Analysis and Design of Algorithms

NUS
National University of Singapore

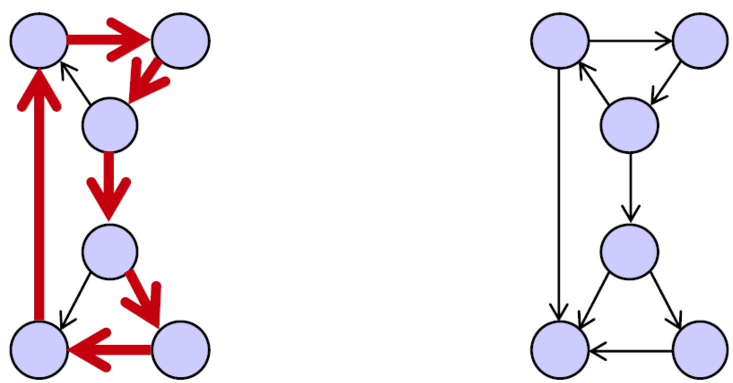
Algorithms
CS3230

Tutorial

Week 13

Directed Hamiltonian Cycle

DIR-HAM-CYCLE: Given a digraph $G = (V, E)$, does there exists a simple directed cycle Γ that contains every node in V exactly once?



Has Hamiltonian cycle No Hamiltonian cycle

Question 1

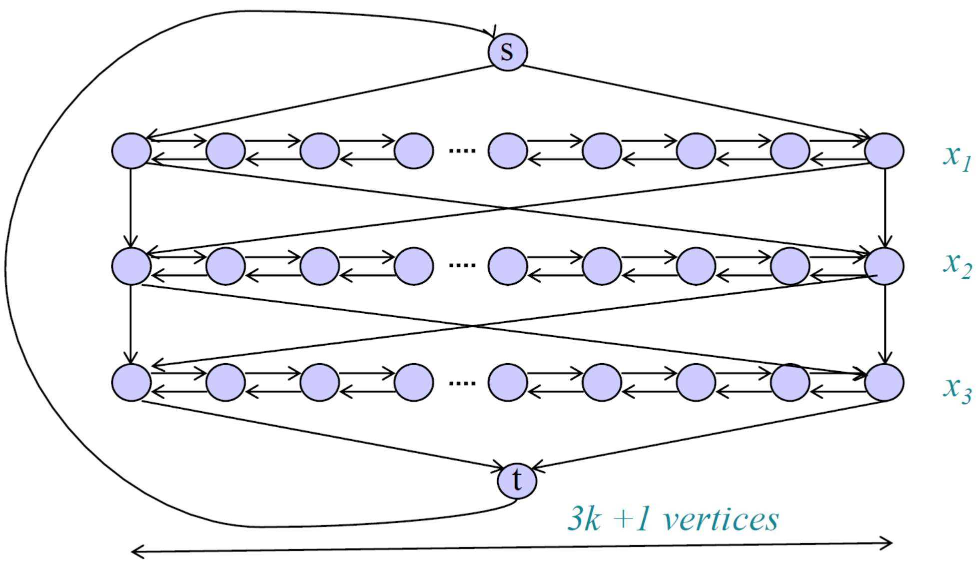
Give a poly time reduction from 3-SAT to DIR-HAM-CYCLE.

3-SAT $\leq_P$ DIR-HAM-CYCLE

Given an instance of Φ of 3-SAT, goal is to construct an instance G of DIR-HAM-CYCLE that has a Hamiltonian cycle **if and only if** Φ is satisfiable.

