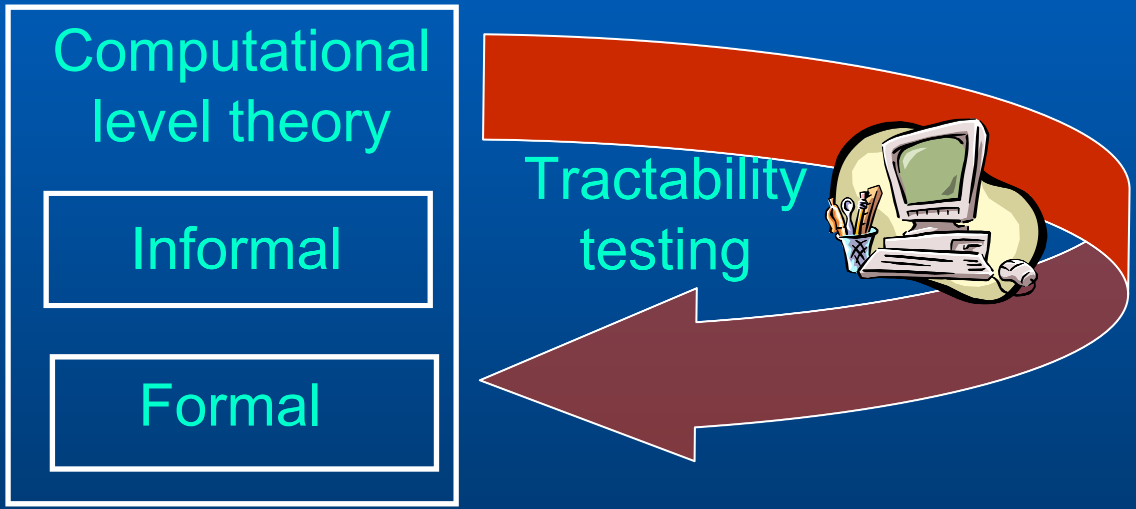
Implementing the Tractable-Design Cycle: Definitions and Techniques

Ulrike Stege

Department of Computer Science

University of Victoria

Implementing the Tractable-Design Cycle: Complexity Analysis—Definitions and Techniques



Computational level theory

Informal

Formal

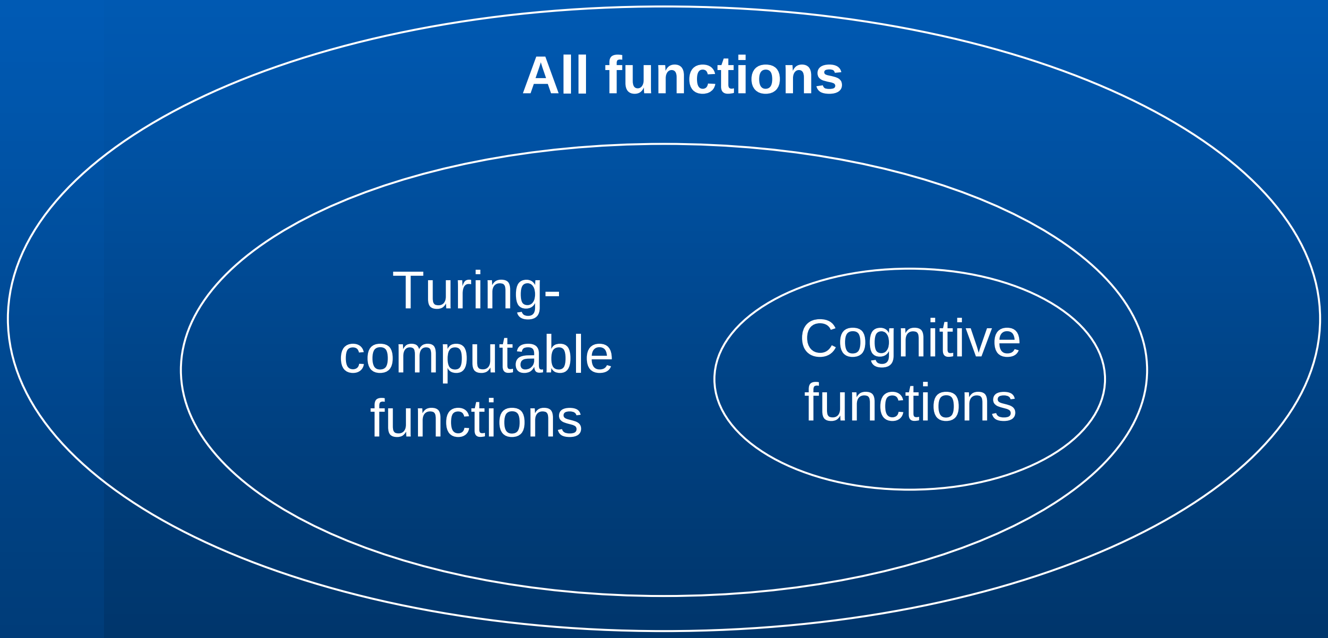
Tractability testing

Joint work with Iris van Rooij, cf. Iris' PhD thesis.

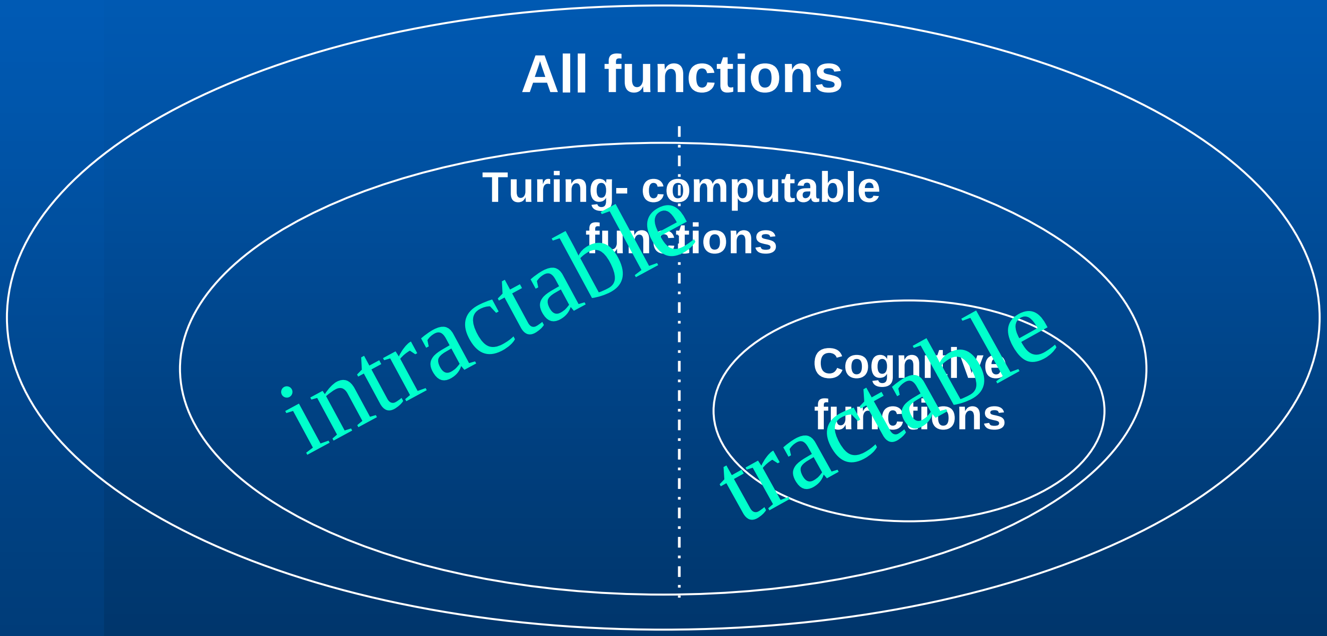
Outline

- Motivation
- Computational problems
  - Optimization Problems versus Decision Problems
- Classical Tractability vs. Classical Intractability
  - Classical Tractability and Polynomial time
  - Nondeterministic polynomial time
  - Classical Intractability: NP-hardness and NP-completeness
- Parameterized decision problems
  - Fixed-Parameter Tractability
  - Fixed-Parameter Intractability

Which functions can describe cognitive systems?



Which functions can describe cognitive systems?



Which cognitive systems are tractable?

- ***Before*** **Testing of Cognitive Theory**
  - **Formalization of the cognitive theory → *computational problem***
  - ***Tractable* or *intractable*? → Analyze complexity of computational problem**
  - **If intractable, revise cognitive theory**

Computational Problems

- Optimization Problems
- Decision Problems
- Optimization Problems versus Decision Problems

Optimization Problems

- **Maximization problems**
- **Minimization problems**

Coherence (informal)

Input: A set of interconnected beliefs.

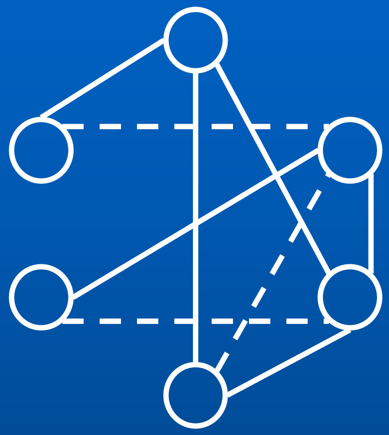
Output: A truth assignment of maximum coherence.

Optimization Problems

Coherence (informal)

Input: A set of interconnected beliefs.

Output: A truth assignment of maximum coherence.



Coherence (formal)

Input: Set of propositions P, set of constraints $C = C^{-} \cup C^{+}$.

Output: A truth assignment to the propositions in P that satisfies a maximum number of constraints. Here a constraint $(p, q) \in C^{-}$ is satisfied if p is ‘false’ and q is ‘true’, and a constraint $(p, q) \in C^{+}$ is satisfied if both p and q are ‘true’ or both p and q are ‘false’

Optimization Problems

Coherence (even more formal)

Input: Network $N = (P,C)$, where C is partitioned into $C = C- \cup C+$.

Output: A subset $A \subseteq P$ such that $|\{(p,q) \in C :$ (p,q) is satisfied$\}|$ is *maximized*.

Here, $(p,q) \in C-$ is *satisfied* if either ($p \in A$ and $q \notin A$) or ($p \notin A$ and $q \in A$), and $(p, q) \in C+$ is *satisfied* if either $p, q \in A$ or $p, q \notin A$.

