Lossy Kernelization II

Recent Advances in Parameterized Complexity

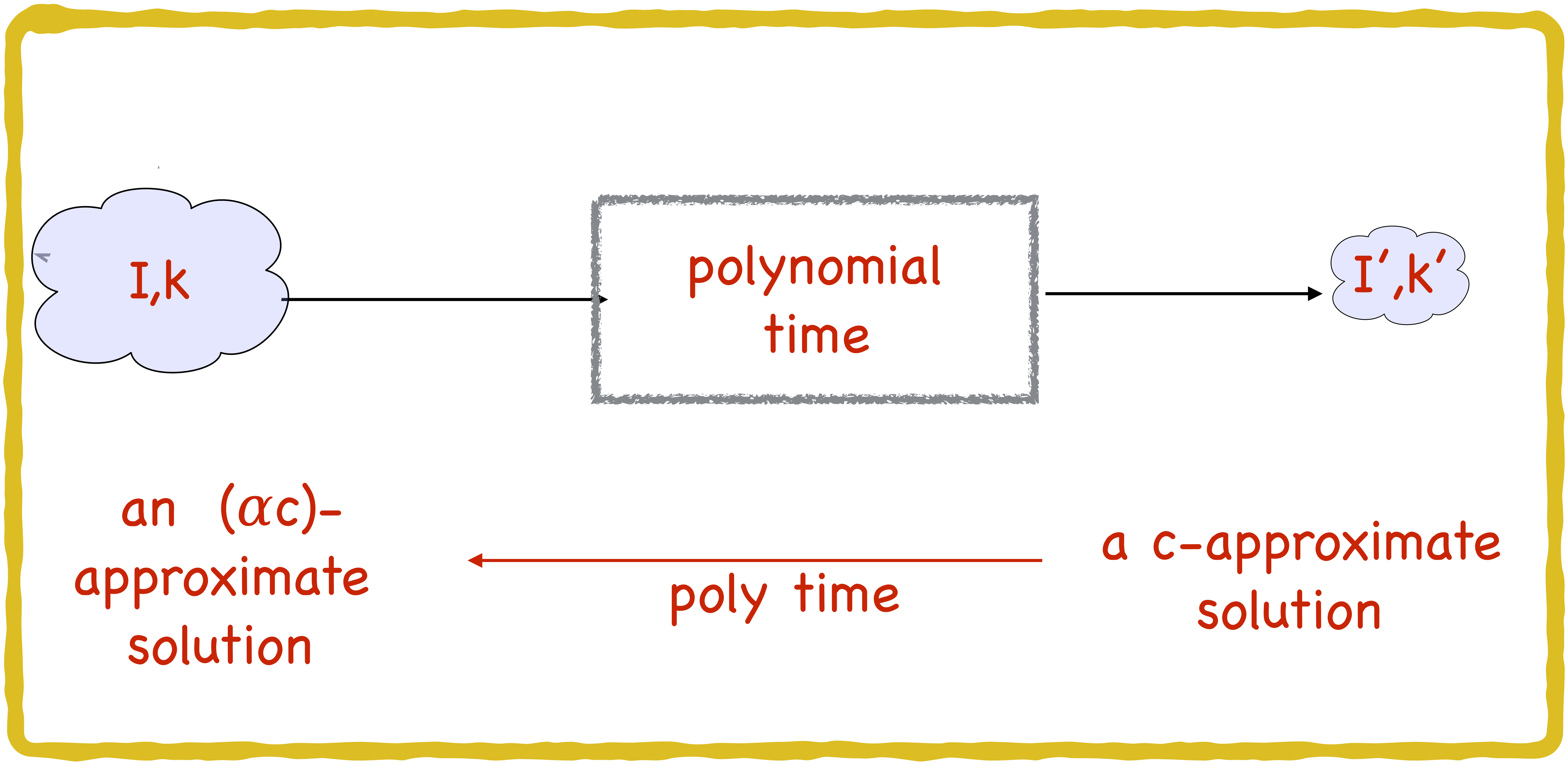
December 2017

Tel Aviv

M. S. Ramanujan

University of Warwick

α-approximate Kernelization



Polynomial α-approximate Kernel if |I'|+k' < poly(k)

Disjoint Factors

a-factor

abdeabdfbafcaddgcag

a-factor

Disjoint Factors

b-factor

abdeabdfbafcaddgcag

b-factor

Disjoint Factors

Given a string L over alphabet Σ, what is the maximum number of DISJOINT factors one can pack?

PARAMETER: |Σ|

In the decision version, we ask: can we pack |Σ| disjoint factors or not?

Disjoint Factors

Given a string L over alphabet Σ, what is the maximum number of DISJOINT factors one can pack?

2-approximable

FPT

APX hard

no polynomial kernel [BTY '11]
(also from Michal's talk)

(1+ϵ)-approximate kernel of polynomial size

PSAKS

PSAKS for Disjoint Factors

Strategy?

1. Express Disjoint Factors as a special kind of independent set problem on interval graphs.

2. Get a PSAKS for this problem

$$(\text{size } |\Sigma|^{O(1/\varepsilon \log 1/\varepsilon)}) = \text{PARAMETER}^{O(1/\varepsilon \log 1/\varepsilon)}$$

PSAKS for Disjoint Factors

1. Express Disjoint Factors as a special kind of independent set problem on interval graphs.

Intervals corresponding to MINIMAL factors

vertex corresponding to this interval with label a

abdeabdfbafcaddgcag

vertex corresponding to this interval with label a

PSAKS for Disjoint Factors

1. Express Disjoint Factors as a special kind of independent set problem on interval graphs.

vertex corresponding
to this interval with
label b

abdeabdfbafcaddgcag

vertex corresponding
to this interval with
label b

Edges defined in the natural way.

PSAKS for Disjoint Factors

1. Express Disjoint Factors as a special kind of independent set problem on interval graphs.

vertex corresponding to this interval with label b

abdeabdfbafcaddgcag

vertex corresponding to this interval with label b

Obtain interval graph G with each vertex having one of |Σ| labels

AND

Disjoint Factors are precisely independent sets of G where no label repeats.

Labeled Interval Independent Set (LIIS)

Given a labeled interval graph G with $|\Sigma|$ labels and chromatic number 2 $|\Sigma|$, find a diversely labeled independent set of maximum size.

PARAMETER: $|\Sigma|$

follows from the fact that we only made vertices corresponding to MINIMAL factors

Labeled Interval Independent Set (LIIS)

Given a labeled interval graph G with $|\Sigma|$ labels and chromatic number 2 $|\Sigma|$, find a diversely labeled independent set of maximum size.

