Network models in GAMS

erwin@amsterdamoptimization.com

Network models in GAMS

- Goal: develop simple random network models:
  - Shortest Path
  - Min-Cost Flow
- Note that transportation model discussed before is an example of a (bipartite) network with supply and demand nodes.
- I focus on **directed** graphs. Directed graphs are more prevalent, and are also easier to deal with.
- I will also talk a bit about exporting data for visualization.
- Accompanying GAMS model: `network.gms`

Network topology

```
set
   n 'nodes' /node1*node50/
   a(n,n) 'arcs'
;

alias (n,i,j);

* sparse random network
a(i,j)$(uniform(0,1)<0.05) = yes;

display n,a;
```

Alternative: `a(i,j) = uniform(0,1)<0.05;`

- *n* is a 1-dim static set
  - Can be used as domain (e.g. for *a*)
- *a* is a 2-dim dynamic set
- Alternative: use different sets for source and destination nodes:

```
set i /node1*node50/
    j /node1*node50/
    a(i,j)
```

- Note: `a(n,n)` is diagonal when used outside declarations.
- $ is the "such-that" operator

Random number generator.

- GAMS uses a (platform independent) pseudo random generator, so runs are **reproducible**.
- Set seed to generate other sequence.
  - `Option seed = 12345;`
  - `execseed = 12345;`
- If you insist on a new sequence each time:
  - `execseed = 1+gmillisec(jnow);`

Exercises

- Rerun the network generation code using different seeds
  - How would one find out the default initial seed (3141)
- Verify the difference between:
  - `a(i,j)$(uniform(0,1)<0.05) = yes;`
  - `a(n,n)$(uniform(0,1)<0.05) = yes;`
- Use `set a(i,j) 'arcs';` instead of `set a(n,n) 'arcs';`
  - This means reordering the declarations a bit
- A better display can be achieved with: `option a:0:0:8; display a;`
  - You may need to set a wider pagewidth
    - Command line parameter pw=200

What is the number of arcs?

```
scalar numarcs 'number of arcs';
numarcs = sum((i,j)$a(i,j),1);
numarcs = sum(a(i,j),1);
numarcs = sum(a,1);
numarcs = card(a);
display numarcs;
```

- Approx 5% of n^2, i.e. 125
- We need to count number of elements in *a*
- This can be evaluated in different ways

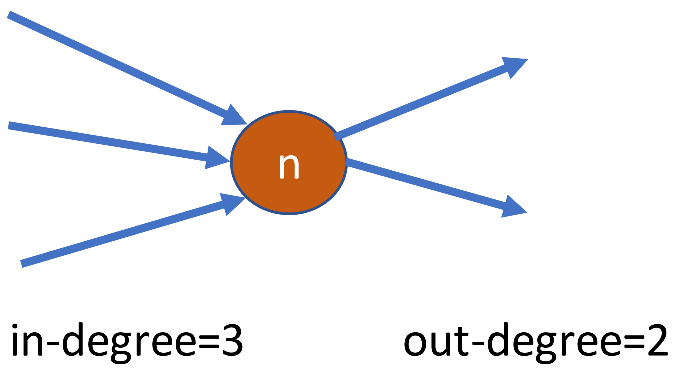
```
----      63 PARAMETER numarcs                =        134.000  number of arcs
```

Summary: In/Out-degree

```
degree(n,'in-degree') = sum(a(i,n),1);
degree(n,'out-degree') = sum(a(n,i),1);

degree('min','in-degree') = smin(n,degree(n,'in-degree'));
degree('min','out-degree') = smin(n,degree(n,'out-degree'));
degree('max','in-degree') = smax(n,degree(n,'in-degree'));
degree('max','out-degree') = smax(n,degree(n,'out-degree'));
```

```
----      72 PARAMETER degree  in- and out-degree

          in-degree  out-degree

node1                    1.000
node2         4.000      3.000
node3         4.000      2.000
node4         1.000      5.000
node5         3.000      4.000
node6         1.000      4.000
node7                    4.000
node8                    3.000
node9         3.000      3.000
node10        1.000      3.000
. . .
node49        4.000      3.000
node50        3.000      2.000
max           7.000      6.000
```

Why is **min** row missing?



