

ASSIGNMENT 2

Due Date: Wednesday 24th September 2003 5:00pm

Please deposit your assignment in Box 183 by the time indicated.

This assignment must be your own work; no discussion or collaboration is permitted. Please make an effort to write clearly, legibly and succinctly.

1. A company called “Breezy” sells airconditioners in number of states. Breezy is considering establishing manufacturing centres in some of the cities in those states.

The company has determined an annual demand d_j in each state j , and a transportation cost c_{ij} per airconditioner shipped from each city i to each state j . Any manufacturing centre can produce up to b airconditioners per year. The annual fixed cost f_i of operating a manufacturing centre in city i has also been determined.

- (a) Formulate the problem of deciding how to minimise the annual cost of supplying airconditioners as a Linear Integer Program. Make sure you think about whether your model is as *good* as it could be. Implement your model as an AMPL model file.

Solution:

```

set STATES;
set CITIES;

param FixedCost{CITIES};
param Demand{STATES};
param ShipCost{CITIES,STATES};

param Capacity;                                # capacity of manufacturing centres

      check: Capacity * card(CITIES) >= sum{j in STATES} Demand[j];
      # check sufficient capacity to meet demand

var x{CITIES} binary;                            # indicates if manufacturing centre established or not
var y{CITIES,STATES} >= 0;                       # amount shipped from centre at each city to each state

minimize TotalCost:                             # combined fixed and shipping costs
      sum{i in CITIES} FixedCost[i]*x[i]
      + sum{i in CITIES,j in STATES} ShipCost[i,j]*y[i,j];

subject to DeliverAll{j in STATES}:             # demand met in each state
      sum{i in CITIES} y[i,j] = Demand[j];

subject to CapConstr{i in CITIES}:              # capacity at each city not exceeded
      sum{j in STATES} y[i,j] <= Capacity*x[i];

subject to OpenCentre{i in CITIES,j in STATES}: # ensures model is good
      y[i,j] <= min(Capacity,Demand[j])*x[i];

```

Note that if the last constraint is omitted, then when solving with the data in the next part, lpsolve requires 9 branch-and-bound nodes to solve the problem, and the value of the LP relaxation is 4639285.71, whereas the optimal integer solution has value 5010000. Contrast this with the case where the last constraint is included. In this case the LP relaxation is 4922142.86 which is closer to the IP value, and fewer branch-and-bound nodes are needed (3 are used).

- (b) Breezy's business in Australia is in fact focussed on the states of central and western Australia, in particular on WA, SA and NT. The cities that are candidates for manufacturing centres are Perth, Adelaide and Alice Springs. The annual demand in each state is as follows: WA, 4,000; SA, 7,500; and NT, 2,000. The cost of shipping an airconditioner from each of these cities to each region is given in the table below.

	WA	SA	NT
Perth	\$350	\$400	\$420
Adelaide	\$450	\$260	\$320
Alice Springs	\$490	\$310	\$280

Table 1: Cost per airconditioner of shipping from city to state

Any manufacturing centre can produce up to 7,000 airconditioners per year. The

annual fixed cost of operating a manufacturing centre in each city is as follows: Perth, \$390,000; Adelaide, \$360,000; and Alice Springs, \$340,000.

Implement an AMPL data file for your AMPL model in part 1a based on this data. Use AMPL to solve the problem. What is the annual cost of supplying airconditioners? Where should manufacturing centers be established, and how many airconditioners should each manufacturing centre supply to each state per year?

Solution:

```
set STATES := WA SA NT;
set CITIES := Perth Adelaide Alice;

param FixedCost :=
    Perth 390000
    Adelaide 360000
    Alice 340000;

param Demand :=
    WA 4000
    SA 7500
    NT 2000;

param ShipCost:
    WA SA NT :=
Perth    350 400 420
Adelaide 450 260 320
Alice    490 310 280;

param Capacity := 7000;
```

Solving this integer program yields a total annual cost of \$5,010,000. Manufacturing centres are built at Adelaide and Perth. Adelaide supplies 7,000 airconditioners to SA, while Perth meets all other demand.

