CS21 Decidability and Tractability

Lecture 20

February 21, 2024

Outline

- NP-complete problems: independent set, vertex cover, clique…
- NP-complete problems: Hamilton path and cycle,Traveling Salesperson Problem
- NP-complete problems: Subset Sum
- NP-complete problems: NAE-3-SAT, max cut

Search vs. Decision

- Definition: given a graph G = (V, E), an independent set in G is a subset $V' \subseteq V$ such that for all $u,w \in V'$ $(u,w) \notin E$
- A problem:

  given G, find the largest independent set
- This is called a search problem
  - searching for *optimal* object of some type
  - comes up frequently

Search vs. Decision

- We want to talk about languages (or decision problems)
- Most search problems have a natural, related decision problem by adding a bound "k"; for example:
  - search problem: given G, find the largest independent set
  - decision problem: given (G, k), is there an independent set of size *at least* k

Ind. Set is NP-complete

Theorem: the following language is NP-complete:

IS = {(G, k) : G has an IS of size $\geq$ k}.

- Proof:
  - Part 1: IS $\in$ NP. Proof?
  - Part 2: IS is NP-hard.
    - reduce from 3-SAT

Ind. Set is NP-complete

- We are reducing from the language:

  3SAT = { φ : φ is a 3-CNF formula that has a satisfying assignment }

  to the language:

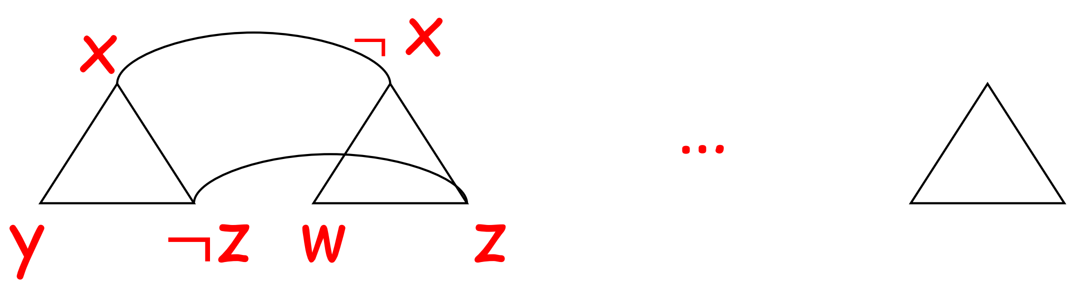
  IS = {(G, k) : G has an IS of size $\geq$ k}.

Ind. Set is NP-complete

The reduction f: given

$$\varphi = (x \lor y \lor \neg z) \land (\neg x \lor w \lor z) \land \ldots \land (\ldots)$$

we produce graph G_φ:

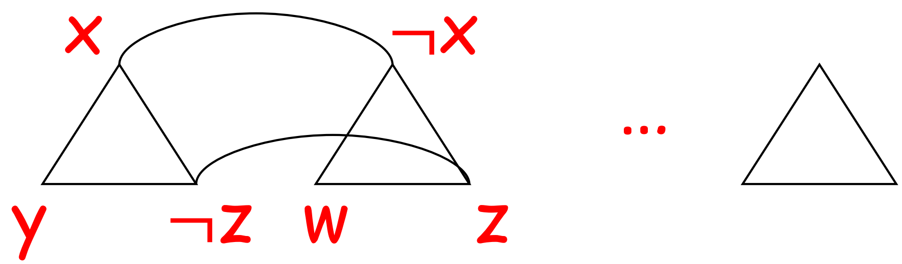


- one triangle for each of m clauses
- edge between every pair of contradictory literals
- set k = m

Ind. Set is NP-complete

$\varphi = (x \vee y \vee \neg z) \wedge (\neg x \vee w \vee z) \wedge \ldots \wedge (\ldots)$

$f(\varphi) =$
(G, # clauses)

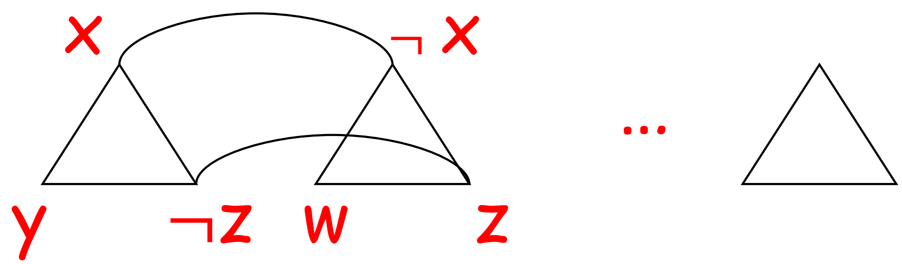


- Is f poly-time computable?
- YES maps to YES?
  - 1 true literal per clause in satisfying assign. A
  - choose corresponding vertices (1 per triangle)
  - IS, since no contradictory literals in A

