CHAPTER 19

EXERCISE 19.1. Consider the following two cases:

Case 1: \( \alpha \geq \beta \). Then \( c+d = (c-d)+2d = (\alpha-\beta)+2d = |\alpha-\beta|+2d \). Hence, \( c+d \) would be minimized when \( d = 0 \). Consequently, minimizing \( (\alpha-\beta) \) is equivalent to minimizing \( (c+d) \).

Case 2: \( \alpha \leq \beta \). This case may be treated in a similar way as Case 1.

EXERCISE 19.3. (a) Construct a network \( G = (N, A) \) representing the physical transformation network. Let the capacity of arc \( (i, j) \) in the network be \( c_{ij} \). A minimum \( s-t \) cut in \( G \) will yield the subset of arcs, the destruction of which will disconnect nodes \( s \) and \( t \) in the cheapest manner.

(b) Perform the node splitting transformation described in Section 2.4. Split up each node \( i \) in the network \( G \) into two nodes \( i' \) and \( i'' \) and set the capacity of arc \( (i', i'') \) to \( c_i \). All arcs entering node \( i \) in the original network now enter node \( i' \) and all arcs emanating from node \( i \) in the original network now emanate from node \( i'' \). The destruction of the arc \( (i', i'') \) is equivalent to the destruction of node \( i \) in the original network. A minimum \( s-t \) cut in the transformed network will yield a subset of nodes and arcs in the original network, the destruction of which will disconnect nodes \( s \) and \( t \) in the cheapest manner.

EXERCISE 19.5. This problem may be formulated as the dual of a minimum cost flow problem. The problem may be stated in the form of a linear program as follows:

\[
\text{Minimize } \left[ \log \Pi_{\{(i, j) : a_{ij} \neq 0\}} \left( \frac{\alpha_i}{\beta_j} \right)^{a_{ij}} \right] \\
\text{or, equivalently, minimize } \left[ \log \Pi_{\{(i, j) : a_{ij} \neq 0\}} \left( \frac{\alpha_i}{\beta_j} \right)^{a_{ij}} \right]
\]

subject to

\[
\log l \leq \log \alpha_i + \log |a_{ij}| - \log \beta_j \leq \log u, \text{ for each } i = 1, ..., p \text{ and } j = 1, ..., q,
\]

because \( |a_{ij}| \)'s are constants. Let \( b(i) \) and \( d(j) \) be the number of nonzero entries in row \( i \) and column \( j \), respectively. Then the objective function can be rewritten as

\[
\log \left[ \Pi_{\{(i, j) : a_{ij} \neq 0\}} \left( \frac{\alpha_i}{\beta_j} \right)^{a_{ij}} \right] = \sum_{1 \leq i \leq p} b(i) \log \alpha_i - \sum_{p+1 \leq j \leq p+q} d(j) \log \beta_j.
\]

Using the same transformation of variables, as in Application 19.5, yields:

\[
\text{Minimize } \sum_{1 \leq i \leq p} b(i) \pi(i) - \sum_{p+1 \leq j \leq p+q} d(j) \pi(j),
\]

subject to \( c_{ij} - \pi(i) + \pi(j) \geq 0 \) for each arc \( (i, j) \in A \) in the network shown in Figure 19.7. Now use the duality theory to transform this problem into a minimum cost flow problem on a bipartite network.

EXERCISE 19.7. The solution of this exercise is rather long and is given in the following paper:

EXERCISE 19.9. (a) The time-expanded replica of a network cannot contain a directed cycle, because each arc in the network is from a node representing an earlier time period to a node representing a later time period. Hence, the time-expanded replica of the network will always be a directed acyclic graph.

(b) See the solution of Exercise 19.12.

(c) The example shown in Figure S19.9 has multiple optimal solutions. When the network given in Figure S19.9(a) is expanded over 3 time periods (see Figure S19.9(b)), then the maximum flow value is one unit and the two alternate solutions are: (i) one unit flow along the path $s^* \rightarrow t^1 \rightarrow t^2 \rightarrow t^3 \rightarrow t^*$; and (ii) one unit flow along the path $s^* \rightarrow t^0 \rightarrow t^2 \rightarrow t^3 \rightarrow t^*$.

