# A Constant-Factor Approximation Algorithm for the Asymmetric Traveling Salesman Problem

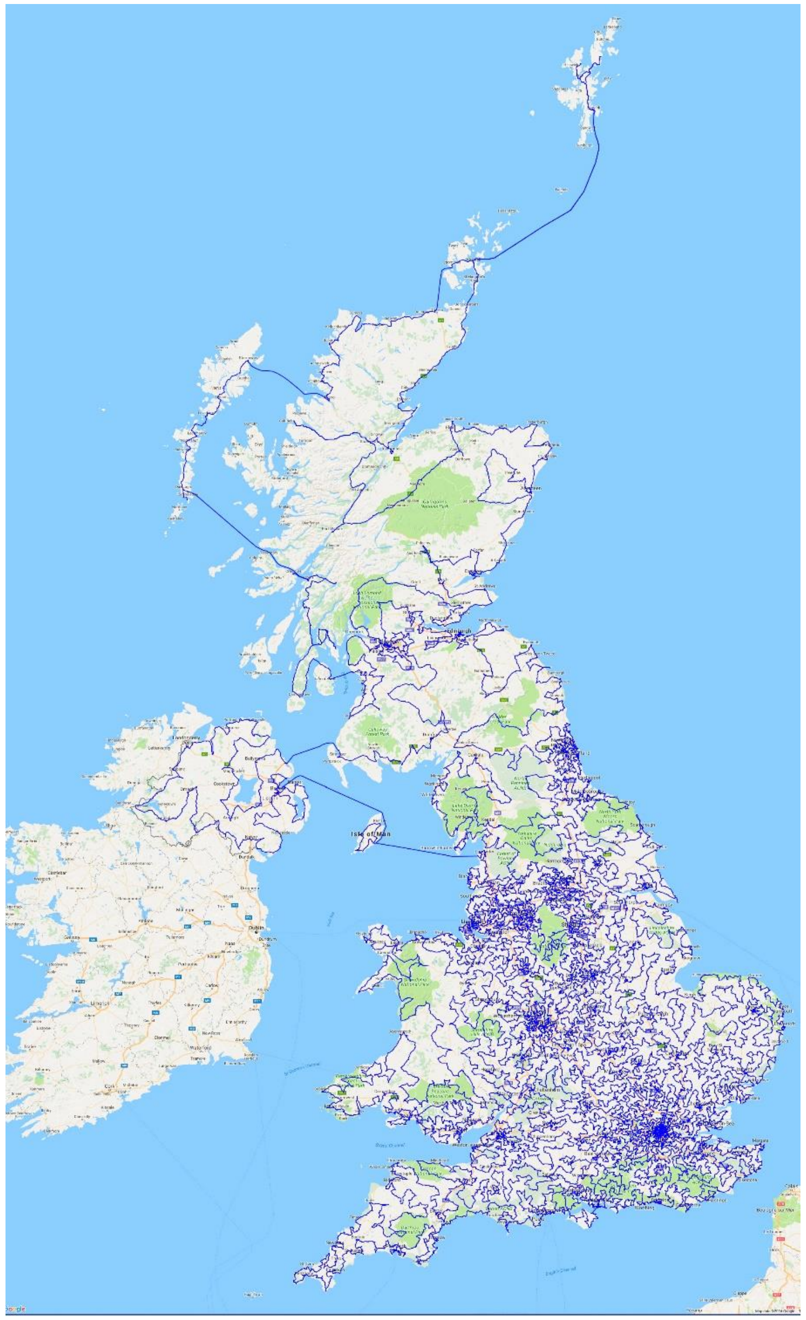
Ola Svensson, *Jakub Tarnawski* and László A. Végh

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

THE LONDON SCHOOL
OF ECONOMICS AND
POLITICAL SCIENCE

# What's the cheapest way to visit all 24727 pubs in the UK?

45,495,239 meters

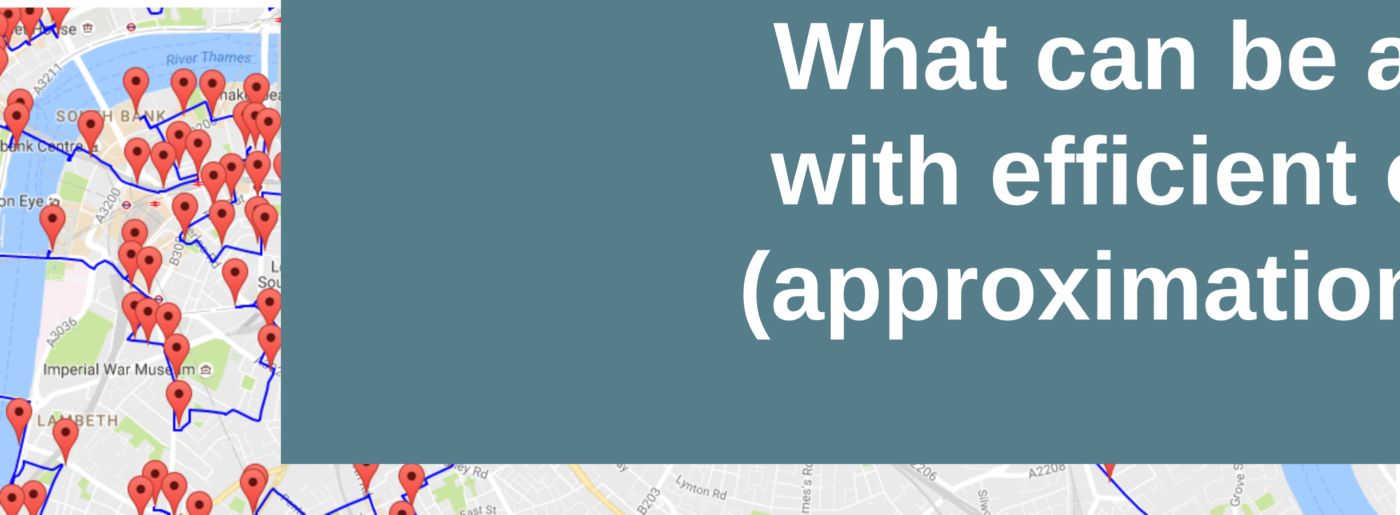


*Cook, Espinoza, Goycoolea, Helsgaun (2015)*

# Find the shortest tour that visits *n* given cities

# Traveling Salesman Problem

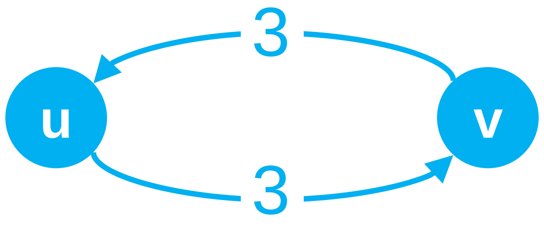
- Variants studied in mathematics by Hamilton and Kirkman already in the 1800’s
- Benchmark problem:
  - one of the most studied NP-hard optimization problems
  - yet our understanding is quite incomplete



**What can be accomplished with efficient computation (approximation algorithms)?**

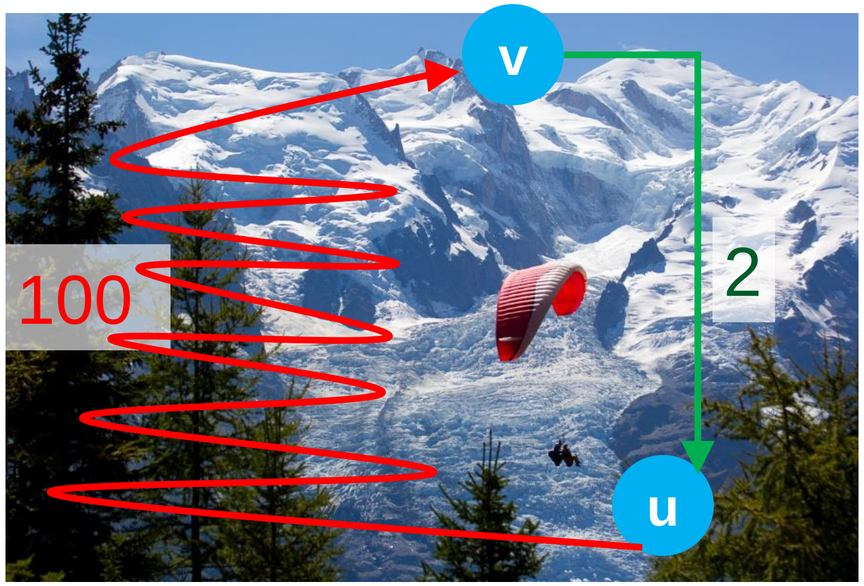
# Two basic versions

**Symmetric:** distance(u,v) = distance(v,u)



2-approximation is trivial
1.5-approximation [Christofides'76] taught in undergrad courses, still unbeaten

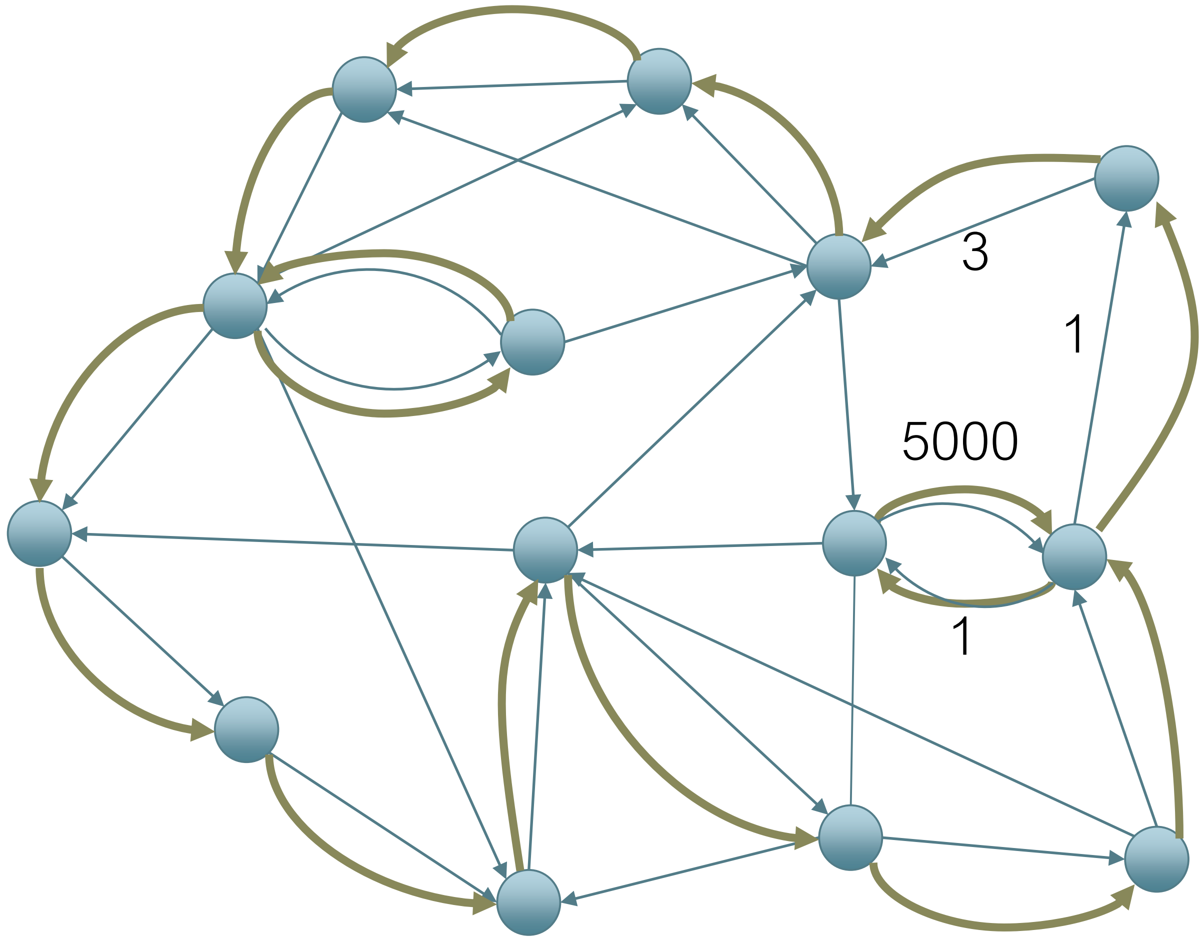
**Asymmetric:** more general, no such assumption is made



# Asymmetric Traveling Salesman Problem

**Input:** an edge-weighted digraph $G = (V, E, w)$

**Output:** a minimum-weight tour that visits each vertex at least once

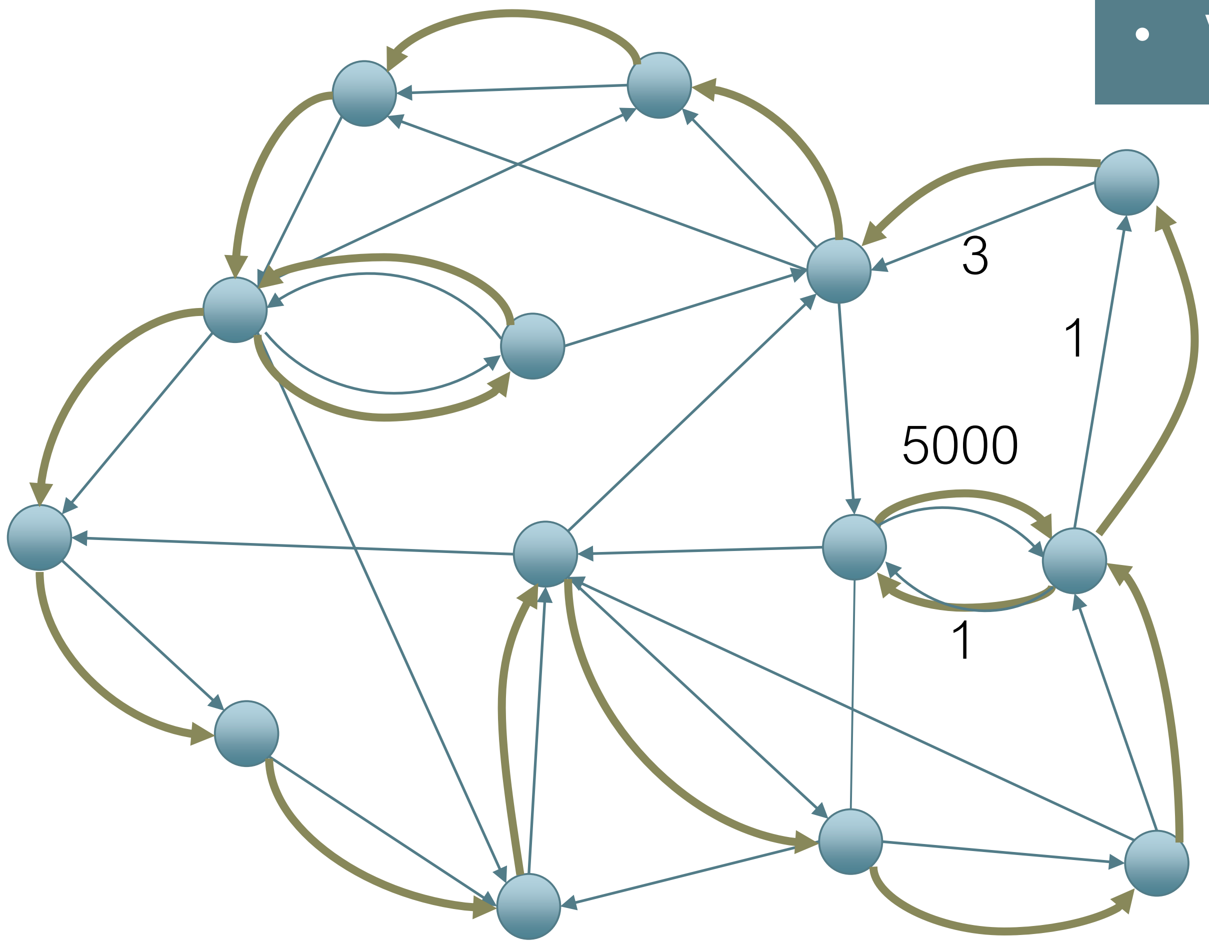


# Asymmetric Traveling Salesman Problem

**Input:** an edge-weighted digraph $G = (V, E, w)$

**Output:** a minimum-weight tour that visits each vertex at least once

Equivalently could have:
- Complete graph with Δ-inequality
- Visit each vertex *exactly* once

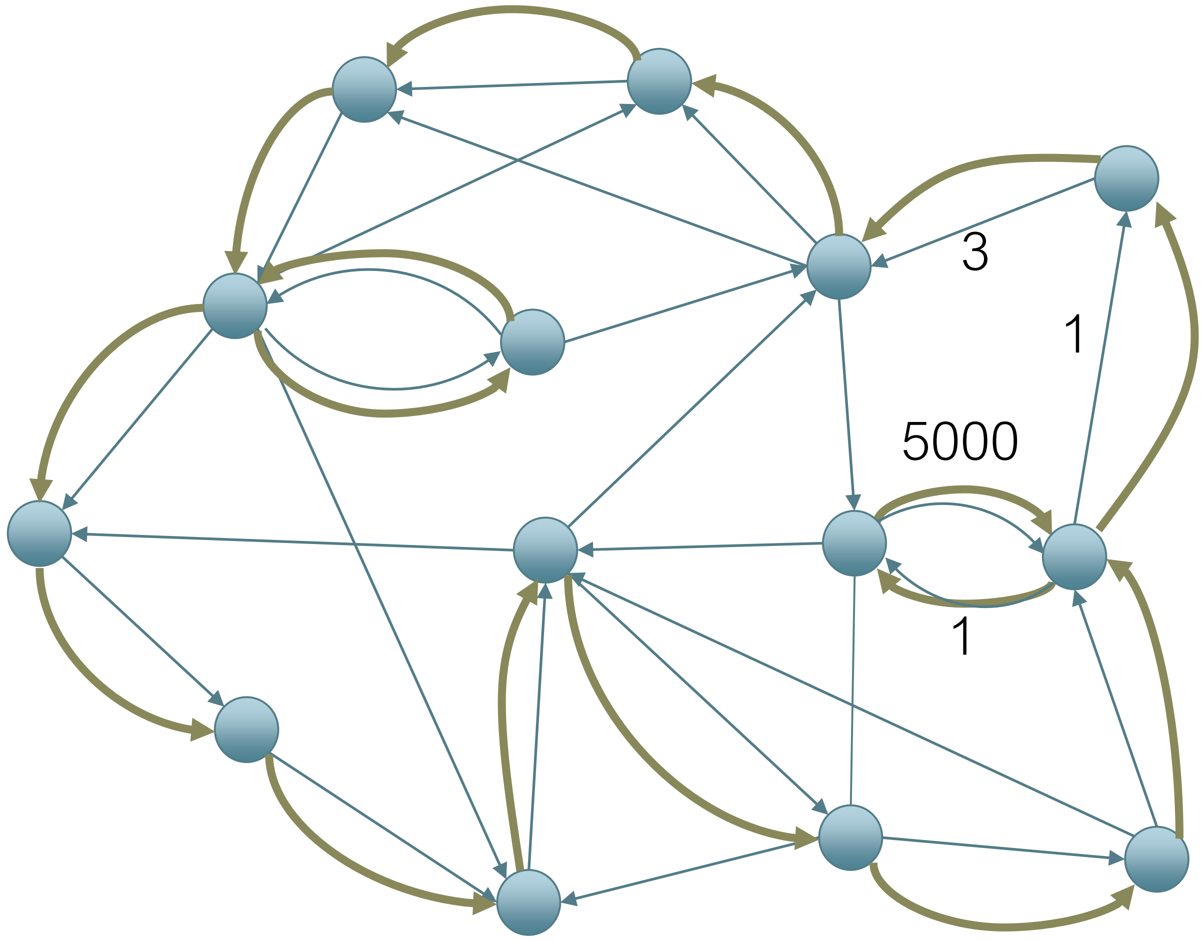


# Asymmetric Traveling Salesman Problem

**Input:** an edge-weighted digraph $G = (V, E, w)$

**Output:** a minimum-weight connected Eulerian multigraph $(V, E')$

in-degree = out-degree



# Asymmetric Traveling Salesman Problem

**Input:** an edge-weighted digraph $G = (V, E, w)$

**Output:** a minimum-weight connected Eulerian multigraph $(V, E')$

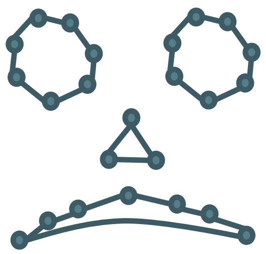
*Variables:* $x_{uv}$ = #times we traverse edge $(u, v)$

in-degree = out-degree

*Minimize:* $\sum_{uv \in E} w(u,v)\, x_{uv}$

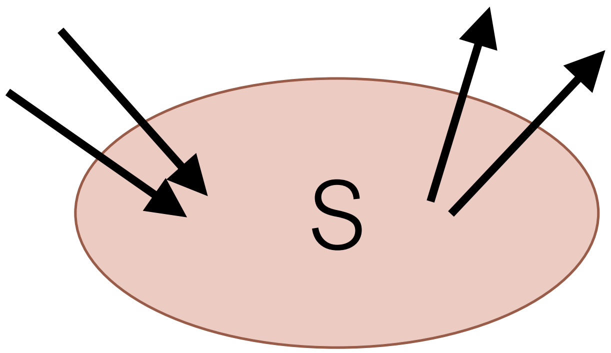
*Subject to:* $\boxed{x(\delta^+(v)) = x(\delta^-(v))}$ for all $v \in V$

$\boxed{x(\delta(S)) \geq 2}$ for all $S \subset V$



$x \geq 0$

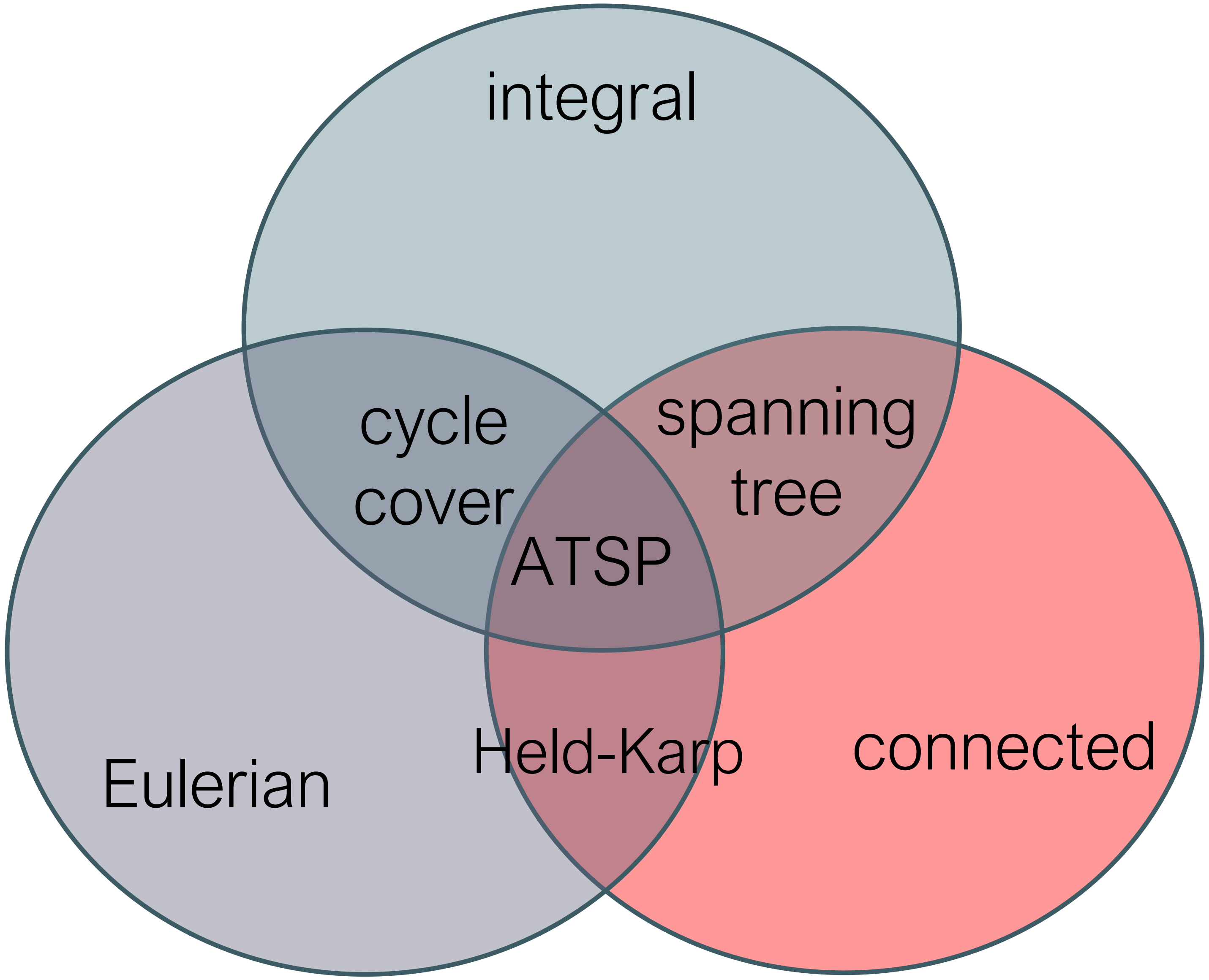
$\delta(S)$ = set of cut edges



# Integrality gap of the

i.e. how far off is that particular algorithm?

# Pick any two…



integral
cycle cover
spanning tree
ATSP
Held-Karp
Eulerian
connected

# Two natural approaches: begin with…

**Output:** a minimum-weight ~~**connected**~~ ~~**Eulerian**~~ multigraph

# Add Eulerian graphs until connected

$\log_2 n$-approximation via repeated cycle covers
[Frieze, Galbiati, Maffioli'82]

$0.99 \log_2 n$-approximation
[Bläser'03]

$0.84 \log_2 n$-approximation
[Kaplan, Lewenstein, Shafrir, Sviridenko'05]

$0.67 \log_2 n$-approximation
[Feige, Singh'07]

# Local-Connectivity ATSP

- Defined new, easier problem
- Reduced $O(1)$-approximation of ATSP to it
- Solved it for unweighted graphs (easy part)

[Svensson'15]

…

Solved it for graphs with two edge weights
[Svensson, T., Vegh'16]

# Start with spanning tree, then make Eulerian

$O(\log n / \log \log n)$-approximation via thin trees
[Asadpour, Goemans, Mądry, Oveis Gharan, Saberi'10]

$O(1)$-approximation for planar & bounded-genus graphs
[Oveis Gharan, Saberi'11]

Integrality gap $\leq \text{poly}(\log \log n)$
via generalization of Kadison-Singer
[Anari, Oveis Gharan'14]

# Hardness

NP-hard to approximate within $1 + \frac{1}{74}$
[Papadimitriou, Vempala'00, Karpinski, Lampis, Schmied'13]

Integrality gap $\geq 2$
[Charikar, Goemans, Karloff'02]

# Theorem:

A $O(1)$-approximati algorithm with respe elaxation

2-edge-weights ATSP

general ATSP

structured ATSP

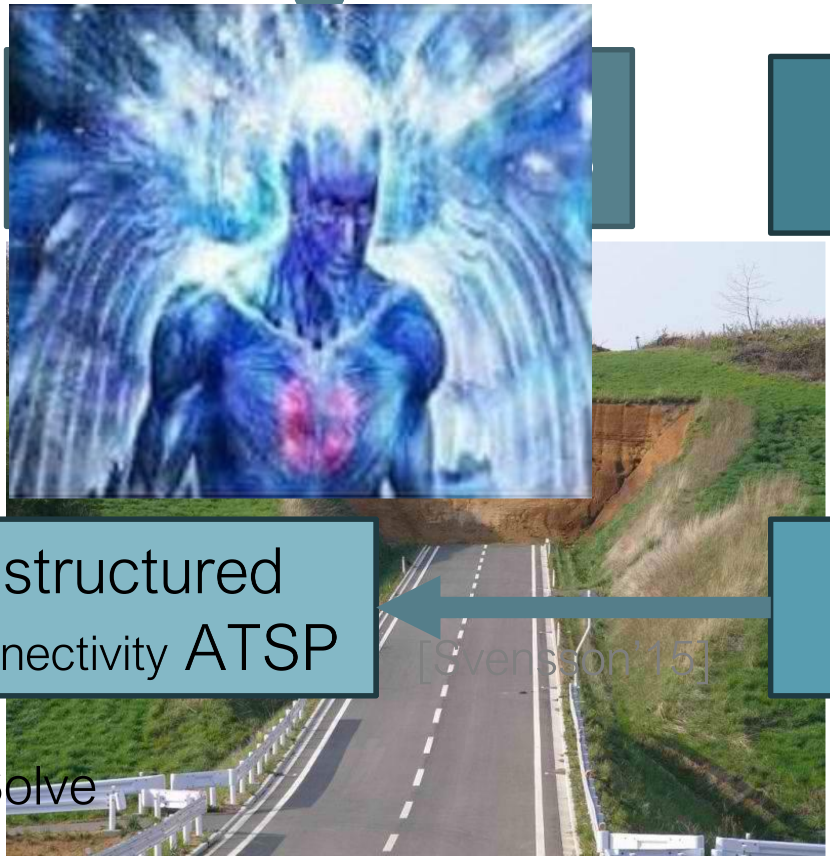
[Svensson'15]

[Svensson'15]

2-edge-weights Local-Connectivity ATSP

✓ Solve [Svensson, T., Vegh'16]

more structured ATSP



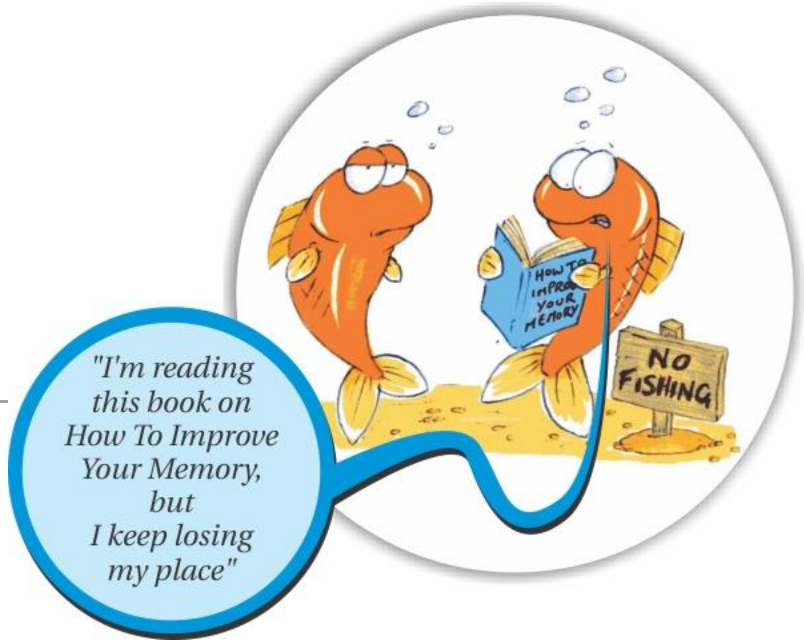
really structured Local-Connectivity ATSP

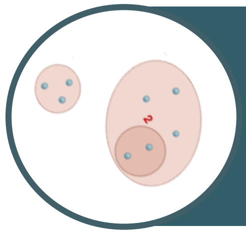
really structured ATSP

[Svensson'15]

✓ Solve

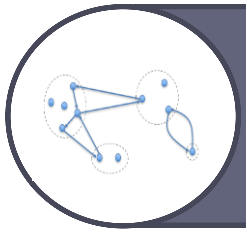
# Outline of reductions

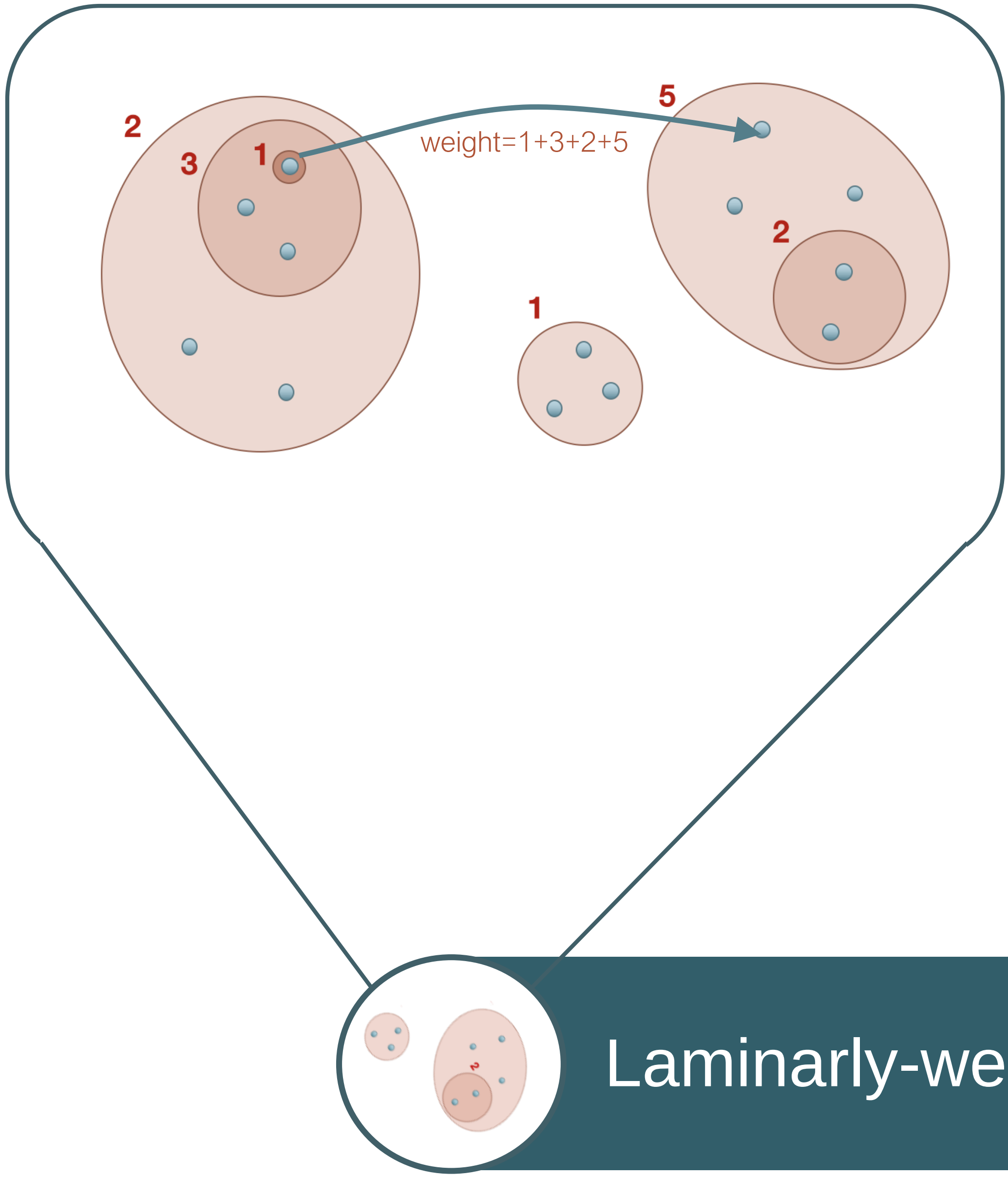


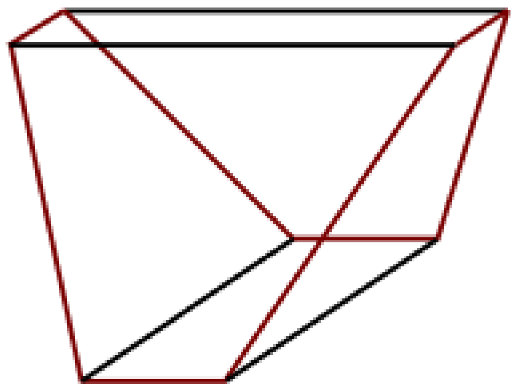

Laminarly-weighted instances


Irreducible instances


Vertebrate pairs


Solving Local-Connectivity ATSP





By amazing power of LP-duality

Laminarly-weighted instances

# Asymmetric Traveling Salesman Problem

**Input:** an edge-weighted digraph $G = (V, E, w)$

**Output:** a minimum-weight connected Eulerian multigraph $(V, E')$

*Variables:* $x_{uv}$ = #times we traverse edge $(u, v)$

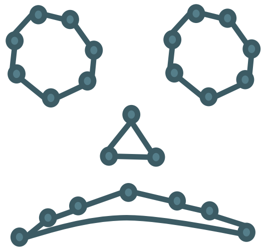
in-degree = out-degree

*Minimize:* $\sum_{uv \in E} w(u,v)\, x_{uv}$

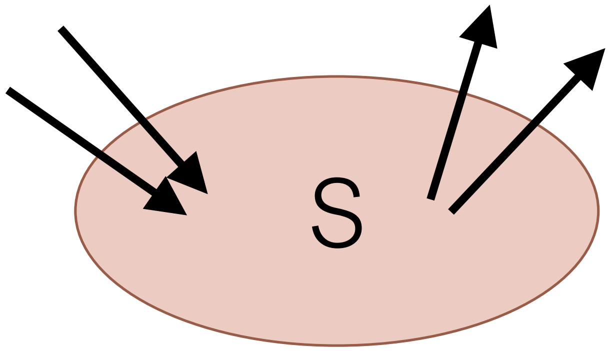
*Subject to:* $\boxed{\boldsymbol{x}\left(\boldsymbol{\delta}^{+}(\boldsymbol{v})\right) = \boldsymbol{x}\left(\boldsymbol{\delta}^{-}(\boldsymbol{v})\right)}$ for all $v \in V$

$\boxed{\boldsymbol{x}\left(\boldsymbol{\delta}(\boldsymbol{S})\right) \geq \mathbf{2}}$ for all $S \subset V$

$x \geq 0$



$\delta(S)$ = set of cut edges



*Minimize:* $\sum_{uv \in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x\big(\delta^{+}(v)\big) = x(\delta^{-}(v))}$ for all $v \in V$