2. Consider again the problem in Question 1. State governments are vying to build the manufacturing base in their states.

Note: In this question, please do not be concerned with maintaining general models; feel free to make your model modifications specific to the questions asked.

- (a) The SA Government is particularly interested in expanding the manufacturing capability in SA, and so has asked Breezy to optimize their supply costs under the assumption that the Adelaide manufacturing centre could be built with sufficient production capacity to meet demand from all states combined. Assume that the fixed cost of operating the manufacturing centre is unchanged. Show how to modify your model and AMPL files from Question 1 to respond to the Government request, and use AMPL to solve the modified problem. In this case, what would the annual cost of supplying airconditioners be, where should manufacturing centers be established, and how many airconditioners should each manufacturing centre supply to each state per year?

Solution: The capacity of the Adelaide centre is now expanded to meet total demand, i.e. should be $4000 + 7500 + 2000 = 13500$ rather than 7000. Since capacity is now different for different centres, we change the capacity parameter in the model file to:

```
param Capacity{CITIES};
```

and the check on sufficient capacity to

```
check: sum{i in CITIES} Capacity[i] >= sum{j in STATES} Demand[j];
```

The body of the constraint on capacity, “CapConstr”, should change to

```
sum{j in STATES} y[i,j] <= Capacity[i]*x[i];
```

and the body of the strengthening constraint, “OpenCentre”, should change to

```
y[i,j] <= min(Capacity[i],Demand[j]) * x[i];
```

In the data file, the capacity now needs to be defined:

```
param Capacity :=
```

```
    Adelaide    13500
```

```
    Perth       7000
```

```
    Alice       7000;
```

Solving this integer program yields a total annual cost

of \$4,740,000. Manufacturing centres are still built at both Adelaide and Perth. Adelaide meets all the SA and NT demand: it supplies 7,500 airconditioners to SA and 2,000 airconditioners to NT. Perth meets all the WA demand of 4,000 airconditioners.

- (b) Pursuing an aggressive policy, the SA Government has promised to discount transport costs for Breezy airconditioners shipped from Adelaide to WA according to the following scheme. The first 1,000 airconditioners are shipped at current rates, the next 2,000 airconditioners are shipped at a 20% discount rate, and any further airconditioners are shipped at a 30% discount rate. So, for example, if Breezy is willing to ship 3,500 airconditioners to WA, the shipping cost will be $\$450(1000) + \$360(2000) + \$315(500)$, since the standard shipping cost is \$450 per airconditioner, 20% off that yields \$360 per airconditioner, and 30% off yields \$315 per airconditioner. Explain briefly why the use of integer programming, rather than linear programming, is appropriate to model this cost structure linearly. Now show how to modify your model and AMPL files from part 2a to yield an integer linear programming model for minimizing Breezy’s supply costs under the SA Government’s regime. Solve the problem using AMPL. Now what would the annual cost of supplying airconditioners be, where should manufacturing centres be established, and how many airconditioners should each manufacturing centre supply to each state per year?

Solution: The cost of shipping airconditioners from Adelaide to WA now is a concave piecewise linear function of the number of airconditioners shipped. To see this, observe that if n denotes the number of airconditions shipped from Adelaide to WA, (corresponding to the variable $y["Adelaide","WA"]$ in the AMPL program), then the cost function is

