Chapter 4

Lecture 4

Advanced Topics in Graph Algorithms
©Tel-Aviv University
Spring Semester, 1994

Lecturer	:	Ron Shamir
Scribe	:	Izik Pe'er
Lecture 4	:	May 12

4.1 Recognizing Triangulated Graphs

4.1.1 Generating a PEO

In lecture 3 we have introduced the *MCS* algorithm for generating a PEO for a triangulated graph, and proved that it is a linear time algorithm. We shall prove the correctness of this algorithm.

Theorem 4.1 *(Tarjan and Yannakakis 85)*
A graph $G = (V, E)$ is triangulated if and only if the MCS *algorithm produces a permutation* σ [1] *which is a PEO.*

Proof:

$\Rightarrow$ Trivial.

$\Leftarrow$ We shall prove the claim by induction on the number of vertices n.

[1]We shall refer to each permutation σ simply as a one-to-one function from $\{1, \ldots, n\}$ to V, so that $\sigma(1)$ is the first vertex in the order, $\sigma(2)$ is the second, etc. Surely, for each vertex v, $\sigma^{-1}(v)$ is v's serial number in the order.

The theorem is trivially true for $n = 1$. Assume correctness for every graph with less than n vertices. Let $G(V, E)$ be a triangulated graph with n vertices, and let σ be the permutation generated by the *MCS* algorithm with input G. According to the induction hypothesis it suffices to show that the vertex $v = \sigma(1)$ is simplicial.

Claim 4.2 *G cannot contain an induced chordless path* $\mu = [u = v_0, v_1, \ldots, v_k, w]$ *for* $k \geq 1$ *so that*

$$\forall i, 1 \leq i \leq k : \sigma^{-1}(v_i) < \sigma^{-1}(u) < \sigma^{-1}(w) \tag{4.1}$$

Proof: Assume, by contrary, that such a path exists. Let μ be a path satisfying 4.1, such that $\sigma^{-1}(u)$ is maximal. u was numbered before v_k, despite the fact that w is a neighbor of v_k and not of u. According to the maximum cardinality rule, this implies there exists a vertex x, such that x is a neighbor of u and not of v_k, and x was numbered before u, as in figure 4.1.

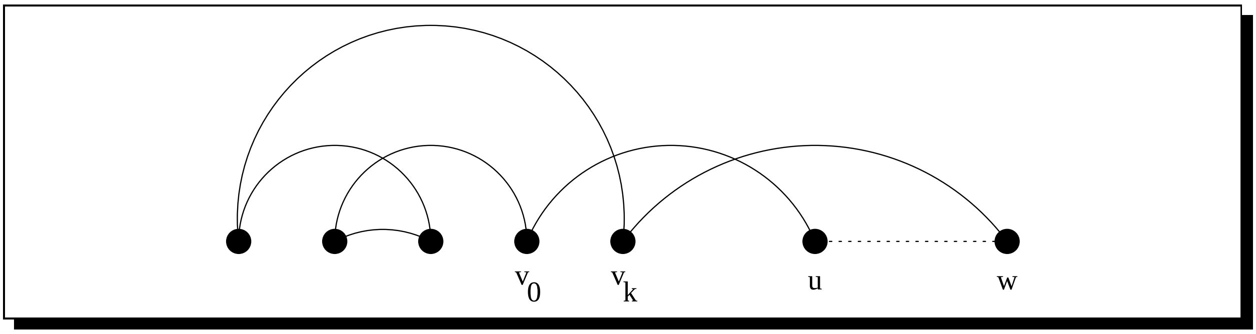

Figure 4.1: The path μ. Vertices appear ordered from left to right according to the *MCS* algorithm output.

Let j be the maximal index such that v_j is a neighbor of x. Surely j is well defined, since $u = v_0$ is a neighbor of x, and $j < k$, by the definition of x. It follows from the maximality of j that the induced path $\mu' = [x, v_j, \ldots, v_k, w]$ is chordless ($xw \notin E$, otherwise $\mu' = [x, v_j, \ldots, v_k, w, x]$ is a chordless cycle of length ≥ 4, contradicting G's chordality).

If $\sigma^{-1}(x) < \sigma^{-1}(w)$ (as in figure 4.3), then for each $j \leq i \leq k$, $\sigma^{-1}(v_i) < \sigma^{-1}(u) < \sigma^{-1}(x)$, so the path μ' meets the condition 4.1, and contradicts the maximality of $\sigma^{-1}(u)$, since $\sigma^{-1}(x) > \sigma^{-1}(u)$. Hence $\sigma^{-1}(x) > \sigma^{-1}(w)$ (as in figure 4.4), and the path $\mu'' = [w, v_k, \ldots, v_j, x]$ satisfies 4.1, contradicting the maximality of $\sigma^{-1}(u)$.

It follows that there is no path satisfying 4.1. ■

Back to the proof of Theorem 4.1:

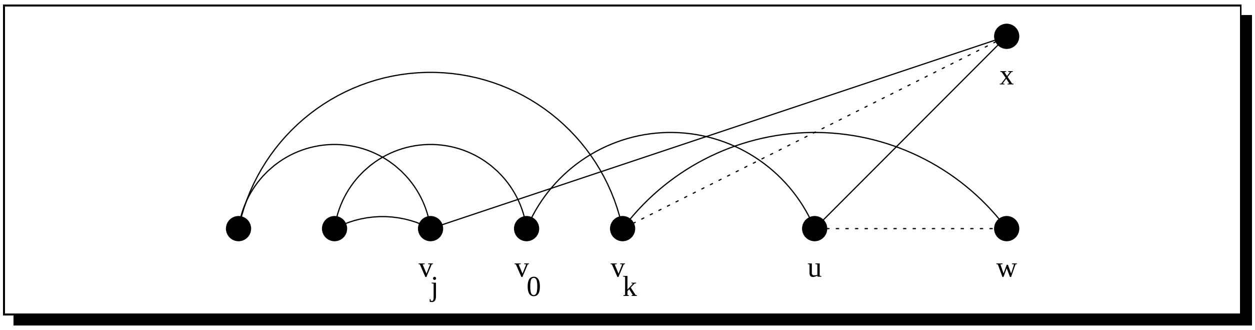

Figure 4.2: The vertices x and v_j

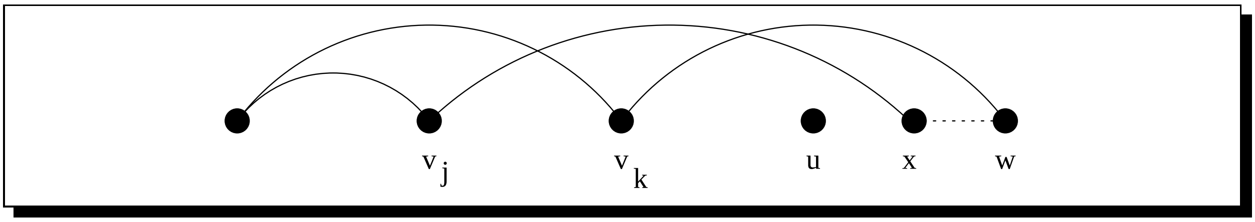

Figure 4.3: The path μ'

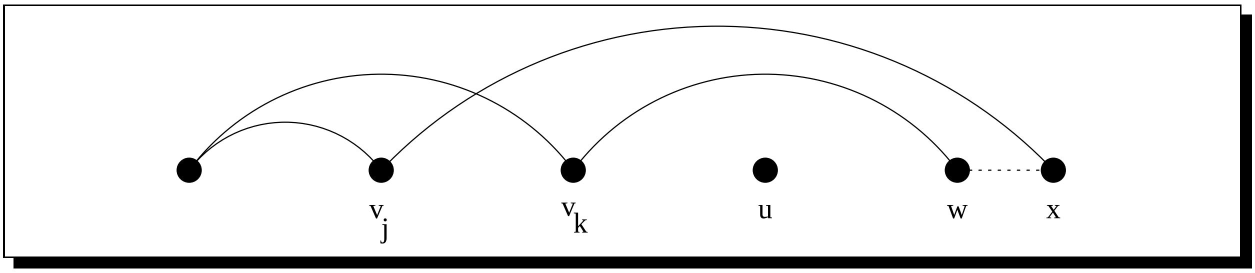

Figure 4.4: The path μ''

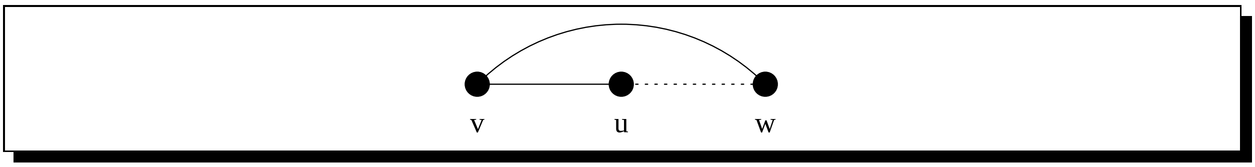

Figure 4.5: A forbidden structure in a chordal graph

Denote $v = \sigma^{-1}(1)$, and assume, by contrary that, v is not a simplicial vertex. Then there exist $u, w \in Adj(v)$ such that $uw \notin E$, and $\sigma^{-1}(u) < \sigma^{-1}(w)$ (as in figure 4.5. It follows that the path $[u, v, w]$ satisfies 4.1 of Claim 4.2, yielding a contradiction.

■

4.1.2 Testing an Elimination Scheme

In order to recognize whether a graph is triangulated, we can use the *MCS* algorithm to get an elimination scheme, but we have yet to describe how we can test whether this elimination scheme is a PEO.

Naive Algorithm

Simulate the elimination process, and for each vertex to be eliminated, check whether its noneliminated neighbors are a clique. This strategy results in a total time complexity of $O(|V| \cdot |E|)$.

Efficient Algorithm

We can improve complexity if we collect a list of desired neighbors for each vertex, and check all the members in this list only once.

```
begin
1. for each vertex v do A(v) ← ∅
2. for i ← 1 to n − 1 do
3.         v ← σ(i)
4.         X ← {x ∈ Adj(v)|σ^{-1}(v) < σ^{-1}(x)}
5.         if X ≠ ∅ then
6.                 u ← σ(min{σ^{-1}(x)|x ∈ X})
7.                 concatenate X \ {u} to A(u)
           end if
8.         if A(v) \ Adj(v) ≠ ∅ then return FALSE
   end for
9. return TRUE
end
```

