ICON 2004 - IEEE International Conference On Networks

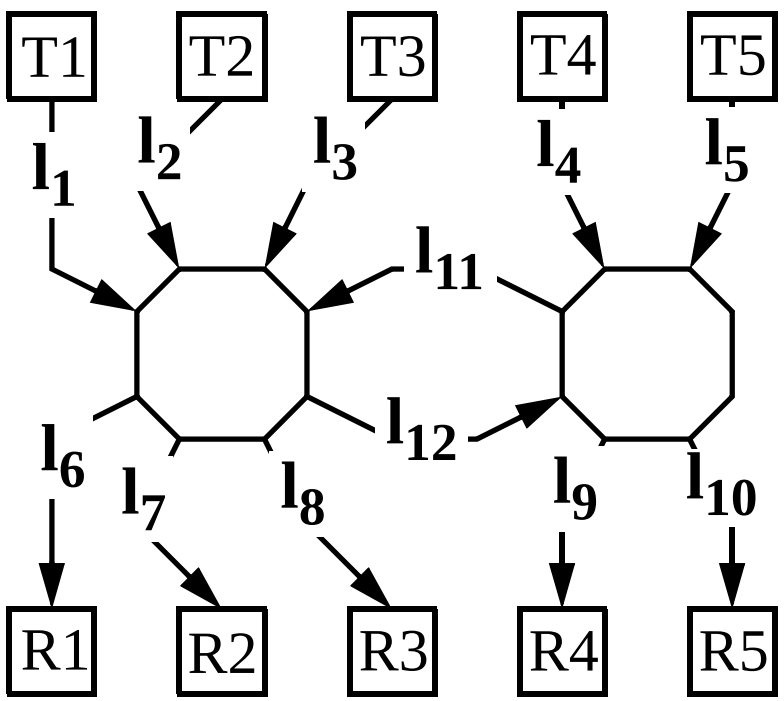
November 16-19, 2004, Singapore, Hilton

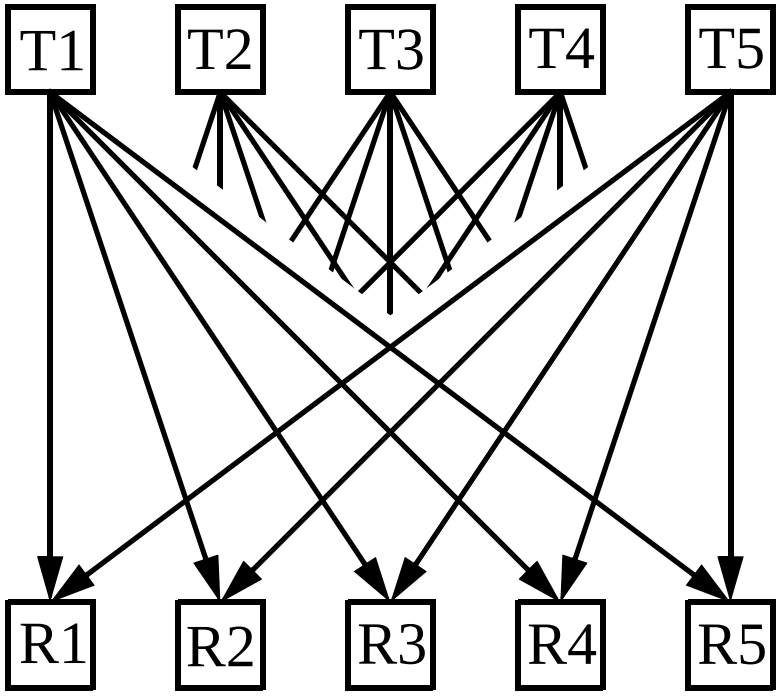
EFFICIENT LIQUID SCHEDULE SEARCH STRATEGIES FOR COLLECTIVE COMMUNICATIONS

Emin Gabrielyan, Roger D. Hersch

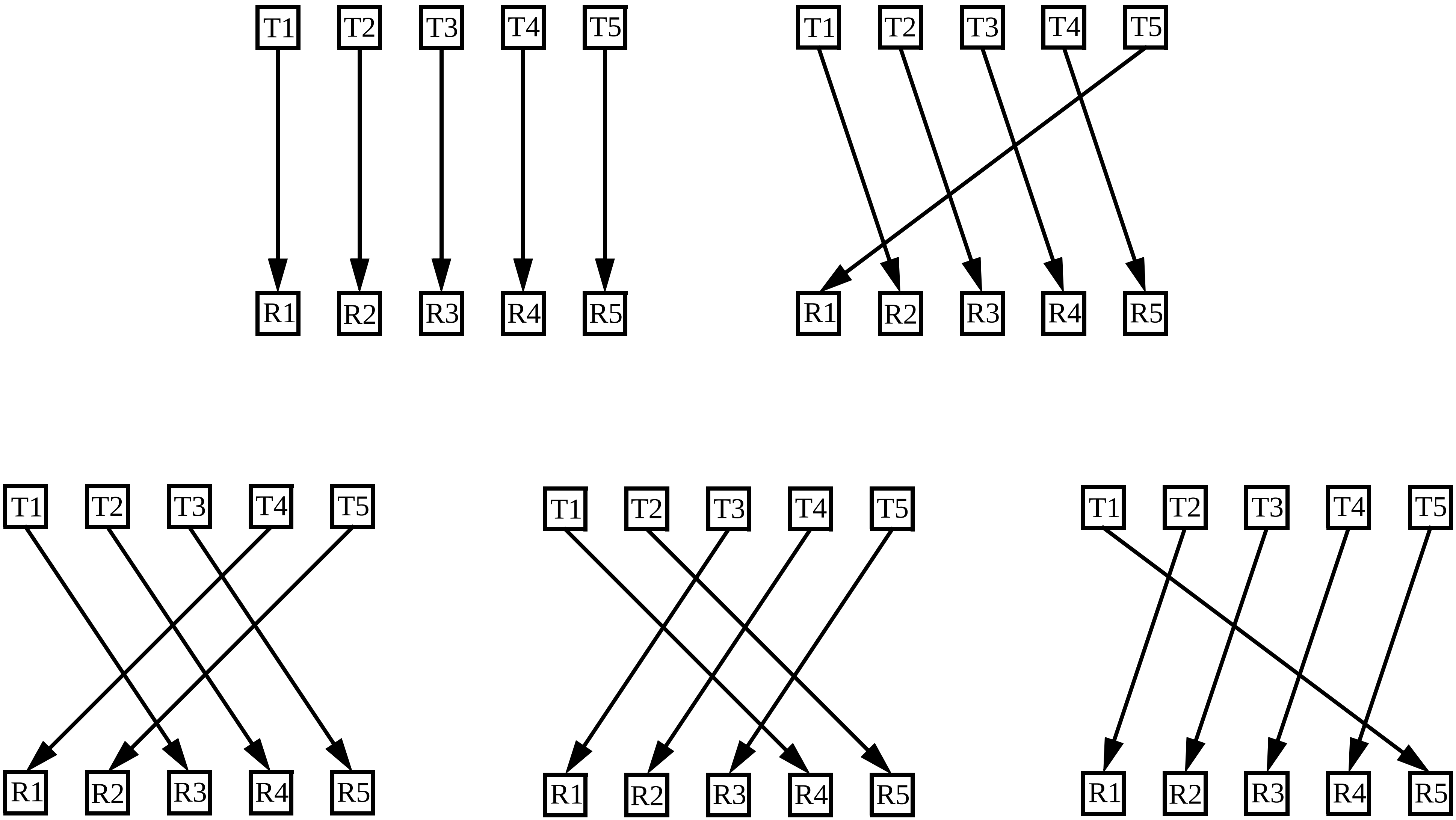
Swiss Federal Institute of Technology - Lausanne

Example: 25 transmissions to be carried out

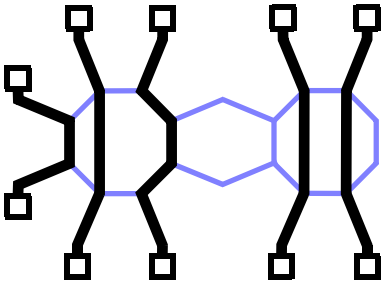




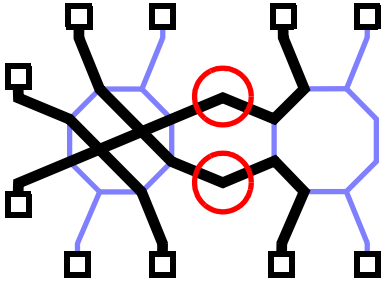
Round-robin schedule



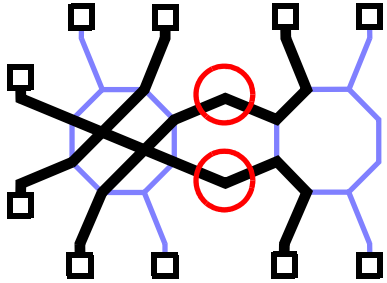
Round-robin Throughput



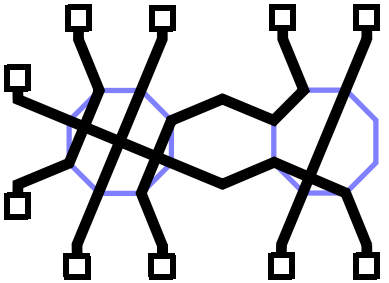
phase 1



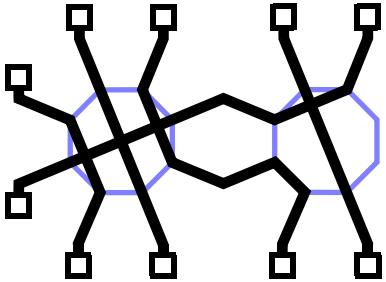
phase 3.1



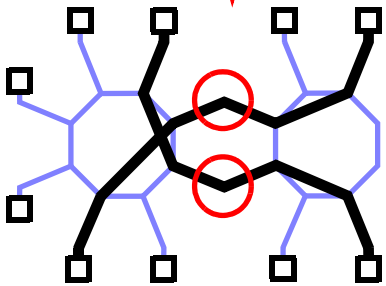
phase 4.1



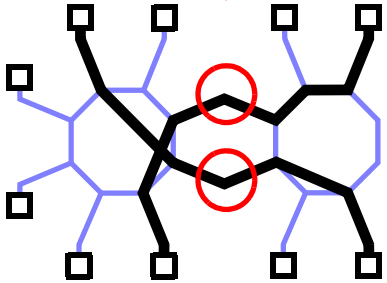
phase 5



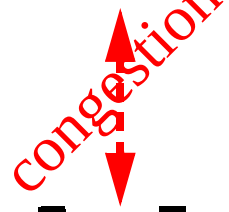
phase 2



phase 3.2

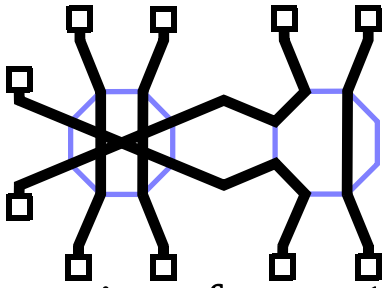


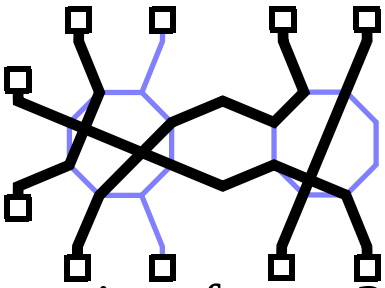
phase 4.2

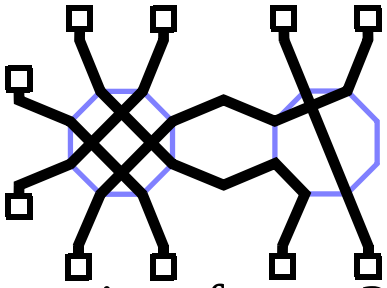


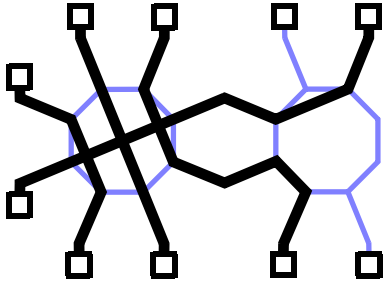
$$T_{roundrobin} = 25/7 \cdot 1Gbps = 3.57Gbps$$

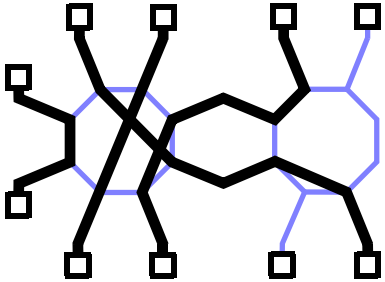
Liquid schedule

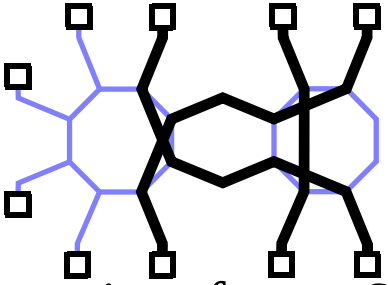

time frame 1


time frame 2


time frame 3


time frame 4

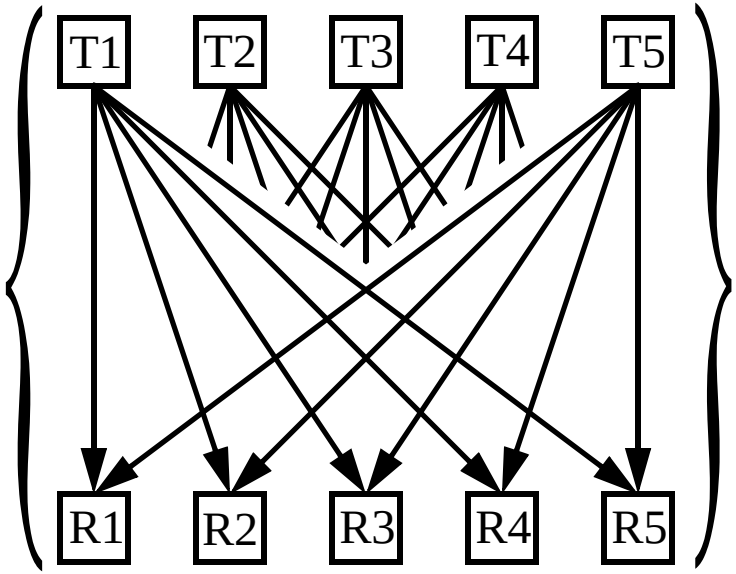

time frame 5

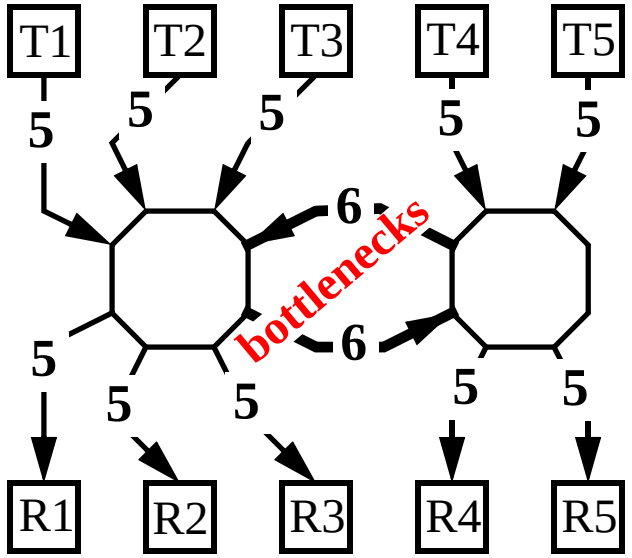

time frame 6

$$T_{liquid} = 25/6 \cdot 1Gbps = 4.16Gbps$$

Transfers and Load of Links

X =



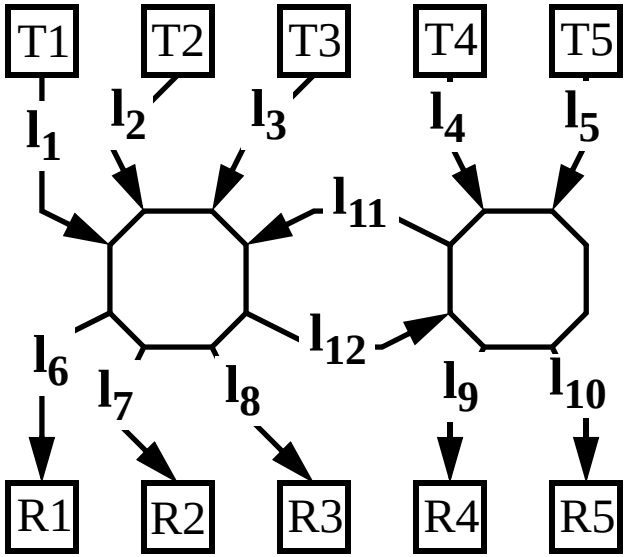


The 25 transfer traffic

$$\lambda(l_1, X) = 5, \ldots \lambda(l_{12}, X) = 6$$

Transfers: $\{l_1, l_6\}, \ldots \{l_1, l_{12}, l_9\}, \ldots$

Duration of Traffic



$$\lambda(l_1, X) = 5, \ldots \lambda(l_{10}, X) = 5$$

$$\lambda(l_{11}, X) = 5, \ldots \lambda(l_{12}, X) = 6$$

$$\Lambda(X) = 6$$

$$X = \left\{ \begin{array}{c} \{l_1, l_6\}, \{l_1, l_7\}, \{l_1, l_8\}, \{l_1, l_{12}, l_9\}, \{l_1, l_{12}, l_{10}\}, \\ \{l_2, l_6\}, \{l_2, l_7\}, \{l_2, l_8\}, \{l_2, l_{12}, l_9\}, \{l_2, l_{12}, l_{10}\}, \\ \{l_3, l_6\}, \{l_3, l_7\}, \{l_3, l_8\}, \{l_3, l_{12}, l_9\}, \{l_3, l_{12}, l_{10}\}, \\ \{l_4, l_{11}, l_6\}, \{l_4, l_{11}, l_7\}, \{l_4, l_{11}, l_8\}, \{l_4, l_9\}, \{l_4, l_{10}\}, \\ \{l_5, l_{11}, l_6\}, \{l_5, l_{11}, l_7\}, \{l_5, l_{11}, l_8\}, \{l_5, l_9\}, \{l_5, l_{10}\} \end{array} \right\}$$

Liquid Throughput

$$X=\left\{\begin{array}{c}\{l_1, l_6\}, \{l_1, l_7\}, \{l_1, l_8\}, \{l_1, l_{12}, l_9\}, \{l_1, l_{12}, l_{10}\},\\ \{l_2, l_6\}, \{l_2, l_7\}, \{l_2, l_8\}, \{l_2, l_{12}, l_9\}, \{l_2, l_{12}, l_{10}\},\\ \{l_3, l_6\}, \{l_3, l_7\}, \{l_3, l_8\}, \{l_3, l_{12}, l_9\}, \{l_3, l_{12}, l_{10}\},\\ \{l_4, l_{11}, l_6\}, \{l_4, l_{11}, l_7\}, \{l_4, l_{11}, l_8\}, \{l_4, l_9\}, \{l_4, l_{10}\},\\ \{l_5, l_{11}, l_6\}, \{l_5, l_{11}, l_7\}, \{l_5, l_{11}, l_8\}, \{l_5, l_9\}, \{l_5, l_{10}\}\end{array}\right\}$$

the throughput of a single link

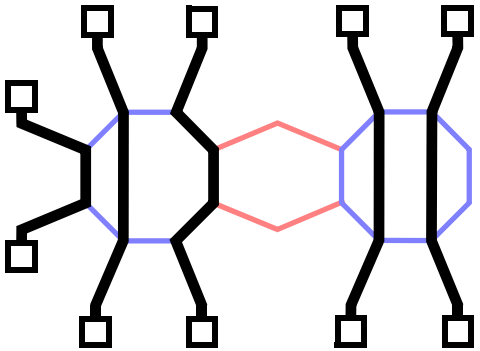
total number of transfers

$$T_{liquid} = \frac{\#(X)}{\Lambda(X)} \cdot T_{link} = \frac{25}{6} \cdot 1Gbps = 4.17Gbps$$

traffic's duration (the load of its bottlenecks)

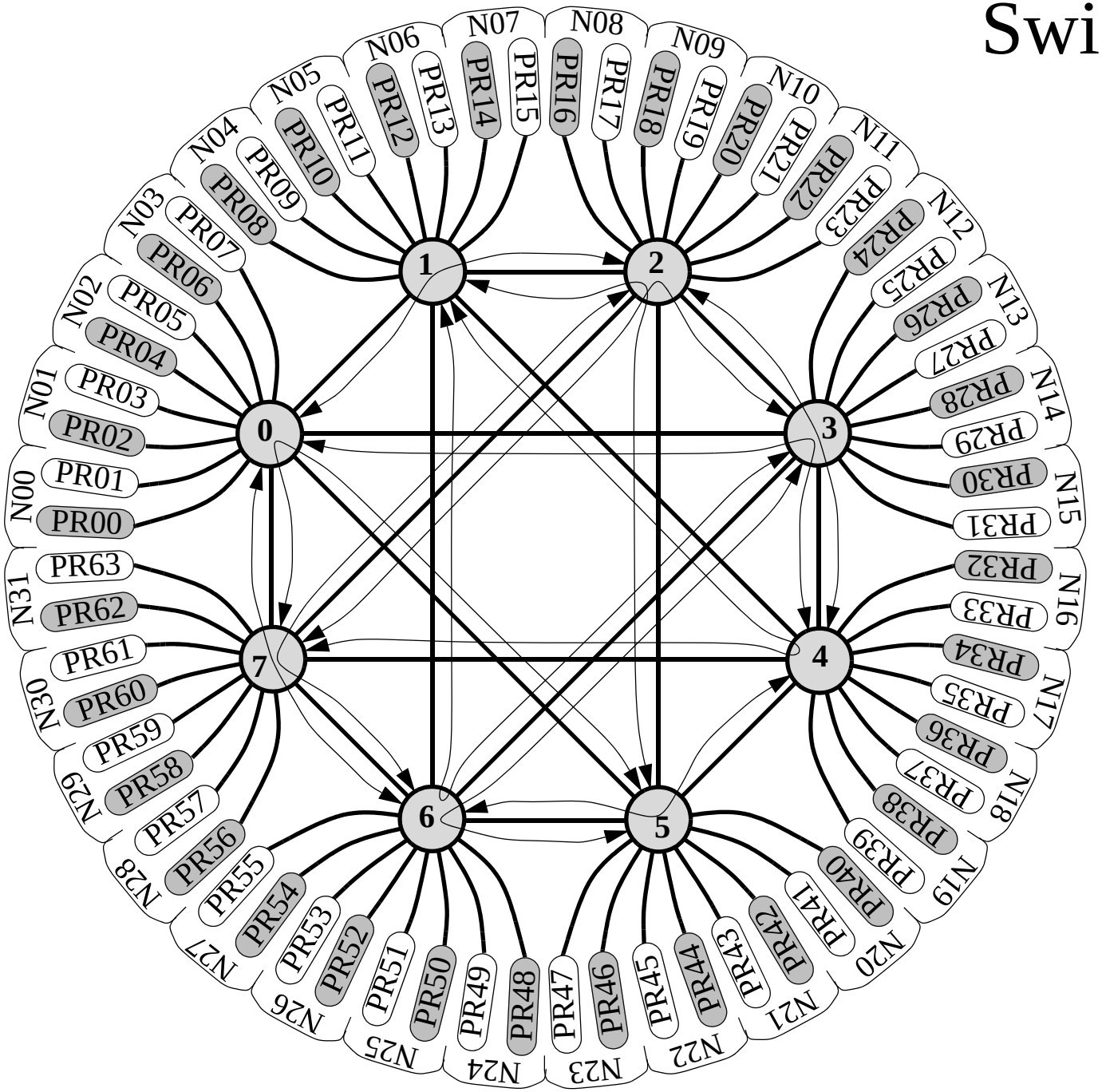
Schedules yielding the liquid throughput

$$X=\left\{\begin{array}{c}
\{l_1, l_6\}, \{l_1, l_7\}, \{l_1, l_8\}, \{l_1, \mathbf{l_{12}}, l_9\}, \{l_1, \mathbf{l_{12}}, l_{10}\}, \\
\{l_2, l_6\}, \{l_2, l_7\}, \{l_2, l_8\}, \{l_2, \mathbf{l_{12}}, l_9\}, \{l_2, \mathbf{l_{12}}, l_{10}\}, \\
\{l_3, l_6\}, \{l_3, l_7\}, \{l_3, l_8\}, \{l_3, \mathbf{l_{12}}, l_9\}, \{l_3, \mathbf{l_{12}}, l_{10}\}, \\
\{l_4, \mathbf{l_{11}}, l_6\}, \{l_4, \mathbf{l_{11}}, l_7\}, \{l_4, \mathbf{l_{11}}, l_8\}, \{l_4, l_9\}, \{l_4, l_{10}\}, \\
\{l_5, \mathbf{l_{11}}, l_6\}, \{l_5, \mathbf{l_{11}}, l_7\}, \{l_5, \mathbf{l_{11}}, l_8\}, \{l_5, l_9\}, \{l_5, l_{10}\}
\end{array}\right\}$$



- Without a right schedule we may have intervals when the access to the bottleneck links is blocked by other transmissions.
- Our goal is to schedule the transfers such that all bottlenecks are always kept occupied ensuring that the liquid throughput is obtained.
- A schedule yielding the liquid throughput we call as a liquid schedule and our objective is to find a liquid schedule whenever it exists.

Swiss-T1 Cluster



N00 Node

0 Switch

PR01 Rx Proc

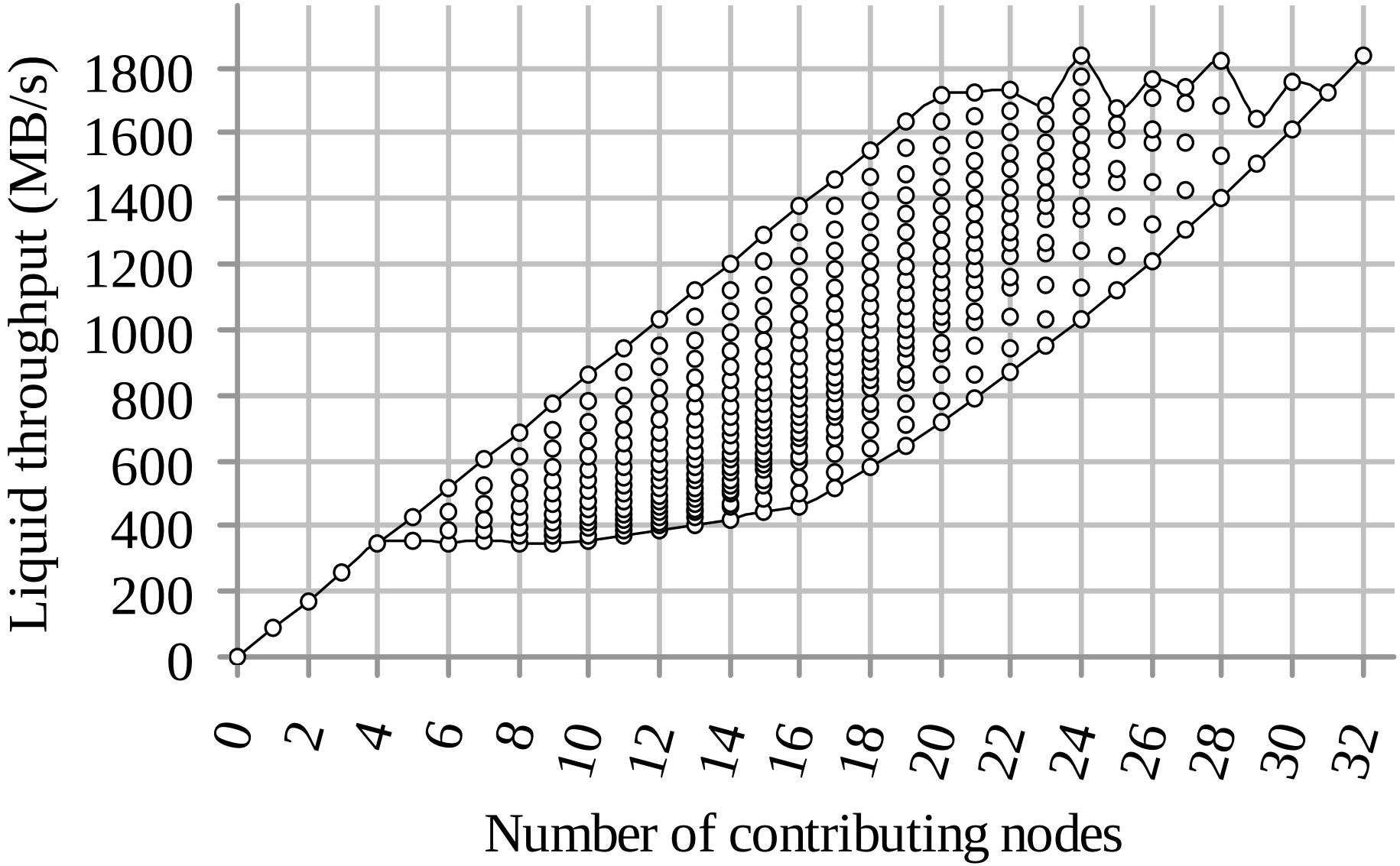
PR00 Tx Proc

Routing

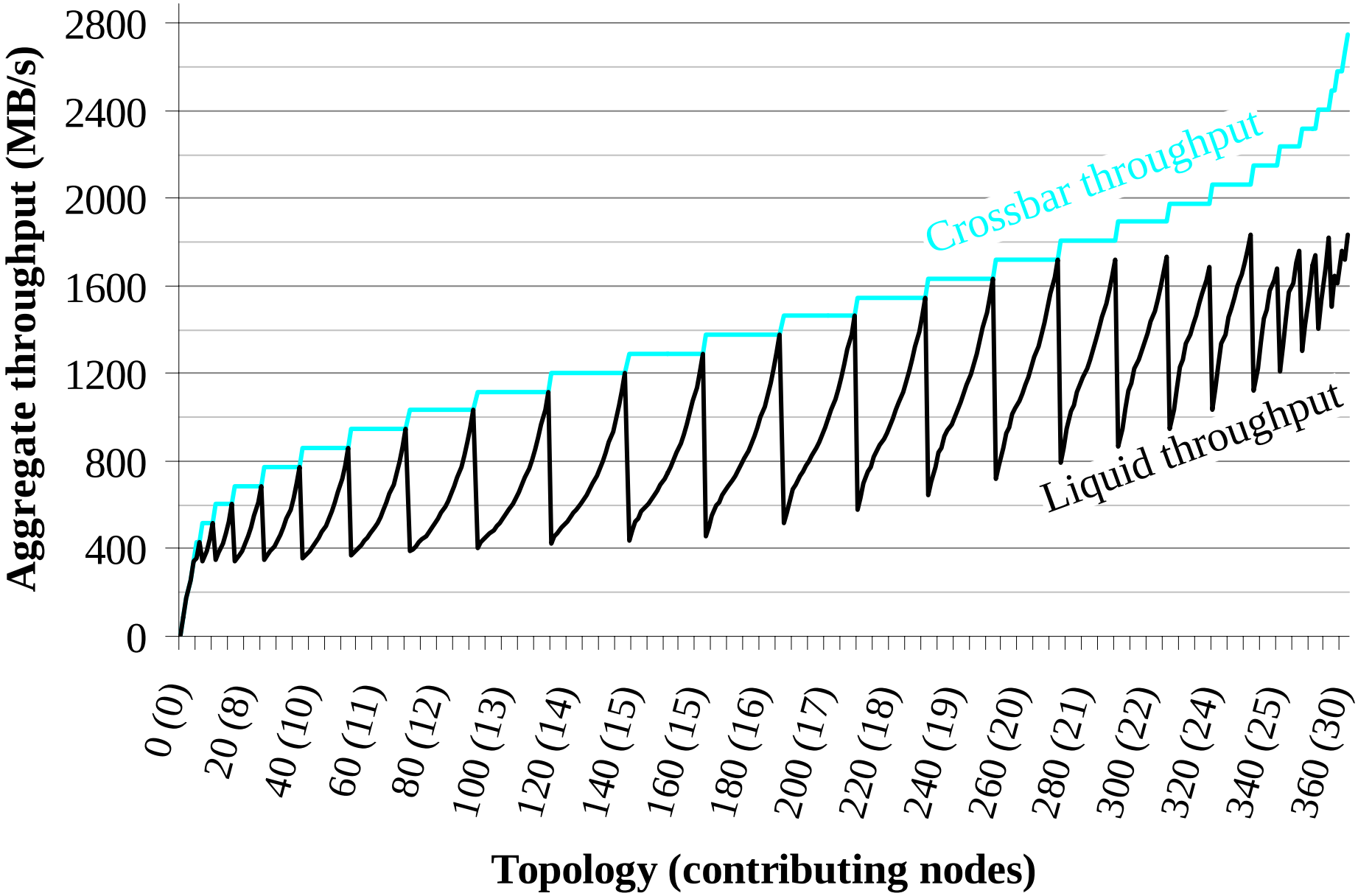
Link

-- ICON 2004, IEEE International Conference On Networks, November 16-19, 2004, Singapore, Hilton --
-- EFFICIENT LIQUID SCHEDULE SEARCH STRATEGIES FOR COLLECTIVE COMMUNICATIONS --

363 Communication Patterns

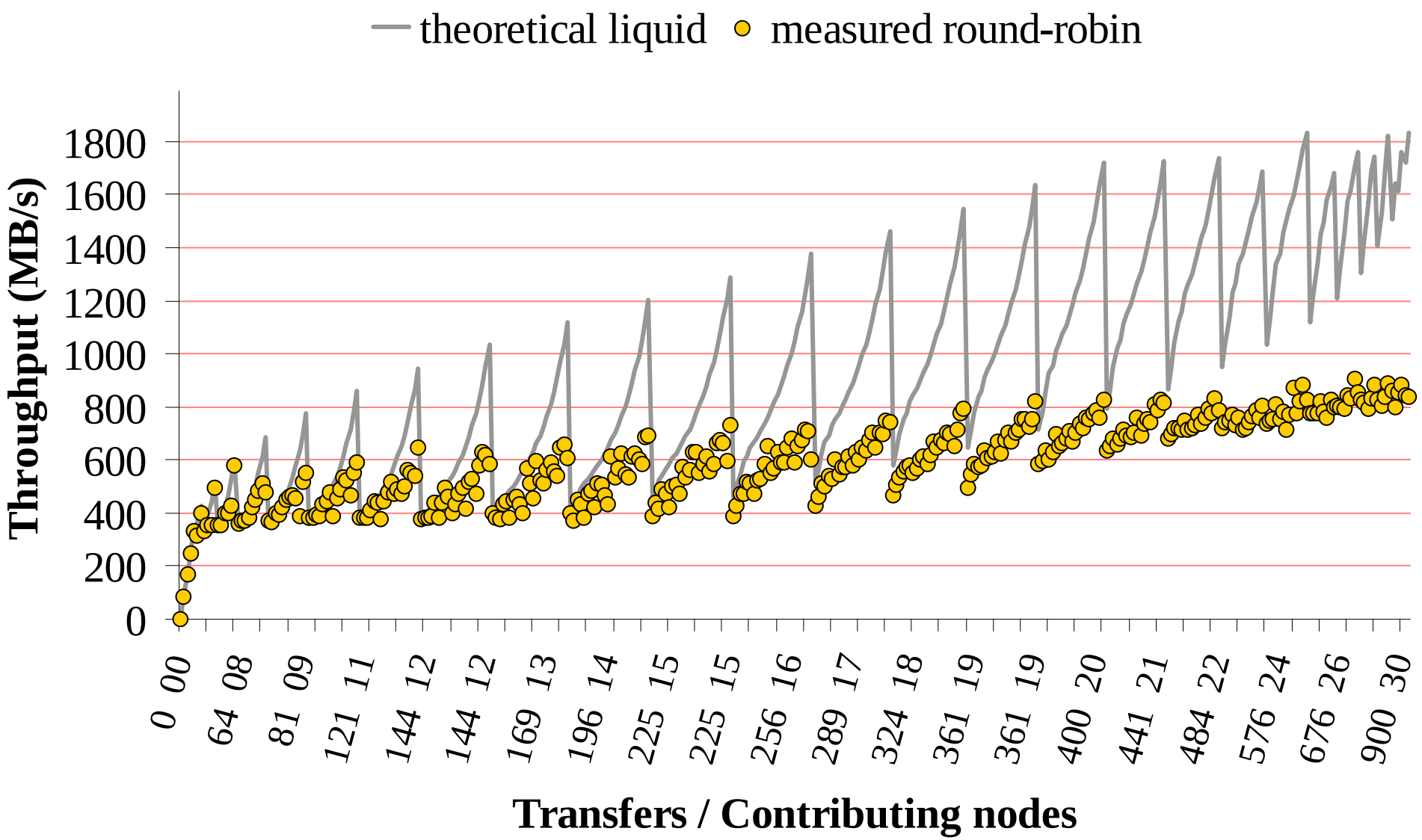


363 Topology Test-bed



-- ICON 2004, IEEE International Conference On Networks, November 16-19, 2004, Singapore, Hilton --
-- EFFICIENT LIQUID SCHEDULE SEARCH STRATEGIES FOR COLLECTIVE COMMUNICATIONS --

Round-robin throughput



Team: a set of mutually non-congesting transfers using all bottlenecks

$$X = \left\{ \begin{array}{c} \{l_1, l_6\}, \{l_1, l_7\}, \{l_1, l_8\}, \{l_1, \mathbf{l_{12}}, l_9\}, \{l_1, \mathbf{l_{12}}, l_{10}\}, \\ \{l_2, l_6\}, \{l_2, l_7\}, \{l_2, l_8\}, \{l_2, \mathbf{l_{12}}, l_9\}, \{l_2, \mathbf{l_{12}}, l_{10}\}, \\ \{l_3, l_6\}, \{l_3, l_7\}, \{l_3, l_8\}, \{l_3, \mathbf{l_{12}}, l_9\}, \{l_3, \mathbf{l_{12}}, l_{10}\}, \\ \{l_4, \mathbf{l_{11}}, l_6\}, \{l_4, \mathbf{l_{11}}, l_7\}, \{l_4, l_{11}, l_8\}, \{l_4, l_9\}, \{l_4, l_{10}\}, \\ \{l_5, \mathbf{l_{11}}, l_6\}, \{l_5, \mathbf{l_{11}}, l_7\}, \{l_5, l_{11}, l_8\}, \{l_5, l_9\}, \{l_5, l_{10}\} \end{array} \right\}$$

schedule α is liquid $\Leftrightarrow$

load of the bottlenecks

number of timeframes

$$\alpha = \left\{ \begin{array}{l} \left\{ \begin{array}{c} \{l_1, \mathbf{l_{12}}, l_9\}, \\ \{l_2, l_7\}, \\ \{l_3, l_8\}, \\ \{l_4, \mathbf{l_{11}}, l_6\}, \\ \{l_5, l_{10}\} \end{array} \right\}, \left\{ \begin{array}{c} \{l_1, \mathbf{l_{12}}, l_{10}\}, \\ \{l_2, l_6\}, \\ \{l_4, \mathbf{l_{11}}, l_7\}, \\ \{l_5, l_9\} \end{array} \right\}, \left\{ \begin{array}{c} \{l_1, \mathbf{l_8}\}, \\ \{l_2, \mathbf{l_{12}}, l_9\}, \\ \{\mathbf{l_3}, l_6\}, \\ \{l_4, l_{10}\}, \\ \{l_5, \mathbf{l_{11}}, l_7\} \end{array} \right\}, \\ \left\{ \begin{array}{c} \{l_1, l_7\}, \\ \{l_2, \mathbf{l_8}\}, \\ \{\mathbf{l_3}, \mathbf{l_{12}}, l_9\}, \\ \{l_5, \mathbf{l_{11}}, l_6\} \end{array} \right\}, \left\{ \begin{array}{c} \{l_1, l_6\}, \\ \{l_2, \mathbf{l_{12}}, \mathbf{l_{10}}\}, \\ \{\mathbf{l_3}, l_7\}, \\ \{\mathbf{l_4}, \mathbf{l_{11}}, \mathbf{l_8}\} \end{array} \right\}, \left\{ \begin{array}{c} \{\mathbf{l_3}, \mathbf{l_{12}}, \mathbf{l_{10}}\}, \\ \{\mathbf{l_4}, \mathbf{l_9}\}, \\ \{\mathbf{l_5}, \mathbf{l_{11}}, \mathbf{l_8}\} \end{array} \right\} \end{array} \right\}$$

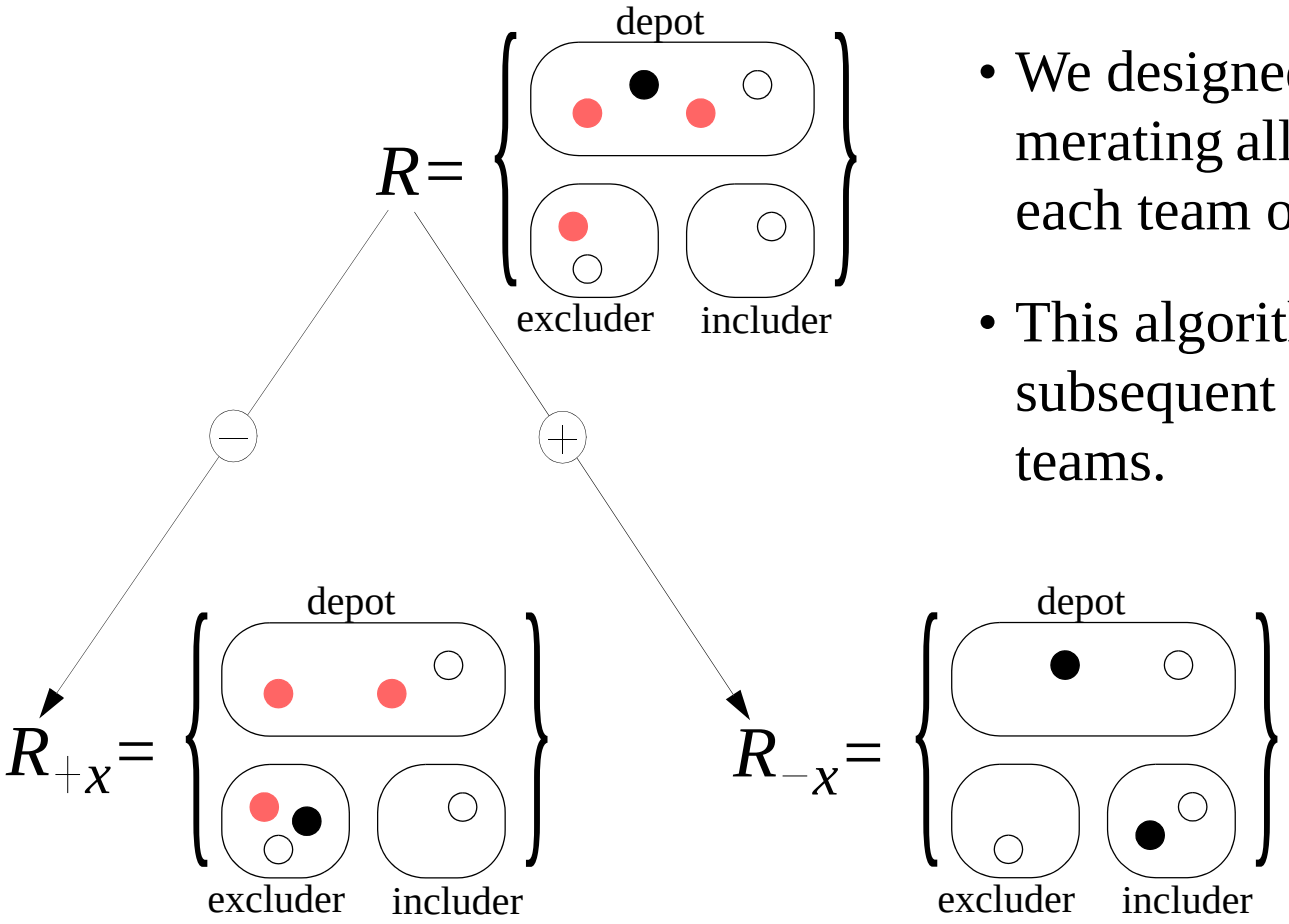
$\Leftrightarrow \#(\alpha) = \Lambda(X) \Leftrightarrow$

$\Leftrightarrow \forall(A \in \alpha)$

A is a team of X

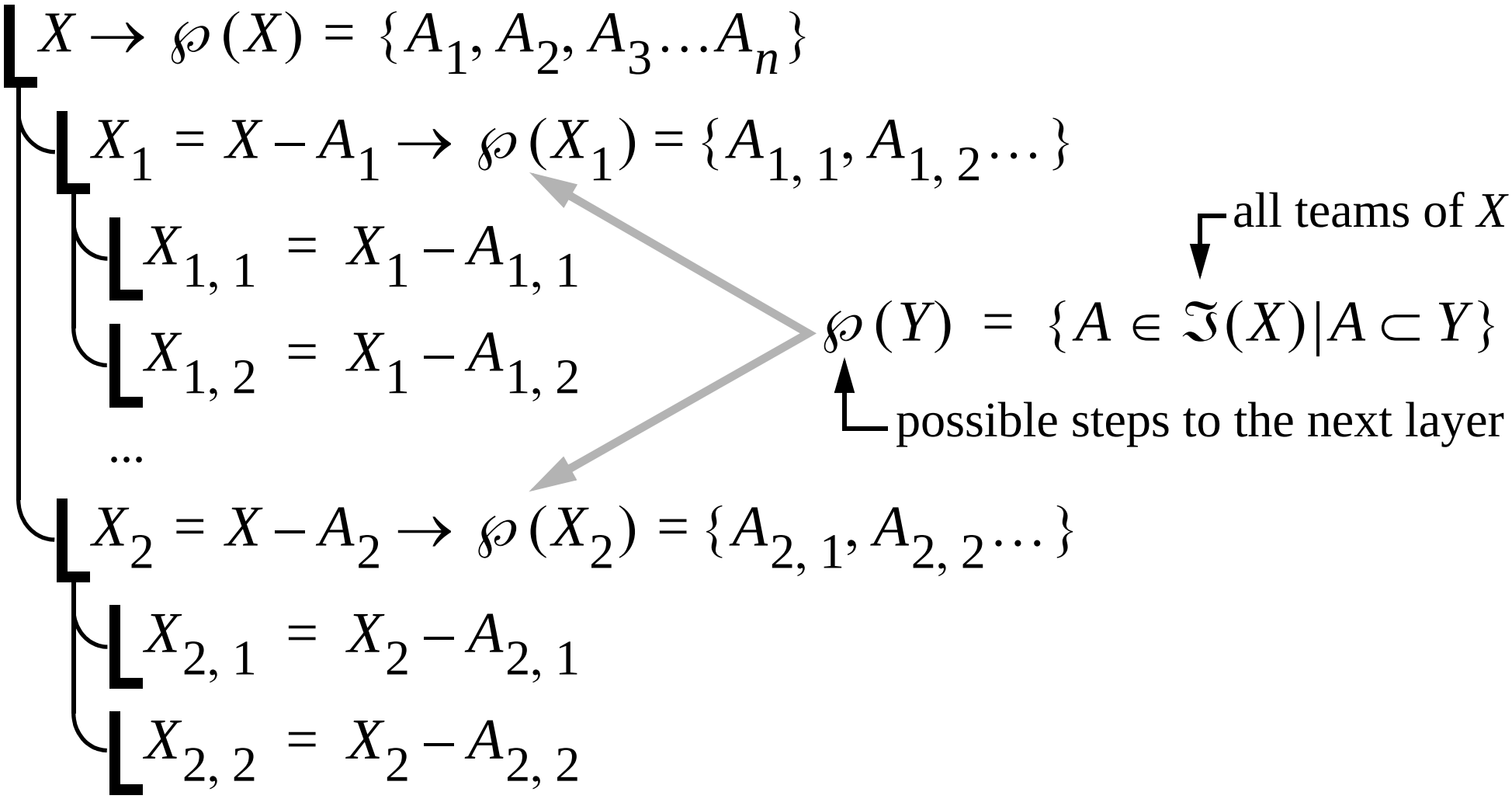
$\Im(X)$, all teams of the traffic X

- ● - transfer x
- ● - transfers congesting with x
- ○ - transfers non-congesting with x

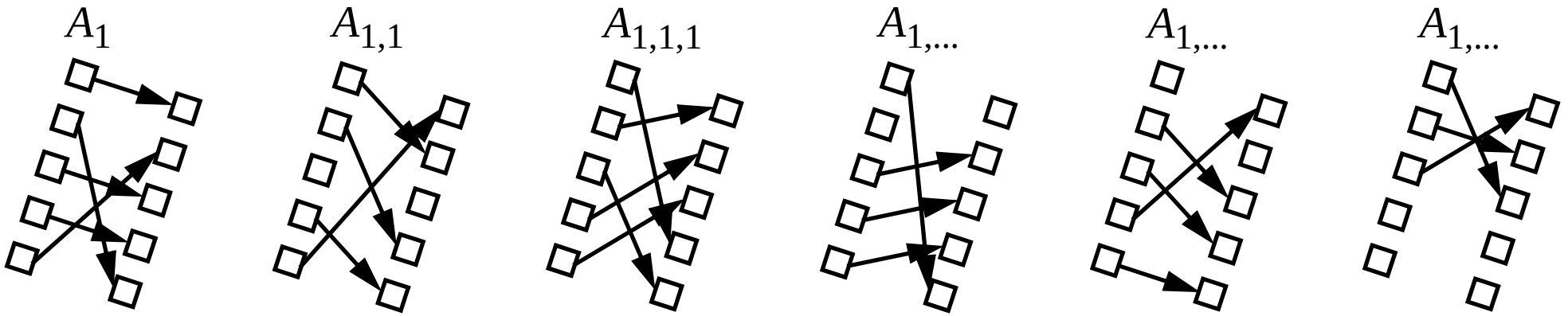


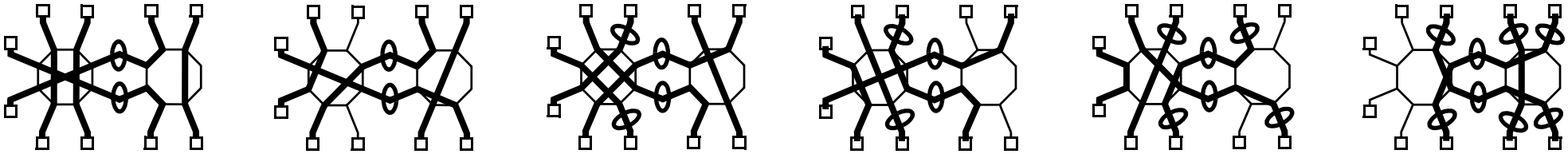
- To cover the full solution space when constructing a liquid schedule an efficient technique obtaining the whole set of possible teams of a traffic is required.
- We designed an efficient algorithm enumerating all teams of a traffic traversing each team once and only once.
- This algorithm obtains each team by subsequent partitioning of the set of all teams.
- We introduced triplets consisting of subsets of the traffic, representing one-by-one partitions of the set of all teams.

Liquid schedule search tree



Additional bottlenecks





2 bottlenecks
$\Lambda(X)=6$

2 bottlenecks
$\Lambda(X_1)=5$

4 bottlenecks
$\Lambda(X_{1,1})=4$

4 bottlenecks
$\Lambda(X_{1,...})=3$

6 bottlenecks
$\Lambda(X_{1,...})=2$

8 bottlenecks
$\Lambda(X_{1,...})=1$

$X_{1,1} = X_1 - A_{1,1}$ (16 transfers)

$X_1 = X - A_1$ (20 transfers)

X (25 transfers)

Prediction of dead-ends and search optimization

- When a team of transfers is carried out - for the remaining traffic we have the same bottleneck links as before - with possibly new additionally emerged bottleneck links.
- Considering new bottleneck links (at every step of construction) in the choice of the further teams substantially reduces the search space.
- Team is a collection of simultaneous transmissions using all bottlenecks of the network. Teams are full if they congest with all other transmissions of the traffic.
- Limiting our choice with only full teams additionally reduces the search space without affecting the solution space.

