

3 Steiner Tree and TSP

In this chapter, we will present constant factor algorithms for two fundamental problems, metric Steiner tree and metric TSP. The reasons for considering the metric case of these problems are quite different. For Steiner tree, this is the core of the problem – the rest of the problem reduces to this case. For TSP, without this restriction, the problem admits no approximation factor, assuming $\mathbf{P} \neq \mathbf{NP}$. The algorithms, and their analyses, are similar in spirit, which is the reason for presenting these problems together.

3.1 Metric Steiner tree

The Steiner tree problem was defined by Gauss in a letter he wrote to Schumacher (reproduced on the cover of this book). Today, this problem occupies a central place in the field of approximation algorithms. The problem has a wide range of applications, all the way from finding minimum length interconnection of terminals in VLSI design to constructing phylogeny trees in computational biology. This problem and its generalizations will be studied extensively in this book, see Chapters 22 and 23.

Problem 3.1 (Steiner tree) Given an undirected graph $G = (V, E)$ with nonnegative edge costs and whose vertices are partitioned into two sets, *required* and *Steiner*, find a minimum cost tree in G that contains all the required vertices and any subset of the Steiner vertices.

We will first show that the core of this problem lies in its restriction to instances in which the edge costs satisfy *the triangle inequality*, i.e., G is a complete undirected graph, and for any three vertices u, v , and w , $\text{cost}(u, v) \leq \text{cost}(u, w) + \text{cost}(v, w)$. Let us call this restriction the *metric Steiner tree problem*.

Theorem 3.2 *There is an approximation factor preserving reduction from the Steiner tree problem to the metric Steiner tree problem.*

Proof: We will transform, in polynomial time, an instance I of the Steiner tree problem, consisting of graph $G = (V, E)$, to an instance I' of the metric Steiner tree problem as follows. Let G' be the complete undirected graph on

vertex set V . Define the cost of edge (u, v) in G' to be the cost of a shortest u - v path in G . G' is called the *metric closure* of G . The partition of V into required and Steiner vertices in I' is the same as in I .

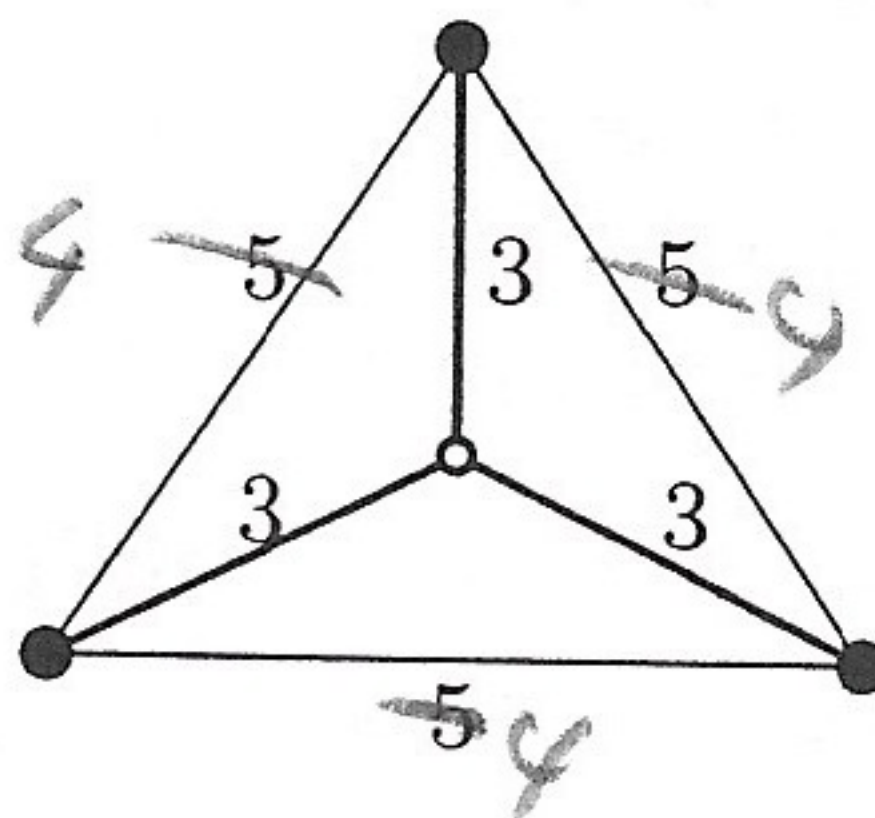
For any edge $(u, v) \in E$, its cost in G' is no more than its cost in G . Therefore, the cost of an optimal solution in I' does not exceed the cost of an optimal solution in I .

Next, given a Steiner tree T' in I' , we will show how to obtain, in polynomial time, a Steiner tree T in I of at most the same cost. The cost of an edge (u, v) in G' corresponds to the cost of a path in G . Replace each edge of T' by the corresponding path to obtain a subgraph of G . Clearly, in this subgraph, all the required vertices are connected. However, this subgraph may, in general, contain cycles. If so, remove edges to obtain tree T . This completes the approximation factor preserving reduction. \square

As a consequence of Theorem 3.2, any approximation factor established for the metric Steiner tree problem carries over to the entire Steiner tree problem.

3.1.1 MST-based algorithm

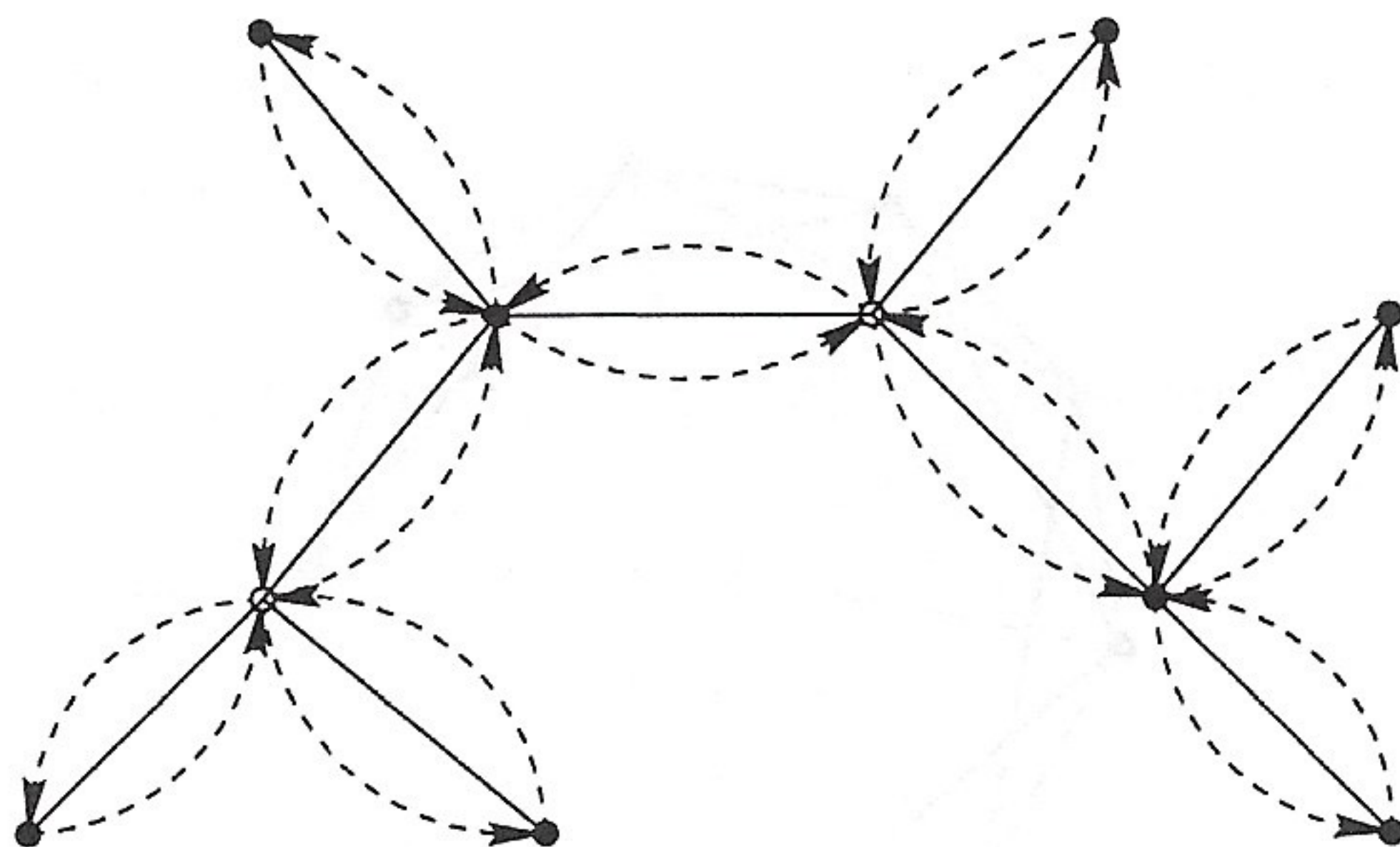
Let R denote the set of required vertices. Clearly, a minimum spanning tree (MST) on R is a feasible solution for this problem. Since the problem of finding an MST is in \mathbf{P} and the metric Steiner tree problem is \mathbf{NP} -hard, we cannot expect the MST on R to always give an optimal Steiner tree; below is an example in which the MST is strictly costlier.



