Discussiones Mathematicae
Graph Theory 15 (1995) 119–145

EFFICIENT ALGORITHMS FOR MINIMAL DISJOINT PATH PROBLEMS ON CHORDAL GRAPHS

C.P. Gopalakrishnan, C.R. Satyan and C. Pandu Rangan

Department of Computer Science
Indian Institute of Technology
Madras 600 036, India
email: rangan@iitm.ernet.in

Abstract

Disjoint paths have applications in establishing bottleneck-free communication between processors in a network. The problem of finding minimum delay disjoint paths in a network directly reduces to the problem of finding the minimal disjoint paths in the graph which models the network. Previous results for this problem on chordal graphs were an $O(\mid V \mid \ \mid E \mid^2)$ algorithm for 2 edge disjoint paths and an $O(\mid V \mid \ \mid E \mid)$ algorithm for 2 vertex disjoint paths. In this paper, we give an $O(\mid V \mid \ \mid E \mid)$ algorithm for 2 vertex disjoint paths and an $O(\mid V \mid + \mid E \mid)$ algorithm for 2 edge disjoint paths, which is a significant improvement over the previous result.

Keywords: chordal graph, minimal paths, disjoint paths, clique, bfs.

1991 Mathematics Subject Classification: 05C38, 05C85.

1. Introduction

Disjoint path problems have applications in establishing communication between processors in a network. In fact, the problem of finding minimal delay communication paths directly reduces to the problem of finding the corresponding minimal disjoint paths in the underlying network. We may state the **minimal disjoint path** problem as follows:

Given an undirected graph $G = (V, E)$ and two pairs of vertices s, t and u, v (all distinct) we seek two vertex/edge *disjoint* paths between each pair such that they are *minimal* paths between these pairs.

In this paper, we solve the above problem on *chordal* graphs (also called *triangulated* graphs), a class of *perfect graphs*. For characterizations and properties of chordal graphs refer [G 80].

The problems concerning minimal disjoint paths have attracted very little attention so far. Some partial results have been obtained by Schwill [S 89, S 90]. But research on these problems for paths without length constraints has been done so far. An $O(\mid V \mid \ \mid E \mid)$ algorithm for the two vertex disjoint paths problem for a general undirected graph was given by Shiloach [S 80] and Ohtsuki [O 80]. The more general n disjoint path problem, i.e. the problem to decide, given a graph and n pairs of vertices, whether the graph contains n pairwise vertex disjoint paths each one connecting a single pair of vertices was shown to be NP-complete by Karp [K 75]. However, if n is fixed, Robertson and Seymour [RS 86] give an algorithm which runs in $O(\mid V \mid^2 \ \mid E \mid)$ steps. The algorithm is, however unpractical due to a leading constant (hidden by the 'big O') of horrible size.

The two paths problem on chordal graphs i.e., finding two vertex/edge-disjoint paths, without any length constraints has been shown to admit a linear algorithm ([PS 78], [KPS 91]). But there is no known polynomial algorithm for the general n disjoint path problem even when restricted to chordal graphs except the complicated and unpractical Robertson and Seymour algorithm.

As far as our problem is concerned (minimal disjoint problem) the earlier known polynomial time results are by Schwill [S 80] who gave an $O(\mid V \mid \mid E \mid^2)$ algorithm for finding 2 minimal edge disjoint paths on a chordal graph. He also claims an $O(\mid V \mid \ \mid E \mid)$ algorithm for the vertex disjoint case, crucially based on a result which appears to be inadequate. In this paper, following a new approach, we present an $O(\mid V \mid \ \mid E \mid)$ algorithm for finding 2 vertex disjoint paths, and an $O(\mid V \mid + \mid E \mid)$ algorithm for finding 2 edge disjoint paths in a chordal graph, which is a significant improvement over the results in [S 89].

2. Preliminaries

A path P of length n is an ordered sequence of $(n+1)$ distinct vertices $v_0, v_1, \ldots, v_n$ such that $(v_i, v_{i+1}) \in E$, for all $0 \leq i < n$. For subpaths we use square brackets and specify the end vertices. '[' and ']' denote the inclusion of left and right end vertices in the subpath respectively and ']' and '[' denote their exclusion. For example, $P[v_0, v_2]$ specifies the subpath $[v_0, v_1, v_2]$ and $P]v_0, v_4[$ specifies the subpath $[v_1, v_2, v_3]$. The operation '.' concatenates two internally disjoint paths at a common end vertex. A path

$P[x, y]$ is said to be *minimal* in G if no other path $P'[x, y]$ has length less than $P[x, y]$.

Definition 2.1. Let $G = (V, E)$ be a chordal graph. Let s, t, u, v be the four given vertices. Consider the *bfs tree* rooted at s. Let t occur in the kth level of the bfs tree (i.e. $d(s, t) = k$). In this tree, a vertex x is said to *lie between* s and t iff there is a minimal path between s and t which passes through x. That is, x is said to lie between s and t iff $d(s, x) + d(x, t) = d(s, t)$.

Let $G' = (V', E')$ be the graph induced by the vertices which lie between s and t. Define

$$S(i) = \{x \mid x \in V' \text{ and } d(s, x) = i\}, \quad 0 \leq i \leq k.$$

Similarly, if $d(u, v) = l$, we get the graph $G'' = (V'', E'')$ and $U(j)$, $0 \leq j \leq l$, from the bfs tree rooted at u. From the definition, it is obvious that the S's and U's correspond to different **levels** in the respective bfs trees. Refer to Figure 1.

Lemma 2.1. *If $x \in S(i)$ (or $U(i)$) and $y \in S(j)$ (or $U(j)$), with $i < j$, and if there is a path $[x = x_0, x_1, \ldots, x_{r-1}, y = x_r]$, where $r = j - i$ then there is a shortest path from s to t (or u to v) containing all the vertices $x_0, x_1, \ldots, x_r$.*

Proof. The path is simply constructed by taking the minimal paths from s to x and y to t and concatenating them to get the path $P[s, x].[x_0, x_1, \ldots, x_{r-1}].P[y, t]$. It can be easily verfied that the length of this path adds up to $d[s, t]$ and hence is minimal. Similarly the other part of the lemma can be proved. ■

Corollary 2.1.1. *If $x \in S(i)$ and $y \in S(j)$, with $i < j$, and if there is a path $[x = x_0, x_1, \ldots, x_{r-1}, y = x_r]$, where $r = j - i$, then $x_f \in S(i + f)$ and hence $x_f \in S$, $0 \leq f \leq r - 1$.*

Lemma 2.2. *Each $S(i)$, $0 \leq i \leq k$, is a clique. Similarly each $U(j)$, $0 \leq j \leq l$, is a clique.*

Proof. Follows from the fact that G' (respectively G'') is a chordal graph and that each $S(i)$ (respectively $U(j)$) is a minimal vertex separator. ■

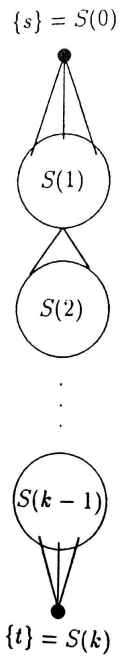

Figure 1. The sets $S(1)$ to $S(k-1)$ are the sets of vertices which lie between s and u at various levels.

Corollary 2.2.1. *If* $x \in S(i)$ $(or\ U(j))$ *and* $y \in S(i+r)$ $(or\ U(j+r))$, *then* $d(x,y) = r$ *or* $(r+1)$.

Proof. $d(x,y) \geq r$, because the shortest path should at least be of length r ($S(i)$ and $S(i+1)$ are levels of a bfs tree). $d(x,y) \leq r+1$ because there is some vertex $z \in S(i)$ such that $d(z,y) = r$, and since each $S(i)$ is a clique, $d(x,z) \leq 1$. So, $d(x,y) \leq d(x,z) + d(y,z) \leq r+1$. ■

Corollary 2.2.2. *If* $x \in S(i)$ *and* $y \in S(i+r)$ *and* $d(x,y) = r+1$, *then there is a shortest path* $x = x_0, x_1, \ldots, x_r, y = x_{r+1}$ *such that* x_0 *and* x_1 *belong to* $S(i)$ (A *similar corollary may be stated for* U).

Definition 2.2. Let $S = \cup_{0 \leq i \leq k} S(i)$ and $U = \cup_{0 \leq j \leq l} U(j)$. We call S (U) as a *strand*.

If $S \cap U = \emptyset$, any minimal path between s and t, and any minimal path between u and v will not have any vertices/edges in common. Thus the problem is trivially solved. So, from now on we will consider graphs where $S \cap U \neq \emptyset$.

Consider, $S \cap U(j)$, for some j. These vertices, should either belong to a single set $S(i)$ or two consecutive sets, $S(i)$ and $S(i+1)$ for some i. This follows directly from the facts that each $U(j)$ is a clique and the end vertices of an edge can either be in two consecutive $S(i)$'s or, in the same $S(i)$. Based on the above observation, we classify the $U(j)$'s into three classes, namely (refer to Figure 2.)

- Type *ZERO* $U(j)$, such that $U(j) \cap S = \emptyset$.
- Type *ONE* $U(j)$, such that $U(j) \cap S$ is a subset of $S(i)$, for some i.
- Type *TWO* $U(j)$, such that $U(j) \cap S$ is a subset of $S(i) \cup S(i+1)$, for some i, and it is neither a subset of $S(i)$ nor a subset of $S(i+1)$.

