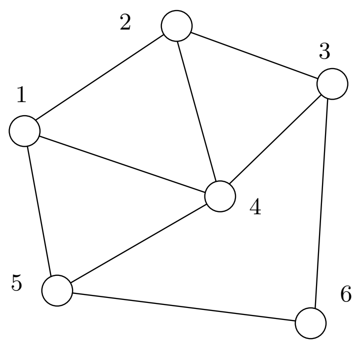
1 Graphs and AMPL

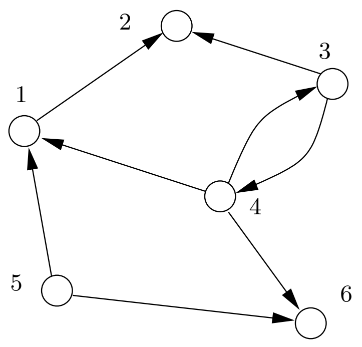
First of all, some info on how to define a graph, directed or nondirected, in AMPL. An **undirected** graph $G = (V, E)$ such as that in Figure 1a can be implemented as follows:

```
param n;
set V := 1..n;
set E within {i in V, j in V: i < j};
# variables, obj and constraints...
data;
param n := 6;
set E := (1,2) (1,4) (1,5) (2,3) (2,4) (3,4) (3,6) (4,5) (5,6);
```

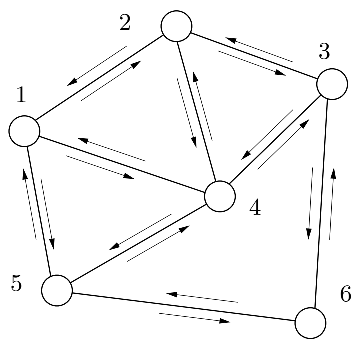
Why the "`i < j`" constraint in defining the edge set E? Because E is formally a set of *subsets*, more specifically a set of subsets of two elements. AMPL may very well handle a set of sets, but we just want to keep it simple[1]: we just assume that a set of two elements can be defined as a *pair* of elements. With "`{i in V, j in V}`", AMPL will consider any pair of elements of V.



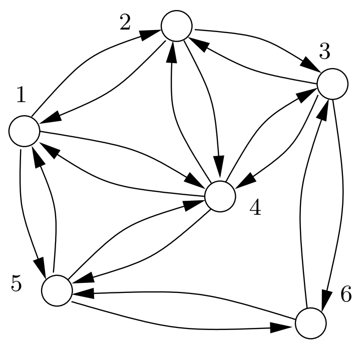
(a) An undirected graph.



(b) A directed graph.



(c) Directions on an undirected graph.



(d) A directed graph.

Fig. 1.

Now, a set of two elements and a pair are very similar, but while the two sets $\{i, j\}$ and $\{j, i\}$ are the same thing, the two pairs (i, j) and (j, i) are **not**. Thus, we

[1] Believe me, you don't want to see the "set of sets" formulation. But if you really do, I'll be very happy to show it to you.

assume that a set of two elements is equivalent to a pair with the first element smaller than the second one (a set, for instance, doesn't have a "first" and a "second" element). That is, we impose the constraint "`i < j`" when defining these pairs. When we want to consider edge $\{2,5\}$, we tell AMPL to consider the *pair* `(2,5)`. AMPL will tell you that the pair `(5,2)` doesn't belong to `E`, but that's because we assume that the pair `(5,2)` does *not* correspond to the set $\{2,5\} = \{5,2\}$, while `(2,5)` does.

This tricks AMPL into thinking that `E` is a set of pairs, but it should not trick us. We know that what AMPL sees as the pair "`(2,5)`" is indeed a *set*, $\{2,5\}$, in our paper world. AMPL thinks a subset of E is simply a set of pairs, and that an element of E is a pair. Now, what is the subset F of edges incident[2] to node 3, for example? On a piece of paper, we would write

$$F = \{\{i,j\} \in E : i = 3\},$$

while in AMPL we need to identify the pairs where 3 is either the first or the second element: we can do so by either telling AMPL to take those pairs (i,j) where $i = 3$ or $j = 3$. The following two lines are equivalent:

```
set F := {i in V, j in V: (i,j) in E and (i=3 or j=3)};
set F := {(i,j) in E: i=3 or j=3};
```

and AMPL will display it as

```
ampl: display F;
set F := (2,3) (3,4) (3,6);
```

where we know that the first condition (`i=3`) matched the second and third pair, and the second condition (`j=3`) matched the first pair.