Liquid schedules construction

teams of the reduced traffic

$$\Im(Y) \subset \{A \in \Im(X) | A \subset Y\}$$

original traffic's teams formed from the reduced traffic

$$\underbrace{\Im^{full}(Y)} \subset \Im(Y)$$

full teams of the reduced traffic

$$Choice = \wp(Y) = \{A \in \Im(X) | A \subset Y\} \rightarrow \wp(Y) = \Im(Y)$$

$$\downarrow$$

$$Choice = \wp(Y) = \Im(Y)$$

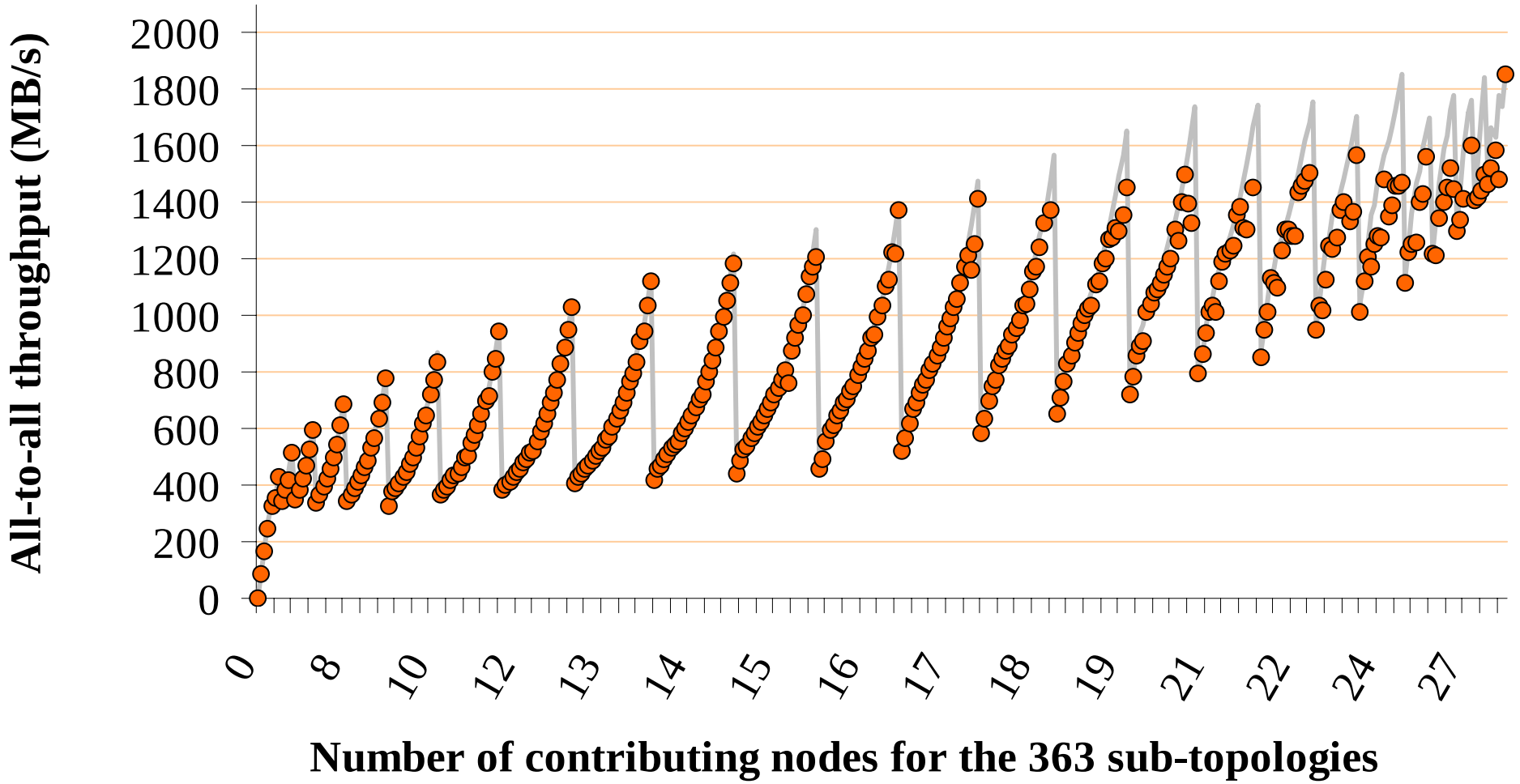
$$\downarrow$$

$$Choice = \wp(Y) = \Im^{full}(Y)$$

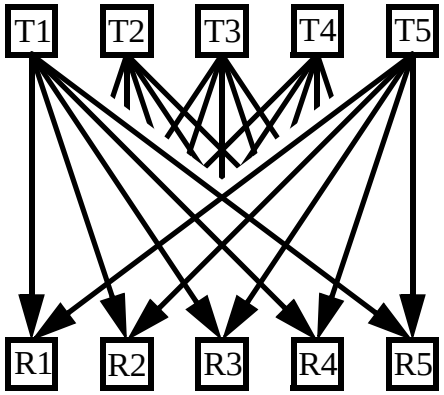
additionally decreasing the search space without affecting the solution space

For more than 90% of the test-bed topologies construction of a global liquid schedule is completed in a fraction of a second (less than 0.1s).

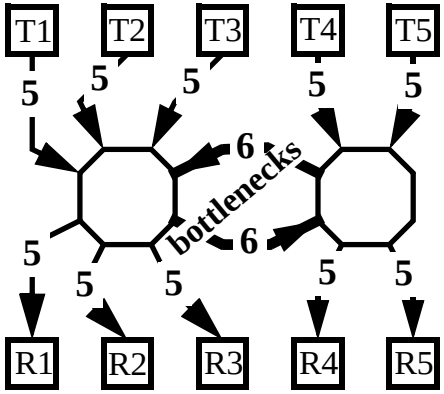
Results

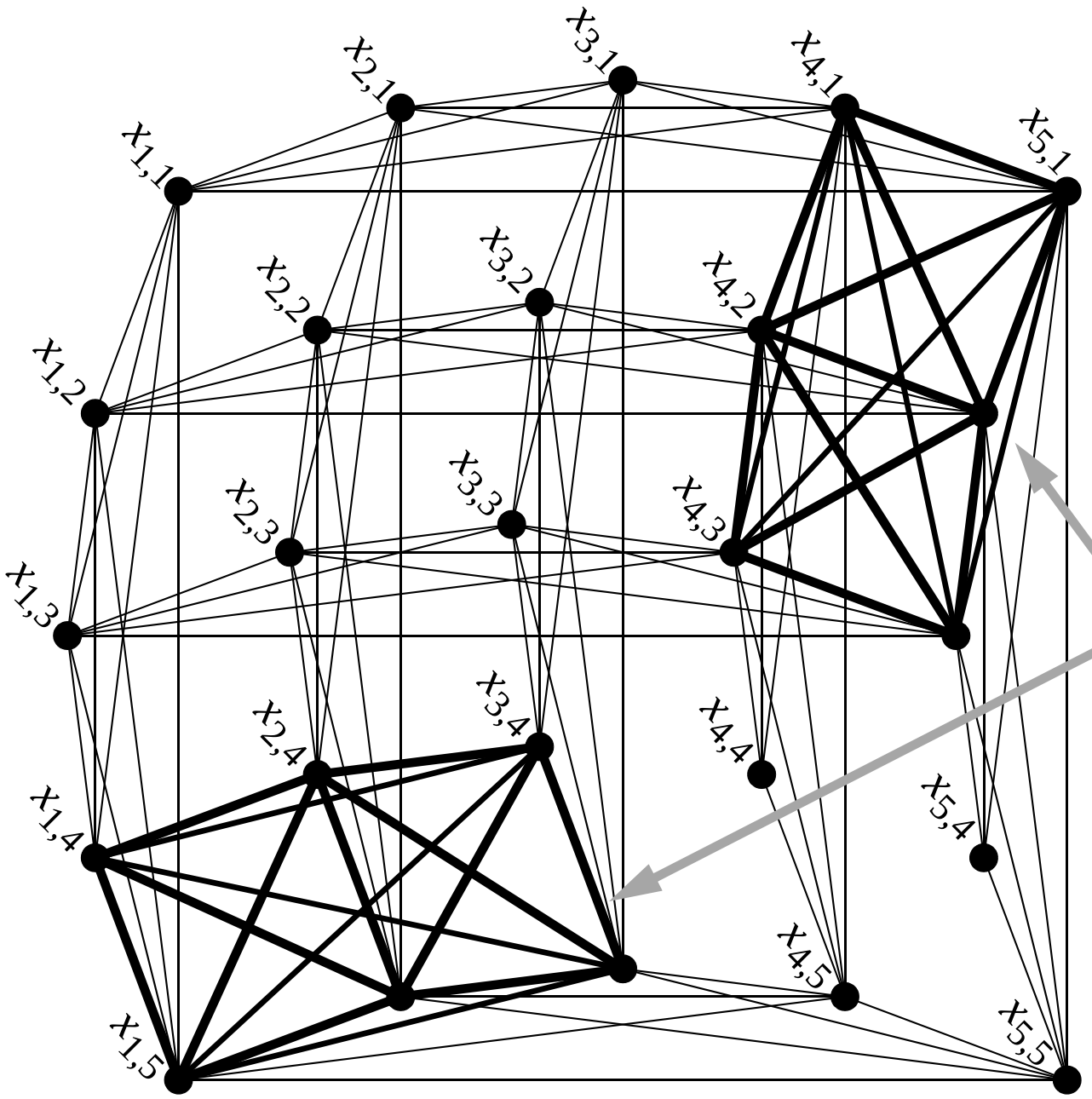

All-to-all throughput (MB/s)
2000
1800
1600
1400
1200
1000
800
600
400
200
0
0
8
10
12
13
14
15
16
17
18
19
21
22
24
27
Number of contributing nodes for the 363 sub-topologies
liquid throughput
carried out according to the liquid schedules

Congestion Graph



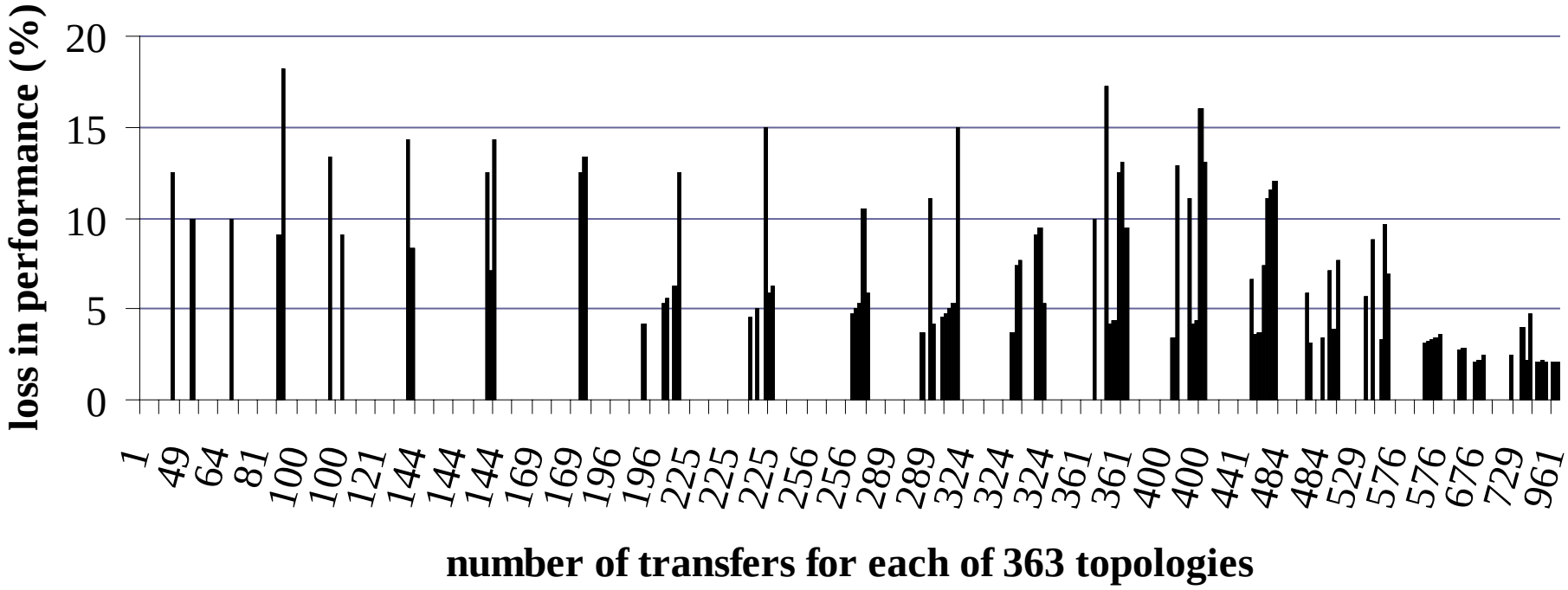
The 25 vertices of the graph represent the 25 transfers transfers. The edges represent congestion relations between transfers, i.e. each edge represents one or more communication links shared by two transfers.





Bold edges represent all congestions due to **bottleneck links**

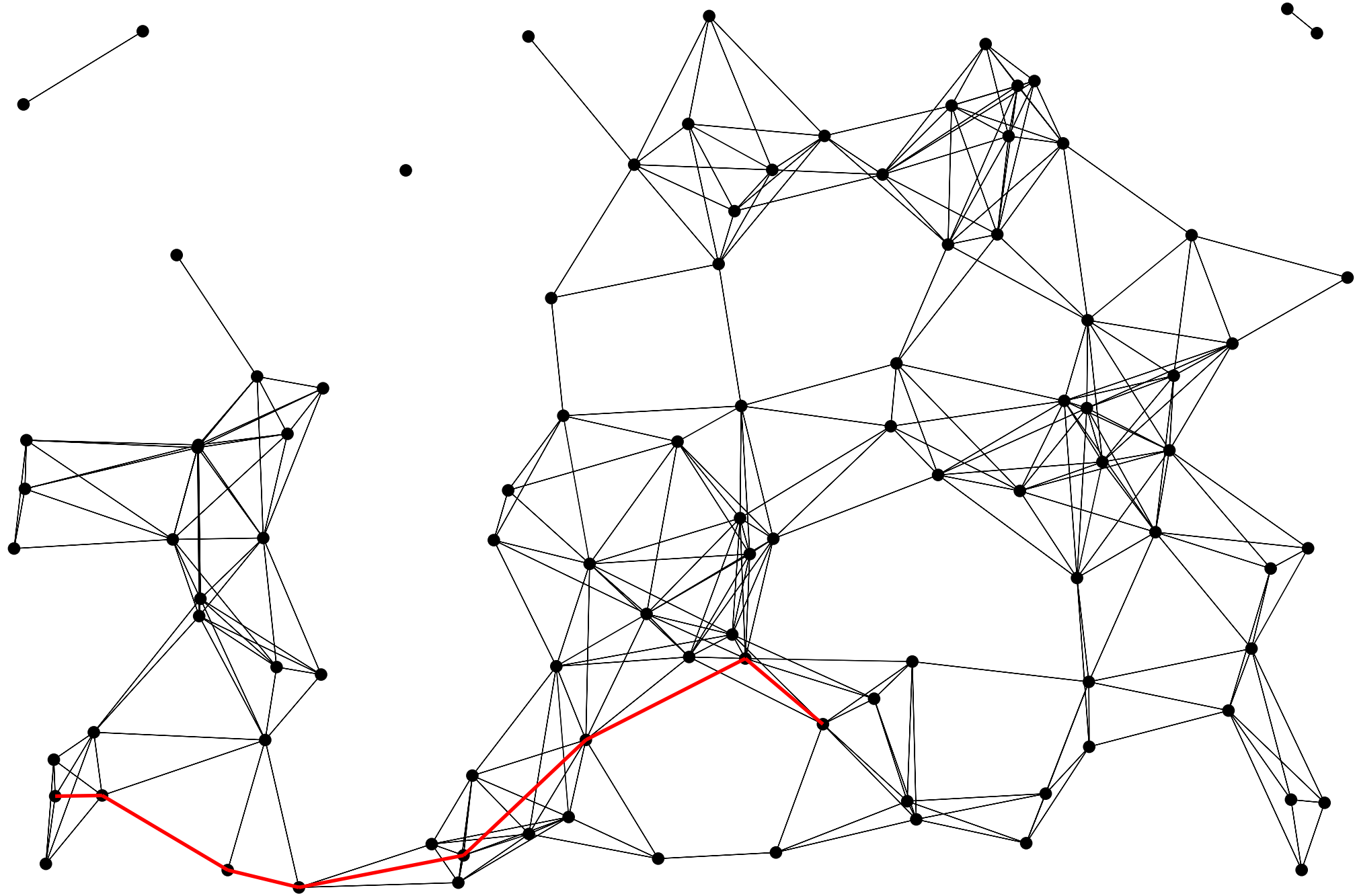
Loss of performance induced by schedules computed with a graph colouring heuristic algorithm



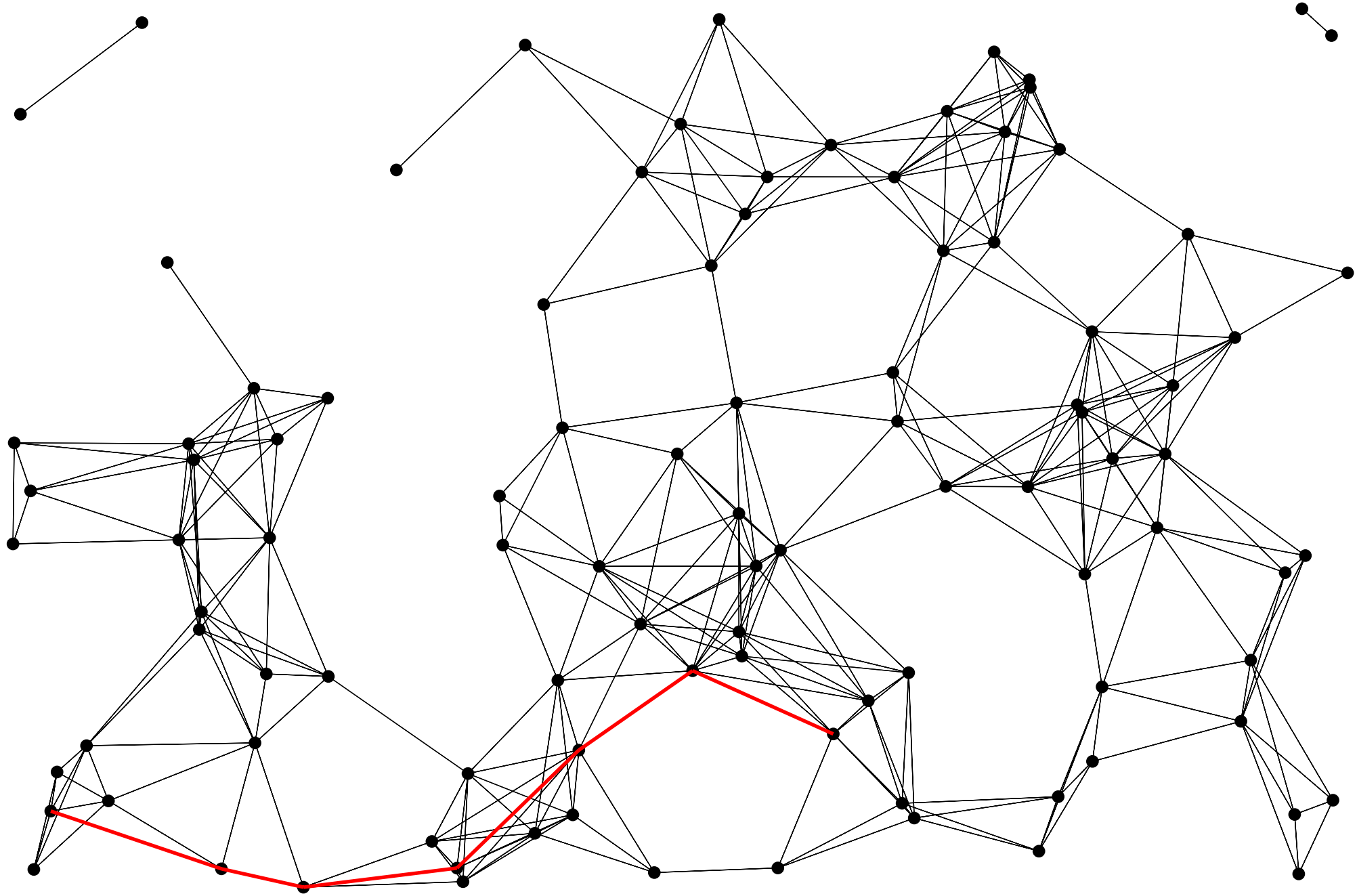
- For 74% of the topologies Dsatur algorithm does not induce a loss of performance.
- For 18% of topologies, the performance loss is bellow 10%.
- For 8% of topologies, the loss of performance is between 10% and 20%.

Conclusion and Future work

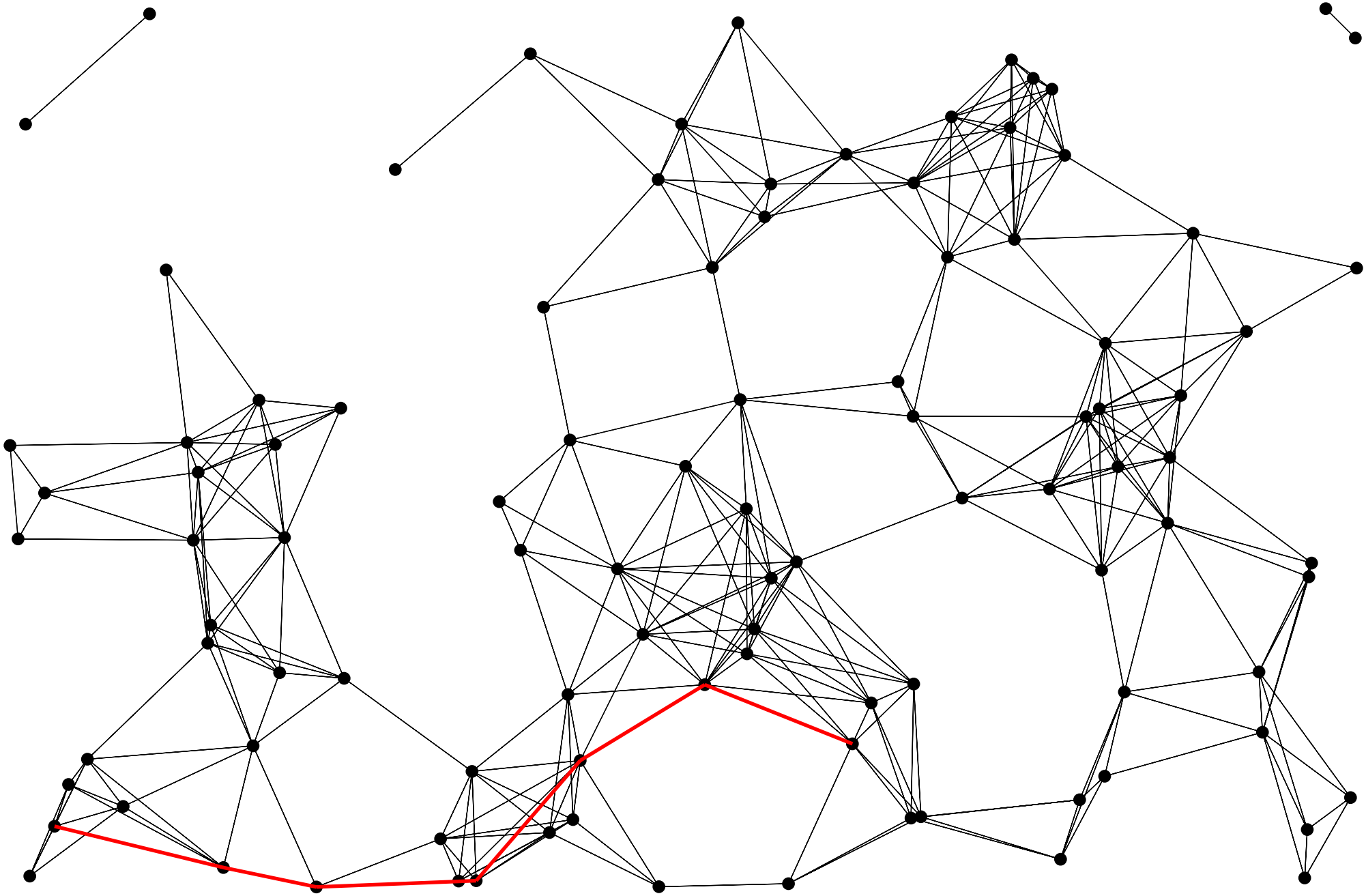
- Data exchanges relying on the liquid schedules may be carried out several times faster compared with topology-unaware schedules.
- Thanks to introduced theoretical model we considerably reduce the liquid schedule search space without affecting the solution space.
- Our method may be applied when high QoS and efficient bandwidth usage of a media is required for continuous streaming applications such as video and voice.
- Liquid scheduling is applicable for TDM wireless networks, optical networks or low latency wormhole/cut-through networks, such as Myrinet. Streams can be transmitted from edge to edge in large chunks, but global synchronization in the network is required.
- Future work: fault-tolerance of transmission by spacial diversification of routing paths. Example of an underlying network: a wireless ad-hoc mobile network seeking to provide end-to-node streaming services, such as packetized voice.



