Shorter Tours by Nicer Ears

7/5-approximation for graphic TSP, 3/2 for the path version, and 4/3 for two-edge-connected subgraphs

Jens Vygen

(joint work with András Sebő)

September 21, 2012

Metric TSP

Given a complete graph G and metric weights $c : E(G) \to \mathbb{R}_{\geq 0}$, find a Hamiltonian circuit in G with minimum total weight.

- *NP*-hard
- best known approximation ratio $\frac{3}{2}$ (Christofides [1976])
- no $\frac{185}{184}$-approximation algorithm exists unless $P = NP$ (Lampis [2012])
- integrality ratio of subtour relaxation between $\frac{4}{3}$ and $\frac{3}{2}$ (Wolsey [1980])

Metric TSP

Given a complete graph G and metric weights $c : E(G) \to \mathbb{R}_{\geq 0}$, find a Hamiltonian circuit in G with minimum total weight.

- *NP*-hard
- best known approximation ratio $\frac{3}{2}$ (Christofides [1976])
- no $\frac{185}{184}$-approximation algorithm exists unless $P = NP$ (Lampis [2012])
- integrality ratio of subtour relaxation between $\frac{4}{3}$ and $\frac{3}{2}$ (Wolsey [1980])

But recently there has been progress for a special case called Graphic TSP:

- approximation ratio $1.5 - \epsilon$ (Gharan, Saberi, Singh [2011])
- approximation ratio 1.461 (Mömke, Svensson [2011])
- approximation ratio 1.445 (Mucha [2012])

Metric TSP

Given a complete graph G and metric weights $c : E(G) \to \mathbb{R}_{\geq 0}$, find a Hamiltonian circuit in G with minimum total weight.

- *NP*-hard
- best known approximation ratio $\frac{3}{2}$ (Christofides [1976])
- no $\frac{185}{184}$-approximation algorithm exists unless $P = NP$ (Lampis [2012])
- integrality ratio of subtour relaxation between $\frac{4}{3}$ and $\frac{3}{2}$ (Wolsey [1980])

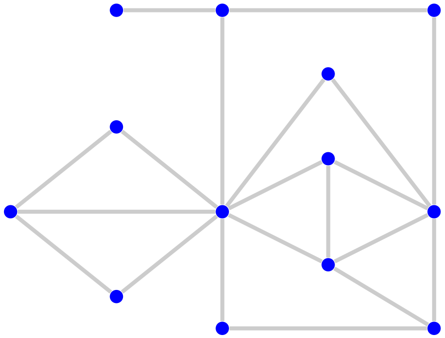
But recently there has been progress for a special case called Graphic TSP:

- approximation ratio $1.5 - \epsilon$ (Gharan, Saberi, Singh [2011])
- approximation ratio 1.461 (Mömke, Svensson [2011])
- approximation ratio 1.445 (Mucha [2012])

We will show an approximation ratio of 1.4.

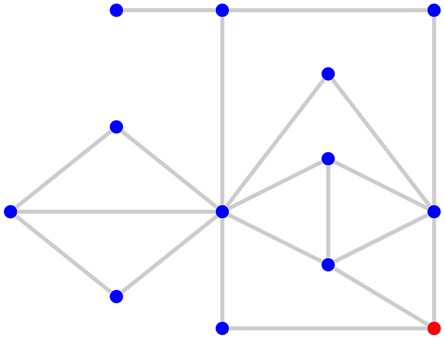
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



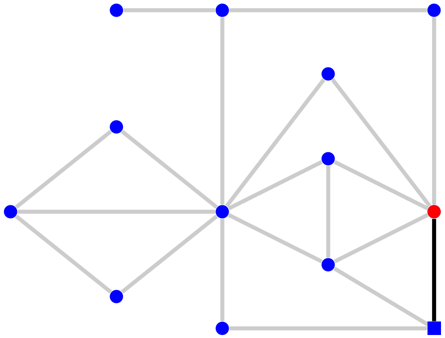
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



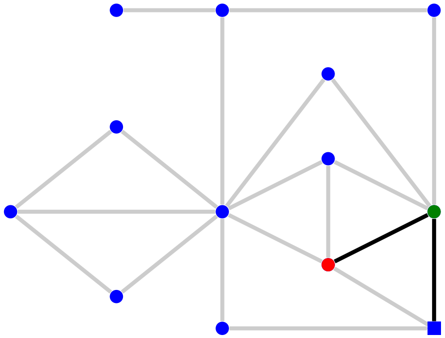
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



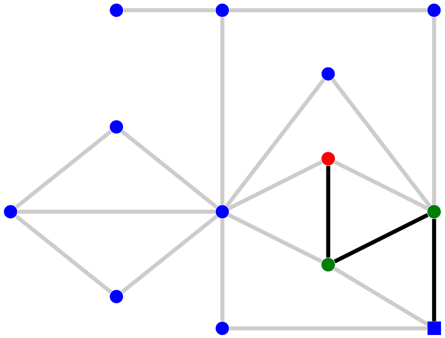
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



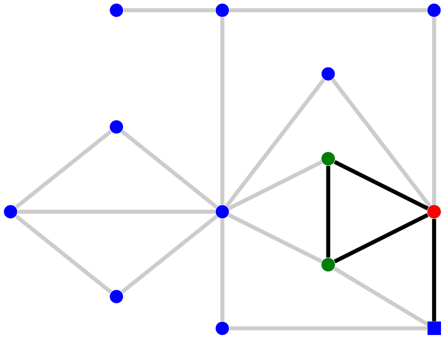
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



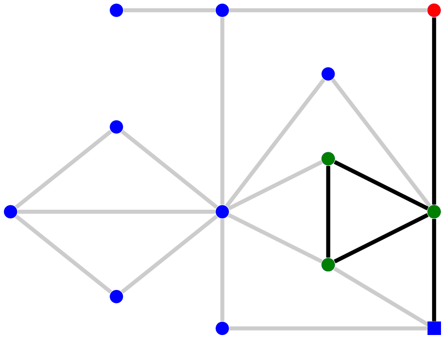
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



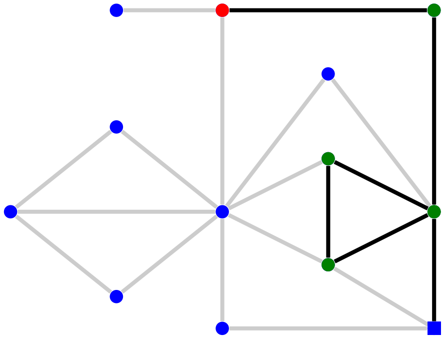
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



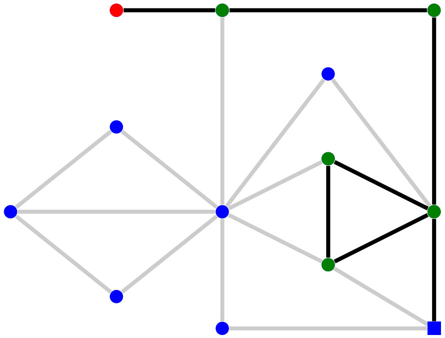
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



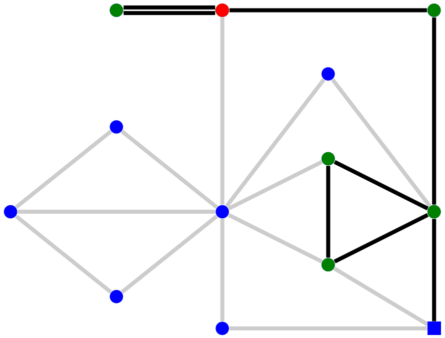
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



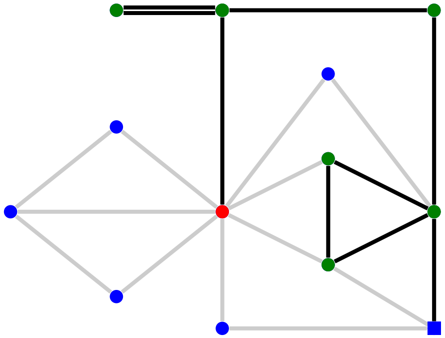
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



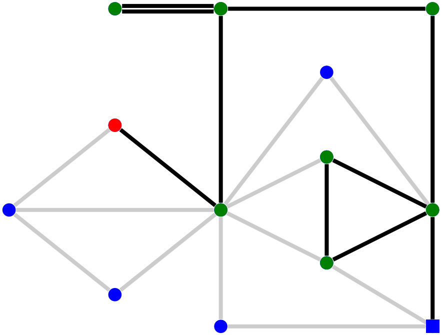
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



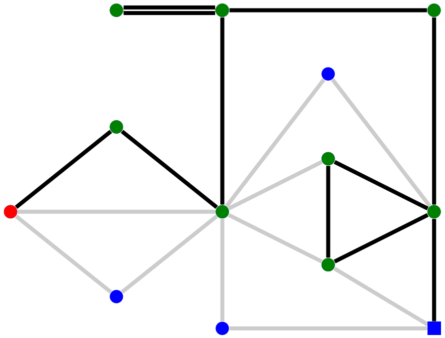
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



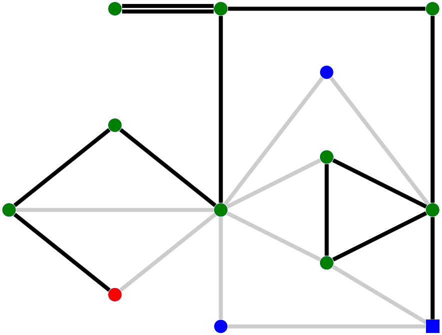
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



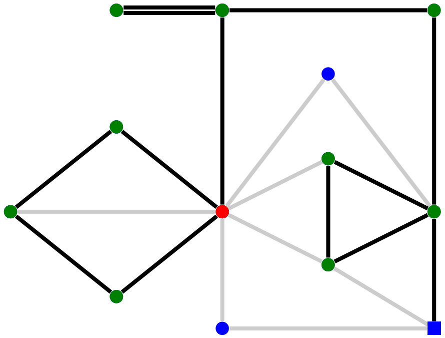
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



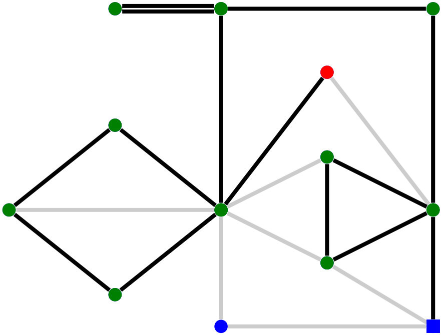
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



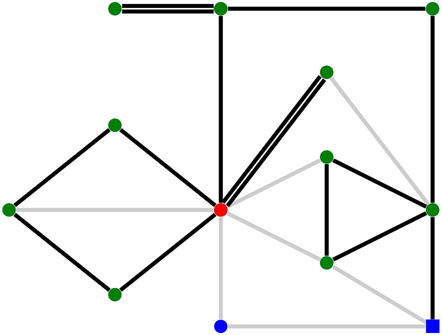
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



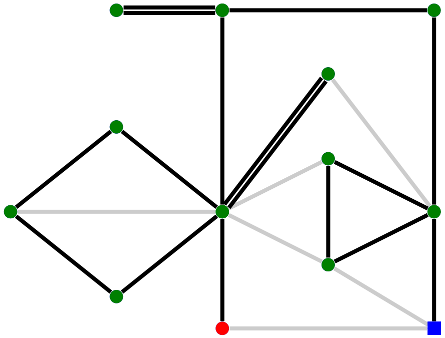
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



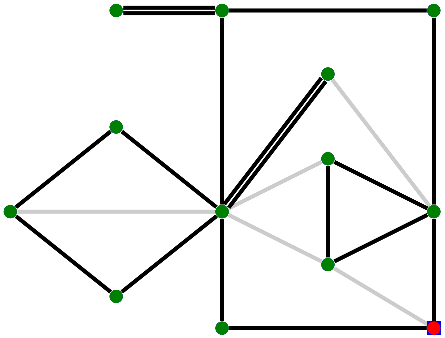
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



