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

Ola Svensson, Jakub Tarnawski and László A. Végh
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 \( \Delta \)-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 \( (V, E') \)

\[
in\text{-degree} = \text{out\text{-degree}}
\]
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 $(u, v)$

**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 $S \subset V$

$x \geq 0$

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

$S$ = set of cut edges
Integrality gap of the

i.e. how far off is that particular algorithm?
Pick any two...

- integral
- Eulerian
- cycle cover
- ATSP
- spanning tree
- Held-Karp
- 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]

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]

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]

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

2-edge-weights ATSP

[Svensson'15]

2-edge-weights Local-Connectivity ATSP

[Svensson, T., Vegh'16]

general ATSP

[Svensson'15]

structured ATSP

[Svensson'15]

more structured ATSP

[Svensson'15]

really structured Local-Connectivity ATSP

[Svensson, T., Vegh'16]

really structured ATSP
Outline of reductions

- Laminarly-weighted instances
- Irreducible instances
- Vertebrate pairs
- Solving Local-Connectivity ATSP
Laminarly-weighted instances

By amazing power of LP-duality
Asymmetric Traveling Salesman Problem

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

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

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

**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 $S \subset V$
- $x \geq 0$

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

$S$ = set of cut 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 \( S \subset V \)
\( x \geq 0 \)
Minimize: \[ \sum_{u,v \in E} w(u, v) x_{uv} \]

Subject to:

1. \[ x(\delta^+(v)) = x(\delta^-(v)) \] for all \( v \in V \)
2. \[ x(\delta(S)) \geq 2 \] for all \( S \subset V \)
3. \( x \geq 0 \)

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

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: \[ x(\delta^+(v)) = x(\delta^-(v)) \quad \text{for all } v \in V \]
\[ x(\delta(S)) \geq 2 \quad \text{for all } S \subset V \]
\[ x \geq 0 \]

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:
- $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$
- $x(\delta(S)) \geq 2$ for all $S \subseteq V$
- $x \geq 0$

LP-value = 22

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

Subject to:
- $\sum_{S: (u,v) \in \delta(S)} y_S + \alpha_u - \alpha_v \leq w(u, v)$ for all $(u, v) \in E$
- $y \geq 0$

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: $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$
$x(\delta(S)) \geq 2$ for all $S \subset V$
$x \geq 0$

Dual value = LP-value = 22

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 $(u, v) \in E$

$y \geq 0$

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 $S$
Maximize: \( \sum_{S \subseteq V} 2 \cdot y_S \)

Subject to:

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

\( y \geq 0 \)

Dual value = LP-value = 22

Dual has variables:

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

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

\[
4 + 3 - 6 = 1
\]
Maximize: $\sum_{S \subseteq V} 2 \cdot y_S$

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

Dual has variables:

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

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

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

$2 + 1 + 4 - 6 = 1$
Minimize: \( \sum_{uv \in E} w(u, v) x_{uv} \)
Subject to: \( x(\delta^+(v)) = x(\delta^-(v)) \) for all \( v \in V \)
\( x(\delta(S)) \geq 2 \) for all \( S \subset V \)
\( x \geq 0 \)

Dual value = LP-value = 22

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 \( (u, v) \in E \)
\( y \geq 0 \)

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

Subject to:
- \( x(\delta^+(v)) = x(\delta^-(v)) \) for all \( v \in V \)
- \( x(\delta(S)) \geq 2 \) for all \( S \subset V \)
- \( x \geq 0 \)

Dual value = LP-value = 22

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 \( (u,v) \in E \)
- \( y \geq 0 \)

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 \( F \)
\[
w(F) = w'(F)
\]

\[
w'(F) = w(A, B) + \alpha_A - \alpha_B + w(B, C) + \alpha_B - \alpha_C + w(C, A) + \alpha_C - \alpha_A = w(F)
\]
Minimize: $\sum_{u,v \in E} w(u,v) x_{uv}$
Subject to:
1. $x(\delta^+(v)) = x(\delta^-(v))$ for all $v \in V$
2. $x(\delta(S)) \geq 2$ for all $S \subset V$
3. $x \geq 0$

Maximize: $\sum_{S \subset V} 2 \cdot y_S$
Subject to:
1. $\sum_{S: (u,v) \in \delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v)$ for all $(u,v) \in E$
2. $y \geq 0$

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

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:

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

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

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

$y \geq 0$

What happened?

