Formalizing Theory of Approximating the Traveling-Salesman Problem

Interdisciplinary Project, Mathematics

Lukas Koller

Thursday 29th June, 2023

Traveling-Salesman Problem

The TRAVELING-SALESMAN PROBLEM (TSP) is a well known combinatorial optimization problem.

Definition (TRAVELING-SALESMAN PROBLEM)

Input: undirected graph G with edge weights $c : E(G) \to \mathbb{R}_+$.

Task: Find a Hamiltonian cycle in G with minimum total weight.

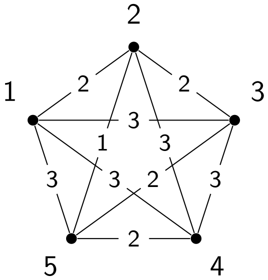
Theorem (15.43, Korte and Vygen, 2018)

The decision problem of TSP *is* NP*-hard.*

$\implies$ We can't solve TSP optimally in polynomial time, unless $\mathrm{P} = \mathrm{NP}$.

Traveling-Salesman Problem

The TRAVELING-SALESMAN PROBLEM (TSP) is a well known combinatorial optimization problem.



Definition (TRAVELING-SALESMAN PROBLEM)

Input: undirected graph G with edge weights $c : E(G) \to \mathbb{R}_+$.

Task: Find a Hamiltonian cycle in G with minimum total weight.

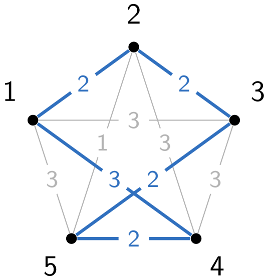
Theorem (15.43, Korte and Vygen, 2018)

The decision problem of TSP *is* NP*-hard.*

$\implies$ We can't solve TSP optimally in polynomial time, unless $\mathrm{P} = \mathrm{NP}$.

Traveling-Salesman Problem

The $\textsc{Traveling-Salesman Problem}$ (TSP) is a well known combinatorial optimization problem.



Definition ($\textsc{Traveling-Salesman Problem}$)

Input: undirected graph G with edge weights $c : E(G) \to \mathbb{R}_+$.

Task: Find a Hamiltonian cycle in G with minimum total weight.

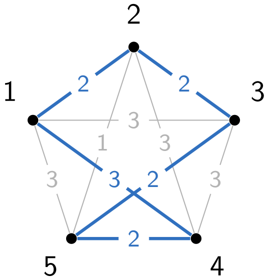
Theorem (15.43, Korte and Vygen, 2018)

The decision problem of TSP *is* NP*-hard.*

$\implies$ We can't solve TSP optimally in polynomial time, unless $\mathrm{P} = \mathrm{NP}$.

Traveling-Salesman Problem

The TRAVELING-SALESMAN PROBLEM (TSP) is a well known combinatorial optimization problem.



Definition (TRAVELING-SALESMAN PROBLEM)

Input: undirected graph G with edge weights $c : E(G) \to \mathbb{R}_+$.

Task: Find a Hamiltonian cycle in G with minimum total weight.

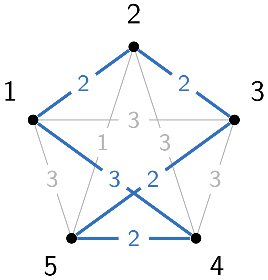
Theorem (15.43, Korte and Vygen, 2018)

The decision problem of TSP *is* NP*-hard.*

$\implies$ We can't solve TSP optimally in polynomial time, unless $\mathrm{P} = \mathrm{NP}$.

Traveling-Salesman Problem

The TRAVELING-SALESMAN PROBLEM (TSP) is a well known combinatorial optimization problem.



Definition (TRAVELING-SALESMAN PROBLEM)

Input: undirected graph G with edge weights $c : E(G) \to \mathbb{R}_+$.

Task: Find a Hamiltonian cycle in G with minimum total weight.

Theorem (15.43, Korte and Vygen, 2018)

The decision problem of TSP *is* NP*-hard.*

$\implies$ We can't solve TSP optimally in polynomial time, unless $\mathrm{P} = \mathrm{NP}$.

Approximation Algorithms for TSP

- An *approximation algorithm* is a polynomial-time algorithm for an optimization problem that returns an approximate solution.
- The *approximation ratio* bounds the distance between the approximate solution and the optimal solution.

Negative result for TSP:

Theorem (Sahni and Gonzalez, 1976)

There is no constant-factor approx. algorithm for TSP, *unless* $\mathrm{P} = \mathrm{NP}$.

We simplify the TSP to the $\textsc{Metric}$ TSP by assuming:

(i) the graph G is complete and

(ii) the edge weights c satisfy the triangle-inequality.

Theorem (Sahni and Gonzalez, 1976)

The decision problem of $\textsc{Metric}$ TSP *is* NP*-hard.*

Approximation Algorithms for TSP

- An *approximation algorithm* is a polynomial-time algorithm for an optimization problem that returns an approximate solution.
- The *approximation ratio* bounds the distance between the approximate solution and the optimal solution.

Negative result for TSP:

Theorem (Sahni and Gonzalez, 1976)

There is no constant-factor approx. algorithm for TSP, *unless* $\mathrm{P} = \mathrm{NP}$.

We simplify the TSP to the $\textsc{Metric}$ TSP by assuming:

(i) the graph G is complete and

(ii) the edge weights c satisfy the triangle-inequality.

Theorem (Sahni and Gonzalez, 1976)

The decision problem of $\textsc{Metric}$ TSP *is* NP*-hard.*

Approximation Algorithms for TSP

- An *approximation algorithm* is a polynomial-time algorithm for an optimization problem that returns an approximate solution.
- The *approximation ratio* bounds the distance between the approximate solution and the optimal solution.

Negative result for TSP:

Theorem (Sahni and Gonzalez, 1976)

There is no constant-factor approx. algorithm for TSP*, unless* $\mathrm{P} = \mathrm{NP}$.

We simplify the TSP to the $\textsc{Metric}$ TSP by assuming:

(i) the graph G is complete and
(ii) the edge weights c satisfy the triangle-inequality.

Theorem (Sahni and Gonzalez, 1976)

The decision problem of $\textsc{Metric}$ TSP *is* NP*-hard.*

Approximation Algorithms for TSP

- An *approximation algorithm* is a polynomial-time algorithm for an optimization problem that returns an approximate solution.
- The *approximation ratio* bounds the distance between the approximate solution and the optimal solution.

Negative result for TSP:

Theorem (Sahni and Gonzalez, 1976)

There is no constant-factor approx. algorithm for TSP, *unless* $\mathrm{P} = \mathrm{NP}$.

We simplify the TSP to the $\textsc{Metric}$ TSP by assuming:

(i) the graph G is complete and
(ii) the edge weights c satisfy the triangle-inequality.

Theorem (Sahni and Gonzalez, 1976)

The decision problem of $\textsc{Metric}$ TSP *is* NP-*hard.*

Overview

(i) I have formalized a proof for the equivalence of the Maximum Matching Problem and the Minimum Matching-Perfect Problem.

I have formalized parts of the section 21.1, *Approximation Algorithms for the TSP* from (Korte and Vygen, 2018).

(ii) I have formally verified two approx. algorithms for Metric TSP.
- DoubleTree and Christofides-Serdyukov algorithm

(iii) I have formalized a L-reduction from VCP4[1] to Metric TSP.
- VCP4 is MaxSNP-hard (Thm. 16.46, Korte and Vygen, 2018) $\implies$ Metric TSP is MaxSNP-hard.

[1] Minimum Vertex Cover Problem with maximum degree of 4

Overview

(i) I have formalized a proof for the equivalence of the MAXIMUM MATCHING PROBLEM and the MINIMUM MATCHING-PERFECT PROBLEM.

I have formalized parts of the section 21.1, *Approximation Algorithms for the TSP* from (Korte and Vygen, 2018).

(ii) I have formally verified two approx. algorithms for METRIC TSP.
- DOUBLETREE and CHRISTOFIDES-SERDYUKOV algorithm

(iii) I have formalized a L-reduction from VCP4[1] to METRIC TSP.
- VCP4 is MAXSNP-hard (Thm. 16.46, Korte and Vygen, 2018) $\implies$ METRIC TSP is MAXSNP-hard.

[1] MINIMUM VERTEX COVER PROBLEM with maximum degree of 4

Overview

(i) I have formalized a proof for the equivalence of the Maximum Matching Problem and the Minimum Matching-Perfect Problem.

I have formalized parts of the section 21.1, *Approximation Algorithms for the TSP* from (Korte and Vygen, 2018).

(ii) I have formally verified two approx. algorithms for Metric TSP.
- DoubleTree and Christofides-Serdyukov algorithm

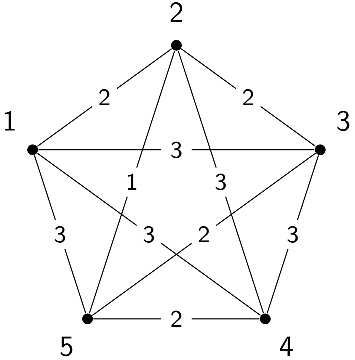
(iii) I have formalized a L-reduction from VCP4[1] to Metric TSP.
- VCP4 is MaxSNP-hard (Thm. 16.46, Korte and Vygen, 2018)
  $\implies$ Metric TSP is MaxSNP-hard.

[1] Minimum Vertex Cover Problem with maximum degree of 4

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

CHRISTOFIDES-SERDYUKOV *is a* $\frac{3}{2}$*-approx. algorithm for* METRIC TSP.



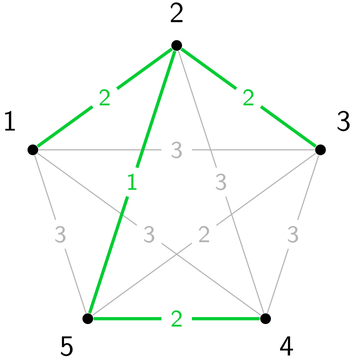
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

CHRISTOFIDES-SERDYUKOV *is a* $\frac{3}{2}$*-approx. algorithm for* METRIC TSP.