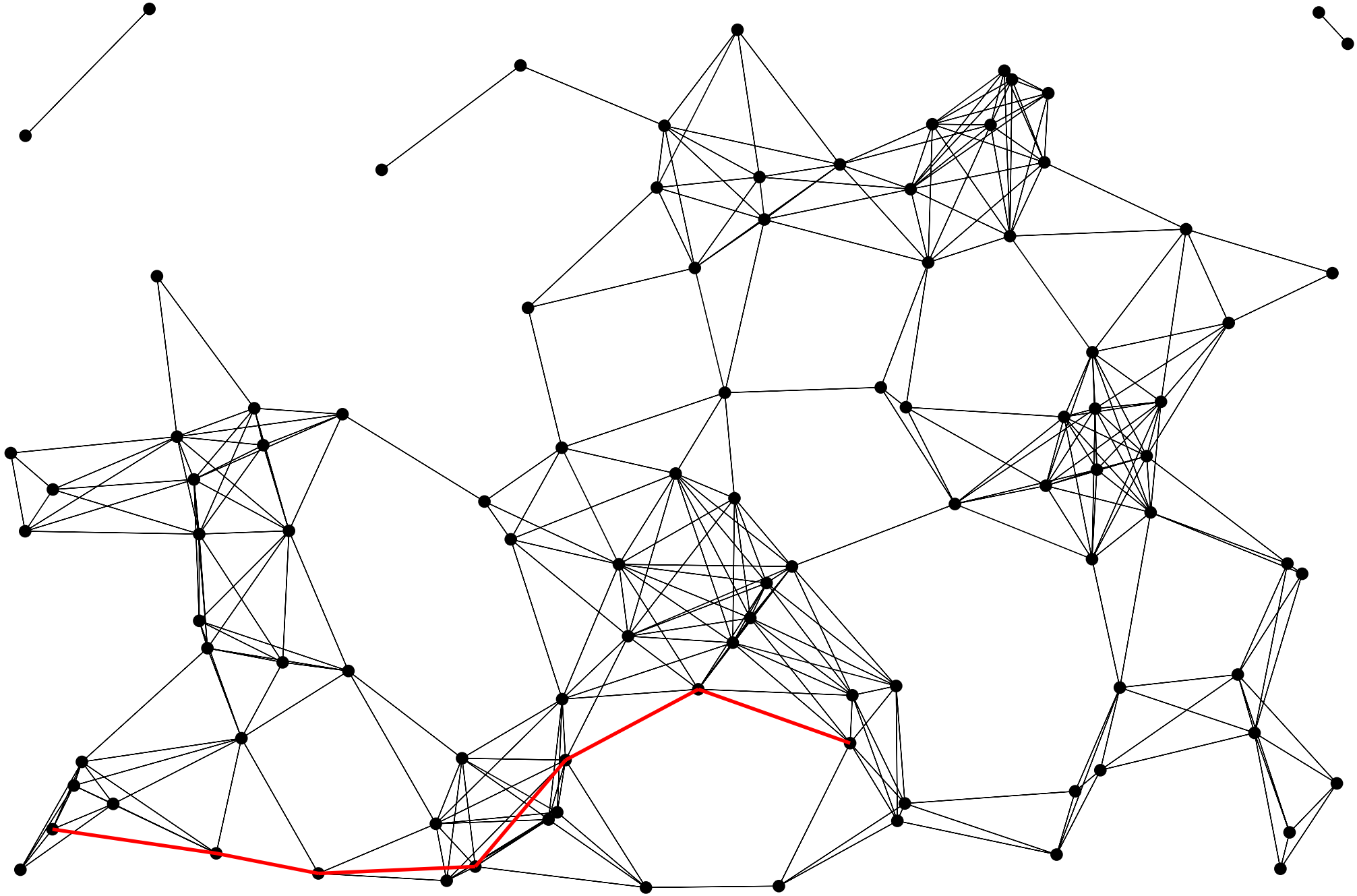
a network of 100 nodes and 668 links

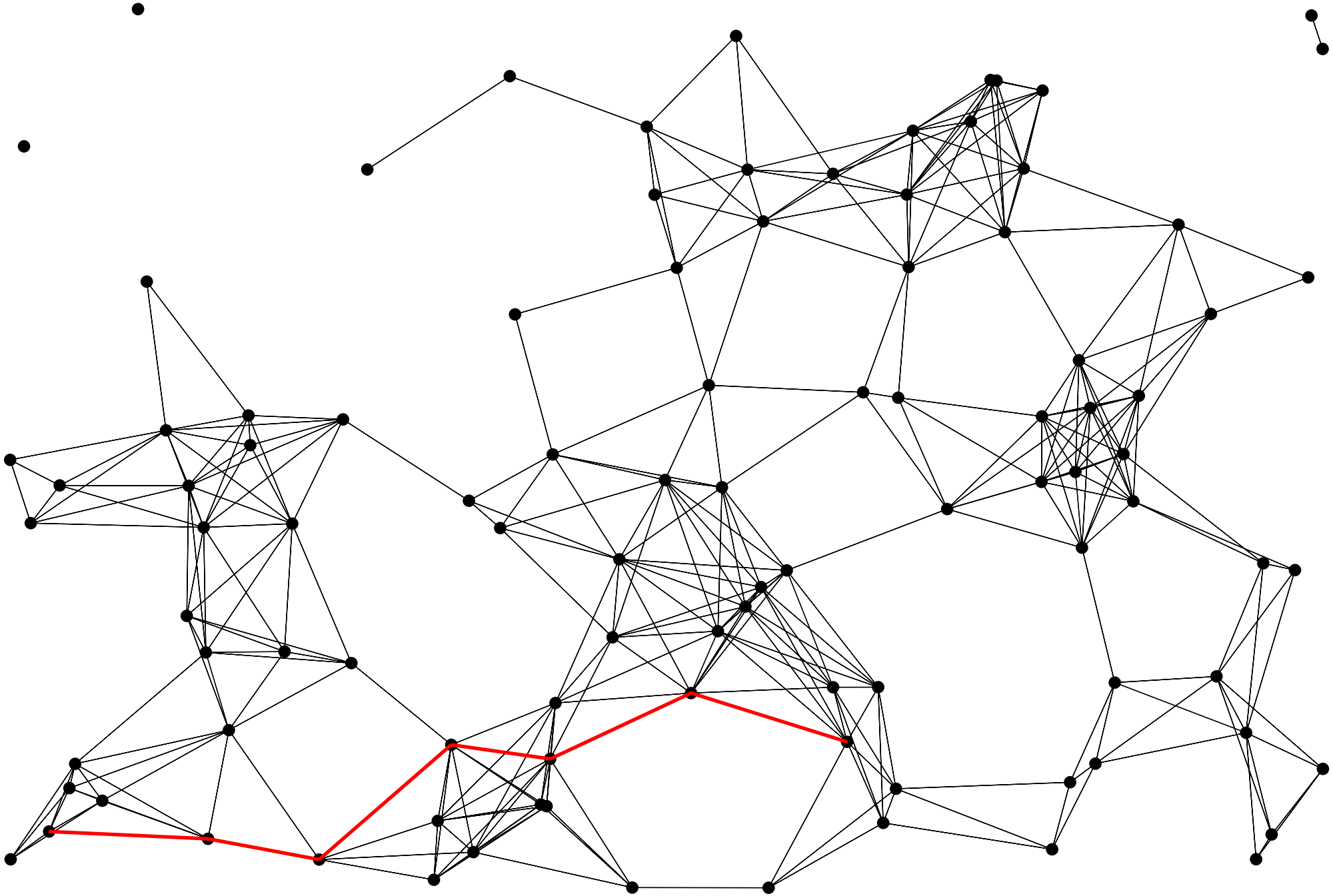


a network of 100 nodes and 682 links

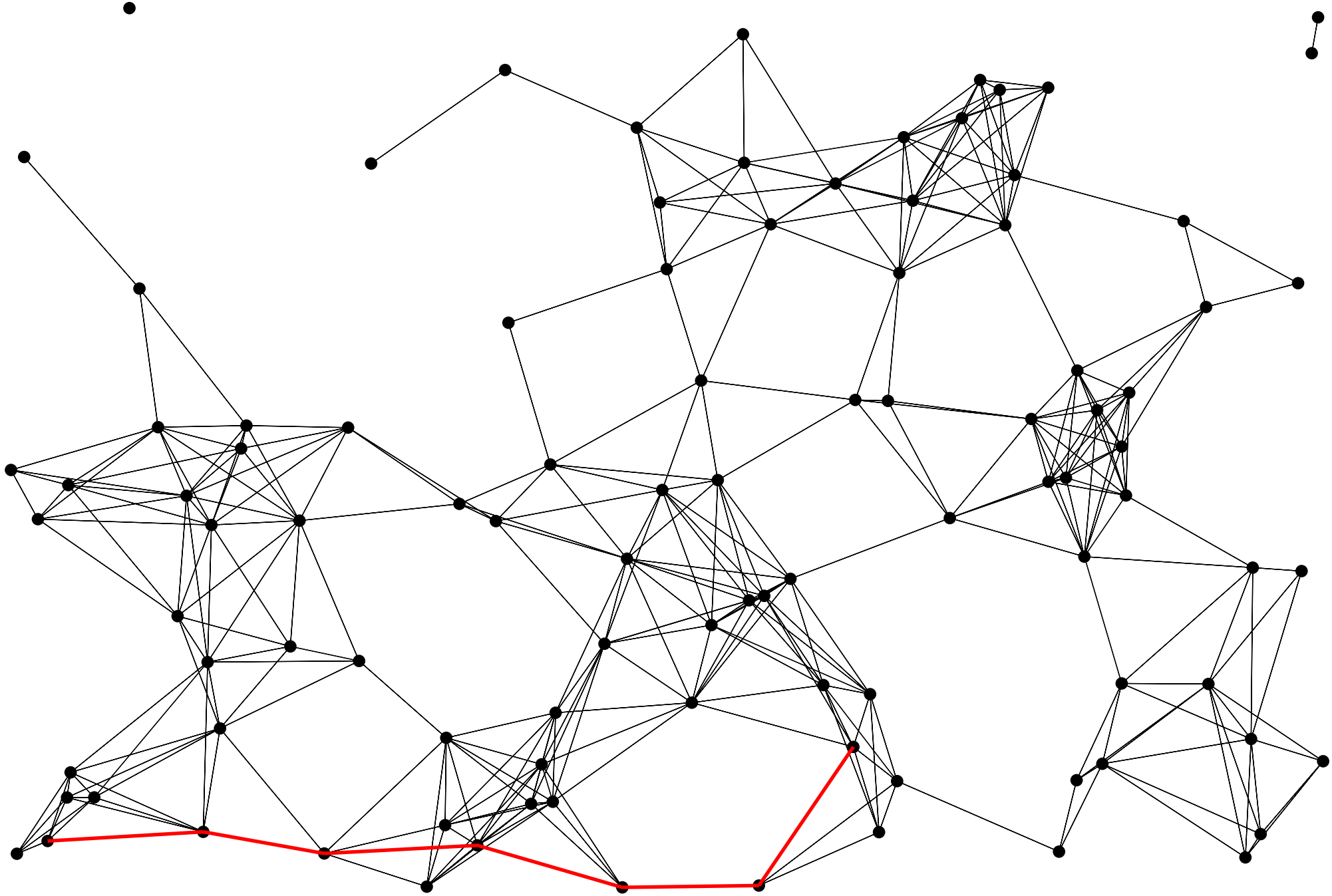


a network of 100 nodes and 682 links

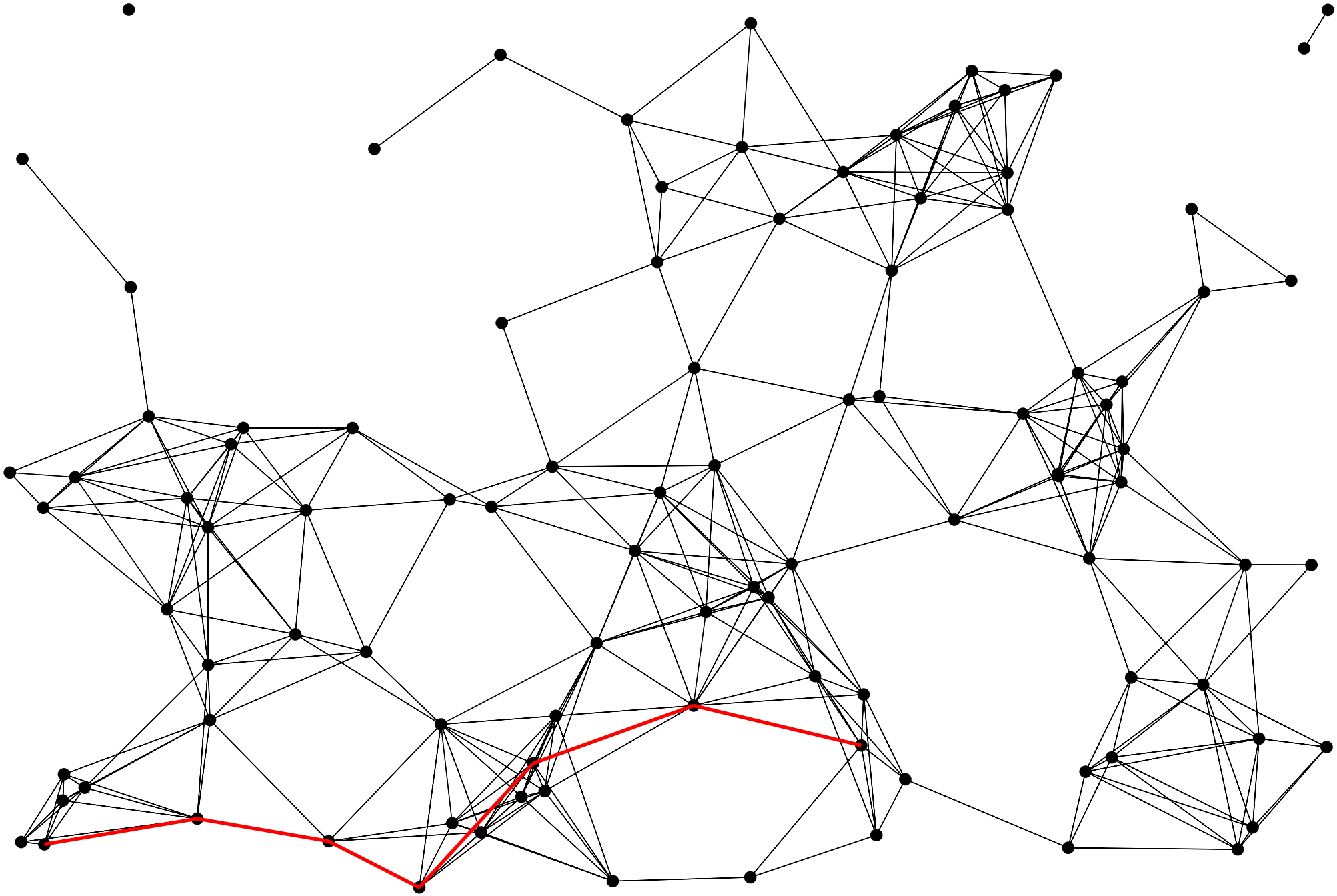

a network of 100 nodes and 694 links



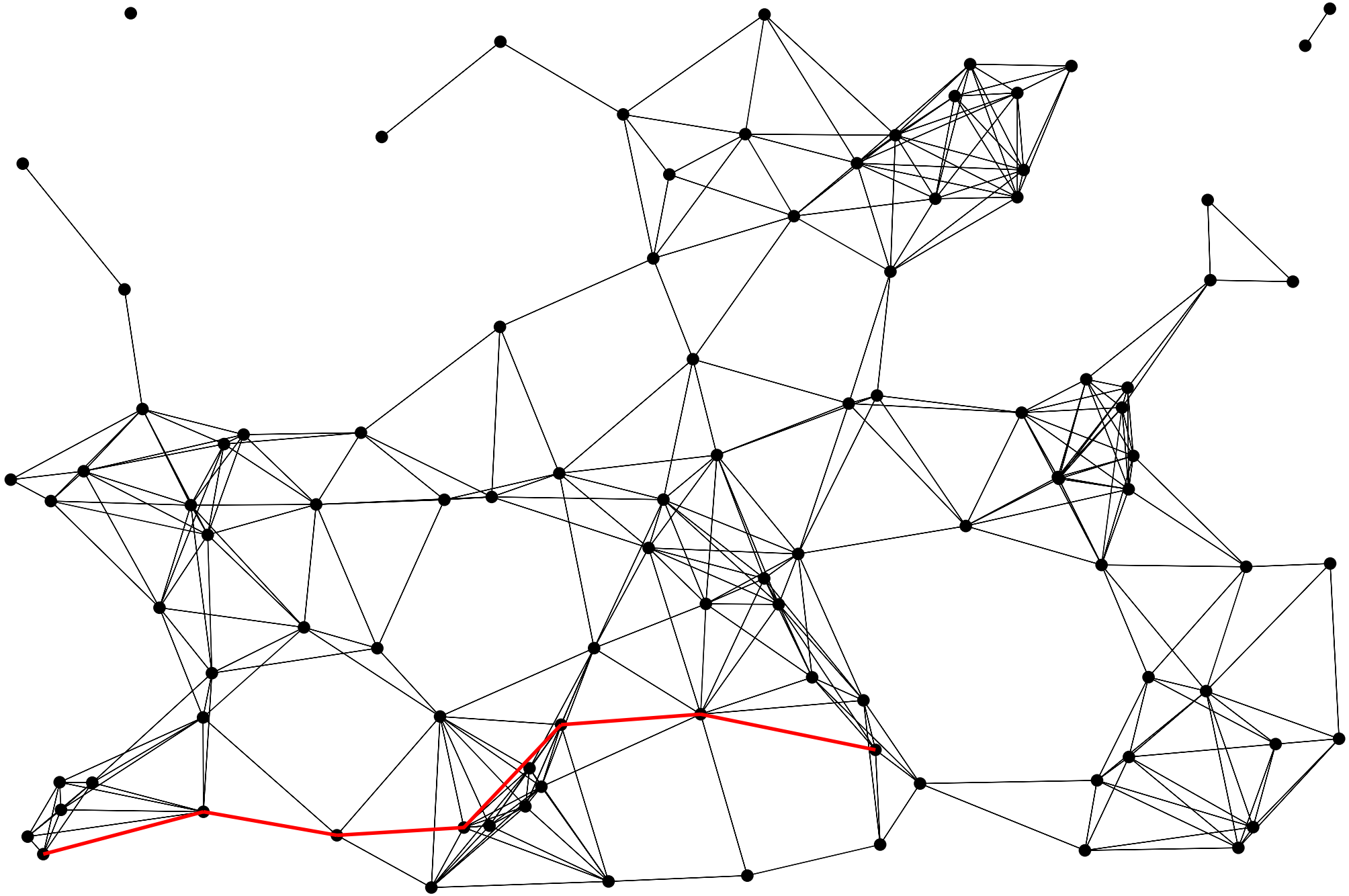
a network of 100 nodes and 688 links



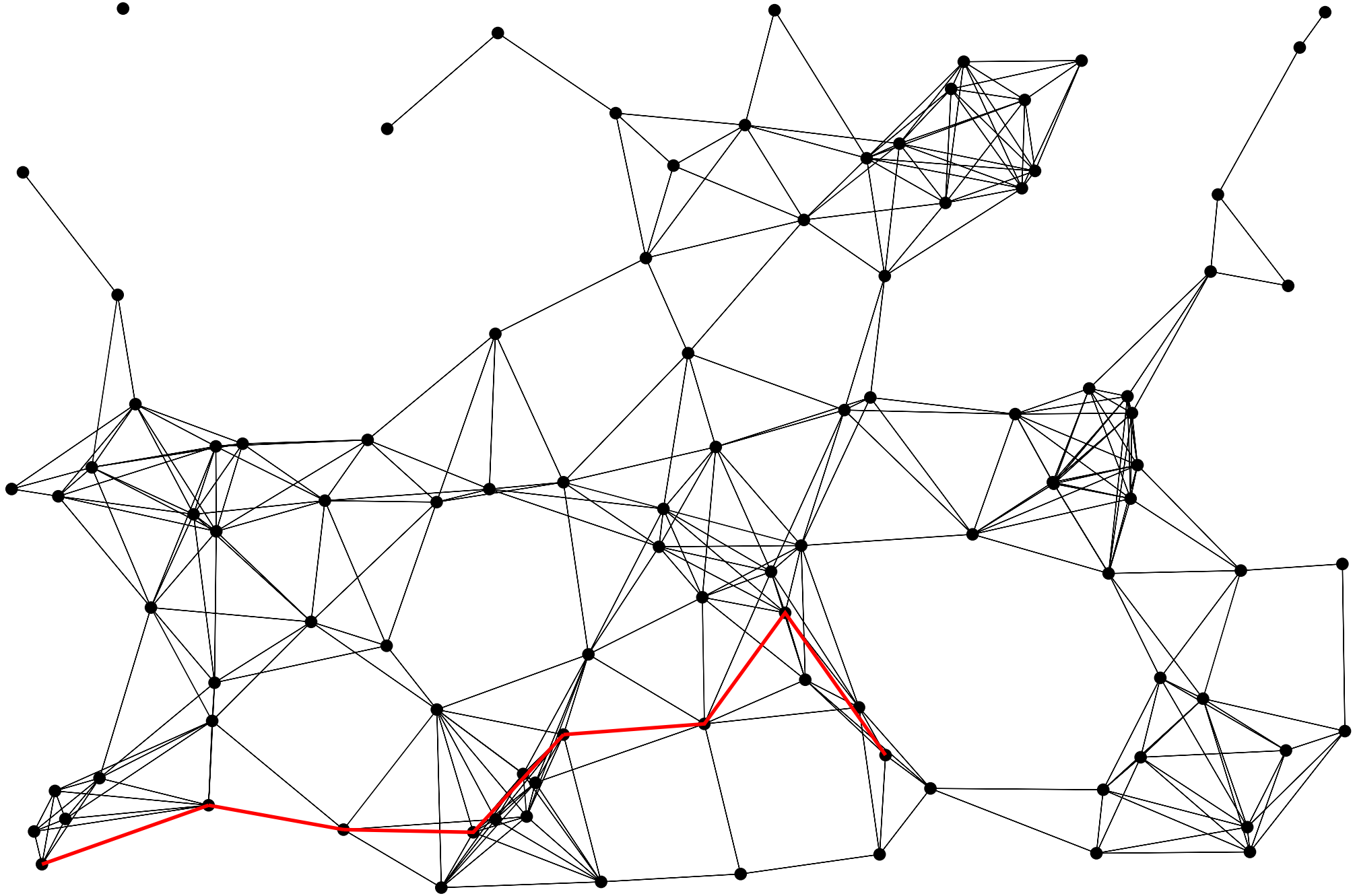
a network of 100 nodes and 702 links



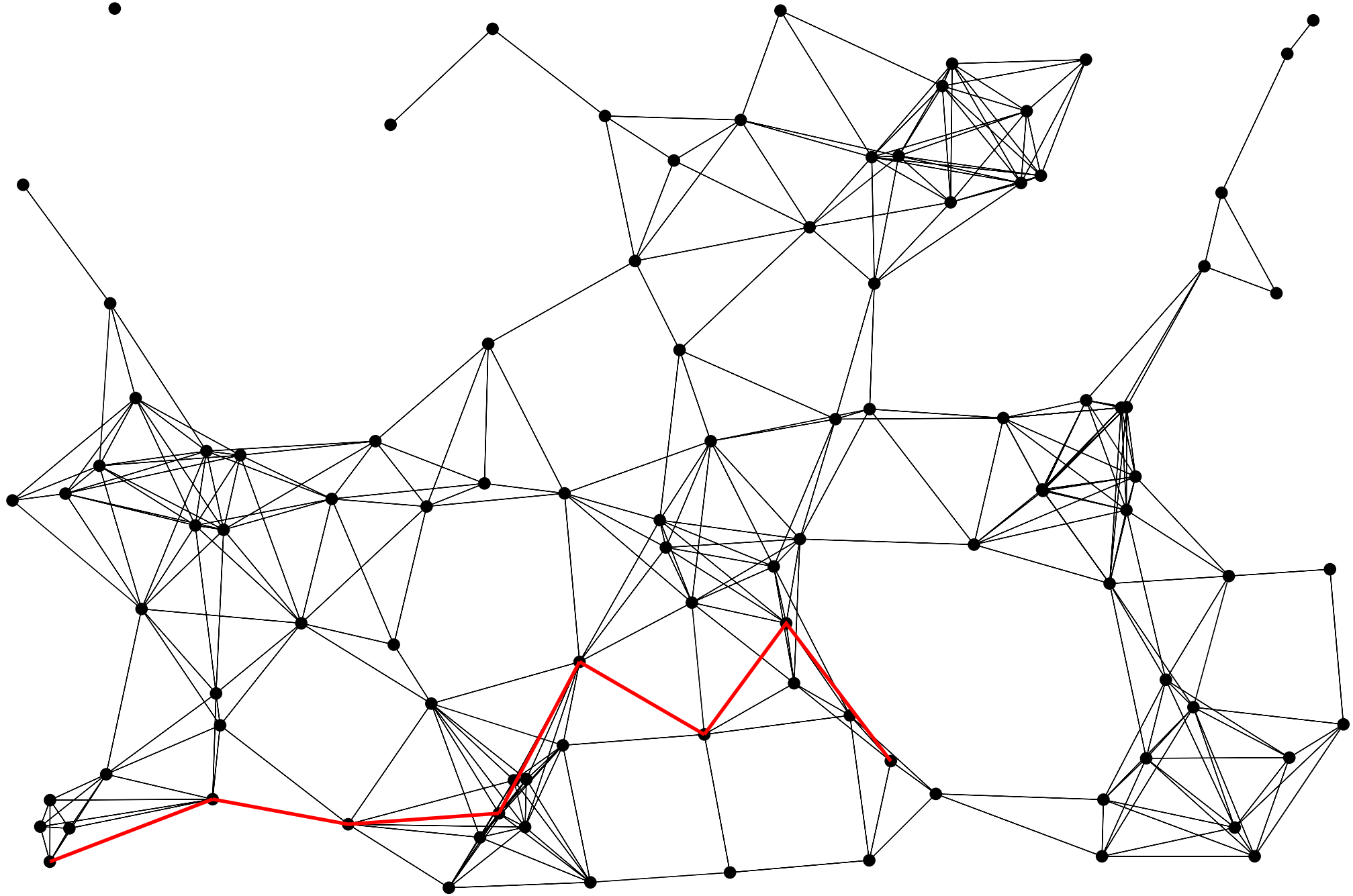
a network of 100 nodes and 708 links



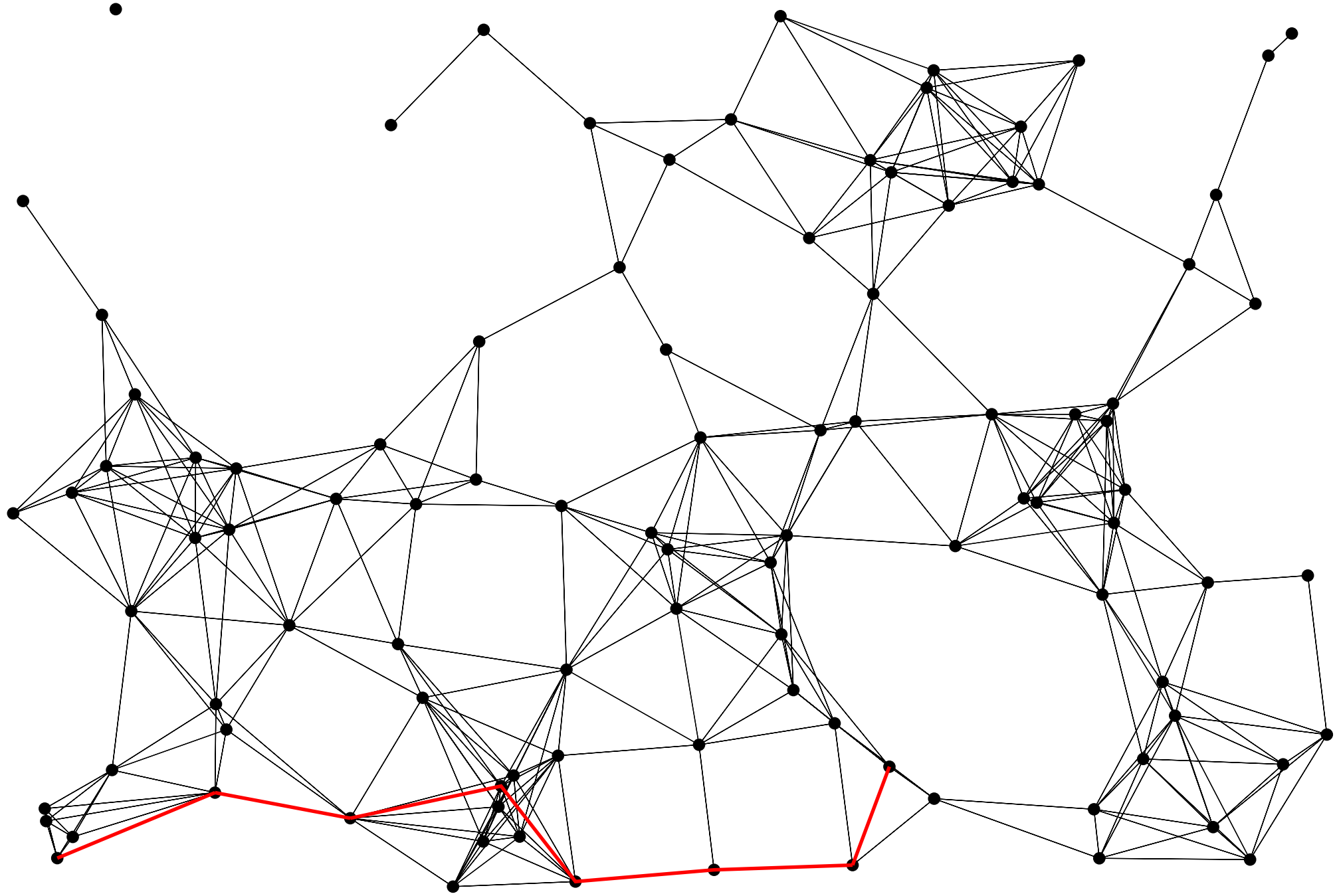
a network of 100 nodes and 702 links



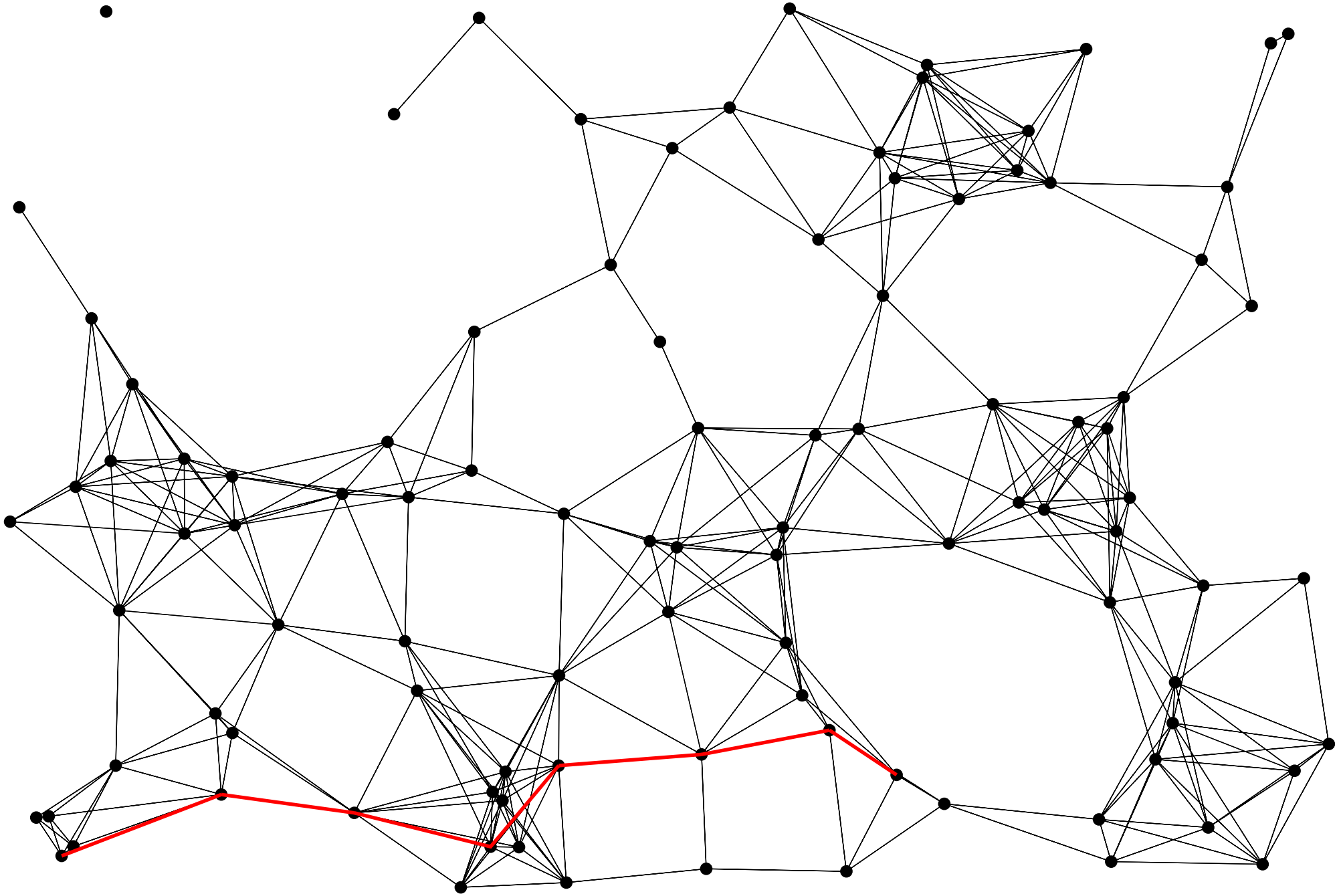
a network of 100 nodes and 708 links



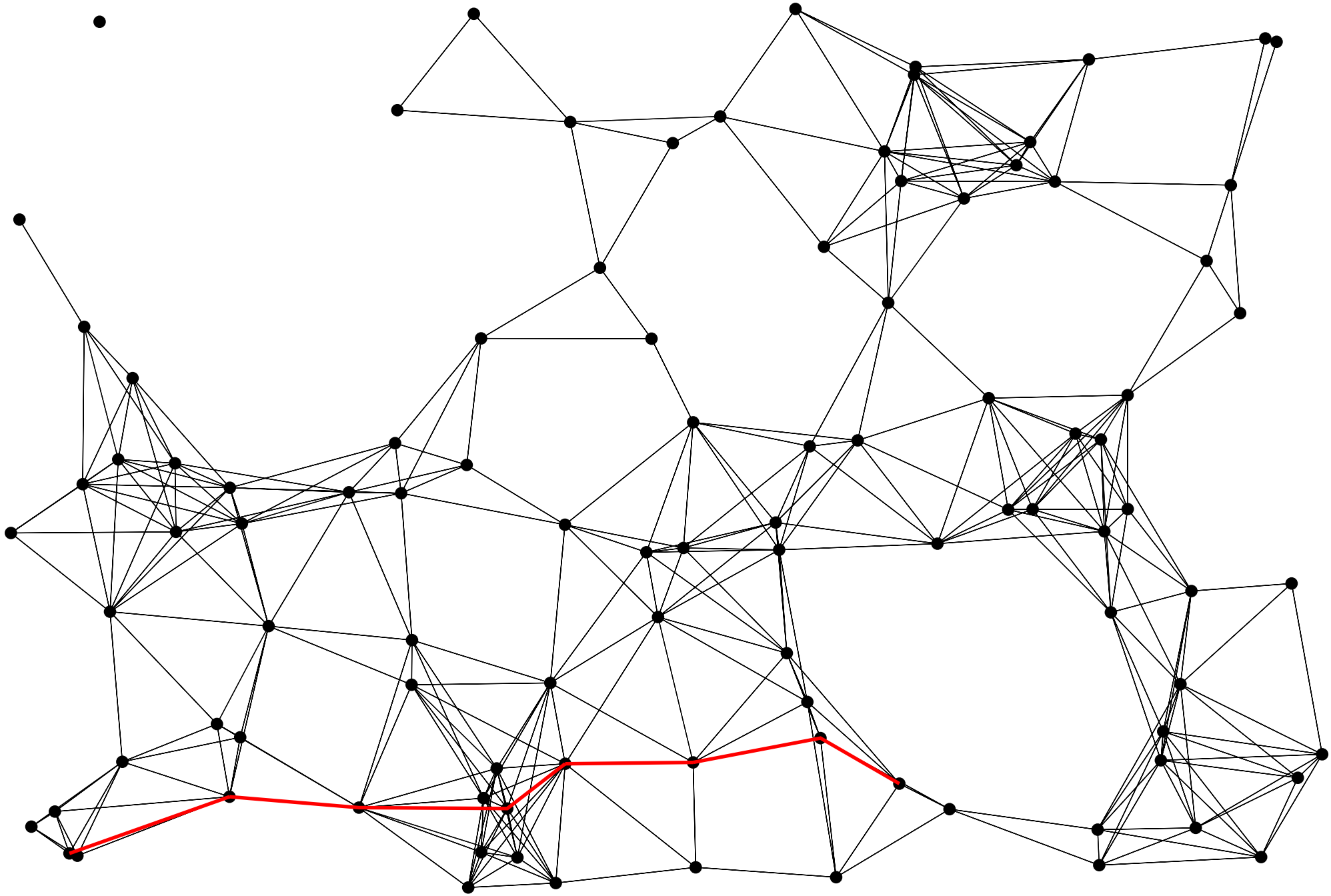
a network of 100 nodes and 708 links



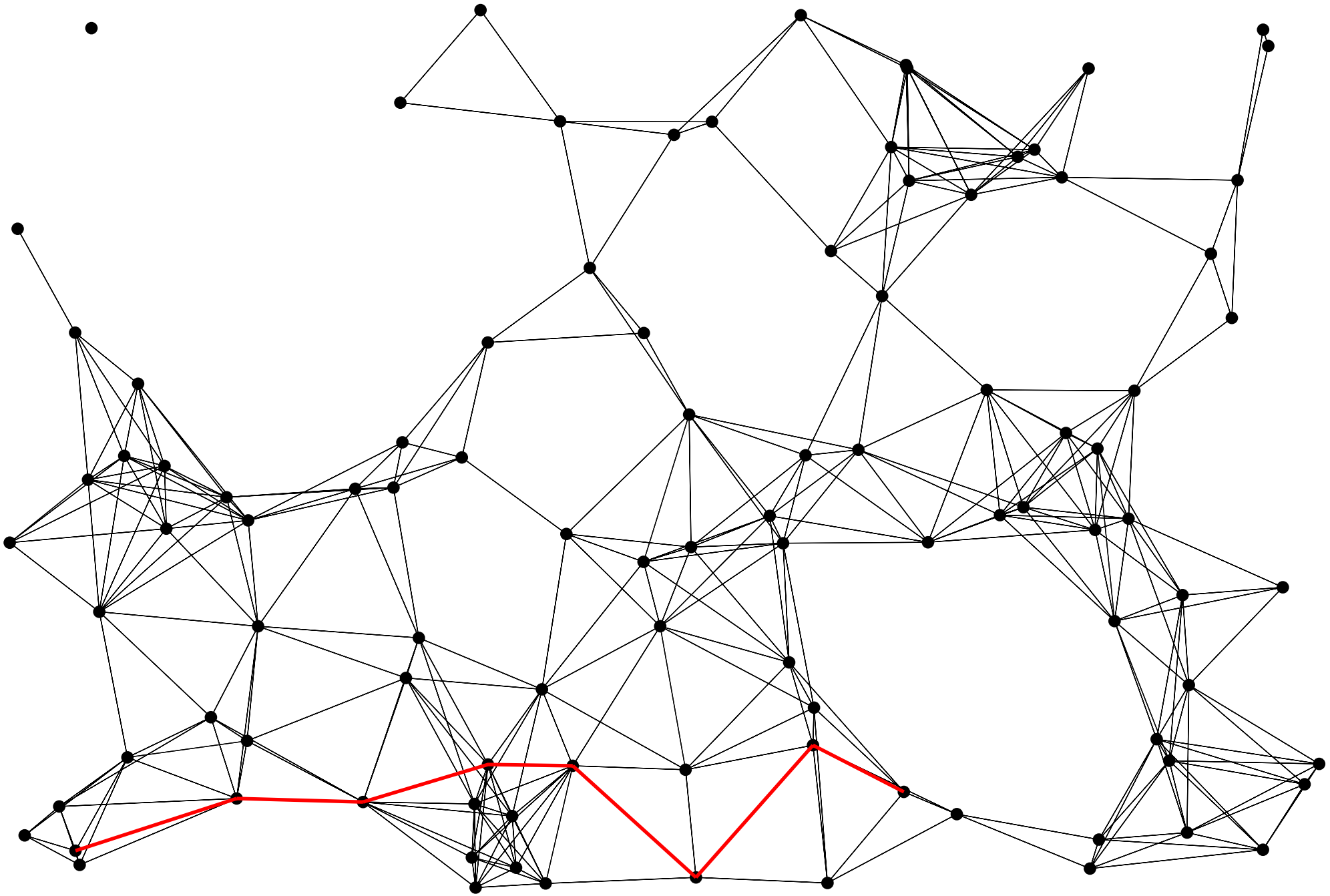
a network of 100 nodes and 714 links



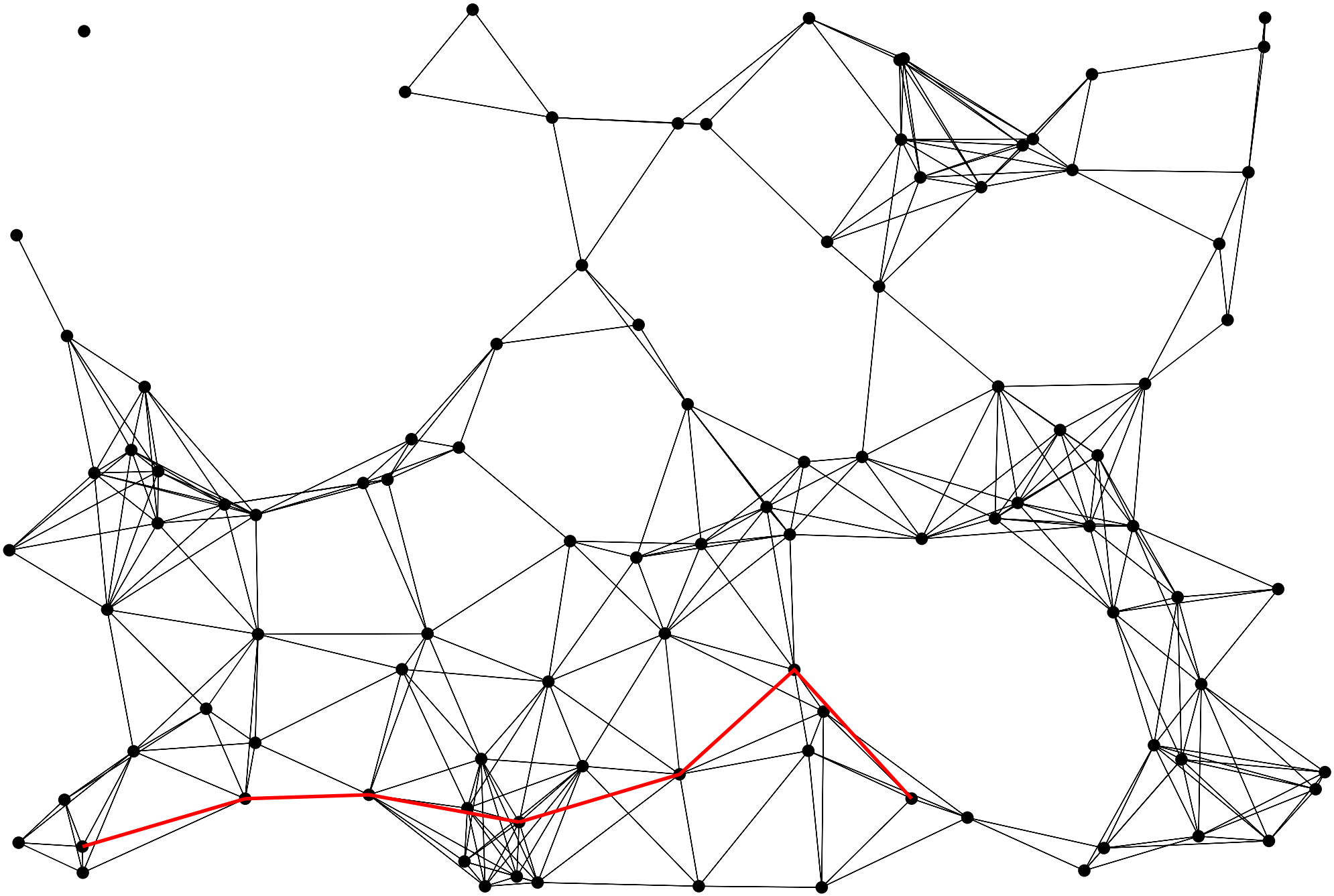
a network of 100 nodes and 718 links



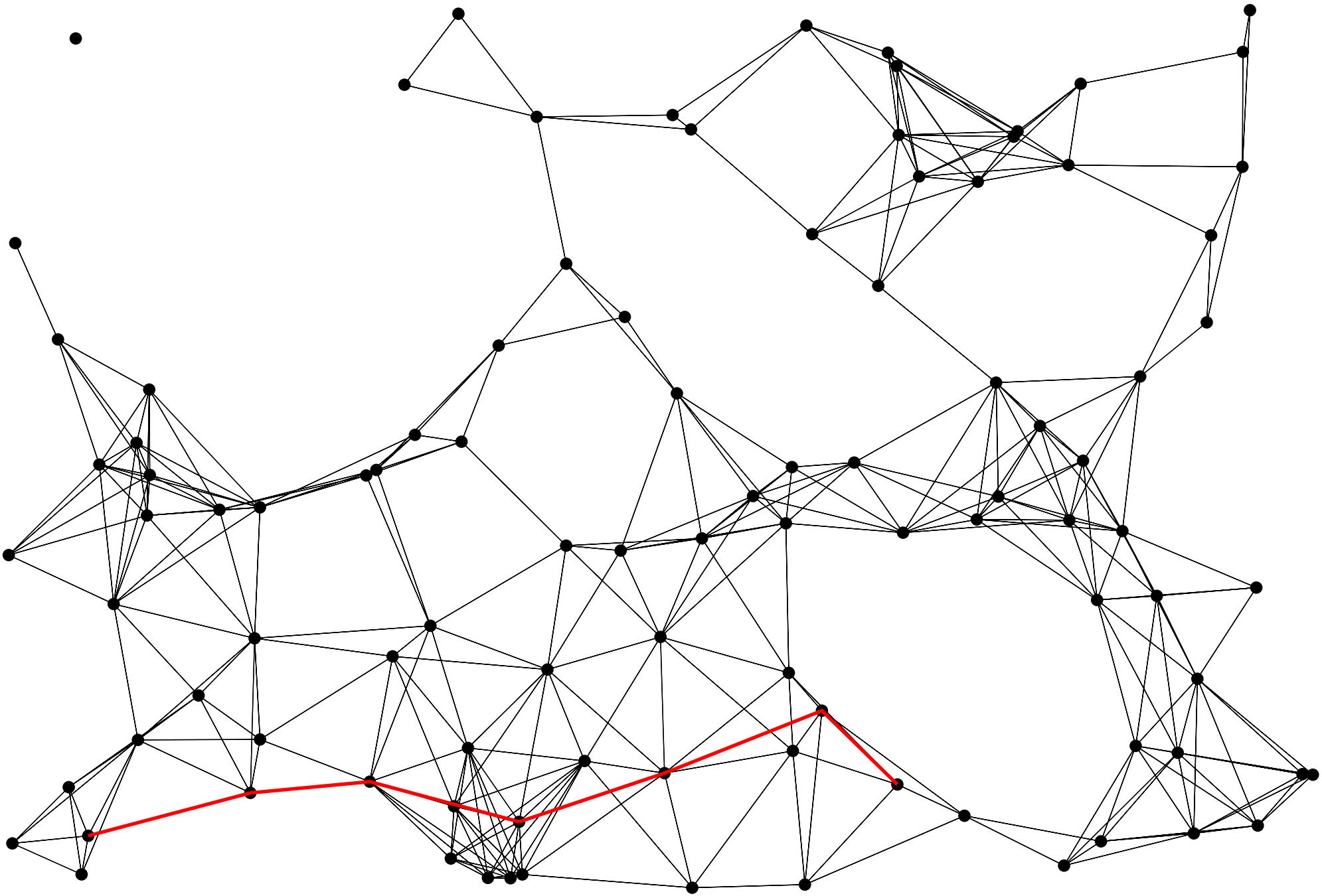
a network of 100 nodes and 718 links



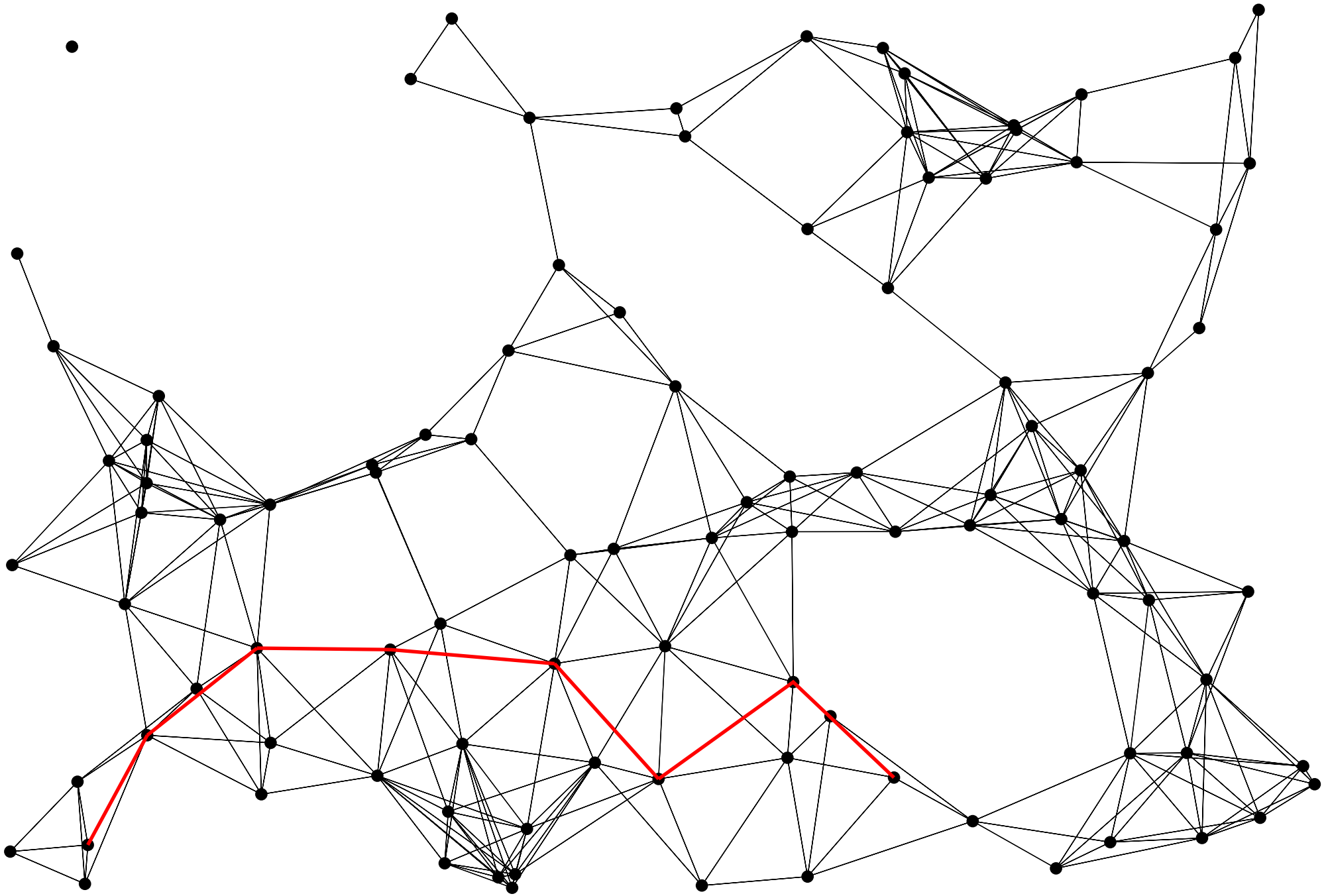
a network of 100 nodes and 712 links



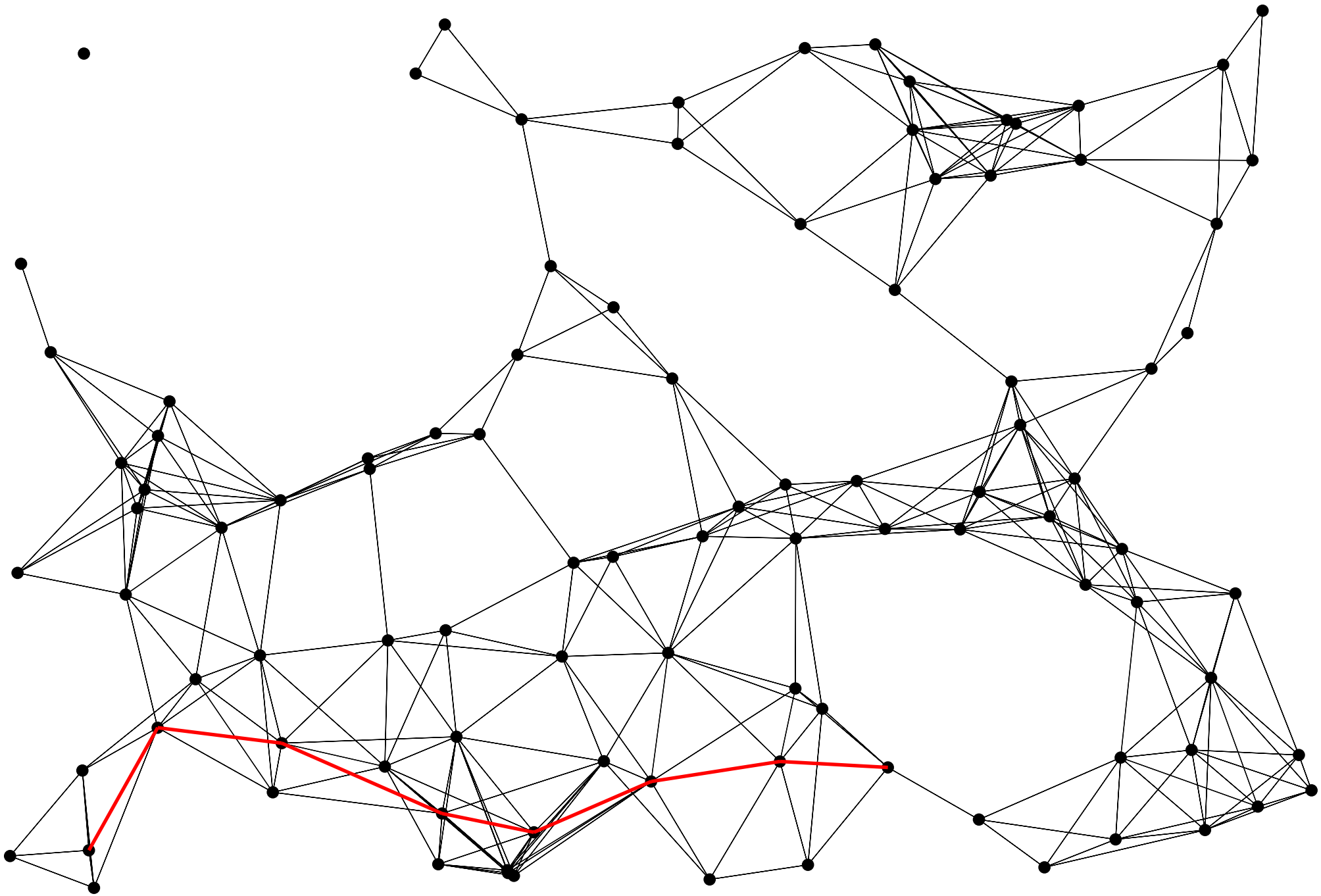
a network of 100 nodes and 704 links



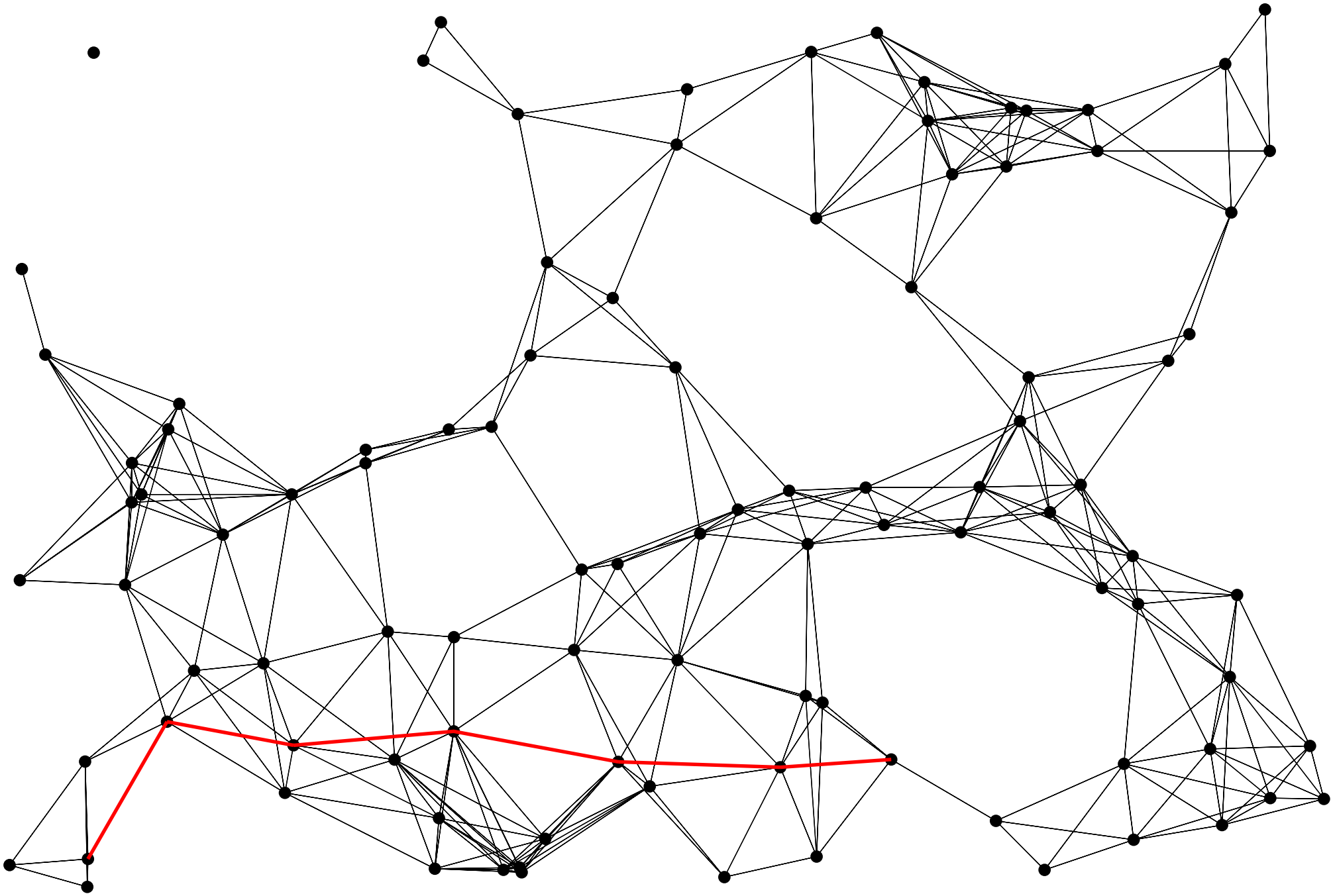
a network of 100 nodes and 692 links



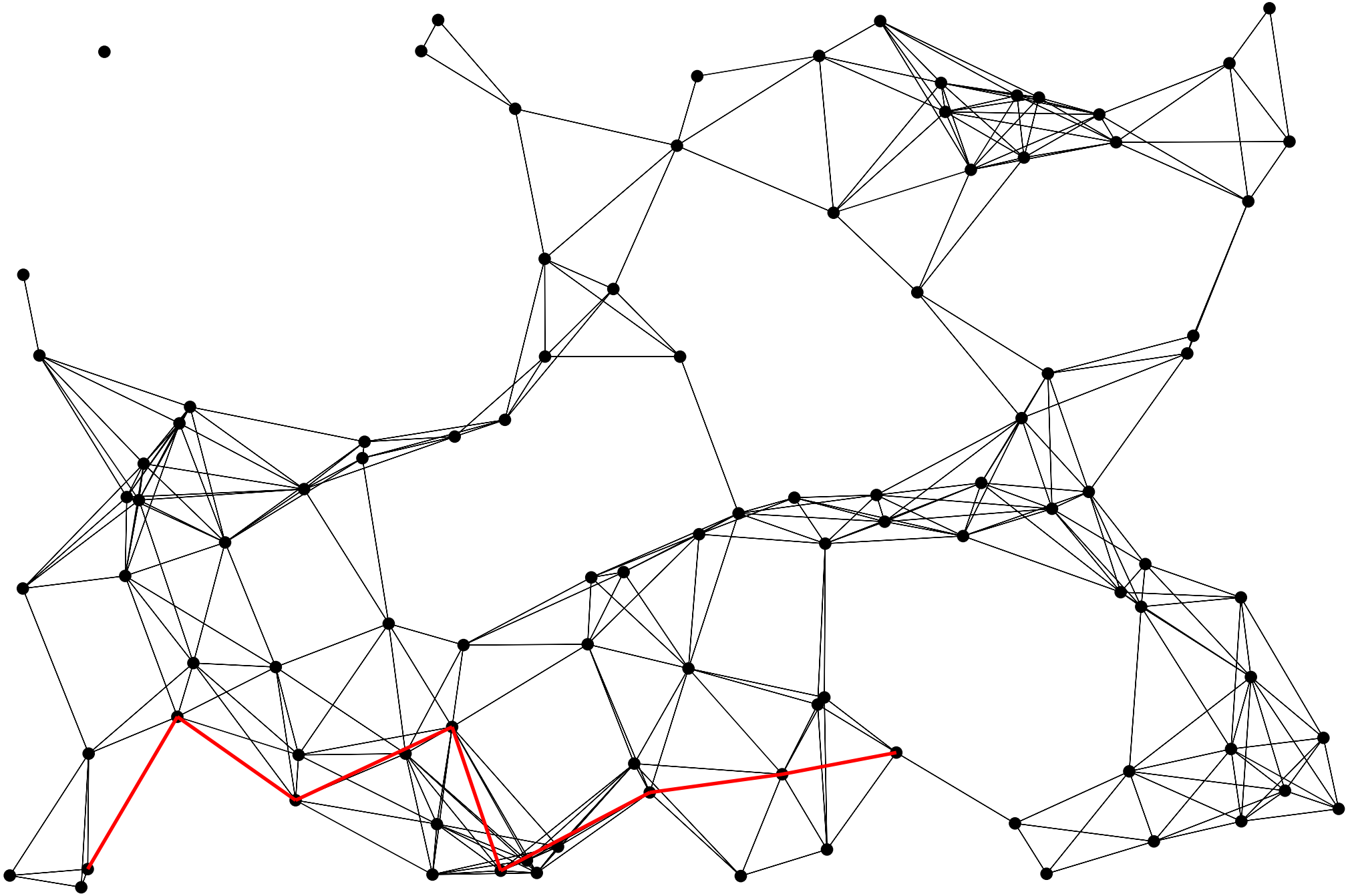