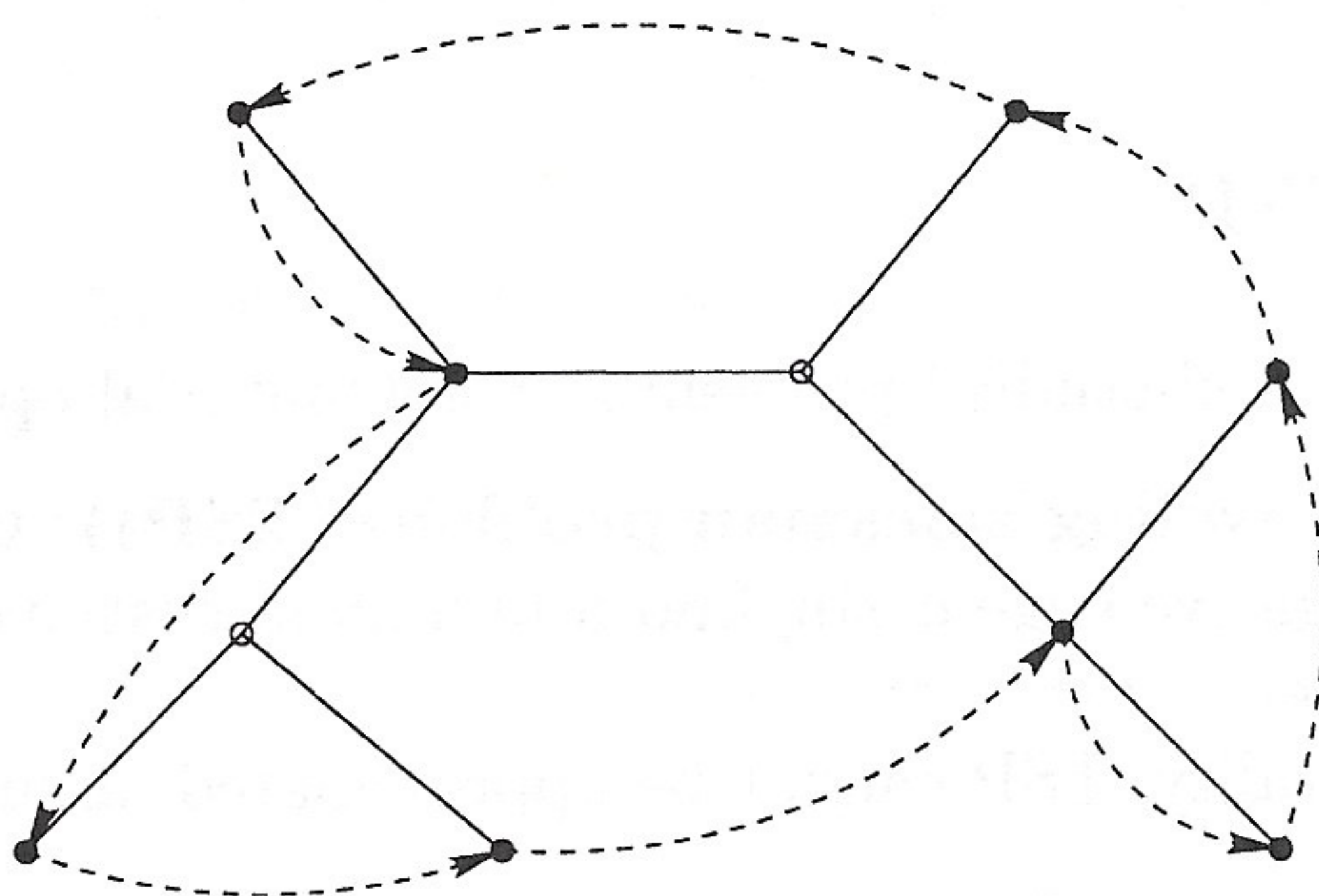
Even so, an MST on R is not much more costly than an optimal Steiner tree:

Theorem 3.3 *The cost of an MST on R is within $2 \cdot \text{OPT}$.*

Proof: Consider a Steiner tree of cost OPT . By doubling its edges we obtain an Eulerian graph connecting all vertices of R and, possibly, some Steiner vertices. Find an Euler tour of this graph, for example by traversing the edges in DFS (depth first search) order:



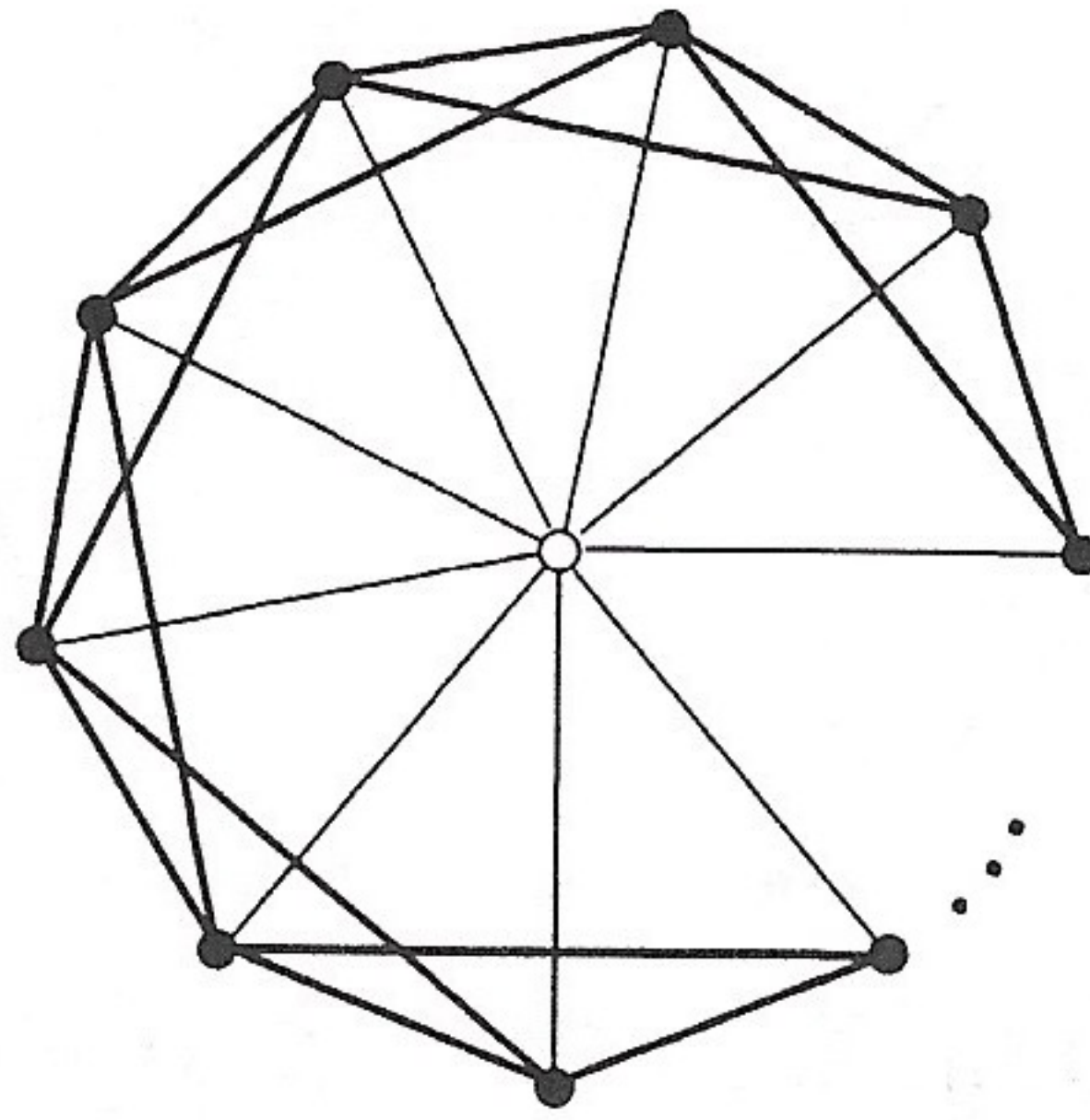
The cost of this Euler tour is $2 \cdot \text{OPT}$. Next obtain a Hamiltonian cycle on the vertices of R by traversing the Euler tour and “short-cutting” Steiner vertices and previously visited vertices of R :



Because of triangle inequality, the shortcuts do not increase the cost of the tour. If we delete one edge of this Hamiltonian cycle, we obtain a path that spans R and has cost at most $2 \cdot \text{OPT}$. This path is also a spanning tree on R . Hence, the MST on R has cost at most $2 \cdot \text{OPT}$. \square

Theorem 3.3 gives a straightforward factor 2 algorithm for the metric Steiner tree problem: simply find an MST on the set of required vertices. As in the case of set cover, the “correct” way of viewing this algorithm is in the setting of LP-duality theory. In Chapters 22 and 23 we will see that LP-duality provides the lower bound on which this algorithm is based and also helps solve generalizations of this problem.