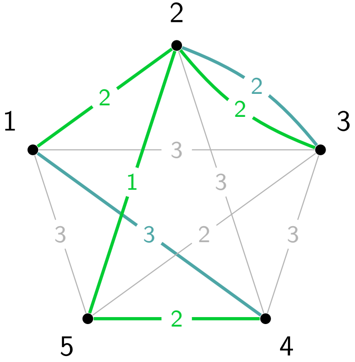
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

CHRISTOFIDES-SERDYUKOV *is a* $\frac{3}{2}$*-approx. algorithm for* METRIC TSP*.*



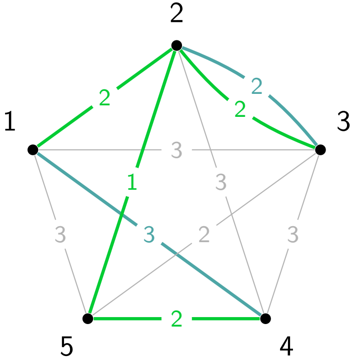
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

Christofides-Serdyukov *is a* $\frac{3}{2}$*-approx. algorithm for* Metric TSP.



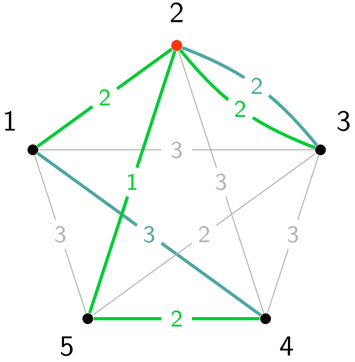
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

CHRISTOFIDES-SERDYUKOV *is a* $\frac{3}{2}$*-approx. algorithm for* METRIC TSP.

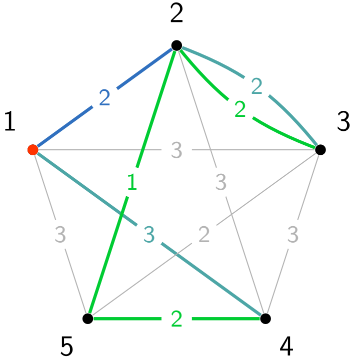


Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)
Christofides-Serdyukov *is a* $\frac{3}{2}$*-approx. algorithm for* Metric TSP.

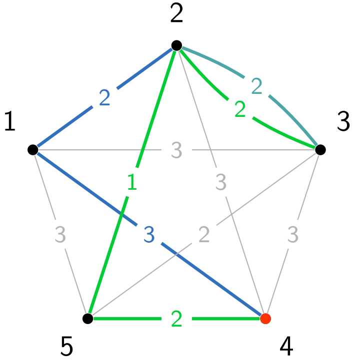


Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)
CHRISTOFIDES-SERDYUKOV *is a* $\frac{3}{2}$*-approx. algorithm for* METRIC TSP.



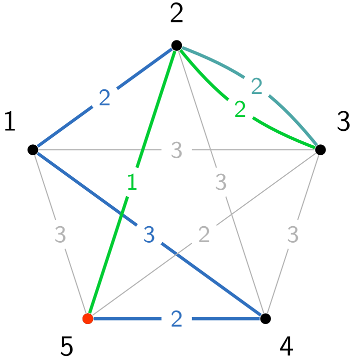
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

Christofides-Serdyukov *is a* $\frac{3}{2}$*-approx. algorithm for* Metric TSP*.*



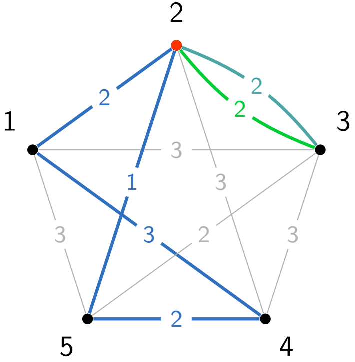
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

Christofides-Serdyukov *is a* $\frac{3}{2}$*-approx. algorithm for* Metric TSP.



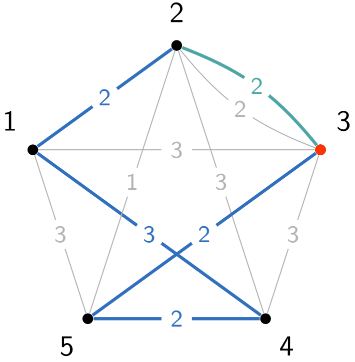
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,2,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

CHRISTOFIDES-SERDYUKOV *is a $\frac{3}{2}$-approx. algorithm for* METRIC TSP.



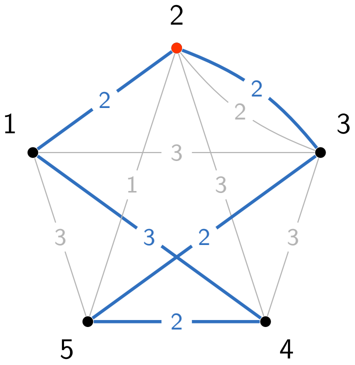
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,~~2~~,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

Christofides-Serdyukov *is a* $\frac{3}{2}$*-approx. algorithm for* Metric TSP.