GAMS Sparsity Rule

Zero and does not exist is the same.

Exercises

- Why is row with "min" missing in the output?
- Try using an expression like: `EPS+smin(n,degree(n,'in’))`
  - EPS values are usually converted to zeros when exported
- Add a row for “average in- and out-degree”
  - Explain the results
  - This is equal to `card(a)/card(n)`
- Add a colum for “degree” where: degree = in-degree+out-degree.

Diagonal

- Outside declarations using `(n,n)` indicates the diagonal.
- For shortest path/min-cost flow models we usually don't mind these self-loops: if costs (lengths) are positive, it is never profitable to use them.

```
* do we have diagonal elements?
set diagonal(n) 'diagonal elements';
diagonal(n) = a(n,n);
display diagonal;
abort$card(diagonal) "Diagonal is not empty";
```

Otherwise remove diagonal elements by:

```
a(n,n) = no;
```

```
----     96 SET diagonal  diagonal elements

node35

----     97 Diagonal is not empty
**** Exec Error at line 97: Execution halted: abort$1 'Diagonal is not empty'
```

Optional

Export to Python

```
embeddedCode Python:
import pickle
nodes = list(gams.get('n'))
arcs = list(gams.get('a'))
coord = list(gams.get('coord'))
data = {'nodes':nodes,
        'arcs':arcs,
        'coord':coord}
pickle.dump(data,open('%picklefile%', 'wb'))
endEmbeddedCode
```

Export data

Data representation

nodes: ['node1', 'node2', 'node3',...]
arcs: [('node1', 'node44'), ('node2', 'node9'),...]
coord:[(('node1', 'x'),11.6), (('node1', 'y'),84.3),...]

GAMS will substitute out `%picklefile%` for a file name

```
import pickle
import networkx as nx

data = pickle.load(open('%picklefile%', 'rb'))

DG = nx.DiGraph()
DG.add_nodes_from(data['nodes'])
DG.add_edges_from(data['arcs'])
print(DG)
```

Import into Python script

Splitting code in two parts makes it easier to debug.