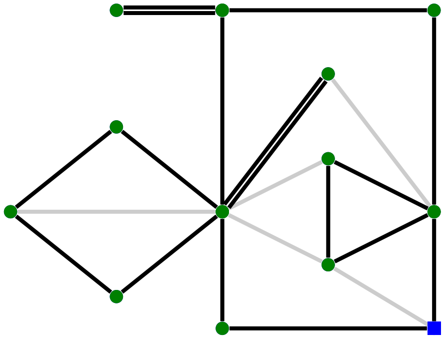
Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



Graphic TSP

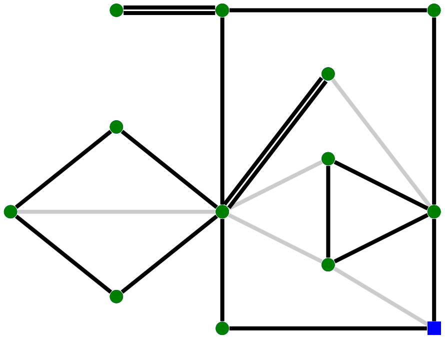
Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



Equivalently:

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

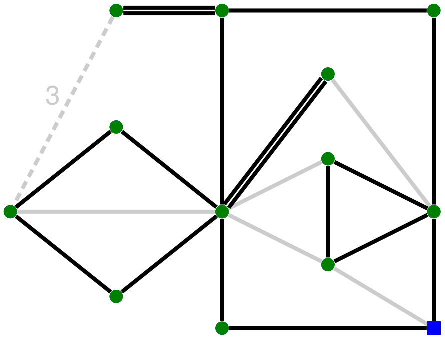


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

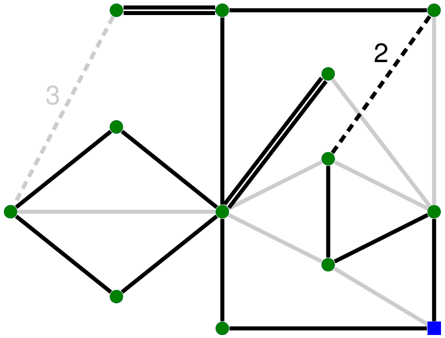


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

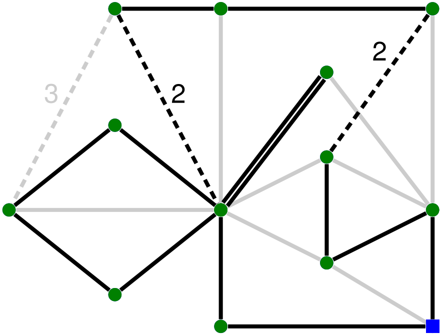


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

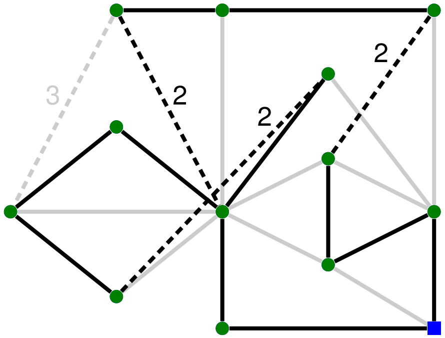


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

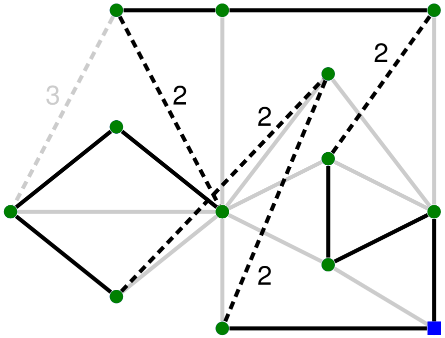


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

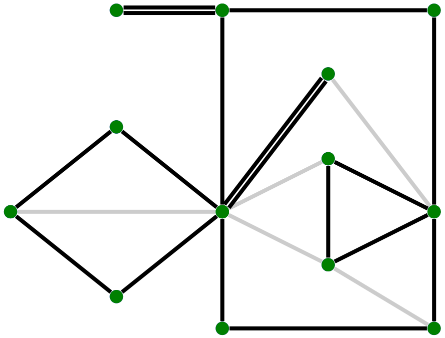


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.

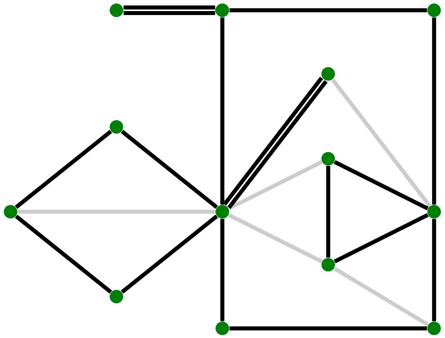


Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G
- find a smallest Eulerian spanning subgraph of $2G$

Graphic TSP

Given a connected graph G, find a minimum length closed edge progression in G that visits every vertex at least once.



Equivalently:

- find a shortest Hamiltonian circuit in the metric closure of G
- find a smallest $\underbrace{\text{Eulerian spanning subgraph of } 2G}_{\text{its edge set is called tour or connected-}\emptyset\text{-join of } G}$

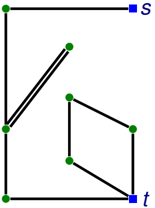
Main results

Let G be a connected graph and $T \subseteq V(G)$ with $|T|$ even.

A connected-T-join of G (aka T-tour) is a set $F \subseteq E(2G)$ such that

- $(V(G), F)$ is connected, and
- $v \in T \Leftrightarrow |F \cap \delta(v)|$ odd.

$T = \{s, t\}$



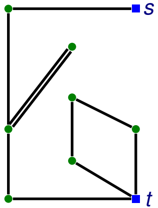
Main results

Let G be a connected graph and $T \subseteq V(G)$ with $|T|$ even.

A connected-T-join of G (aka T-tour) is a set $F \subseteq E(2G)$ such that

- $(V(G), F)$ is connected, and
- $v \in T \Leftrightarrow |F \cap \delta(v)|$ odd.

$T = \{s, t\}$



We improve the approximation ratio for:

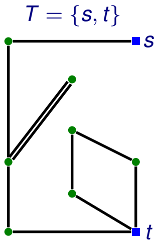
- Graphic TSP (smallest tour = connected-$\emptyset$-join): from $\frac{13}{9}$ (Mucha [2012]) to $\frac{7}{5}$
- Connected-T-join (smallest connected-T-join): from $\frac{5}{3}$ (Christofides [1976], Hoogeveen [1991]), 1.578 for $T = \{s, t\}$ (An, Kleinberg, Shmoys [2012]), to $\frac{3}{2}$
- 2ECSS (smallest 2-edge-connected spanning subgraph): from $\frac{17}{12}$ (Cheriyan, Sebő, Szigeti [2001]) to $\frac{4}{3}$

Main results

Let G be a connected graph and $T \subseteq V(G)$ with $|T|$ even.

A connected-T-join of G (aka T-tour) is a set $F \subseteq E(2G)$ such that

- $(V(G), F)$ is connected, and
- $v \in T \Leftrightarrow |F \cap \delta(v)|$ odd.

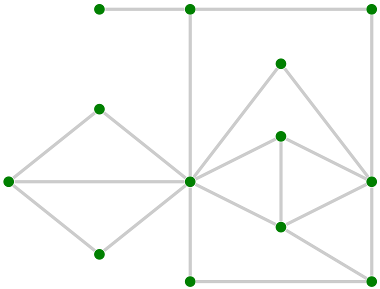


We improve the approximation ratio for:

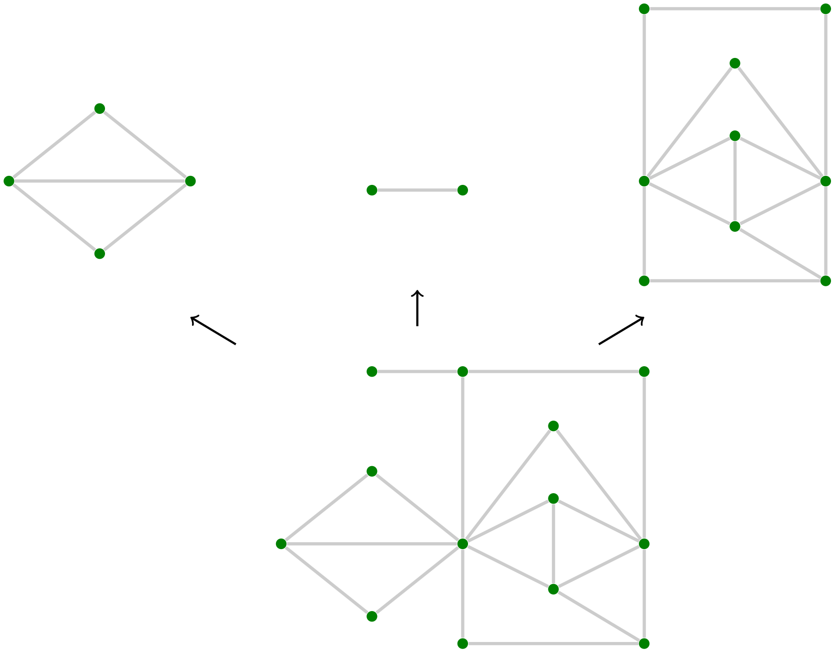
- Graphic TSP (smallest tour = connected-$\emptyset$-join): from $\frac{13}{9}$ (Mucha [2012]) to $\frac{7}{5}$
- Connected-T-join (smallest connected-T-join): from $\frac{5}{3}$ (Christofides [1976], Hoogeveen [1991]), 1.578 for $T = \{s, t\}$ (An, Kleinberg, Shmoys [2012]), to $\frac{3}{2}$
- 2ECSS (smallest 2-edge-connected spanning subgraph): from $\frac{17}{12}$ (Cheriyan, Sebő, Szigeti [2001]) to $\frac{4}{3}$

Note that doubling edges is necessary (except for 2ECSS) and tripling edges does not help.

