Graph Theory: Lecture 7

Chordal Graphs

Michael Lampis

November 25, 2024

Forbidden Subgraph Characterizations

Wider question: how does **local** structure lead to **global** structure?

- A graph is a forest if and only if it has no C_k (induced) subgraph.
- A graph is bipartite if and only if it has no C_{2k+1} (induced) subgraph.
- A graph is planar if and only if it has no $K_{3,3}, K_5$ topological minor.

Forbidden Subgraph Characterizations

Wider question: how does **local** structure lead to **global** structure?

- A graph is a forest if and only if it has no C_k (induced) subgraph.
- A graph is bipartite if and only if it has no C_{2k+1} (induced) subgraph.
- A graph is planar if and only if it has no $K_{3,3}, K_5$ topological minor.

Are the first two statements above still true for **induced** subgraphs?

Forbidden Subgraph Characterizations

Wider question: how does **local** structure lead to **global** structure?

- A graph is a forest if and only if it has no C_k (induced) subgraph.
- A graph is bipartite if and only if it has no C_{2k+1} (induced) subgraph.
- A graph is planar if and only if it has no $K_{3,3}, K_5$ topological minor.

In other words:

- If I promise you that a small (bad) structure H does not appear in a larger graph G, what (else) does this tell us about G?

Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

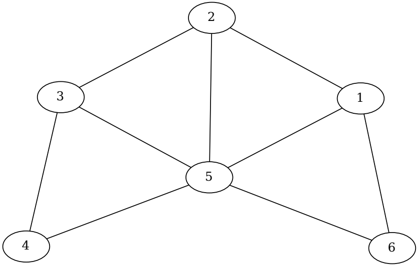
Examples:

Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

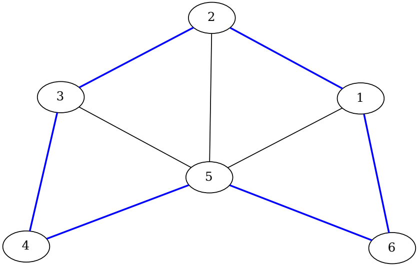


Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

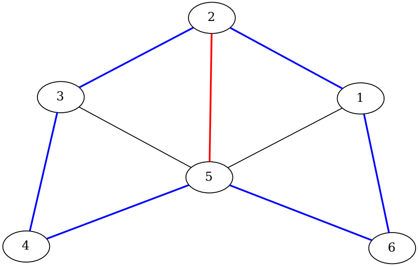


Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

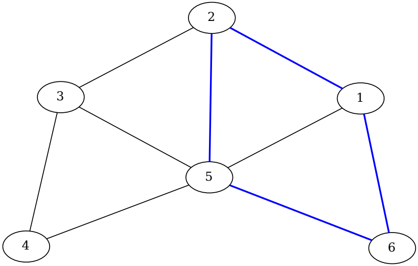


Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

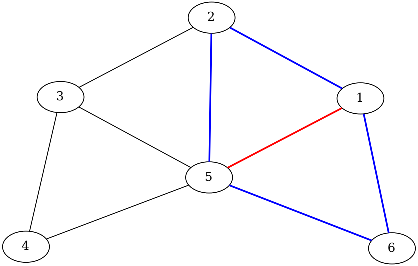


Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

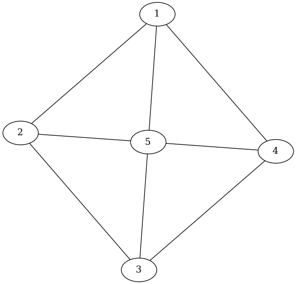


Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

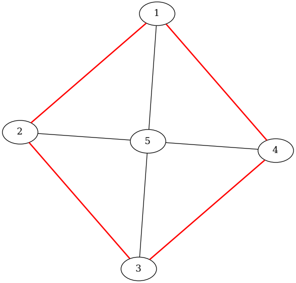


Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:



Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

Are the following chordal?

- Forests?
- Cliques?
- Bipartite graphs?
- Planar graphs?

Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

Chordal recognition is in:

- NP?
- coNP?
- P?

Chordal Graphs

Definition

A graph G is **chordal** if G does not contain any cycle C_k, for $k \geq 4$ as an **induced** subgraph.