Type ONE and Type TWO can further be subdivided into subtypes, **A** or **B** depending on whether $U(j) = S \cap U(j)$ or not. Note, that similarly $S(i)$'s can be classified with respect to U.

Lemma 2.3. *The vertices in* $U(i)$ *can be linearly ordered (numbered) as* $x_1, x_2, \ldots, x_k$, *where* $k =\mid U(i) \mid$, *such that* $N(x_j, U(i+1))$ *is a subset of* $N(x_{j+1}, U(i+1))$ *for each* $1 \leq j \leq k$, *where* $N(x, U(i+1))$ *denotes the neighbourhood of* x *in* $U(i+1)$.

Proof. We construct such a linear order of vertices. Find $N(x_t, U(i+1))$ $\forall x_t \in U(i)$. Arrange the x_t's in linear order (say nondecreasing) according to the cardinality of $N(x_t, U(i+1))$. We claim that this is the required order. If not let x_k violate the order at k, i.e. $N(x_k, U(i+1)) \not\supset N(x_{k-1}, U(i+1))$. Now consider the 4-cycle x_k, x_{k+1}, y, z where y and z are neighbours of x_k and x_{k-1} in $U(i+1)$. Further y is not a neighbour of x_{k-1} and z is not a neighbour of x_k, because of the non-superset claim (which implies the presence of such a y and z ($(y,z) \in E$ in view of Lemma 2.2 in the respective neighbourhood sets of x_k and x_{k-1} in $U(i+1)$). Now, by the triangulation property we arrive at a contradiction. ■

Corollary 2.3.1. There is at least one vertex in $U(i)$ which is adjacent to all vertices in $U(i+1)$. This is called the *special vertex.*

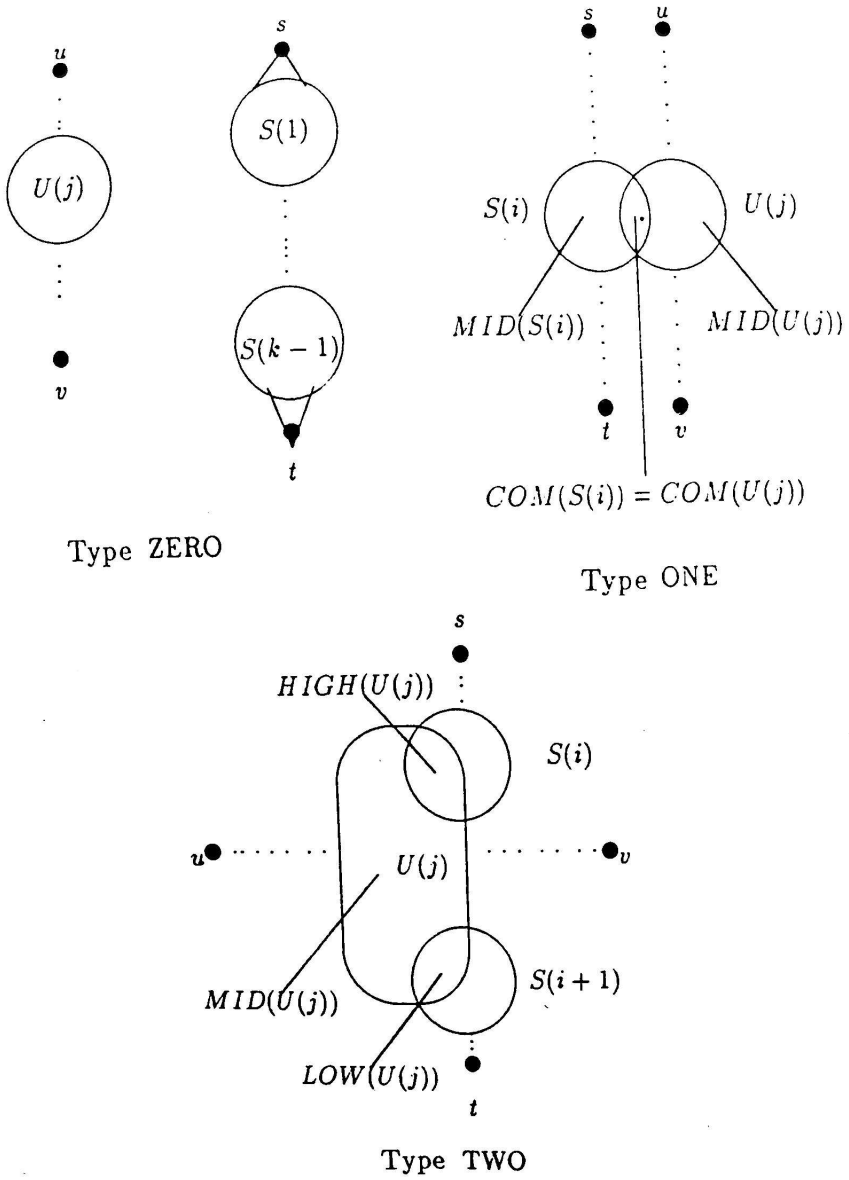

Figure 2. Different types of $U(j)$.

Proof. It is clear that the cardinality of the highest numbered vertex in the linear order is $\mid U(i+1) \mid$. ■

By symmetry there exists at least one special vertex in $U(i+1)$ which is adjacent to all vertices in $U(i)$. This is called the *upward special vertex* and

the former is called the *downward special vertex*.

Definition 2.3. The *Projection* of a vertex $x \in U(i)$ (denoted by $Proj(x)$) over the level $U(j)$ is defined to be the set of vertices in $U(j)$ which are accesible to x in the shortest possible distance. By generalizing Lemma 2.3 it follows that in level $U(j)$ the projections of $x_1, x_2, \ldots, x_k$ (these vertices belong to $U(i)$) obey the same superset property (i.e) $Proj(x_1) \subset Proj(x_2) \subset \cdots \subset Proj(x_k)$. The projection of a set of vertices of a level on a different level is the union of individual projections. Projection can be upward or downward.

Lemma 2.4. *If* $S(p1) \cap U \neq \emptyset$ *and* $S(p1 + q1) \cap U \neq \emptyset$ *for some* $p1$ *and* $q1$, *then* $S(i) \cap U \neq \emptyset$, *for all* $p1 < i < (p1 + q1)$.

Proof. Assume the contrary. Then there exists an integer r, such that $p1 < r < (p1 + q1)$ and $S(r) \cap U = \emptyset$. Let p be the greatest integer such that $p < r$, and $S(p) \cap U \neq \emptyset$. Similarly let q be the least integer such that $(p + q) > r$ and $S(p + q) \cap U \neq \emptyset$. It is easily seen that for every $i1$, $p < i1 < (p + q)$, $(Si1) \cap U = \emptyset$.
Now, let $U(j1)$ be one of the sets such that $U(j1) \cap S(p) \neq \emptyset$ and $U(j2)$ be one such that $U(j2) \cap S(p + q) \neq \emptyset$. Let $x \in U(j1) \cap S(p)$ and $y \in U(j2) \cap S(p + q)$. By Corollary 2.2.1, $d(x, y)$ is either q or $q + 1$.
If $d(x, y) = q$, then $\mid j1 - j2 \mid$ must be either q or $q - 1$ by Corollary 2.2.1. If $\mid j1 - j2 \mid$ is q, then the path $[x_0 \ldots x_q]$ of length q constructed by Corollary 2.2.2 will make each $x_i \in S(p + i)$ and also $U(j1 + i)$. If it is $(q-1)$ then the path $[x, x_1, x_2 \ldots, x_{q-1}, y]$ connecting x, y in G'' makes each $x_i \in S(p + i)$, for $0 < i < q$ (by Lemma 2.1). Thus, we have contradicted the assumption that $S(p + i) \cap U = \emptyset$.
If $d(x, y) = q + 1$, consider two shortest paths (both of length $q + 1$), one in G' and the other in G''. Let these paths be $x = x_0, x_1, x_2, \ldots, x_q, y = x_{q+1}$ and $x = y_0, y_1, y_2, \ldots, y_q, y = y_{q+1}$, respectively. Here x_0 and x_1 are chosen to be in the same level in G' (Corollary 2.2.2). Since $d(x, y) = q + 1$, $\mid j1 - j2 \mid$ must be either q or $q + 1$ by Corollary 2.2.1. If $\mid j1 - j2 \mid$ is $q + 1$, then the path in G' with length $q + 1$ makes $x_i \in U\,(0 < i < q + 1)$ (Lemma 2.1), and contradicts the assumption. Otherwise, if $\mid j1 - j2 \mid = q$, then we can choose the path in G'' such that y_0 and y_1 belong to the same level (Corollary 2.2.2). Now, these two paths form a cycle in the induced graph of the union of vertices in G' and in G''. Since both are shortest paths, there is no chord between two vertices in the same path. So, the only possible chords are of the types (x_i, y_i) or (x_i, y_{i+1}) or (x_{i+1}, y_i) for $0 < i < q + 1$ (chords like (x_i, y_{i+2}) etc. violate the shortest path

property). Also all the chords cannot be of the first type only, since it still leaves the chordless 4-cycle $(x_i, y_i, y_{i+1}, x_{i+1}, x_i)$. Thus there is at least one chord that is not of the first type for every i. Let it be (x_i, y_{i+1}). Now, the path $[x = x_0, \ldots, x_i, y_{i+1}, \ldots, y_{q+1} = y]$ has length $q+1$. Therefore, by Lemma 2.1, each $x_j, 0 < j \leq i$ belongs to U, giving a contradiction again. ■

The above lemma shows that if $U \cap S \neq \emptyset$, then the U's which intersect S form a contiguous subsequence of $U(0) \ldots U(l)$, say $U(a) \ldots U(b)$. Similarly, the S's which intersect U form a contiguous subsequence.