Something complicated with no structure

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

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

Optimal primal and dual $x$ and $(y,0)$

Want more structure!
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:
\[ w(e) = \sum_{S:(u,v) \in \delta(S)} y_S \]
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: \[ x(\delta^+(v)) = x(\delta^-(v)) \quad \text{for all } v \in V \]
\[ x(\delta(S)) \geq 2 \quad \text{for all } S \subset V \]
\[ x \geq 0 \]

Maximize: \[ \sum_{S \subset V} 2 \cdot y_S \]
Subject to:
\[ \sum_{S:(u,v) \in \delta(S)} y_S + \alpha_u - \alpha_v \leq w(u, v) \quad \text{for all } (u, v) \in E \]
\[ y \geq 0 \]

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

Again by complementarity slackness
\[ x(\delta(S)) = 2 \quad \text{for every } S \in \mathcal{L} \]

So every \( S \in \mathcal{L} \) is a tight set!
Minimize: \( \sum_{uv \in E} w(u, v) x_{uv} \)
Subject to: 
\[
x(\delta^+(v)) = x(\delta^-(v)) \quad \text{for all } v \in V
\]
\[
x(\delta(S)) \geq 2 \quad \text{for all } S \subseteq V
\]
\[x \geq 0\]

A lot of structure:
\[
w(e) = \sum_{S: (u,v) \in \delta(S)} y_S
\]

Maximize: \( \sum_{S \subseteq V} 2 \cdot y_S \)
Subject to:
\[
\sum_{S: (u,v) \in \delta(S)} y_S + \alpha_u - \alpha_v \leq w(u,v) \quad \text{for all } (u, v) \in E
\]
\[y \geq 0\]

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

Again by complementarity slackness
\[
x(\delta(S)) = 2 \quad \text{for every } S \in \mathcal{L}
\]

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

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

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

- $x, y$ 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 = $\text{OPT} = 2 \cdot \sum_{S \in \mathcal{L}} y_S$ ($=28$ in example)

**Theorem:**

A $\rho$-approximation algorithm for laminarly-weighted instances yields a $\rho$-approximation algorithm for general ATSP
Reduced our task to:

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

- $x, y$ 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 $y$:
  \[
  w(e) = \sum_{S \in \mathcal{L} : e \in \delta(S)} y_S
  \]
Basic idea: recursively solving smaller instances is not dangerous if optimum drops.
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 \( \leq \text{OPT} \)
Repeated cycle cover

[Frieze, Galbiati, and Maffioli’82]

Find min-cost cycle cover
“Contract“
Repeat until graph is connected

Cost of cycle cover $\leq OPT$
Repeated cycle cover

[Frieze, Galbiati, and Maffioli’82]

Find min-cost cycle cover

“Contract“

Repeat until graph is connected

Cost of cycle cover ≤OPT

Cost of cycle cover ≤OPT
Repeated cycle cover

[Frieze, Galbiati, and Maffioli’82]

Find min-cost cycle cover
“Contract“
Repeat until graph is connected

Cost of cycle cover ≤ \( OPT \)
Repeated cycle cover

[Frieze, Galbiati, and Maffioli’82]

Find min-cost cycle cover
“Contract“
Repeat until graph is connected

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

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

Cost of cycle cover $\leq \text{OPT}$
Repeated cycle cover
[Frieze, Galbiati, and Maffioli’82]

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

Cost of cycle cover \( \leq OPT \)
Cost of cycle cover \( \leq OPT \)
Cost of cycle cover \( \leq OPT \)

Total cost \( \leq 3 \cdot OPT \)

\( \log_2 n \)-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

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

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

We pursue it using the structure of laminarily-weighted instances
Le retour
Laminarly-weighted

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

- $x, y$ 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 $y$:
  $$w(e) = \sum_{S \in \mathcal{L} : e \in \delta(S)} y_S$$ for every edge $e$

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

What to do? Simply add a shortest path

Set so that we always pay for the rewiring in lift

Lift tour in contracted instance to subtour in original instance
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 y-value of new set to pay for maximum cost over all possible ways to enter and exit the original set

In example:

\[ ? = 5+2+2+1+4+3 = 17 \]  \( (\text{path crosses every tight set}) \)

**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:

\[ 2 \times (5 + 2 + 2 + 4 + 3) - 2 \times 17 + 2 \times (5 + 1 + 4) - 2 \times 17 \leq 0 \]

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 \(\Rightarrow\) **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 $\rho$-approximation algorithm for irreducible instances yields a $8\rho$-approximation algorithm for laminarily-weighted instances, and thus for general ATSP.

Let $\mathcal{A}$ be a $\rho$-approximation algorithm for irreducible instances…
If instance is irreducible, simply run $\mathcal{A}$
Otherwise select \textit{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)
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 $\mathcal{A}$

Alg for reducible instances

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

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

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

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

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

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

**Simplifying assumption:**
instance obtained by restricting to vertices inside $S$ is feasible
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 $2\rho \left( 2 \cdot \sum_{R \in \mathcal{L}: R \subset S} y_R \right)$

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

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
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
Algorithms 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 $p$-approximation algorithm for irreducible instances yields a $8p$-approximation algorithm for laminarly-weighted instances, and thus for general ATSP.
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!
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

Alg for perfect irreducible

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
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

Alg for perfect irreducible

Node-weighted instance
Use 28-approximation by Ola
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