PARAMETER: $|\Sigma|$

follows from the fact that we only made vertices corresponding to MINIMAL factors

Can be used to ensure that each label induces an independent set in G while blowing up the label set by a factor of 2 $|\Sigma|$

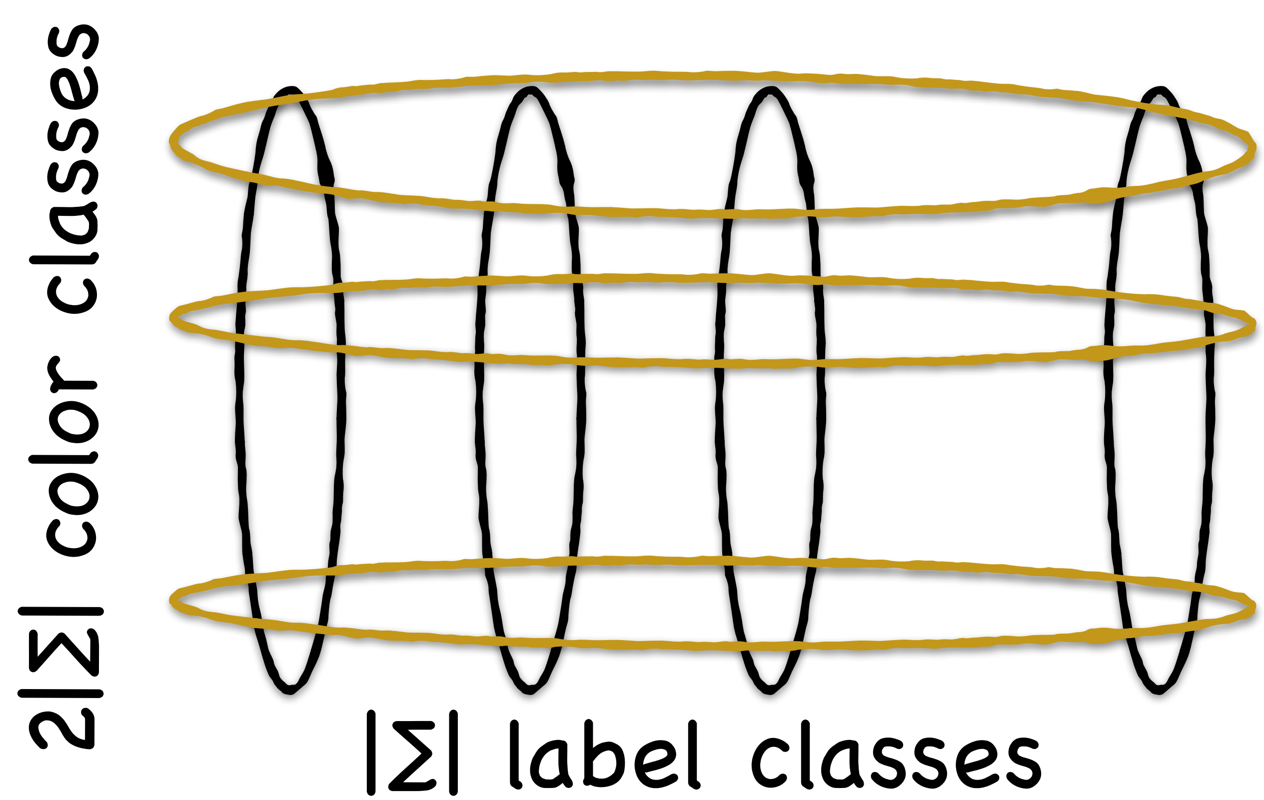
Labeled Interval Independent Set (LIIS)

Given a labeled interval graph G with |Σ| labels and chromatic number 2 |Σ|, find a diversely labeled independent set of maximum size.

PARAMETER: |Σ|

follows from the fact that we only made vertices corresponding to MINIMAL factors

Can be used to ensure that each label induces an independent set in G while blowing up the label set by a factor of only 2 |Σ|

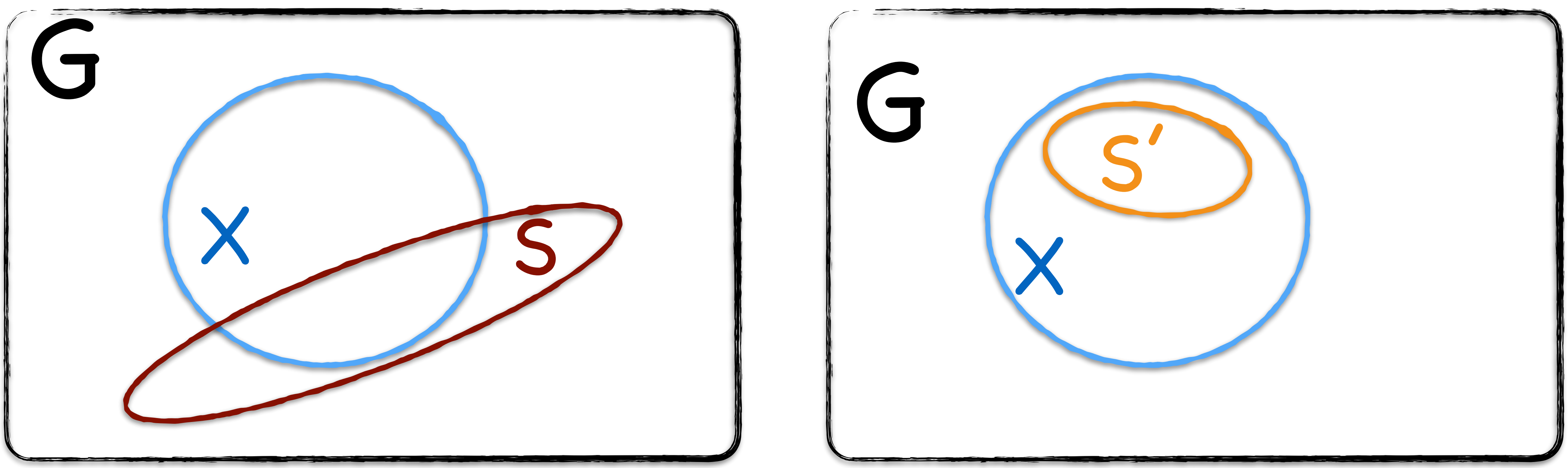


Labeled Interval Independent Set (LIIS)

Cute Covering Lemma: G has a vertex subset X of size $|\Sigma|^{O(1/\varepsilon \log 1/\varepsilon)}$ such that

Labeled Interval Independent Set (LIIS)

Cute Covering Lemma: G has a vertex subset X of size $|\Sigma|^{O(1/\varepsilon \log 1/\varepsilon)}$ such that