Definition 2.4. $U(j)$ is said to be *intersecting* if $U(j) \cap S(i) \neq \emptyset$. If the sequence of U's which intersects S is $U(a)..U(b)$, and if the sequence of S's which intersects U is $S(c)..S(d)$, such that $U(a)$ intersects $S(c)$ and $U(b)$ intersects $S(d)$, then the sequences are said to be in the *same direction* if $(d-c)(a-b) \leq 0$.

From now on we assume that the sequence of intersecting U's are in the same direction as a minimal path from s to t. If this is not so, i.e. $U(a)$ intersects $S(d)$ and $U(b)$ intersects $S(c)$, then we can simply interchange the labels of u and v to get the required proper order.

Let $x_a \in U(a)$, $y_b \in U(b)$, $x_c \in S(c)$, $y_d \in S(d)$. If there are paths $P[x_a, y_b]$, $Q[x_c, y_d]$ such that P and Q are vertex/edge disjoint and minimal, then they can be extended to get vertex/edge disjoint paths between s, t and u, v.

Based on the structure of intersection of S and U we identify two types and treat each one of them separately initially. The intersection is said to be a *C1-type* intersection if it contains at least one U of type TWO otherwise it is said to be of *C2–type*. First we consider C1-type.

3. C1-type Intersection

Lemma 3.1. *If there are i, j such that $S(i) \cap U(j) \neq \emptyset$ and $S(i+1) \cap U(j+2) \neq \emptyset$ then $U(j+1)$ is of type TWO.*

Proof. By Lemma 2.4, $U(j+1) \cap S \neq \emptyset$. Since there are edges between $U(j)$ and $U(j+1)$, $U(j+1)$ can only intersect with $S(i-1), S(i), S(i+1)$. Since there are edges between $U(j+1)$ and $U(j+2)$, $U(j+1)$ can ony intersect with $S(i+1), S(i), S(i+2)$. Thus the common sets are $S(i), S(i+1)$. We now show that $U(j+1)$ intersects both (i.e. it is type TWO). Consider $x \in (S(i) \cap U(j))$ and $y \in (S(i+1) \cap U(j+2))$. x is adjacent to at least one vertex in $S(i+1)$ say z. $z \neq y$, because there can be no edge from any

vertex in $U(j)$ to any vertex in $U(j+2)$. Similarly, there is an edge (w, y), $w \in S(i), w \neq x$. Also (y, z) is an edge because $S(i+1)$ is a clique. Thus we can now use the path $[x, z, y]$ and prove (by Lemma 2.1) that $z \in U(j+1)$. Similarly $w \in U(j+1)$. Hence both $U(j+1) \cap S(i)$ and $U(j+1) \cap S(i+1)$ are non-empty. ■

Lemma 3.2. *If there is at least one U of type TWO (say $U(j)$) then*
1. *if $U(j+1)$ is of type ONE, $j+1$ is the greatest integer r such that $U(r) \cap S \neq \emptyset$.*
2. *if $U(j-1)$ is of type ONE, $j-1$ is the least integer r such that $U(r) \cap S \neq \emptyset$.*

Proof. (1) Assume the contrary. By Lemma 2.4, if there is an integer r greater than $j+1$, such that $U(r) \cap S \neq \emptyset$, then $U(j+2) \cap S \neq \emptyset$. Let $U(j)$ intersect S in $S(i)$ and $S(i+1)$ (because $U(j)$ is of type TWO). Now, $U(j+1)$ must intersect $S(i)$ or $S(i+1)$ because it is type ONE. Without loss of generality, let it intersect $S(i+1)$. Since there are edges between $U(j+1)$ and $U(j+2)$, the latter can intersect S only in $S(i)$ or $S(i+1)$ or $S(i+2)$. But it can neither intersect $S(i)$ nor $S(i+1)$ because then there would be an edge between $U(j)$ and $U(j+2)$. Hence, $U(j+2)$ should intersect $S(i+2)$. This along with the fact that $U(j) \cap S(i+1) \neq \emptyset$ and Lemma 3.1 implies that $U(j+1)$ should be of type TWO, which contradicts the assumption that it is of type ONE.
(2) The proof of this part is similar to proof of (1). ■

Corollary 3.2.1. *If there is a U of type TWO, then a U of type ONE cannot be inbetween other intersecting U's and also at most one U of type ONE can be at the ends of the sequence of U's which intersect S. That is if there is a U of type TWO and if $U(a) \ldots U(b)$ is the sequence of intersecting U's only $U(a)$ or $U(b)$ or both can be of type ONE.*

Proof. Follows directly from Lemma 3.2. ■

Lemma 3.3. *If there exist i, j such that $S(i) \cap U(j) \neq \emptyset$, $S(i) \cap U(j+1) \neq \emptyset$ and $S(i+1) \cap U(j) \neq \emptyset$ and $S(i+1) \cap U(j+1) \neq \emptyset$ then, $U(r) \cap S = \emptyset$ for all r satisfying $0 \leq r < j$ and $j+1 < r \leq l$.*

Proof. Let if possible $U(r) \cap S \neq \emptyset$ for some $r > (j+1)$. This implies (by Lemma 2.4) that $U(j+2) \cap S \neq \emptyset$. Since $U(j+1)$ intersects $S(i)$ and $S(i+1)$, this implies that $U(j+2)$ should intersect at least one of $S(i)$ or $S(i+1)$. If it intersects $S(i)$, it implies that there is an edge between $U(j)$ and $U(j+2)$, because both have some vertices common with $S(i)$ and $S(i)$

is a clique. Again if $U(j+2)$ intersects $S(i+1)$, then $S(i+1)$ intersects $U(j)$, $U(j+1)$, $U(j+2)$ which is not possible. So, we get a contradiction in both ways. ■

Definition 3.1. (Refer Figure 2) Let $U(j)$ be of type TWO and let it intersect the S's at $S(i)$ and $S(i+1)$. Define $HIGH(U(j)) = (U(j) \cap S(i))$ and $LOW(U(j)) = (U(j) \cap S(i+1))$. For a $U(j)$ of type ONE which intersects $S(i)$, $HIGH(U(j)) = LOW(U(j)) = U(j) \cap S(i)$. For a $U(j)$ of type TWO-B or ONE-B, we define $MID(U(j)) = U(j) \backslash (U(j) \cap S)$ (the part which is not in common with S). For type A's $MID(U(j)) = \emptyset$. We define $COM(U(j)) = U(j) \backslash MID(j)$ (the common part). Similar definitions can be given for $S(i)$'s with respect to U.

Definition 3.2. Let $U(a)$ intersect $S(i)$ and $S(i+1)$ if it of type TWO, else let it intersect $S(i+1)$ if it is of type ONE. We call the sequence $U(a) \ldots U(b)$, a *TWO-chain* if

1. Each $U(j)$, $a < j < b$, is of type TWO and $U(a), U(b)$ are each type ONE or TWO.
2. Each $U(j)$, $a < j < b$, intersects $S(i+j-a)$ and $S(i+j-a+1)$. $U(b)$ intersects $S(i+b-a)$. It also intersects $S(i+b-a+1)$ if it is of type TWO (Refer to Figure 3).

Lemma 3.4. *In a C1-type intersection, if there are more than two U's which intersect S then they form a TWO-chain.* (*By symmetry a similar lemma can be stated for S's.*)

Proof. Let the intersecting $U's$ be $U(a) \ldots U(b)$. First take the case when there is only one U of type TWO. Thus the other U's which intersect should be type ONE. But Corollary 3.2.1 implies that there should be exactly two U's and they must be $U(a)$ and $U(b)$ (where $b = a+2$). Further, $U(a)$ and $U(b)$ should be of type ONE and $U(a+1)$ should be of type TWO. Let $U(a+1)$ intersect S in $S(i+1)$ and $S(i+2)$. This implies that $U(a)$ intersects $S(i+1)$ and $U(b)$ intersects $S(i+2)$. Thus the 3 U's form a TWO-chain.