Example 3.4 For a tight example, consider a graph with n required vertices and one Steiner vertex. An edge between the Steiner vertex and a required vertex has cost 1, and an edge between two required vertices has cost 2 (not all edges of cost 2 are shown below). In this graph, any MST on R has cost $2(n - 1)$, while $\text{OPT} = n$.



□

3.2 Metric TSP

The following is a well-studied problem in combinatorial optimization.

Problem 3.5 (Traveling salesman problem (TSP)) Given a complete graph with nonnegative edge costs, find a minimum cost cycle visiting every vertex exactly once.

In its full generality, TSP cannot be approximated, assuming $\mathbf{P} \neq \mathbf{NP}$.

Theorem 3.6 *For any polynomial time computable function $\alpha(n)$, TSP cannot be approximated within a factor of $\alpha(n)$, unless $\mathbf{P} = \mathbf{NP}$.*

Proof: Assume, for a contradiction, that there is a factor $\alpha(n)$ polynomial time approximation algorithm, \mathcal{A} , for the general TSP problem. We will show that \mathcal{A} can be used for deciding the Hamiltonian cycle problem (which is \mathbf{NP} -hard) in polynomial time, thus implying $\mathbf{P} = \mathbf{NP}$.

The central idea is a reduction from the Hamiltonian cycle problem to TSP, that transforms a graph G on n vertices to an edge-weighted complete graph G' on n vertices such that

- if G has a Hamiltonian cycle, then the cost of an optimal TSP tour in G' is n , and
- if G does not have a Hamiltonian cycle, then an optimal TSP tour in G' is of cost $> \alpha(n) \cdot n$.

Observe that when run on graph G' , algorithm \mathcal{A} must return a solution of cost $\leq \alpha(n) \cdot n$ in the first case, and a solution of cost $> \alpha(n) \cdot n$ in the second case. Thus, it can be used for deciding whether G contains a Hamiltonian cycle.

The reduction is simple. Assign a weight of 1 to edges of G , and a weight of $\alpha(n) \cdot n$ to nonedges, to obtain G' . Now, if G has a Hamiltonian cycle, then the corresponding tour in G' has cost n . On the other hand, if G has

no Hamiltonian cycle, any tour in G' must use an edge of cost $\alpha(n) \cdot n$, and therefore has cost $> \alpha(n) \cdot n$. \square

Notice that in order to obtain such a strong nonapproximability result, we had to assign edge costs that violate triangle inequality. If we restrict ourselves to graphs in which edge costs satisfy triangle inequality, i.e., consider *metric TSP*, the problem remains NP-complete, but it is no longer hard to approximate.

3.2.1 A simple factor 2 algorithm

We will first present a simple factor 2 algorithm. The lower bound we will use for obtaining this factor is the cost of an MST in G . This is a lower bound because deleting any edge from an optimal solution to TSP gives us a spanning tree of G .

Algorithm 3.7 (Metric TSP – factor 2)

