CHAPTER 16

EXERCISE 16.1. (a) The Lagrangian multiplier problem for the equality formulation is $L^* = \max_\mu L(\mu)$ with $L(\mu) = \min\{cx + \mu(\mathbf{A}x + s - b): x \in \mathbf{X}, s \geq 0\}$.

(b) If some $\mu_i < 0$, then we can increase the value of $s_i$ in any feasible solution as much as desired, and in doing so decrease the value of the objective function $cs + \mu(\mathbf{A}x + s - b)$ to any preassigned negative number; therefore, $L(\mu) = -\infty$.

If $\mu_i > 0$, then decreasing the value of $s_i$ in any feasible solution of the problem (while keeping other variables constant) will strictly decrease the value of the objective function. Since $s_i \geq 0$ in any feasible solution, $s_i = 0$ in the optimal solution.

(c) Part (b) shows that for any value of the Lagrangian multiplier $\mu \geq 0$, $L(\mu) = \min\{cx + \mu(\mathbf{A}x + s - b): x \in \mathbf{X}, s \geq 0\} = \min\{cx + \mu(\mathbf{A}x - b), x \in \mathbf{X}\}$. It also shows that $L(\mu) = -\infty$ if $\mu_i < 0$ for some $i$. Therefore, in solving the Lagrangian dual problem, we can restrict attention to values of $\mu \geq 0$ and so $L = \min\{cx + \mu(\mathbf{A}x - b), x \in \mathbf{X}\}$.

EXERCISE 16.3. (a) If some $(c-\mu \mathbf{A})_j < 0$, then we can increase the value of $x_j$ in any feasible solution of the problem as much as desired so that the value of the objective function $(c-\mu \mathbf{A})x + \mu b$ decreases to any preassigned negative number, i.e., $L(\mu) = -\infty$.

(b) If some $(c-\mu \mathbf{A})_j > 0$, then decreasing the value of $x_j$ in any feasible solution of the problem (while keeping the other variables constant) will strictly decrease the value of the objective function. Therefore, $x_j$ will assume the smallest possible feasible value in any optimal solution, i.e., $x_j = 0$.

(c) Part (a) implies that we can restrict the optimal Lagrangian multiplier $\mu^*$ to those values satisfying the conditions $c-\mu^* \mathbf{A} \geq 0$. But then part (b) implies that $(c-\mu^0 \mathbf{A})x^* = 0$ for any optimal solution $x^*$ whenever the Lagrangian objective value $L(\mu^0)$ is bounded. Therefore, $L^* = \max_\mu L(\mu) = \max_\mu \{\min_{x \geq 0} \{(c-\mu \mathbf{A})x + \mu b\}\} = \max_\mu \{\mu b\}: c-\mu \mathbf{A} \geq 0\}$, which is the dual of the given linear program.

EXERCISE 16.5. As shown in 16.2, the path 1-2-5-6 has a cost of 5 and a travel time of 15; the path 1-3-2-5-6 has a cost of 15 and a travel time of 10. The convex combination of these paths with a weight of 0.8 on path 1-2-5-6 and 0.2 on path 1-3-2-5-6 has a cost of 7 and a travel time of 14. Therefore, this convex combination is feasible and has a cost equal to $L^* = 7$.

Note that the paths 1-2-5-6 and 1-3-2-5-6 define the lines that intersect at the maximum value of the Lagrangian function $L(\mu)$, at $\mu = 2$, in Figure 16.3. (Recall that the shaded region corresponds to the linear program defined by the Lagrangian dual problem.) Therefore, if we consider the linear programming relaxation of the model (16.1) and set the linear programming dual variable of the constraint (16.1c) to the value $\mu = 2$, then these paths solve the resulting shortest path problems. Consequently, $L^* = 7$ is a lower bound on the linear programming relaxation of
the problem (16.1). The weight $0.8$ and $0.2$ on the paths $1$-$2$-$5$-$6$ and $1$-$3$-$2$-$5$-$6$ defines arc flows of $x_{12} = 0.8$, $x_{13} = x_{32} = 0.2$, $x_{25} = x_{56} = 1$ and $x_{ij} = 0$ otherwise. This solution satisfies the constraints of the linear programming relaxation of (16.1) and has a cost of $7$. Since this value equals the Lagrangian lower bound on the linear program, the $x$ values solve the linear program. Seen in another way, we can define dual variables for the linear programming relaxation of problem (16.1) as $\mu = 2$ (for constraint 16.1c), and $\pi_1 = 0$, $\pi_2 = -21$, $\pi_3 = -16$, $\pi_4 = -24$, $\pi_5 = -29$, and $\pi_6 = -35$ (for the constraints 16.1b). Note that $\pi_j$ is the negative of the shortest path distance from node $l$ to node $j$ with respect to $c_{ij} + 2t_{ij}$. With this choice of (feasible) dual variables, the objective function of dual of the linear programming relaxation of (16.1) is $-\pi_6 + \pi_1 - \mu T = 35 - 2(14) = 7$. Therefore, the linear programming primal value of $7$ equals a feasible dual value and so is optimal.

**EXERCISE 16.7.** (a) Associating a Lagrangian multiplier $-\pi(i)$ with a mass balance constraint of the $i$th node, the Lagrangian subproblem becomes

$$L(\pi) = \text{minimize} \{ \sum_{(i, j) \in A} c_{ij} x_{ij} - \sum_{i \in N} \pi(i) \sum_{j: (i, j) \in A} x_{ij} - \sum_{j: (i, j) \in A} x_{ji} - b(i) \}; \ 0 \leq x_{ij} \leq u_{ij} \}$$

or, equivalently, minimize

$$\{ \sum_{(i, j) \in A} c_{ij} x_{ij} - \sum_{i \in N} \pi(i) b(i); \ 0 \leq x_{ij} \leq u_{ij} \}$$

i.e., minimize

$$\sum_{(i, j) \in A} (c_{ij} - \pi(i) + \pi(j)) x_{ij}; \ 0 \leq x_{ij} \leq u_{ij}.$$  

Since each variable $x_{ij}$ appears in a single constraint in the relaxed problem, the optimal solution is easy to determine: we set $x_{ij} = 0$ if $c_{ij} > 0$; $x_{ij} = u_{ij}$ if $c_{ij} < 0$ and $x_{ij}$ to any value $0 \leq x_{ij} \leq u_{ij}$ if $c_{ij} = 0$. But these are exactly the complementary slackness conditions of Theorem 9.4 or, equivalently, the optimality conditions $c_{ij} \pi_{ij} \geq 0$ for all arcs $(i, j)$ in $G(x)$ (see Theorem 9.3). Therefore, the pseudo flow solves the Lagrangian subproblem.

(b) We have just seen that the current pseudo flow is optimal for the Lagrangian subproblem. Now consider the new Lagrange multipliers $\pi' = \pi - d$ (recall that $d$ is the vector of shortest path distances with respect to the reduced costs $c_{ij}'$). Lemma 9.11 shows that $x$ satisfies the reduced cost optimality conditions for the vector $\pi'$ and so by part (a), it also solves the Lagrangian subproblem for the Lagrange multiplier $\pi'$. Moreover, by Lemma 9.12, the flow $x'$ we obtain by augmenting on the shortest path from an excess (node $s$) to a deficit node (node $l$) satisfies the reduced cost optimality conditions for the vector $\pi'$, so it also solves the Lagrangian subproblem for the Lagrange multiplier $\pi'$. Therefore, since $L(\pi) = \sum_{(i, j) \in A} c_{ij} x_{ij} - \sum_{i \in N} \pi(i) \sum_{j: (i, j) \in A} x_{ij} - \sum_{j: (j, i) \in A} x_{ji} - b(i)$ and $x$ solves the Lagrangian subproblem for the Lagrange multipliers $\pi$ and $\pi'$,

$$L(\pi) - L(\pi') = \sum_{i \in N} d(i) \sum_{j: (i, j) \in A} x_{ij} - \sum_{j: (j, i) \in A} x_{ji} - b(i) = d(l) e(l) - d(s) e(s) = d(l) e(l) < 0$$

since $d(l) > 0$ and $e(l) < 0$. Thus $L(\pi')$ provides better lower bound on the optimal objective function value than does $L(\pi)$.

Finally, note that if after the augmentation, the problem has no excess nodes (and so no deficit nodes), then $L(\pi) = \{ \sum_{(i, j) \in A} c_{ij} x'_{ij} - \sum_{i \in N} d(i) \sum_{j: (i, j) \in A} x'_{ij} - \sum_{j: (j, i) \in A} x'_{ji} - b(i) \} = \sum_{(i, j) \in A} c_{ij} x'_{ij}$. But then the lower bound $L(\pi)$ equals the objective value of a feasible solution and so the solution $x'_{ij}$ is an optimal flow and the Lagrangian multipliers $\pi'$ solve the Lagrangian multiplier problem.
EXERCISE 16.9.  (a) If we relax the constraints 16.11(b), the Lagrangian subproblem becomes: minimize $\sum_{i \in I} (c_{ij} + \lambda_i) x_{ij} + \sum_{j \in J} F_j y_j - \sum_{i \in I} d_i x_{ij}$ subject to $\sum_{i \in I} d_i x_{ij} \leq K_j y_j$ for all $j \in J$; $0 \leq x_{ij} \leq 1$ and $y_j = 0$ or 1.

This Lagrangian problem decomposes into a separate subproblem for each location $j$: minimize $\sum_{i \in I} (c_{ij} + \lambda_i) x_{ij}$ subject to $\sum_{i \in I} d_i x_{ij} \leq K_j y_j$; $0 \leq x_{ij} \leq 1$ and $y_j = 0$ or 1. To solve this Lagrangian subproblem, we choose the least cost solution obtained by solving the two linear programs: one fixing $y_j$ to 0 and the other fixing $y_j$ to 1.

(b) If we relax the constraints 16.11(c), the Lagrangian subproblem becomes: minimize $\sum_{i \in I} \sum_{j \in J} (c_{ij} + d_i \mu_j) x_{ij}$ subject to $\sum_{j \in J} x_{ij} = 1$ for all $i \in I$; $0 \leq x_{ij} \leq 1$ and $y_j = \{0, 1\}$. We can determine the value of the zero-one variable $y_j$ by checking the sign of $(F_j \mu_j K_j)$, choosing $y_j = 1$ if $(F_j \mu_j K_j) \leq 0$ and $y_j = 0$ if $(F_j \mu_j K_j) > 0$. Once we have decided on the value of $y_j$, the subproblem becomes: minimize $\sum_{i \in I} \sum_{j \in J} (c_{ij} + d_i \mu_j) x_{ij}$ subject to $\sum_{j \in J} x_{ij} = 1$ for all $i \in I$; $0 \leq x_{ij} \leq 1$. This problem decomposes into a separate subproblem for each customer $i$: minimize $\sum_{j \in J} (c_{ij} + d_i \mu_j) x_{ij}$ subject to $\sum_{j \in J} x_{ij} = 1$, $0 \leq x_{ij} \leq 1$. We obtain the solution of subproblem $i$ by setting $x_{ij} = 1$ for that value of $j \in J$ so that $(c_{ij} + d_i \mu_j)$ is as small as possible (and setting $x_{ij} = 0$ for all other values of $j$).

(c) When we relax the integrality constraint of (16.11b), the optimal solution of the Lagrangian problem yields $\mu_1 = -F_1$, $y_1 = 1$, and $x_{11} = 1$ and so $L^*_1 = F_1 > 0$. When we relax the constraint 16.11(c), we obtain $\mu_1 = F_1/10$, $y_1 = 1$, and $x_{11} = 1$, which yields $L^*_2 = F_1/2 < L^*_1$. In general, the solution we obtain by relaxing (16.11b) will be sharper than the solution obtained by relaxing (16.11c) because the latter satisfies the integrality property and the former does not. (Theorem 16.9 and 16.10 establish this result.)

EXERCISE 16.11#.  (Note: the added constraint should be $x_{ij} \leq \min \{y_j, K_j/d_j\}$, not $x_{ij} \leq \min \{y_j, K_j\}$)

(a) Let $L^1(\mu)$ be the Lagrangian subproblem obtained from the original model and $L^2(\mu)$ be the Lagrangian subproblem in the restricted problem. Since the set of feasible solutions in $L^2(\mu)$ is a subset of the set of feasible solutions in $L^1(\mu)$ and we are minimizing, $L^2(\mu) \geq L^1(\mu)$. Consequently, $\max_{\mu} L^2(\mu) \geq \max_{\mu} L^1(\mu)$.

(b) If we relax the constraints 16.11(b), we can use the same a method used in the solution of Exercise 16.9(a), except that in this case we introduce the additional constraints $x_{ij} \leq \min \{y_j, K_j/d_j\}$ for all $i \in I$ in the Lagrangian subproblem $j$ of the decomposition. Therefore, for each $j$, and for $y_j = 0$ and $y_j = 1$, we again solve a linear program, but now with upper bounds that might not be one (if $K_j/d_j < 1$) constraints.

If we relax the constraints 16.11(c), the subproblem contains the constraints $x_{ij} \leq \min \{y_j, K_j/d_j\}$ or $x_{ij} \leq y_j$ and $x_{ij} \leq K_j/d_j$. Therefore, the subproblem becomes an uncapacitated facility location problem (since once we open location $j$, the constraint $x_{ij} \leq y_j$ does not restrict the fraction of the demand we can send from any customer $i$ to location $j$) together with an upper bound on the flow between any individual customer and location (that is, the problem does not impose joint customer capacity on any location). To solve this problem, we would have to use any algorithm for the uncapacitated facility location problem; for example, we could use Lagrangian relaxation again, relaxing either the constraints 16.11(b) or the forcing constraints $x_{ij} \leq y_j$.  

(c) In this case, the model is slightly different since if $K_j/d_i < 1$, we cannot assign customer $i$ to location $j$. Therefore, we simply eliminate the variable $x_{ij}$ from the problem. The remaining variables have the constraints $x_{ij} \leq y_j$. In this case, once we have eliminated the variables with $K_j/d_i < 1$, if we relax the constraints 16.11(b), the problem once again decomposes by location $j$ and so we solve the subproblem as in Exercise 16.9. If we relax the constraints 16.11(c), the resulting problem is an uncapacitated version of the facility location problem as in part (b), though without the variables upper bounds $x_{ij} \leq K_j/d_i$. Therefore, we would solve the problem as indicated in part (b).

**EXERCISE 16.13** #. (Note: The problem statement contains an error: it should be $x_{kj} \geq x_{ij}$, not $x_{ik} \geq x_{ij}$.)

(a) The added constraint says that if we assign node $i$ to a multiplexer at node $j$ (or to the switching center if $j = 1$), then $x_{ij} = 1$ and so $x_{ik} = 1$. Therefore, we must assign every node $k$ on the path $P_{ij}$ from node $i$ to node $j$ to the same multiplexer at node $j$.

(b) The relaxed problem has an objective function of the form

$$\min \sum_{i \in S} \sum_{j \in S} \alpha_{ij} x_{ij} + \sum_{(i, j) \in T} c_{ij} y_{ij} + F \sum_{j \in S} z_j + \text{constant}.$$  

Since the only constraint in the relaxation imposed upon $y_{ij}$ is $y_{ij} \geq 0$, the solution sets $y_{ij} = 0$ if $\gamma_{ij} \geq 0$ and $y_{ij} = -\infty$ if $\gamma_{ij} < 0$. Since $\gamma_{kl} = (c_{kl} - \lambda_{kl})$, the Lagrangian multiplier $\lambda_{kl}$ imposed upon arc $(k, l)$ must satisfy the condition $\lambda_{kl} \leq c_{kl}$. With this choice, in the optimal solution to the Lagrangian subproblem $\Sigma_{(i, j) \in T} \gamma_{ij} y_{ij} = 0$.

The remaining problem is a facility location problem on a tree with the stipulation that the set of nodes assigned to any node $j$ must be in a subtree of $T$ containing node $j$. Define $\alpha_{jj} = F$ for any node $j$. With this definition, all the costs become assignment costs (that is, $\alpha_{jj}$ is the cost of assigning node $j$ to node $j$).

To develop a dynamic programming recursion for this problem, we root the tree $T$ at node 1 and let $T_j$ denote the subtree of $T$ rooted at node $j$ (that is, $T_j$ contains all nodes $i$ whose path to node 1 in $T$ passes through node $j$). We then define two quantities:

$$C(T_j) = \text{optimal cost of a tree rooted at node } j, \text{ assuming all the nodes in } T_j \text{ are assigned to a concentrator in } T_j.$$  

$$C(T_j, q) = \text{optimal cost of a tree rooted at node } j, \text{ assuming node } j \text{ is assigned to node } q \text{ (in or out of } T_j)$$

Let $S_j$ be the set of immediate successors of node $j$ in $T$; that is, those nodes whose path to node 1 has node $j$ as its first intermediate node.

We will compute the quantities $C(T_j)$ and $C(T_j, q)$ by moving “upward” in the tree $T$, starting at its endnodes. For an endnode $j$, $C(T_j) = F$ since node $j$ must have its own concentrator and $C(T_j, q)$ for $q \neq j$ equals $\alpha_{jq}$, the cost of assigning node $j$ to node $q$.

We make two observations that are consequences of the contiguity assumption:

1. If node $i \in S_j$ is assigned to some node $q$ outside of $T_i$, then node $j$ must be assigned node $q$ as well, and
(2) if node \( j \) is assigned to some node in \( T_i \) and \( i \in S_j \), then every node in \( T_i \) must be assigned to a node in \( T_i \).

These observations imply the following results for computing \( C(T_j, q) \).

If \( q \notin T_j \) or \( q = j \),
\[
C(T_j, q) = \alpha_{jq} + \sum_{i \in S_j} \min\{C(T_i), C(T_i, q)\}.
\]

If \( q \in T_i \) for some \( i \in S_j \),
\[
C(T_j, q) = \alpha_{jq} + C(T_i) + \sum_{p \in S_j, p \neq i} \min\{C(T_i), C(T_i, q)\}.
\]

Having computed these values for all \( q \), we set
\[
C(T_j) = \min_{q \in T_j} C(T_j, q).
\]

**EXERCISE 16.15.** Let \( s \in S \) be designated as the source node for each commodity and let each \( k \neq s \in S \) define a commodity. Therefore, for each commodity \( k \neq s \in S \), we wish to send one unit from node \( s \) to node \( k \). We set the routing cost on each arc in the network design model to value zero and the fixed cost of each arc \((i, j)\) to \( F_{ij} = c_{ij} \) (for each undirected arc \([i, j]\) in the original graph, we include the two directed arcs \((i, j)\) and \((j, i)\), each with the same fixed cost, in the network design formulation). In order to send one unit from node \( s \) to each node \( k \neq s \in S \), the network design solution must define a connected graph connecting the nodes in \( S \). Since the routing cost are zero and the fixed costs \( F_{ij} = c_{ij} \) are nonnegative, the network design problem contains a solution with no cycles.

Therefore, since we can direct any Steiner tree solution to be a directed tree out of node \( s \), the network design problem and the Steiner tree problem are equivalent.

To solve the network design formulation of the Steiner tree problem, we could, as indicated in the text, relax the forcing constraints so that the subproblem decomposed into a separate shortest path problem for each commodity. We can use the subgradient optimization technique to obtain the optimal value of the Lagrange multipliers.

**EXERCISE 16.17.** (a) The formulation is exactly the same as described in Application 16.4, except that in this case we modify the "forcing constraints" so they become \( x_{ij}^k \leq \min\{y_{ij}, y_{ji}\} \), or, \( x_{ij}^k \leq y_{ij} \) and \( x_{ij}^k \leq y_{ji} \). We can once again solve the problem by relaxing the "forcing constraints" and in so doing decompose the problem into \( K \) independent shortest path problems (one for each commodity \( k \)).

(b) Since the problem has a commodity between every pair of nodes, the network designed must be a connected network. Moreover, since flow costs are zero, we need to obtain a minimum cost connected network. Finally, since all the fixed arc costs are nonnegative, a minimum cost connected network must be a minimum spanning tree.

**EXERCISE 16.19.** (a) If we relax the mass balance constraint \( N x = b \), then the resulting objective function becomes \( \sum_{(i, j) \in A} c_{ij} y_{ij} + \lambda(N x - b) \) which can be transformed to the form \( \sum_{(i, j) \in A} [c_{ij} y_{ij} + (\lambda_i - \lambda_j) x_{ij}] + \lambda b \).

For a particular value of \( \lambda \), if \( \lambda_i - \lambda_j \geq 0 \), then the value of \( x_{ij} \) in some optimal solution will be as small as possible (i.e., zero) and if \( \lambda_i - \lambda_j < 0 \), then the value of \( x_{ij} \) in the optimal solution will be as large as possible (i.e., \( (n-1)y_{ij} \)). Therefore, we can remove the variables \( x_{ij} \) (and, of course, the constraints 16.3(e) and 16.3(f)). Note that
the resulting problem is simply an assignment problem. (The zero-one constraints on \( y_{ij} \) actually become redundant because the assignment problem will always have a zero-one solution.)

(b) If we relax the assignment constraints, then the objective function becomes \( \sum_{(i, j) \in A} (c_{ij} + \lambda_i + \mu_j)y_{ij} - \sum_{i=1}^{n} \lambda_i - \sum_{j=1}^{n} \mu_j \). Note that for a fixed value of the Lagrange multipliers, the Lagrangian subproblem is actually a network design problem: the fixed cost of arc \((i, j)\) is \((c_{ij} + \lambda_i + \mu_j)\), and the flow cost of arc \((i, j)\) is zero.

(c) Because the relaxations described in the text and in part (a) of this problem satisfy the integrality property, the optimal objective function values for both these relaxations equal that of the linear programming relaxation. However, the Lagrangian subproblem in (b) need not satisfy the integrality property, and so, in general, the bound obtained by this relaxation will be sharper.

**EXERCISE 16.21#.** (Note: the exercises referred to in the hint should be 16.20, not 16.21, and 13.38, not 13.37. The Lagrangian subproblem in case 2 of part (b) should be a matching, not a circulation.)

(a) minimize \( \sum_{(i, j) \in A} c_{ij} y_{ij} \)

subject to

\[
\sum_{j \neq i} y_{ij} \leq 2 \quad \text{for all } i = 1, 2, \ldots, n
\]

\[
\sum_{i \in Q} \sum_{j \in Q} y_{ij} \leq |Q| - 1 \quad \text{for all subsets } Q \text{ of } \{2, 3, \ldots, n\}
\]

\[
\sum_{(i, j) \in A} y_{ij} = n \quad \text{for all } i > j \text{ and } (i, j) \in A.
\]

(b) (1) If we relax the degree 2 constraints, the Lagrangian subproblem solution must contain \( n \) arcs. The subtour constraints imply that the solution on the nodes \{2,3,\ldots,n\} is a forest. If this forest constrain \( k \) components, then node 1 must have \( k + 1 \) additional arcs incident to it. Let \( A' \) denote the arcs not incident to node 1. If we add the additional redundant constraint \( \sum_{(i, j) \in A'} y_{ij} = n - 2 \) to the formulation and relax the degree 2 constraints, then any feasible solution to the subproblem must be a spanning trees on the nodes \{2, 3, \ldots, n\} plus 2 additional arcs incident to node 1. This is a particular type of 1-tree as defined in Exercise 13.38, one with the added arc incident to a specific node, node 1, and with a specified degree (equal to 2) on node 1. (Note: The literature often defines a 1-tree as this special case.)

(b) (2) Since each of the \( n \) arcs adds 1 unit to the degree of exactly 2 nodes, the total degree for an \( n \) arc graph, summed over all nodes, is \( 2n \). But since each of the \( n \) nodes has a degree at most 2, if any node has a degree less than 2, then the total degree would be less than \( 2n \). So each node has degree exactly 2. This problem is a special case of perfect b-matching problem and the transformation described in Section 12.7 shows how to convert this problem into a matching problem.

**EXERCISE 16.23.** (a) \( \sum_{1 \leq i \leq n} \sum_{1 \leq j \leq n} y_{ij} = \sum_{i \in S} \sum_{j \in S} y_{ij} + \sum_{i \in S} \sum_{j \not\in S} y_{ij} + \sum_{i \not\in S} \sum_{j \in N-S} y_{ij} + \sum_{i \not\in S} \sum_{j \not\in S} y_{ij} \).
The assignment constraints imply that the lefthand side equals \( n = |N| \) and that the last term equals \( |N-S| = n - |S| \).

Therefore, \( \sum_{i \in S} \sum_{j \in S} y_{ij} + \sum_{i \in S} \sum_{j \in S-N} y_{ij} = |S| \) and so the first term is less than or equal to \( |S| - 1 \) if and only if the second term is greater than or equal to 1. Noting that this result is true if the variable \( y_{ij} \) are fractional or integer and establishes the desired conclusion.

(b) We consider two cases for the cut condition \( \sum_{i \in S} \sum_{j \in S-N} y_{ij} \geq 1 \): (i) node 1 belongs to \( S \), (ii) node 1 does not belong to \( S \). In case (i), by the max-flow min cut theorem, the cut condition holds if and only if the network with arc capacities \( y_{ij} \) can carry a flow of 1 unit between node 1 and every other node; these are the conditions \( N x_{ik} = b_k \) in the multicommodity flow-based formulation given in Exercise 16.22. In case (ii), the cut condition holds if and only if the network with arc capabilities \( y_{ij} \) can carry a flow of one unit from every node \( k \neq 1 \) to node 1; these are the conditions \( N z_{ik} = d_k \) in the multicommodity flow formulation. Since the flows \( x_{ik} \) and \( z_{ik} \) have no associated cost, we have shown that every feasible solution \( (x,y) = (x^k, z^k, y) \) in the multicommodity flow-based formulation corresponds to a feasible solution \( y \) with the same cost in the “cut” formulation and vice-versa. Since by part (a), the cut formulation is equivalent to the assignment-based formulation, this formulation is equivalent to the multicommodity flow-based formulation.

**EXERCISE 16.25.** (a) The formulation is the same as (16.3) or the model discussed in Exercise 16.20 except that \( K \) arcs must enter and leave node 1; that is, the constraint (16.3b) for node \( i = 1 \) and constraint (16.3c) for \( j = 1 \) become constraints (16.4e) and (16.4f).

We can apply Lagrangian relaxation to this problem much as we apply the method to the traveling salesman problem. For example, in the modified version of model (16.3), if we dualize the forcing constraints (16.3e), the problem decomposes into a network flow problem in the variables \( x \) and a transportation problem in the variable \( y \). If we consider the modified formulation of the model considered in Exercise 20 and dualize the transportation constraints (16.3b) and (16.3c), as modified above, the Lagrangian subproblem becomes a minimum spanning tree problem on the nodes 2, 3, ..., \( n \) (for the arcs adjacent to node 1, we set \( y_{1j} = K \) and \( y_{il} = K \) in the subproblem, then we solve it by finding a minimum spanning tree on nodes, 2, 3, ..., \( n \) plus the \( K \) least cost arcs out of and into node 1 (with respect to the Lagrangian cost coefficients are negative and to value 0 otherwise).

Several other relaxations are possible; for example, if we modify the last relaxation by retaining the constraints
\[
\sum_{1 \leq j \leq n} y_{1j} = K \quad \text{and} \quad \sum_{1 \leq j \leq n} y_{il} = K
\]

in the subproblem, then we solve it by finding a minimum spanning tree on nodes, 2, 3, ..., \( n \) plus the \( K \) least cost arcs out of and into node 1 (with respect to the Lagrangian cost coefficients).

(b) We create copies \( 1^1, 1^2, ..., 1^K \) of node 1. We replace every arc \((l,s)\) with arcs \((1^p,s)\) for \( p = 1, 2, ..., K \), each with same cost as arc \((l,s)\). We replace every arc \((r,l)\) with arcs \((r,1^q)\) for \( q = 1, 2, ..., K \), each with the same cost as arc \((r,l)\). We give each arc \((1^p,1^q)\) a cost \( M > \) the cost of the minimum \( K \)-TSP tour (e.g., \( M > \) the sum of the absolute value of all arc costs).

Every solution to the \( K \)-traveling salesman problem is a set of \( K \) cycles all containing node 1. Suppose we replace the arc \((l,q)\) leaving node 1 in the \( k \)th cycle with arc \((1^K,q)\) and the arc \((p,1)\) entering node 1 in the \( k \)th cycle with the arc \((p,1^{K+1})\), with the convention \( 1^{K+1} = 1^1 \). The result is a TSP tour that contains no arc of the form \((1^p,1^q)\). By renumbering the copies of node 1, we can represent very TSP tour not containing any arc \((1^p,1^q)\) in this
form. Since any tour containing an arc of the form \((l^p, l^q)\) cannot be optimal, solving the TSP on the expanded network solves the K-TSP problem.

**EXERCISE 16.27.**

1. To model different vehicle capacities, we simply replace \(u\) with \(u^k\) in constraint (16.4g).
2. To model the delivery time restriction for each vehicle \(k\), we add the constraint \(\sum_{1 \leq i \leq n} \sum_{1 \leq j \leq n} t_{ij} x_{ij}^k \leq T\). Since for each \(k\), the \(x_{ij}^k\) variables are 0-1 incidence vectors of the route taken by vehicle \(k\), the lefthand side of this constraint is the total time traveled by vehicle \(k\). (Note that these constraints are independent knapsack constraints. Therefore, if we relax all the other constraints in the model, the Lagrangian relaxation becomes \(K\) independent knapsack problems.)

**EXERCISE 16.29.**

Using the Lagrangian relaxation method described in Section 16.6, we relax the degree constraint and solve a MST problem for the objective function \(L(\mu) = cx + \mu (\sum_{j=2}^{n} x_{1j} - k)\), with \(k = 8\). When \(\mu = 0\), the MST solution, which has a cost 10.8, contains the arcs \((1,12), (1,2), (1,6), (1,8), (2,4), (2,13), (2,3), (4,5), (8,7), (8,9), (8,10)\) and \((10,11)\). In this solution, the degree of node 1 is 4 and so is less than 8. Thus we need to decrease \(\mu\) so that the arcs incident to node 1 become more attractive.

At \(\mu = -0.2\) we obtain a new solution (i.e., the basis changes). We obtain this solution by adding arc \((1,4)\), removing arc \((2,4)\), adding arc \((1,10)\) and removing arc \((8,10)\) to the solution in which \(\mu = 0\). The solution now has the arcs \((1,12), (1,2), (1,4), (1,6), (1,8), (1,10), (2,13), (2,3), (4,5), (8,7), (8,9)\) and \((10,11)\). Node 1 now has degree 6 and the solution has cost 11.2. The Lagrangian value is \(11.2 - 0.2 (6 - 8) = 11.6\).

At \(\mu = -1.5\) we obtain a new MST solution (a new basis). We obtain this solution by adding arc \((1,3)\), removing arc \((2,3)\), adding arc \((1,9)\) and removing arc \((8,9)\) to the solution in which \(\mu = -0.2\). The new solution has the arcs \((1,12), (1,2), (1,4), (1,6), (1,8), (1,10), (1,3), (1,9), (2,13), (4,5), (8,7)\) and \((10,11)\). Node 1 has degree 8. The cost of the solution is 14.2 and the Lagrangian value is 14.2. Since the Lagrangian is a lower bound on the cost of the optimal solution, we have obtained the optimal solution.

Note that when we require the degree of node 1 to be 5 the Lagrangian relaxation becomes \(L(\mu) = cx + \mu (\sum_{j=2}^{n} x_{1j} - k)\), with \(k = 5\). In solving the relaxed problem, the value of \(k\) is irrelevant, and so is also irrelevant to the breakpoints where the basis changes, since it is a constant. Thus, we can use the results from the case \(k = 8\) and avoid resolving the problem. Another way to state this conclusion is that when we solved for the problem when node 1 had degree 8, we started with \(\mu = 0\) and obtained a solution in which node 1 had degree 4. For some Lagrange multiplier value between \(\mu = 0\) and the multiplier value when node 1 had degree 8, we must have also obtained an optimal solution to the problem with node 1 restricted to have degree 5. When \(\mu = 0\), the solution is the MST solution. When \(\mu = -0.2\), we add arc \((1,4)\) and delete arc \((2,4)\). This gives us a solution with arcs \((1,12), (1,2), (1,6), (1,8), (1,4), (2,13), (2,3), (4,5), (8,7), (8,9)\) and \((8,10)\) and \((10,11)\). In this solution, node 1 has degree 5. The cost is 11.0 and Lagrangian relaxation value (lower bound) is also 11 so we have the optimal solution.

**EXERCISE 16.31\#.**

(Note: In the definition of the polyhedron in part (a), \(x_{ij}\) should be \(x_{1j}\).)
(a) The discussion of Section 16.6 of the degree constrained MST shows that the relaxation gap is zero for any objective coefficient $c$. Thus the result proved in Exercise 16.30 applied with the single constraint $\sum_{j \neq 1}^n x_{1j} = k$ in place of $Ax = b$ shows that the extreme points of $\{x: x \in H(X) \text{ and } \sum_{j \neq 1}^n x_{1j} = k\}$ are integral.

(b) Theorem 13.10 shows that if we remove the cardinality constraint $\sum_{j \neq 1}^n x_{1j} = k$ from the formulation, the resulting polyhedron describes the incidence vectors $X$ of spanning tree solutions (i.e., its extreme point are $X$). But this means that the polyhedron described by $\{x: \sum_{(i, j) \in A} x_{ij} = n - 1; \sum_{(i, j) \in A(S)} x_{ij} \leq |S| - 1 \text{ for any set } S \text{ of nodes, } x_{ij} \geq 0\}$ is the same as $H(X)$ of part (a). Therefore the result now follows from part (a).

**Exercise 16.33.** (a) In the degree constrained minimum spanning tree problem, the degree of a node (let us call it the root node) is specified. If the degree of the root node is specified to be $k$, then it is possible for any subtree off the root node (that is any subtree we obtain by eliminating the root and its incident edges) to contain as many as $n - k$ nodes. In a capacitated minimum spanning tree problem, the capacities are described in quite a different way: the constraint is on the subtrees off the root node. A subtree of the root node can have no more than a specified number (say $u$) of nodes. The only way this restriction capacitates the degree of the root node is that we can have no fewer than $\lceil N/u \rceil$ subtrees. Thus, a star network (one in which each node is connected directly to the root node) is always feasible. Figure 16.5 illustrates the difference between these formulations.

![Given Network](image)

![Optimal Degree Constrained MST (K = 2)](image)

![Optimal CMST (u = 2)](image)

![Star is Always Feasible for CMST](image)

**Figure S16.5 Constrained and Capacitated Minimum Spanning Trees**

(b) Let $x_{ij} = 1$ if arc $(i, j)$ is in the capacitated minimum spanning tree and $x_{ij} = 0$ otherwise. Define $A(S)$ as in Exercise 16.31. We can formulate the capacitated minimum spanning tree problem as follows:
\[
\begin{align*}
\text{min} & \quad cx \\
\text{subject to} & \\
\sum_{(i, j) \in A} x_{ij} & = n - 1 \quad \text{(i)} \\
\sum_{(i, j) \in A(T)} x_{ij} & \leq |T| - 1 \quad \text{for any set } T \text{ of nodes} \quad \text{(ii)} \\
\sum_{(i, j) \in A(S)} x_{ij} & \leq |S| - \left\lceil \frac{|S|}{u} \right\rceil \quad \text{for any set } S \text{ with } 1 \notin S \quad \text{(iii)} \\
x_{ij} & \geq 0 \text{ and integer} \quad (16.33d)
\end{align*}
\]

The only new constraint in the formulation is (iii) which we call the packing constraint. This constraint says that any set of \(|S|\) nodes other than the root node must belong to at least \(\left\lceil \frac{|S|}{u} \right\rceil\) different components. Thus the solution can contain at most \(|S| - \left\lceil \frac{|S|}{u} \right\rceil\) edges joining these nodes.

(c) If we relax the packing constraints, the Lagrangian subproblem is a minimum spanning tree problem. Note that the number of packing constraints is exponential. Thus keeping track of all of them is computationally intensive and inefficient. Instead, we maintain only a subset of these constraints that are the “crucial” ones. In the Lagrangian relaxation procedure, we would solve the subproblem with the reduced set of packing constraints. Then we check to see if the Lagrangian solution is feasible. If so, then we would change the multipliers using the subgradient optimization procedure. If not, then we need to find a violated constraint (or constraints) and add it (them) to the objective function. In intermediate steps, we might delete constraints that have been inactive for a while, i.e., ones with Lagrangian multiplier zero.

**EXERCISE 16.35.** We can view the vehicle routing problem as a degree constrained minimum spanning tree problem in which each subtree off the root is a path plus an arc completing this path to a tour. Let us define \(x_{ij}\) to be 1 if arc \((i,j)\) belongs to any such path and let \(y_{ij}\) be the arc that makes the path into a tour. Therefore, \(y_{ij} = 1\) if arc \((i,1)\) is the arc that completes the tour. Using these variable definitions, we can formulate the identical vehicle customer routing problem as follows:

\[
\begin{align*}
\text{min} & \quad \sum_{(i, j) \in A} c_{ij} x_{ij} + \sum_{j \neq 1} c_{j1} y_{j1} \\
\text{subject to} & \\
\sum_{(i, j) \in A} x_{ij} & = n - 1 \quad \text{(i)} \\
\sum_{j \neq 1} x_{1j} & = K \quad \text{(ii)}
\end{align*}
\]
\[ \sum_{(i, j) \in A(T)} x_{ij} \leq |T| - 1 \text{ for any set } T \text{ of nodes} \quad (iii) \]

\[ \sum_{(i, j) \in A(S)} x_{ij} \leq |S| - \left[ \sum_{(i, j) \in A} (d_i + d_j) x_{ij} + \sum_{i \neq 1} d_i y_{i1} \right] \times 2u, \text{ for any set } S \text{ subject to } 1 \notin S \quad (iv) \]

\[ \sum_{j \neq i} x_{ij} + y_{i1} = 2, \text{ for each } i \neq 1 \quad (v) \]

\[ \sum_{i} x_{ij}, y_{i1} \geq 0 \text{ and integer.} \quad (vi) \]

All of the constraints are fairly straightforward except (iv). This is a generalized version of the packing constraint 16.34(c). It says any set of \(|S|\) customers must use at least \[ \left[ \sum_{(i, j) \in A} (d_i + d_j) x_{ij} + \sum_{i \neq 1} d_i y_{i1} \right] \times 2u \] vehicles. Therefore, any feasible solution can contain at most \(|S| - \left[ \sum_{(i, j) \in A} (d_i + d_j) x_{ij} + \sum_{i \neq 1} d_i y_{i1} \right] \times 2u\) type tree (\(x_{ij}\)) edges joining these customers.

Relaxing constraints (iv) and (v), we obtain a degree constrained spanning tree problem. We can therefore use the Lagrangian relaxation method by relaxing constraints (iv) and (v). Notice that the subproblem is easy to solve.