Directed graphs

When we have a directed graph, we are more at ease as we don't need to trick AMPL into anything. A directed graph (or digraph) $G = (V,A)$ has a set of nodes and a set A of *arcs*. Each arc is defined as an ordered pair (i,j) where i is the "tail" and j is the "head" of the arc. A digraph $G = (V,A)$ such as the one in Figure 1b can be defined in AMPL similarly to the nondirected graph:

```
param n;
set V := 1..n;
set A within {i in V, j in V: i != j};
# variables, obj and constraints...
data;
param n := 6;
set A := (1,2) (3,2) (3,4) (4,3) (4,1) (4,6) (5,1) (5,6);
```

Notice that the condition on the pairs changed: it is now "`i != j`", which means i is only required to be different, not smaller, than j. How do we get all nodes such that there is an arc *from* node 3 to them?

```
ampl: set H = {(i,j) in A: i=3};
ampl: display H;
set H := (3,2) (3,4);
```

How about all nodes that have an arc *to* node 3?

```
ampl: set H2 = {(i,j) in A: j=3};
ampl: display H2;
set H2 := (4,3);
```

[2] "incident in i" = "with i as an endnode".

Directions on an undirected graph

What happens when we have an undirected graph $G = (V, E)$, but we have both variables defined on the edges of E and variables that use the two directions on each edge? That is, we have a graph with edges $\{i, j\}$, but we may define a quantity in the direction $i \to j$ and another in the direction $j \to i$. On paper, that is, from the modeling standpoint, that amounts to defining, for a set E of subsets, a set of pairs A as follows: $A = \{(i, j) : i \in V, j \in V, \{i, j\} \in E\}$. Strange: it seems we have one pair for each edge, and we should have two! Well, for the graph in 1c, A contains pair $(1, 4)$ and $(4, 1)$, because both $\{1, 4\}$ and $\{4, 1\}$ are in E – they are actually the same subset.

We use a similar trick with AMPL. After we define the set of edges with

```
set E within {i in V, j in V: i < j};
```

we define a set `ED` with a pair `(i,j)` and a pair `(j,i)` for each edge `(i,j)`:

```
set ED := {i in V, j in V: ((i,j) in E) or ((j,i) in E)};
```

What's this? It is a set defined on E, so that in the data file we only need to specify E, not `ED`. Also, it contains all pairs `(i,j)` such that either `(i,j)` or `(j,i)` are, according to AMPL, pairs of the set of edges. If we define `ED` after defining `E` in the AMPL code at the beginning of the document, we get:

```
ampl: display ED;
set ED :=
(1,2)  (1,5)  (2,3)  (3,2)  (3,6)  (4,2)  (4,5)  (5,4)  (6,3)
(1,4)  (2,1)  (2,4)  (3,4)  (4,1)  (4,3)  (5,1)  (5,6)  (6,5);
```

which looks like the directed graph in Figure 1d, where each edge $\{i, j\}$ of graph in Fig. 1a is turned into two arcs, (i, j) and (j, i).

2 The Minimum Spanning Tree (MST) problem

Problem. Given a graph $G = (V, E)$ and a function $c : E \to \mathbb{R}_+$, find the subset S of E that minimizes $\sum_{\{i,j\} \in S} c_{ij}$ and such that for any two nodes k and l there is a path connecting k and l using only nodes in S.

Solution. There are two ways to approach this, as we've seen in class. The first one uses flow variables, the second uses the concept of cut. I will go over the first model only. Let us define binary variables y_{ij}, equal to 1 if $\{i, j\}$ is included in the solution, 0 otherwise. Now we can already write the objective function,

$$\sum_{\{i,j\} \in E} c_{ij} y_{ij},$$

to be minimized. What constraints do we need? We want to make sure that there is, for each pair of nodes k and l, a path from k to l that uses the edges of E for which $y_{ij} = 1$. That is, we could look for that path, for any pair (k, l), and impose that it uses the edges for which $y_{ij} = 1$.