Ind. Set is NP-complete

$\varphi = (x \vee y \vee \neg z) \wedge (\neg x \vee w \vee z) \wedge \ldots \wedge (\ldots)$

$f(\varphi) =$
(G, # clauses)



- NO maps to NO?
  - IS can have at most 1 vertex per triangle
  - IS of size ≥ # clauses must have exactly 1 per
  - since IS, no contradictory vertices
  - can produce satisfying assignment by setting these literals to true

Vertex cover

- Definition: given a graph G = (V, E), a vertex cover in G is a subset V' ⊆ V such that for all (u,w) ∈ E, u ∈ V' or w ∈ V'
- A search problem:

  given G, find the smallest vertex cover

- corresponding language (decision problem):

  VC = {(G, k) : G has a VC of size ≤ k}.

Vertex Cover is NP-complete

<u>Theorem</u>: the following language is NP-complete:

VC = {(G, k) : G has a VC of size $\leq$ k}.

- Proof:
  - Part 1: VC $\in$ NP. Proof?
  - Part 2: VC is NP-hard.
    - reduce from?

Vertex Cover is NP-complete

- We are reducing from the language:

$$IS = \{(G, k) : G \text{ has an IS of size} \geq k\}$$

to the language:

$$VC = \{(G, k) : G \text{ has a VC of size} \leq k\}.$$

Vertex Cover is NP-complete

- How are IS, VC related?

- Given a graph G = (V, E) with n nodes
  - if V' ⊆ V is an independent set of size k
  - then V-V' is a vertex cover of size n - k
- Proof:
  - suppose not. Then there is some edge with neither endpoint in V-V'. But then both endpoints are in V'. contradiction.

Vertex Cover is NP-complete

- How are IS, VC related?

- Given a graph G = (V, E) with n nodes
  - if V' $\subseteq$ V is a vertex cover of size k
  - then V-V' is an independent set of size n - k
- Proof:
  - suppose not. Then there is some edge with both endpoints in V-V'. But then neither endpoint is in V'. contradiction.

Vertex Cover is NP-complete

The reduction:

- given an instance of IS: (G, k) f produces the pair (G, n-k)

- f poly-time computable?
- YES maps to YES?
  - IS of size $\geq$ k in G $\Rightarrow$ VC of size $\leq$ n-k in G
- NO maps to NO?
  - VC of size $\leq$ n-k in G $\Rightarrow$ IS of size $\geq$ k in G

Clique

- Definition: given a graph $G = (V, E)$, a clique in G is a subset $V' \subseteq V$ such that for all $u,v \in V'$, $(u, v) \in E$
- A search problem:

  given G, find the largest clique

- corresponding language (decision problem):

  CLIQUE = $\{(G, k) : G$ has a clique of size $\geq k\}$.

Clique is NP-complete

Theorem: the following language is NP-complete:

CLIQUE = {(G, k) : G has a clique of size $\geq$ k}

- Proof:
  - Part 1: CLIQUE $\in$ NP. Proof?
  - Part 2: CLIQUE is NP-hard.
    - reduce from?

Clique is NP-complete

- We are reducing from the language:

IS = {(G, k) : G has an IS of size $\geq$ k}

to the language:

CLIQUE = {(G, k) : G has a CLIQUE of size $\geq$ k}.

Clique is NP-complete

- How are IS, CLIQUE related?
- Given a graph G = (V, E), define its complement G' = (V, E' = {(u,v) : (u,v) $\notin$ E})
  - if V' $\subseteq$ V is an independent set in G of size k
  - then V' is a clique in G' of size k
- Proof:
  - *Every* pair of vertices u,v $\in$ V' has no edge between them in G. Therefore they have an edge between them in G'.

Clique is NP-complete

- How are IS, CLIQUE related?
- Given a graph $G = (V, E)$, define its complement $G' = (V, E' = \{(u,v) : (u,v) \notin E\})$
  - if $V' \subseteq V$ is a clique in G' of size k
  - then V' is an independent set in G of size k
- Proof:
  - *Every* pair of vertices $u,v \in V'$ has an edge between them in G'. Therefore they have no edge between them in G.

Clique is NP-complete

The reduction:

- given an instance of IS: (G, k) f produces the pair (G’, k)

- f poly-time computable?
- YES maps to YES?
  - IS of size $\geq$ k in G $\Rightarrow$ CLIQUE of size $\geq$ k in G’
- NO maps to NO?
  - CLIQUE of size $\geq$ k in G’ $\Rightarrow$ IS of size $\geq$ k in G

Hamilton Path

- Definition: given a directed graph G = (V, E), a Hamilton path in G is a directed path that touches every node exactly once.

- A language (decision problem):

  HAMPATH = {(G, s, t) : G has a Hamilton path from s to t}

HAMPATH is NP-complete

Theorem: the following language is NP-complete:

HAMPATH = {(G, s, t) : G has a Hamilton path from s to t}

- Proof:
  - Part 1: HAMPATH ∈ NP. Proof?
  - Part 2: HAMPATH is NP-hard.
    - reduce from?