3-SAT $\leq_P$ DIR-HAM-CYCLE

3-SAT instance Φ with n variables and k clauses

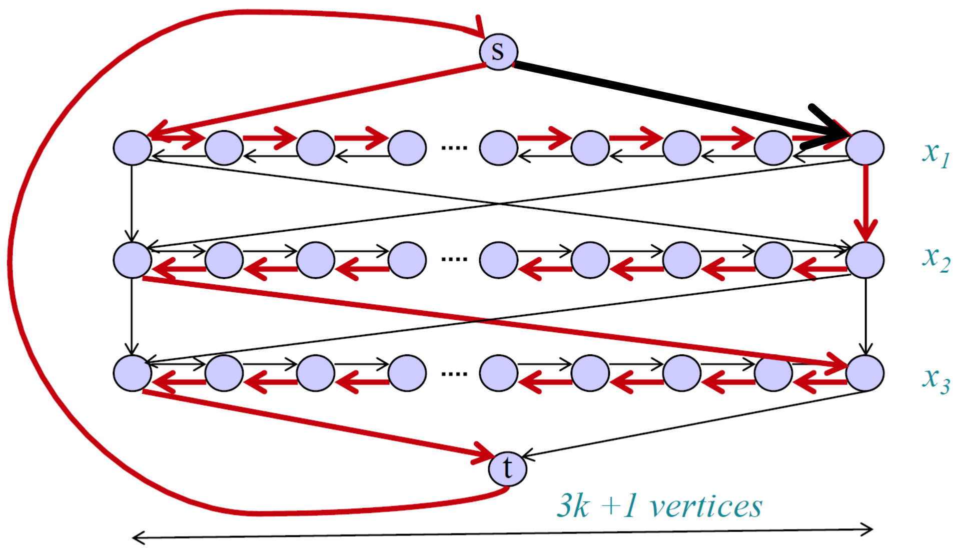


First, construct G with:

- Distinguished vertices s and t
- n rows of vertices corresponding to n variables
- $3k$ bidirectional edges between $3k + 1$ columns of vertices corresponding to k clauses.