Consider again the graph in Fig. 1, and node pair $(1, 6)$. We don't know what edges will be included in the solution S, but it is necessary that the path from 1 to 6 uses the edges of S. Note that "a path from 1 to 6" and "a path from 6 to 1" are equivalent here, as an edge included in S can be crossed in the two directions on a path between 1 and 6, and therefore on a path from 6 to 1, too.

That allows us to simplify things a little: instead of ensuring, **for any pair** (k, l) of nodes, that a path exists between k and l, we pick a node r (say node 2), that we call *root*, and ensure that, **for any node** k, a path exists between r and k – and that implies that a path also exists between k and r, which is the same from r to k, only in the opposite direction. If we can ensure a path between the root node r and any k, then we can ensure a path between any pair of nodes k and l: simply concatenate the paths $k \to r$ and $r \to l$.

OK, but we still have to ensure that condition. Suppose we need a path from our root node, $r = 2$, to $k = 6$. We could define *flow* variables x_{ij} and x_{ji} for the two directions of an edge $\{i, j\}$ of E. These flow variables are binary as $x_{ij} = 1$ if the path from 2 to 6 uses edge $\{i, j\}$ from i to j, 0 otherwise. Also, since we want it to be a flow from 2 to 6, it should satisfy the flow constraints at 2, i.e., there should be a positive flow balance,

$$\text{(node 2)} \qquad x_{21} + x_{23} + x_{24} - x_{12} - x_{32} - x_{42} = 1$$

and at all intermediate nodes:

$$\begin{array}{lllll}
\text{(node 1)} & x_{12} + x_{14} + x_{15} & -x_{21} - x_{41} - x_{51} & = 0 \\
\text{(node 3)} & x_{32} + x_{34} + x_{36} & -x_{23} - x_{43} - x_{63} & = 0 \\
\text{(node 4)} & x_{41} + x_{42} + x_{43} + x_{45} & -x_{14} - x_{24} - x_{34} - x_{54} & = 0 \\
\text{(node 5)} & x_{51} + x_{54} + x_{56} & -x_{15} - x_{45} - x_{65} & = 0.
\end{array}$$

We don't need to ensure any flow condition at the destination node, 6, because we know that if a flow enters the network at node 2 and conserves at all nodes but 6, it must exit at 6. We can re-write the above constraints as follows:

$$\begin{array}{ll}
\text{(node 2)} & \sum_{j \in V:\{2,j\} \in E} x_{2j} - \sum_{j \in V:\{2,j\} \in E} x_{j2} = 1 \\
\text{(node 1)} & \sum_{j \in V:\{1,j\} \in E} x_{1j} - \sum_{j \in V:\{1,j\} \in E} x_{j1} = 0 \\
\text{(node 3)} & \sum_{j \in V:\{3,j\} \in E} x_{3j} - \sum_{j \in V:\{3,j\} \in E} x_{j3} = 0 \\
\text{(node 4)} & \sum_{j \in V:\{4,j\} \in E} x_{4j} - \sum_{j \in V:\{4,j\} \in E} x_{j4} = 0 \\
\text{(node 5)} & \sum_{j \in V:\{5,j\} \in E} x_{5j} - \sum_{j \in V:\{5,j\} \in E} x_{j5} = 0
\end{array}$$

or even more succintly

$$\begin{array}{llll}
\text{(node 2)} & \sum_{j \in V:\{2,j\} \in E} x_{2j} - \sum_{j \in V:\{2,j\} \in E} x_{j2} = 1 & \\
\text{(nodes 1,3,4,5)} & \sum_{j \in V:\{i,j\} \in E} x_{ij} - \sum_{j \in V:\{i,j\} \in E} x_{ji} = 0 & \forall i \in \{1, 3, 4, 5\}
\end{array}$$

This gives a flow from the root node to node 6. We should write similar constraints for a path from the root node to node 5, and then from 2 to 4, from 2 to 3, and from 2 to 1. Let's rewrite the above constraints to reflect that those variables are used for the path to node 6, and not for the other paths:

$$\begin{array}{llll}
\text{(node 2)} & \sum_{j \in V:\{2,j\} \in E} x^6_{2j} - \sum_{j \in V:\{2,j\} \in E} x^6_{j2} = 1 & \\
\text{(nodes 1,3,4,5)} & \sum_{j \in V:\{i,j\} \in E} x^6_{ij} - \sum_{j \in V:\{i,j\} \in E} x^6_{ji} = 0 & \forall i \in \{1, 3, 4, 5\}
\end{array}$$

Now let's write constraints for the path from the root node, 2, to node 4:

$$\begin{array}{llll}
\text{(node 2)} & \sum_{j \in V:\{2,j\} \in E} x^4_{2j} - \sum_{j \in V:\{2,j\} \in E} x^4_{j2} = 1 & \\
\text{(nodes 1,3,5,6)} & \sum_{j \in V:\{i,j\} \in E} x^4_{ij} - \sum_{j \in V:\{i,j\} \in E} x^4_{ji} = 0 & \forall i \in \{1, 3, 5, 6\}
\end{array}$$

and for the remaining nodes other than the root, i.e., 1, 3, and 5:

$$\begin{array}{llll}
\text{(node 2)} & \sum_{j \in V:\{2,j\} \in E} x^1_{2j} - \sum_{j \in V:\{2,j\} \in E} x^1_{j2} = 1 & \\
\text{(nodes 3,4,5,6)} & \sum_{j \in V:\{i,j\} \in E} x^1_{ij} - \sum_{j \in V:\{i,j\} \in E} x^1_{ji} = 0 & \forall i \in \{3, 4, 5, 6\}
\end{array}$$

$$\begin{array}{lll}
\text{(node 2)} & \sum_{j\in V:\{2,j\}\in E} x^3_{2j} - \sum_{j\in V:\{2,j\}\in E} x^3_{j2} = 1 & \\
\text{(nodes 1,4,5,6)} & \sum_{j\in V:\{i,j\}\in E} x^3_{ij} - \sum_{j\in V:\{i,j\}\in E} x^3_{ji} \; = 0 & \forall i \in \{1,4,5,6\}
\end{array}$$

$$\begin{array}{lll}
\text{(node 2)} & \sum_{j\in V:\{2,j\}\in E} x^5_{2j} - \sum_{j\in V:\{2,j\}\in E} x^5_{j2} = 1 & \\
\text{(nodes 1,3,4,5)} & \sum_{j\in V:\{i,j\}\in E} x^5_{ij} - \sum_{j\in V:\{i,j\}\in E} x^5_{ji} \; = 0 & \forall i \in \{1,3,4,5\}
\end{array}$$

It actually makes sense to rewrite all constraints above as follows:

$$\begin{array}{lll}
\text{(node 2)} & \sum_{j\in V:\{2,j\}\in E} x^k_{2j} - \sum_{j\in V:\{2,j\}\in E} x^k_{j2} = 1 & \forall k \in \{1,3,4,5,6\} = V \setminus \{2\} \\
\text{(nodes} \neq k) & \sum_{j\in V:\{i,j\}\in E} x^k_{ij} - \sum_{j\in V:\{i,j\}\in E} x^k_{ji} \; = 0 & \forall i \in V : 2 \neq i \neq k, \;\; \forall k \in V \setminus \{2\}
\end{array}$$

Well, now that we have a ton of x^k_{ij} what do we do with them? We said that there can be a flow x^k_{ij} on edge $\{i,j\}$ from i to j on its path to k only if the relative $y_{ij} = 1$. That means

$$\begin{array}{lll}
x^k_{ij} \leq y_{ij} & \forall \{i,j\} \in E, \forall k \in V \setminus \{k\} & \text{(direction from } i \text{ to } j) \\
x^k_{ji} \leq y_{ij} & \forall \{i,j\} \in E, \forall k \in V \setminus \{k\} & \text{(direction from } i \text{ to } j)
\end{array}$$