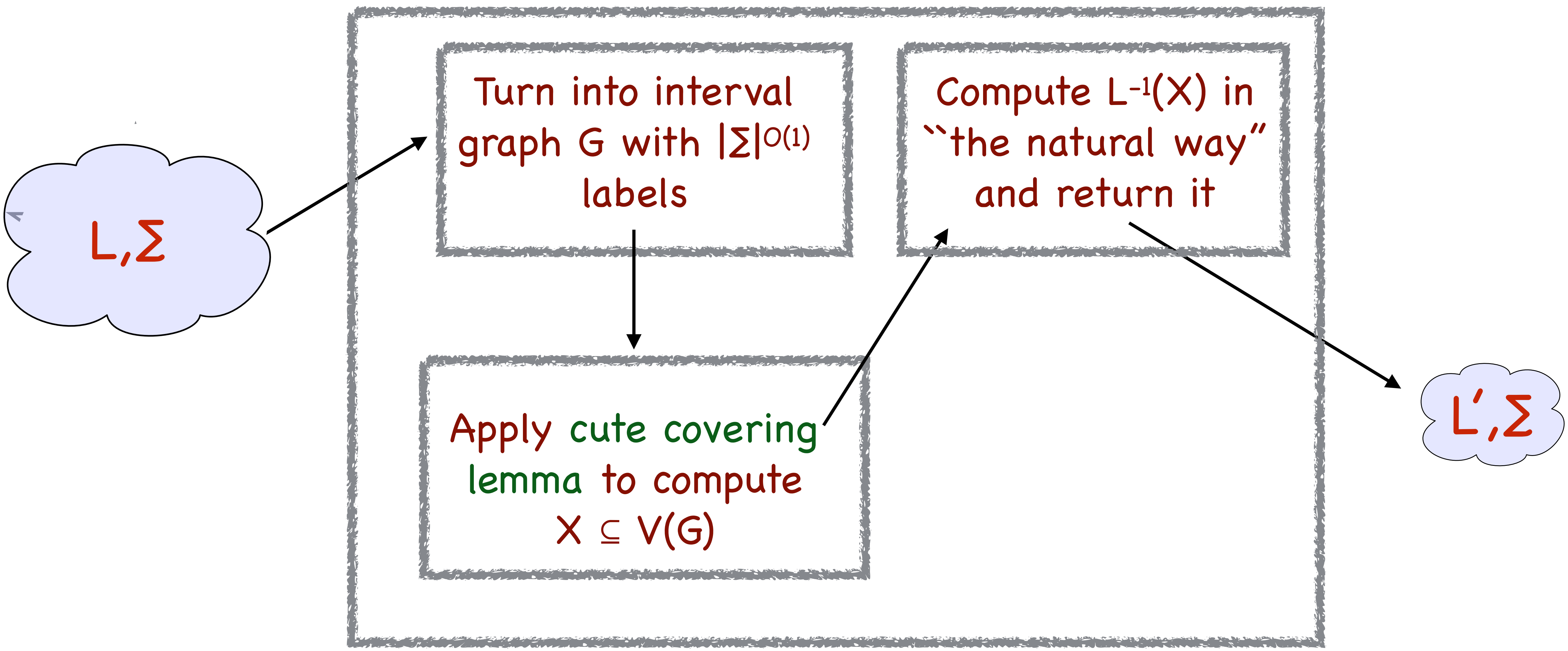


For any independent set S in G

There is an independent set S' in X

1. Label-set(S') ⊆ Label-set(S) and
2. |Label-set(S')| >= (1-ε) |Label-set(S)|

PSAKS for Disjoint Factors



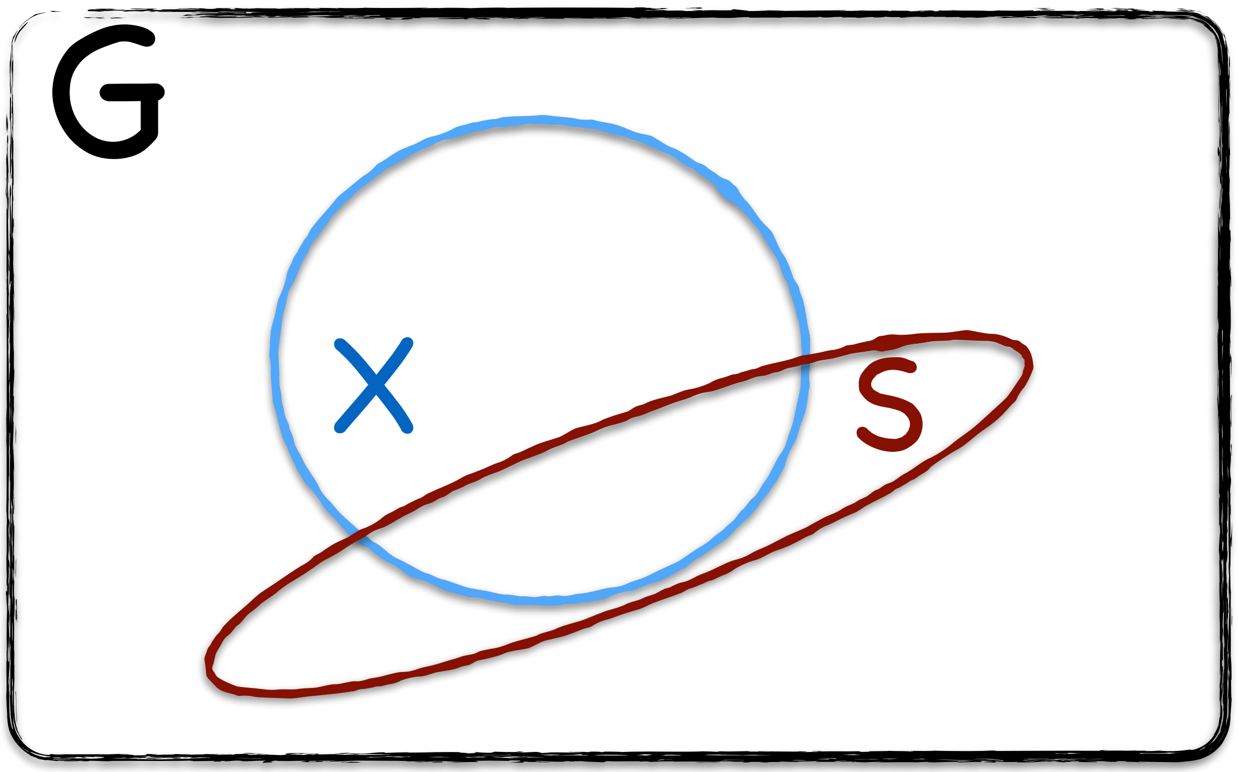
an ((1-ε) c)-approximate solution

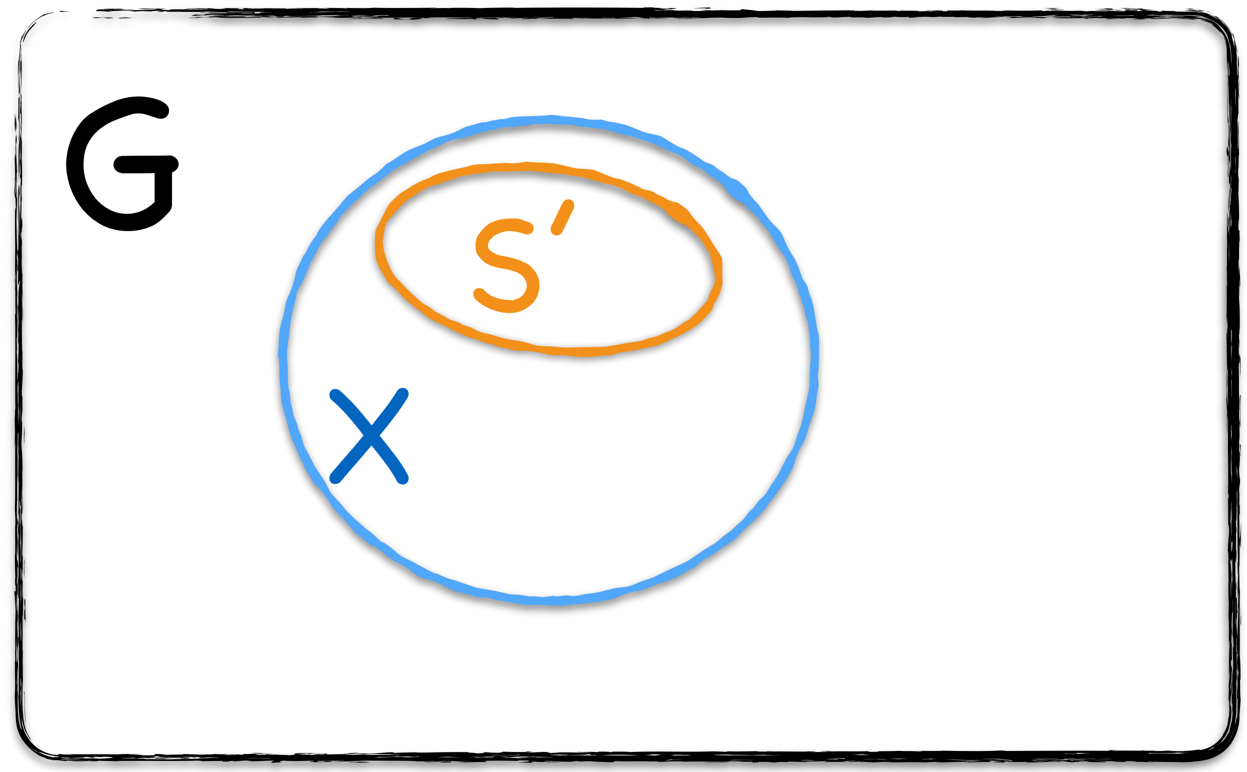
poly time

a c-approximate solution

Labeled Interval Independent Set (LIIS)

Cute Covering Lemma: G has a vertex subset X of size $|\Sigma|^{O(1/\varepsilon \log 1/\varepsilon)}$ such that





For any independent set S in G

There is an independent set S′ in X

1. Label-set(S′) $\subseteq$ Label-set(S) and
2. |Label-set(S′)| >= $(1-\varepsilon)$ |Label-set(S)|

Labeled Interval Independent Set (LIIS)

Rich Labels

Have more than K occurrences

Poor Labels

Have fewer than K occurrences

K=poly(|Σ|) is the total number of labels

Labeled Interval Independent Set (LIIS)

Rich Labels

Have more than K occurrences

We can always find an independent set in G spanning ALL rich labels

Greedily pick the left-most inclusion-wise minimal ``unspanned" label, delete it and recurse.

Labeled Interval Independent Set (LIIS)

Rich Labels

Have more than K occurrences

