On the Sequential and Parallel Complexity of Matching in Chordal and Strongly Chordal Graphs

Elias Dahlhaus

Basser Department of Computer Science
University of Sydney
Australia

Marek Karpinski*

Department of Computer Science
University of Bonn
53117 Bonn

and

International Computer Science Institute
Berkeley, California

January, 1994

Abstract

In this paper we consider the sequential and parallel complexity of the maximum matching problem in chordal and strongly chordal graphs. We prove that, given a strongly perfect elimination ordering, a maximum matching in a strongly chordal graph can be found in a linear time. On the other hand we observe that the matching problem restricted to chordal (paths) graphs is of the same parallel complexiy as a general bipartite matching.

*Supported in part by the DFG Grant KA 673/4–1, by the ESPRIT BR Grants 7097 and ECUS030, and by the Volkswagen–Stiftung. E Mail: marek@cs.uni-bonn.de.

1 Introduction

Chordal graphs became interesting as a generalization of interval graphs (see for example [8]). We call a graph chordal if every cycle of length greater than three has a chord, i.e. an edge that joins two non consecutive vertices of the cycle. Note that interval graphs are not only chordal but strongly chordal as defined in [3]. Strongly chordal graphs are just those chordal graphs having a so called strongly perfect elimination ordering.

In this paper we consider the sequential and parallel complexity of the maximum matching problem in chordal and strongly chordal graphs. Note that in general a linear time algorithm for perfect matching is not known. Here we shall show that, provided a strongly perfect elimination ordering is known, a maximum matching in a strongly chordal graph can be found in linear time by a simple greedy algorithm. This algorithm can be turned into a (non optimal) parallel algorithm. The random bits in the algorithm of [9] can be eliminated in the special case of strongly chordal graphs.

On the other hand, we shall find out that matching restricted to chordal graphs (also restricted to path graphs) is of the same parallel complexity degree as bipartite matching.

In section 2, we shall introduce the basic notation. In section 3 we consider the sequential and parallel complexity of maximum matching restricted to strongly chordal graphs. In section 4 we discuss the parallel complexity of matching restricted to path graphs.

2 Notation and Basic Definitions

A *graph* $G = (V, E)$ consists of a *vertex set* V and an *edge set* E. Multiple edges and loops are not allowed. The edge joining x and y is denoted by xy.

We say that x is a *neighbor* of y iff $xy \in E$. The *full neighborhood* of x is the set $\{x\} \cup \{y : xy \in E\}$ consisting of x and all neighbors of x and is denoted by $N(x)$.

A *path* is a sequence $(x_1 \dots x_k)$ of distinct vertices such that $x_i x_{i+1} \in E$.

A *cycle* is a closed path, that means a sequence $(x_0 \dots x_{k-1} x_0)$ such that x_i $x_{i+1 \pmod k} \in E$.

A *subgraph* of (V, E) is a graph (V', E') such that $V' \subset V$, $E' \subset E$.

An *induced subgraph* is an edge-preserving subgraph, that means (V', E') is an induced subgraph of (V, E) iff $V' \subset V$ and $E' = \{xy \in E : x, y \in V'\}$.

A graph (V, E) is *chordal* iff each cycle $(x_0 \dots x_{k-1} x_0)$ of length greater than 3 has an edge $x_i x_j \in E, j - i \neq \pm 1 \bmod k$ (which joins vertices which are not neighbors in the cycle). Sometimes they are also called *triangulated* or *rigid circuit* graphs. We remark that this notion is equivalent to the nonexistence of an induced cycle of length greater than 3.

Independently Gavril [5] and Buneman [1] proved the following:

Theorem: *A graph is chordal iff it is the intersection graph of vertices of subtrees of a tree, i.e. the vertices of the chordal graph corresond to subtrees of a fixed*

tree and two vertices of the chordal graph are joined by an edge iff the corresonding subtrees share a vertex of the tree.

A *path graph* is the intersection graph of a collection of paths of a tree.

We also can define chordal graphs by characteristic orderings.

Theorem: [4] A graph $G = (V, E)$ is chordal iff there is an ordering $<$ of V, such that with $x < y$, $x < z$, $xy \in E$, and $xz \in E$, we have $yz \in E$. Such an ordering is called a *perfect elimination ordering.*

A graph $G = (V, E)$ is called *strongly chordal* [3] iff there is an ordering $<$ on the vertices of V such that

1. for $xy, xz \in E$, such that $x < y$ and $x < z$, also $yz \in E$,
2. for $x_1y_2, x_2y_1, x_1x_2 \in E$, such that $x_1 < y_1$ and $x_2 < y_2$, we have $y_1y_2 \in E$.

Such an ordering is called a *perfect elimination ordering.*

A *matching* of $G = (V, E)$ is a subset M of E such that no two edges share a vertex. A matching of maximal size is called a *maximum matching.* If all vertices of G belong to an edge of the matching M then M is called a *perfect matching.*

3 Maximum Matching Algorithms for Strongly Chordal Graphs

We assume that a strongly perfect elimination ordering $<$ of the vertex set of the graph $G = (V, E)$ i.e. the corresponding enumeration $(v_1, \ldots v_n)$ is given.

We claim that the following algorithm computes a maximum matching in a strongly chordal graph.

1. $V' := V$; $M := \emptyset$;
2. Repeat

   uv is an edge in E with $u, v \in V'$, u is minimal with respect to $<$, v is the $<$-smallest vertex in V' than is adjacent to u;

   $M := M \cup \{uv\}$; $V' := V' \setminus \{u, v\}$

   until there are no edges in E with both incident vertices in V'.

It is easily seen that this algorithm has a time bound of $O(n + m)$.

We have to show the correctness.

For a matching M of G, we call a pair of edges u_1u_2 and w_1w_2 in M a *defect* of M if

1. $u_1w_1 \in E$,
2. $u_1 < w_2$, and $w_1 < u_2$.

Lemma 1: If there is a matching of cardinality k then there is a defect free matching of the same cardinality k.

Proof: We label the edge v_iv_j with $l_{v_iv_j} := (i-j)^2$. Suppose there is a defect consisting of the pair u_1u_2 and w_1w_2. Then, by definition $u_1w_1 \in E$. Since $u_1 < w_2$ and $w_1 < u_2$ and $<$ is a strongly perfect elimination ordering, $u_2w_2 \in E$. Therefore we get a matching M' where the edges u_1u_2 and w_1w_2 are replaced by the edges u_1w_1 and u_2w_2.

Claim: $\Sigma_{e\in M'} l_e < \Sigma_{e\in M} l_e$.

Proof of Claim: For simplicity, we identify the vertices with their indices v_i.

We consider the following subcases:

First case $u_1 < w_2 < w_1 < u_2$:

$$\begin{aligned}(u_2-w_2)^2 + (w_1-u_1)^2 &= (u_2-w_2)^2 + ((w_1-w_2)+(w_2-u_2))^2\\ &= (u_2-w_2)^2 + (w_1-w_2)^2 + 2(w_1-w_2)(w_2-u_1) + (w_2-u_1)^2\\ &< (w_1-w_2)^2 + (u_2-w_2)^2 + 2(u_2-u_1)(w_2-u_1) + (w_2-u_1)^2\\ &= (w_1-w_2)^2 + (u_2-u_1)^2.\end{aligned}$$

Second case $u_1 < w_1 < w_2 < u_2$:

$$(w_1-u_1)^2 + (u_2-w_2)^2 < (u_2-u_1)^2 < (u_2-u_1)^2 + (w_2-w_1)^2$$

Third case $u_1 < w_1 < u_2 < w_2$: Then the inequality $(w_1-u_1)^2 - (w_2-u_2)^2 < (u_2-u_1)^2 + (w_2-w_1)^2$ follows immediately.

All other possible cases are permutations of the cases as considered.

□ (Claim)

Clearly after the removal of several defects, we find a matching of the same cardinality with a minimum sum of labels l_e. This matching is free of defects.

□ (Lemma)

Lemma 2: The maximum matching obtained by above algorithm is defect free.