$$C(n) = \begin{cases} 450n, & 0 \leq n < 1000 \\ 90000 + 360n, & 1000 \leq n < 3000 \\ 225000 + 315n, & 3000 \leq n \leq 4000. \end{cases}$$


```

minimize TotalCost:                                     # total annual cost of meeting demand
  sum{i in CITIES} FixedCost[i]*x[i] +
  sum{i in CITIES,j in STATES: i <> "Adelaide" or j <> "WA"} ShipCost[i,j]*y[i,j] +
  sum{k in 1..NBreaksAdlWA} BreakCostAdlWA[k]*ShipWtAdlWA[k];

subject to DeliverAll{j in STATES}:
  sum{i in CITIES} y[i,j] = Demand[j];

subject to CapConstr{i in CITIES}:
  sum{j in STATES} y[i,j] <= Capacity[i]*x[i];

subject to OpenCentre{i in CITIES,j in STATES}:
  y[i,j] <= min(Capacity[i],Demand[j])*x[i];

subject to SetAdlWAShip:                               # amount shipped from Adelaide to WA is
  y["Adelaide","WA"] =                               # weighted combination of breakpoints
  sum{k in 1..NBreaksAdlWA} BreakAdlWA[k]*ShipWtAdlWA[k];

subject to SumOneAdlWAShipWt:                         # combination of breakpoints used is convex
  sum{k in 1..NBreaksAdlWA} ShipWtAdlWA[k] = 1;

subject to SumOneAdlWAShip:                           # can only use one cost regime
  sum{k in 1..(NBreaksAdlWA-1)} ShipAdlWA[k] = 1;

subject to AdjBreaksUsed{k in 1..NBreaksAdlWA}:      # only breakpoints at either end of the cost regime
  ShipWtAdlWA[k] <=                                  # selected can be used in the combination
  (if k <> NBreaksAdlWA then ShipAdlWA[k] else 0)
  +
  (if k <> 1 then ShipAdlWA[k-1] else 0);

```

```

set STATES := WA SA NT;
set CITIES := Perth Adelaide Alice;

param: FixedCost Capacity:=
Perth      390000    7000
Adelaide   360000    13500
Alice      340000    7000;

param Demand :=
WA         4000
SA         7500
NT         2000;

param ShipCost:
           WA      SA      NT :=
Perth     350     400     420
Adelaide  450     260     320
Alice     490     310     280;

param NBreaksAdlWA := 4;

param: BreakAdlWA BreakCostAdlWA :=
1         0         0
2         1000     450000
3         3000     1170000
4         4000     1485000;

```

Solving this mixed integer program yields a total annual cost of \$4,435,000. The only manufacturing centres are built is at Adelaide, which meets demand in all states.

- (c) Hearing about the SA Government's plans, the WA Government retaliates, with its own discount policy. It promises to halve the Perth manufacturing centre annual operating costs if it is built with capacity to meet demand from all states combined, and its production exceeds 7,000 airconditioners per year. Show how to modify your model and AMPL files from part 2b to reflect this new cost structure for the Perth centre, ensuring your model is still an integer linear program. Is the WA Government's strategy effective? Justify your answer.

Solution: To model this cost structure, we can introduce a binary variable to indicate whether or not the WA Government's discount will apply. We call the AMPL variable "PerExtra", and introduce it into the AMPL program via:

```
var PerExtra binary;
```

Now the cost function will need to have half the annual cost of the Perth manufacturing centre subtracted off if the WA Govt. discount applies, so we simply add

```
- 0.5*FixedCost["Perth"]*PerExtra
```

to the objective function. Now the WA Govt. needed two conditions to be met before giving the discount, so if PerExtra = 1 then we would expect the Perth manufacturing centre to have the capacity to supply all states combined, and we would expect the centre to be manufacturing at least 7,000 airconditions. We can model

the first requirement by exempting Perth from the normal capacity constraint via

```
subject to CapConstr{i in CITIES: i <> "Perth"}:  
    sum{j in STATES} y[i,j] <= Capacity[i]*x[i];
```

and adding the following constraint specifically for Perth capacity:

```
subject to WAGovtCapConstr:  
    sum{j in STATES} y["Perth",j] <= Capacity["Perth"]*x["Perth"]  
    + (sum{j in STATES} Demand[j] - Capacity["Perth"])*PerExtra;
```

We can model the second requirement with the following constraint:

```
subject to WAGovtMinProdConstr:  
    sum{j in STATES} y["Perth",j] >= 7000 - 7000*(1 - PerExtra);
```

Finally, we must of course ensure that if `PerExtra = 1` then the Perth manufacturing facility is open, i.e. that `x["Perth"] = 1`. This can be modelled by

```
subject to OpenPerthIfExtra:  
    PerExtra <= x["Perth"];
```

Solving the AMPL model with these modifications yields *no change* in the solution. In other words, the only manufacturing centre that would be opened would be in Adelaide, which would supply all demand. Thus the WA Government's strategy is *not* effective.