Figure 4.6: Code for PEO testing algorithm

Example

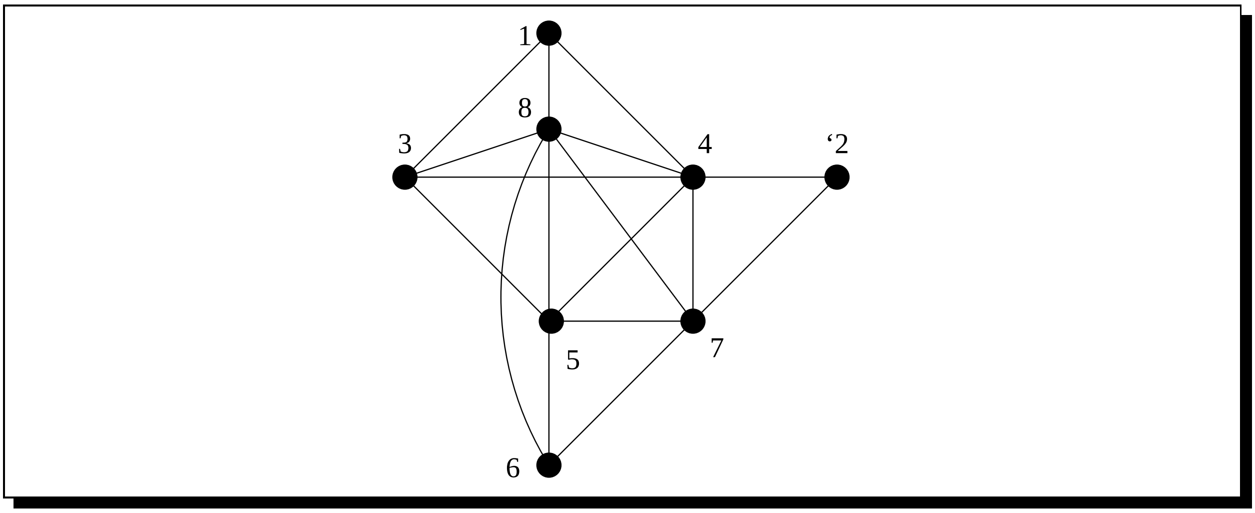

Figure 4.7: An input chordal graph and elimination order

When applying the algorithm in 4.6 to the input graph and elimination order in 4.7, the algorithm does the following iterations:

Iteration 1		$u = 3$	$A(3) = \{4, 8\}$	$X = 3, 4, 8$
Iteration 2	$X = 4, 7$	$u = 4$	$A(4) = \{7\}$	$A(2) \subseteq Adj(2)$
Iteration 3	$X = 4, 5, 8$	$u = 4$	$A(4) = \{7, 5, 8\}$	$A(3) \subseteq Adj(3)$
Iteration 4	$X = 5, 7, 8$	$u = 5$	$A(5) = \{7, 8\}$	$A(4) \subseteq Adj(4)$
Iteration 5	$X = 6, 7, 8$	$u = 6$	$A(6) = \{7, 8\}$	$A(5) \subseteq Adj(5)$
Iteration 6	$X = 7, 8$	$u = 7$	$A(7) = \{8\}$	$A(6) \subseteq Adj(6)$

Correctness of the Algorithm

Theorem 4.3 *The algorithm in figure 4.6 returns* TRUE *if and only if* σ *is a PEO.*

Proof:

$\Leftarrow$ Suppose the algorithm returns *FALSE* on iteration number $\sigma^{-1}(u)$. This may happen only if in stage 8 at that iteration there exists an unnumbered vertex $w \in A(u) \setminus Adj(u)$. The vertex w was added to $A(u)$ at stage 7 of a prior iteration, number $\sigma^{-1}(v)$. It follows that $\sigma^{-1}(v) < \sigma^{-1}(u) < \sigma^{-1}(w)$, for $u, w \in Adj(v)$, but since $uw \notin E$ (as in figure 4.5), σ is not a PEO.

$\Rightarrow$ Suppose σ is not a PEO, and the algorithm returns *TRUE*. Let v be a vertex with maximal $\sigma^{-1}(v)$, such that the set $X_v = \{w | w \geq Adj(v), \sigma^{-1}(v) < \sigma^{-1}(w)\}$ does not induce a clique. The vertex v is well defined, because if all of the X_v's are cliques, then σ is a PEO. Let u be the vertex defined in stage 6 of iteration $\sigma^{-1}(v)$. In stage 7 of this iteration, $X_v \setminus \{u\}$ is added to $A(u)$. Since the algorithm returns *TRUE* in stage 8 of iteration number $\sigma^{-1}(u)$, the condition of stage 8 (i.e. The condition $A(u) \subseteq Adj(u)$) is not met, so every $x \in X_v \setminus \{u\}$ is a neighbor of u.

Since $\sigma^{-1}(v)$ is chosen to be the maximal vertex for which X_v is not a clique, and v is prior to u, X_u is a clique, so all u's neighbors in $X_v \setminus \{u\}$ are adjacent to each other (as in figure 4.8, hence X_v is a clique, and a contradiction follows.

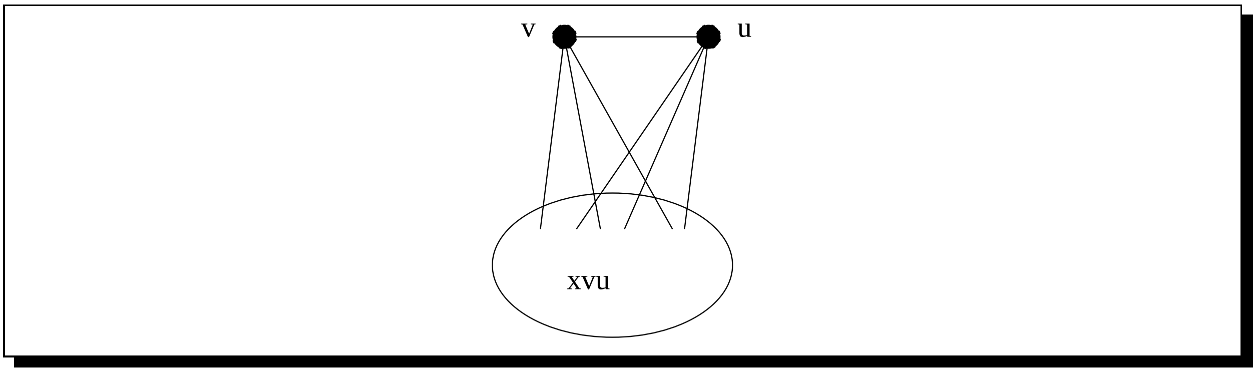

Figure 4.8: $X_v \setminus \{u\}$ is a clique

■

Complexity of the Algorithm

Theorem 4.4 *The algorithm 4.6 runs in time $O(|V| + |E|)$.*

Proof: We shall hold σ and σ^{-1} as vectors, and for each v we shall hold $Adj(v)$ and $A(v)$ as lists, allowing duplicates to be concatenated to $A(v)$. Stage 8 of iteration $\sigma^{-1}(v)$ can be implemented in $O(|Adj(v)| + |A(v)|)$ by holding a bit vector $Test$ of length n, and the condition $A(v) \setminus Adj(v) = \emptyset$ is implemented in the following way :