(d) Let node $t$ be the sink node in the original network, and $t^0$, $t^1$, $t^2$, ..., $t^p$ be the copies of this node in the time-expanded network. First obtain a maximum flow from node $s^*$ to node $t^0$; let $x^0$ be a maximum flow. Now with $x^0$ as the starting flow, obtain a maximum flow from node $s^*$ to node $t^1$; let $x^1$ be a maximum flow. With $x^1$ as the starting flow, obtain a maximum flow from node $s^*$ to $t^2$. Repeat this process until we obtain a maximum flow from node $s^*$ to $t^p$.

EXERCISE 19.11#. This exercise contains a typographic error. In line 4 of the exercise, $s^p$ should be $s^r$ and $t^p$ should be $t^r$.

(a) In order to solve this problem we need to perform $O(\log p)$ applications of any maximum flow problem. We do binary search on the number of time periods $r$ over which the time expanded replica $G^r$ is to be constructed. By using binary search on $r$, we can determine in $O(\log p)$ application of any maximum flow algorithm the smallest index $r$ so that the maximum flow from the nodes in $s^1$, $s^2$, ..., $s^r$ to the nodes in $t^1$, $t^2$, ..., $t^r$ equal at least $B$.

(b) Add a super-source node $s^*$ with supply equal to $B$ and connect to the nodes $s^1$, $s^2$, ..., $s^p$ through zero cost uncapacitated arcs. Similarly, add a super-sink node $t^*$ with demand equal to $B$ and connect each of the nodes $t^1$, $t^2$, ...
..., \( t' \) to node \( t^* \) by uncapacitated arcs. The cost of the arc \( (t^k, t^*) \) is \( M^k \) for every \( 1 \leq k \leq r \) for some sufficiently large constant \( M \). An minimum cost flow in the network will give the desired solution. The optimal \( r \) is the largest index such that \( (t^r, t^*) \) carries positive flow.

**EXERCISE 19.13.** Suppose that the arc traversal times are in some appropriate time units. We decompose 24 hours of the day in the same time units. Suppose that we decompose a day in \( H \) time units. (For example, if arc traversal times are multiples of 10 minutes, then \( H = (24 \times 60)/10 = 144 \).) Let \( c^0_{ij} \), \( 1 \leq h \leq H \), denote the traversal time of arc \((i, j)\) if we start traversing the arc at time \( h \). To solve the problem, we construct a time-expanded network as described in Exercise 19.12 and replace each node \( i \) by \( H \) copies, \( i^1, i^2, \ldots, i^H \). We then examine each arc \((i, j) \in A\) one by one, and add the following arcs to the time expanded network: \((i^1, j^{1+c_{ij}})\), \((i^2, j^{2+c_{ij}})\), \((i^3, j^{3+c_{ij}})\), ..., as long as \( h + c_{ij} \leq H \). Add in the arc \((a^k, a^{k+1})\) for each \( k \) to denote waiting at node \( i \) from time \( k \) to time \( (k+1) \). If we want to determine the shortest path from \( s \) to node \( t \) starting at time \( p \), then apply the search starting at node \( s^p \) and identify all reachable nodes. If \( t^j \) is the lowest indexed node reachable from node \( s^p \), then \((q-p)\) is the length of the shortest path. We assume that the data is integral and arc traversal times are positive.

**EXERCISE 19.15.** This exercise assumes that all arc costs are nonnegative and that the subgraph has odd number of incident arcs to some node in \( S \) and even number of incident arcs to each node not in \( S \). Let \( c'_{ij} \) be the cost of the shortest path from node \( i \) to node \( j \) and \( p_{ij} \) be the minimum cost path. Define a complete graph \( G' \) on nodes in \( S \) where the cost of arc \((i,j)\) is \( c'_{ij} \). Find the minimum cost weighted (nonbipartite) matching in \( G' \). The union of the shortest paths \( p_{ij} \) for the arcs in the minimum matching in the original graph; suppose \( H \) is the resulting subgraph. This is the optimal solution of the original problem if arcs are allowed to be repeated. To eliminate repetition of arcs, we need to perform one more step. If \( H \) has an odd number of arcs between any pair of nodes, then replace these by a single arc, and if \( H \) has even number of arcs between any pair of nodes, then delete all the arcs; notice that this does not worsen the cost of solution because arc costs \( c'_{ij} \) are nonnegative and also does not change the parity (odd/even degrees of arcs incident on a node) of any node. The resulting subgraph is the desired solution.