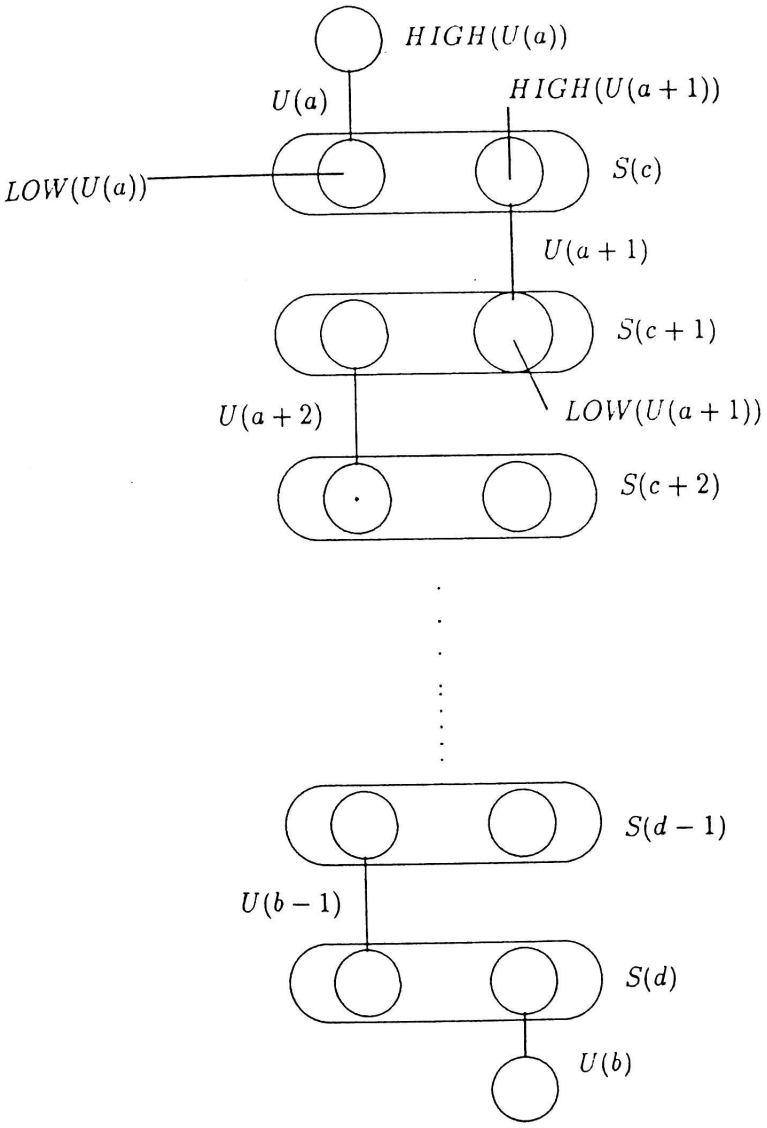

Figure 3. A TWO-chain.

Now consider the case when there are at least 2 U's of type TWO. Take the least j such that $U(j) \cap S \neq \emptyset$ and $U(j)$ is of type TWO. From Corollary 3.2.1, $j = a$ or $a+1$. Let $U(j)$ intersect S in $S(i)$ and $S(i+1)$ for some i. By Corollary 3.2.1, $U(j+1)$ is of type TWO. Then $U(j+1)$ must intersect $S(i-1)$ and $S(i)$ or $S(i+1)$ and $S(i+2)$ (it cannot intersect $S(i)$ and $S(i+1)$ because of Lemma 3.3). It intersects $S(i+1)$

and $S(i+2)$, because we have assumed that the S's and U's are in the same direction (see Definition 2.4). Now, $U(j+2)$ can only intersect $S(i+2)$ and $S(i+3)$. This is because it can neither intersect $S(i)$ nor $S(i+1)$ (as $U(j)$ intersects both of these). Thus we see by an easy extrapolation that if $U(e)$ is of type TWO, then it intersects $S(e-j+i)$ and $S(e-j+i+1)$. Also let $S(d-1)$ and $S(d)$ which are intersected by the set $U(f)$ where f is the largest integer such that $U(f)$ is of type TWO. If $U(a)$ is of type ONE, we can see that it should intersect $S(i)$ and similarly $U(b)$ should intersect $S(d)$. Thus the sequence forms a TWO-chain. ■

Thus we see that C1-type intersection can be further classified into:
Type C1.1: There is only one intersecting U of type TWO.
Type C1.2: There are two intersecting $U's$.
Type C1.2.1: Both the $U's$ are of type TWO.
Type C1.2.2: One of type TWO U and the other is of type ONE.
Type C1.3: There are more than two intersecting $U's$ forming a TWO-chain

First we give lemmas which takes care of the trivial cases: C1.1 and C1.2.

Lemma 3.5. *Let there be a C1.1-type intersection in G and let the only intersecting $U(j)$ of type TWO intersect $S(i)$ and $S(i+1)$. Then*
1. *Edge-disjoint paths exist.*
2. *Vertex disjoint paths exist iff at least one of the conditions given below is true:*
(a) *$U(j)$ is of type TWO-B.*
(b) $\mid S(i) \mid > 1$ *or* $\mid S(i+1) \mid > 1$.

Proof. (1) Take any vertex $x \in S(i)$. A shortest path from u to v which is of the form $[u \ldots x].[x \ldots v]$ can share only one vertex with any shortest path from s to t. Thus the path $[u \ldots x].[x \ldots v]$ and any path from s to t are the required edge-disjoint paths.

(2a) Take any vertex $x \in MID(U(j))$. The path $[u \ldots x].[x \ldots v]$ and any shortest path form s to t will not have any vertex in common (x does not belongs to S).

(2b) Take a vertex $x \in HIGH(U(j))$ and another vertex $y \in S(i)$. Then the shortest paths $[u \ldots x].[x \ldots v]$ and $[s \ldots y].[y \ldots t]$ are vertex-disjoint. ■

Definition 3.3. The type C1.2.1 further consists of two types which are
C1.2.1.1: The two interesting $U's$ intersect $S(i)$ and $S(i+1)$.
C1.2.1.2: The two intersecting $U's$ intersect $S(i)$, $S(i+1)$ and $S(i+2)$.

Lemma 3.6. *Let there be a C1.2.1.1-type intersection in G and let the two $U's$ be $U(j)$ and $U(j+1)$ and the $S's$ which intersect them be $S(i)$ and $S(i+1)$. Then*
1. *Edge-disjoint paths exist.*
2. *Vertex-disjoint paths exist iff one of the following is true*
   (a) $\mid S(i) \mid > 2$ *or* $\mid S(i+1) \mid > 2$.
   (b) *At least one of $U(j)$, $U(j+1)$ is of type TWO-B.*
   (c) *if $x, y \in S(i)$ and $z, w \in S(i+1)$ and $x, w \in U(i)$ and $y, z \in U(i+1)$, then (x, z) and (y, w) are edges.*

Proof. Similar to Lemma 3.5. ■

Lemma 3.7. *If $U(j)$ and $U(j+1)$ are of type TWO, and if $U(j)$ intersects $S(i)$ and $S(i+1)$ and $U(j+1)$ intersects $S(i+1)$ and $S(i+2)$, then every vertex in $HIGH(U(j))$ is adjacent to some vertex in $HIGH(U(j+1))$ and every vertex in $LOW(U(j+1))$ is adjacent to some vertex in $LOW(U(j))$.*

Proof. First let us assume that $U(j+1)$ is of type TWO-A. Since every vertex in $HIGH(U(j))$ is adjacent to some vertex in $U(j+1)$, it must be adjacent to either a vertex in $HIGH(U(j+1))$ or $LOW(U(j+1))$. But it cannot be adjacent to a vertex in $LOW(U(j+1))$, because this would mean that there is an edge between $S(i)$ and $S(i+2)$ which is not possible. So it should be adjacent to some vertex in $HIGH(U(j+1))$.

Now consider the case that $U(j+1)$ is of type TWO-B. As said above, a vertex in $HIGH(U(j))$ cannot be adjacent to a vertex in $LOW(U(j+1))$. Thus it can only be adjacent to a vertex in $HIGH(U(j+1))$ or $MID(U(j+1))$. Suppose it is adjacent to a vertex in $MID(U(j+1))$. Let (x, y) be that edge. Since $U(j+1)$ is a clique, y is adjacent to all vertices in $LOW(U(j+1))$, in particular to a vertex z (which exists since $U(j+1)$ is of type TWO). Now consider the path of $[x, y, z]$ of length 2. Now, $x \in S(i)$ and $z \in S(i+2)$ and Lemma 2.1 imply that $y \in S(i+1)$, a contradiction to the fact that y does not belong to S. Thus even in the case a vertex in $HIGH(U(j))$ can only be adjacent to a vertex in $HIGH(U(j+1))$. The proof of the other part of the lemma is similar to the proof given above. ■

Lemma 3.8. *Let there be a C1.2.1.2-type intersection in G and let the two $U's$ be $U(j)$ and $U(j+1)$, and let them intersect $S(i)$, $S(i+1)$ and $S(i+1)$, $S(i+2)$, respectively. According to Lemma 3.7 there exist edges between $HIGH(U(j))$ and $HIGH(U(j+1))$ and between $LOW(U(j+1))$ and $LOW(U(j))$, let these edges be (x, y) and (w, z), respectively, where $x \in S(i)$, $y, z \in S(i+1)$ and $w \in S(i+2)$. Then*

1. *Edge-disjoint paths exist.*
2. *Vertex-disjoint paths exist iff one of the following conditions hold*
   (a) *At least one of* $U(j)$ *and* $U(j+1)$ *is of type TWO-B.*
   (b) $\mid S(i+1) \mid > 2$.
   (c) *There is at least one vertex in* $S(i)$ *which is not* x *(say* d*) and which is adjacent to* z *or there is at least one vertex in* $S(i+2)$ *which is not* w *(say* c*) and which is connected to* y.

