CS 5003: Parameterized Algorithms

Lecture 19

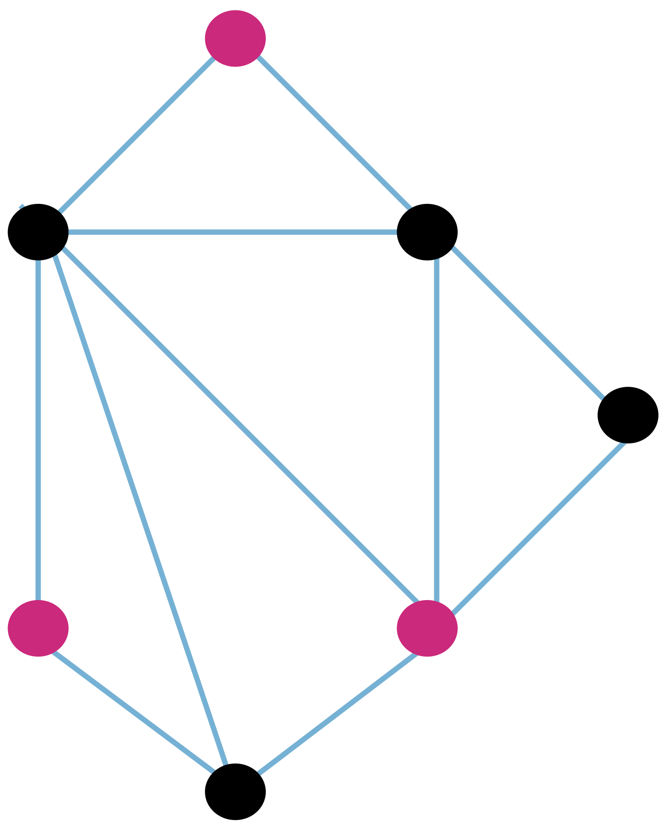
Krithika Ramaswamy

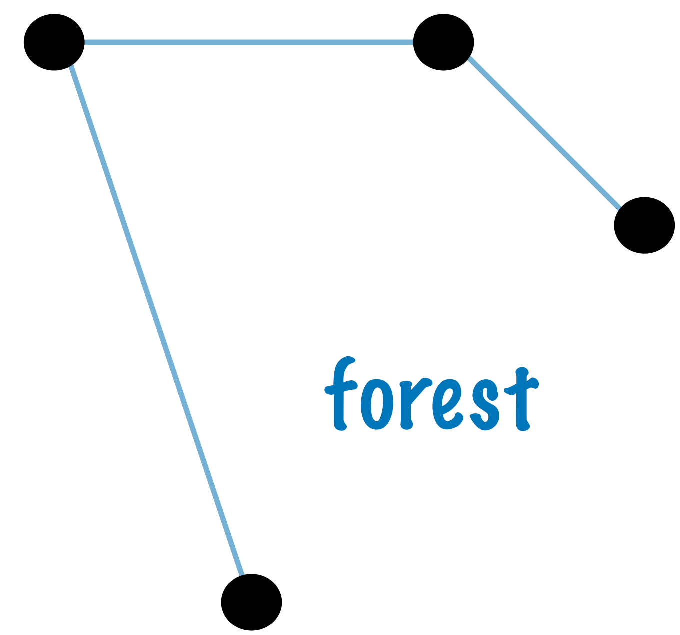
IIT Palakkad

References: Parameterized Algorithms by Cygan et al. and Kernelization by Fomin et al.

Feedback Vertex Set

FVS - set of vertices that has at least one vertex of every cycle



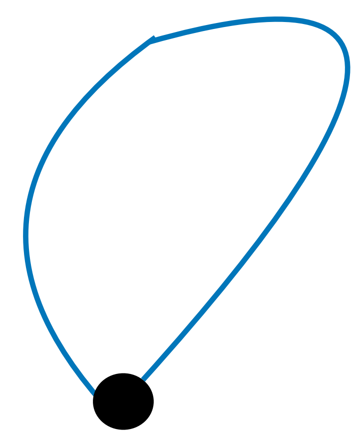


Feedback Vertex set
<u>Instance:</u> An undirected graph G and an integer k
<u>Question:</u> Does there exist a feedback vertex set of G of size at most k?
<u>Parameter:</u> k

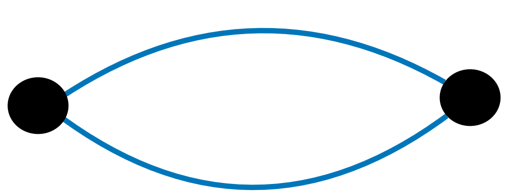
Feedback Vertex Set

Assume graph is a multigraph

* **Reduction Rule 1:** Delete isolated vertices
* **Reduction Rule 2:** Delete degree-1 vertices
* **Reduction Rule 3:** If there is a loop at a vertex v, delete v from the graph and reduce the parameter by 1

