COMPUTING THE MINIMUM FILL-IN IS NP-COMPLETE

presentation by Jeffrey Hellrung

based on the paper by Mihalis Yannakakis, *Computing the Minimum Fill-In is NP-Complete*, SIAM J. Alg. Disc. Meth., Vol. 2, No. 1, March 1981

Outline

- Brief Overview of NP-Completeness (3 slides)
- Terminology and Preliminaries (3 slides)
- The Reduction (15 slides)

Brief Overview of NP-Completeness

- *P* ("deterministic Polynomial time") is the class of decision problems which can be solved in polynomial time (relative to the size of the "input string").
  - linear programming
  - computing the greatest common divisor (Euclidean algorithm)
  - determining if a number is prime (2002)

Brief Overview of NP-Completeness

- *NP* ("Non-deterministic Polynomial time") is the class of decision problems whose solutions can be "verified" in polynomial time.
- Given a "certificate", or "candidate proof", an algorithm exists which can verify, in polynomial time, whether the solution given by the certificate is correct.
  - *P* is a subset of *NP*
  - graph isomorphism problem
  - "Traveling Salesman" problem

Brief Overview of NP-Completeness

- A problem *C* is in *NP-Complete* if *C* is in *NP* and if every other problem in *NP* is polynomial-time reducible to *C*.
- To show *C* is *NP-Complete,* since polynomial-time reductions are closed under composition, need only show that *some* particular *NP-Complete* problem can be polynomial-time reduced to *C*.
- Will show that minimizing fill-in is *NP-Complete* via reduction from Optimal Linear Arrangement problem, which is known to be *NP-Complete*.

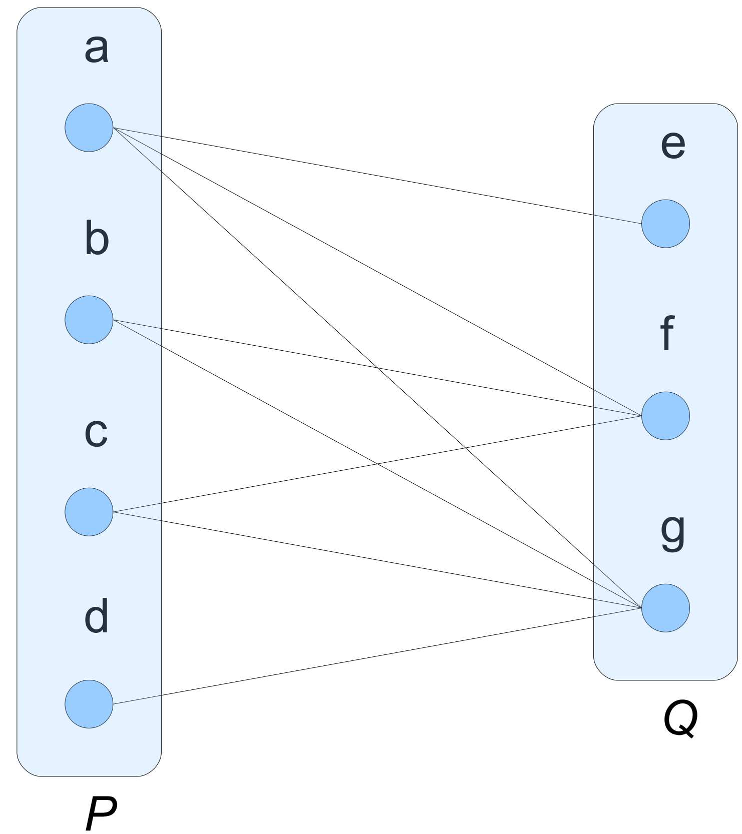
Terminology and Preliminaries

- Should know these terms:
  - *graph*, $G = (N,E)$; *nodes*, N; *edges*, E
  - *adjacent*; *neighborhood* of v, $\Gamma(v)$; *degree* of v, $d(v) = |\Gamma(v)|$
  - *clique*; *independent*; *subgraph*
  - *bipartite* graph, (P,Q,E)
- A bipartite graph is a *chain graph* if the neighborhoods of P form a chain; i.e., there exists an ordering π of P such that

$$\Gamma(\pi(1)) \supset \Gamma(\pi(2)) \supset \cdots \supset \Gamma(\pi(|P|))$$

Terminology and Preliminaries

- Example of a chain graph.
- $\pi(1)$ = a, $\pi(2)$ = b, $\pi(3)$ = c, $\pi(4)$ = d
- Notice that the neighborhoods of *Q* likewise form a chain (ordered as (g, f, e)).



