minimum cost subgraph of G that has a path connecting each receiver to a sender (any sender suffices). Partition the instances into two cases: $S \cup R = V$ and $S \cup R \neq V$. Show that these two cases are in **P** and **NP**-hard, respectively. For the second case, give a factor 2 approximation algorithm.
Hint: Add a new vertex which is connected to each sender by a zero cost edge. Consider the new vertex and all receivers as required and the remaining vertices as Steiner, and find a minimum cost Steiner tree.

3.3 Give an approximation factor preserving reduction from the set cover problem to the following problem, thereby showing that it is unlikely to have a better approximation guarantee than $O(\log n)$.

Problem 3.14 (Directed Steiner tree) $G = (V, E)$ is a directed graph with nonnegative edge costs. The vertex set V is partitioned into two sets, *required* and *Steiner*. One of the required vertices, r, is special. The problem is to find a minimum cost tree in G rooted into r that contains all the required vertices and any subset of the Steiner vertices.
Hint: Construct a three layer graph: layer 1 contains a required vertex corresponding to each element, layer 2 contains a Steiner vertex corresponding to each set, and layer 3 contains r.

3.4 (Hoogeveen [137]) Consider variants on the metric TSP problem in which the object is to find a simple path containing all the vertices of the graph. Three different problems arise, depending on the number (0, 1, or 2) of endpoints of the path that are specified. Obtain the following approximation algorithms.

- If zero or one endpoints are specified, obtain a 3/2 factor algorithm.
- If both endpoints are specified, obtain a 5/3 factor algorithm.

Hint: Use the idea behind Algorithm 3.10.

3.5 (Papadimitriou and Yannakakis [227]) Let G be a complete undirected graph in which all edge lengths are either 1 or 2 (clearly, G satisfies the triangle inequality). Give a 4/3 factor algorithm for TSP in this special class of graphs.
Hint: Start by finding a minimum 2-matching in G. A 2-matching is a subset S of edges so that every vertex has exactly 2 edges of S incident at it.

3.6 (Frieze, Galbiati, and Maffioli [95]) Give an $O(\log n)$ factor approximation algorithm for the following problem.

Problem 3.15 (Asymmetric TSP) We are given a directed graph G on vertex set V, with a nonnegative cost specified for edge $(u \to v)$, for each pair $u, v \in V$. The edge costs satisfy *the directed triangle inequality*, i.e., for any three vertices u, v, and w, $\text{cost}(u \to v) \leq \text{cost}(u \to w) + \text{cost}(w \to v)$. The problem is to find a minimum cost cycle visiting every vertex exactly once.

Hint: Use the fact that a minimum cost cycle cover (i.e., disjoint cycles covering all the vertices) can be found in polynomial time. Shrink the cycles and recurse.

3.7 Let $G = (V, E)$ be a graph with edge costs satisfying the triangle inequality, and $V' \subseteq V$ be a set of even cardinality. Prove or disprove: The cost of a minimum cost perfect matching on V' is bounded above by the cost of a minimum cost perfect matching on V.

3.8 Given n points in $\mathbf{R}^2$, define the optimal Euclidean Steiner tree to be a minimum length tree containing all n points and any other subset of points from $\mathbf{R}^2$. Prove that each of the additional points must have degree three, with all three angles being 120°.

3.9 (Rao, Sadayappan, Hwang, and Shor [238]) This exercise develops a factor 2 approximation algorithm for the following problem.

Problem 3.16 (Rectilinear Steiner arborescence) Let $p_1, \ldots, p_n$ be points given in $\mathbf{R}^2$ in the positive quadrant. A path from the origin to point p_i is said to be *monotone* if it consists of segments traversing in the positive x direction or the positive y direction (informally, going right or up). The problem is to find a minimum length tree containing monotone paths from the origin to each of the n points; such a tree is called *rectilinear Steiner arborescence*.

For point p, define x_p and y_p to be its x and y coordinates, and $|p|_1 = |x_p| + |y_p|$. Say that point p *dominates* point q if $x_p \leq x_q$ and $y_p \leq y_q$. For sets of points A and B, we will say that A *dominates* B if for each point $b \in B$, there is a point $a \in A$ such that a dominates b. For points p and q, define $\text{dom}(p, q) = (x, y)$, where $x = \min(x_p, x_q)$ and $y = \min(y_p, y_q)$. If p dominates q, define segments(p, q) to be a monotone path from p to q. Consider the following algorithm.

Algorithm 3.17 (Rectilinear Steiner arborescence)

1. $T \leftarrow \emptyset$.
2. $P \leftarrow \{p_1, \ldots, p_n\} \cup \{(0,0)\}$.
3. while $|P| > 1$ do:
   Pick $p, q = \arg\max_{p,q \in P}(|\text{dom}(p,q)|_1)$.
   $P \leftarrow (P - \{p, q\}) \cup \{\text{dom}(p,q)\}$.
   $T \leftarrow T \cup \text{segments}(\text{dom}(p,q), p) \cup \text{segments}(\text{dom}(p,q), q)$.
4. Output T.

by the algorithm has weight $(k-1)(2-\varepsilon)$. On the other hand, the optimal k-cut picks all edges of weight 1, and has weight k. □

4.3 Exercises

4.1 Show that Algorithm 4.3 can be used as a subroutine for finding a k-cut within a factor of $2-2/k$ of the minimum k-cut. How many subroutine calls are needed?