Proof: Suppose there is a defect u_1w_1, u_2w_2 with $u_1 < w_2$, $u_2 < w_1$ and $u_1u_2 \in E$. Suppose $u_1 < u_2$. Then w_1 is not the minimal choice of a neighbor as required by the algorithm.

□ (Lemma)

Theorem 1: The matching computed by the above algorithm is a maximum matching.

Proof: We consider any defect free maximum matching M and the matching M' computed by the above algorithm.

Let x be the smallest vertex y such that M restriced to $\{u|u \le y\}$ and M' restricted to $\{u|u \le y\}$ are different. Then M restricted to $\{u|u < x\}$ and M' restricted to $\{u|u < x\}$ coincide. It cannot be that x is covered by an edge of M but not by an edge of M', because necessarily x has a neighbor that is not covered

by M restricted to $\{u|u < x\}$ and the edge joining x with the minimum neighbor t must be in M'(if $t < x$ then x is chosen as the smallest neighbor of t not covered by the matching considered before. If $x < t$ then t is chosen as the smallest neighbor of x by above algorithm). Suppose x is in an edge of M' but does not appear in an edge of M. Note that x has a neighbor t that is not in an edge of M restricted to $\{u|u < x\}$. We add xt to M and delete the edge $ty \in M$ if such an edge exists.

Therefore we may assume that there are edges xt of M and an edge xt' of M' that are incident with x. Note that $t' < t < x$ and t' does not belong to an edge of M restricted to $\{u|u < x\}$. Moreover, it cannot belong to an edge of M. Otherwise there is an edge $t'y \in M$ with $y > x$ and $t'y$ and xt forms a defect. Therefore in M, we can replace xt by xt' and the new matchin M coincides with M' in $\{u|u \leq x\}$. By induction, we get a maximum matching that coincides with M'.

□ (theorem)

Corollary: For strongly chordal graphs, a maximum matching can be computed in linear time

Theorem 2: In strongly chordal graphs, one can find a perfect matching by a CREW-PRAM in $O(\log^2 n)$ time with a polynomial processor bound if a perfect matching exists.

Proof: We prove that there is at most one defect free perfect matching. Since this is the perfect matching with the minimum sum of labels $l_{uv} = (u - v)^2$, we get a perfect matching by the minimum perfect matching algorithm of [9] in $O(\log^2 n)$ time with a polynomial processor bound.

Lemma 3: There exists at most one defect free perfect matching.

Proof: Assume there are defect-free perfect matchings M and M'. Assume M and M' coincide in $\{u|u < x\}$ but not in $\{u|u \leq x\}$. Suppose $xt \in M$ and $xt' \in M'$ are the edges in M and M' respectively that are incident with x. Without loss of generality, we assume that $t' < t$. Both t and t' do not appear in any edge of M and M' with both incident vertices in $\{u|u < x\}$. Therefore the edge $t'u$ in M that is incident with t' must have the property that $u > x$. But then $t'u$ and xt form a defect in M. This is a contradiction.

□ (lemma)

□ (theorem)

Remark: A strongly perfect matching of a strongly chordal graph can be computed in $O(\log^4 n)$ time with a linear processor number [2]. Therefore it is possible to get an NC-algorithms to compute a perfect matching in strongly chordal graphs also without the knowledge of a strongly perfect elimination ordering.

4 The Parallel Complexity of Maximum Matching in Path Graphs

Theorem 3: Suppose the we can find a perfect matching of a path graph in polylogarithmic time with a polynomial processor bound. Then we can find a perfect matching in a bipartite graph in polylogarithmic time with a polynomial processor bound, i.e. the marriage problem is in NC.

Proof: We construct a reducton from the the bipartite perfect matching problem into the perfect matching problem restricted to path graphs that can be computed in logarithmic time with $O(n^2)$ processors.

Given a bipartite graph $B = (V \cup W, E)$ with all edges incident with exactly one vertex in V and exactly one vertex in W. Note that B has only a perfect matching if V and W have the same size.

We construct an interval representation as follows.