Terminology and Preliminaries

- Also should be familiar with the following:
  - *chordal* graph
  - correspondence between symmetric matrices and their adjacency graphs
  - *eliminating* a node; *simplicial* nodes
  - *fill-in* $F(\pi)$ when eliminating nodes according to the ordering π
  - π is a *perfect elimination ordering* if $F(\pi)$ is empty.

The Reduction (Introduction)

- A graph has a perfect elimination ordering if and only if it is chordal.
- Thus, given an ordering π, the *augmented graph* $G_\pi = (N, E \cup F(\pi))$ is chordal, and π is a perfect elimination ordering of G_π.
- It follows that the minimum fill-in $|F(\pi)|$ over all orderings π is equal to the minimum number of edges whose addition to G gives a chordal graph.

The Reduction (Overview)

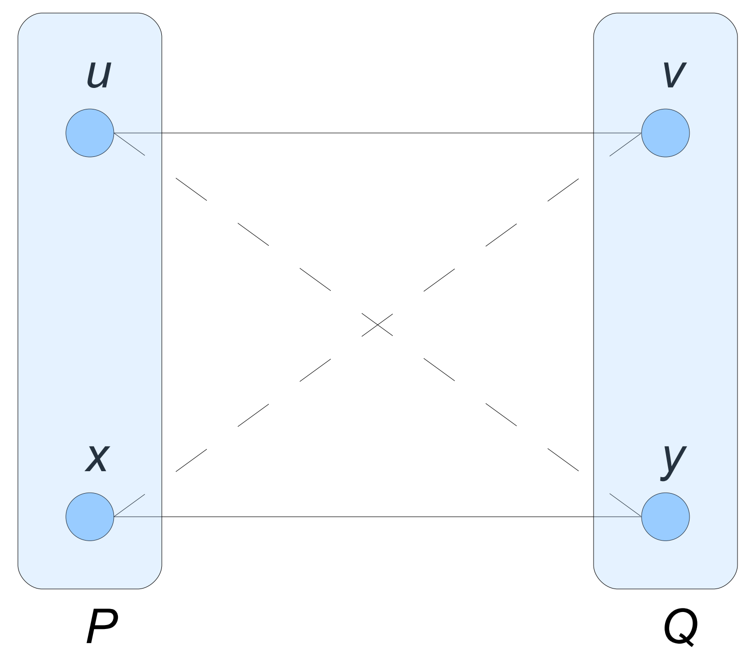
- The proof reduces the Optimal Linear Arrangement Problem to the problem of finding the minimum number of edges whose addition to a bipartite graph gives a chain graph.
- Further, the latter problem can be reduced to finding the minimum number of edges whose addition to a graph makes it chordal.
- Since OLAP is *NP-Complete*, it follows that determining the minimum fill-in is *NP-Complete*.

The Reduction (Lemma 1)

- Two edges (*u*,*v*), (*x*,*y*) are *independent* if the nodes *u*,*v*,*x*,*y* are distinct and the subgraph induced by these nodes consists of exactly the two edges.
- **Lemma 1: A bipartite graph is a chain graph if and only if it does not contain a pair of independent edges.**

The Reduction (Lemma 1)

- Easy to prove.
- Idea for “only if” direction:
  - If (u,v) and (x,y) are independent edges, then neither $\Gamma(u)$ contains $\Gamma(x)$ nor vice versa, hence G cannot be a chain graph.