a network of 100 nodes and 678 links



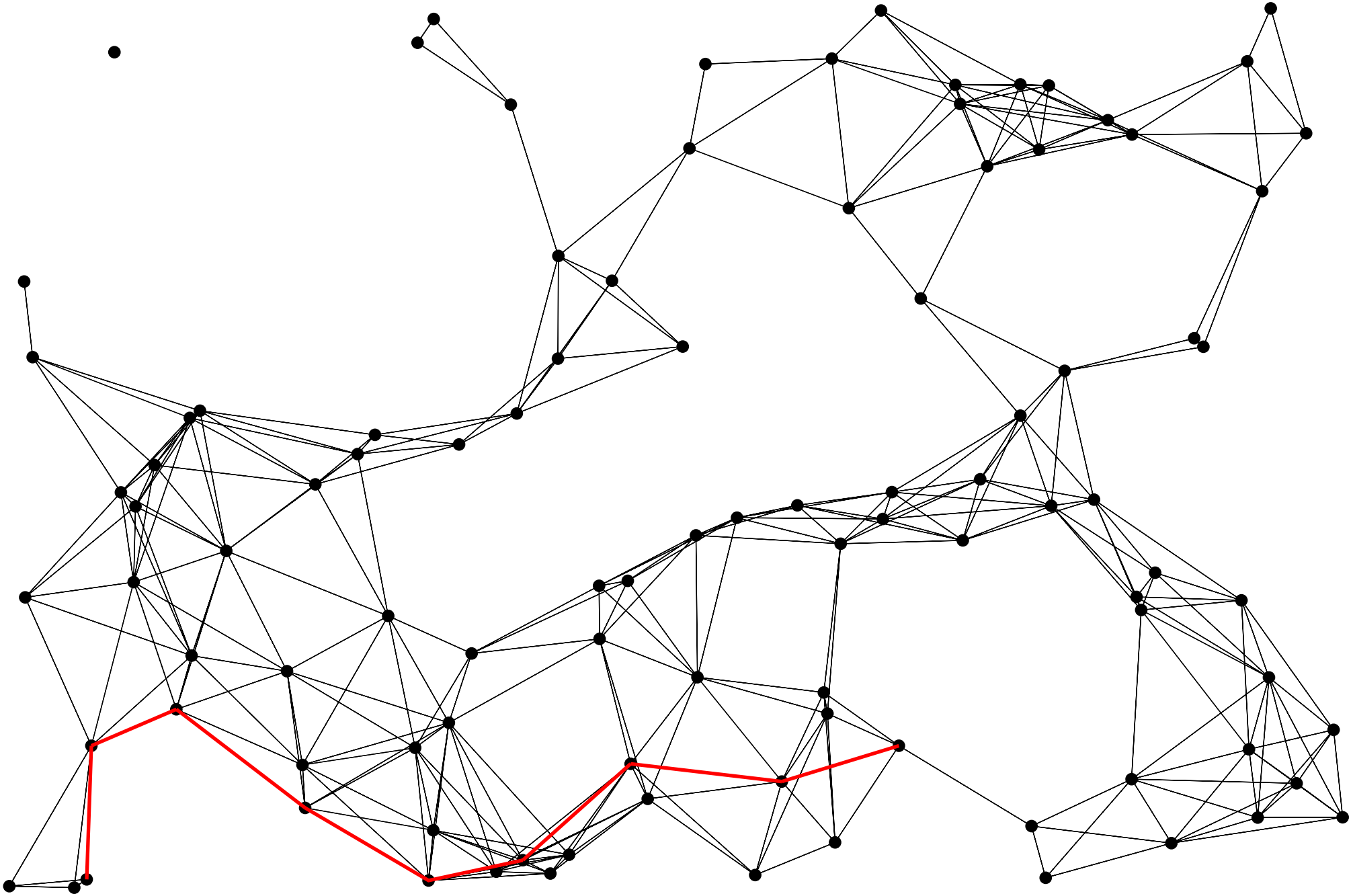
a network of 100 nodes and 680 links



a network of 100 nodes and 694 links

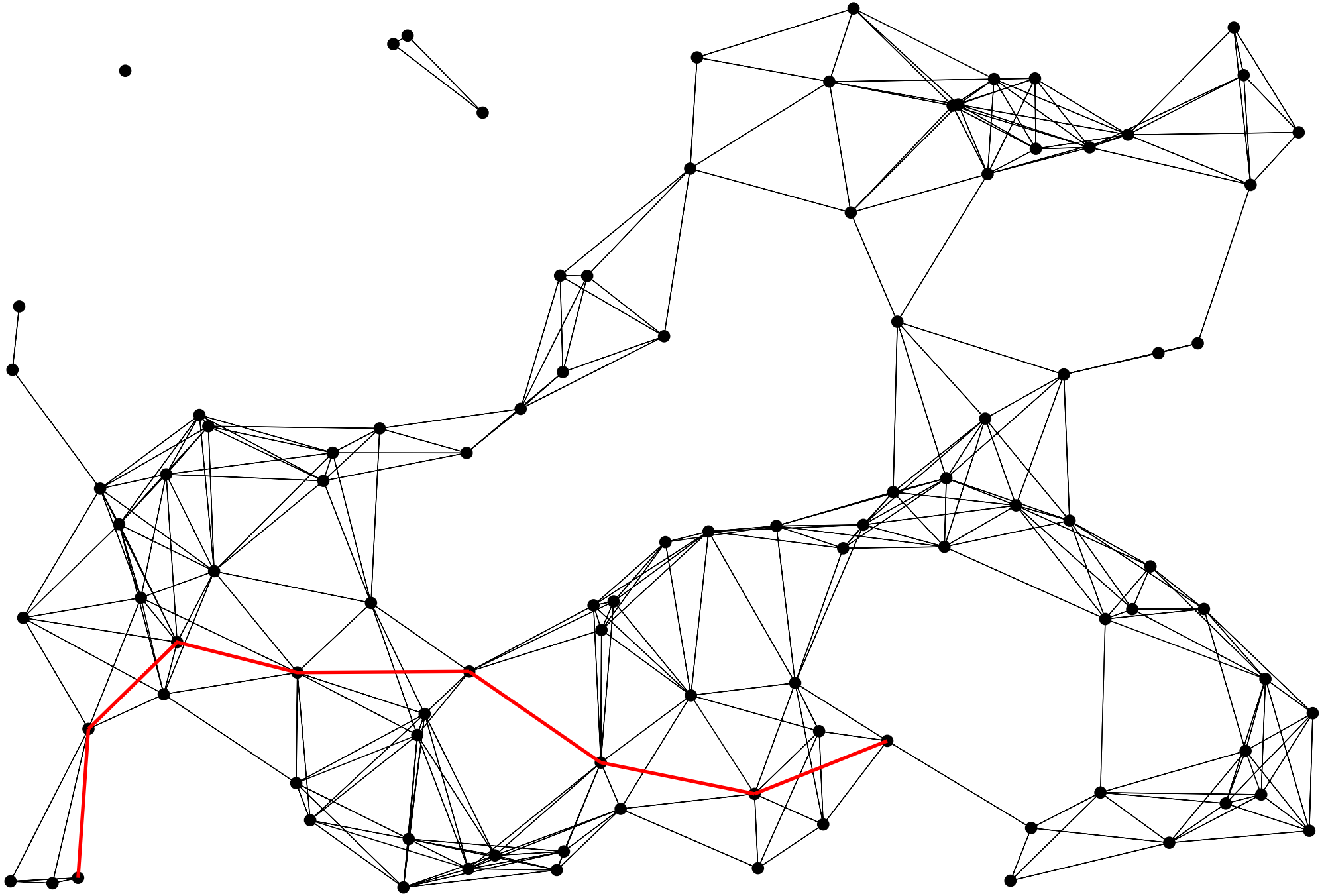


a network of 100 nodes and 682 links

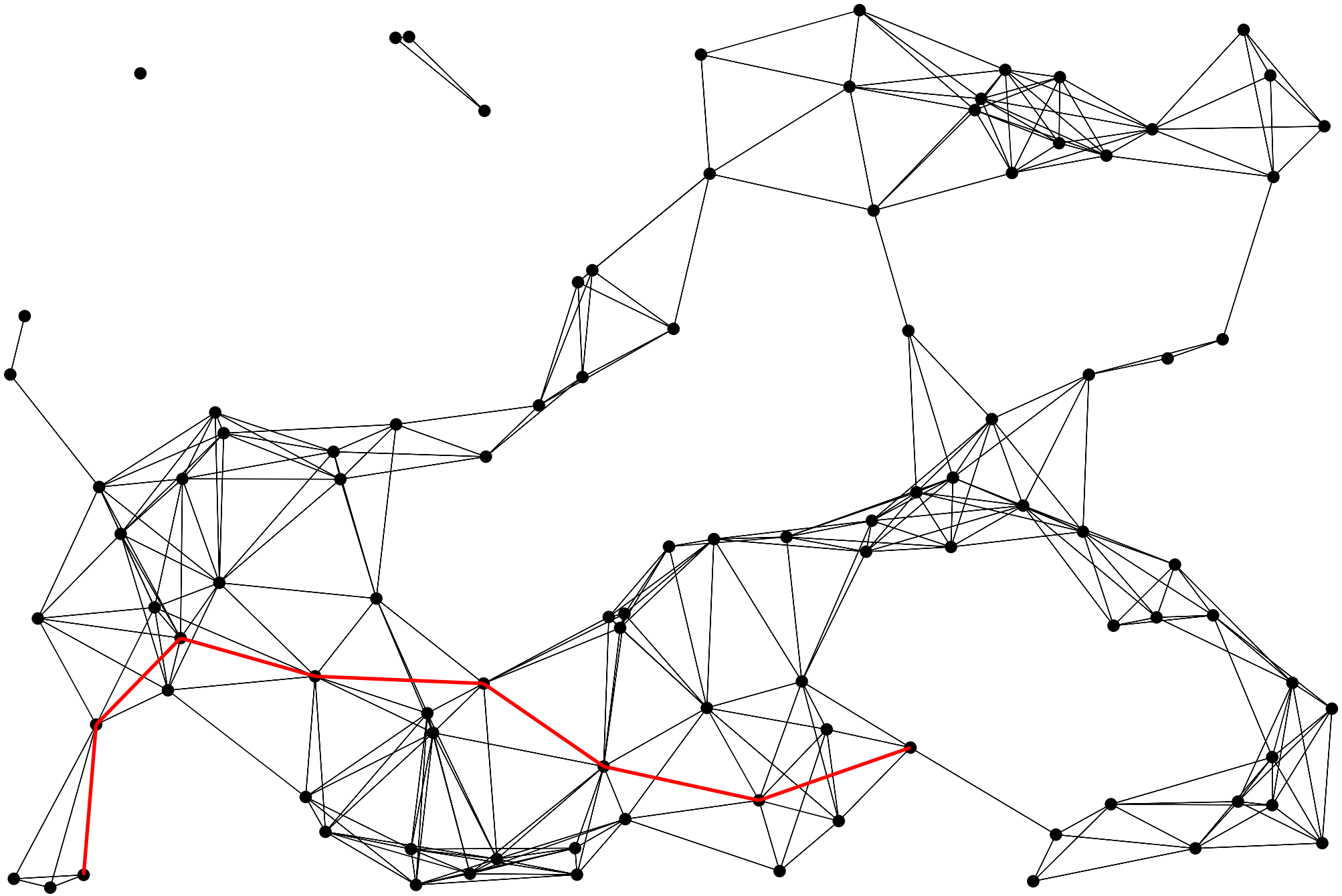


a network of 100 nodes and 672 links

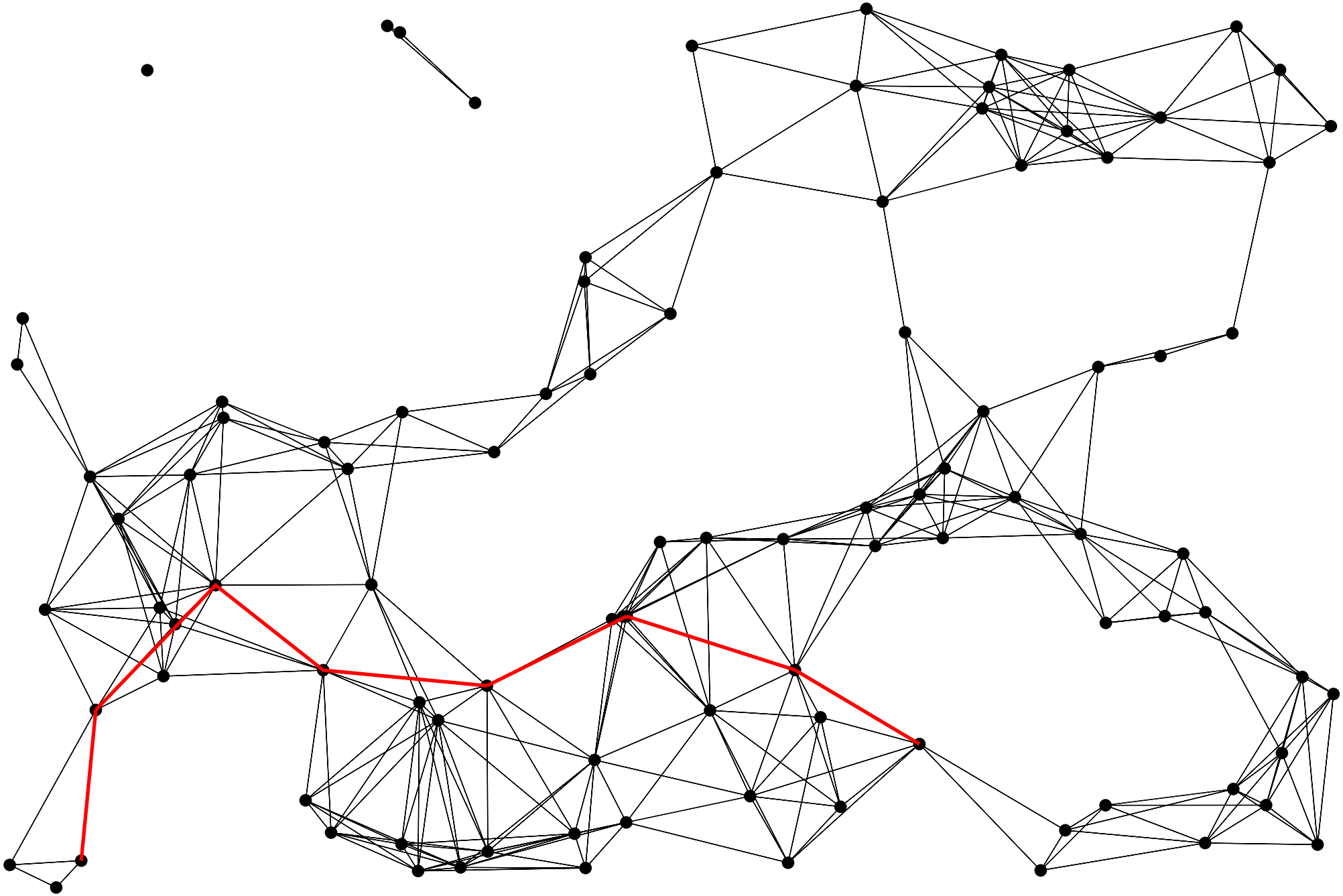
a network of 100 nodes and 682 links



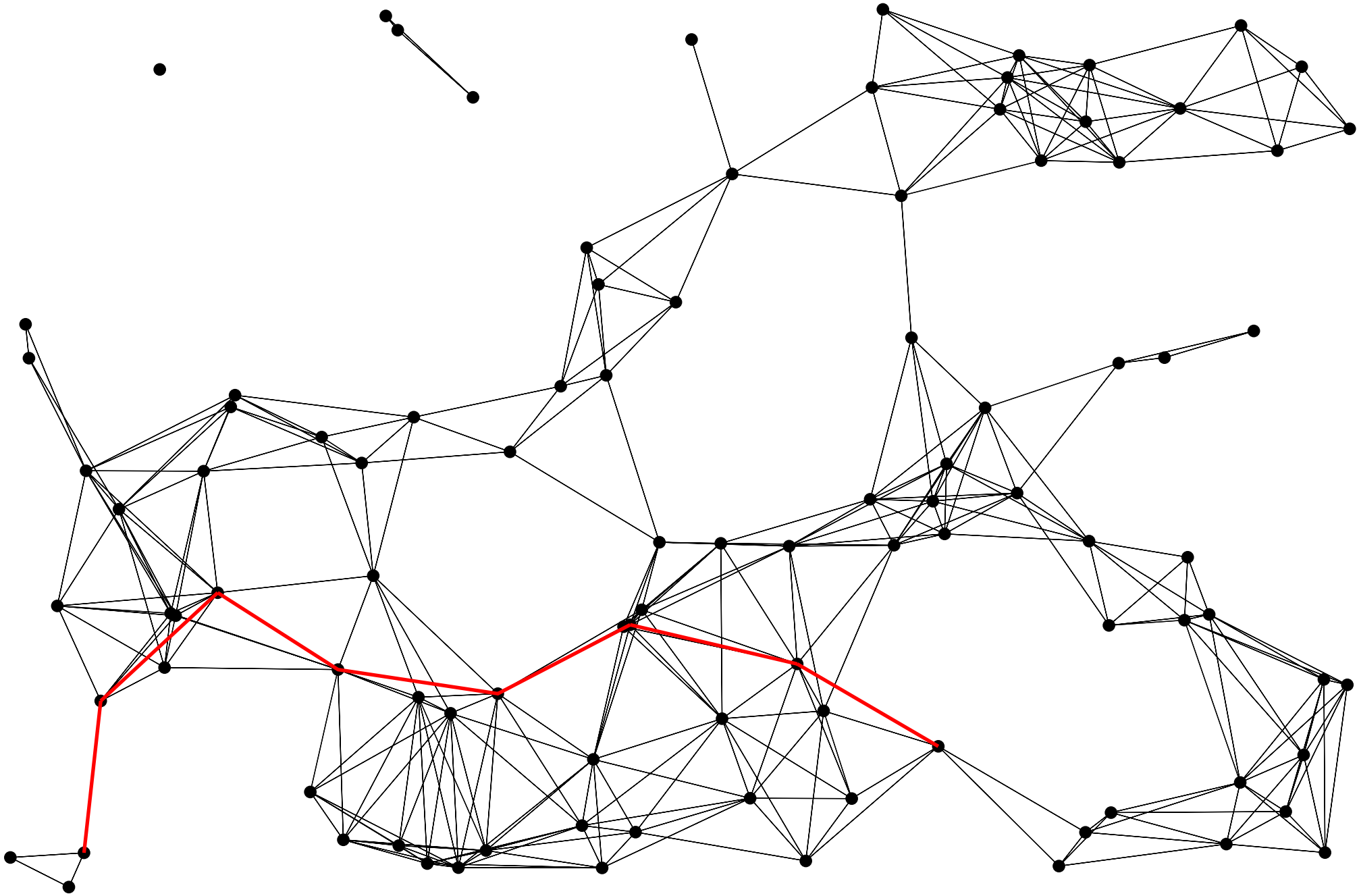
a network of 100 nodes and 680 links

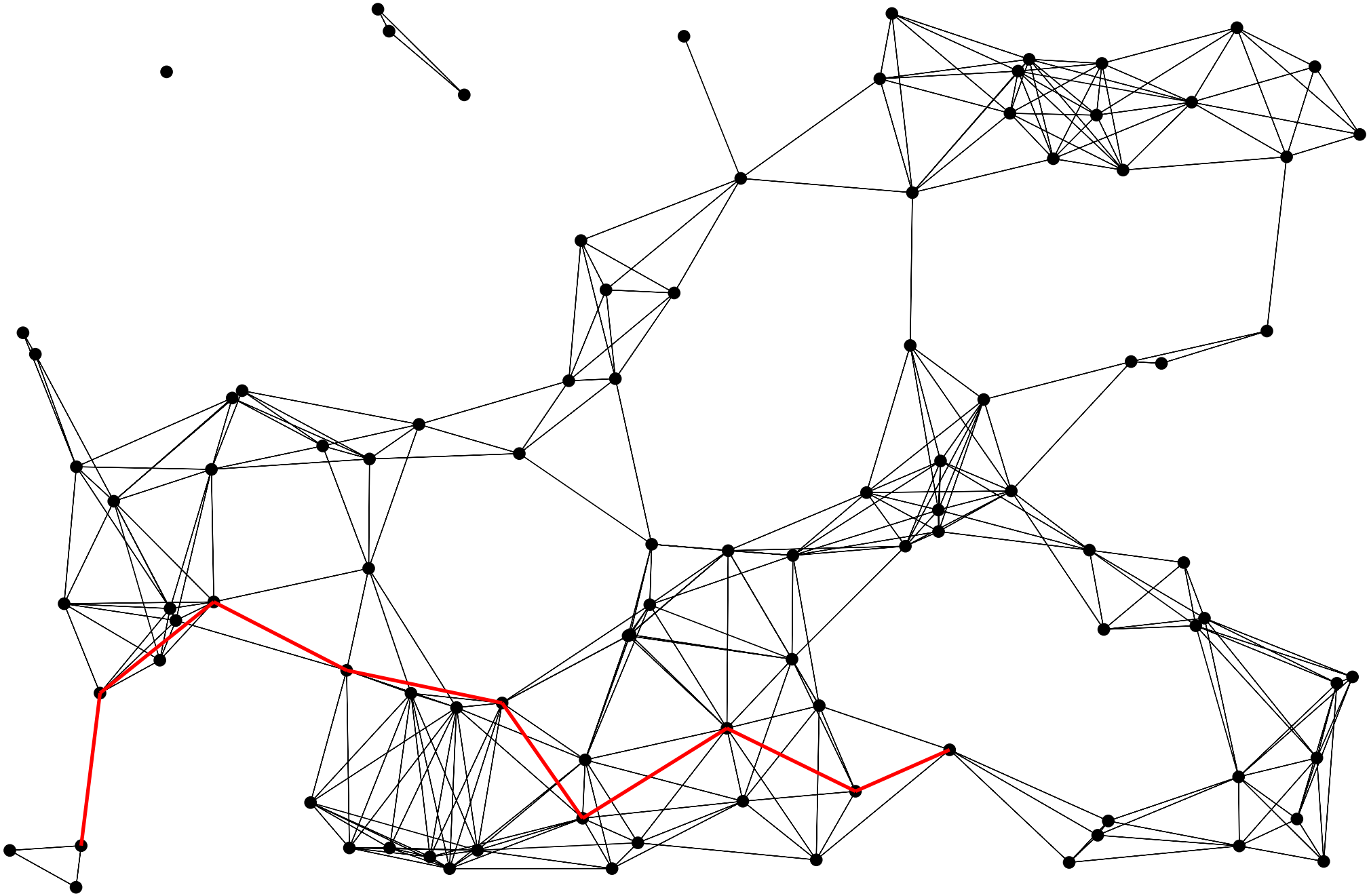


a network of 100 nodes and 680 links

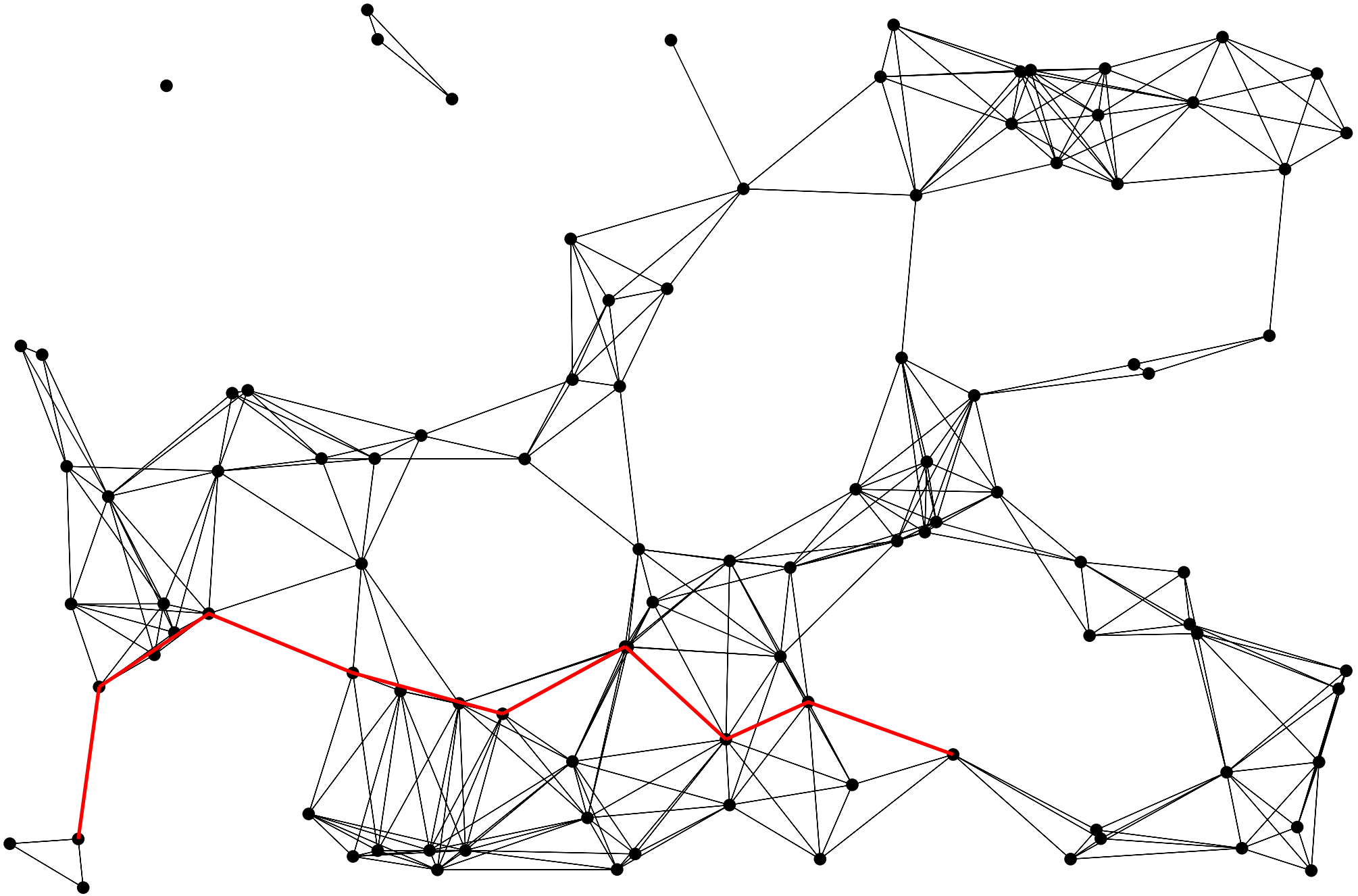


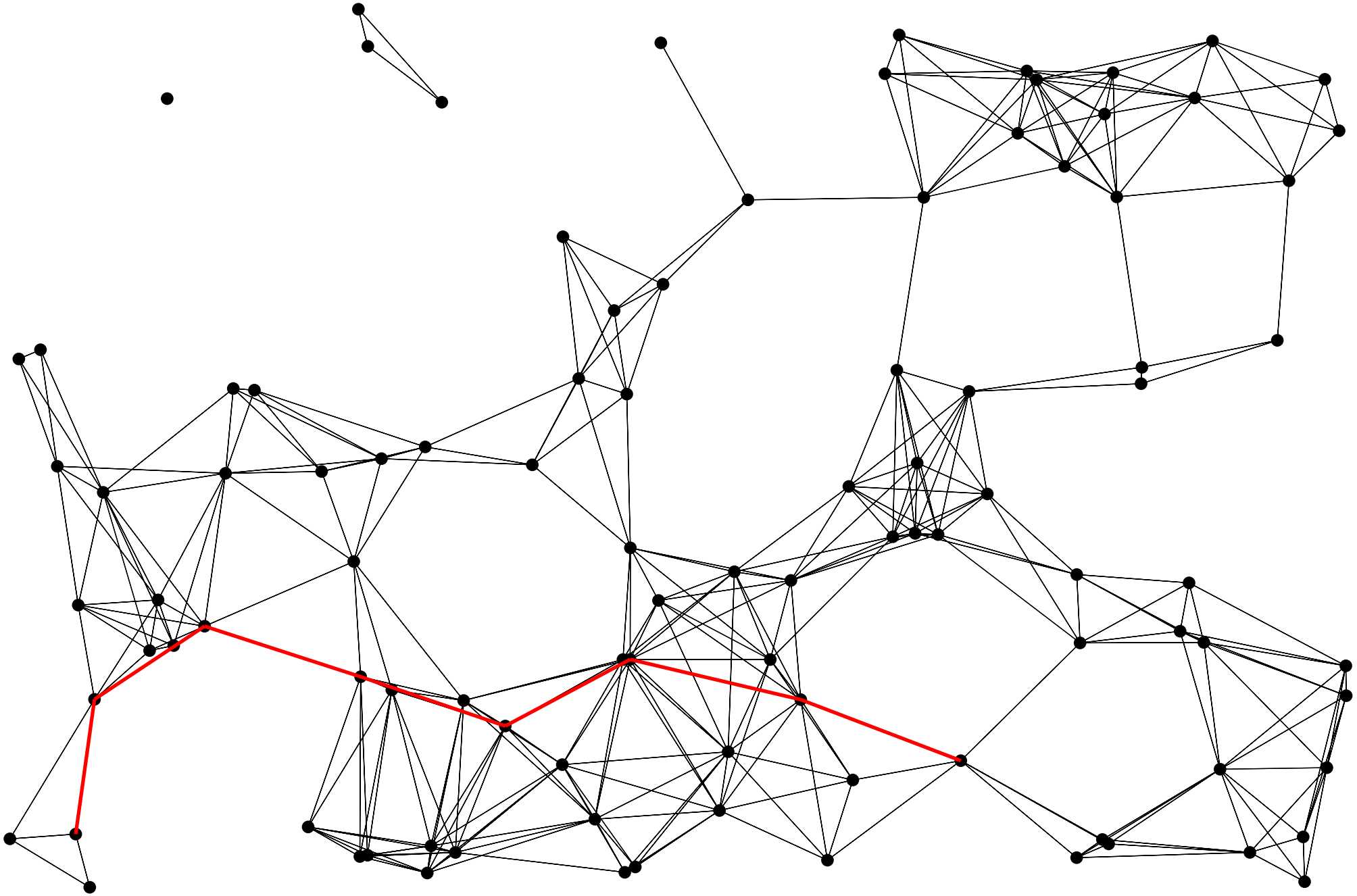
a network of 100 nodes and 694 links


a network of 100 nodes and 696 links

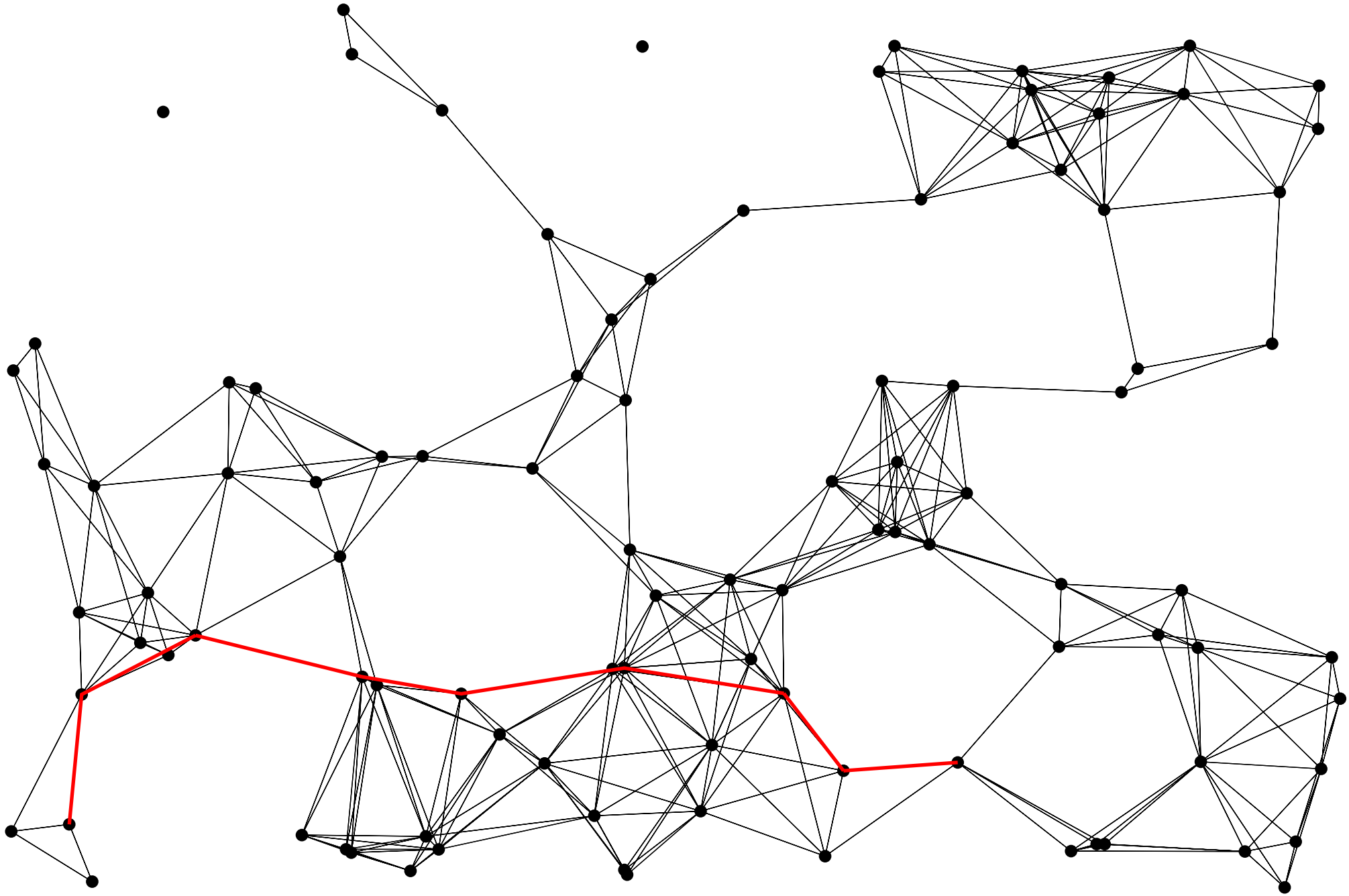


a network of 100 nodes and 688 links

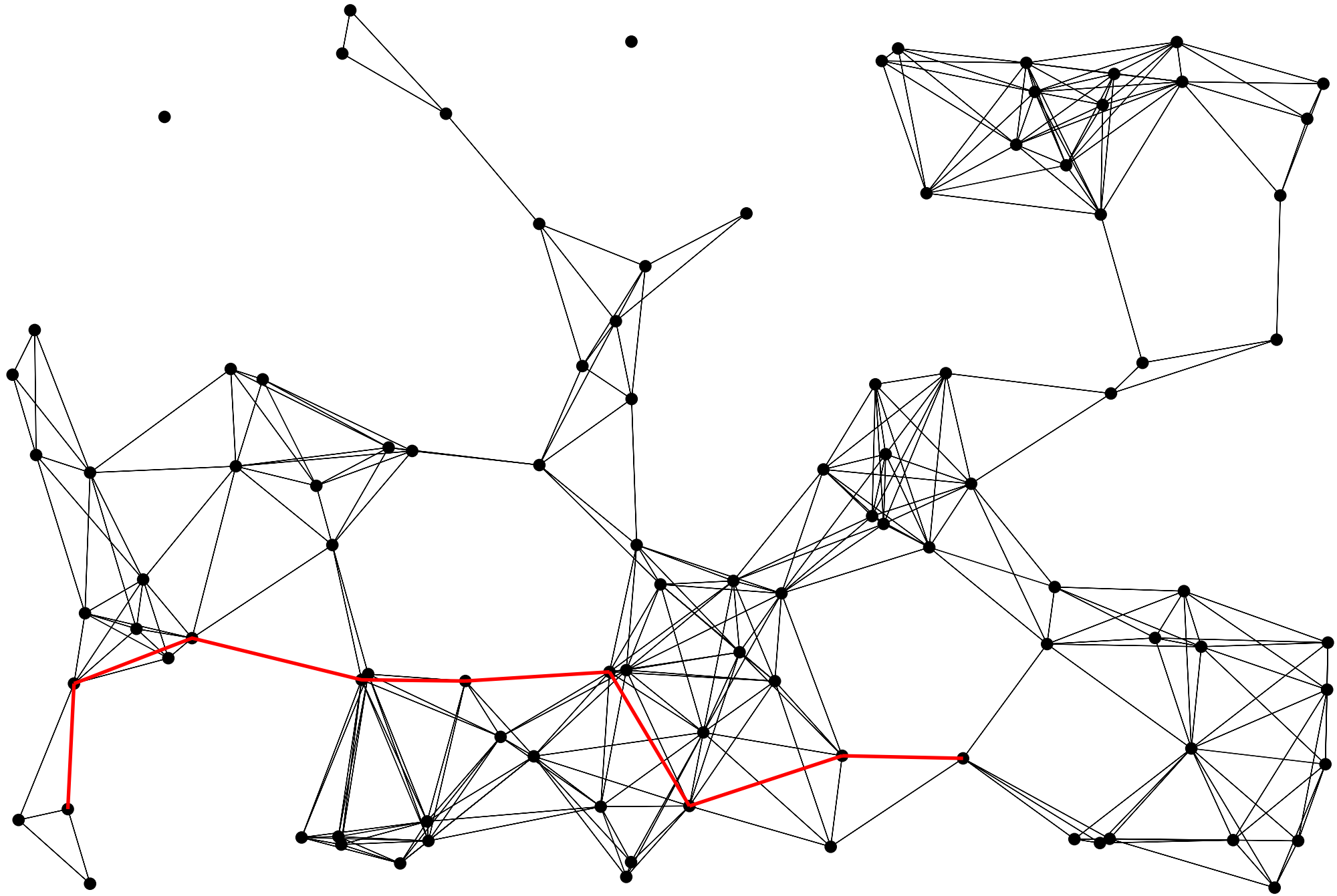

a network of 100 nodes and 692 links



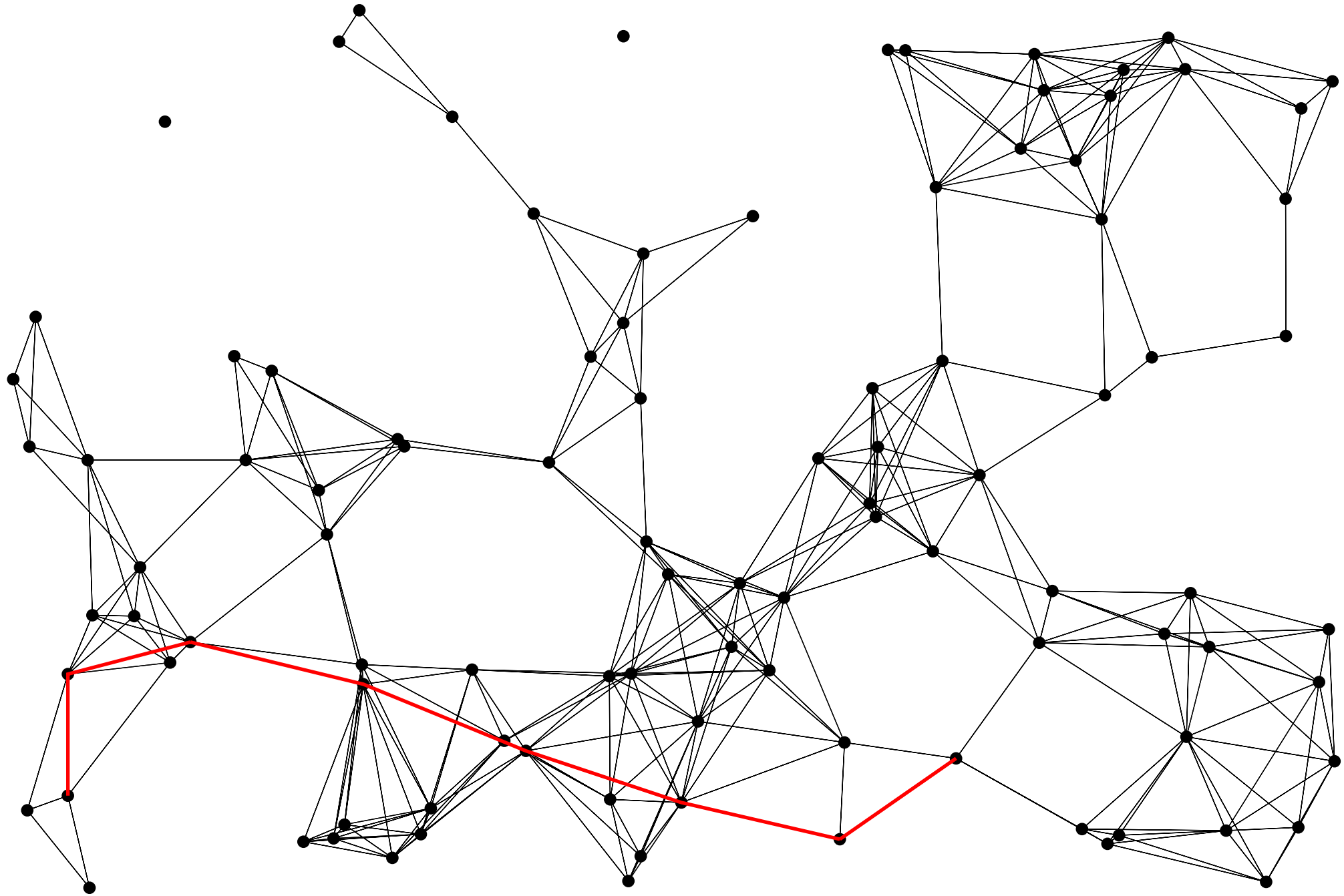
a network of 100 nodes and 706 links



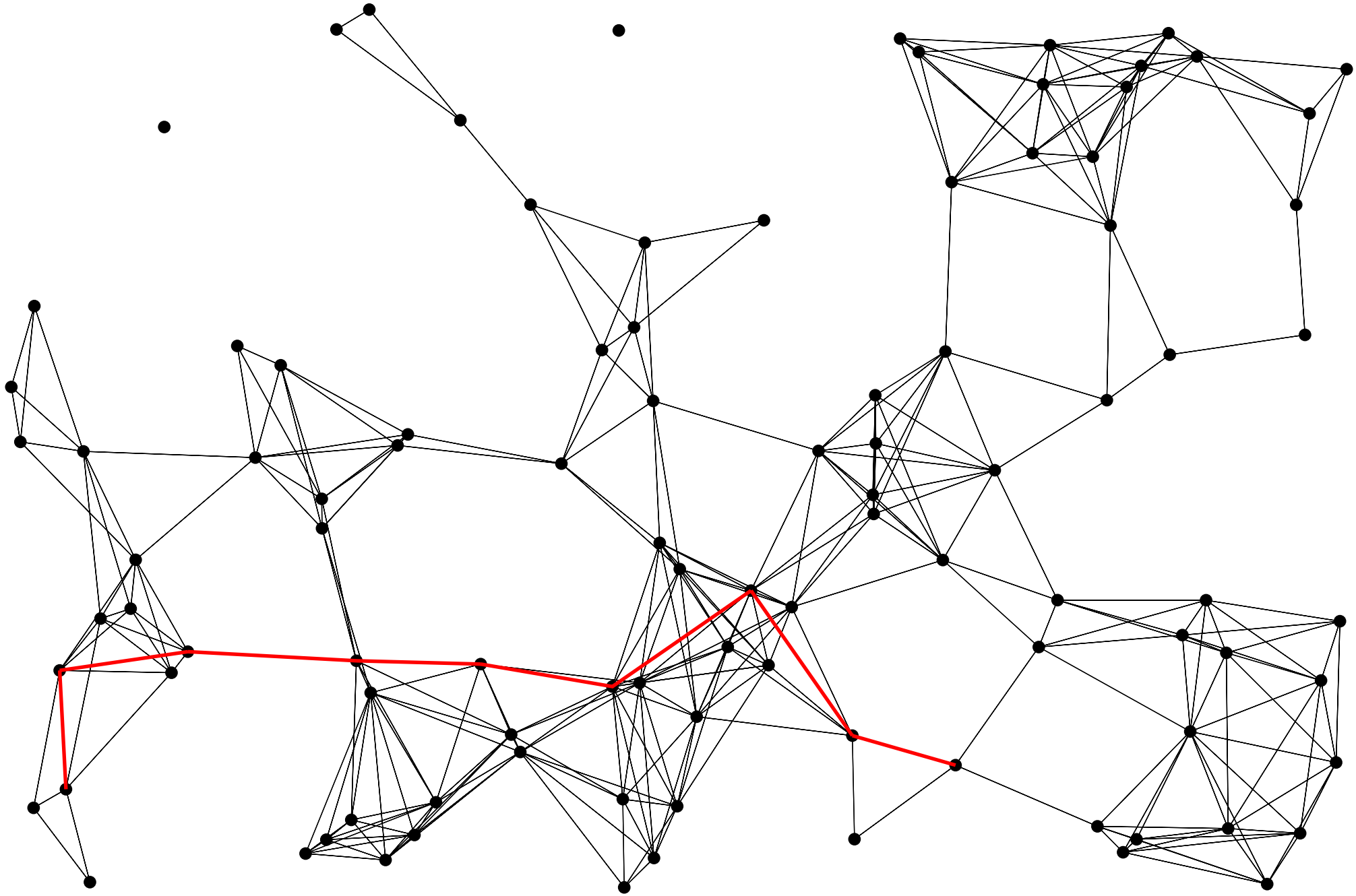
a network of 100 nodes and 684 links



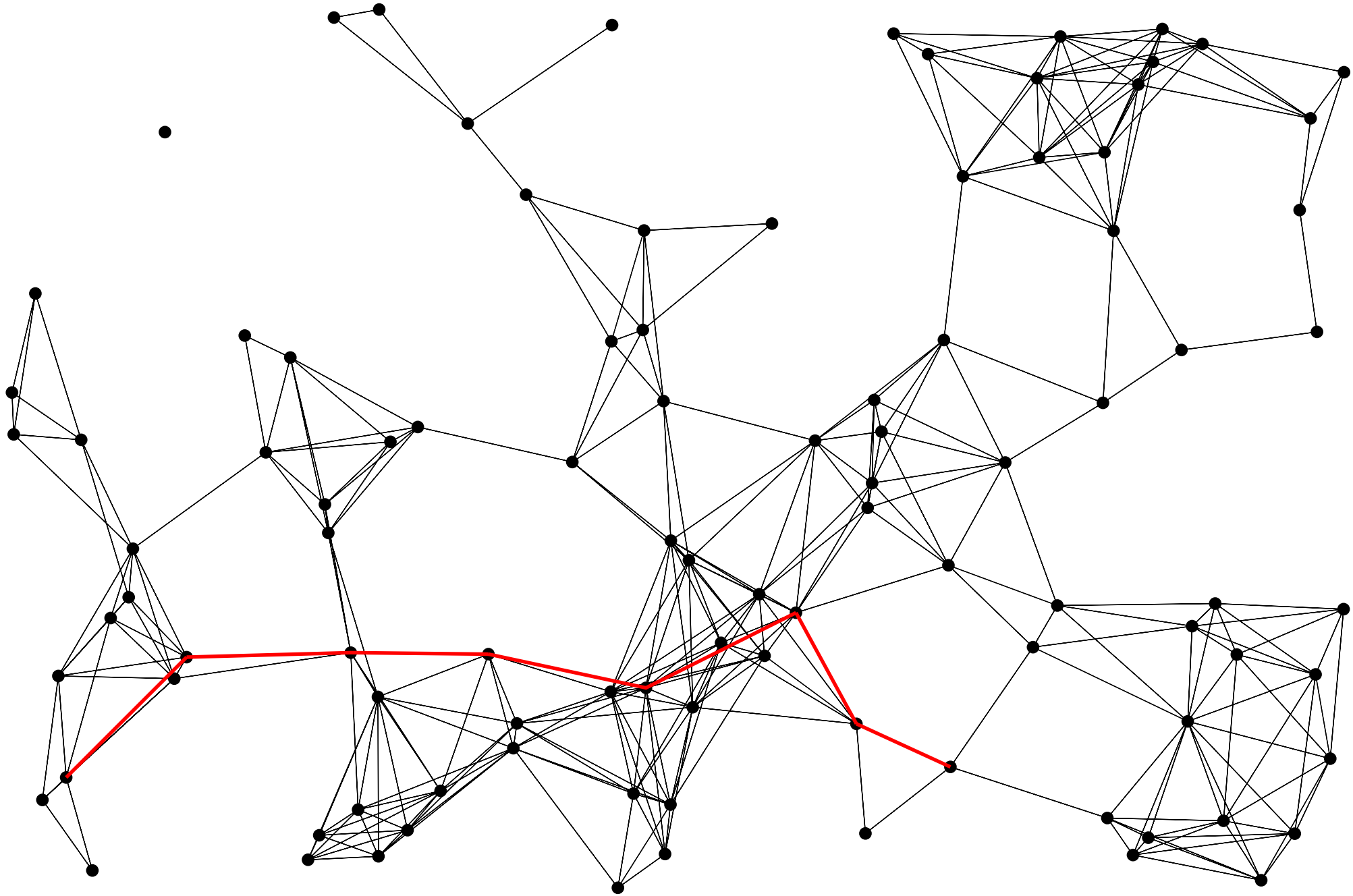
a network of 100 nodes and 698 links



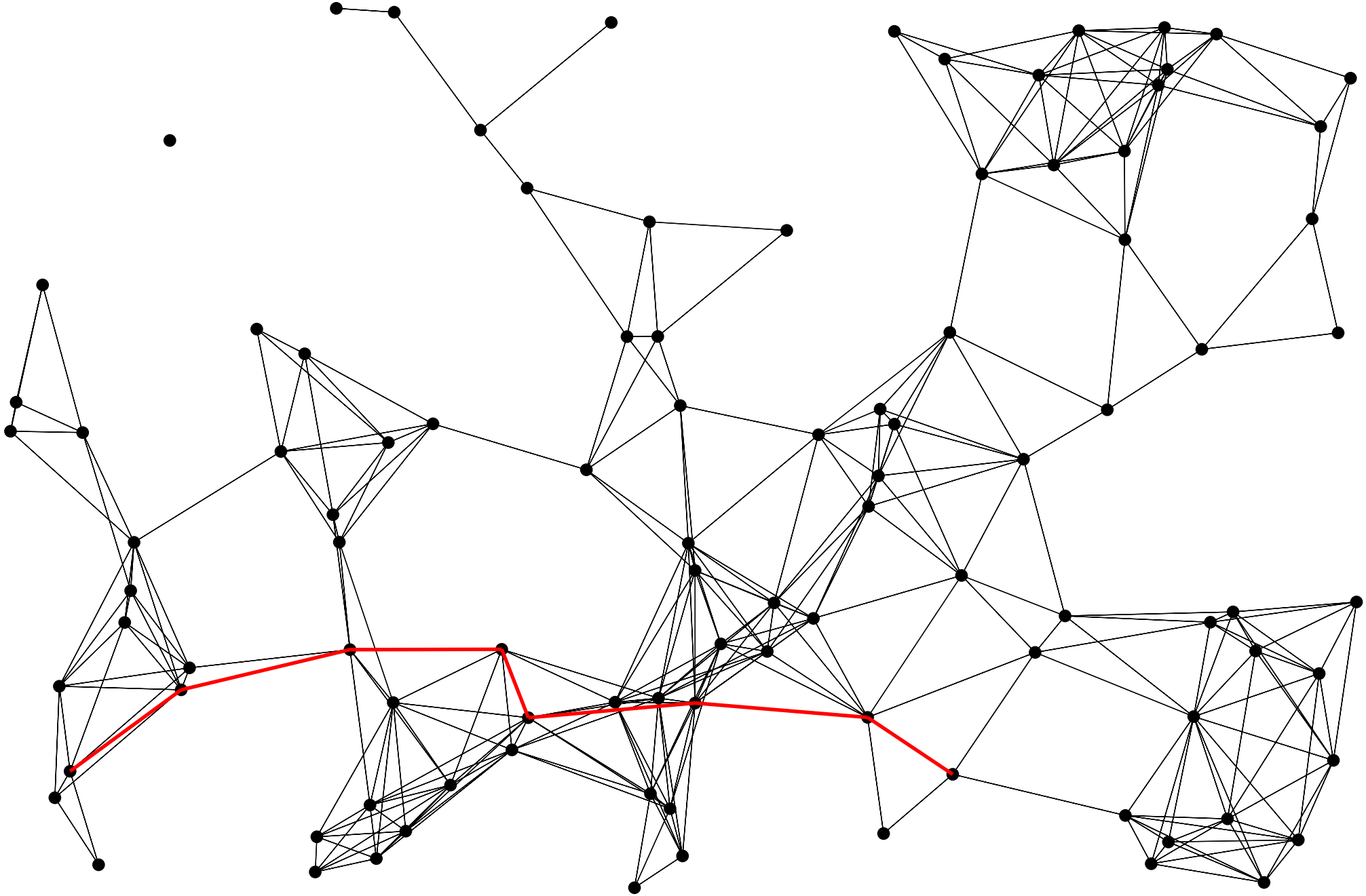
a network of 100 nodes and 686 links



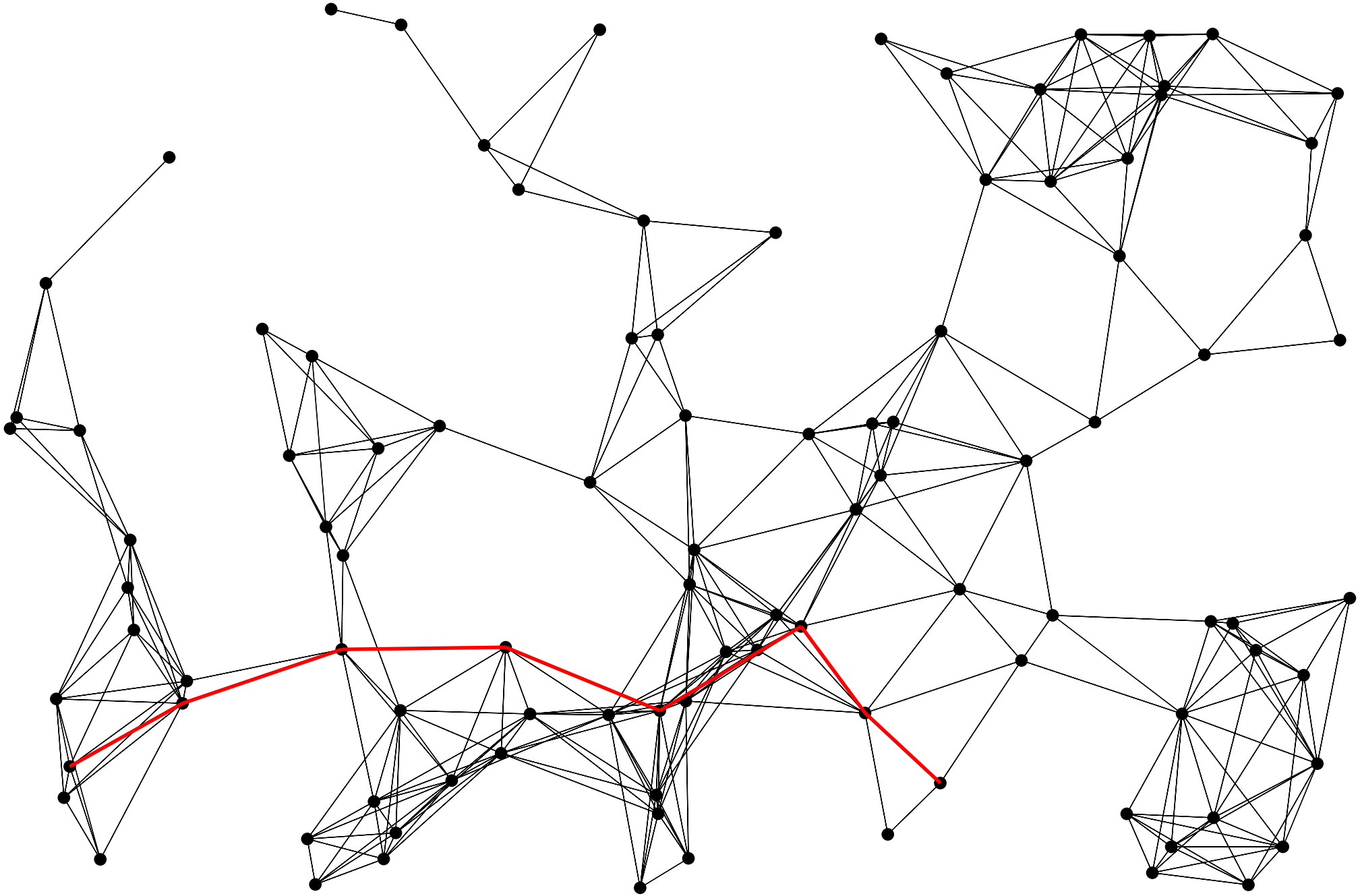
a network of 100 nodes and 684 links

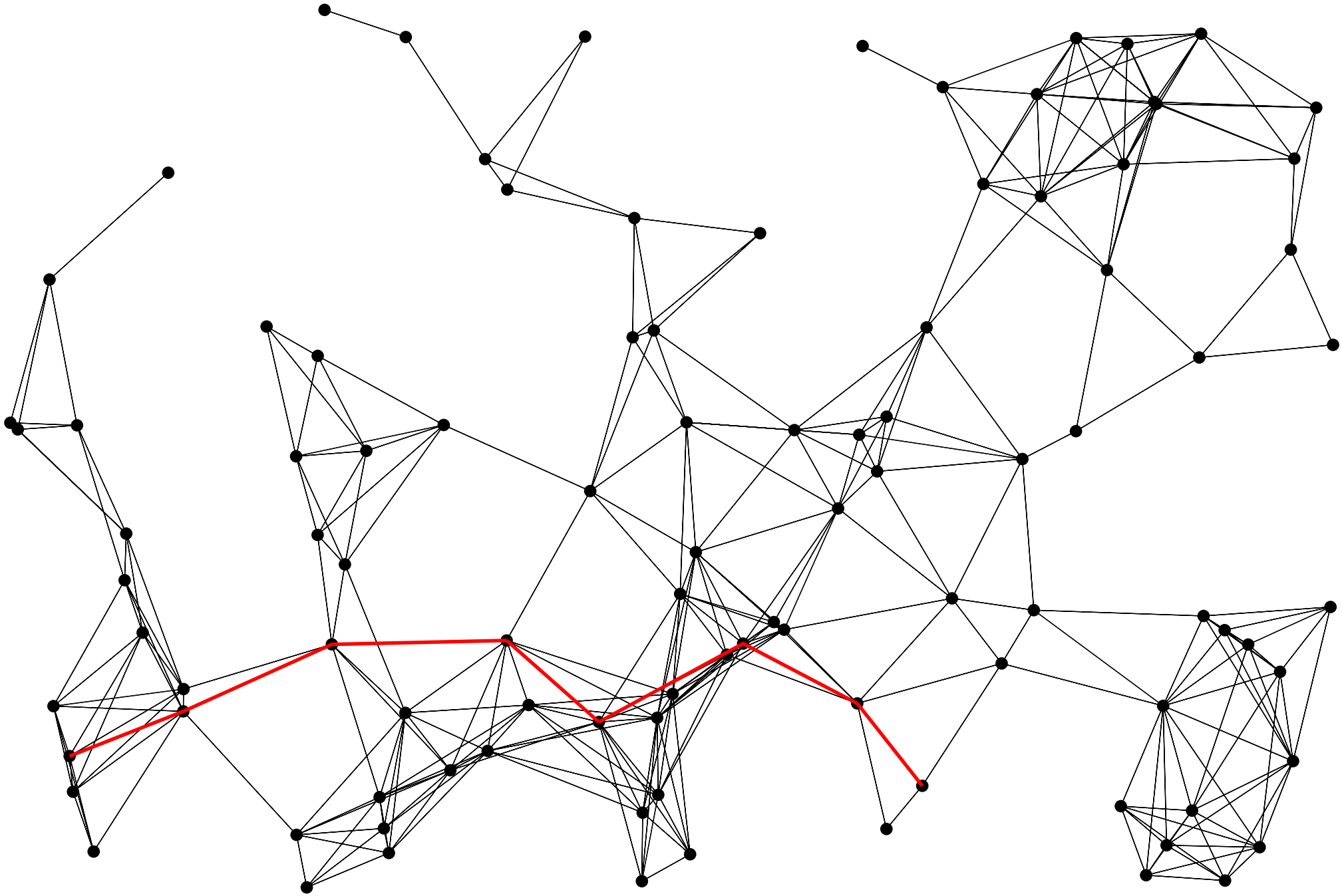