1. for each $w \in Adj(v)$, set the bit $Test(w)$ to 1.
2. for each $w \in A(v)$, if $Test(w)$ is 0, return *FALSE*.
3. for each $w \in Adj(v)$, clear the bit $Test(w)$ to 0.
4. return *TRUE*

The resulting overall complexity is $O(|V| + \sum v \in V|Adj(v)| + \sum v \in V|A(v)|)$, where $|A(v)|$ is the last (maximal) size of $A(v)$, including the duplicates. For each $v \in V$, $Adj(v)$ adds vertices only to one list $A(u)$, and the number of added vertices is at most $|Adj(v) - 1|$. Therefore $\sum v \in V|A(v)| < \sum v \in V|Adj(v)| = O(|E|)$ and the overall time complexity is $O(|V| + |E|)$. ■

Corollary 4.5 *There is a linear time algorithm for checking whether a given graph is triangulated.*

4.2 Triangulated Graphs as Intersection Graphs

In Lecture 1 it was shown that every interval graph is triangulated, and that interval graphs are characterized as intersection graphs. Is there such a characterization for triangulated graphs as well? We shall see that a graph G is triangulated if and only if G is an intersection graph of a family of subtrees of a tree. The tree containing all the subtrees is called the *host tree.*

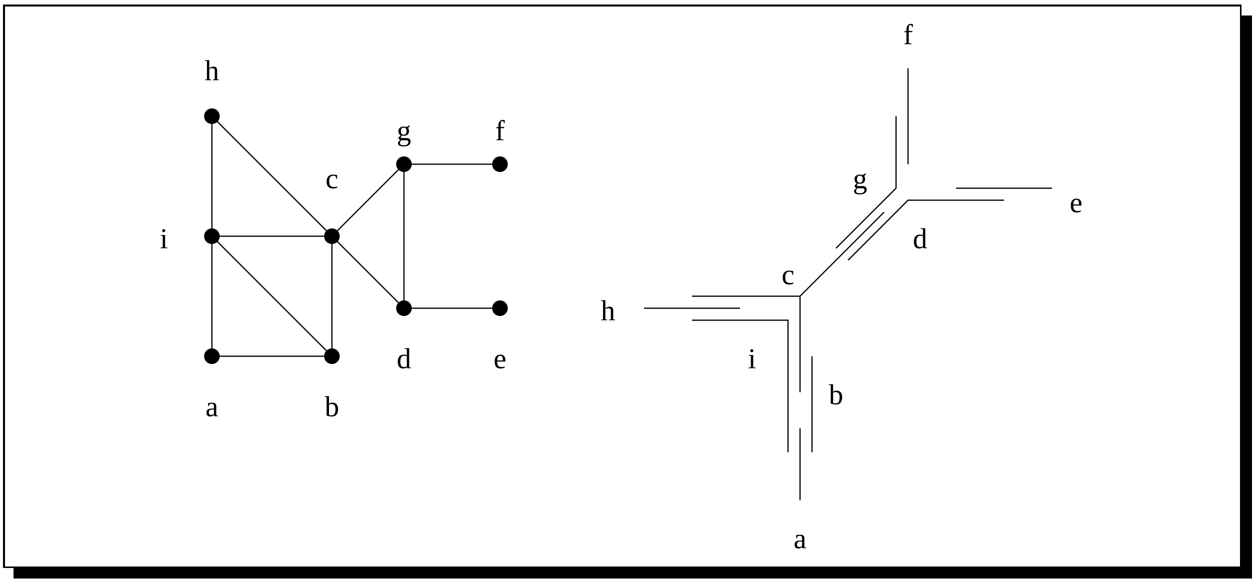

Figure 4.9: The intersection graph of a set of subtrees of a host tree is triangulated

4.2.1 Evolutionary Trees

An application for this characterization of triangulated graphs is the evolutionary tree, also called the phylogenetic tree. This model is formulated in the study of evolution, where we wish to determine the way a group of species evolved in time to their current properties. We use the following assumptions :

- There exists a single ancient species, from which all the other species have evolved.
- Each ancient species might have split into two subspecies, because of a mutation changing a property of one of the subspecies.
- Mutations are rare, so each mutation can occur only once in the tree.

We can therefore define the evolutionary tree, which is a rooted binary tree. In this tree, every vertex corresponds to a species (every leaf corresponds to a current species). Because of the third assumption for every property the set of all the species having that property corresponds to a set of tree vertices that induces a *connected subgraph*, i.e. a subtree.

We can build a graph with vertices that match properties. Two vertices will be adjacent if their properties appear together in a certain species (as in figure 4.10). This graph must be *triangulated.*