Computational Problems

- Optimization Problems
- Decision Problems
- Optimization Problems versus Decision Problems

Decision Problems

- Answer: yes / no

Decision Problem

Coherence (Decision Version)

Input: $N = (P,C)$, C is partitioned into $C = C^- \cup C^+$, a positive integer k

Question: Does there exist $A \subseteq P$ such that $|\{(p,q) \in C : (p,q) \text{ is satisfied}\}| \geq k$?

Decision Problems

- Answer: yes / no
- Often answer is *constructive*: If yes, we also know a solution that is a *witness* for answer.

Computational Problems

- Optimization Problems
- Decision Problems
- Optimization Problems versus Decision Problems

Decision Problems versus Optimization Problems

- *Goal* : Determine whether the formalized (optimization) problem is tractable or intractable.
- *Complexity Theory* : set up for decision problems
- ???What about our optimization problem???

Decision Problems versus Optimization Problems

- **In classical and parameterized framework we can show**

  If decision problem is tractable, then optimization problem is tractable, and vice versa!

Why?

Decision problem is tractable ⇒ optimization problem is tractable

- We first introduce our framework and then reconsider this issue.

- Classical Complexity
  - Tractability ≅ Polynomial Time ≅ $\mathcal{P}$
  - A decision problem L is ***decidable in polynomial time*** iff for each instance $<x,k>$ it can be decided in $|x|^c$ (c **is constant**) time whether $<x,k> \in L$ or $<x,k> \notin L$.

$\boxed{L \in \mathcal{P}}$

The Polynomial Time Class $\mathcal{P}$

- **Examples:** $|x|, |x|^2, |x|^3, |x|^{81}$

n	$O(n^2)$	$O(2^n)$	$O(2^{\kappa}n)$, $\kappa = 10$
5	0.15 msec	0.19 msec	0.51 sec
20	0.04 sec	1.75 min	2.05 sec
50	0.25 sec	8.4 x 10^3 yrs	5.12 sec
100	1.00 sec	9.4 x 10^{17} yrs	10.2 sec
1000	1.67 min	7.9 x 10^{288} yrs	1.71 min

The Polynomial Time Class $\mathcal{P}$

- **How can we prove that a decision problem $L \in \mathcal{P}$?**

Decision Problems versus Optimization Problems

- **In classical and parameterized framework we can show**

  If optimization problem is tractable, then decision problem is tractable.

$$L_{\text{opt}} \in \mathcal{P} \Rightarrow L \in \mathcal{P}$$

$L_{\text{opt}} \in \mathcal{P} \Rightarrow L \in \mathcal{P}$

Decision Problems versus Optimization Problems

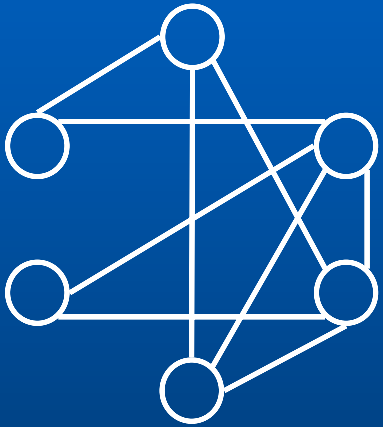
- In classical and parameterized framework we can show

If decision problem is tractable, then optimization problem is tractable!

Why?

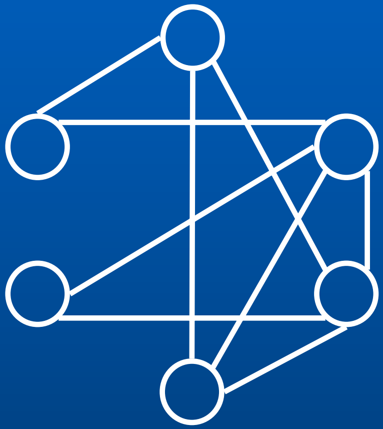
Coherence for $C = C+$

$k = 6$



$P' = ?$

Coherence for $C = C+$ (opt. Version)



$P' = ?$

$L \in \mathcal{P} \Rightarrow L_{\mathrm{opt}} \in \mathcal{P}$

- $x \in L_{\mathrm{opt}}$?
- Range of k
- $<x,k>$
- $\underbrace{(\text{Range of } k)|x|^{c}}_{\in \mathcal{P}}$

Decision Problems versus Optimization Problems

- $L \in \mathcal{P} \Leftrightarrow L_{\text{opt}} \in \mathcal{P}$

Coherence (Decision Version)

Input: $N = (P,C)$, C is partitioned into $C = C^{-} \cup C^{+}$, a positive integer k

Question: Does there exist $A \subseteq P$ such that $|\{(p,q) \in C : (p,q) \text{ is satisfied}\}| \geq k$?

- **Is Coherence $\in \mathcal{P}$?**
- **Answer: we don't really know.**
- **What do we know?**

The Polynomial Time Class $\mathcal{P}$

- **How can we prove that a decision problem** $L \in \mathcal{P}$ **?**

- **How can we prove that a decision problem** $L \notin \mathcal{P}$ **?**

The Nondeterministic Polynomial Time Class $\mathcal{NP}$

- **A decision problem L is *decidable in nondeterministic polynomial time* iff for each instance <x,k> and any solution S, it can be verified in polynomial time ($|x|^c$) if S proves that** <x,k> is a yes-instance for L.

The Nondeterministic Polynomial Time Class $\mathcal{NP}$

- For a decision problem $L \in \mathcal{NP}$ and an instance $<x,k>$ we can determine whether $<x,k>$ is a yes-instance for L or $<x,k>$ is a no-instance for L in *exponential time*.
- That is, we just have to try out each candidate solution!

Coherence $\in \mathcal{NP}$

- What is a possible solution/witness for Coherence?

- Show
  - The witness is “short”
  - We can verify in a “short” time if the witness is a correct solution.

$\mathcal{P}$ versus $\mathcal{NP}$

- **A task that we can complete fast, we can also complete slow(er).**
- **Thus:** $\mathcal{P} \subseteq \mathcal{NP}$
- **But: the converse does not necessarily hold!**

$\mathcal{P} = \mathcal{NP}$?

- Million dollar question!
- Assumption: $\mathcal{P} \neq \mathcal{NP}$
- We assume there is a decision problem L such that $L \in \mathcal{P}$ and $L \notin \mathcal{NP}$.
- We say a problem is $\mathcal{NP}$*–hard* if it is at least as hard as any problem in $\mathcal{NP}$.
- We say a problem is $\mathcal{NP}$*–complete* if it is

(1) $\mathcal{NP}$-hard and (2) also in $\mathcal{NP}$ itself.

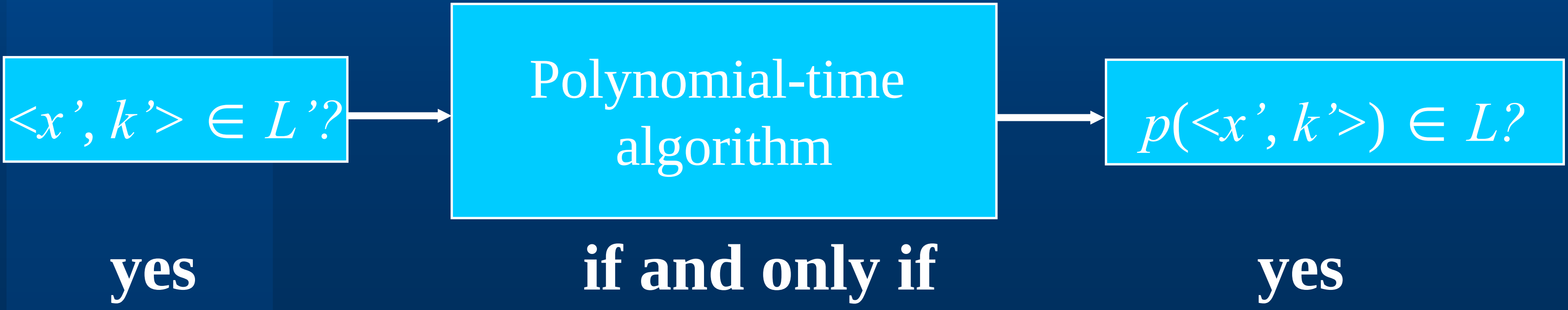
(Classically) intractable

- An $\mathcal{NP}$–hard decision problem is viewed as (classically) intractable.
- To prove that $\mathcal{P} = \mathcal{NP}$ it is enough to show that there exists an $\mathcal{NP}$–hard problem that is in $\mathcal{P}$!
- So far, nobody was able to do so …

Proving $\mathcal{NP}$-hardness

Let *L* be the problem we want to show $\mathcal{NP}$–hardness for.

- **Show that there is an $\mathcal{NP}$–hard problem *L’* that can be polynomial-time reduced to *L*.**



<*x’*, *k’*> ∈ *L’*?

Polynomial-time algorithm

p(<*x’*, *k’*>) ∈ *L*?

yes **if and only if** **yes**

Proving $\mathcal{NP}$–hardness

- If L is $\mathcal{NP}$–hard, then a polynomial-time algorithm for L would also imply a polynomial-time algorithm for L'.
- How do we find L'?
  - There is a huge catalogue of problems that are shown to be $\mathcal{NP}$–hard, just pick one that works without too much trouble.
  - Not very difficult, but experience helps.

Coherence is $\mathcal{NP}$–hard

- We reduce from a problem called Max-Cut
- Max-Cut is known to be $\mathcal{NP}$–complete [GJ'79].
- More details in demo session tomorrow.

Corollary

- Coherence is $\mathcal{NP}$-complete
  - even for $C = C$-

Is Coherence (really) intractable?

Reconsider / specify the task.

- Can we reduce the input space?
- Can we parameterize?

Reducing the input space: $C = C+$

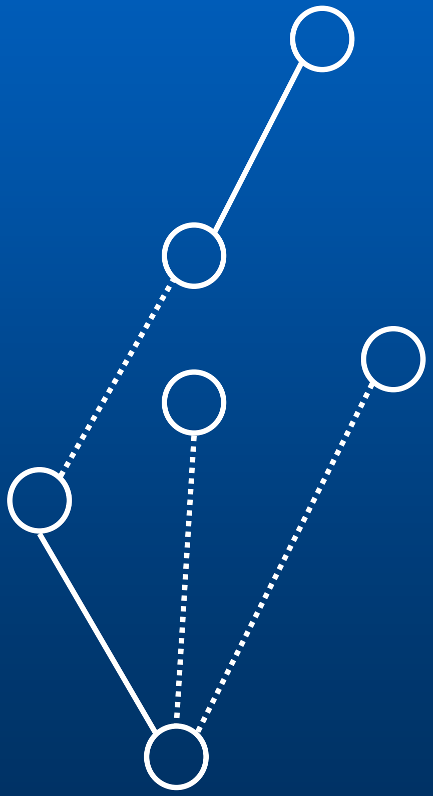
- **We can decide Coherence in polynomial time if $C = C+$.**

Reducing the input space: Is a network consistent?