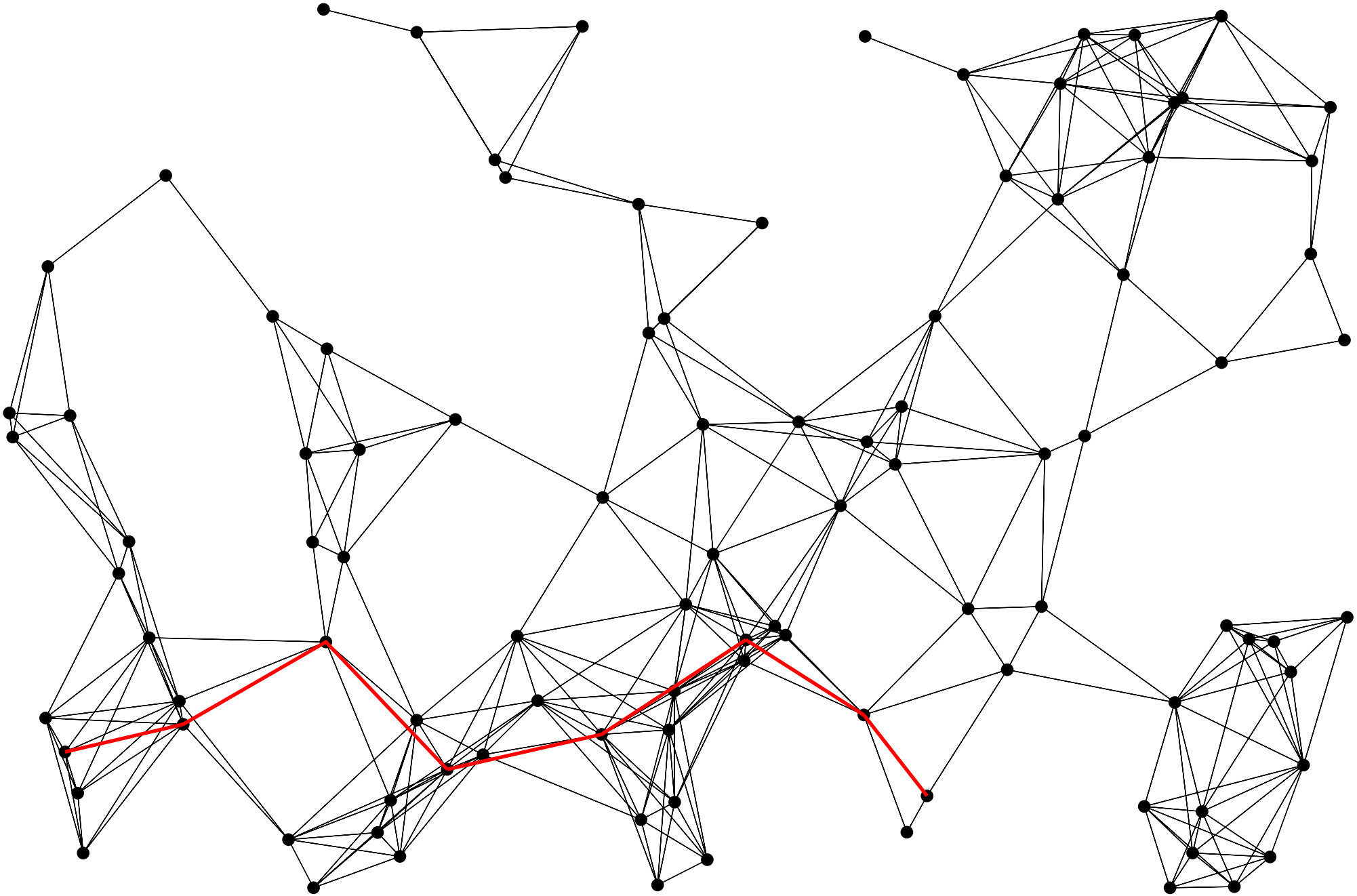
a network of 100 nodes and 680 links



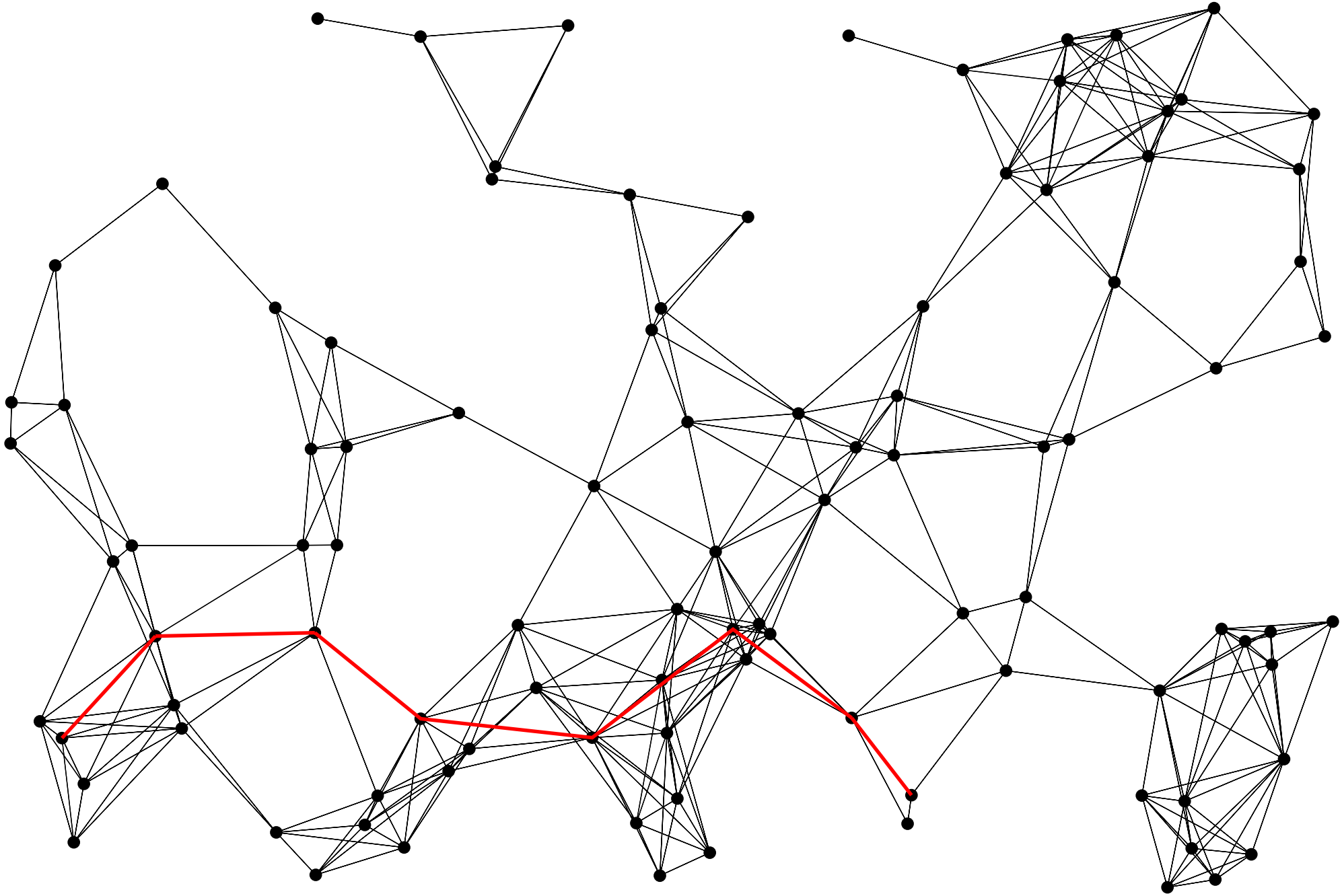
a network of 100 nodes and 668 links


a network of 100 nodes and 684 links

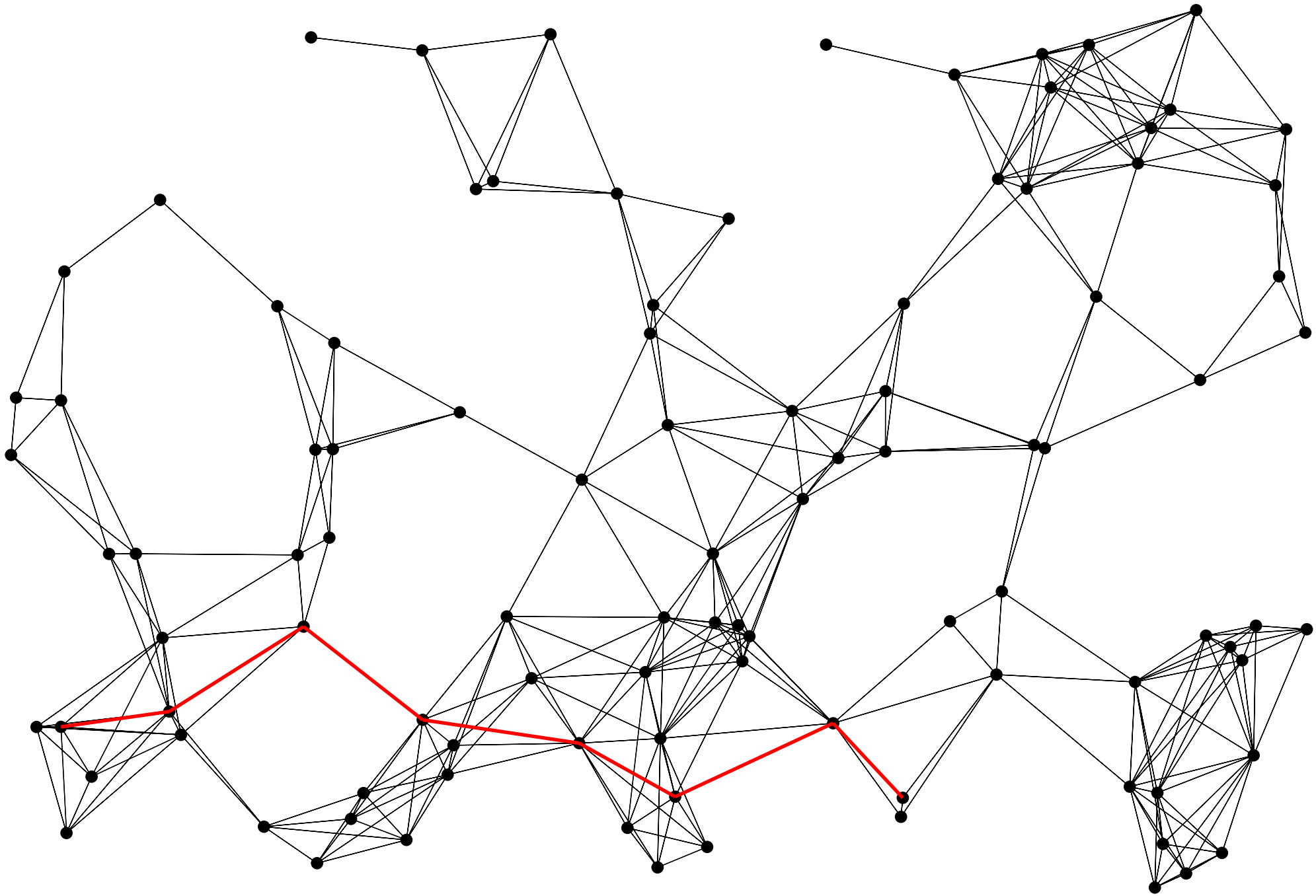

a network of 100 nodes and 678 links



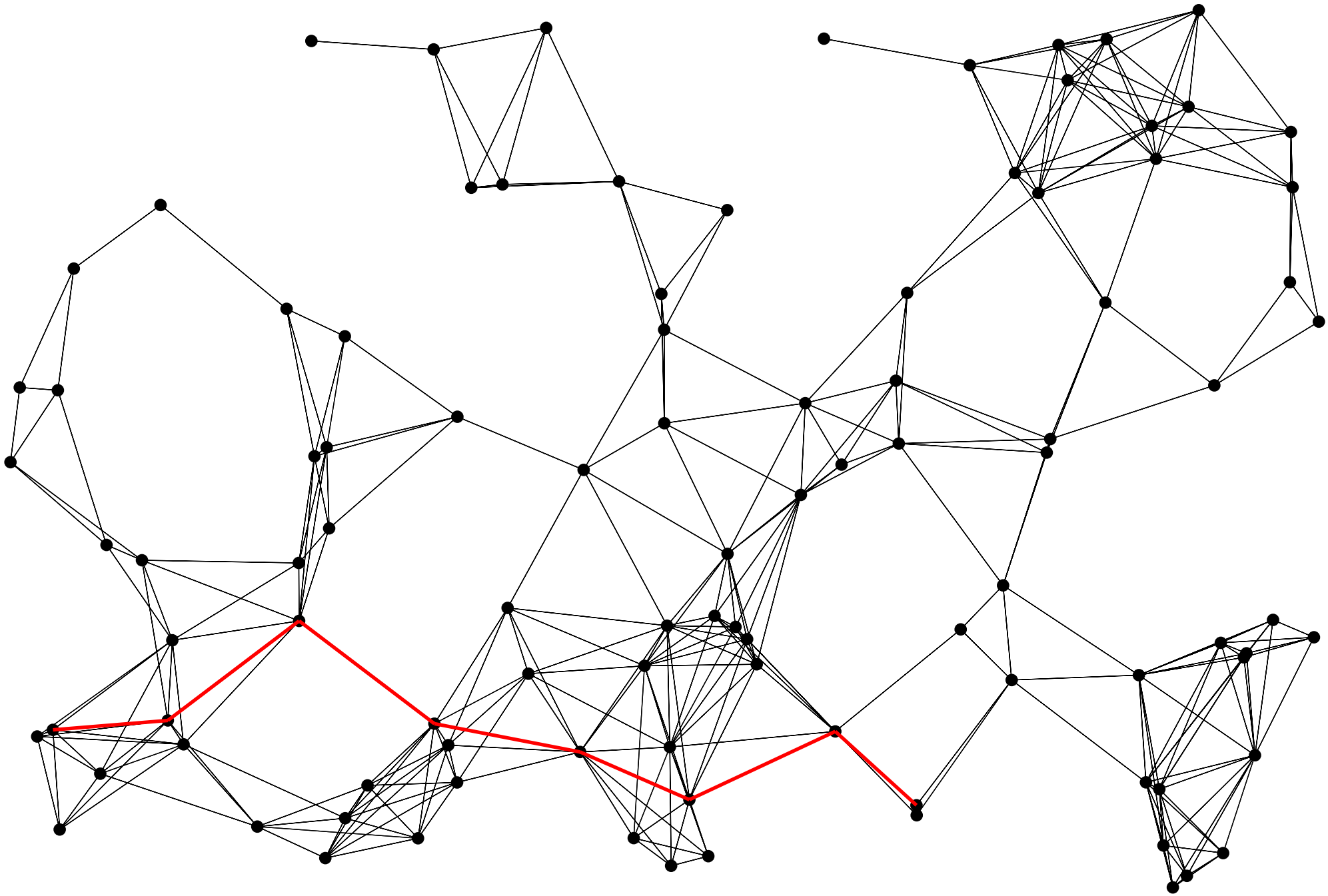
a network of 100 nodes and 680 links



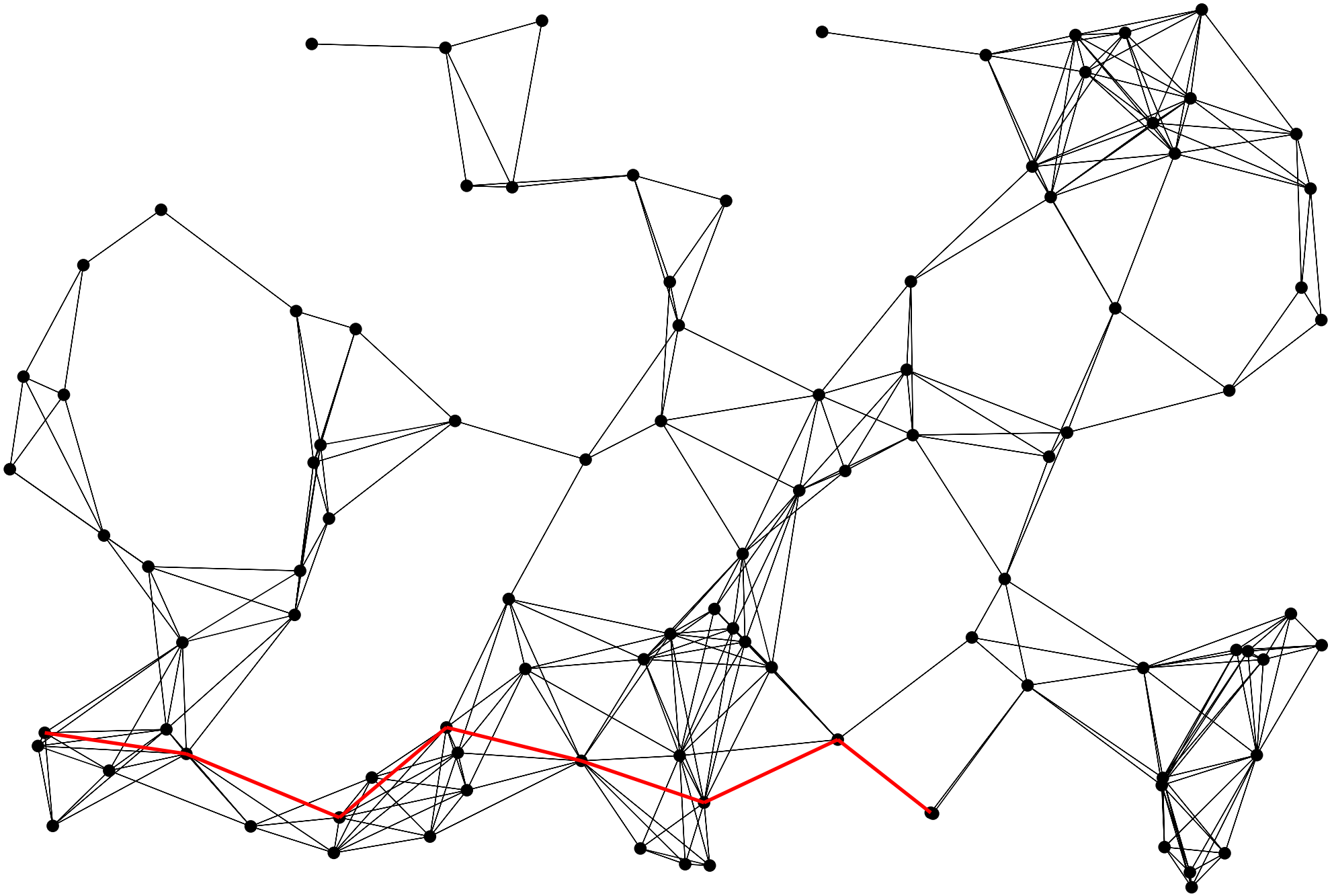
a network of 100 nodes and 678 links



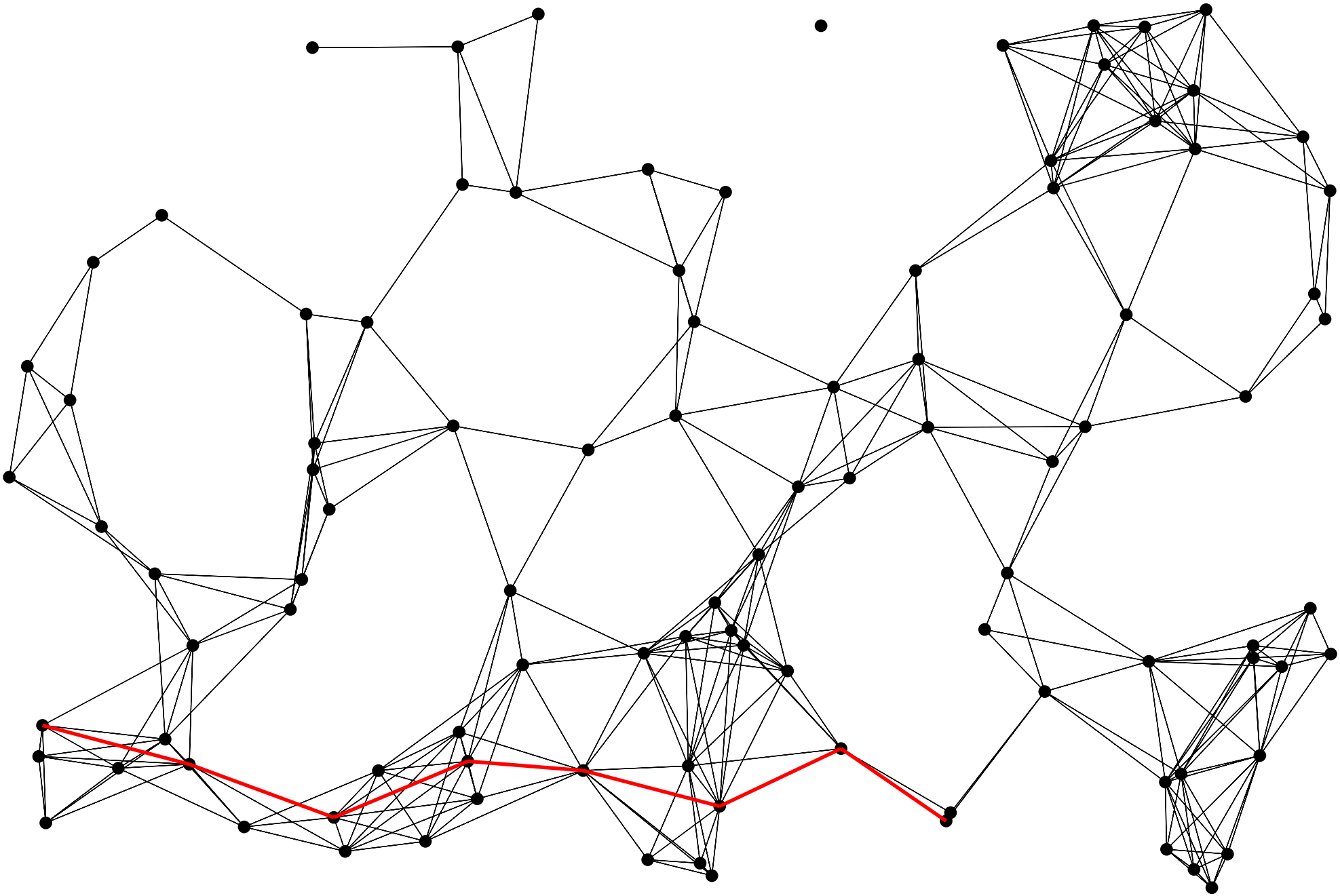
a network of 100 nodes and 672 links



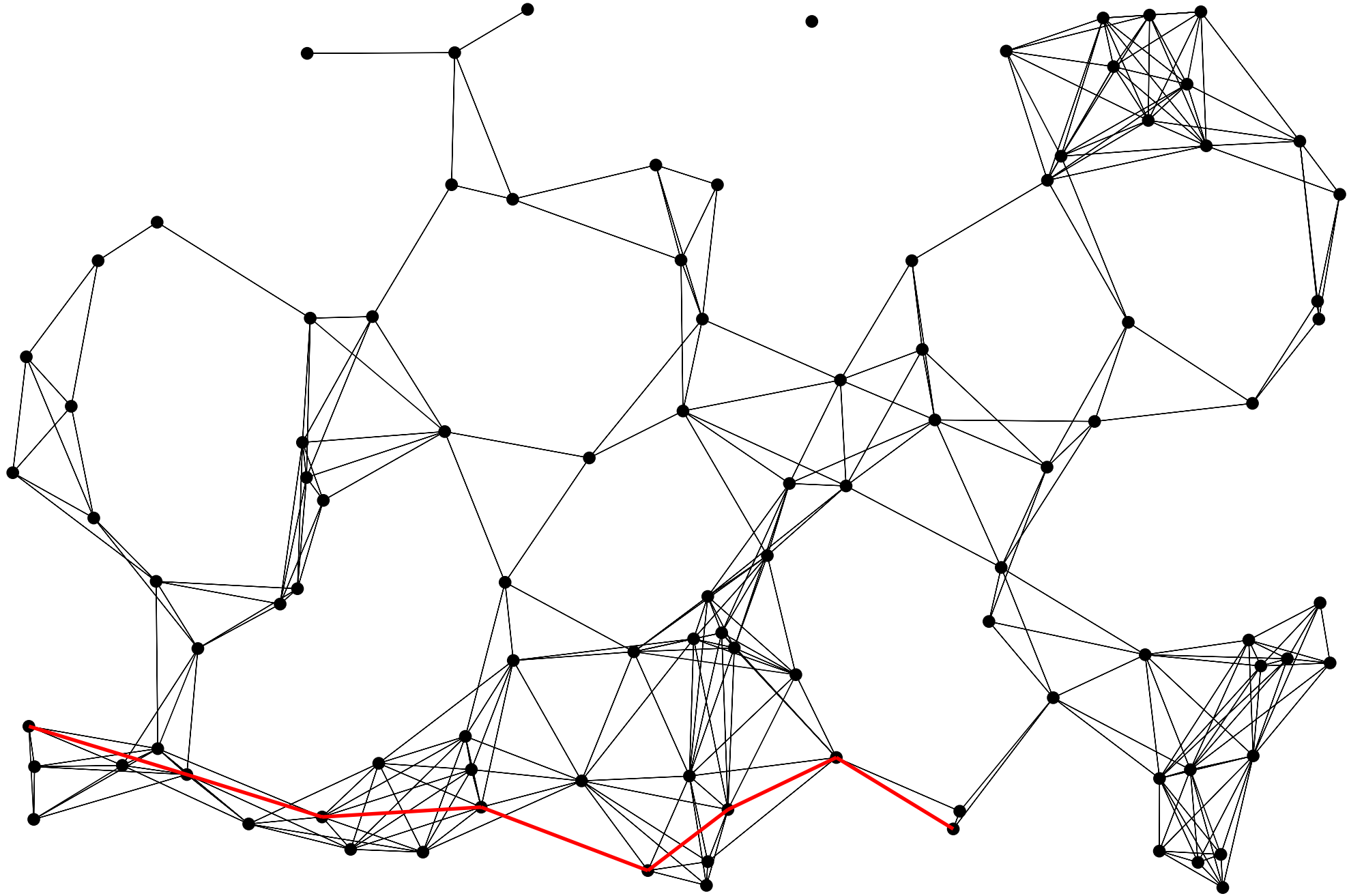
a network of 100 nodes and 680 links



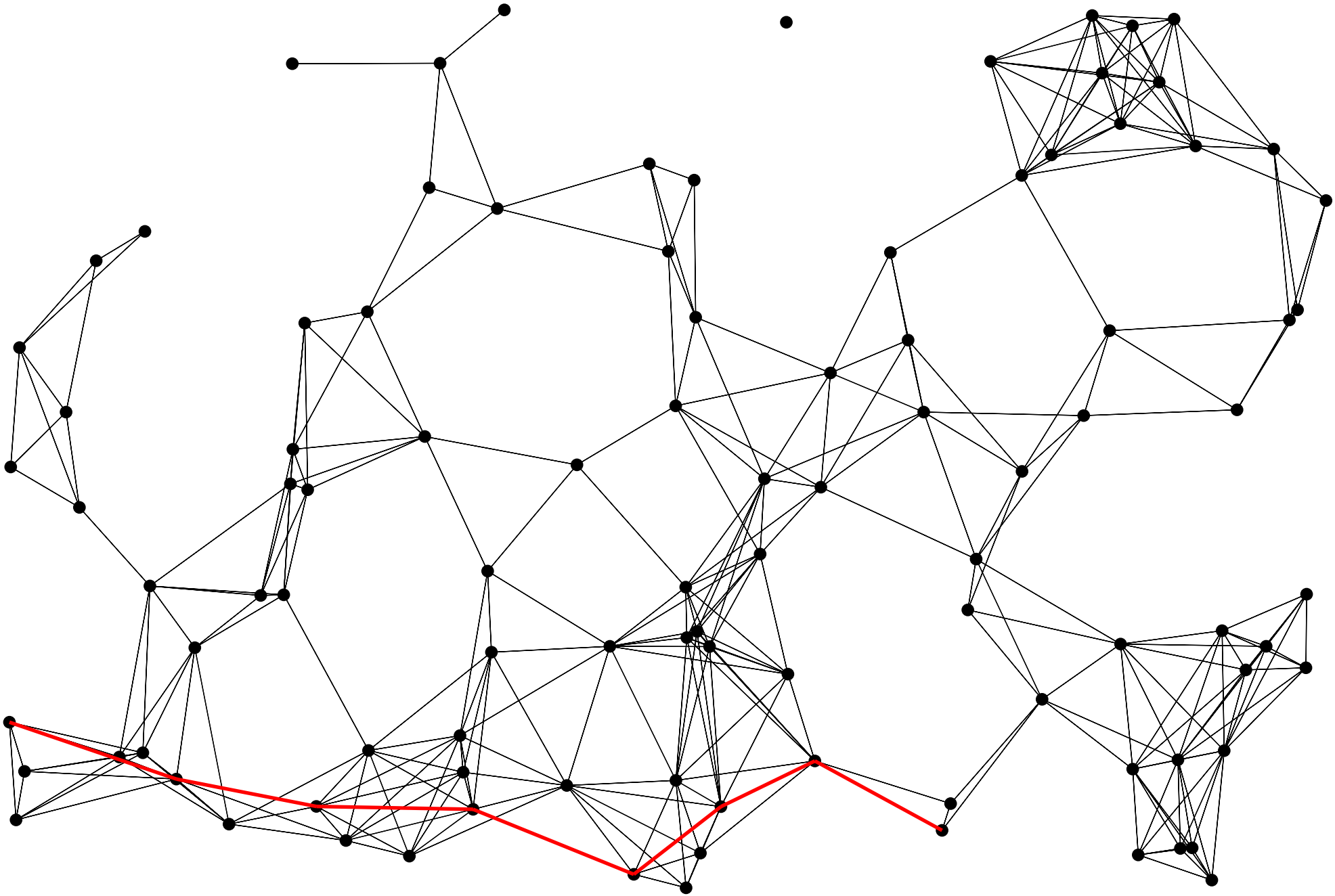
a network of 100 nodes and 674 links



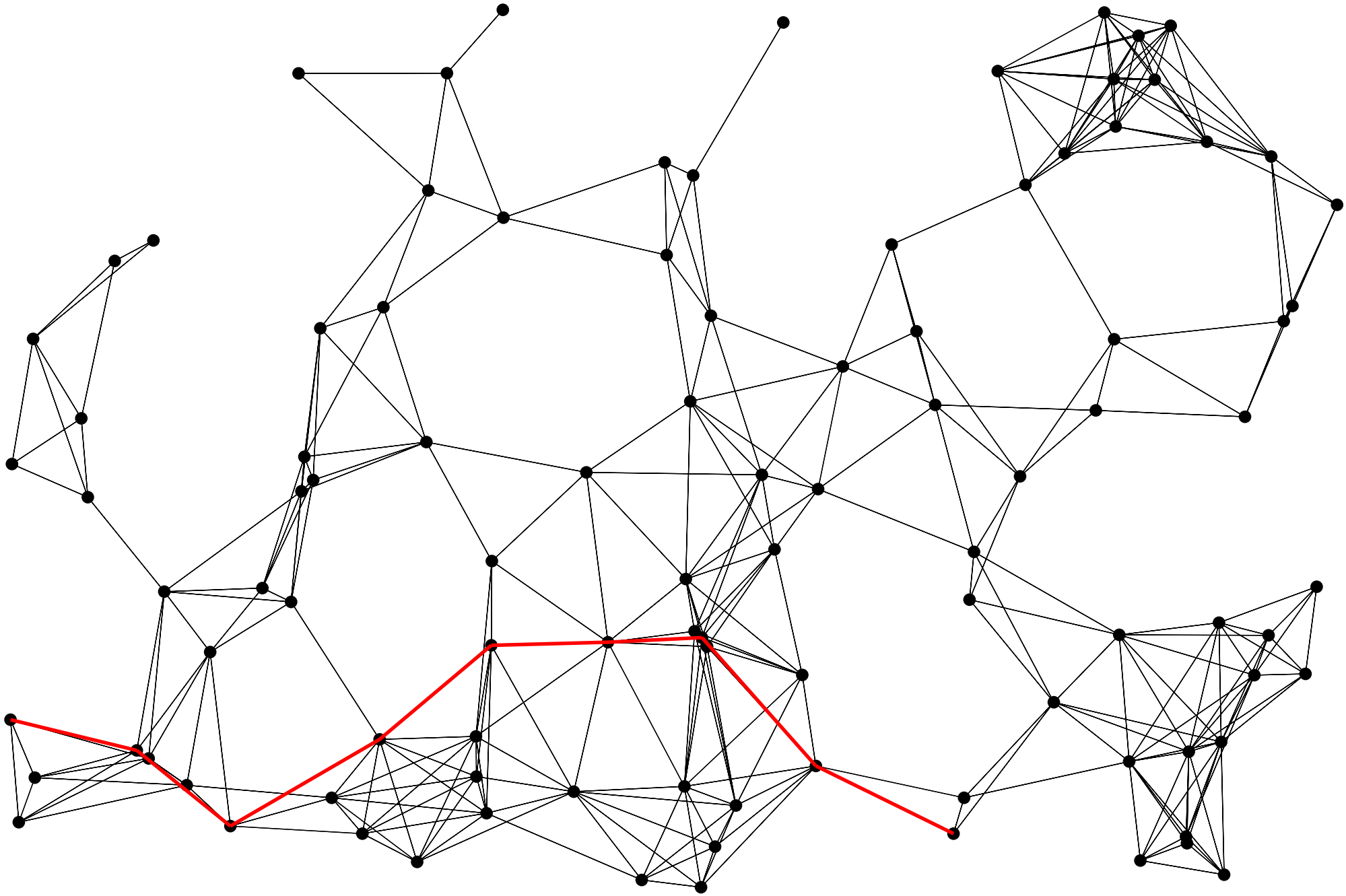
a network of 100 nodes and 662 links



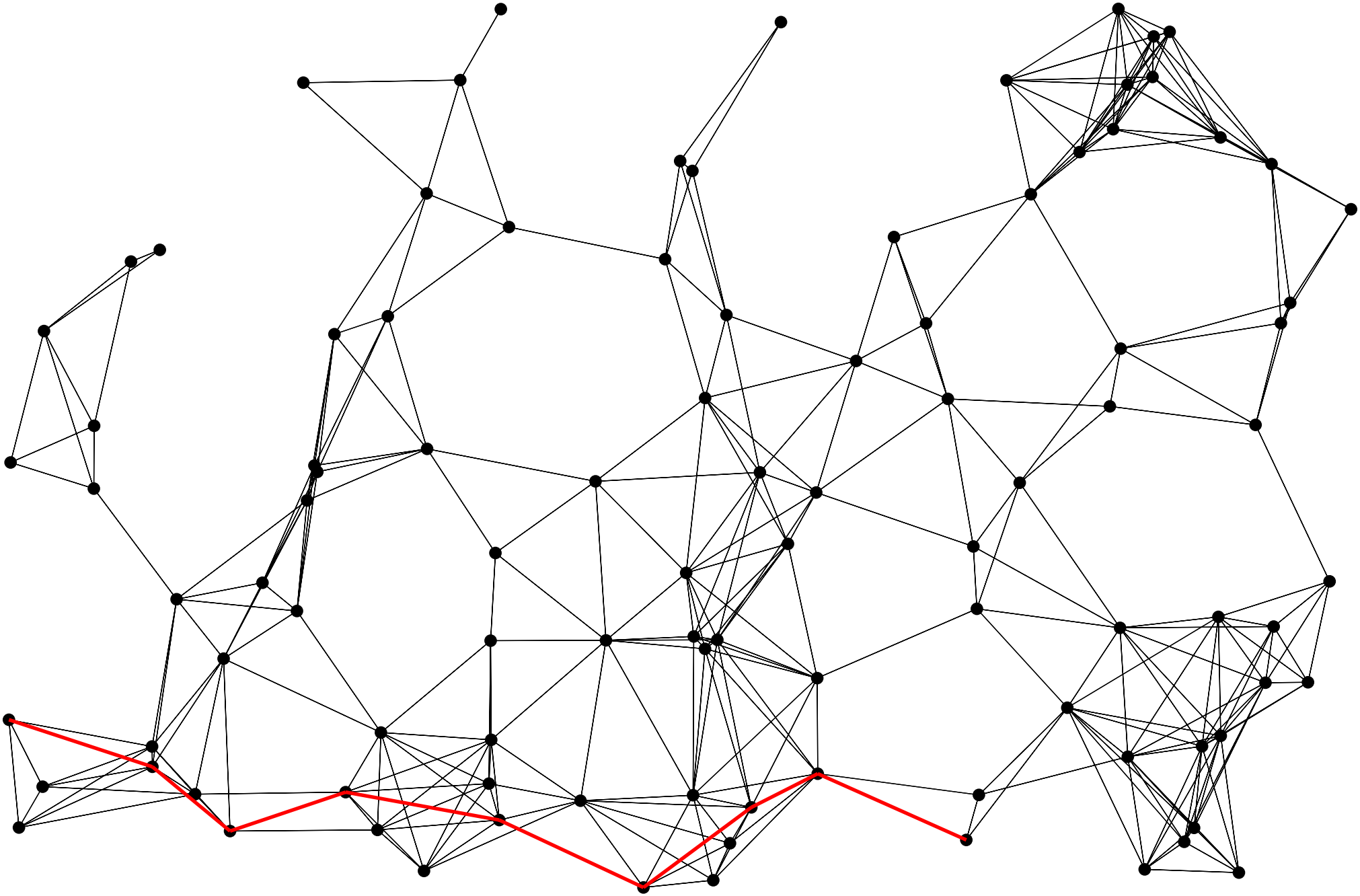
a network of 100 nodes and 662 links



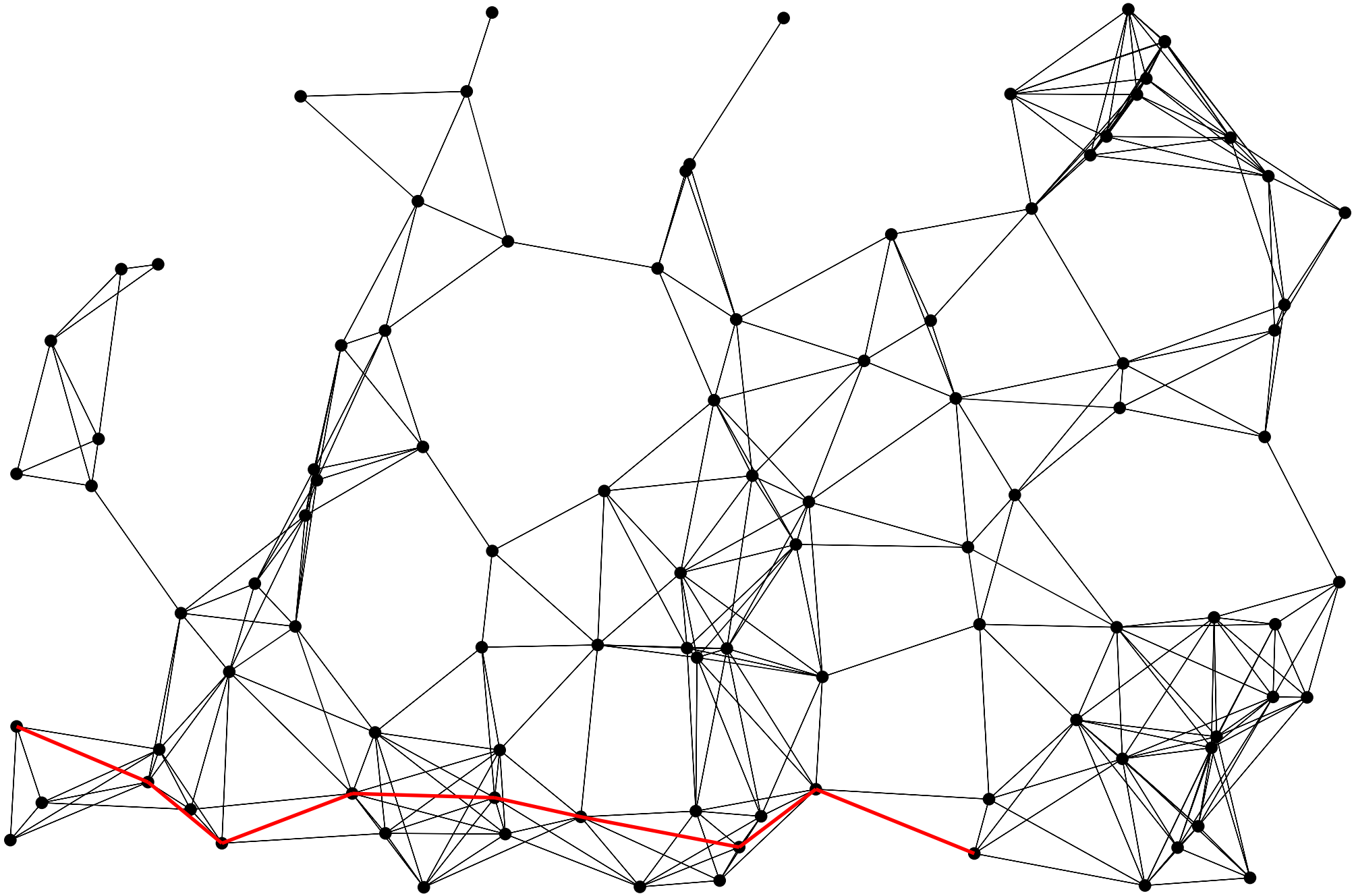
a network of 100 nodes and 666 links



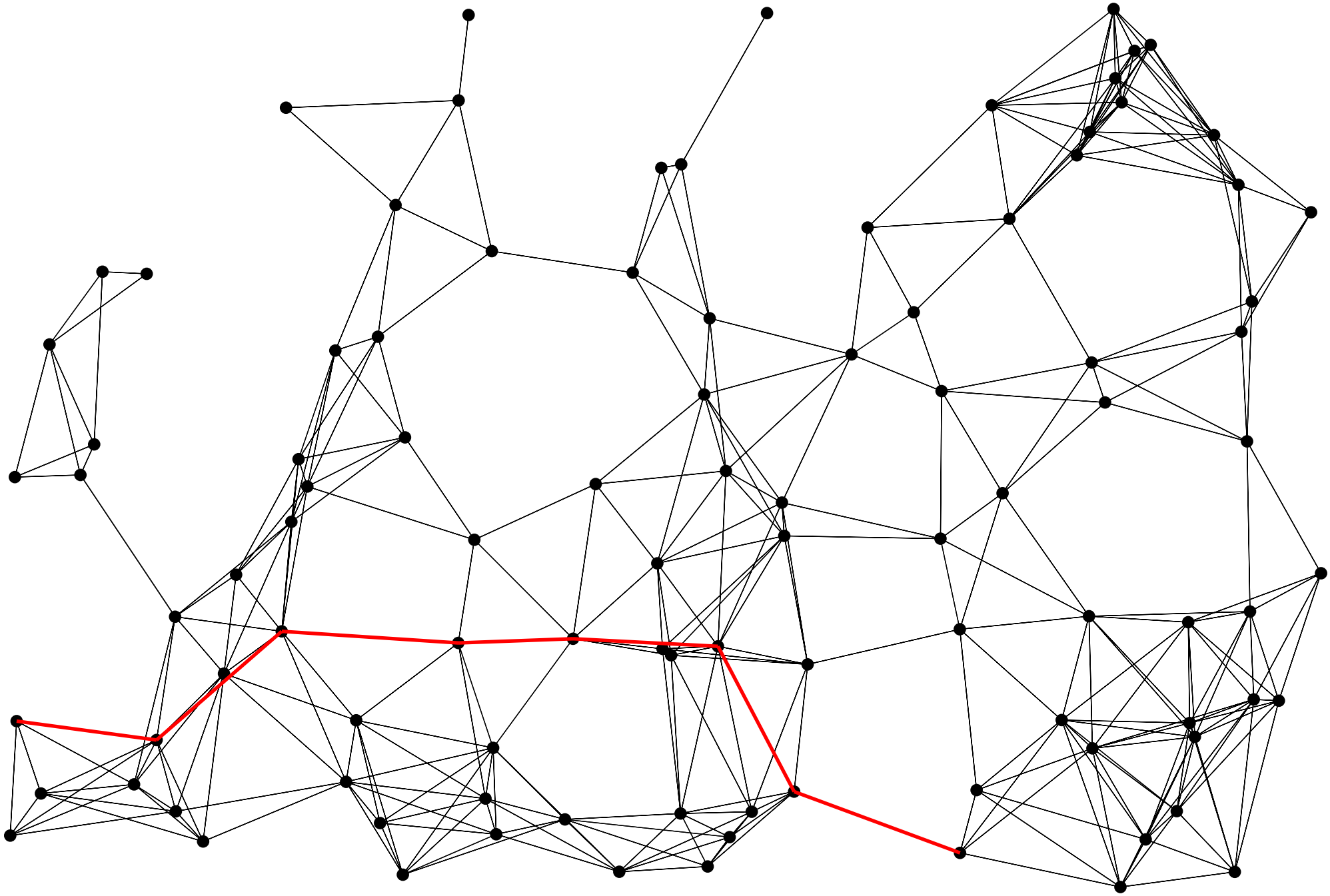
a network of 100 nodes and 670 links



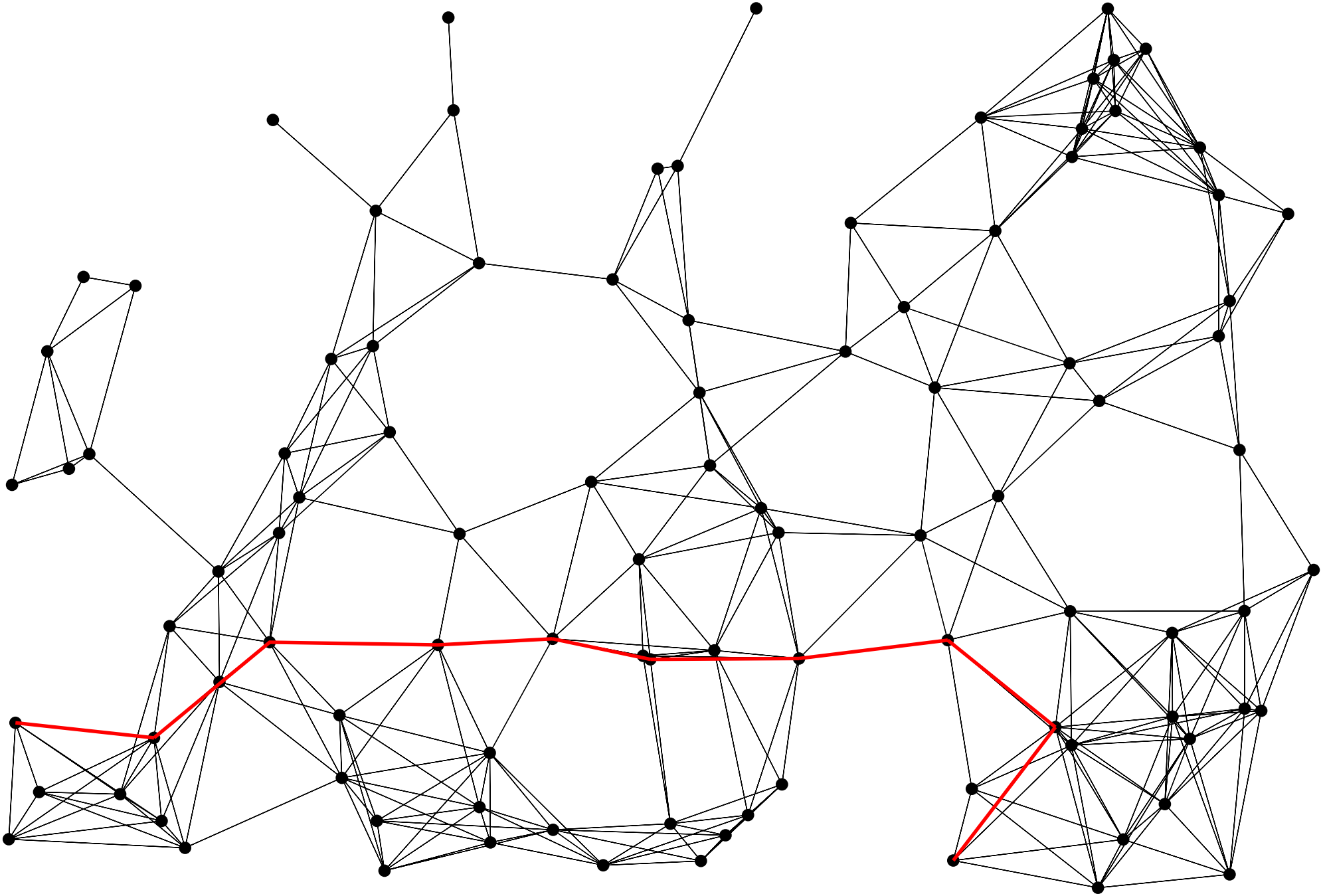
a network of 100 nodes and 684 links



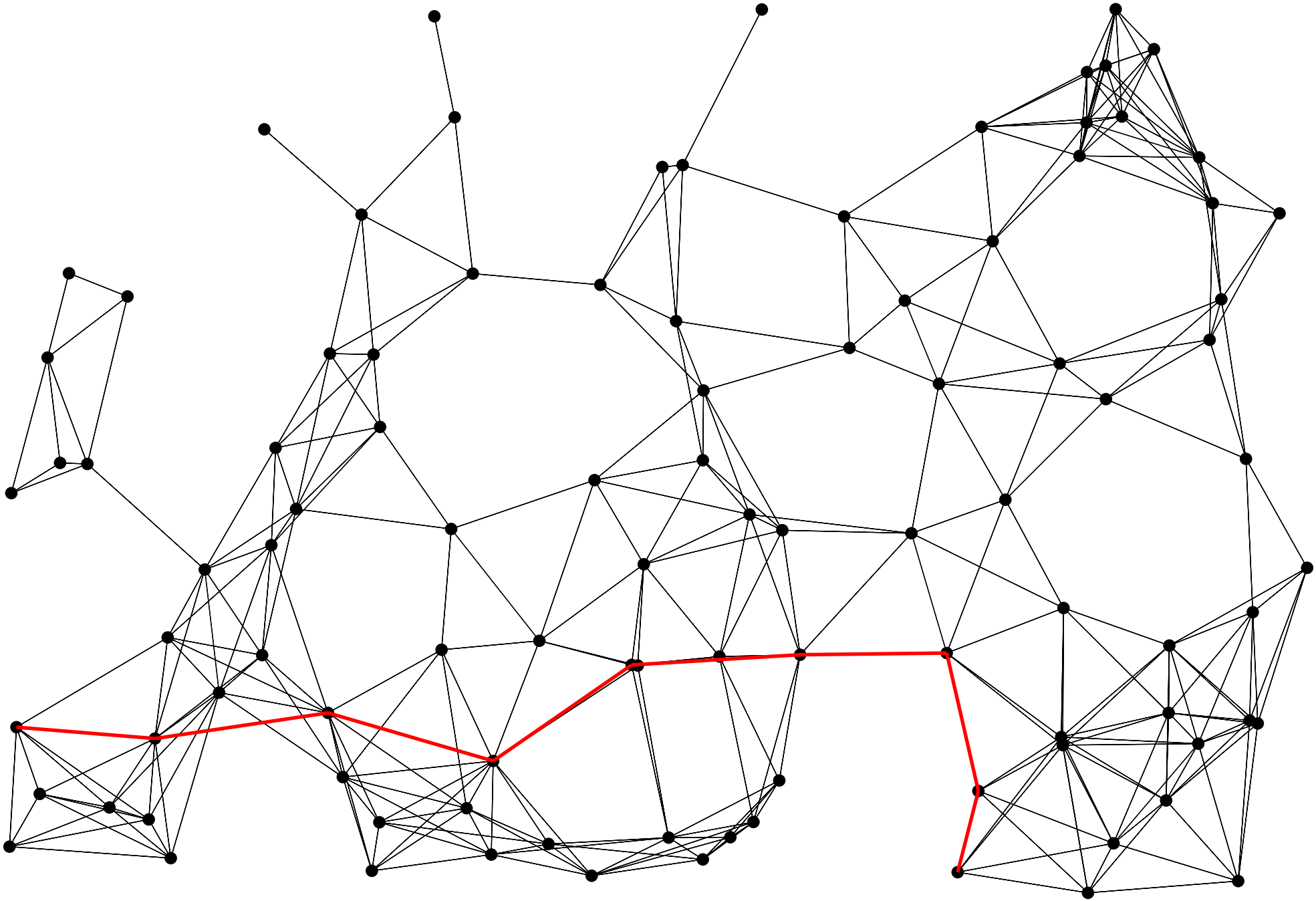
a network of 100 nodes and 702 links



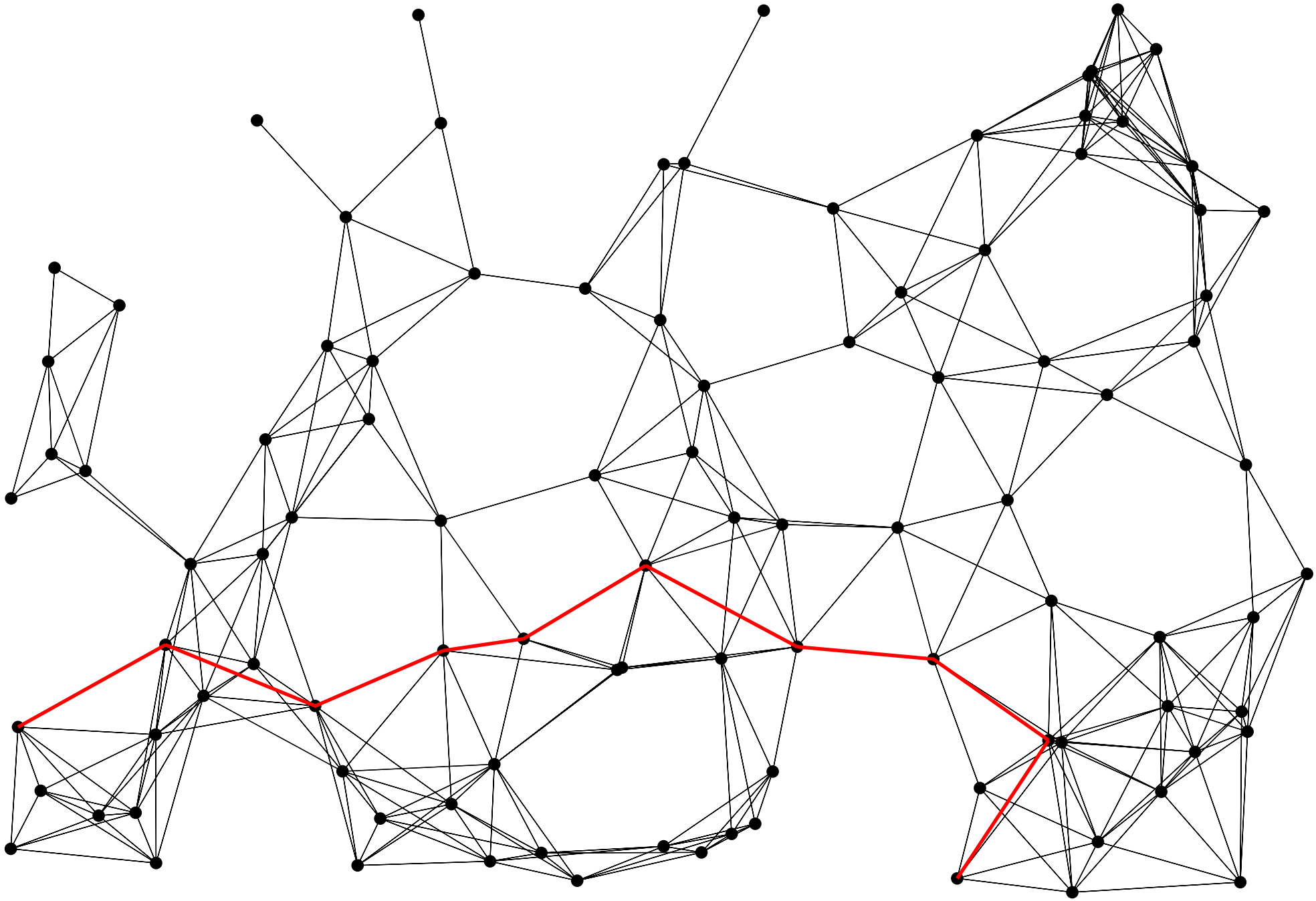
a network of 100 nodes and 698 links



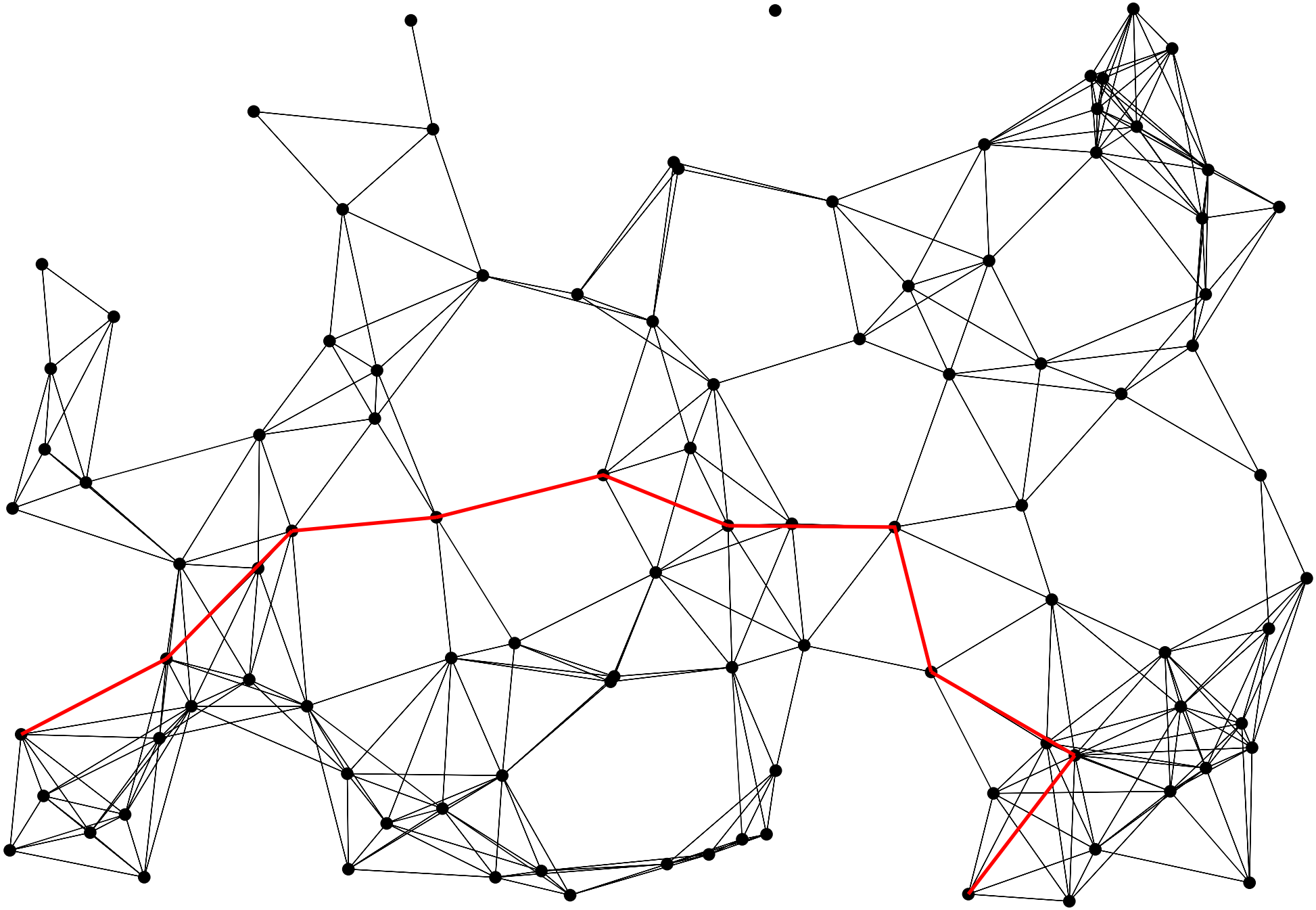
a network of 100 nodes and 692 links