Examples:

Chordal recognition is in:

- NP?
  Certificate: ??
- coNP?
  Counter-certificate: Long Induced Cycle
- P?

Chordal Graphs and Separators

Theorem

A graph G is chordal if and only if every minimal vertex separator of G induces a clique.

Chordal Graphs and Separators

Theorem

A graph G is chordal if and only if every minimal vertex separator of G induces a clique.

Sanity check:

- Trees are chordal.
- Every minimal vertex separator of a tree is a single vertex (K_1).

Chordal Graphs and Separators

Theorem

A graph G is chordal if and only if every minimal vertex separator of G induces a clique.

Need to prove that:

- G is chordal $\Rightarrow$ all minimal separators are cliques.
- G is not chordal $\Rightarrow$ some minimal separator is not a clique.

Which part is easy?

Chordal Graphs and Separators

Theorem

A graph G is chordal if and only if every minimal vertex separator of G induces a clique.

Proof.

(Easy part): G is not chordal $\Rightarrow$ some minimal separator is not a clique

- G has an induced cycle $v_1, v_2, \ldots, v_k$, $k \geq 4$
- Take a minimal $v_1 v_3$ separator S.
- $v_2 \in S$ and at least one $v_i \in S \cap \{v_4, \ldots, v_k\}$.
- $v_2 v_i \notin E$, therefore S is not a clique.

□

Chordal Graphs and Separators

Theorem

A graph G is chordal if and only if every minimal vertex separator of G induces a clique.

Proof.

(Harder part): G is not chordal $\Leftarrow$ some minimal separator is not a clique

- Let S be a minimal xy-separator that is not a clique
- Let $a, b \in S$ such that $ab \notin E$
- a, b have neighbors in both components of $G - S$ that contain x, y (because S is minimal).
- Take a shortest $a \to b$ path in each component, their union is an induced cycle (why?) of length at least 4, so G is not chordal.

□

Simplicial Vertices

Definition

A vertex v of a graph G is called **simplicial** if $G[N(v)]$ induces a clique.

Simplicial Vertices

Definition

A vertex v of a graph G is called **simplicial** if $G[N(v)]$ induces a clique.

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Simplicial Vertices

Definition

A vertex v of a graph G is called **simplicial** if $G[N(v)]$ induces a clique.

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Sanity check:

- If G is a tree
- and G is not a clique $\Leftrightarrow$ G is not K_2

Simplicial Vertices

Definition

A vertex v of a graph G is called **simplicial** if $G[N(v)]$ induces a clique.

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Sanity check:

- If G is a tree
- and G is not a clique $\Leftrightarrow$ G is not K_2
- G contains at least two non-adjacent leaves

Simplicial Vertices continued

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Proof.

Proof by induction on n. □

Simplicial Vertices continued

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Proof.

Proof by induction on n.

- Base case: $n = 3$, $G = P_3$, good.

□

Simplicial Vertices continued

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Proof.

Proof by induction on n.

- Let x, y be two non-adjacent vertices, S a minimal xy-separator
- S is a clique, X, Y are components of $G - S$ that contain x, y
- Claim: Each of X, Y contains a simplicial vertex of G, there are no edges from X to Y, so these are non-adjacent.

□

Simplicial Vertices continued

Theorem

If G is chordal and G is not a clique, then G contains at least two non-adjacent simplicial vertices.

Proof.

Proof by induction on n.

- Claim: X has a simplicial vertex of G
- Case 1: $G[X \cup S]$ is a clique
  - All vertices of X are simplicial, good.
- Case 2: $G[X \cup S]$ is not a clique
  - Inductive hypothesis applies on $G' = G[X \cup S]$
  - $\Rightarrow$ two non-adjacent simplicial vertices in G'
  - Both of them cannot be in S (which is a clique), so one is in X, good.

□

Recognizing Chordality

- "Is vertex v simplicial?" is in P.

Recognizing Chordality

- "Is vertex v simplicial?" is in P.
- "Does G have a simplicial vertex?" is in P.

Recognizing Chordality

- "Is vertex v simplicial?" is in P.
- "Does G have a simplicial vertex?" is in P.

Theorem

A chordal graph G contains at least one simplicial vertex.

Recognizing Chordality

- "Is vertex v simplicial?" is in P.
- "Does G have a simplicial vertex?" is in P.

Theorem

A chordal graph G contains at least one simplicial vertex.

- Alternative coNP counter-certificate: check that G has no simplicial vertex.

Recognizing Chordality

- "Is vertex v simplicial?" is in P.
- "Does G have a simplicial vertex?" is in P.

Theorem

A chordal graph G contains at least one simplicial vertex.

- Alternative coNP counter-certificate: check that G has no simplicial vertex.
- Can we use simplicial vertices to show that chordality recognition is in NP?

Recognizing Chordality

- "Is vertex v simplicial?" is in P.
- "Does G have a simplicial vertex?" is in P.

Theorem

A chordal graph G contains at least one simplicial vertex.

- Alternative coNP counter-certificate: check that G has no simplicial vertex.
- Can we use simplicial vertices to show that chordality recognition is in NP?
- Key insight: simplicial vertices cannot be involved in long induced cycles.

Recognizing Chordality continued

Definition

A **Perfect Elimination Ordering** of the vertices of a graph $G = (V, E)$ is an ordering of $V = \{v_1, \dots, v_n\}$ such that for all i we have that v_i is simplicial in $G[\{v_i, v_{i+1}, \dots, v_n\}]$.

Theorem

G has a perfect elimination ordering if and only if G is chordal.

Recognizing Chordality continued

Definition

A **Perfect Elimination Ordering** of the vertices of a graph $G = (V, E)$ is an ordering of $V = \{v_1, \dots, v_n\}$ such that for all i we have that v_i is simplicial in $G[\{v_i, v_{i+1}, \dots, v_n\}]$.

Theorem

G has a perfect elimination ordering if and only if G is chordal.

Proof.

G is not chordal $\Rightarrow$ G has no perfect elimination ordering

- Suppose G contains cycle C_k with $k \geq 4$.
- Build an ordering, let v_i be the first vertex of C_k in the ordering.
- The two neighbors of v_i in the cycle are non-adjacent, come later
- $\Rightarrow$ v_i is not simplicial in the rest of the graph, contradiction.

Recognizing Chordality continued

Definition

A **Perfect Elimination Ordering** of the vertices of a graph $G = (V, E)$ is an ordering of $V = \{v_1, \ldots, v_n\}$ such that for all i we have that v_i is simplicial in $G[\{v_i, v_{i+1}, \ldots, v_n\}]$.

Theorem

G has a perfect elimination ordering if and only if G is chordal.

Proof.

G is chordal $\Rightarrow$ G has a perfect elimination ordering

- G has a simplicial vertex v, place it first.
- Inductively construct an ordering of $G - v$ (which is chordal).

□

Recognizing Chordality

Theorem

There is a polynomial-time algorithm that decides if a given graph G is chordal.

Proof.

Key ideas:

- Finding a simplicial vertex is in P.
- If no such vertex, say No.
- If v is simplicial, then G chordal $\Leftrightarrow$ $G - v$ chordal, recurse.

Recognizing Chordality

Theorem

There is a polynomial-time algorithm that decides if a given graph G is chordal.

Proof.

Key ideas:

- Finding a simplicial vertex is in P.
- If no such vertex, say No.
- If v is simplicial, then G chordal $\Leftrightarrow$ $G - v$ chordal, recurse.
- Recursion sequence gives a perfect elimination ordering.

□

Applications

Maximum Independent Set

Basic algorithm:

1. Pick a vertex v
2. Compute (recursively) $s_1 = \alpha(G - v)$
3. Compute (recursively) $s_2 = 1 + \alpha(G - N[v])$
4. Return $\max\{s_1, s_2\}$

Maximum Independent Set

Basic algorithm:

1. Pick a vertex v
2. Compute (recursively) $s_1 = \alpha(G - v)$
3. Compute (recursively) $s_2 = 1 + \alpha(G - N[v])$
4. Return $\max\{s_1, s_2\}$

- Basic algorithm is bad (exponential-time).
- What if we have a way to select a “good” vertex v?

Maximum Independent Set – Simplicial vertices

Theorem

If v is a simplicial vertex of G, then there exists a maximum independent set S of G with $v \in S$.

Maximum Independent Set – Simplicial vertices

Theorem

If v is a simplicial vertex of G, then there exists a maximum independent set S of G with $v \in S$.

Proof.

Exchange argument:

- If $v \notin S$ and $N(v) \cap S = \emptyset$, contradiction, as $S \cup \{v\}$ is a larger independent set.
- If $v \notin S$ and $N(v) \cap S \neq \emptyset$, then $|N(v) \cap S| = 1$, as $N(v)$ is a clique.
- Let $S \cap N(v) = \{u\}$. Then $(S \setminus \{u\}) \cup \{v\}$ is another maximum independent set.

□

Maximum Independent Set

Basic algorithm:

1. Pick a vertex v
2. Compute (recursively) $s_1 = \alpha(G - v)$
3. Compute (recursively) $s_2 = 1 + \alpha(G - N[v])$
4. Return $\max\{s_1, s_2\}$

Maximum Independent Set

Basic algorithm:

1. Pick a simplicial vertex v
2. ~~Compute (recursively)~~ $s_1 = \alpha(G - v)$
3. Compute (recursively) $s_2 = 1 + \alpha(G - N[v])$
4. ~~Return $\max\{s_1, s_2\}$~~ $\rightarrow$ Return s_2

Maximum Independent Set

Basic algorithm:

1. Pick a simplicial vertex v
2. ~~Compute (recursively)~~ $s_1 = \alpha(G - v)$
3. Compute (recursively) $s_2 = 1 + \alpha(G - N[v])$
4. ~~Return max$\{s_1, s_2\}$~~ $\rightarrow$ Return s_2

Correctness:

- Running time is polynomial (no branching)
- v is simplicial $\Rightarrow$ some optimal independent set contains it.

Maximum Clique

Basic algorithm:

1. Pick a vertex v
2. Compute (recursively) $s_1 = \omega(G - v)$
3. Compute (recursively) $s_2 = 1 + \omega(G[N(v)])$
4. Return $\max\{s_1, s_2\}$

- Basic algorithm is bad (exponential-time).
- What if we have a way to select a "good" vertex v?

Maximum Clique

Basic algorithm:

1. Pick a simplicial vertex v
2. Compute (recursively) $s_1 = \omega(G - v)$
3. ~~Compute (recursively) $s_2 = 1 + \omega(G[N(v)])$~~ $s_2 = 1 + |N(v)|$
4. Return $\max\{s_1, s_2\}$

Maximum Clique

Basic algorithm:

1. Pick a simplicial vertex v
2. Compute (recursively) $s_1 = \omega(G - v)$
3. ~~Compute (recursively) $s_2 = 1 + \omega(G[N(v)])$~~ $s_2 = 1 + |N(v)|$
4. Return $\max\{s_1, s_2\}$

Correctness:

- Running time is polynomial (no branching)
- v is simplicial $\Rightarrow$ if v is in our clique, all of $N(v)$ can be placed in our clique.

Coloring

Recall the First-Fit coloring algorithm:

- Order vertices $v_1, \dots, v_n$
- Assign each vertex lowest available color

Coloring

Recall the First-Fit coloring algorithm:

- Order vertices $v_1, \dots, v_n$
- Assign each vertex lowest available color

Idea: execute this with **the opposite of** a PEO.

Coloring

Recall the First-Fit coloring algorithm:

- Order vertices $v_1, \ldots, v_n$
- Assign each vertex lowest available color

Idea: execute this with **the opposite of** a PEO.

Correctness:

- Claim: If some vertex receives color k, it is part of a clique of size k

Coloring

Recall the First-Fit coloring algorithm:

- Order vertices $v_1, \ldots, v_n$
- Assign each vertex lowest available color

Idea: execute this with **the opposite of** a PEO.

Correctness:

- Claim: If some vertex receives color k, it is part of a clique of size k
- When we color v_i, its previously colored neighbors form a clique
- If we use color k, the clique must be using colors $\{1, \ldots, k-1\}$, so it has size $k-1$, so we have a clique of size k.
- Recall: $\chi(G) \geq \omega(G)$.