Complexity of Finding Embeddings in a *k*-Tree

S. Arnborg, D. G. Corneil, and A. Proskurowski

Graphical Models Reading Group

Presented by Chris Bartels

University of Washington

Department of Electrical Engineering

January 29, 2004

Overview

- Computer Science Background
- Graph Theory Background
- PARTIAL K-TREE is NP-complete
- Recognition of partial k-trees for fixed k

Overview

- Computer Science Background
  - Turing Machines
  - Classes P and NP
  - Reductions
  - NP-Completeness
  - NP-hard
- Graph Theory Background
- PARTIAL K-TREE is NP-complete
- Recognition of partial k-trees for fixed k

Turing Machines

Alan Turing, 1936

- Our mathematical model of a computer
- Defined by
  - A set of states, Q
  - Special states, q_{start}, q_{accept}, q_{reject}
  - An infinite memory "tape"
  - Input alphabet, Σ, tape alphabet, Γ
  - A transition function,
    - $Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\}$

Turing Machines

- Turing machines provide binary answers
  - q_{accept}, q_{reject}, or never stop running
- We will only deal with TMs that are *deciders*
- The set of input strings a TM accepts is its *language*
- Functionality usually described on a high level

Turing Machines

- Provides a model for what can/can't be done on a real computer
- Variations on the Turing machine as well as completely different models
  - Can't do things Turing machines can't
  - Differences are polynomial in space and time

The class P

- Class of problems solvable by a polynomial time TM
- That is, can be solved in an amount of time that is polynomial in the length of the input
  - PATH: Given a graph, is there a path from *s* to *t*
  - PRIME: Given an integer, is it a prime number
- The class P is only decision problems

The class NP

- Class of problems solvable by a non-deterministic polynomial time TM
- When you don’t know what to do, run all options in parallel
- If any branch *accepts*, the machine *accepts*
- If all branches *reject*, the machine *rejects*
- Total time is the time of longest branch

The class NP

- Also, the class of languages that have polynomial time verifiers
- Examples:
  - HAMPATH
  - SAT
  - CLIQUE
- ~HAMPATH does not appear to be in NP

P vs. NP

- For some problems we can prove that they take an exponential amount of time, EXPTIME
- We can prove that:

$$P \subseteq NP \subseteq EXPTIME$$

$$P \subset EXPTIME$$

- It is speculated that:

$$P \subset NP \subset EXPTIME$$

Reductions

- Language A is reducible to language B

$$A \leq_P B$$

- $A \leq_P B$ iff there exists a function

$$f : \Sigma^* \rightarrow \Sigma^*$$

where for every

$$\omega \in A \Leftrightarrow f(\omega) \in B$$

- Example 3SAT $\leq_P$ CLIQUE
  - 3SAT = $\{<\phi> \mid \phi$ is a satisfiable 3-cnf formula $\}$
  - CLIQUE = $\{<G,k> \mid G$ is a graph with a k-clique$\}$

NP-Completeness

Cook-Levin

- A language A is NP-complete if:
  - A is in NP
  - All other languages in NP $\leq_P$ A
- Beginning with a language and an input, we can form a Boolean equation that is true iff the input is in the language
- If a language is in NP, the equation can be written in polynomial time
- SAT is NP-complete

Suppose CLIQUE $\in$ P

- If CLIQUE $\in$ P then P=NP. How?
  - Start with NP-complete language A
  - Reduce to Boolean formula
  - Reduce to 3-cnf Boolean formula
  - Reduce to graph
  - Use magical CLIQUE solver to get answer

NP-Hard

- A problem is NP-hard if:
  - All problems in NP are polynomial time reducible to it
- NP-hard problems do not have to be in NP
- Do not have to be decision problems
- Often the optimization version of a NP-complete decision problem
  - Finding optimal k-tree vs. does the graph have a k-tree

Overview

- Computer Science Background
- Graph Theory Background
  - Triangulation and k-trees
  - Elimination
  - Chain Graphs
  - Minimum cut linear arrangement
- PARTIAL K-TREE is NP-complete
- Recognition of partial k-trees for fixed k

Triangulation and k-trees

- A graph is triangulated if
  - All cycles of length >= 4 have a chord
  - Iff it has a perfect elimination order
  - Iff it has a Junction Tree
  - ect...
- A k-tree
  - Triangulated with all max cliques size k
- A partial k-tree
  - Any subgraph of a k-tree