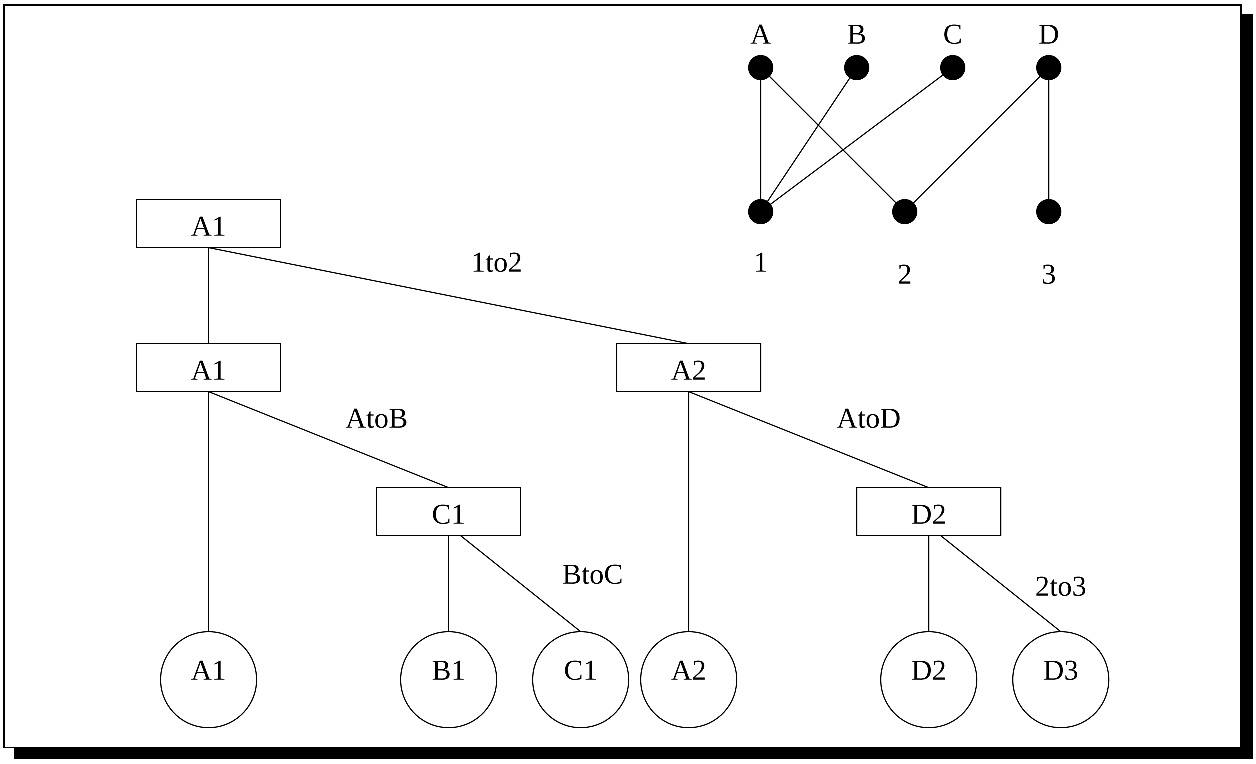

Figure 4.10: An evolutionary tree describing the properties $A, B, C, D, 1, 2, 3$, and the corresponding triangulated intersection graph

4.2.2 Triangulated Graphs as Intersection Graphs

Definition Let $S = \{T_i\}_{i \in I}$ be a family of subsets of a set T. S has the *Helly* property if and only if for every $J \subseteq I$, if $\forall i, j \in J : T_i \cap T_j \neq \emptyset$ then $\bigcap_{j \in J} T_j \neq 0$. (figure)

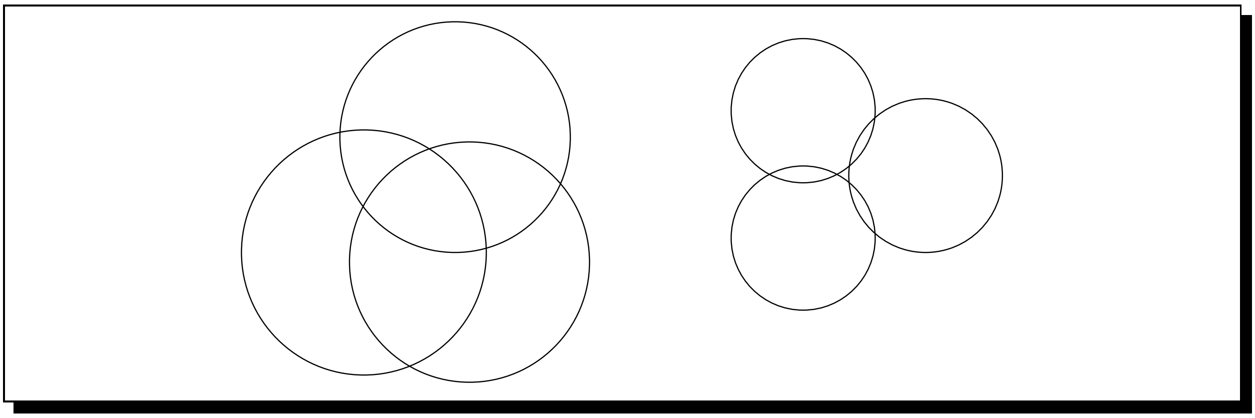

Figure 4.11: The left family has the Helly property, while the right family does not

Lemma 4.6 *Suppose that $\{T_i\}_{i \in S}$ is a set of subtrees of T, and that there exist points a, b, c on T, such for each $i \in S$, T_i contains at least two of them, then $\bigcap_{i \in S} T_i \neq \emptyset$*

Proof: Denote: P_1 the simple path from a to b in T, P_2 the simple path from b to c in T, and P_3 the simple path from c to a in T. Since T is a tree, $P_1 \cap P_2 \cap P_3 \neq \emptyset$ (otherwise $P_1 - P_2 - P_3$ contains a cycle). Every T_i contains at least one of the paths $P_i, i = 1, 2, 3$. Hence, $\bigcap_{i \in S} T_i \supseteq P_1 \cap P_2 \cap P_3 \neq \emptyset$.

