Improved Inapproximability for TSP

The Role of Bounded Occurrence CSPs

Michael Lampis
KTH Royal Institute of Technology

January 22, 2013

The Story

> Good research involves good storytelling
>
> Mike Fellows

The Story

- The Traveling Salesman problem is famous and important. Unfortunately, it's NP-hard.
  - How well can we approximate it?
  - Big breakthroughs in algorithms recently. We set out to improve on inapproximability results.

The Story

- The Traveling Salesman problem is famous and important. Unfortunately, it's NP-hard.
  - How well can we approximate it?
  - Big breakthroughs in algorithms recently. We set out to improve on inapproximability results.

Main idea

- Hardness obtained through a reduction from a Constraint Satisfaction Problem (CSP)

The Story

- The Traveling Salesman problem is famous and important. Unfortunately, it's NP-hard.
  - How well can we approximate it?
  - Big breakthroughs in algorithms recently. We set out to improve on inapproximability results.

Main idea

- Reduction is easier if CSP has bounded # of occurrences

The Story

- The Traveling Salesman problem is famous and important. Unfortunately, it's NP-hard.
  - How well can we approximate it?
  - Big breakthroughs in algorithms recently. We set out to improve on inapproximability results.

Main idea

- We need inapproximability results for CSPs with bounded # of occurrences

The Story

- The Traveling Salesman problem is famous and important. Unfortunately, it's NP-hard.
  - How well can we approximate it?
  - Big breakthroughs in algorithms recently. We set out to improve on inapproximability results.

Main idea

- Such results use expander graphs

The Story

- The Traveling Salesman problem is famous and important. Unfortunately, it's NP-hard.
  - How well can we approximate it?
  - Big breakthroughs in algorithms recently. We set out to improve on inapproximability results.

Main idea

- Good expanders $\rightarrow$
  $\rightarrow$Hardness for bounded occurrence CSPs $\rightarrow$
  $\rightarrow$Hardness for TSP

The Actual Story

Better Expanders

- A local improvement argument gives (slightly) better expander graphs than those already in the literature!

TSP inapproximability

- A reduction from a 5-occurrence CSP gives a better inapproximability constant!

The Actual Story

Better Expanders

- A local improvement argument gives (slightly) better expander graphs than those already in the literature!

TSP inapproximability

- A reduction from a 5-occurrence CSP gives a better inapproximability constant!

The Actual Story

Better Expanders

- A local improvement argument gives (slightly) better expander graphs than those already in the literature! ✓

TSP inapproximability

- A reduction from a 5-occurrence CSP gives a better inapproximability constant! ✓

The Actual Story

Better Expanders

- A local improvement argument gives (slightly) better expander graphs than those already in the literature!

TSP inapproximability

- A reduction from a 5-occurrence CSP gives a better inapproximability constant!

The catch:

The Actual Story

Better Expanders

- A local improvement argument gives (slightly) better expander graphs than those already in the literature!

TSP inapproximability

- A reduction from a 5-occurrence CSP gives a better inapproximability constant!

The catch:

The reduction does not use the new expanders! Instead we rely on an amplifier construction by Berman and Karpinski.

The Traveling Salesman Problem

The Traveling Salesman Problem

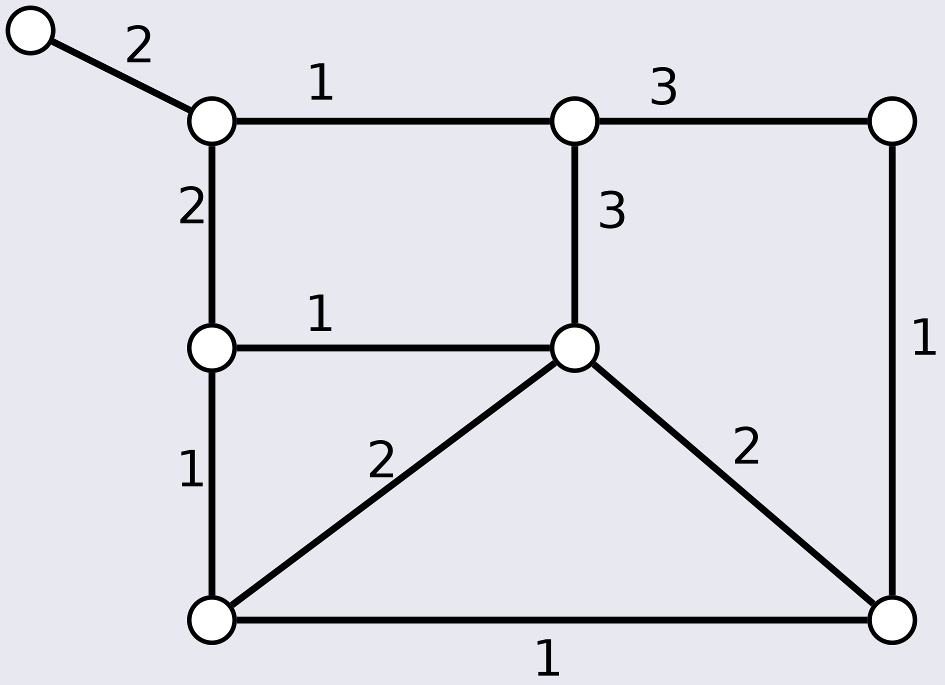
Input:

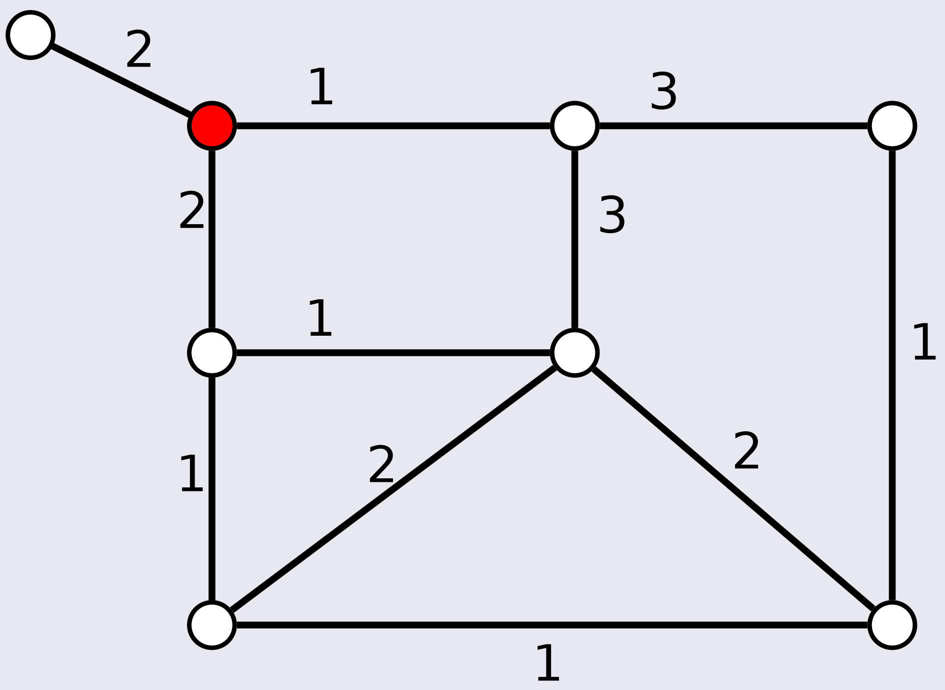
- An edge-weighted graph $G(V, E)$

Objective:

- Find an ordering of the vertices $v_1, v_2, \dots, v_n$ such that $d(v_1, v_2) + d(v_2, v_3) + \dots + d(v_n, v_1)$ is minimized.
- $d(v_i, v_j)$ is the shortest-path distance of v_i, v_j on G

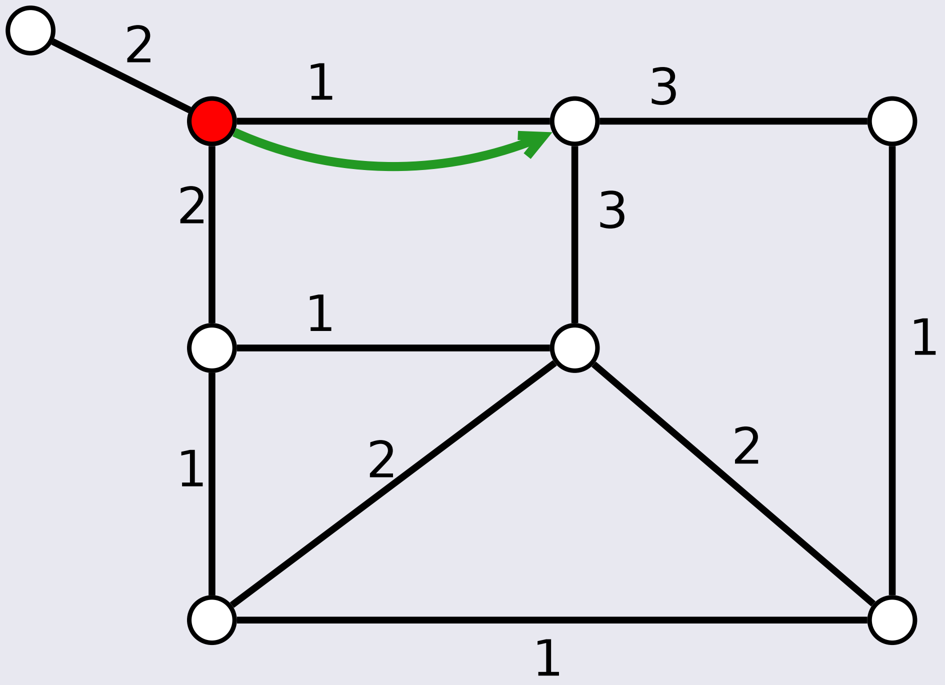
The Traveling Salesman Problem



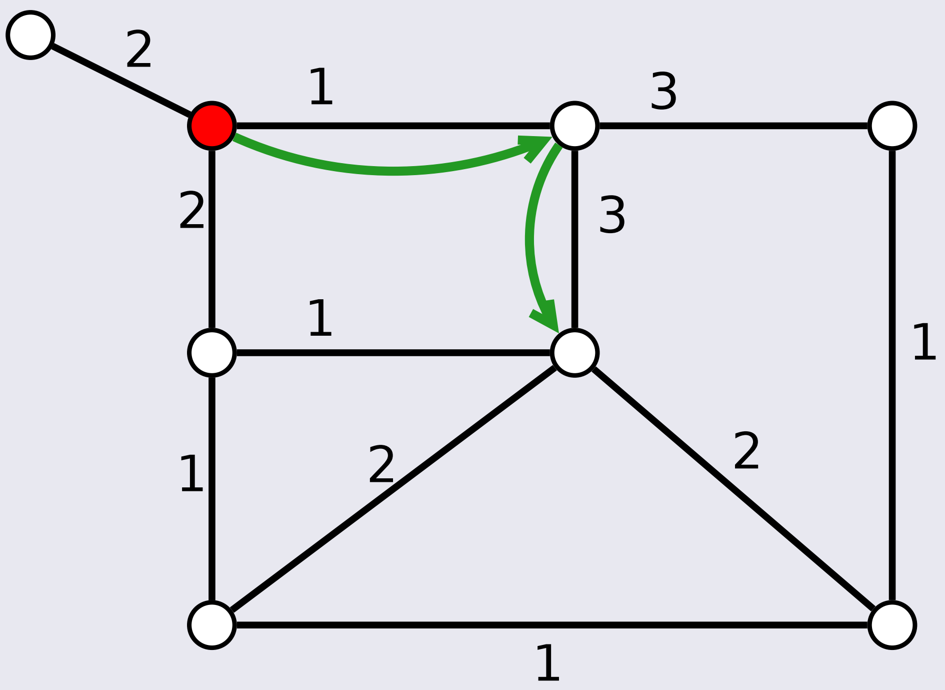

2
1
3
2
3
1
1
1
2
2
1


ACME
INSURANCE

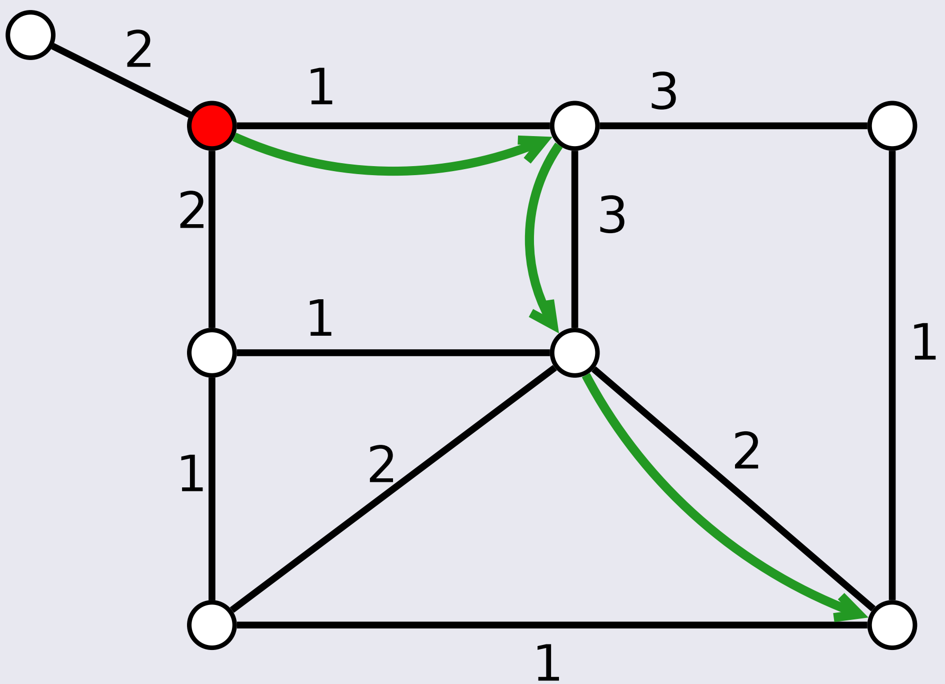
The Traveling Salesman Problem



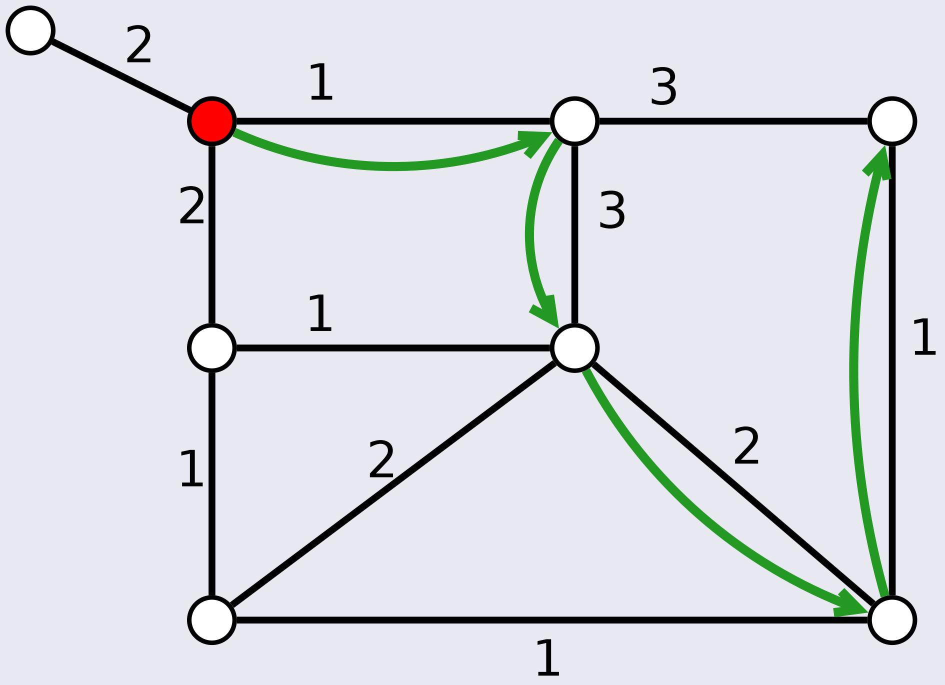
The Traveling Salesman Problem

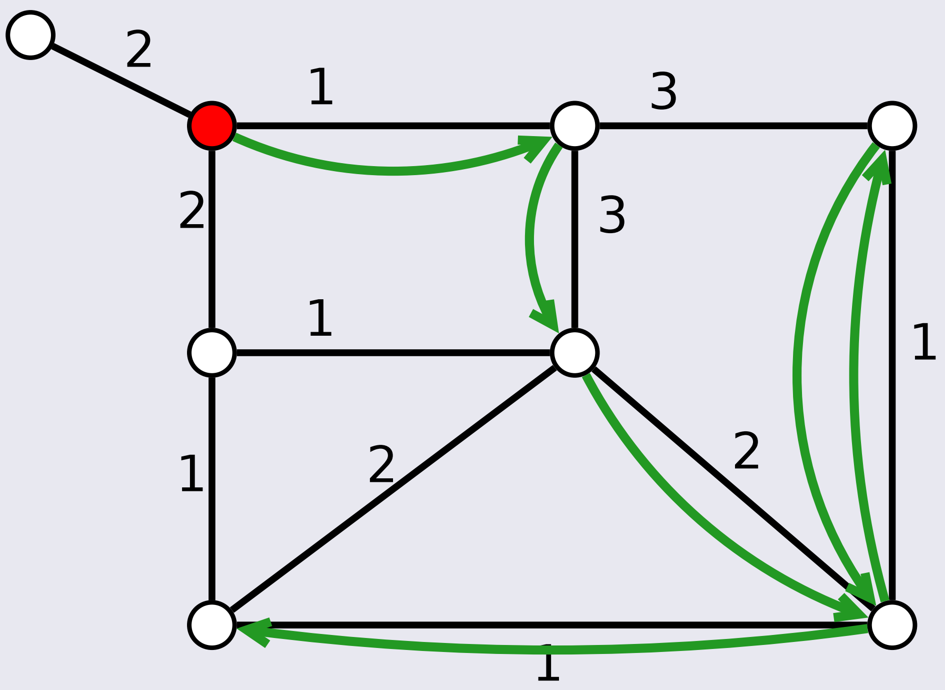


The Traveling Salesman Problem



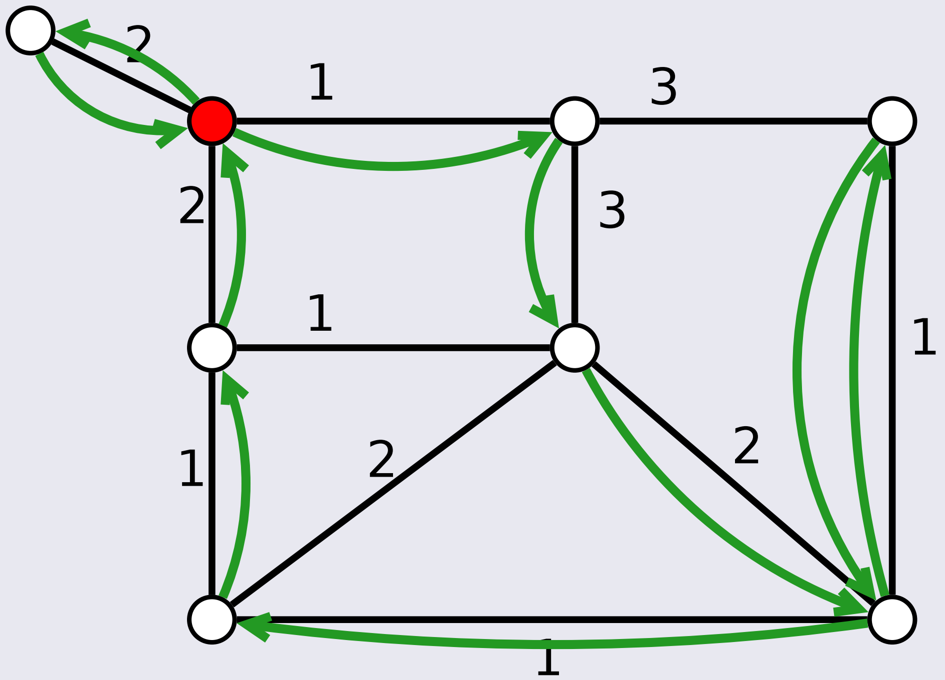
The Traveling Salesman Problem




2
1
3
2
3
1
1
1
2
2
1


ACME
INSURANCE

The Traveling Salesman Problem



TSP Approximations – Upper bounds

- $\frac{3}{2}$ approximation (Christofides 1976)