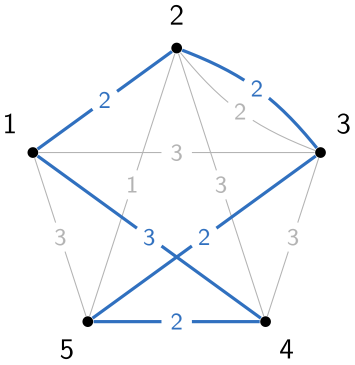
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,~~2~~,3,2

Christofides-Serdyukov Algorithm (Christofides, 1976; Serdyukov, 1978)

Theorem (21.5, Korte and Vygen, 2018)

CHRISTOFIDES-SERDYUKOV *is a* $\frac{3}{2}$*-approx. algorithm for* METRIC TSP*.*



Algorithm:

1. compute minimum spanning-tree (MST)
2. compute minimum perfect-matching between vertices with odd degree in MST
3. compute Eulerian tour on MST + matching, e.g. 2,1,4,5,2,3,2
4. shortcut Eulerian tour, e.g. 2,1,4,5,~~2~~,3,2

Formalization of the Christofides-Serdyukov Algorithm

I have formalized and verified the Christofides-Serdyukov algorithm in Isabelle/HOL (Nipkow et al., 2002).

- I build on an exiting graph formalization by Abdulaziz (2020) for Berge's theorem (Berge, 1957).
- The Christofides-Serdyukov algorithm is formalized with the assumption of the existence of necessary algorithms.
  - minimum spanning-tree, minimum perfect-matching, and Eulerian tour.
- The feasibility and the approximation ratio of the Christofides-Serdyukov algorithm are proven.
- The definition of the Christofides-Serdyukov algorithm is refined to a `WHILE`-program using Hoare-Logic.

L-Reduction (Linear Reduction)

An L-reduction is a special type of reduction that is used to prove the $\textsc{MaxSNP}$-hardness of optimization problems.

Definition (L-Reduction, Papadimitriou and Yannakakis, 1991)

Let A and B be optimization problems with cost functions c_A and c_B. An L-Reduction from A to B is a pair of function f and g (computable in polynomial time) with $\alpha, \beta > 0$ s.t. for any instance x_A of A

(i) $f(x_A)$ is an instance of B and

$$\mathrm{OPT}(f(x_A)) \leq \alpha\, \mathrm{OPT}(x_A),$$

(ii) for any feasible solution y_B of $f(x_A)$, $g(x_A, y_B)$ is a feasible solution of x_A s.t.

$$|c_A(x_A, g(x_A, y_B)) - \mathrm{OPT}(x_A)| \leq \beta |c_B(f(x_A), y) - \mathrm{OPT}(f(x_A))|.$$

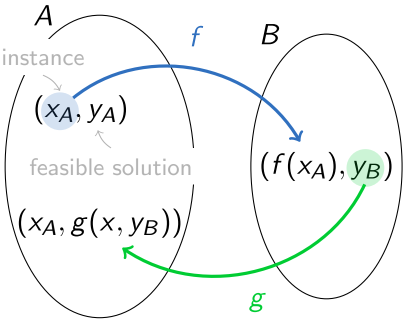
$\mathrm{OPT}(\cdot)$ denotes the cost of the optimal solution.

L-Reduction (Linear Reduction)

An L-reduction is a special type of reduction that is used to prove the MaxSNP-hardness of optimization problems.

Definition (L-Reduction, Papadimitriou and Yannakakis, 1991)

Let A and B be optimization problems with cost functions c_A and c_B. An L-Reduction from A to B is a pair of function f and g (computable in polynomial time) with $\alpha, \beta > 0$ s.t. for any instance x_A of A



(i) $f(x_A)$ is an instance of B and

$$\mathrm{OPT}(f(x_A)) \leq \alpha\, \mathrm{OPT}(x_A),$$

(ii) for any feasible solution y_B of $f(x_A)$, $g(x_A, y_B)$ is a feasible solution of x_A s.t.

$$|c_A(x_A, g(x_A, y_B)) - \mathrm{OPT}(x_A)| \leq \beta |c_B(f(x_A), y) - \mathrm{OPT}(f(x_A))|.$$

OPT(·) denotes the cost of the optimal solution.

L-Reduction for Metric TSP (Korte and Vygen, 2018) I

VCP4[1]

Input: undir. graph G with maximum degree of 4.

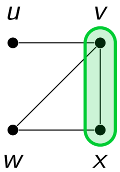
Task: Find a vertex cover of G with minimum cardiality.

METRIC TSP

Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.

*Not all edges are shown.



$\xmapsto{f}$

[1] MINIMUM VERTEX COVER PROBLEM with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) I

VCP4[1]

Input: undir. graph G with maximum degree of 4.

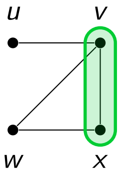
Task: Find a vertex cover of G with minimum cardiality.

Metric TSP

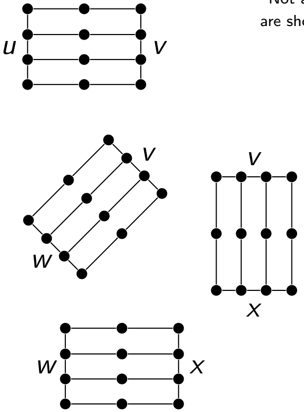
Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.

*Not all edges are shown.



$\overset{f}{\longmapsto}$



[1] Minimum Vertex Cover Problem with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) I

VCP4[1]

Input: undir. graph G with maximum degree of 4.

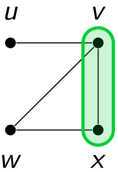
Task: Find a vertex cover of G with minimum cardiality.