Proof. Similar to Lemma 3.5. ■

Lemma 3.9. *Let there be a C1.2.2-type intersection in* G. *If* $U(j)$ *is of type ONE which* $S(i)$ *and* $S(i+1)$, *then*
1. *Edge-disjoint paths exist.*
2. *Vertex-disjoint paths exist iff one of the following conditions is true*
   (a) *At least one of* $U(j)$ *and* $U(j+1)$ *is of type TWO-B.*
   (b) $\mid S(i) \mid > 2$.
   (c) *There are two edges* (x, y), (z, w) *with* x, y, z, w *all different such that* $x \in LOW(U(j))$, $y \in LOW(U(j+1))$, $z \in HIGH(U(j+1))$ *and* $w \in S(i+1)$.

Proof. Similar to Lemma 3.5. Note that a similar result holds when $U(j)$ is of type TWO and $U(j+1)$ is of type ONE. ■

Lemma 3.10. *Let* $U(a) \ldots U(b)$ *be a sequence of* $U's$ *forming a TWO-chain. For any* $j < b-1$, *let* $U(j+1)$ *intersect* $S(i)$ *and* $S(i+1)$ *for some* i. *Then every vertex in* $LOW(U(j))$ *is adjacent to every vertex in* $LOW(U(j+1))$.

Proof. Since $U(j+1)$ is neither the first nor the last set of the TWO-chain it should be of type TWO. Since the $U's$ form a TWO-chain, $U(j)$ should intersect $S(i)$. Take any vertex $x \in LOW(U(j))$. This vertex is also a member of $S(i)$. Thus it is adjacent to some vertex in $S(i+1)$, say y. This y cannot belong to $U(j+2)$ (which exists, because $U(j+1)$ is not the last U in the TWO-chain). Let $z \in HIGH(U(j+1))$. Since $U(j+2)$ belongs to a TWO-chain, z should belong to $S(i+1)$. Now the fact that the path (x, y, z) is of length 2 and $x \in U(j)$ and $z \in U(j+2)$ make $y \in U(j+1)$ (Lemma 2.1). Since y belongs to both $S(i+1)$ and $U(j+1)$, it should also belong to $LOW(U(j+1))$. Since, we have taken an arbitrary x, the lemma is true for all vertices belonging to $LOW(U(j)$.

Lemma 3.11. *If* $U(a) \ldots U(b)$ *forms a TWO-chain, then there exist vertex-disjoint paths* $[x_0, x_1, \ldots, x_{p-1}, x_p]$ *and* $[y_0, y_1, \ldots, y_{p-1}, y_p]$ *where*

$x_0 \in LOW(U(a))$, $x_p \in LOW(U(b-1))$, $y_0 \in HIGH(U(a+1))$, *and* $y_p \in U(b)$ *where* $p = b - a - 1$.

Proof. Refer Figure 3. If $U(a)$ is of type ONE, let it intersect $S(c)$. If it is of type TWO, let it intersect $S(c-1)$ and $S(c)$. We build the paths iteratively (by induction). The induction hypothesis is that if at any stage $[x_0, x_1, \ldots, x_i]$ and $[y_0, y_1, \ldots, y_i]$ are the two paths then

(1) The paths are vertex-disjoint.

(2) If $i \neq p$, then x_i and y_i are in the same level $U(a+i)$.

(3) $x_i \in LOW(a+i)$.

(4) If $i \neq p$, then $y_i \in HIGH(U(a+i+1))$ else if $i = p$ then $y_i \in U(a+i+1)$ (which is the same as $U(b)$).

Initially take any vertex $x_0 \in LOW(a)$ and any vertex $y_0 \in HIGH(a+1)$. Since it is a TWO-chain, these two vertices lie in the same level $S(c)$ and they are obviously distinct.

Suppose it is true at the jth step, i.e., we have the two paths $[x_0, x_1, \ldots, x_j]$ and $[y_0, y_1, \ldots, y_j]$ which satisfy the four properties. If j is already equal to p, then we have found the required paths. If not, this implies that $y_j \in HIGH(a+j+1)$ and $x_j \in LOW(a+j)$. Since $j < p$, $(a+j+1) < (a+p+1) = (a+b-a-1+1) = b$. Thus $U(a+j+1)$ is not the last U in the chain. This implies that it is of type TWO. By Lemma 3.10, we have that x_j is adjacent to some vertex in $LOW(U(a+j+1))$ (say x_{j+1}) in the next level (which is $S(a+j+1)$). Now, suppose $j+1 < p$, then this implies that $(a+j+2) < b$. Thus $U(a+j+2)$ is also of type TWO. Since $U(a+j+1)$ is also of type TWO (because $(a+j+2) > a$, i.e., it is not the first level), this implies by Lemma 3.7 that y_j should be adjacent to some vertex in $HIGH(a+j+2)$ (say y_{j+1}) in the next level (which is $S(a+j+1)$). Thus in this case the induction hypothesis is true for $j+1$ also. If $j+1 = p$, then because every vertex of $U(a+j+1) = U(a+p) = U(b-1)$, is adjacent to some vertex of $U(b)$, y_j should be adjacent to some vertex j_{j+1} in $U(b)$. Thus even in this case the induction hypothesis holds good.

The induction gives the result immediately. ■

Lemma 3.12. *If* $U(a) \ldots U(b)$ *are the sets intersecting* S *and there is a TWO-chain, then edge-disjoint paths exist.*

Proof. Let $[x_0, x_1, \ldots, x_p]$ and $[y_0, y_1, \ldots, y_p]$ be the two paths which are mentioned in Lemma 3.11 ($p = s - a - 1$). Now consider the paths $[u \ldots x_0, y_0, y_1, \ldots, y_p, \ldots, v]$ and $[s \ldots x_0, x_1, \ldots, x_p \ldots t]$. These paths have only one vertex in common i.e., x_0. But these two do not have any edge

in common because the vertices y_0 and x_1 are distinct (they in fact lie in distinct levels). ■

Remark 3.1. The problem with these two paths is that the path starting from the first U i.e., $U(a)$, does not go upto $U(b)$, but stops in $U(b-1)$. Thus if we are able to extend this path until $U(b)$ still keeping the paths vertex-disjoint, we will get the required two paths. Note that even though a TWO-chain exists, there may not be any vertex-disjoint paths.

Lemma 3.13. *If $U(a) \ldots U(b)$ are the sets intersecting S and if they form a TWO-chain, vertex-disjoint paths exist iff one of the following conditions hold:*
1. *If $[x_0, x_1, \ldots, x_p]$ and $[y_0, y_1, \ldots, y_p]$ are the two vertex-disjoint paths mentioned in Lemma 3.11 and $S(c) \ldots S(c+p)$ are the levels which contain the above vertices, then for at least one e, $c \leq e \leq c+p$ we have that $\mid S(e) \mid > 2$.*
2. *There is altelast one integer i, $a \leq i \leq b$, such that $U(i)$ is of type TWO-B.*
3. *$U(a)$ is of type TWO, and there exist two distinct vertices, $x, y \in S(c)$, such that $x \in HIGH(U(a))$ and y is adjacent to at least one vertex in $LOW(U(a))$ or $U(b)$ is of type TWO, and there exist two distinct vertices $x, y \in S(d)$, such that $x \in LOW(U(b))$ and y is adjacent to at least one vertex in $HIGH(U(b))$.*
4. *At least one of $U(a)$, $U(b)$ is of type ONE-B (say $U(a)$) and there is a vertex $x \in MID(U(a))$ which is adjacent to some vertex in $HIGH(U(a+1))$.*

Proof. The important point to note here is that the starting vertices x_0 and y_0 may be any vertex of $LOW(U(a))$ and $HIGH(U(a+1))$, respectively. Also there cannot be any edges between x_i and y_{i+1} because they belong to $LOW(U(i))$ and $HIGH(U(i+2))$, respectively.

(1) Let $S(e)$ be the level such that $\mid S(e) \mid > 2$. Let $i = e-c$. Then there is a vertex x which is neither x_i nor y_i in the level $S(e)$. Now we claim that there are vertex-disjoint paths from x_0 to y_i and s to x. Take any shortest path from s to x (say P) and the path $[x_0, x_1, \ldots, x_i, y_i]$. If these two paths do not intersect then our claim is true. Suppose these two paths intersect at some vertex x_j which is nearest along the path P to x. $j \neq i$ because no shortest path from s to x will pass through x_i as x_i and x are in the same level. Let z be the vertex in the path $P[x_j, x]$ which is adjacent to x_i. By our choice of x_j, z is not any of $x_j \ldots x_i$. Consider the four-cycle $x_j, z, y_{j+1}, y_j, x_j$. The edges (z, y_{j+1}) and (y_j, x_j) exist because they are in

the same level. The chords that this four-cycle can have are (x_j, y_{j+1}) and (z, y_j). But since the edge (x_j, y_{j+1}) cannot exist (because $x_j \in U(a+j)$ and $y_{j+1} \in U(a+j+2)$), the edge (z, y_j) must exist. Now consider the paths $[x_0, x_1, \ldots, x_i, y_i]$ and $[s \ldots y_0, y_1, \ldots, y_j, z].P[z, x]$. These paths are obviously vertex-disjoint. In a similar manner one can show that there are vertex- disjoint paths $[x \ldots w, x_m, \ldots, x_p]$ and $[y_i, y_{i+1}, \ldots, y_p]$. Here w and x_m are analogous to the vertices z and y_j mentioned above. Thus the vertex-disjoint paths are $[u, \ldots, x_0, x_1, \ldots, x_i, y_i, y_{i+1}, \ldots, y_m, \ldots, v]$ and $[s, \ldots, y_0, y_1, \ldots, y_j, z, \ldots x \ldots w, x_m, x_{m+1}, \ldots, x_p, \ldots, t]$.