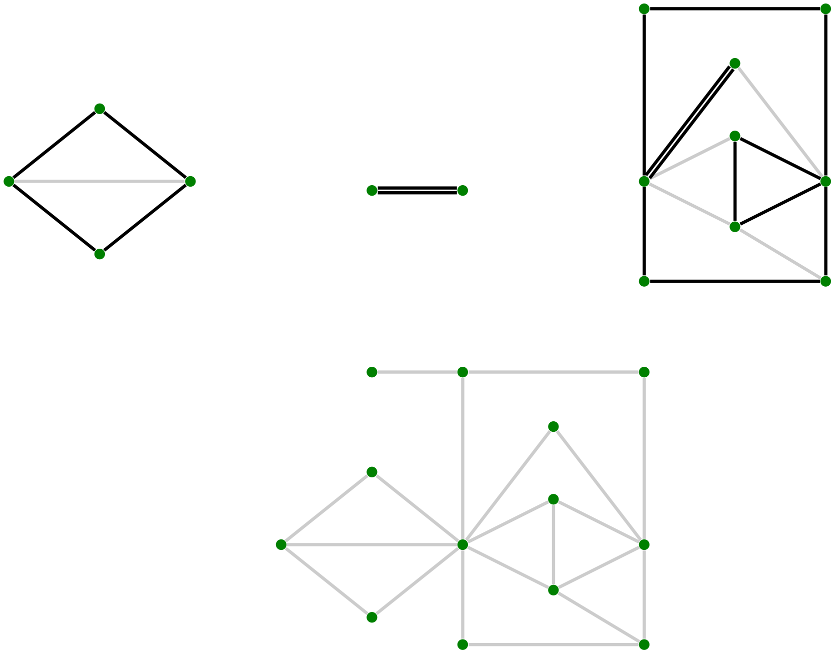
Consider blocks separately



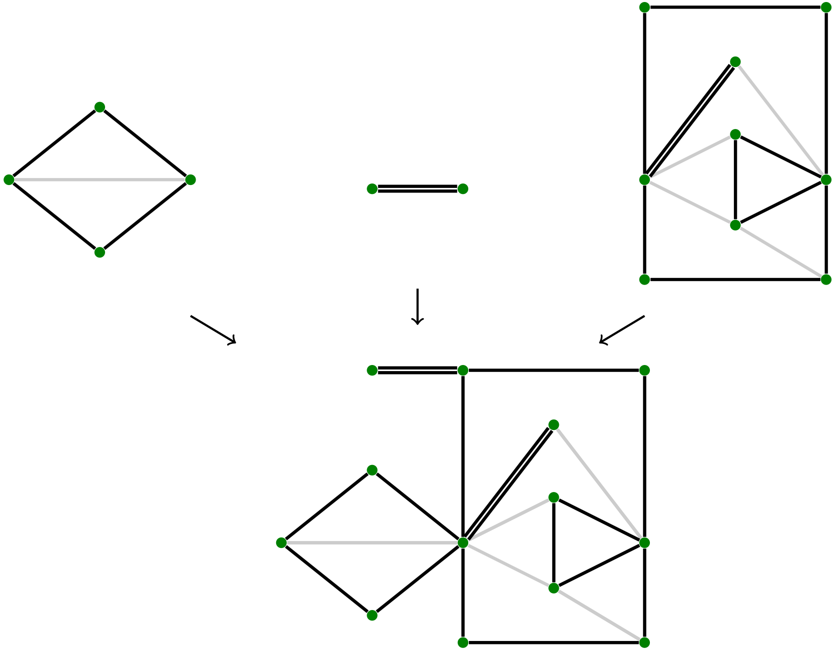
Consider blocks separately



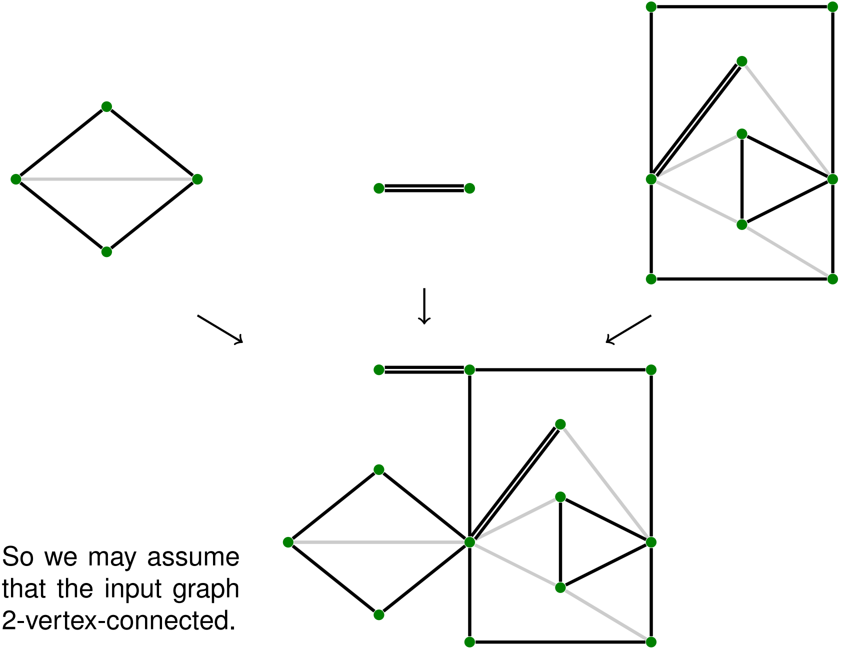
Consider blocks separately



Consider blocks separately



Consider blocks separately

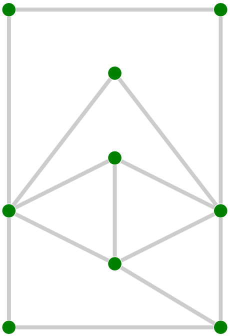


So we may assume that the input graph 2-vertex-connected.

Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

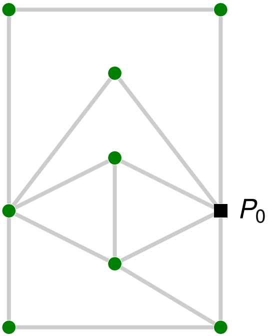
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

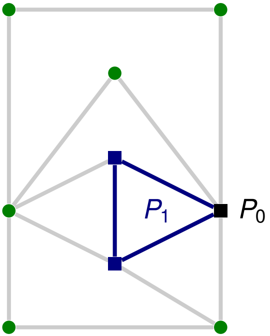
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

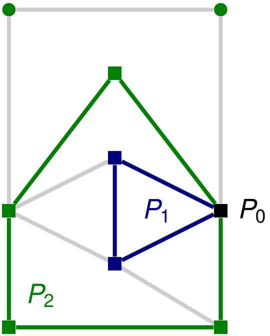
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

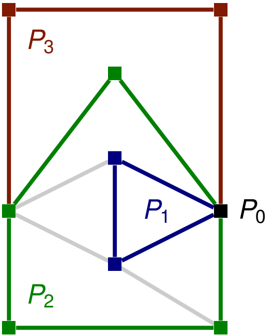
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

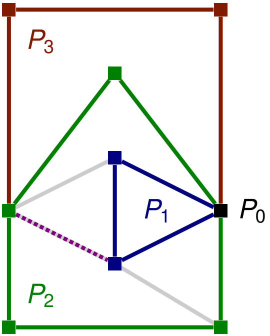
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

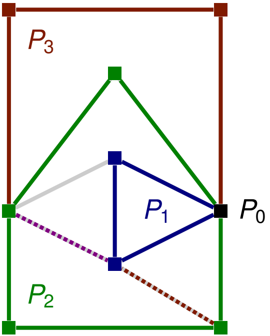
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

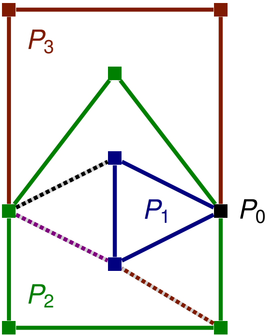
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

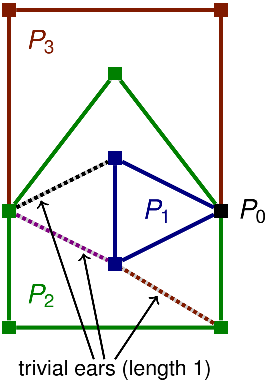
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

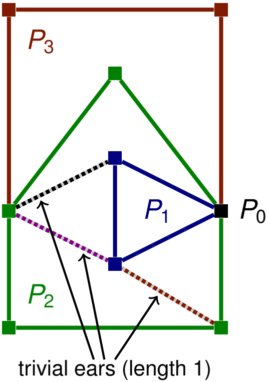
- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.

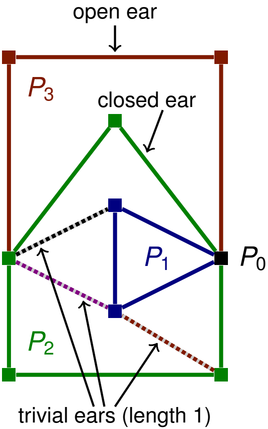


- A graph is 2-edge-connected iff it has an ear-decomposition.

Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.

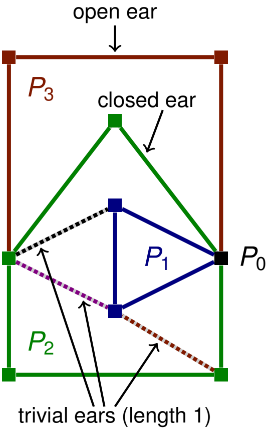


- A graph is 2-edge-connected iff it has an ear-decomposition.

Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.

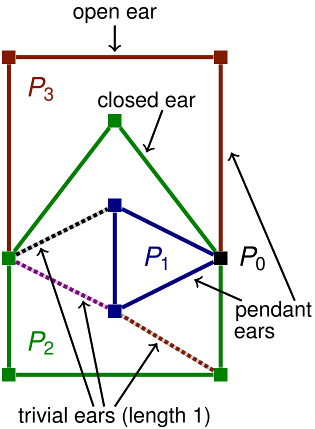


- A graph is 2-edge-connected iff it has an ear-decomposition.
- A graph is 2-vertex-connected iff it has an open ear-decomposition. ($P_2, \ldots, P_k$ are all open ears = paths.)

Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.

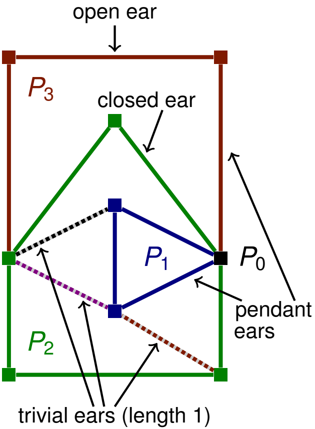


- A graph is 2-edge-connected iff it has an ear-decomposition.
- A graph is 2-vertex-connected iff it has an open ear-decomposition.
- A nontrivial ear is called pendant if none of its internal vertices is endpoint of another nontrivial ear.

Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.

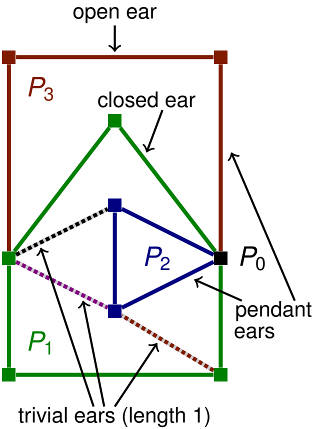


- A graph is 2-edge-connected iff it has an ear-decomposition.
- A graph is 2-vertex-connected iff it has an open ear-decomposition.
- A nontrivial ear is called pendant if none of its internal vertices is endpoint of another nontrivial ear.
- W.l.o.g., pendant ears come last, followed only by trivial ears.

Ear-decompositions

Write $G = P_0 + P_1 + \cdots + P_k$, where P_0 is a single vertex, and each P_i ($i = 1, \ldots, k$) is either

- a circuit sharing exactly one vertex with $P_0 + \cdots + P_{i-1}$, or
- a path sharing exactly its endpoints with $P_0 + \cdots + P_{i-1}$.



- A graph is 2-edge-connected iff it has an ear-decomposition.
- A graph is 2-vertex-connected iff it has an open ear-decomposition.
- A nontrivial ear is called pendant if none of its internal vertices is endpoint of another nontrivial ear.
- W.l.o.g., pendant ears come last, followed only by trivial ears.

Ear-decompositions with fewest even ears

For a 2-edge-connected graph G, let $\varphi(G)$ denote the minimum number of even ears in an ear-decomposition of G.

Theorem (Lovász [1972])

$\varphi(G) = 0$ *if and only if G is factor-critical.*

Ear-decompositions with fewest even ears

For a 2-edge-connected graph G, let $\varphi(G)$ denote the minimum number of even ears in an ear-decomposition of G.

Theorem (Lovász [1972])

$\varphi(G) = 0$ *if and only if* G *is factor-critical.*

Corollary (Lovász and Plummer [1986])

For every 2-vertex-connected factor-critical graph one can compute an open odd ear-decomposition in polynomial time.

Ear-decompositions with fewest even ears

For a 2-edge-connected graph G, let $\varphi(G)$ denote the minimum number of even ears in an ear-decomposition of G.

Theorem (Lovász [1972])

$\varphi(G) = 0$ *if and only if G is factor-critical.*

Corollary (Lovász and Plummer [1986])

For every 2-vertex-connected factor-critical graph one can compute an open odd ear-decomposition in polynomial time.

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ *even ears can be computed in polynomial time,*

Ear-decompositions with fewest even ears

For a 2-edge-connected graph G, let $\varphi(G)$ denote the minimum number of even ears in an ear-decomposition of G.

Theorem (Lovász [1972])

$\varphi(G) = 0$ if and only if G is factor-critical.

Corollary (Lovász and Plummer [1986])

For every 2-vertex-connected factor-critical graph one can compute an open odd ear-decomposition in polynomial time.