Metric TSP

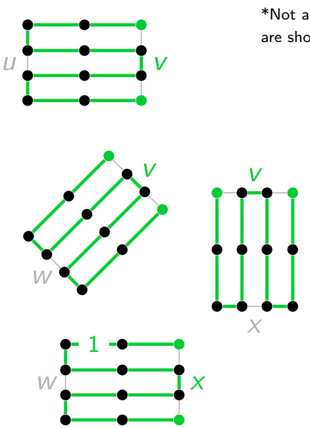
Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.

*Not all edges are shown.



$$\overset{f}{\longmapsto}$$



[1] Minimum Vertex Cover Problem with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) I

VCP4[1]

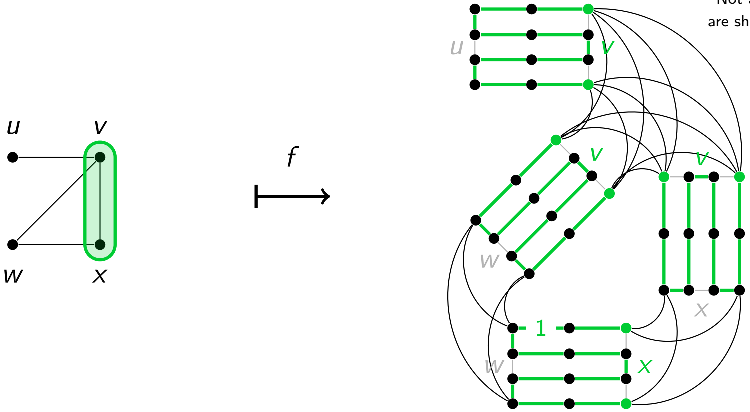
Input: undir. graph G with maximum degree of 4.

Task: Find a vertex cover of G with minimum cardiality.

Metric TSP

Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.



*Not all edges are shown.

[1] Minimum Vertex Cover Problem with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) I

VCP4[1]

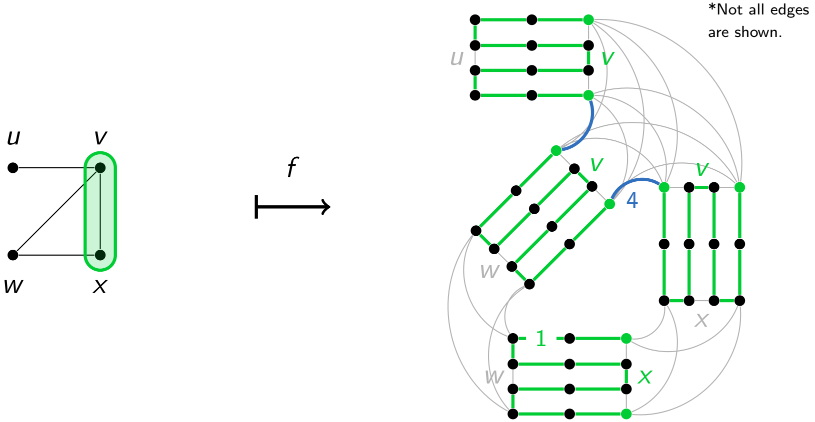
Input: undir. graph G with maximum degree of 4.

Task: Find a vertex cover of G with minimum cardiality.

Metric TSP

Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.



[1] Minimum Vertex Cover Problem with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) I

VCP4[1]

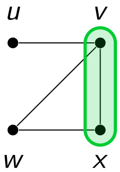
Input: undir. graph G with maximum degree of 4.

Task: Find a vertex cover of G with minimum cardiality.

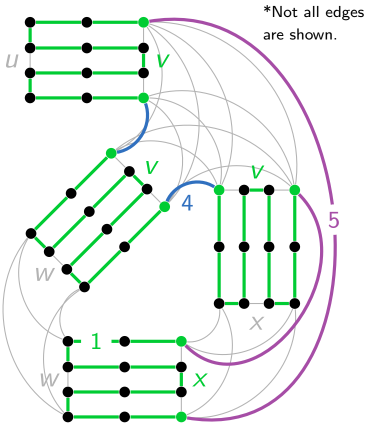
Metric TSP

Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.



$\xmapsto{f}$



[1] Minimum Vertex Cover Problem with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) II

VCP4[1]

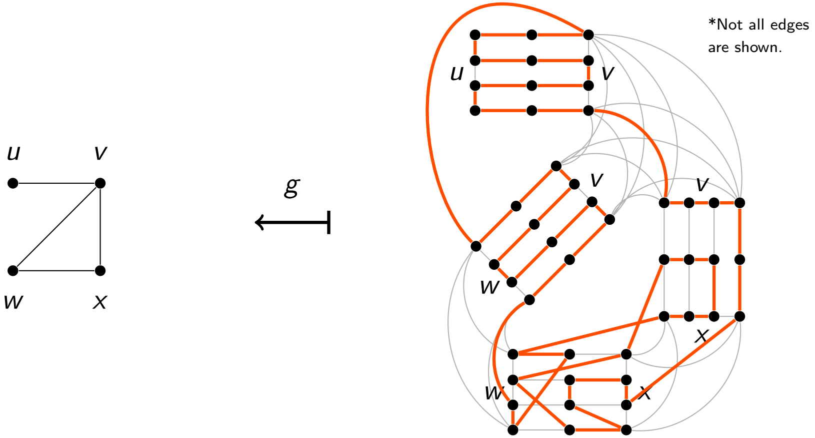
Input: undir. graph G with maximum degree of 4.