$\boldsymbol{x\big(\delta(S)\big) \geq 2}$ for all $S \subset V$

$x \geq 0$

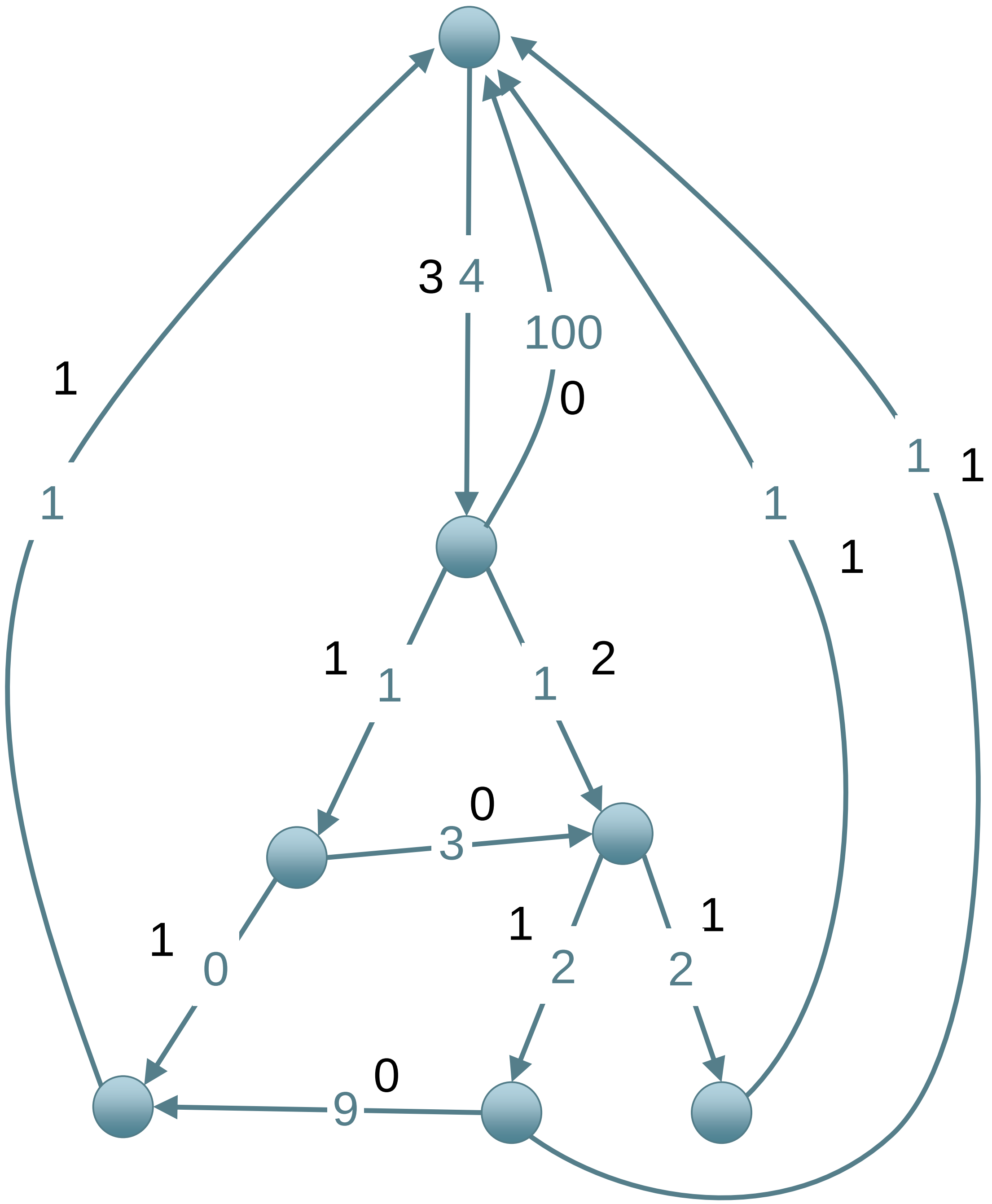
*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x\left(\delta^{+}(v)\right) = x\left(\delta^{-}(v)\right)$ for all $v \in V$

$x\left(\delta(S)\right) \geq 2$ for all $S \subset V$

$x \geq 0$

LP-value = 22



1. Solve LP to obtain solution depicted in black
2. Forget edges with LP-value = 0
    - Doesn’t change LP-value
    - Any tour is smaller instance is a tour in original instance
3. Now all edges have positive LP-value

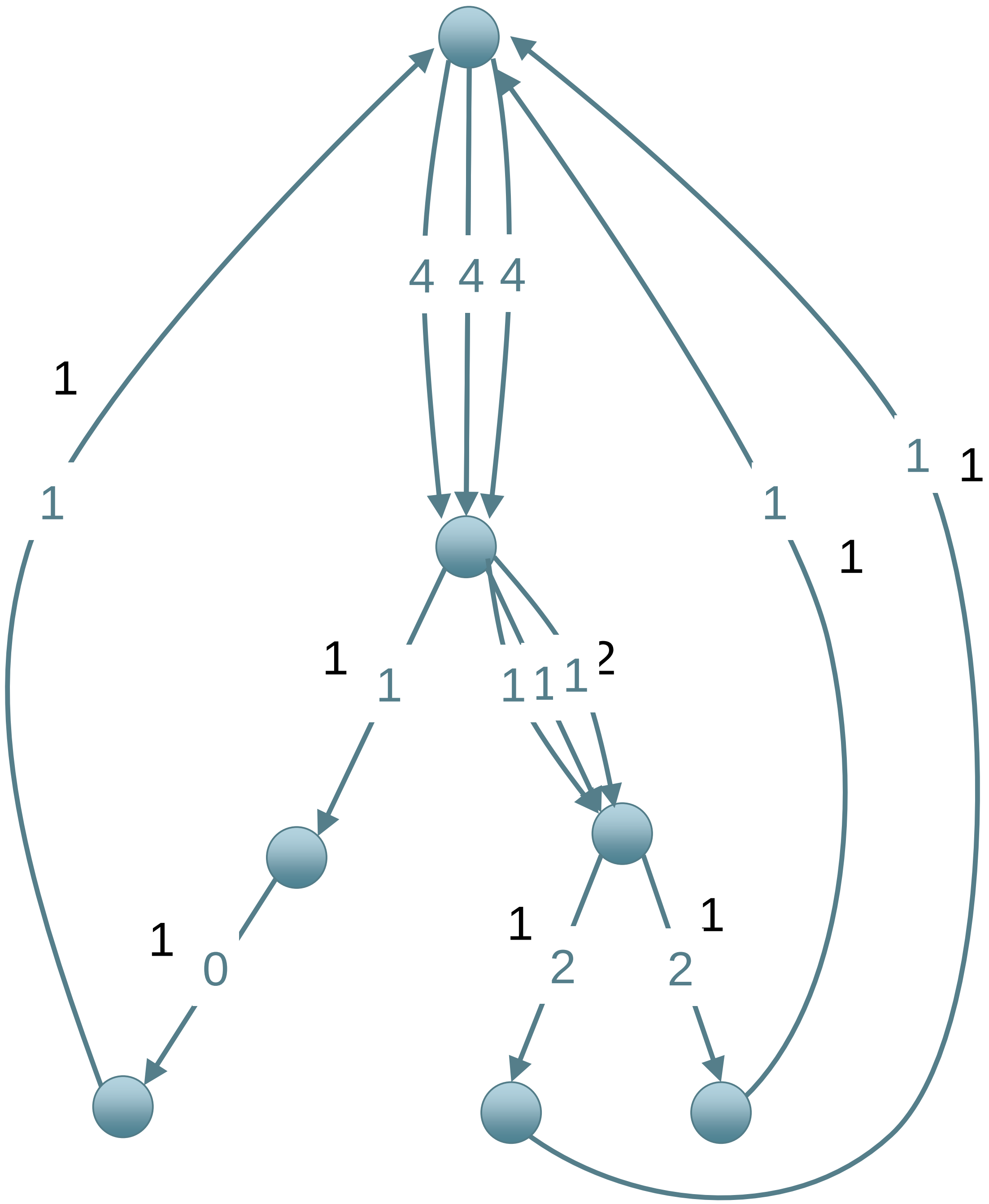
*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^{+}(v)) = x(\delta^{-}(v))$ for all $v \in V$

$x(\delta(S)) \geq 2$ for all $S \subset V$

$x \geq 0$

LP-value = 22



1. Solve LP to obtain solution depicted in black
2. Forget edges with LP-value = 0
    - Doesn't change LP-value
    - Any tour is smaller instance is a tour in original instance
3. Now all edges have positive LP-value

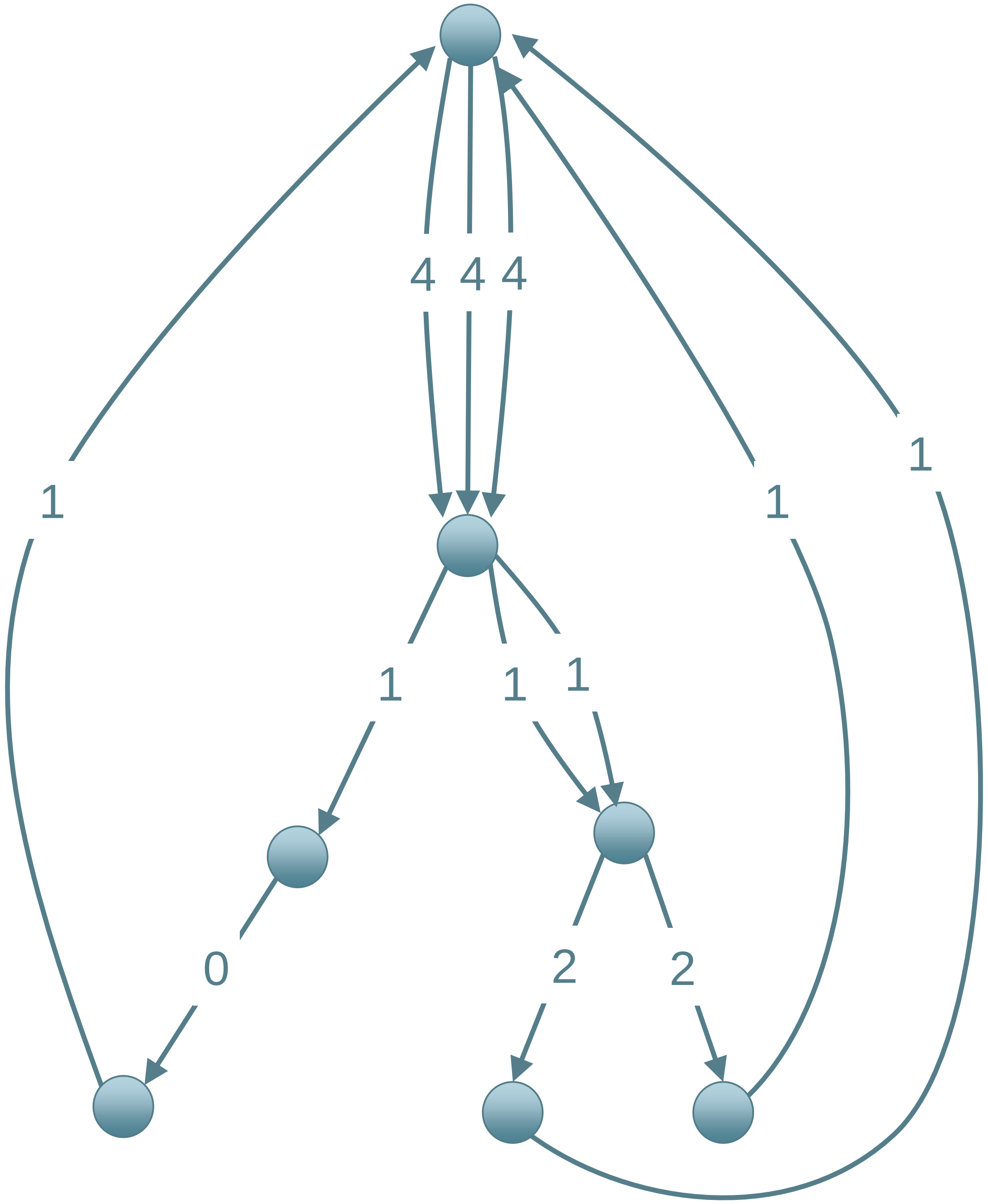
*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$

$x(\delta(S)) \geq 2$ for all $S \subset V$

$x \geq 0$

LP-value = 22



1. Solve LP to obtain solution depicted in black
2. Forget edges with LP-value = 0
   - Doesn't change LP-value
   - Any tour is smaller instance is a tour in original instance
3. Now all edges have positive LP-value

Do these edges have structure?

By complementarity slackness, each remaining edge corresponds to tight constraint in **dual**

Minimize: $\sum_{uv\in E} w(u,v)\, x_{uv}$

Subject to: $\boldsymbol{x}(\boldsymbol{\delta}^{+}(\boldsymbol{v})) = \boldsymbol{x}(\boldsymbol{\delta}^{-}(\boldsymbol{v}))$ for all $v \in V$

$\boldsymbol{x}(\boldsymbol{\delta}(\boldsymbol{S})) \geq \boldsymbol{2}$ for all $S \subset V$

$x \geq 0$

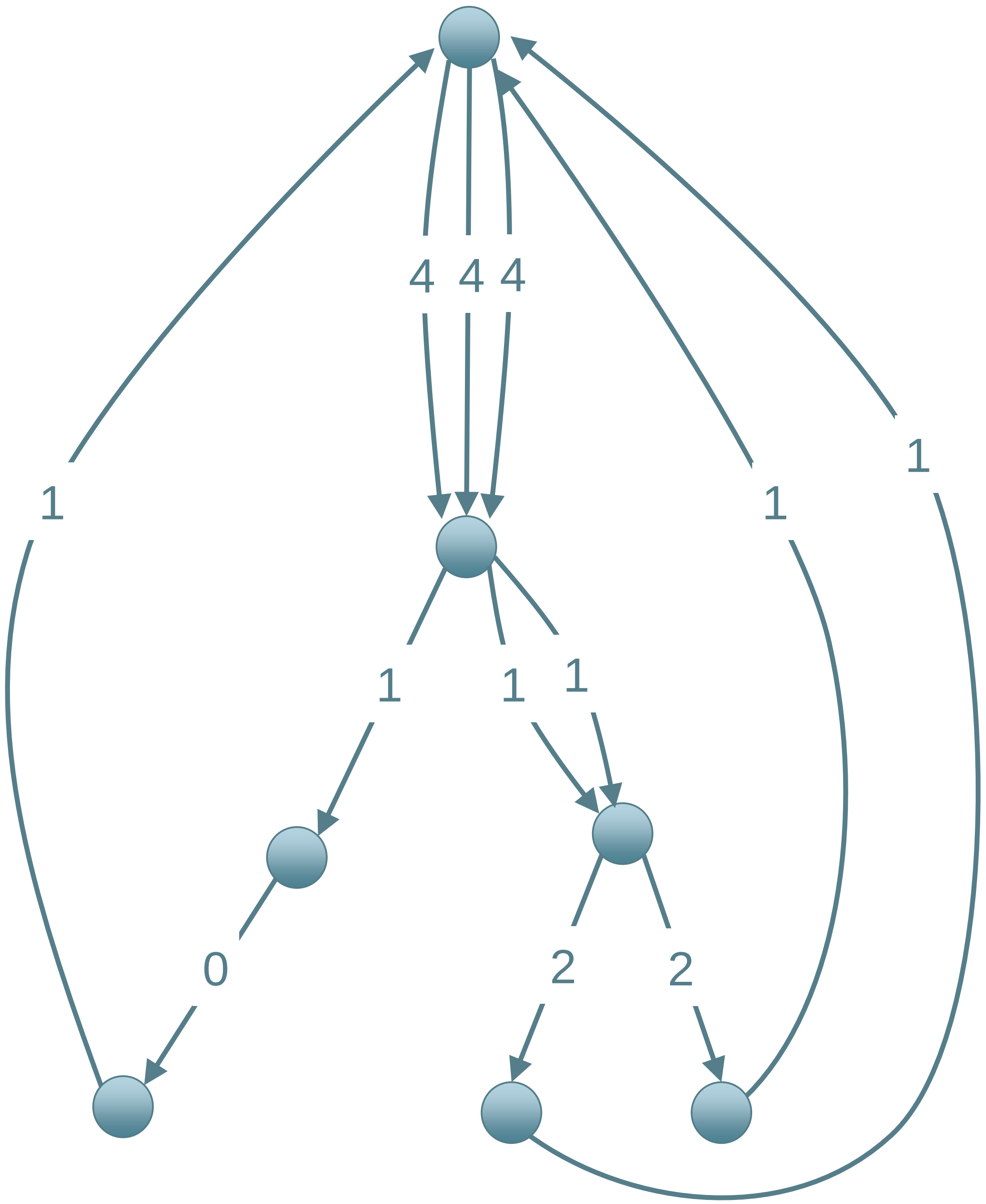
Maximize: $\sum_{S\subset V} 2 \cdot y_S$

Subject to:

$\boxed{\sum_{\boldsymbol{S}:(\boldsymbol{u},\boldsymbol{v})\in\boldsymbol{\delta}(\boldsymbol{S})} \boldsymbol{y_S} + \boldsymbol{\alpha_u} - \boldsymbol{\alpha_v} \leq \boldsymbol{w}(\boldsymbol{u},\boldsymbol{v})}$ for all $(\mathrm{u}, \mathrm{v}) \in E$

$\mathrm{y} \geq 0$

LP-value = 22



Sum of y-values cutting (u,v)
+ tail potential
- head potential
is at most the edge-weight

Dual has variables:

- $\alpha_v$ - vertex potential for each v
- $y_S$ - value for each cut $S$

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x(\delta^{+}(v)) = x(\delta^{-}(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

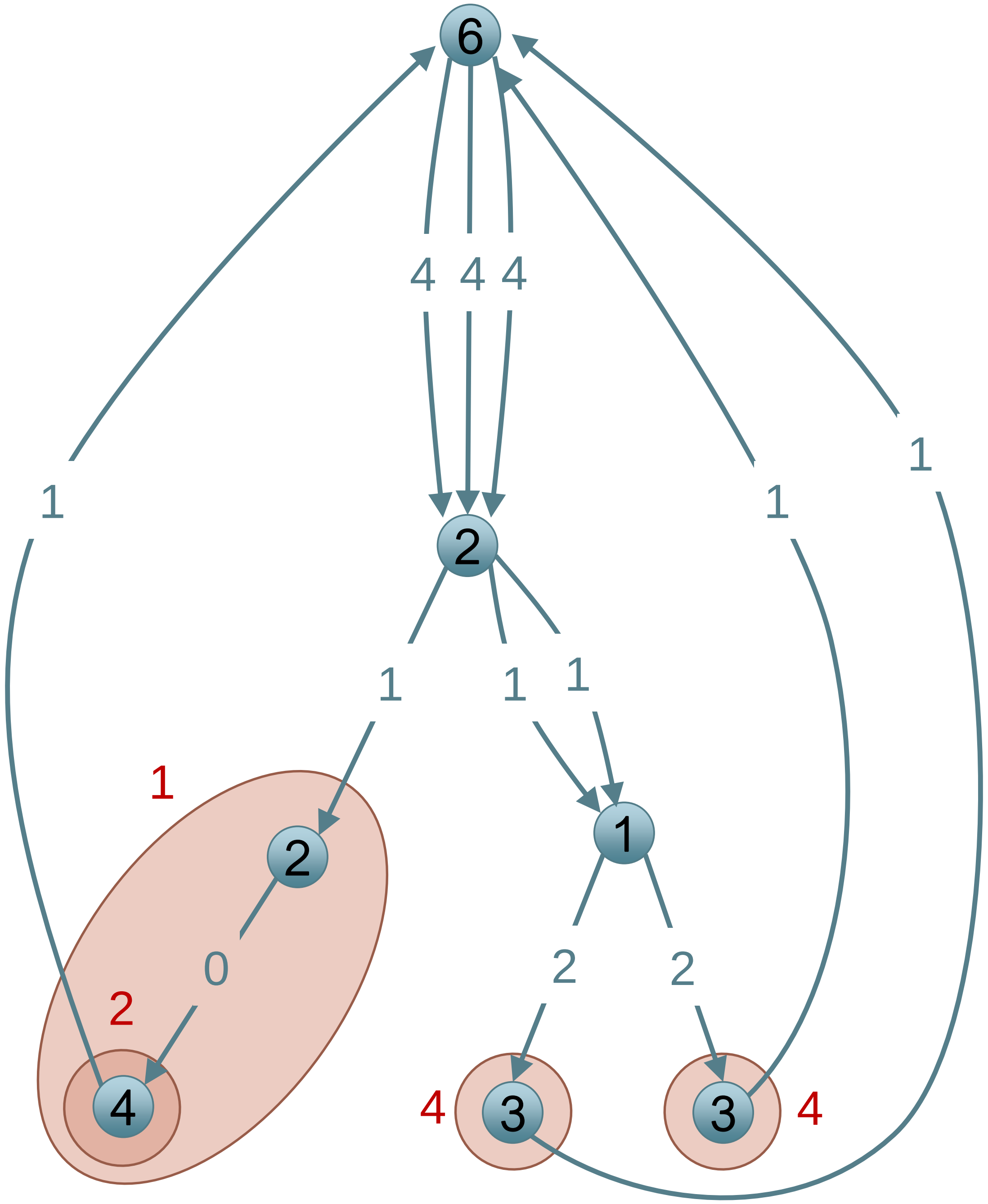
*Subject to:*

$\boxed{\boldsymbol{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)}}$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

Sum of y-values cutting (u,v)
+ tail potential
- head potential
is at most the edge-weight

Dual value = LP-value = 22



Dual has variables:

- $\alpha_v$ - vertex potential for each v
- $y_S$ - value for each cut $S$

*Minimize:* $\sum_{uv \in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x(\delta^{+}(v)) = x(\delta^{-}(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S \subset V} 2 \cdot y_S$

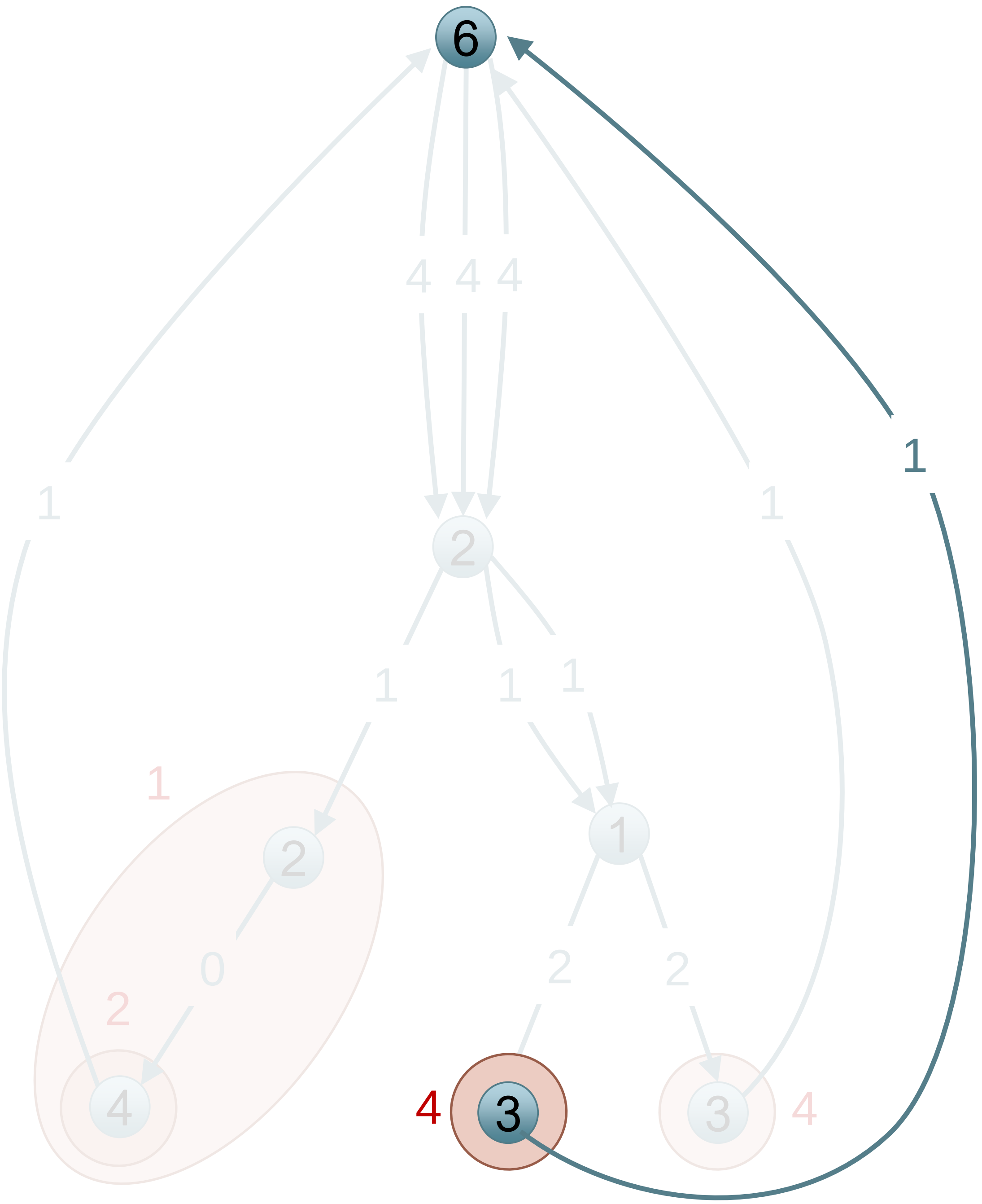
*Subject to:*

$\boxed{\boldsymbol{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)}}$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

Sum of y-values cutting (u,v)
+ tail potential
- head potential
is at most the edge-weight

Dual value = LP-value = 22



Dual has variables:

- $\alpha_v$ - vertex potential for each v
- $y_S$ - value for each cut $S$

$$\sum_{S:(u,v)\in\delta(S)} \boldsymbol{y_S + \alpha_u - \alpha_v \leq w(u,v)}$$

4 + 3 - 6 = 1

*Minimize:* $\sum_{uv \in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$

$x(\delta(S)) \geq 2$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S \subset V} 2 \cdot y_S$

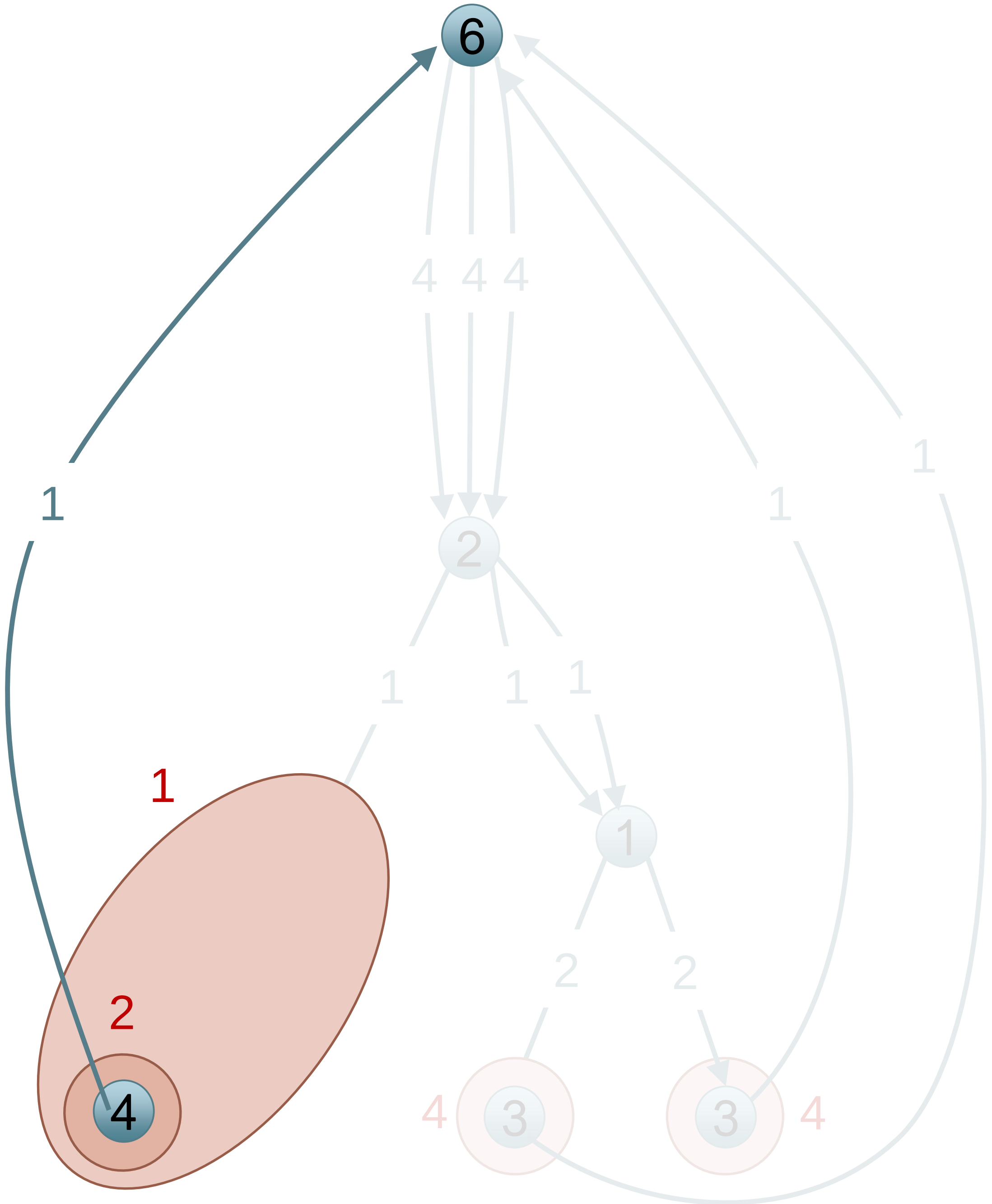
*Subject to:*

$\boxed{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)}$ for all $(u,v) \in E$

$y \geq 0$

Sum of y-values cutting (u,v)
+ tail potential
- head potential
is at most the edge-weight

Dual value = LP-value = 22



Dual has variables:

- $\alpha_v$ - vertex potential for each v
- $y_S$ - value for each cut $S$

$$\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)$$

2+1 + 4 - 6 = 1

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^{+}(v)) = x(\delta^{-}(v))$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

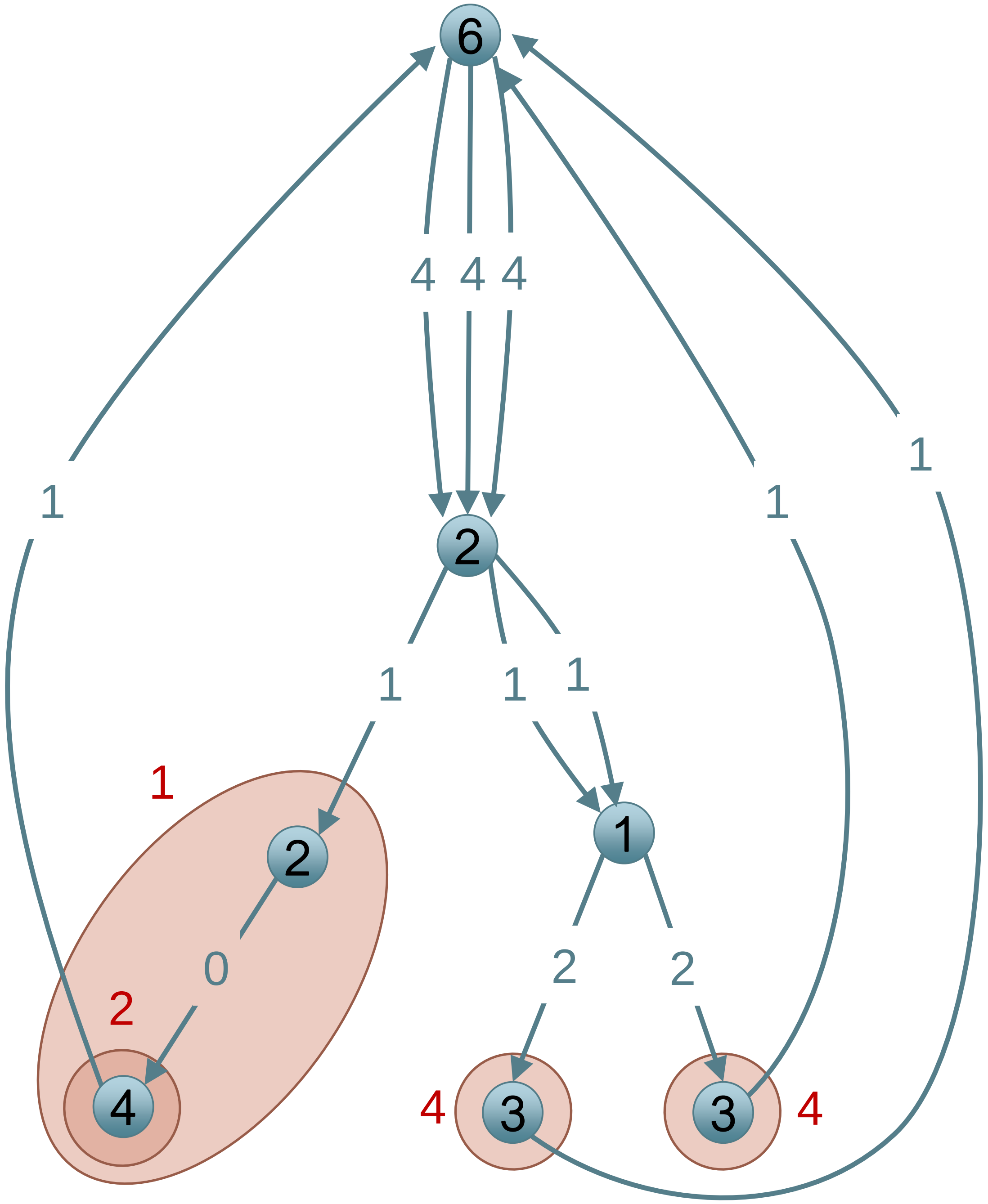
*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

*Subject to:*

$$\sum_{S:(u,v)\in\delta(S)} \boldsymbol{y_S + \alpha_u - \alpha_v \leq w(u,v)} \quad \text{for all } (u,v) \in E$$

$$y \geq 0$$

Dual value = LP-value = 22



By complementarity slackness:

$$\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v = w(u,v)$$

for every edge (u,v) *(since we only kept positive edges)*

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^{+}(v)) = x(\delta^{-}(v))$ for all $v \in V$

$x(\delta(S)) \geq 2$ for all $S \subset V$

$x \geq 0$

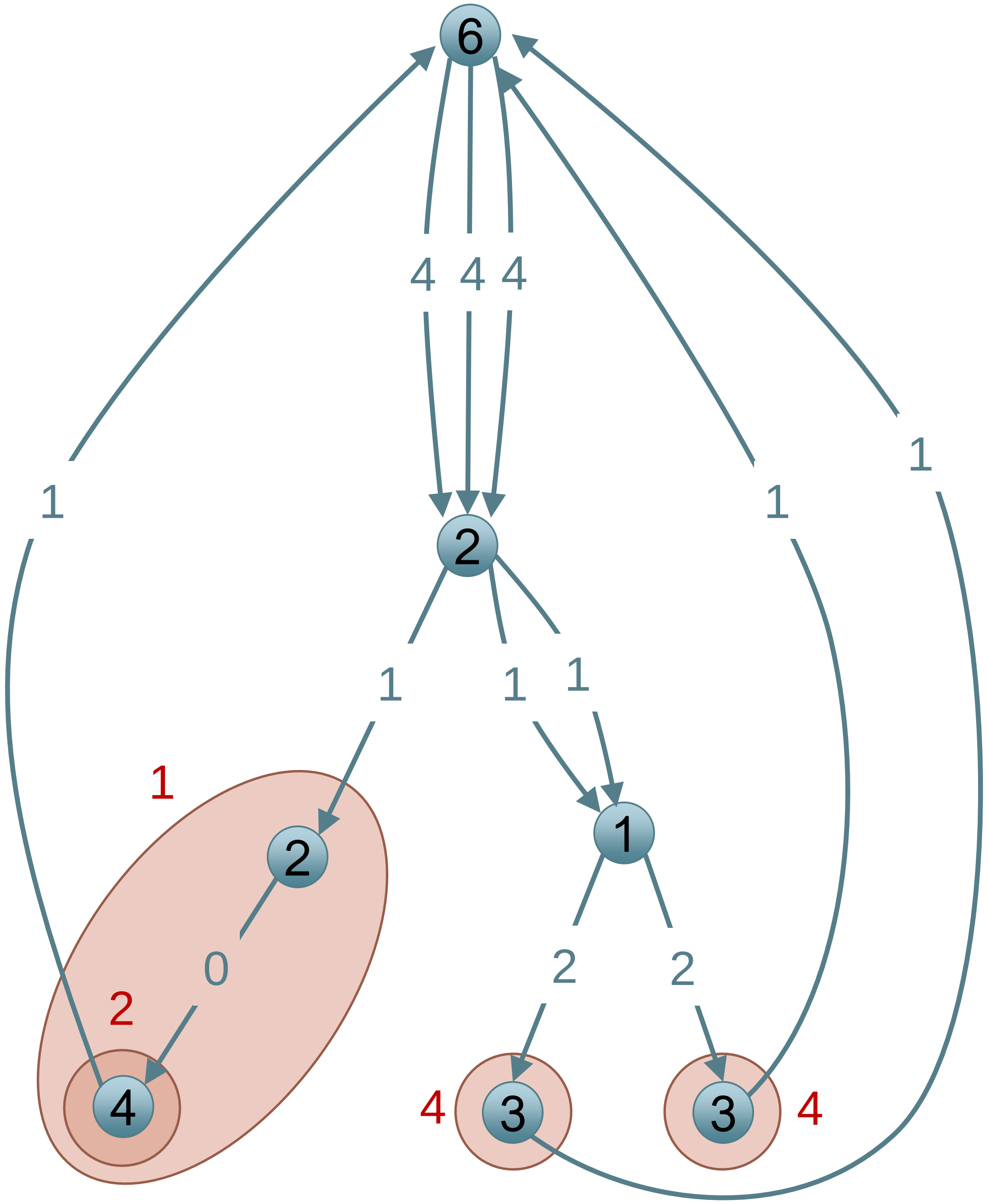
*Maximize:* $\sum_{S\subset V} 2\cdot y_S$

*Subject to:*

$\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)$ for all $(u,v) \in E$

$y \geq 0$

Dual value = LP-value = 22



By complementarity slackness:

$$\sum_{S:(u,v)\in\delta(S)} y_S = w(u,v) - \alpha_u + \alpha_v$$

for every edge (u,v) *(since we only kept positive edges)*

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x(\delta^+(v)) = x(\delta^-(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

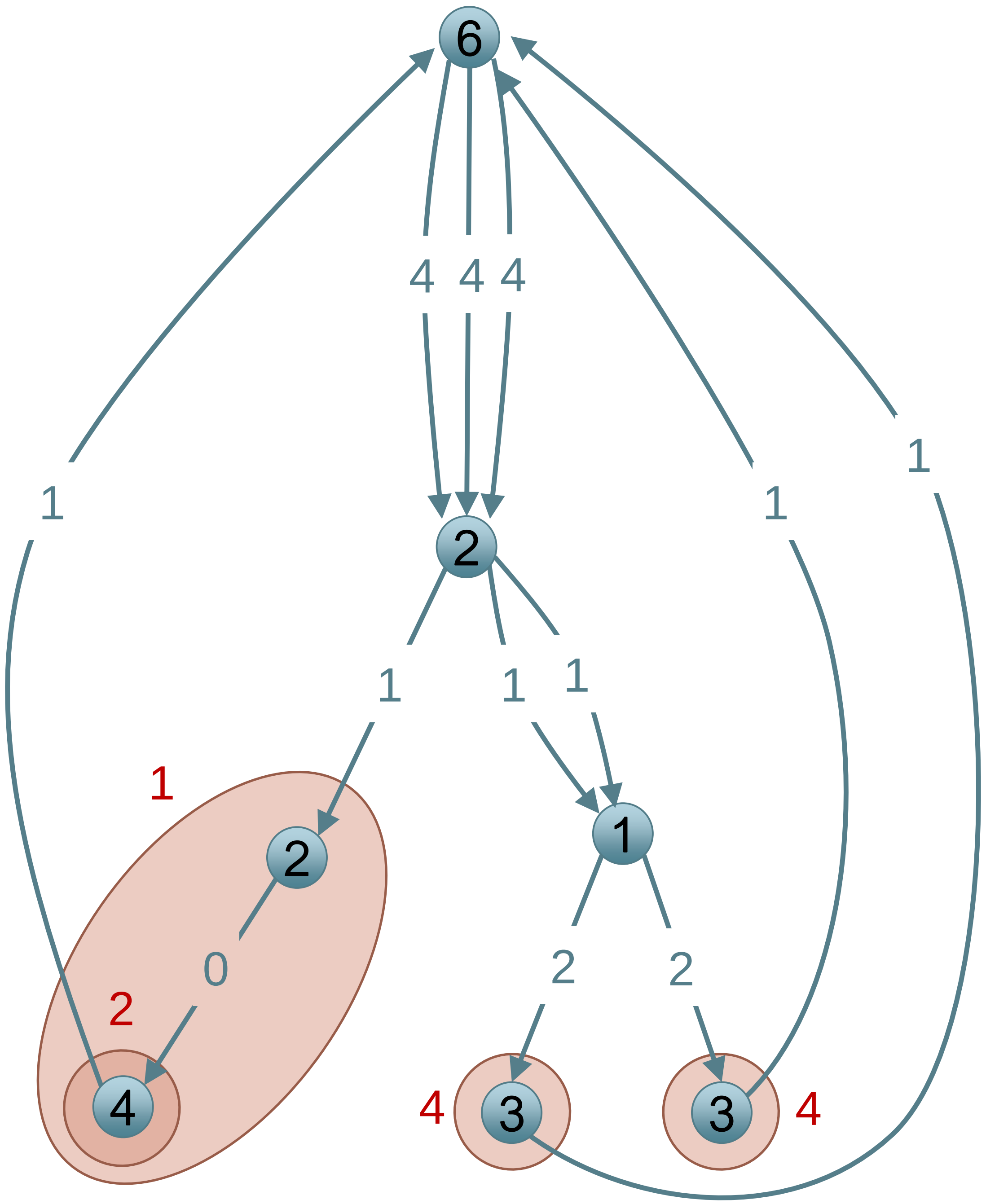
*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

*Subject to:*

$$\sum_{S:(u,v)\in\delta(S)} \boldsymbol{y_S + \alpha_u - \alpha_v \leq w(u,v)} \quad \text{for all } (u,v) \in E$$

$$y \geq 0$$

Dual value = LP-value = 22



By complementarity slackness:

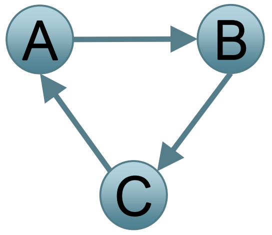
$$\sum_{S:(u,v)\in\delta(S)} y_S = w(u,v) - \alpha_u + \alpha_v =: w'(u,v)$$

for every edge (u,v) *(since we only kept positive edges)*

**Observation:**

For any Eulerian edge set F

$$w(F) = w'(F)$$



$$\begin{aligned} w'(F) &= w(A,B) + \alpha_A - \alpha_B \\ &+ w(B,C) + \alpha_B - \alpha_C \\ &+ w(C,A) + \alpha_c - \alpha_A \\ &= w(F) \end{aligned}$$

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x(\delta^+(v)) = x(\delta^-(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

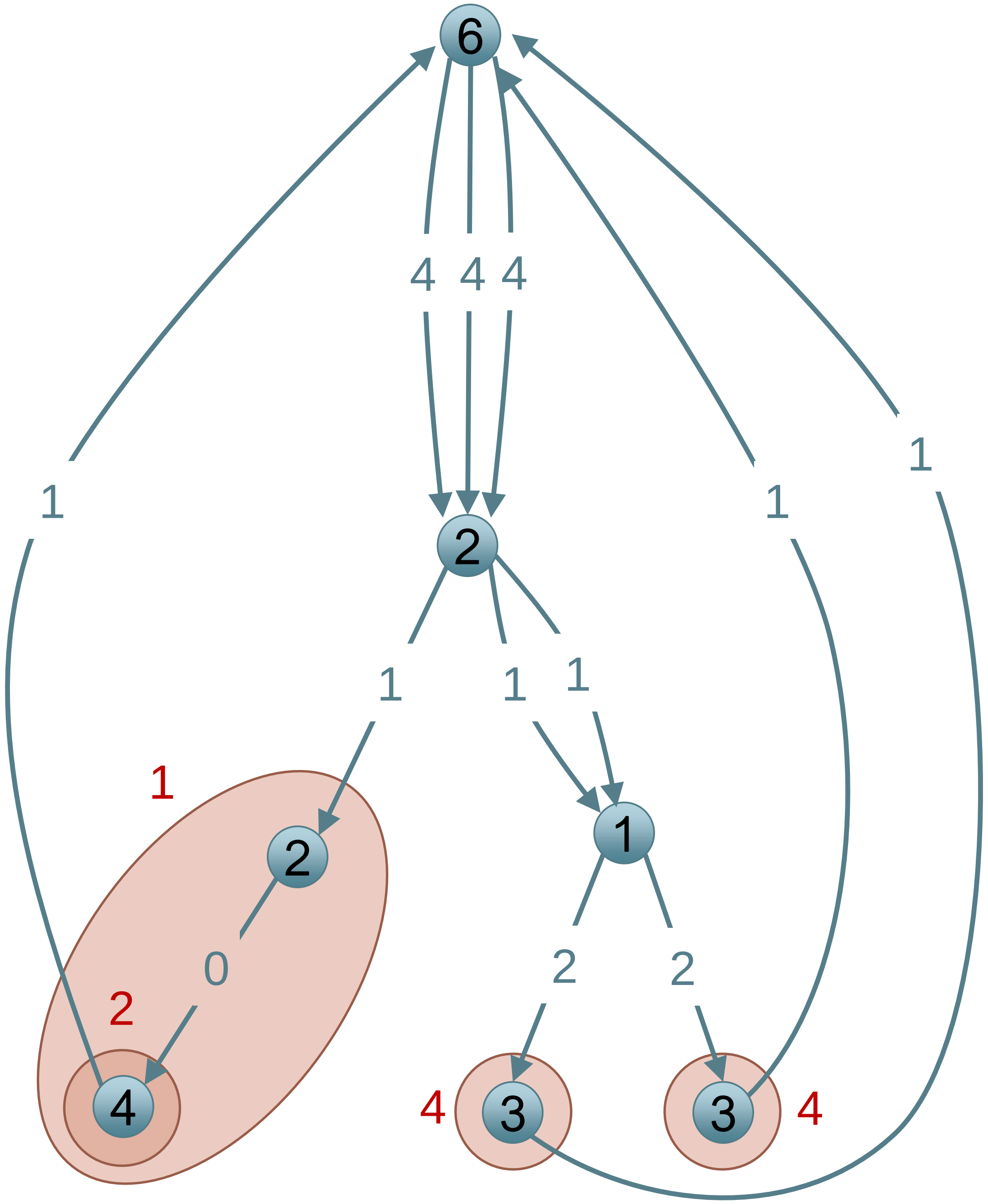
*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

*Subject to:*

$$\boldsymbol{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)} \text{ for all } (\mathrm{u},\mathrm{v}) \in E$$

$$\mathrm{y} \geq 0$$

Dual value = LP-value = 22



By complementarity slackness:

$$\sum_{S:(u,v)\in\delta(S)} y_S = w(u,v) - \alpha_u + \alpha_v =: w'(u,v)$$

for every edge (u,v) *(since we only kept positive edges)*

**Observation:**

For any Eulerian edge set F

$$w(F) = w'(F)$$

Thus equivalent to consider weight function $w'$:

$$w'(u,v) = \sum_{S:(u,v)\in\delta(S)} y_S$$

So normalize and forget about vertex potentials

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x}(\boldsymbol{\delta}^{+}(\boldsymbol{v})) = \boldsymbol{x}(\boldsymbol{\delta}^{-}(\boldsymbol{v}))$ for all $v \in V$

$\boldsymbol{x}(\boldsymbol{\delta}(\boldsymbol{S})) \geq \boldsymbol{2}$ for all $S \subset V$

$x \geq 0$

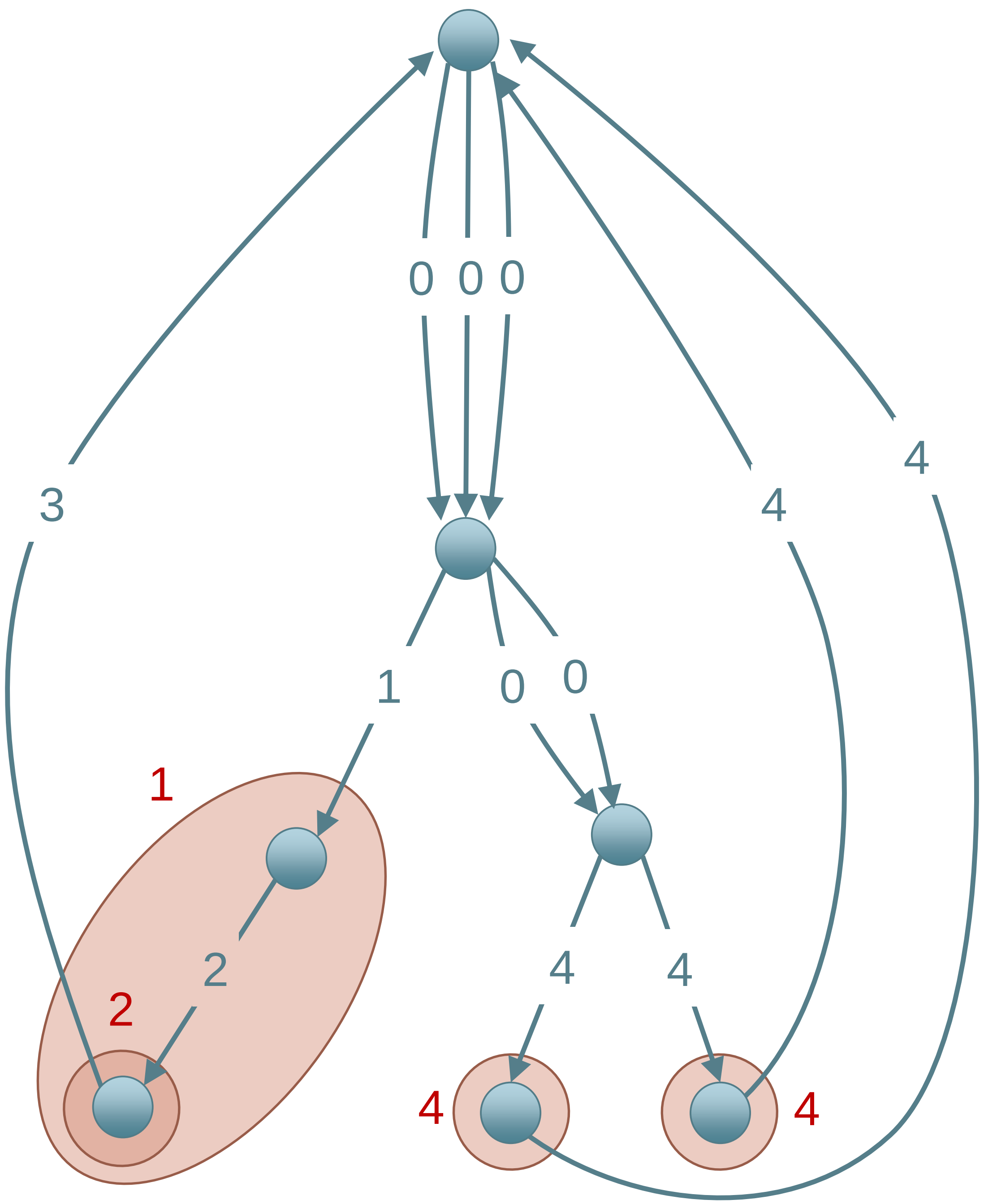
*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

*Subject to:*

$\sum_{\boldsymbol{S}:(\boldsymbol{u},\boldsymbol{v})\in\boldsymbol{\delta}(\boldsymbol{S})} \boldsymbol{y_S} + \boldsymbol{\alpha_u} - \boldsymbol{\alpha_v} \leq \boldsymbol{w}(\boldsymbol{u},\boldsymbol{v})$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

Dual value = LP-value = 22



By complementarity slackness:

$$\sum_{S:(u,v)\in\delta(S)} y_S = w(u,v) - \alpha_u + \alpha_v =: w'(u,v)$$

for every edge (u,v) *(since we only kept positive edges)*

**Observation:**

For any Eulerian edge set F

$w(F) = w'(F)$

Thus equivalent to consider weight function $w'$:

$$w'(u,v) = \sum_{S:(u,v)\in\delta(S)} y_S$$

So normalize and forget about vertex potentials

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x(\delta^+(v)) = x(\delta^-(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

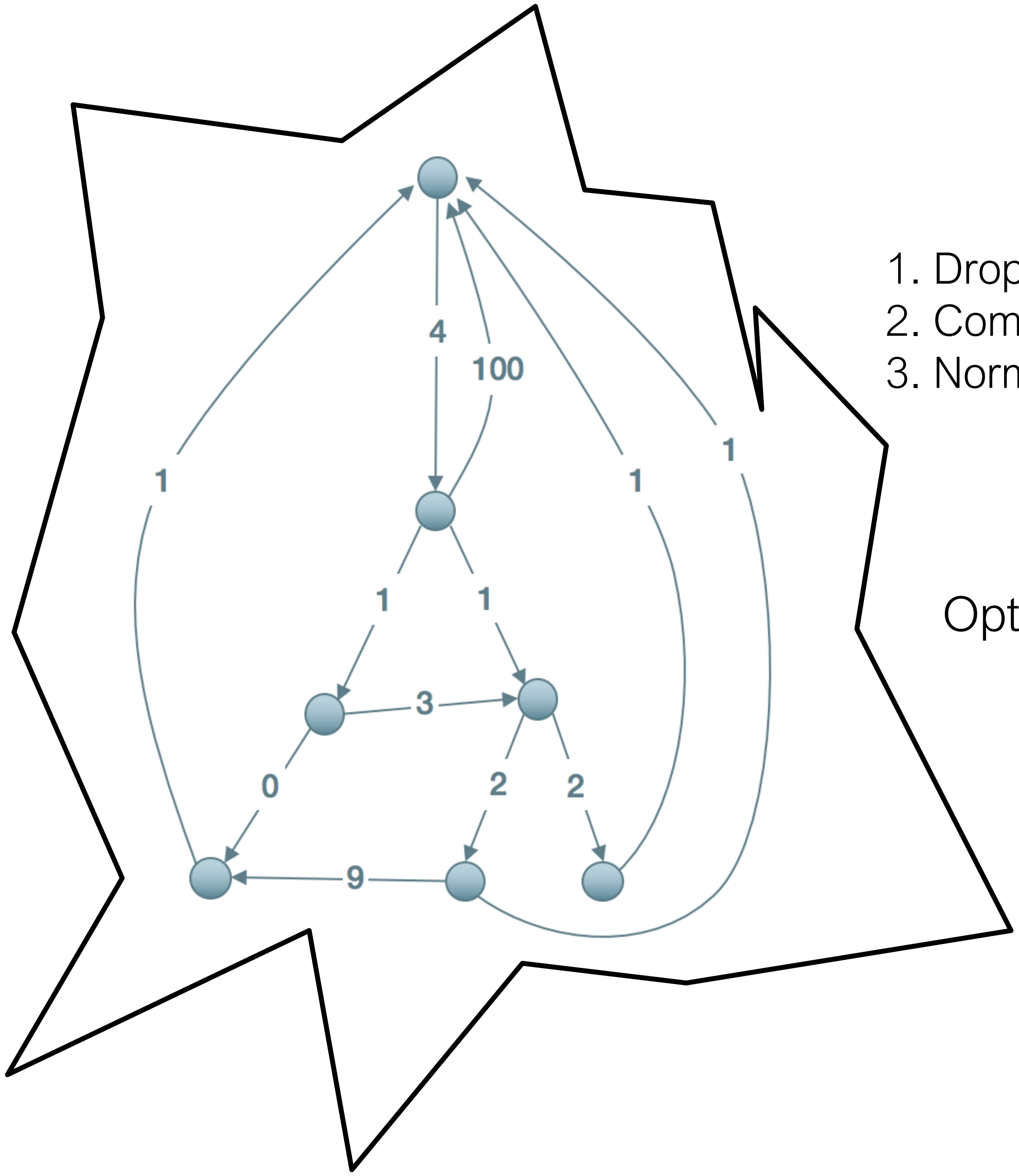
*Subject to:*

$\boldsymbol{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)}$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

# What happened?

Something complicated with no structure

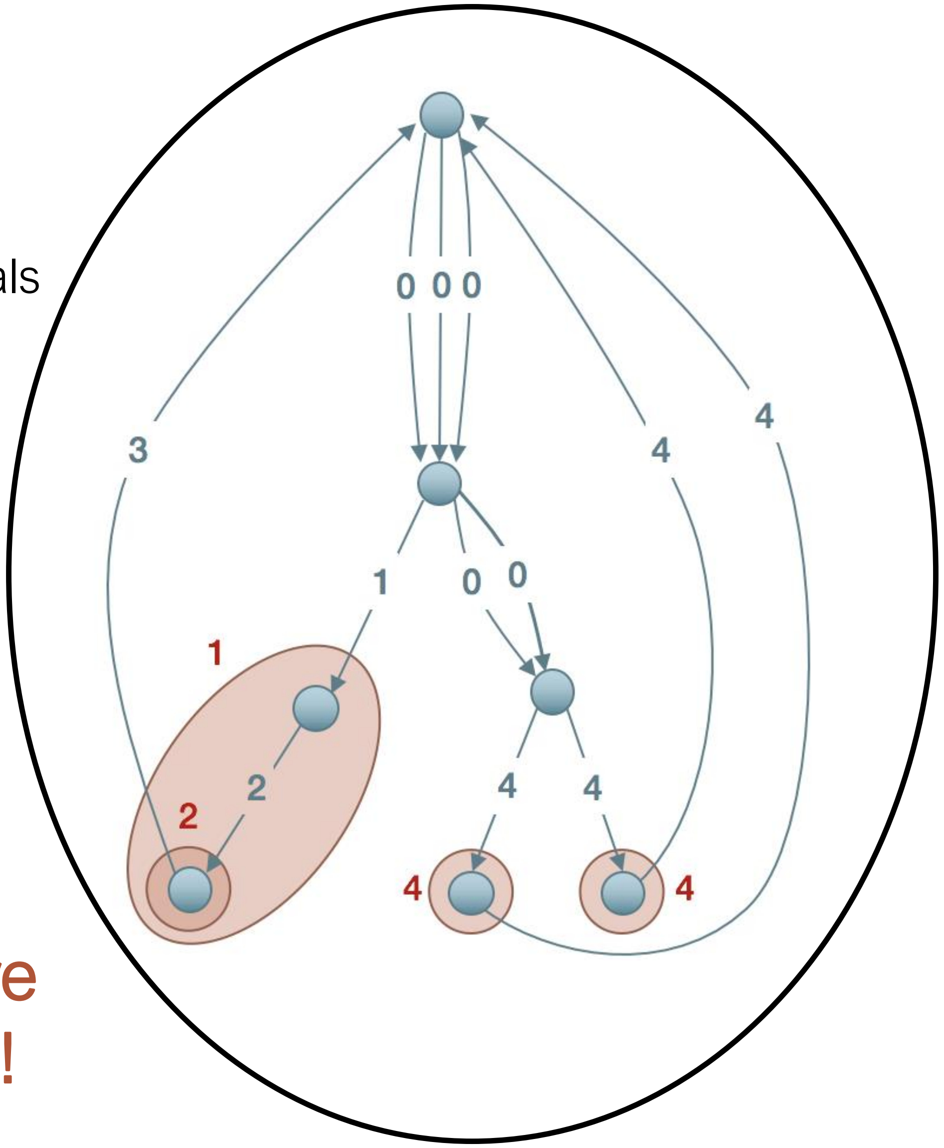


1. Drop 0-edges
2. Complementarity slackness
3. Normalize with vertex potentials

Optimal primal and dual
x and (y,0)

A lot of structure:

$$w(e) = \sum_{S:(u,v)\in\delta(S)} y_S$$



Want more structure!

Minimize: $\sum_{uv\in E} w(u,v)\, x_{uv}$

Subject to: $\boldsymbol{x(\delta^+(v)) = x(\delta^-(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

Maximize: $\sum_{S\subset V} 2 \cdot y_S$

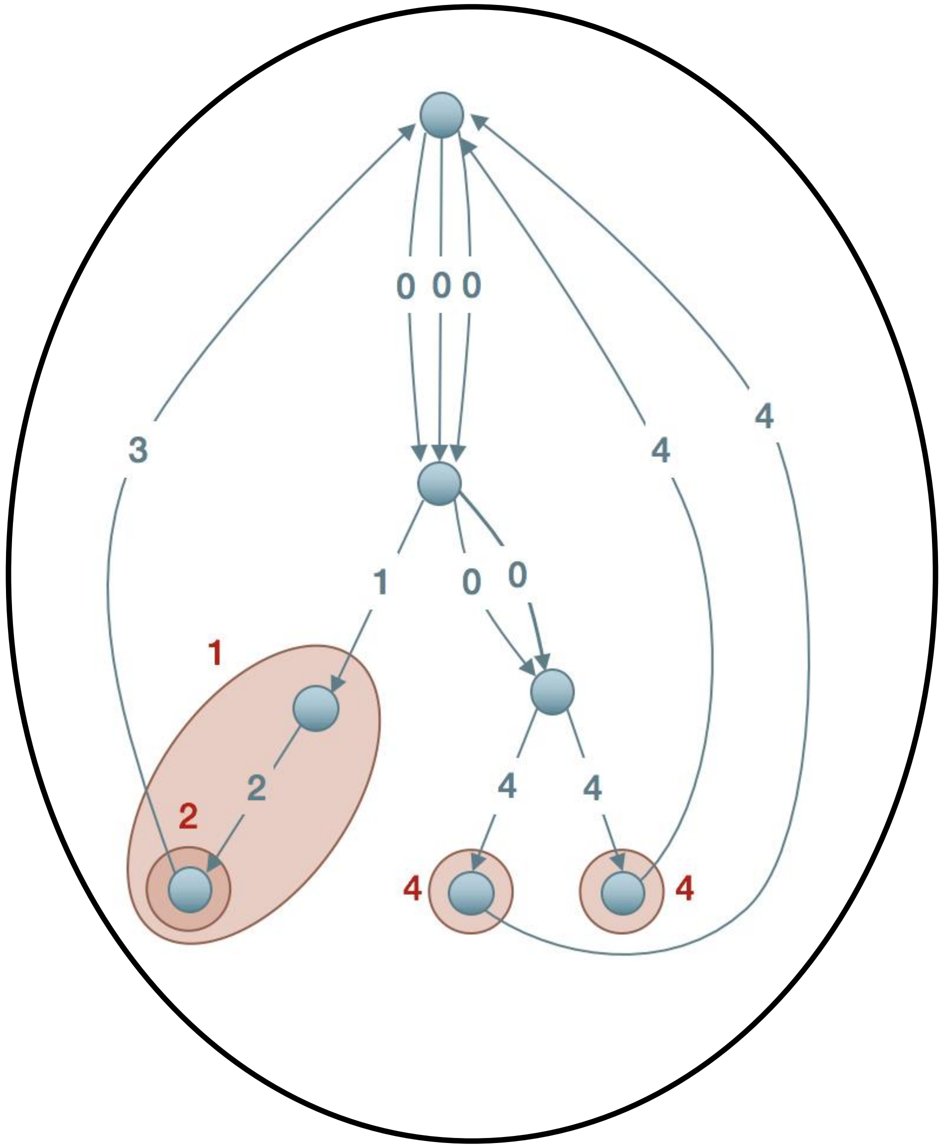
Subject to:

$\boldsymbol{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)}$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

A lot of structure:

$$w(e) = \sum_{S:(u,v)\in\delta(S)} y_S$$



*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$

$x(\delta(S)) \geq 2$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

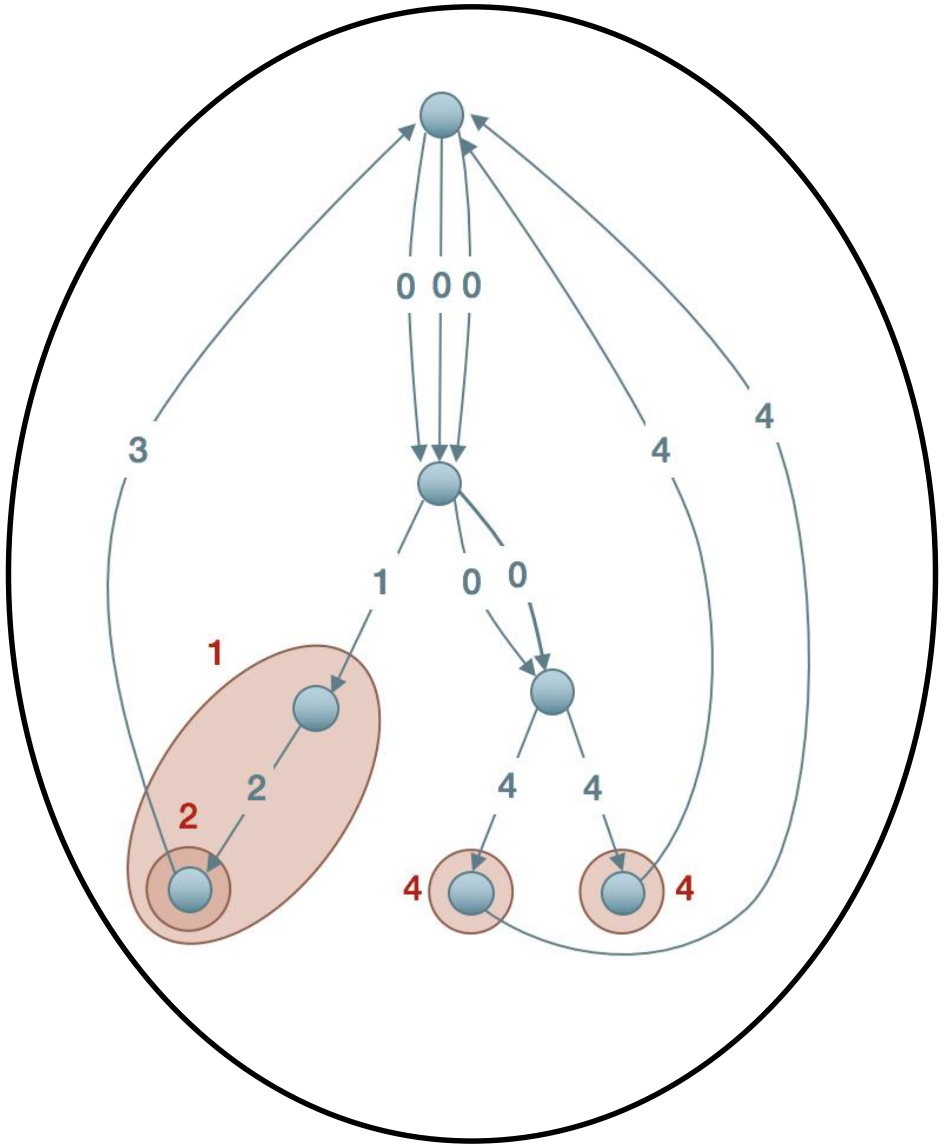
*Subject to:*

$\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)$ for all $(u,v) \in E$

$y \geq 0$

A lot of structure:

$$w(e) = \sum_{S:(u,v)\in\delta(S)} y_S$$



Let $\mathcal{L} = \{S: y_S > 0\}$ be support of dual solution

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $\boldsymbol{x(\delta^{+}(v)) = x(\delta^{-}(v))}$ for all $v \in V$

$\boldsymbol{x(\delta(S)) \geq 2}$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

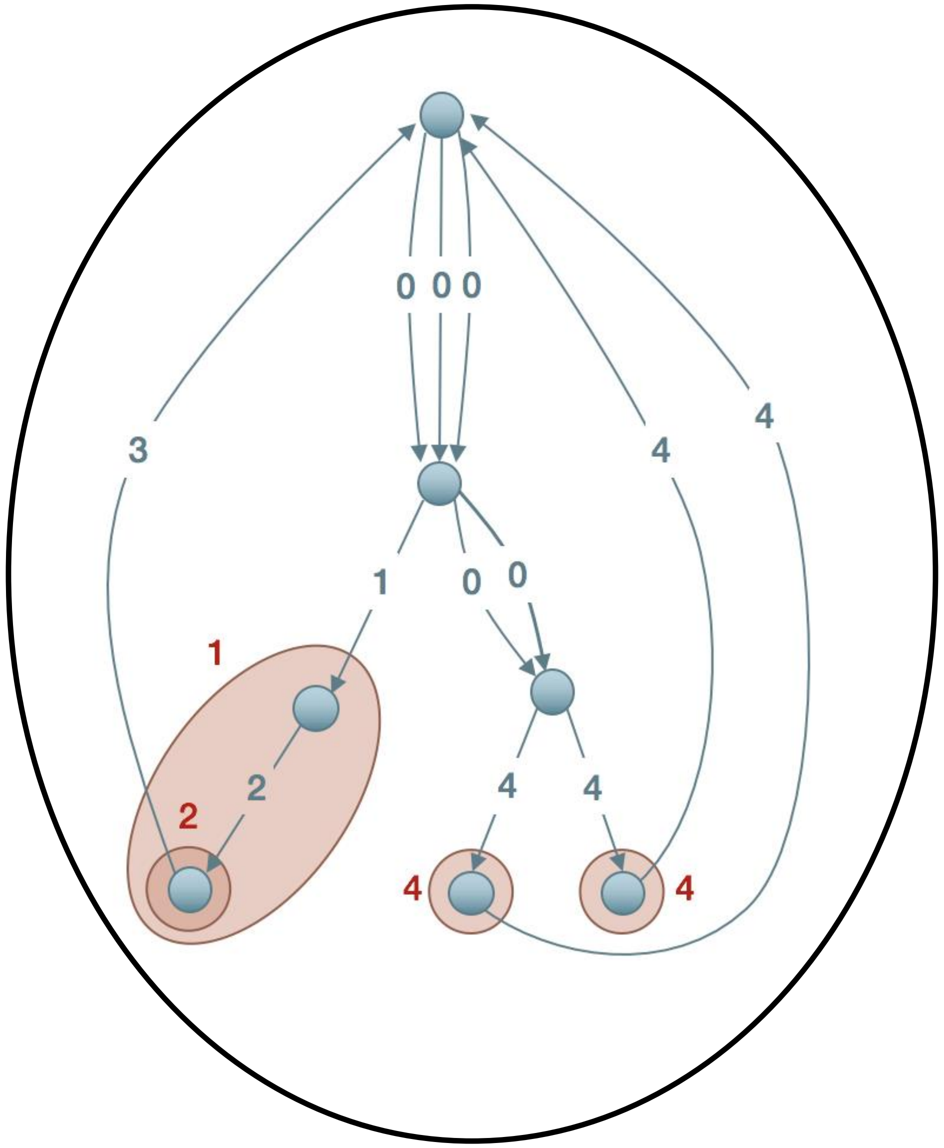
*Subject to:*

$\boldsymbol{\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)}$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

A lot of structure:

$$w(e) = \sum_{S:(u,v)\in\delta(S)} y_S$$



Let $\mathcal{L} = \{S: y_S > 0\}$ be support of dual solution

Again by complementarity slackness

$$x(\delta(S)) = 2 \text{ for every } S \in \mathcal{L}$$

*So every $S \in \mathcal{L}$ is a tight set!*

*Minimize:* $\sum_{uv\in E} w(u,v)\, x_{uv}$

*Subject to:* $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$

$x(\delta(S)) \geq 2$ for all $S \subset V$

$x \geq 0$

*Maximize:* $\sum_{S\subset V} 2 \cdot y_S$

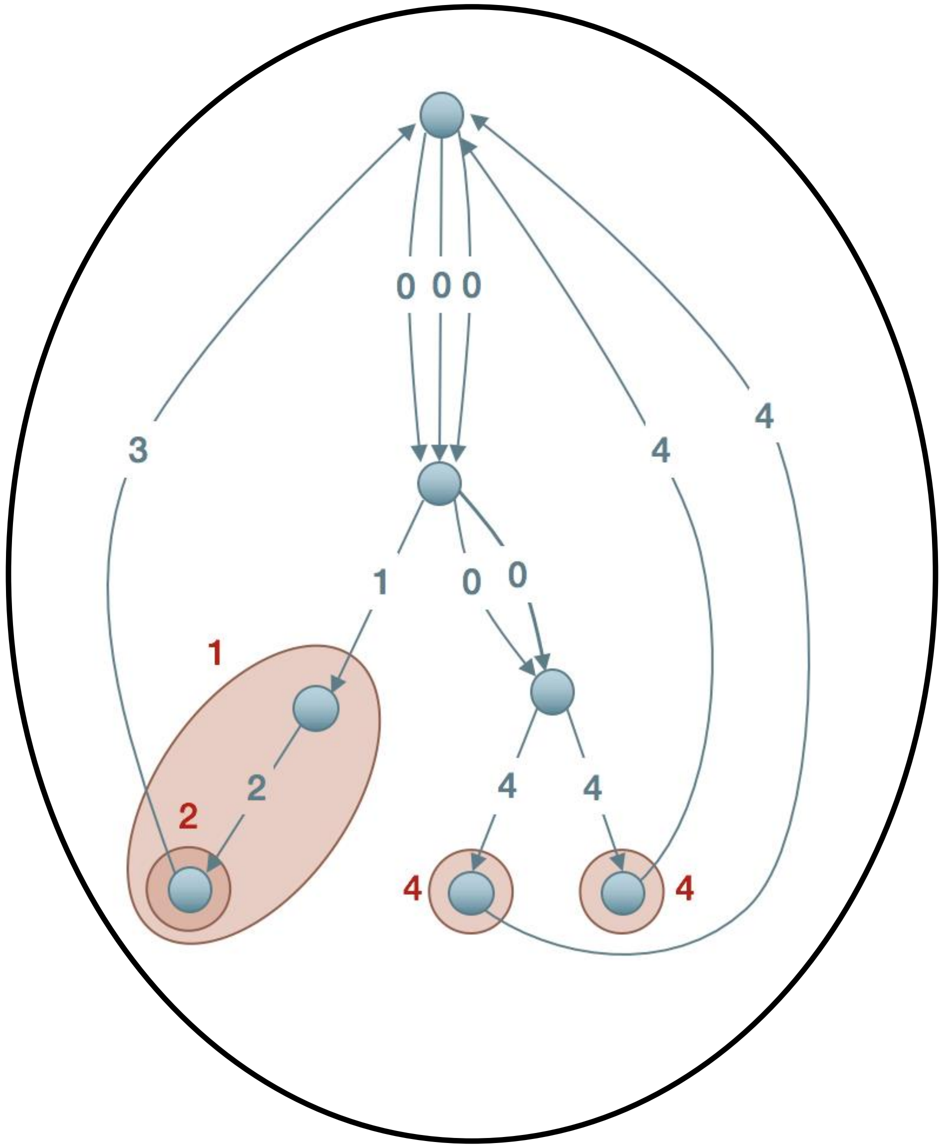
*Subject to:*

$\sum_{S:(u,v)\in\delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)$ for all $(\mathrm{u},\mathrm{v}) \in E$

$\mathrm{y} \geq 0$

A lot of structure:

$$w(e) = \sum_{S:(u,v)\in\delta(S)} y_S$$



Let $\mathcal{L} = \{S: y_S > 0\}$ be support of dual solution

Again by complementarity slackness

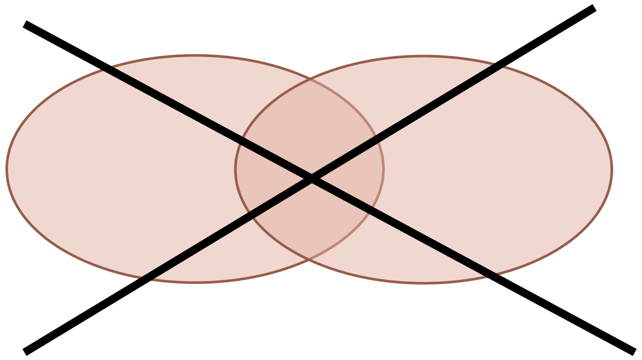
$$x(\delta(S)) = 2 \text{ for every } S \in \mathcal{L}$$

*So every $S \in \mathcal{L}$ is a tight set!*

By “standard” uncrossing techniques:

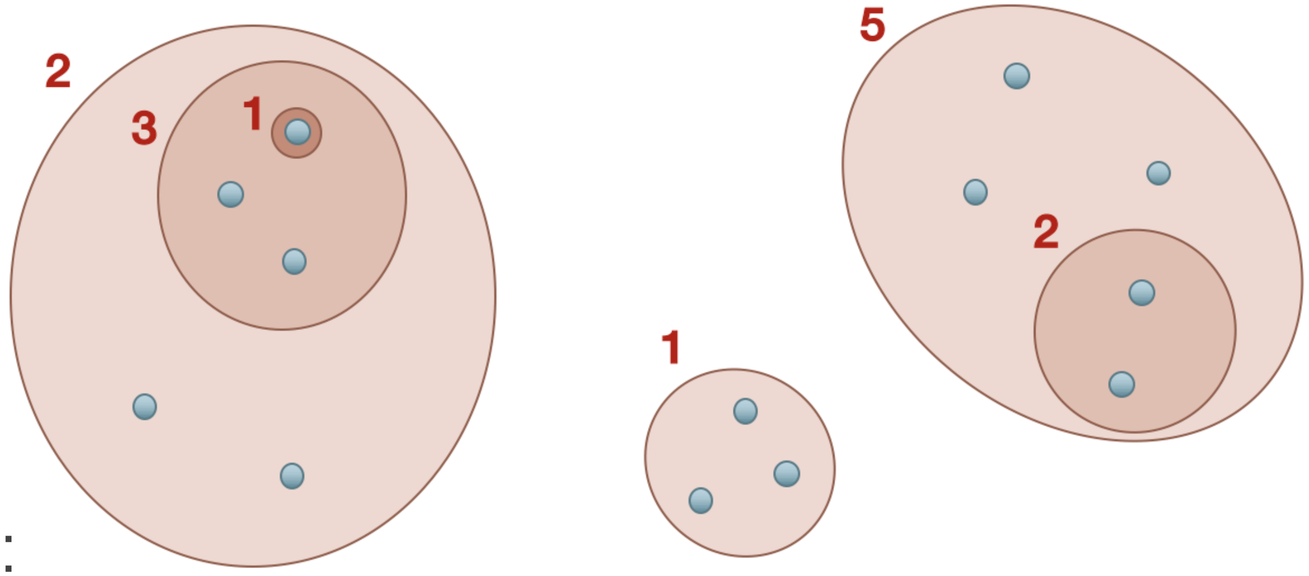
$\mathcal{L}$ is a laminar family

*Any two sets are either disjoint or one is a subset of the other*



No two sets intersect non-trivially

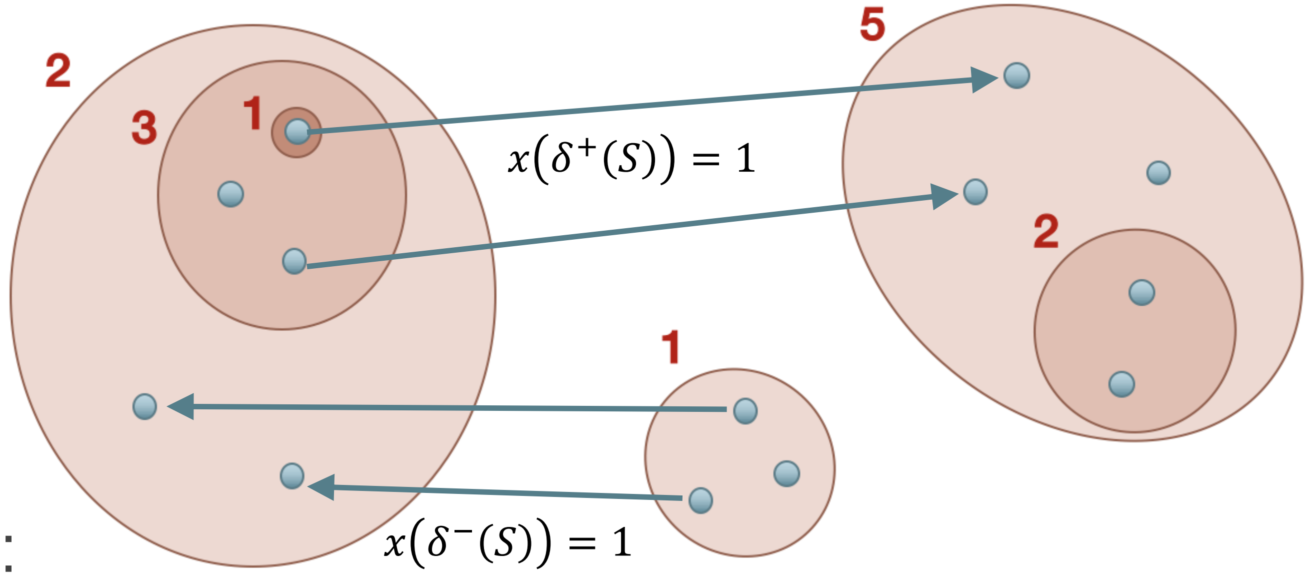
# Laminarly-weighted



Laminarly-weighted instance $\mathcal{I} = (G, \mathcal{L}, x, y)$:

- $x, y$ primal and dual solutions (that will be optimal by definition)

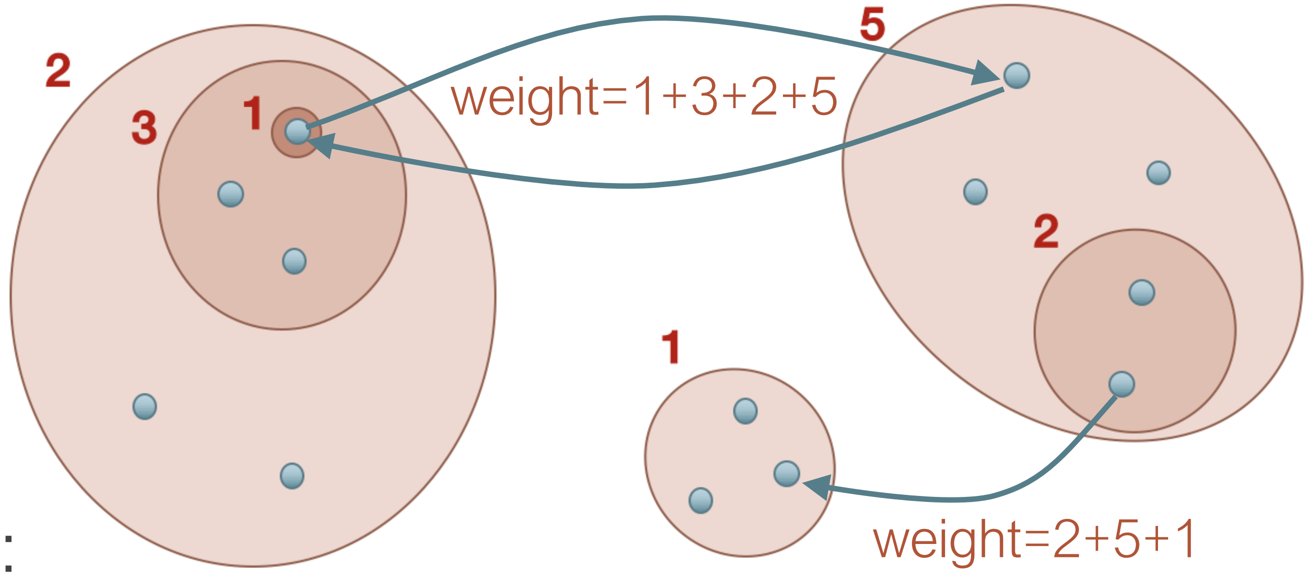
# Laminarly-weighted



Laminarly-weighted instance $\mathcal{I} = (G, \mathcal{L}, x, y)$:

- $x, y$ primal and dual solutions (that will be optimal by definition)
- $\mathcal{L} = \{S : y_S > 0\}$ is a *laminar* family of *tight* sets *(LP says that we should visit each such set once)*

# Laminarly-weighted



Laminarly-weighted instance $\mathcal{I} = (G, \mathcal{L}, x, y)$:

- $x, y$ primal and dual solutions (that will be optimal by definition)
- $\mathcal{L} = \{S: y_S > 0\}$ is a *laminar* family of *tight* sets *(LP says that we should visit each such set once)*
- weights induced by $\mathcal{L}$ and $y$:

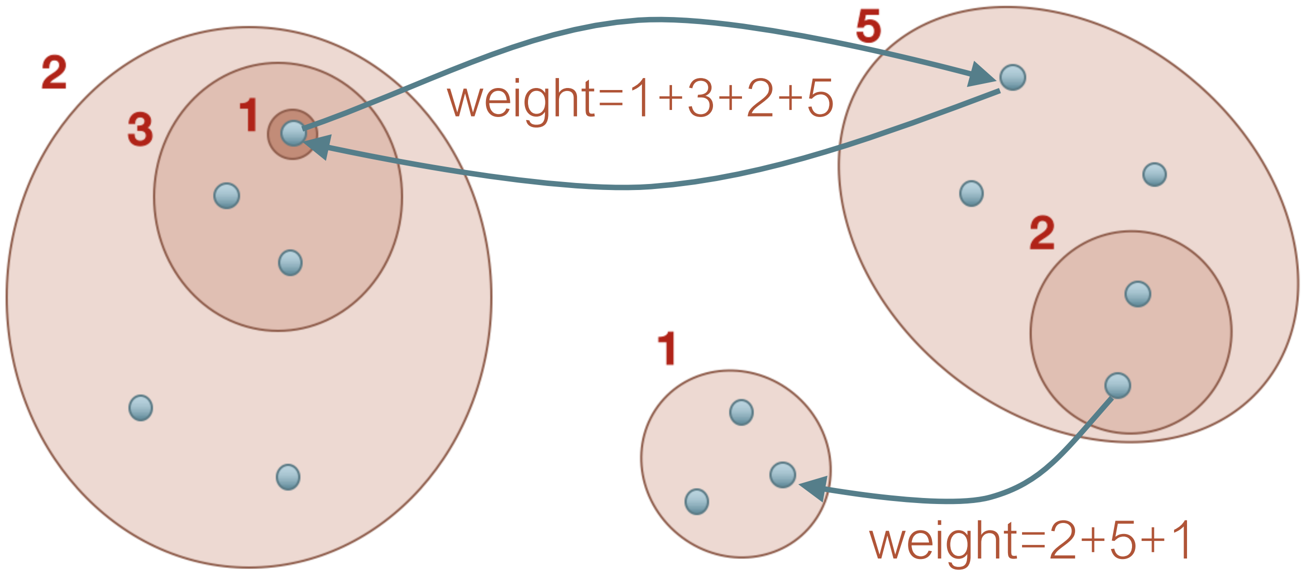
$$w(e) = \sum_{S \in \mathcal{L}: e \in \delta(S)} y_S \quad \text{for every edge } e$$

Held-Karp lower bound = OPT = $\mathbf{2} \cdot \sum_{S \in \mathcal{L}} y_S$ (=28 in example)

**Theorem:**

A ρ-approximation algorithm for laminarly-weighted instances yields a ρ-approximation algorithm for general ATSP

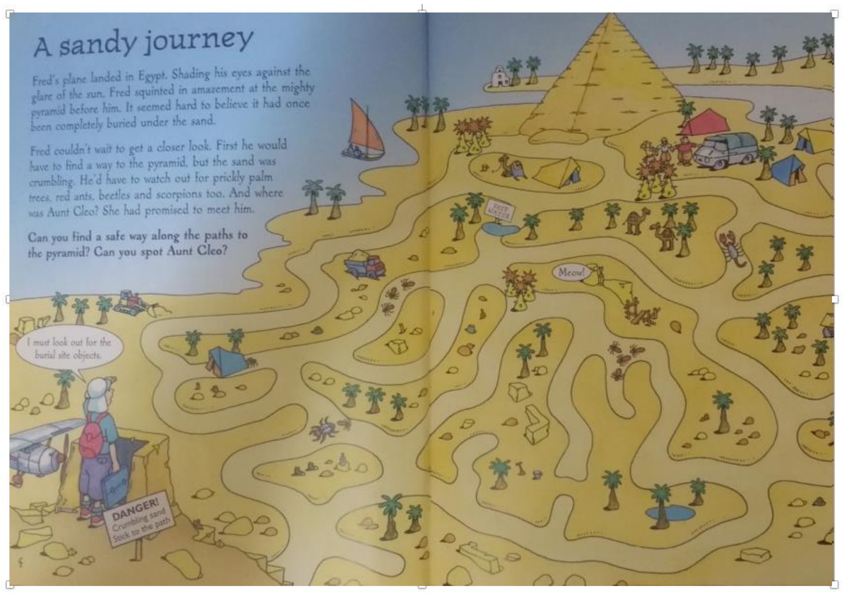
# Reduced our task to:



Laminarly-weighted instance $\mathcal{I} = (G, \mathcal{L}, x, y)$:

- $x, y$ primal and dual solutions (which will be optimal by definition)
- $\mathcal{L} = \{S : y_S > 0\}$ is a *laminar* family of *tight* sets *(LP says that we should visit each such set once)*
- weights induced by $\mathcal{L}$ and $y$:

$$w(e) = \sum_{S \in \mathcal{L} : e \in \delta(S)} y_S$$



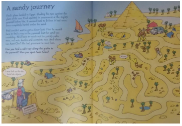
A sandy journey

Fred's plane landed in Egypt. Shading his eyes against the glare of the sun, Fred squinted in amazement at the mighty pyramid before him. It seemed hard to believe it had once been completely buried under the sand.

Fred couldn't wait to get a closer look. First he would have to find a way to the pyramid, but the sand was crumbling. He'd have to watch out for prickly palm trees, red ants, beetles and scorpions too. And where was Aunt Cleo? She had promised to meet him.

**Can you find a safe way along the paths to the pyramid? Can you spot Aunt Cleo?**

Basic idea:
recursively solving smaller instances
is not dangerous
if optimum drops



Irreducible instances

# Let’s take a detour

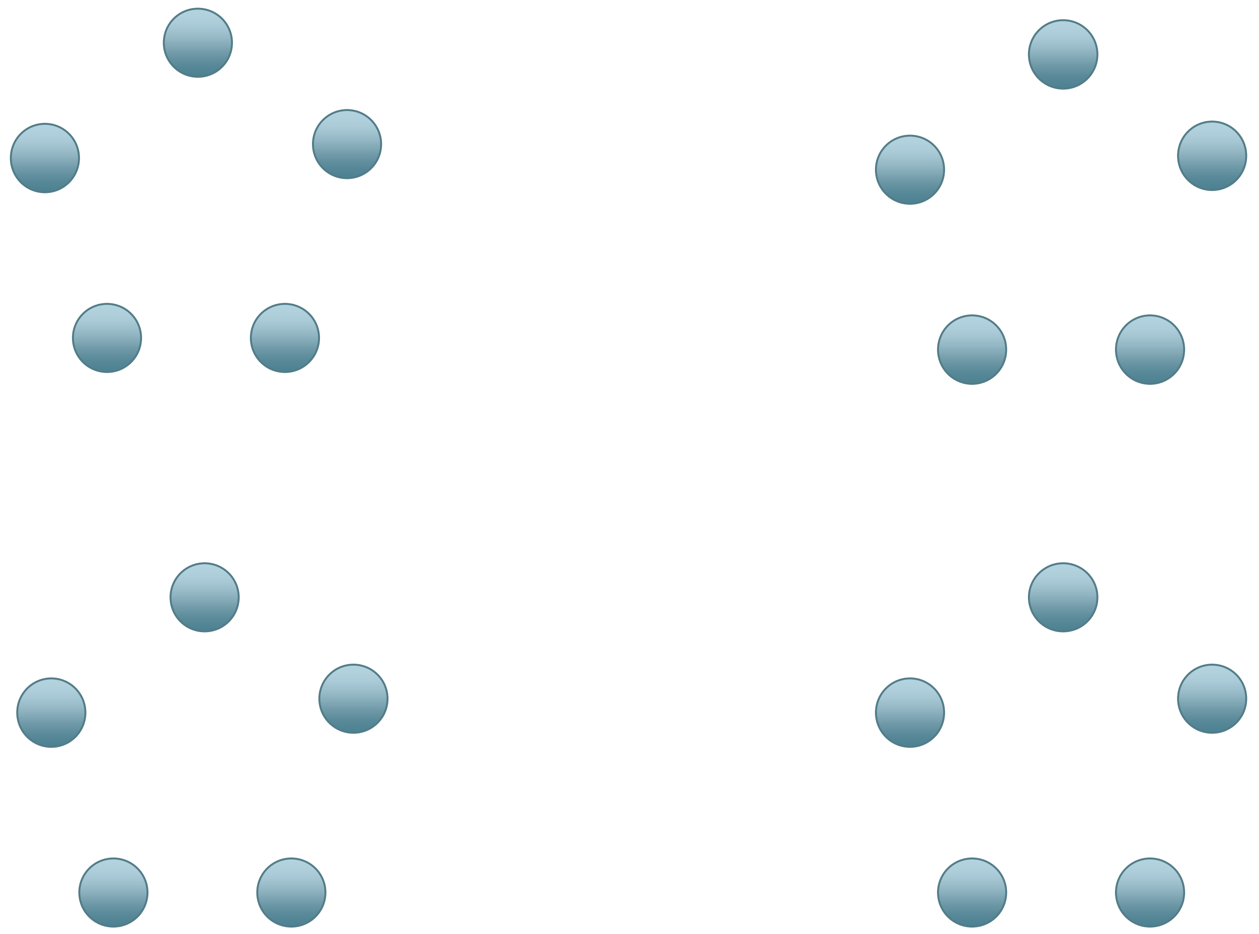
# Repeated cycle cover

[Frieze, Galbiati, and Maffioli'82]

Find min-cost cycle cover

“Contract“

Repeat until graph is connected



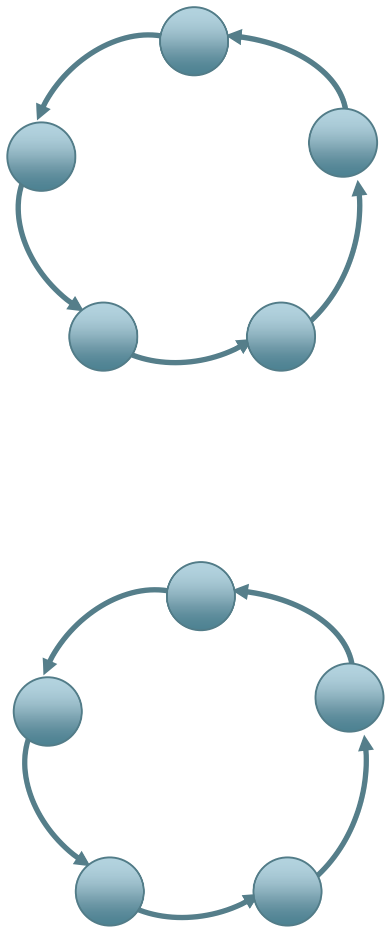
# Repeated cycle cover

[Frieze, Galbiati, and Maffioli’82]

Find min-cost cycle cover

“Contract“

Repeat until graph is connected



Cost of cycle cover $\leq \boldsymbol{OPT}$

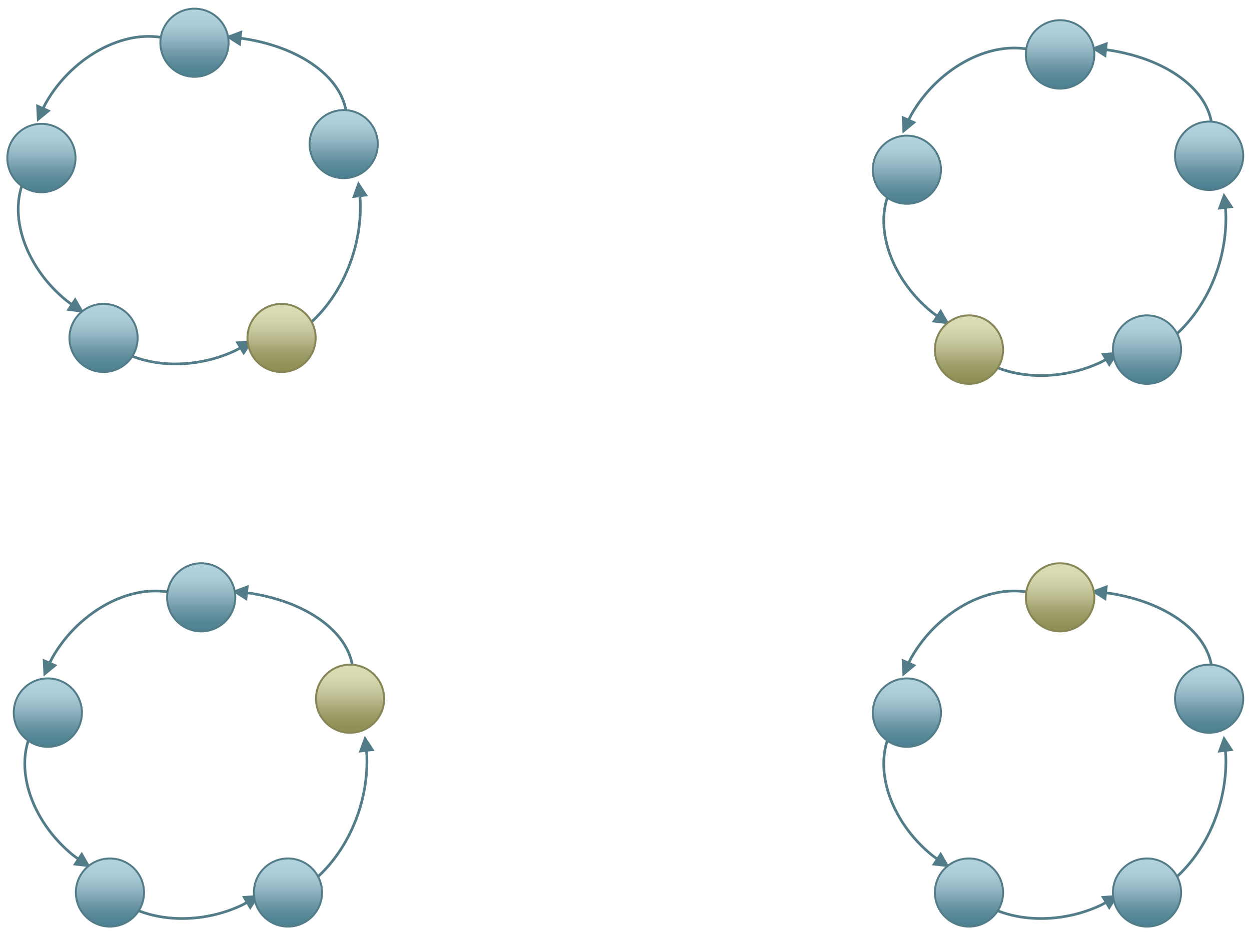
# Repeated cycle cover

[Frieze, Galbiati, and Maffioli'82]

Find min-cost cycle cover

“Contract“

Repeat until graph is connected



Cost of cycle cover $\leq \boldsymbol{OPT}$

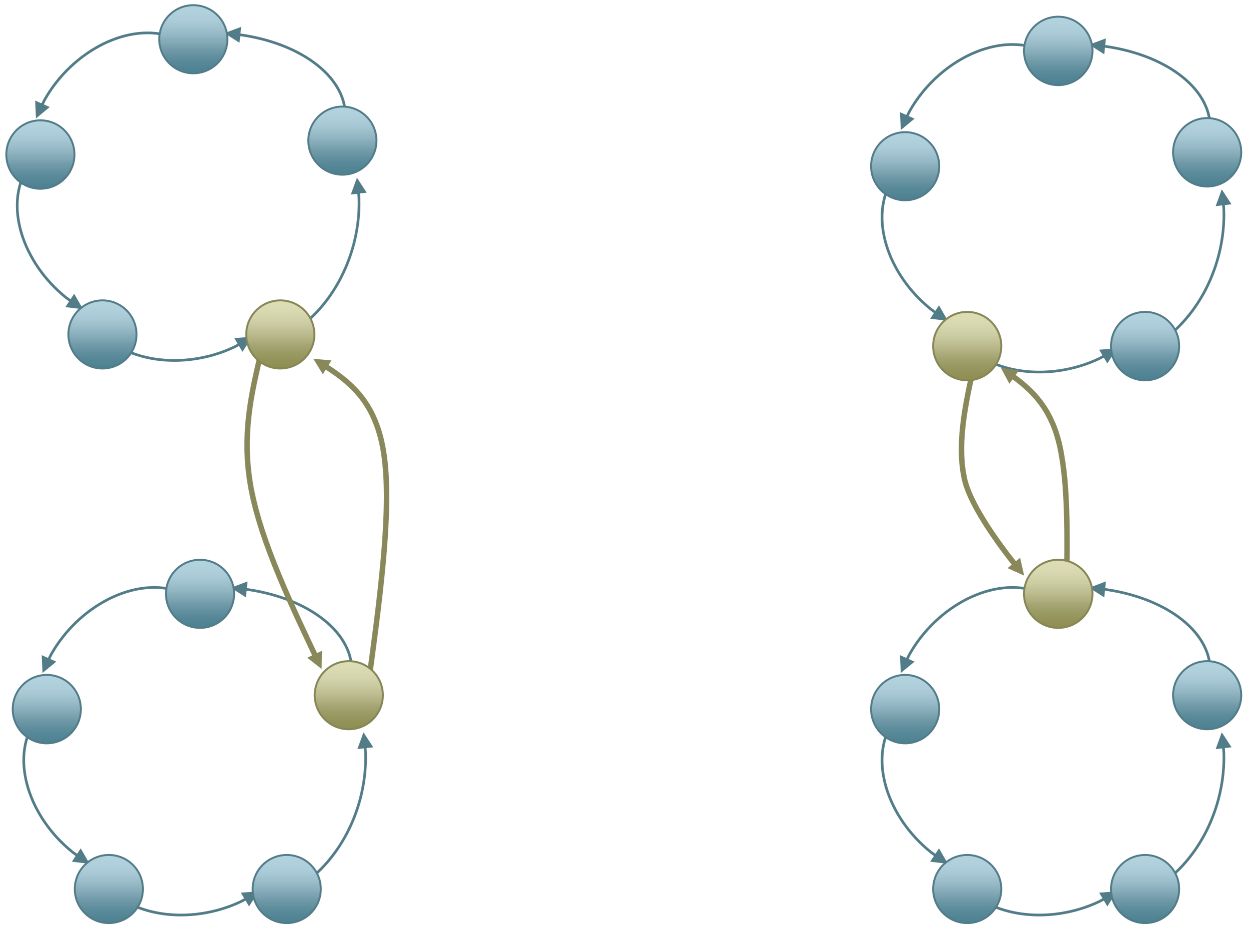
# Repeated cycle cover

[Frieze, Galbiati, and Maffioli'82]

Find min-cost cycle cover

"Contract"

Repeat until graph is connected



Cost of cycle cover $\leq \boldsymbol{OPT}$

Cost of cycle cover $\leq \boldsymbol{OPT}$

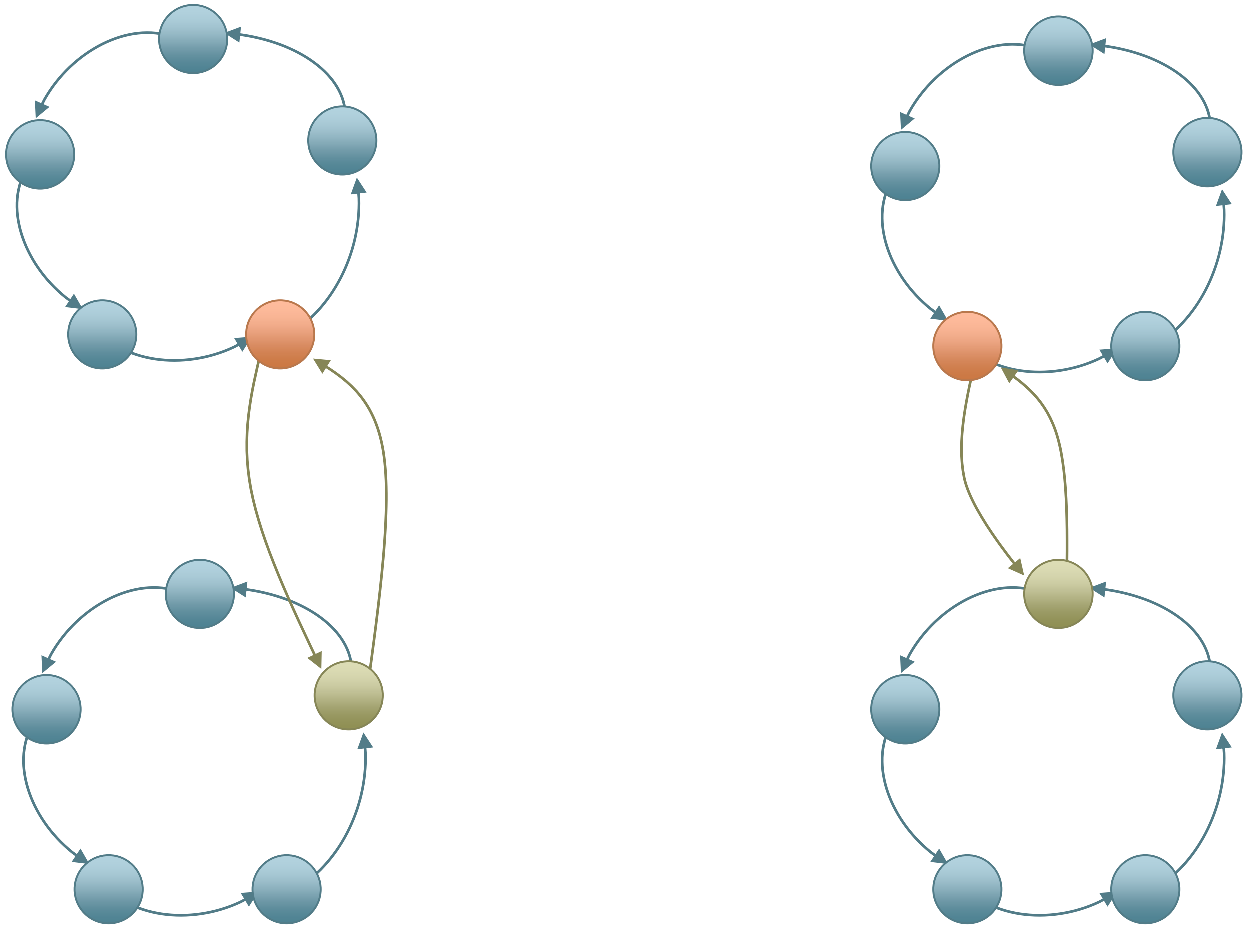
# Repeated cycle cover

[Frieze, Galbiati, and Maffioli'82]

Find min-cost cycle cover

"Contract"

Repeat until graph is connected



Cost of cycle cover $\leq \boldsymbol{OPT}$

Cost of cycle cover $\leq \boldsymbol{OPT}$

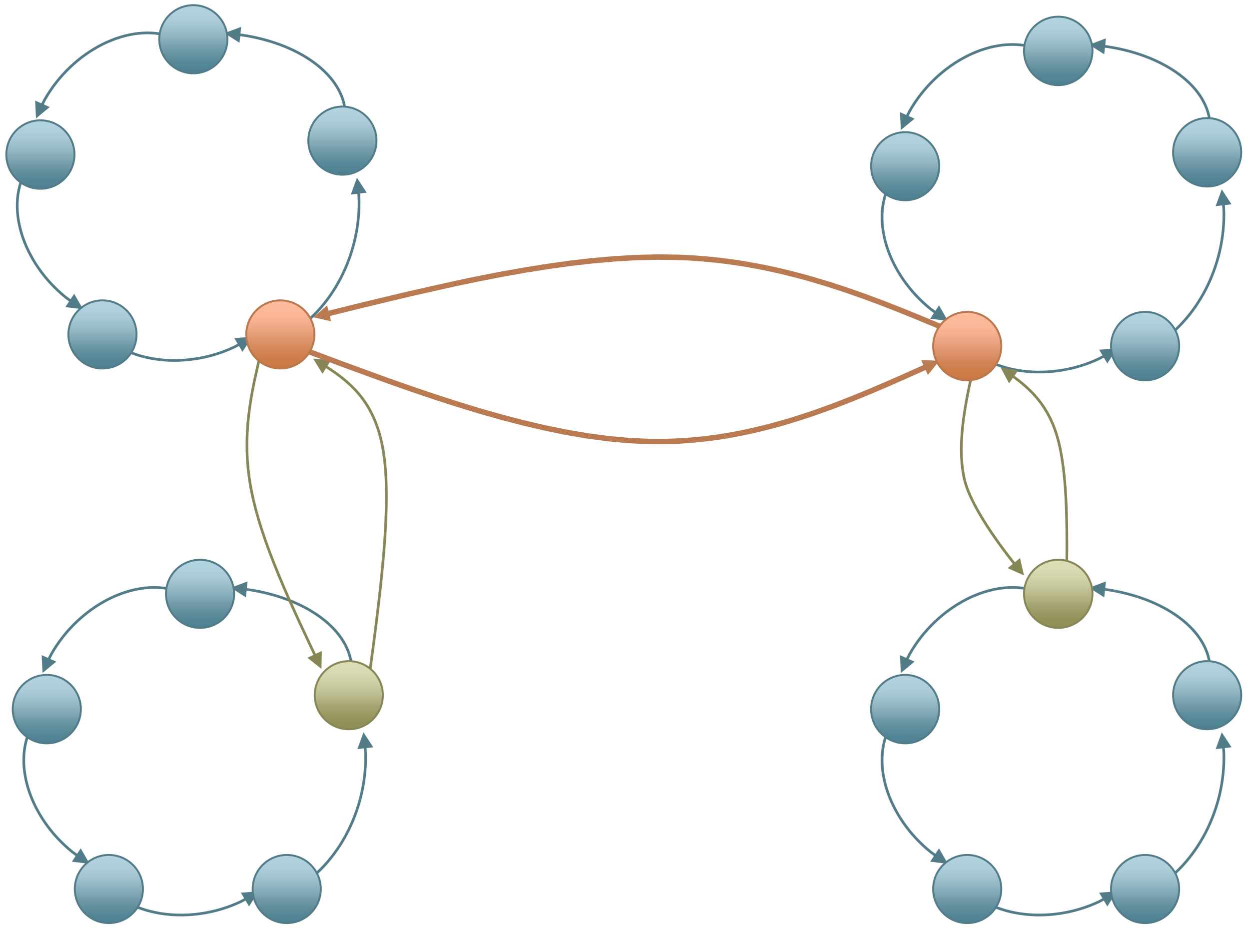
# Repeated cycle cover

[Frieze, Galbiati, and Maffioli'82]

Find min-cost cycle cover

"Contract"

Repeat until graph is connected



Cost of cycle cover $\leq \boldsymbol{OPT}$

Cost of cycle cover $\leq \boldsymbol{OPT}$

Cost of cycle cover $\leq \boldsymbol{OPT}$

# Repeated cycle cover

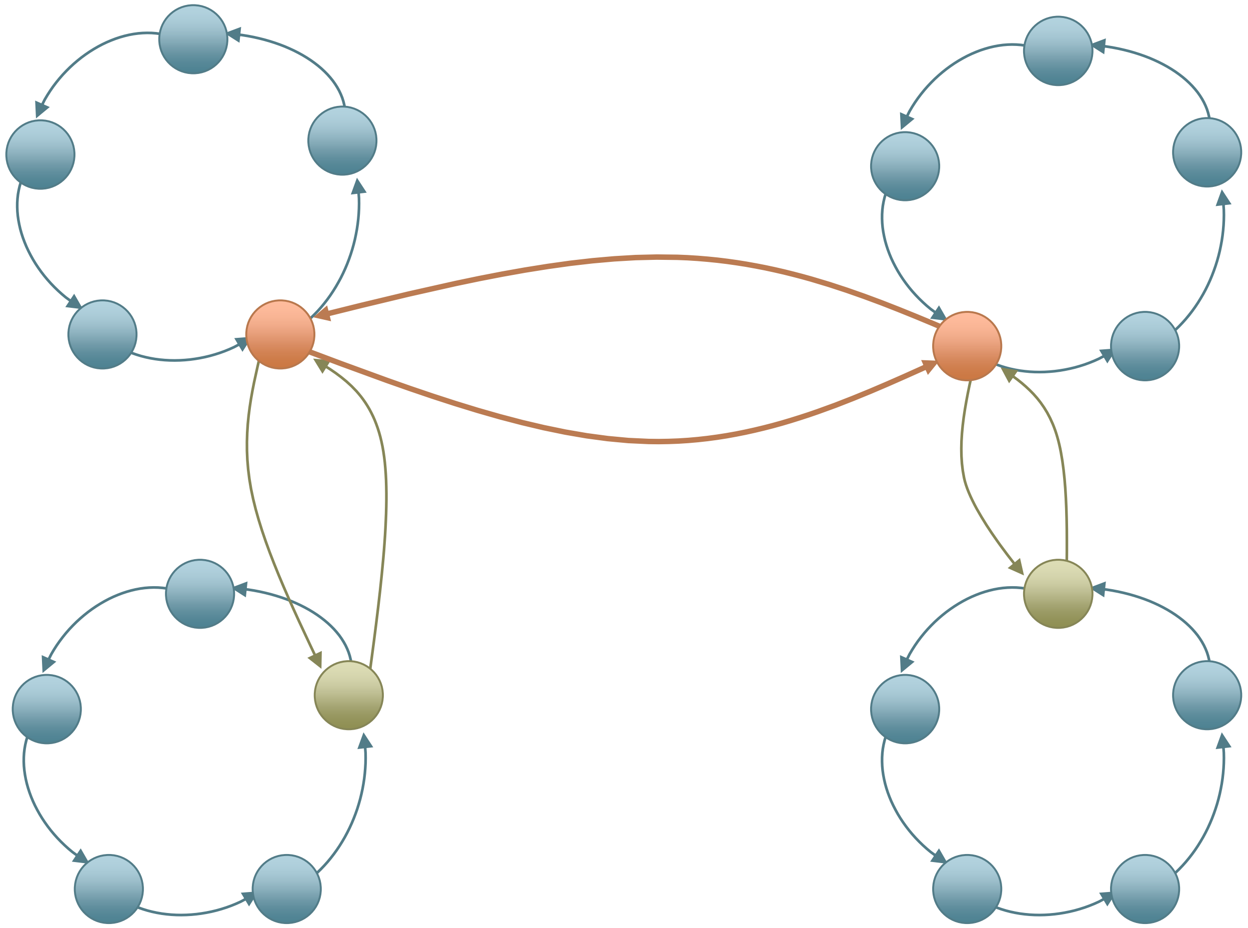
[Frieze, Galbiati, and Maffioli'82]

Find min-cost cycle cover

"Contract"

Repeat until graph is connected

Worst case:  all cycles have length 2 so need to repeat $\log_2 n$ times (each time cost $OPT_{LP}$)



Cost of cycle cover $\leq \boldsymbol{OPT}$

Cost of cycle cover $\leq \boldsymbol{OPT}$

Cost of cycle cover $\leq \boldsymbol{OPT}$

Total cost $\leq \mathbf{3} \cdot \boldsymbol{OPT}$

$\boldsymbol{log_2\, n}$-approximation

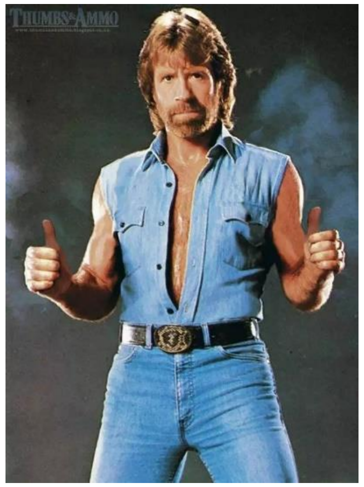
# Recursive algorithm fine if value drops

Each time we take a cycle cover we make some progress

What if the value of OPT drops by say a factor 9/10 each time?