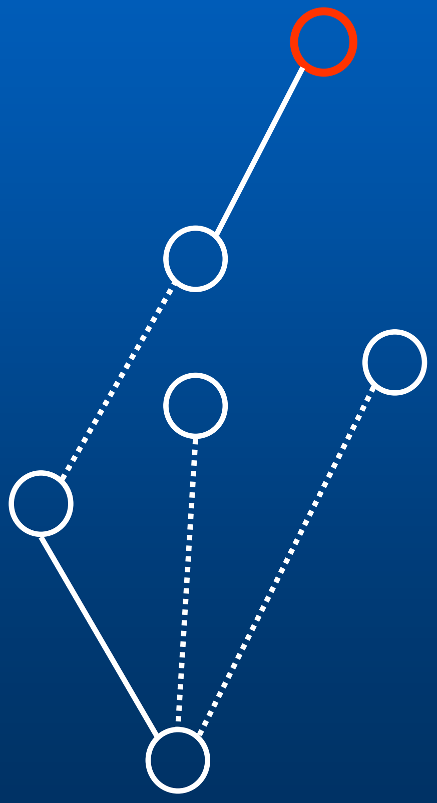
- A network is consistent if *every* edge (constrained) can be satisfied.

- We can decide in polynomial time whether or not a network $N = (P, C)$ is consistent.

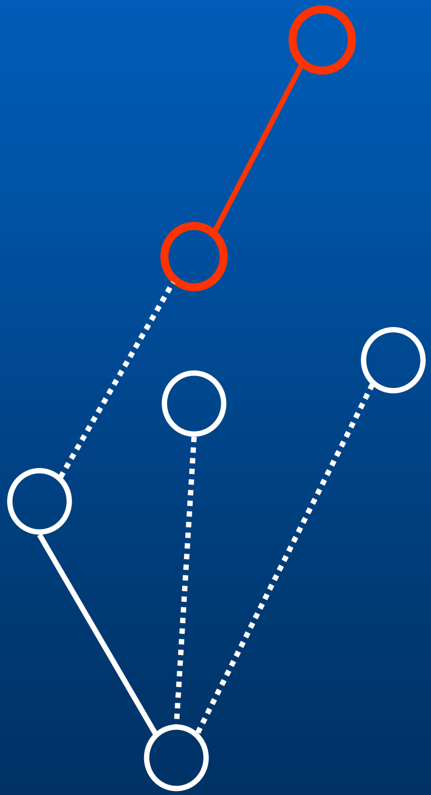
Reducing the input space: Coherence for trees



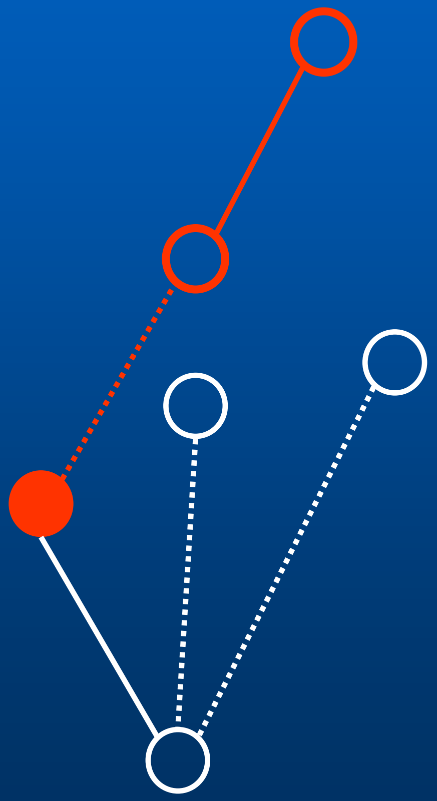
Reducing the input space: Coherence for trees



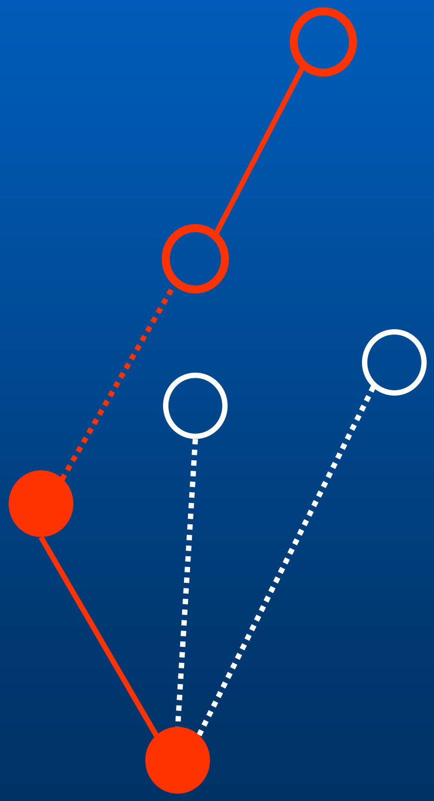
Reducing the input space: Coherence for trees



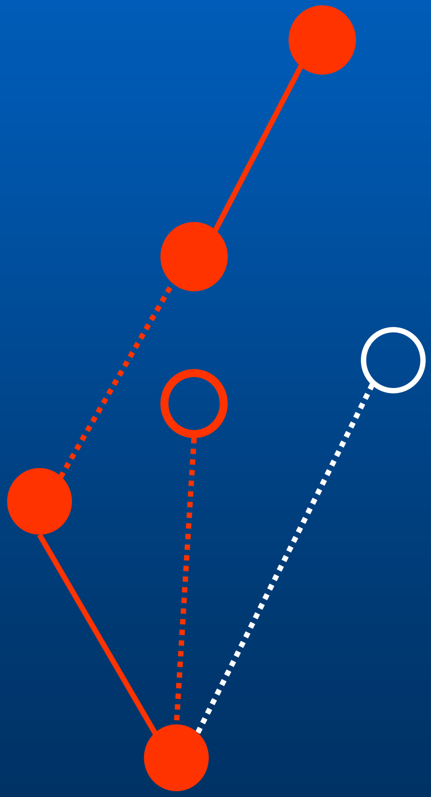
Reducing the input space: Coherence for trees



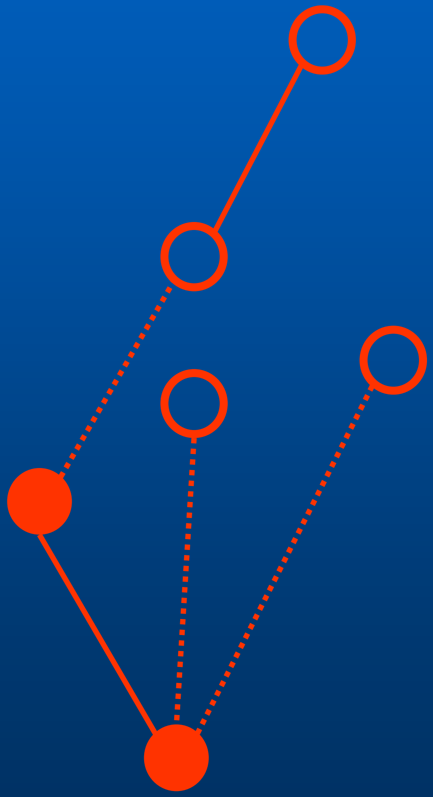
Reducing the input space: Coherence for trees



Reducing the input space: Coherence for trees

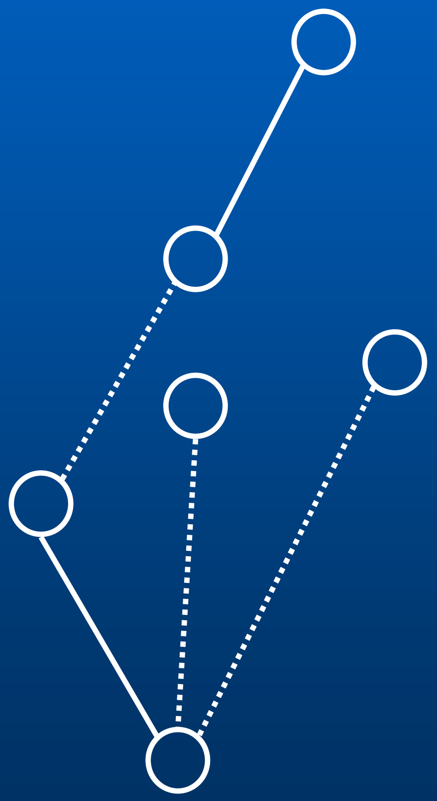


Reducing the input space: Coherence for trees



Reducing the input space: Coherence for trees

Every tree is a consistent network!



Is Coherence (really) intractable?

Reconsider / specify the task.

- Can we reduce the input space?
- Can we parameterize?

Parameterized Complexity

- **Parameterized decision problem**
- **Parameterized Complexity Classes**
  - $\mathcal{FPT}$
  - $\mathcal{W}$**[1]**
  - $\mathcal{W}$**[2]**
  - **…**

Parameterized decision problem

- **Like decision problem, but a parameter (explicit or implicit) is specified.**

k-Coherence (Parameterized Decision Version)

Input: An (inconsistent) network $N = (P,C)$, C is partitioned into $C = C^{-} \cup C^{+}$, a positive integer k

Parameter: k

Question: Does there exist $A \subseteq P$ such that $|\{(p,q) \in C : (p,q) \text{ is satisfied}\}| \geq k$?

Fixed-Parameter Tractability

A parameterized decision problem ***L*** is fixed-parameter tractable (*fpt*) if there exists a constant α and an algorithm Φ such that Φ decides if $\langle x, k \rangle$ is a yes-instance for ***L*** in time $f(k) \cdot |x|^{\alpha}$ where ***f*** is an arbitrary function of the parameter ***k*****.**

Examples for *FPT* running times

instance size |*x*| = *n*, parameter *k*

$2^k n$

$2817^{2k} + n^3$

n^{91}

$k^{k^{k^k}} + n$

Remarks

A problem that is $\mathcal{NP}$–hard or $\mathcal{NP}$–complete for can be fixed-parameter tractable for a chosen parameter!