(2) This condition implies that $\mid U(f) \mid > 2$ for some f. If we interchange the roles of s, t and u, v, this condition will appear as condition (1) for the interchanged version. Thus in a similar manner to the proof of (1) the vertex-disjoint paths can be shown to exist.

(3) This condition implies that x is adjacent to some vertex (say y_0) in $HIGH(U(a+1))$ (by Lemma 3.10) and y is adjacent to some vertex in $LOW(U(a))$ say x_0. We immediately have the paths $[u \ldots x, y_0, \ldots, y_p, \ldots, v]$ and $[s \ldots y, x_0, \ldots, x_p, \ldots t]$. A similar construction can be given for the other case also.

(4) When $s-t$ and $u-v$ are interchanged, this condition manifests itself as condition (3). Thus we can again construct vertex-disjoint paths.
If all the above conditions are false, then it is obvious that vertex-disjoint paths do not exist. ■

Theorem 3.1. *The minimal vertex/edge-disjoint path problem can be solved when there is a C1-type intersection in a chordal graph G in $O(\mid V \mid + \mid E \mid)$ time.*

Proof. The various lemmas given before takes care for the corresponding subclasses of C1-type of intersection. We now show that using these lemmas we can easily implement a linear algorithm. We assume an adjacency list representation of G. The sets S and U can be found out using a standard bfs search (implemented in linear time). Also, the type of intersection can be found out (whether it is C1 or C2) by scanning the sets of $S's$ and $U's$. We are concerned now only with C1-type of intersection. Depending on whether it is C1.1, C1.2 or C1.3, the corresponding lemmas give the simple conditions to check and output the vertex-disjoint paths. As can be noted edge-disjoint paths always exist in this type of intersection and also be output in linear time. ■

4. C2-type Intersection

In this type of intersection there are no type TWO U's or S's. All are of type ONE.

Lemma 4.1. *If there is an integer j such that $U(j) \cap S(i) \neq \emptyset$ and $U(j+1) \cap S(i) \neq \emptyset$ and both $U(j)$ and $U(j+1)$ are of type ONE, then for no other r which is neither j nor $j+1$, is $U(r) \cap S \neq \emptyset$ true.*

Proof. Similar to Lemma 3.2. ■

Definition 4.1. We define a sequence $U(a) \ldots U(b)$ to be a *ONE-chain* if
1. Each $U(r), a \leq r \leq b,$ is of type ONE.
2. If $U(a) \cap S(i) \neq \emptyset$, then for each $r, a \leq r \leq b$, $U(r) \cap S(i + r - a) \neq \emptyset$ (refer to Figure 4).

From Lemma 4.1, it is clear that if a C2-type intersection exists in a chordal graph then it can only be of the following types:
Type C2.1: There is only one U.
Type C2.2: There are only two $U's$ and they intersect the same S.
Type C2.3: The $U's$ in the chordal graph form a ONE-chain.

First we give lemmas which takes care of the trivial cases: C2.1 and C2.2.

Lemma 4.2. *Let there be a type C2.1-type intersection in G, and let $U(j)$ be the only U which intersects $S(i)$, then*
1. *Edge-disjoint paths exist.*
2. *Vertex-disjoint paths exist iff one of the following conditions is true*
   (a) *$U(j)$ is of type ONE-B*
   (b) $| S(i) | > 1$.

Proof. Easy. ■

Lemma 4.3. *Let there be a C2.2-type intersection in G, and let the two intersecting $U's$ be $U(j)$ and $U(j+1)$ which intersect $S(i)$, then*
1. *Edge-disjoint paths exist.*
2. *Vertex-disjoint paths exist iff one of the following conditions is true*
   (a) *$U(j)$ or $U(j+1)$ is of type ONE-B.*
   (b) $| S(i) | > 2$.

Proof. Easy. ■

We now turn our attention to C2.3-type intersection characterized by the ONE-chain.

We now study how the paths are organized in a ONE-chain. In other words, we look at how each minimal path from s to t in S (or similarly from u to v in U) is characterized by the ONE-chain. We prove the following general lemma.

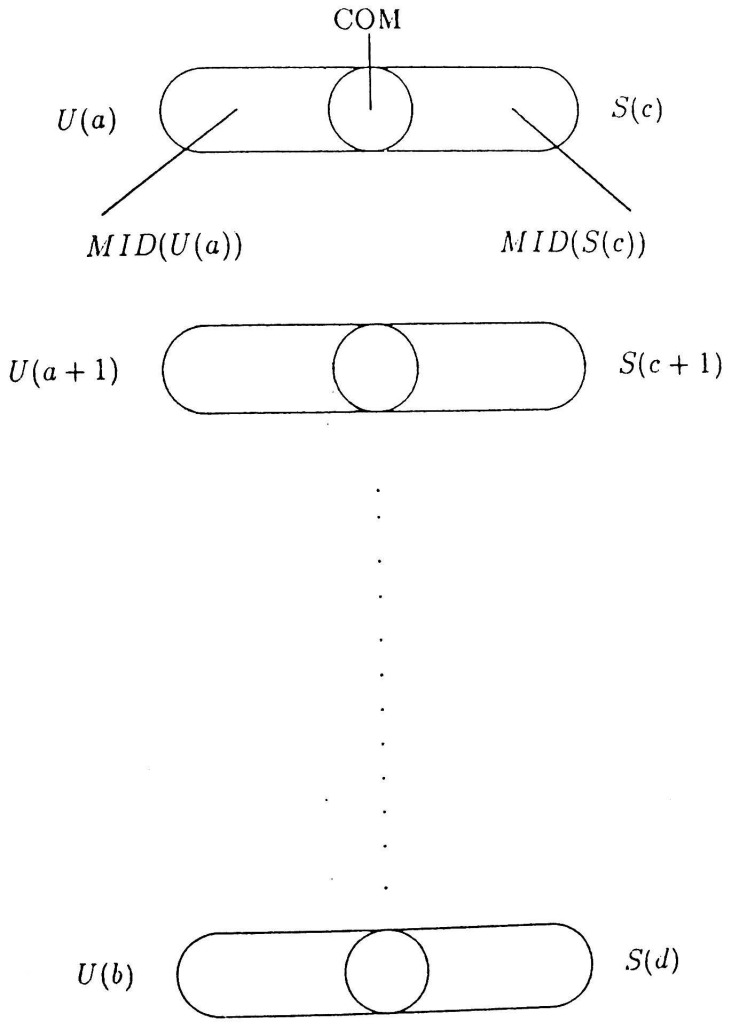

Figure 4. A ONE-chain. (The MID parts for some levels can be empty).

Lemma 4.4. *In a ONE-chain* $U(a) \ldots U(b)$ *any path (shortest)* $[x_a \ldots x_b]$ *with* $x_a \in U(a)$ *and* $x_b \in U(b)$ *has the property that all the vertices which belong to COM (refer Definition* 3.1*) are present consecutively. In other words, the MID vertices if present are at the ends.*

Proof. Assume the contrary. Suppose there is a path such that the COM vertices are not consecutive. That is, there is a MID vertex present between two COM vertices (the placing of the COM vertices in the path does not matter). Let this path be $[x_a \ldots x_i \ldots x_j \ldots x_k \ldots x_b]$, where $x_a \in U(a)$, $x_b \in U(b)$, $x_i \in COM(U(i))$, $x_j \in MID(U(j))$, $x_k \in COM(U(k))$ and $x_b \in U(b)$. Now, there is a path $[y_c \ldots x_i \ldots x_j \ldots x_k \ldots y_d]$ starting from $S(c)$ and ending in $S(d)$, because x_i and x_k also belong to S (they are COM vertices). Hence by Lemma 2.1 this path belongs to S, which implies that x_j is a COM vertex. We arrive at a contradiction. ■

Definition 4.2. We say that there is a *JUMP* between j and $j+1$ in a ONE-chain $U(a) \ldots U(b)$ if $\mid MID(U(j)) \mid \geq 1$ and there is no edge between any vertex belonging to $MID(U(j))$ and any belonging to $MID(U(j+1))$. The reverse of a JUMP (JUMP in the other direction) is called an *RJUMP* which is said to exist between j and $j+1$ if $\mid MID(U(j+1) \mid \geq 1$ and there is no edge between any vertex belonging to $MID(U(j+1))$.

We see from the above definition that a JUMP obviously exists between j and $j+1$, if $U(j)$ is of type ONE-B and $U(j+1)$ is of type ONE-A (because $MID(U(j+1)) = \emptyset$). An RJUMP always exists between j and $j+1$ if $U(j)$ is of type ONE-A and $U(j+1)$ is of type ONE-B. What we have said here equally applies to the S strand.

