CS 5003: Parameterized Algorithms

Lecture 3

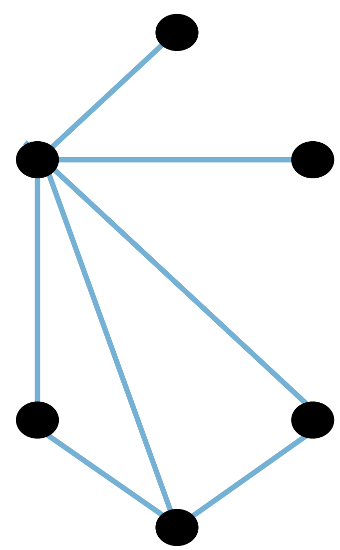
Krithika Ramaswamy

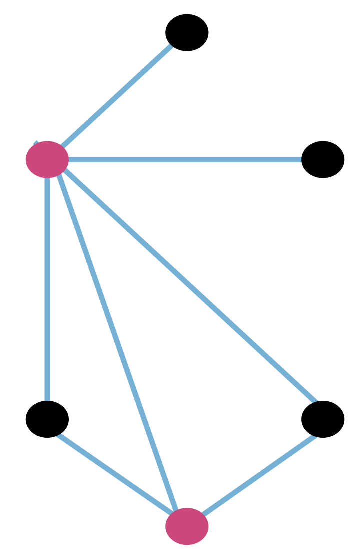
IIT Palakkad

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

Vertex Cover

Vertex cover - set of vertices that has at least one endpoint of each edge



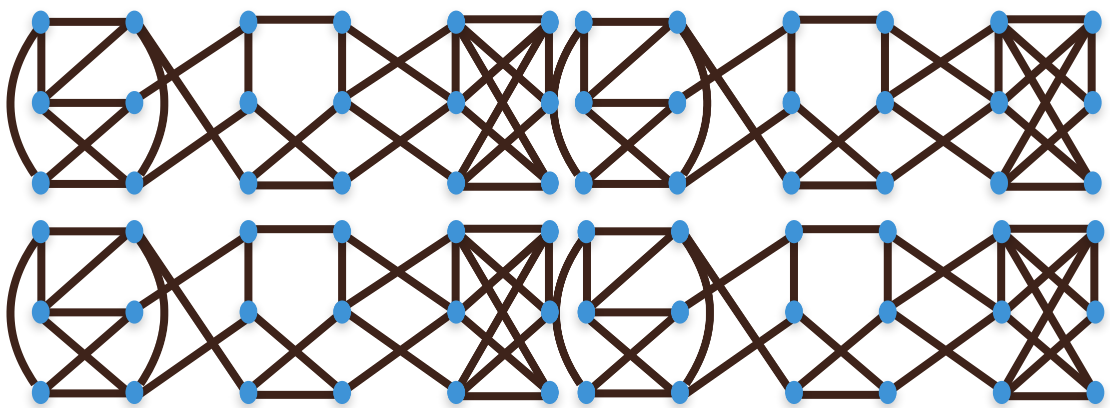


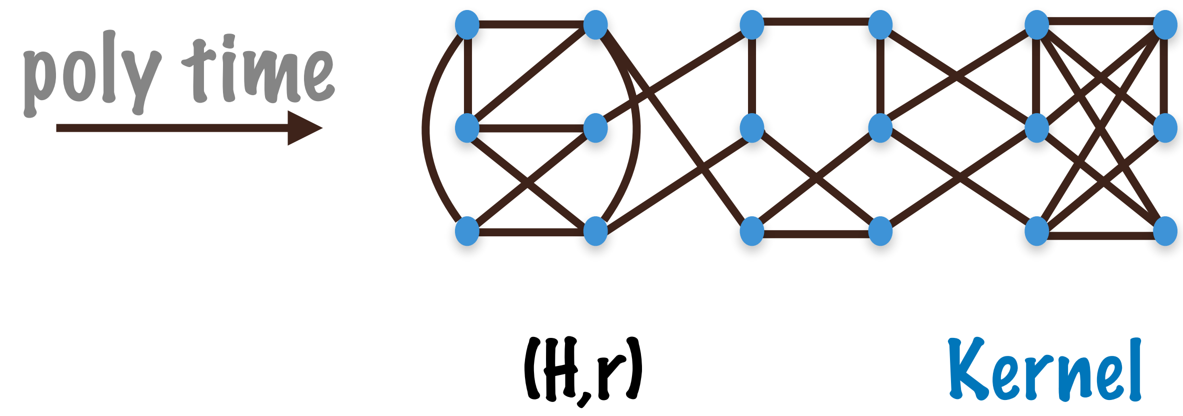
Instance: A graph G on n vertices m edges and integer k
Question: Does G have a vertex cover of size at most k?
Parameter: k