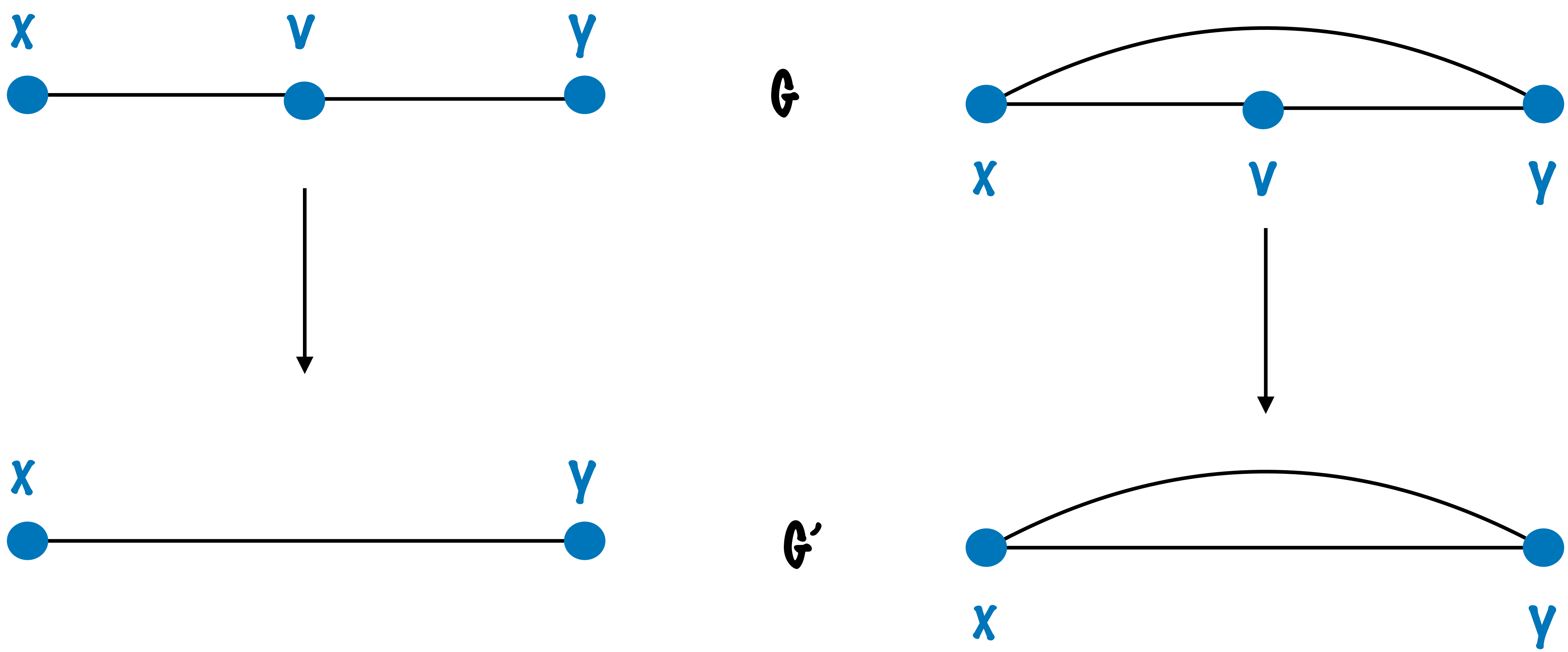


* **Reduction Rule 4:** If there is an edge with multiplicity > 2, reduce it to 2