4.2 A natural greedy algorithm for computing a multiway cut is the following. Starting with G, compute minimum s_i–s_j cuts for all pairs s_i, s_j that are still connected and remove the lightest of these cuts; repeat this until all pairs s_i, s_j are disconnected. Prove that this algorithm also achieves a guarantee of $2-2/k$.

The next 4 exercises provide background and an algorithm for finding Gomory–Hu trees.

4.3 Let $G=(V,E)$ be a graph and $w: E \rightarrow \mathbf{R}^+$ be an assignment of nonnegative weights to its edges. For $u,v \in V$ let $f(u,v)$ denote the weight of a minimum u–v cut in G.

1. Let $u,v,w \in V$, and suppose $f(u,v) \le f(u,w) \le f(v,w)$. Show that $f(u,v)=f(u,w)$, i.e., the two smaller numbers are equal.
2. Show that among the $\binom{n}{2}$ values $f(u,v)$, for all pairs $u,v \in V$, there are at most $n-1$ distinct values.
3. Show that for $u,v,w \in V$,

$$f(u,v) \ge \min\{f(u,w), f(w,v)\}.$$

4. Show that for $u,v,w_1,\ldots,w_r \in V$

$$f(u,v) \ge \min\{f(u,w_1), f(w_1,w_2), \ldots, f(w_r,v).\} \tag{4.1}$$

4.4 Let T be a tree on vertex set V with weight function w' on its edges. We will say that T is a *flow equivalent tree* if it satisfies the first of the two Gomory–Hu conditions, i.e., for each pair of vertices $u,v \in V$, the weight of a minimum u–v cut in G is the same as that in T. Let K be the complete graph on V. Define the weight of each edge (u,v) in K to be $f(u,v)$. Show that any maximum weight spanning tree in K is a flow equivalent tree for G.
Hint: For $u,v \in V$, let $u, w_1, \ldots, w_r, v$ be the unique path from u to v in T. Use (4.1) and the fact that since T is a maximum weight spanning tree, $f(u,v) \le \min\{f(u,w_1),\ldots,f(w_r,v)\}$.

4.5 Let $(A,\bar{A})$ be a minimum s–t cut such that $s \in A$. Let x and y be any two vertices in A. Consider the graph G' obtained by collapsing all vertices of $\bar{A}$ to a single vertex $v_{\bar{A}}$. The weight of any edge $(a, v_{\bar{A}})$ in G' is defined to be the sum of the weights of edges (a,b) where $b \in \bar{A}$. Clearly, any cut in G' defines a cut in G. Show that a minimum x–y cut in G' defines a minimum x–y cut in G.

4.6 Now we are ready to state the Gomory–Hu algorithm. The algorithm maintains a partition of V, $(S_1, S_2, \ldots S_t)$, and a spanning tree T on the vertex set $\{S_1,\ldots,S_t\}$. Let w' be the function assigning weights to the edges of T. Tree T satisfies the following invariant.
Invariant: For any edge (S_i,S_j) in T there are vertices a and b in S_i and S_j respectively, such that $w'(S_i,S_j)=f(a,b)$, and the cut defined by edge (S_i,S_j) is a minimum a–b cut in G.

The algorithm starts with the trivial partition V, and proceeds in $n-1$ iterations. In each iteration, it selects a set S_i in the partition such that $|S_i| \ge 2$ and refines the partition by splitting S_i, and finding a tree on the refined partition satisfying the invariant. This is accomplished as follows. Let x and y be two distinct vertices in S_i. Root the current tree T at S_i, and consider the subtrees rooted at the children of S_i. Each of these subtrees is collapsed into a single vertex, to obtain graph G' (besides these collapsed vertices, G' contains all vertices of S_i). A minimum x–y cut is found in G'. Let (A,B) be the partition of the vertices of G' defining this cut, with $x \in A$ and $y \in B$, and let w_{xy} be the weight of this cut. Compute $S_i^x = S_i \cap A$ and $S_i^y = S_i \cap B$, the two sets into which S_i splits.

The algorithm updates the partition and the tree as follows. It refines the partition by replacing S_i with two sets S_i^x and S_i^y. The new tree has the edge (S_i^x, S_i^y), with weight w_{xy}. Consider a subtree T' that was incident at S_i in T. Assume w.l.o.g. that the node corresponding to T' lies in A. Then, T' is connected by an edge to S_i^x. The weight of this connecting edge is the same as the weight of the edge connecting T' to S_i. All edges in T' retain their weights.

Show that the new tree satisfies the invariant. Hence show that the algorithm terminates (when the partition consists of singleton vertices) with a Gomory–Hu tree for G.

Consider the graph: