique minimal separators*, Res. Rep. LIM, Marseille (May 1997), submitted to SIAM J. Disc. Math.

[2] J.Blair, P.Heggerness, J.A.Telle, *Making an arbitrary filled graph minimal by removing fill edges*, SWAT'96 proceedings.

[3] D.G Corneil, S.Olariu, L.Stewart, *Asteroidal Triple-Free Graphs*, SIAM J. Disc. Math. 10(1997)399-430.

[4] D.R.Fulkerson,O.A.Gross, *Incidence Matrices and Interval Graphs*, Pacific J. Math. 15(1965)835-855.

[5] T.Kloks,D.Kratsch,H.Müller, *Approximating the bandwidth for asteroidal triple-free graphs*, ESA'95 proceedings.

[6] C.G.Lekkerkerker, J.C.Boland, *Representation of a finite graph by a set of intervals on the real line*, Fund. Math. 51(1962)45-64.

[7] D.Rose, R.E.Tarjan and G.Lueker, *Algorithmic aspects of Vertex Elimination on Graphs*, SIAM J. Comput. 5(1976)266-283.

[8] R.E.Tarjan, *Decomposition by clique separators*, Disc. Math. 55(1985)221-232.

[9] R.E.Tarjan and M.Yannakakis, *Simple Linear-Time Algorithms to Test Chordality of Graphs*, SIAM J. Comput. 13(1984)566-579.

[10] M.Yannakakis, *Computing the minimum fill-in is NP-Complete*, SIAM J. Alg. Disc. Meth. 2(1981)77-79.