Feedback Vertex Set

* Reduction Rule 5: Short circuit degree-2 vertices

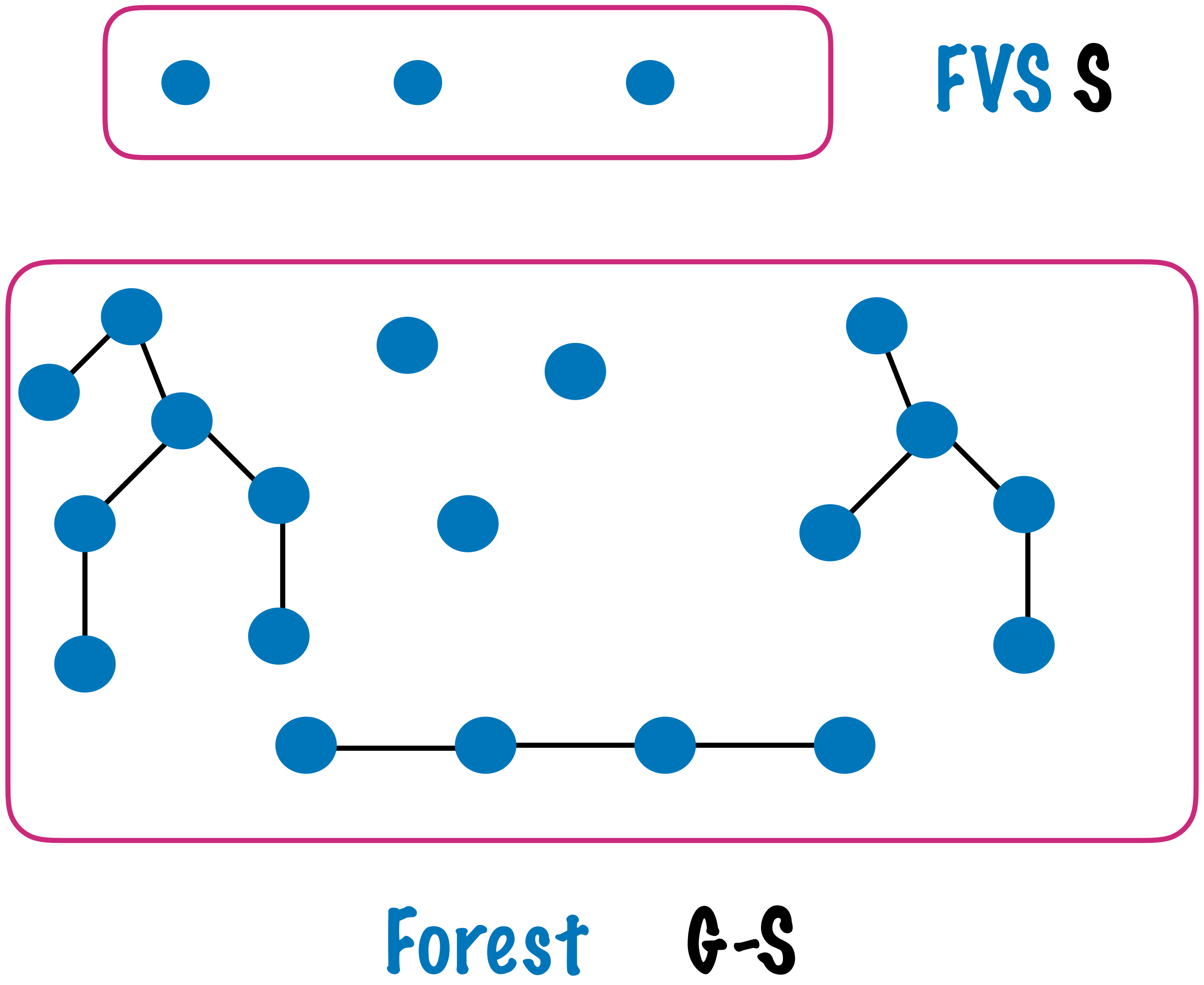


There exists a minimum FVS that does not contain v

(G,k) is an yes-instance iff (G′,k) is an yes-instance

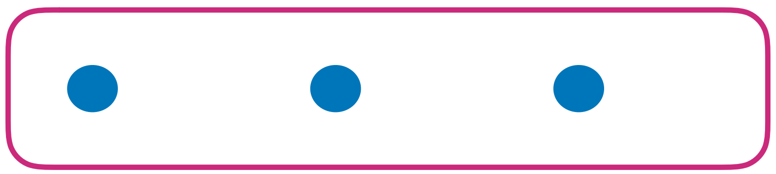
Feedback Vertex Set

Lemma: If G is graph with minimum degree >= 3, then number of edges incident to any FVS S is > |E(G)|/2



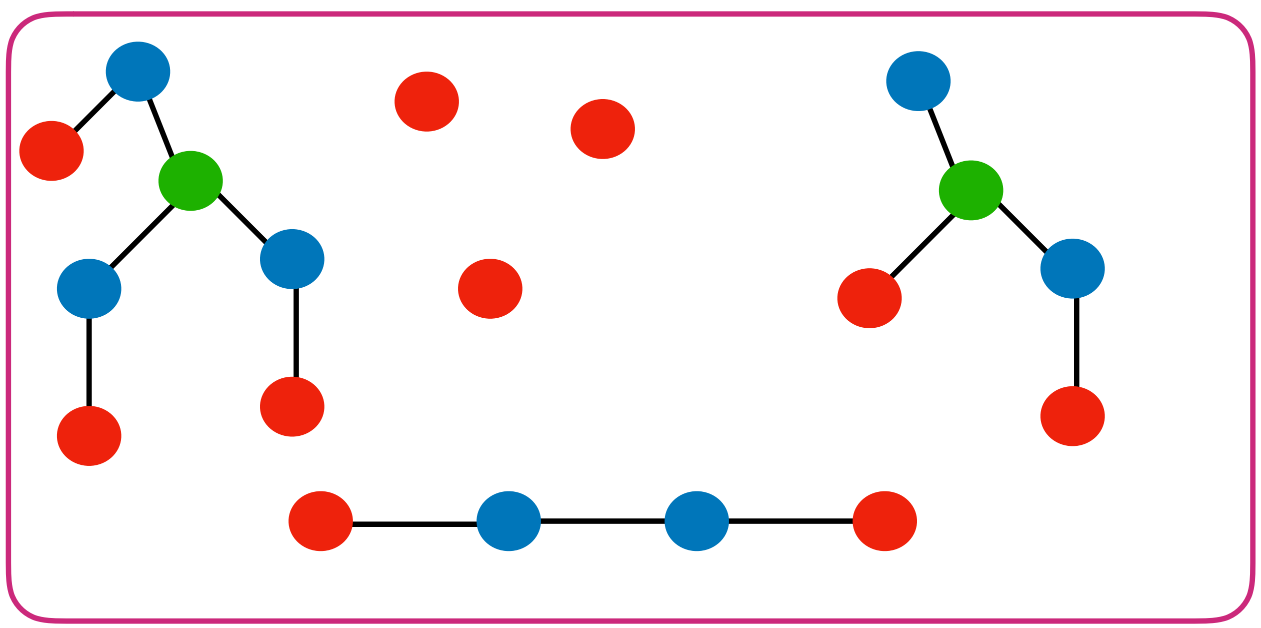
Feedback Vertex Set

Lemma: If G is graph with minimum degree >= 3, then number of edges incident to any FVS S is > |E(G)|/2