HAMPATH is NP-complete

- We are reducing from the language:

3SAT = { φ : φ is a 3-CNF formula that has a satisfying assignment }

to the language:

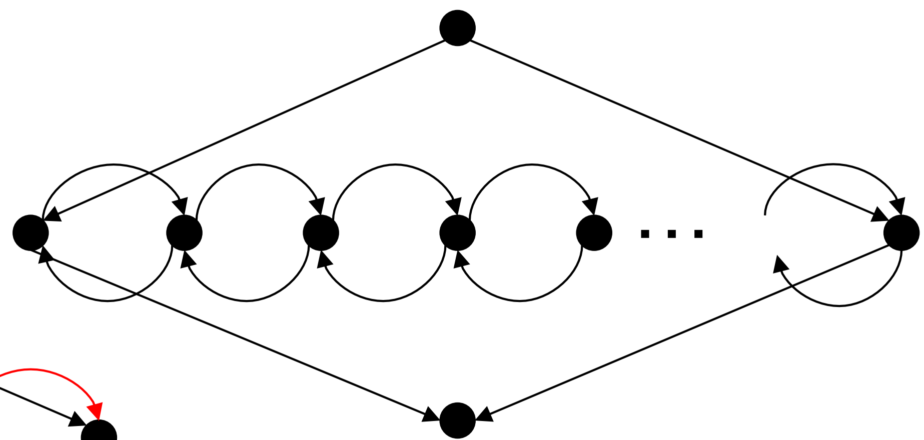
HAMPATH = {(G, s, t) : G has a Hamilton path from s to t}

HAMPATH is NP-complete

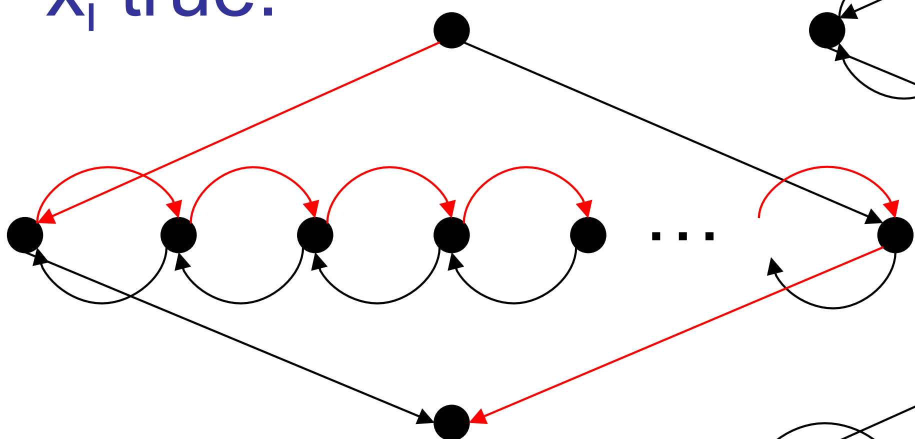
- We want to construct a graph from φ with the following properties:
  - a satisfying assignment to φ translates into a Hamilton Path from s to t
  - a Hamilton Path from s to t can be translated into a satisfying assignment for φ
- We will build the graph up from pieces called gadgets that “simulate” the clauses and variables of φ.

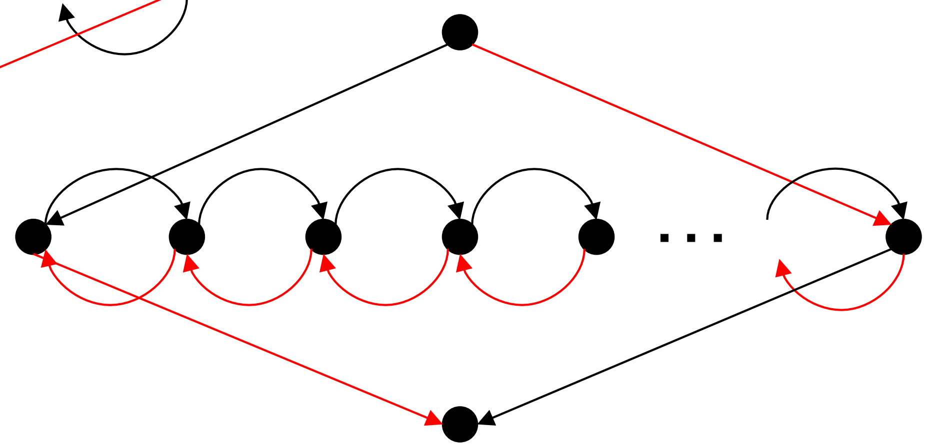
HAMPATH is NP-complete

- The variable gadget (one for each x_i):



