I618 Advanced Computer Science II (Part II)

12/21 11:00-12:30
1/ 7 15:10-16:40
○1/ 9 9:20-10:50
1/11 11:00-12:30
1/16 9:20-10:50

Ryuhei Uehara
uehara@jaist.ac.jp
http://www.jaist.ac.jp/~uehara

I will give you some report problems on January.

Algorithms on Interval/Chordal Graphs

- Some efficient algorithms on the graphs
  - based on the graph properties, especially;
    - [Thm 5] Every interval graph has a *compact representation* in $[1..|V|]$, and all maximal cliques appear on the representation.

      … we can solve many problems by "sweeping" from left to right.
    - [Thm 6] In a chordal graph, every *separator* is a clique.

      … which allows us to use "divide-and-conquer."
    - [Thm 9] A chordal graph is an intersection graph of subtrees of a tree.

      … which generalizes the results on interval graphs;
      - each node of the tree *can* correspond to a maximal clique,
      - the number of nodes *can* be at most $|V|$.

      … which allows us "dynamic programming" on the tree.

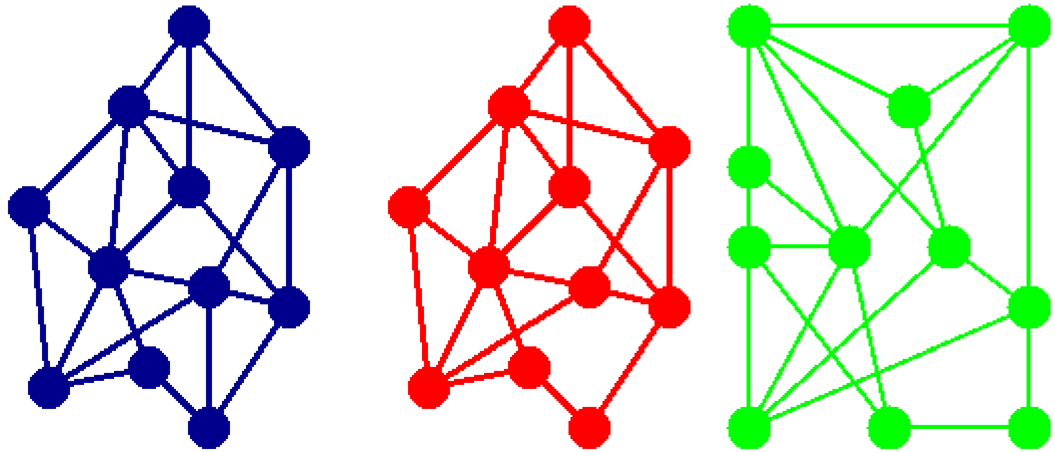
The tree is called "clique tree," but it requires more detailed analysis to make it "compact" since [Thm 9] does not construct a compact one.

Algorithms on Interval/Chordal Graphs

- Basic problems
  - graph isomorphism asks if two graphs are essentially "same".
    - its difficulty is *hereditary*; superclass is more difficult and subclass is easier.
    - graph isomorphism is *hard* for chordal graphs (and its superclasses)
    - graph isomorphism is *linear time solvable* for interval graphs (and its subclasses)
  - graph recognition determines if a given graph is in the class.
    - its difficulty is *not hereditary*; we need specified algorithm for each graph class
    - chordal graphs can be recognized in linear time
    - interval graphs can be recognized in linear time