f(k) poly(n,m)

Theorem: VC is FPT with respect to the solution size as parameter

Kernelization: A way to get an FPT Algorithm





Compress input so that its size is bounded by a function of k

Solving original instance (G,k) is equivalent to solving reduced instance (H,r)

Kernel => FPT

A Quadratic Kernel for Vertex Cover

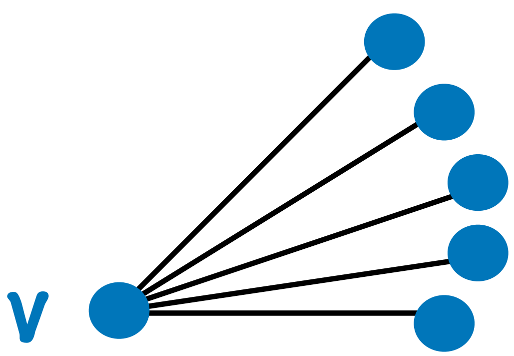
Reduction Rule 1: Delete isolated vertex v

Resulting Instance: (G-v,k)

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

n <= 2m

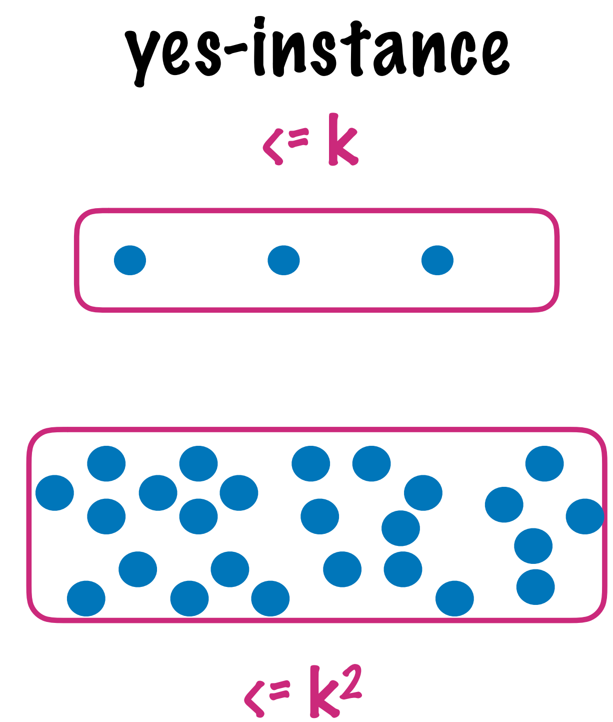
Reduction Rule 2: Delete high degree vertices



>= k+1 neighbours

Resulting Instance: (G-v,k-1)