**Alg for perfect irreducible**

**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
Algorithm 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 \((J, B)\)
- \(J = (G, \mathcal{L}, x, y)\) 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 $(J, B)$ 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 (Svenssson 2015)

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

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

Input: partition of the vertex set $V = V_1 \cup V_2 \cup \cdots \cup V_k$
Local-Connectivity ATSP (Svensson 2015)

Instance $I = (G, \mathcal{L}, x, y)$ with induced weights $w: E \rightarrow \mathbb{R}_+$
Lower bound function $lb: V \rightarrow \mathbb{R}_+$ with $\sum_{v \in V} lb(v) = OPT$
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 $I = (G, L, x, y)$ with induced weights $w : E \rightarrow \mathbb{R}_+$
Lower bound function $lb : V \rightarrow \mathbb{R}_+$ with $\sum_{v \in V} lb(v) = OPT$
Input: partition of the vertex set $V = V_1 \cup V_2 \cup \ldots \cup V_k$
Output: subtour $F$ that crosses each $V_i$

$\alpha$-light algorithm: for every component $C$ of $F$, $w(E(C)) \leq \alpha \cdot lb(V(C))$

“Every component locally pays for itself”
Local-Connectivity ATSP (Svensson 2015)

\begin{itemize}
  \item $\alpha$-light algorithm for Local-Connectivity ATSP
  \item 9$\alpha$-approximation for ATSP
  \item (Svensson 2015) 27-approximation for node-weighted ATSP
  \item Want: O(1)-light algorithm for vertebrate instances
\end{itemize}
Local-Connectivity ATSP: node-weighted case

- Instance $I = (G, \mathcal{L}, x, y)$, with $\mathcal{L}$ containing only singletons (ignore $B$)
  \[ w(u, v) = y_{\{u\}} + y_{\{v\}} \]
- Define $lb(u) = 2y_{\{u\}} \quad \forall u \in V$
- Partition $V = V_1 \cup V_2 \cup \ldots \cup V_k$
- Modify $G$ and $x$, and solve an integer circulation problem
Local-Connectivity ATSP: node-weighted case

• Instance $I = (G, \mathcal{L}, x, y)$, with $\mathcal{L}$ containing only singletons (ignore $B$)
  
  \[ w(u, v) = y\{u\} + y\{v\} \]

• Define $lb(u) = 2y\{u\} \quad \forall u \in V$

• Partition $V = V_1 \cup V_2 \cup \ldots \cup V_k$

• 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 $\leq 1$ 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 $C$, $w(E(C)) = \sum_{(u,v) \in E(C)} y\{u\} + y\{v\} \leq 4 \sum_{v \in C} y\{v\}$
Local-Connectivity ATSP: node-weighted case

- The rerouted $x$ is feasible for the circulation problem, of weight $\text{OPT}$
- Flow integrality: there exists also integer solution of weight $\leq \text{OPT}$
- After mapping back, every vertex (with $y_v > 0$) has in-degree $\leq 2$
- For a component $C$, $w(E(C)) = \sum_{(u,v) \in E(C)} y_u + y_v \leq 4 \sum_{v \in C} y_v$
- $\text{lb}(V(C)) = 2 \sum_{v \in C} y_v \implies$ 2-light algorithm
Local-Connectivity ATSP: one non-singleton set in \( L \)

- Vertebrate pair \((J, B)\). Assume \( L \) has a single non-singleton component \( S \). Thus,

\[
w(u, v) = \begin{cases} 
y\{u\} + y\{v\} + y_s & \text{if } (u, v) \in \delta(S) 
y\{u\} + y\{v\} & \text{if } (u, v) \notin \delta(S) \end{cases}
\]

- Define \( lb(u) = 2y\{u\} \) as before, but on one backbone vertex \( u \in V(B) \) put \( lb(u) = w(B) \) instead

\[
\sum_{v \in V} lb(v) = \Theta(OPT), \text{ since } w(B) = \Theta(OPT)
\]
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 $S$, $w(E(C)) \leq 2 \text{lb}(V(C))$ exactly as in the node-weighted case

• Giant component $C_0$ containing $B$:
  • Contains all edges in $F$ crossing $S$
  • Has lower bound $\text{lb}(V(C_0)) \geq \text{lb}(u) = \Theta(\text{OPT})$
  • $w(E(C_0)) \leq w(F) \leq O(\text{OPT})$

• 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

1. Amazing power of LP-duality
   - Laminarly-weighted instances

2. Recursive approach as long as OPT drops
   - Irreducible instances

3. Irreducible instances behave like node-weighted
   - Vertebrate pairs

4. Complete backbone to tour using circulations and
   - Solving Local-Connectivity ATSP

---

"I'm reading this book on How To Improve Your Memory, but I keep losing my place"
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 $S \subseteq V$
  $$|\delta(S) \cap T| \leq O(1) \cdot x(\delta(S))$$
  (would also imply apx for Bottleneck ATSP [An, Kleinberg, Shmoys’10])

Thank you!