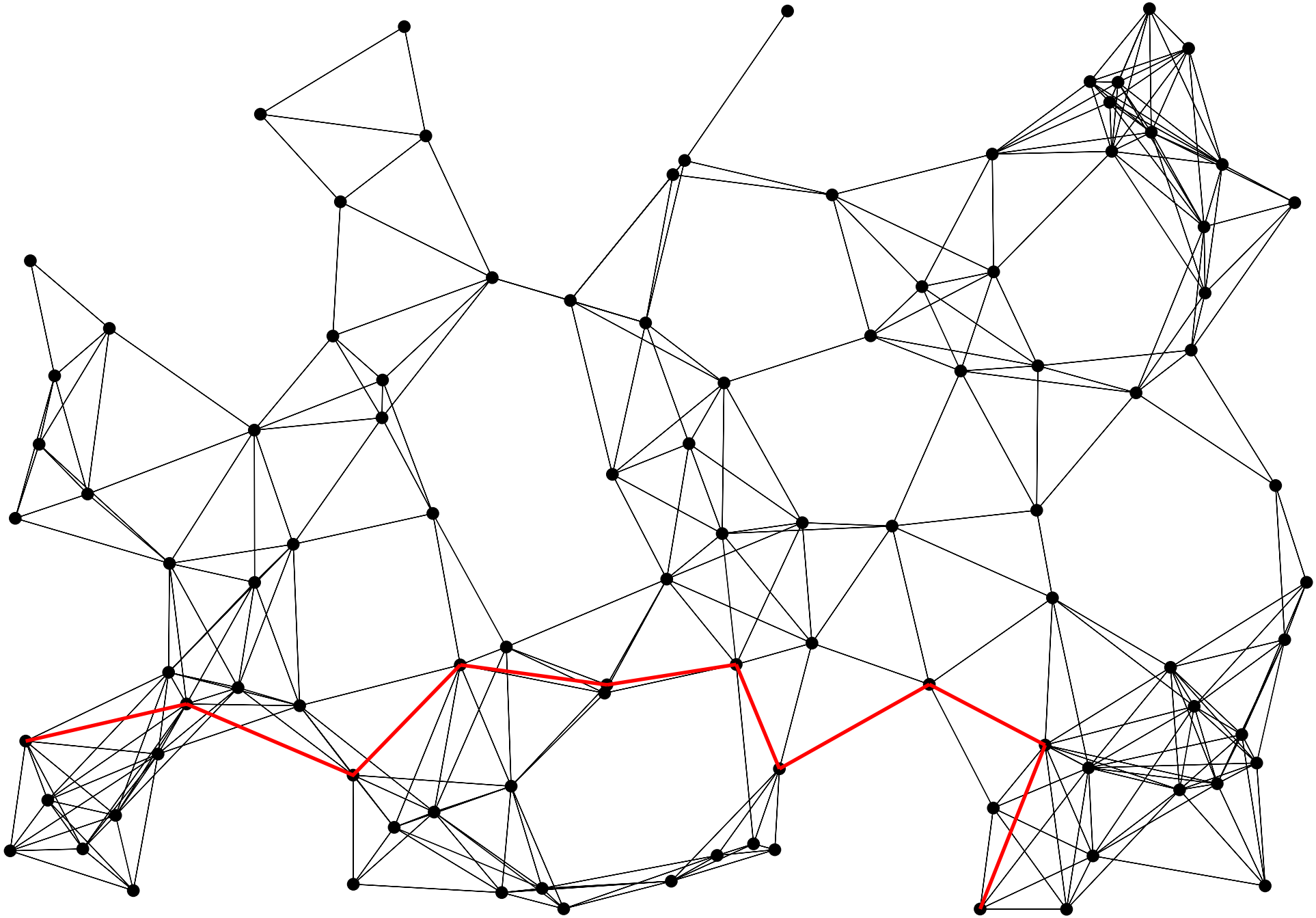
a network of 100 nodes and 678 links



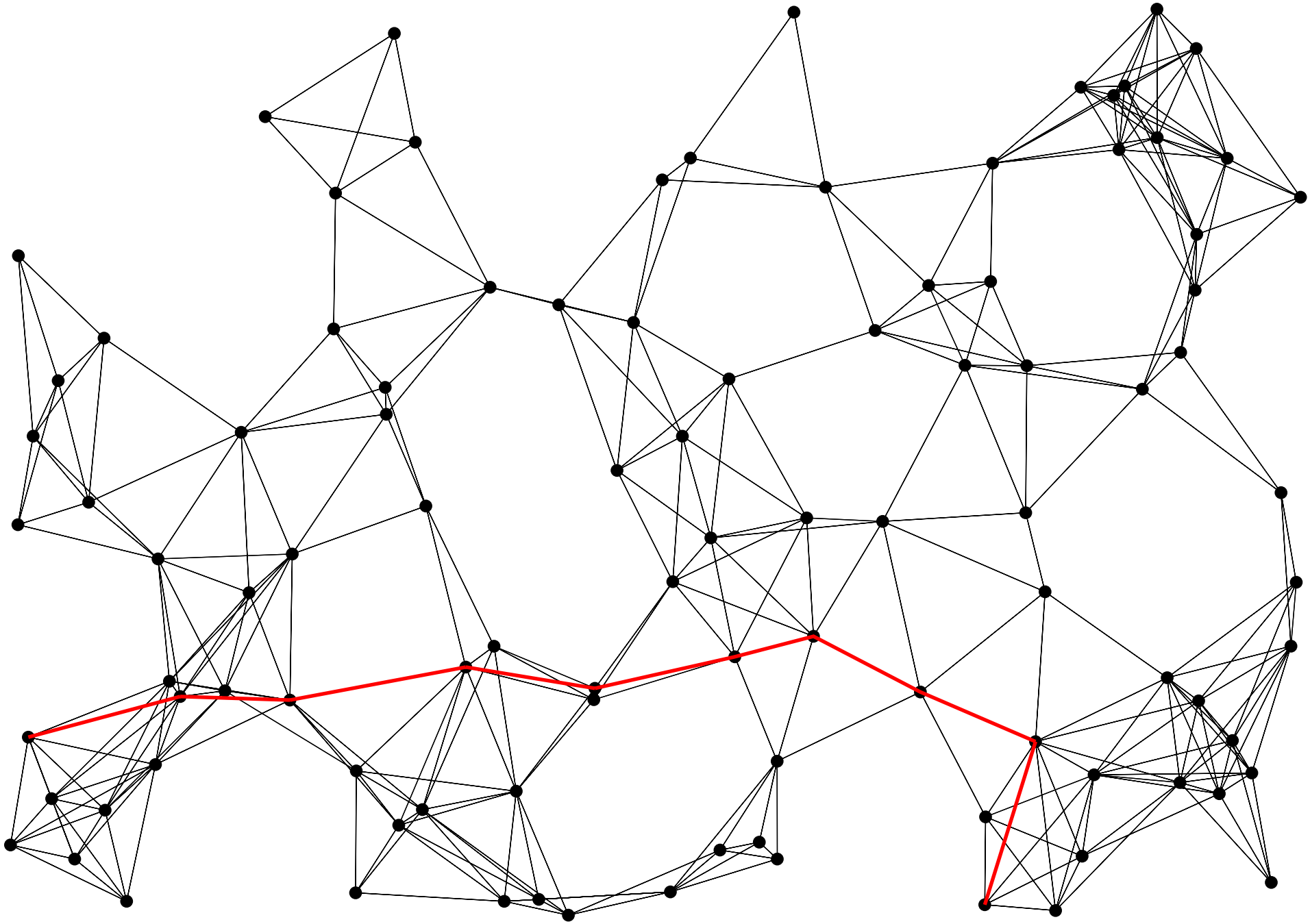
a network of 100 nodes and 690 links



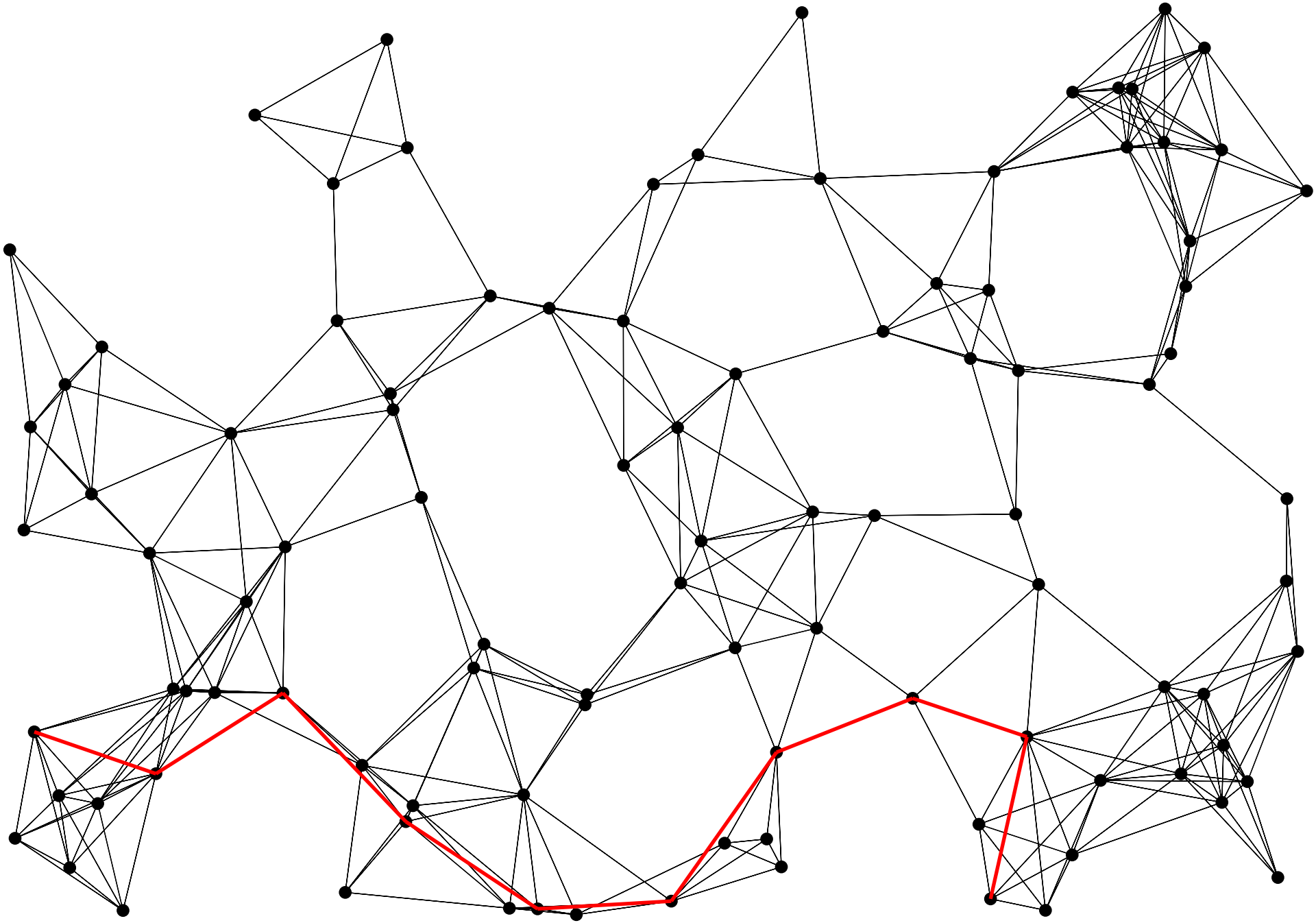
a network of 100 nodes and 698 links



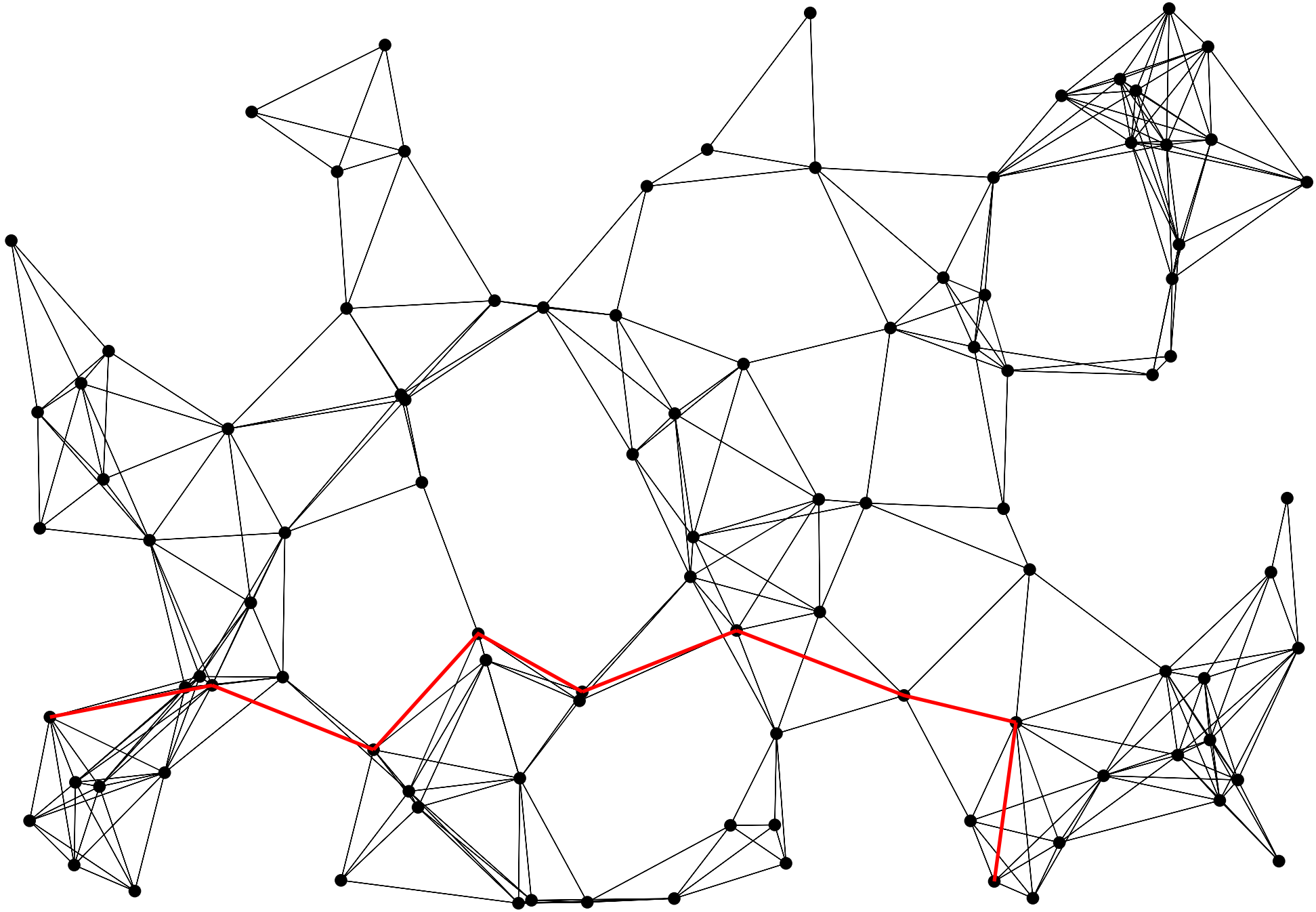
a network of 100 nodes and 700 links



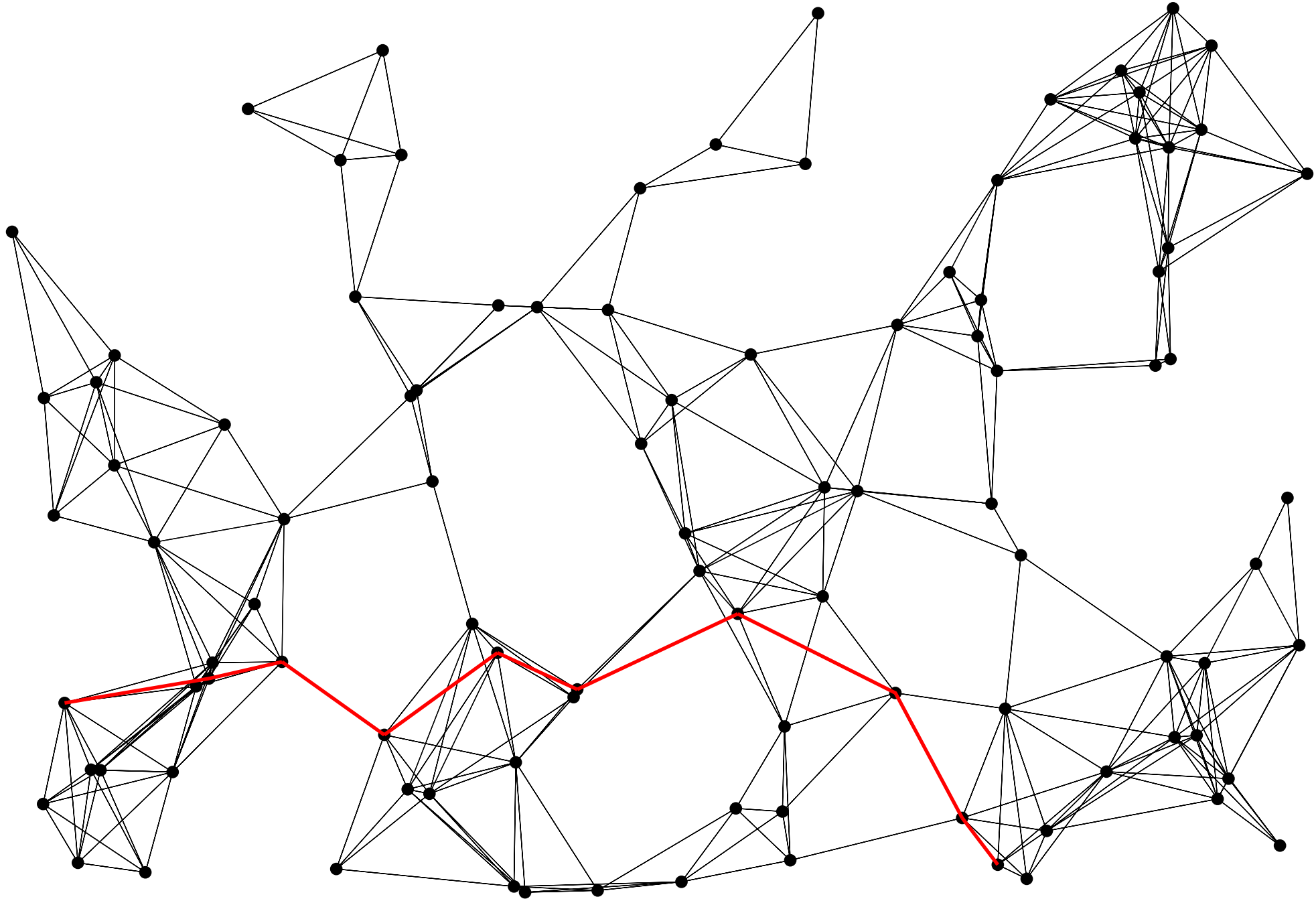
a network of 100 nodes and 680 links



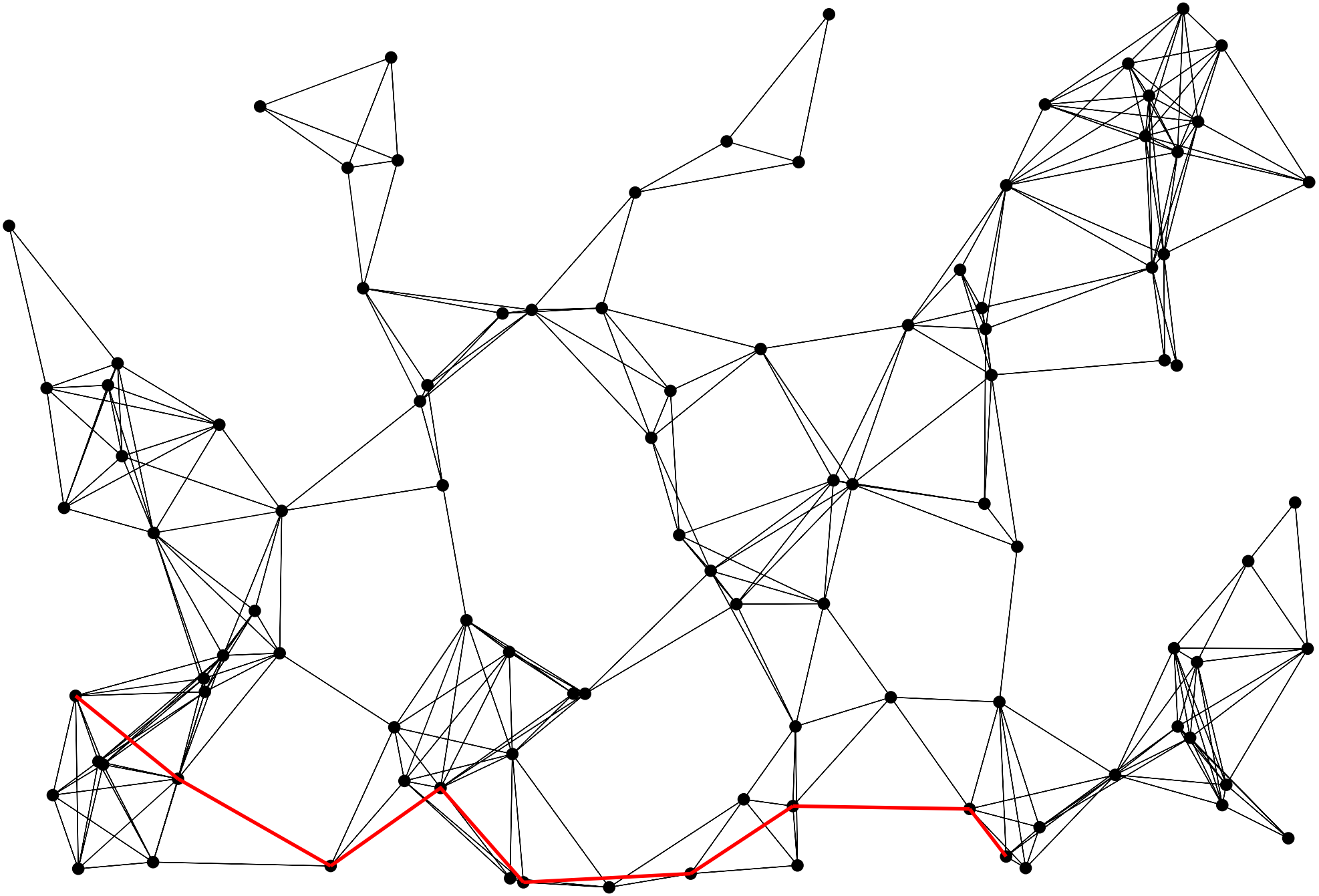
a network of 100 nodes and 672 links



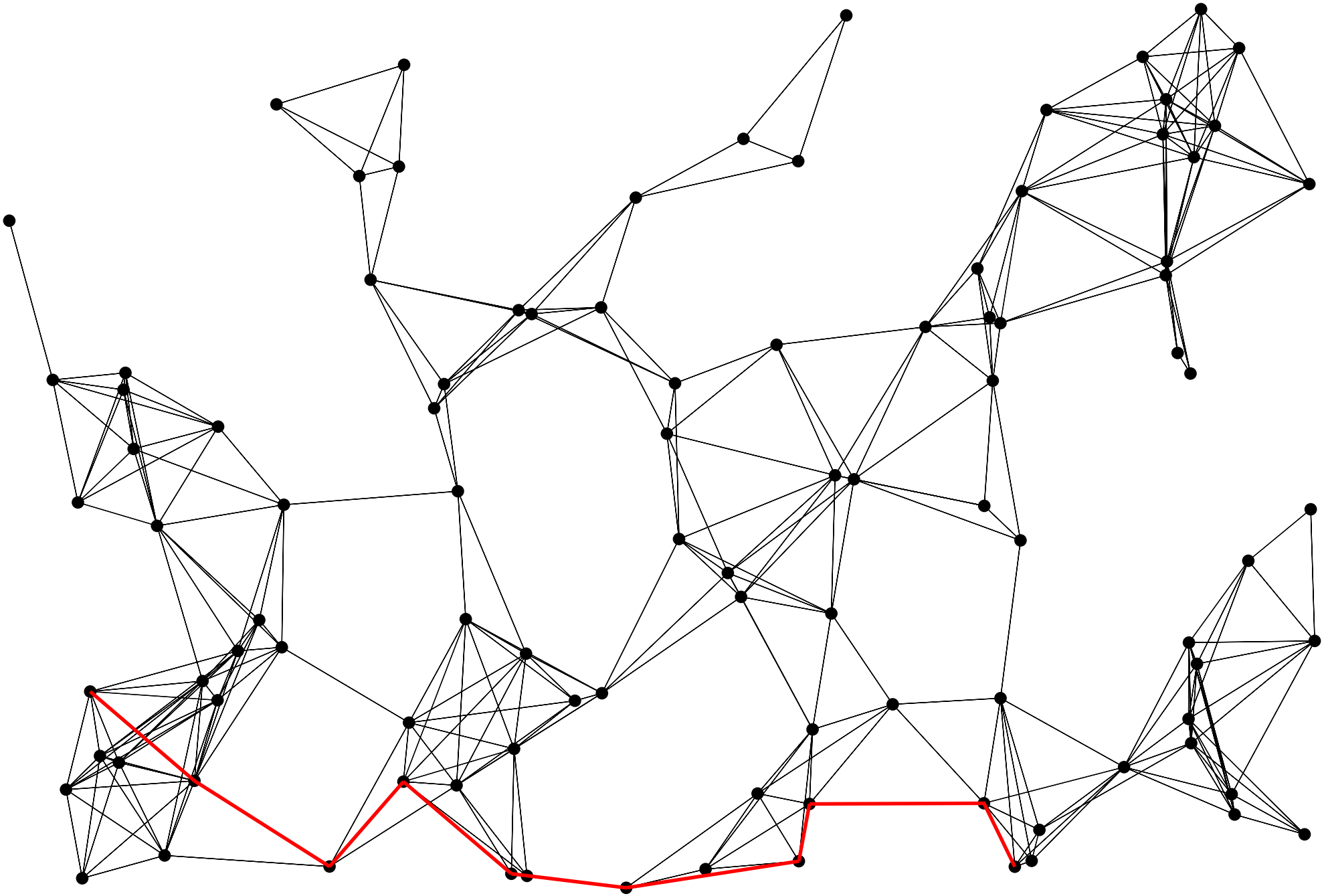
a network of 100 nodes and 678 links



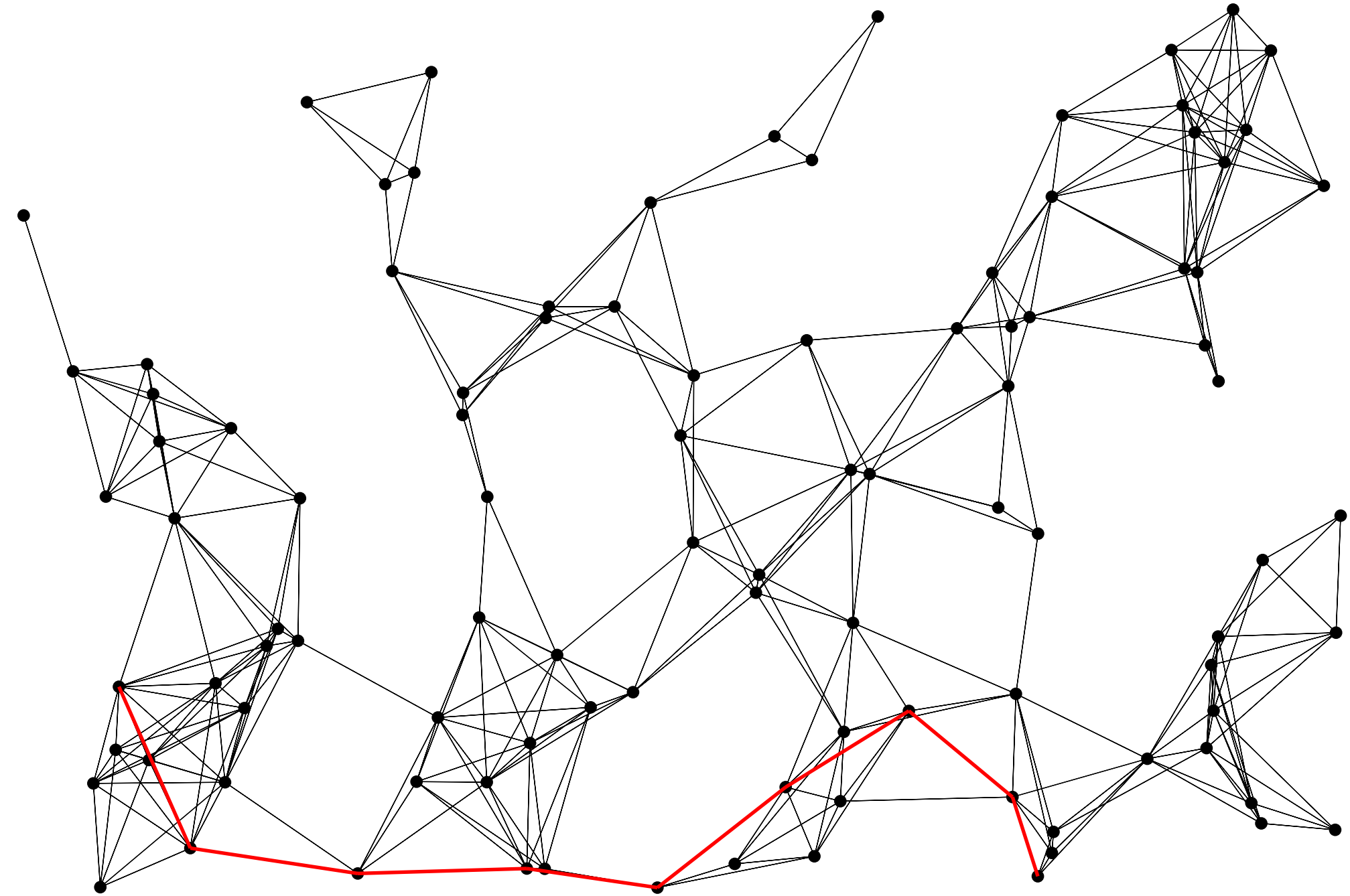
a network of 100 nodes and 670 links



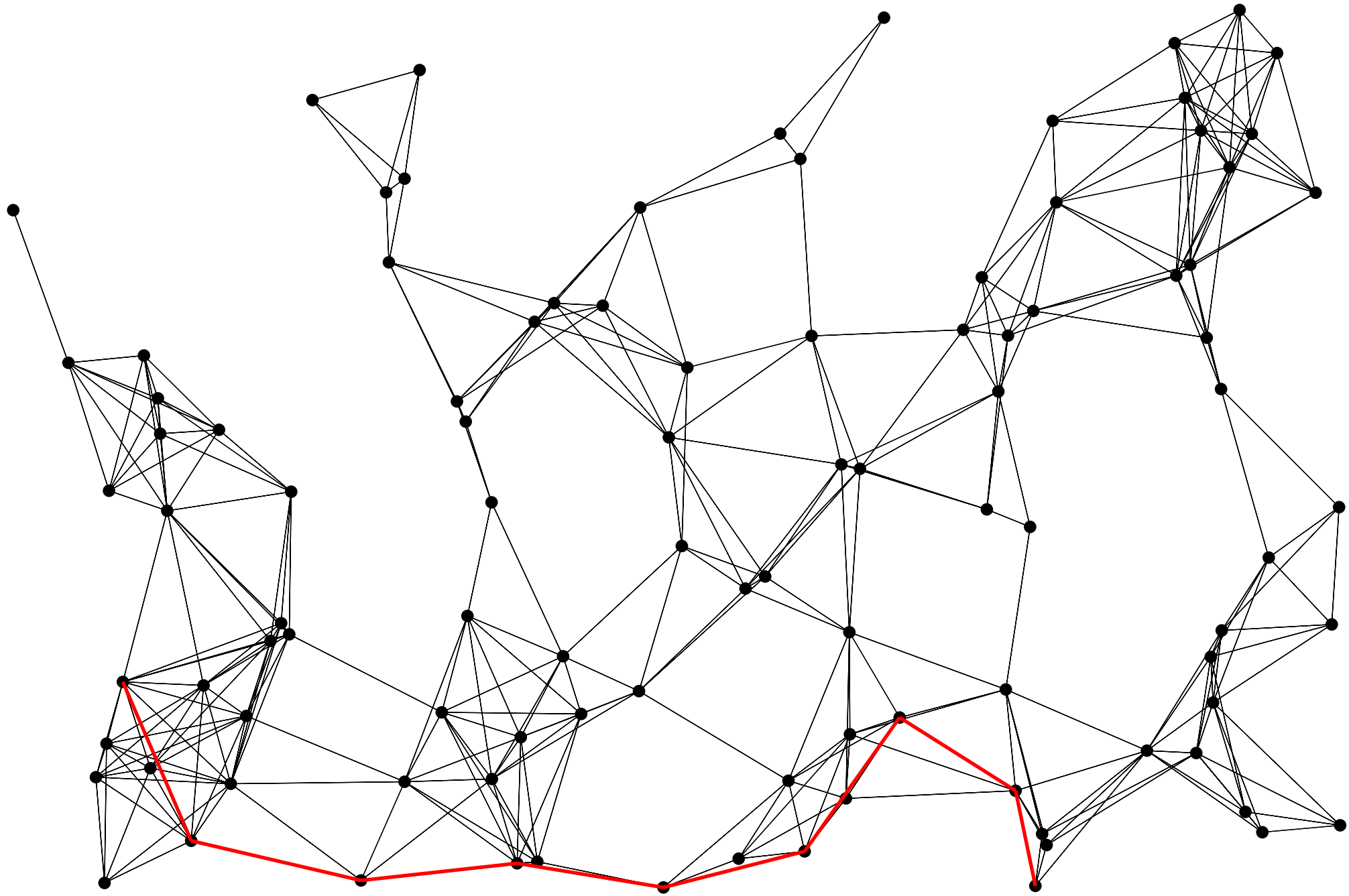
a network of 100 nodes and 666 links



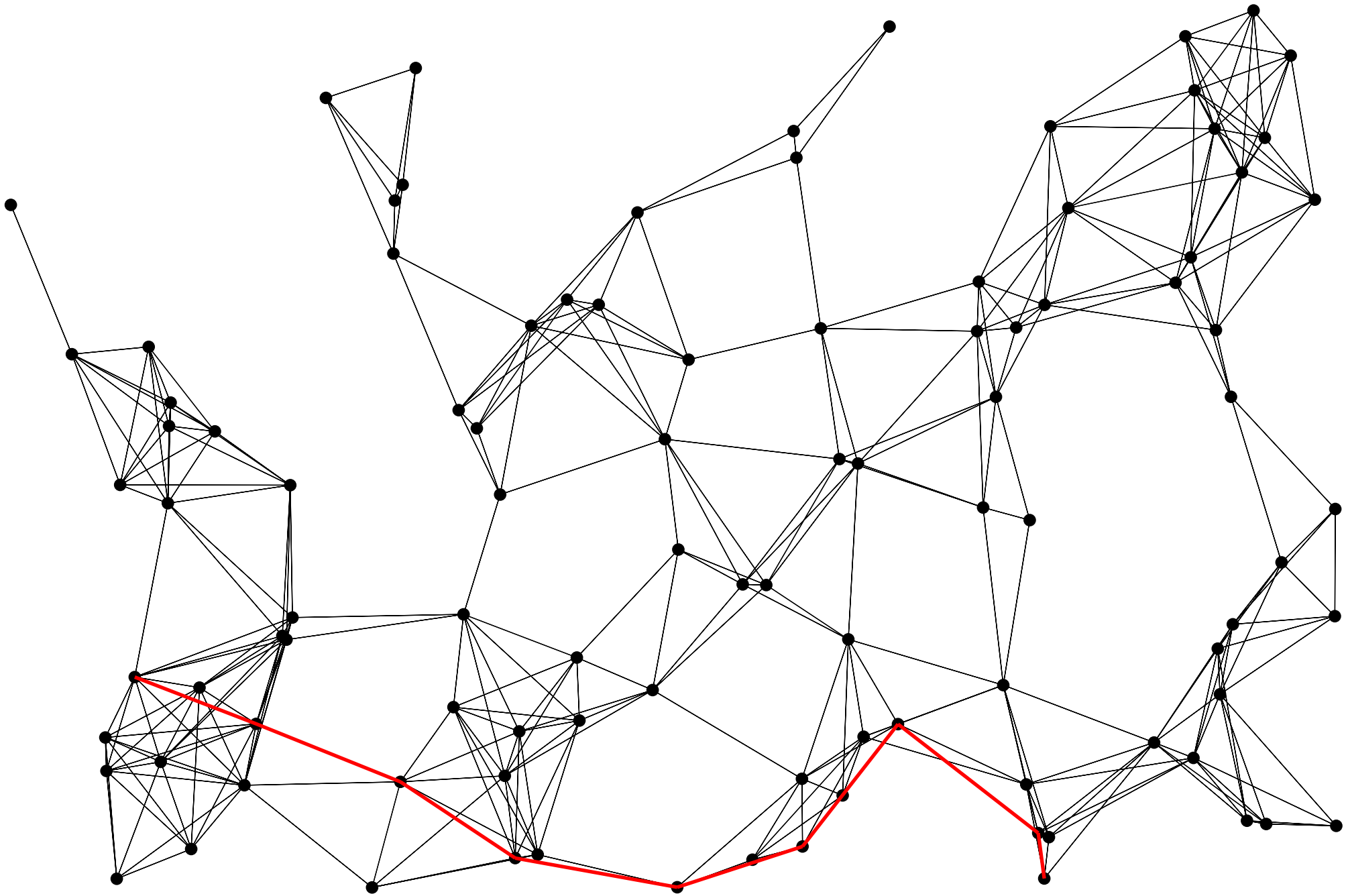
a network of 100 nodes and 676 links



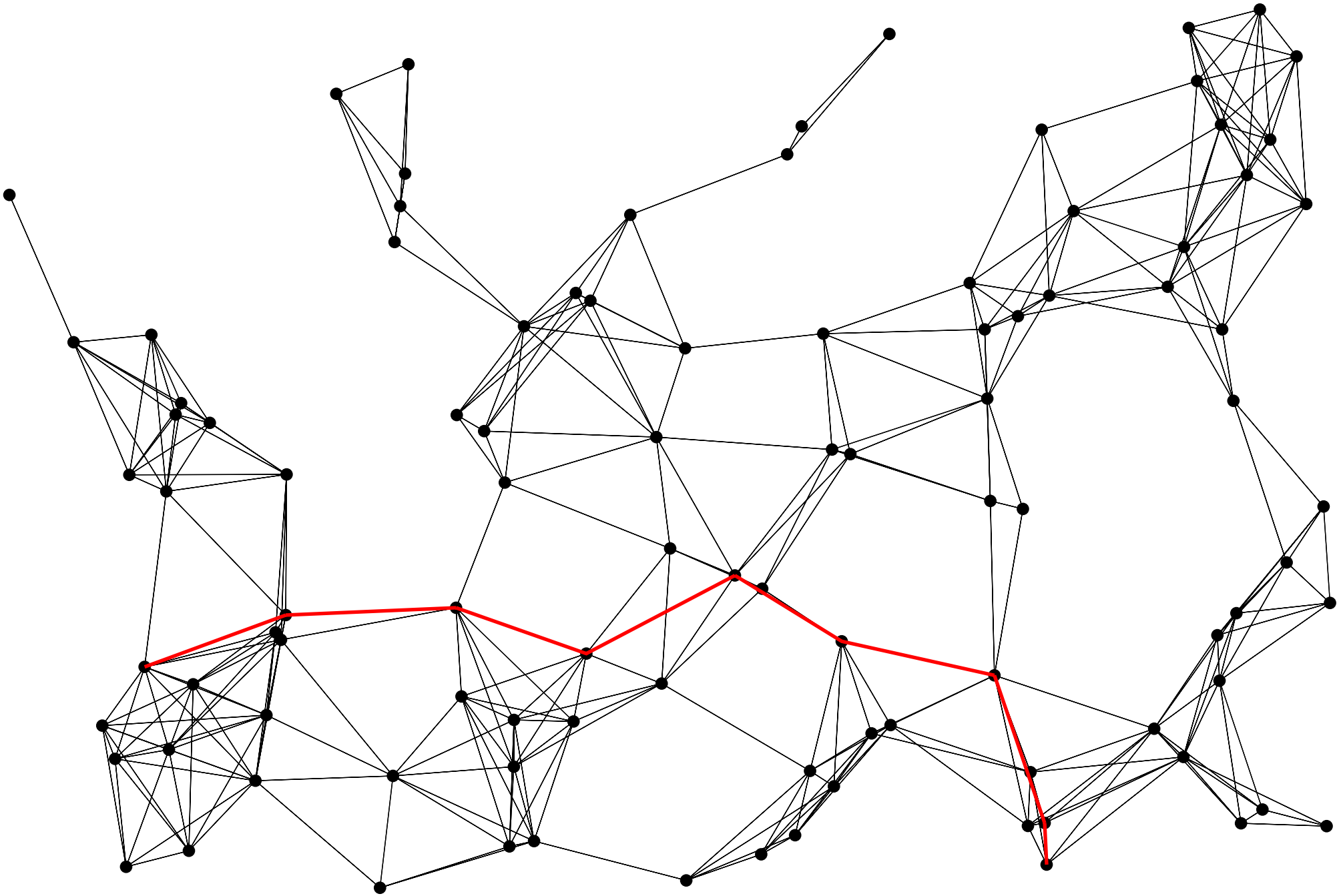
a network of 100 nodes and 694 links



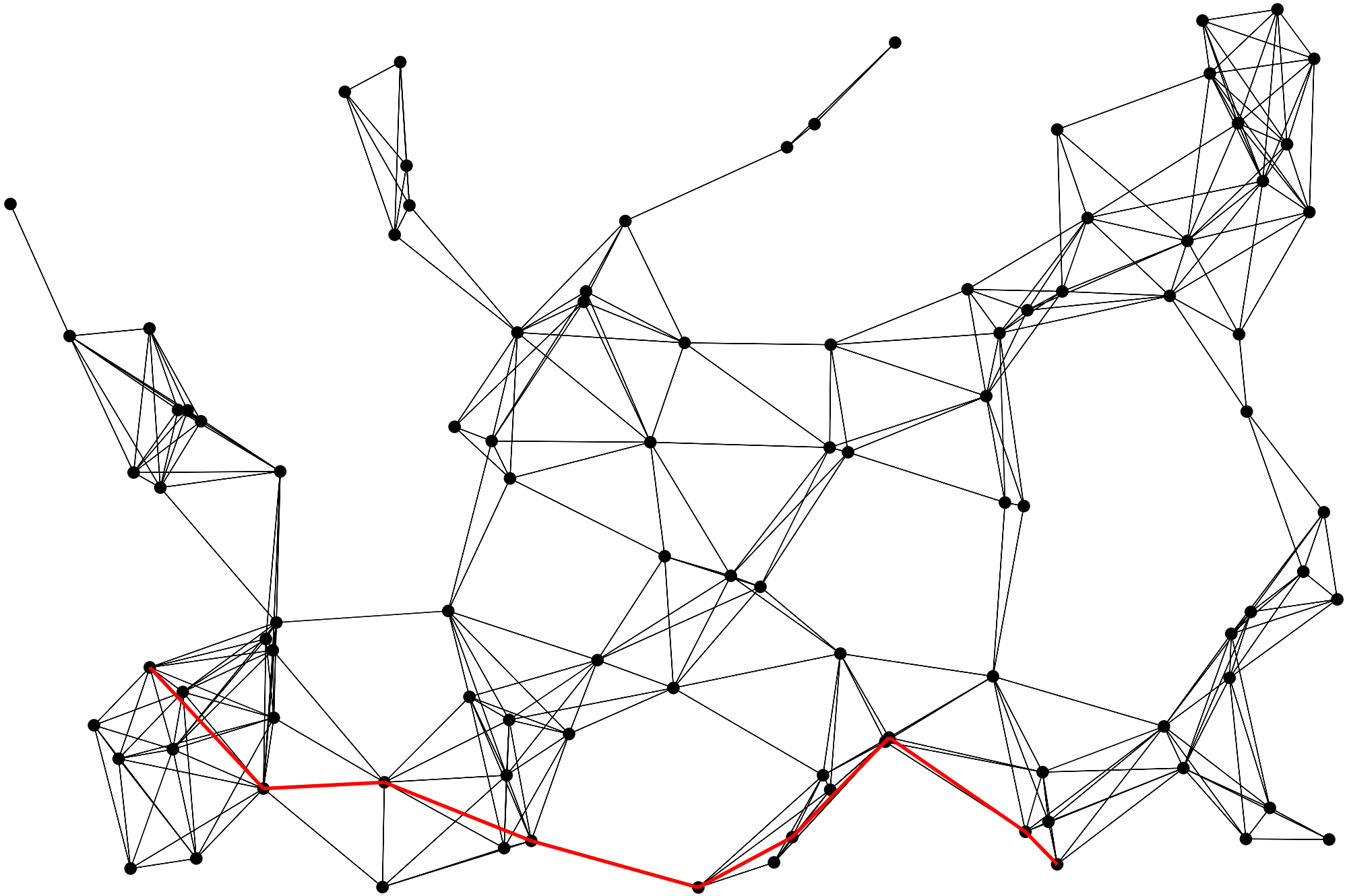
a network of 100 nodes and 710 links



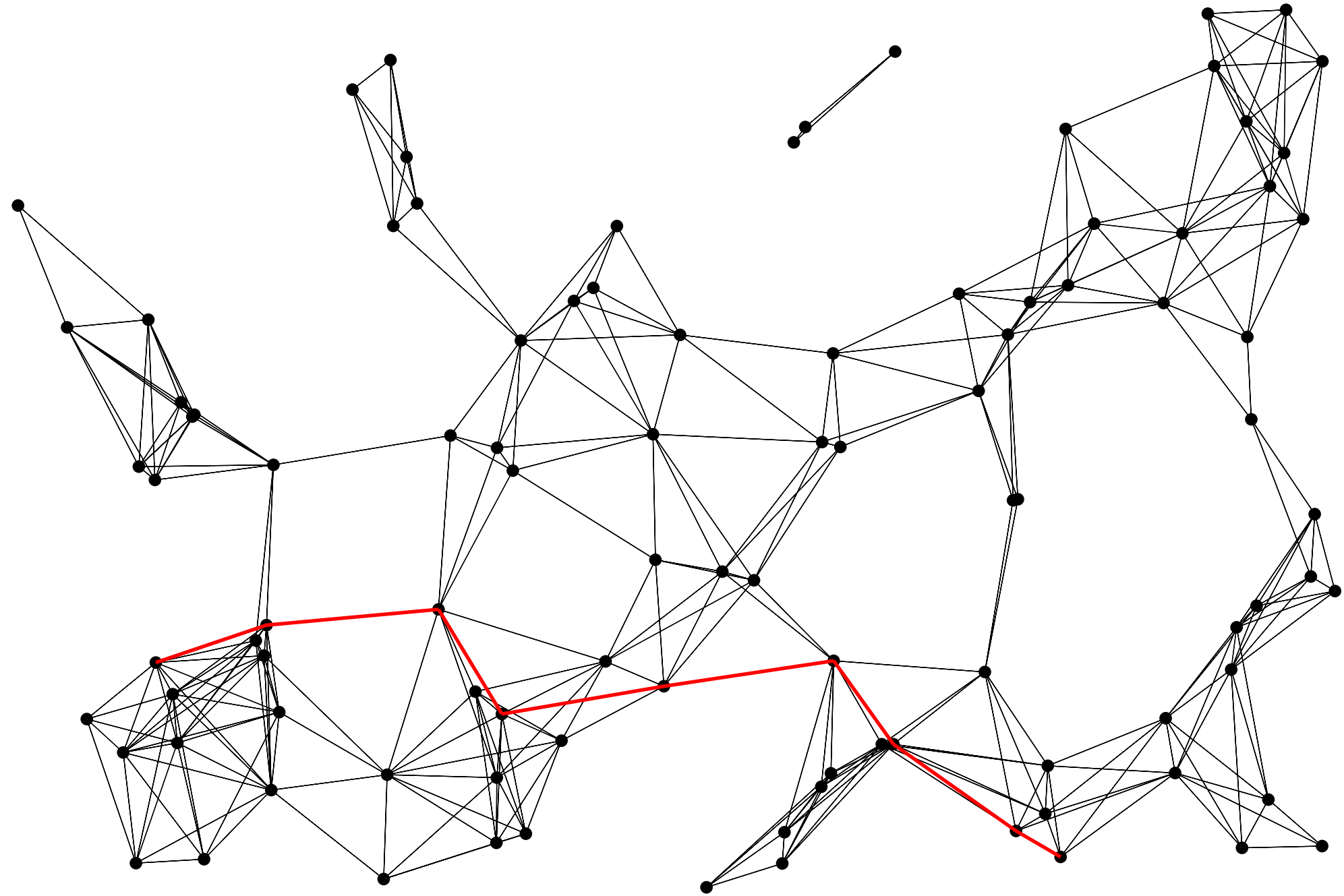
a network of 100 nodes and 710 links

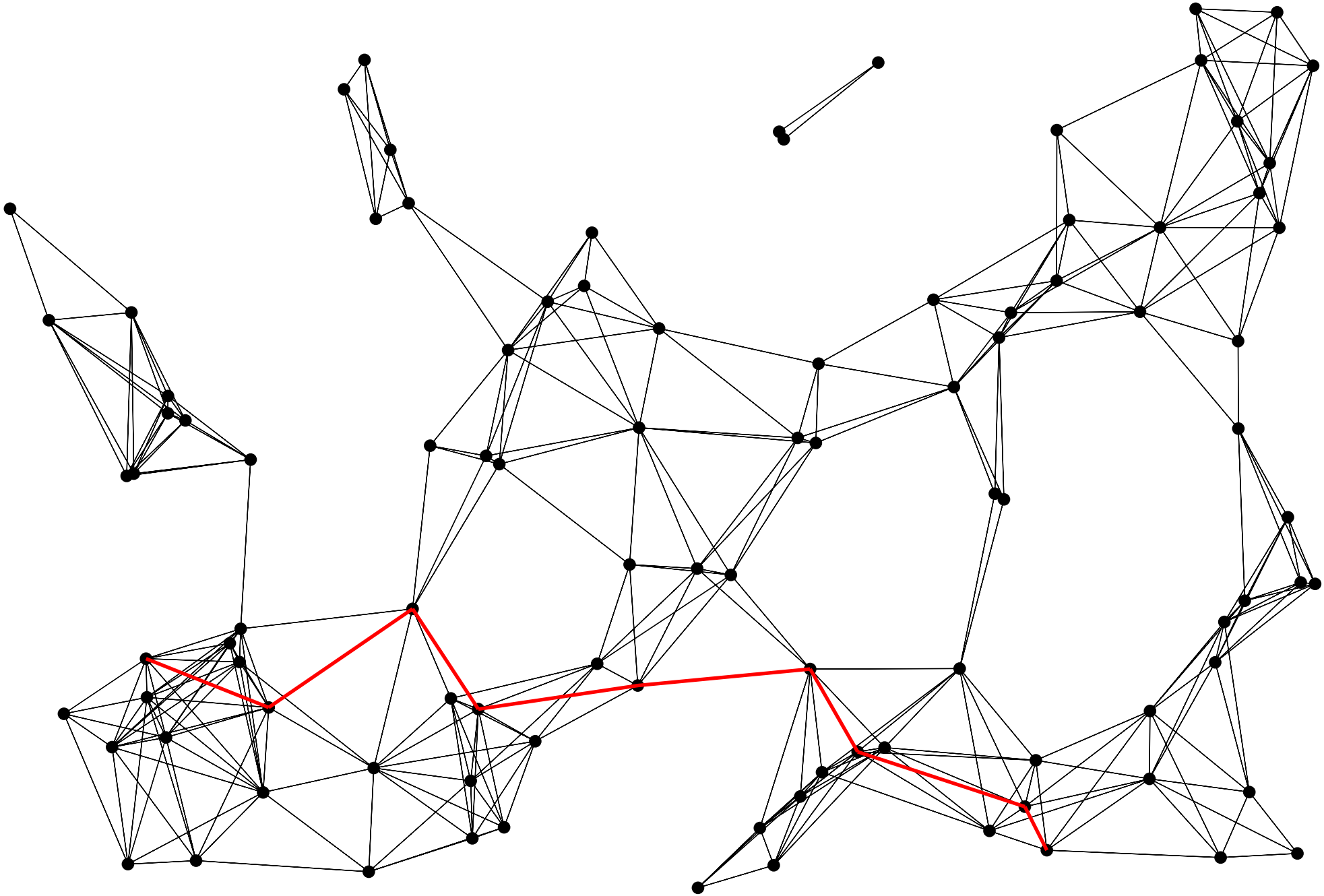


a network of 100 nodes and 698 links

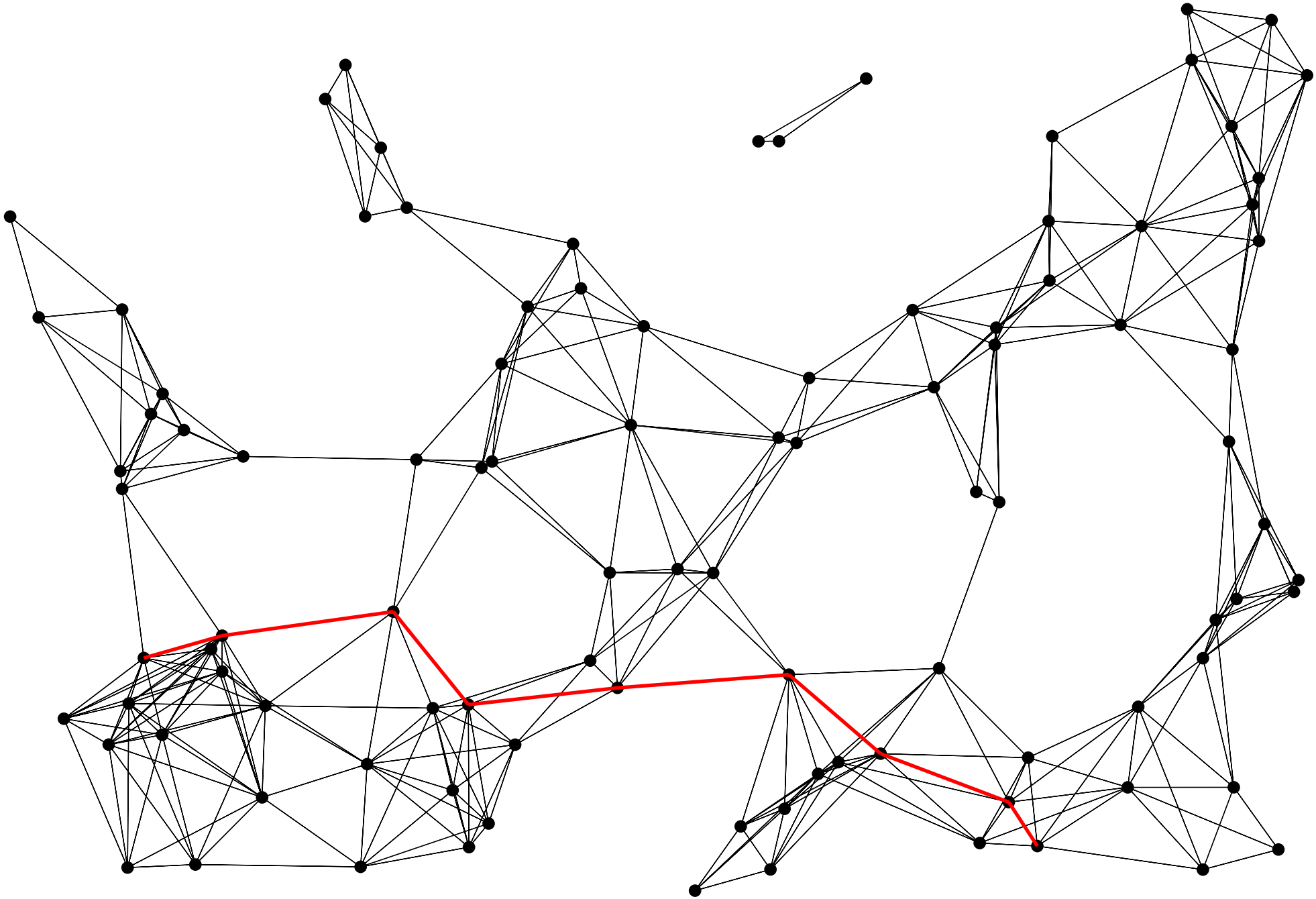


a network of 100 nodes and 704 links

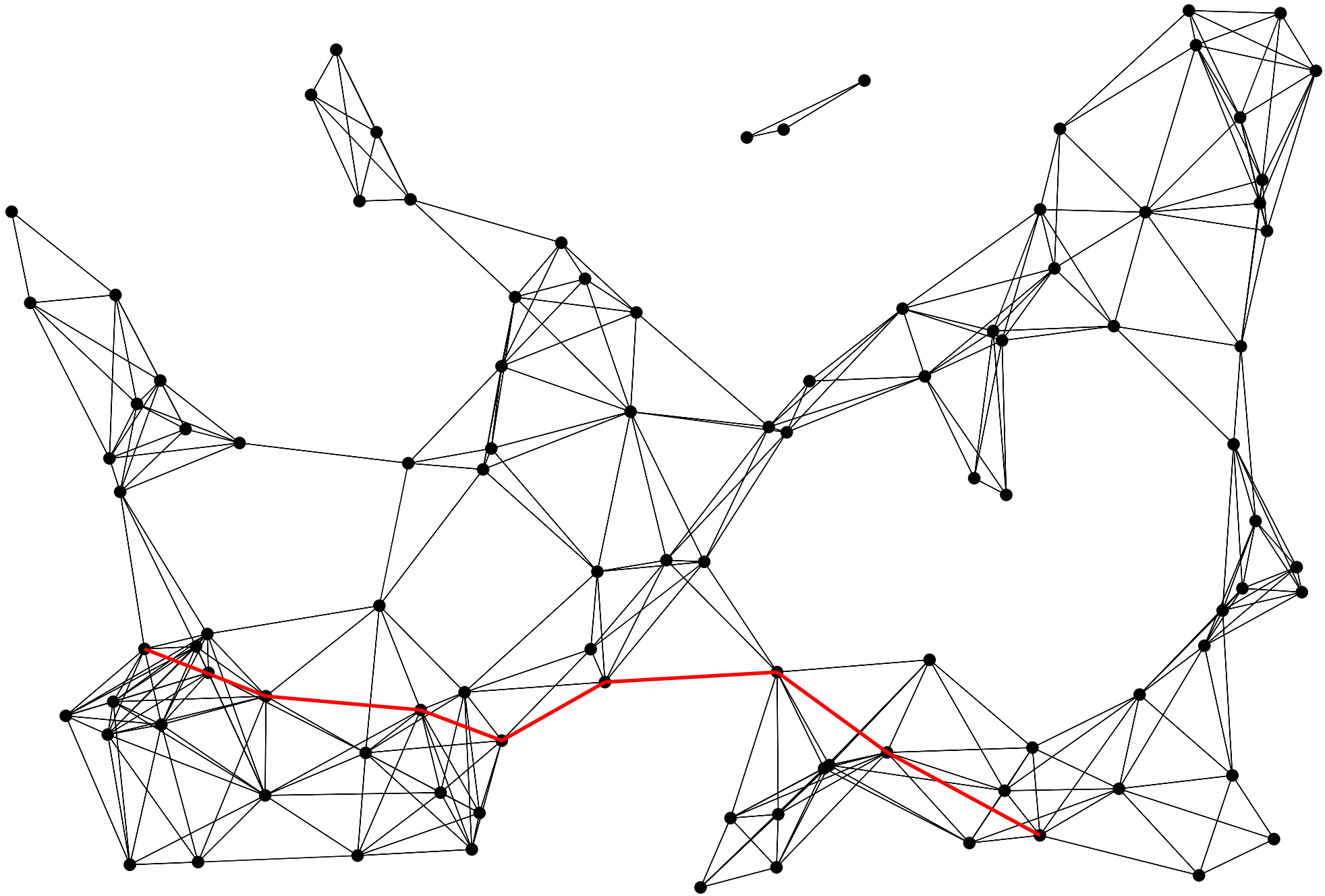

a network of 100 nodes and 702 links



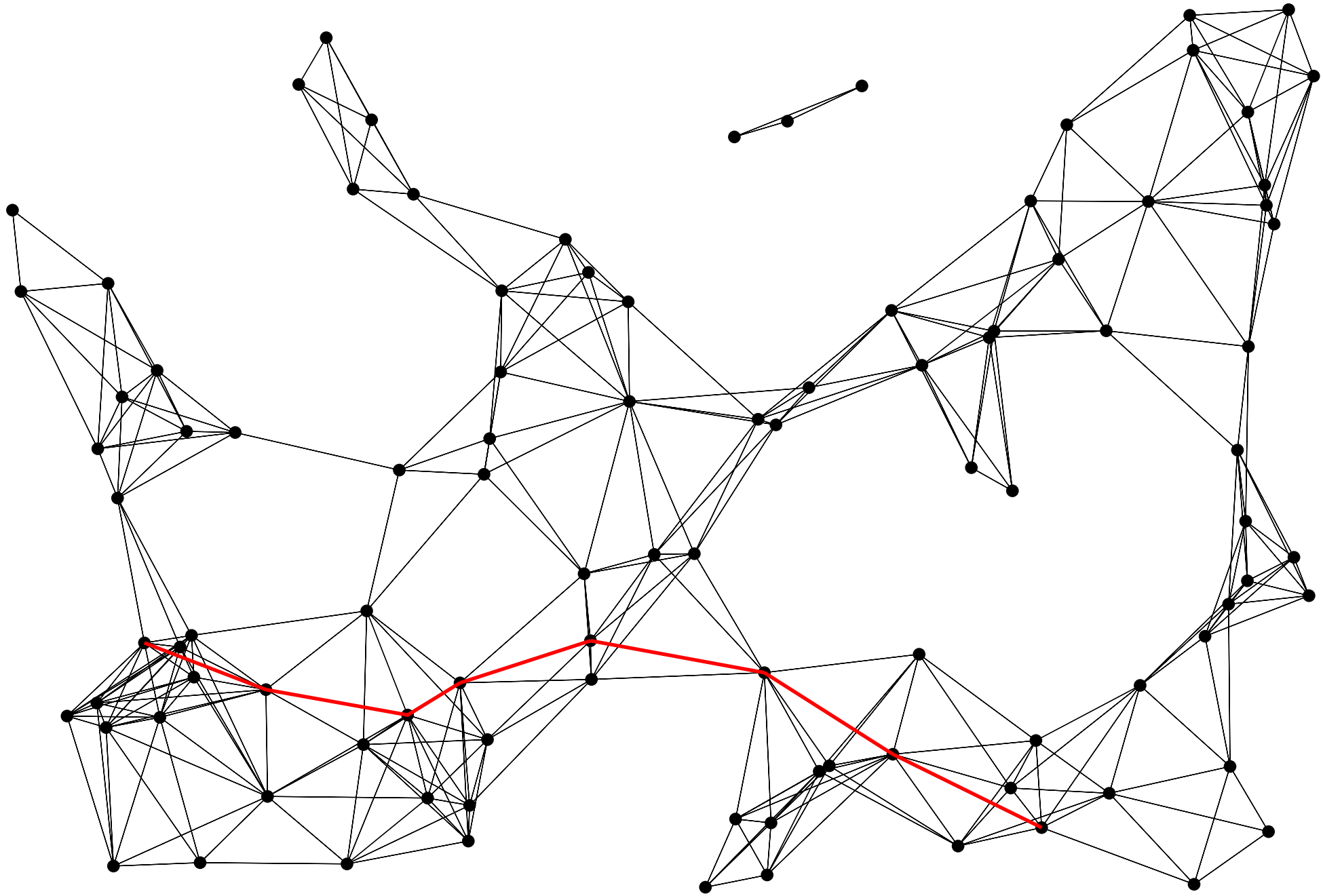