(G,k) is an yes-instance iff (G-v,k-1) is an yes-instance

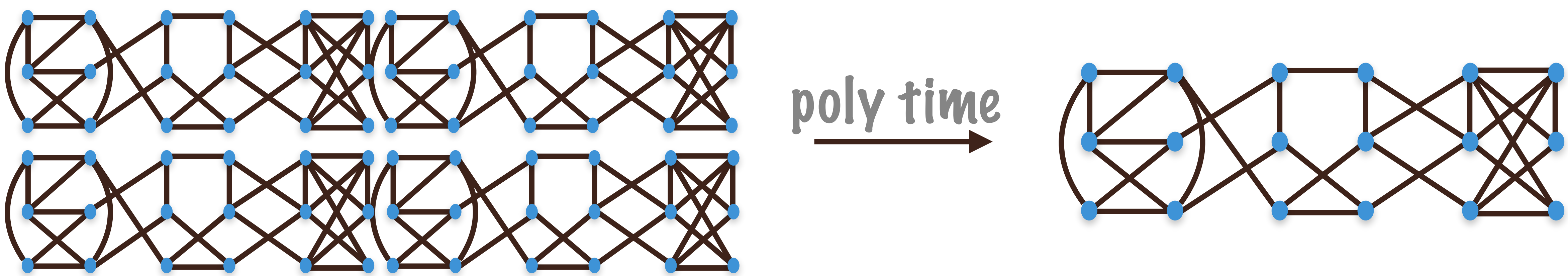


Theorem: Vertex Cover admits a kernel with <= k(k + 1) vertices & k^2 edges.

Kernelization: An Equivalent Notion of FPT

A problem parameterized is FPT iff it has a kernel!

How small can a kernel be?



(H,r)

Poly-size kernel

(G,k)

Compress input so that its size is bounded by a polynomial function of k

Solving original instance (G,k) is equivalent to solving reduced instance (H,r)

Not all problems that are FPT have polynomial kernels!