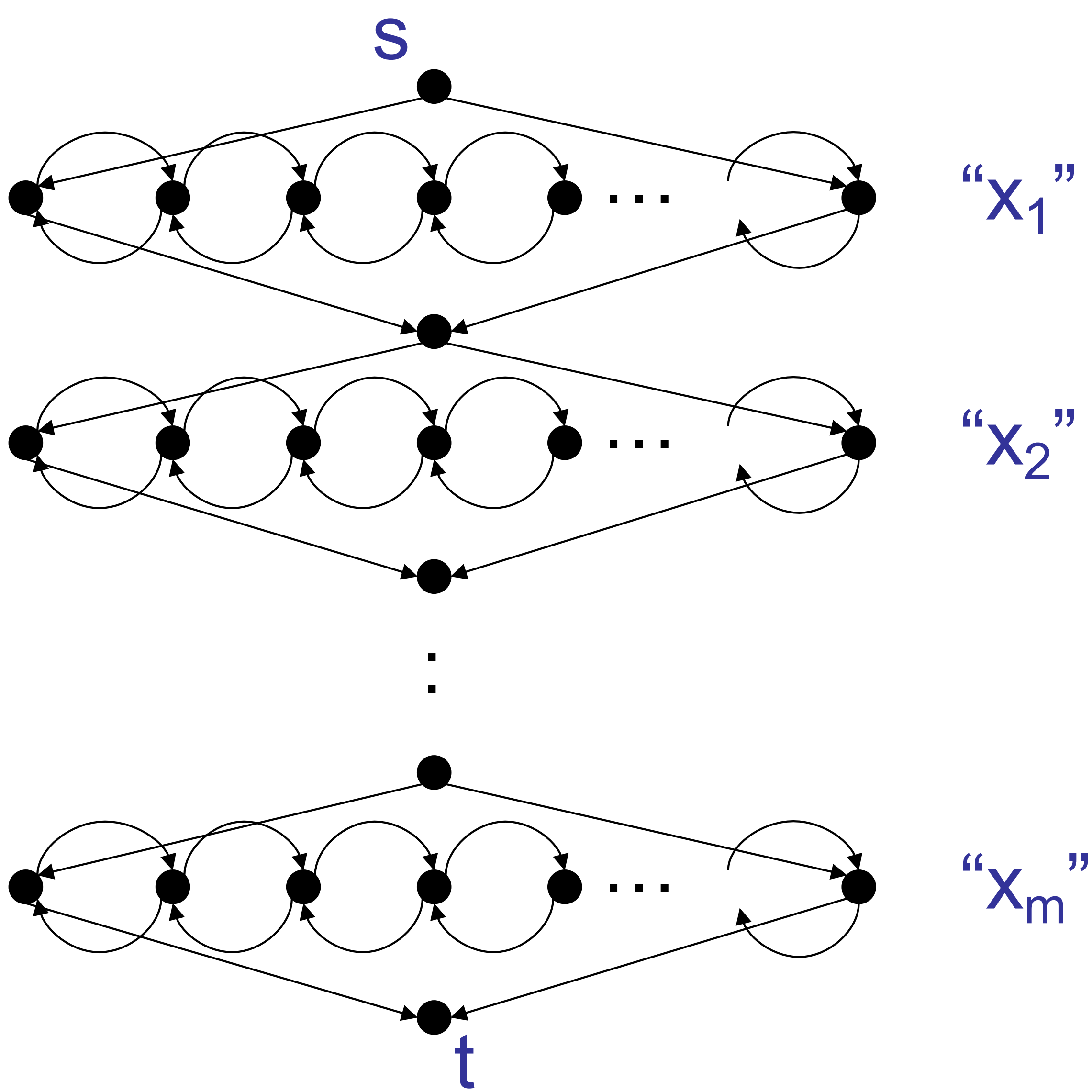
x_i true:





x_i false:

HAMPATH is NP-complete



- path from s to t translates into a truth assignment to $x_1 \ldots x_m$
- why must the path be of this form?