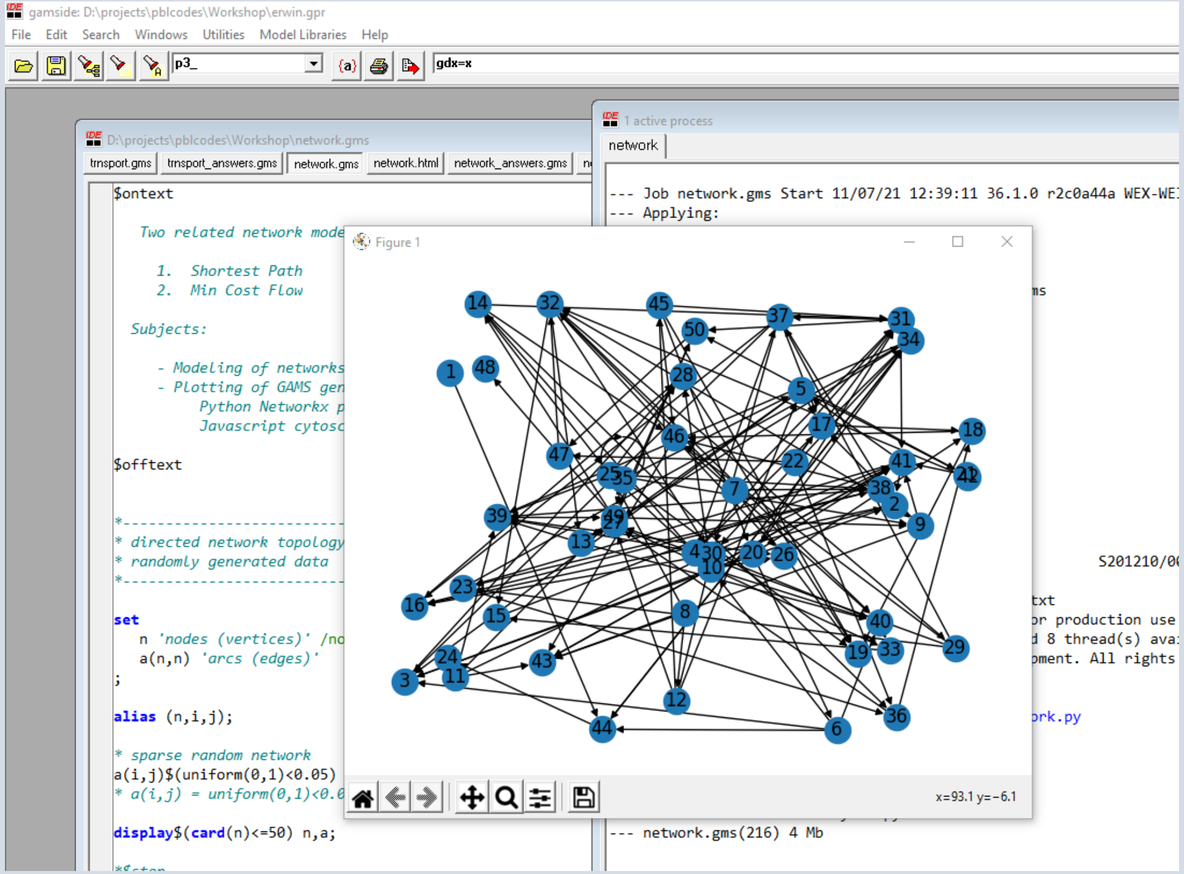
Optional

Running GAMS model with Python code

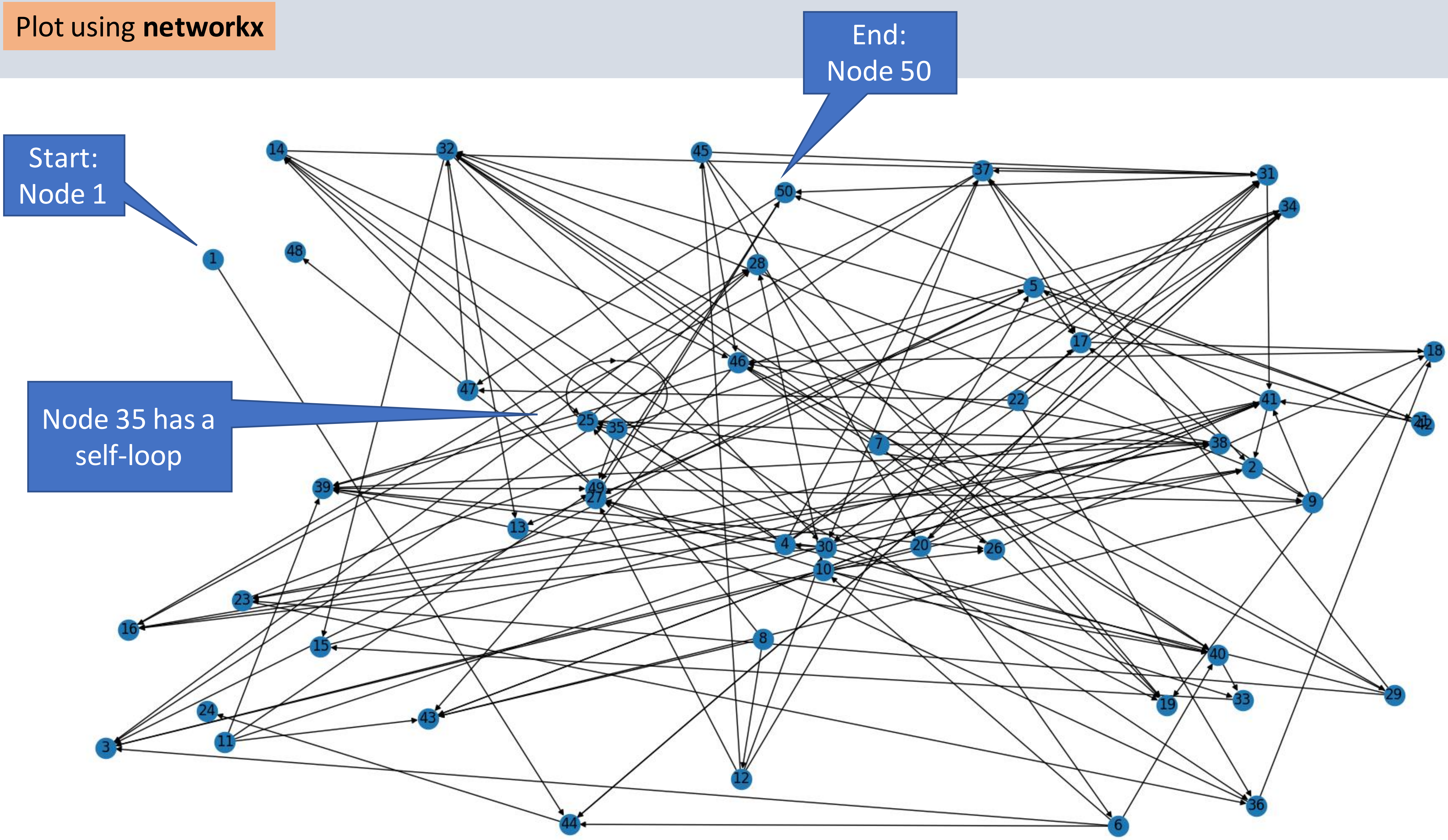
- Nodes are shown as 1,2,…
- We also generate random coordinates in GAMS