A Smaller Kernel for Vertex Cover

k^2 edges and $2k^2/3$ vertices

Reduction Rule 1: Delete isolated vertices

$$n <= 2m$$

Reduction Rule 2: Add vertices of degree >= k+1 to solution

$$n <= k(k + 1) \text{ and } m <= k^2$$

Let v be a deg 1 vertex in G. Then there exist a min v.c. S s.t. v is not in S

Reduction Rule 3: Add neighbours of degree-1 vertices to solution

$2k^2 >= 2m = \Sigma \deg(v) >= 2n$

$$n <= k^2 \text{ and } m <= k^2$$

A Smaller Kernel for Vertex Cover

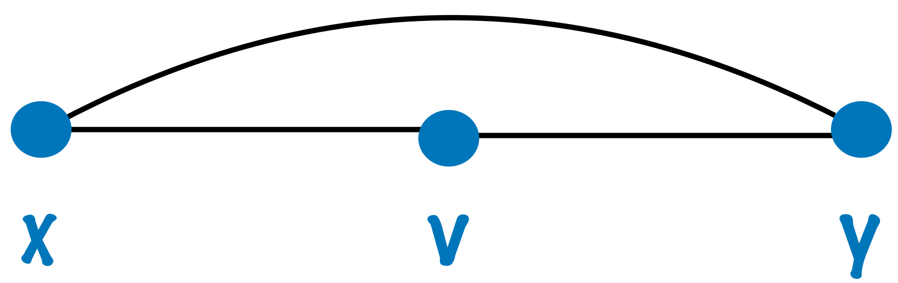
k^2 edges and $2k^2/3$ vertices

Suppose min deg is at least 3

$$\boxed{2k^2 >= 2m = \Sigma \deg(v) >= 3n}$$

Case 1: Suppose x, y are adjacent

Reduction Rule 4: For degree-2 vertices



Any VC has at least 2 from {v,x,y}

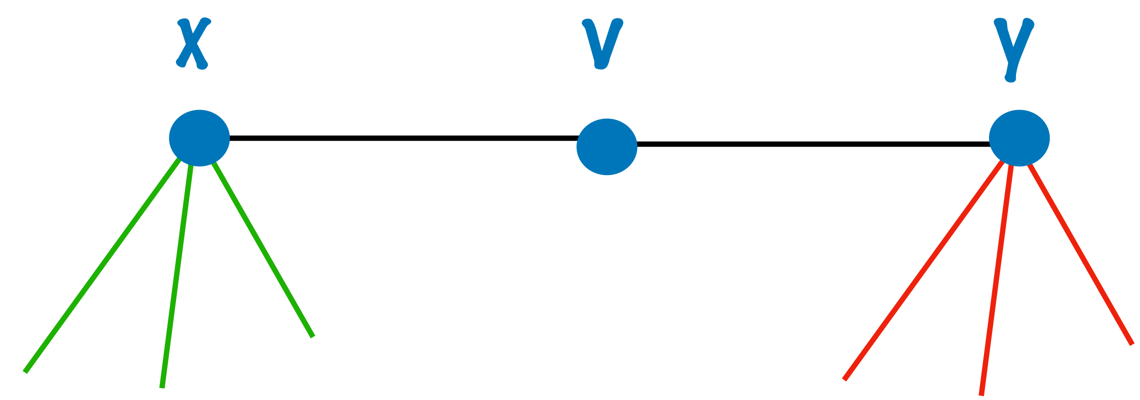
Add x and y to solution and the resulting instance is (G-{v,x,y},k-2)

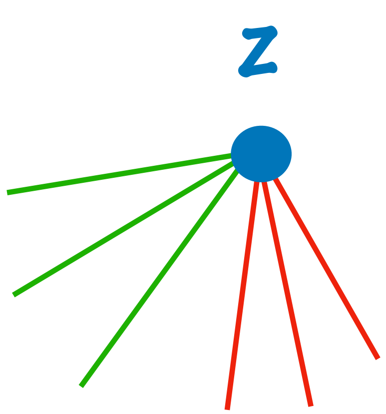
Ex: Prove Correctness of the rule

A Smaller Kernel for Vertex Cover

k^2 edges and $2k^2/3$ vertices

Reduction Rule 4: For degree-2 vertices



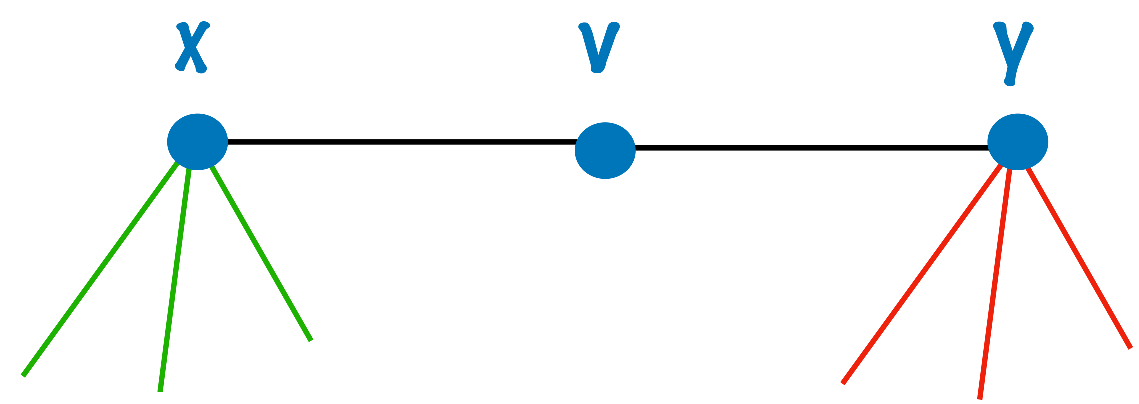


Resulting instance is (G',k-1)

(G,k) is an yes-instance iff (G',k-1) is an yes-instance

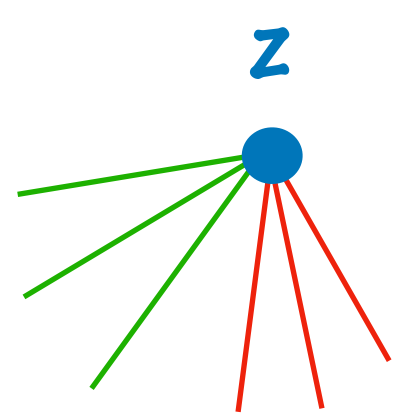
A Smaller Kernel for Vertex Cover

k^2 edges and $2k^2/3$ vertices



Suppose (G,k) is an yes-instance

* S is vertex cover of <= k vertices
* Not all 3 vertices need to be in a minimal vertex cover
* Either v is in S or x and y are in S