HAMPATH is NP-complete

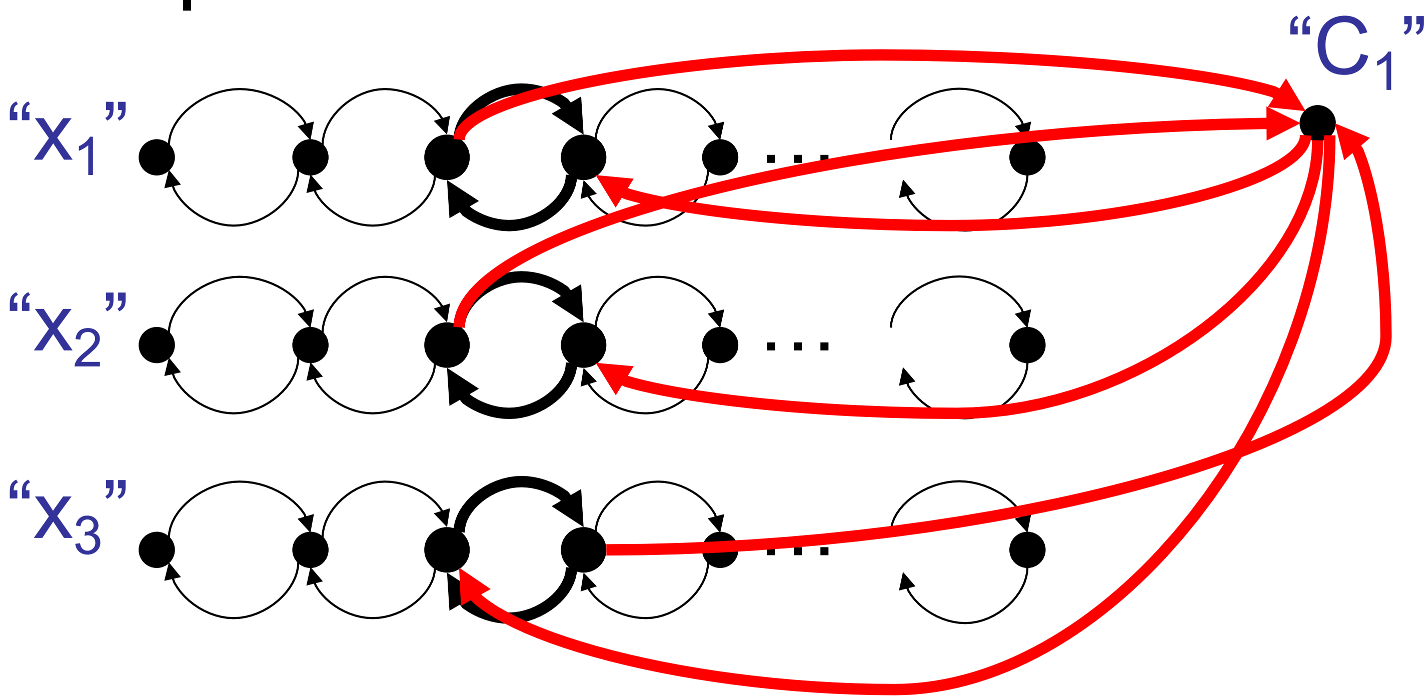
$\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \wedge \ldots \wedge (\ldots)$

- How to ensure that all k clauses are satisfied?
- need to add nodes
  - can be visited in path if the clause is satisfied
  - if visited in path, implies clause is satisfied by the assignment given by path through variable gadgets

HAMPATH is NP-complete

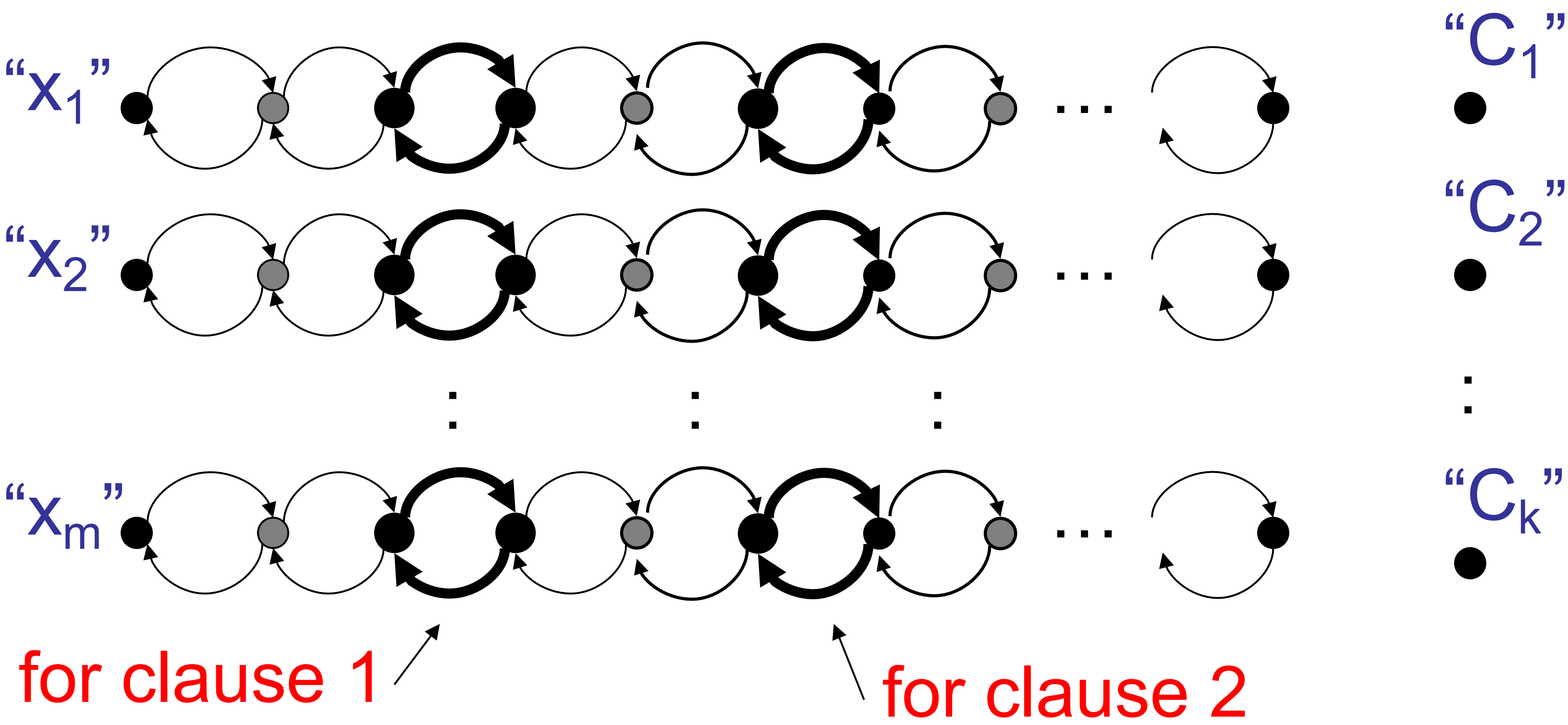
$\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \wedge \ldots \wedge (\ldots)$