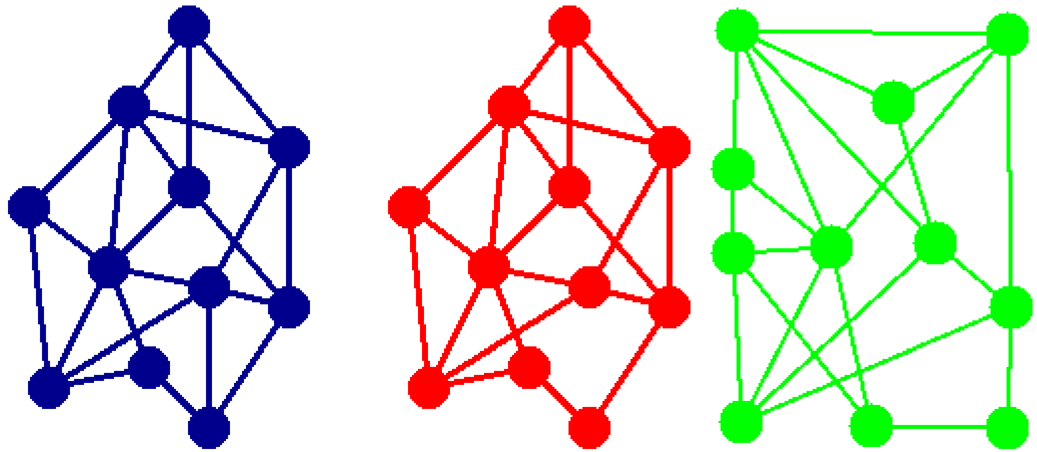
Graph Isomorphism

- The graph isomorphism problem
  - asks if there is a one-to-one mapping of vertex sets which keeps adjacency relationship.



Graph Isomorphism

- The graph isomorphism problem
  - asks if there is a one-to-one mapping of vertex sets which keeps adjacency relationship.



Graph Isomorphism

- The graph isomorphism problem
  - asks if there is a one-to-one mapping of vertex sets which keeps adjacency relationship.
- The GI problem is very natural basic problem.
  - It is in $\mathcal{NP}$, but it is not known if it is in $\mathcal{P}$ or $\mathcal{NP}$-complete
    - long standing open problem
    - *many* researchers feel it is easier than $\mathcal{NP}$-complete problems
    - one candidate between $\mathcal{P}$ and $\mathcal{NP}$-complete problems.
  - Hence we introduce 'GI-completeness';
    - the GI problem is *GI-complete* on a graph class $\mathcal{C}$ if the GI problem is still as hard as the usual one even on the class $\mathcal{C}$.

Graph Isomorphism

- The GI problem is very natural basic problem.
  - Hence we introduce 'GI-completeness';
    - the GI problem is *GI-complete* on a graph class $\mathcal{C}$ if the GI problem is still as hard as the usual one even on the class $\mathcal{C}$.

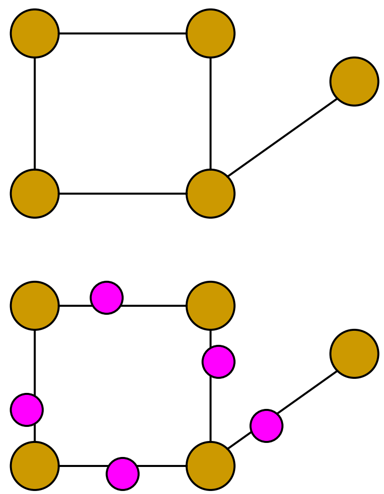
[Example 1] The GI problem for bipartite graphs is GI-complete.

(Proof) For any given graph $G=(V,E)$, we construct $G'=(V',E')$ as follows;

1. $V' := V \cup E$
2. $E' := \{\{e,v\} \mid v \in e \in E\}$

It is easy to see that

1. G' is bipartite for any G, and
2. G_1' and G_2' are isomorphic iff G_1 and G_2 are isomorphic. □



Graph Isomorphism

- The GI problem is very natural basic problem.
  - For our graph classes…
    - The GI problem on interval graphs is solvable in linear time.
    - Chordal graphs are GI-complete.

[Theorem 10] The GI problem for trees can be solved in linear time.

(Proof) Exercise! (Or report?) □

[Note] Theorem 10 is strongly related to the GI-problem for several graph classes including interval graphs.

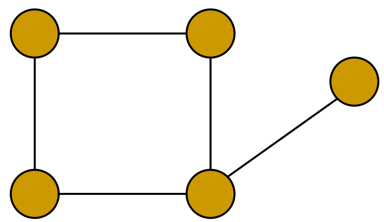
> [Today's First Goal]
> 1. The results for interval graphs are postponed after recognition.
> 2. GI-completeness of chordal graphs.

Graph Isomorphism

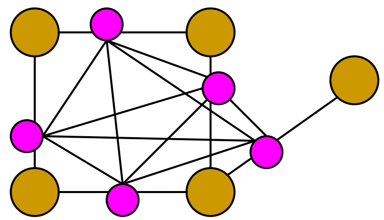
- The GI problem for chordal graphs

[Theorem 11] The GI problem for chordal graphs is GI-complete.