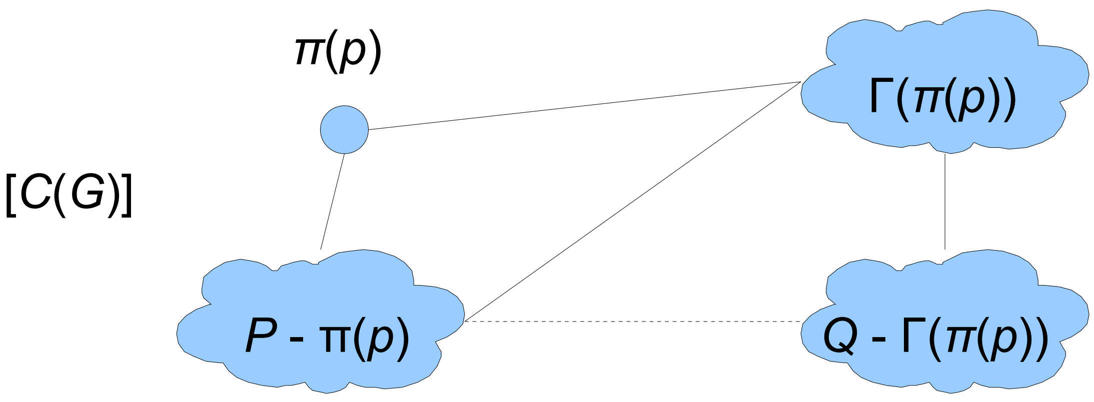


The Reduction (Lemma 2)

- If $G = (P,Q,E)$ is a bipartite graph, denote by $C(G)$ the same graph, but making P and Q cliques.
- **Lemma 2: $C(G)$ is chordal if and only if G is a chain graph.**
- “only if” is an easy corollary of Lemma 1.
- For “if”, let π be a “chain ordering” of P in G. Then show that $\pi(p = |P|)$ is a simplicial node in $C(G)$, hence, by induction, $C(G)$ has a perfect elimination ordering.

The Reduction (Lemma 2)

- Let
  - $\Gamma(\pi(p))$ = neighborhood of $\pi(p)$ in G
  - $\Gamma'(\pi(p))$ = neighborhood of $\pi(p)$ in $C(G)$
- Then $\Gamma'(\pi(p)) = (P - \pi(p)) \cup \Gamma(\pi(p))$.



The Reduction (Lemma 3)

- For a bipartite graph *G*, Lemma 2 implies that the minimum number of edges whose addition to *C*(*G*) makes it chordal is equal to the minimum number of edges (between *P* and *Q*) whose addition to *G* gives a chain graph.
- **Lemma 3: It is *NP-Complete* to find the minimum number of edges whose addition to *G* gives a chain graph.**
- Reduction is from Optimal Linear Arrangement Problem (OLAP).