3. Let $G = (V, A)$ be an undirected graph with nodes $V = \{1, \dots, n\}$ and let c_a be the cost of each arc $a \in A$. The *minimum cost degree-constrained spanning tree problem* is defined to be the problem of finding a spanning tree for G , such that number of arcs incident to any vertex does not exceed some given value b , and such that the tree has minimum cost. To illustrate, consider the example below in Figure 3, in which G and c are given, and in which we take $b = 3$. G has nodes $V = \{1, 2, 3, 4, 5, 6, 7\}$ and arcs $A = \{\{1, 2\}, \{1, 5\}, \{1, 6\}, \{2, 3\}, \{2, 4\}, \{2, 5\}, \{2, 6\}, \{3, 5\}, \{4, 5\}, \{5, 6\}, \{5, 7\}, \{6, 7\}\}$. The costs on each are given by $c_{\{1,2\}} = 2$, $c_{\{1,5\}} = 4$, $c_{\{1,6\}} = 8$, $c_{\{2,3\}} = 9$, $c_{\{2,4\}} = 11$, $c_{\{2,5\}} = 3$, $c_{\{2,6\}} = 6$, $c_{\{3,5\}} = 12$, $c_{\{4,5\}} = 14$, $c_{\{5,6\}} = 7$, $c_{\{5,7\}} = 4$ and $c_{\{6,7\}} = 10$.

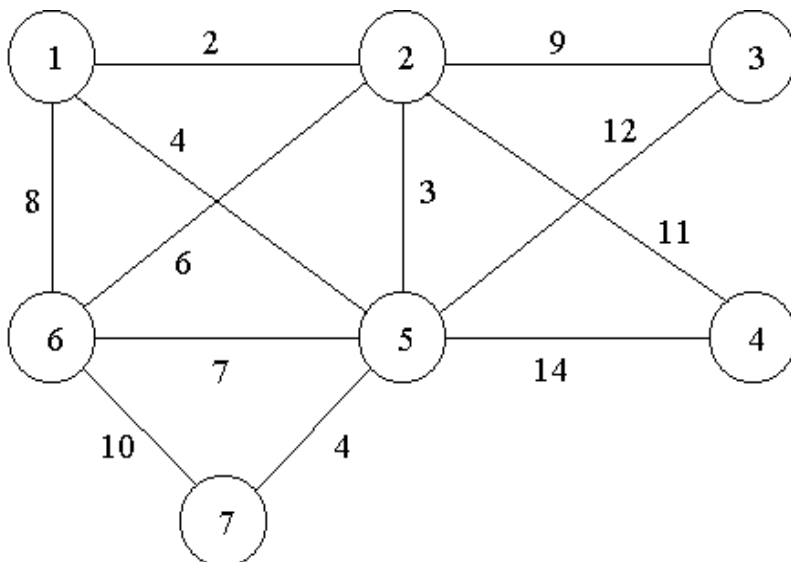


Figure 1: Network with costs on arcs

- (a) Observe that if the degree constraint is ignored, then the minimum cost degree-constrained spanning tree problem is simply a minimum cost spanning tree problem. Verify, using Kruskal's algorithm, and showing all working, that the minimum cost spanning tree for the instance given in Figure 3 is the tree with arcs $\{1, 2\}$, $\{2, 3\}$, $\{2, 4\}$, $\{2, 5\}$, $\{2, 6\}$, and $\{5, 7\}$. What is the cost of this tree, and how many arcs in the tree are incident to each node? Explain briefly why the degree constraint for $b = 3$ is violated by node 2. Is the degree constraint violated by any node other than node 2?