The tree T is consists of a main node c, vertices t_v, for each $v \in V$, and vertices $s_{w,i}$, $1 \leq i < deg(w)$, $w \in W$. The parent of each t_v and each $s_{w,1}$ is c and the the parent of each $s_{w,i}$ is $s_{w,i-1}$, for $i \neq 1$.

The collection $\mathcal{P}$ of paths is constructed as follows. For each node $t \neq c$ of T, we provide a one node path p_t containing exactly t, and for each $vw \in E$, we have a path q_{vw} containing t_v, c, and all nodes $s_{w,i}$.

It is easily seen that this path representation and therefore also the resulting path graph $G = (\mathcal{P}, E_G)$ can be constructed in $O(\log n)$ time with $O(n^2)$ processors by a CREW-PRAM.

It remains to show that each perfect matching in G induces a perfect matching in B and vice versa.

Suppose a perfect matching M og G is given. Note that there are as many paths p_t as paths q_{vw}. Note that each path p_t shares a node only with a path q_{vw}. Therefore a perfect matching of G consists only of edges of the form $p_t q_{vw}$. Since there are $deg(w) - 1$ nodes $s_{w,i}$, exactly and $deg(w)$ many paths $q_{v,w}$, exactly one path $q_{v,w}$ is matched with $p_{t,v}$, say $q_{v_w,w}$. Then $M' = \{v_w w | w \in W\}$ defines a perfect matching in B.

Vice versa, we assume that a perfect matching M' of B is given. For each $vw \in M'$, let $p_{t_v} q_{vw} \in M$ and for each $v'w \in E$ with $v' \neq v$, choose a distinguished number $i_{v'} < deg(w)$ and let $s_{w,i_{v'}} q_{v'w} \in M$. M defines a perfect matching of G.

□ (theorem)

5 Conclusions

We would like to mention that the parallel perfect matching algorithm for strongly chordal graphs is not optimal. It remains an intersting problem to find an optimal parallel perfect elimination algorithm for strongly chordal graphs.

Finally we would like to remark that strongly chordal graphs are exactly the chordal graphs that are complements of comparability graphs [6]. It is known that the perfect matching problem restricted to complements of cocomparability graphs is equivalent to 2-processor scheduling, and this can be done in $\log^2 n)$ time with a polynomial processor bound [7]. It might be interesting to find a reasonable upper class of strongly chordal graphs and complements of comparability graphs such that the perfect matching problem can still be parallelized.

6 Acknowledgements

Recently Stephan Olariu has mentioned to us that R. Lin got similar results to ours. We are grateful to Avi Wigderson, Joseph Naor, and Alejandro Schaeffer for a number of intersting conversations.

References

[1] P. Bunemann, *A Characterization of Rigid Circuit Graphs*, Discrete Mathematics 9 (1974), pp. 205-212.

[2] E. Dahlhaus, *Chordale Graphen im besonderen Hinblick auf parallele Algorithmen*, Habilitation Thesis, University of Bonn, 1991.

[3] M. Farber, *Characterizations of Strongly Chordal Graphs*, Discrete Mathematics 43 (1983), pp. 173-189.

[4] D. Fulkerson, O. Gross, *Incidence Matrices and Interval Graphs*, Pacific Journal of Mathematics 15 (1965), pp.835-855.

[5] F. Gavril, *The Intersection Graphs of Subtrees in Trees Are Exactely the Chordal Graphs*, Journal of Cobinatorial Theory Series B, vol. 16(1974), pp. 47-56.

[6] P. Gilmore, A. Hoffman, *A Characterization of Cocomparability Graphs and of Interval Graphs*, Canadian Journal of Mathematics 16 (1964), pp. 539-548.

[7] D. Helmbold, E. Mayr, *Two Processor Scheduling is in NC*, in *VLSI Algorithms and Architectures* (F. Makedon et al. ed.), LNCS 227 (1986), pp. 12-15.

[8] C. Lekkerkerker, J. Boland, *Representation of a Finite Graph by a Set of Intervals on the Real Line*, Fundamenta Mathematicae 51.

[9] K. Mulmuley, U. Vazirani, V. Vazirani, *Matching is as easy as matrix inversion*, Combinatorica 7 (1987), pp. 105-113.