a network of 100 nodes and 702 links



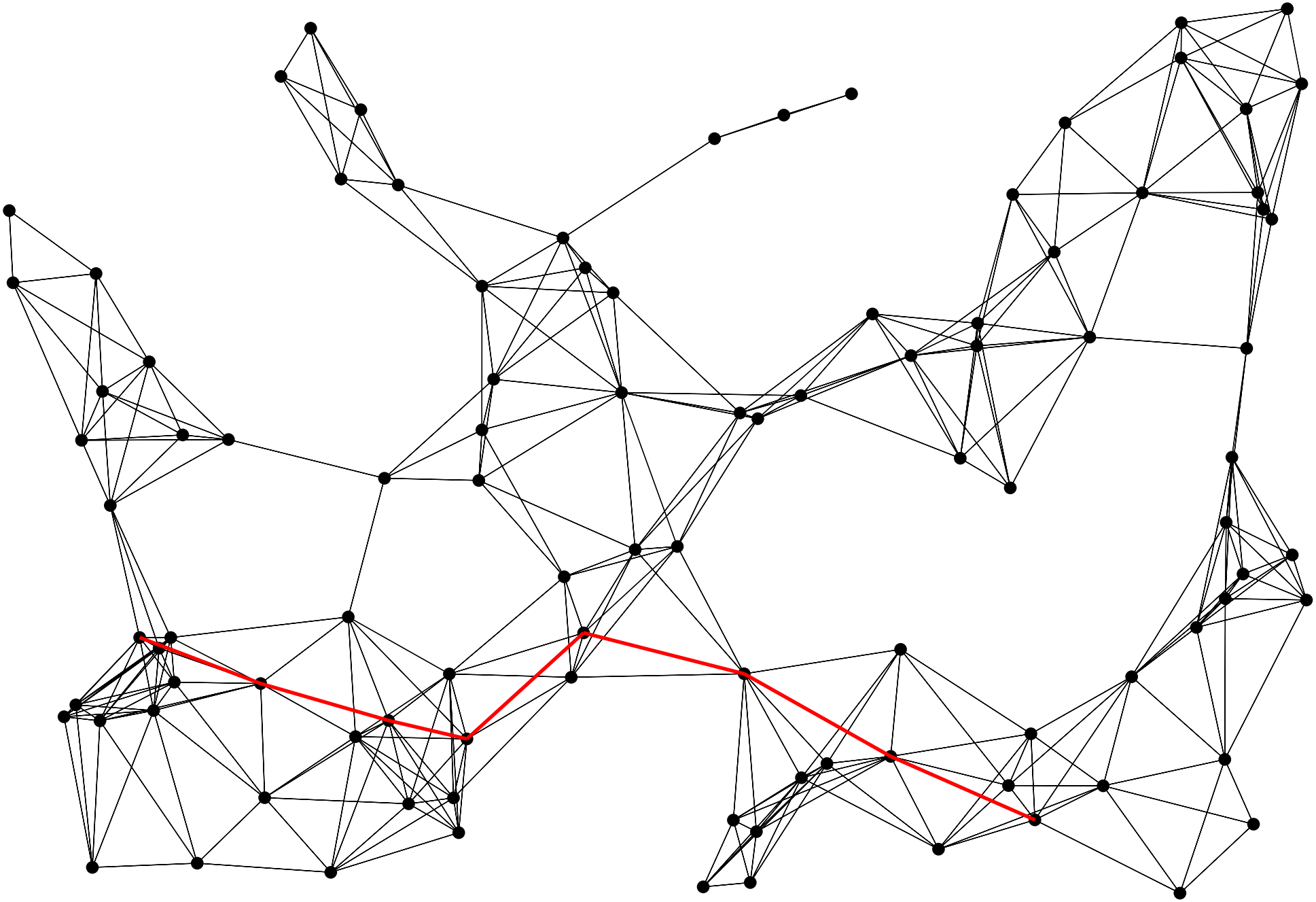
a network of 100 nodes and 708 links



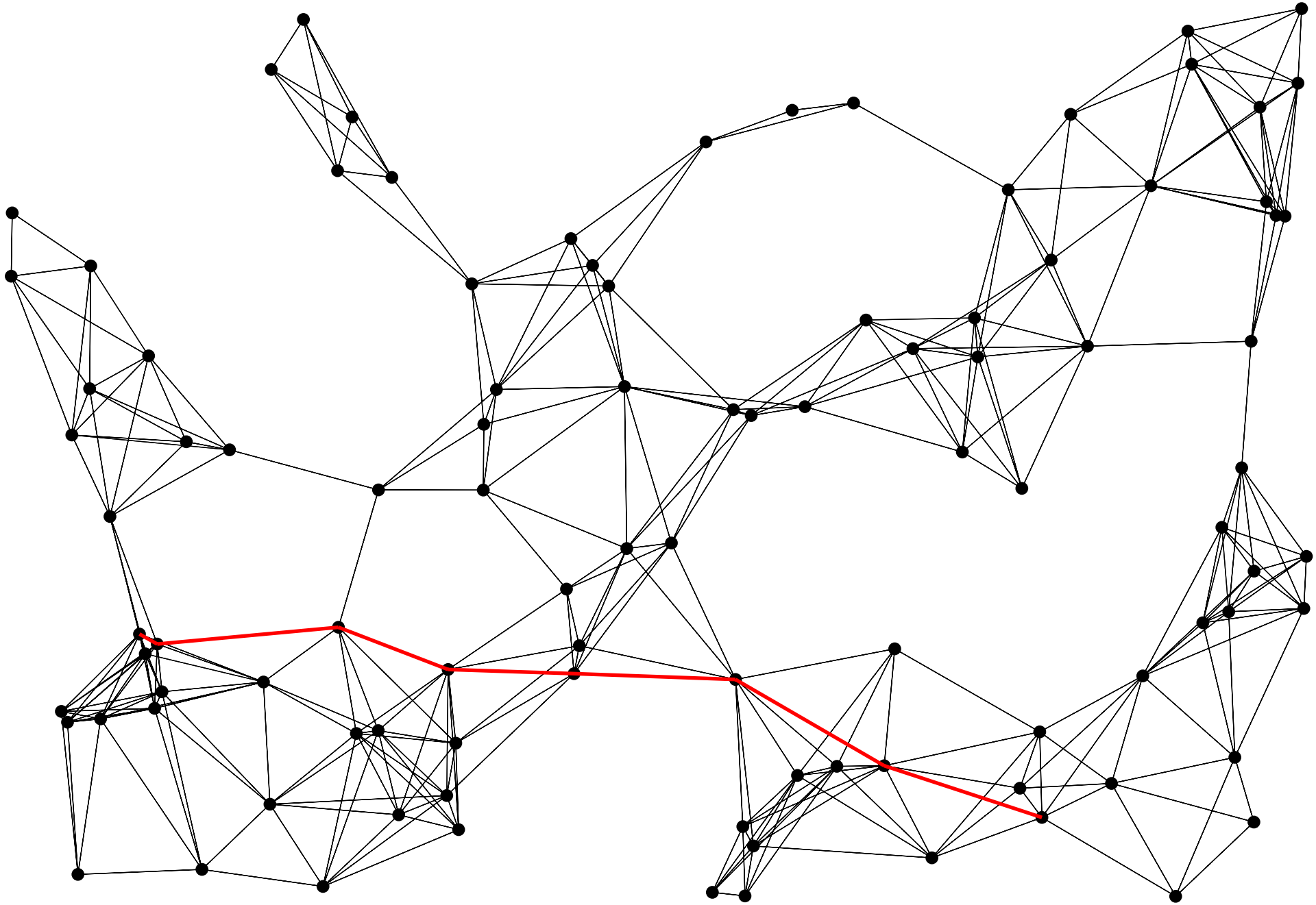
a network of 100 nodes and 708 links



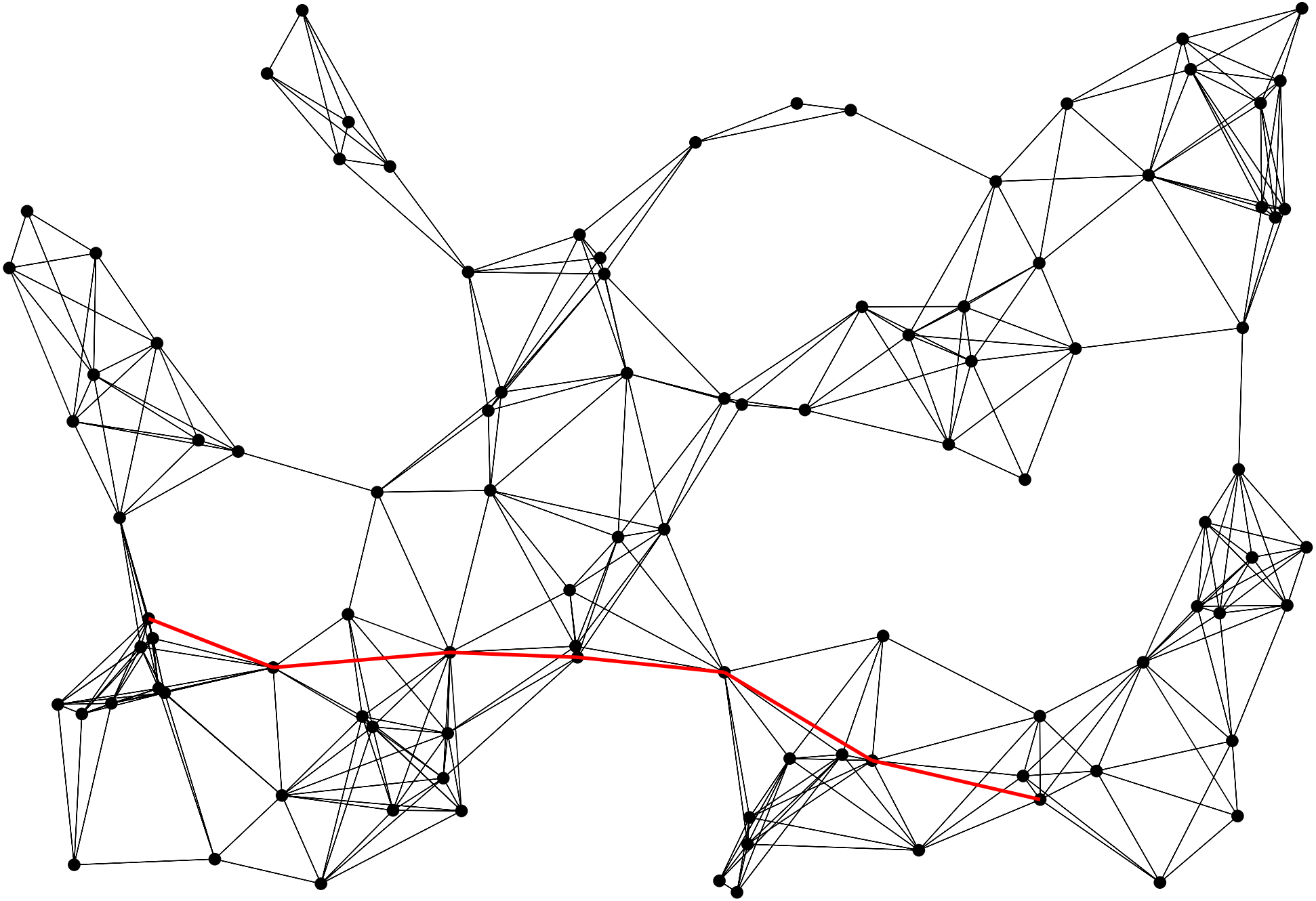
a network of 100 nodes and 710 links



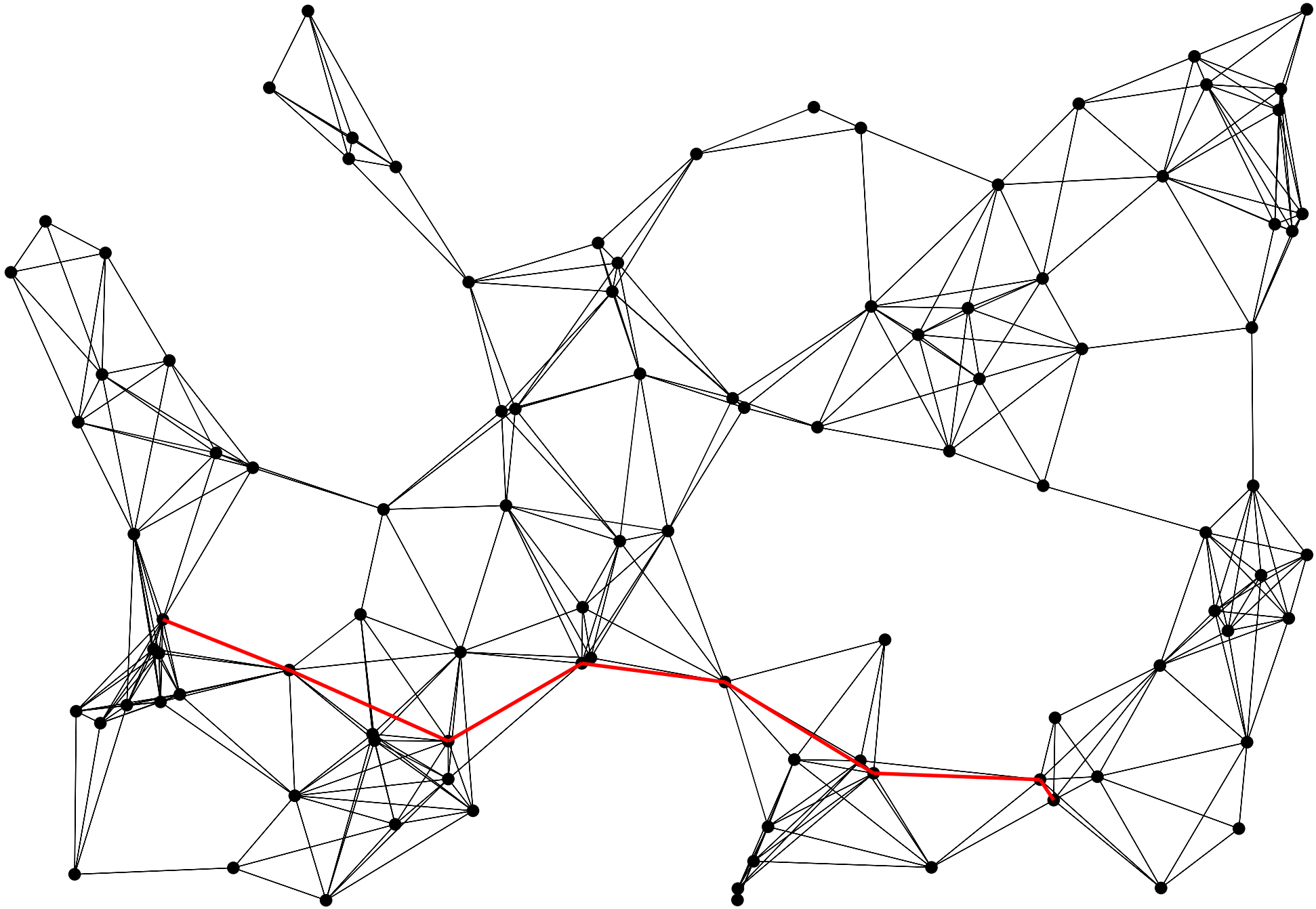
a network of 100 nodes and 702 links



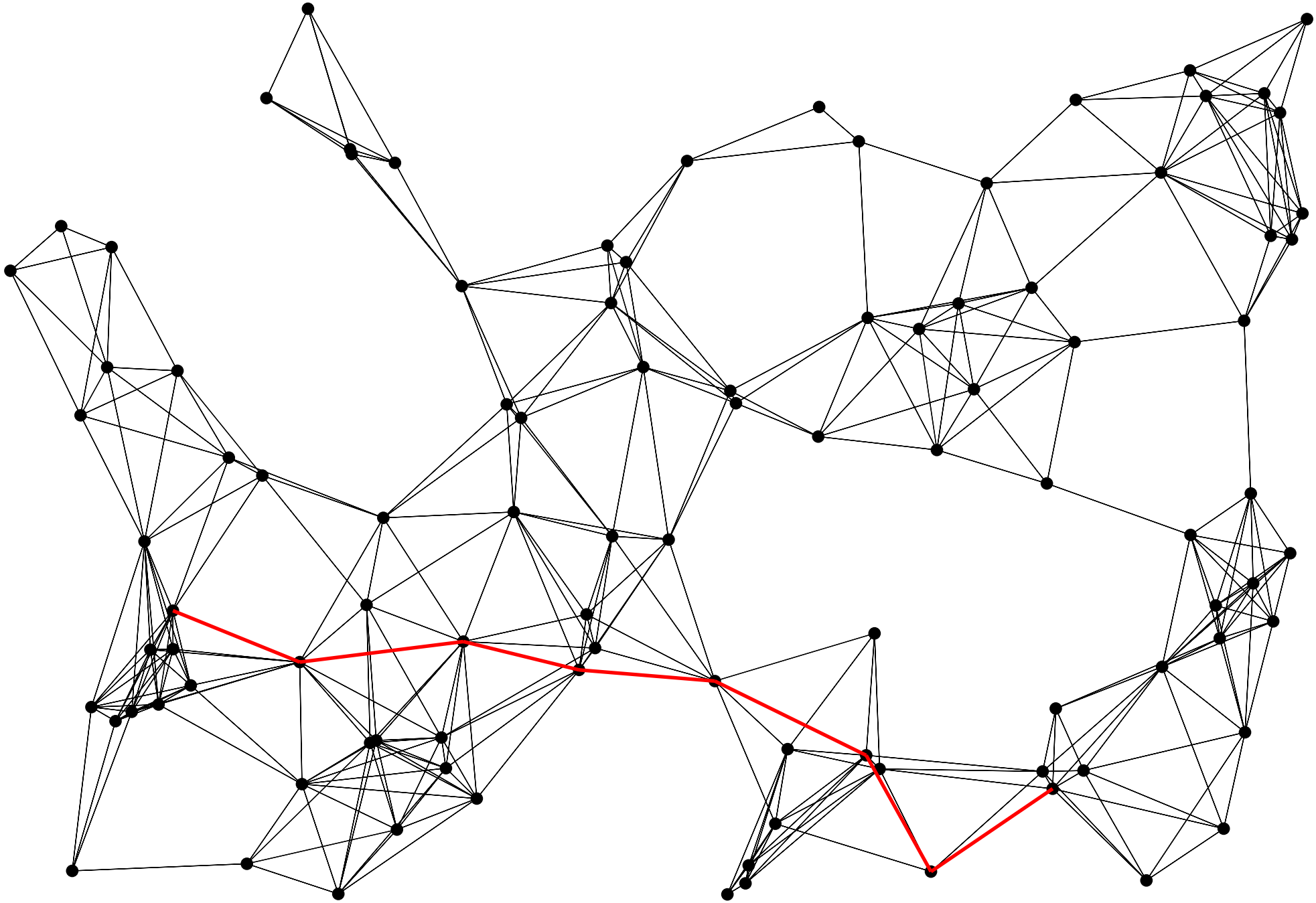
a network of 100 nodes and 706 links



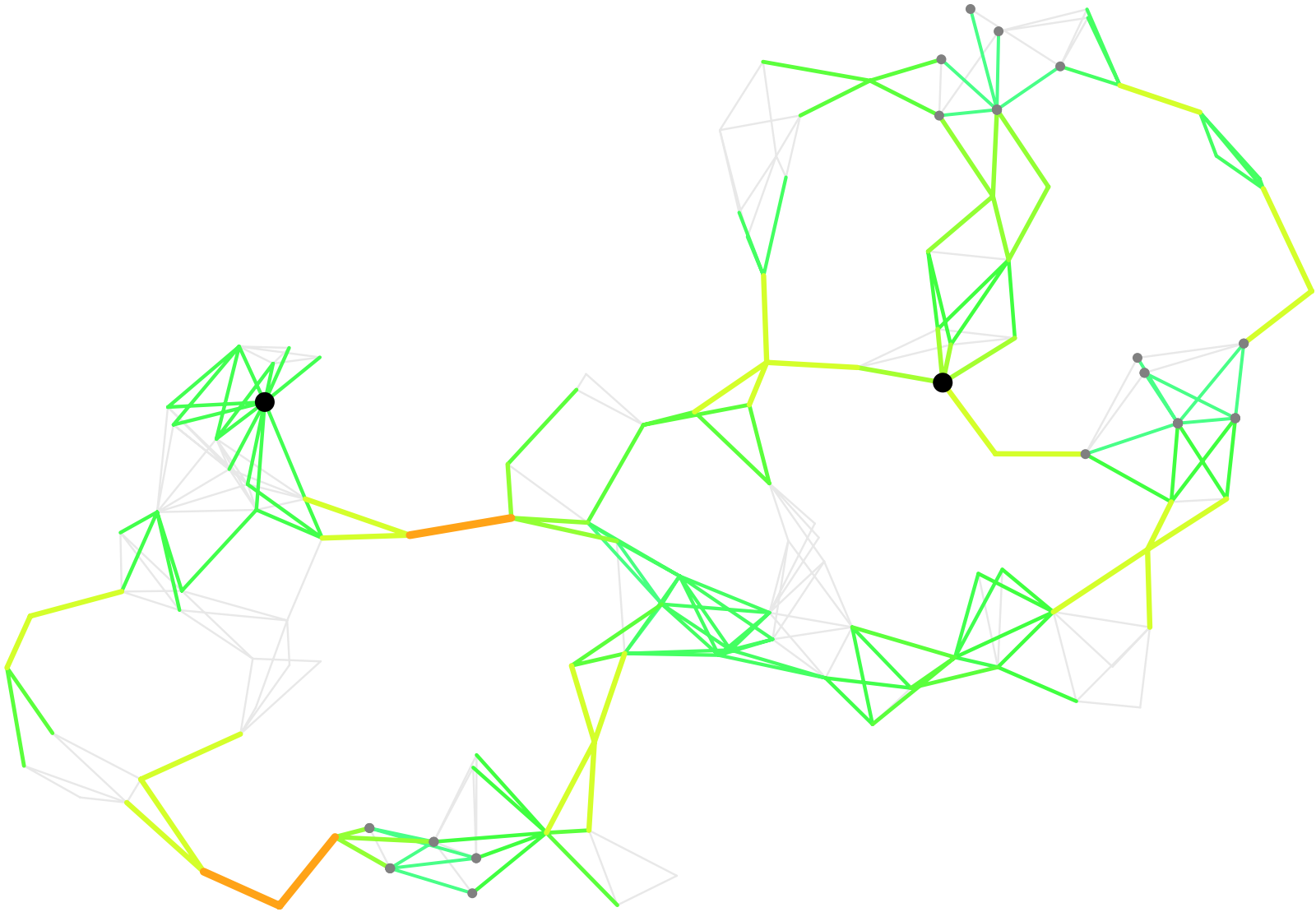
a network of 100 nodes and 718 links



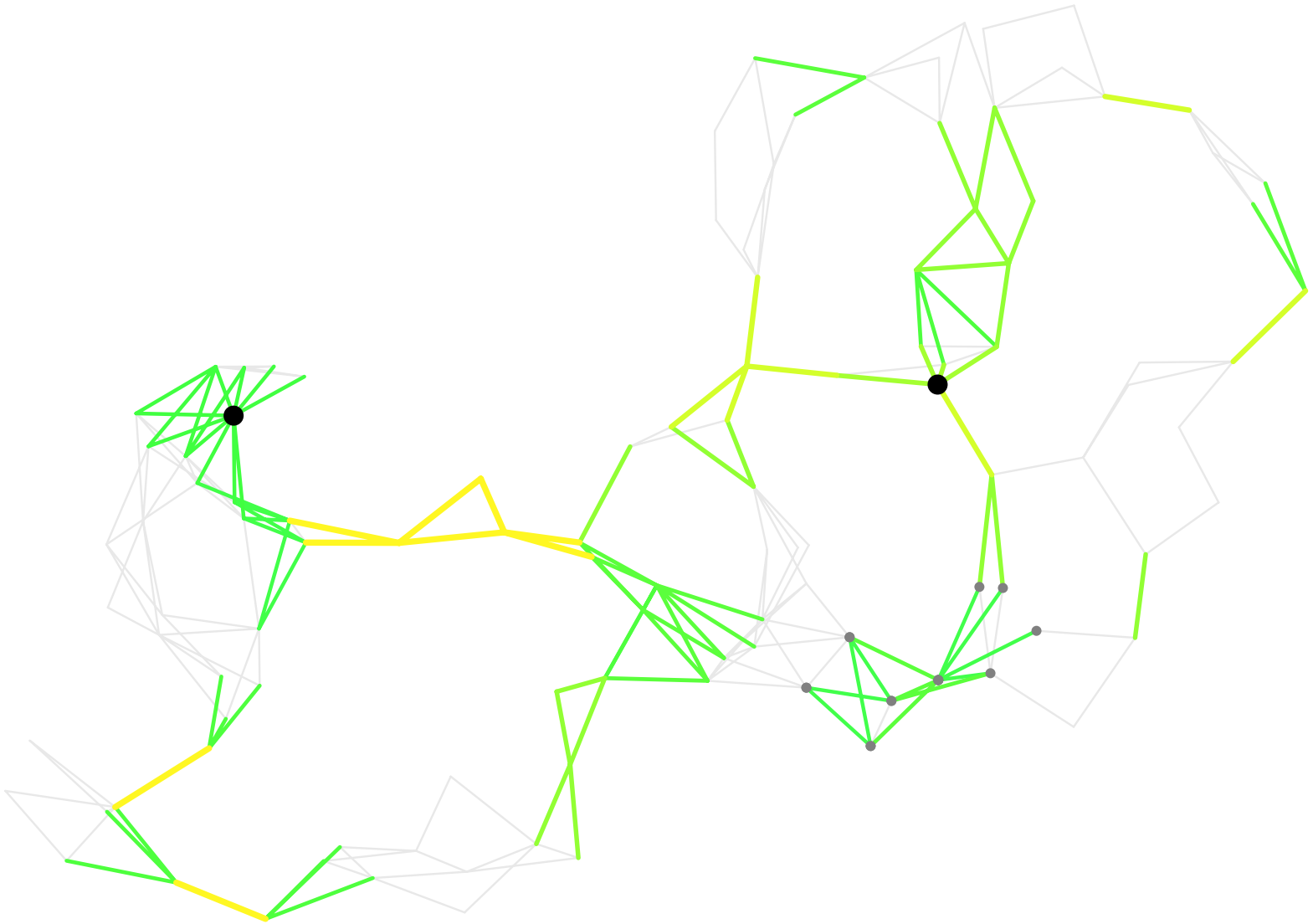
a network of 100 nodes and 728 links



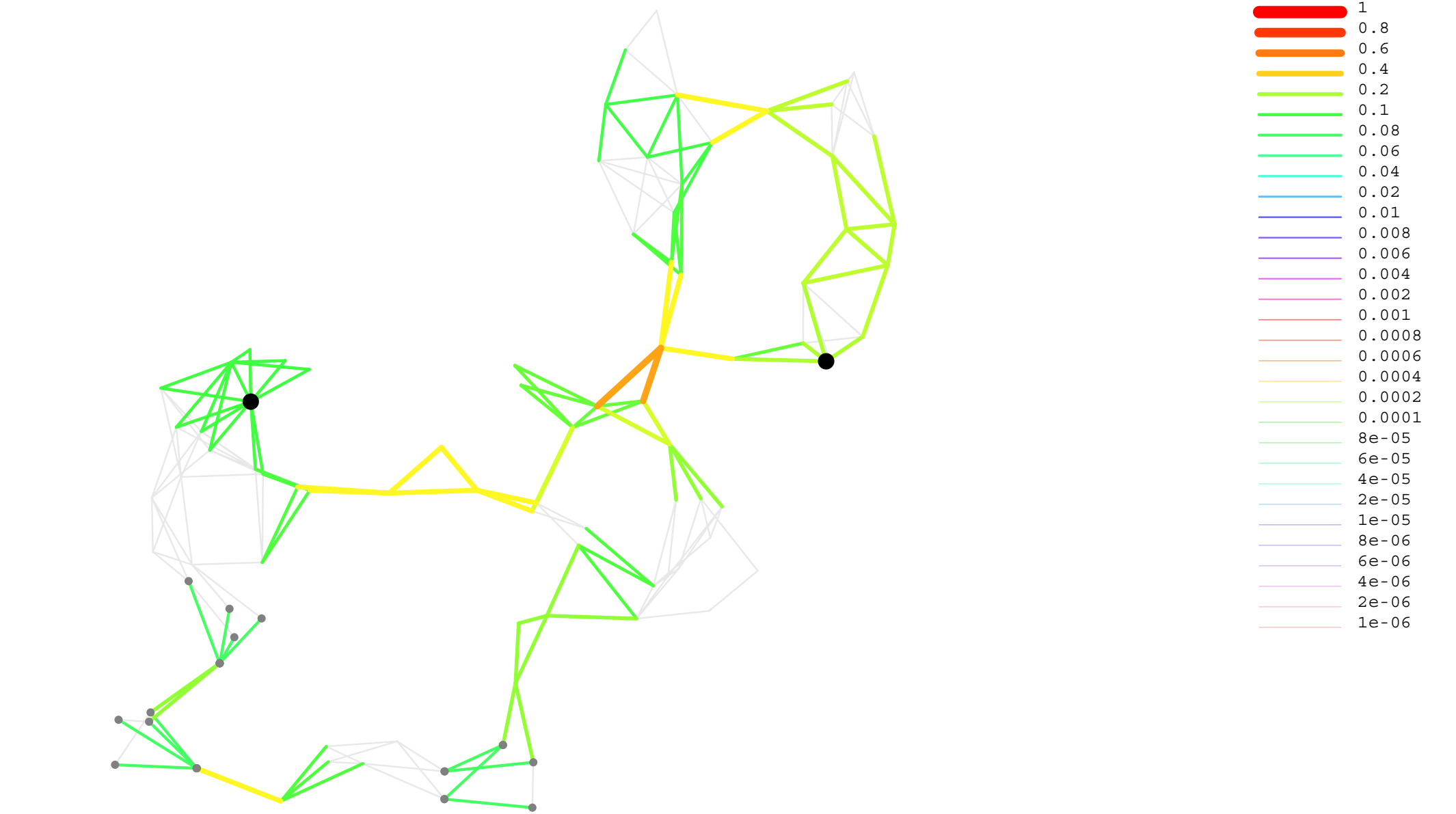
a network of 100 nodes and 736 links



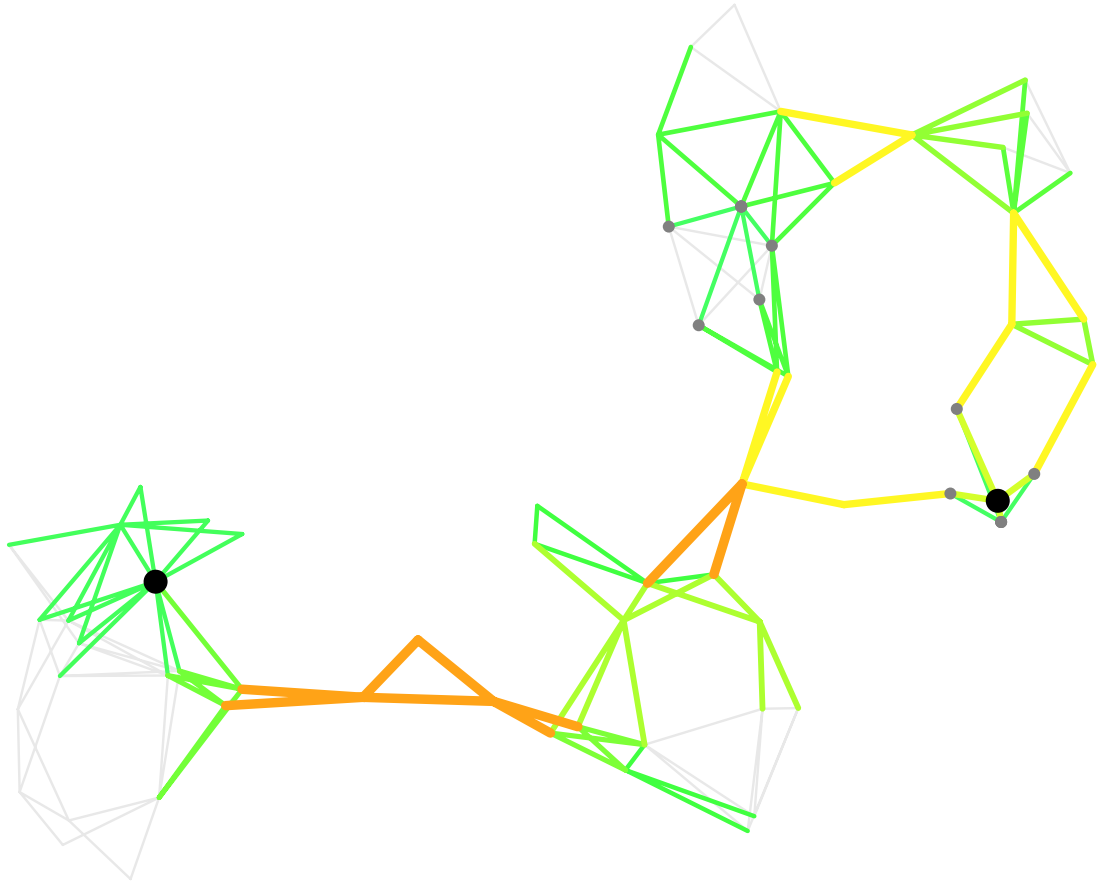
180 nodes, clock 1, layer 12, 17 bottlenecks at load 0.0666666667



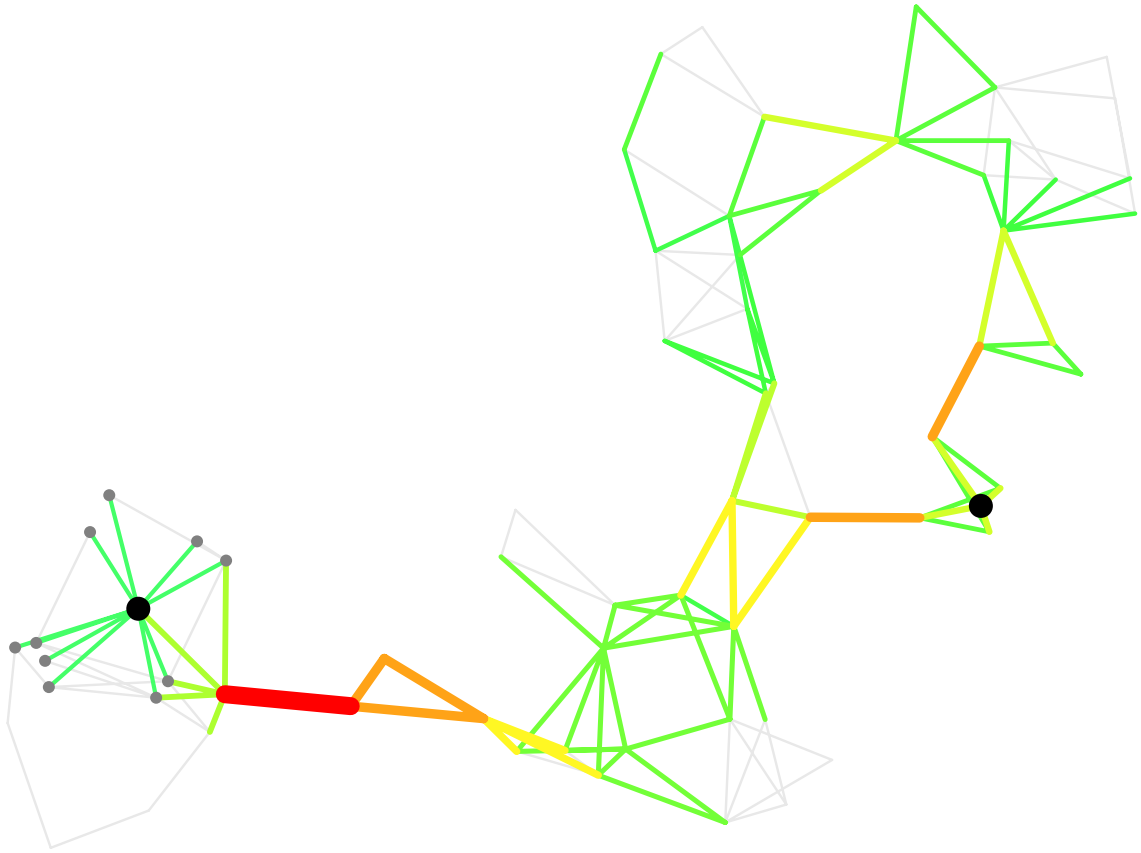
180 nodes, clock 2, layer 12, 8 bottlenecks at load 0.09375



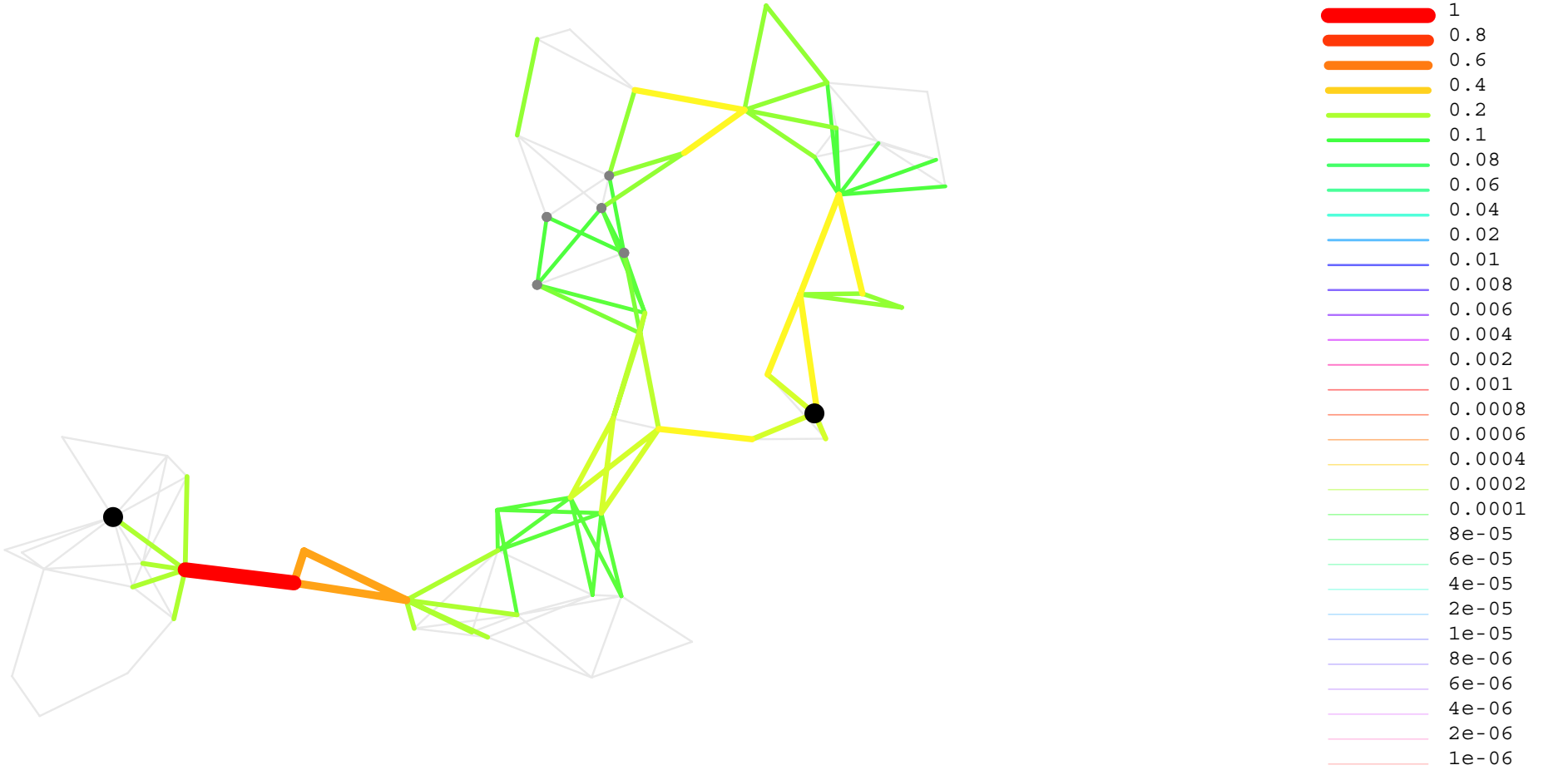
180 nodes, clock 3, layer 12, 12 bottlenecks at load 0.0833333333



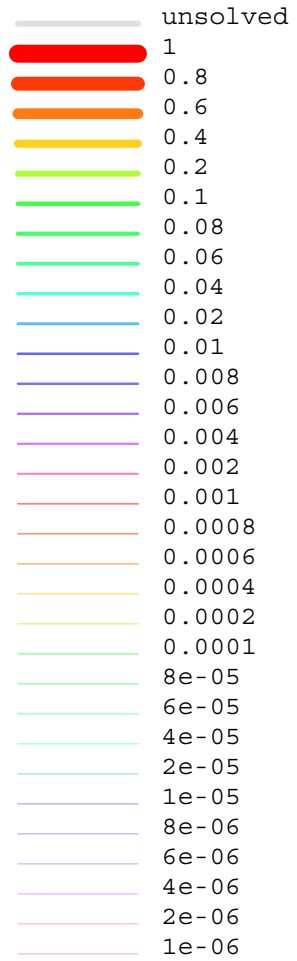
180 nodes, clock 4, layer 12, 7 bottlenecks at load 0.0833333333



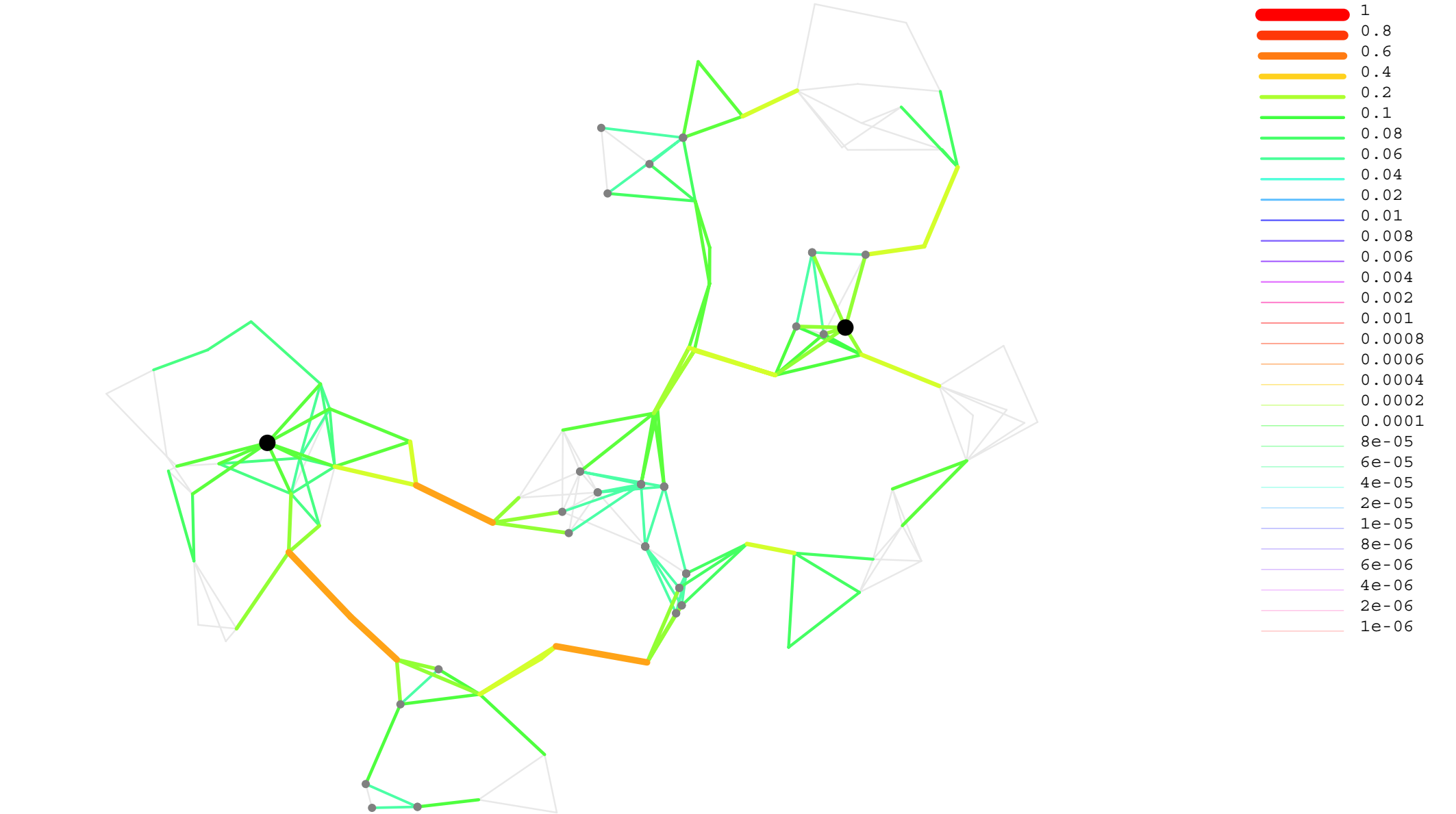
180 nodes, clock 5, layer 12, 10 bottlenecks at load 0.08



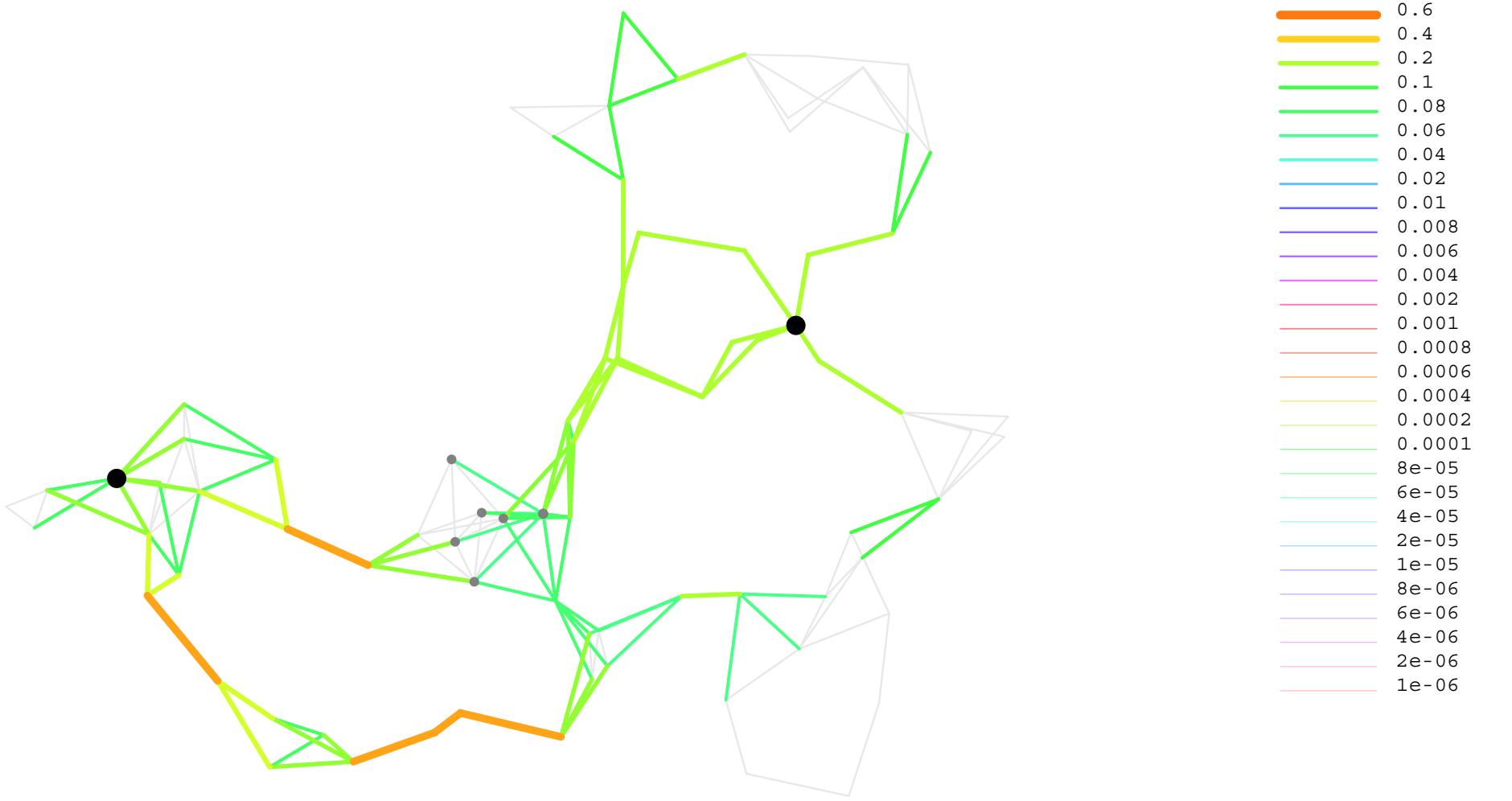
180 nodes, clock 6, layer 12, 5 bottlenecks at load 0.1086419753