A problem that is in $\mathcal{P}$ is fixed-parameter tractable for any chosen parameter.

k-Coherence

Input: An (inconsistent) network $N = (P,C)$, C is partitioned into $C = C^{-} \cup C^{+}$, a positive integer k

Parameter: k

Question: Does there exist $A \subseteq P$ such that $|\{(p,q) \in C : (p,q) \text{ is satisfied}\}| \geq k$?

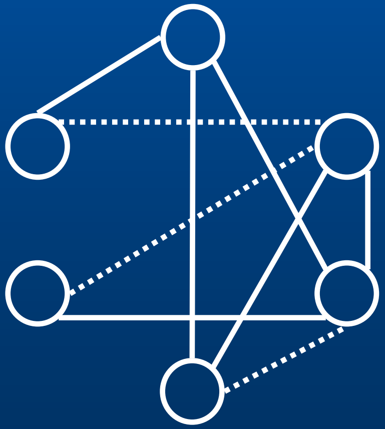
k-Coherence for connected networks is in $\mathcal{FPT}$

- **A network N is connected if for every pair of nodes there exists a path in N.**

k-Coherence for connected networks is in $\mathcal{FPT}$

Lemma. **Let <*N*, *k*> be an instance for *k*-Coherence, *N* connected. If |*P*| > *k*, then <*N*, *k*> is a yes-instance.**

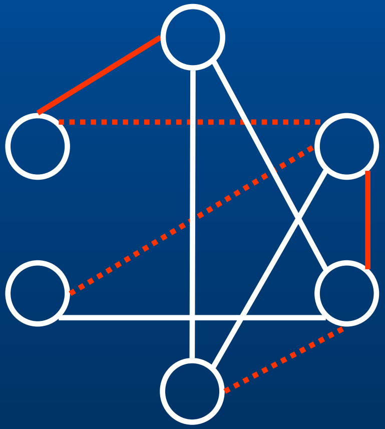
Proof.



k-Coherence for connected networks is in $\mathcal{FPT}$

Lemma. Let <*N*, *k*> be an instance for *k*-Coherence, *N* connected. If |*P*| > *k*, then <*N*, *k*> is a yes-instance.

Proof.

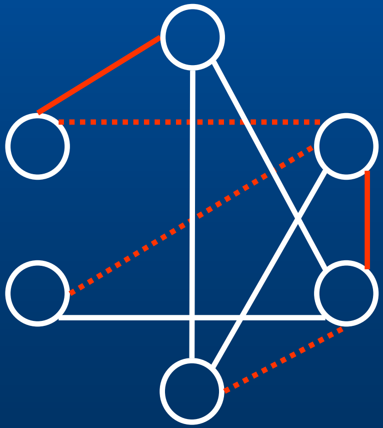


Pick a spanning subtree of *N*.

k-Coherence for connected networks is in $\mathcal{FPT}$

Lemma. Let <*N*, *k*> be an instance for *k*-Coherence, *N* connected. If |*P*| > *k*, then <*N*, *k*> is a yes-instance.

Proof.



Pick a spanning subtree of *N*.⇒ At least |*P*| edges are satisfied!

k-Coherence for connected networks is in $\mathcal{FPT}$

- $\langle N, k \rangle$ instance for k-Coherence
- N connected
- If $|P| > k$ then $\langle N, k \rangle$ is a yes-instance.
- Else $|P| \leq k$.
- How much time does it take to decide the answer for $\langle N, k \rangle$ with $|P| \leq k$?
- $|E| = ?$

k-Coherence for connected networks is in $\mathcal{FPT}$

- $\langle N, k\rangle$ **instance for** k**-Coherence**
- N **connected**
- **If** $|P| > k$ **then** $\langle N, k\rangle$ **is a yes-instance.**
- **Else** $|P| \leq k$.
- How much time does it take to decide the answer for $\langle N, k\rangle$ with $|P| \leq k$?
- $|E| \leq \binom{k}{2}$

k-Coherence for connected networks is in $\mathcal{FPT}$

- N is bounded in size in a function of k
  - Problem Kernel
- Even if we have to try out every possible solution, the number of those is still a function of k
- We can answer in fixed-parameter-tractable time for parameter k

Proving *FPT*-Membership

- Give an algorithm
  - Problem Kernel / Kernelization
  - Bounded Search Tree
- Prove existence of a problem kernel
  - Boundary Lemma
- Graph Minor Theorem
- More details in demo session tomorrow.

How do we show that a (parameterized decision) problem is parameterized intractable?

- **Prove that the problem is hard for class $\mathcal{W}$[1] or class $\mathcal{W}$[2] or ...**
- **Prove that: if the problem is in $\mathcal{FPT}$, then $\mathcal{P} = \mathcal{NP}$.**

$\mathcal{W}[1]$

- $\mathcal{FPT} \subseteq \mathcal{W}$**[1]**
- **Conjecture:** $\mathcal{FPT} \neq \mathcal{W}$**[1]**
- **Hard for** $\mathcal{W}$**[1]**
  - **Problems that are likely not fixed-parameter tractable**
  - **Running times are something like** n^k

Prove that a (parameterized) problem is hard for class $\mathcal{W}[1]$

- Via parameterized reduction from a problem that is known to be hard for $\mathcal{W}[1]$ and that further preserves the parameter.
- Similar idea as in $\mathcal{NP}$–hardness proofs.
  - ☺ Time permitted is in $\mathcal{FPT}$ (any function in parameter is allowed, rest polynomial)
  - ☹ Parameter has to be preserved!

☺ Many of the “classic” $\mathcal{NP}$–hardness reductions in the literature are already parameterized.

Summary

- We investigated techniques from computer science to prove (in)tractability for decision problems and optimization problems.
- We also observed: If a special case of a decision problem is ($\mathcal{NP}$-)hard, then the problem itself is ($\mathcal{NP}$-)hard itself.
- Further: If we can prove that a problem is tractable, then its special cases are tractable as well.

Demo Session

Ulrike Stege (University of Victoria)

Iris van Rooij (TU Eindhoven)

Topics

- $\mathcal{NP}$–completeness proofs
  - Membership
  - Polynomial-time reduction
- $\mathcal{FPT}$–algorithms
  - Technique of building a problem kernel
  - Technique of bounded a search tree
  - combination

Topics

- $\mathcal{FPT}$–membership
  - Existence of a problem kernel
  - Graph Minor Theorem
- Parameterized intractability
  - Parameterized reduction
  - Not in $\mathcal{FPT}$ unless $\mathcal{P} = \mathcal{NP}$
  - Membership in $\mathcal{W}[1]$

Coherence (Decision Version)

Input: An (inconsistent) network $N = (P,C)$, C is partitioned into $C = C^{-} \cup C^{+}$, a positive integer k

Question: Does there exist $A \subseteq P$ such that $|\{(p,q) \in C : (p,q) \text{ is satisfied}\}| \geq k$?

Coherence is $\mathcal{NP}$–hard

We reduce from

Max-Cut (decision version)

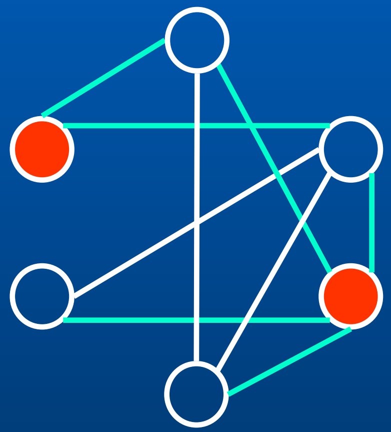
Input : A graph $G = (V, E)$. A positive integer m.
Question : Does there exist a partition of V into sets A and R such that $\left|\{(u,v) \in E : u \in A, v \in R\}\right| \geq m$?

- Max-Cut is known to be $\mathcal{NP}$–complete [GJ'79].

Max Cut (Example)

$m = 4$

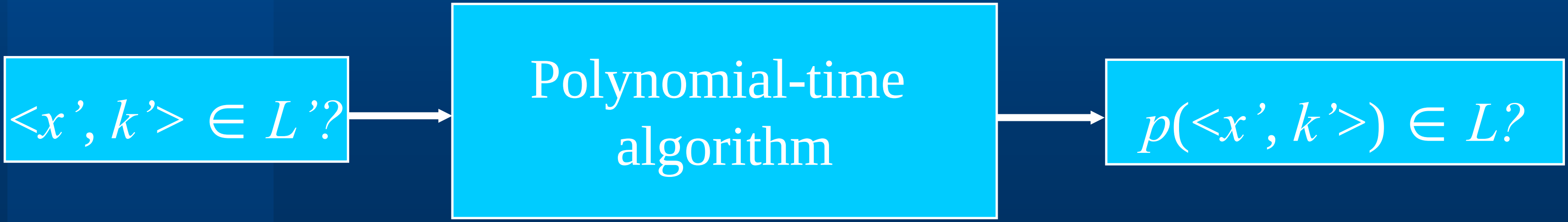
A R



Proving $\mathcal{NP}$-hardness

Let *L* be the problem we want to show $\mathcal{NP}$–hardness for.

- Show that there is an $\mathcal{NP}$–hard problem *L'* that can be **polynomial-time reduced to** *L*.



$<x', k'> \in L'?$

Polynomial-time algorithm

$p(<x', k'>) \in L?$

yes if and only if yes

Coherence is $\mathcal{NP}$-hard

- Let (G,m) be an instance for Max Cut.
- We define an instance <N,k> for Coherence as follows.
  - $P = V$

$$\left.\begin{array}{l} - \; C = E \\ - \; C- = E \end{array}\right\} \; N = G$$

  - $k = m$
- We still have to prove

<G,m> is a yes-instance for Max-Cut if and only if <N,k> is a yes-instance for Coherence

$<G,m>$ is a yes-instance for Max-Cut if and only if $<N,k>$ is a yes-instance for Coherence.

"$\Rightarrow$"

- Since $<G,m>$ is a yes-instance for Max-Cut, we can assume V be partitioned into A and R. Further let $\left|\{(u,v) \in E : u \in A, v \in R\}\right| \geq m$.
- We show A is a solution for N. Consider an edge $e \in \{(u,v) \in E : u \in A, v \in R\}$. Edge e is satisfied!
- There are at least $m = k$ many of those edges!