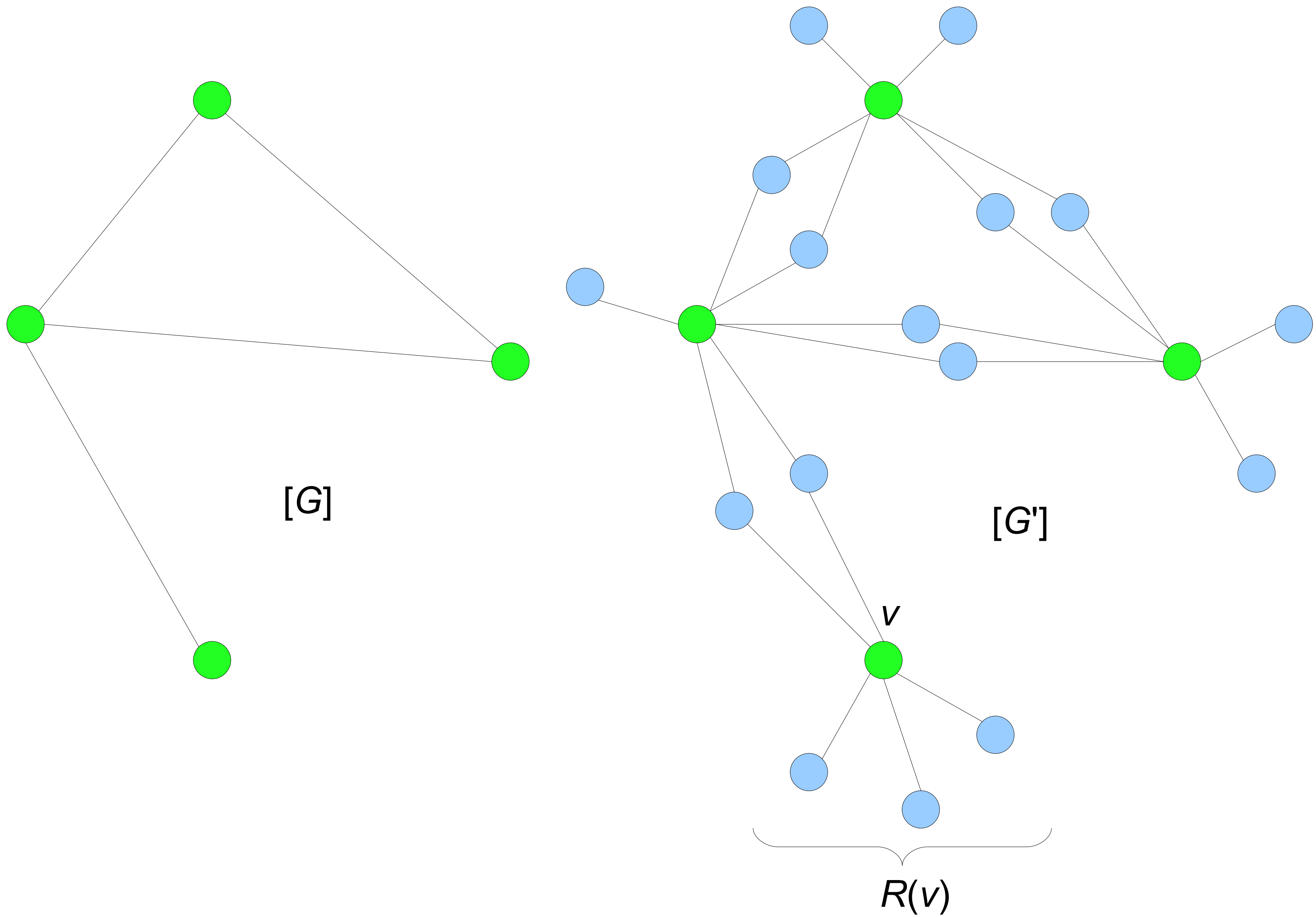
The Reduction (OLAP)

- A linear arrangement of a graph *G* = (*N*,*E*) is an ordering *π* of *N*.
- For an edge *e* = (*u*,*v*) of *G*, define
  - $\delta(e, \pi) = \left|\pi^{-1}(u) - \pi^{-1}(v)\right|$
- The cost *c*(*π*) of *π* is $$c(\pi) = \sum_{e \in E} \delta(e, \pi)$$
-
- The OLAP is to decide, given *G* and *k*, whether there exists a linear arrangement *π* such that *c*(*π*) ≤ *k*.

The Reduction (Lemma 3)

- First, given an instance (G = (N,E); k) of OLAP, construct the bipartite graph G' = (P,Q,E') as follows:
  - $P = N$
  - Q has two nodes for each edge e of G and $n - d(v)$ nodes (denoted $R(v)$) for each node v of G.

The Reduction (Lemma 3)



The Reduction (Lemma 3)

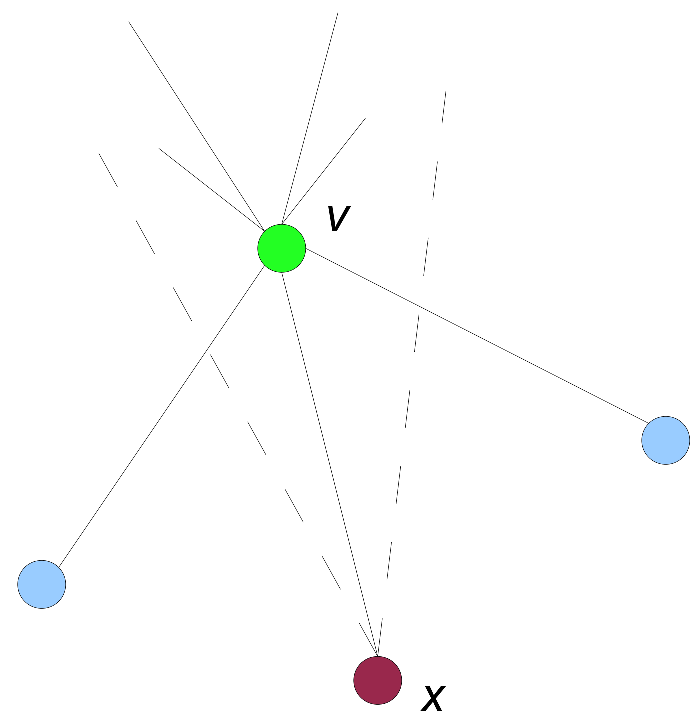
- If $l(G)$ is the minimum cost of a linear arrangement of G, and $h(G')$ is the minimum number of edges whose addition to G' gives a chain graph, then the claim is that
- $$h(G') = l(G) + \frac{n^2(n-1)}{2} - 2m$$
-
- Thus, $l(G) \le k$ if and only if $h(G') \le k$.