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ even ears can be computed in polynomial time, and

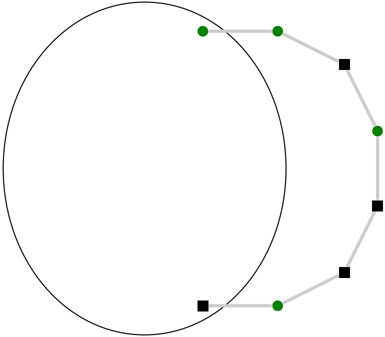
$$\frac{|V(G)|-1+\varphi(G)}{2} \;=\; \max\Big\{\min\{|J| : J \text{ is a } T\text{-join}\} : T \subseteq V(G),\ |T| \text{ even}\Big\}.$$

Proof of the easy direction: ear induction

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ even ears can be computed in polynomial time, and

$$\frac{|V(G)|-1+\varphi(G)}{2} = \max\Big\{\min\{|J| : J \text{ is a } T\text{-join}\} : T \subseteq V(G),\ |T| \text{ even}\Big\}.$$



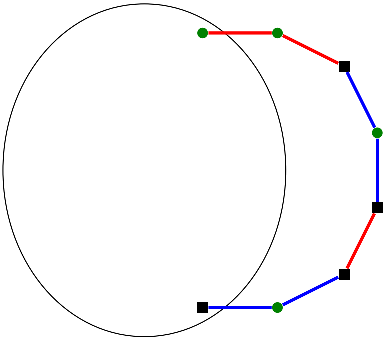
Proof of "$\geq$": fix ear-decomposition and T

Proof of the easy direction: ear induction

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ even ears can be computed in polynomial time, and

$$\tfrac{|V(G)|-1+\varphi(G)}{2} = \max\Big\{\min\{|J| : J \text{ is a } T\text{-join}\} : T \subseteq V(G),\ |T| \text{ even}\Big\}.$$



Proof of "$\geq$": fix ear-decomposition and T

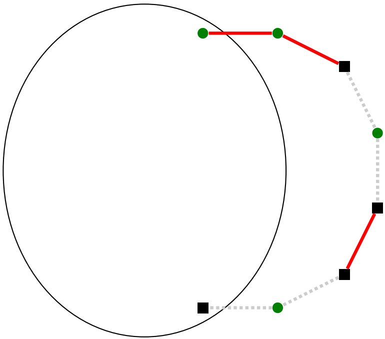
- Split pendant ear P at the vertices of T into red and blue part

Proof of the easy direction: ear induction

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ even ears can be computed in polynomial time, and

$$\frac{|V(G)|-1+\varphi(G)}{2} = \max\Big\{\min\{|J| : J \text{ is a } T\text{-join}\} : T \subseteq V(G),\ |T| \text{ even}\Big\}.$$



Proof of "$\geq$": fix ear-decomposition and T

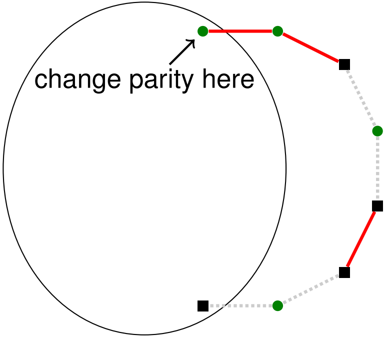
- Split pendant ear P at the vertices of T into red and blue part
- Take the smaller part

Proof of the easy direction: ear induction

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ even ears can be computed in polynomial time, and

$$\frac{|V(G)|-1+\varphi(G)}{2} = \max\Big\{\min\{|J| : J \text{ is a } T\text{-join}\} : T \subseteq V(G),\ |T| \text{ even}\Big\}.$$



Proof of "$\geq$": fix ear-decomposition and T

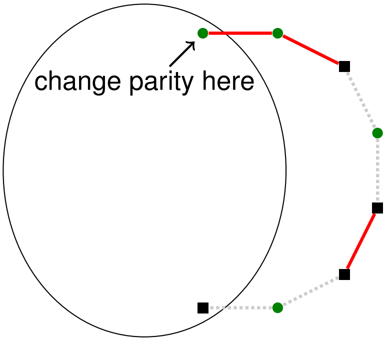
- Split pendant ear P at the vertices of T into red and blue part
- Take the smaller part
- Change parity of an endpoint of P if necessary; delete P; iterate

Proof of the easy direction: ear induction

Theorem (Frank [1993])

Let G be a 2-edge-connected graph. Then an ear-decomposition with $\varphi(G)$ even ears can be computed in polynomial time, and

$$\frac{|V(G)|-1+\varphi(G)}{2} = \max\Big\{\min\{|J| : J \text{ is a } T\text{-join}\} : T \subseteq V(G),\ |T| \text{ even}\Big\}.$$



(k_{even} := # even ears)

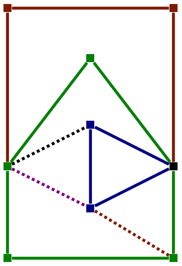
Proof of "$\geq$": fix ear-decomposition and T

- Split pendant ear P at the vertices of T into red and blue part
- Take the smaller part
- Change parity of an endpoint of P if necessary; delete P; iterate
- This yields a T-join with $\leq \frac{1}{2}(|V(G)| - 1 + k_{\text{even}})$ edges

□

Ear-decompositions for 2ECSS

Simple algorithm for 2ECSS: delete all trivial ears.

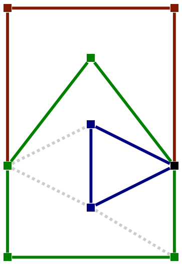


Ear-decompositions for 2ECSS

Simple algorithm for 2ECSS: delete all trivial ears.
The remaining number of edges is at most

$$\frac{5}{4}(|V(G)| - 1 + k_{\mathrm{even}}) + \frac{1}{2}k_3,$$

where k_{even} is the number of even ears,
and k_3 is the number of ears of length 3.

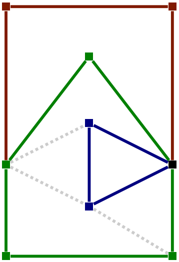


Ear-decompositions for 2ECSS

Simple algorithm for 2ECSS: delete all trivial ears.
The remaining number of edges is at most

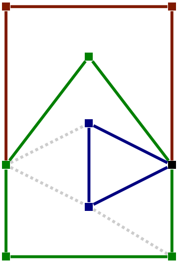
$$\frac{5}{4}(|V(G)| - 1 + k_{\mathrm{even}}) + \frac{1}{2}k_3,$$

where k_{even} is the number of even ears,
and k_3 is the number of ears of length 3.



Note: Every ear-decomposition has at least $\varphi(G)$ nontrivial ears
$\Rightarrow$ Every 2ECSS has at least $L_\varphi := |V(G)| - 1 + \varphi(G)$ edges.

Ear-decompositions for 2ECSS



Simple algorithm for 2ECSS: delete all trivial ears.
The remaining number of edges is at most

$$\tfrac{5}{4}(|V(G)| - 1 + k_{\mathrm{even}}) + \tfrac{1}{2}k_3,$$

where k_{even} is the number of even ears,
and k_3 is the number of ears of length 3.

Note: Every ear-decomposition has at least $\varphi(G)$ nontrivial ears
$\Rightarrow$ Every 2ECSS has at least $L_\varphi := |V(G)| - 1 + \varphi(G)$ edges.

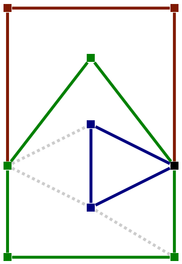
By Frank's theorem, we get $k_{\mathrm{even}} = \varphi(G)$. Note $k_3 \leq \frac{1}{2}(|V(G)| - 1)$.

Ear-decompositions for 2ECSS

Simple algorithm for 2ECSS: delete all trivial ears.
The remaining number of edges is at most

$$\frac{5}{4}(|V(G)| - 1 + k_{\text{even}}) + \frac{1}{2}k_3,$$

where k_{even} is the number of even ears,
and k_3 is the number of ears of length 3.



Note: Every ear-decomposition has at least $\varphi(G)$ nontrivial ears
$\Rightarrow$ Every 2ECSS has at least $L_\varphi := |V(G)| - 1 + \varphi(G)$ edges.

By Frank's theorem, we get $k_{\text{even}} = \varphi(G)$. Note $k_3 \leq \frac{1}{2}(|V(G)| - 1)$.

This immediately yields a $\frac{3}{2}$-approximation for 2ECSS

(was improved to $\frac{17}{12}$ by Cheriyan, Sebő and Szigeti [2001])

New algorithm for 2ECSS

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

New algorithm for 2ECSS

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

New algorithm for 2ECSS

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

New algorithm for 2ECSS

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

New algorithm for 2ECSS

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

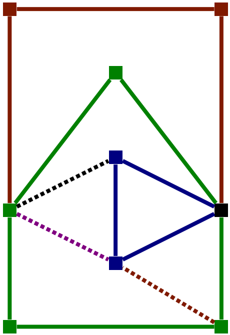
Nice ear-decompositions

An ear-decomposition is called nice if

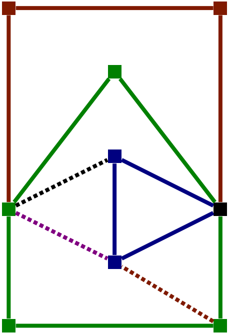
(i) the number of even ears is minimum,

(ii) all short ears (length 2 or 3) are pendant,

(iii) and there are no edges connecting internal vertices of different short ears.



Nice ear-decompositions



An ear-decomposition is called nice if

(i) the number of even ears is minimum,

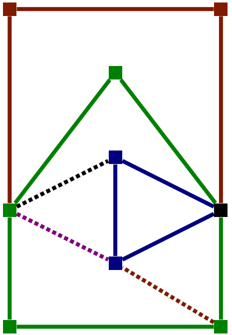
(ii) all short ears (length 2 or 3) are pendant,

(iii) and there are no edges connecting internal vertices of different short ears.

Lemma (Cheriyan, Sebő, Szigeti [2001])

A nice ear-decomposition can be computed in polynomial time.

Nice ear-decompositions



An ear-decomposition is called nice if

(i) the number of even ears is minimum,

(ii) all short ears (length 2 or 3) are pendant,

(iii) and there are no edges connecting internal vertices of different short ears.

Lemma (Cheriyan, Sebő, Szigeti [2001])

A nice ear-decomposition can be computed in polynomial time.

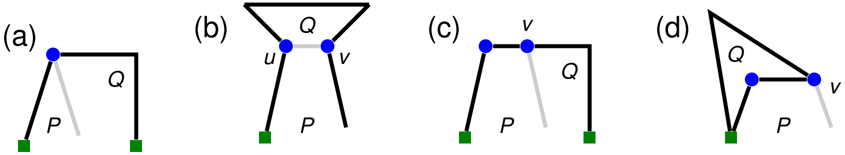
Sketch of Proof:

- Compute an ear-decomp. with fewest even ears (Frank [1993])
- Subdivide one edge of each even ear (⇒ factor-critical graph)
- Compute an open odd ear-decomp. (Lovász, Plummer [1986])
- Undo subdivisions ⇒ open ear-decomp. with fewest even ears
- Replace non-pendant short ears
- Replace adjacent short ears

□

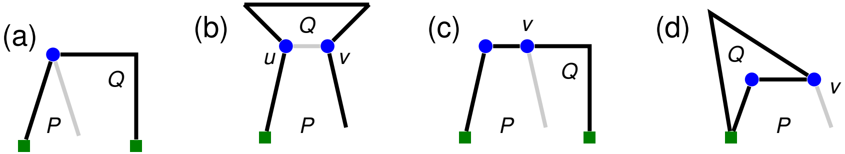
Sketch of proof (some details)

- Replace non-pendant short ears

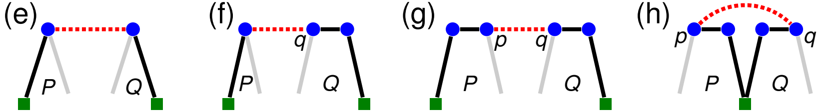


Sketch of proof (some details)

- Replace non-pendant short ears

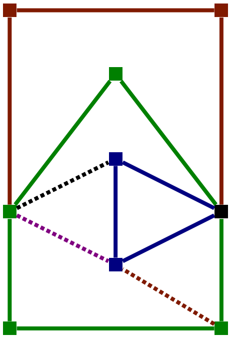


- Replace adjacent short ears



Optimizing short ears

- Adding all short ears leaves some number of connected components

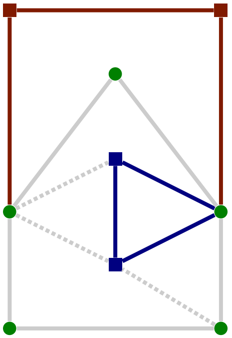


Recall: An ear-decomposition is called nice if

(i) the number of even ears is minimum,
(ii) all short ears (length 2 or 3) are pendant,
(iii) and there are no edges connecting internal vertices of different short ears.