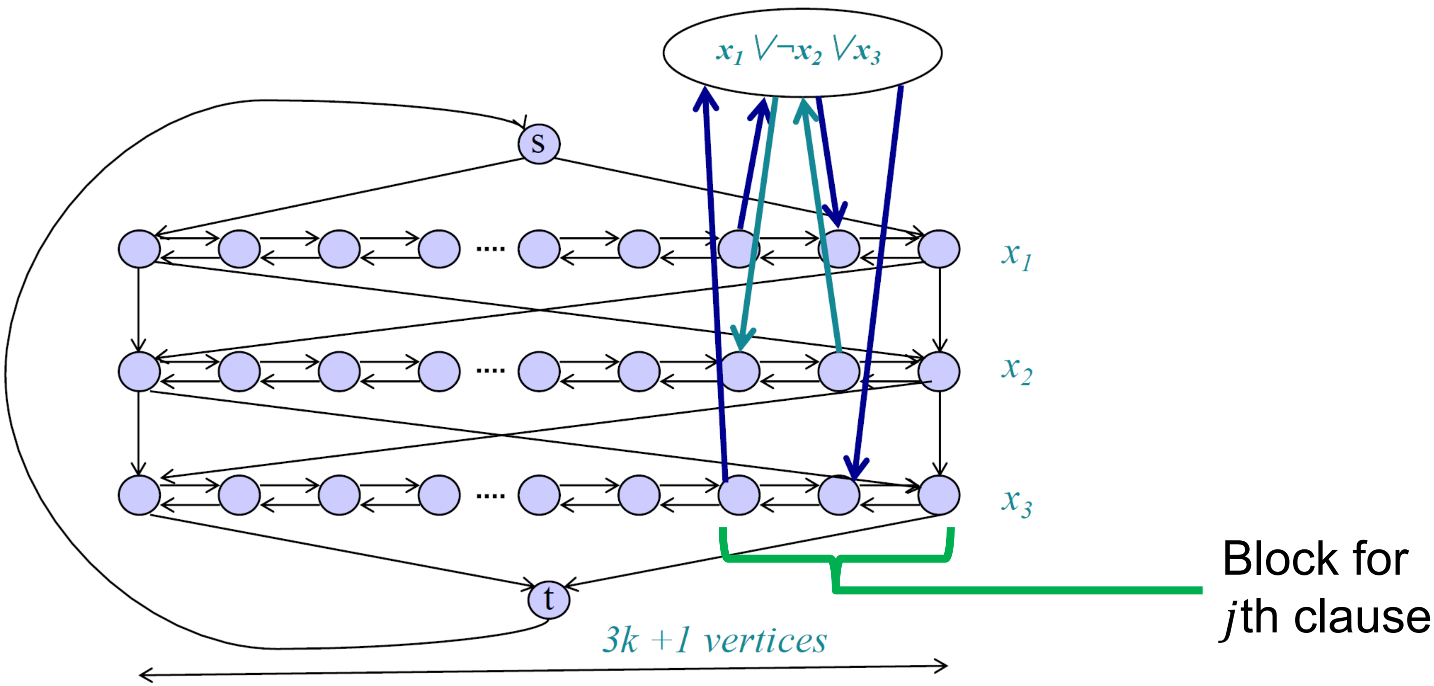
3-SAT $\leq_P$ DIR-HAM-CYCLE

There are 2^n **directed Hamiltonian cycles** corresponding to moving right or moving left on row i. Think of moving right as setting $x_i = 1$ and moving left as $x_i = 0$. Here is the cycle for the assignment:



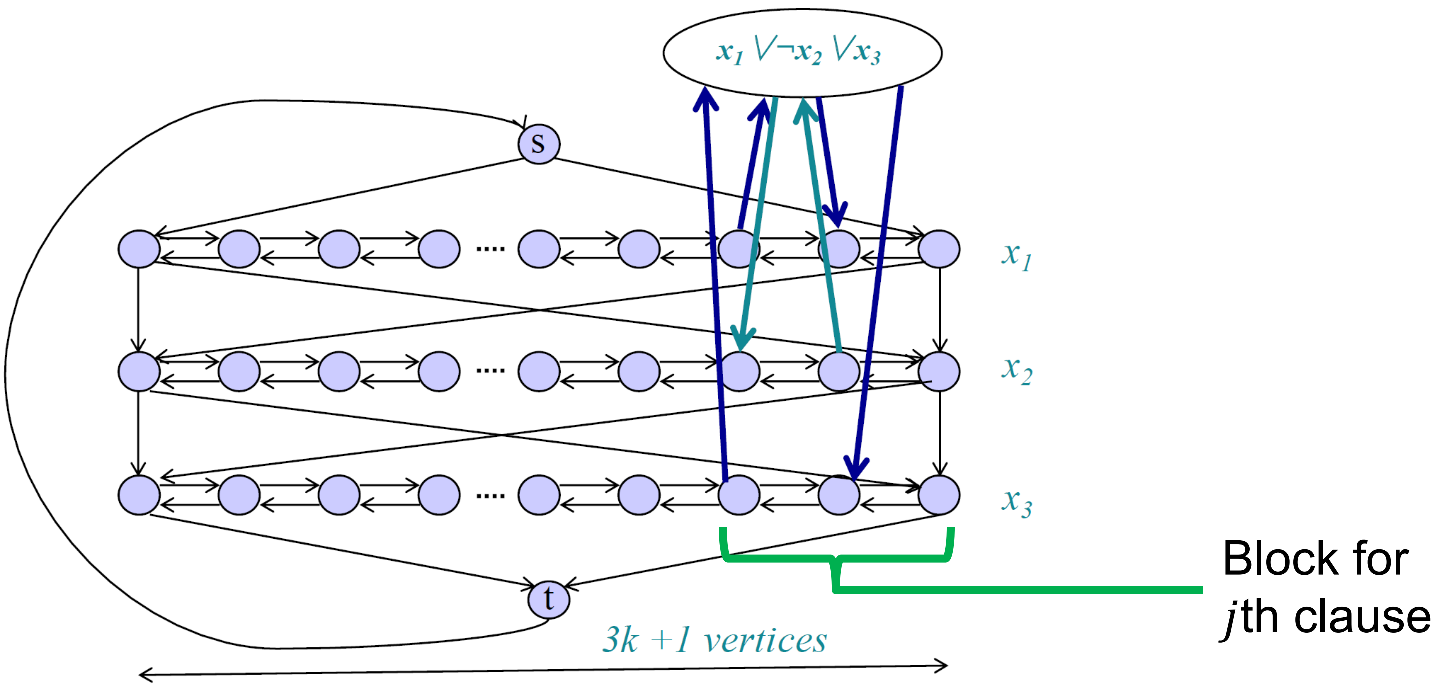
3-SAT $\leq_P$ DIR-HAM-CYCLE

Then, add a vertex for each clause (corresponding to its 3 columns) to enforce the clause constraints.



3-SAT $\leq_P$ DIR-HAM-CYCLE

- If clause contains x_i, edges set direction towards right
- If clause contains $\bar{x_i}$, edges set direction towards left



3-SAT $\leq_P$ DIR-HAM-CYCLE

Claim: If Φ is satisfiable, then G has a directed Ham cycle.

3-SAT $\leq_P$ DIR-HAM-CYCLE

Claim: If Φ is satisfiable, then G has a directed Ham cycle.

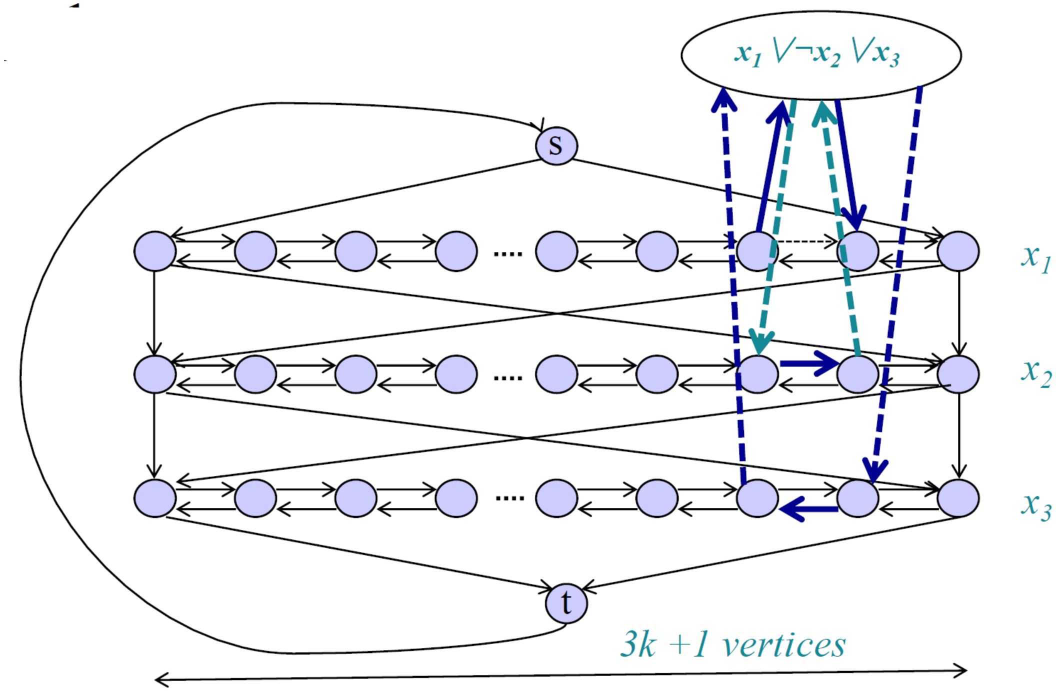
Proof: Suppose Φ has satisfying assignment x^*. Then, define a directed Ham cycle in G as follows:

- If $x_i^* = 1$, go from left to right on row i
- If $x_i^* = 0$, go from right to left on row i

As the assignment is satisfying, for each clause C_j, there will be at least one row i in which we are going in the right direction, so that the vertex for C_j can be spliced in.

3-SAT $\leq_P$ DIR-HAM-CYCLE

Example: $x_1 = 1, x_2 = 1, x_3 = 0$ splices in the clause vertex by taking the solid, instead of dotted, edges.