Then total cost would be

$$\sum_{i=0}^{\log_2 n} \left(\frac{9}{10}\right)^i OPT \leq \sum_{i=0}^{\infty} \left(\frac{9}{10}\right)^i OPT = 10 \cdot OPT$$

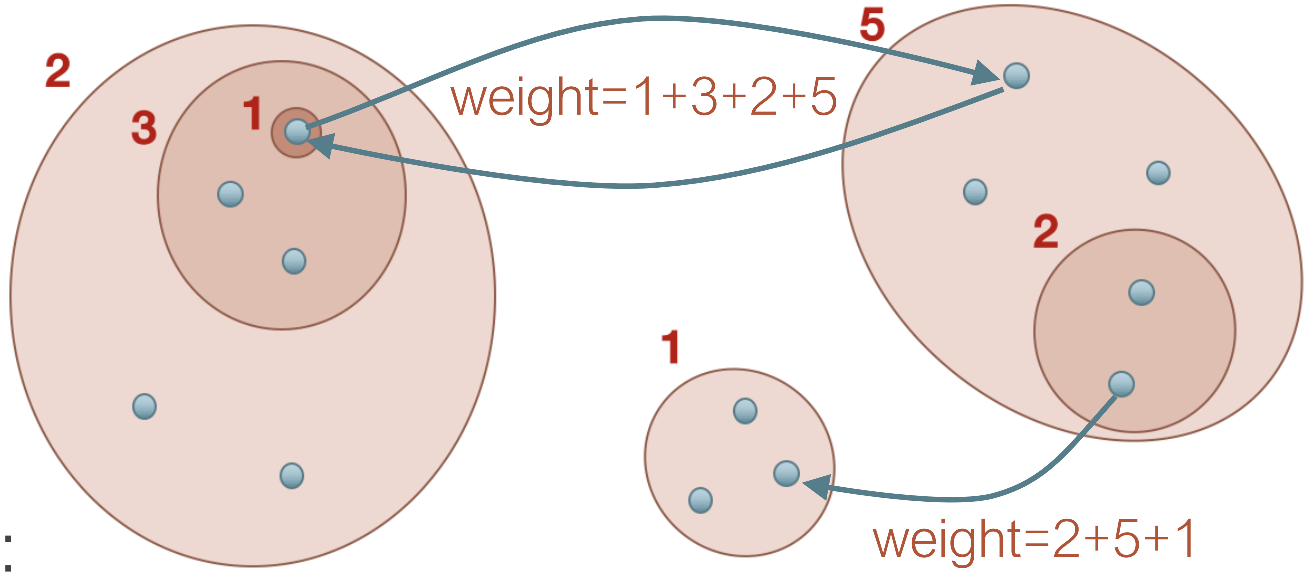


No one has been able to pursue this strategy with cycle cover approach

**We pursue it using the structure of laminarly-weighted instances**

# Le retour

# Laminarly-weighted



Laminarly-weighted instance $\mathcal{I} = (G, \mathcal{L}, x, y)$:

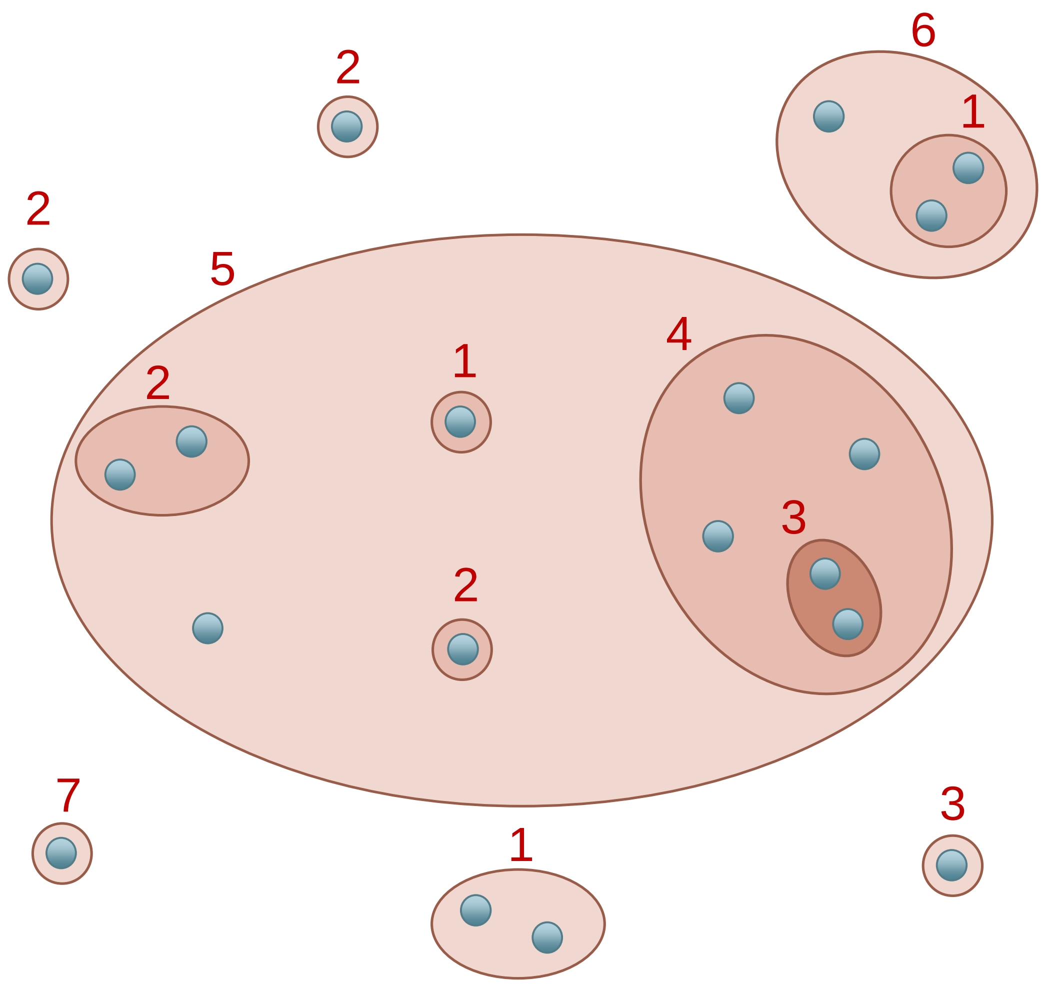
- $x, y$ primal and dual solutions (which will be optimal by definition)
- $\mathcal{L} = \{S : y_S > 0\}$ is a *laminar* family of *tight* sets *(LP says that we should visit each such set once)*
- weights induced by $\mathcal{L}$ and $y$:

$$w(e) = \sum_{S \in \mathcal{L} : e \in \delta(S)} y_S \quad \text{for every edge } e$$

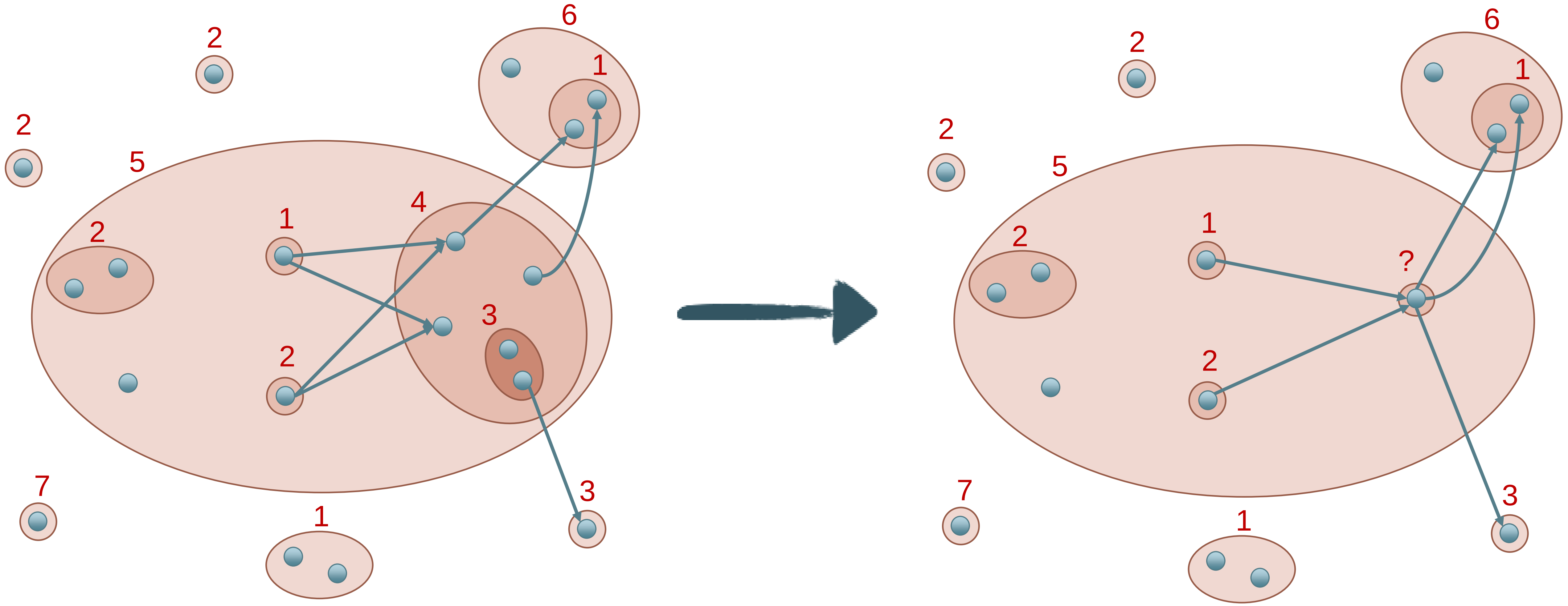
Held-Karp lower bound = OPT = $\mathbf{2} \cdot \sum_{S \in \mathcal{L}} y_S$ (=28 in example)

# Contraction and lift

# Contraction of tight sets in $\mathcal{L}$



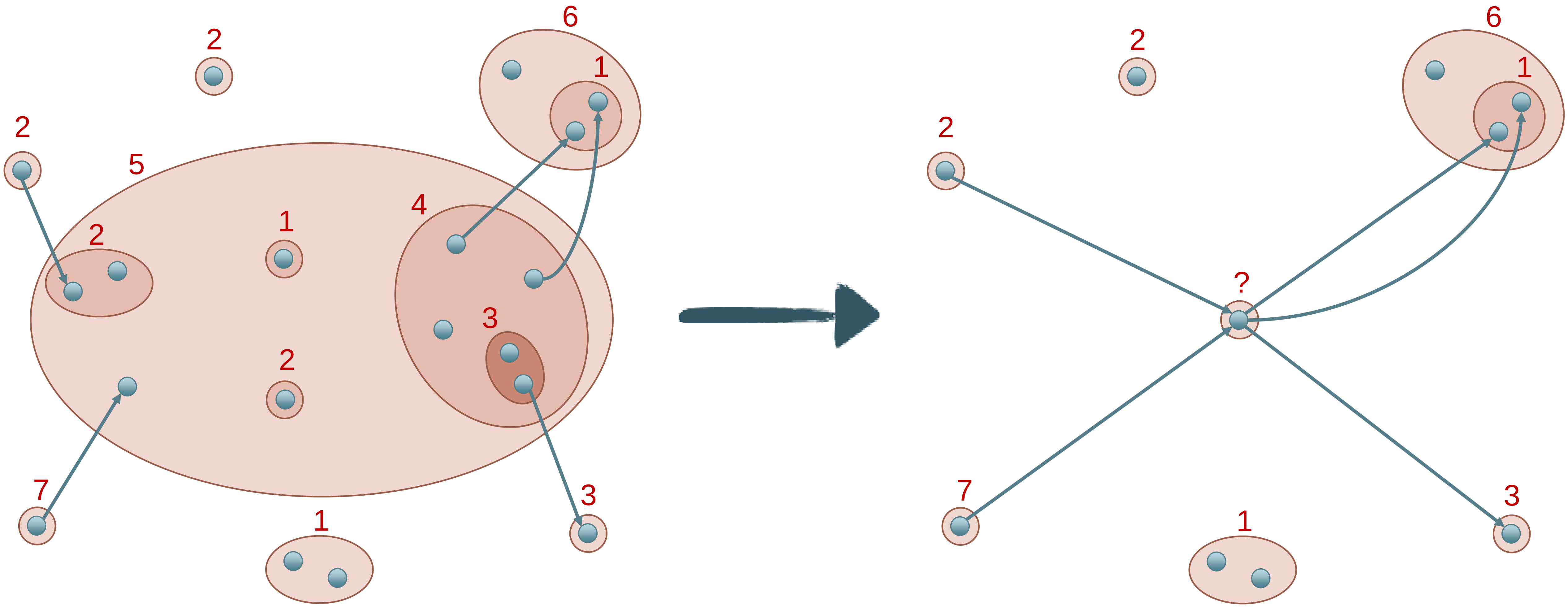
# Contraction of tight sets in $\mathcal{L}$



Contraction gives smaller instance: G, x, $\mathcal{L}$ easy to contract

Remains to specify y-value of new vertex/set

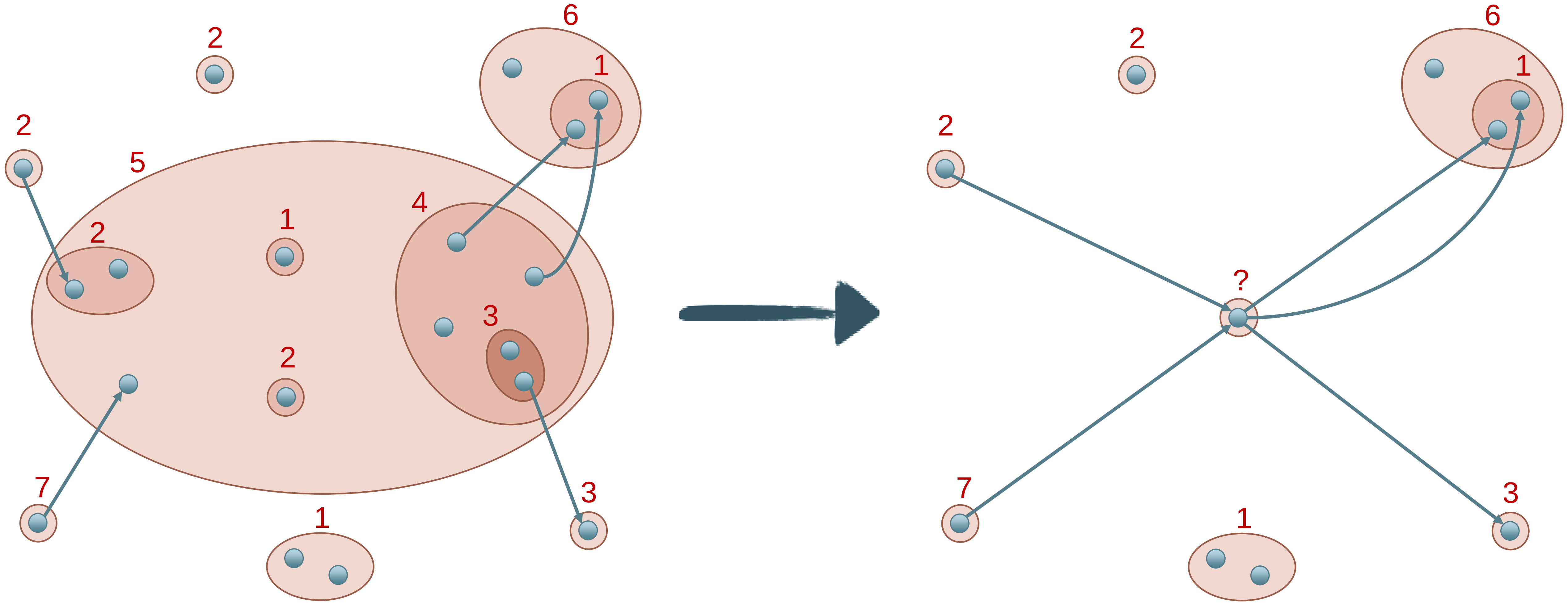
# Contraction of tight sets in $\mathcal{L}$



Contraction gives smaller instance: G, x, $\mathcal{L}$ easy to contract

Remains to specify y-value of new vertex/set

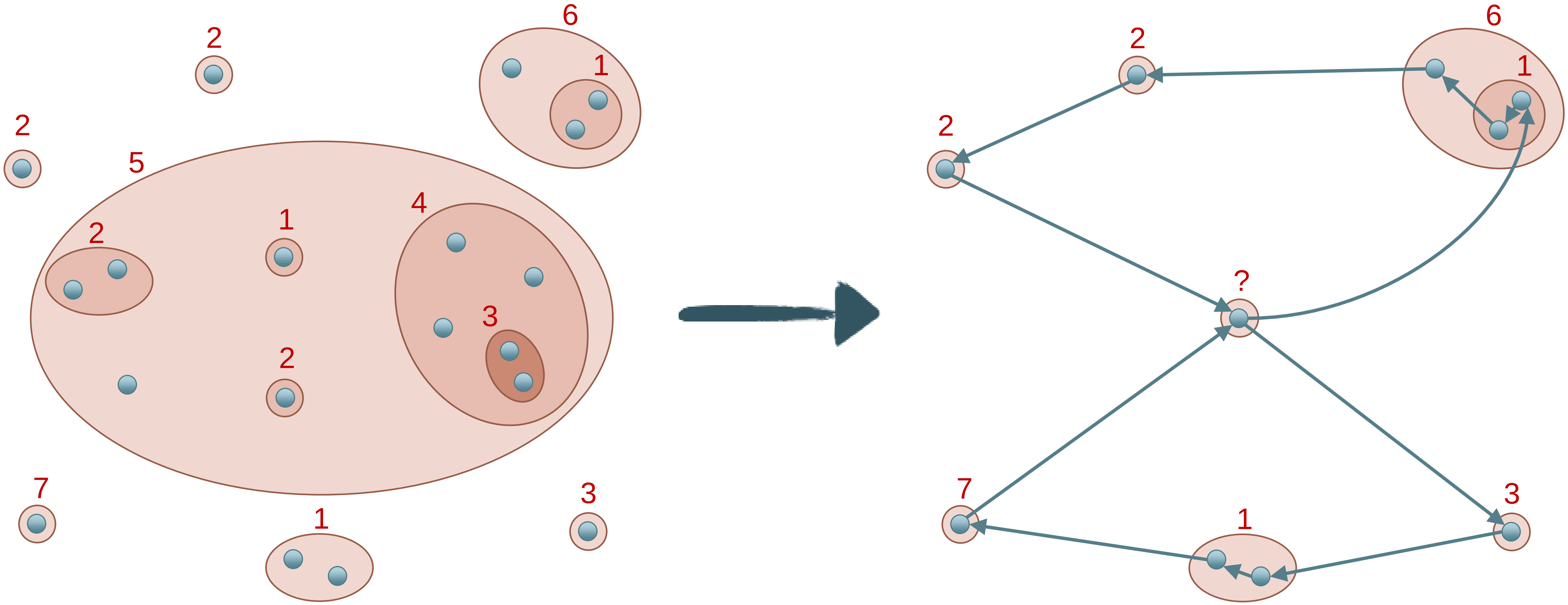
# Contraction of tight sets in $\mathcal{L}$



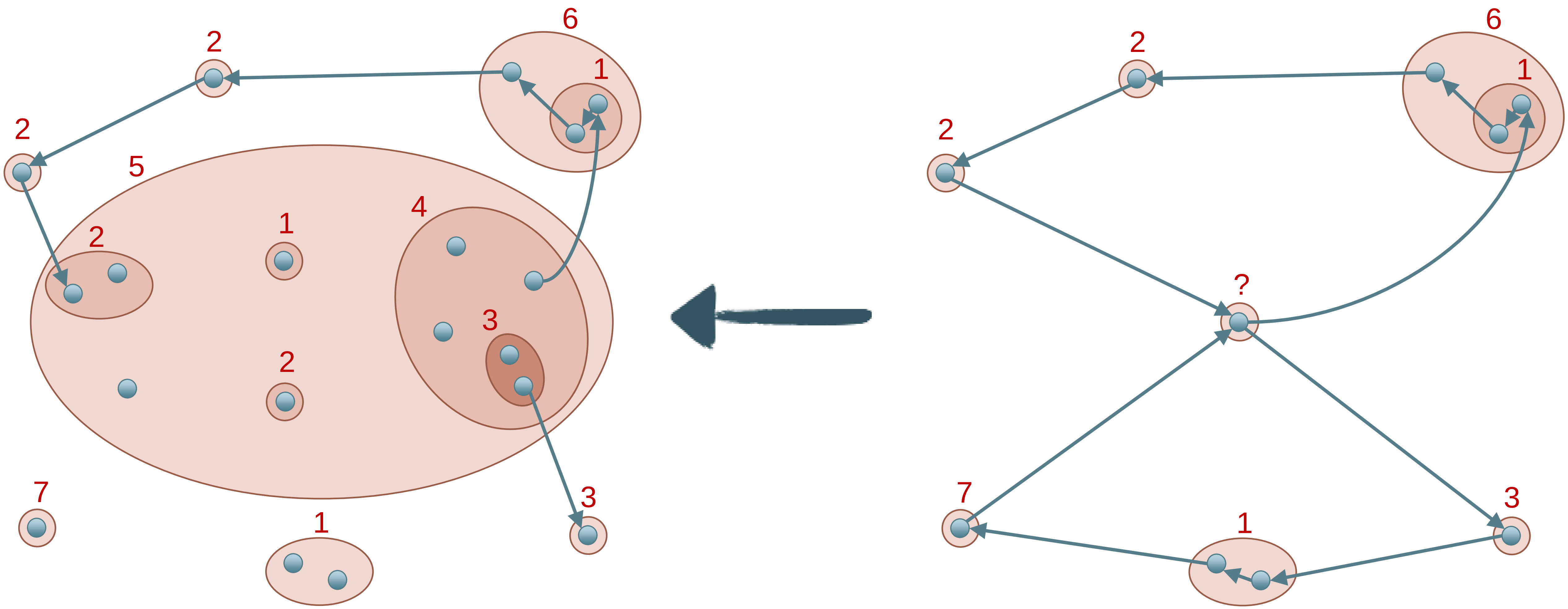
Contraction gives smaller instance: G, x, $\mathcal{L}$ easy to contract

**Remains to specify y-value of new vertex/set**

# Lifting a tour in the contracted instance



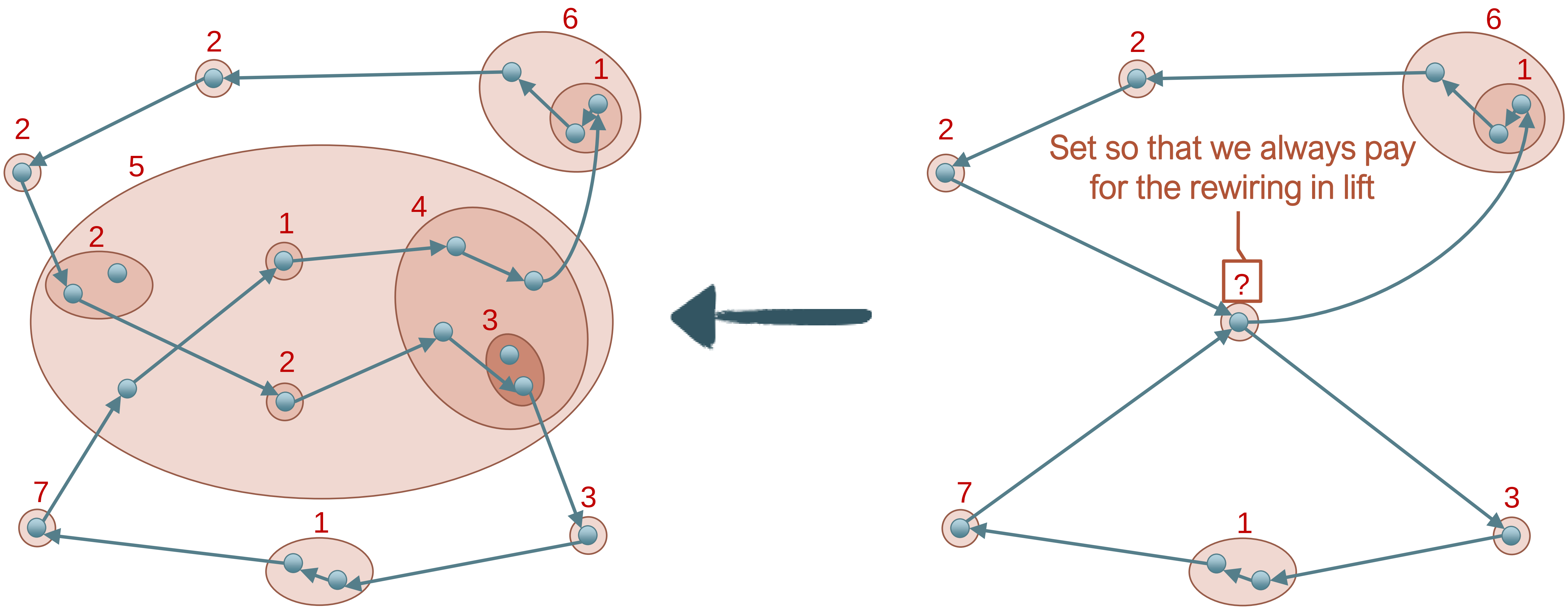
# Lifting a tour in the contracted instance



What to do?

Lift tour in contracted instance to subtour in original instance

# Lifting a tour in the contracted instance

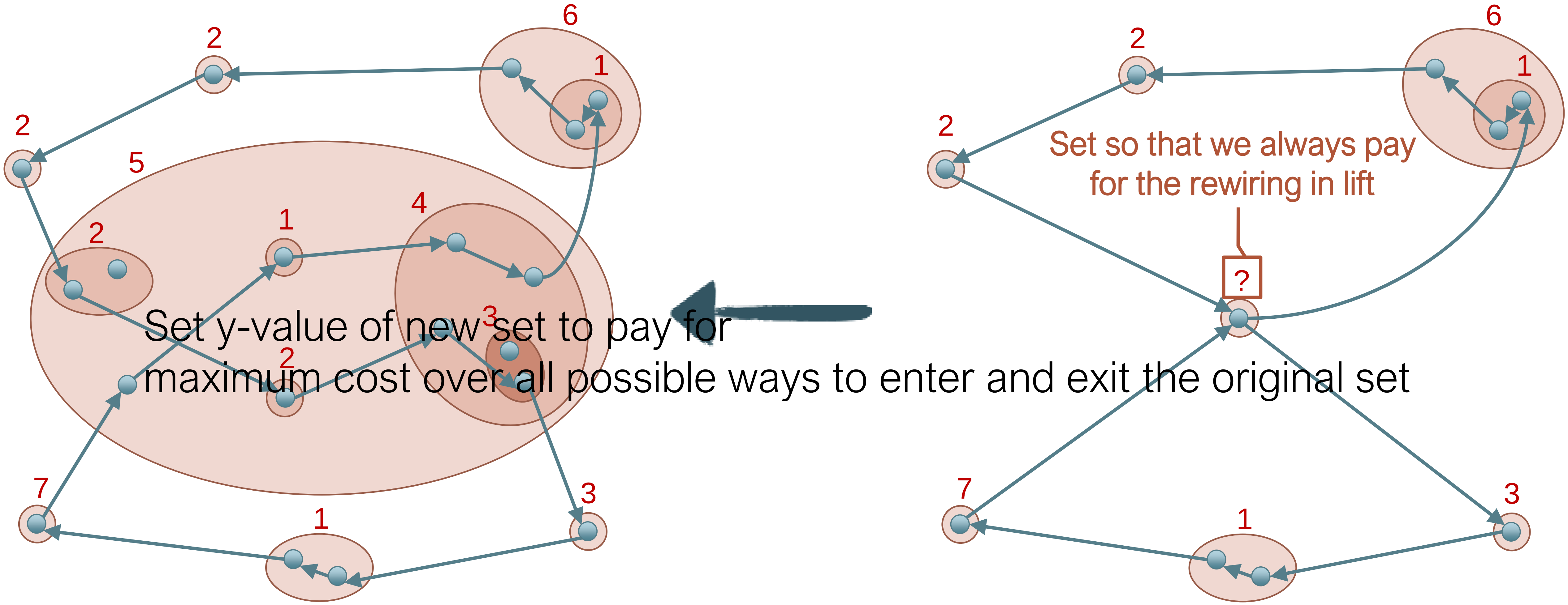


Set so that we always pay for the rewiring in lift

What to do?

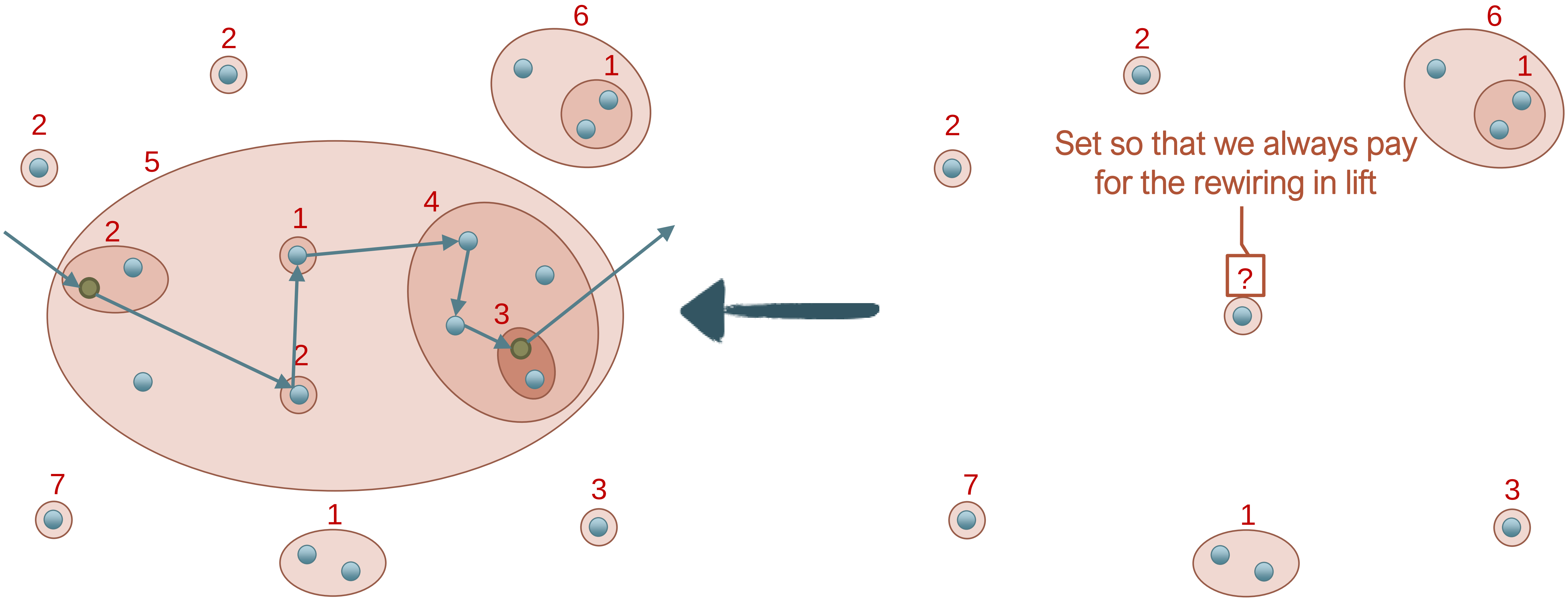
Simply add a shortest path

Lift tour in contracted instance to subtour in original instance



Set so that we always pay for the rewiring in lift

Set y-value of new set to pay for
maximum cost over all possible ways to enter and exit the original set

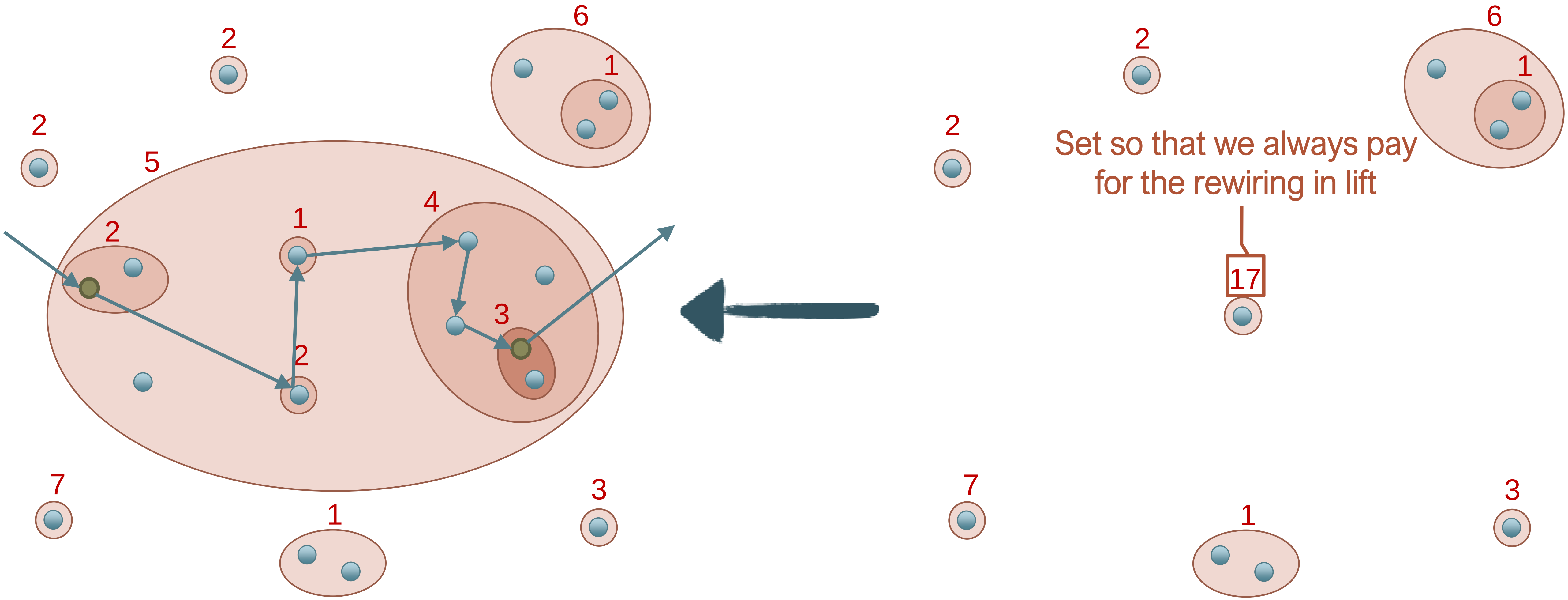


Set so that we always pay for the rewiring in lift

Set y-value of new set to pay for
maximum cost over all possible ways to enter and exit the original set

In example:

? = 5+2+2+1+4+3 = 17 *(path crosses every tight set)*


Set so that we always pay for the rewiring in lift

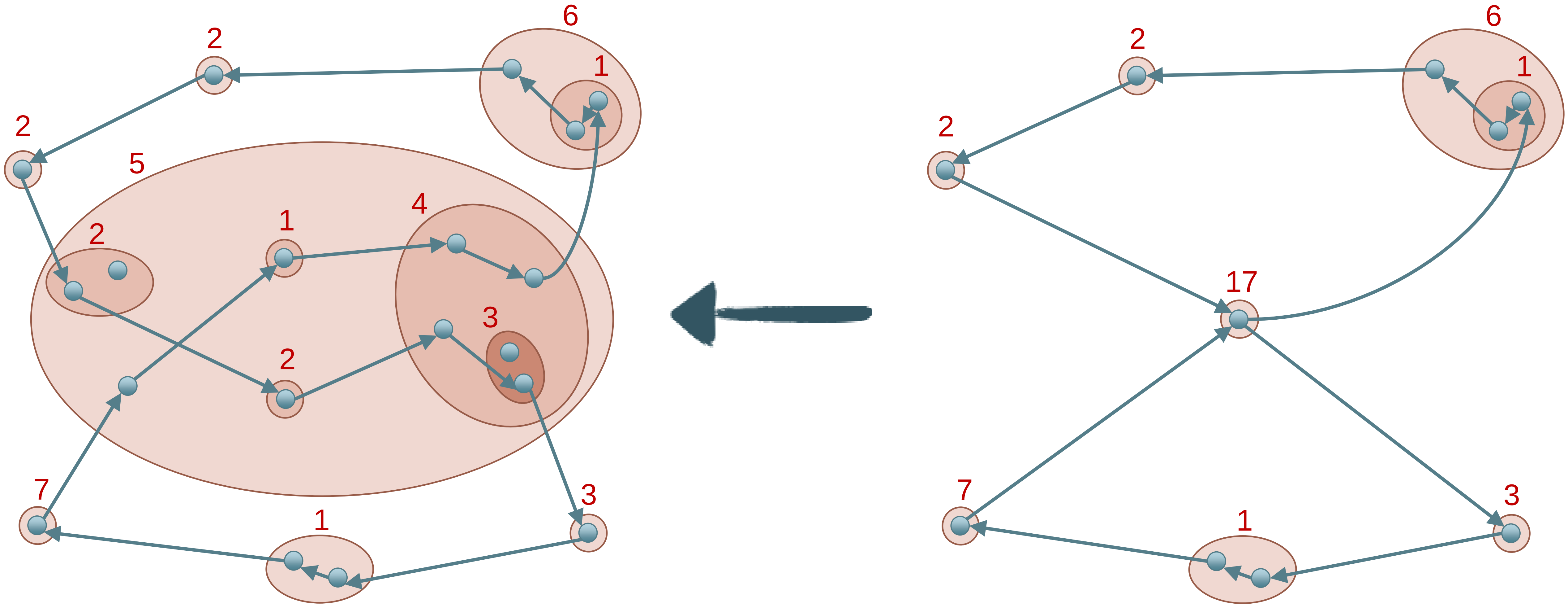
Set y-value of new set to pay for
maximum cost over all possible ways to enter and exit the original set

In example:

? = 5+2+2+1+4+3 = 17 *(path crosses every tight set)*

**Fact:** No matter how we enter and exit, there exists a path that enters and exits each set at most once => contraction does not increase LP-value

*Generalization of the fact: if there is a path from u to v then there is one without cycles*



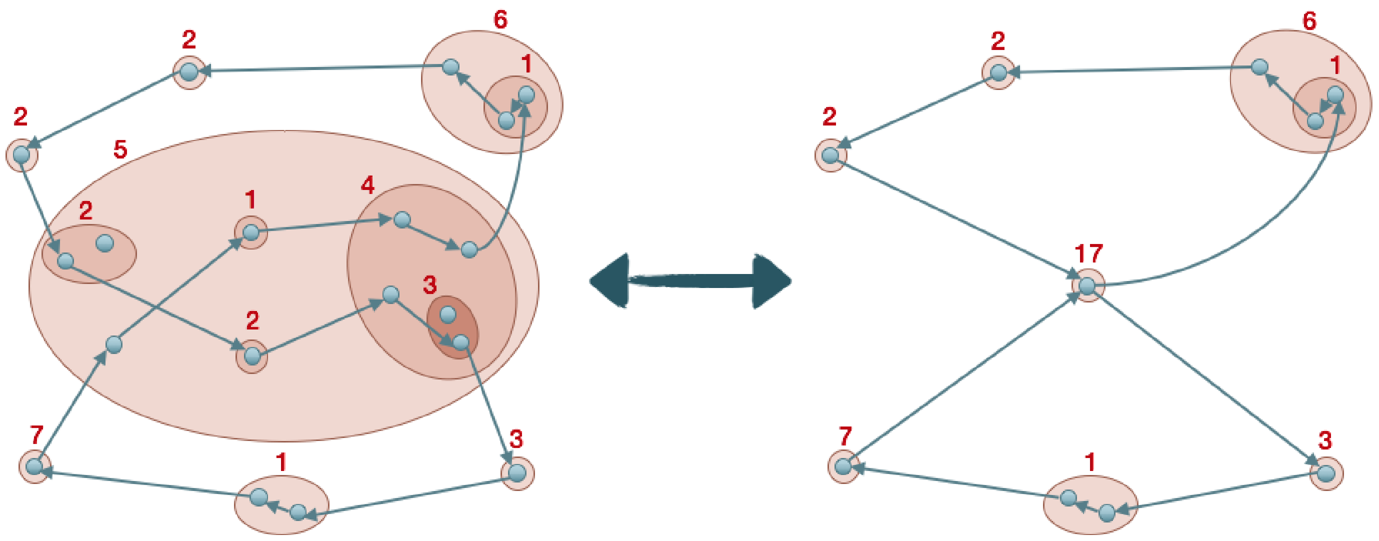
## Change of cost in example:

$$\underbrace{2*(5+2+2+4+3)}_{\text{cost of first visit in lift}} - \underbrace{2*17}_{\text{cost of first visit in tour}} + \underbrace{2*(5+1+4)}_{\text{cost of 2}^{\text{nd}}\text{ visit in lift}} - 2*\underbrace{17}_{\text{cost of 2}^{\text{nd}}\text{ visit in tour}} \leq 0$$

## By design:

**Fact:** Lift no more expensive than tour in contracted instance

# Facts about contraction



**Fact:** No matter how we enter and exit, there exists a path that enters and exits each set at most once => **contraction does not increase LP-value**

**Fact:** Lift no more expensive than tour in contracted instance

Lift is a subtour but may not be a tour:

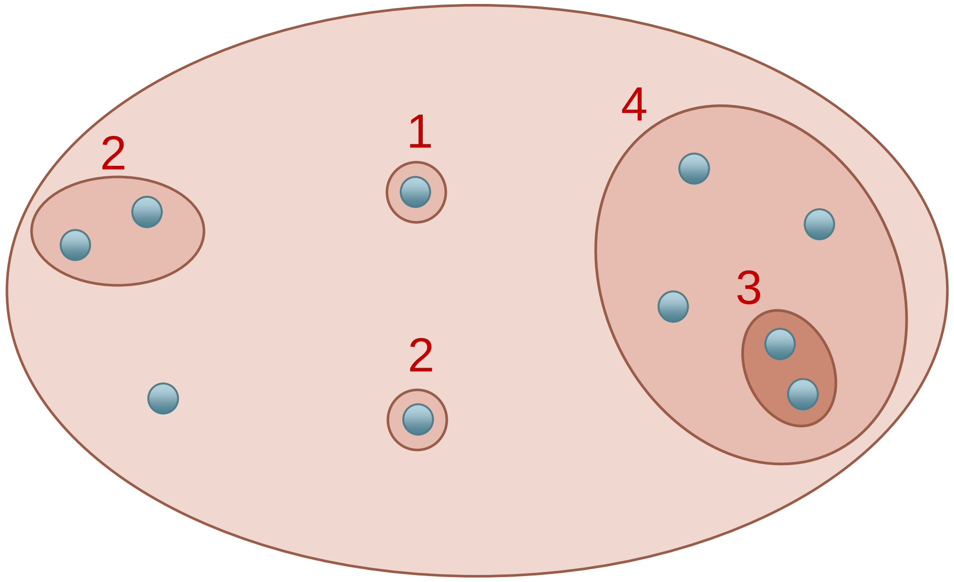
it visits all vertices outside contracted set but not inside