The Reduction (Lemma 3)

- If $h(\pi)$ denotes the minimum number of edges whose addition to G' makes it a chain graph with respect to the ordering π (of $P = N$), it suffices to show that, for every such ordering π,
- $$h(\pi) = c(\pi) + \frac{n^2(n-1)}{2} - 2m$$
-
- We thus consider an ordering π of P, and count the (minimum) additional edges required to make G' a chain graph.

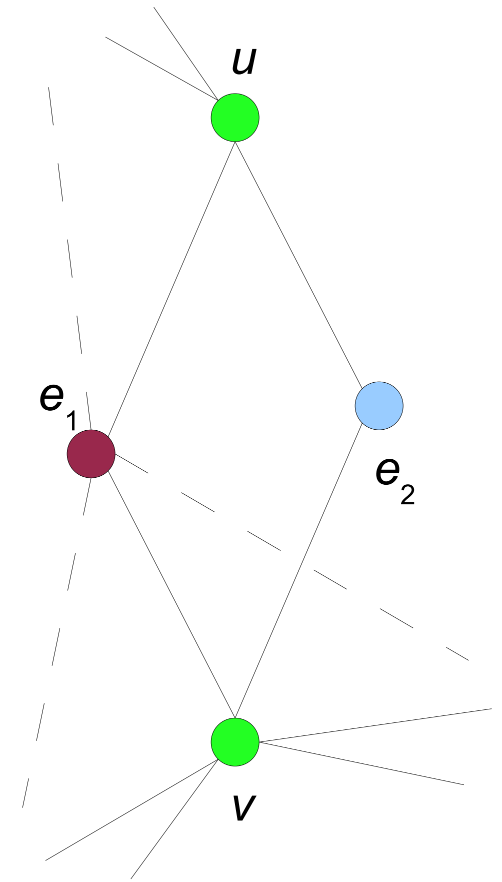
The Reduction (Lemma 3)

- For x in $R(v)$, v in N: x is only adjacent to v in G', but x must be adjacent to all nodes ordered before v in order to make G' a chain graph. Hence, x's "contribution" to $h(\pi)$ is $\pi^{-1}(v) - 1$.



The Reduction (Lemma 3)

- Let $e = (u,v)$ be an edge of G, and consider the the corresponding nodes e_1 and e_2 in G'. As in the previous case, e_i must be adjacent to all nodes ordered before v or before u, giving e_i's “contribution” to $h(\pi)$ as $\max\{\pi^{-1}(u), \pi^{-1}(v)\} - 2$.



The Reduction (Lemma 3)

- To summarize:
  - Each *v* in *N*, *x* in *R*(*v*) contributes $\pi^{-1}(v) - 1$.
  - Each $e = (u,v)$ in *E* contributes $\pi^{-1}(u) + \pi^{-1}(v) + \delta(e,\pi) - 4$ (after some algebraic manipulations).
- Upon summing, the claim just “falls out”:
- $h(\pi) = \sum_{v \in N} \sum_{x \in R(v)} (\pi^{-1}(v) - 1) + \sum_{e=(u,v) \in E} (\pi^{-1}(u) + \pi^{-1}(v) + \delta(e,\pi) - 4)$
- $= c(\pi) + \frac{n^2(n-1)}{2} - 2\mathrm{m}$
-
- This completes the reduction.