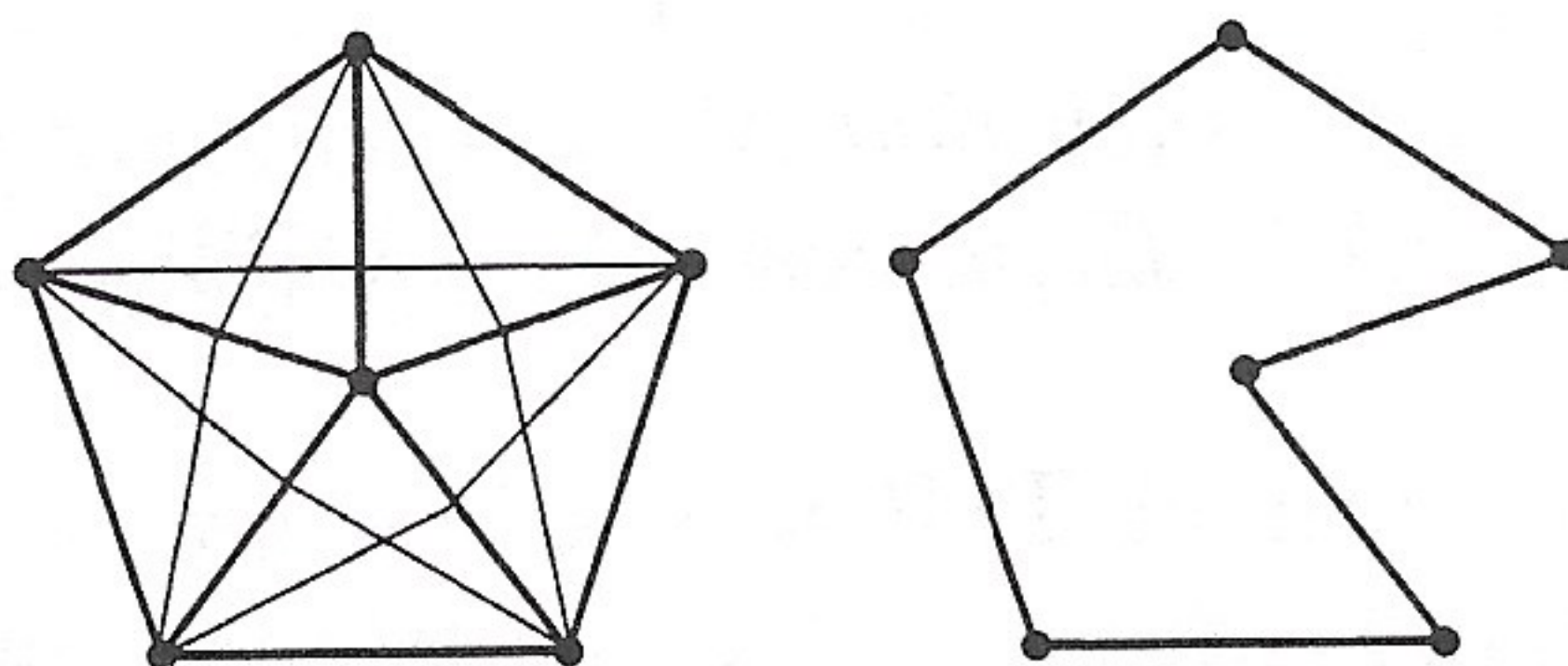
1. Find an MST, T , of G .
2. Double every edge of the MST to obtain an Eulerian graph.
3. Find an Eulerian tour, \mathcal{T} , on this graph.
4. Output the tour that visits vertices of G in the order of their first appearance in \mathcal{T} . Let \mathcal{C} be this tour.

Notice that Step 4 is similar to the “short-cutting” step in Theorem 3.3.