Phew! The final model is (let's replace node 2 with the more general r):

$$\begin{array}{ll}
\min \sum_{\{i,j\}\in E} c_{ij} y_{ij} & \\
\quad \sum_{j\in V:\{r,j\}\in E} x^k_{rj} - \sum_{j\in V:\{r,j\}\in E} x^k_{jr} = 1 & \forall k \in V \setminus \{r\} \\
\quad \sum_{j\in V:\{i,j\}\in E} x^k_{ij} - \sum_{j\in V:\{i,j\}\in E} x^k_{ji} = 0 & \forall i \in V : r \neq i \neq k, \;\; \forall k \in V \setminus \{r\} \\
\quad x^k_{ij} \leq y_{ij} & \forall \{i,j\} \in E, \forall k \in V \setminus \{r\} \\
\quad x^k_{ji} \leq y_{ij} & \forall \{i,j\} \in E, \forall k \in V \setminus \{r\} \\
\quad x^k_{ij}, x^k_{ji} \in \{0,1\} & \forall \{i,j\} \in E, \forall k \in V \setminus \{r\} \\
\quad y_{ij} \in \{0,1\} & \forall \{i,j\} \in E.
\end{array}$$

How do we implement this stuff in AMPL? Easy, Section 1 gives some hints on all of this. Adding an index to a formula is as easy as adding a set to the definition of a variable in AMPL.

```
param n; # number of nodes

set V=1..n;
set E within {i in V, j in V: i<j};
set ED     = {i in V, j in V: (i,j) in E or (j,i) in E};

param r in V;

param c {E};

var y {E} binary;
var x {(i,j) in ED, k in V: k != r} binary;

minimize totalcost: sum {(i,j) in E} c [i,j] * y [i,j];

conservation_root {k in V: k != r}:
  sum {j in V: (r,j) in ED} x [r,j,k] -
  sum {j in V: (j,r) in ED} x [j,r,k] = 1;

conservation_intermed {i in V, k in V: (i!=r) and (i!=k) and (k!=r)}:
  sum {j in V: (i,j) in ED} x [i,j,k] -
  sum {j in V: (j,i) in ED} x [j,i,k] = 0;
```

```
use_edge_ij {(i,j) in E, k in V: (k!=r)}:
  x [i,j,k] <= y [i,j];

use_edge_ji {(i,j) in E, k in V: (k!=r)}:
  x [j,i,k] <= y [i,j];
```

The data file for the graph in Fig. 1 is (I placed random values for c_{ij}):

```
param n = 6;
set E := (1,2) (1,4) (1,5) (2,3) (2,4) (3,4) (3,6) (4,5) (5,6);

param r = 2;

param c :=
1 2 21
1 4 42
1 5 22
2 3 35
2 4 49
3 4 10
3 6 13
4 5 40
5 6 21;
```

A little exercise: do you think the condition

$$x^k_{ij} + x^k_{ji} \le y_{ij} \quad \forall \{i,j\} \in E, \quad \forall k \in V \setminus \{k\}$$

makes any sense? What does it mean? Do you think it's a valid constraints, that is, does it exclude any feasible solution?

3 The Network Design problem

Problem. We want to design a telecommunication network that connects a set of cities using their railway network, defined as a graph $G = (V, E)$. Installing a unit of network capacity on edge $\{i, j\} \in E$ (for instance, a fiber optics cable) costs c_{ij}. Once installed, it carries up to U Mb/s of data traffic in both directions (U Mb/s $i \to j$ and U Mb/s $j \to i$). For any *ordered* pair (k, l) of cities in a set P, there is a planned amount of data traffic from k to l denoted as d_{kl}.

Determine how much capacity to install on all edges of G in order to satisfy all traffic demands $(k, l) \in P$ while minimizing total installation cost.

Solution. Let's assume that the communication from k to l is carried out on one single path from k to l, and let also remember that the data from k to l may follow a different path than the data from l to k. This time we don't have to choose what edges to include in the solution, but how much capacity should be installed on each edge.

On an edge $\{i, j\}$ with one fiber, we know that at most U Mb/s of traffic can be accommodated. We don't know what city pairs are using that edge, but depending on the variables that tell us who is using what edge, we can compute the total traffic being routed on $\{i, j\}$.