For graphic (un-weighted) case

- $\frac{3}{2} - \epsilon$ approximation (Oveis Gharan et al. FOCS '11)
- 1.461 approximation (Mömke and Svensson FOCS '11)
- $\frac{13}{9}$ approximation (Mucha STACS '12)
- 1.4 approximation (Sebö and Vygen arXiv '12)

TSP Approximations – Lower bounds

- Problem is APX-hard (Papadimitriou and Yannakakis '93)
- $\frac{5381}{5380}$-inapproximable (Engebretsen STACS '99)
- $\frac{3813}{3812}$-inapproximable (Böckenhauer et al. STACS '00)
- $\frac{220}{219}$-inapproximable (Papadimitriou and Vempala STOC '00, Combinatorica '06)

TSP Approximations – Lower bounds

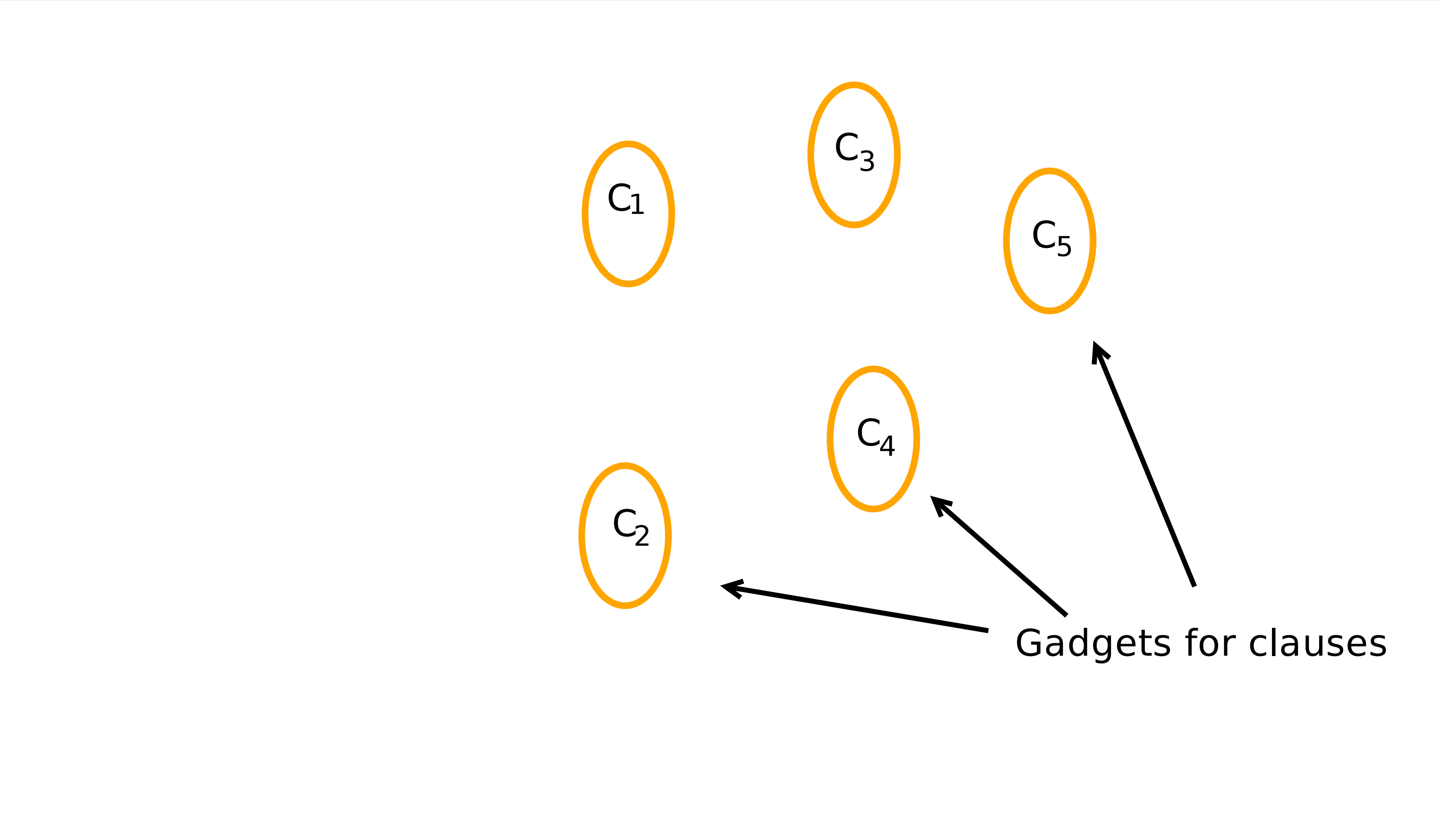
- Problem is APX-hard (Papadimitriou and Yannakakis '93)
- $\frac{5381}{5380}$-inapproximable (Engebretsen STACS '99)
- $\frac{3813}{3812}$-inapproximable (Böckenhauer et al. STACS '00)
- $\frac{220}{219}$-inapproximable (Papadimitriou and Vempala STOC '00, Combinatorica '06)

This talk:

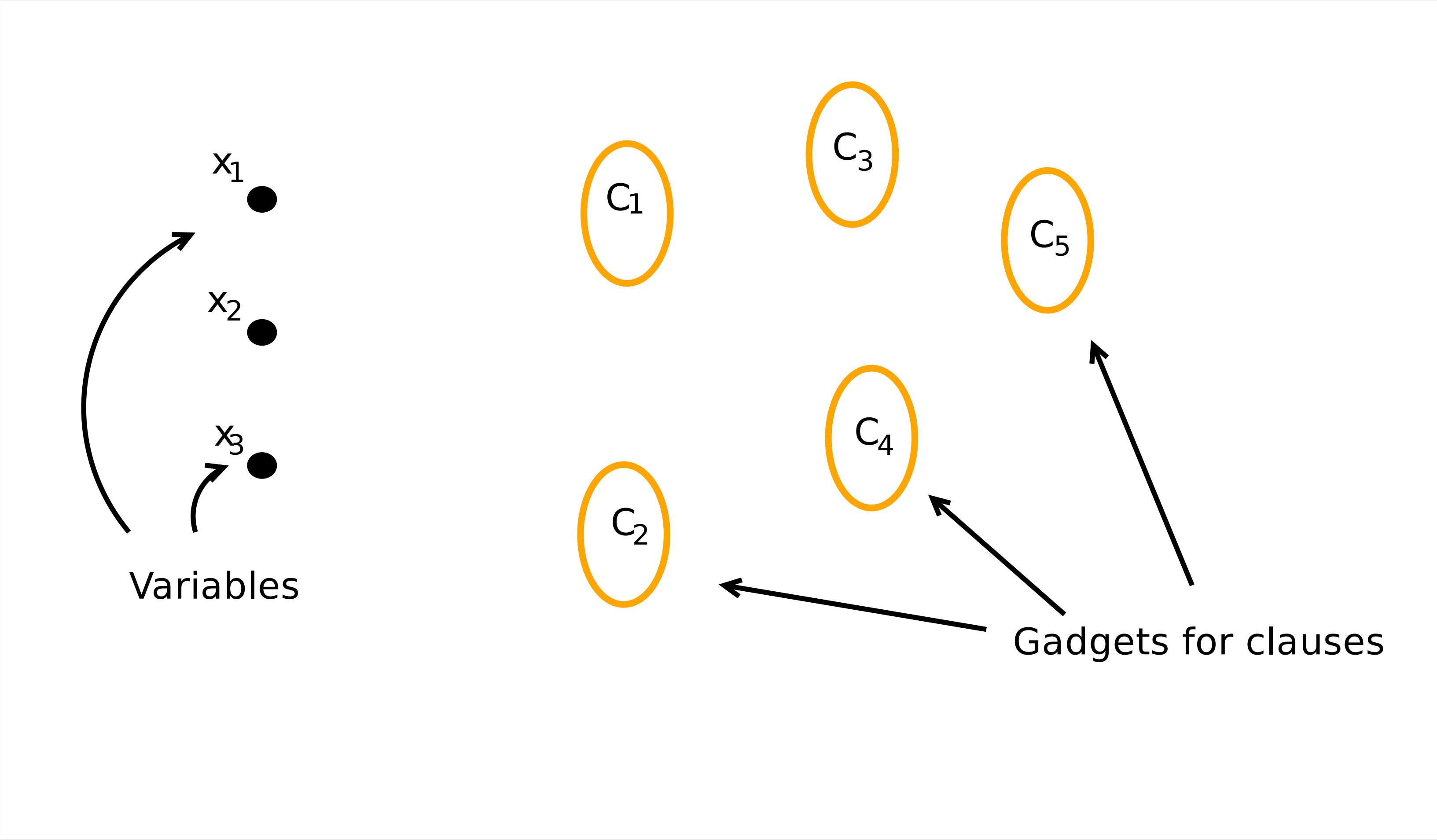
Theorem
There is no $\frac{185}{184}$-approximation algorithm for TSP, unless P=NP.

Reduction Technique

We reduce some inapproximable CSP (e.g. MAX-3SAT) to TSP.

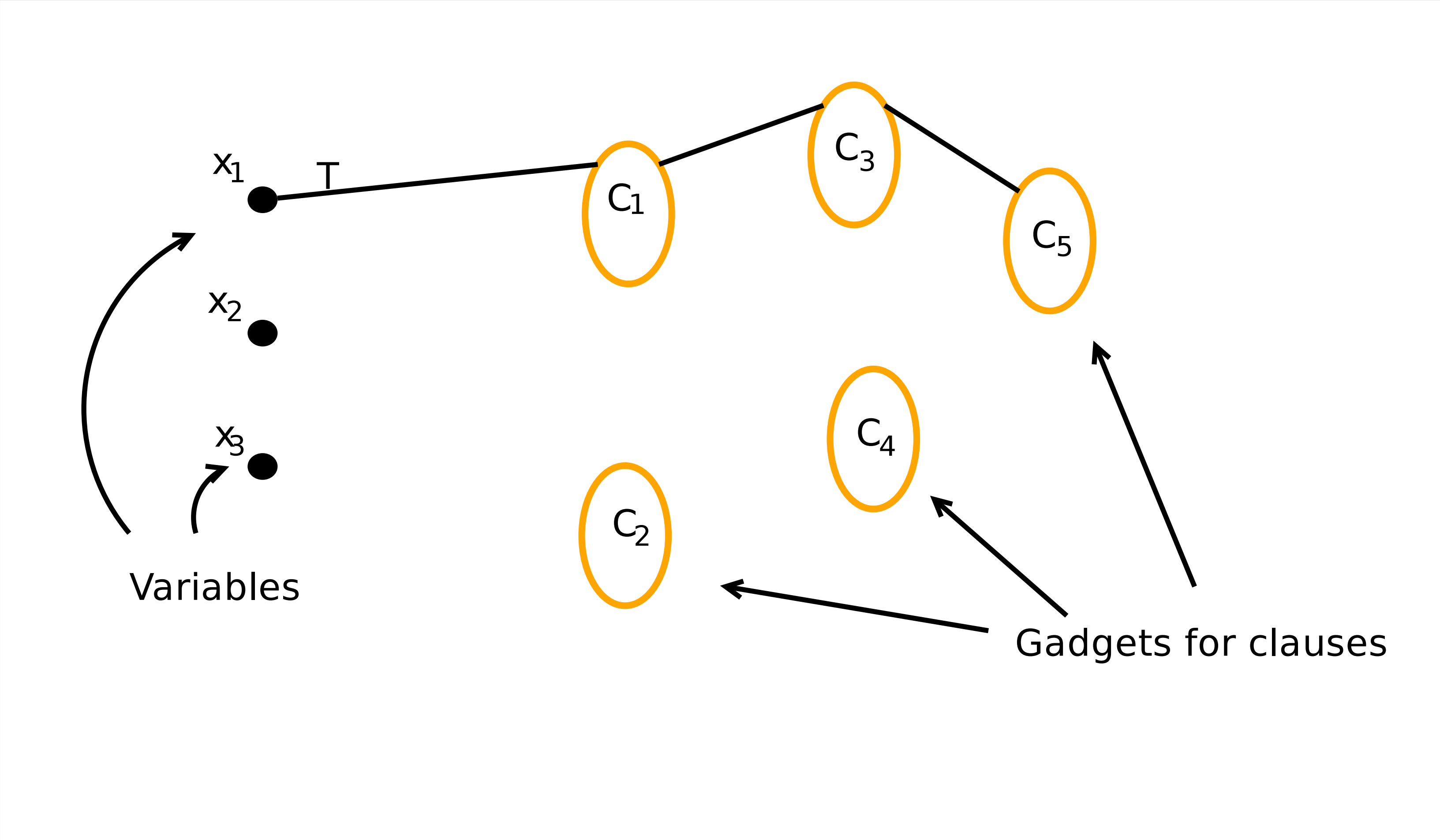


First, design some gadgets to represent the clauses



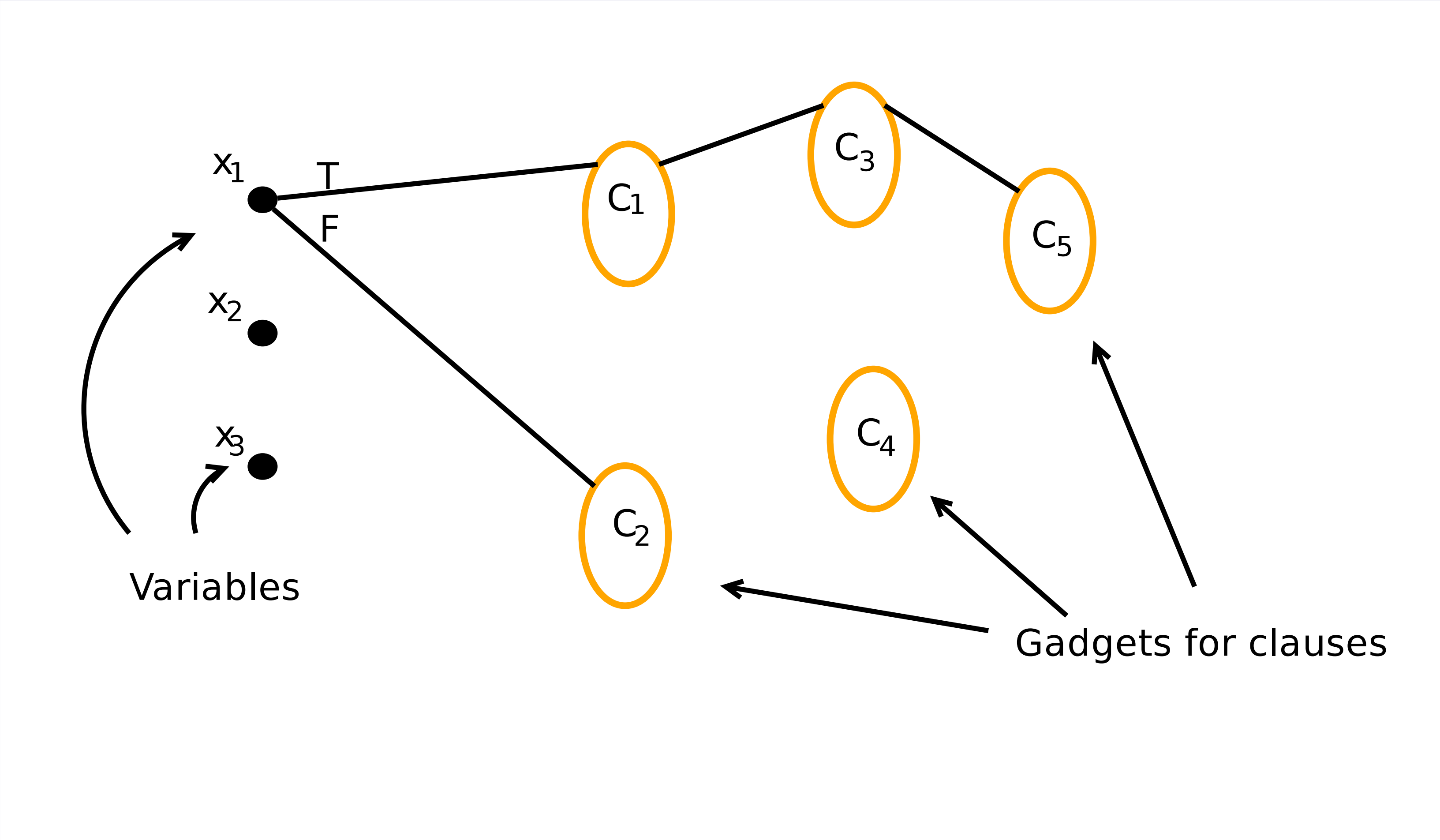
Then, add some choice vertices to represent truth assignments to variables