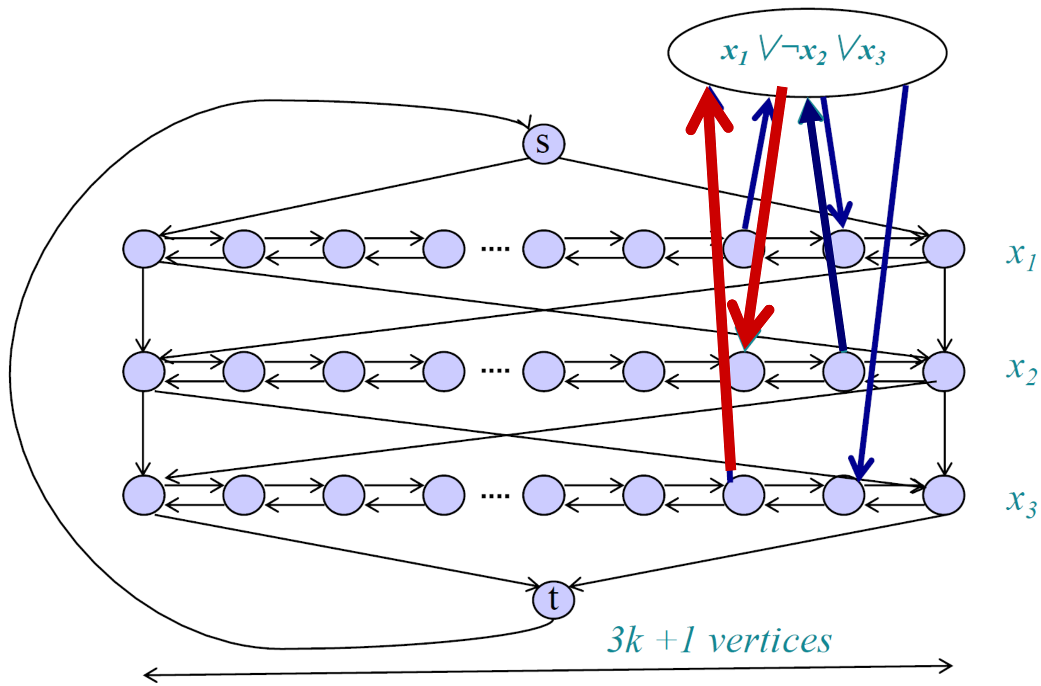
3-SAT $\leq_P$ DIR-HAM-CYCLE

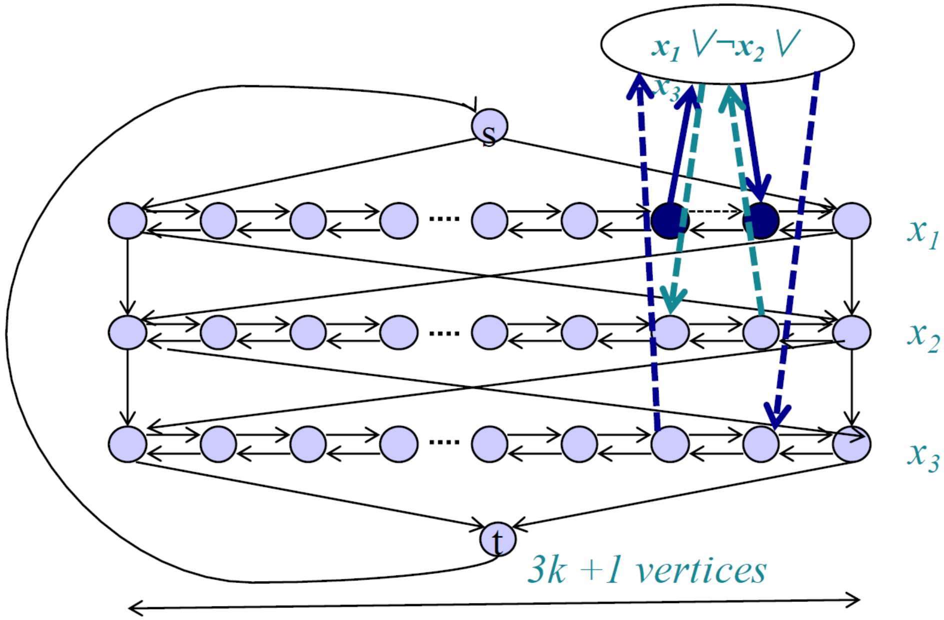
Claim: If G has a directed Ham cycle, then Φ is satisfiable.

3-SAT $\leq_P$ DIR-HAM-CYCLE

Claim: If G has a directed Ham cycle, then Φ is satisfiable.

Why not?