Corollary 4.4.1. *Let $U(a) \ldots U(b)$ be a ONE-chain. If a JUMP occurs between p and $p+1$, then every vertex in $MID(U(j))$ is adjacent to some vertex in $MID(U(j-1))$, for $a < j \leq p$.*

Proof. Assume the contrary. If some vertex, say x in $MID(U(j))$, $a < j \leq p$, is not adjacent to any vertex in $MID(U(j-1))$, then it should be adjacent to some vertex (say y) in $COM(U(j-1))$. But this means that a path exists in U with a MID vertex (x) inbetween y (a COM vertex) and another COM vertex of $COM(U(p+1))$, because of the JUMP. This contradicts Lemma 4.4. ■

Similarly, if a RJUMP exists between q and $q+1$, every vertex belonging to $MID(U(j))$ is adjacent to some vertex belonging to $MID(U(j+1))$, for $q+1 \leq j < b$.

Corollary 4.4.2. *Let $U(a) \ldots U(b)$ be a ONE-chain. Let there be a JUMP between p and $p+1$. Then for every j, $a < j \leq p$, there is an edge between every vertex in $COM(U(j))$ and some vertex in $MID(U(j-1))$.*

Proof. Take any vertex x in $COM(U(j))$. If it is not adjacent to any vertex in $MID(U(j-1))$, it must be adjacent to some vertex in $COM(U(j-1))$ (say y). Since $a < j \leq p$, by Corollary 4.4.1 there must exist an edge between a vertex (say z) in $MID(U(j))$ and a vertex (say t) in $MID(U(j-1))$. Now consider the 4-cycle (x, y, z, t, x). One of the chords (x, z) or (y, t) must exist. But by Lemma 4.4 the edge (y, t) cannot exist, because it violates the 'consecutive COM vertices' property. Hence the edge (x, z) should exist. Thus x is adjacent to some vertex in $MID(U(j-1))$. A contradiction. ■

A similar corollary can be stated for the case of the RJUMP.

Corollary 4.4.3. *In a ONE-chain, there cannot be 2 JUMP's (or 2 RJUMP's), nor there can be an RJUMP before a JUMP (i.e if a JUMP occurs between p and $p+1$, then an RJUMP cannot occur between q and $q+1$, where $q < p$).*

Proof. Follows from Lemma 4.4 and Corollaries 4.4.1 and 4.4.2, for if there are 2 JUMP's (or 2 RJUMP's) or if a RJUMP exists before a JUMP, it implies that there would be an edge between a vertex in $MID(U(j))$ (where $j \leq p$) and a vertex in $COM(U(j-1))$. ■

The above corollary implies that the most general form of a ONE-chain has a JUMP and an RJUMP with the latter following the former. See Figure 5. A ONE-chain need not have a JUMP or an RJUMP. It can also have only one of them. Hereafter, we consider the ONE-chain in its most general form i.e with both a JUMP and an RJUMP.

In a ONE-chain $U(a) \ldots U(b)$, if a JUMP occurs between p and $p+1$ and a RJUMP occurs between q and $q+1$ then, we can call the U's from $U(a)$ till $U(p)$ as the pre-JUMP sequence and the U's from $U(q+1)$ till $U(b)$ as the post-RJUMP sequence. In the pre-JUMP sequence of a ONE-chain, every vertex in $MID(U(j))$ is adjacent to some vertex in $MID(U(j-1))$ (Corollary 4.4.1). Similarly, in the post-RJUMP sequence, every vertex in $MID(U(k))$ is adjacent to some vertex in $MID(U(k+1))$. Hence, we can form a path consisting entirely of MID vertices (MID vertices are disjoint with the other set) in the pre-JUMP and post-RJUMP sequences. The inbetween vertices must be all COM vertices. This inbetween sequence between JUMP and RJUMP, we call as the COM region. This is the region where really a *collision* between paths can occur. A *region* is nothing but a set of consecutive levels.

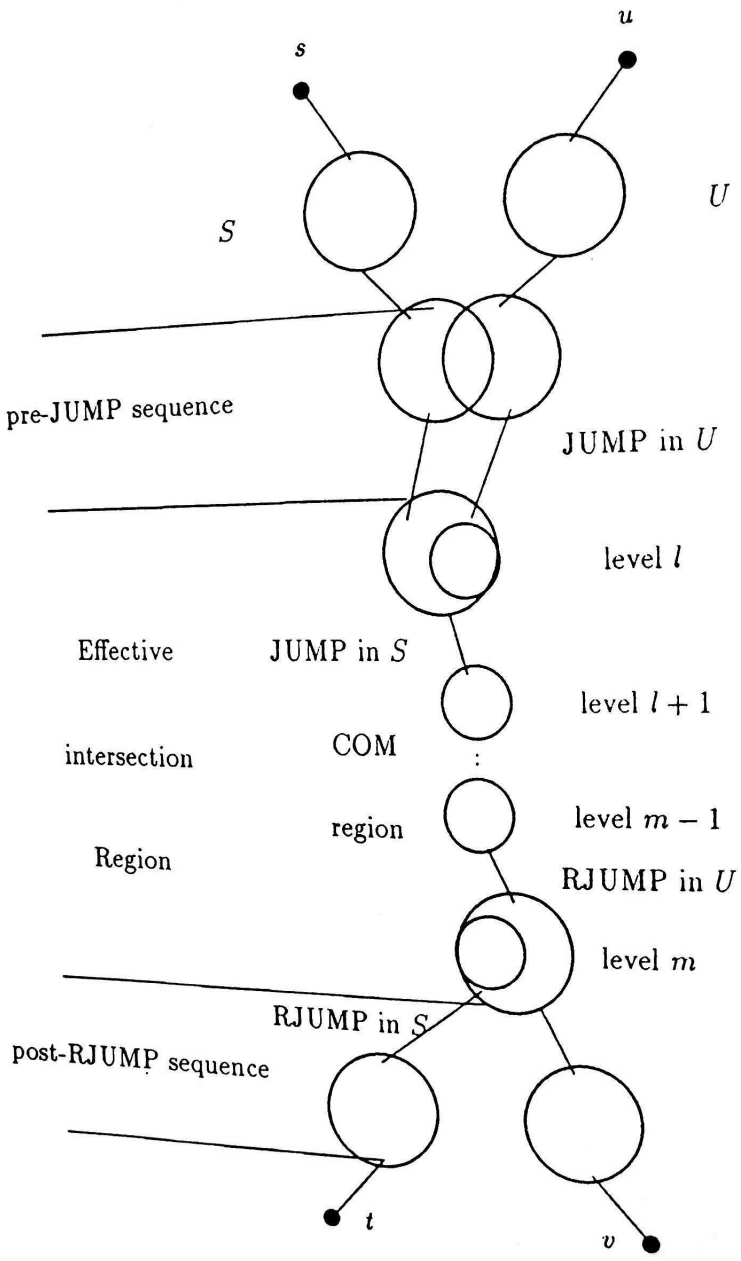

Figure 5. The general form of the ONE-chain.

Since we are interested in finding two disjoint paths, the idea is to construct each of the path using as much of MID vertices as possible. In the case where we can traverse the entire ONE-chain using only MID vertices of both S

and U, we have two minimal disjoint paths easily. This is possible only if there are no JUMPS or RJUMPS. In the general case where we assume that there is a JUMP and an RJUMP, we have to include some COM vertices to traverse the ONE-chain. In the following section, carrying on the points made above we reduce the disjoint path problem to a *bfs* problem whose solution yeilds the disjoint paths in a ONE-chain.

5. The BFS Problem

Definition 5.1. (Definition of a general BFS problem) Given the levels $X_0 \dots X_l$ of a graph we should find k vertex/edge disjoint paths $x_0, x_1, \dots, x_l$ and $y_0, y_1, \dots, y_l$ such that $x_0, y_0 \in X(0)$, $x_l, y_l \in X(l)$.

We now solve the above problem, keeping in mind that the underlying graph is chordal. As can be noted we give an algorithm to find k vertex/edge-disjoint paths instead of just two, because the bfs problem can be encountered in many situations where a general algorithm is needed.

Algorithm 5.1. (k vertex disjoint paths)

1. Number the vertices in each level according to the numbering imposed by Lemma 2.3. That is in a level X_i, if there are n vertices they could be ordered $1 \dots n$.

2. In X_0 choose the k highest numbered vertices. These are the ones which have the largest projection (refer Definition 2.3) on its next level (and also subsequent levels). If there are less than k vertices in this level then output-'no k vertex disjoint paths are possible'. Stop.

3. Suppose we have got the k paths till level $X(i)$. Let the vertices at level $X(i)$ be $x_1, x_2, \dots, x_k$, where x_1 is lower in ordering compared to x_2 etc. imposed by the linear ordering of Lemma 2.3. Now for continuing the path from x_1, choose the highest numbered vertex in its projection on the next level. If a distinct vertex in the next level cannot be chosen for some vertex, then output-'no k vertex disjoint paths are possible'. Stop.

4. If level $X(l)$ (last level) has been reached, output the k vertex disjoint paths constructed and stop. Else goto step 3.