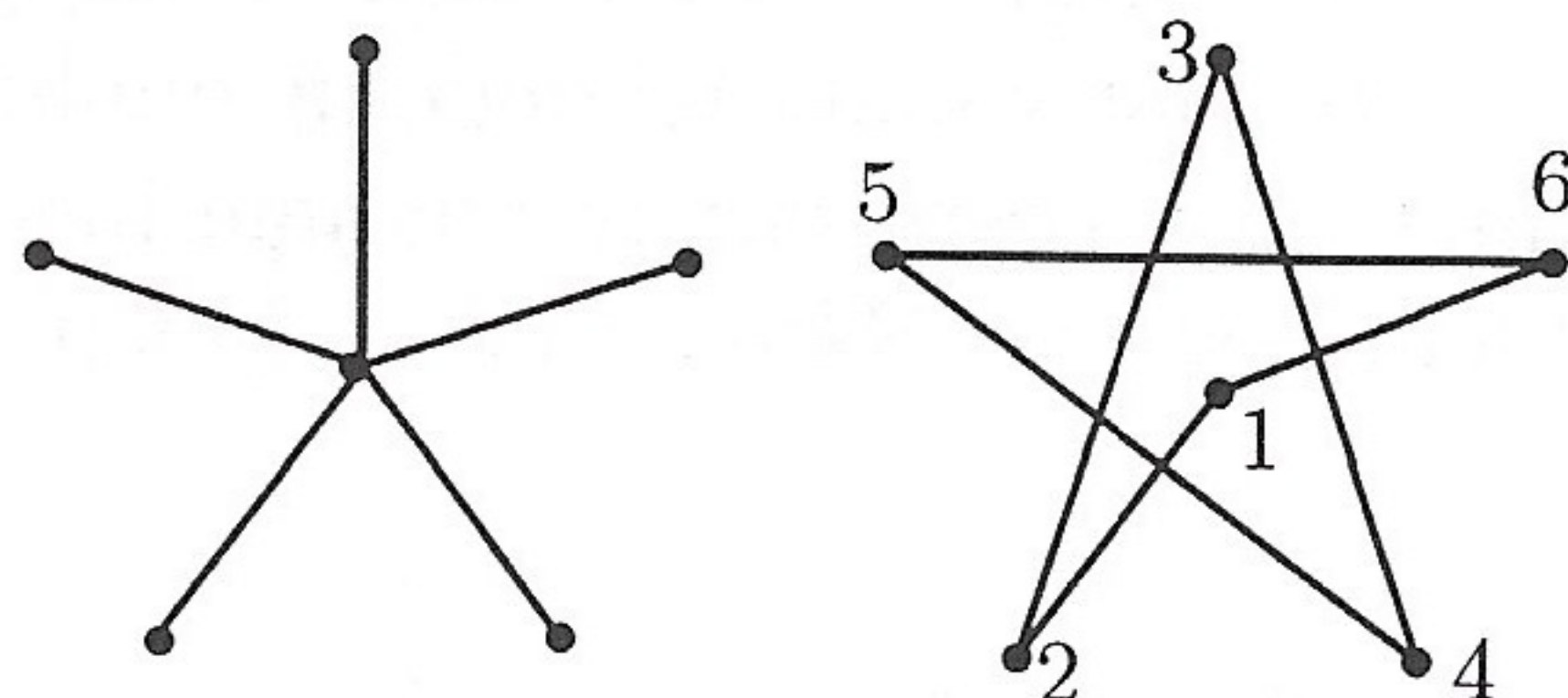
Theorem 3.8 *Algorithm 3.7 is a factor 2 approximation algorithm for metric TSP.*

Proof: As noted above, $\text{cost}(T) \leq \text{OPT}$. Since \mathcal{T} contains each edge of T twice, $\text{cost}(\mathcal{T}) = 2 \cdot \text{cost}(T)$. Because of triangle inequality, after the “short-cutting” step, $\text{cost}(\mathcal{C}) \leq \text{cost}(\mathcal{T})$. Combining these inequalities we get that $\text{cost}(\mathcal{C}) \leq 2 \cdot \text{OPT}$. \square

Example 3.9 A tight example for this algorithm is given by a complete graph on n vertices with edges of cost 1 and 2. We present the graph for $n = 6$ below, where thick edges have cost 1 and remaining edges have cost 2. For arbitrary n the graph has $2n - 2$ edges of cost 1, with these edges forming the union of a star and an $n - 1$ cycle; all remaining edges have cost 2. The optimal TSP tour has cost n , as shown below for $n = 6$:



Suppose that the MST found by the algorithm is the spanning star created by edges of cost 1. Moreover, suppose that the Euler tour constructed in Step 3 visits vertices in order shown below for $n = 6$:



Then the tour obtained after short-cutting contains $n - 2$ edges of cost 2 and has a total cost of $2n - 2$. Asymptotically, this is twice the cost of the optimal TSP tour. \square

3.2.2 Improving the factor to $3/2$

Algorithm 3.7 first finds a low cost Euler tour spanning the vertices of G , and then short-cuts this tour to find a traveling salesman tour. Is there a cheaper Euler tour than that found by doubling an MST? Recall that a graph has an Euler tour iff all its vertices have even degrees. Thus, we only need to be concerned about the vertices of odd degree in the MST. Let V' denote this set of vertices. $|V'|$ must be even since the sum of degrees of all vertices in the MST is even. Now, if we add to the MST a minimum cost perfect matching on V' , every vertex will have an even degree, and we get an Eulerian graph. With this modification, the algorithm achieves an approximation guarantee of $3/2$.

Algorithm 3.10 (Metric TSP – factor $3/2$)

1. Find an MST of G , say T .
2. Compute a minimum cost perfect matching, M , on the set of odd-degree vertices of T . Add M to T and obtain an Eulerian graph.
3. Find an Euler tour, \mathcal{T} , of this graph.
4. Output the tour that visits vertices of G in order of their first appearance in \mathcal{T} . Let \mathcal{C} be this tour.

Interestingly, the proof of this algorithm is based on a second lower bound on OPT.