However, if contraction **causes significant decrease** in value, then we can **use remaining budget to complete** the lift into tour

# Implementing recursive strategy

# (Ir)reducible sets in $\mathcal{L}$

**DEF:** A set $S \in \mathcal{L}$ is *reducible* if worst way to enter/exit crosses at most a weighted $\frac{3}{4}$-fraction of the sets strictly inside $S$

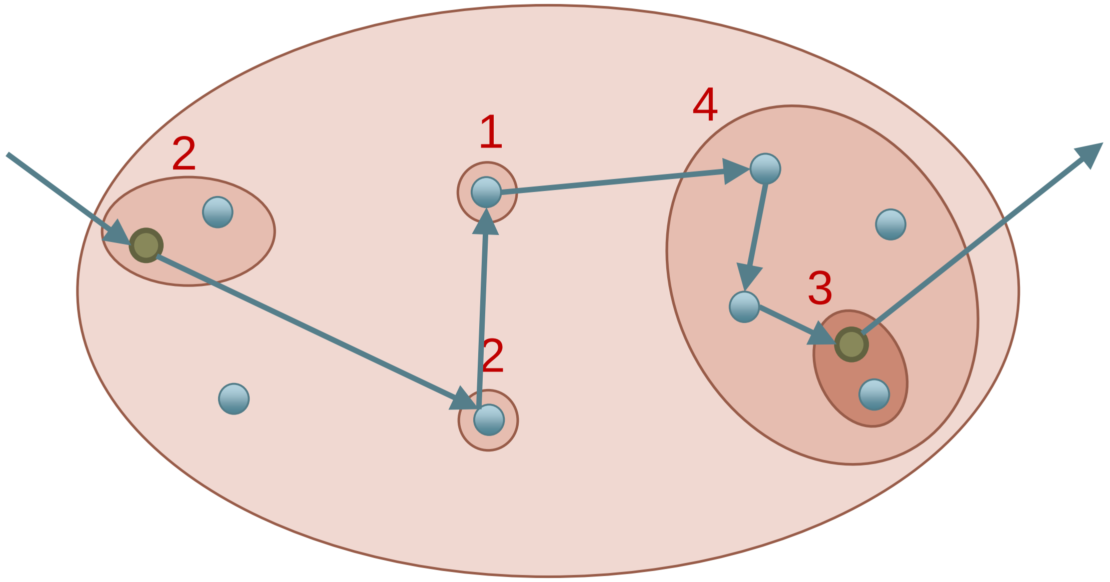


Total value inside $S$ = 2+2+1+4+3 = 12

So worst way to enter/exit should cross sets of value **at most 9** to be reducible

# (Ir)reducible sets in $\mathcal{L}$

**DEF:** A set $S \in \mathcal{L}$ is *reducible* if worst way to enter/exit crosses at most a weighted $\frac{3}{4}$-fraction of the sets strictly inside $S$



Total value inside $S$ = 2+2+1+4+3 = 12

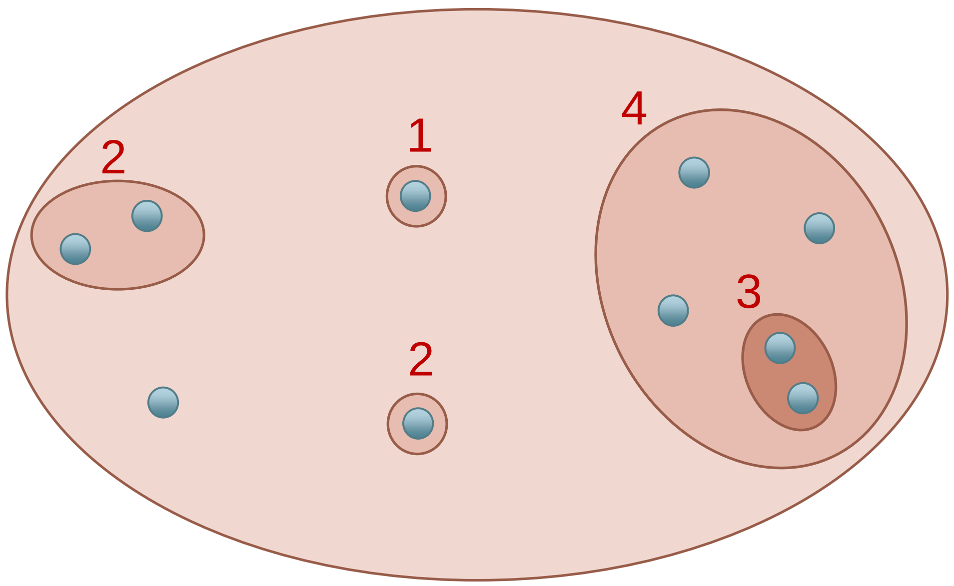
So worst way to enter/exit should cross sets of value **at most 9** to be reducible

Worst way to enter/exit crosses sets of **value = 12**

IRREDUCIBLE

# (Ir)reducible sets in $\mathcal{L}$

**DEF:** A set $S \in \mathcal{L}$ is *reducible* if worst way to enter/exit crosses at most a weighted $\frac{3}{4}$-fraction of the sets strictly inside $S$



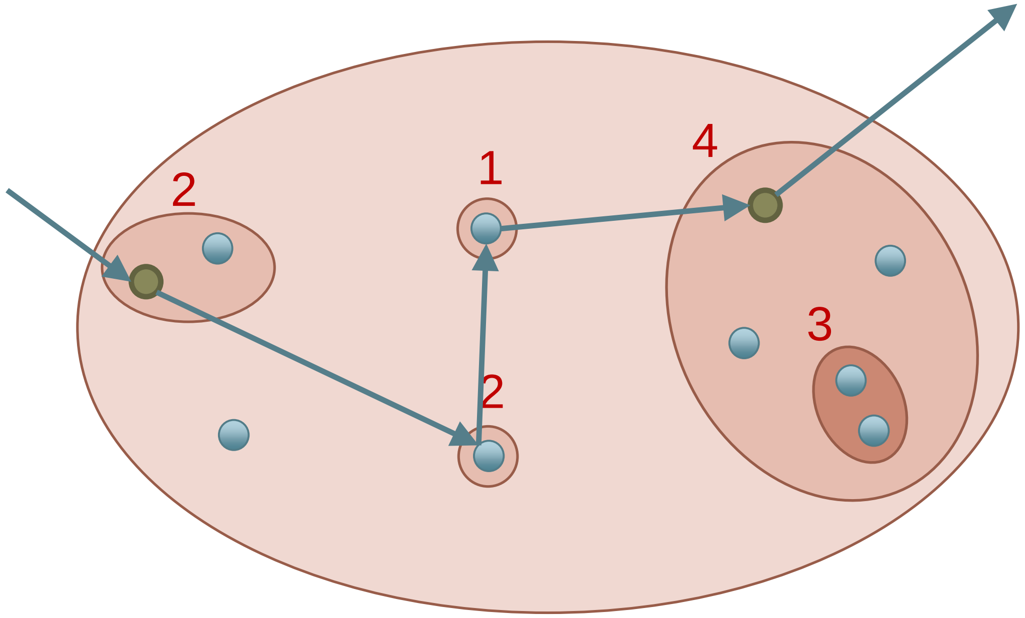
Total value inside $S$ = 2+2+1+4+3 = 12

So worst way to enter/exit should cross sets of value **at most 9** to be reducible

# (Ir)reducible sets in $\mathcal{L}$

**DEF:** A set $S \in \mathcal{L}$ is *reducible* if worst way to enter/exit crosses at most a weighted $\frac{3}{4}$-fraction of the sets strictly inside $S$

We say that an instance is irreducible if no set in $\mathcal{L}$ is *reducible*



Total value inside $S$ = 2+2+1+4+3 = 12

So worst way to enter/exit should cross sets of value **at most** 9 to be reducible

Worst way to enter/exit crosses sets of **value** = 9

REDUCIBLE

## Theorem:

A **ρ-approximation algorithm for irreducible instances** yields a **8ρ-approximation algorithm** for laminarly-weighted instances, and thus for general ATSP

Let $\mathcal{A}$ be a $\rho$-approximation algorithm for irreducible instances…

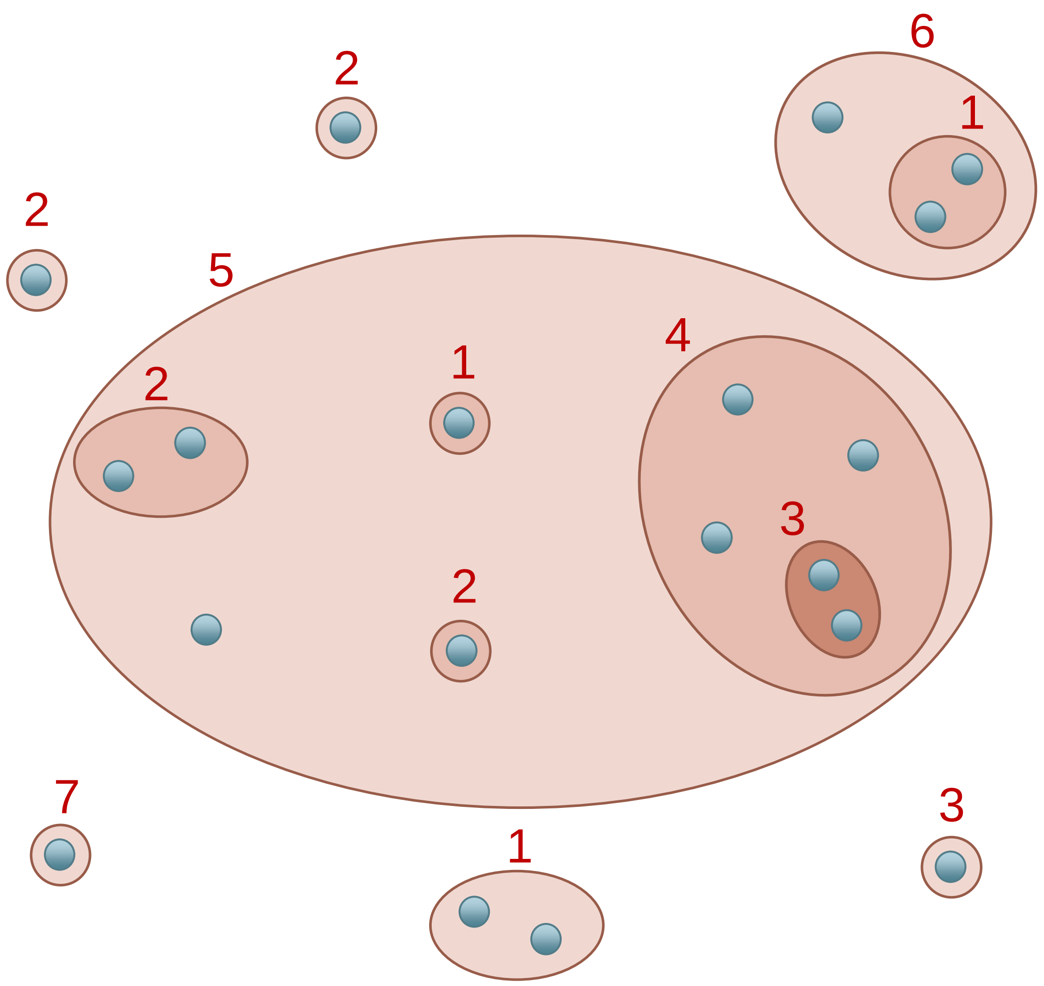
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



If irreducible:
simply run $\mathcal{A}$ to obtain $\rho$-approximate tour
($\rho < \ 8\rho$, so okay)

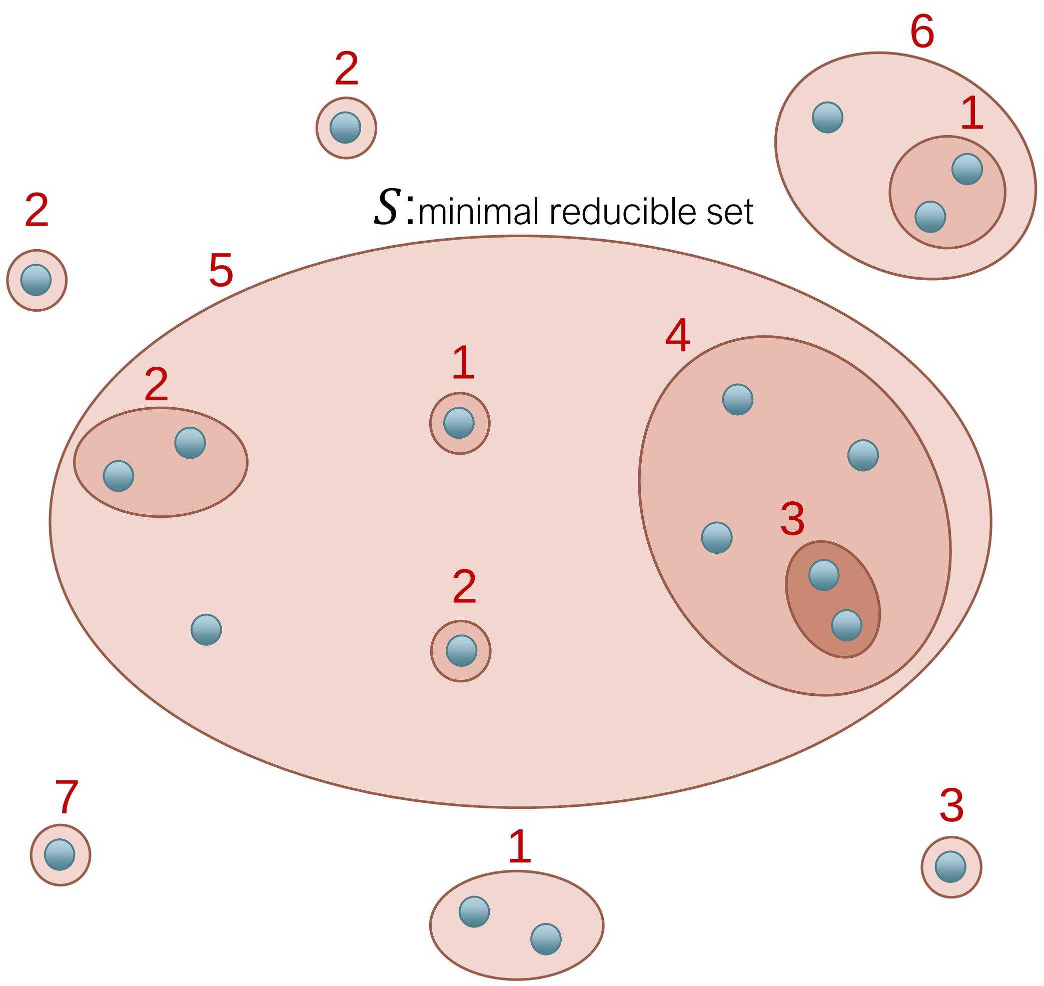
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



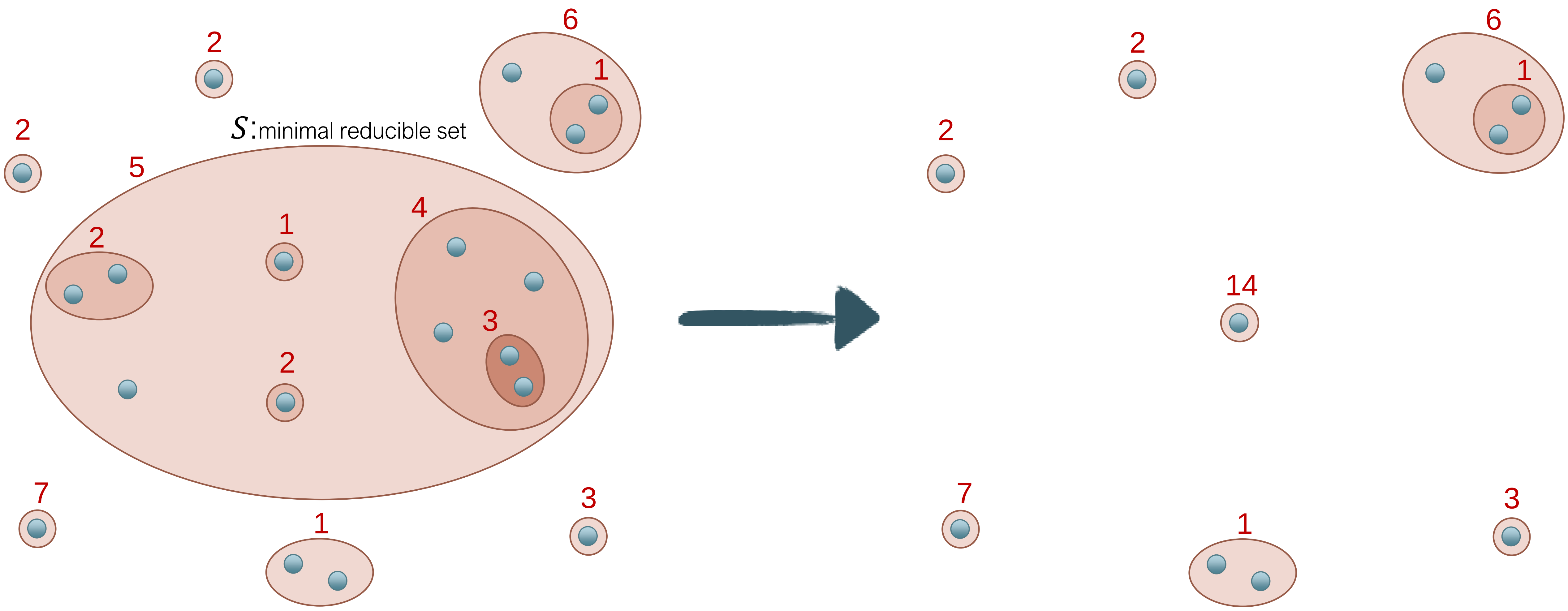
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



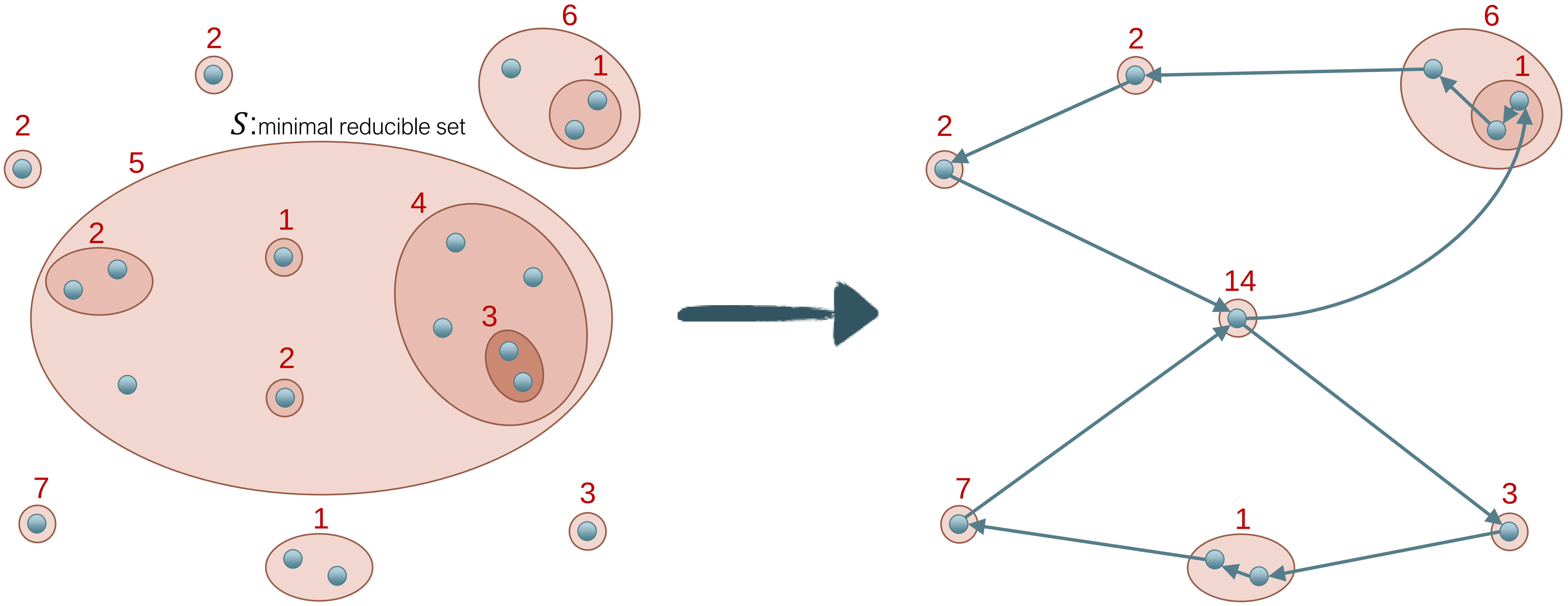
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



Recursive call returns $8\rho$-approximate solution $T$ on smaller instance:

$$w(T) \le 8\rho\left(OPT - \frac{1}{4}\left(2\cdot\sum_{R\in\mathcal{L}:R\subset S} y_R\right)\right) = 8\rho OPT - 2\rho\left(2\cdot\sum_{R\in\mathcal{L}:R\subset S} y_R\right)$$

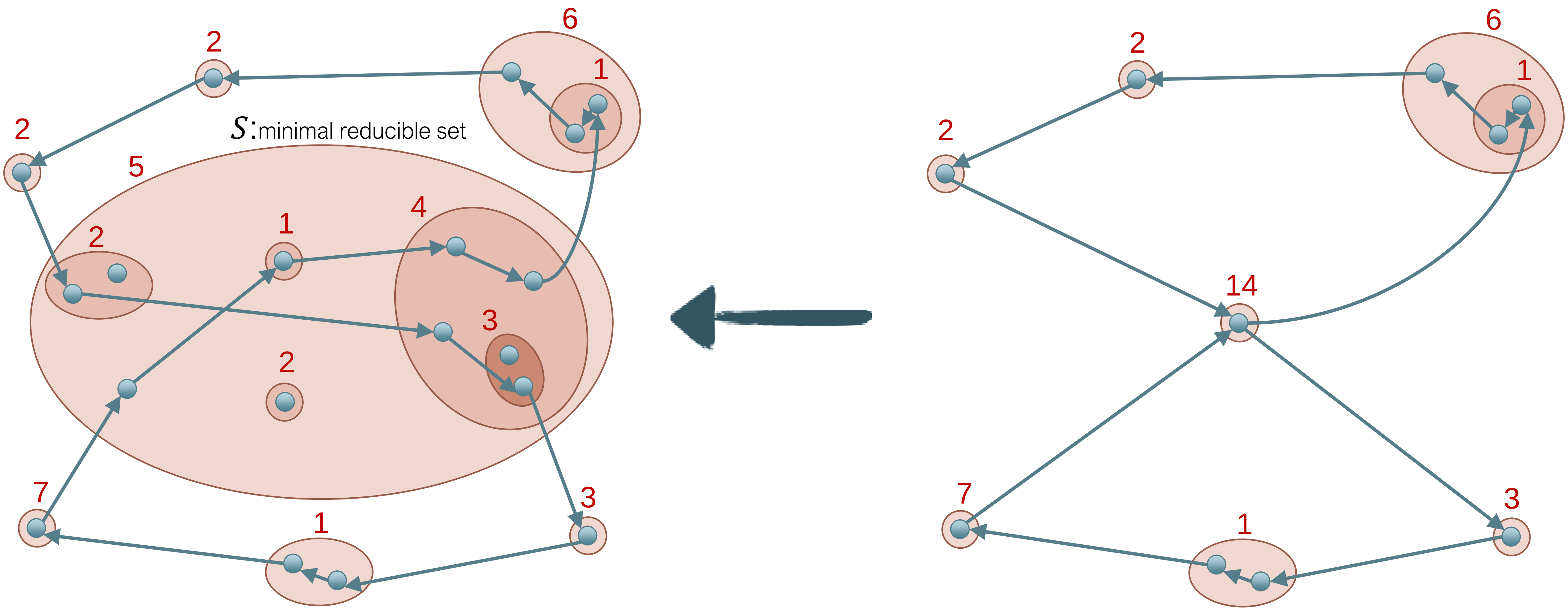
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



Recursive call returns $8\rho$-approximate solution $T$ on smaller instance:

$$w(lift) \le w(T) \le 8\rho\left(OPT - \frac{1}{4}\left(2\cdot\sum_{R\in\mathcal{L}:R\subset S} y_R\right)\right) = 8\rho OPT - \underbrace{2\rho\left(2\cdot\sum_{R\in\mathcal{L}:R\subset S} y_R\right)}$$

**Remaining task:** complete lift to a tour using $\mathcal{A}$ while paying at most the above

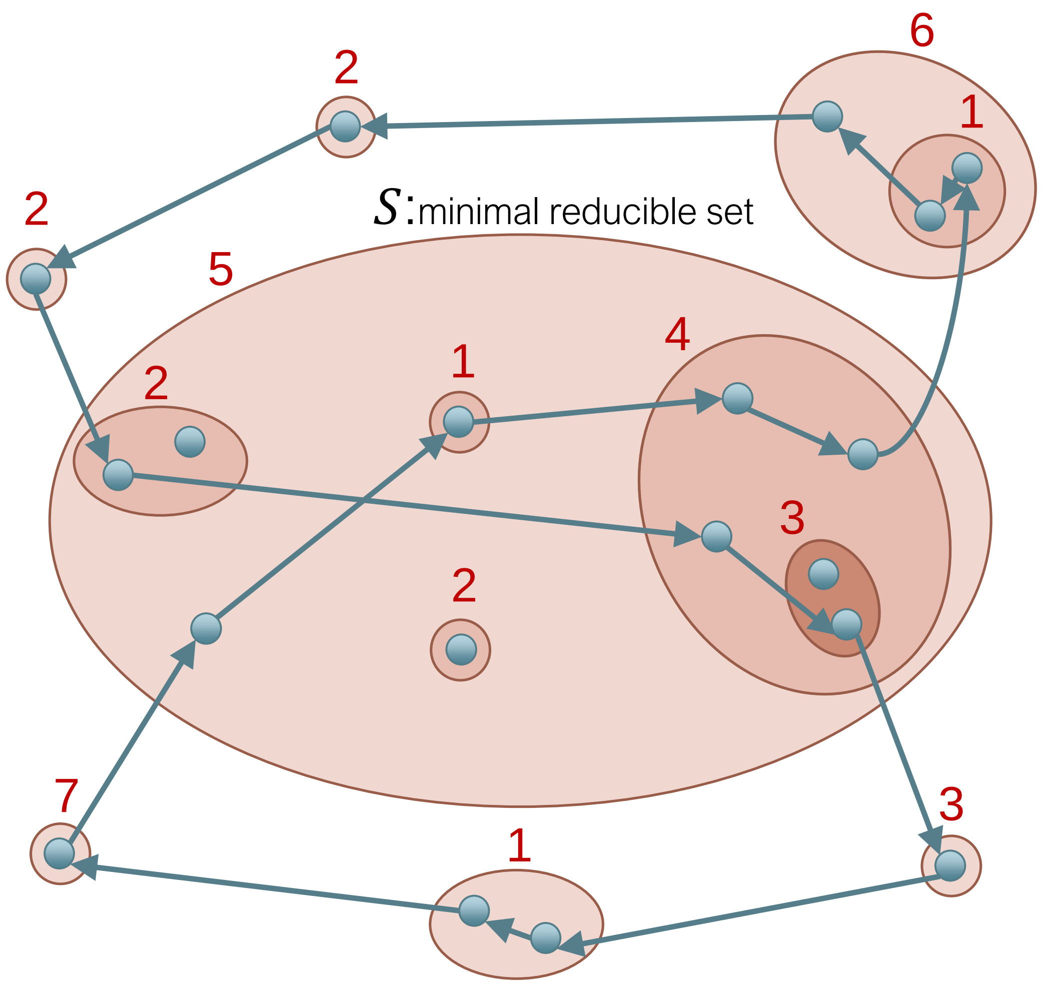
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



**Task:** complete to tour while paying at most $2\rho( 2 \cdot \sum_{R\in\mathcal{L}:R \subset S} y_R)$

- We need to only connect unvisited vertices inside $S$

**Simplifying assumption:**

instance obtained by restricting to vertices inside S is feasible

# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

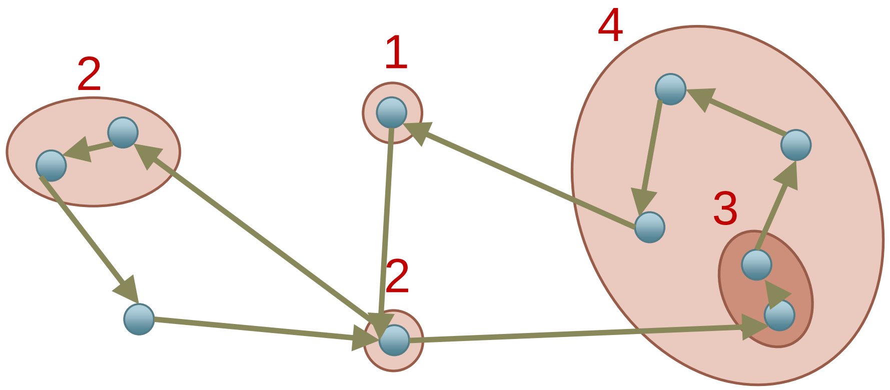
Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$

**Task:** complete to tour while paying at most $2\rho(\, 2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R)$

$S$:minimal reducible set



- We need to only connect unvisited vertices inside $S$

**Simplifying assumption:**

instance obtained by restricting to vertices inside S is feasible

An **irreducible instance** since $S$ was a *minimal* reducible set

Held-Karp value = 2 times dual values

$$= 2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R$$

$$\left.\right\}$$

Solve this instance with $\mathcal{A}$ to find tour on $S$ of weight

$$\leq \rho \cdot \left( 2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R \right)$$

Better by a factor 2 than needed

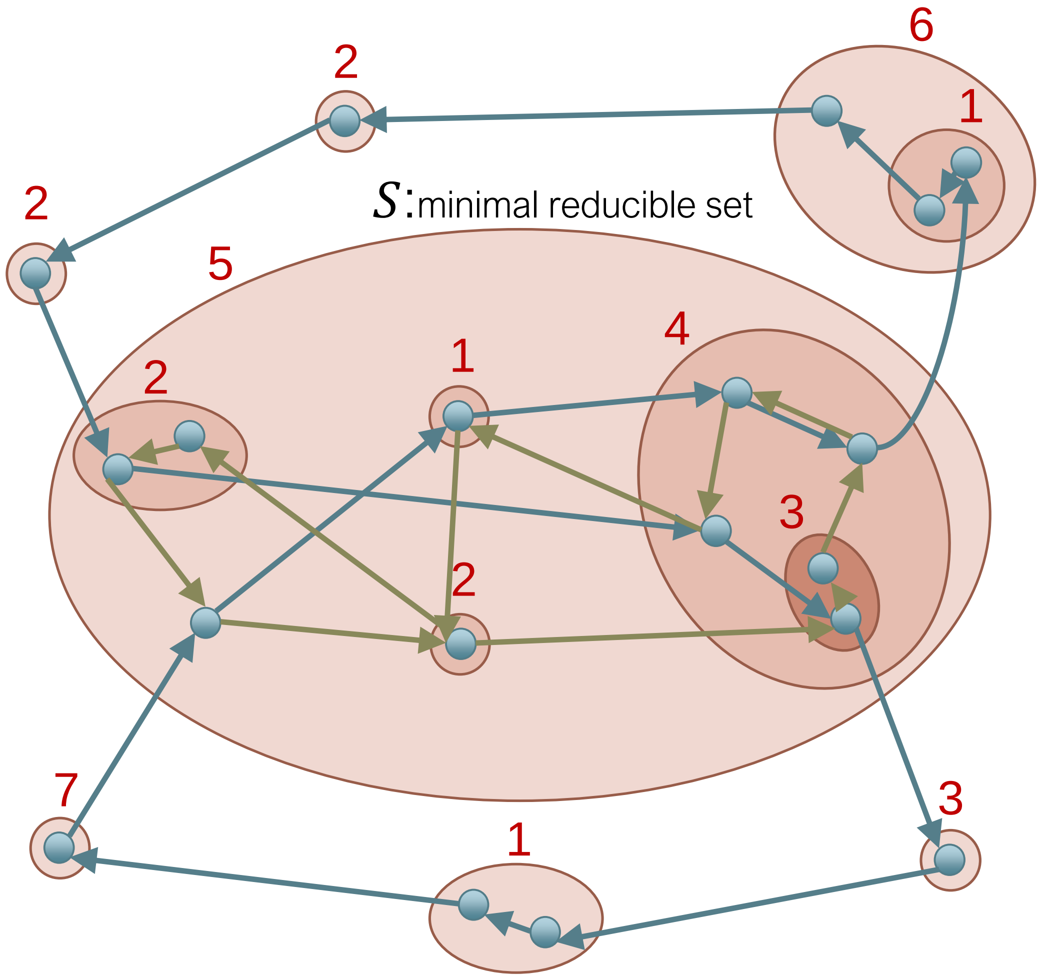
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



Contract and recursively find lift (subtour) of weight

$$\leq 8\rho OPT - 2\rho\left(2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R\right)$$

Under simplifying assumption, find tour on S of weight

$$\leq \rho\left(2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R\right)$$

Final tour has value at most

$$\leq 8\rho OPT \; - \; \rho\left(2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R\right)$$

**Simplifying assumption not true in general:**

We define the operation of inducing on $S$ for ATSP in paper. Makes us lose another factor of 2

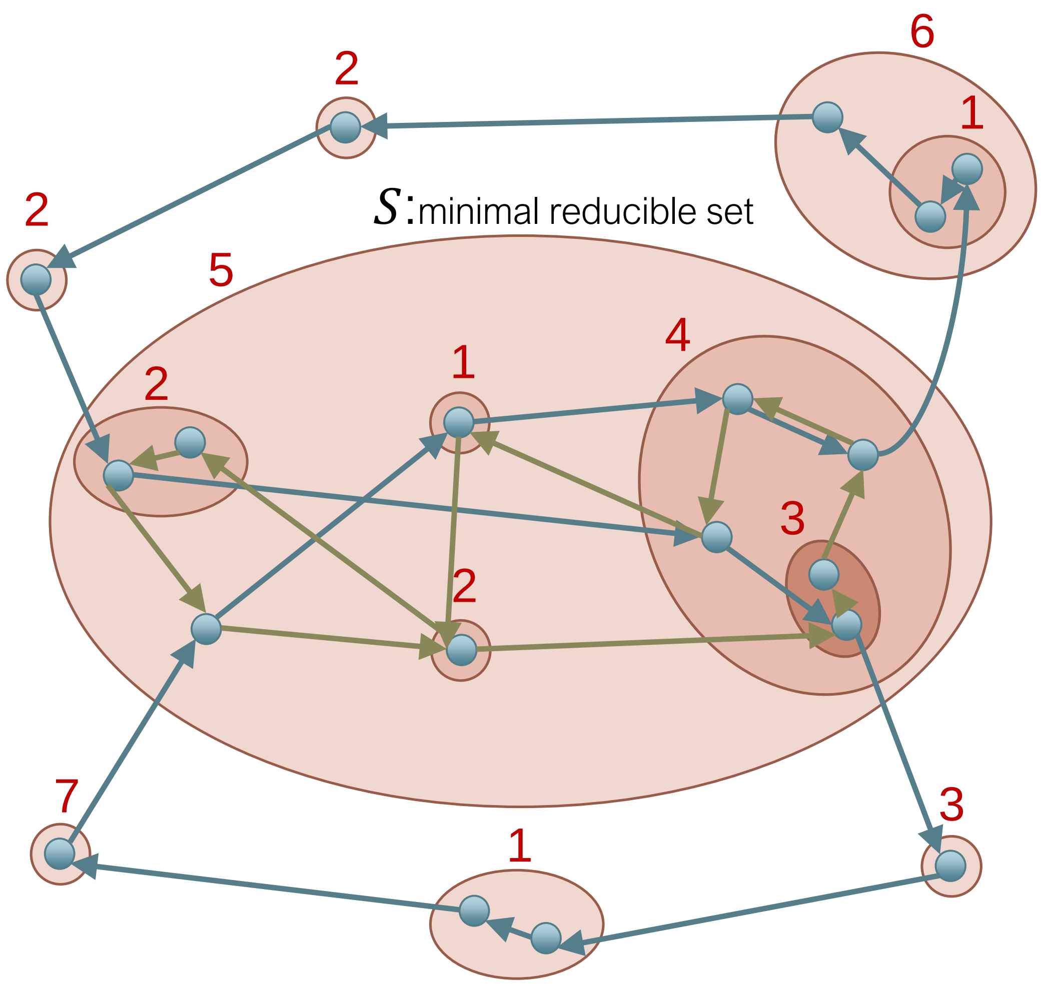
# Alg for reducible instances

If instance is irreducible, simply run $\mathcal{A}$

Otherwise select *minimal* reducible set $S \in \mathcal{L}$

Recursively find tour $T$ in instance with $S$ contracted

Complete lift of $T$ to a tour in original instance using $\mathcal{A}$



Contract and recursively find lift (subtour) of weight

$$\leq 8\rho OPT - 2\rho\left(2\cdot\sum_{R\in\mathcal{L}:R\subset S} y_R\right)$$

Eulerian set of edges

Under simplifying assumption, find ~~tour~~ on S of weight

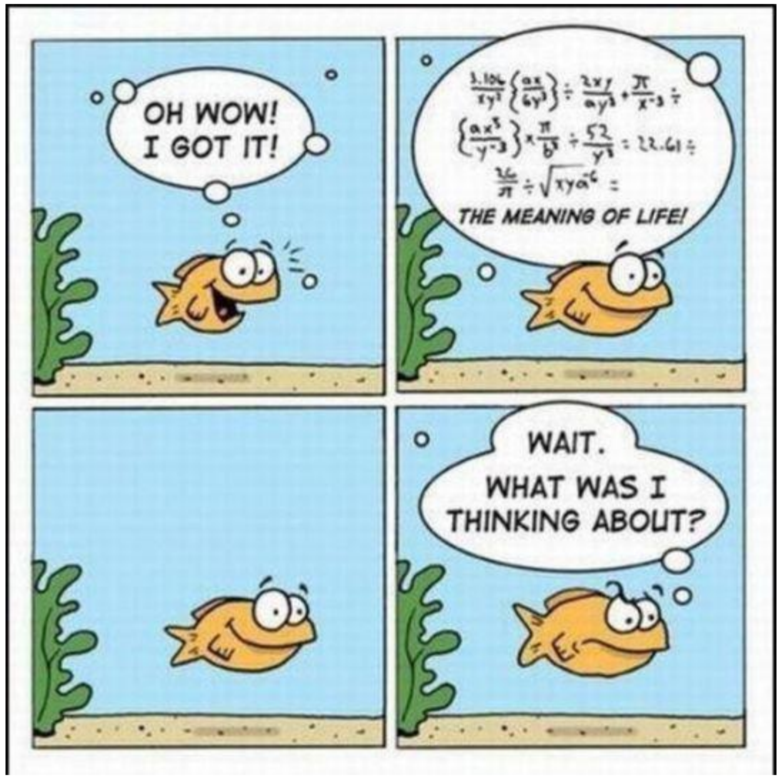
$$\leq \rho\left(2\cdot\sum_{R\in\mathcal{L}:R\subset S} y_R\right) * 2$$

Final tour has value at most

$$\leq 8\rho OPT$$

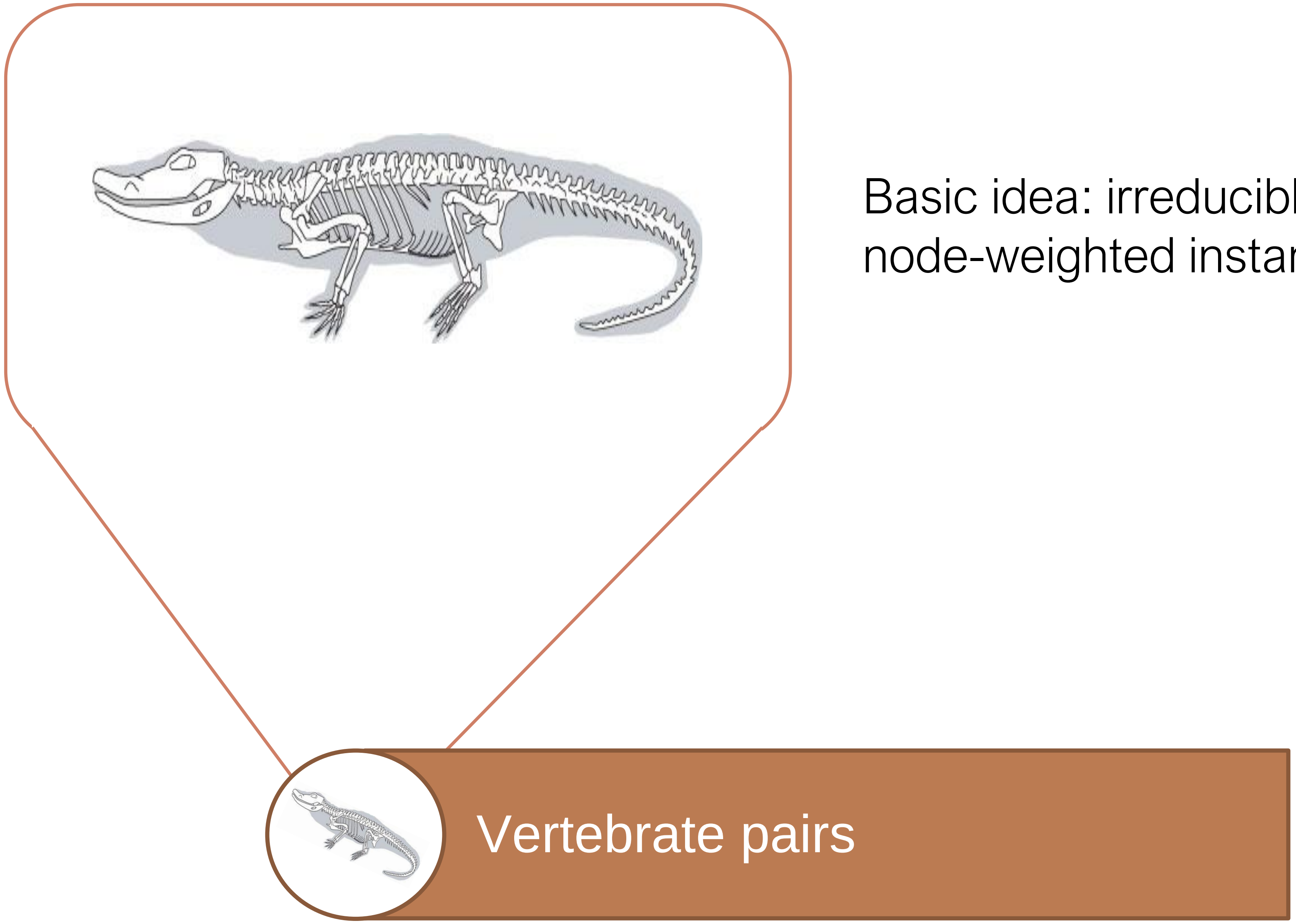
**Simplifying assumption not true in general:**

We define the operation of inducing on $S$ for ATSP in paper. Makes us lose another factor of 2



Theorem:

A **ρ-approximation algorithm for irreducible instances** yields a **8ρ-approximation algorithm** for laminarly-weighted instances, and thus for general ATSP



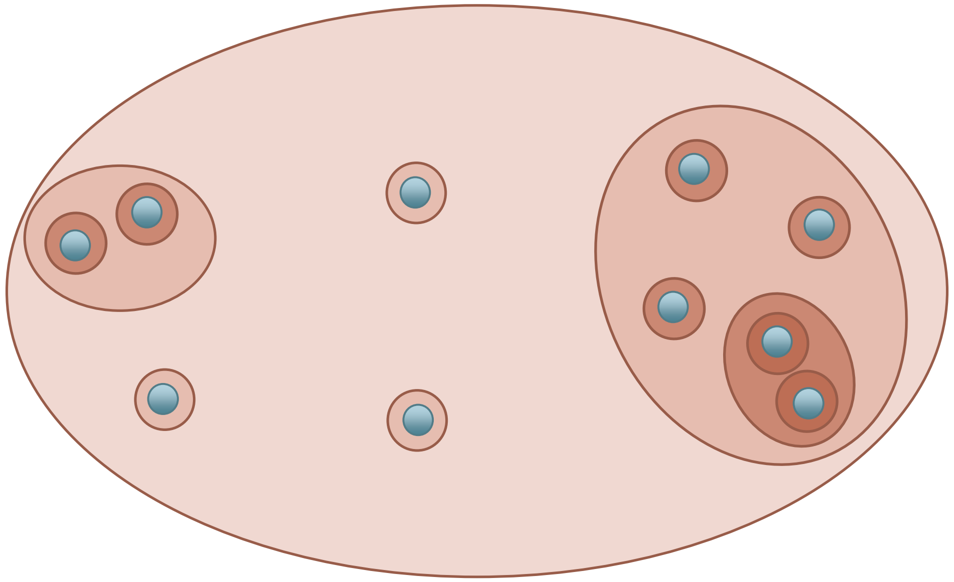
Vertebrate pairs

Basic idea: irreducible instances are almost node-weighted instances

# Simplifying assumptions

- $\mathcal{L}$ contains all singletons (every vertex has a node-weight)
- The instance is perfectly irreducible:

  the contraction of any set causes *no* decrease in LP-value



When contracting a set, the LP-decrease is proportional to #sets not crossed by path in worst way to enter/exit

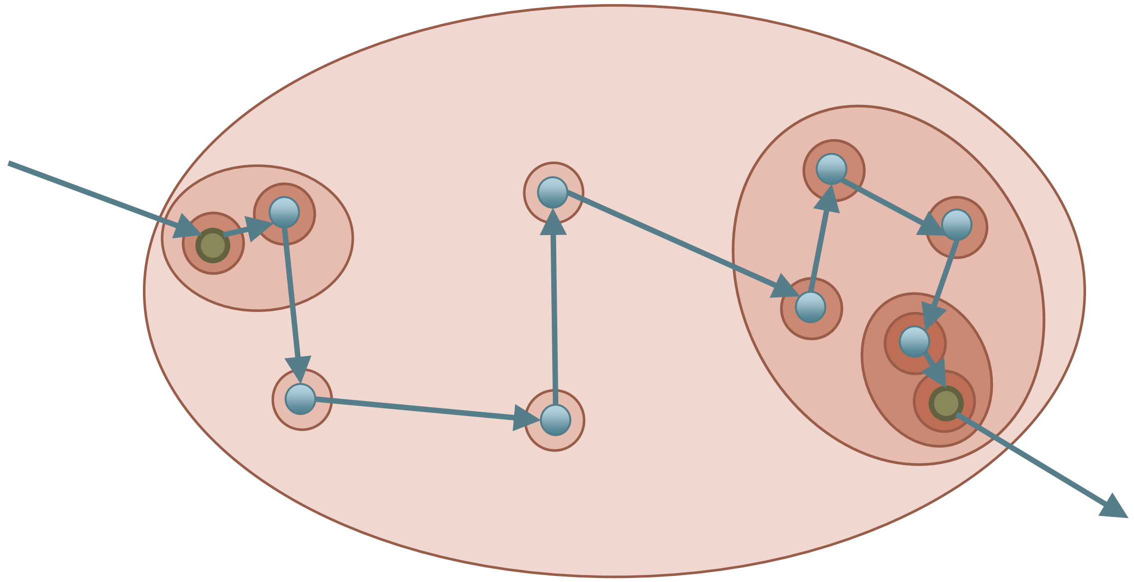
Since all singletons in $\mathcal{L}$ and no LP-decrease, worst way to enter/exit must visit all vertices!

# Simplifying assumptions

- $\mathcal{L}$ contains all singletons (every vertex has a node-weight)

- The instance is perfectly irreducible:

  the contraction of any set causes *no* decrease in LP-value



When contracting a set, the LP-decrease is proportional to #sets not crossed by path in worst way to enter/exit

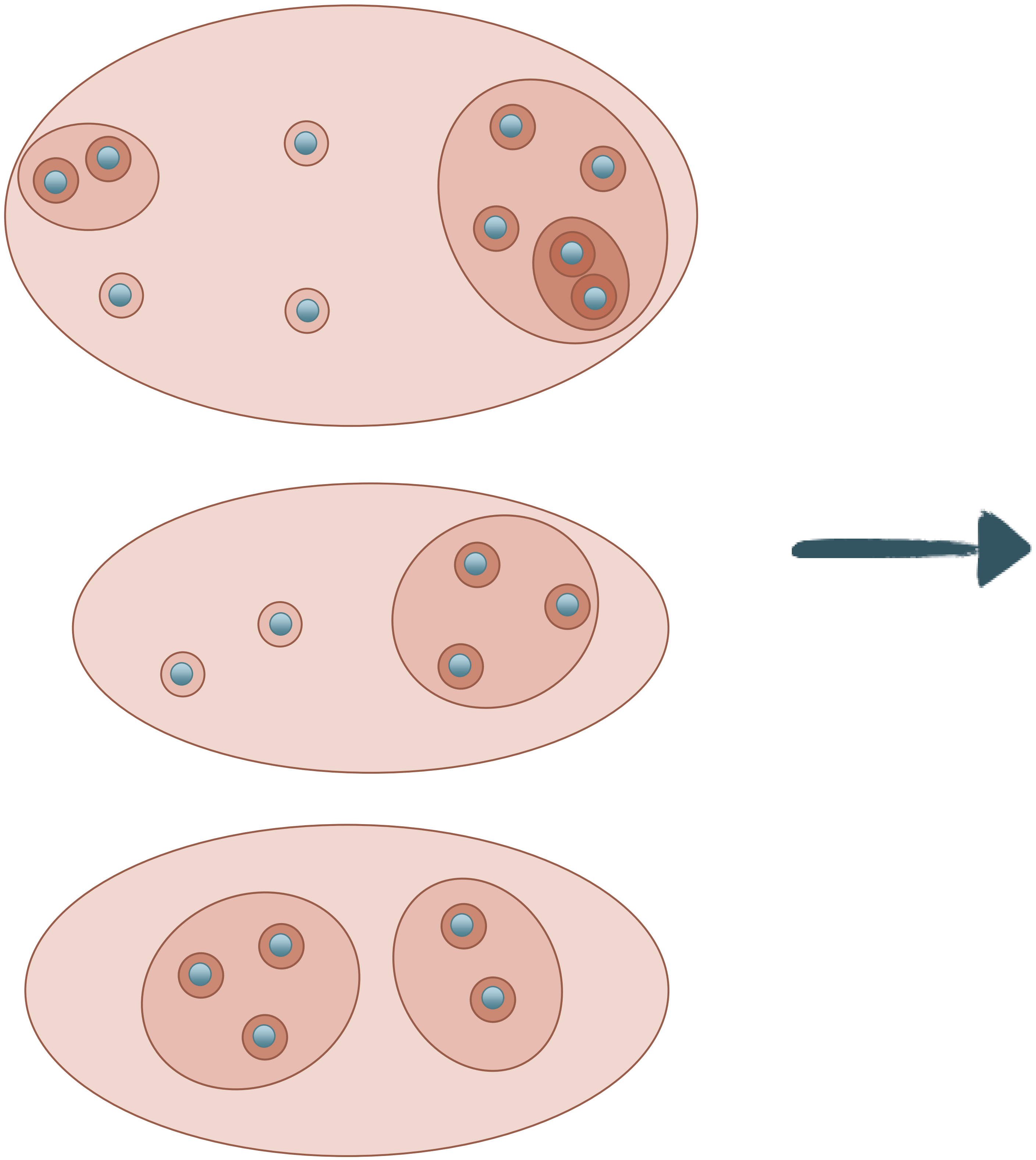
Since all singletons in $\mathcal{L}$ and no LP-decrease, worst way to enter/exit must visit all vertices!

# Alg for perfect irreducible

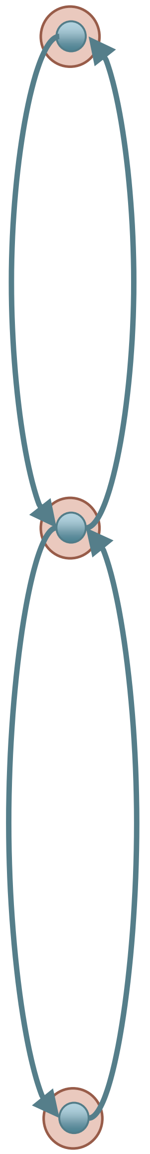
Contract all maximal sets in $\mathcal{L}$

Resulting instance is node-weighted, use Svensson’15 to obtain a 28-approximate tour

Obtain lift of tour and rewire first visit so as to make sure to visit worst enter/exit path



Node-weighted instance
Use 28-approximation by Ola

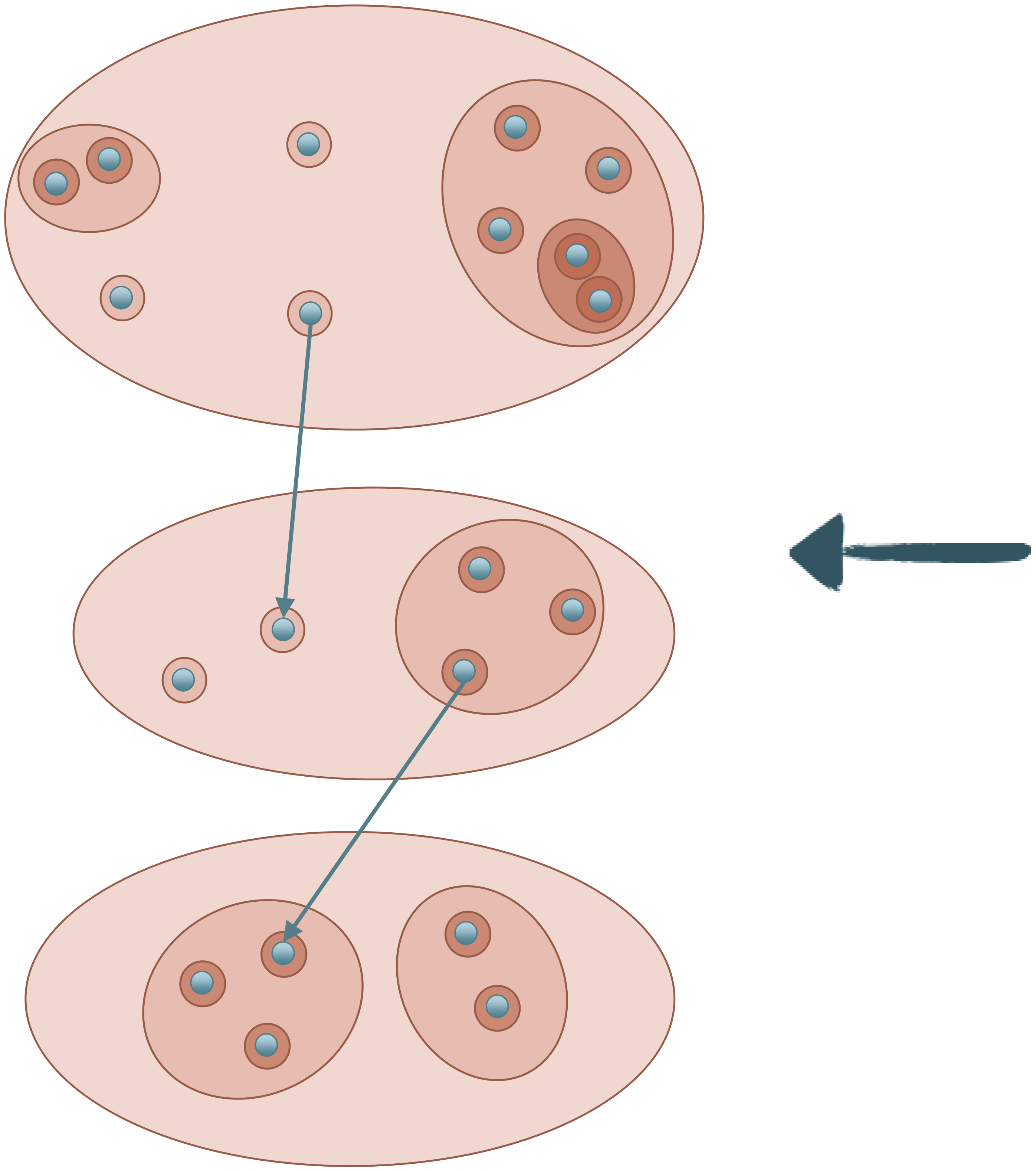


# Alg for perfect irreducible

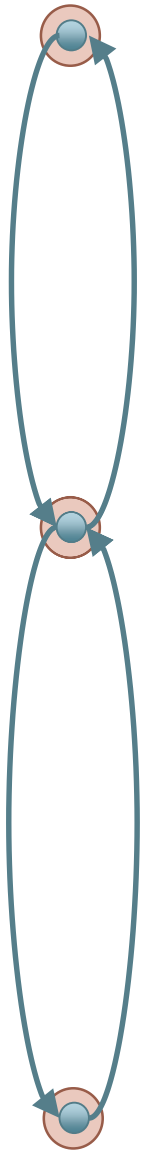
Contract all maximal sets in $\mathcal{L}$

Resulting instance is node-weighted, use Svensson'15 to obtain a 28-approximate tour

Obtain lift of tour and rewire first visit so as to make sure to visit worst enter/exit path



Node-weighted instance
Use 28-approximation by Ola



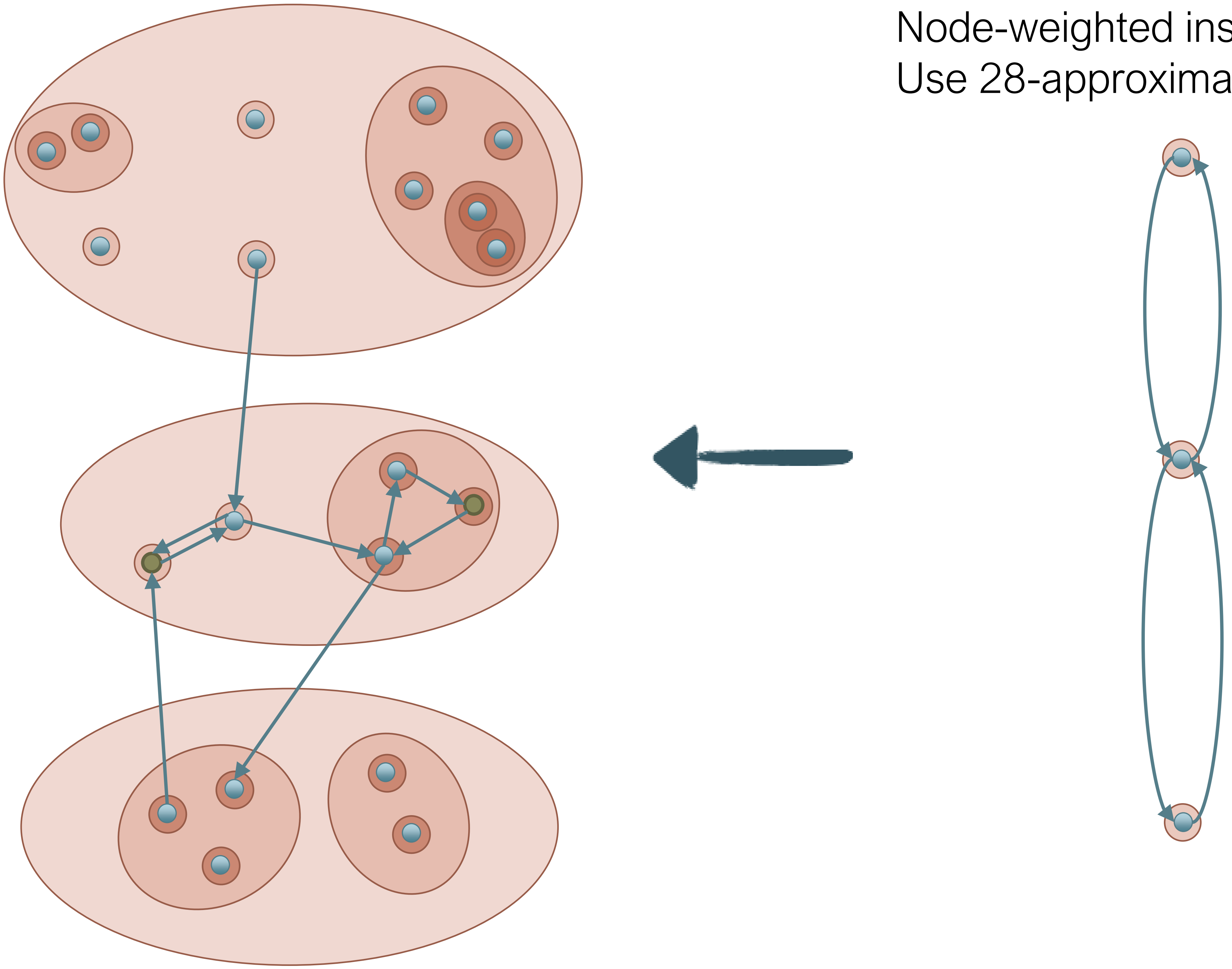
# Alg for perfect irreducible

Contract all maximal sets in $\mathcal{L}$

Resulting instance is node-weighted, use Svensson'15 to obtain a 28-approximate tour

Obtain lift of tour and rewire first visit so as to make sure to visit worst enter/exit path

Node-weighted instance
Use 28-approximation by Ola



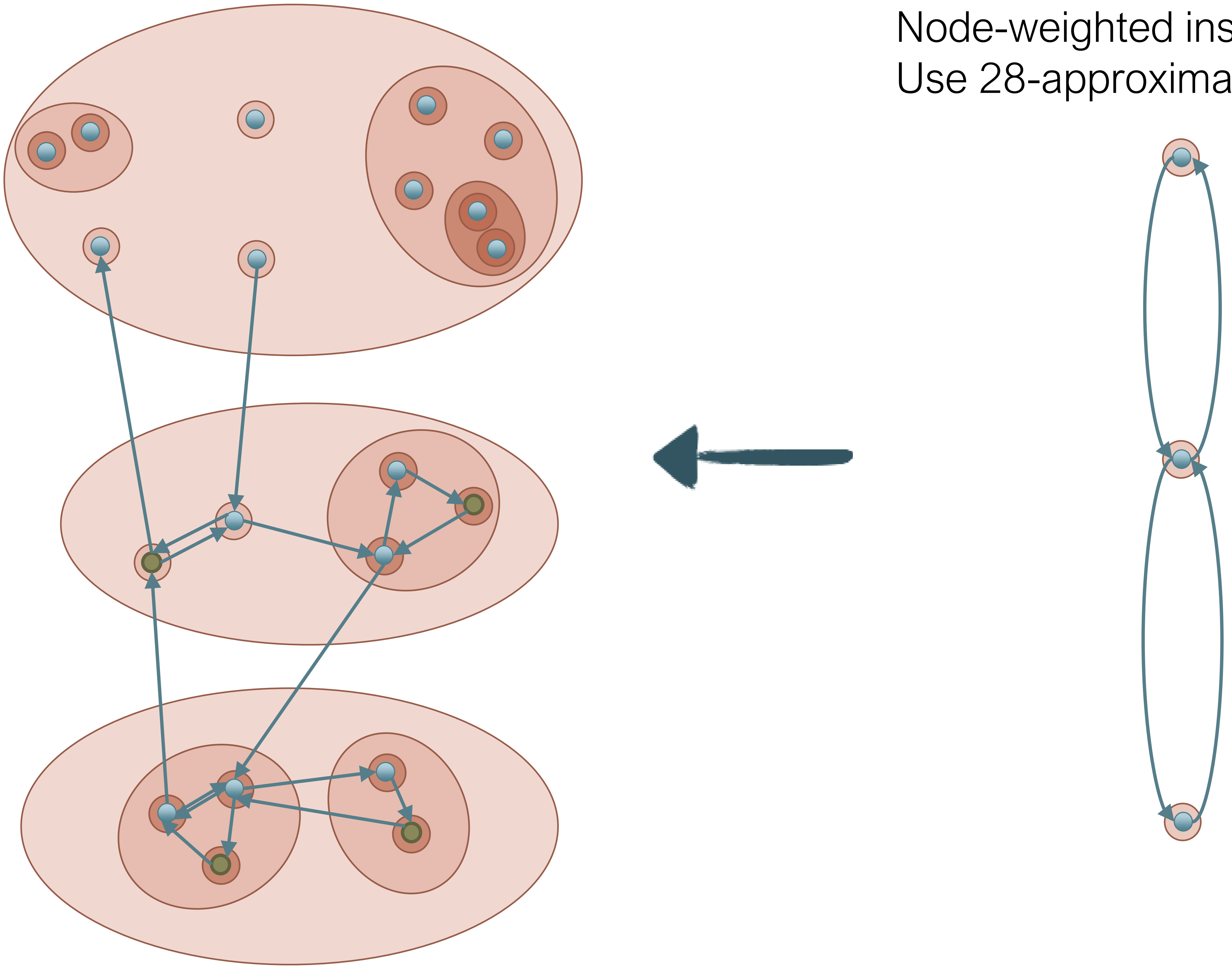
# Alg for perfect irreducible

Contract all maximal sets in $\mathcal{L}$

Resulting instance is node-weighted, use Svensson'15 to obtain a 28-approximate tour

Obtain lift of tour and rewire first visit so as to make sure to visit worst enter/exit path

Node-weighted instance
Use 28-approximation by Ola

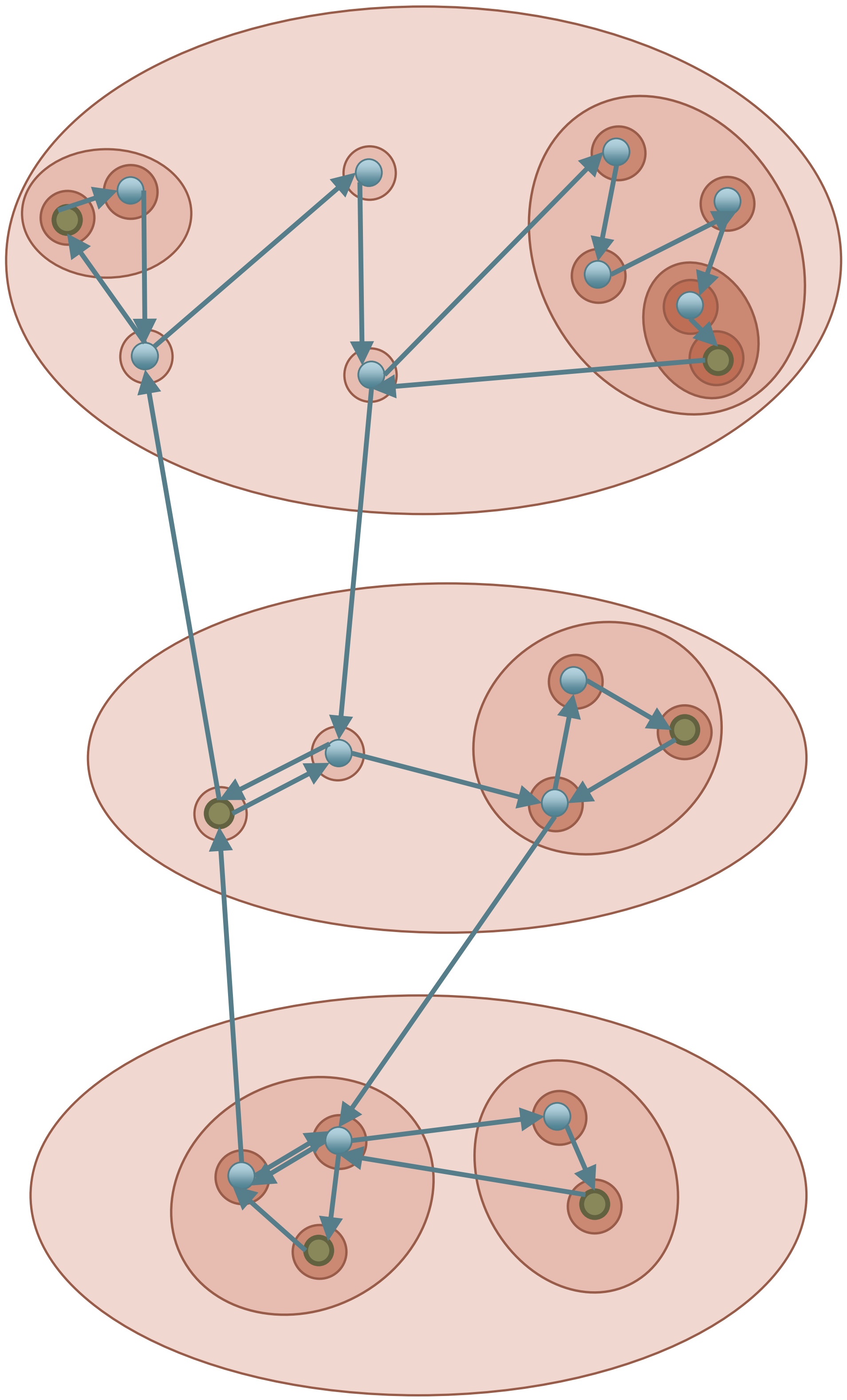


# Alg for perfect irreducible

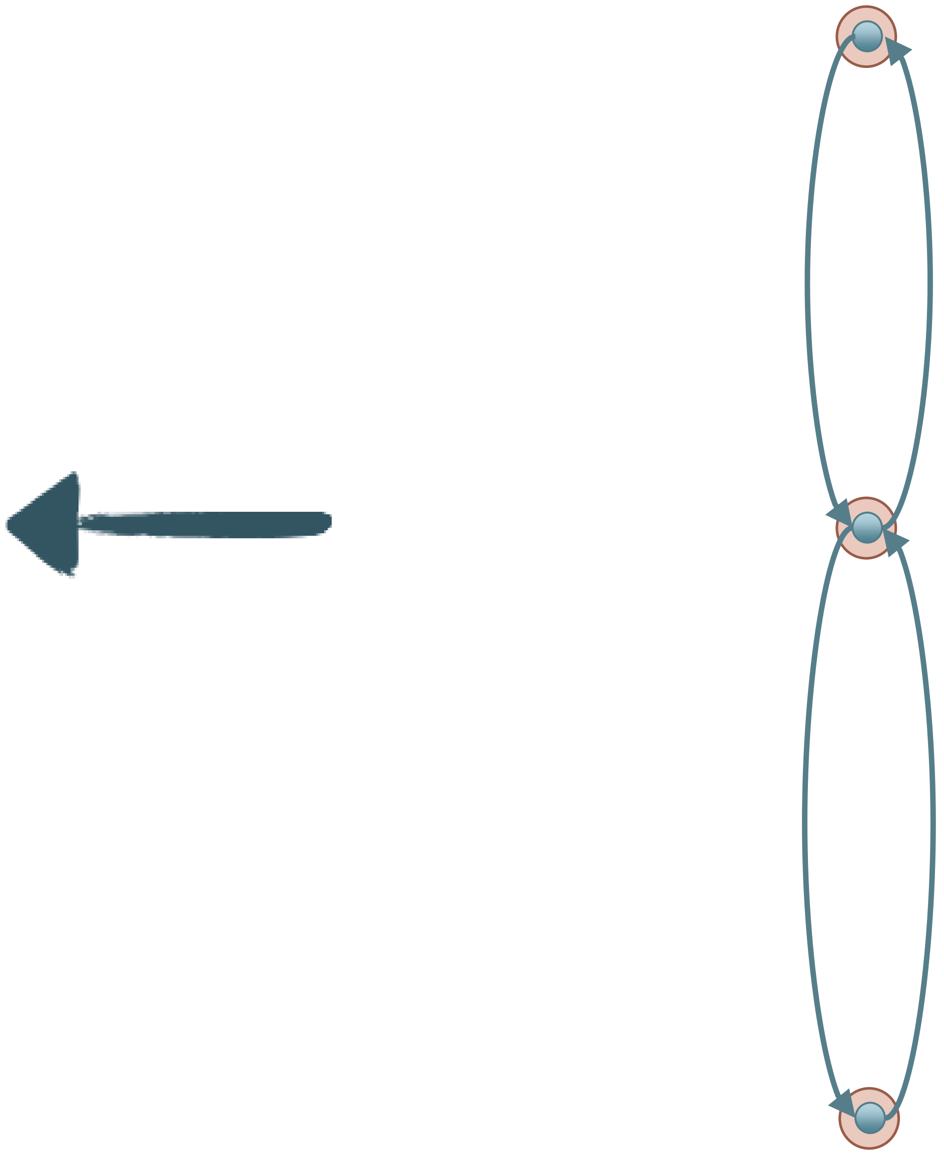
Contract all maximal sets in $\mathcal{L}$

Resulting instance is node-weighted, use Svensson’15 to obtain a 28-approximate tour

Obtain lift of tour and rewire first visit so as to make sure to visit worst enter/exit path



Node-weighted instance
Use 28-approximation by Ola

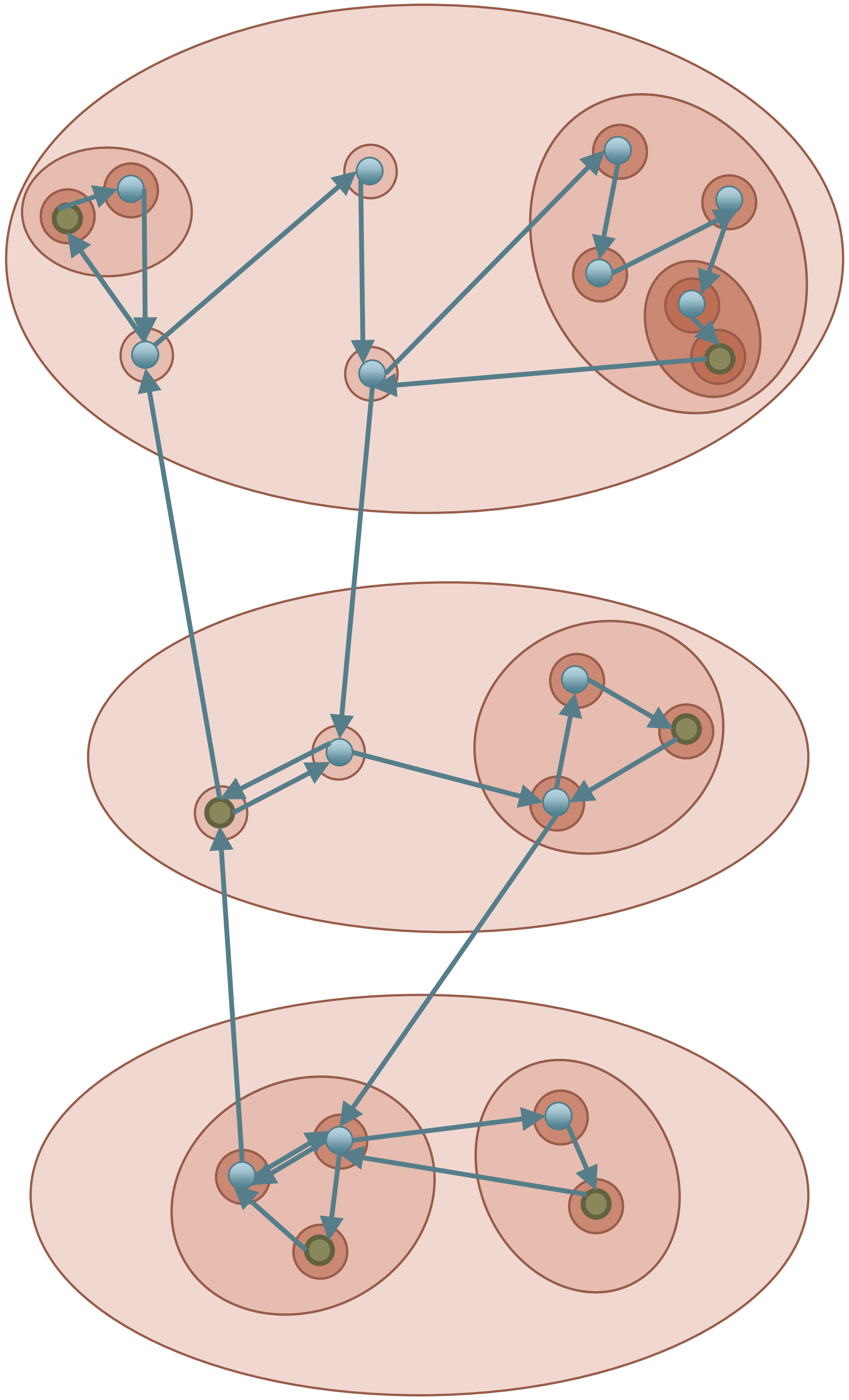


# Alg for perfect irreducible

Contract all maximal sets in $\mathcal{L}$

Resulting instance is node-weighted, use Svensson'15 to obtain a 28-approximate tour

Obtain lift of tour and rewire first visit so as to make sure to visit worst enter/exit path



Cost of tour:

$$w(lift) + w(paths)$$

$$w(lift) \leq 28 \cdot OPT$$

We add 3 paths per maximal set
Cost of each path bounded by the LP-value inside that set

$$w(paths) \leq 3 \cdot OPT$$

Total cost $\leq 31 \cdot OPT$

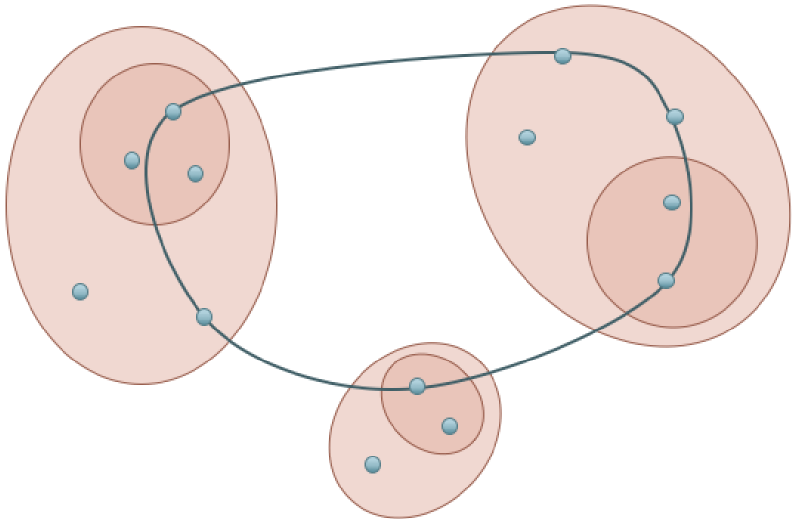
In general not perfect irreducibility:

Worst enter/exit path only crosses most sets in $\mathcal{L}$

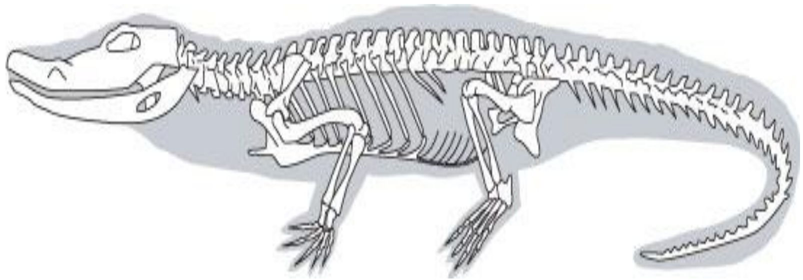
We further reduce to the case when we are given subtour $B$ such that:

- $w(B) \leq 31 \cdot OPT$
- $B$ crosses all non-singleton sets of $\mathcal{L}$

  (to get this, we contract the sets it doesn't cross, and solve them recursively; it's okay because there are few)

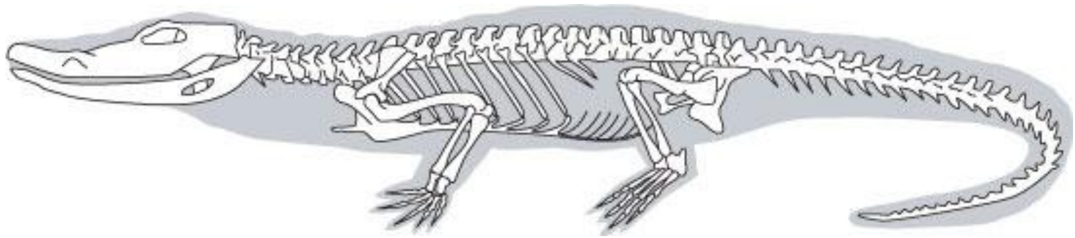


$B$ is called the backbone and together with the instance they form a *vertebrate pair*



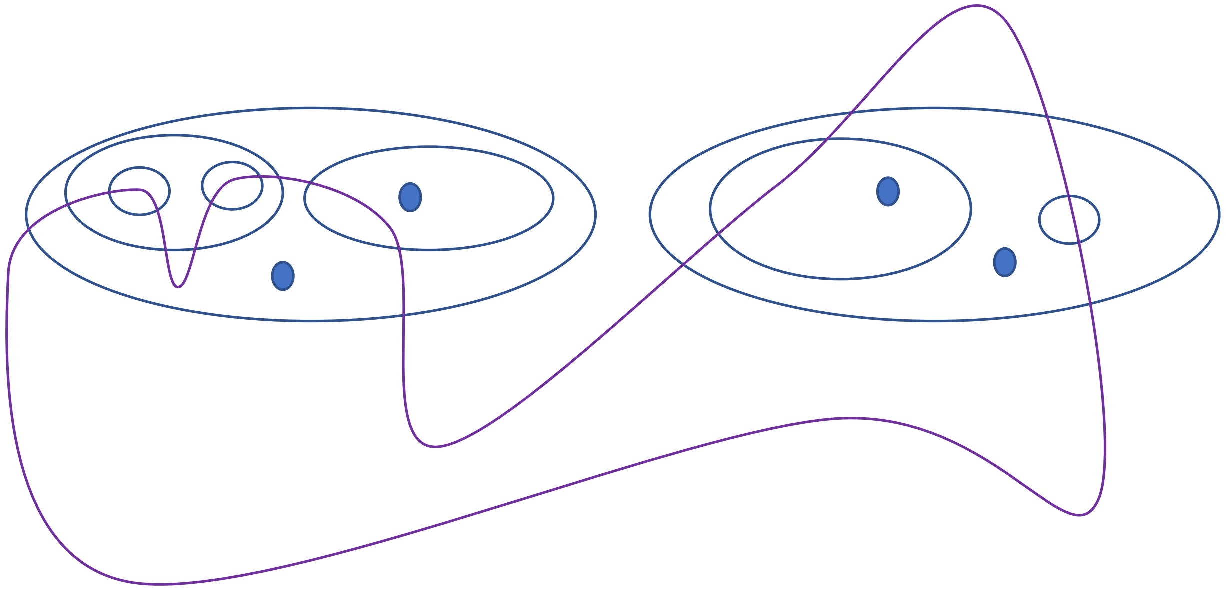
# Solving Local-Connectivity ATSP on vertebrate pairs

# Vertebrate pairs



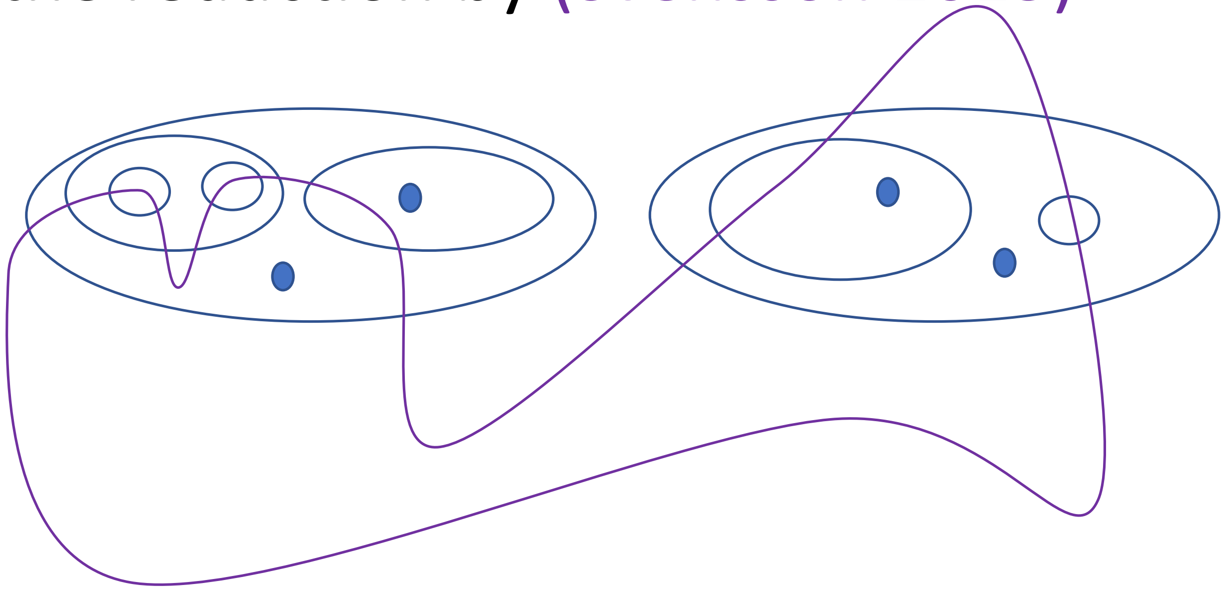
Vertebrate pair $(\mathcal{I}, B)$

- $\mathcal{I} = (G, \mathcal{L}, x, y)$ instance
- $B$: backbone = subtour that crosses every non-singleton set in $\mathcal{L}$



# Vertebrate pairs

- We have reduced general ATSP to solving ATSP for a vertebrate pair $(\mathcal{J}, B)$ with $w(B) = \Theta(OPT)$
- We want to solve Local-Connectivity ATSP on such instances and apply the reduction by (Svensson 2015)

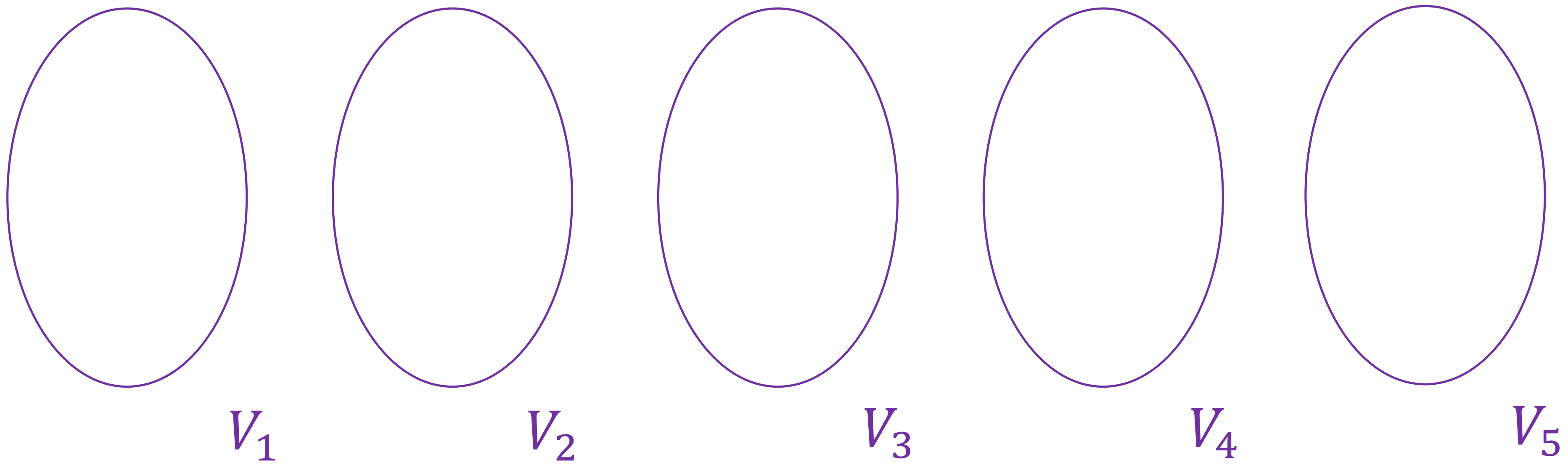


# Local-Connectivity ATSP (Svensson 2015)

Instance $\mathcal{I} = (G, \mathcal{L}, x, y)$ with induced weights $w: E \to \mathbb{R}_+$

Lower bound function $\mathrm{lb}: V \to \mathbb{R}_+$ with $\sum_{v \in V} \mathrm{lb}(v) = OPT$

Input: partition of the vertex set $V = V_1 \cup V_2 \cup \cdots \cup V_k$



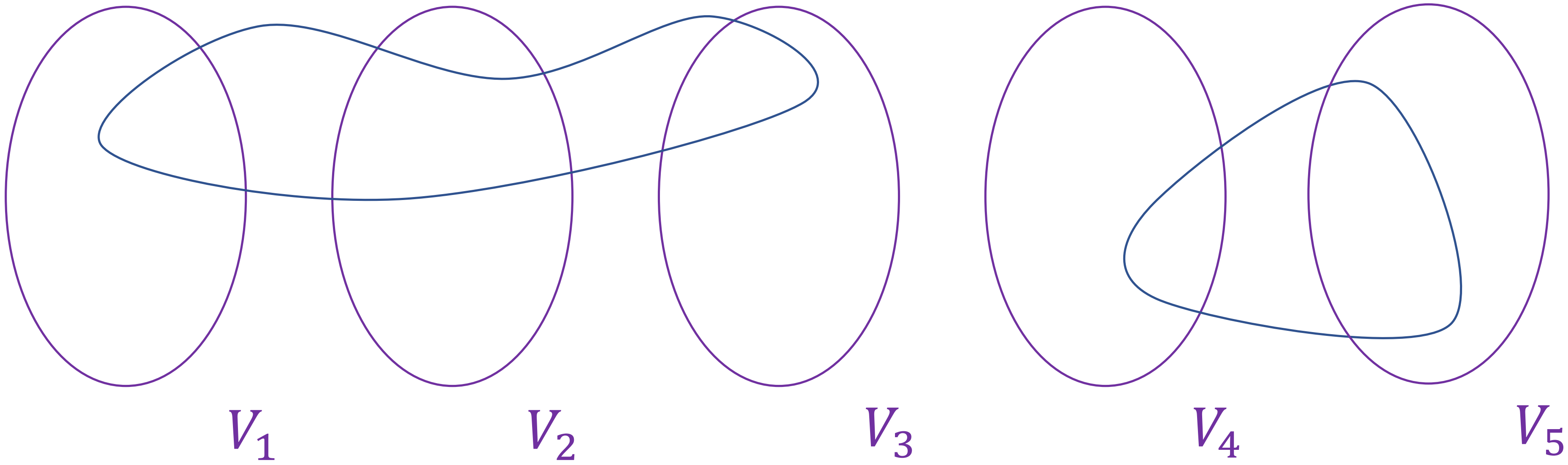
# Local-Connectivity ATSP (Svensson 2015)

Instance $\mathcal{I} = (G, \mathcal{L}, x, y)$ with induced weights $w: E \to \mathbb{R}_+$

Lower bound function $\text{lb}: V \to \mathbb{R}_+$ with $\sum_{v \in V} \text{lb}(v) = OPT$

Input: partition of the vertex set $V = V_1 \cup V_2 \cup \cdots \cup V_k$

Output: subtour $F$ that crosses each $V_i$



# Local-Connectivity ATSP (Svensson 2015)

Instance $\mathcal{I} = (G, \mathcal{L}, x, y)$ with induced weights $w: E \to \mathbb{R}_+$

Lower bound function $\mathrm{lb}: V \to \mathbb{R}_+$ with $\sum_{v \in V} \mathrm{lb}(v) = OPT$

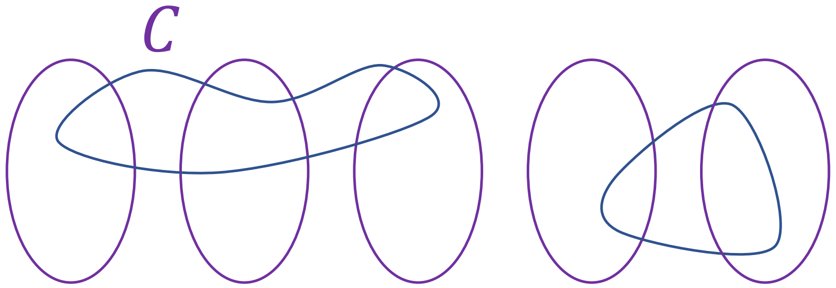
Input: partition of the vertex set $V = V_1 \cup V_2 \cup \cdots \cup V_k$

Output: subtour $F$ that crosses each $V_i$

$\alpha$-light algorithm: for every component $C$ of $F$,

$$w(E(C)) \leq \alpha \,\mathrm{lb}(V(C))$$

*“Every component locally pays for itself”*



# Local-Connectivity ATSP (Svensson 2015)

$\alpha$-light algorithm for Local-Connectivity ATSP

$9\alpha$-approximation for ATSP

(Svensson 2015)
27-approximation for node-weighted ATSP

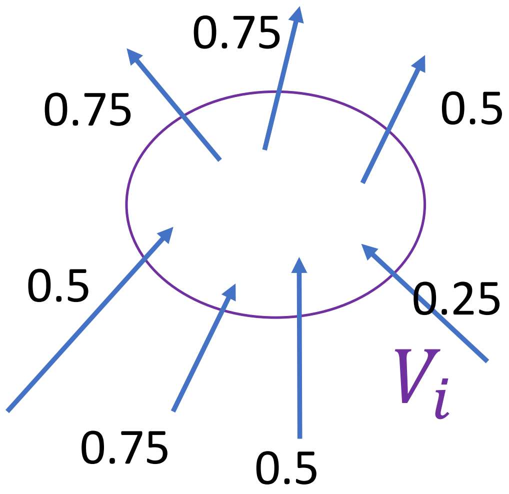
Want:
O(1)-light algorithm for vertebrate instances

# Local-Connectivity ATSP: node-weighted case

- Instance $\mathcal{I} = (G, \mathcal{L}, x, y)$, with $\mathcal{L}$ containing only singletons (ignore $B$)

$$w(u, v) = y_{\{u\}} + y_{\{v\}}$$

- Define $\mathrm{lb}(u) = 2y_{\{u\}} \quad \forall u \in V$
- Partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$
- Modify $G$ and $x$, and solve an integer circulation problem

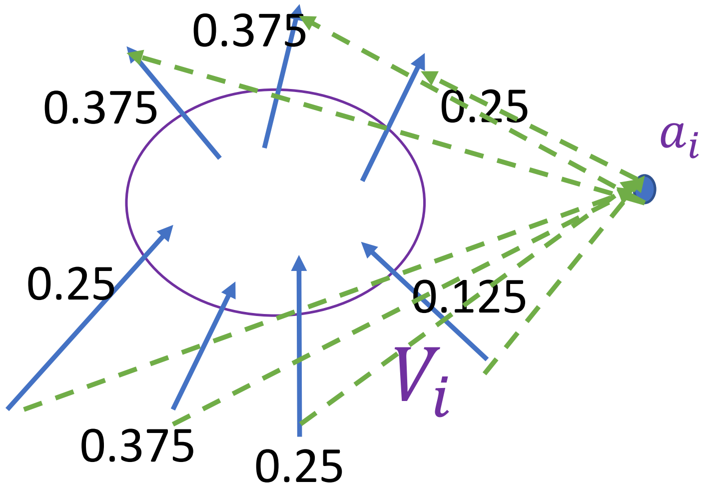


# Local-Connectivity ATSP: node-weighted case

- Instance $\mathcal{I} = (G, \mathcal{L}, x, y)$, with $\mathcal{L}$ containing only singletons (ignore $B$)

$$w(u, v) = y_{\{u\}} + y_{\{v\}}$$

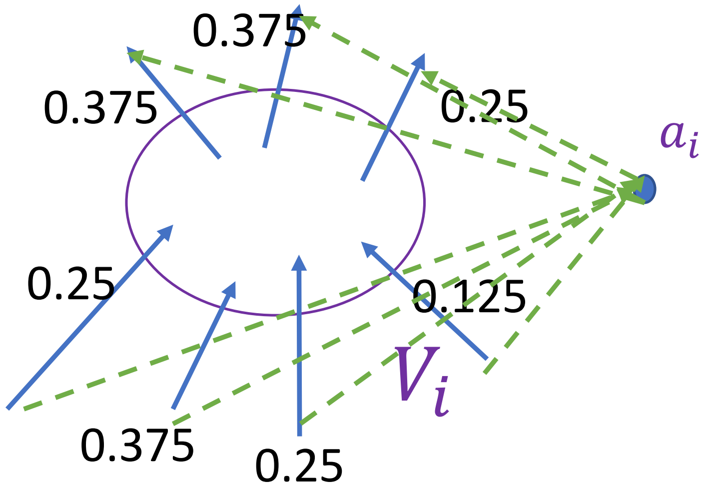
- Define $\mathrm{lb}(u) = 2y_{\{u\}} \quad \forall u \in V$
- Partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$
- Modify $G$ and $x$, and solve an integer circulation problem



- For each $V_i$, create auxiliary vertex $a_i$
- Reroute 1 fractional unit of incoming and outgoing flow $x$ to $a_i$
- Solve integer circulation problem routing =1 unit through each $a_i$ (and ≤1 unit through each $v$ with $y_v > 0$)
- Map back to original $G$

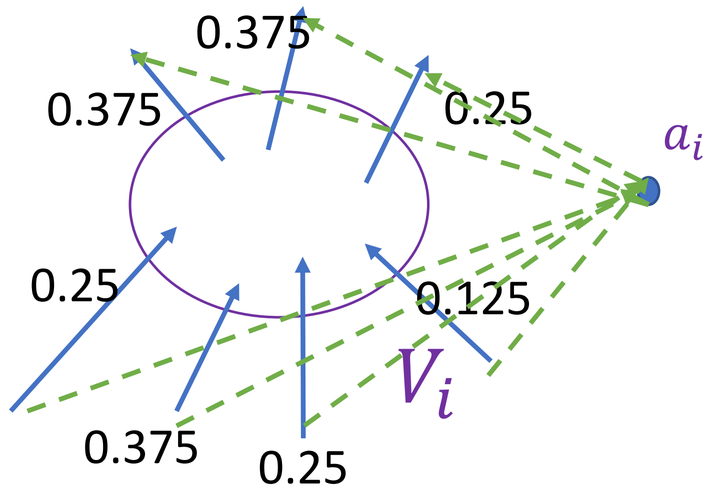
# Local-Connectivity ATSP: node-weighted case

- The rerouted $x$ is feasible for the circulation problem, of weight $OPT$



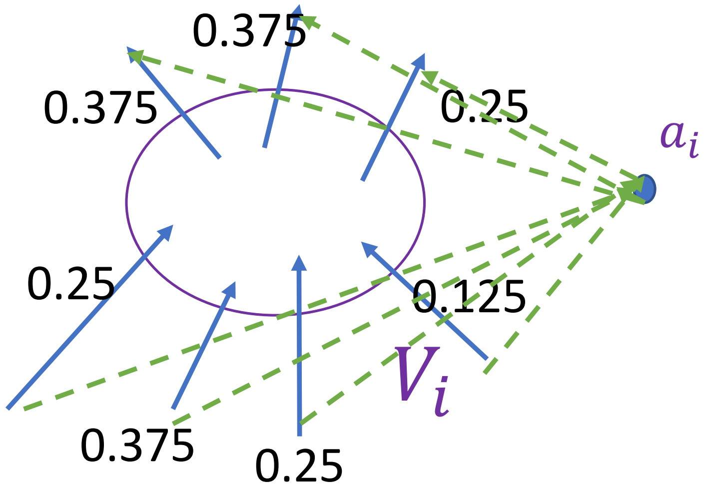
# Local-Connectivity ATSP: node-weighted case

- The rerouted $x$ is feasible for the circulation problem, of weight $OPT$
- **Flow integrality:** there exists also integer solution of weight $\leq OPT$



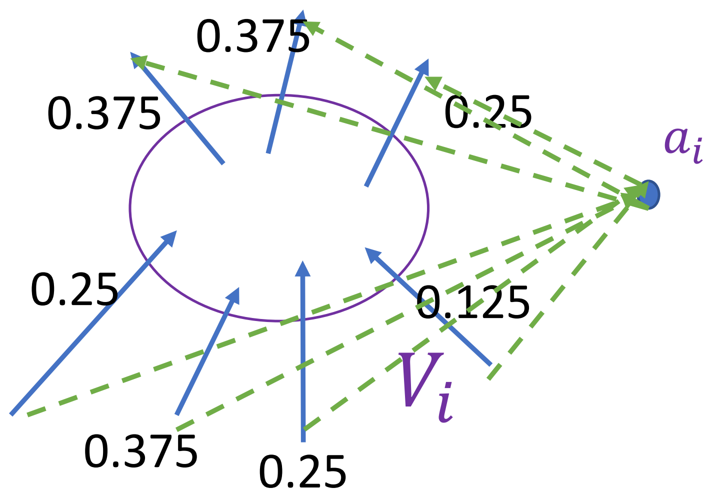
# Local-Connectivity ATSP: node-weighted case

- The rerouted $x$ is feasible for the circulation problem, of weight $OPT$
- Flow integrality: there exists also integer solution of weight $\leq OPT$
- After mapping back, every vertex (with $y_v > 0$) has in-degree $\leq 2$



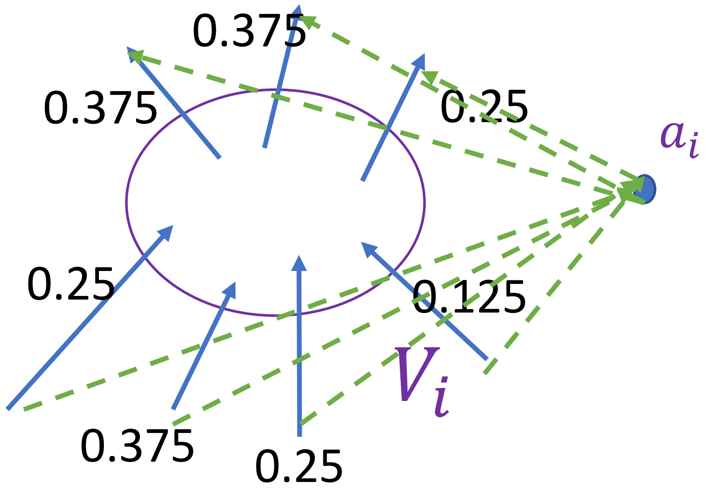
# Local-Connectivity ATSP: node-weighted case

- The rerouted $x$ is feasible for the circulation problem, of weight $OPT$
- **Flow integrality:** there exists also integer solution of weight $\leq OPT$
- After mapping back, every vertex (with $y_v > 0$) has in-degree $\leq 2$
- For a component $C$, $w\big(E(C)\big) = \sum_{(u,v)\in E(C)} y_{\{u\}} + y_{\{v\}} \leq 4 \sum_{v\in C} y_{\{\mathrm{v}\}}$

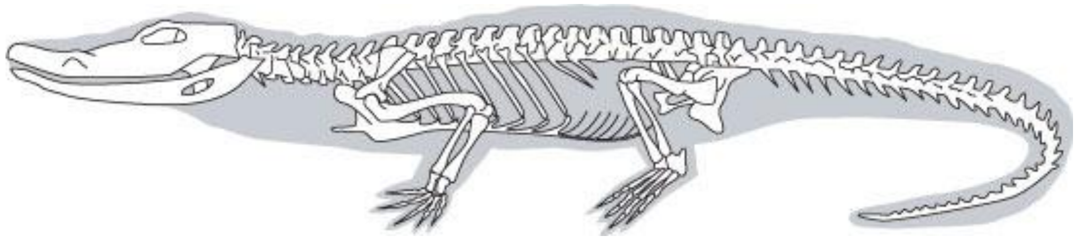


# Local-Connectivity ATSP: node-weighted case

- The rerouted $x$ is feasible for the circulation problem, of weight $OPT$
- **Flow integrality:** there exists also integer solution of weight $\leq OPT$
- After mapping back, every vertex (with $y_v > 0$) has in-degree $\leq 2$
- For a component $C$, $w\big(E(C)\big) = \sum_{(u,v)\in E(C)} y_{\{u\}} + y_{\{v\}} \leq 4 \sum_{v\in C} y_{\{\mathrm{v}\}}$
- $\mathrm{lb}\big(V(C)\big) = 2\sum_{v\in C} y_{\{\mathrm{v}\}}$ $\Longrightarrow$ 2-light algorithm



# Local-Connectivity ATSP: one non-singleton set in $\mathcal{L}$

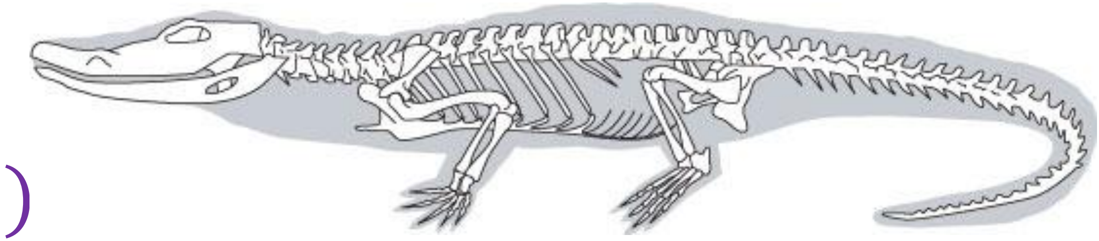


- Vertebrate pair $(\mathcal{I}, B)$.
  Assume $\mathcal{L}$ has a single non-singleton component $S$.
  Thus,

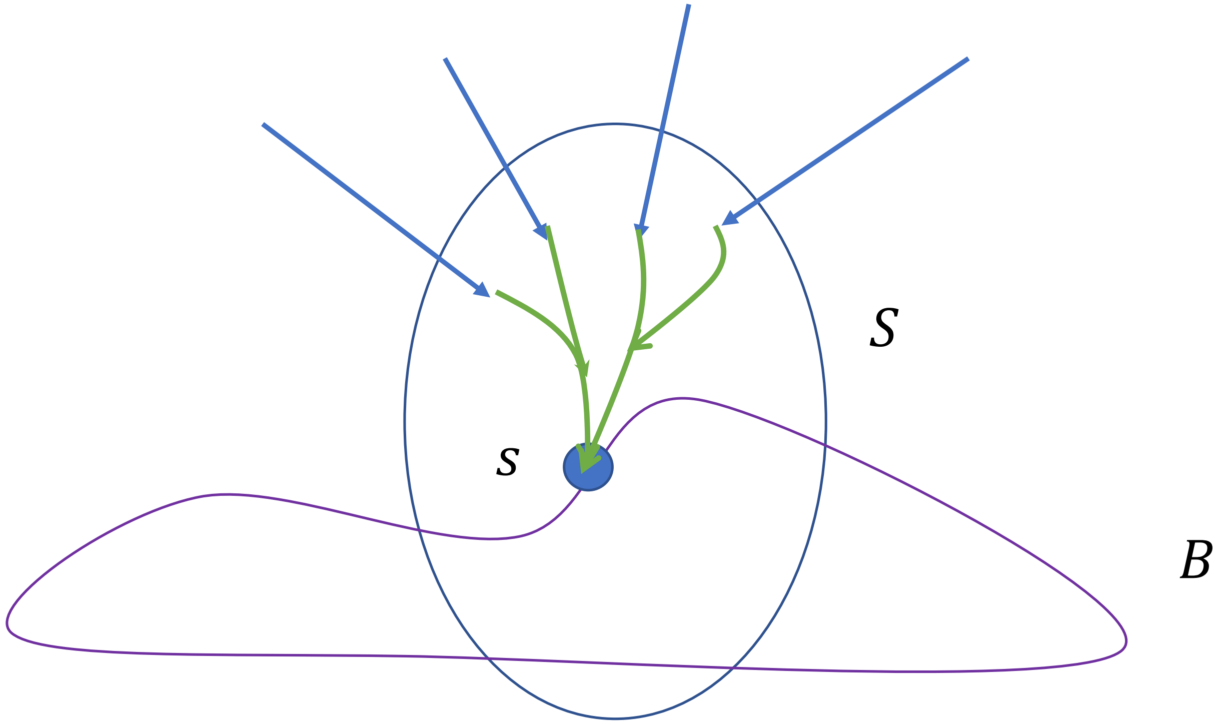
$$w(u,v) = \begin{cases} y_{\{u\}} + y_{\{v\}} + y_S & if\ (u,v) \in \delta(S) \\ y_{\{u\}} + y_{\{v\}} & if\ (u,v) \notin \delta(S) \end{cases}$$

- Define $\mathrm{lb}(u) = 2y_{\{u\}}$ as before,
  but on one backbone vertex $u \in V(B)$ put $\mathrm{lb}(u) = w(B)$ instead
- $\sum_{v \in V} \mathrm{lb}(v) = \Theta(OPT)$, since $w(B) = \Theta(OPT)$

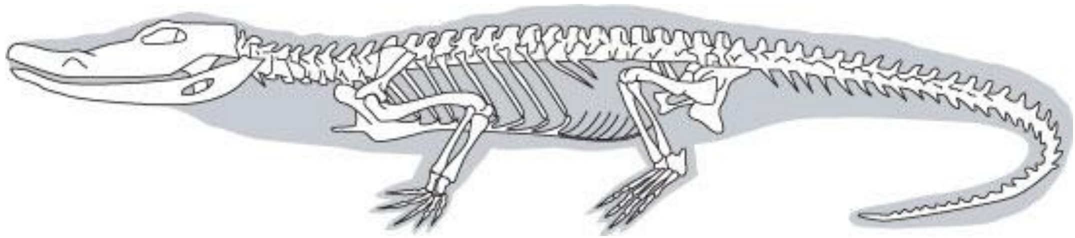
# Local-Connectivity ATSP: one non-singleton set in $\mathcal{L}$



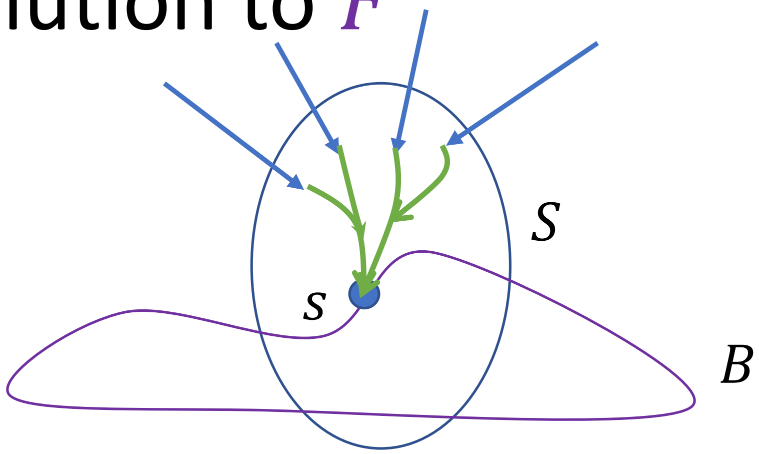
- By assumption, $x(\delta^{in}(S)) = x(\delta^{out}(S))$
- Backbone property: there is a node $s \in V(B) \cap S$
- Flow argument: we can route the incoming 1 unit of flow of $S$ to $s$ (within $x$)



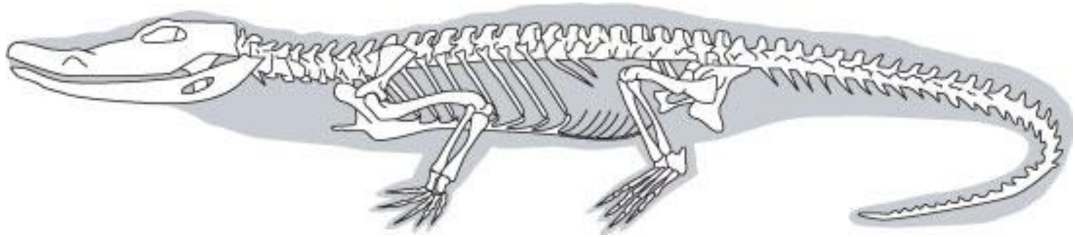
# Local-Connectivity ATSP:
# one non-singleton set in $\mathcal{L}$



- Partition $V = V_1 \cup V_2 \cup \cdots \cup V_k$
- Add backbone $B$ as initial content of the Eulerian output set $F$
- Via flow splitting [Svensson, T., Vegh'16]
  "force" all edges entering $S$ to proceed to $s \in V(B)$
- Create auxiliary vertices $a_i$ as before
- Solve integral circulation problem, and add solution to $F$

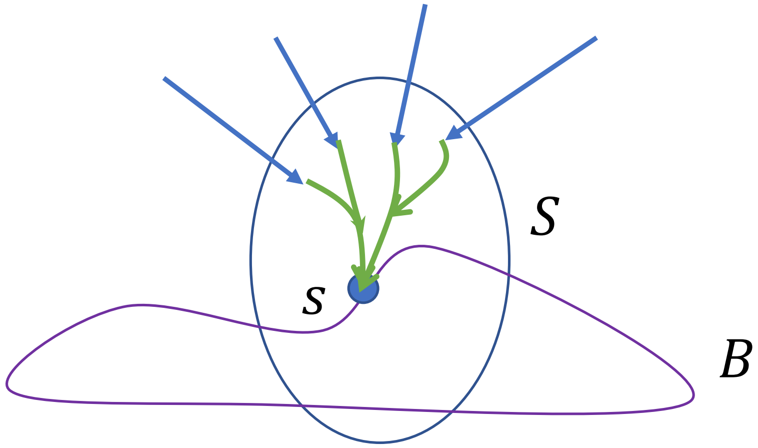


# Local-Connectivity ATSP: one non-singleton set in $\mathcal{L}$



## Analysis

- For all components $C$ not crossing $S$, $w(E(C)) \leq 2\,\mathrm{lb}(V(C))$ exactly as in the node-weighted case
- Giant component $C_0$ containing $B$:
  - Contains all edges in $F$ crossing $S$
  - Has lower bound $\mathrm{lb}(V(C_0)) \geq \mathrm{lb}(u) = \Theta(OPT)$
  - $w(E(C_0)) \leq w(F) \leq O(OPT)$



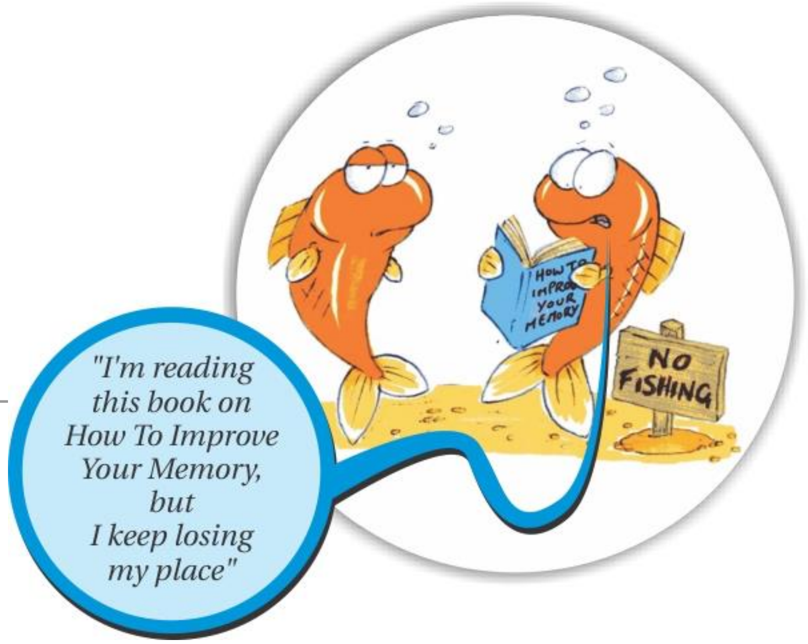
- Therefore solution is $O(1)$-light.
- Same approach extends to arbitrary $\mathcal{L}$: enforce that every subtour crossing a non-singleton set in $\mathcal{L}$ must intersect the backbone.

# Summary and open problems…

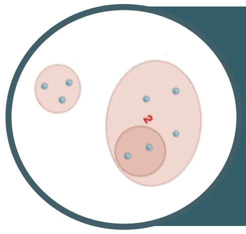
**Theorem:**

A $O(1)$-approximation algorithm with respect to Held-Karp relaxation

# Sequence of reductions



Amazing power of LP-duality



Laminarly-weighted instances

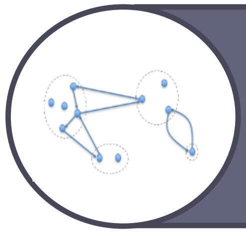
Recursive approach as long as OPT drops

Irreducible instances

Irreducible instances behave like node-weighted

Vertebrate pairs

Complete backbone to tour using circulations and Local-Connectivity ATSP



Solving Local-Connectivity ATSP

# Open questions

- Is the right ratio 2?
    - Unoptimized constant $= 5500$
    - By optimizing our approach, we believe we can get an upper bound in the hundreds. New ideas are needed to get close to lower bound of 2

- Bottleneck ATSP: find tour with minimum max-weight edge

- Thin tree conjecture: Is there a tree $T$ such that for every $S \subset V$

$$|\delta(S) \cap T| \leq O(1)\, x\big(\delta(S)\big)$$

(would also imply apx for Bottleneck ATSP [An, Kleinberg, Shmoys'10])

Thank you!