Task: Find a vertex cover of G with minimum cardiality.

Metric TSP

Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.



[1] Minimum Vertex Cover Problem with maximum degree of 4

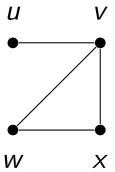
L-Reduction for Metric TSP (Korte and Vygen, 2018) II

VCP4[1]

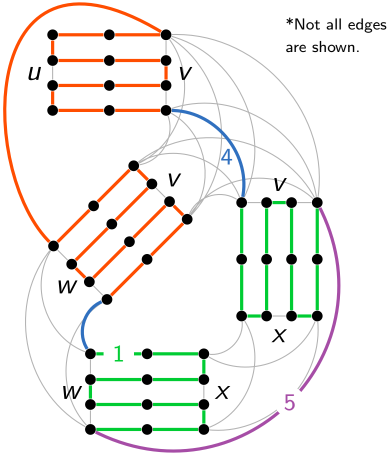
Input:	undir. graph G with maximum degree of 4.
Task:	Find a vertex cover of G with minimum cardiality.

Metric TSP

Input:	undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.
Task:	Find a Hamiltonian cycle in G with minimum total weight.



$\overset{g}{\longleftarrow\!\!\!|}$



[1] Minimum Vertex Cover Problem with maximum degree of 4

L-Reduction for Metric TSP (Korte and Vygen, 2018) II

VCP4[1]

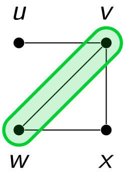
Input: undir. graph G with maximum degree of 4.

Task: Find a vertex cover of G with minimum cardiality.

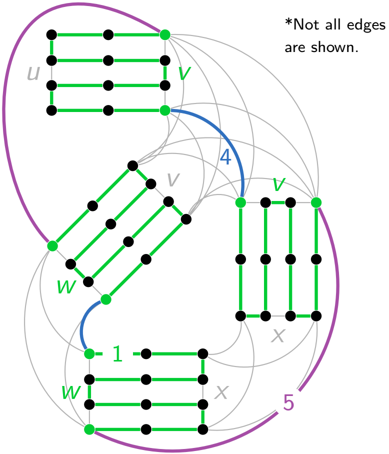
Metric TSP

Input: undir. & compl. graph G with edge weights $c : E(G) \to \mathbb{R}_+$ that satisfy the tri.-inequality.

Task: Find a Hamiltonian cycle in G with minimum total weight.



$\xleftarrow{g}$



[1] Minimum Vertex Cover Problem with maximum degree of 4

Formalization of L-Reduction for Metric TSP

I have formalized the L-reduction in Isabelle/HOL.

- A locale provides an abstract executable graph representation based on an adjacency map.
- The locale assumes `fold`-functions to do computation on the graphs.
  - This approach is adapted from (Abdulaziz, 2022).
- Abstract versions of the reduction functions f and g are defined using the `fold`-functions.
- The feasibility of the function f and g and the linear inequalities are proven.
- The locale is instantiated with concrete implementations to obtain executable versions of the functions f and g.

The L-reduction in (Thm. 21.6, Korte and Vygen, 2018) is incomplete.

(i) The proof for the linear inequalities fails if the optimal vertex cover has a cardinality of 1.

(ii) Not all cases are covered when constructing a vertex cover from a Hamiltonian cycle.

Formalization of L-Reduction for Metric TSP

I have formalized the L-reduction in Isabelle/HOL.

- A locale provides an abstract executable graph representation based on an adjacency map.
- The locale assumes `fold`-functions to do computation on the graphs.
  - This approach is adapted from (Abdulaziz, 2022).
- Abstract versions of the reduction functions f and g are defined using the `fold`-functions.
- The feasibility of the function f and g and the linear inequalities are proven.
- The locale is instantiated with concrete implementations to obtain executable versions of the functions f and g.

The L-reduction in (Thm. 21.6, Korte and Vygen, 2018) is incomplete.

(i) The proof for the linear inequalities fails if the optimal vertex cover has a cardinality of 1.

(ii) Not all cases are covered when constructing a vertex cover from a Hamiltonian cycle.

Conclusion and Future Work

- I have formally verified two approx. algorithms for METRIC TSP.
- I have formalized a L-reduction from VCP4[1] to METRIC TSP, which proves the MAXSNP-hardness of METRIC TSP.

- Prove the existence of the necessary algorithms for the CHRISTOFIDES-SERDYUKOV algorithm.
  - Write an adaptor to the existing formalization of Prim's algorithm (Lammich and Nipkow, 2019).
  - Formalize and verify algorithms for minimum perfect-matching and Eulerian tour (Sec. 11.2 & 2.4, Korte and Vygen, 2018).
- Formalize an encoding of multigraphs with simple graphs.
- Prove the polynomial running time of reduction functions f and g.
  - Refine the functions f and g to `IMP-` and prove their running time.

[1] MINIMUM VERTEX COVER PROBLEM with maximum degree of 4

Conclusion and Future Work

- I have formally verified two approx. algorithms for METRIC TSP.
- I have formalized a L-reduction from VCP4[1] to METRIC TSP, which proves the MAXSNP-hardness of METRIC TSP.