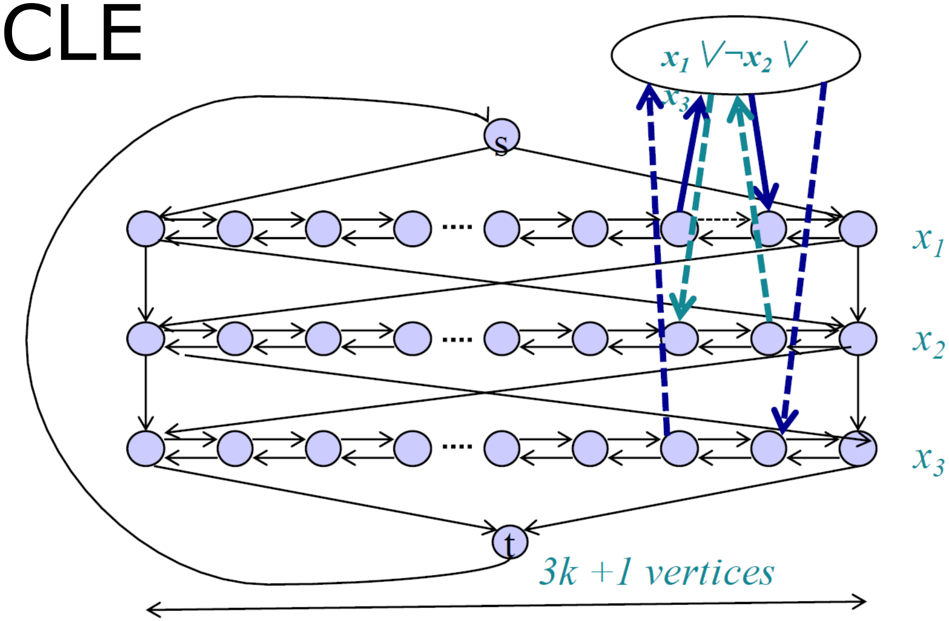
Assume that the cycle enters the clause at the highlighted (left) vertex. If it does not exit the clause through the highlighted vertex next to it, when it visits that vertex from the right it would be unable to proceed further.

3-SAT $\leq_P$ DIR-HAM-CYCLE

- So, if an edge enters a clause vertex through row i, it also exits through row i

- Given a Ham cycle, we can construct an assignment x:
  - If row i is traversed from left to right, set $x_i = 1$
  - If row i is traversed from right to left, set $x_i = 0$

3-SAT $\leq_P$ DIR-HAM-CYCLE



Consider a clause vertex C_j where the incoming edge in the Hamiltonian cycle comes from the left (respectively right) on row k.

- By construction, $x_k = 1$ (resp. $x_k = 0$), as assigned by A satisfies the clause.

Question 2

Given a simple undirected graph $G=(V,E)$, we would like to find the minimum vertex cover of the graph (i.e. find a subset of $V' \subseteq V$ such that for any $(u,v) \in E$, either $u \in V'$ or $v \in V'$

Idea: Instead of minimizing, it is often useful to ask the question of maximizing a related quantity. In this case, let's ask the question of maximizing the number of edges that we can find in E such that no two edges share a vertex in G.

A set of edges where no two edges share a vertex in common is called a matching (independent edge set).

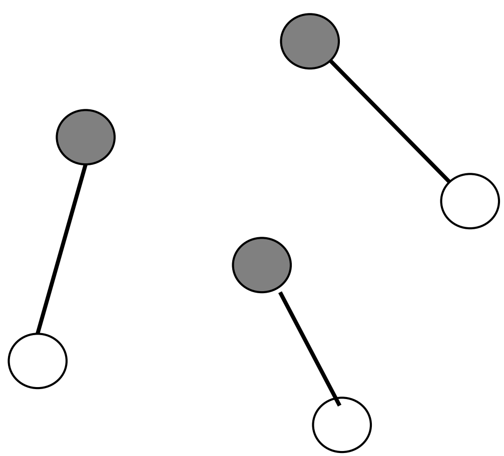
If we have a matching M of the graph G, what is the minimum number of vertices from V required to cover the edges in M?

Question 2

Answer: $|M|$

In a matching M, no two edges share the same vertex.

Hence, require exactly $|M|$ vertices to cover all the edges in M.