$$coord_{n,xy} \sim U(0,100)$$

- A little bit of work to transform them from a GAMS datastructure (sparse) to a dict suited for networkx
- GAMS pauses until you close the window



Plot using **networkx**



Shortest Path Problem

- In our example model, length/cost is just the Euclidean distance between nodes
  - But can be anything. Assume they are positive.
- Instead of using an shortest path algorithm (Dijkstra) we use an LP formulation

Edsger Wiebe Dijkstra

$$\min \quad \sum_{i,j} cost_{i,j} f_{i,j}$$

$$\sum_{i} f_{i,n} + inflow_n = \sum_{j} f_{n,j} + outflow_n \quad \forall n$$

$$f_{i,j} \in \{0,1\}$$

$$inflow_n = \begin{cases} 1 & \text{if } n \text{ is the start node} \\ 0 & \text{otherwise} \end{cases}$$

$$outflow_n = \begin{cases} 1 & \text{if } n \text{ is the end node} \\ 0 & \text{otherwise} \end{cases}$$

These vectors are extremely sparse.
GAMS will only store the nonzero elements.

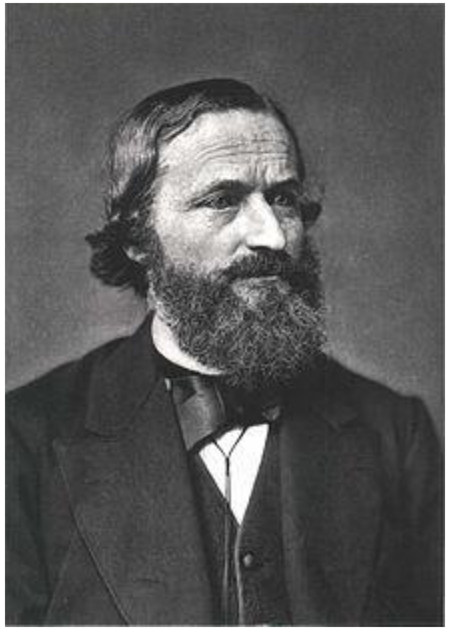
- The solution is automatically integer, so we can use continuous variables (LP) instead of binary variables (MIP).
- The node-balance equation sums over **sparse** network topology.
- A.k.a. flow-conservation or Kirchoff equations.

