CSC 565: Graph Theory **North Carolina State University**
Fall 2019 **Computer Science**

Lecture 27: Nov 25, 2019

Lecturer: Don Sheehy <drsheehy@ncsu.edu> *Scribe: K. Gardner, G. Miller*

Lat week we discussed how to solve systems of linear equations of the form

$$Mx = b$$

in the special case where M is positive semidefinite (P.D.) using the Cholesky decomposition. While this can be done using the Cholesky decomposition the performance depends on the structure of the matrix. In the case of sparse matrices, for example, we would want to keep the matrix sparse. This requires an ordering on the vertices which minimizes the number of contractions.

The ordering is given by a permutation matrix P known as the *pivot order.* We then solve

$$PMP^\top = Pb$$

and return $x = P^\top y$.

For some ordering $v_1, \ldots, v_n$ on the vertices of a graph G let G_i be the graph after vertices 1-i are eliminated. The *fill graph* is the graph with the same vertex set and all of the edges resulting from contractions:

$$G^+ = (\{1, \ldots, n\}, \bigcup_{i=1}^{n} E_{G_i}).$$

These edges are referred to as the *fill edges* E^+.

We will build an *elimination tree* rooted on the last vertex eliminated v_n where the parent of every other node v_i is defined to be $\min\{j > i \mid (i,j) \in E^+\}$.

$$|E^+| = \sum_{e \in E} |m^{-1}(e)|.$$

Throughout we assume an edge written (b, c) implies $b < c$. We would like a mapping $m : E^+ \to E$ that tells us which edge in the initial graph to "charge" We define $m(b, c) = (a, c)$ where a is the minimal descendant of b in the elimination tree such that $(a, c) \in E$. This process will work up from the leaves.

Lemma 1. For every $(b, c) \in E^+$ there exists $(a, c) \in E$ such that a is a descendant of b

Proof. If b is minimal then set $a = b$ and we are done. So we may assume w.l.o.g. that $(b, c) \notin E$.

Suppose, for the sake of contradiction, that there exists some $(b, c) \in E^+$ with no descendant a of b such that $(a, c) \in E$. So (b, c) was added by a reduction after removing a vertex x that was adjacent to both b and c. Let $x = max\{v < b \mid (x, b), (x, c) \in E^+\}$ and let y be the parent of x in the elimination tree. That is, y is the smallest larger value connected to a vertex in the fill graph.

Note that $y \neq c$ as c is connected to b. If $y \neq b$ then $x < y \leq b$, but y is connected to b and c in the fill graph, contradicting our choice of x. There exists a descendant a of x such that $(a, c) \in E$ in the same way, therefore (a, c) is a descendant of b. □

In nested dissection a separator is used between subgraphs to order eliminations. That is, a vertex set V is broken as the disjoint union $V = A \sqcup B \sqcup C$ and the seperator order is $\text{order}(V) = (\text{order}(A), \text{order}(B), C)$. We can group vertices in an elimination tree where the high level is the separator tree. Total height $4\sqrt{n}$. Each fill edge in the elimination tree is along a path between the vertices of the corresponding edge that gets "charged" in the initial graph. The length of this path is at most the depth of the tree.

In the top level seperator there are $2\sqrt{n}$ vertices and height at most $4\sqrt{n}$. So every vertex in the seperator has degree at most 4 in E. So there are at most $32n$ fill edges in the top-level separator, reducing geometrically as we move down the tree. We can write the following recurrence relation

$$f(n) \leq 32n + 4f(n/4)$$

which gives $O(n \log n)$ fill edges.

We will fill with nested dissection in $O(n \log n)$. If every seperator has ...

A *perfect elimination graph* is a graph $G = (V, E)$ such that there exists an ordering of V such that no new fill edges are introduced in the elimination tree. That is, for each vertex removed there exist edges between all of its neighbors in E. Examples of perfect elimination graphs include trees and triangulated polygons.

Definition 1. A *chordal graph* is a graph G such that every induced cycle of G is K_3.

We can glue graphs along shared k-cliques using the *k-clique sum*, depicted in Figure.Note that clique sums of cliques is always a chorded graph. Note that $A_i \cap G_{i-1} \subseteq A_j$ for some $j < i$. In other words, every clique in G_i is contained in A_j for some $j \leq i$.

We can think of the gluing of cliques as edges in a tree with cliques as vertices. For an ordered set of cliques $A_1, \ldots, A_r$ and $G_i = G_{i-1} + A_i$ we define a tree with vertex set $V = \{A_1, \ldots, A_r\}$ rooted at A_1 such that the parent of all other nodes in the tree is the clique $\text{parent}(A_i) = A_j$ such that $A_i \cap G_{i-1} \subseteq A_j$ and $j < i$. We note that every chordal graph can be constructed in this way. A_r is a leaf in the tree. If this vertex is removed all its neighbors are in the corresponding clique - contains $r \in G_{r-1}$.

Note that we have no new fill because... cliques. All new vertices added in that step can be removed without adding fill.

Corollary 1. *Clique sums of cliques have perfect elimination orders.*

Lemma 2. Chorded graphs are clique sums of cliques.

Proof. If we have a corded graph and a minimal separator then there must be a path between any two vertices in the separator. If the graph is chordal then min separators must be cliques. Note that every minimum separator in a chordal graph is a clique.

If the graph is a clique we are done. Otherwise take any minimal separator C which separates subgraphs A and B. C is a clique and, because subgraphs of chordal graphs are chordal, A and B are chordal. Therefore $G[A \cup C]$ and $G[B \cup C]$ are clique sums of cliques by induction. □

Theorem 1. *G is a perfect elimination graph if and only if it is chordal.*

Proof. (⇒) Suppose there is a perfect elimination ordering and an induced cycle that is not a clique. If we have a perfect elimination ordering one of these vertices must be removed. First that would add an edge to the fill graph, a chord, not induced.

(⇐) By Lemma 2 G is a sum of cliques, which has a perfect elimination order by Corollary 1. □