Optimizing short ears

- Adding all short ears leaves some number of connected components

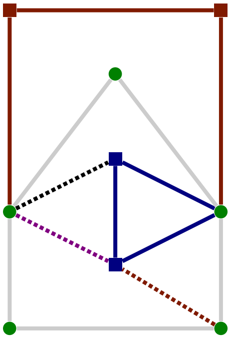


Recall: An ear-decomposition is called nice if

(i) the number of even ears is minimum,
(ii) all short ears (length 2 or 3) are pendant,
(iii) and there are no edges connecting internal vertices of different short ears.

Optimizing short ears

- Adding all short ears leaves some number of connected components
- Internal vertices of short ears may be incident to trivial ears

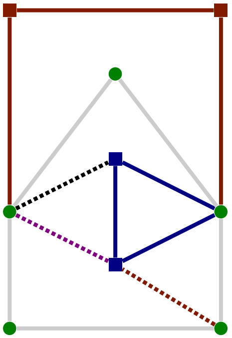


Recall: An ear-decomposition is called nice if

(i) the number of even ears is minimum,
(ii) all short ears (length 2 or 3) are pendant,
(iii) and there are no edges connecting internal vertices of different short ears.

Optimizing short ears

- Adding all short ears leaves some number of connected components
- Internal vertices of short ears may be incident to trivial ears
- These can be used to replace some short ears by other short ears

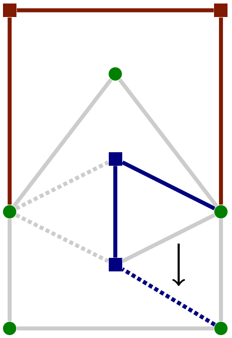


Recall: An ear-decomposition is called nice if

(i) the number of even ears is minimum,
(ii) all short ears (length 2 or 3) are pendant,
(iii) and there are no edges connecting internal vertices of different short ears.

Optimizing short ears

- Adding all short ears leaves some number of connected components
- Internal vertices of short ears may be incident to trivial ears
- These can be used to replace some short ears by other short ears
- Goal: minimize the resulting number of connected components

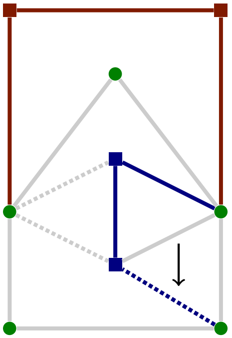


Recall: An ear-decomposition is called nice if

(i) the number of even ears is minimum,
(ii) all short ears (length 2 or 3) are pendant,
(iii) and there are no edges connecting internal vertices of different short ears.

Optimizing short ears

- Adding all short ears leaves some number of connected components
- Internal vertices of short ears may be incident to trivial ears
- These can be used to replace some short ears by other short ears
- Goal: minimize the resulting number of connected components



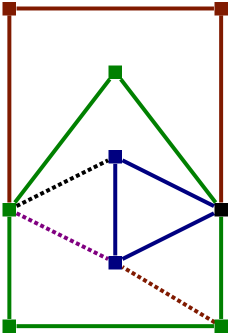
Note: Replacing some short ears by other ears (with the same internal vertices) will maintain a nice ear-decomposition.

Recall: An ear-decomposition is called nice if

(i) the number of even ears is minimum,
(ii) all short ears (length 2 or 3) are pendant,
(iii) and there are no edges connecting internal vertices of different short ears.

Eardrums and earmuffs

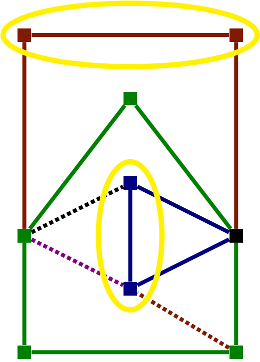
An eardrum is the set of components of an induced subgraph with maximum degree 1.



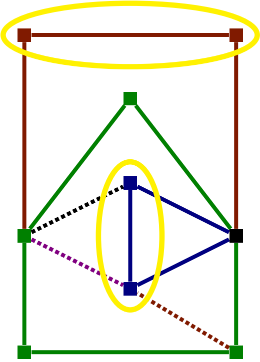
Eardrums and earmuffs

An eardrum is the set of components of an induced subgraph with maximum degree 1.

Example: the sets of internal vertices of short ears in a nice ear-decomposition.



Eardrums and earmuffs

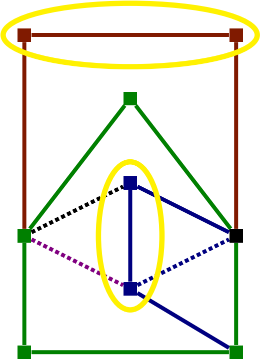


An eardrum is the set of components of an induced subgraph with maximum degree 1.

Example: the sets of internal vertices of short ears in a nice ear-decomposition.

Let M be an eardrum. For $f \in M$ let $\mathcal{P}_f$ be the set of paths with internal vertices f.

Eardrums and earmuffs



An eardrum is the set of components of an induced subgraph with maximum degree 1.

Example: the sets of internal vertices of short ears in a nice ear-decomposition.

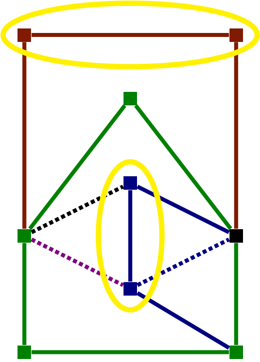
Let M be an eardrum. For $f \in M$ let $\mathcal{P}_f$ be the set of paths with internal vertices f.

An earmuff is a set of paths $\{P_f : f \in F\}$ with $F \subseteq M$, $P_f \in \mathcal{P}_f$ for $f \in F$, and $(V(G), \bigcup_{f \in F} E(P_f))$ is a forest.

Theorem

A maximum earmuff can be computed in polynomial time.

Eardrums and earmuffs



An eardrum is the set of components of an induced subgraph with maximum degree 1.

Example: the sets of internal vertices of short ears in a nice ear-decomposition.

Let M be an eardrum. For $f \in M$ let $\mathcal{P}_f$ be the set of paths with internal vertices f.

An earmuff is a set of paths $\{P_f : f \in F\}$ with $F \subseteq M$, $P_f \in \mathcal{P}_f$ for $f \in F$, and $(V(G), \bigcup_{f \in F} E(P_f))$ is a forest.

Theorem

A maximum earmuff can be computed in polynomial time.

Call the maximum $\mu(G, M)$.

First solution: matroid intersection

- Represent each path $P \in \mathcal{P}_f$ ($f \in M$) by the set e_P of its two endpoints; let $E_f := \{e_P : P \in \mathcal{P}_f\}$.
- Let r be the rank function of the cycle matroid of the complete graph on $V(G)$.

First solution: matroid intersection

- Represent each path $P \in \mathcal{P}_f$ ($f \in M$) by the set e_P of its two endpoints; let $E_f := \{e_P : P \in \mathcal{P}_f\}$.
- Let r be the rank function of the cycle matroid of the complete graph on $V(G)$.

Theorem (Rado [1942])

Let E be a finite set and r the rank function of a matroid on E. Let $E_1, E_2, \ldots, E_k \subseteq E$. Then

$$\max\{r(\{e_1, \ldots, e_k\}) : e_i \in E_i \ (i = 1, \ldots, k)\} = \min\left\{r\left(\bigcup_{i \in I} E_i\right) + k - |I| : I \subseteq \{1, \ldots, k\}\right\}.$$

First solution: matroid intersection

- Represent each path $P \in \mathcal{P}_f$ ($f \in M$) by the set e_P of its two endpoints; let $E_f := \{e_P : P \in \mathcal{P}_f\}$.
- Let r be the rank function of the cycle matroid of the complete graph on $V(G)$.

Theorem (Rado [1942])

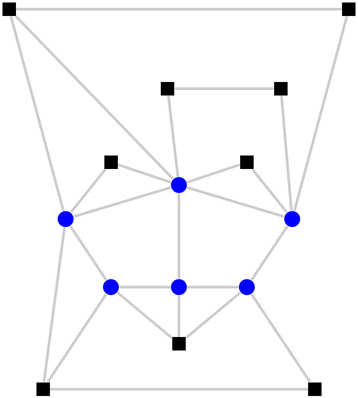
Let E be a finite set and r the rank function of a matroid on E. Let $E_1, E_2, \ldots, E_k \subseteq E$. Then

$$\max\{r(\{e_1, \ldots, e_k\}) : e_i \in E_i \ (i = 1, \ldots, k)\} = \min\left\{r\left(\bigcup_{i \in I} E_i\right) + k - |I| : I \subseteq \{1, \ldots, k\}\right\}.$$

Note: Special case of matroid intersection.

Earmuff maximization: example

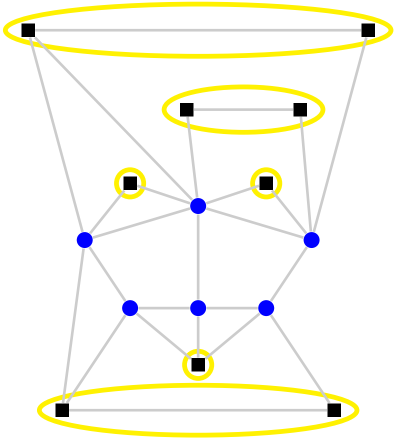
- ■ vertex in V_M
- ● vertex in $V(G) \setminus V_M$



Earmuff maximization: example

■ vertex in V_M

● vertex in $V(G) \setminus V_M$



○ eardrum

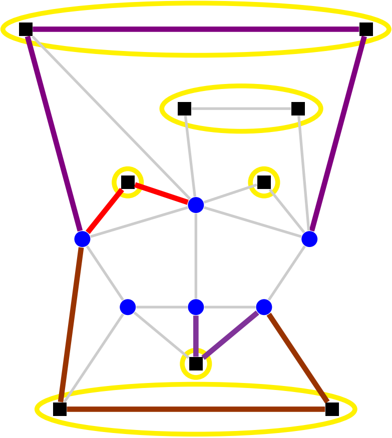
Earmuff maximization: example

- vertex in V_M
- vertex in $V(G) \setminus V_M$

maximum earmuff

other edges

eardrum

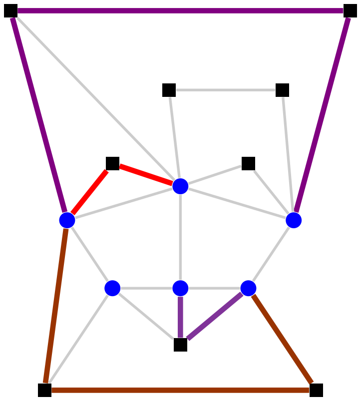


Earmuff maximization: example

- vertex in V_M
- vertex in $V(G) \setminus V_M$

maximum earmuff

other edges



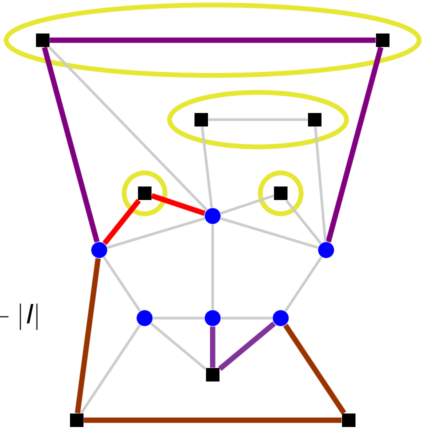
Earmuff maximization: example

- vertex in V_M
- vertex in $V(G) \setminus V_M$

maximum earmuff

other edges

sets in I, with
$\mu(G, M) = r(\bigcup_{i \in I} E_i) + |M| - |I|$



Earmuff maximization: example

■ vertex in V_M

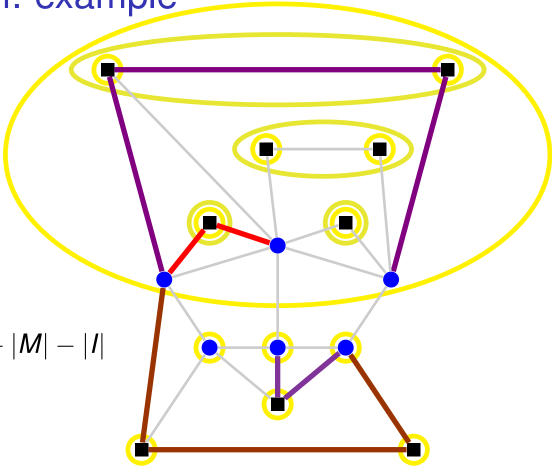
● vertex in $V(G) \setminus V_M$

maximum earmuff

other edges

sets in I, with
$\mu(G, M) = r(\bigcup_{i \in I} E_i) + |M| - |I|$

cuts in dual solution



No edge belongs to more than 2 cuts. Number of cuts $\geq$
$|V(G)| + |I| - r(\bigcup_{i \in I} E_i) - 1 \; = \; |V(G)| - 1 + |M| - \mu(G, M)$