(*G*,*m*) is a yes-instance for Max-Cut if and only if (*N*,*k*) is a yes-instance for Coherence.

"$\Leftarrow$"

- **Let *P'* be a solution for *N*.**
- **We define a partition *A* = *P'*, *R*=*V*-*P'* for *G*. Let *e*** be satisfied in *N*. Then $e \in \{(u,v) \in E : u \in A, v \in R\}$. **Then**

$$\left|\{(u,v) \in E : u \in A, v \in R\}\right| \geq p = m$$

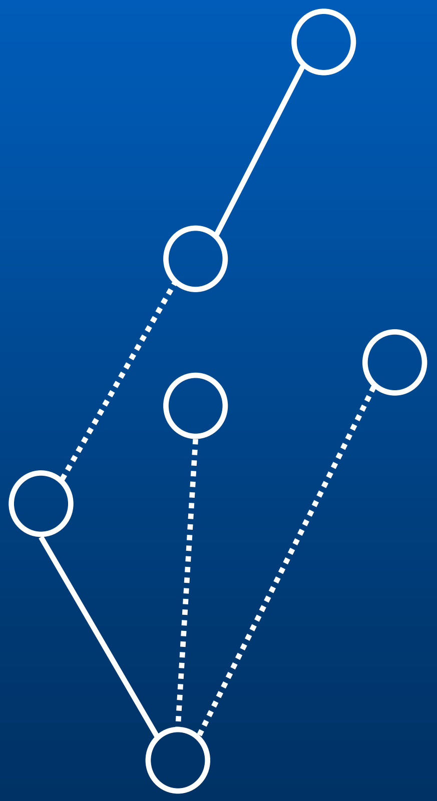
Corollary

- Coherence is $\mathcal{NP}$–complete
  - even for $C = C$-

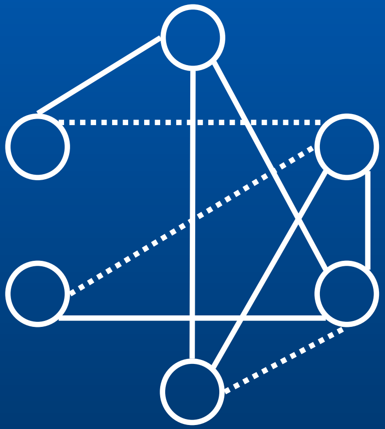
To determine whether a network is consistent is in $\mathcal{P}$.

Coherence for trees

Every tree is a consistent network!



To determine whether a network is consistent is in $\mathcal{P}$.



Technique of Problem Kernel

k-Coherence (Parameterized Decision Version)

Input: An (inconsistent) network $N = (P,C)$, C is partitioned into $C = C^{-} \cup C^{+}$, a positive integer k

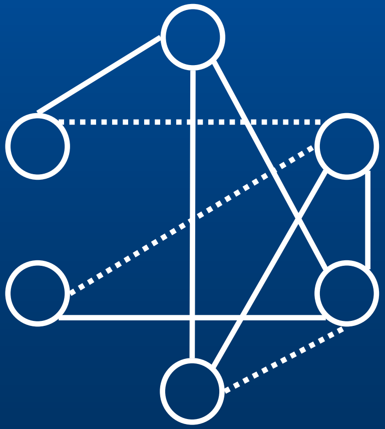
Parameter: k

Question: Does there exist $P' \subseteq P$ such that $|\{(p,q) \in C : (p,q) \text{ is satisfied}\}| \geq k$?

k-Coherence for connected networks is in $\mathcal{FPT}$

Lemma. Let <N, k> be an instance for k-Coherence, N connected. If $|P| > k$, then <N, k> is a yes-instance.

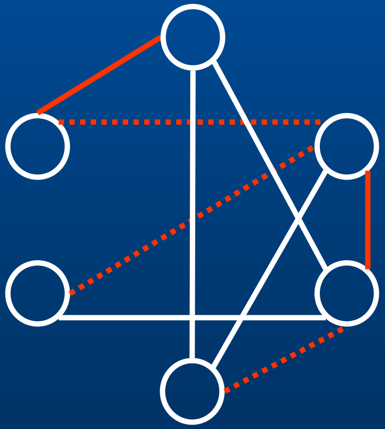
Proof.



k-Coherence for connected networks is in $\mathcal{FPT}$

Lemma. **Let <N, k> be an instance for k-Coherence, N connected. If |P| > k, then <N, k> is a yes-instance.**

Proof.

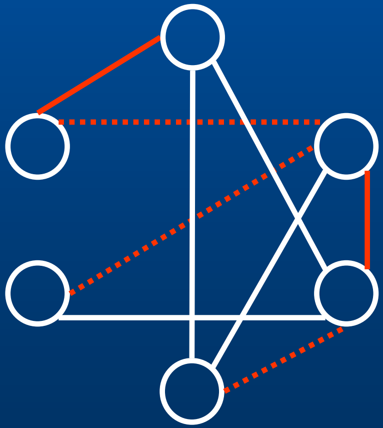


Pick a spanning subtree of N.

k-Coherence for connected networks is in $\mathcal{FPT}$

Lemma. Let $\langle N, k \rangle$ be an instance for k-Coherence, N connected. If $|P| > k$, then $\langle N, k \rangle$ is a yes-instance.

Proof.



Pick a spanning subtree of N.$\Rightarrow$ At least $|P|$ edges are satisfied!

k-Coherence for connected networks is in $\mathcal{FPT}$

- $<N, k>$ **instance for** k**-Coherence**
- N **connected**
- **If** $|P| > k$ **then** $<N, k>$ **is a yes-instance.**
- **Else** $|P| \leq k$.
- How much time does it take to decide the answer for $<N, k>$ with $|P| \leq k$?
- $|E|$ = ?

k-Coherence for connected networks is in $\mathcal{FPT}$

- $<N, k>$ **instance for k-Coherence**
- N **connected**
- **If $|P| > k$ then $<N, k>$ is a yes-instance.**
- **Else $|P| \leq k$.**
- How much time does it take to decide the answer for $<N, k>$ with $|P| \leq k$?
- $|E| \leq \binom{k}{2}$

k-Coherence for connected networks is in $\mathcal{FPT}$

- N is bounded in size in a function of k
  - N is a problem Kernel
- Even if we have to try out every possible solution, the number of those is still a function of k
- We can answer in fixed-parameter-tractable time for parameter k

k-Coherence for connected networks is in $\mathcal{FPT}$

- *N* is bounded in size in a function of ***k***
  - *N* is a problem Kernel
- However: Often this is just the 1st step of an *fpt*-algorithm.

Technique of bounded search trees

- **We show using this technique that the problem |C_-|-Coherence is in $\mathcal{FPT}$.**

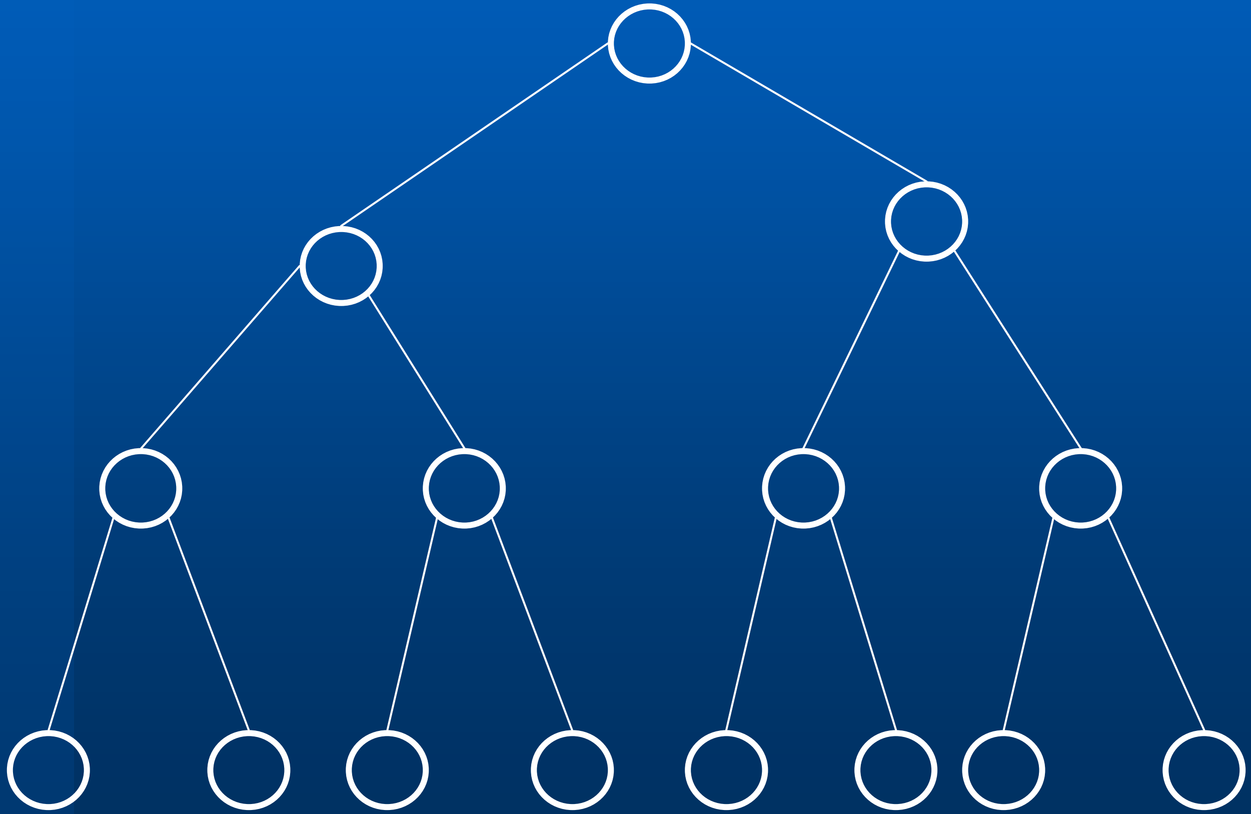
$|C-|$-Coherence (Parameterized Decision Version)

Input: An (inconsistent) network $N = (P,C)$, C is partitioned into $C = C- \cup C+$, a positive integer k

Parameter: $|C-|$

Question: Does there exist $P' \subseteq P$ such that $|\{(p,q) \in C : (p,q)$ is satisfied$\}| \geq k$?