$$\min \sum_{i,j|A(i,j)} cost_{i,j} f_{i,j}$$

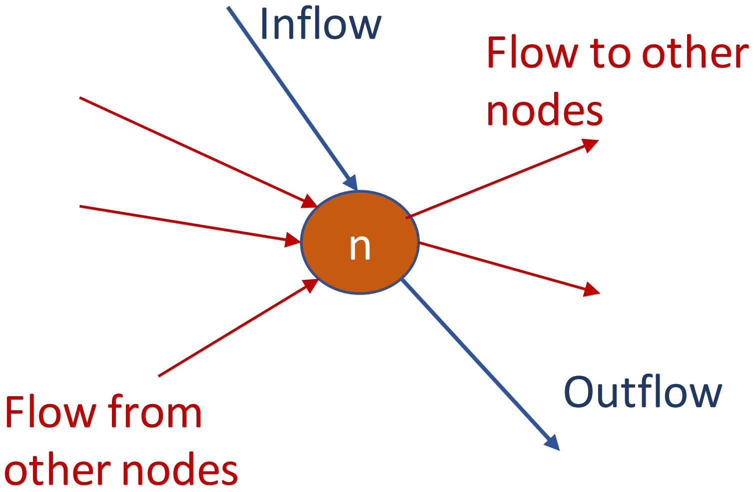
$$\sum_{i|A(i,n)} f_{i,n} + inflow_n = \sum_{j|A(n,j)} f_{n,j} + outflow_n \quad \forall n$$

$$f_{i,j} \geq 0$$

- We can combine the inflow and outflow vectors into one vector.



Gustav Kirchoff



```
parameters
   inflow(n)    'exogenous inflow at node'     / node1  1.0 /
   outflow(n)   'exogenous outflow at node'    / node50 1.0 /
;

positive variable f(i,j) 'flow from node i -> node j';

variable totalLength 'objective: minimize';

equations
    nodeBalance(n) 'kirchoff equations'
    objective      'minimize'
;
```

Declaration: |n| x |n| flow variables

```
objective.. totalLength =e= sum(a,length(a)*f(a));

nodeBalance(n)..
     sum(a(i,n), f(a)) + inflow(n) =e=
     sum(a(n,j), f(a)) + outflow(n);

model shortestPath /all/;
solve shortestPath using lp minimizing totalLength;

display f.l;
```

Size: |n|+1 equations, |a|+1 variables
Only variables occurring in equations count!

```
----    147 VARIABLE f.L  flow from node i -> node j

            node20      node31      node34      node44      node50

node1                                            1.000
node20                   1.000
node31                                                       1.000
node34       1.000
node44                               1.000
```

Form path (not so easy)

```
sets
  step /step1*step50/
  path(step,n) 'easier to read than f'
;
singleton set cur(i) 'current node';
cur('node1') = yes;
* while we have a current node
loop(step$card(cur),
* record current node
   path(step,cur) = yes;
* current node := next node
   cur(j) = f.l(cur,j)>0.5;
* to debug add this
*   display cur;
);
option path:0:0:1;
display path;
```

```
----     171 SET path  easier to read than f

step1.node1
step2.node44
step3.node34
step4.node20
step5.node31
step6.node50
```

Note:
In GAMS we cannot have something like
`[node1,node44,node34,node20,node31,node50]`
The ordering of nodes is predetermined by /node1*node50/

Exercise: write this piece of GAMS code without the use of singleton sets.