We can easily take care of independent sets in G that contain only rich labels.

i.e.: the set X that we need is easily defined

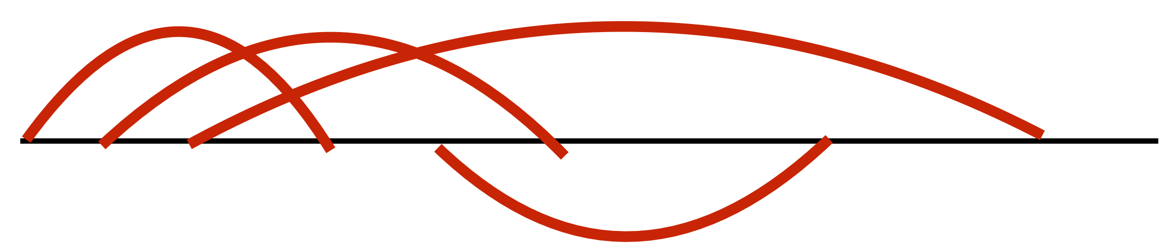
Labeled Interval Independent Set (LIIS)

Poor Labels

Have fewer than K occurrences

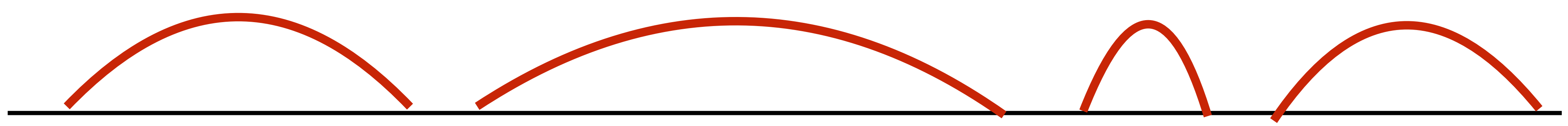
FEW occurrences of poor labels anyway – only K occurrences.

So these labels define a set of K^2 intervals in G



Labeled Interval Independent Set (LIIS)

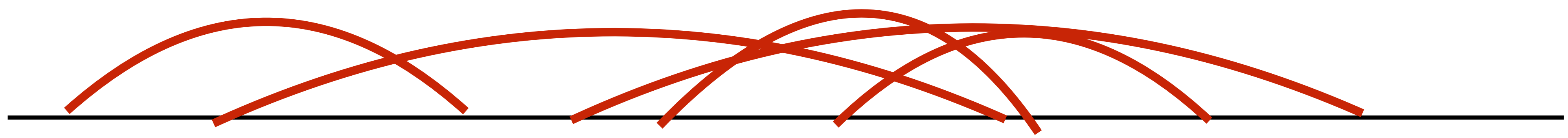
Construction of X



1. Compute rich labels and greedily find an independent set spanning them (one exists!). Mark the endpoints of the corresponding intervals.

Labeled Interval Independent Set (LIIS)

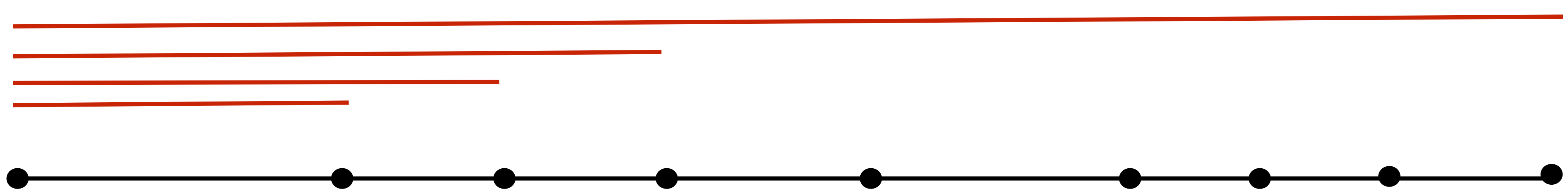
Construction of X



2. Compute poor labels and mark all $O(K^2)$ endpoints of the corresponding intervals.

Labeled Interval Independent Set (LIIS)

Construction of X

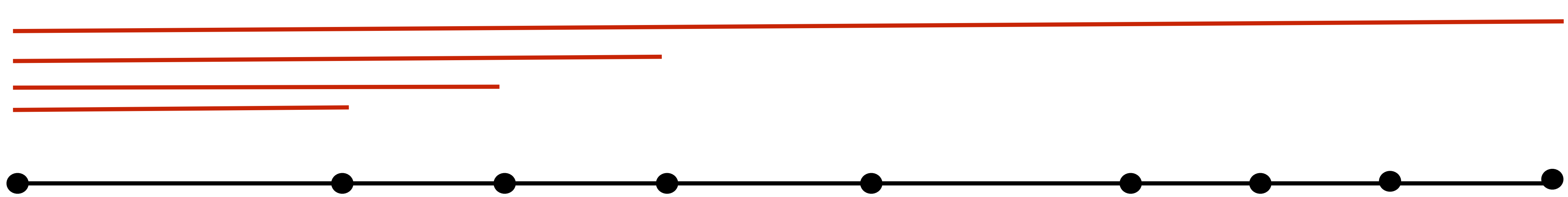


3. Zoom into all $O(K^4)$ intervals defined by the marked set of endpoints and simply recursively compute the set X corresponding to each interval

Finally, return the union of the sets returned by the recursions and the vertices corresponding to the intervals we marked in this step.