Explain briefly why the cost of this tree is a *lower bound* on the cost of the minimum cost degree-constrained spanning tree. [Hint: explain why the minimum cost spanning tree problem is a *relaxation* of the minimum cost degree-constrained spanning tree problem.]

Solution:

Arc Considered	Cost	Added to Tree? (Y/N)
{1,2}	2	Y
{2,5}	3	Y
{1,5}	4	N - would form cycle (1,2,5,1)
{5,7}	4	Y
{2,6}	6	Y
{5,6}	7	N - would form cycle (2,5,6,2)
{1,6}	8	N - would form cycle (1,2,6,1)
{2,3}	9	Y
{6,7}	10	N - would form cycle (2,6,7,5,2)
{2,4}	11	Y

We can stop at this point, because we have added six arcs to the tree; there are only seven nodes in the graph so the tree must now span the graph. The six arcs added are exactly the six arcs suggested in the question.

The cost of the tree is $2 + 3 + 4 + 6 + 9 + 11 = 35$. The number of tree arcs incident to each vertex are given in the table below.

Vertex	1	2	3	4	5	6	7
Degree in tree	1	5	1	1	2	1	1

For $b = 3$, the degree constraint asks that the degree of every vertex in the tree is no more than 3. Since the degree of vertex 2 in the tree is $5 > 3 = b$, the degree constraint is violated by vertex 2. All other vertices have degree in the tree either 1 or 2, which are no more than $b = 3$, so all other vertices satisfy the degree constraint. Any tree which is feasible for the minimum cost degree-constrained spanning tree problem is a spanning tree for the graph, and so is feasible for the minimum cost spanning tree problem on the same graph. The cost of a tree in either problem is identical, so the minimum cost spanning tree in the graph is either the minimum cost degree-constrained spanning tree in the graph, in which case the problems have identical value, or it is some other spanning tree having no greater cost. Thus the value of the minimum cost spanning tree in a graph gives a lower bound on the value of the minimum cost degree-constrained spanning tree in that graph.

- (b) Develop a branch-and-bound method for solving the minimum cost degree-constrained spanning tree problem, in general. Use lower bounds given by simple minimum cost spanning trees. [Hint: each node of the branch-and-bound tree will be associated with some *subgraph* of G .] Clearly define the branching rule in your branch-and-bound algorithm, and explain its effect on the minimum cost spanning tree problem to be solved at each child node defined by the branching rule. [Hint: the minimum spanning tree may violate the degree constraint, so use the branching rules to correct the violation, in such a way the the new problem at each node of the branch-and-bound tree is again a simple minimum cost spanning tree problem.] Make clear

how you will detect a leaf node of the branch-and-bound tree. You may find it helpful to use the following notation: if $T(\hat{G}) \subseteq A$ is the set of arcs in a minimum cost spanning tree of some subgraph \hat{G} of G , then use $T_i(\hat{G}) = \{\{i, j\} \in T(\hat{G})\}$ to denote the set of arcs in $T(\hat{G})$ which are incident to node i .

Solution: At each node of the branch-and-bound tree, we solve a minimum cost spanning tree problem on a graph $\hat{G} = (V, \hat{A})$ with costs \hat{c} , where V is the set of vertices in the original graph $G = (V, A)$, $\hat{A} \subseteq A$, and $\hat{c}_a = c_a$ for all $a \in \hat{A}$. At the root node, $\hat{A} = A$. Consider a node of the branch-and-bound tree defined by $\hat{G} = (V, \hat{A})$ and \hat{c} . Now either the graph is not connected, or we may find the minimum cost spanning tree given by the set of arcs $T(\hat{G})$. In the first case, the node is a *leaf node* and is *infeasible*. In the second case, as discussed in the previous part, the cost of $T(\hat{G})$, given by $\sum_{a \in T(\hat{G})} \hat{c}_a$ is a lower bound on the minimum cost