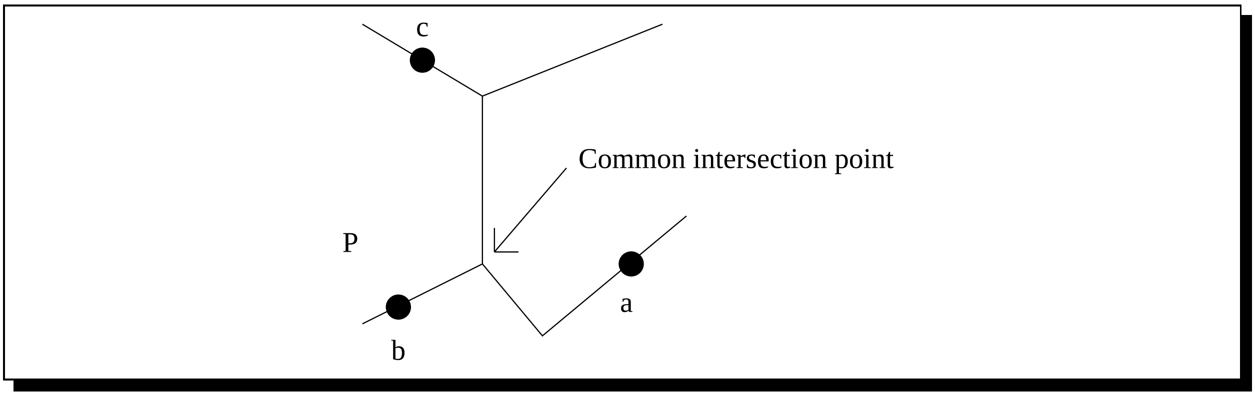

Figure 4.12: The paths P_1, P_2, and P_3 must intersect

■

Theorem 4.7 *A family of subtrees of a host tree has the* Helly *property.*

Proof: We shall prove the claim by induction on k, the size of the family J. For $k = 2$ the theorem is trivially true. Assume correctness for $|J| \leq k$. Let J be a family of $k+1$ trees : $\{T_1, \ldots, T_{k+1}\}$, each pair of which having a non-empty intersection. By the induction hypothesis:

- $\bigcap_{j=1}^{k} T_j \neq \emptyset$ so there exists $a \in \bigcap_{j=1}^{k} T_j$.
- $\bigcap_{j=2}^{k+1} T_j \neq \emptyset$ so there exists $b \in \bigcap_{j=2}^{k+1} T_j$.
- In addition, by assumption, $T_1 \cap T_{k+1} \neq \emptyset$ so there exists $c \in T_1 \cap T_{k+1}$.

Lemma 4.6 applied for $\{T_1, \ldots, T_{k+1}\}$ and the points a, b, c, yields $\bigcap_{j=1}^{k+1} T_j \neq \emptyset$. ■

Theorem 4.8 *(Walter 72, Gavril 74, Buneman 74)*
Let $G(V, E)$ be an undirected graph. The following statements are equivalent :

(1) G is triangulated.

(2) G is the intersection graph of a family of subtrees of a tree.

(3) There exists a tree $T = (\bar{K}, \bar{E})$ (as in figure 4.13) such that $\bar{K}$ is the set of maximal cliques in G, and for every $v \in V$, if $\bar{K}_v = \{C \in \bar{K} | v \in C\}$, then $T_{\bar{K}_v}$, the subgraph induced by $\bar{K}_v$, is connected (hence a subtree).

Proof:

(3) $\Rightarrow$ (2) Let T be a tree satisfying (3). We shall prove that $vw \in E \Leftrightarrow T_{\bar{K}_v} \cap T_{\bar{K}_w} \neq \emptyset$:

- $\Rightarrow$ If $v, w \in V$, $vw \in E$, then for some maximal clique $C \in \bar{K}$, $v, w \in C$ so $T_{\bar{K}_v} \cap T_{\bar{K}_w} \neq \emptyset$.
- $\Leftarrow$ If $T_{\bar{K}_v} \cap T_{\bar{K}_w} \neq \emptyset$ then there are maximal cliques that contain both w and v, so $vw \in E$.

It follows that G is an intersection graph of the family $\{T_{\bar{K}_v}\}_{v \in V}$ of subtrees of T.

(2) $\Rightarrow$ (1) Let $\{T_{\bar{K}_v}\}_{v \in V}$ be a family of subtrees of a tree T, for which $vw \in E \Leftrightarrow T_{\bar{K}_v} \cap T_{\bar{K}_w} \neq \emptyset$. Suppose that G contains an induced chordless cycle $[v_0, v_1, \ldots, v_{k-1}, v_0]$, for $k > 3$. Let the corresponding subtrees be $T_0, T_1, \ldots T_{k-1}$, respectively. It is obvious that $|i - j| \leq 1 (mod k) \Leftrightarrow T_i \cap T_j \neq \emptyset$. Let a_i be an arbitrary point in $T_i \cap T_{i+1}$, for $i = 0, \ldots, k-1$ (throughout, indices are mod k). We would like to prove that the a_i's form a cycle,