Earmuff maximization: example

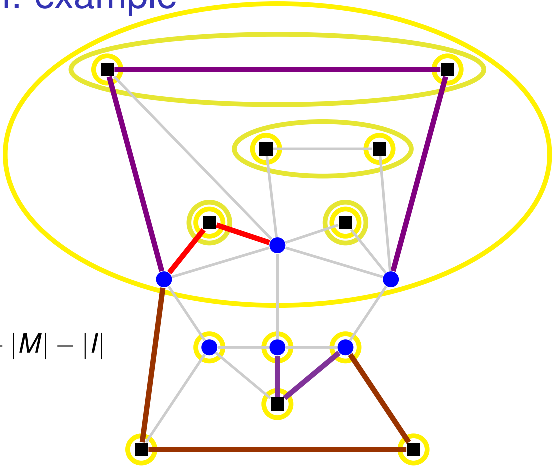
- vertex in V_M
- vertex in $V(G) \setminus V_M$

maximum earmuff

other edges

sets in I, with $\mu(G,M) = r(\bigcup_{i\in I} E_i) + |M| - |I|$

cuts in dual solution



No edge belongs to more than 2 cuts. Number of cuts $\geq$
$|V(G)| + |I| - r(\bigcup_{i\in I} E_i) - 1 \ = \ |V(G)| - 1 + |M| - \mu(G,M)$

Theorem

Any 2ECSS has at least $L_\mu := |V(G)| - 1 + |M| - \mu(G,M)$ *edges.* $\square$

New algorithm for 2ECSS

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

New algorithm for 2ECSS

✓ Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

New algorithm for 2ECSS

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

New algorithm for 2ECSS

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- ✓ Take all edges of pendant ears.
- ▶ Add edges to obtain connectivity.
- ▶ Add edges to correct parity.

Alternatively:

- ▶ Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

New algorithm for 2ECSS

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- ✓ Take all edges of pendant ears.
- ✓ Add edges to obtain connectivity.
- ▸ Add edges to correct parity.

Alternatively:

- ▸ Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

New algorithm for 2ECSS

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- ✓ Take all edges of pendant ears.
- ✓ Add edges to obtain connectivity.
- ✓ Add edges to correct parity.

Alternatively:

- Take all edges of nontrivial ears.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Alternative yields an 2ECSS with at most $\frac{5}{4}L + \frac{1}{2}\pi$ edges.

$\longrightarrow$ The better of the two 2ECSSs has at most $\frac{4}{3}L$ edges.

New algorithm for 2ECSS: proof of main theorem

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- ✓ Take all edges of pendant ears.
- ✓ Add edges to obtain connectivity.
- ✓ Add edges to correct parity.

$\left.\right\}$ $L_\mu + \pi_{\mathrm{long}}$ edges.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

New algorithm for 2ECSS: proof of main theorem

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- ✓ Take all edges of pendant ears.
- ✓ Add edges to obtain connectivity.

$\}$ $L_\mu + \pi_{\mathrm{long}}$ edges.

- ✓ Add edges to correct parity. $\}$ $\frac{1}{2}(|V_0| - 1 + \varphi_0(G))$ edges.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

New algorithm for 2ECSS: proof of main theorem

- ✓ Compute a nice ear-decomposition.
- ✓ Optimize short ears so that they serve best for connectivity.

Note: number of even ears is minimum, all short ears are pendant

- ✓ Take all edges of pendant ears.
- ✓ Add edges to obtain connectivity.
- ✓ Add edges to correct parity.

$\}$ $L_\mu + \pi_{\text{long}}$ edges.

$\}$ $\frac{1}{2}(|V_0|-1+\varphi_0(G))$ edges.

Theorem

The new algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Proof: Since $|V_0| \leq |V(G)| + \varphi_\pi(G) - 2\pi_{\text{short}} - 4\pi_{\text{long}}$, correcting parity needs at most $\frac{1}{2}L_\varphi - \pi - \pi_{\text{long}}$ edges. $\square$

New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.

- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.
- Delete all 1-ears. In each of the resulting blocks:
- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.
- Delete all 1-ears. In each of the resulting blocks:
- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Theorem

In each block, this algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.
- Delete all 1-ears. In each of the resulting blocks:
- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Apply lemma of Mömke-Svensson.

Theorem

In each block, this algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.
- Delete all 1-ears. In each of the resulting blocks:
- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Apply lemma of Mömke-Svensson.

Theorem

In each block, this algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Theorem

Mömke-Svensson yields a tour with at most $\frac{4}{3}L + \frac{2}{3}\pi$ edges.

$\longrightarrow$ The better of the two tours has at most $\frac{7}{5}L$ edges.

The Mömke-Svensson lemma

Definition (Mömke and Svensson [2011])

Let G be a 2-vertex-connected graph.
A removable pairing in G consists of a set R of removable edges and a set of pairwise disjoint pairs of elements of R such that

- deleting any edge set $S \subseteq R$ that contains at most edge out of each pair does not disconnect the graph
- the two edges of any pair share a vertex, and this vertex is incident to another edge

Theorem (Mömke and Svensson [2011])

Given a 2-vertex-connected graph G and a removable pairing $(R, \mathcal{P})$. Then one can find a tour with at most $\frac{4}{3}|E(G)| - \frac{2}{3}|R|$ edges in polynomial time.

Mömke-Svensson applied to ear-decompositions

Theorem

Given a 2-vertex-connected graph G with an ear-decomposition in which all ears are nontrivial. Then one can find a tour with at most $\frac{4}{3}(|V(G)| - 1) + \frac{2}{3}\pi$ *edges in polynomial time.*

Mömke-Svensson applied to ear-decompositions

Theorem

Given a 2-vertex-connected graph G with an ear-decomposition in which all ears are nontrivial. Then one can find a tour with at most $\frac{4}{3}(|V(G)| - 1) + \frac{2}{3}\pi$ *edges in polynomial time.*

Proof: We define the removable pairing as follows:

- For each pendant ear, R will contain exactly one of its edges.
- For each non-pendant ear, R will contain a pair of edges incident to an internal vertex that is endpoint of another ear.

Mömke-Svensson applied to ear-decompositions

Theorem

Given a 2-vertex-connected graph G with an ear-decomposition in which all ears are nontrivial. Then one can find a tour with at most $\frac{4}{3}(|V(G)| - 1) + \frac{2}{3}\pi$ edges in polynomial time.

Proof: We define the removable pairing as follows:

- For each pendant ear, R will contain exactly one of its edges.
- For each non-pendant ear, R will contain a pair of edges incident to an internal vertex that is endpoint of another ear.

So $|R| = 2(|E(G)| - (|V(G)| - 1)) - \pi$.

Mömke-Svensson applied to ear-decompositions

Theorem

Given a 2-vertex-connected graph G with an ear-decomposition in which all ears are nontrivial. Then one can find a tour with at most $\frac{4}{3}(|V(G)| - 1) + \frac{2}{3}\pi$ edges in polynomial time.

Proof: We define the removable pairing as follows:

- For each pendant ear, R will contain exactly one of its edges.
- For each non-pendant ear, R will contain a pair of edges incident to an internal vertex that is endpoint of another ear.

So $|R| = 2(|E(G)| - (|V(G)| - 1)) - \pi$.

The Mömke-Svensson lemma yields a tour with at most $\frac{4}{3}|E(G)| - \frac{2}{3}|R|$ edges. $\square$

Proof of Mömke-Svensson lemma

Theorem (Mömke and Svensson [2011])

Given a 2-vertex-connected graph G and a removable pairing $(R, \mathcal{P})$. Then one can find a tour with at most $\frac{4}{3}|E(G)| - \frac{2}{3}|R|$ edges in polynomial time.

Proof of Mömke-Svensson lemma

Theorem (Mömke and Svensson [2011])

Given a 2-vertex-connected graph G and a removable pairing $(R, \mathcal{P})$. Then one can find a tour with at most $\frac{4}{3}|E(G)| - \frac{2}{3}|R|$ edges in polynomial time.

Proof: Find an odd join J containing at most one edge of each pair.
Add a second copy of each edge in $J \setminus R$.
Delete the edges in $J \cap R$.
We get a tour with $|E(G)| + c(J)$ edges,
where $c(e) = -1$ for $e \in R$ and $c(e) = 1$ for $e \notin R$.

Proof of Mömke-Svensson lemma

Theorem (Mömke and Svensson [2011])

Given a 2-vertex-connected graph G and a removable pairing $(R, \mathcal{P})$. Then one can find a tour with at most $\frac{4}{3}|E(G)| - \frac{2}{3}|R|$ edges in polynomial time.

Proof: Find an odd join J containing at most one edge of each pair.
Add a second copy of each edge in $J \setminus R$.
Delete the edges in $J \cap R$.
We get a tour with $|E(G)| + c(J)$ edges,
where $c(e) = -1$ for $e \in R$ and $c(e) = 1$ for $e \notin R$.

Construct G' as follows. For each pair P with edges $\{u, v\}, \{v, w\}$, add a vertex v_P and an edge $\{v_P, v\}$ of weight 0, and replace the two edges in P by $\{u, v_P\}, \{v_P, w\}$.

Proof of Mömke-Svensson lemma

Theorem (Mömke and Svensson [2011])

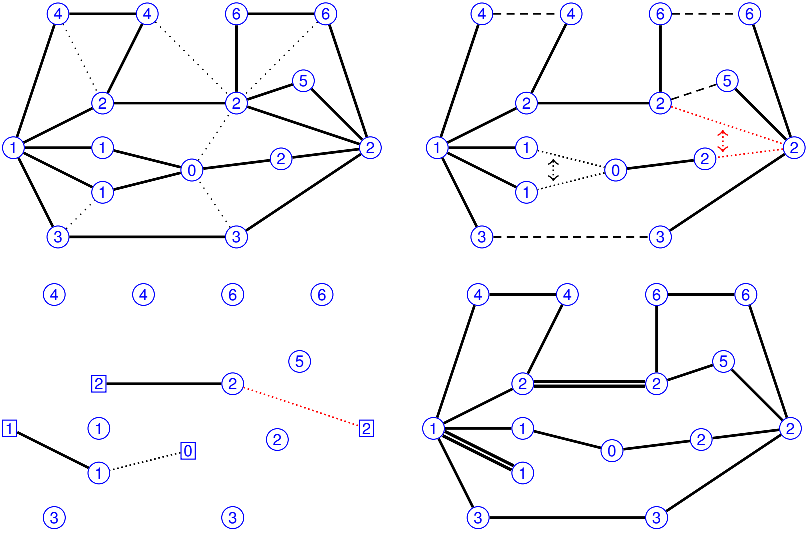
Given a 2-vertex-connected graph G and a removable pairing $(R, \mathcal{P})$. Then one can find a tour with at most $\frac{4}{3}|E(G)| - \frac{2}{3}|R|$ edges in polynomial time.