degree-constrained spanning tree for \hat{G} . Furthermore, if $T(\hat{G})$ satisfies the degree constraints, then it must be the minimum cost degree-constrained spanning tree in \hat{G} . Since \hat{G} is a subgraph of G having the same vertex set, $T(\hat{G})$ must also be a feasible degree-constrained spanning tree of G . Thus the node is a *leaf node*, $T(\hat{G})$ is a *feasible* tree for the original problem, and its cost can be used to update the upper bound, if less than the current upper bound. Otherwise, if $T(\hat{G})$ does *not* satisfy the degree constraints, there must exist vertex $i \in V$ with $|T_i(\hat{G})| > b$. We may choose any such i , but a good heuristic might be to choose i with maximum degree in the tree, i.e. to maximize $|T_i(\hat{G})|$. Clearly any feasible tree cannot use every arc in $T_i(\hat{G})$, since there are more than b of them. Say $n_i(\hat{G}) = |T_i(\hat{G})|$ and $T_i(\hat{A}) = \{a_1, a_2, \dots, a_{n_i(\hat{G})}\}$. So a valid branching rule would be to create $n_i(\hat{G})$ child nodes, with the subgraph at the j th node given by $(V, \hat{A} \setminus \{a_j\})$ for each $j = 1, \dots, n_i(\hat{G})$. (Note: stronger branching rules are possible. If you are interested in hearing more about these, please ask!)

Note that correct branch-and-bound methods can be based on quite different branching rules. For example, is it possible to create a child node for each combination of $n_i(\hat{G})$ choose b arcs incident to vertex i , with all arcs incident to i but not in that combination deleted in the graph at the child node. Whilst this is technically correct, it is not a highly desirable branching scheme in general, as the number of child nodes is exponential in the problem parameters (in particular, in b). This property is generally to be avoided, and branch-and-bound trees with modest numbers of child nodes preferred.

- (c) Apply your branch-and-bound algorithm to the problem instance given in Figure 3, with $b = 3$. Use an initial upper bound of ∞ , which may be updated once your branch-and-bound search yields a feasible solution. Use the best bound tree search strategy and label all branch-and-bound nodes in the order in which you created them. Clearly indicate where the branch-and-bound tree was pruned, which branch-and-bound nodes are leaf nodes, and why.

Solution: We give the nodes of the branch-and-bound tree in the table below. The left hand column indicates the depth of the node in the branch-and-bound tree. “Arcs” indicates the arcs in the graph at each node, i.e. the set \hat{A} . “MST” gives the set of arcs in the minimum cost spanning tree at that node. The lower bound “LB” given is, of course, the cost of the MST, while the upper bound, “UB”, is the

upper bound recorded in the branch-and-bound procedure at the point at which the node is processed. Note that at no node is the network disconnected, so there are no leaf nodes which are infeasible, i.e. for which a MST does not exist. Thus all nodes are either degree infeasible, in which case they are either pruned or branch out into child nodes, or they are feasible leaf nodes. Note also that this branching process is somewhat inefficient. For example, the graph at node 8 is identical to the graph at 12; there is no point in evaluating both. Another example is that since $\{2, 4\}$ is omitted from the graph at node 13, the LB at that node cannot be better than that at node 5, which has already been pruned; again there is no point in evaluating node 13. These flaws can be corrected with a *stronger* branching scheme, which *partitions* the solution space. The branching scheme given here does not, e.g. any tree that does not contain either arc $\{2, 5\}$ or arc $\{2, 6\}$ is feasible for node 3 as well as for node 4. (If you are interested in hearing more about stronger branching rules, please ask!)

The optimal solution is the tree given at nodes 8 and 12 of the branch-and-bound tree, and it has cost 37.