(Proof) It is sufficient to show that the GI problem for general graphs can be reduced to the GI problem for chordal graphs by a polynomial time reduction.



Let $G=(V,E)$ be a given (general) graph, and $G'=(V',E')$ be a graph constructed as follows;



1. $V' := V \cup E$
2. E' consists of
   1. $\{u,e\}$, $\{v,e\}$ if $e=\{u,v\}$ in E,
   2. $\{e_1,e_2\}$ for all e_1, e_2 in E.

Graph Isomorphism

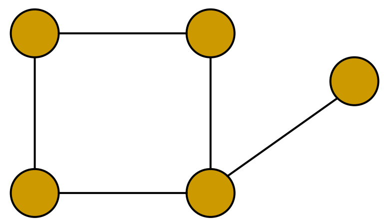
- The GI problem for chordal graphs

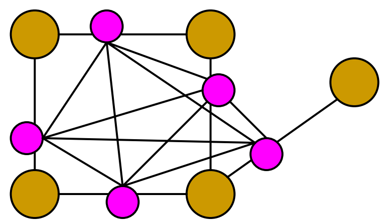
[Theorem 11] The GI problem for chordal graphs is GI-complete.

(Proof) It is sufficient to show that the GI problem for general graphs can be reduced to the GI problem for chordal graphs by a polynomial time reduction.

It is sufficient to show that

1. *G'* is a chordal graph, and
2. G_1 and G_2 are isomorphic iff so are G_1' and G_2'.





Graph Isomorphism

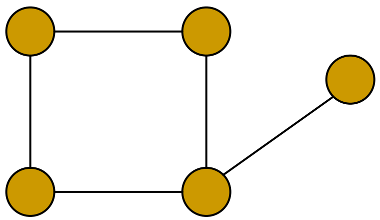
- The GI problem for chordal graphs

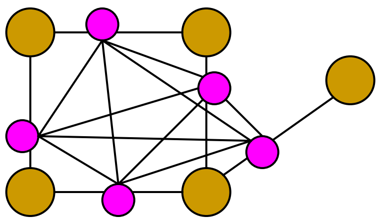
[Theorem 11] The GI problem for chordal graphs is GI-complete.

(Proof) It is sufficient to show that the GI problem for general graphs can be reduced to the GI problem for chordal graphs by a polynomial time reduction.

1. *G'* is a chordal graph;

   any cycle $C=(v_1,v_2,\ldots,v_k,v_1)$ of length at least 4 contains at least two vertices in E, which are joined by a chord.





Graph Isomorphism

- The GI problem for chordal graphs

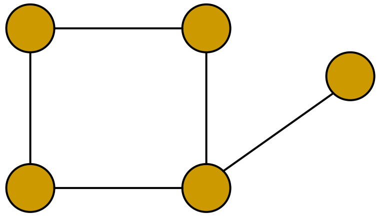
[Theorem 11] The GI problem for chordal graphs is GI-complete.

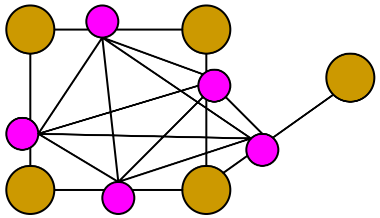
(Proof) 2. G_1 is isomorphic to G_2 iff so are G_1' and G_2'

⇔ G can be reconstructed from G' up to isomorphism.

For given $G'=(V',E')$,

1. E can be determined by the set of vertices
   1. E induces a clique
   2. each of them have two neighbors in V'-E.
2. V is determined by V'-E, and G can be reconstructed uniquely. □





Graph Isomorphism

- The GI problem for chordal graphs

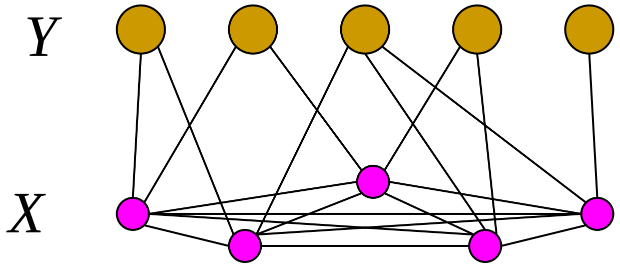
[Theorem 11] The GI problem for chordal graphs is GI-complete.