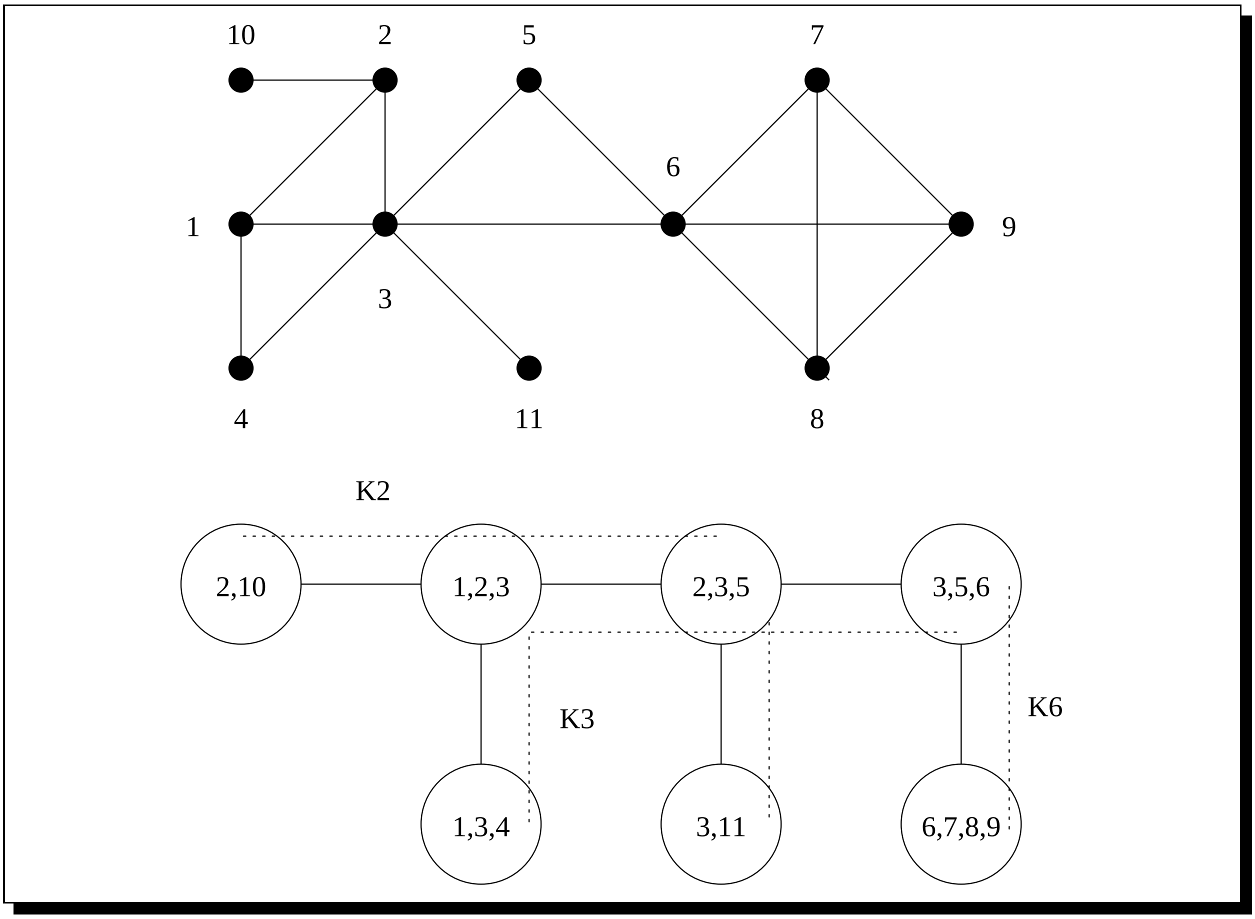

Figure 4.13: The tree $T = (\bar{K}, \bar{E})$

but we have to "throw away" possible branches, which are not on the main cycle, but that the a_i's may lie on. Let b_i be the last common point in the simple paths $a_i \to a_{i-1}$ and $a_i \to a_{i+1}$. The path $a_i \to a_{i-1}$ is in T_i and the path $a_i \to a_{i+1}$ is in T_{i+1}, therefore $b_i \in T_i \cap T_{i+1}$.

Denote by P_i the simple path from b_{i-1} to b_i. It follows that $P_i \subseteq T_i$, hence $P_i \cap P_j = \emptyset$ for every $|i - j| > 1$. On the other hand, $P_i \cap P_{i+1} = \{b_i\}$ for every i. We have shown that $\bigcup_{i=1}^{k} P_i$ is a simple cycle in T, contradicting T being a tree.