Nodes								
1	<p>Node 1 Root node Arcs: A MST: $\{\{1, 2\}, \{2, 3\}\}$ $\{2, 4\}, \{2, 5\}$ $\{2, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 2 LB = 35 UB = ∞</p>							
2	<p>Node 2 Parent node 1 Arcs: $A \setminus \{1, 2\}$ MST: $\{\{1, 5\}, \{2, 3\}\}$ $\{2, 4\}, \{2, 5\}$ $\{2, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 2 LB = 37 UB = ∞ Pruned by node 8</p>	<p>Node 3 Parent node 1 Arcs: $A \setminus \{2, 6\}$ MST: $\{\{1, 2\}, \{2, 3\}\}$ $\{2, 4\}, \{2, 5\}$ $\{5, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 2 LB = 36 UB = ∞</p>	<p>Node 4 Parent node 1 Arcs: $A \setminus \{2, 5\}$ MST: $\{\{1, 2\}, \{2, 3\}\}$ $\{2, 4\}, \{1, 5\}$ $\{2, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 2 LB = 36 UB = ∞</p>	<p>Node 5 Parent node 1 Arcs: $A \setminus \{2, 4\}$ MST: $\{\{1, 2\}, \{2, 3\}\}$ $\{4, 5\}, \{2, 5\}$ $\{2, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 2 LB = 38 UB = ∞ Pruned by node 8</p>	<p>Node 6 Parent node 1 Arcs: $A \setminus \{2, 3\}$ MST: $\{\{1, 2\}, \{3, 5\}\}$ $\{2, 4\}, \{2, 5\}$ $\{2, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 2 LB = 38 UB = ∞ Pruned by node 8</p>			
3	<p>Node 7 Parent node 3 Arcs: $A \setminus \{\{2, 6\}, \{1, 2\}\}$ MST: $\{\{1, 5\}, \{2, 3\}\}$ $\{2, 4\}, \{2, 5\}$ $\{5, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 4 LB = 38 UB = ∞ Pruned by node 8</p>	<p>Node 8 Parent node 3 Arcs: $A \setminus \{\{2, 6\}, \{2, 5\}\}$ MST: $\{\{1, 5\}, \{2, 3\}\}$ $\{2, 4\}, \{1, 2\}$ $\{5, 6\}, \{5, 7\}$ MST degree feasible LB = 37 UB = 37 Feasible leaf</p>	<p>Node 9 Parent node 3 Arcs: $A \setminus \{\{2, 6\}, \{2, 4\}\}$ MST: $\{\{1, 2\}, \{2, 3\}\}$ $\{2, 5\}, \{4, 5\}$ $\{5, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 4 LB = 39 UB = 37 Pruned by node 8</p>	<p>Node 10 Parent node 3 Arcs: $A \setminus \{\{2, 6\}, \{2, 3\}\}$ MST: $\{\{1, 2\}, \{2, 4\}\}$ $\{2, 5\}, \{3, 5\}$ $\{5, 6\}, \{5, 7\}$ Degr. constr. not satisfied vert. 4 LB = 39 UB = 37 Pruned by node 8</p>	<p>Node 11 Parent node 4 Arcs: $A \setminus \{\{2, 5\}, \{1, 2\}\}$ MST: $\{\{1, 5\}, \{2, 3\}\}$ $\{2, 4\}, \{2, 6\}$ $\{5, 6\}, \{5, 7\}$ MST degree feasible LB = 41 UB = 37 Feasible leaf</p>	<p>Node 12 Parent node 4 Arcs: $A \setminus \{\{2, 5\}, \{2, 6\}\}$ MST: $\{\{1, 5\}, \{2, 3\}\}$ $\{2, 4\}, \{1, 2\}$ $\{5, 6\}, \{5, 7\}$ MST degree feasible LB = 37 UB = 37 Feasible leaf</p>	<p>Node 13 Parent node 4 Arcs: $A \setminus \{\{2, 5\}, \{2, 4\}\}$ MST: $\{\{1, 2\}, \{2, 3\}\}$ $\{2, 6\}, \{4, 5\}$ $\{1, 5\}, \{5, 7\}$ MST degree feasible LB = 39 UB = 37 Feasible leaf</p>	<p>Node 14 Parent node 4 Arcs: $A \setminus \{\{2, 5\}, \{2, 3\}\}$ MST: $\{\{1, 2\}, \{2, 4\}\}$ $\{2, 6\}, \{3, 5\}$ $\{1, 5\}, \{5, 7\}$ MST degree feasible LB = 39 UB = 37 Feasible leaf</p>