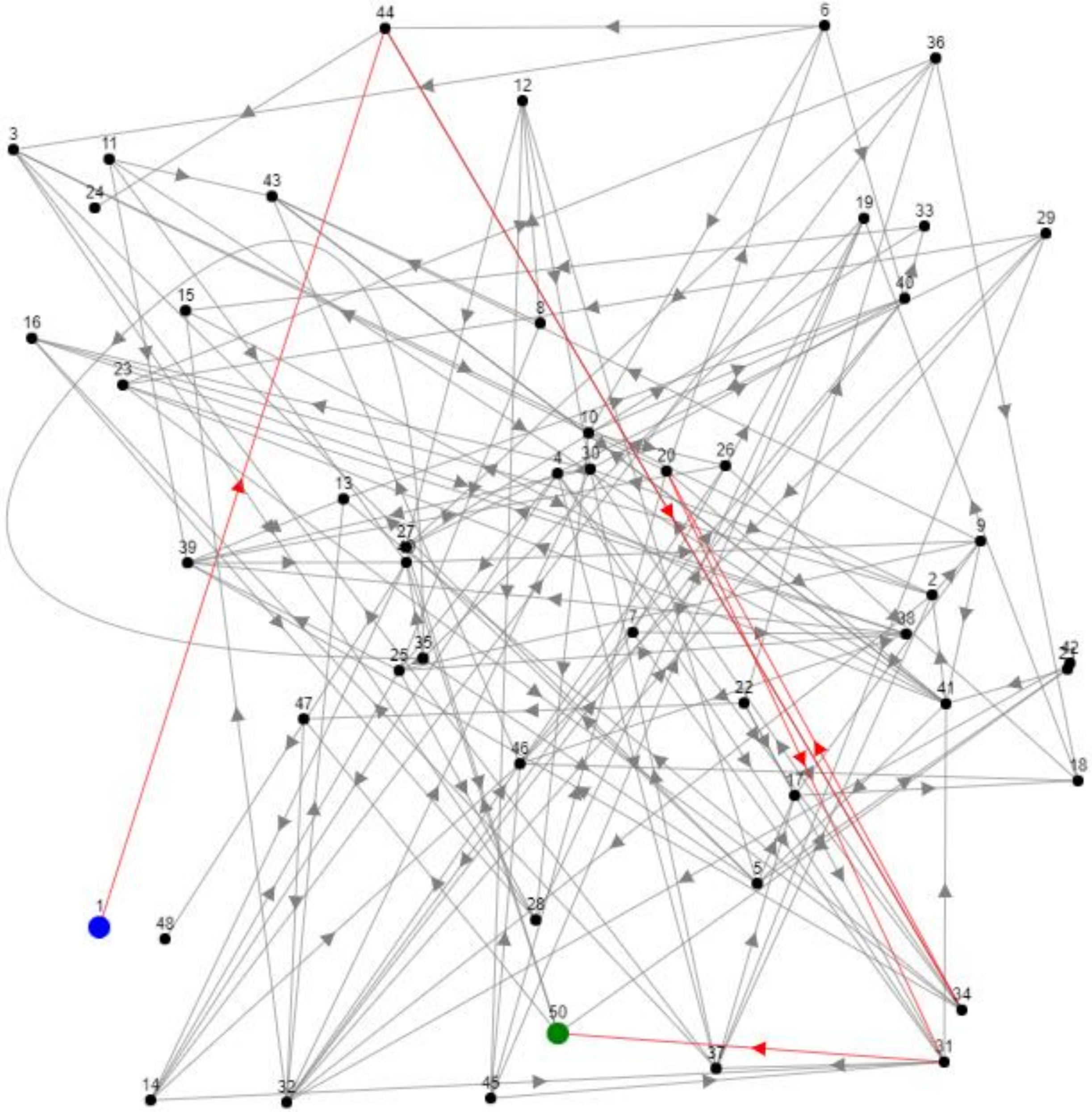
Generate HTML/Javascript

- To make a **browser**-based plot in we can generate HTML/Javascript from the GAMS model.
- In this example, I used the **PUT facility** to write the data file (javascript code).
- Then some HTML/Javascript was used generate the plot.
  - Javascript network package: **cytoscape.js**

GAMS Shortest Path

Number of nodes: 50
Number of arcs: 134

From	To	Length
node1	node44	84.526
node44	node34	102.056
node34	node20	55.080
node20	node31	58.517
node31	node50	34.789
Total length		334.968



HTML + JS document

Min-Cost Flow LP

$$\min \sum_{i,j|A(i,j)} cost_{i,j} f_{i,j}$$

$$\sum_{i|A(i,n)} f_{i,n} + inflow_n = \sum_{j|A(n,j)} f_{n,j} + outflow_n \quad \forall n$$

$$0 \leq f_{i,j} \leq capacity_{i,j}$$

- It is easy to generalize the Shortest Path LP model to a more generic Min-Cost Flow LP model.
  - Allow multiple Supply and Demand Nodes
    - These have a nonzero inflow or outflow
    - The remaining nodes are transshipment nodes
  - The lengths become costs
  - Capacity limits on the arcs
    - `f.up(a) = capacity(a);`
    - Simple bound instead of full-blown constraint
    - This may split a flow to different paths

Exercises

- Adapt the HTML/Javascript code to report the Min-Cost Flow solution.
- Implement the trnsport.gms model as a min-cost flow problem.