- Prove the existence of the necessary algorithms for the CHRISTOFIDES-SERDYUKOV algorithm.
  - Write an adaptor to the existing formalization of Prim's algorithm (Lammich and Nipkow, 2019).
  - Formalize and verify algorithms for minimum perfect-matching and Eulerian tour (Sec. 11.2 & 2.4, Korte and Vygen, 2018).
- Formalize an encoding of multigraphs with simple graphs.
- Prove the polynomial running time of reduction functions f and g.
  - Refine the functions f and g to `IMP-` and prove their running time.

[1] MINIMUM VERTEX COVER PROBLEM with maximum degree of 4

References I

Abdulaziz, M. (2020). *Formalization of Berge's Theorem*. GitHub Repository. https://github.com/wimmers/archive-of-graph-formalizations/blob/master/Undirected_Graphs/Berge.thy.

Abdulaziz, M. (2022). *Formalization Depth-First Search Algorithm*. GitHub Repository. https://github.com/cmadlener/archive-of-graph-formalizations/blob/master/Pair_Graph/DFS.thy.

Berge, C. (1957). "Two Theorems in Graph Theory". In: *Proceedings of the National Academy of Sciences of the United States of America* 43.9, pp. 842–844. ISSN: 00278424.

Christofides, N. (1976). *Worst-case analysis of a new heuristic for the travelling salesman problem*. Technical Report 388. Graduate School of Industrial Administration, Carnegie Mellon University.

Korte, B. and J. Vygen (2018). *Combinatorial Optimization: Theory and Algorithms*. 6th. Springer Berlin, Heidelberg. ISBN: 978-3-662-56039-6.

References II

Lammich, P. and T. Nipkow (June 2019). "Purely Functional, Simple, and Efficient Implementation of Prim and Dijkstra". In: *Archive of Formal Proofs*. https://isa-afp.org/entries/Prim_Dijkstra_Simple.html, Formal proof development. ISSN: 2150-914x.

Nipkow, T., L. C. Paulson, and M. Wenzel (2002). *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*. Lecture Notes in Computer Science. Springer.

Papadimitriou, C. H. and M. Yannakakis (1991). "Optimization, approximation, and complexity classes". In: *Journal of Computer and System Sciences* 43.3, pp. 425–440.

Sahni, S. and T. Gonzalez (July 1976). "P-Complete Approximation Problems". In: *Journal of the ACM* 23.3, pp. 555–565. DOI: 10.1145/321958.321975.

Serdyukov, A. I. (1978). "On some extremal tours in graphs (translated from Russian)". In: *Upravlyaemye Sistemy (in Russian)* 17, pp. 76–79.

Appendix – Equiv. Max. Matching and Min. Matching

Let G be an undirected graph with edge weights c.

(i) MaxMatch[1] solves MinMatch[2]. Assume f solves MaxMatch.

$$c'(e) := 1 + -c(e) + \sum_{e' \in E(G)} c(e')$$

- If matching $M := f(G, c')$ is perfect return M otherwise None.

(ii) MinMatch solves MaxMatch. Assume f solves MinMatch.

$$E_1 := \{e \times \{1\} \mid e \in E(G)\} \qquad E_2 := \{e \times \{2\} \mid e \in E(G)\}$$

$$E_V := \{\{(v,1),(v,2)\} \mid v \in V(G)\}$$

$$H := (V(G) \times \{1,2\}, E_1 \cup E_2 \cup E_V) \quad c'(e) := \begin{cases} 1 & \text{if } e \in E_V \\ -c(e) & \text{otherwise} \end{cases}$$

- Return matching $M := f(H, c') \cap E_1$.

[1] Maximum Matching Problem

[2] Minimum Matching-Perfect Problem

Appendix – Approximation Algorithms for Metric TSP

Let (G, c) be an instance of METRIC TSP.

Lemma (21.3, Korte and Vygen, 2018)

Given a connected Eulerian graph G' that spans $V(G)$, we can compute (in polynomial time) a Hamiltonian cycle for G with total weight of at most $c(E(G'))$.

Proof Idea.

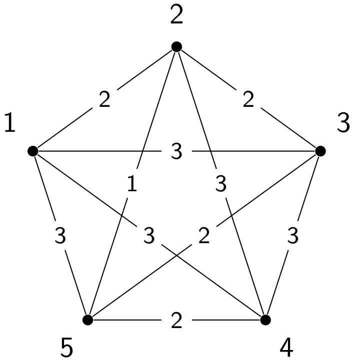
1. Compute a Eulerian tour in G'.
2. Remove duplicate vertices ("shortcutting").

$\Longrightarrow$ We only need to find Eulerian graph that spans $V(G)$ with the smallest total weight!

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



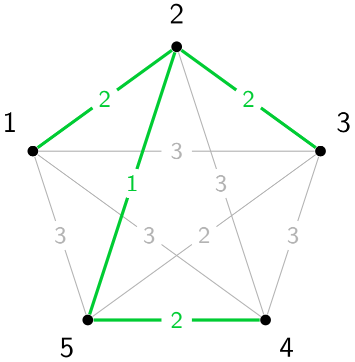
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DoubleTree *is a 2-approximation algorithm for* Metric TSP.