Technique of bounded search trees – smart exhaustive search



$|C-|$-Coherence

Input: An (inconsistent) network $N = (P,C)$, C is partitioned into $C = C- \cup C+$, a positive integer k

Parameter: $|C-|$

Question: Does there exist $P' \subseteq P$ such that $|\{(p,q) \in C : (p,q)$ is satisfied$\}| \geq k$?

$|C_-|$-Coherence is in *FPT*

- **Generalize $|C_-|$-Coherence to $|C_-|$-Annotated Coherence**
- **Apply technique of bounded search tree to $|C_-|$-Annotated Coherence**

Generalization of $|C-|$-Coherence

$|C-|$-Annotated Coherence

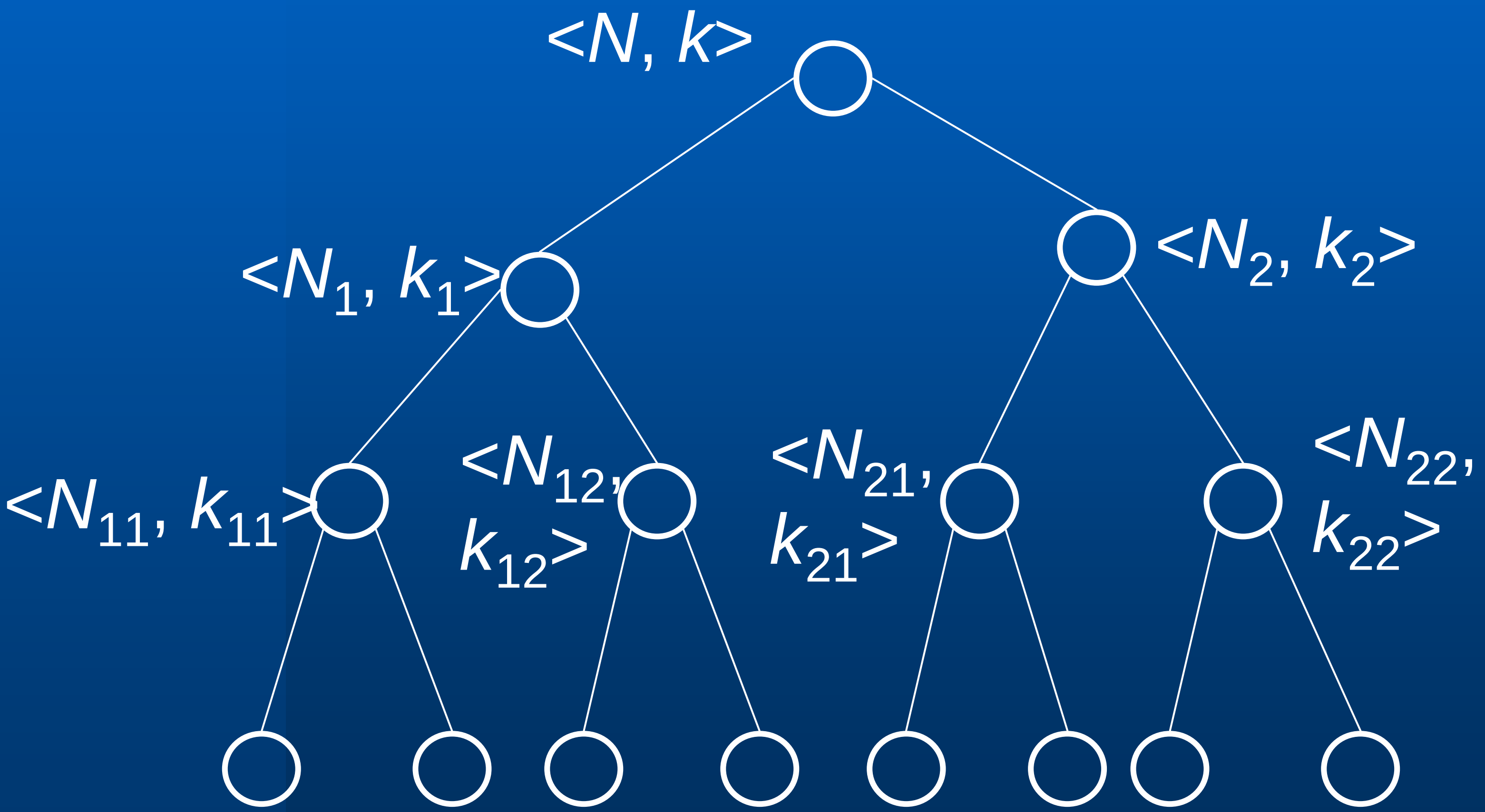
Input: A network *N* = (*P*, *C*). Here, *P* is partitioned into *U**, *P**, and *R**, and *C* is partitioned into *C*+ and *C*-. A positive integer *k*.

Parameter: $|C-|$

Question: Does there exist a partition of *P* into *P'* and *R* such that $P^* \subseteq P'$, $R' \subseteq R$, and at least *k* edges are satisfied by *A* and *R*?

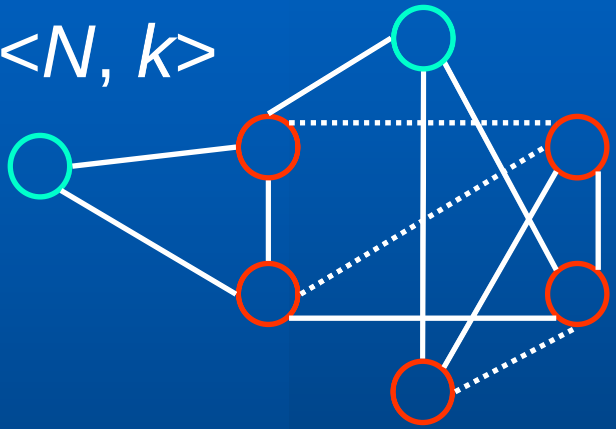
|*C*–|-Coherence is a special case of |*C*–|-Annotated Coherence

An *FPT*-Algorithm for $|C_-|$-Annotated Coherence



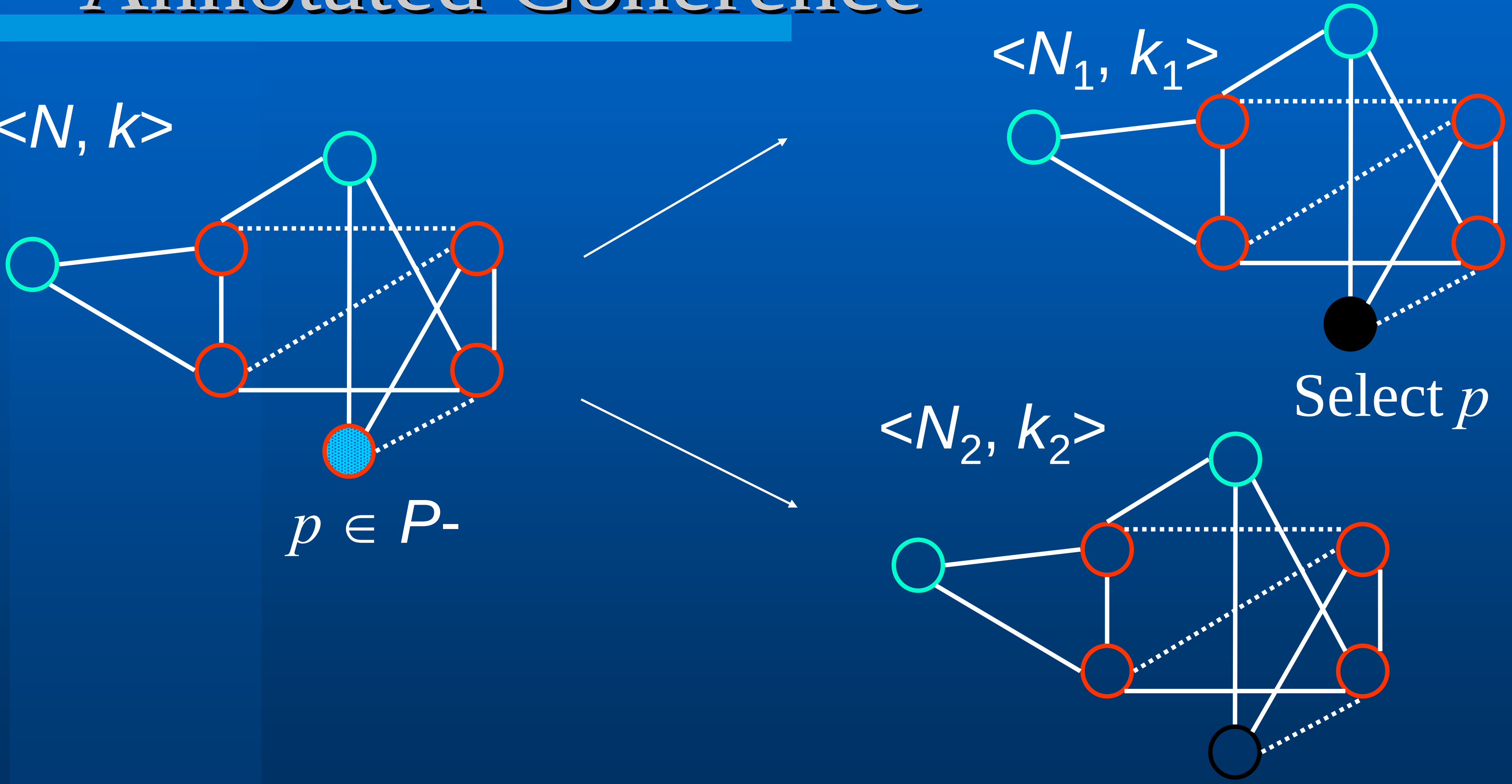
An *FPT*-Algorithm for |*C*–|-Annotated Coherence

<*N*, *k*>



$P = P- \cup P+$

An *FPT*-Algorithm for |*C*–|-Annotated Coherence



<*N*, *k*>

p ∈ *P*-

<N_1, k_1>

Select *p*

<N_2, k_2>

Do *not* select *p*

(P-)-Element-decision Branching-Rule

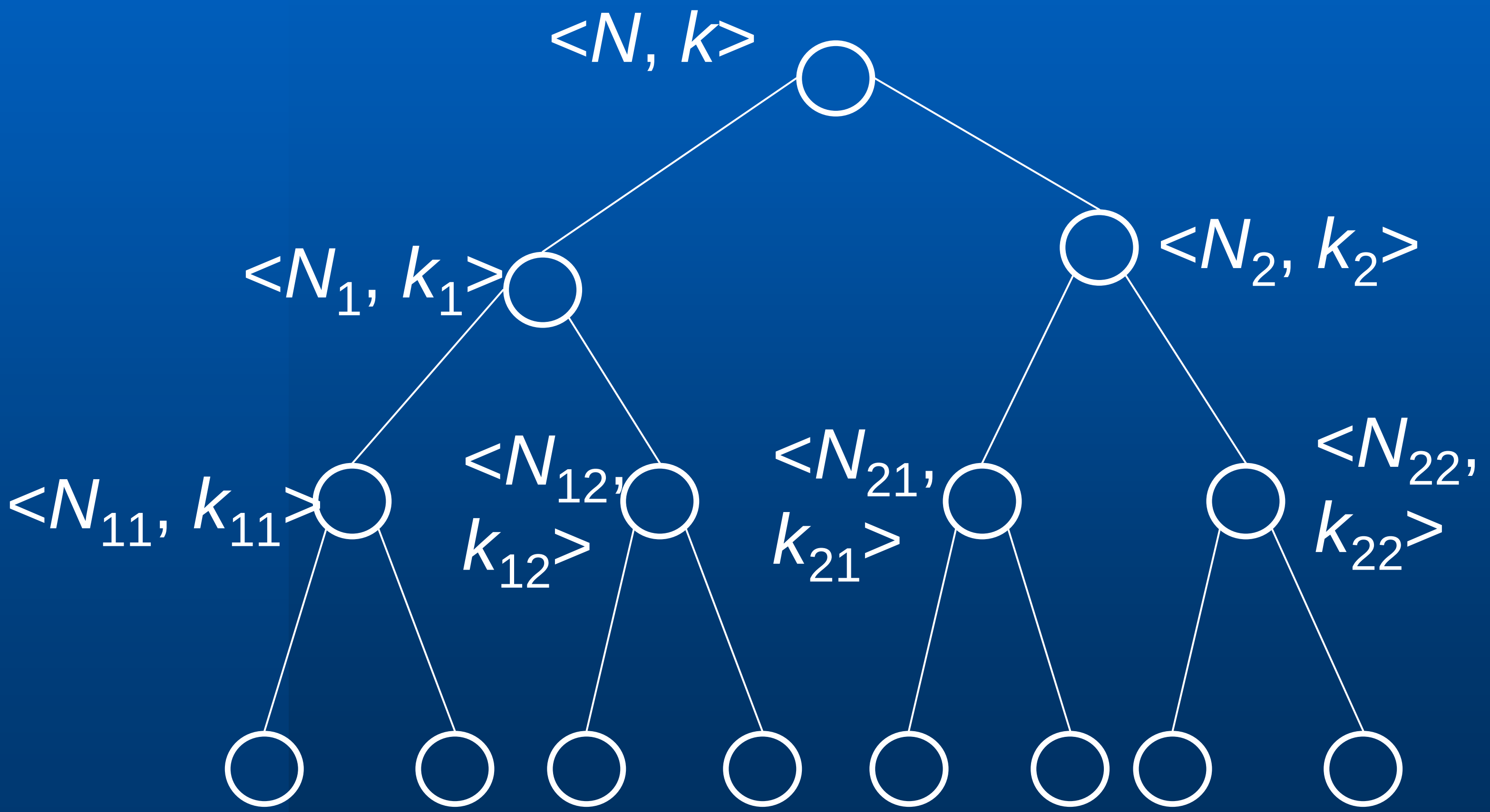
$<N, k>$: instance for $|C\text{-}|$-Annotated Coherence

- $N = (P, C)$
- $P = U' \cup P' \cup R'$, and
- $P = P\text{-} \cup P\text{+}$.
- Let $p \in U' \cap P\text{-}$.

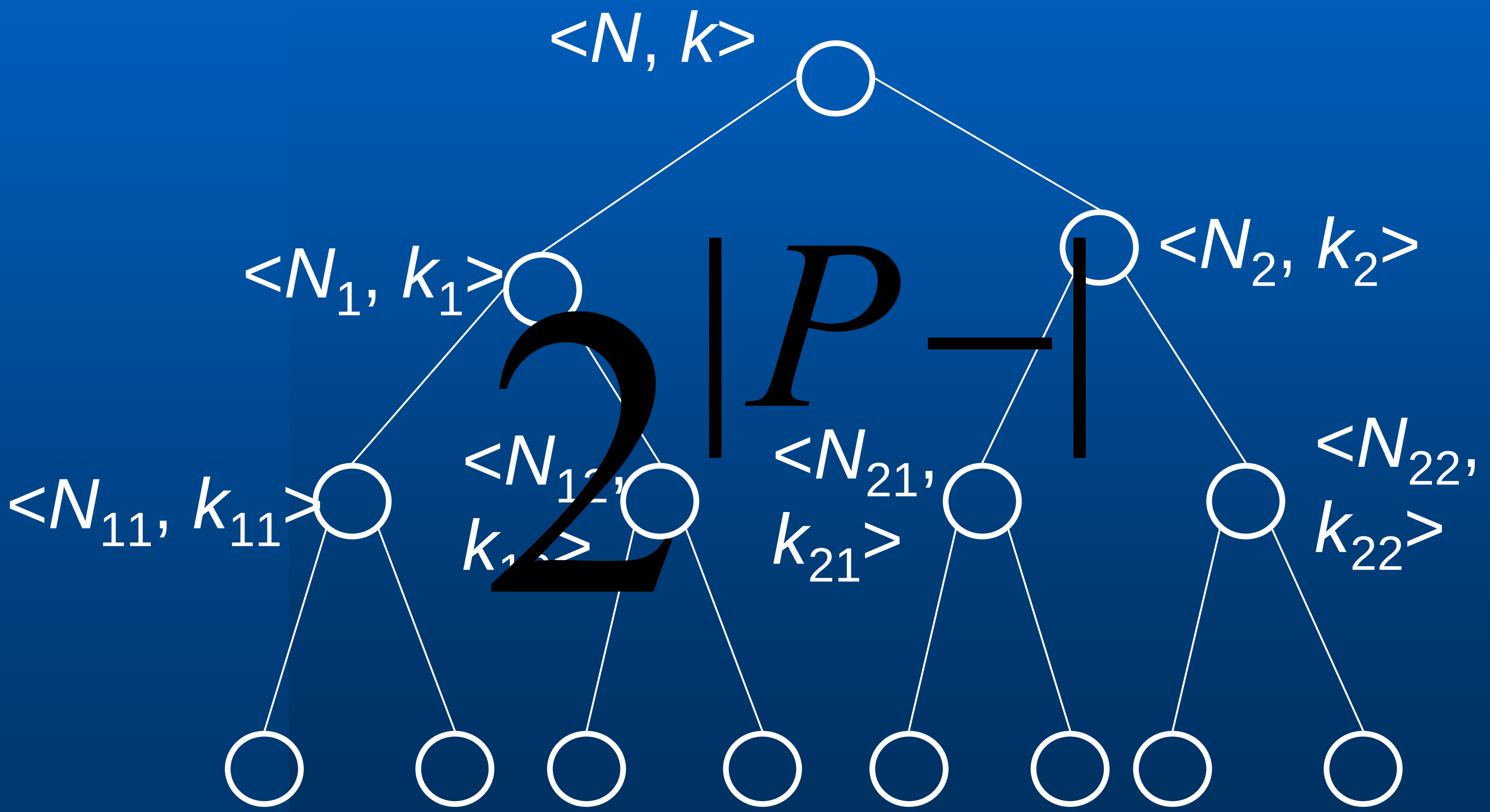
Create in the search tree two children of $<N, k>$.

- $\mathbf{<N_1, k_1>}$: $N_1 = (P_1, C_1)$ with $U_1' = U' \backslash \{p\}$, $P_1' = P' \cup \{p\}$, $R_1' = R'$, $k_1 = k$
- $\mathbf{<N_2, k_2>}$: $N_2 = (P_2, C_2)$ with $U_2' = U' \backslash \{p\}$, $P_2' = P'$, $R_2' = R' \cup \{p\}$, $k_2 = k$.

How big is the search tree after applying the reduction rule as often as possible?



How big is the search tree after applying the reduction rule as often as possible?



$$2^{|P-|}$$

An *FPT*-Algorithm for $|C–|$-Annotated Coherence

- $2^{|P\text{-}|} \leq$?
- $|P\text{-}| \leq 2|E\text{-}|$
- $2^{|P\text{-}|} \leq 2^{2|E\text{-}|}$
- Running time so far: $2^{2|E\text{-}|}|N|$
- If not solved: How does an instance look like after this branching process?