FVS S

In H, let V1 denote vertices with degree atmost 1 in H, V2 degree = 2,



Forest H=G-S

$$E(H,S) >= 2\,V_1 + V_2 > V_1 + V_2 + V_3 > E(H)$$

$$E(H) = E(G) - E(H,S) - E(S)$$
$$< E(G) - E(H) - E(S)$$
$$<= E(G) - E(H)$$

$$E(H) < E(G)/2$$

$$E(S) + E(H,S) > E(G)/2$$

Algorithm

* **Step 1:** Initialize $S=\emptyset$
* **Step 2:** Execute the following steps k times
    * **Step 2.1:** Apply preprocessing rules to get equivalent instance (G',k')
    * **Step 2.2:** Pick an edge uniformly at random
        * Edge $e=\{u,v\}$ is picked w.p $1/|E(G')|$
    * **Step 2.3:** Pick a vertex x from $\{u,v\}$ uniformly at random
        * Vertex v is picked w.p 1/2 and vertex u is picked w.p 1/2
    * **Step 2.4:** Add x to S and delete x from G
* **Step 3:** If S is an FVS of G return yes, otherwise return no.

Analysis

* Running Time: Polynomial
* Correctness:
  * If (G,k) is a no-instance then Algorithm always outputs no.
  * Suppose (G,k) is an yes-instance and F is a <=k FVS
    * Let H=G-F
    * Pr (an edge in $E(F) \cup E(H,F)$ is chosen) > 1/2
    * Pr (a vertex from F is chosen) > 1/2 . 1/2 = 1/4
    * Pr(S=F) > $(1/4)^k$
    * Pr(Algorithm says yes) > $(1/4)^k$

Theorem: Feedback Vertex Set can be solved in randomized polynomial time, with success probability at least 4^{-k}.

Algorithm

Given an input instance (G,k), run the following algorithm 4^k times. If none of the executions return yes, then declare that (G,k) is a no-instance. Otherwise, declare that (G,k) is an yes-instance.

* Step 1: Initialize $S=\emptyset$
* Step 2: Execute the following steps k times
  * Step 2.1: Apply preprocessing rules to get equivalent instance (G',k')
  * Step 2.2: Pick an edge uniformly at random
    * Edge $e=\{u,v\}$ is picked w.p $1/|E(G')|$
  * Step 2.3: Pick a vertex x from $\{u,v\}$ uniformly at random
    * Vertex v is picked w.p 1/2 and vertex u is picked w.p 1/2
  * Step 2.4: Add x to S and delete x from G
* Step 3: If S is an FVS of G, return yes

Analysis

Identity: $(1-p)^t \le (e^{-p})^t$

* Running Time: $O^*(4^k)$
* Correctness:
    * If (G,k) is a no-instance then Algorithm always outputs no.
    * Suppose (G,k) is an yes-instance and F is a <=k FVS
        * Let H=G-F
        * Pr (an edge in $E(F) \cup E(H,F)$ is chosen) > 1/2
        * Pr (a vertex from F is chosen) > 1/2 . 1/2 = 1/4
        * $Pr(S=F) > (1/4)^k$
        * Pr(Algorithm says no) < $(1-(1/4)^k)$ ^ $(4)^k \le 1/e$
        * Pr(Algorithm says yes) > 1- 1/e >= 1/2

Theorem: Feedback Vertex Set can be solved in randomized $O^*(4^k)$ time, with constant success probability.