Lemma 5.1. *Algorithm 5.1 finds k vertex disjoint paths in the bfs problem in $O(\mid V \mid + \mid E \mid)$ time and space.*

Proof. We give a natural inductive proof.
Base case. Handled by step 2. If there are not k vertices in the first level, then a negative result is output.

Inductive case. Suppose we have constructed the k paths till level $X(i)$. Let the vertices at level $X(i)$ be $x_1, x_2, \ldots, x_k$, where x_1 is lower in order compared to x_2, etc. That is on the next level, $Proj(x_1) \subset Proj(x_2) \subset \cdots \subset Proj(x_k)$. By induction, at level $X(i)$ the k vertices chosen are the k highest numbered vertices, which means that the projection of x_1 (the lowest numbered among the k vertices) is a superset of the projection of any other vertex unchosen at this level. In step 3 where we extend the path to the next level, we preserve the above property. Because for every vertex x at level $X(i)$ we choose the highest numbered vertex within its (x's) projection in the next level.By starting this procedure from the lowest ordered vertex (x_1), we make sure that the lower ones get the preference in selection (the higher ones anyway have a wider superset choice to choose from). Thus at each level we get the k highest numbered vertices. If for some vertex x_i we cannot choose it successor, it means that all the vertices in its projection have already been chosen and hence we have to stop. The time taken for step 1 (numbering vertices at each level) is $O(\mid V \mid + \mid E \mid)$, the same time that it takes for a bfs search. The numbering can be done by just comparing the cardinality of its projection on the next level. For other steps, the time taken is within $O(\mid V \mid + \mid E \mid)$. Hence overall time complexity is $O(\mid V \mid + \mid E \mid)$. Also it is easy to see that only $O(\mid V \mid + \mid E \mid)$ space is used. ■

Algorithm 5.2. (k edge disjoint paths)

1. Same as step 1 of Algorithm 5.1.

2. In $X(0)$ choose the k highest numbered vertices. However if there are less than k vertices, we do a second round of choosing starting from the highest numbered vertex, and we do as many rounds needed to select the k vertices. Of course, a vertex will be repeated, which means that multiple edges will emanate from this vertex. If it is still not possible to choose, we output: k edge disjoint paths not possible. Stop.

3. Suppose we have formed edge disjoint paths till level $X(i)$. To extend the paths to the next level we do the same as in step 3 of Algorithm 5.1. The only difference is that the vertices are repeated. That is in the first round for each vertex (starting from the lowest ordered) we choose that edge which goes to the highest numbered vertex in its projection and we do the same procedure for the higher numbered vertices. In the second round, we again start from the lowest ordered *among* the repeated vertices and repeat the procedure. If distinct edges cannot be chosen output: no k edge disjoint paths are possible. Stop.

4. If level $X(l)$ (last level) has been reached, output the k edge disjoint

paths constructed and stop. Else goto step 3.

Lemma 5.2. *Algorithm* 5.2 *finds* k *edge disjoint paths in bfs problem in* $O(\mid V \mid + \mid E \mid)$ *time and space.*

Proof. The proof is very similar to that of Lemma 5.1. ∎

The purpose of solving the general bfs problem (k disjoint paths instead of 2) is that this comes in handy when we generalize for k vertex/edge minimal disjoint paths in a later section.

Now we reduce the problem of finding 2 vertex/edge disjoint paths in a ONE-chain to the bfs problem we have just solved.

Lemma 5.3. *The problem of finding 2 vertex/edge disjoint paths in a ONE-chain reduces to the bfs problem.*

Proof. We consider the general form of the ONE-chain (Figure 5) where there is a JUMP and an RJUMP for both S and U. For S, let the JUMP exist between $s1$ and $s1+1$ and the RJUMP between $s2$ and $s2+1$. Similarly for U, let the JUMP exist between $u1$ and $u1+1$ and the RJUMP between $u2$ and $u2+1$. See Figure. Our aim now is to construct two disjoint paths, one belonging to S and the other belonging to U. We try to use as much MID vertices as possible. This is naturally accomplished in the pre-JUMP and post-RJUMP sequences. But in the COM region we have to use vertices which are common to both S and U to complete the disjoint paths. Consider the levels which fall in the COM region of both S and U. We call this as the effective intersection region, because once we have constructed vertex/edge disjoint paths in this region, we can easily extend in either direction to get the full disjoint paths. In figure the effective intersection region is between levels l and m. We have two cases:

Case (a). Take the COM of level l and the subsequent levels till the COM of level m. We have a bfs problem. Apply Algorithms 5.1 and 5.2 (with $k = 2$) to get 2 vertex/edge disjoint paths. Each of them can be extended in either direction, in their respective sequences (S or U) to get the full disjoint paths from s to t and from u to v. If we get a negative result in the bfs problem, we apply case (b).

Case (b). In case (a) we used only the COM vertices of level l and m. We didn't use the MID vertices of these levels. In this case, we take care of all vertices in levels l and m and also reduce the size of the problem, giving a natural recursive solution. Construct a smaller ONE-chain as follows. Take the projection of $MID(U(l))$ on the next level i.e $COM(U(l+1))$. The vertices which are not in the projection form the new COM part at this

level. Do a bfs search till level $m-1$. Similarly, repeat the procedure for the S strand. We now get a new ONE-chain, to which we again apply case (a). Thus we have to solve the bfs problem with a smaller size. ■

The time complexity of the above procedure can be calculated as follows. There can be at most be $\mid V \mid$ reductions in the size of the problem, which is the worst case. In each intermediate step the amount of time taken is $O(\mid V \mid + \mid E \mid)$ because this is the time needed for solving the bfs problem (Lemma 5.2) and for a bfs search. Hence, the overall complexity is $O(\mid V \mid \ \star \ (\mid V \mid + \mid E \mid))$ which is $O(\mid V \mid \star \mid E \mid)$.

However, if we want to find only *edge* disjoint paths we can improve the complexity to $O(\mid V \mid + \mid E \mid)$ as follows. At level l, if the downward special vertex in the COM set (this vertex is connected to all the vertices in the next level), take this as the starting vertex. Otherwise, take two downward special vertices which belong to the MID parts of S and U at level l. Now solve the bfs problem for 2 edge disjoint paths starting from $COM(U(l+1))$ till $COM(U(m-1))$. Depending on the starting vertex at level l extend the path to level m by adding edges to the appropriate upward special vertices at level m. The time taken is only the time to solve one bfs problem and hence is $O(\mid V \mid + \mid E \mid)$.

Theorem 5.1. *The minimal vertex-disjoint path problem can be solved when there is a C2-type intersection in a chordal graph in* $O(\mid V \mid\mid E \mid)$ *time. The corresponding complexity for the edge-disjoint version is* $O(\mid V \mid + \mid E \mid)$.

Proof. Follows from the above discussion and the fact that C2.1 and C2.2-type intersections can be easily implemented in a similar way as mentioned in Theorem 3.1 in linear time. ■

Combining Theorems 3.1 and 5.1 we can state the following final result.

Theorem 5.2. *In a chordal graph the 2 minimal edge-disjoint path problem can be solved in* $O(\mid V \mid + \mid E \mid)$ *time and the 2 minimal vertex-disjoint path problem in* $O(\mid V \mid \star \mid E \mid)$ *time.*

6. Conclusion and Open Problems

In this paper, we have given efficient algorithms for the minimal disjoint path problem on chordal graphs, which not only gives disjoint paths for communication, but also minimises the time required for communication. Whereas the edge-disjoint version on chordal graphs admits a linear solution, the vertex-disjoint version does not. It would be interesting to see whether

the vertex-disjoint version of the problem has a linear solution when the input graph is restricted to the class of interval graphs, a subclass of chordal graphs. All we can say is that there is a simple linear algorithm for this problem on proper interval graphs—a much smaller class of graphs.

Also it would be interesting to study the complexity of this problem on general graphs, as well as other classes of perfect graphs such as permutation graphs and comparability graphs.

References

[G 80] M.C. Golumbic, Algorithmic Graph Theory and Perfect Graphs (Academic Press, 1980).

[K 75] R.M. Karp, *On the computational complexity of combinatorial problems*, Networks **5** (1975) 45–68.

[O 80] T. Ohtsuki, *The two disjoint path problem and wire routing design*, in: Proc. of the 17th Symp. of Res. Inst. of Electrical Comm. (1980) 257–267.

[PS 78] Y. Perl, Y. Shiloach, *Finding two disjoint paths between two pairs of vertices in a graph*, J. of the ACM **25** (1978) 1–9.

[RS 86] N. Robertson, P.D. Seymour, *Graph minors XIII. The disjoint paths problem*, Manuscript 1986.

[S 80] Y. Shiloach, *A polynomial solution to the undirected two paths problem*, J. of the ACM **27** (1980) 445–456.

[S 89] A. Schwill, *Shortest edge-disjoint paths in graphs*, in: Proc. of the 6th STACS (1989) 505–516.

[S 90] A. Schwill, *Nonblocking graphs: Greedy algorithms to compute disjoint paths*, in: Proc. of the 7th STACS (1990) 250–262.

[KPS 91] S.V. Krishnan, C. Pandu Rangan, S. Seshadri, A. Schwill, *Two Disjoint Paths in Chordal graphs*, Technical report, 2/91, February 1991, University of Oldenburg, Germany.

Received 9 May 1995