Labeled Interval Independent Set (LIIS)

Terminating Condition?



Just stop the recursion at a depth of c. $1/\varepsilon \log 1/\varepsilon$. So the size of the set returned satisfies the required bound.

DANGER: we could lose out on many labels of the target independent set when we make the termination in such an ``arbitrary” manner.

Labeled Interval Independent Set (LIIS)

Let I be the target independent set we want X to "capture"

I contains $\geq (1-\varepsilon)|I|$ rich labels

I contains $< (1-\varepsilon)|I|$ rich labels

Labeled Interval Independent Set (LIIS)

Let I be the target independent set we want X to "capture"

I contains >=(1-ε)|I| rich labels

We already greedily picked an independent set that spans ALL rich labels

We lose <= ε|I| labels of I

Labeled Interval Independent Set (LIIS)

Let I be the target independent set we want X to "capture"

We already marked the endpoints of ALL poor labels

I contains $<(1-\varepsilon)|I|$ rich labels

and pushed the job of approximating the rich labels to the recursive calls

Labeled Interval Independent Set (LIIS)

Let I be the target independent set we want X to "capture"

We already marked the endpoints of ALL poor labels

I contains $<(1-\varepsilon)|I|$ rich labels

We push $\leq (1-\varepsilon)|I|$ labels of I to the next level of recursion

Labeled Interval Independent Set (LIIS)

Let I be the target independent set we want X to "capture"

We already marked the endpoints of ALL poor labels

I contains $<(1-\varepsilon)|I|$ rich labels

Roughly $\leq (1-\varepsilon)^{(c.\ 1/\varepsilon \log 1/\varepsilon)} \leq \varepsilon|I|$ labels of I survive without being captured at the bottom of recursion

Labeled Interval Independent Set (LIIS)

Let I be the target independent set we want X to ``capture"

I contains $>=(1-\varepsilon)|I|$ rich labels

I contains $<(1-\varepsilon)|I|$ rich labels

In total, $<= 2\varepsilon|I|$ labels of I can be missing in the very end!

Labeled Interval Independent Set (LIIS)

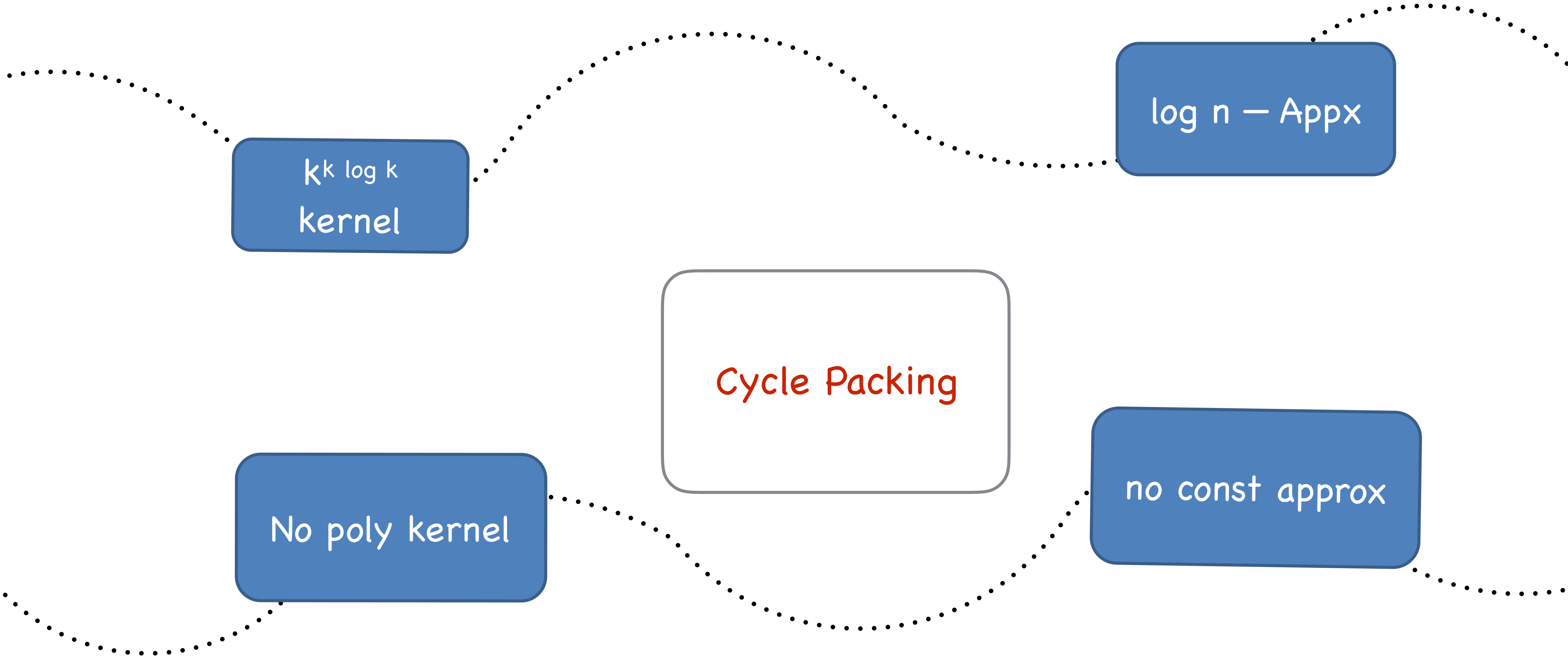
For any I, the set X we computed spans all but $\le 2\varepsilon|I|$ labels of I

Cute Covering Lemma proved!

Disjoint Factors

Cute Covering Lemma →
PSAKS for Disjoint Factors

Cycle Packing



Cycle Packing

$k^{k \log k}$ kernel

log n – Appx

Cycle Packing

No poly kernel

no const approx

(1+ϵ)-approximate kernel of polynomial size

Lower Bounds

Lower Bounds

A problem has a poly time α-approximation
if and only if
it has a α-approximate kernel of constant size

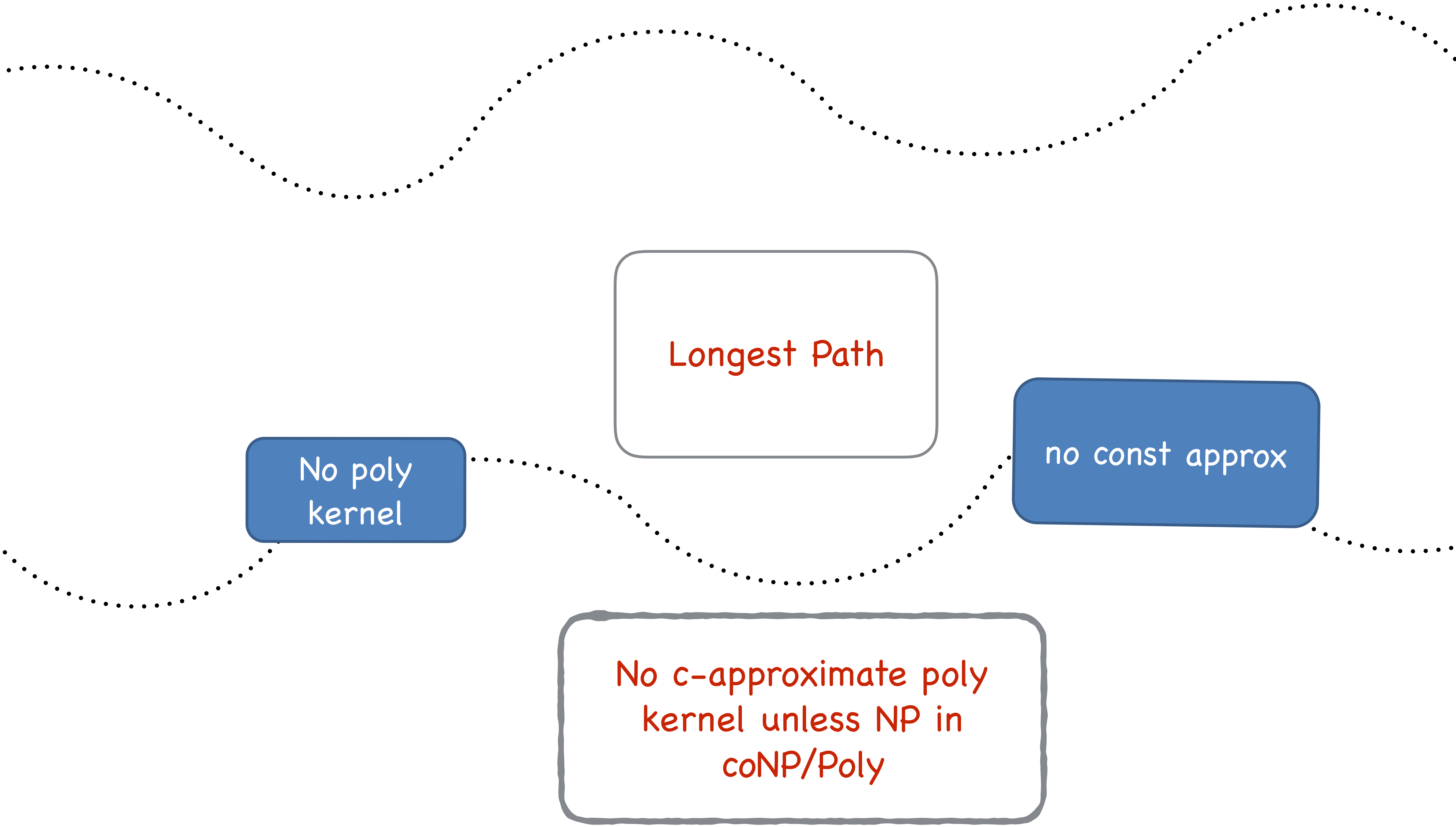
A problem has an FPT α-approximation
if and only if
it has a α-approximate kernel of SOME size

Longest Path

Longest Path

Given a graph G, what is
the length of a longest path in G?