Lemma 3.11 *Let $V' \subseteq V$, such that $|V'|$ is even, and let M be a minimum cost perfect matching on V' . Then, $\text{cost}(M) \leq \text{OPT}/2$.*

Proof: Consider an optimal TSP tour of G , say τ . Let τ' be the tour on V' obtained by short-cutting τ . By the triangle inequality, $\text{cost}(\tau') \leq$

$\text{cost}(\tau)$. Now, τ' is the union of two perfect matchings on V' , each consisting of alternate edges of τ' . Thus, the cheaper of these matchings has cost $\leq \text{cost}(\tau')/2 \leq \text{OPT}/2$. Hence the optimal matching also has cost at most $\text{OPT}/2$. \square

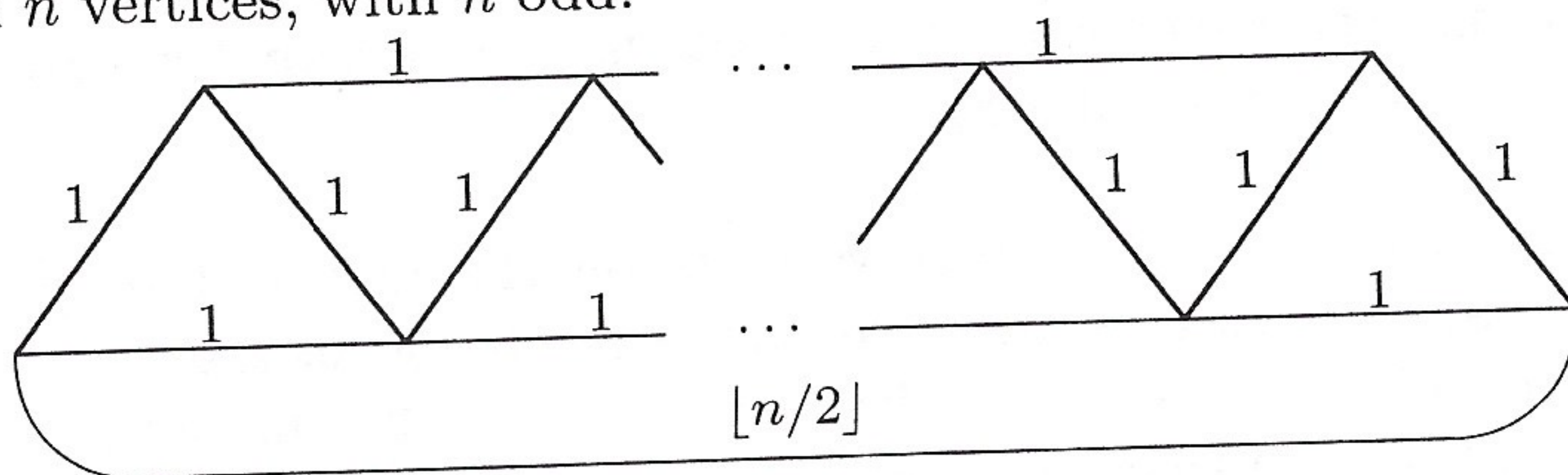
Theorem 3.12 *Algorithm 3.10 achieves an approximation guarantee of $3/2$ for metric TSP.*

Proof: The cost of the Euler tour,

$$\text{cost}(\mathcal{T}) \leq \text{cost}(T) + \text{cost}(M) \leq \text{OPT} + \frac{1}{2}\text{OPT} = \frac{3}{2}\text{OPT},$$

where the second inequality follows by using the two lower bounds on OPT . Using the triangle inequality, $\text{cost}(\mathcal{C}) \leq \text{cost}(\mathcal{T})$, and the theorem follows. \square

Example 3.13 A tight example for this algorithm is given by the following graph on n vertices, with n odd:



Thick edges represent the MST found in step 1. This MST has only two odd degree vertices, and by adding the edge joining them we obtain a traveling salesman tour of cost $(n-1) + \lfloor n/2 \rfloor$. In contrast, the optimal tour has cost n . \square

Finding a better approximation algorithm for metric TSP is currently one of the outstanding open problems in this area. Many researchers have conjectured that an approximation factor of $4/3$ may be achievable.

3.3 Exercises

3.1 The hardness of the Steiner tree problem lies in determining the optimal subset of Steiner vertices that need to be included in the tree. Show this by proving that if this set is provided, then the optimal Steiner tree can be computed in polynomial time.

Hint: Find an MST on the union of this set and the set of required vertices.

3.2 Let $G = (V, E)$ be a graph with nonnegative edge costs. S , the *senders* and R , the *receivers*, are disjoint subsets of V . The problem is to find a