Proof: Find an odd join J containing at most one edge of each pair.
Add a second copy of each edge in $J \setminus R$.
Delete the edges in $J \cap R$.
We get a tour with $|E(G)| + c(J)$ edges,
where $c(e) = -1$ for $e \in R$ and $c(e) = 1$ for $e \notin R$.

Construct G' as follows. For each pair P with edges $\{u, v\}, \{v, w\}$, add a vertex v_P and an edge $\{v_P, v\}$ of weight 0, and replace the two edges in P by $\{u, v_P\}, \{v_P, w\}$.

The all-$\frac{1}{3}$-vector is in the odd join polytope of G', and in its face defined by $x(\delta(v_P)) = 1$ for all pairs P.
Hence there is an odd join J as required with
$c(J) \leq \frac{1}{3}c(E(G')) = \frac{1}{3}|E(G)| - \frac{2}{3}|R|$. $\square$

Example: application of Mömke-Svensson lemma



New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.
- Delete all 1-ears. In each of the resulting blocks:
- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Apply lemma of Mömke-Svensson.

Theorem

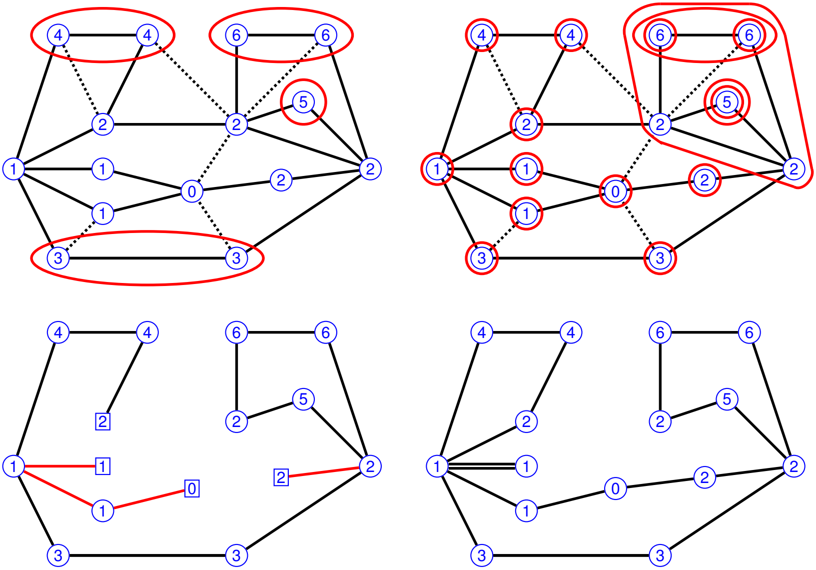
In each block, this algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Theorem

Mömke-Svensson yields a tour with at most $\frac{4}{3}L + \frac{2}{3}\pi$ edges.

$\longrightarrow$ The better of the two tours has at most $\frac{7}{5}L$ edges.

Example: Shorter tour by nicer ears



New algorithm for TSP

- Compute a nice ear-decomposition.
- Optimize short ears so that they serve best for connectivity.
- Delete all 1-ears. In each of the resulting blocks:
- Take all edges of pendant ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Apply lemma of Mömke-Svensson.

Theorem

In each block, this algorithm yields a tour with at most $\frac{3}{2}L - \pi$ edges, where L is a lower bound on the number of edges in any 2ECSS, and π is the number of pendant ears (after optimization).

Theorem

Mömke-Svensson yields a tour with at most $\frac{4}{3}L + \frac{2}{3}\pi$ edges.

$\longrightarrow$ The better of the two tours has at most $\frac{7}{5}L$ edges.

New algorithm for connected-T-joins

- Compute a nice ear-decomposition.
- Optimize clean ears so that they serve best for connectivity.

(Clean ears are short ears without an internal vertex in T.)

- Take all edges of clean ears.
- Apply ear induction to pendant but not clean ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Apply ear induction to all ears.

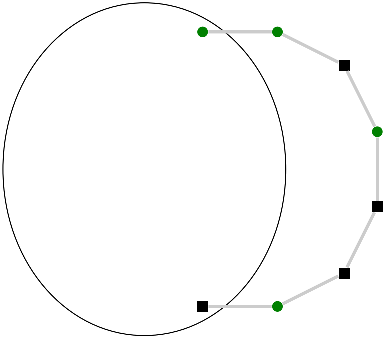
Theorem

The new algorithm yields a connected-T-join with at most $\frac{3}{2}L + \frac{1}{2}\varphi(G) - \pi$ edges, where L is a lower bound and π is the number of pendant ears (after optimization).

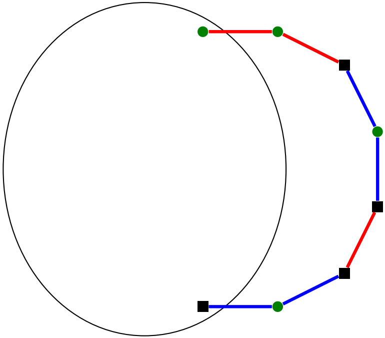
Alternative yields at most $\frac{3}{2}L - \frac{1}{2}\varphi(G) + \pi$ edges.

$\longrightarrow$ The better of the two has at most $\frac{3}{2}L$ edges.

Tool for connected-T-joins: ear induction

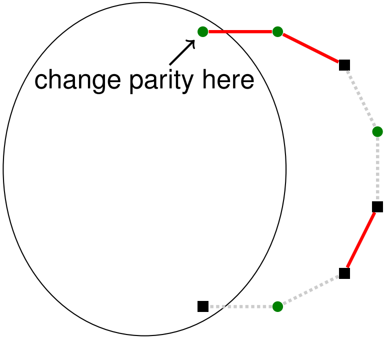


Tool for connected-T-joins: ear induction



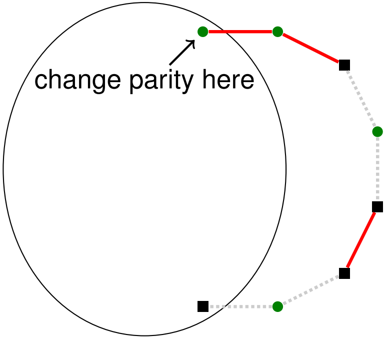
- Split pendant ear at vertices in T (that have wrong parity so far)

Tool for connected-T-joins: ear induction



- Split pendant ear at vertices in T (that have wrong parity so far)
- Take smaller part for obtaining a T-join

Tool for connected-T-joins: ear induction

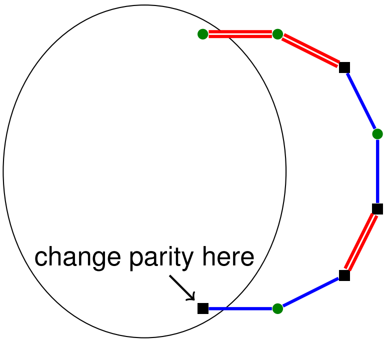


- Split pendant ear at vertices in T (that have wrong parity so far)
- Take smaller part for obtaining a T-join

This yields a

- T-join with $\leq \frac{1}{2}(|V(G)| - 1 + k_{\text{even}})$ edges

Tool for connected-T-joins: ear induction

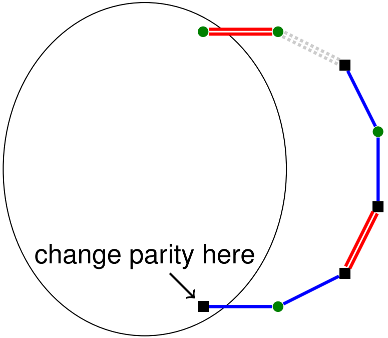


- Split pendant ear at vertices in T (that have wrong parity so far)
- Take smaller part for obtaining a T-join
- Double smaller part for obtaining a connected-T-join

This yields a

- T-join with $\leq \frac{1}{2}(|V(G)| - 1 + k_{\text{even}})$ edges

Tool for connected-T-joins: ear induction

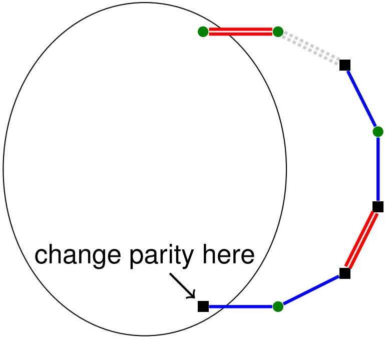


- Split pendant ear at vertices in T (that have wrong parity so far)
- Take smaller part for obtaining a T-join
- Double smaller part for obtaining a connected-T-join
- May delete one pair of parallel edges (if there is one)

This yields a

- T-join with $\leq \frac{1}{2}(|V(G)| - 1 + k_{\text{even}})$ edges

Tool for connected-T-joins: ear induction



- Split pendant ear at vertices in T (that have wrong parity so far)
- Take smaller part for obtaining a T-join
- Double smaller part for obtaining a connected-T-join
- May delete one pair of parallel edges (if there is one)

This yields a

- T-join with $\leq \frac{1}{2}(|V(G)| - 1 + k_{\text{even}})$ edges
- connected T-join with $\leq \frac{3}{2}(|V(G)| - 1) + \pi_{\text{clean}} - \frac{1}{2}k_{\text{even}} - k_{\text{odd}}$ edges

$\leq \frac{3}{2}(|V(G)|-1) + \frac{1}{2}k_{\text{even}} - \pi_{\text{notclean}}$ and $\leq \frac{3}{2}(|V(G)|-1) - \frac{1}{2}k_{\text{even}} + \pi$

New algorithm for connected-T-joins

- Compute a nice ear-decomposition.
- Optimize clean ears so that they serve best for connectivity.

Clean ears are short ears without an internal vertex in T.

- Take all edges of clean ears.
- Apply ear induction to pendant but not clean ears.
- Add edges to obtain connectivity.
- Add edges to correct parity.

Alternatively:

- Apply ear induction to all ears.

Theorem

The new algorithm yields a connected-T-join with at most $\frac{3}{2}L + \frac{1}{2}\varphi(G) - \pi$ edges, where L is a lower bound and π is the number of pendant ears (after optimization).

Alternative yields at most $\frac{3}{2}L - \frac{1}{2}\varphi(G) + \pi$ edges.

$\longrightarrow$ The better of the two has at most $\frac{3}{2}L$ edges.

Using nicer ears, more generally and formally

Definition

Let G be a graph with a nice ear-decomposition, $T \subseteq V(G)$, $|T|$ even.
An ear is clean if it is short and T contains none of its internal vertices.
Let M contain for each clean ear the set of its internal vertices.

Theorem

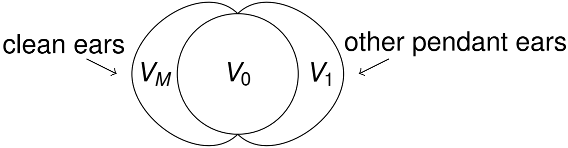
Let G, T, M be as above. Suppose the given ear-decomposition contains a maximum earmuff for M (of cardinality μ).
Then a connected-T-join of cardinality at most $L_\mu + \frac{1}{2}L_\varphi - \pi$ can be constructed in $O(|V(G)|^3)$ time.

Recall:

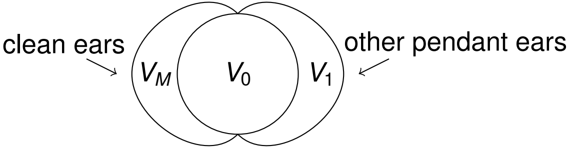
$$\begin{aligned} \pi &= \text{number of pendant ears} \\ L_\varphi &= |V(G)| - 1 + \varphi(G) \\ L_\mu &= |V(G)| - 1 + |M| - \mu \end{aligned}$$

L_μ is a lower bound on the optimum (in fact, on the LP value).
L_φ is a lower bound if $T = \emptyset$.