Longest Path



Longest Path

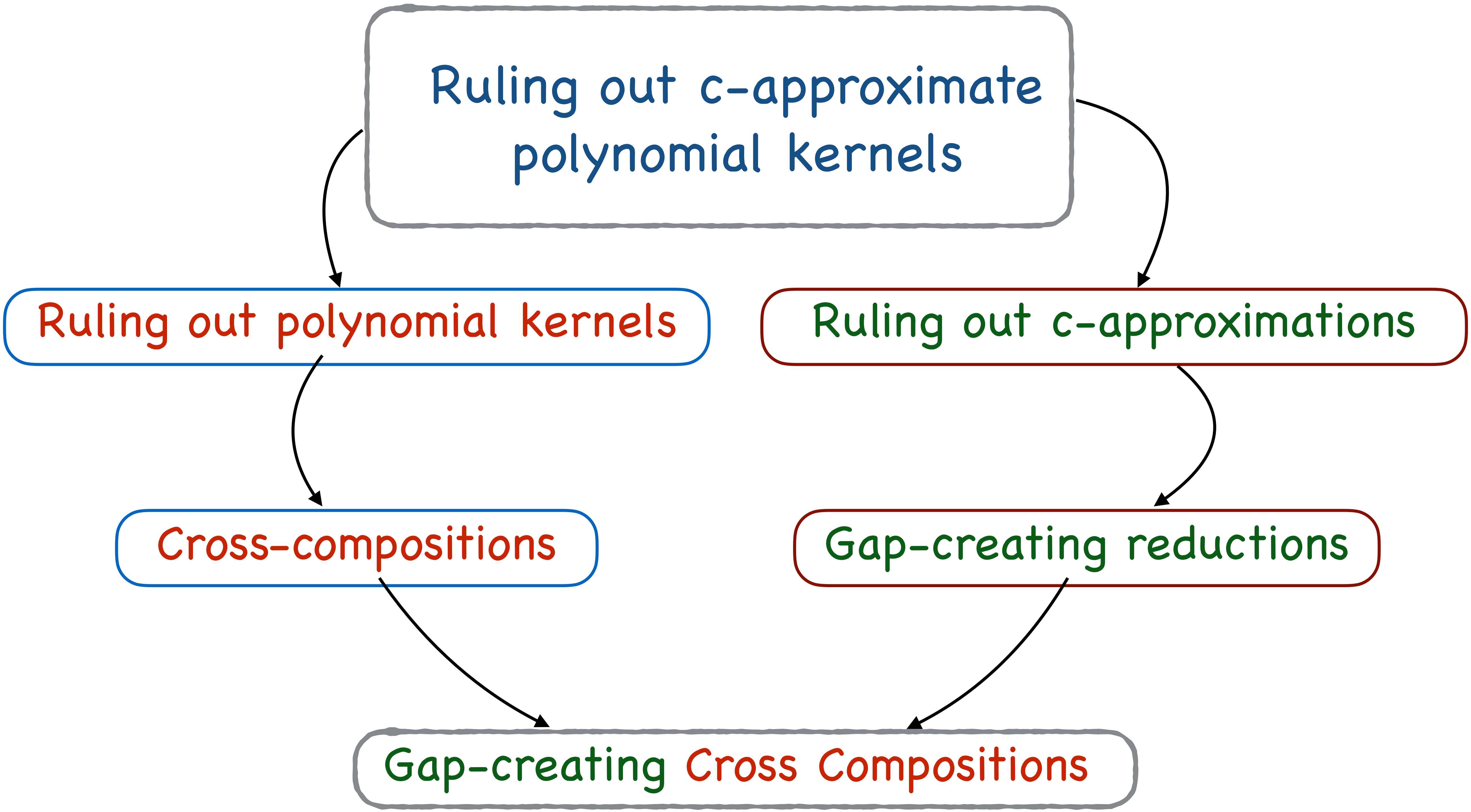
Longest Path

No poly kernel

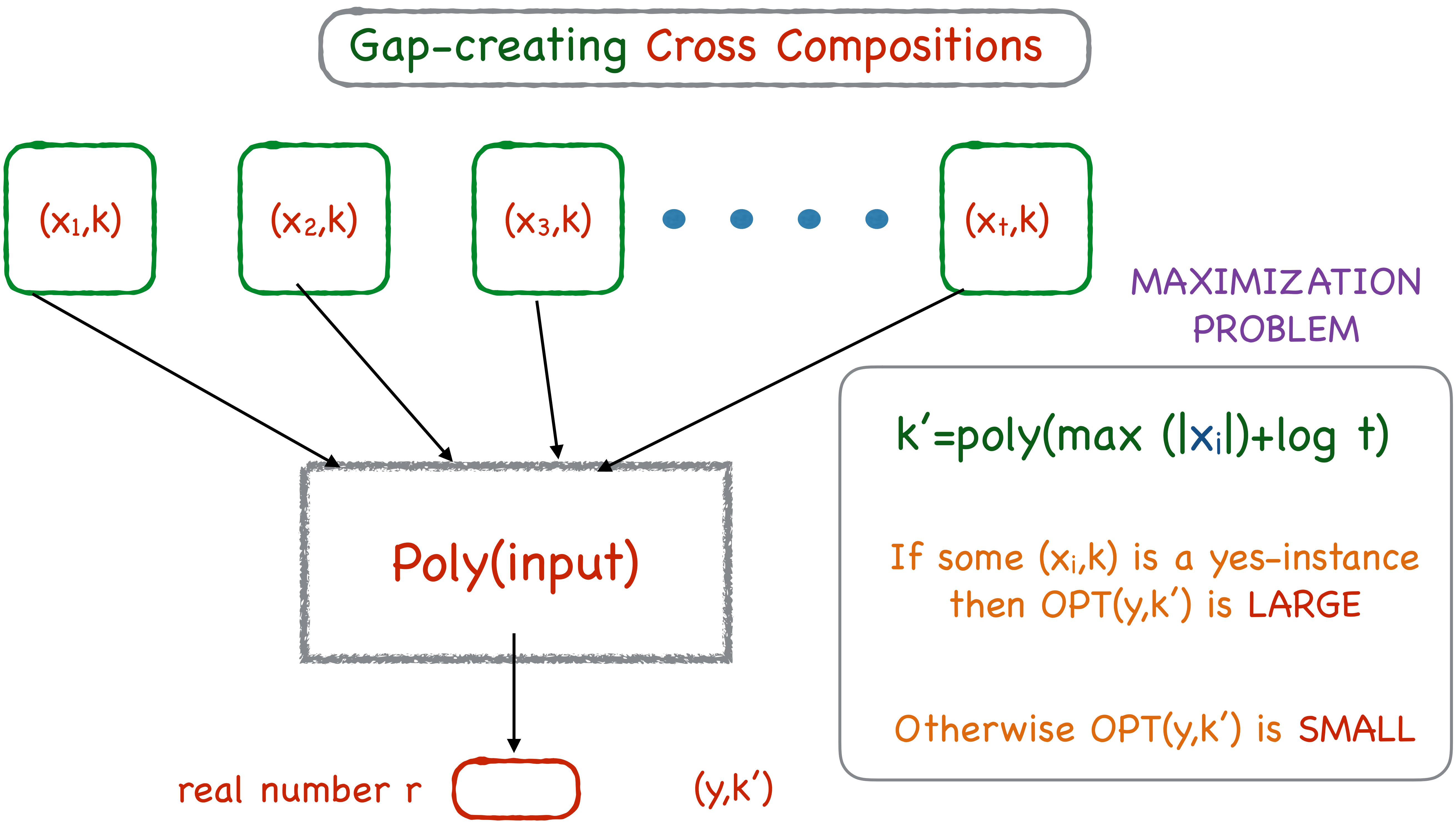
no const approx

No c-approximate poly kernel unless NP in coNP/Poly

Lower Bounds



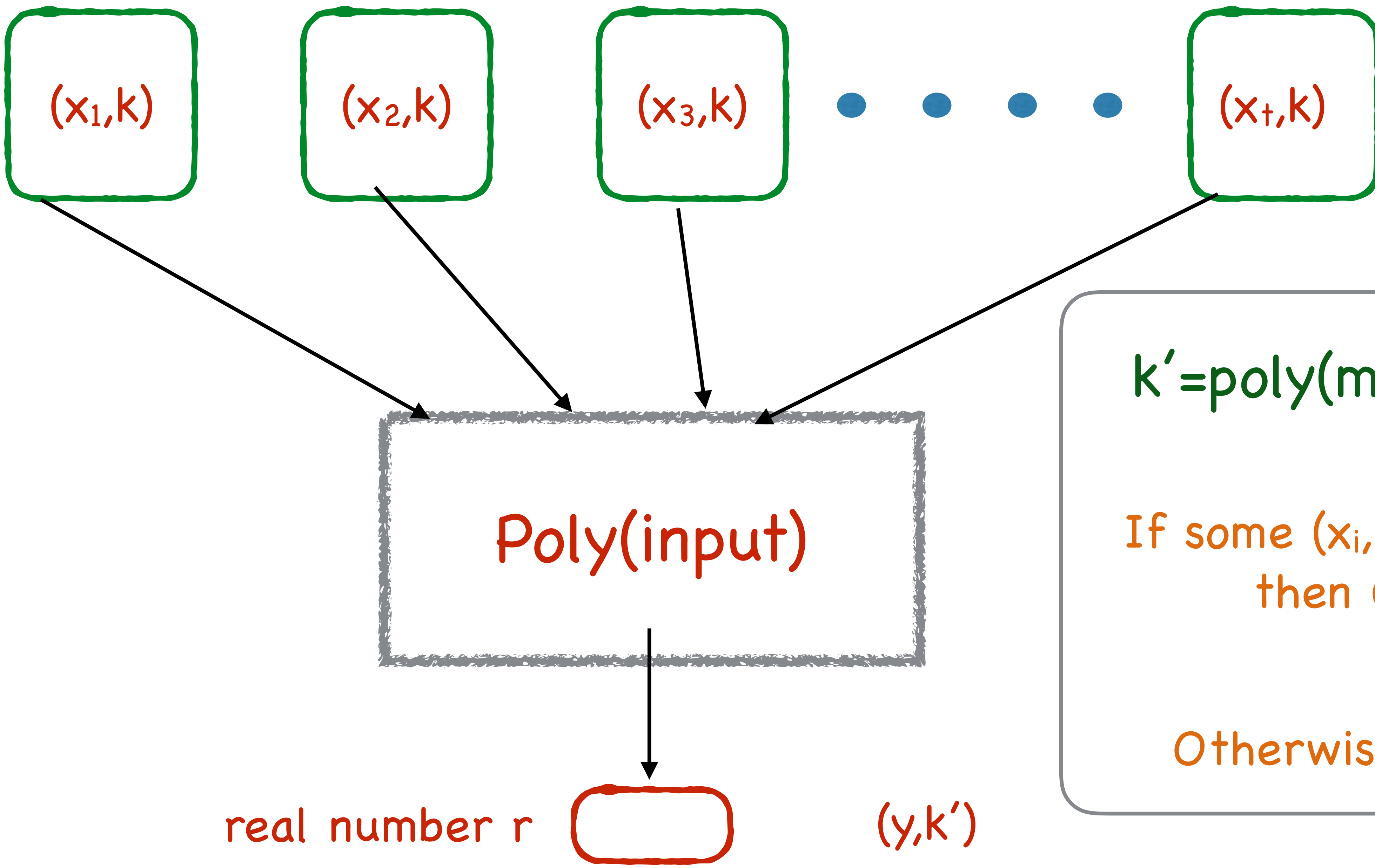
Lower Bounds



Lower Bounds

"""α"""

Gap-creating Cross Compositions



MAXIMIZATION PROBLEM

$k' = poly(\max(|x_i|) + \log t)$

If some (x_i,k) is a yes-instance then $OPT(y,k') >= r$

Otherwise $OPT(y,k') < r/\alpha$

Lower Bounds

``α''

Gap-creating Cross Compositions

Problem P

α-Gap-creating Cross Composition +
α-approximate polynomial
compression

Lower Bounds

α-gap Long Path α-gap cross composes into Longest Path

↓

Longest Path has no α-approximate polynomial kernel unless NP in coNP/poly.