- Clause gadget allows "detour" from "assignment path" for each true literal in clause



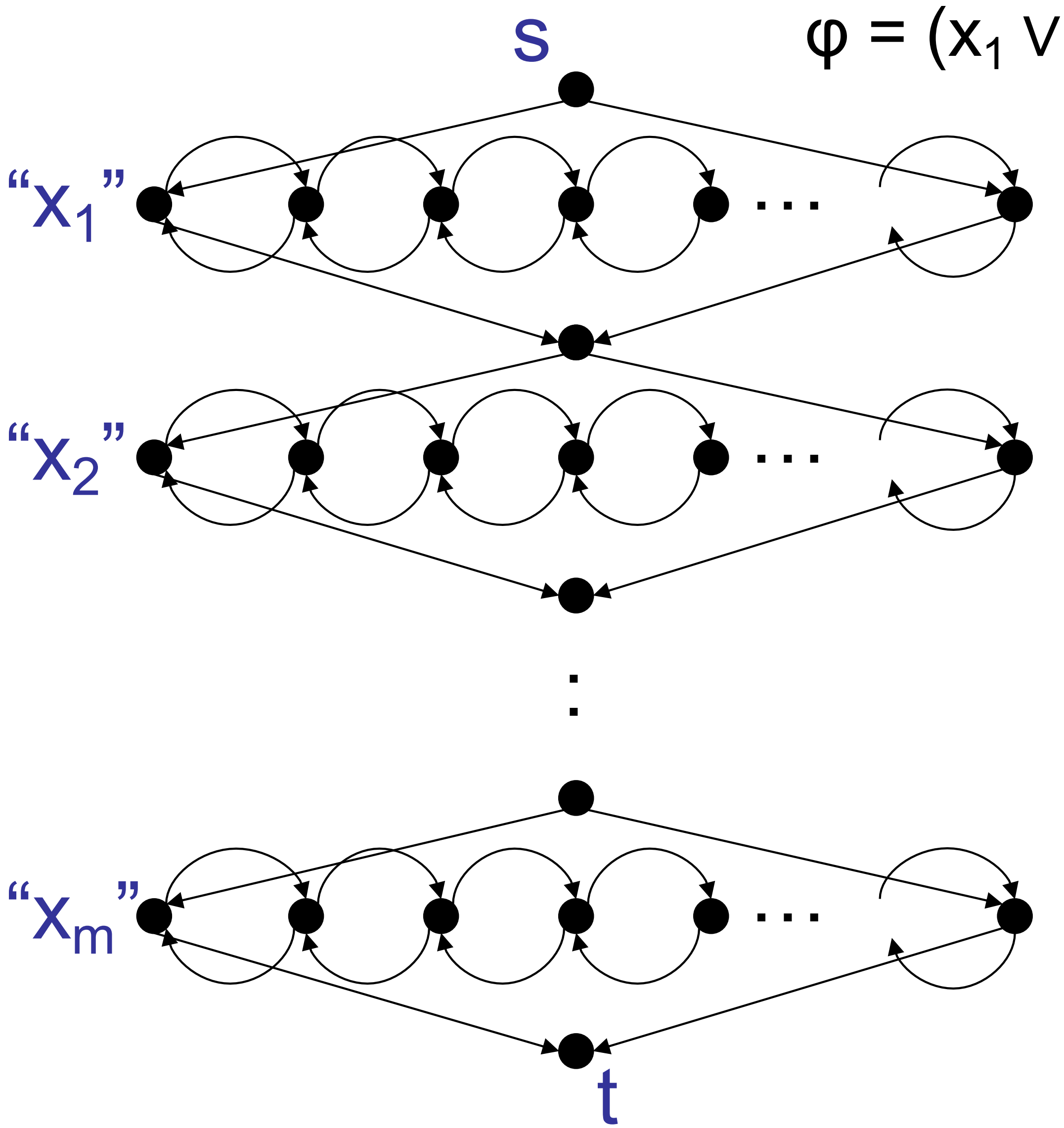
HAMPATH is NP-complete

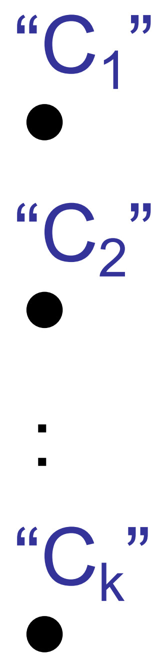
- One clause gadget for each of k clauses:



HAMPATH is NP-complete