Proof

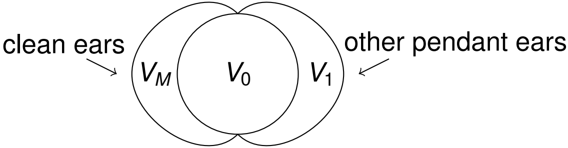


Proof



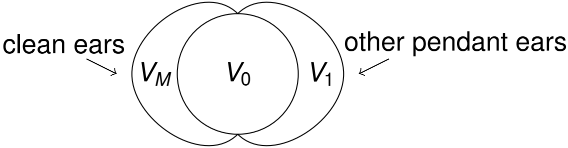
- take all edges of clean ears $\frac{3}{2}|V_M| + \frac{1}{2}\varphi_M$

Proof



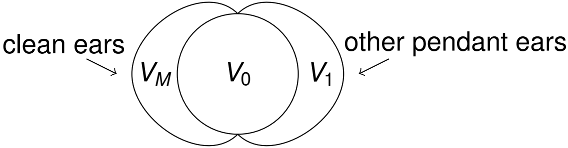
- take all edges of clean ears $\qquad \frac{3}{2}|V_M| + \frac{1}{2}\varphi_M$
- add edges of $G[V_0]$ so that
  $(V_M \cup V_0, E_1 \cup E_2)$ is connected $\qquad |V_0| - 1 - \mu$

Proof



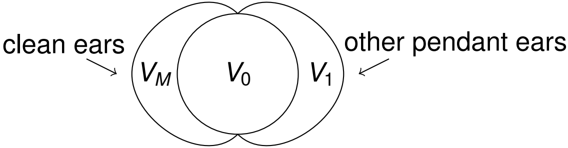
- take all edges of clean ears $\qquad \frac{3}{2}|V_M| + \frac{1}{2}\varphi_M$
- add edges of $G[V_0]$ so that $(V_M \cup V_0, E_1 \cup E_2)$ is connected $\qquad |V_0| - 1 - \mu$
- apply ear induction to non-clean pendant ears $\qquad \frac{3}{2}|V_1| + \frac{1}{2}\varphi_1 - (\pi - |M|)$

Proof



- take all edges of clean ears $\quad \frac{3}{2}|V_M| + \frac{1}{2}\varphi_M$
- add edges of $G[V_0]$ so that $(V_M \cup V_0, E_1 \cup E_2)$ is connected $\quad |V_0| - 1 - \mu$
- apply ear induction to non-clean pendant ears $\quad \frac{3}{2}|V_1| + \frac{1}{2}\varphi_1 - (\pi - |M|)$
- Correct parities on $G[V_0]$ $\quad \frac{1}{2}(|V_0| + \varphi_0 - 1)$

Proof



► take all edges of clean ears	$\frac{3}{2}\lvert V_M\rvert + \frac{1}{2}\varphi_M$
► add edges of $G[V_0]$ so that $(V_M \cup V_0, E_1 \cup E_2)$ is connected	$\lvert V_0\rvert - 1 - \mu$
► apply ear induction to non-clean pendant ears	$\frac{3}{2}\lvert V_1\rvert + \frac{1}{2}\varphi_1 - (\pi - \lvert M\rvert)$
► Correct parities on $G[V_0]$	$\frac{1}{2}(\lvert V_0\rvert + \varphi_0 - 1)$

Adding up, using $\varphi(G) = \varphi_0 + \varphi_1 + \varphi_M$, yields

$$\frac{3}{2}(|V(G)| - 1) + |M| - \mu + \frac{1}{2}\varphi(G) - \pi \quad = \quad L_\mu + \frac{1}{2}L_\varphi - \pi$$

□

Second solution to earmuff maximization

Let M be a eardrum with $V_M \cap T = \emptyset$ (e.g., contain for each clean ear the set of its internal vertices).

- Let $U = V(G) \setminus V_M$, where $V_M = \bigcup M$.
- Represent $\mathcal{P}_f$ ($f \in M$) by the set U_f of its endpoints.
- Sufficient to find a maximum cardinality subset $F \subseteq M$ with a forest representative system $(e_f)_{f \in F}$ of $(U_f)_{f \in F}$, i.e.,
  - $e_f \in \binom{U_f}{2}$ for all $f \in F$,
  - $e_f \neq e_{f'}$ for $f \neq f'$, and
  - the graph $(U, \{e_f : f \in F\})$ is a forest.

Second solution to earmuff maximization

Let M be a eardrum with $V_M \cap T = \emptyset$ (e.g., contain for each clean ear the set of its internal vertices).

- Let $U = V(G) \setminus V_M$, where $V_M = \bigcup M$.
- Represent $\mathcal{P}_f$ ($f \in M$) by the set U_f of its endpoints.
- Sufficient to find a maximum cardinality subset $F \subseteq M$ with a forest representative system $(e_f)_{f \in F}$ of $(U_f)_{f \in F}$, i.e.,
  - $e_f \in \binom{U_f}{2}$ for all $f \in F$,
  - $e_f \neq e_{f'}$ for $f \neq f'$, and
  - the graph $(U, \{e_f : f \in F\})$ is a forest.

Theorem (≈ Lovász [1970])

This maximum is $\mu =$

$$\min\left\{ |M| - \sum_{W \in \mathcal{W}} \big(|\{f \in M : U_f \subseteq W\}| - (|W| - 1) \big) : \mathcal{W} \textit{ is a partition of } U \right\}.$$

(This holds for any finite sets U, M, $(U_f)_{f \in M}$ with $\emptyset \neq U_f \subseteq U$.)

Second solution to earmuff maximization

Let M be a eardrum with $V_M \cap T = \emptyset$ (e.g., contain for each clean ear the set of its internal vertices).

- Let $U = V(G) \setminus V_M$, where $V_M = \bigcup M$.
- Represent $\mathcal{P}_f$ ($f \in M$) by the set U_f of its endpoints.
- Sufficient to find a maximum cardinality subset $F \subseteq M$ with a forest representative system $(e_f)_{f \in F}$ of $(U_f)_{f \in F}$, i.e.,
  - $e_f \in \binom{U_f}{2}$ for all $f \in F$,
  - $e_f \neq e_{f'}$ for $f \neq f'$, and
  - the graph $(U, \{e_f : f \in F\})$ is a forest.

Theorem (≈ Lovász [1970])

This maximum is $\mu =$

$$\min\left\{ |M| - \sum_{W \in \mathcal{W}} \big(|\{f \in M : U_f \subseteq W\}| - (|W| - 1) \big) : \mathcal{W} \textit{ is a partition of } U \right\}.$$

(This holds for any finite sets U, M, $(U_f)_{f \in M}$ with $\emptyset \neq U_f \subseteq U$.)

Note: We have an algorithm with running time $O(|V(G)||E(G)|)$

Lower bounds and LP relaxations

Theorem (Cheriyan, Sebő, Szigeti [2001])

$$L_\varphi := |V(G)| - 1 + \varphi(G) \le \mathrm{LP}(G) :=$$

$$\min\left\{x(E(G)) : x \in \mathbb{R}^{E(G)}_{\geq 0},\ x(\delta(W)) \geq 2 \textit{ for all } \emptyset \neq W \subset V(G)\right\}$$

Proof (Sketch): Frank's theorem $\Rightarrow$ T $\Rightarrow$ 2-packing of T-cuts. $\square$

Lower bounds and LP relaxations

Theorem (Cheriyan, Sebő, Szigeti [2001])

$$L_\varphi \;:=\; |V(G)| - 1 + \varphi(G) \;\leq\; \mathrm{LP}(G) \;:=$$

$$\min\left\{x(E(G)) \;:\; x \in \mathbb{R}_{\geq 0}^{E(G)},\; x(\delta(W)) \geq 2 \textit{ for all } \emptyset \neq W \subset V(G)\right\}$$

Proof (Sketch): Frank's theorem $\Rightarrow$ T $\Rightarrow$ 2-packing of T-cuts. $\square$

Theorem

$$L_\mu \;:=\; |V(G)| - 1 + |M| - \mu \;\leq\; \mathrm{LP}(G, T) \;:=$$

$$\min\Big\{x(E(G)) \;:\; x \in \mathbb{R}_{\geq 0}^{E(G)},$$
$$x(\delta(W)) \geq 2 \textit{ for all } \emptyset \neq W \subset V(G) \textit{ with } |W \cap T| \textit{ even},$$
$$x(\delta(\mathcal{W})) \geq |\mathcal{W}| - 1 \textit{ for all partitions } \mathcal{W} \textit{ of } V(G)\Big\}$$

Proof (Sketch): 2-packing of cuts, including the partition from min-max theorem for forest representation systems. $\square$

Summary of Results

We obtained an improved approximation ratio of:

- $\frac{7}{5}$ for Graphic TSP
- $\frac{3}{2}$ for Connected-T-join
- $\frac{4}{3}$ for 2ECSS

- All algorithms combinatorial, running time $O(|V(G)|^3)$
- These bounds are tight.
- These are also upper bounds on the integrality ratios of the natural LPs for unit weights.

Open problems

- improve approximation ratios (to $\frac{4}{3}$ for Graphic TSP?)
- extend to weights (general metrics)
- extend to directed graphs

Open problems

- improve approximation ratios (to $\frac{4}{3}$ for Graphic TSP?)
- extend to weights (general metrics)
- extend to directed graphs

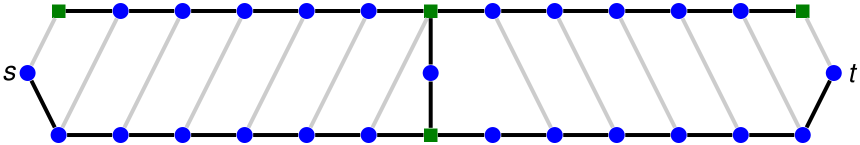
Thank you!

Open problems

- improve approximation ratios (to $\frac{4}{3}$ for Graphic TSP?)
- extend to weights (general metrics)
- extend to directed graphs

Thank you!

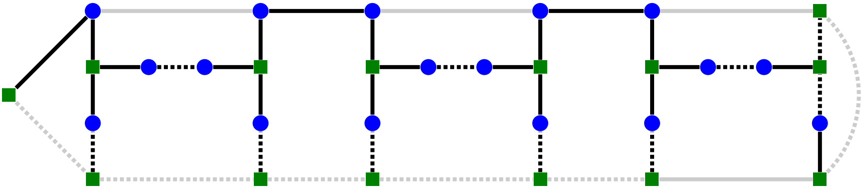
Tight example for connected-T-join



$|V(G)| = 8k + 5$ (Here $k = 3$.)
$T = \{s, t\}$
$OPT = 8k + 4$
$\varphi(G) = 2$
$\pi = 1 = \frac{1}{2}\varphi(G)$.

Algorithm computes solution with $12k + 6$ edges.

Tight example for graphic TSP



$|V(G)| = OPT = 10k + 1$ (Here $k = 3$.)

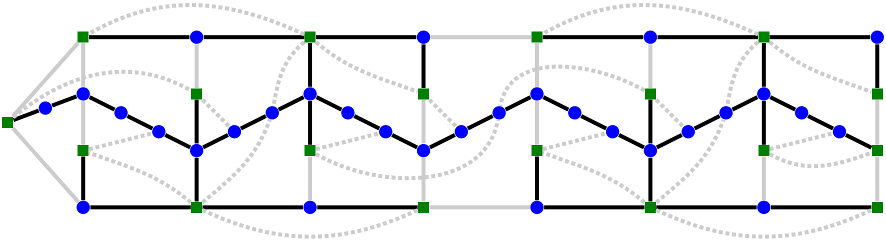
$\varphi(G) = 0$

$L = 10k$

$\pi = k = \frac{1}{10}L$.

Algorithm computes solution with $14k$ edges.

Tight example for 2ECSS



$L = |V(G)| = OPT = 24k$ (Here $k = 2$.)
$\varphi(G) = 1$
$\pi = 4k = \frac{1}{6}L.$

Algorithm computes solution with $32k - 1$ edges.