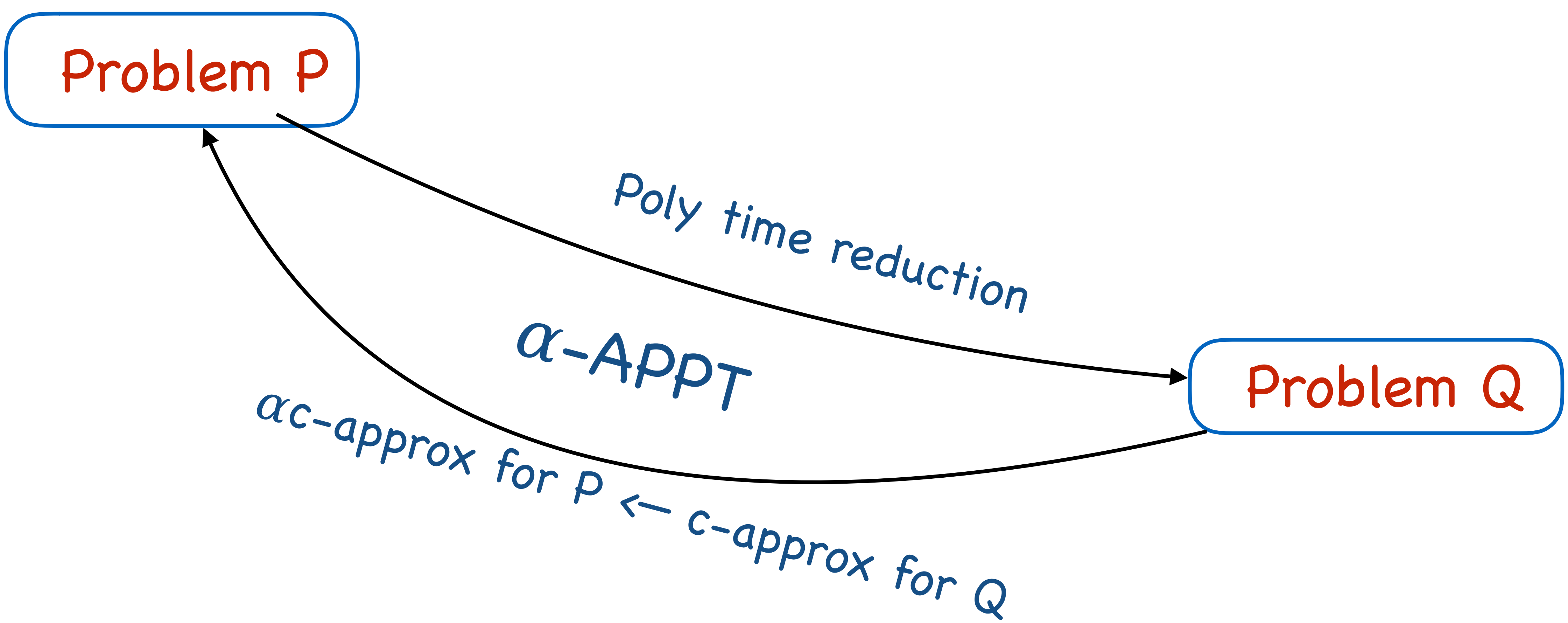
More Lower Bounds

Set Cover (universe size) has no α-approximate polynomial kernel unless NP in coNP/poly.

In previous lower bounds for (standard) kernels, this problem was very useful starting point for polynomial parameter transformations.

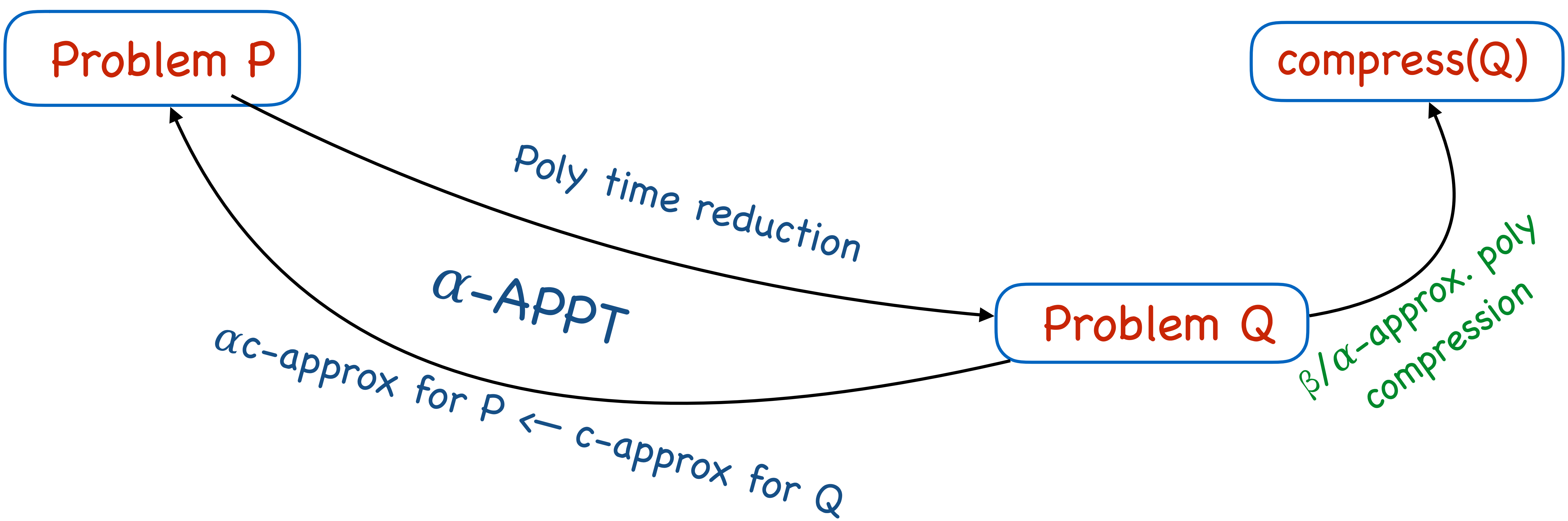
α-approximate Poly. Param. Transformations

Reductions to rule out approximate polynomial kernels



α-approximate Poly. Param. Transformations

Reductions to rule out approximate polynomial kernels



α-approximate Poly. Param. Transformations

Reductions to rule out approximate polynomial kernels

Problem P

β-approximate polynomial compression

compress(Q)

no β-approximate polynomial compression for P + α-APPT from P to Q —>
no β/α-approx. poly compression for Q

Take home message

- Many problems for which we have no polykernel results —> what about approximate kernels for these?

- Or are they like Longest Path, Set Cover (n) and don't even have an approximate kernel?

- What about Directed Feedback Vertex Set? Does it at least have an approximate polynomial kernel?

Thank you for your attention!