$\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \ldots$

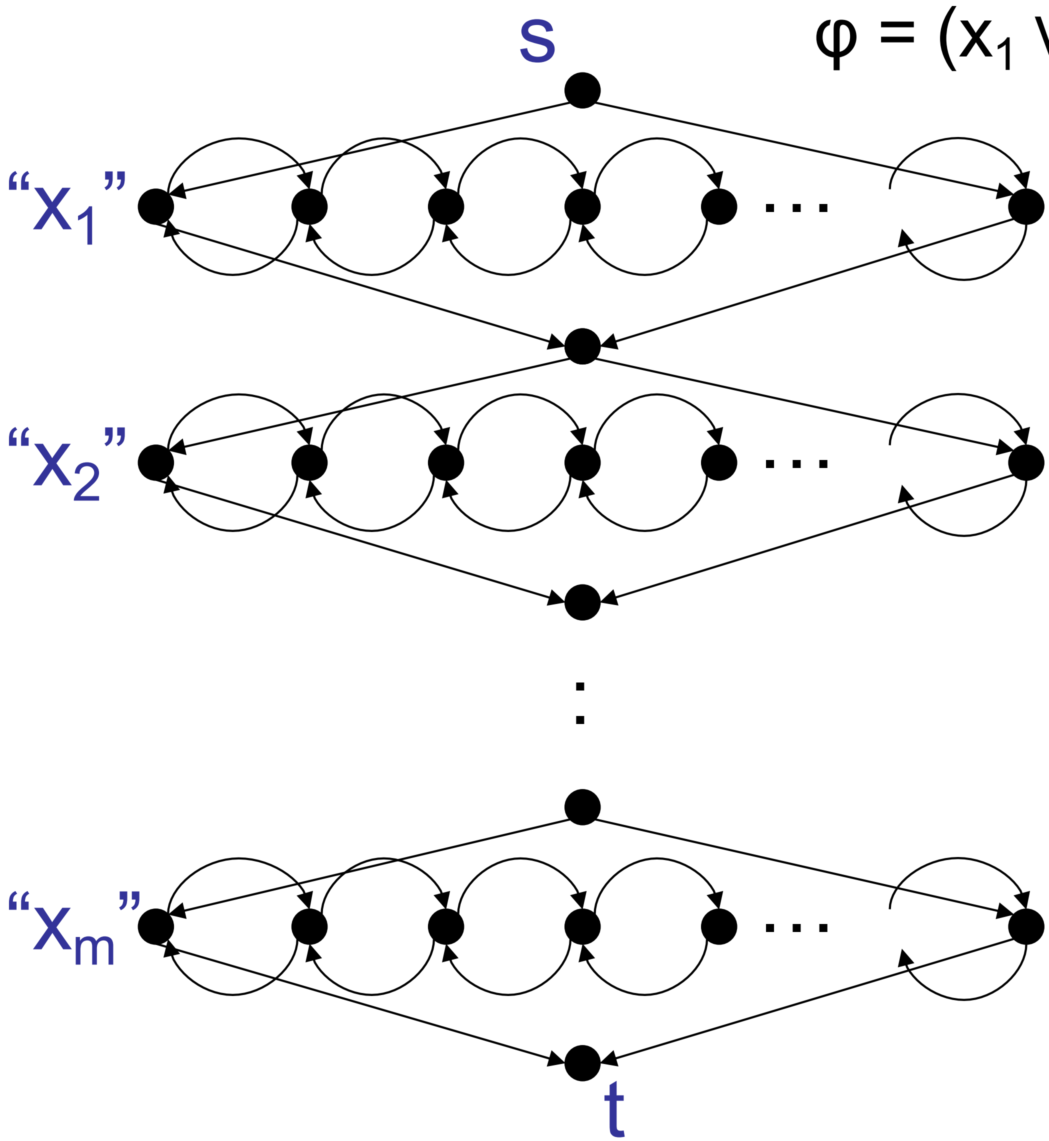


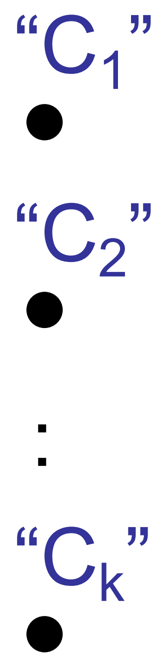


- f(φ) is this graph (edges to/from clause nodes not pictured)
- f poly-time computable?
  - # nodes = O(km)