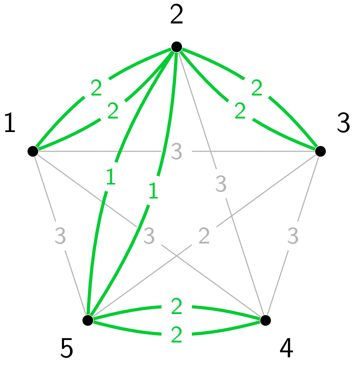
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



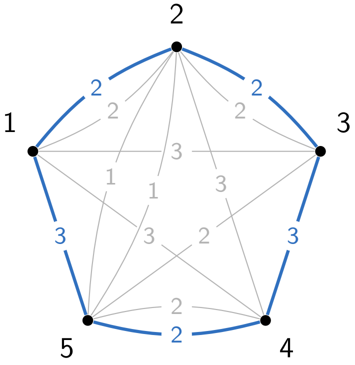
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DoubleTree *is a 2-approximation algorithm for* Metric TSP.



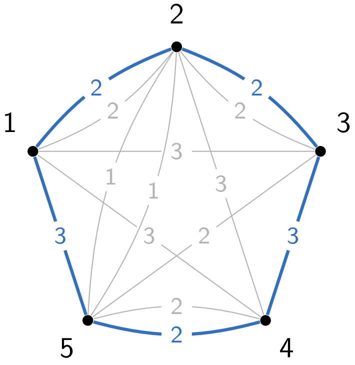
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



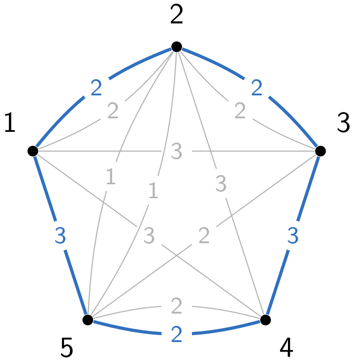
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



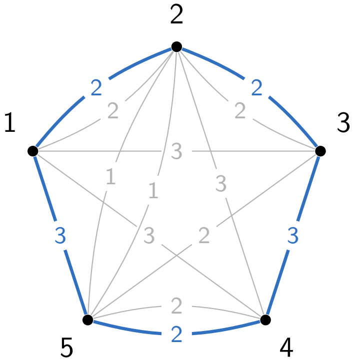
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



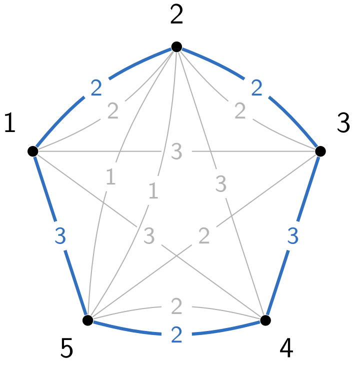
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



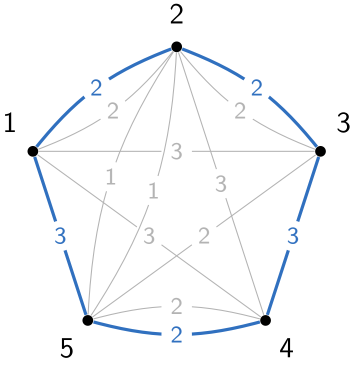
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



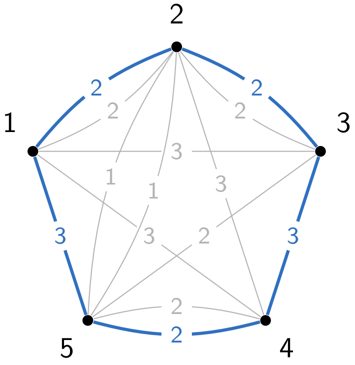
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DoubleTree *is a 2-approximation algorithm for* Metric TSP.



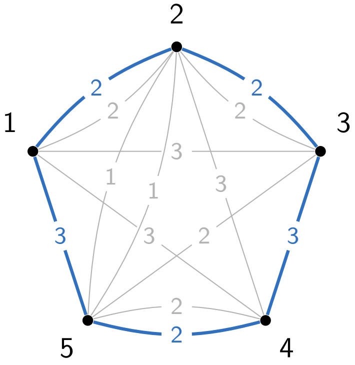
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DOUBLETREE *is a 2-approximation algorithm for* METRIC TSP.



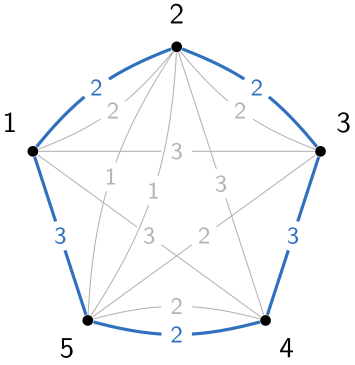
Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2

Appendix – DoubleTree Algorithm

Theorem (21.4, Korte and Vygen, 2018)

DoubleTree *is a 2-approximation algorithm for* Metric TSP.



Algorithm:

1. compute minimum spanning-tree (MST)
2. compute Eulerian tour on doubled MST, e.g. 2,1,2,5,4,5,2,3,2
3. shortcut Eulerian tour, e.g. 2,1,~~2~~,5,4,~~5~~,~~2~~,3,2