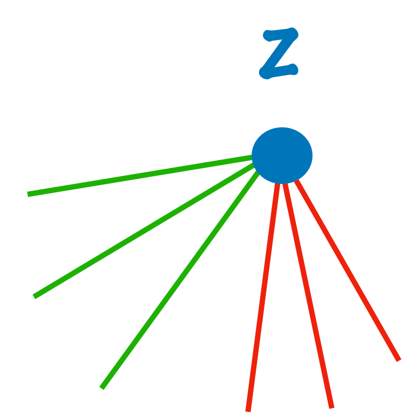
- Suppose x and y are in S
  - Delete x and y from S
  - Add z to S

- Suppose v is in S
  - Both x and y are not in S
  - N(x), N(y) in S
  - Delete v from S

Vertex cover <= k-1 for G'

A Smaller Kernel for Vertex Cover

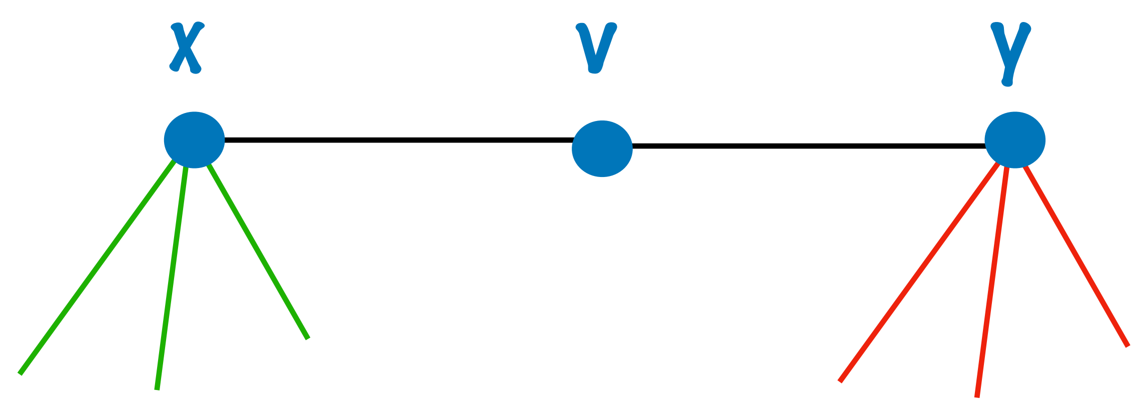
k^2 edges and $2k^2/3$ vertices



Suppose (G',k-1) is an yes-instance

* T is vertex cover of <= k-1 vertices
* Either z in T or z is not in T

- Suppose z is in T
  - Delete z from T
  - Add x and y to T



- Suppose z is not in T
  - N(x), N(y) in T
  - Add v to T

Vertex cover <= k for G

An Even Smaller Kernel for Vertex Cover

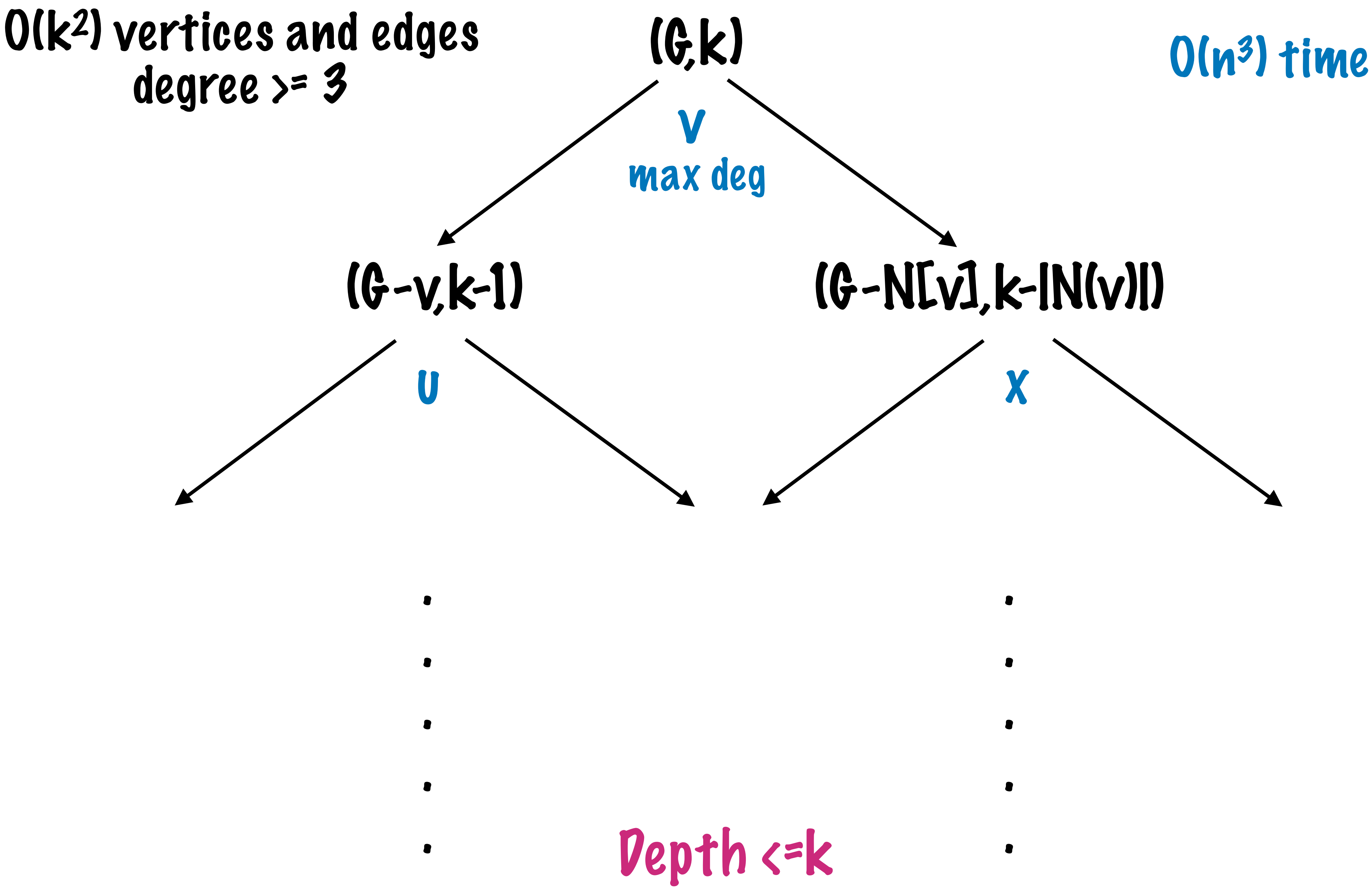
k^2 edges and $2k$ vertices

No $O(k^{2-c})$ edges kernel is likely!

for any $c > 0$

A Faster FPT Algorithm

An $O^*(1.4656^k)$ algorithm



Apply preprocessing rules (reduction rules) at each node

A Faster FPT Algorithm

How many nodes are there?

q leaves => no. of nodes <= 2q-1

Let T(k) denote the no. of leaves in the tree rooted at instance with parameter k

(G,k)

v

max deg

k drops by 1 (G-v,k-1) (G-N[v],k-|N(v)|) k drops by 3

. . . .

$c^k = c^{k-1} + c^{k-3}$

$T(k) \le \begin{cases} T(k-1)+T(k-3) & \text{if } k \ge 3 \\ 1 & \text{otherwise} \end{cases}$

c=1.4656

$O(n^3 + 1.4656^k k^3)$ time algorithm

Branching or Depth-Bounded Search Trees