HAMPATH is NP-complete

$\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \ldots$

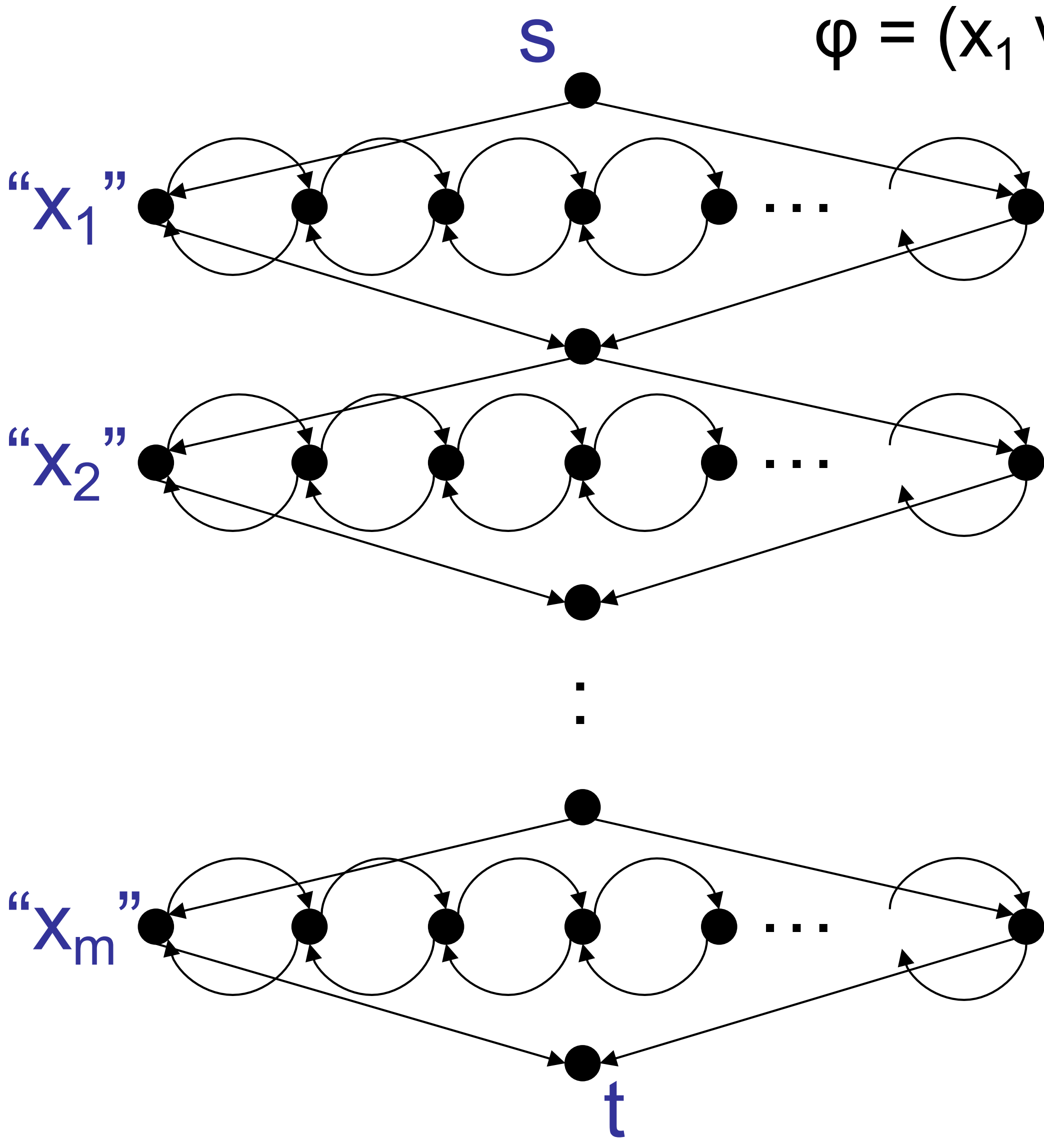


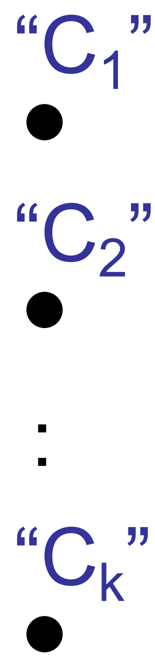


- YES maps to YES?
  - first form path from satisfying assign.
  - pick true literal in each clause and add detour

HAMPATH is NP-complete

$\varphi = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4 \vee x_3) \ldots$





- NO maps to NO?
  - try to translate path into satisfying assignment
  - if path has “intended” form, OK.