As the communication for a city pair (k, l) is on a single path, we can use binary flow variables and constraints similar to those for the MST problem. Consider the example in Fig. 1a, and assume that the set of ordered demand pairs is $P = \{(1, 3), (1, 6), (3, 1), (3, 5)\}$ and the data to be transmitted for each node pair is $d_{13} = 12$Mb/s, $d_{16} = 41$Mb/s, $d_{31} = 29$Mb/s, and $d_{35} = 31$Mb/s. Finally, let's assume $U = 50$. A possible solution, showing how these node pairs exchange data between each other, is in Fig. 2. The demand for pair $(1, 3)$ is routed on edges $\{1, 4\}$ and $\{4, 3\}$, that for pair $(1, 6)$ is routed on $\{1, 4\}$, $\{4, 3\}$, and $\{3, 6\}$, and so on.

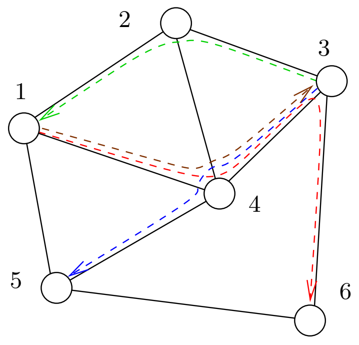


Fig. 2.

Let us define a binary variable x_{ij}^{kl}: it is 1 if edge $\{i, j\}$ is used, in the direction from i to j, to carry traffic that originates in k and will end in l, and 0 otherwise. For the solution above, we have all x variables equal to zero except for the following:

$$\begin{aligned}
&\text{node pair } (1,3): x_{14}^{13} = x_{43}^{13} = 1;\\
&\text{node pair } (1,6): x_{14}^{16} = x_{43}^{16} = x_{36}^{16} = 1;\\
&\text{node pair } (3,1): x_{32}^{31} = x_{21}^{31} = 1;\\
&\text{node pair } (3,5): x_{34}^{35} = x_{45}^{35} = 1;
\end{aligned}$$

This is clearly only one of all possible solutions. How much does it cost? We must compute the number or fiber cables to be installed on each edge. For example, we need one fiber cable on $\{1, 2\}$ and on $\{1, 3\}$, as these carry 29 Mb/s of node pair $(3, 1)$ and thus need a single fiber with 50 Mb/s. Edge $\{4, 5\}$ also gets one fiber as it only carries traffic for demand $(3, 5)$.

Edge $\{3, 4\}$ carries traffic for node pairs $(1, 3)$ and $(1, 6)$ in the direction $3 \to 4$ and for node pair $(3, 5)$ in the opposite direction. Therefore, we need $41 + 12 = 53$ Mb/s in the direction $3 \to 4$ and 31 Mb/s in the opposite one. We need two fibers to accommodate the traffic $3 \to 4$, with a total of 100 Mb/s capacity. The contribution of edge $\{3, 4\}$ to the objective cost, which has to be minimized, is $c_{ij} \cdot 2$.

We can use integer variables y_{ij} for each edge $\{i, j\}$, defining the number of fibers to be installed on edge $\{i, j\}$. What are the constraints? For any node pair (k, l), we must ensure that there be a flow from k to l, for example by introducing flow conservation constraints. For node pair $(1, 3)$, these would be:

$$\text{(node 1)} \qquad x_{12}^{13} + x_{14}^{13} + x_{15}^{13} - x_{21}^{13} - x_{41}^{13} - x_{51}^{13} = 1$$

for the origin node, while at all intermediate nodes we have:

$$\begin{array}{llll}
\text{(node 2)} & x_{21}^{13} + x_{23}^{13} + x_{24} & -x_{12}^{13} - x_{32}^{13} - x_{42}^{13} & = 0 \\
\text{(node 3)} & x_{32}^{13} + x_{34}^{13} + x_{36} & -x_{23}^{13} - x_{43}^{13} - x_{63}^{13} & = 0 \\
\text{(node 4)} & x_{41}^{13} + x_{42}^{13} + x_{43} + x_{45}^{13} & -x_{14}^{13} - x_{24}^{13} - x_{34}^{13} - x_{54}^{13} & = 0 \\
\text{(node 5)} & x_{51}^{13} + x_{54}^{13} + x_{56} & -x_{15}^{13} - x_{45}^{13} - x_{65}^{13} & = 0.
\end{array}$$