Reduction Technique



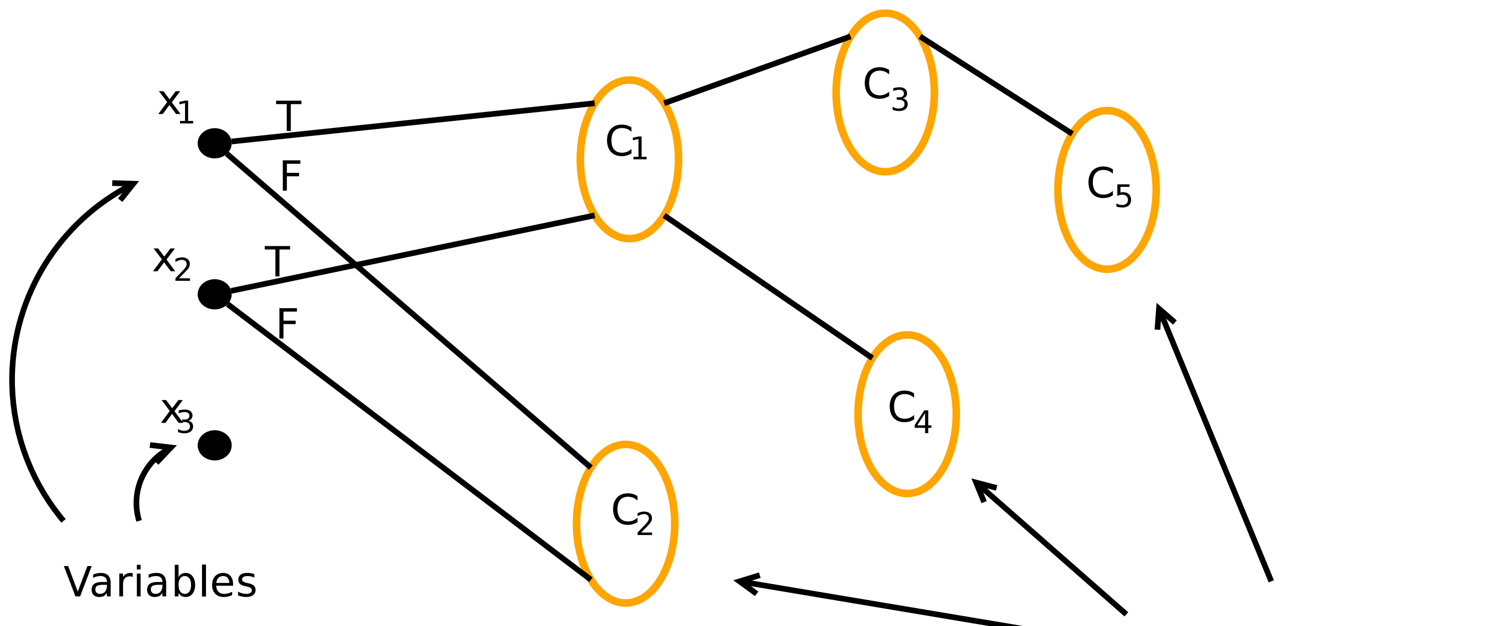
For each variable, create a path through clauses where it appears positive

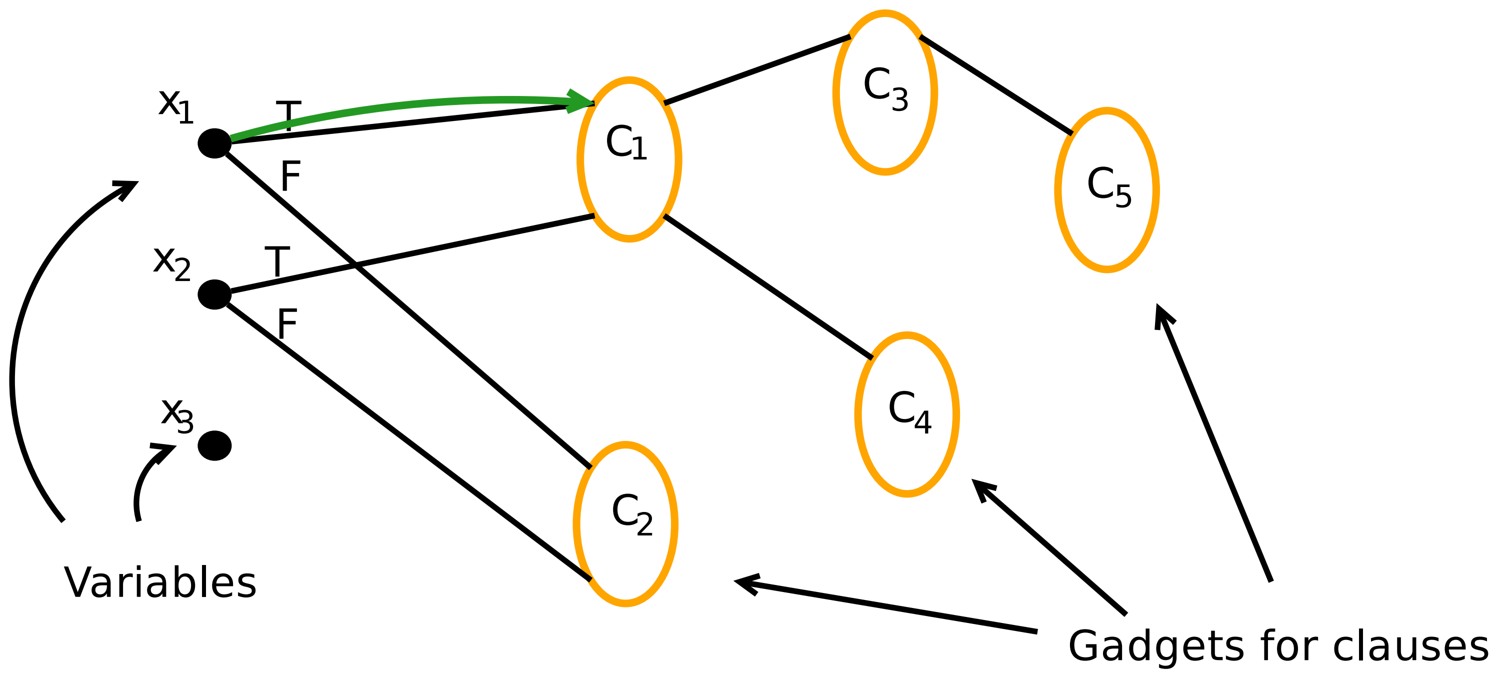
Reduction Technique



. . . and another path for its negative appearances

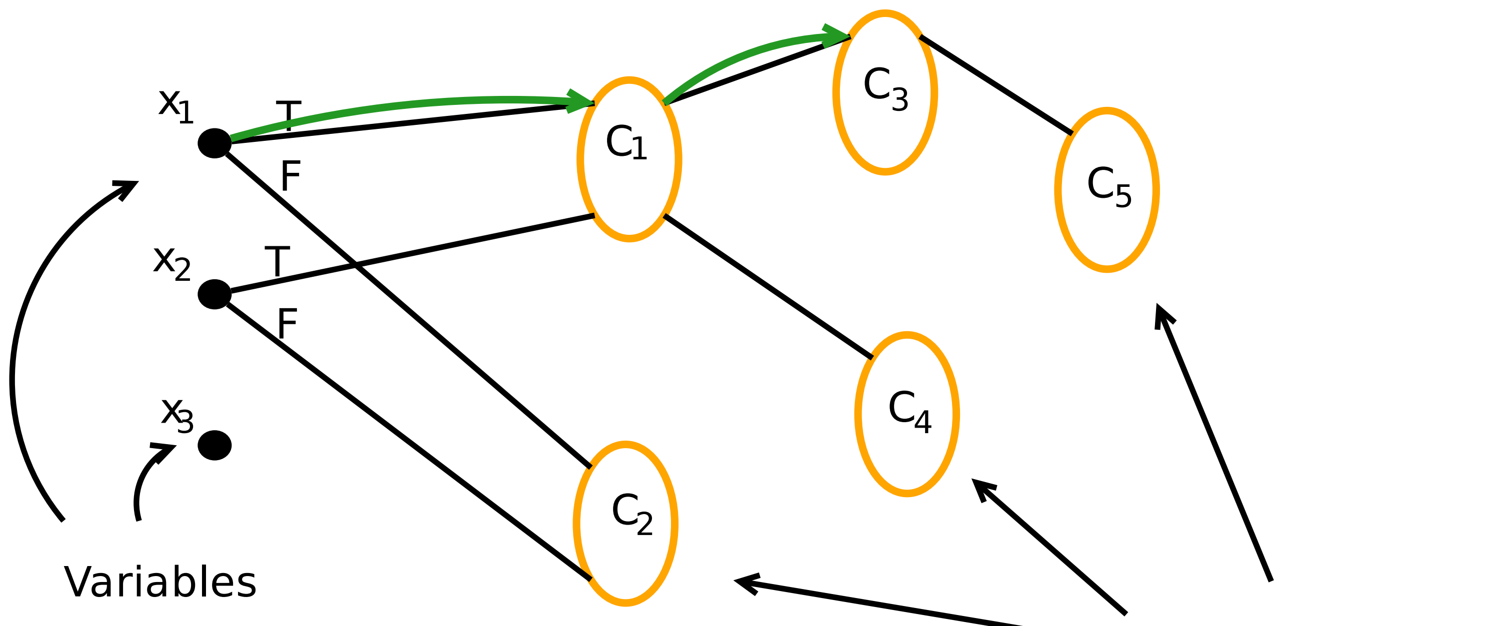
Reduction Technique



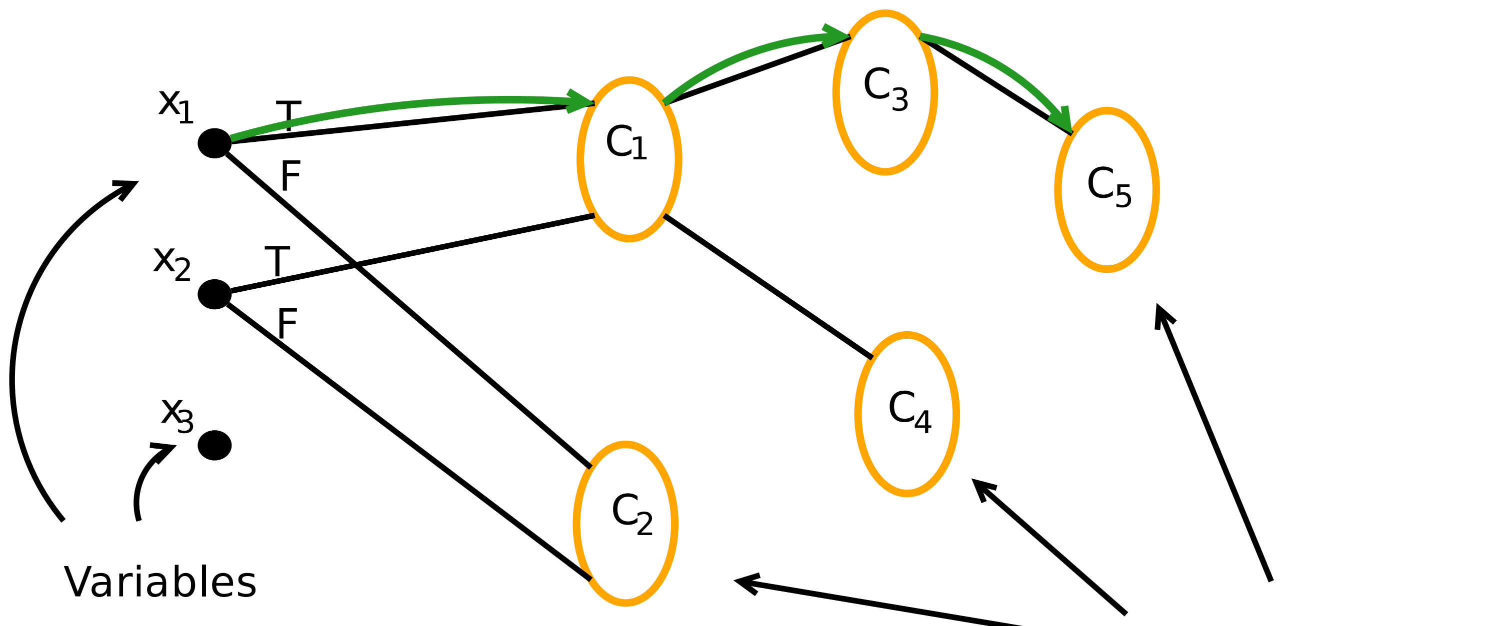


A truth assignment dictates a general path

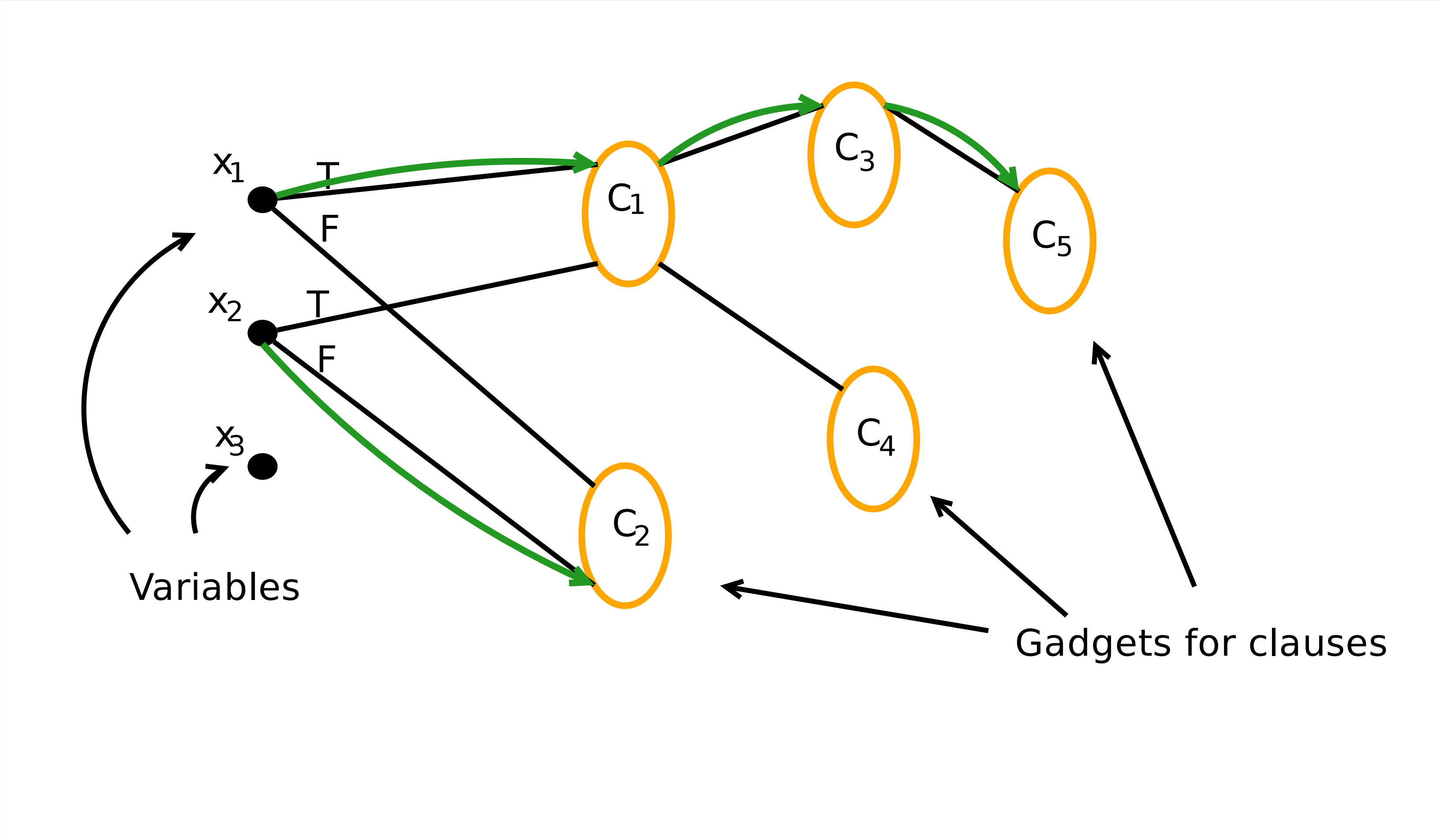
Reduction Technique



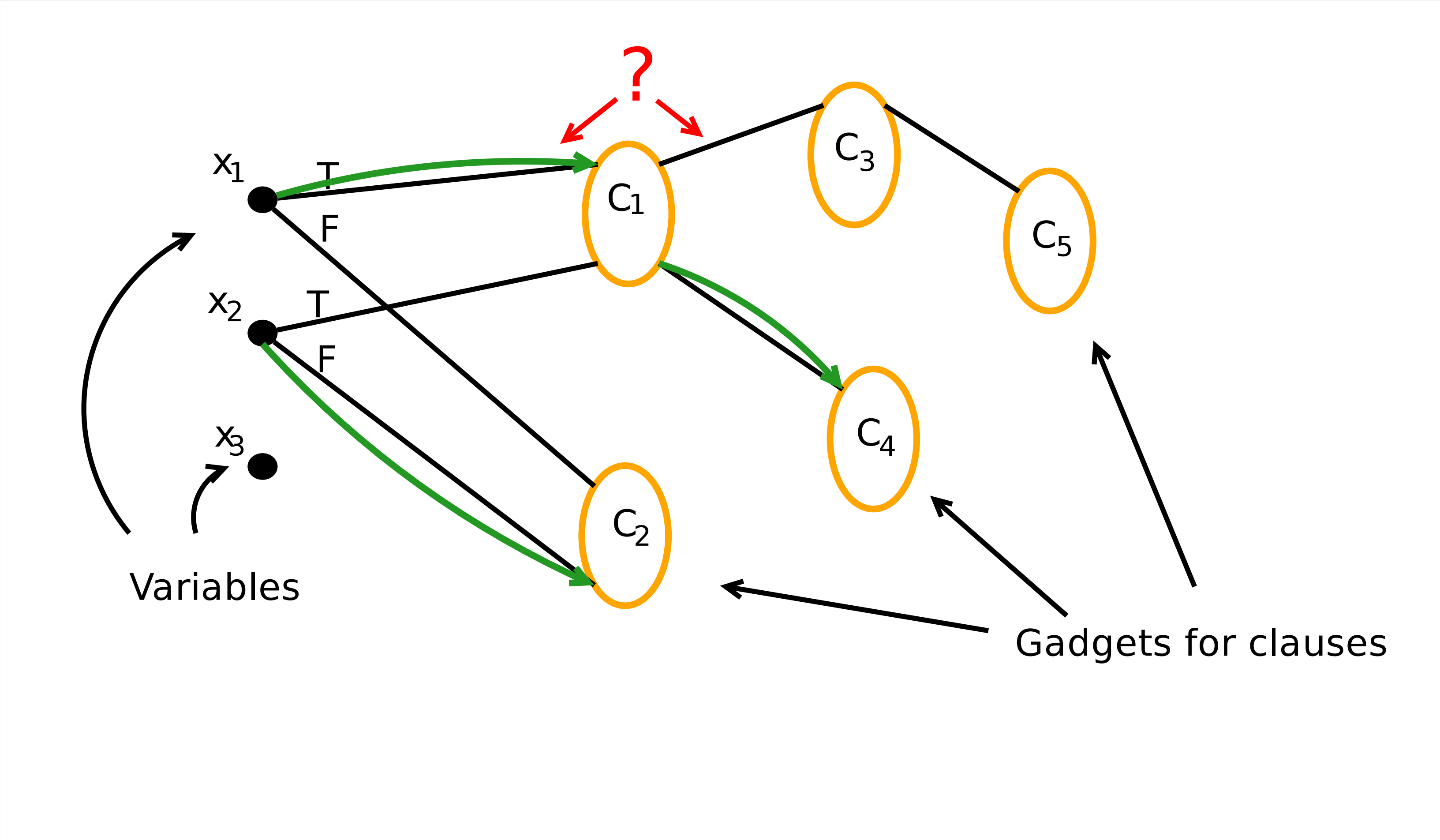
Reduction Technique



Reduction Technique



We must make sure that gadgets are cheaper to traverse if corresponding clause is satisfied



For the converse direction we must make sure that ”cheating” tours are not optimal!