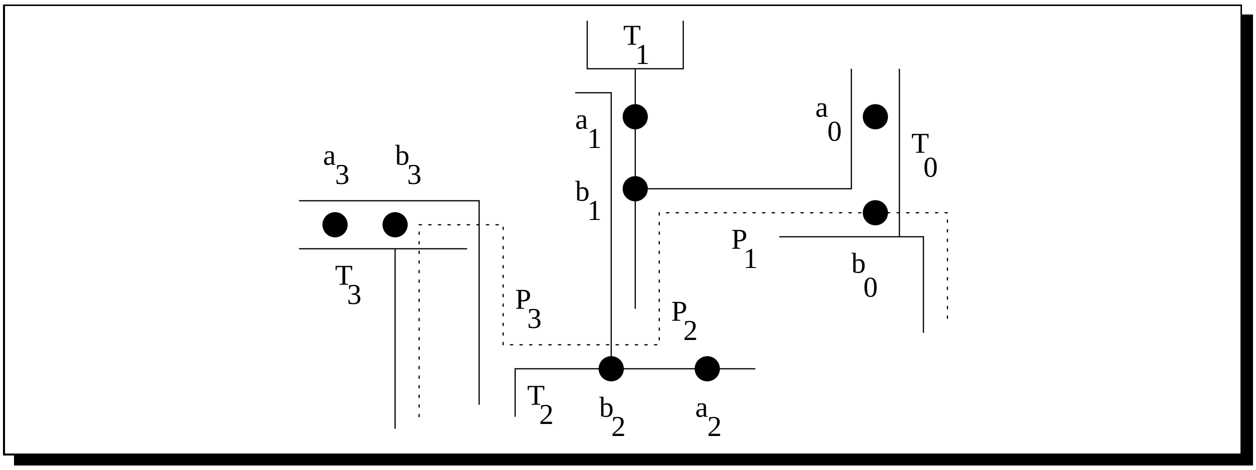

Figure 4.14: The paths $P_0, \ldots, P_{k-1}$ form a cycle

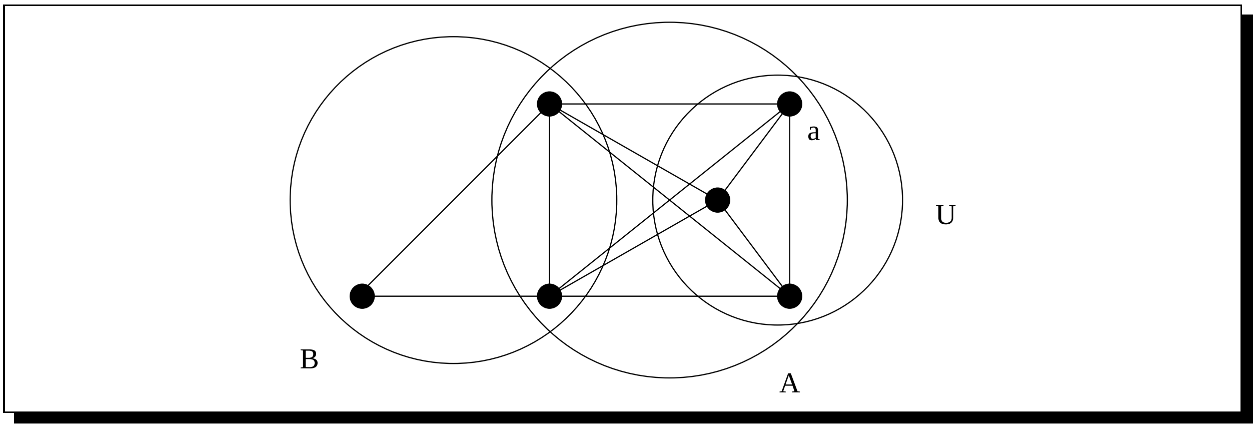

Figure 4.15: The case $B \neq Y$

(1) $\Rightarrow$ (3) We shall prove the claim by induction on n, the number of vertices in G. Assume correctness for every graph with less then n vertices.

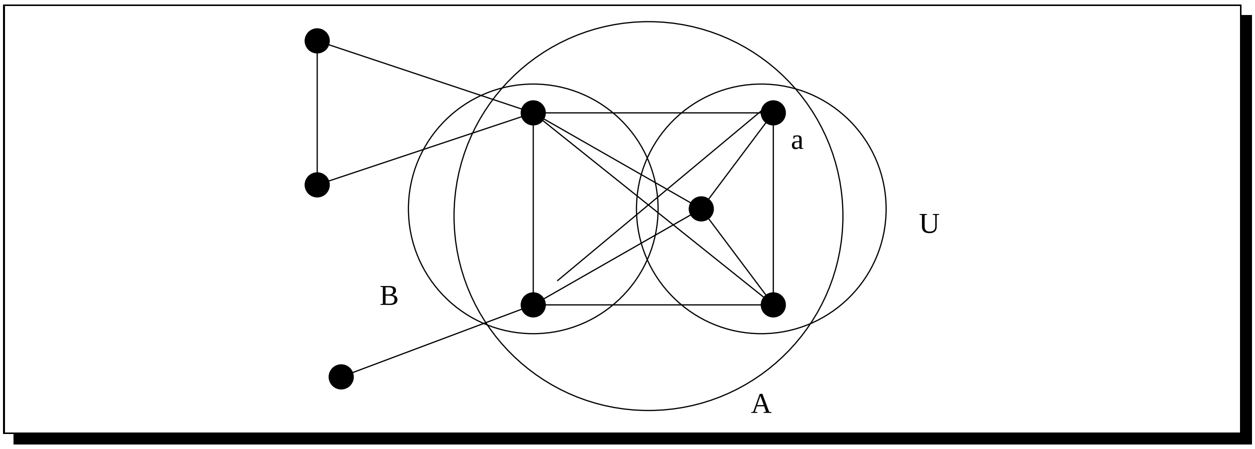

Figure 4.16: The case $B = Y$

- If G is the complete graph then $\bar{K} = V$, and T contains one vertex making the claim trivial (this applies also for $n = 1$).
- If G is not connected with connected components $G_1, \ldots, G_k$, then by the induction hypothesis for each G_i there is a host tree T_i satisfying (3), for $i = 1, \ldots, k$. For each $i = 1, \ldots, k-1$, connect T_i to T_{i+1} by adding an arbitrary edge. The resulting graph is a tree T satisfying (3).
- If G is a connected, not complete, triangulated graph, then let a be a simplicial vertex in G. The set $A = \{a\} \cup Adj(a)$ is a maximal clique in G. Denote $U = \{u \in A | Adj(u) \subset A\}$ and $Y = A \setminus U$.
  * $U \neq \emptyset$ because $a \in U$.
  * $Y \neq \emptyset$ because G is not complete.
  * $V \setminus A \neq \emptyset$ because G is connected and not complete.

  The graph $G' = G_{V \setminus U}$ is triangulated and contains less than n vertices, therefore, by the induction hypothesis there is a tree $T' = (\bar{K}', E')$, whose vertices correspond to the maximal cliques in G', in a way that for each vertex $u \in V \setminus U$, the set $\bar{K}'_v = \{C \in \bar{K}' | v \in C\}$ induces a connected subtree of T'.

  Let us examine the connection between $\bar{K}$ and $\bar{K}'$, the sets of maximal cliques in G and in G', respectively. Let B be the maximal clique in G' that contains Y. There are two cases :

$B = Y$ In that case T can be generated from T' by renaming B as A. For each $y \in Y$: $\bar{K}_y = \bar{K}'_y + \{A\} \setminus \{B\}$ and $\bar{K}_y$ induces the same subtree as $\bar{K}'_y$, because all we did is renaming B as A (as in figure 4.16).

$B \neq Y$ In that case T can be generated from T' by adding a vertex A, and connecting it to B. For each $y \in Y$: $\bar{K}_y = \bar{K}'_y + \{A\}$ and $\bar{K}_y$ induces a connected subtree (as in figure 4.15).

In both cases, for each vertex $u \in U$, $\{A\} = \bar{K}_u$, and for each vertex $v \in V \setminus A$, $\bar{K}_v = \bar{K}'_v$, and these sets induce connected subtrees in T, respectively.

■