Matching of size 3
Cover of size 3

Question 3

Given a simple undirected graph $G=(V,E)$, we would like to find a *matching* in the graph, i.e. a set of edges *M* such that *no two edges share a vertex*.

Consider the following algorithm for finding *M*:

1. Select an arbitrary edge
2. Add edge (u,v) to M and remove all edges covered by *u* and *v* from E.
3. If $E \neq \phi$, goto step 1.

Let V_M be the set of vertices incident to the edges in *M* (all vertices connected to the edges) i.e. $V_M=\{v : \exists\ u \in V$ such that $\{u,v\} \in M\}$.

When the algorithm terminates, which statements are correct?

1. M is a matching
2. V_M is a vertex cover.
3. If M is a maximum matching, then V_M is a minimum vertex cover.

Question 3

Answer: (1) and (2) are correct.

Let initial set of edges be *E'*.

Invariant: *M* contains a matching of removed edges $E' - E$, V_M covers $E' - E$, and *E* does not share any vertex with *M*.

Initialization: True as start with empty *M* and no removed edges.

Maintainance: Assume true at start of iteration. Can add edge to *M* as *E* does not share vertex with *M*. Remove all edges in *E* that shares vertex with newly added edge, so *E* does not share vertex after iteration. Both vertices from added edge added to V_M so the removed edges covered by V_M.

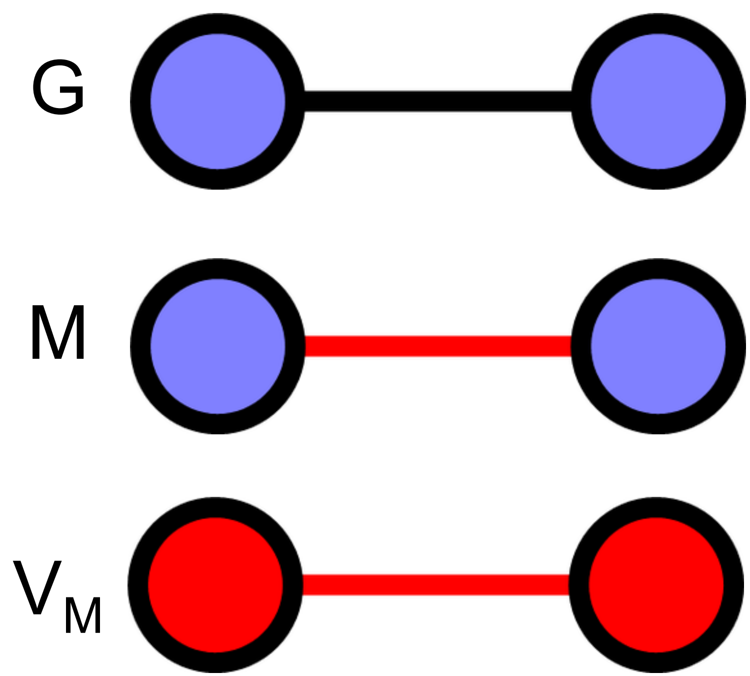
Termination: *E* is empty, so we have a matching and a cover for *E'*.

Question 3

Answer: (3) is not correct.

If M is a maximum matching, then V_M may not be a minimum vertex cover.

Counter example:



Question 4

Given a simple undirected graph $G = (V, E)$, we would like to find a *matching* in the graph, i.e. a set of edges *M* such that *no two edges share a vertex*.

Consider the following algorithm for finding *M*:

1. Select an arbitrary edge
2. Add edge (u, v) to M and remove all edges covered by *u* and *v* from E.
3. If $E \neq \phi$, goto step 1.

Let V_M be the set of vertices incident to the edges in *M* (all vertices connected to the edges) i.e. $V_M = \{v : \exists u \in V \text{ such that} \{u, v\} \in M\}$.

Argue that the algorithm for finding V_M is a 2-approximation algorithm for vertex cover.

Question 4

Answer:

- Question 2 shows that size of optimal cover for M (hence for E, since M is subset of E) is at least $|M|$.
  - Opt $\geq |M|$
- Question 3 shows that the vertices incident to edges in M forms a vertex cover V_M of size $2|M|$
  - $|V_M| = 2|M| \leq 2$ Opt