This looks very much like the MST. Indeed, to ensure that there is a path for node pair $(1,3)$, we can write similar constraints:

$$\begin{array}{lll}
\text{(node 1)} & \sum_{j \in V:\{1,j\} \in E} x_{1j}^{13} - \sum_{j \in V:\{1,j\} \in E} x_{j1}^{13} = 1 & \\
\text{(nodes 2,4,5,6)} & \sum_{j \in V:\{i,j\} \in E} x_{ij}^{13} - \sum_{j \in V:\{i,j\} \in E} x_{ji}^{13} = 0 & \forall i \in V : 1 \neq i \neq 3
\end{array}$$

and we can write them for the three remaining node pairs. We need now a constraint that defines the value of y variables based on the x variables. Let's take again edge $\{3,4\}$. There will be y_{34} fibers once we figure out what y_{34} is, and that will correspond to $Uy_{ij} = 50y_{ij}$ Mb/s of capacity on that edge. That capacity must accommodate the total traffic on edge $\{3,4\}$ in either direction. **Regardless of the value of the variables**, the total traffic from 3 to 4 is

$$12x_{34}^{13} + 41x_{34}^{16} + 29x_{34}^{31} + 31x_{34}^{35},$$

while that in the opposite direction is

$$12x_{43}^{13} + 41x_{43}^{16} + 29x_{43}^{31} + 31x_{43}^{35},$$

therefore y_{ij} must accommodate for the maximum of them:

$$\begin{array}{l}
Uy_{34} \geq 12x_{34}^{13} + 41x_{34}^{16} + 29x_{34}^{31} + 31x_{34}^{35} \\
Uy_{34} \geq 12x_{43}^{13} + 41x_{43}^{16} + 29x_{43}^{31} + 31x_{43}^{35}.
\end{array}$$

In general, the *capacity* constraints are

$$\begin{array}{l}
Uy_{ij} \geq 12x_{ij}^{13} + 41x_{ij}^{16} + 29x_{ij}^{31} + 31x_{ij}^{35} \\
Uy_{ij} \geq 12x_{ji}^{13} + 41x_{ji}^{16} + 29x_{ji}^{31} + 31x_{ji}^{35};
\end{array}$$

even more in general, the right-hand sides of these constraints is $\sum_{(k,l) \in P} d_{kl} x_{ij}^{kl}$ and $\sum_{(k,l) \in P} d_{kl} x_{ji}^{kl}$, respectively, so we can finally write the model:

$$\begin{array}{lll}
\min & \sum_{\{i,j\} \in E} c_{ij} y_{ij} & \\
& \sum_{j \in V:\{k,j\} \in E} x_{kj}^{kl} - \sum_{j \in V:\{k,j\} \in E} x_{jk}^{kl} = 1 & \forall (k,l) \in P \\
& \sum_{j \in V:\{i,j\} \in E} x_{ij}^{kl} - \sum_{j \in V:\{i,j\} \in E} x_{ji}^{kl} = 0 & \forall i \in V : k \neq i \neq l, \;\; \forall (k,l) \in P \\
& Uy_{ij} \geq \sum_{(k,l) \in P} d_{kl} x_{ij}^{kl} & \forall \{i,j\} \in E \\
& Uy_{ij} \geq \sum_{(k,l) \in P} d_{kl} x_{ji}^{kl} & \forall \{i,j\} \in E \\
& x_{ij}^{kl}, x_{ji}^{kl} \in \{0,1\} & \forall \{i,j\} \in E, \forall (k,l) \in P \\
& y_{ij} \in \{0,1\} & \forall \{i,j\} \in E.
\end{array}$$

The translation in AMPL is very similar to that for the MST: we just need an extra set to describe the node pairs and a slightly different definition of the x variables:

```
set P within {i in V, j in V: i != j};
var x {ED, P} binary;
```

and the corresponding parameters will be as follow

```
set P = (1,3) (1,6) (3,1) (3,5);
```