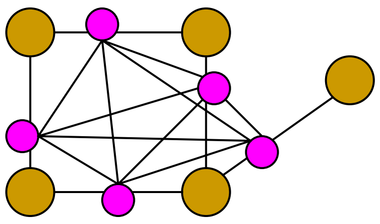
[Note] The chordal graph in the proof of [Theorem 11] is a special chordal graph $G=(V,E)$;

- V can be partitioned into two sets X and Y such that $G[X]$ induces a clique and $G[Y]$ induces an independent set.

Such graphs are called “*split graphs*.”

[Corollary 2] The GI problem for split graphs is GI-complete.





Algorithms on Interval/Chordal Graphs

- The graph recognition problem
  - difficulty is *not hereditary*; we need specified algorithm for each graph class
    - chordal graphs can be recognized in linear time
    - interval graphs can be recognized in linear time
- Very rough history…
  - chordal graph
    - Lexicographically breadth first search
    - Maximum cardinality search
  - interval graph
    - based on canonical tree representations
    - based on LexBFS
    - based on modular decompositions

Recognition of Interval/Chordal Graphs

- Rough history of the graph recognition…
  - chordal graph… linear time recognition by
    - Lexicographically breadth first search (LexBFS)
      - [Rose, Tarjan, Lueker 1976]
    - Maximum cardinality search
      - [Tarjan, Yannakakis 1984]

> Those two algorithms are so "good" that we have no chance to "improve"

  - interval graph… linear time recognition by
    - 1970s-80s; based on canonical tree representations
      - [Booth, Lueker 1976], [Lueker, Booth 1979], [Korte, Möhring 1989]
    - 1990s; based on LexBFS
      - several papers…,[Corneil, Olariu, Stewart 1998]
    - 2000-?; based on modular decompositions
      - some papers…, [McConnell, de Montgolfier 2005]

> Those algorithms have more detailed history, which will be explained later (not today)…

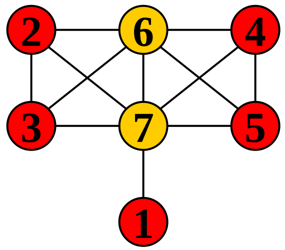
Recognition of a Chordal Graph

[Today's Next Goal]
3. Brief introduction of MCS (and LexBFS).

- Rough history of the graph recognition of chordal graphs
  - Lexicographically breadth first search (LexBFS)
    - by [Rose, Tarjan, Lueker 1976]
    - LexBFS is used to recognize several graph classes including
      - chordal graphs, interval graphs, cographs, Ptolemaic graphs, unit interval graphs, …
    - A survey for (only?) LexBFSs can be found in [Corneil 2004], which is an invited talk at WG 2004.
  - Maximum cardinality search (MCS)
    - by [Tarjan, Yannakakis 1984]
    - A relatively *few* related results are known about MCS.
  - LexBFS and MCS are simple for implementation, have good property, and hence they are well investigated.

Recognition of a Chordal Graph

- LexBFS and MCS are a kind of "search" algorithms.
  - Both algorithms find *reverse* of a PEO as follows;
  1. put any vertex as v_n;
  2. for each i=n-1, n-2, …, 1
     1. find the next vertex and put it as v_i



[Observation 1] *Any* vertex can be the last vertex of a PEO on a chordal graph.

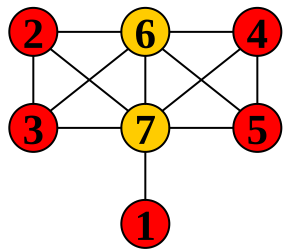
(Proof) Exercise!! (Hint: consider the tree model.)

[Point] How can we find the next vertex?

Recognition of a Chordal Graph

- LexBFS and MCS are a kind of "search" algorithms.
  - Both algorithms find *reverse* of a PEO as follows;

1. put any vertex as v_n;
2. for each i=n-1, n-2, …, 1
   1. find the next vertex and put it as v_i



[Point] How can we find the next vertex?

[MCS] the next vertex v_i is determined by

$$v_i := \max |N(v_i) \cap \{v_{i+1}, v_{i+2}, \ldots, v_n\}|,$$

which is the reason why we call it "maximum cardinality" search. (Ties are broken in any way.)