**EXERCISE 19.17.** Whenever a user whose destination is block \( i \) and requires a parking facility of class \( k \) is assigned the parking facility \( j \), we incur two kinds of costs: (1) the walking cost which depends upon the destination block \( i \) to which the user desires to go, and the cost of the parking facility \( j \) to which the user is actually assigned (let \( c_j \) denote the sum of these two costs); and (2) the cost of maintaining a parking slot of class \( k \) (let \( c_{jk} \) denote the cost of maintaining a parking slot of class \( k \) in the parking facility \( j \) (let this cost be denoted by \( p_{jk} \)). Now construct a bipartite network \( G = (N_1 \cup N_2 \cup \{s, t\}, A) \), in which \( N_1 \) contains 1.K nodes. Each node \( i^k \) (\( 1 \leq i \leq 1, 1 \leq k \leq K \)) in \( N_1 \) has a supply of \( D_{ik} \), which denotes the number of users belonging to block \( i \) requiring a parking facility of class \( k \). The node set \( N_2 \) contains a node for each parking facility \( j \), which is connected to the sink node \( t \) by a directed arc \((j, t)\) of capacity \( s_j \) (this denotes the capacity constraint of facility \( j \)). The network \( G \) also contains an arc \((i^k, j)\) for each \( i^k \in N_1 \), and \( j \in N_2 \) with cost \( c_{ij} + p_{jk} \). The flow on this arc denotes the number of users belonging to block \( i \) who require a parking facility of class \( k \) and are assigned to class \( j \). A minimum cost flow in this network yields an optimal assignment of users to parking facilities.

**EXERCISE 19.19.** We use the same transformation as described in Application 6.6 (the tanker scheduling problem) with the following changes: (a) each "shipment arc" in Application 6.6 is replaced by a "cargo arc" whose lower bound is equal to the number of trucks in the cargo; (b) the cost of each arc emanating from the source node to the origin \( j \) of some cargo arc is equal to the cost of driving to that origin (i.e., \( c_{ij} \)); (c) the cost of the arc which represents "switching" to job \( i \) after completion of job \( i \) is set to \( f_{ij} \); (d) the cost of the cargo arc for job \( i \) is set to \( d_i \), because \( d_i \) is the cost incurred per truck in performing job \( i \) (note that "per truck" is missing in the statement of the
exercise); and (e) we join the sink node to the source node by a zero cost uncapacitated arc \((t, s)\). A minimum cost flow in this network will yield an optimal solution of the truck scheduling problem.

**EXERCISE 19.21.** We can formulate the optimal deployment problem as a transportation problem in a network \(G = (N_1 \cup N_2, A)\), where the set \(N_1\) contains \(p\) nodes and has a node \(i\) for companies at location \(i\) with supply \(u_i\). The set \(N_2\) contains \(j\) subsets of nodes and the \(j\)th subset has \(v_j\) nodes with unit demand. We represent these nodes by \([\{1^1, 1^2, \ldots, 1^{v_1}\}, \{2^1, 2^2, \ldots, 2^{v_2}\}, \ldots, \{q^1, q^2, \ldots, q^{v_q}\}]\), where a node \(j^k\) represents the arrival of the \(k\)th company at event \(j\). Each node \(i\) in \(N_1\) has an uncapacitated arc \((i, j^k)\) to each node \(j^k\) in \(N_2\), and the unit cost of flow on this arc is \(\tau_{ij} \alpha_{jk}\). It is easy to establish one-to-one correspondence feasible flows in this network and assignment of companies to incidents. Notice, however, that this formulation is an instance of the unbalanced transportation problem (i.e., total supplies exceed the total demands) but can be converted to a standard transportation problem by adding a dummy demand node.