How to ensure consistency

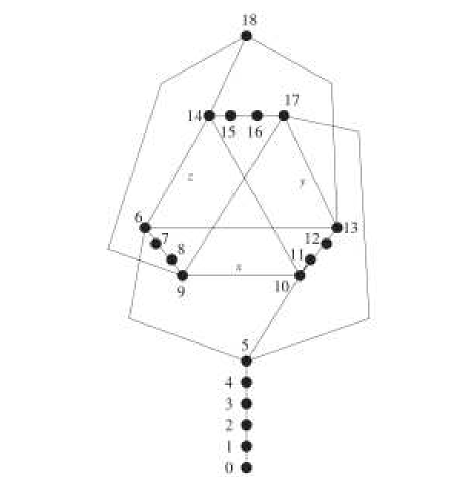


Figure 6. Equation gadget for the symmetric TSP

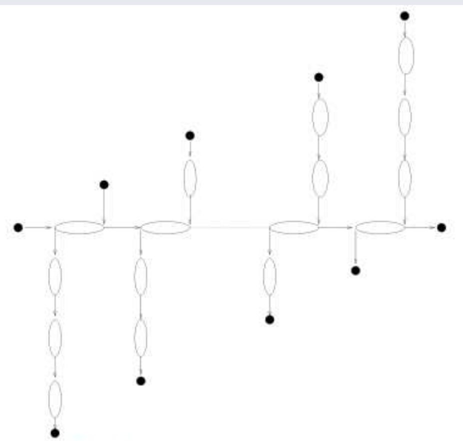


Figure 3. The neighborhood of an edge gadget

- Papadimitriou and Vempala design a gadget for Parity.
- They eliminate variable vertices altogether.
- Consistency is achieved by hooking up gadgets ”randomly”
  - In fact gadgets that share a variable are connected according to the structure dictated by a special graph
  - The graph is called a ”pusher”. Its existence is proved using the probabilistic method.

How to ensure consistency

- Basic idea here: consistency would be easy if each variable occurred at most c times, c a constant.
  - Cheating would only help a tour ”fix” a bounded number of clauses.

How to ensure consistency

- Basic idea here: consistency would be easy if each variable occurred at most c times, c a constant.
  - Cheating would only help a tour ”fix” a bounded number of clauses.
- We will rely on techniques and tools used to prove inapproximability for bounded-occurrence CSPs.
  - This is where expander graphs are important.
  - Main tool: an ”amplifier graph” construction due to Berman and Karpinski.

How to ensure consistency

- Basic idea here: consistency would be easy if each variable occurred at most c times, c a constant.
  - Cheating would only help a tour ”fix” a bounded number of clauses.
- We will rely on techniques and tools used to prove inapproximability for bounded-occurrence CSPs.
  - This is where expander graphs are important.
  - Main tool: an ”amplifier graph” construction due to Berman and Karpinski.
- Result: an easier hardness proof that can be broken down into independent pieces, and also gives an improved bound.

Expander and Amplifier Graphs

Expander Graphs

- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

Expander Graphs

- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

- Definition:

  A graph $G(V, E)$ is an expander if

  - For all $S \subseteq V$ with $|S| \leq \frac{|V|}{2}$ we have for some constant c

    $$\frac{|E(S, V \setminus S)|}{|S|} \geq c$$

  - The maximum degree Δ is bounded

Expander Graphs

- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Expander Graphs

- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

Expander Graphs

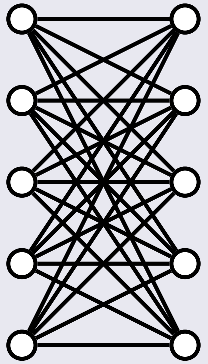
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

A complete bipartite graph is well-connected but not sparse.



Expander Graphs

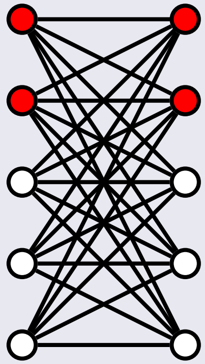
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

A complete bipartite graph is well-connected but not sparse.



Expander Graphs

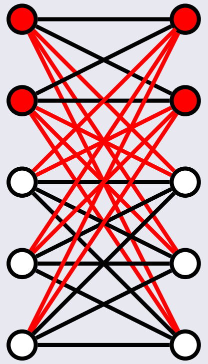
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

A complete bipartite graph is well-connected but not sparse.



Expander Graphs

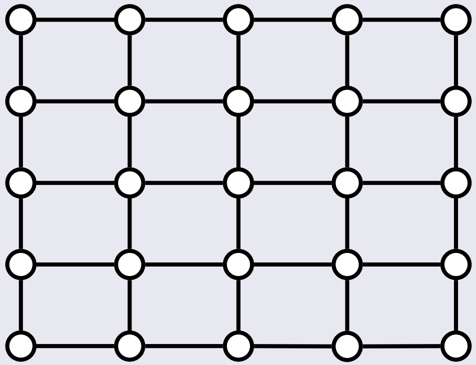
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

A grid is sparse but not well-connected.



Expander Graphs

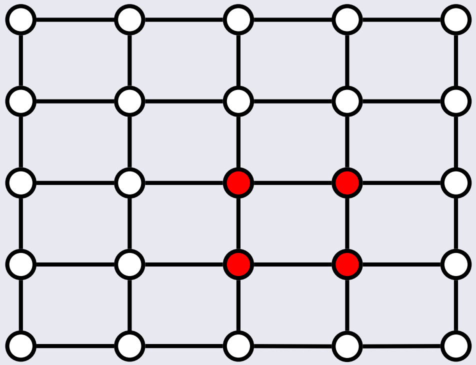
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

A grid is sparse but not well-connected.



Expander Graphs

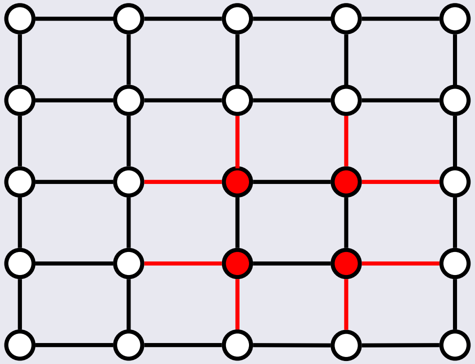
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

A grid is sparse but not well-connected.



Expander Graphs

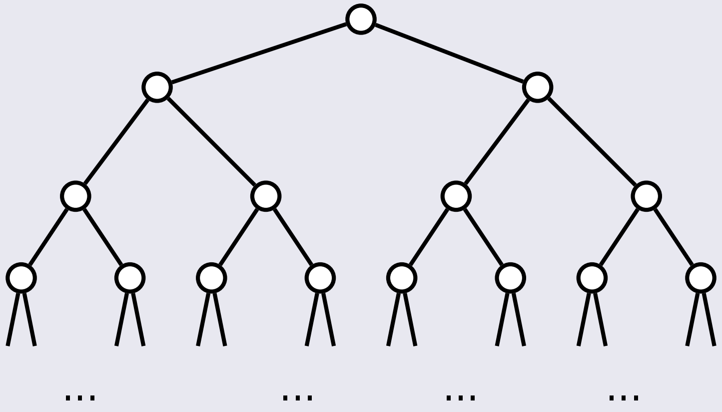
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

An infinite binary tree is a good expander.



Expander Graphs

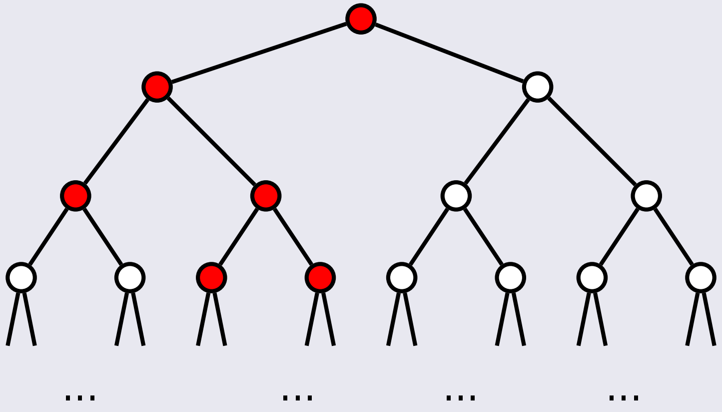
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

An infinite binary tree is a good expander.



Expander Graphs

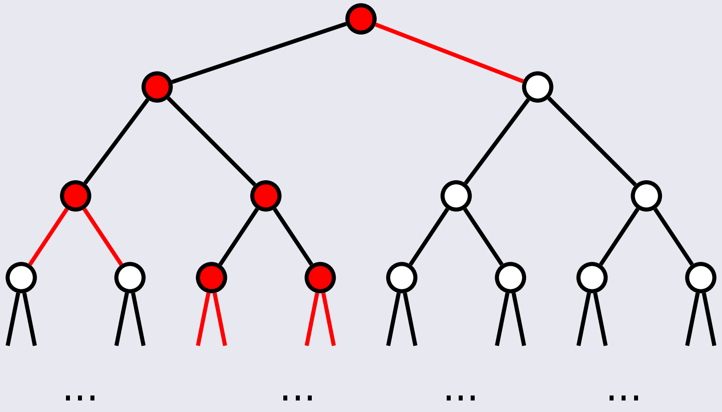
- Informal description:

  An expander graph is a **well-connected** and **sparse** graph.

  - In any possible partition of the vertices into two sets, there are **many** edges crossing the cut.
  - This is achieved even though the graph has low degree, therefore few edges.

Example:

An infinite binary tree is a good expander.



Applications of Expanders

Expander graphs have a number of applications

- Proof of PCP theorem
- Derandomization
- Error-correcting codes

Applications of Expanders

Expander graphs have a number of applications

- Proof of PCP theorem
- Derandomization
- Error-correcting codes
- ... and inapproximability of bounded occurrence CSPs!

Expanders and inapproximability

- Consider the standard reduction from 3-SAT to 3-OCC-3-SAT
  - Replace each appearance of variable x with a fresh variable $x_1, x_2, \ldots, x_n$
  - Add the clauses $(x_1 \to x_2) \wedge (x_2 \to x_3) \wedge \ldots \wedge (x_n \to x_1)$

Expanders and inapproximability

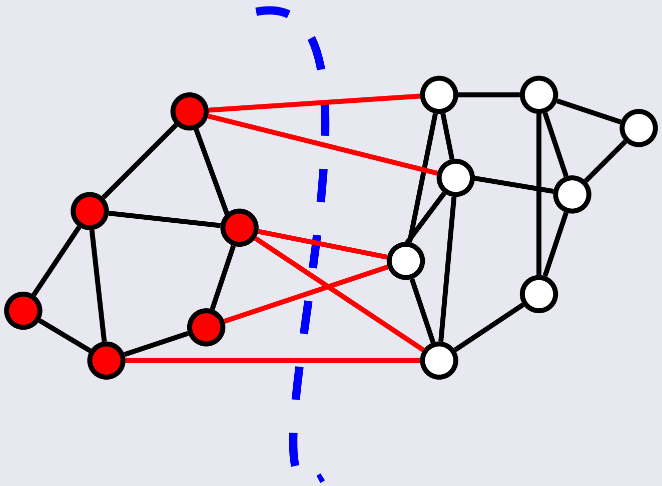
- Consider the standard reduction from 3-SAT to 3-OCC-3-SAT
  - Replace each appearance of variable x with a fresh variable $x_1, x_2, \ldots, x_n$
  - Add the clauses $(x_1 \rightarrow x_2) \wedge (x_2 \rightarrow x_3) \wedge \ldots \wedge (x_n \rightarrow x_1)$

Problem: This does not preserve inapproximability!

- We could add $(x_i \rightarrow x_j)$ for all i, j.
- This ensures consistency but adds too many clauses and does not decrease number of occurrences!

Expanders and inapproximability

- We modify this using a 1-expander [Papadimitriou Yannakakis 91]
  - Recall: a 1-expander is a graph s.t. in each partition of the vertices the number of edges crossing the cut is larger than the number of vertices of the smaller part.



Expanders and inapproximability

- We modify this using a 1-expander [Papadimitriou Yannakakis 91]
  - Replace each appearance of variable x with a fresh variable $x_1, x_2, \ldots, x_n$
  - Construct an n-vertex 1-expander.
  - For each edge (i, j) add the clauses $(x_i \to x_j) \wedge (x_j \to x_i)$

Applications of Expanders

Why does this work?

- Suppose that in the new instance the optimal assignment sets some of the x_i's to 0 and others to 1.
- This gives a partition of the 1-expander.
- Each edge cut by the partition corresponds to an unsatisfied clause.
- Number of cut edges $>$ number of minority assigned vertices $=$ number of clauses lost by being consistent.

Hence, it is always optimal to give the same value to all x_i's.

- Also, because expander graphs are sparse, only linear number of clauses added.
- This gives some inapproximability constant.

Where are all the expanders?

- Expanders sound useful. But how good expanders can we get?

  We want:

  - Low degree – few edges
  - High expansion

  These are conflicting goals!

Where are all the expanders?

- Expanders sound useful. But how good expanders can we get?

  We want:

  - Low degree – few edges
  - High expansion

  These are conflicting goals!

For given Δ what is the highest possible expansion $\phi(\Delta)$ any graph can have?

Where are all the expanders?

- Expanders sound useful. But how good expanders can we get?

  We want:

  - Low degree – few edges
  - High expansion

  These are conflicting goals!

> For given Δ what is the highest possible expansion $\phi(\Delta)$ any graph can have?

- Construction method not obvious!
- Note that for $\Delta = 2$ we have $\phi(\Delta) \to 0$.

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]
  - No graph has expansion more than $\frac{\Delta}{2} - \Omega(\sqrt{\Delta})$ [Alon 97]

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph
  - Such a graph is constructed by taking Δn vertices, selecting u.a.r. a perfect matching and then merging groups of Δ vertices into one.

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph
- Consider a fixed set of vertices $S \subseteq V$.
  - What is the probability that this set has small expansion?

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph
- Consider a fixed set of vertices $S \subseteq V$.
  - What is the probability that this set has small expansion?
  - If this probability is $< 2^{-n}$ we are done, by union bound.

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph
- Consider a fixed set of vertices $S \subseteq V$.
  - What is the probability that this set has small expansion? We can calculate it exactly!

$$P(S,c) = \binom{\Delta|S|}{c}\binom{\Delta n - \Delta|S|}{c} c! \frac{(\Delta|S| - c)!!(\Delta n - \Delta|S| - c)!!}{(\Delta n)!!}$$

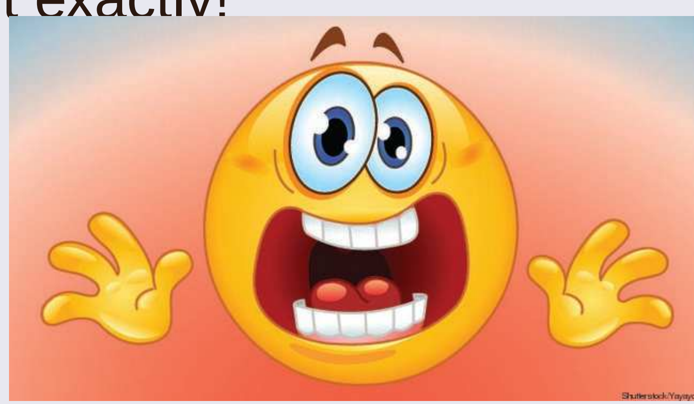
Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph
- Consider a fixed set of vertices $S \subseteq V$.
  - What is the probability that this set has small expansion? We can calculate it exactly!

$$P(S,c) = \binom{\Delta|S|}{c}$$



$$\frac{(\Delta n - \Delta|S| - c)!!}{(\Delta n)!!}$$

Random Graphs are Expanders

- Most graphs are good expanders!
  - Random Δ-regular graphs have expansion at least $\frac{\Delta}{2} - O(\sqrt{\Delta})$ whp. [Bollobás 88]

Proof Sketch:

- Consider a random Δ-regular graph
- Consider a fixed set of vertices $S \subseteq V$.
  - What is the probability that this set has small expansion? We can calculate it exactly!

$$P(S, c) = \binom{\Delta|S|}{c}\binom{\Delta n - \Delta|S|}{c} c! \frac{(\Delta|S| - c)!!(\Delta n - \Delta|S| - c)!!}{(\Delta n)!!}$$

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
  - In particular, random 6-regular graphs are 1-expanders.

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
- Can we improve on these concrete numbers?

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
- Can we improve on these concrete numbers?

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
- Can we improve on these concrete numbers?

High-level argument:

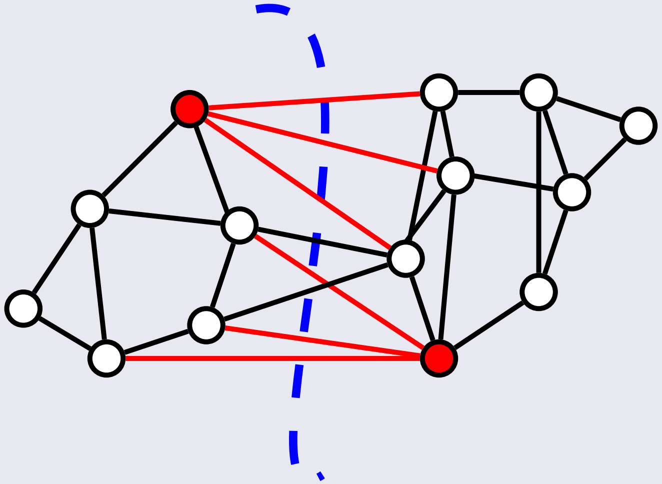
- Suppose a bad set S exists
- If we can exchange a vertex from S with one from $V \setminus S$ and decrease the cut, we have a worse set
- Eventually this process will stop
- Bad set exists $\rightarrow$ locally optimal bad set exists
- $\rightarrow$ Only need to bound probability of a locally optimal bad set

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
- Can we improve on these concrete numbers?

High-level argument:

- (Informally) In a locally optimal bad set all vertices have the majority of their neighbors in the set



Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
- Can we improve on these concrete numbers?

High-level argument:

- The probability of this happening is significantly smaller
  - $\rightarrow$ Better bounds for small specific values of Δ
  - $\rightarrow$ Better coefficient of $\sqrt{\Delta}$ in asymptotics

Improving on Bollobás

- The analysis by Bollobás gives an asymptotically optimal bound, and concrete numbers for specific values of Δ.
- Can we improve on these concrete numbers?

High-level argument:

- The probability of this happening is significantly smaller
  - $\rightarrow$ Better bounds for small specific values of Δ
  - $\rightarrow$ Better coefficient of $\sqrt{\Delta}$ in asymptotics
- But improvement too small!
- Analysis is hard – must be good for something...

Amplifiers

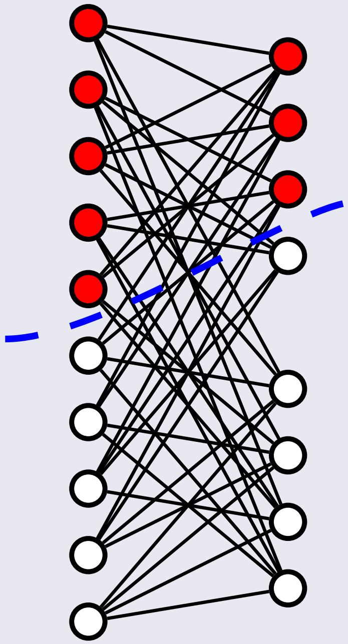
- Previous idea gives noticeable improvement in expansion for $\Delta > 20$
- In TSP reduction we need much smaller Δ
- Better idea: use **existing** amplifier constructions

Amplifiers

- Previous idea gives noticeable improvement in expansion for $\Delta > 20$
- In TSP reduction we need much smaller Δ
- Better idea: use **existing** amplifier constructions

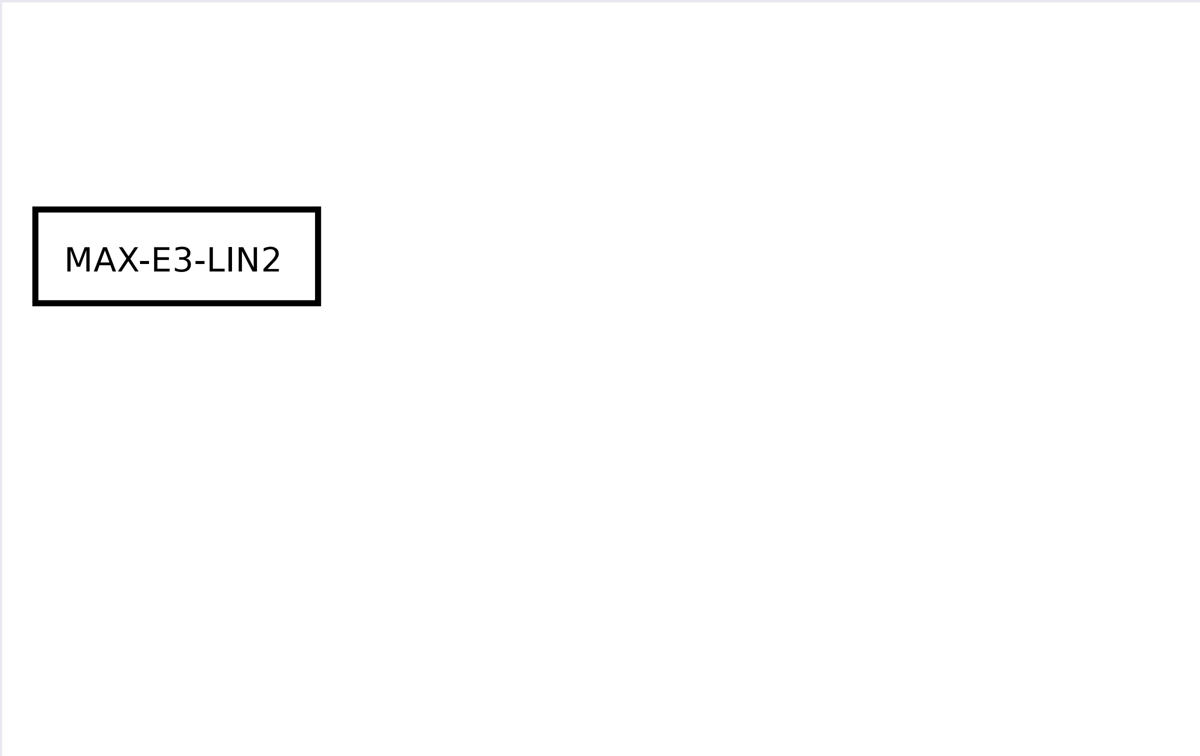
5-regular amplifier [Berman Karpinski 03]

- Bipartite graph. n vertices on left, $0.8n$ vertices on right.
- 4-regular on left, 5-regular on right.
- Graph constructed randomly.
- Crucial Property: whp any partition cuts more edges than the number of left vertices on the smaller set.



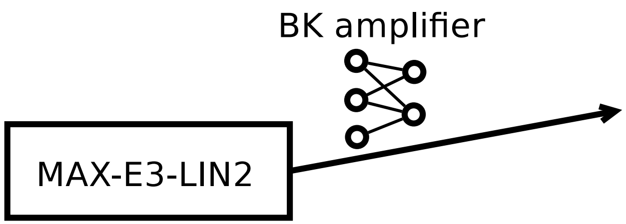
Back to the Reduction

Overview



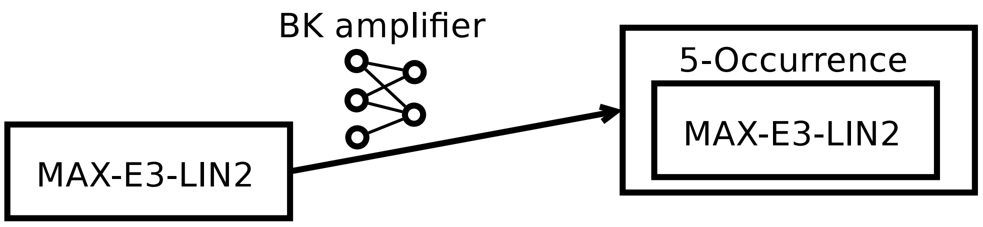
We start from an instance of MAX-E3-LIN2. Given a set of linear equations (mod 2) each of size three satisfy as many as possible.

Problem known to be 2-inapproximable (Håstad)

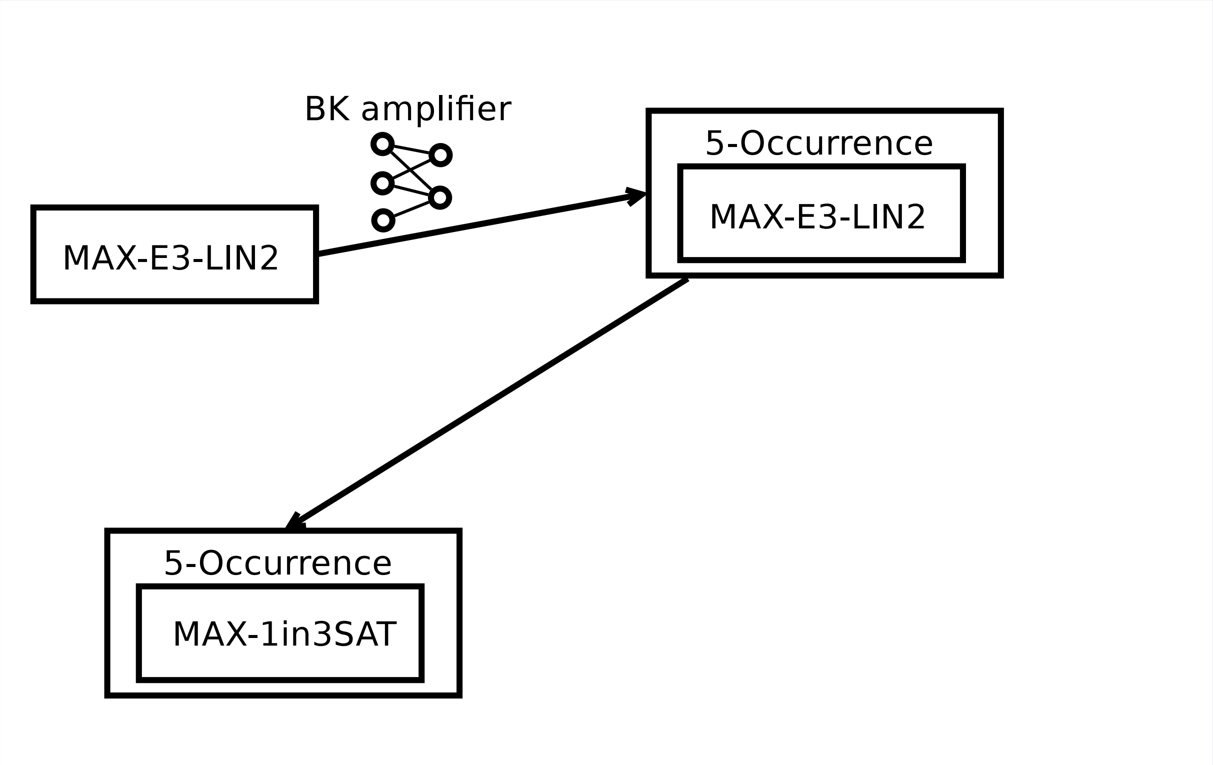


We use the Berman-Karpinski amplifier construction to obtain an instance where each variable appears exactly 5 times (and most equations have size 2).

Overview

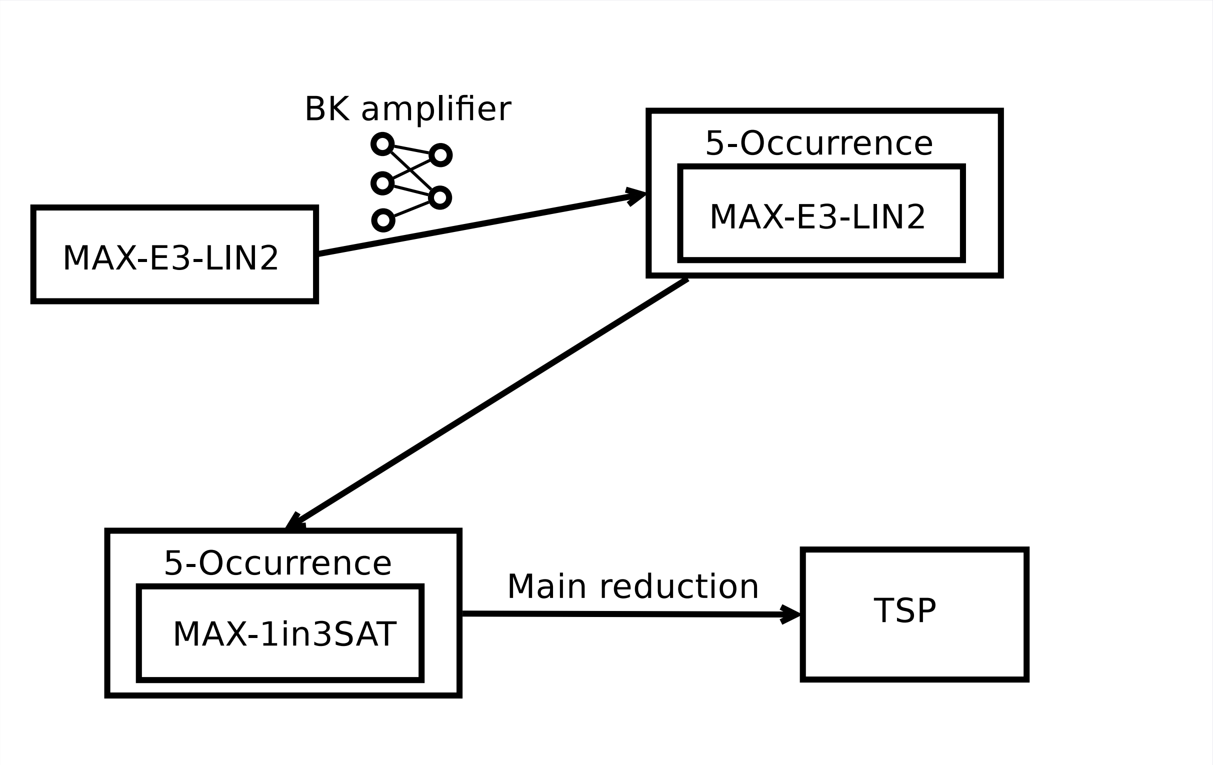


Overview



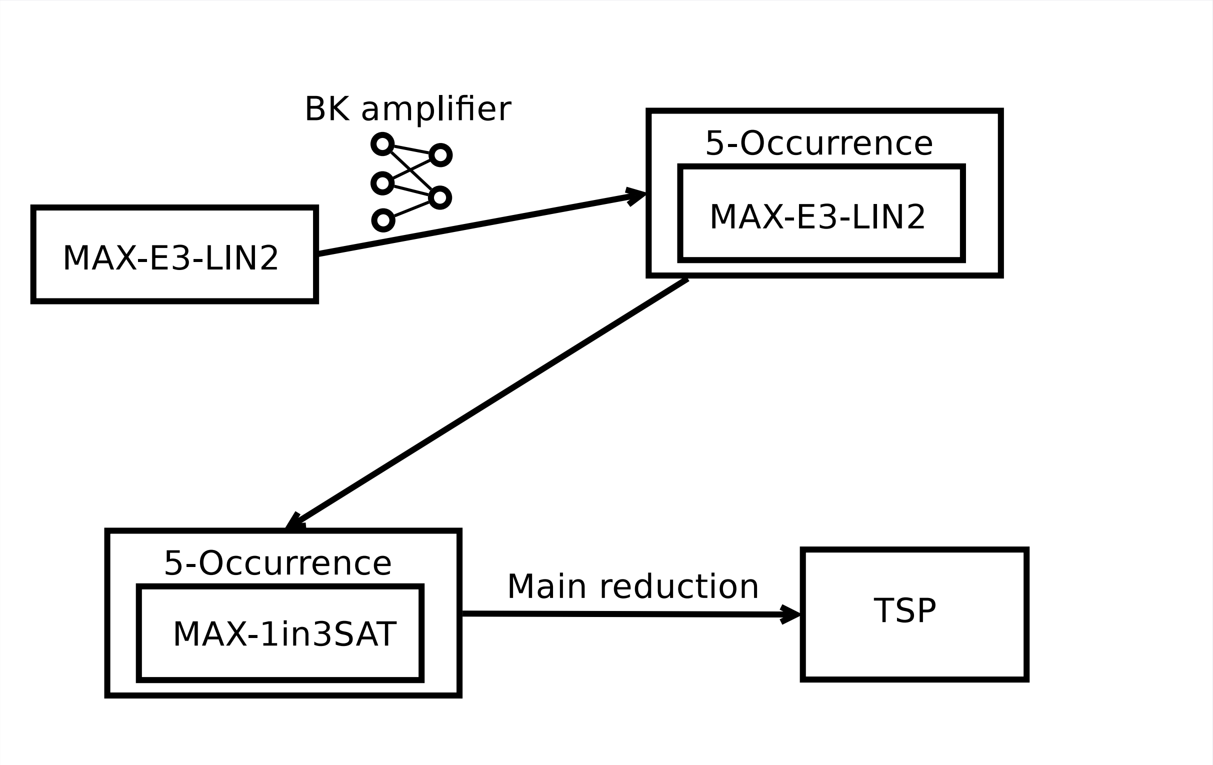
A simple trick reduces this to the 1in3 predicate.

Overview



From this instance we construct a graph.

Overview



From this instance we construct a graph.

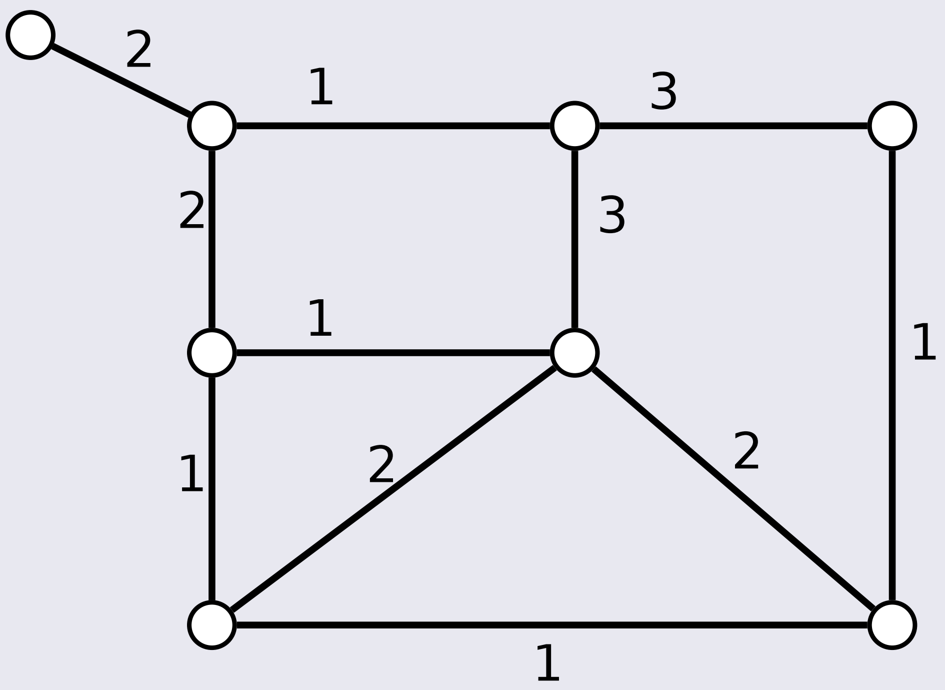
Rest of this talk: some more details about the construction.

1in3-SAT

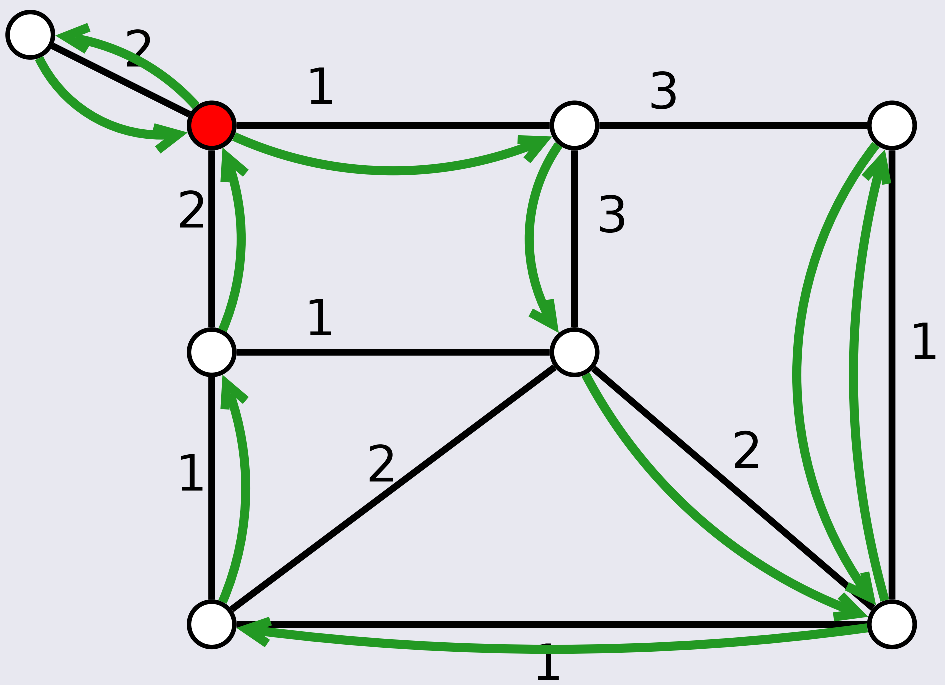
Input:
A set of clauses $(l_1 \vee l_2 \vee l_3)$, l_1, l_2, l_3 literals.
Objective:
A clause is satisfied if exactly one of its literals is true. Satisfy as many clauses as possible.

- Easy to reduce MAX-LIN2 to this problem.
  - Especially for size two equations $(x + y = 1) \leftrightarrow (x \vee y)$.
- Naturally gives gadget for TSP
  - In TSP we'd like to visit each vertex at least once, but not more than once (to save cost)

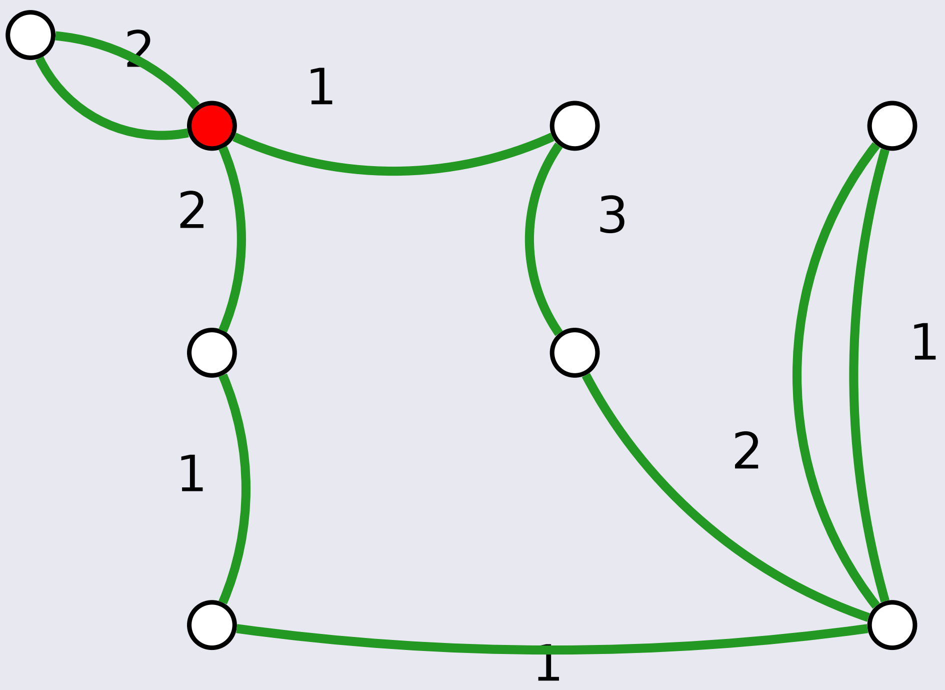
TSP and Euler tours



TSP and Euler tours



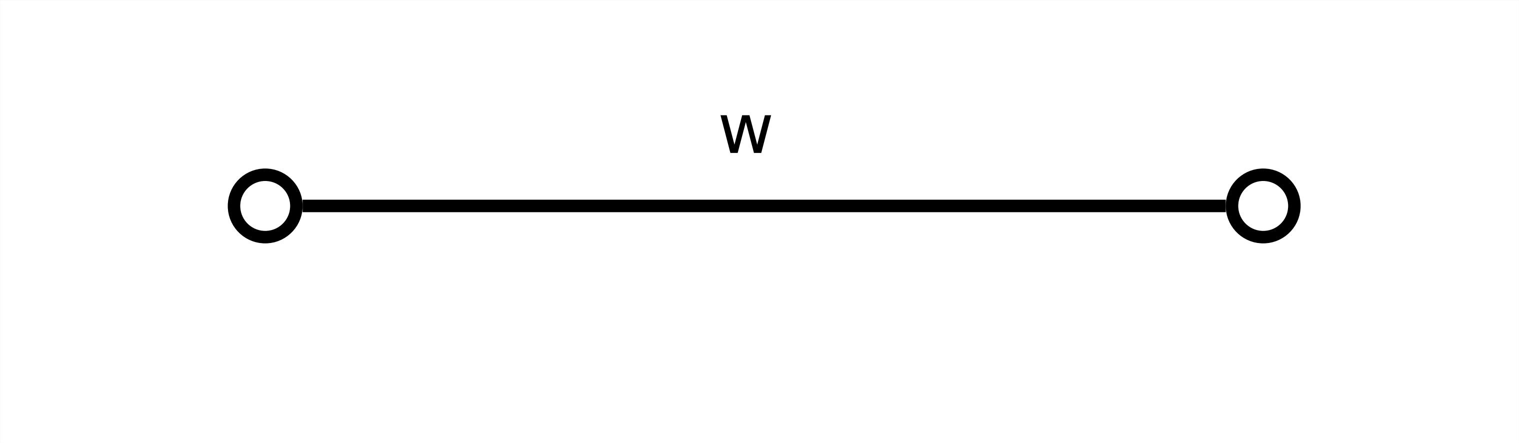
TSP and Euler tours



TSP and Euler tours

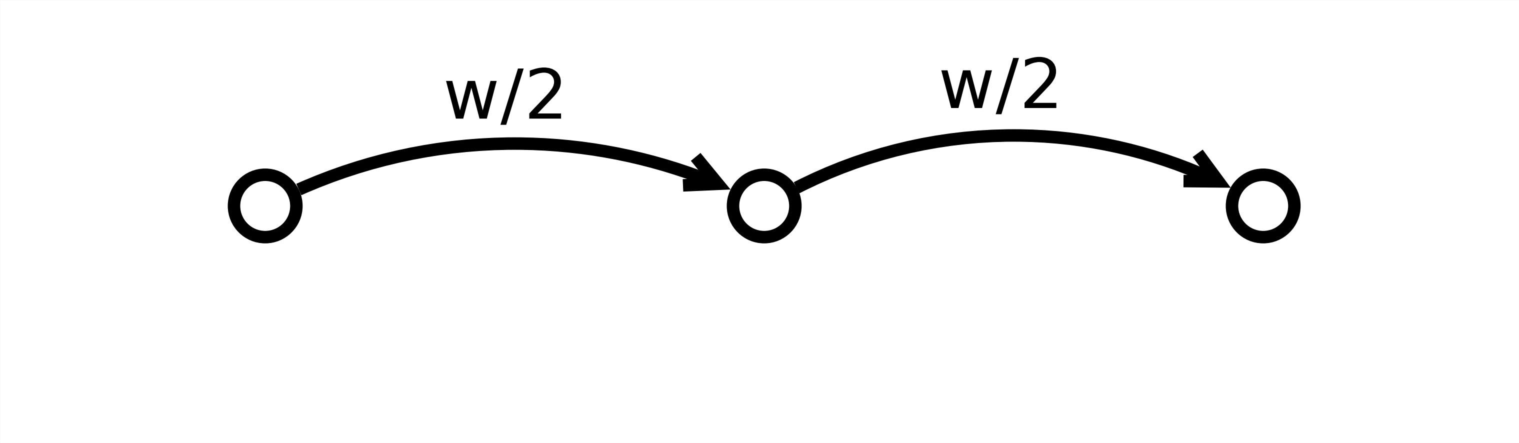
- A TSP tour gives an Eulerian multi-graph composed with edges of G.
- An Eulerian multi-graph composed with edges of G gives a TSP tour.
  - TSP $\equiv$ Select a multiplicity for each edge so that the resulting multi-graph is Eulerian and total cost is minimized
  - **Note**: no edge is used more than twice

Gadget – Forced Edges



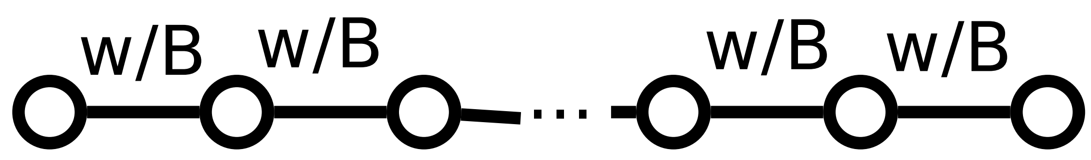
We would like to be able to dictate in our construction that a certain edge has to be used at least once.

Gadget – Forced Edges



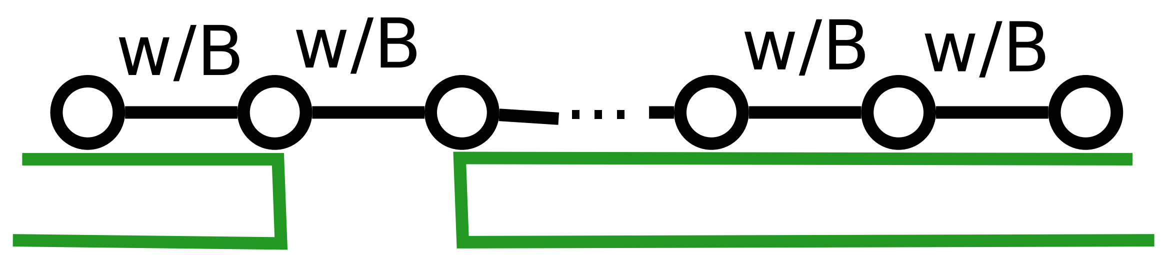
If we had directed edges, this could be achieved by adding a dummy intermediate vertex

Gadget – Forced Edges



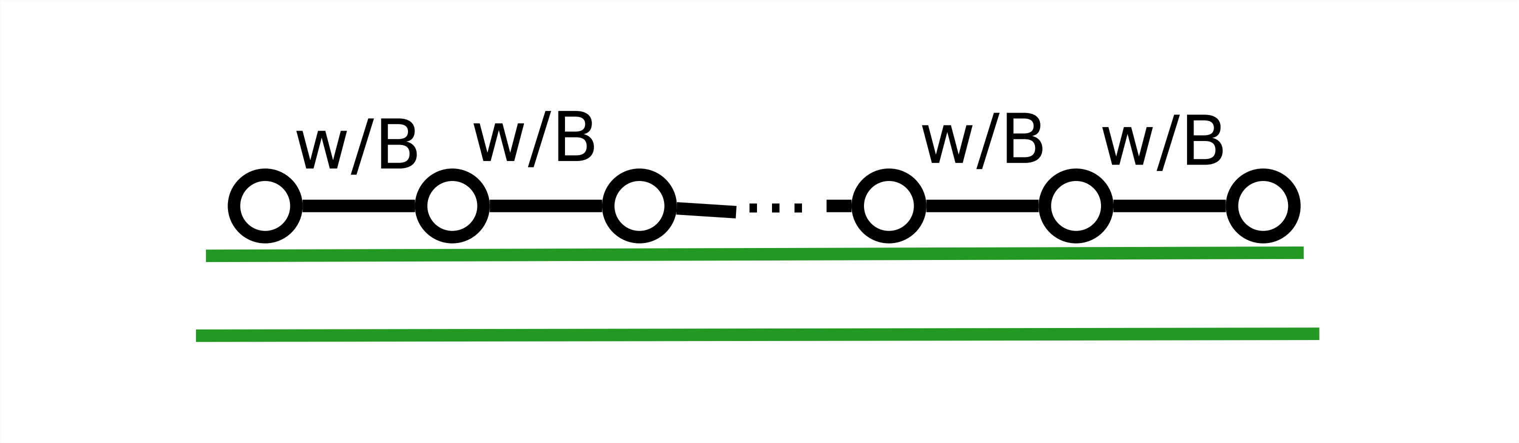
Here, we add many intermediate vertices and evenly distribute the weight w among them. Think of B as very large.

Gadget – Forced Edges



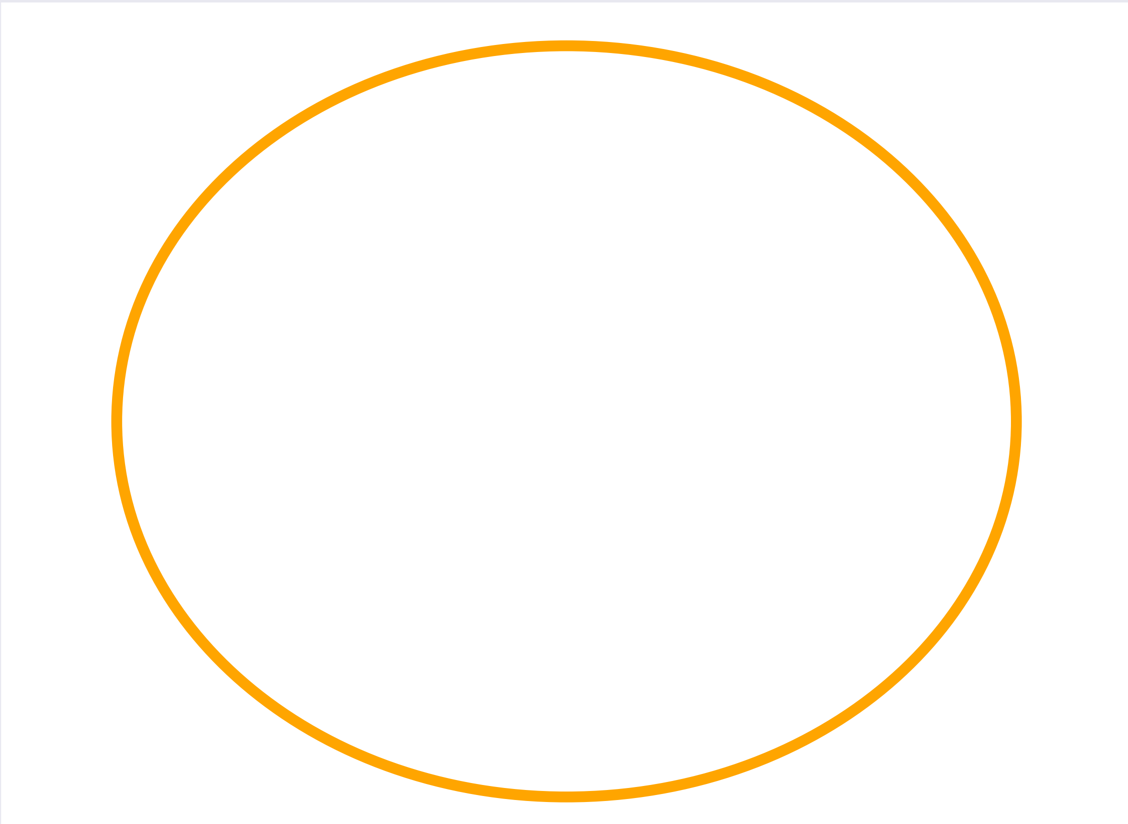
At most one of the new edges may be unused, and in that case all others are used twice.

Gadget – Forced Edges



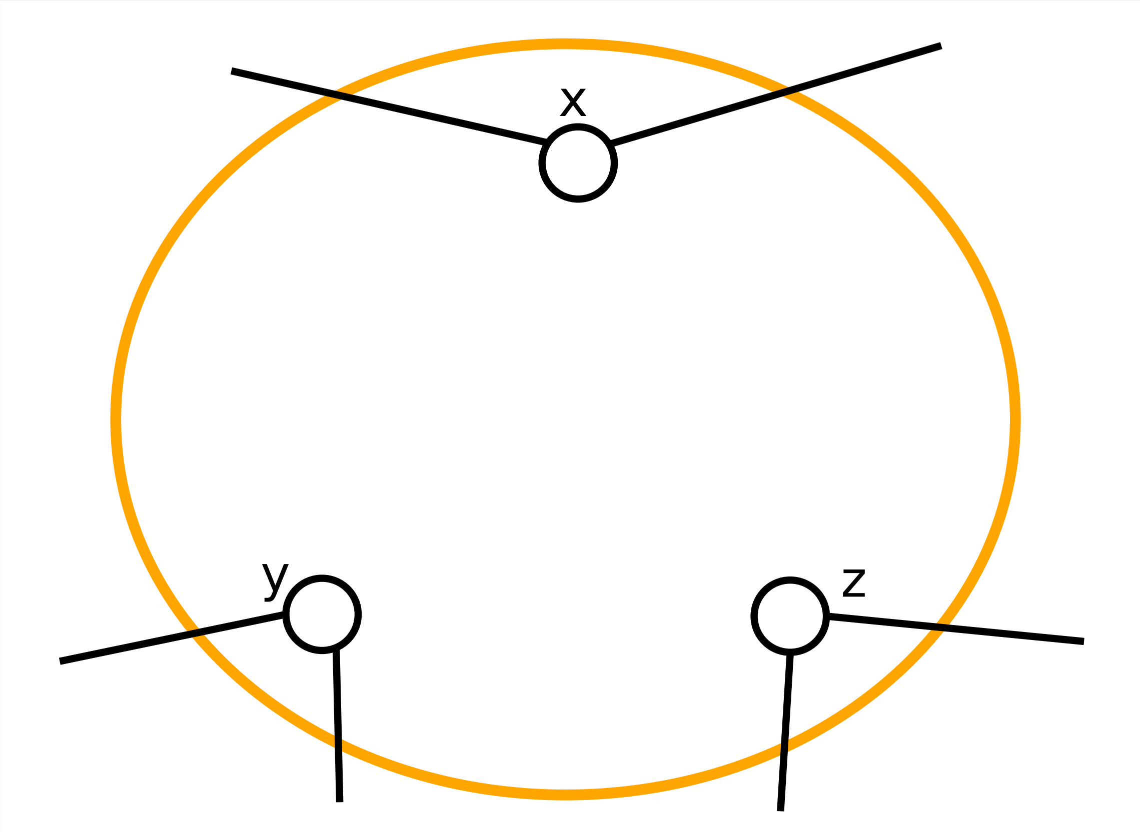
In that case, adding two copies of that edge to the solution doesn't hurt much (for B sufficiently large).

1in3 Gadget



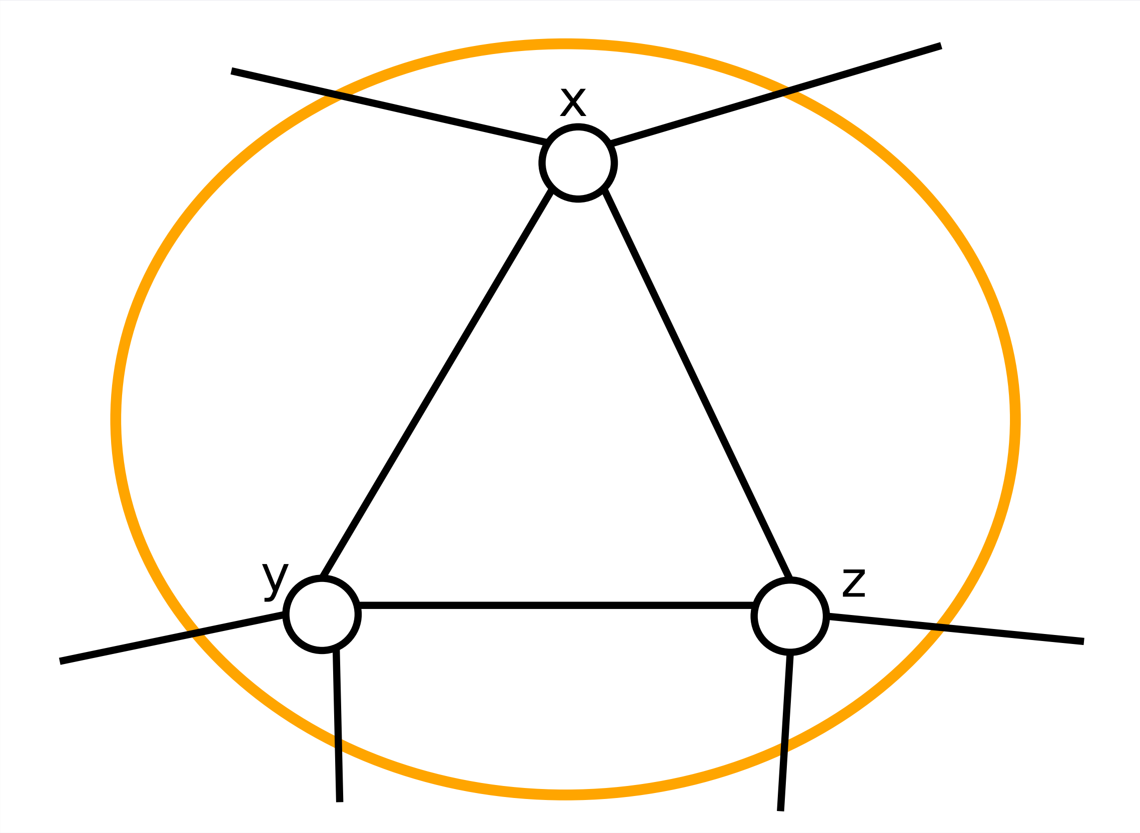
Let's design a gadget for $(x \vee y \vee z)$

1in3 Gadget



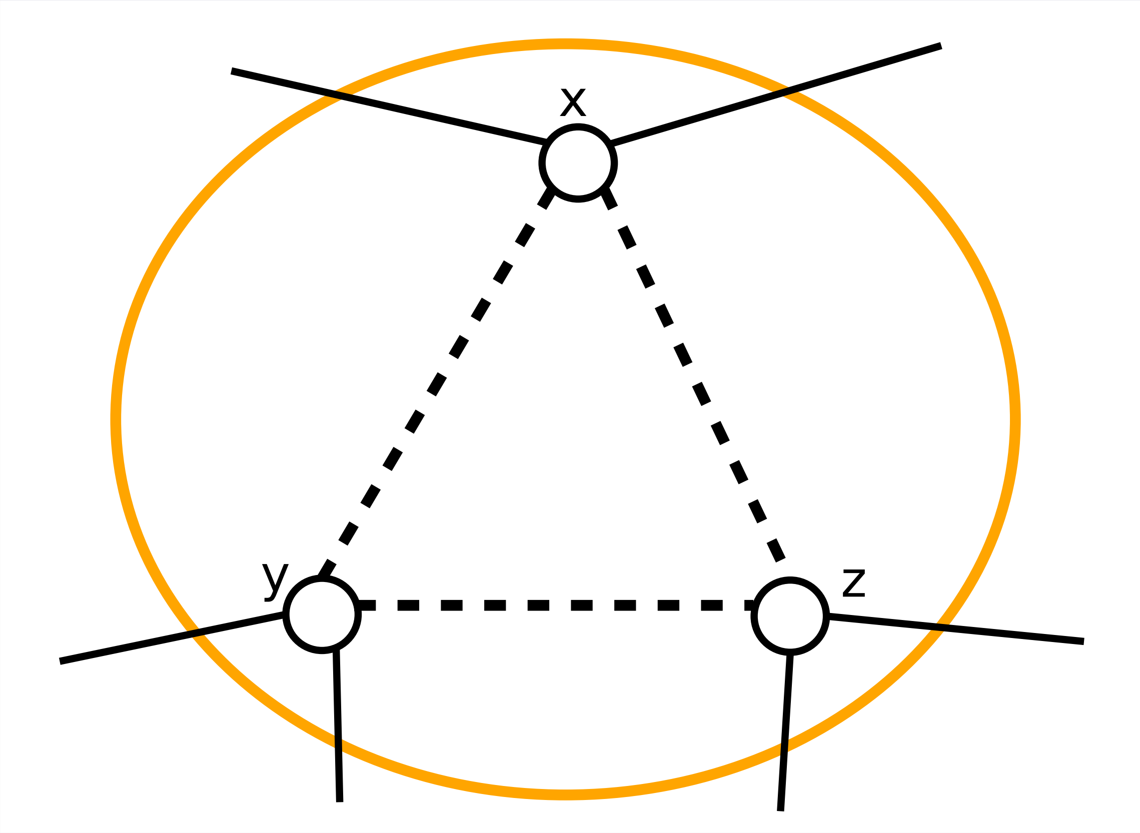
First, three entry/exit points

1in3 Gadget



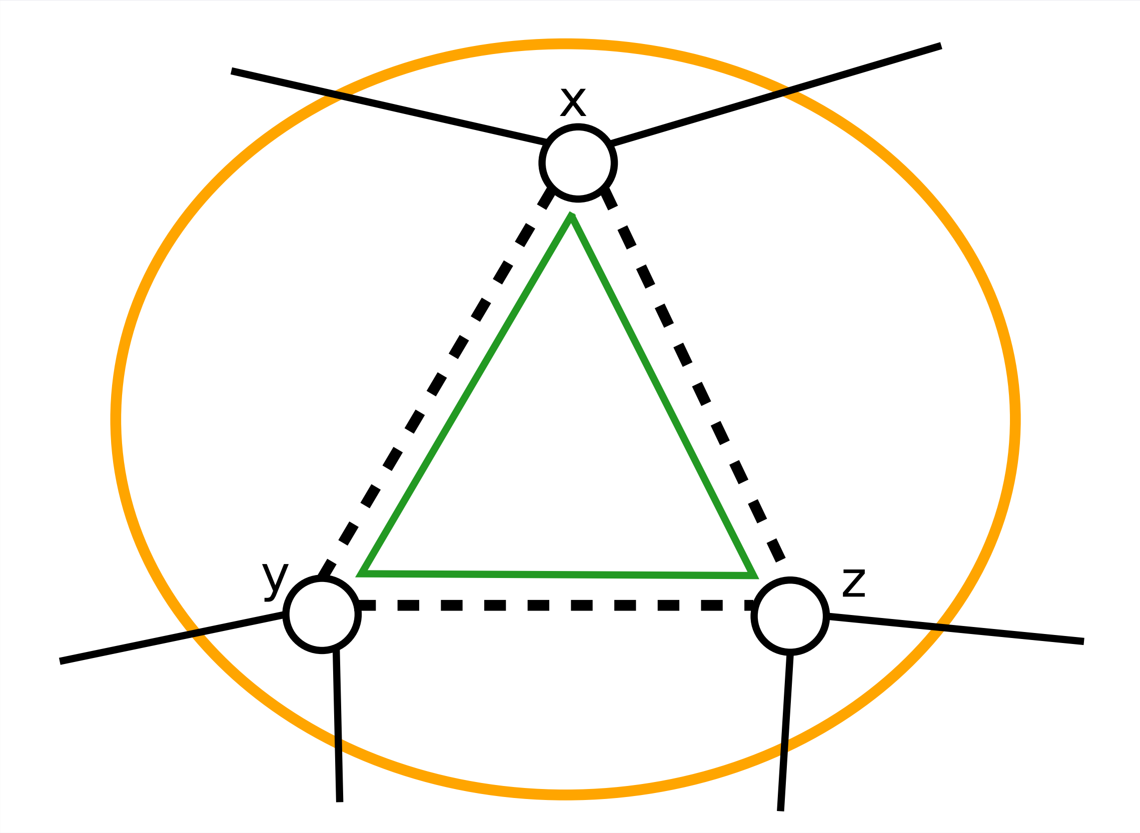
Connect them . . .

1in3 Gadget



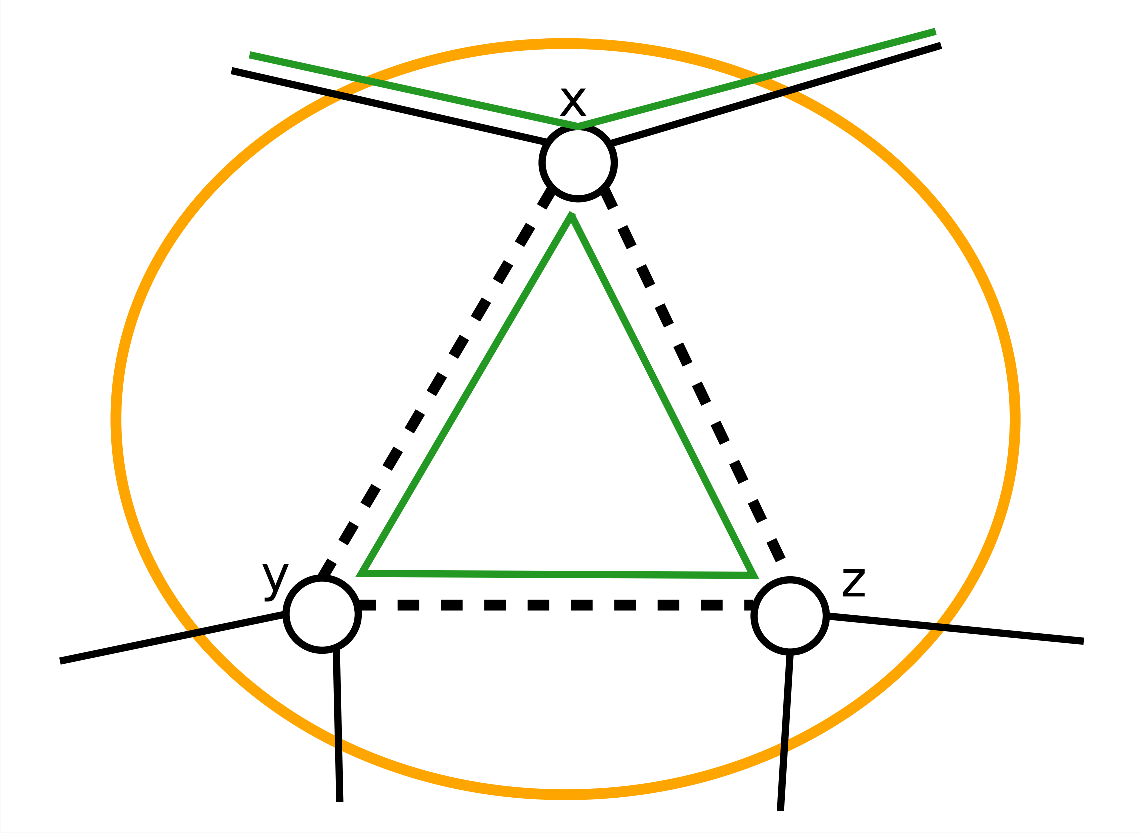
. . . with forced edges

1in3 Gadget



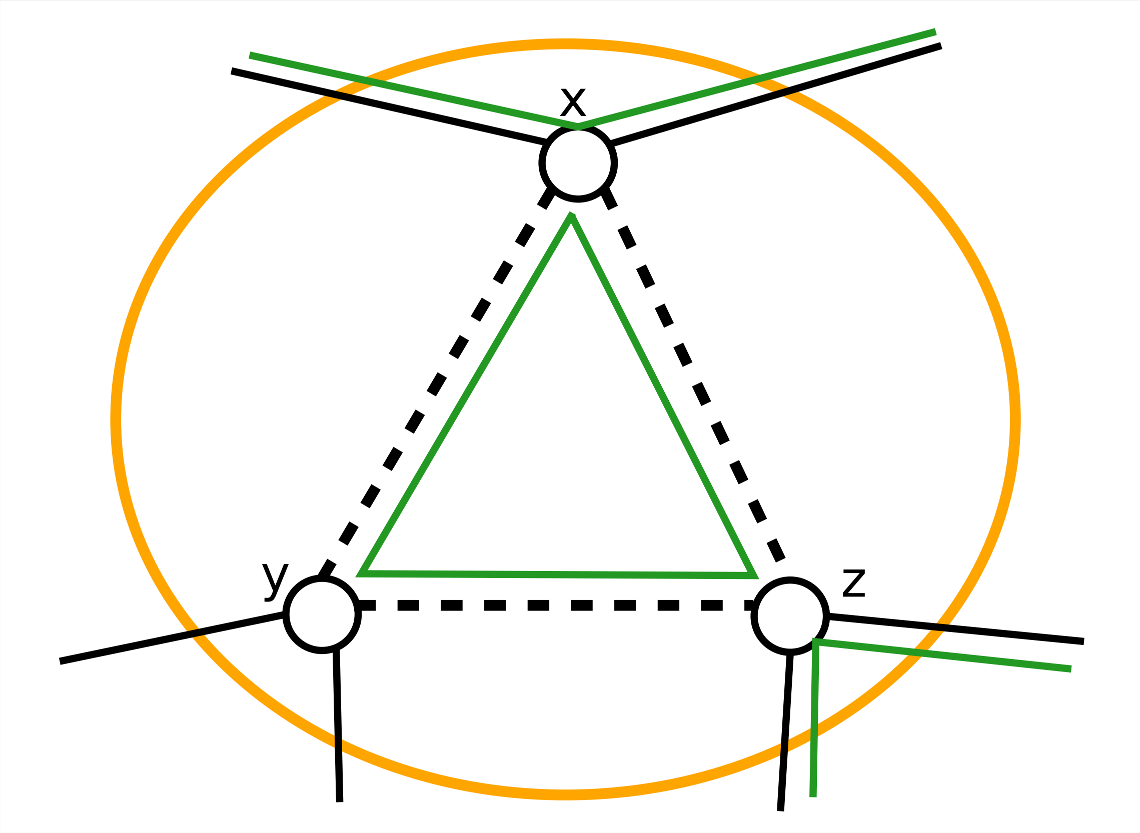
The gadget is a connected component. A good tour visits it once.

1in3 Gadget



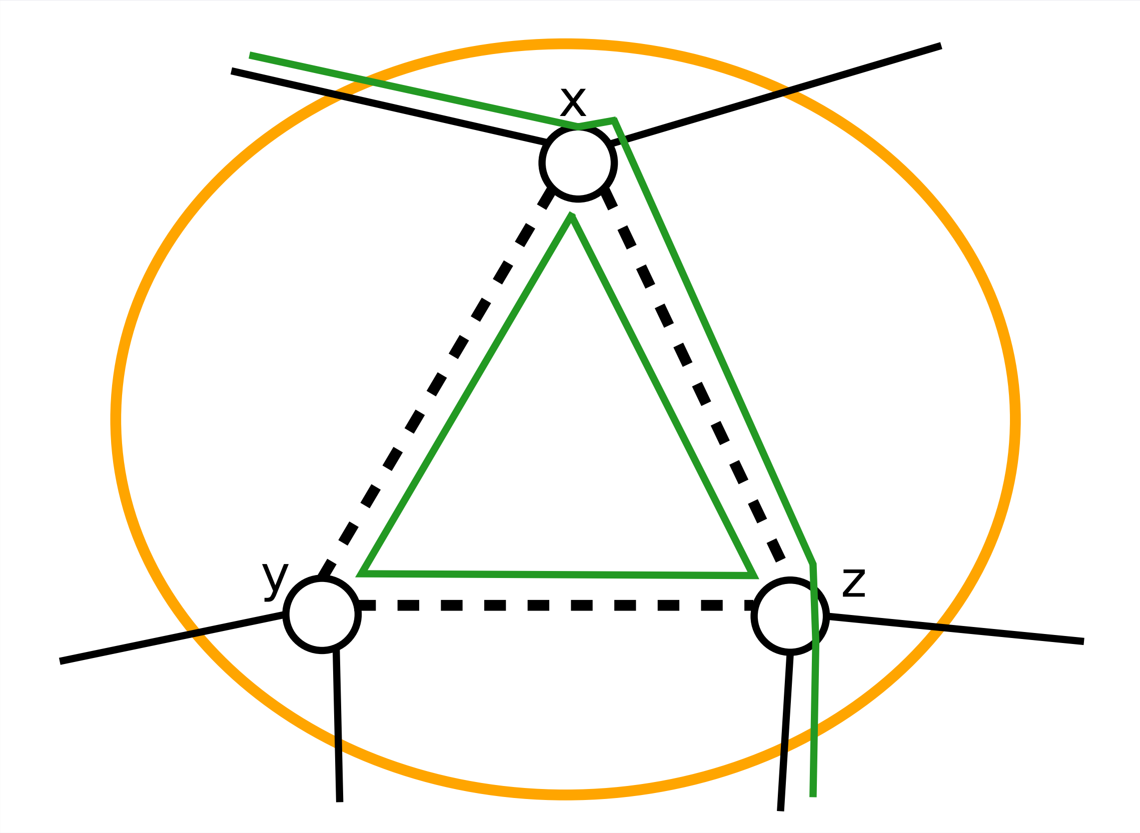
. . . like this

1in3 Gadget

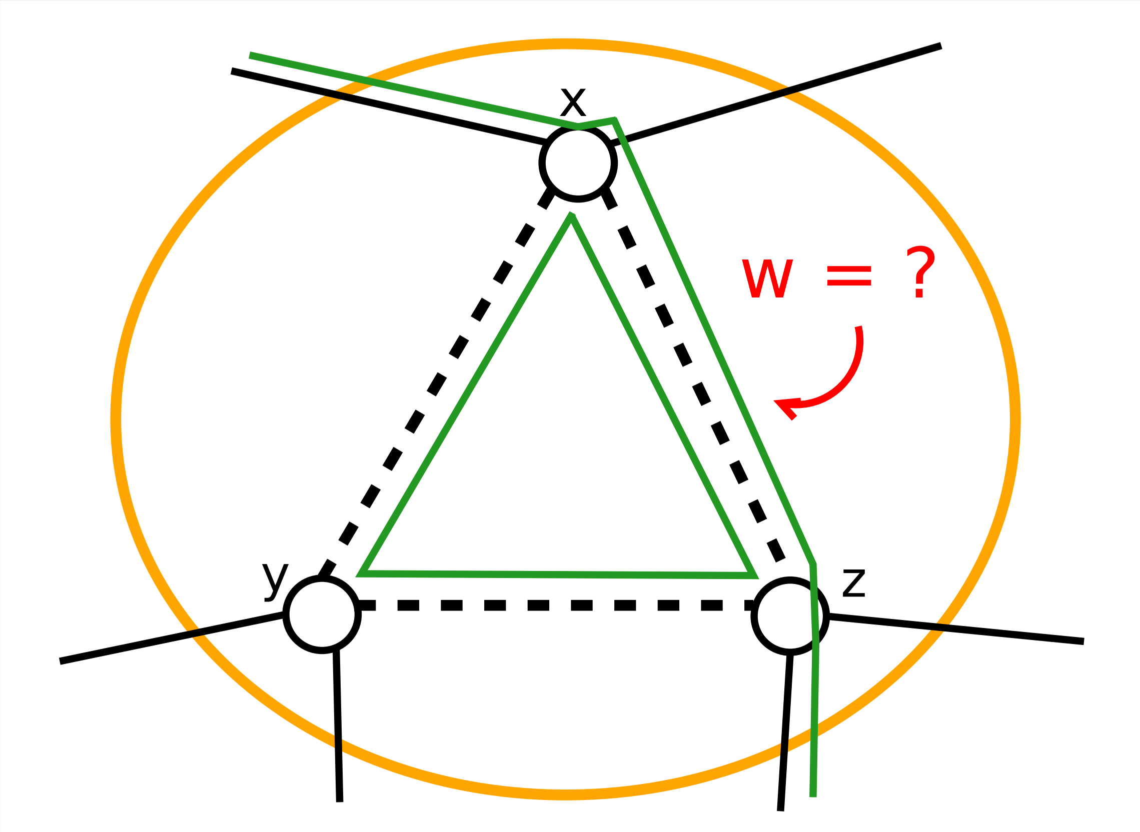


This corresponds to an unsatisfied clause

1in3 Gadget



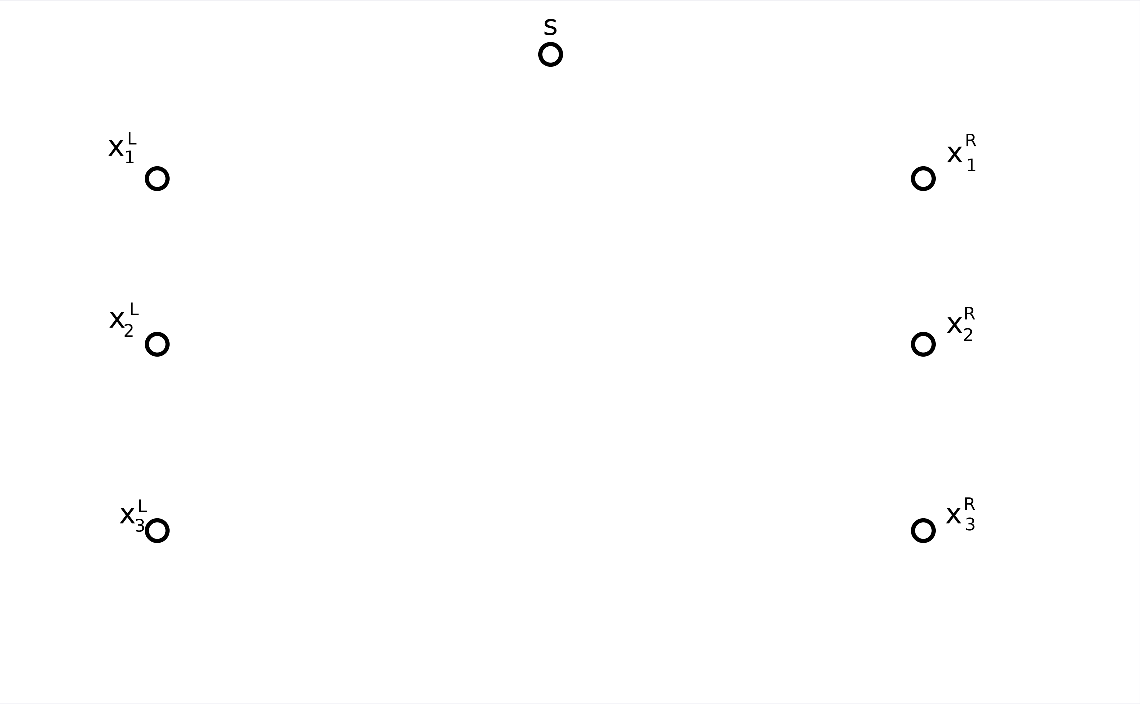
This corresponds to a dishonest tour



The dishonest tour pays this edge twice. How expensive must it be before cheating becomes suboptimal?

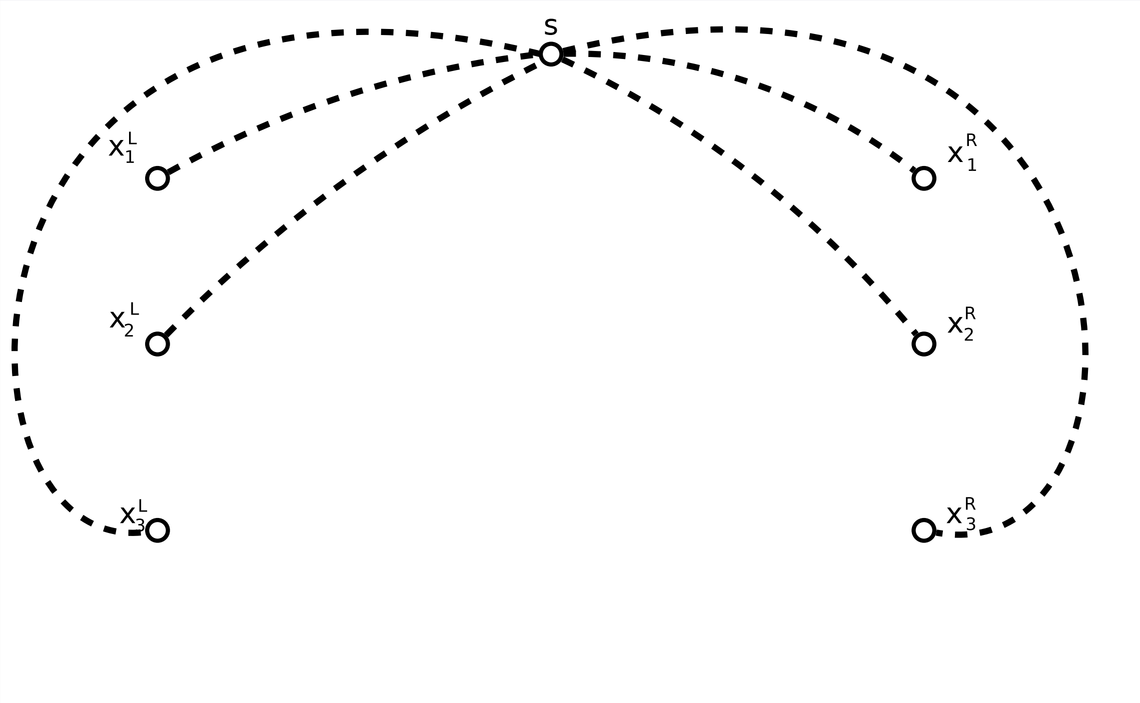
Note that $w = 10$ suffices, since the two cheating variables appear in at most 10 clauses.

Construction



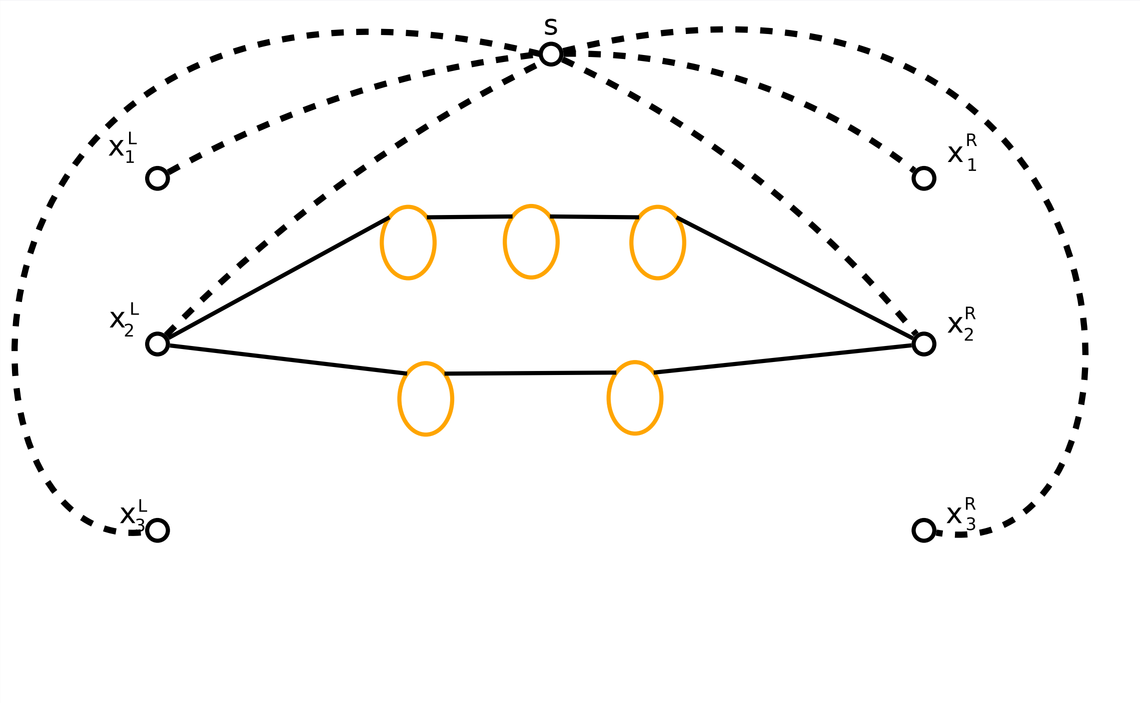
High-level view: construct an origin s and two terminal vertices for each variable.

Construction



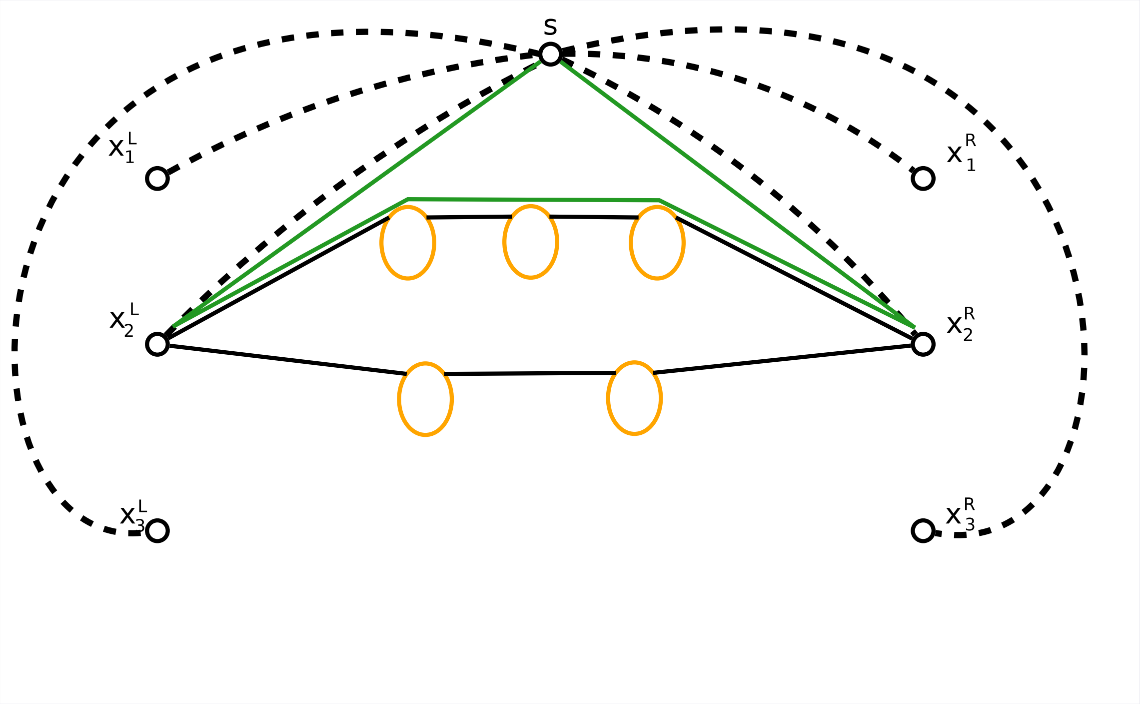
Connect them with forced edges

Construction



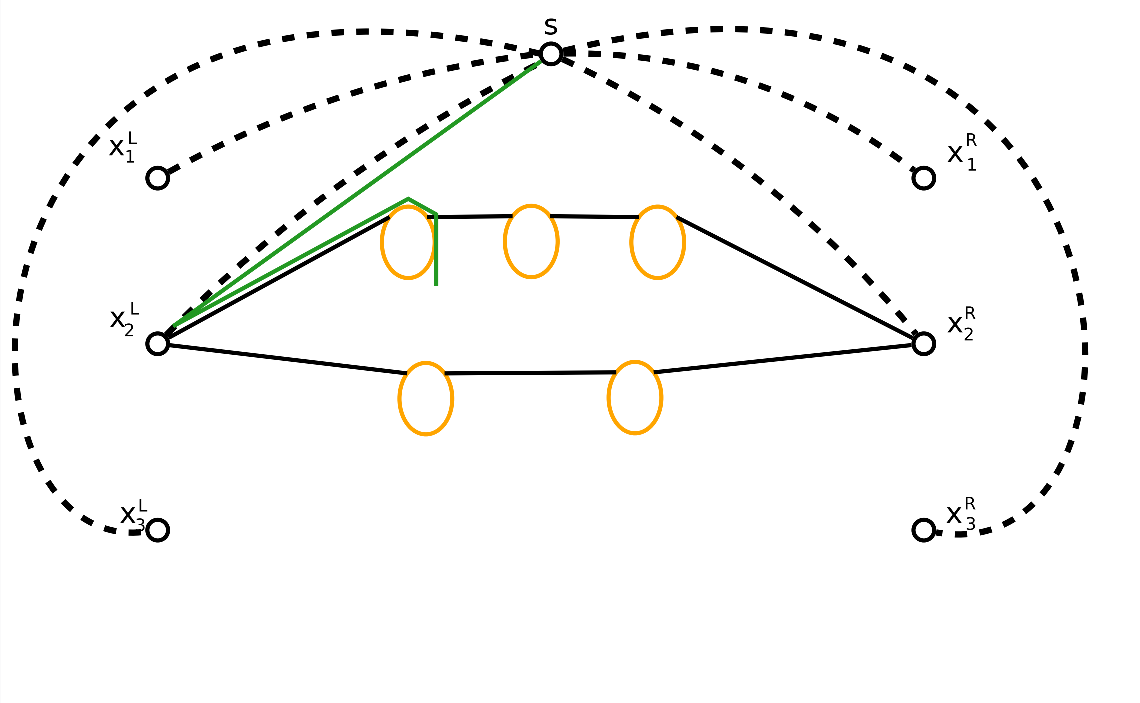
Add the gadgets

Construction



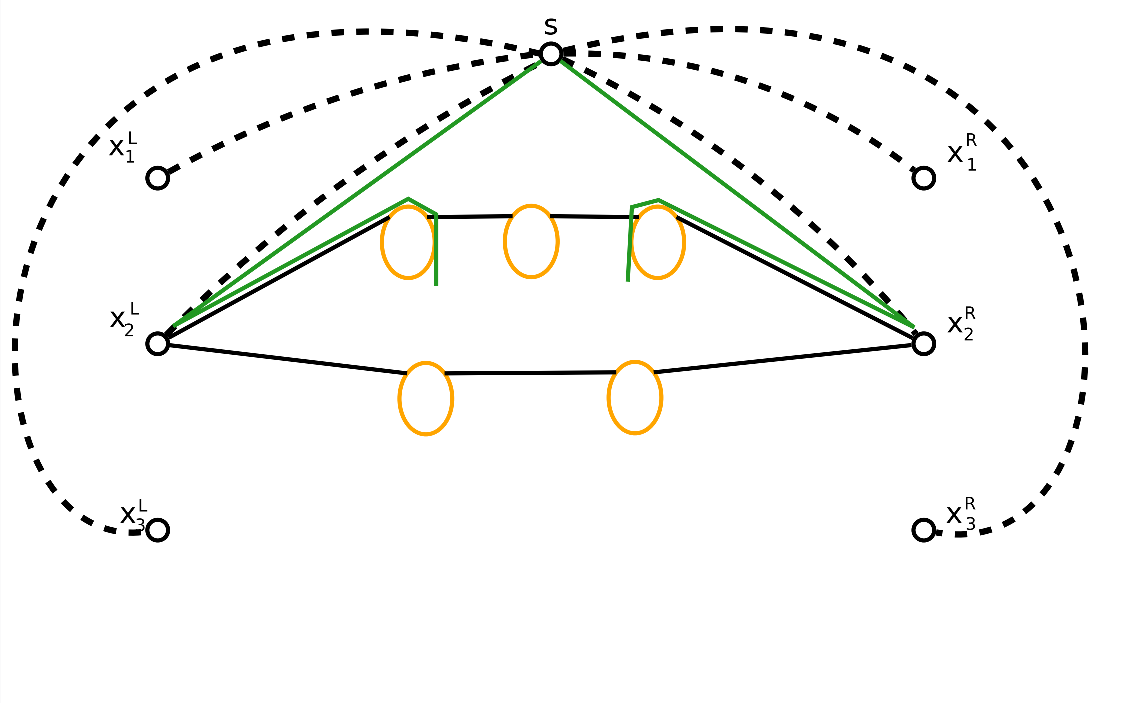
An honest traversal for x_2 looks like this

Construction



A dishonest traversal looks like this. . .

Construction

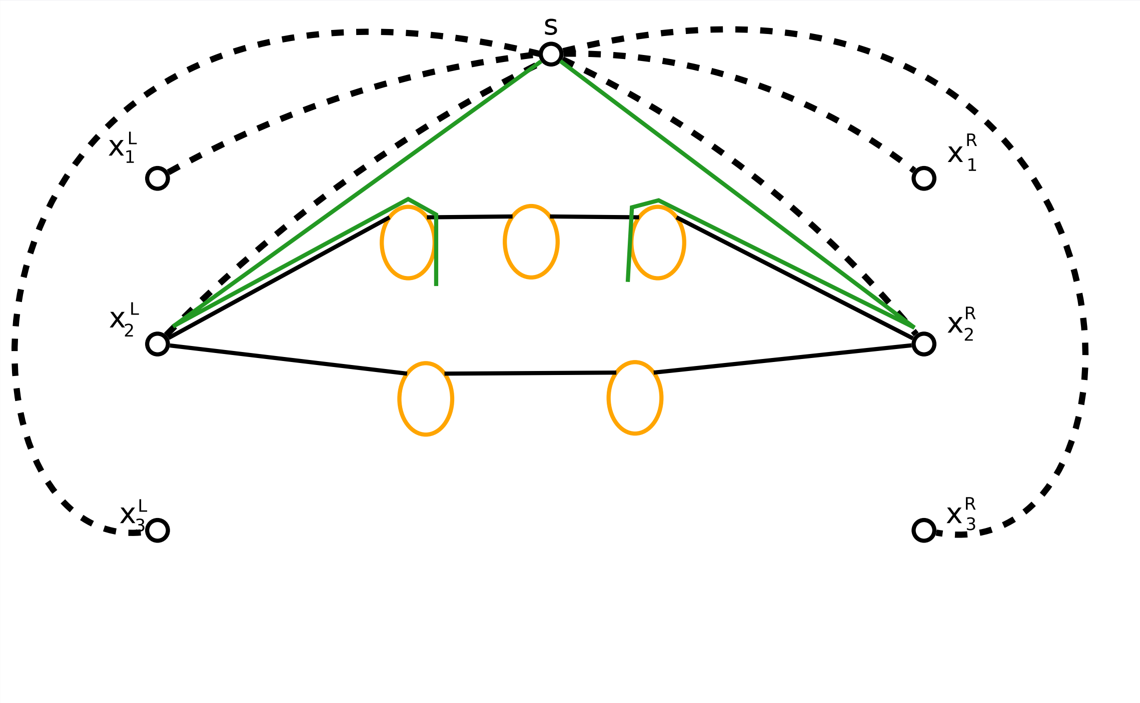


...but there must be cheating in two places

There are as many doubly-used forced edges as affected variables

$\rightarrow w \leq 5$

Construction



. . . but there must be cheating in two places

There are as many doubly-used forced edges as affected variables
$\to w \leq 5$

In fact, no need to write off affected clauses. Use random assignment for cheated variables and some of them will be satisfied

Under the carpet

- Many details missing
  - Dishonest variables are set randomly but not independently to ensure that some clauses are satisfied with probability 1.
  - The structure of the instance (from BK amplifier) must be taken into account to calculate the final constant.

Under the carpet

- Many details missing
  - Dishonest variables are set randomly but not independently to ensure that some clauses are satisfied with probability 1.
  - The structure of the instance (from BK amplifier) must be taken into account to calculate the final constant.

Theorem:
There is no $\frac{185}{184}$ approximation algorithm for TSP, unless P=NP.

Conclusions – Open problems

- A simpler reduction for TSP and a better inapproximability threshold
  - But, constant still very low!

Future work

- Better amplifier constructions?
- Application for improved expanders?
- ATSP

Conclusions – Open problems

- A simpler reduction for TSP and a better inapproximability threshold
  - But, constant still very low!

Future work

- Better amplifier constructions?
- Application for improved expanders?
- ATSP
- . . . **Reasonable** inapproximability for TSP?

The end

Questions?