Triangulated graphs are magic

- Many NP-hard problems become polynomial on triangulated graphs
  - Finding the maximal cliques

Lemma 2.1

- $k_t(G) = k_c(G)$
  - Minimum *k* such that G is a partial k-tree is equal to minimum maxclique size of all triangulations
- Interested in k-trees because we want to minimize calculations in the junction tree algorithm
- Prove that the problem is NP-hard and therefore unlikely to have a shortcut

Elimination

- Algorithm
  - Elimination always yields triangulated graphs
- Perfect elimination orderings
  - Triangulated graphs always have many perfect elimination orderings
- Not all triangulations are obtainable by an elimination ordering
  - All edge minimal triangulations are obtainable by elimination

Chain Graphs

- Bipartite graphs: $G=(A \cup B, E)$
- A chain graph
  - There exists an ordering s.t.

$$\#(u)<\#(v) \text{ iff } \Gamma(u) \supseteq \Gamma(v)$$

- C(G) is formed by completing A and B
- A bipartite graph is a chain graph iff C(G) is triangulated

Minimum Cut Linear Arrangement

$$C_{\#}(G) = \max_i | \{(u,v) \in E : \#(u <= I < \#(v)\} |$$

$$MCLA = \{<G,k> \mid \text{Is there an arrangement s.t.t } c_{\#}(G) <= k\}$$

Blocks of a graph

- Blocks are sets of nodes that have the same neighborhood, Γ(u)
- Lemma 3.1
  - H is a minimal triangulation of G. There exists an elimination order which is block contiguous in both G and H s.t. H is the triangulation of G w.r.t. this order.

Reduction from MCLA to PARTIAL K-TREE

- Give mapping function
- If C(G') is a a partial k'-tree:
  - There exists a block contiguous elimination order,π', s.t. no vertex has degree > k' when eliminated.
  - F is the filled graph after elimination in order π'
  - We can choose π' to be any perfect elimination order of F *(might have to assume F is minimal)*

Reduction from MCLA to PARTIAL K-TREE

- F is chordal, so it is a chain graph
- F has a perfect elimination order which is the reverse chain ordering & is block contiguous. Choose one of these for π'.
- π is ordering of blocks in π', also the ordering of the original nodes in G

Reduction from MCLA to PARTIAL K-TREE

- Consider the graph after eliminating the first (i-1) blocks of C(G'). Each vertex in A_i is adjacent to:
  - $\Delta(G)$ other vertices in A_i
  - $\Delta(G)+1$ other vertices in A_{i+1} A_{i+2} ...$A_{|V|}$
  - $\Delta(G)+1\text{-deg}(j)$ vertices in B_j for j=1...l
  - 2 vertices, B_e, for each edge connected to a vertex in {1,...l}

Reduction from MCLA to PARTIAL K-TREE

- The vertices in A contribute

$$\Delta(G)+(\Delta(G)+1)(|V|-i)$$

- The vertices in B contribute
  - Consider the -deg(j) term

$$\text{sum}(\deg(j), j=1\ldots I) = 2|E_2^i|+|E_1^i|$$

  - Were $|E_2^i|$ has both vertices in {1,...,I}
  - Were $|E1^i|$ has one vertex in {1,...,I}

$$(\Delta(G)+1)I - (2|E_2^i|+|E_1^i|) + (2|E_2^i|+2|E_1^i|)$$

Reduction from MCLA to PARTIAL K-TREE

- Simplifies to:

$$(\Delta(G)+1)(|V|+1)-1+|E_1^i|$$

- $|E1^i|$ is the number edges with one vertex in {1,...,i} and one vertex in {i+1,...,|V|}
- Therefore, C(G') is a partial k'-tree implies G has a minimum linear cut value k

Reduction from MCLA to PARTIAL K-TREE

- Assume G has a minimum linear cut value k.
- There exists a π which gives π'
- The largest clique in F has size k'+1

PARTIAL K-TREE in NP-complete

- PARTIAL K-TREE is in NP
  - Give an elimination order s.t. the elimination graph is k-chordal

Reduction from MCLA to PARTIAL K-TREE

- deg(x): degree, number of neighbors of node x
- $\Delta(G)$ maximum degree of all nodes