An *FPT*-Algorithm for $|C{-}|$-Annotated Coherence

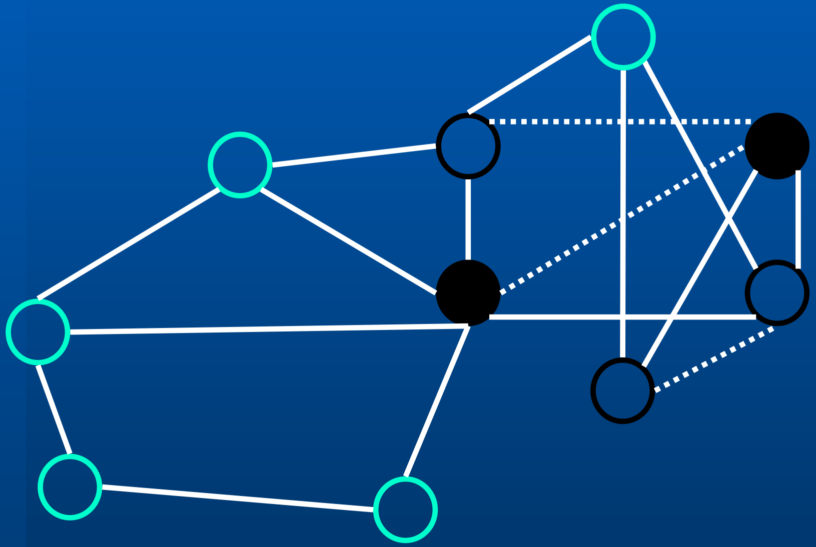
- In N are only vertices from $P+$ left! That means we are left with only positive undecided constraints.
- we can clean up the decided constraints, i.e. we remove them from the network
- Afterwards we can also remove the isolated vertices

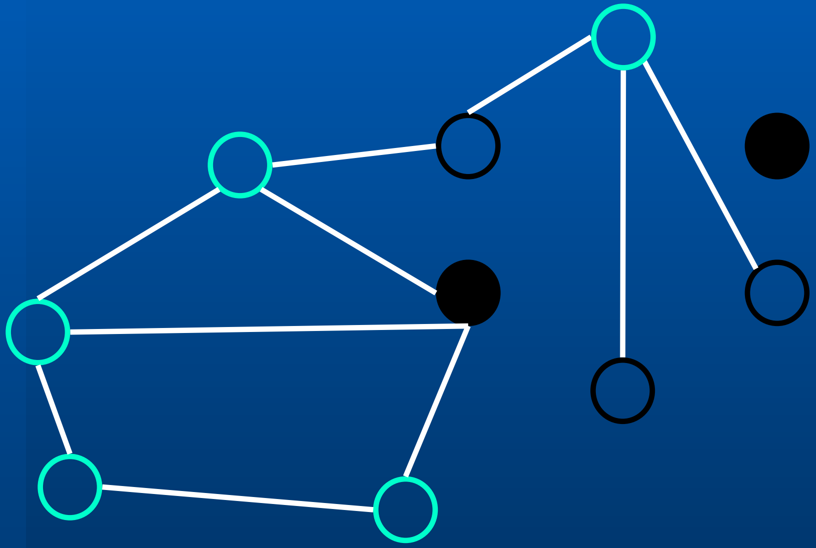
$|C-|$-Annotated Coherence with $P = P+$

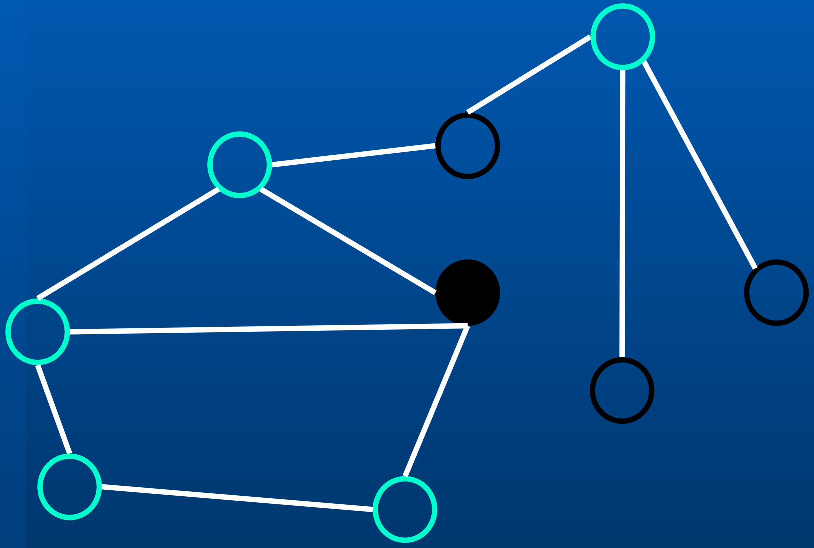
Input: A network $N = (P, C)$. Here, P is partitioned into U^*, P^*, and R^*, and $C=C+$. A positive integer k.

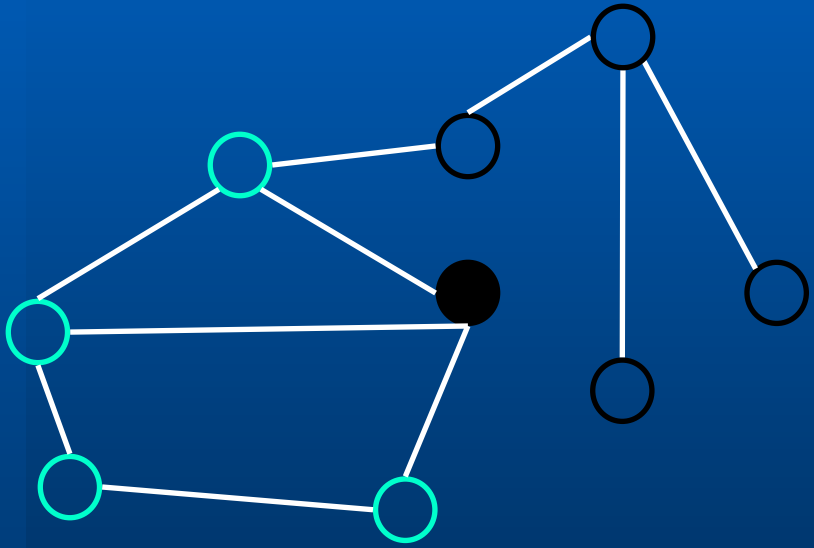
Parameter: $|C-|$

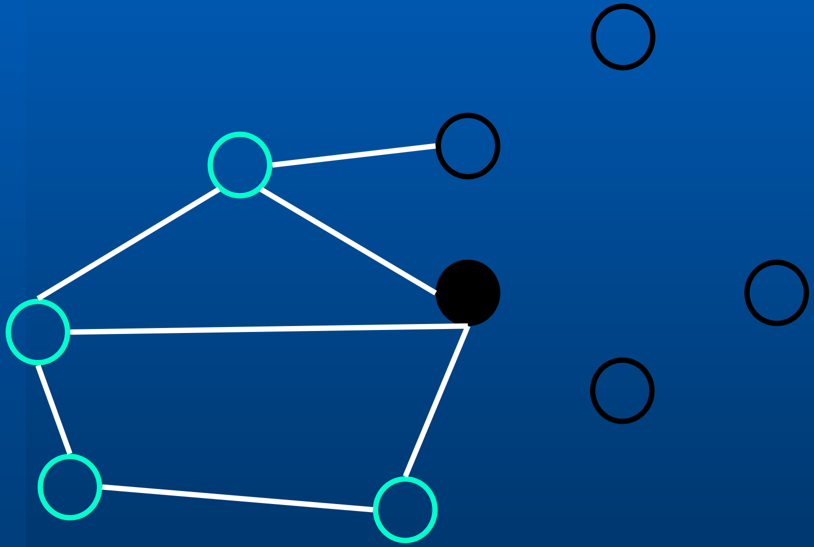
Question: Does there exist a partition of P into P' and R such that $P^* \subseteq P'$, $R' \subseteq R$, and at least k edges are satisfied by A and R?

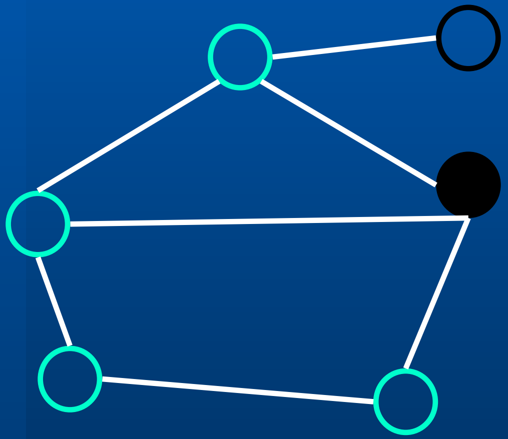






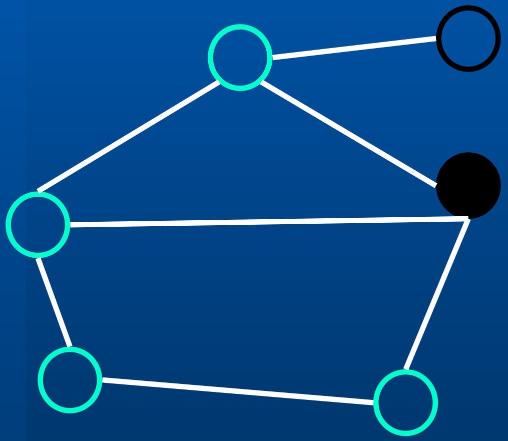






Reduction Rule

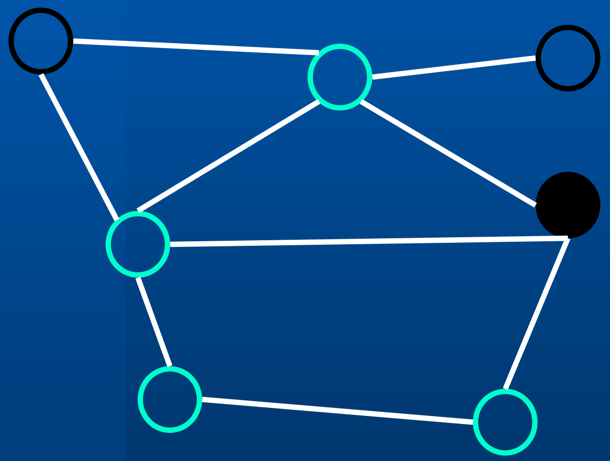
- Let p be a vertex where all neighbors are already selected. If $|N(v) \cap A^*| > |N(v) \cap R^*|$ then accept p, else reject p.

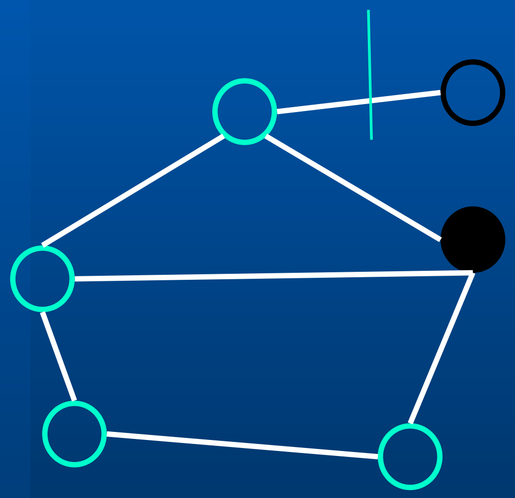


If all nodes that are selected are in A^*, then $P = A$.

If all nodes that are selected are in R^*, then $P = R$.

Otherwise network is inconsistent.





Corollary: $|C+|$-Coherence $\notin \mathcal{FPT}$ (unless $\mathcal{P} = \mathcal{NP}$)

Parameterized Reduction