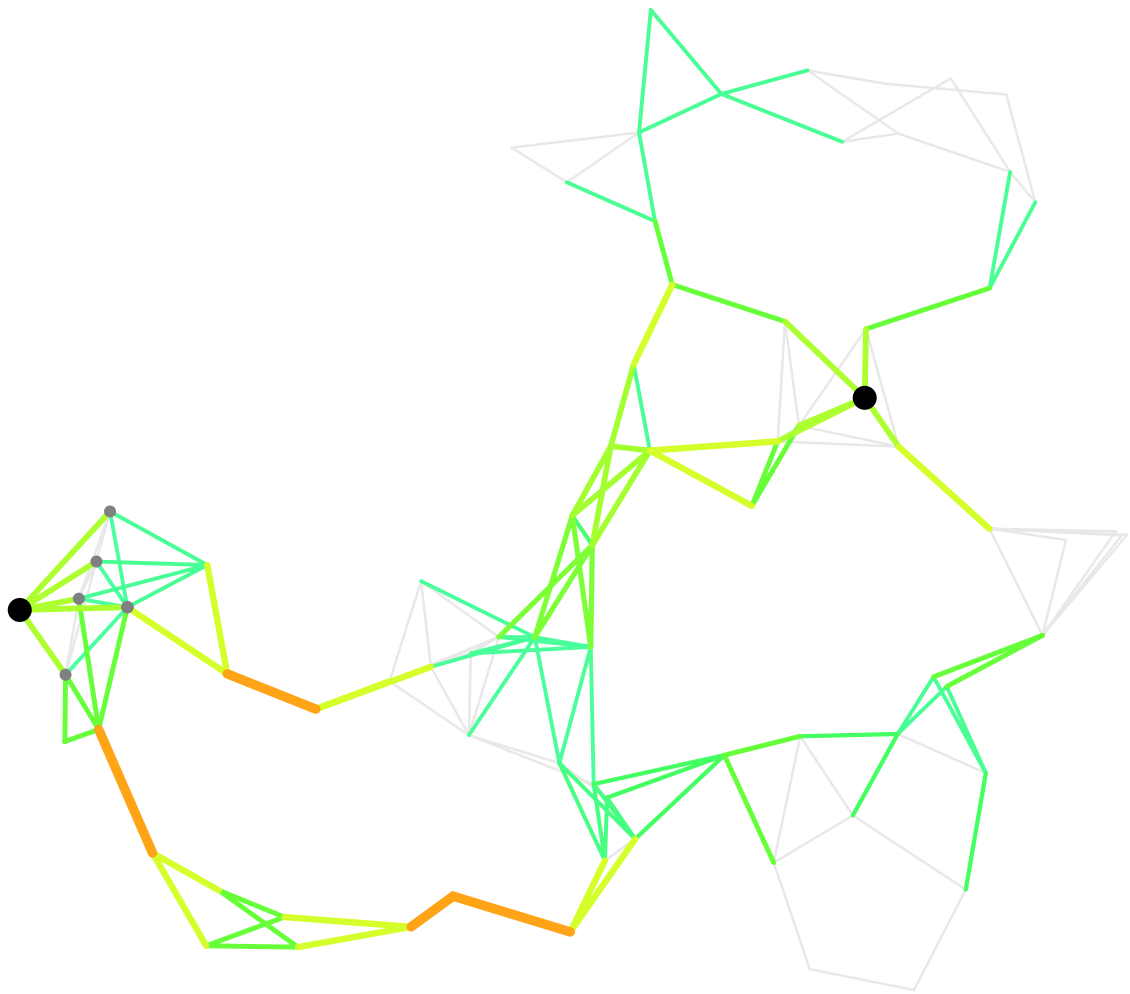
180 nodes, clock 7, layer 0, 0 bottlenecks at load N/A



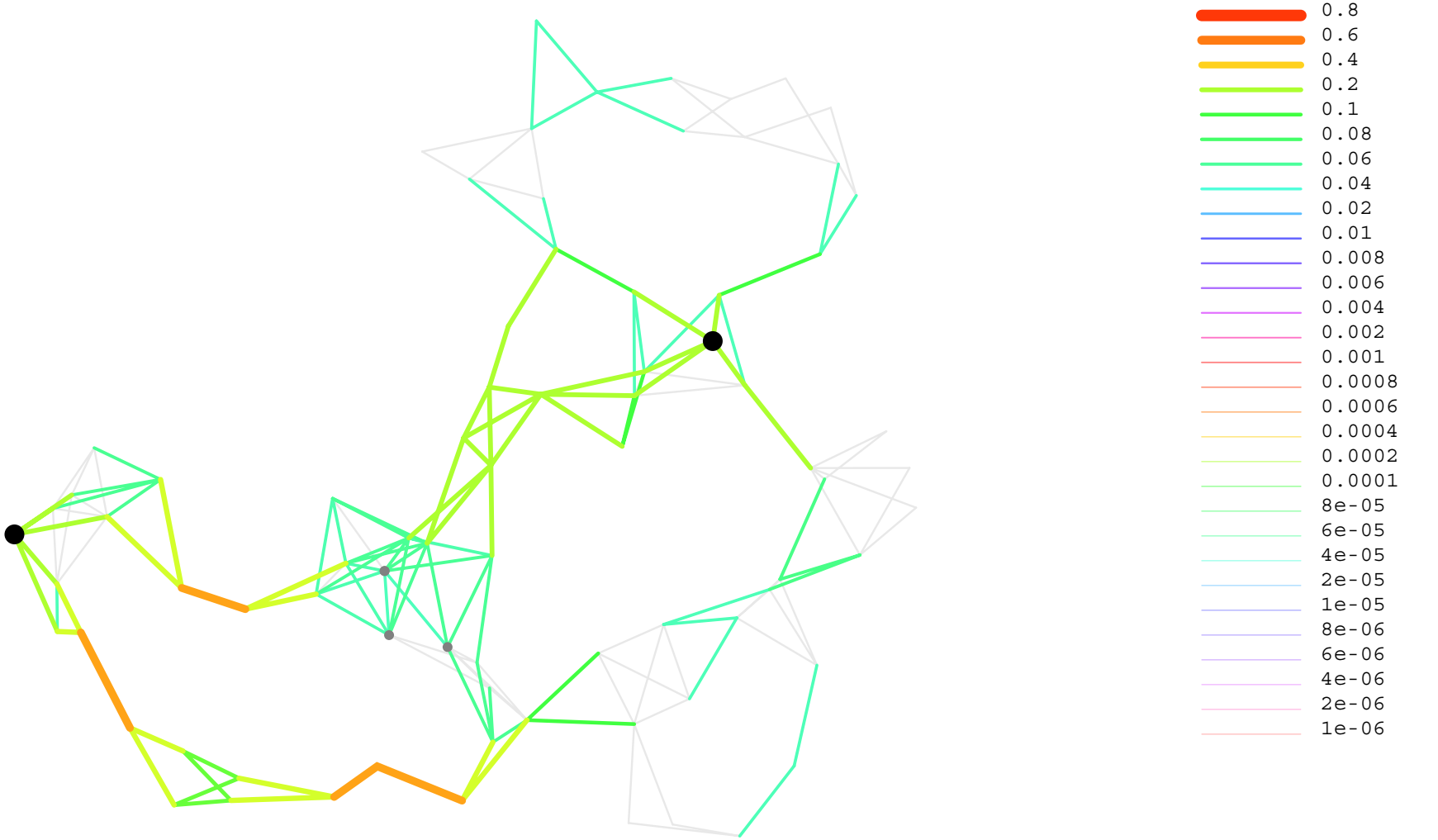
180 nodes, clock 8, layer 12, 24 bottlenecks at load 0.0555555556



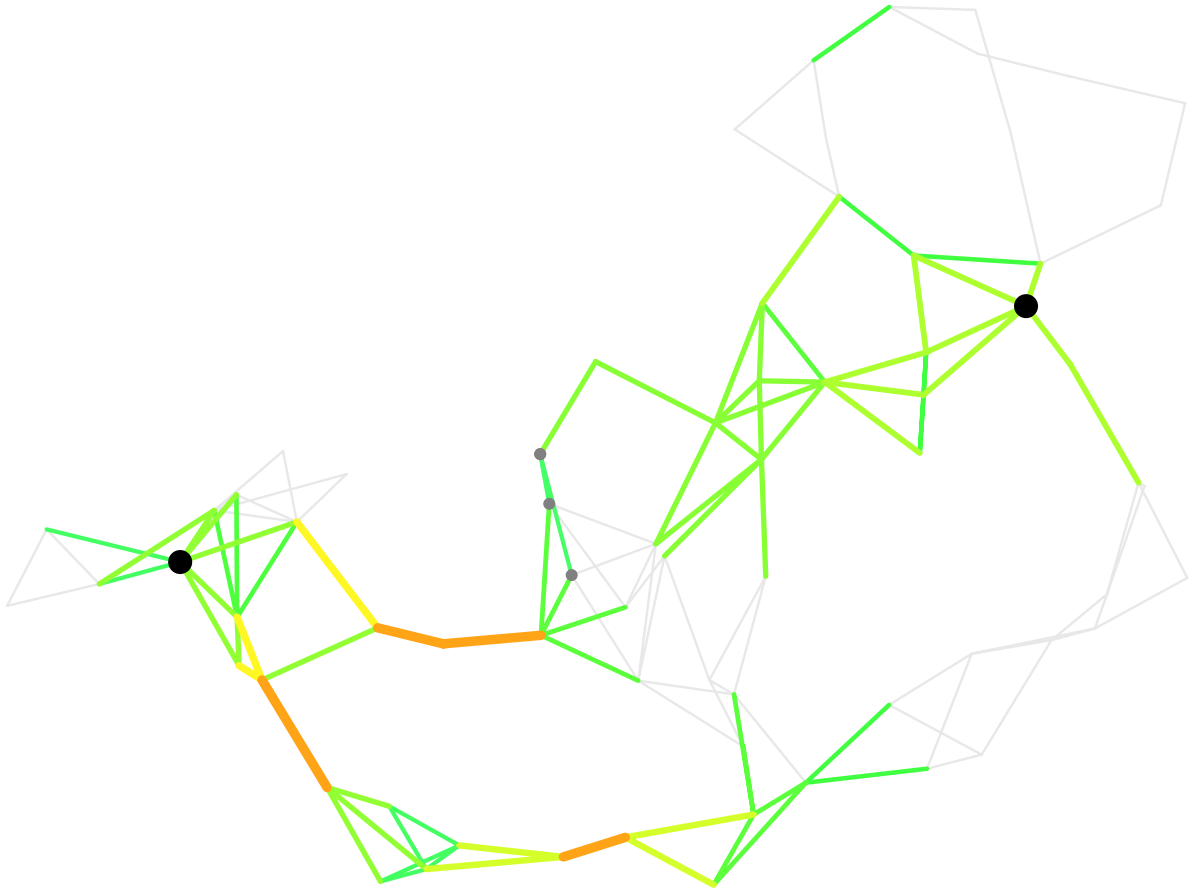
180 nodes, clock 9, layer 12, 5 bottlenecks at load 0.0653333333



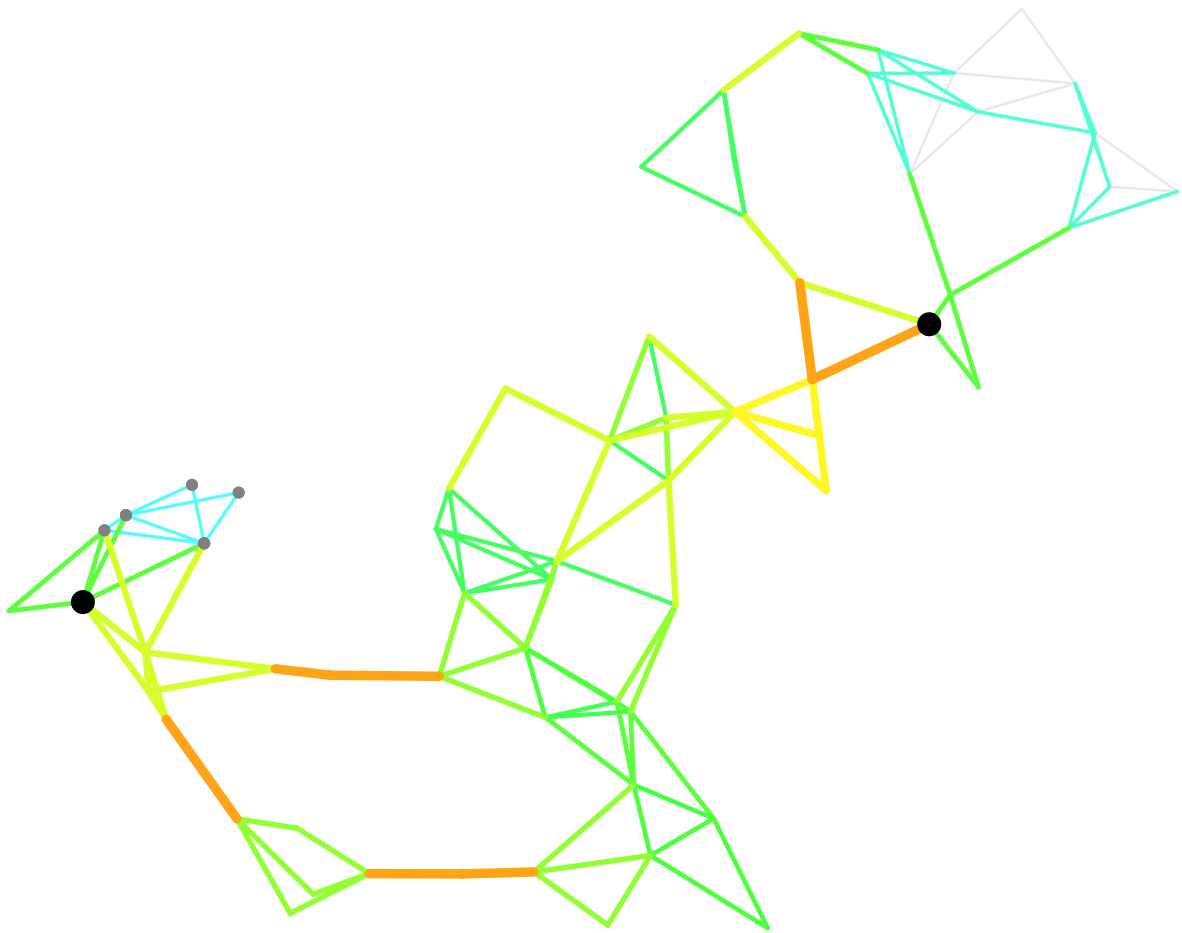
180 nodes, clock 10, layer 12, 4 bottlenecks at load 0.059375



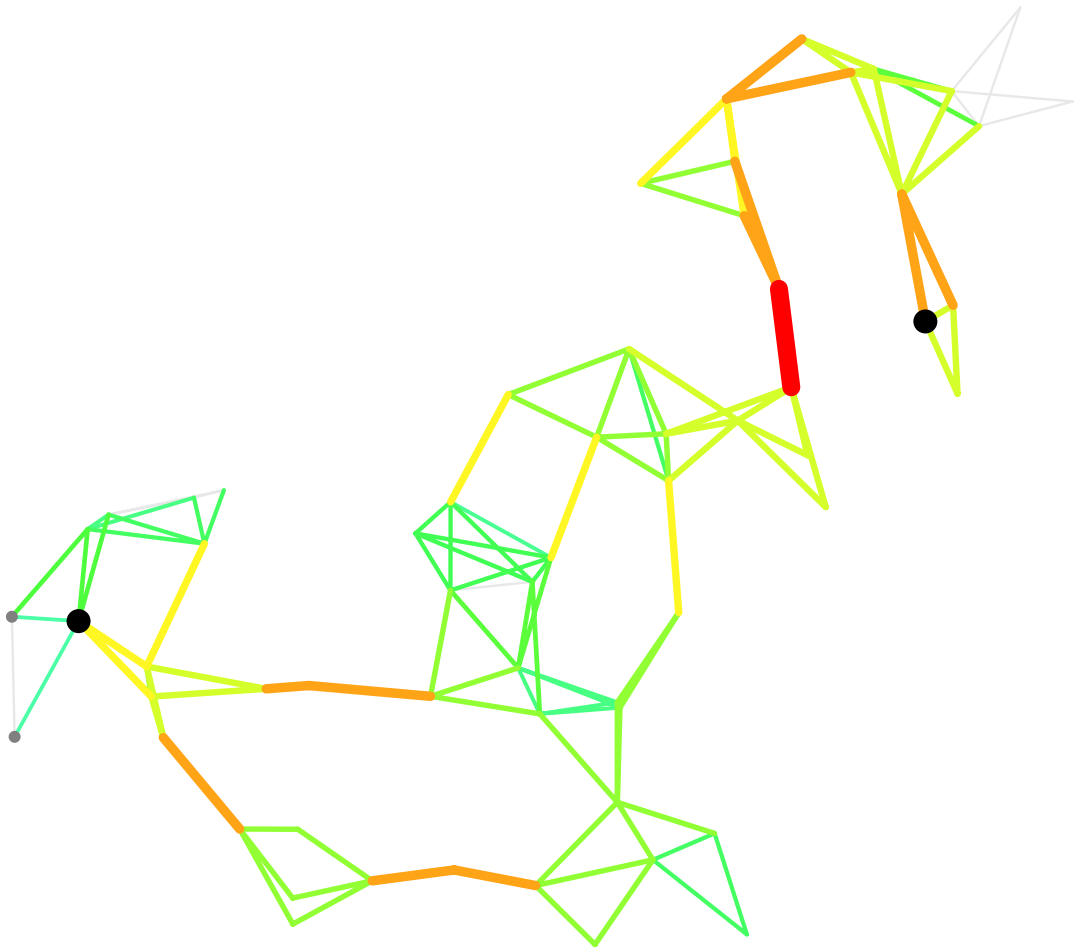
180 nodes, clock 11, layer 12, 2 bottlenecks at load 0.0487179487



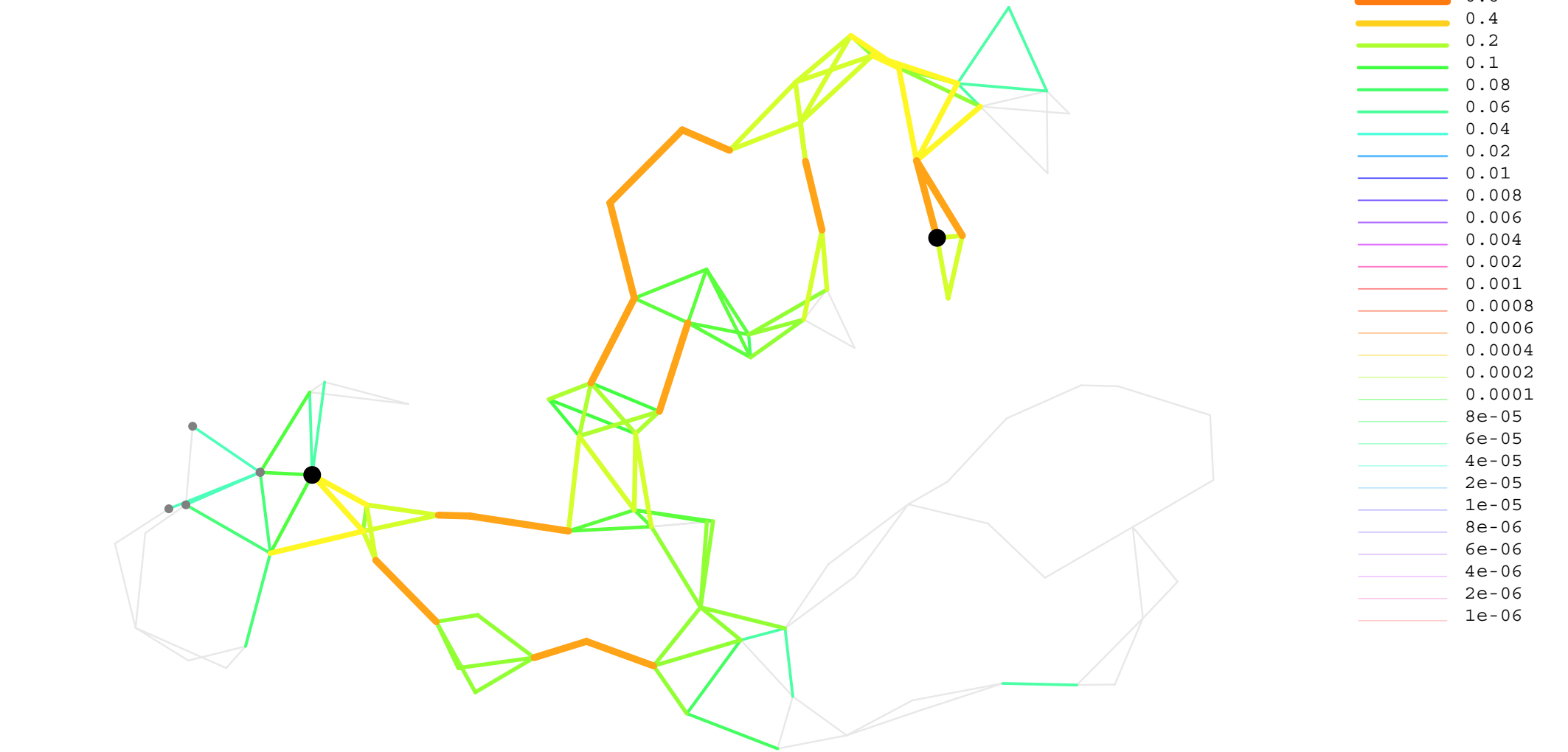
180 nodes, clock 12, layer 12, 2 bottlenecks at load 0.08



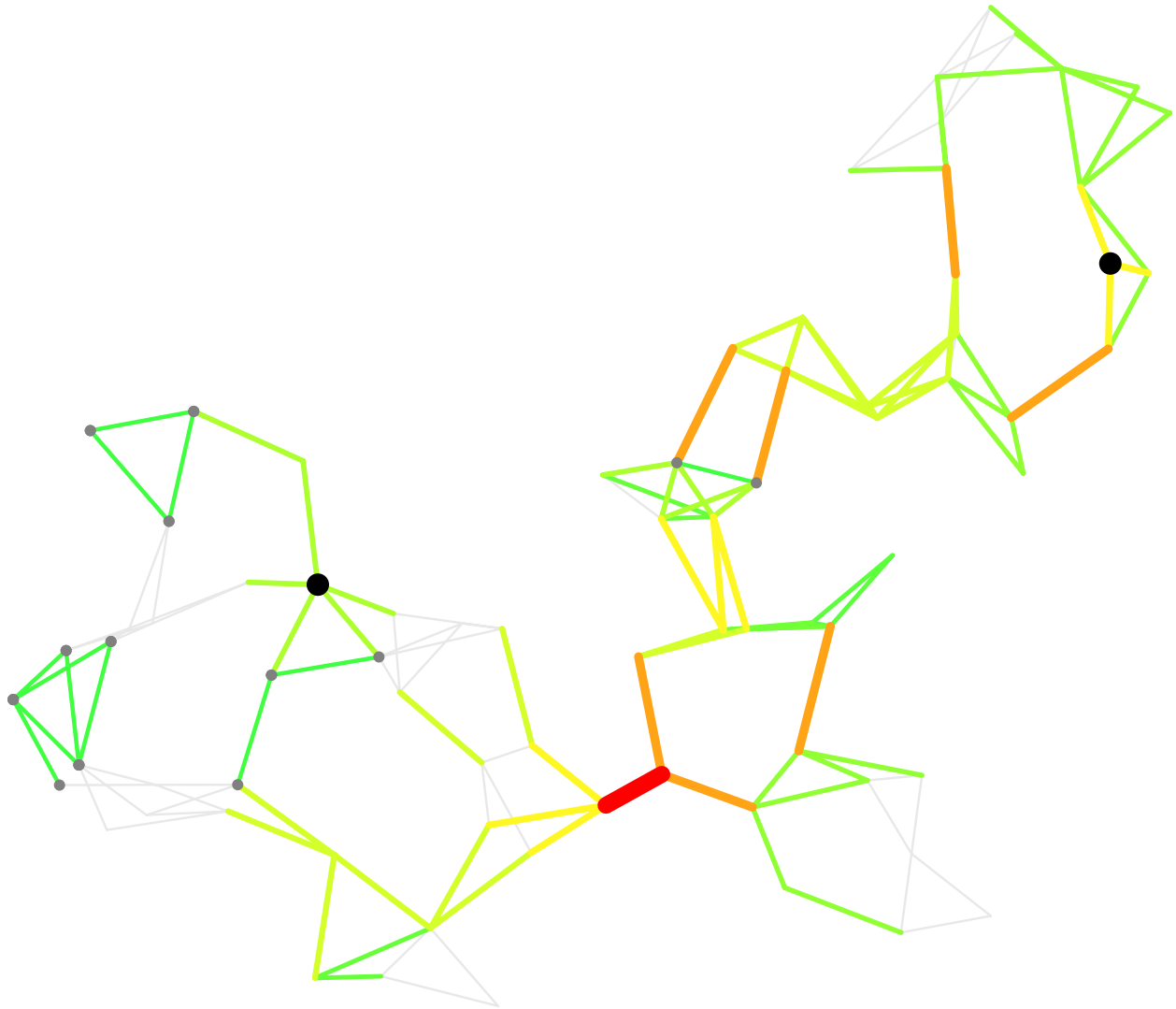
180 nodes, clock 13, layer 12, 7 bottlenecks at load 0.03125



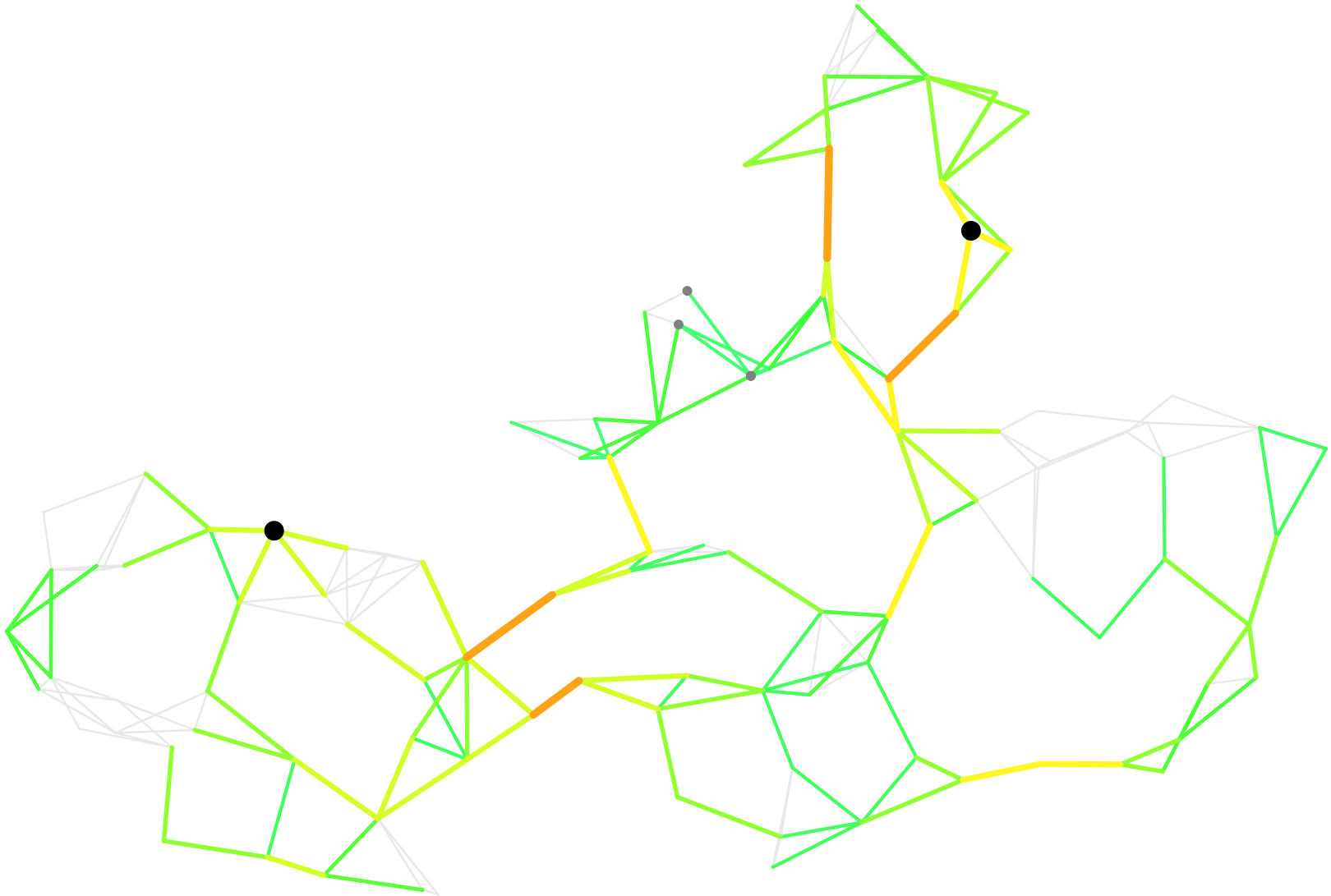
180 nodes, clock 14, layer 12, 2 bottlenecks at load 0.0555555556



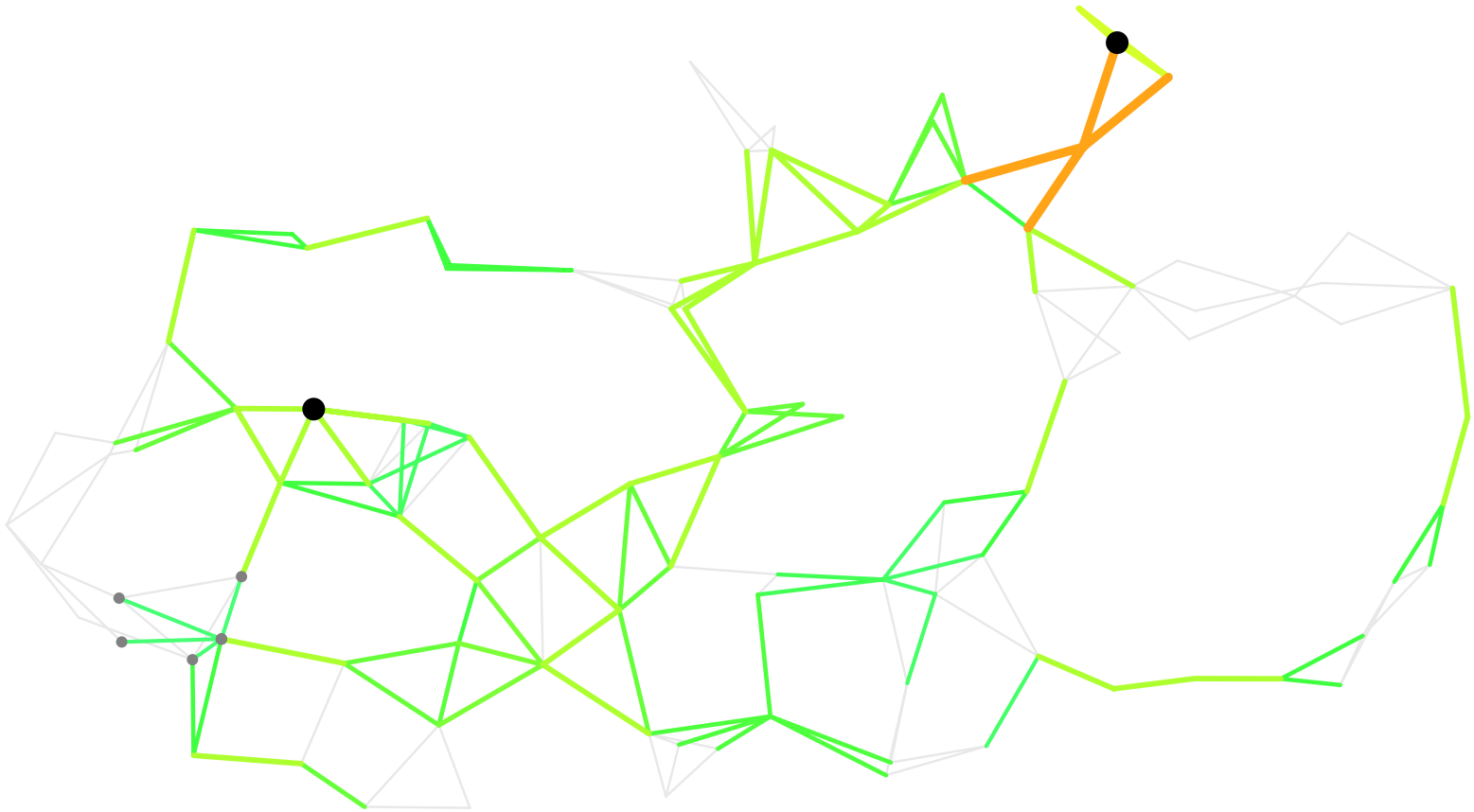
180 nodes, clock 15, layer 12, 3 bottlenecks at load 0.049382716



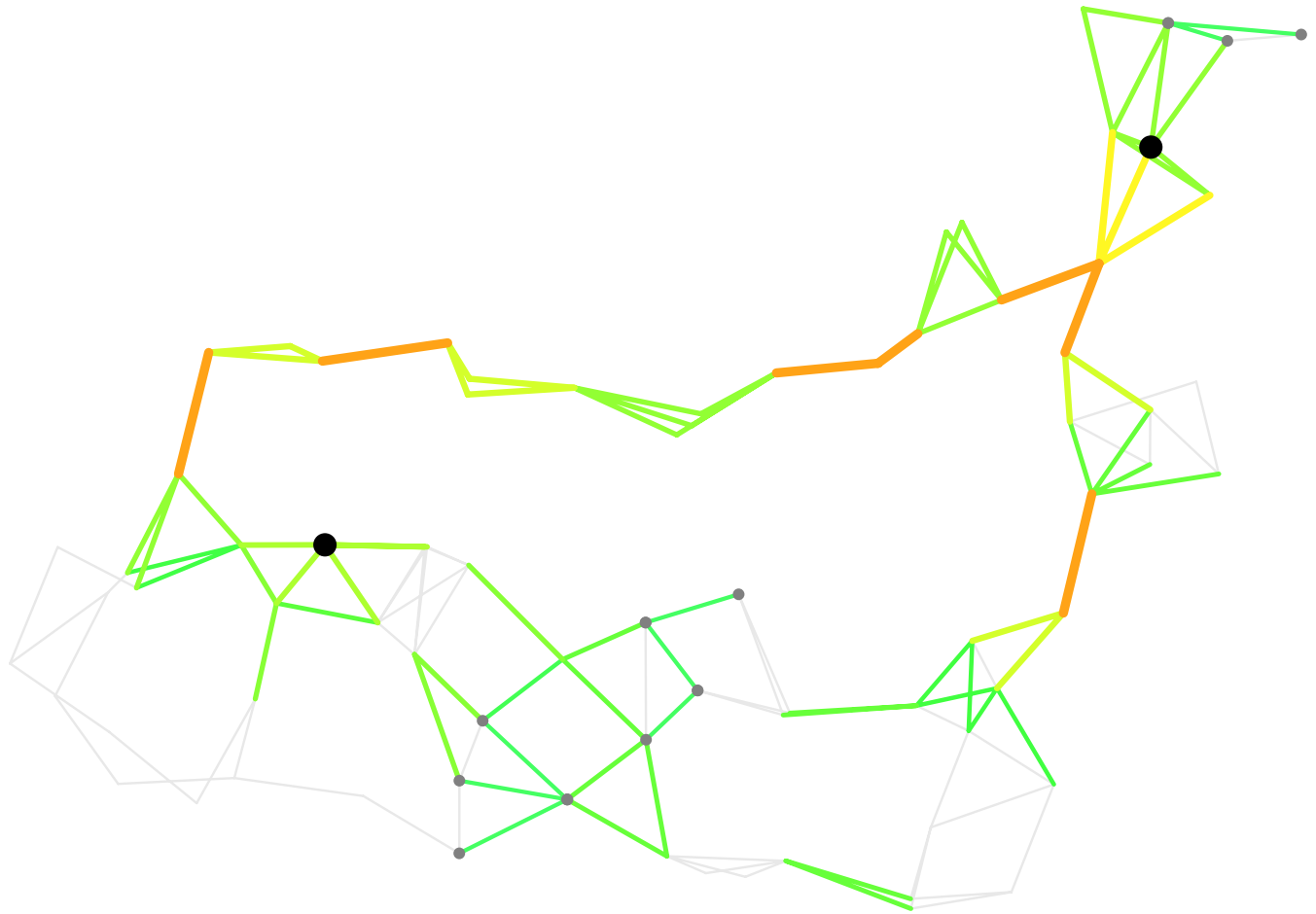
180 nodes, clock 16, layer 12, 12 bottlenecks at load 0.1



180 nodes, clock 17, layer 12, 2 bottlenecks at load 0.0729166667

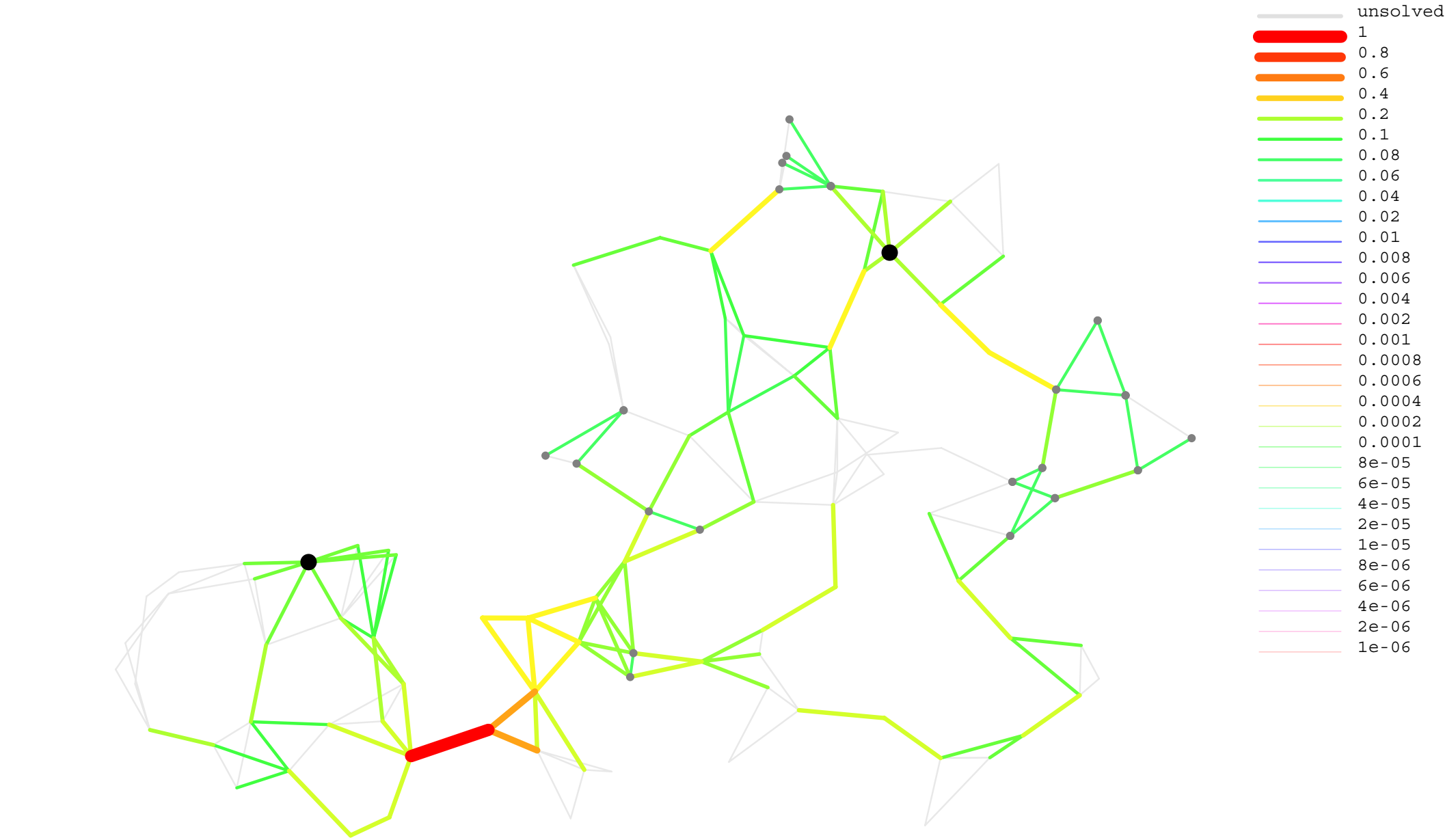


180 nodes, clock 18, layer 12, 4 bottlenecks at load 0.075

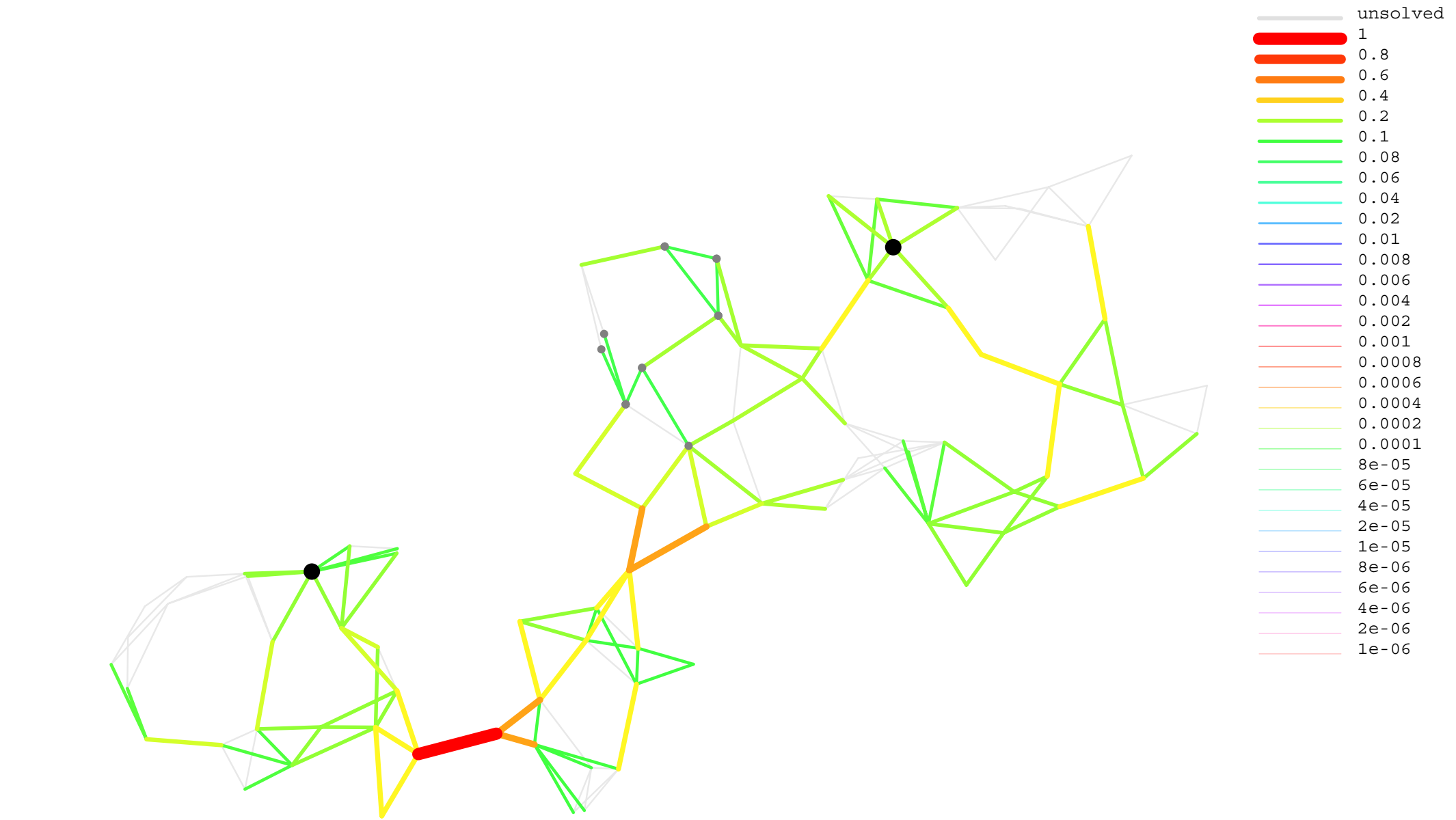


180 nodes, clock 19, layer 12, 8 bottlenecks at load 0.0833333333

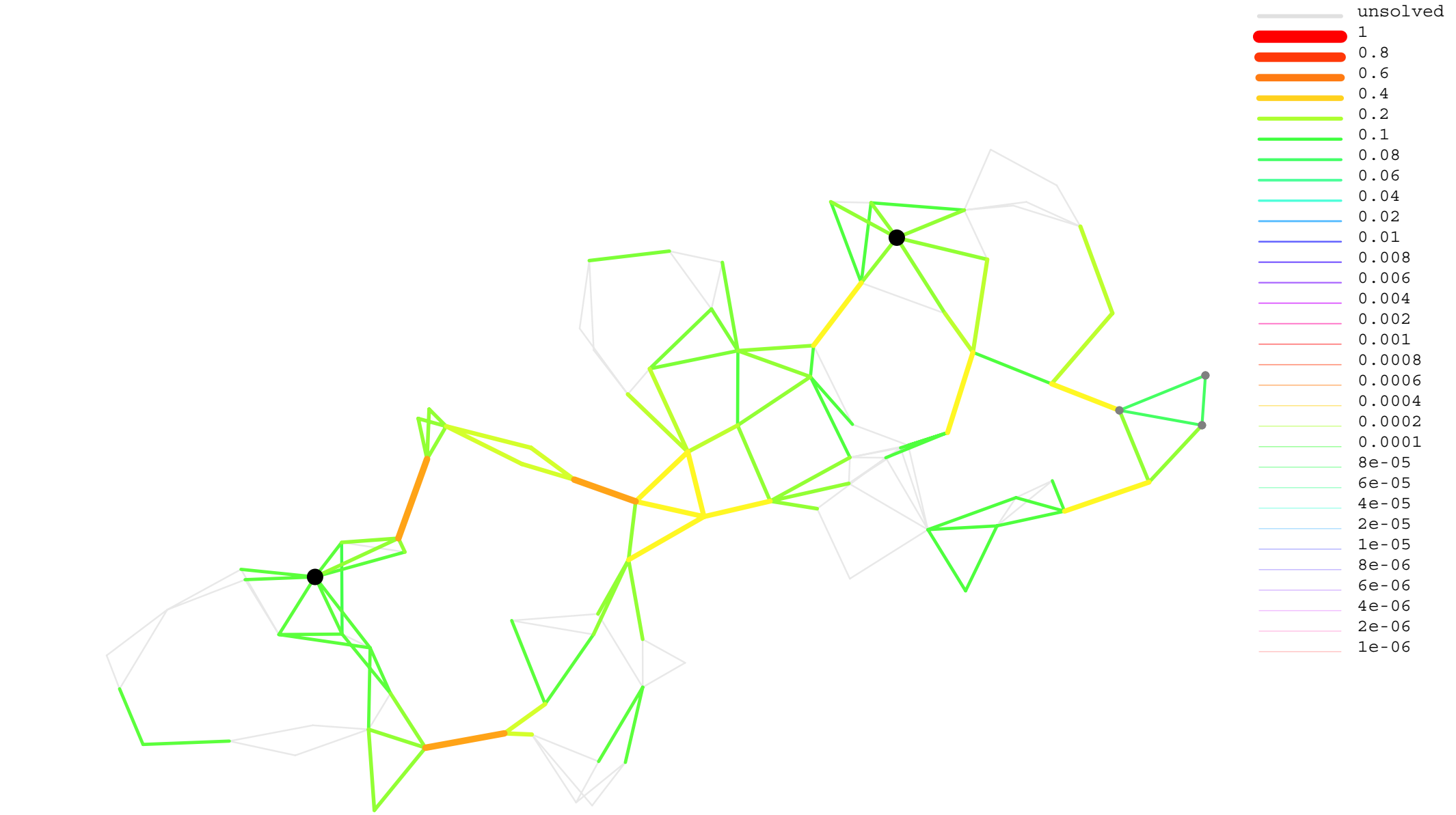
180 nodes, clock 20, layer 12, 5 bottlenecks at load 0.0916666667



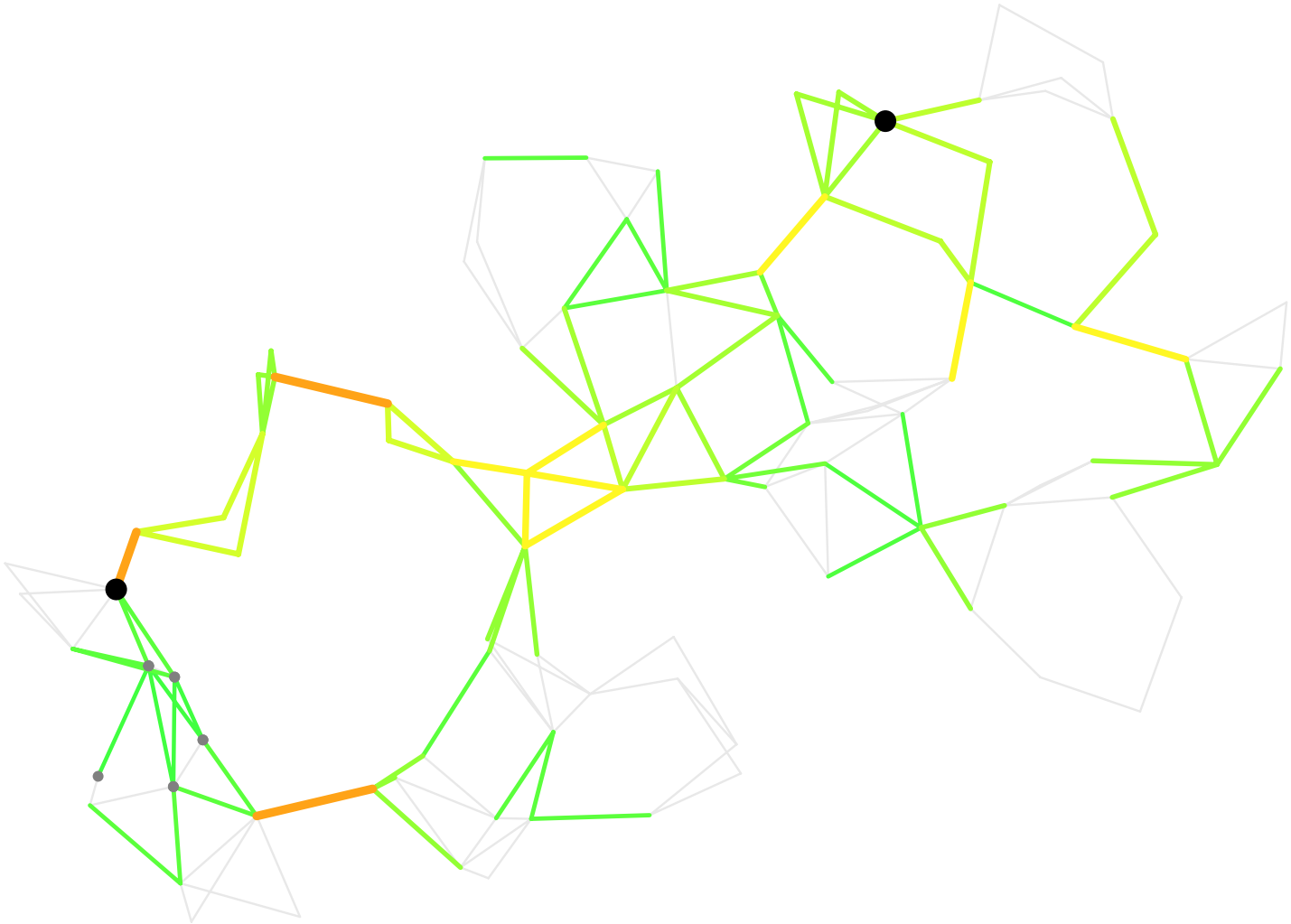
180 nodes, clock 21, layer 12, 17 bottlenecks at load 0.0833333333



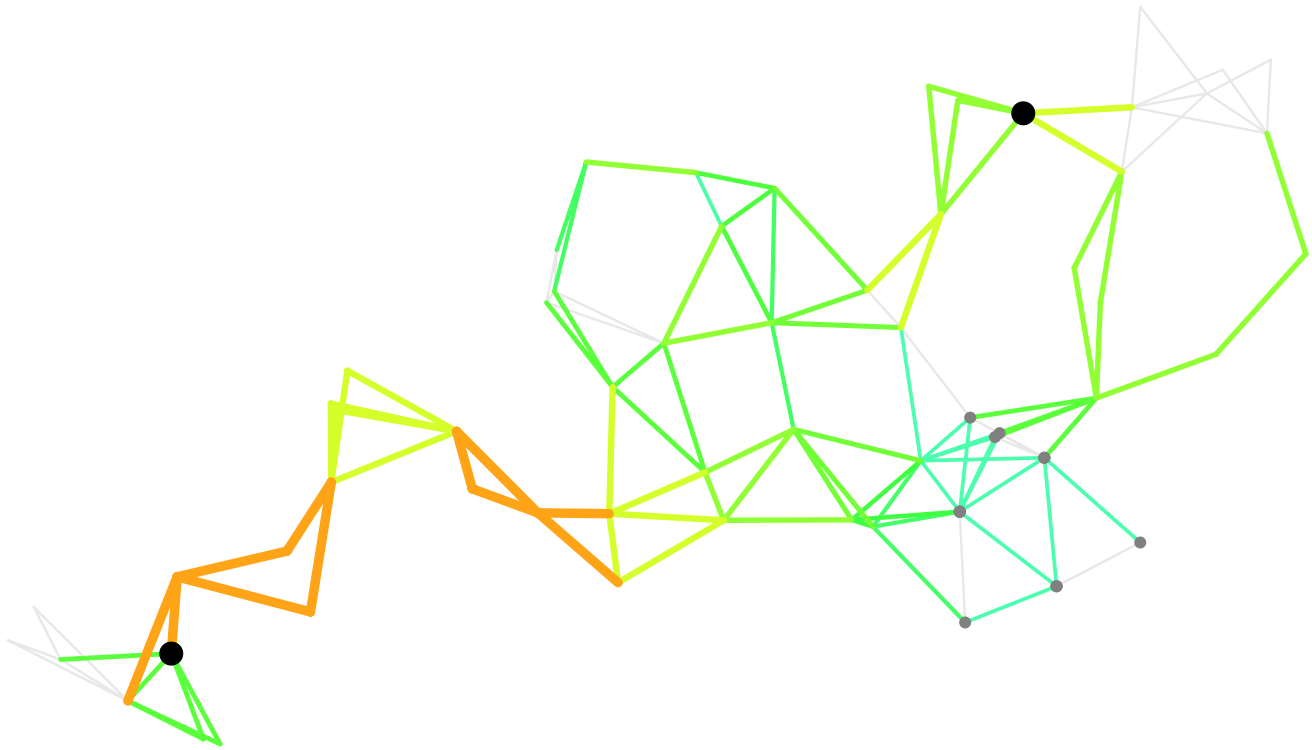
180 nodes, clock 22, layer 12, 7 bottlenecks at load 0.09375



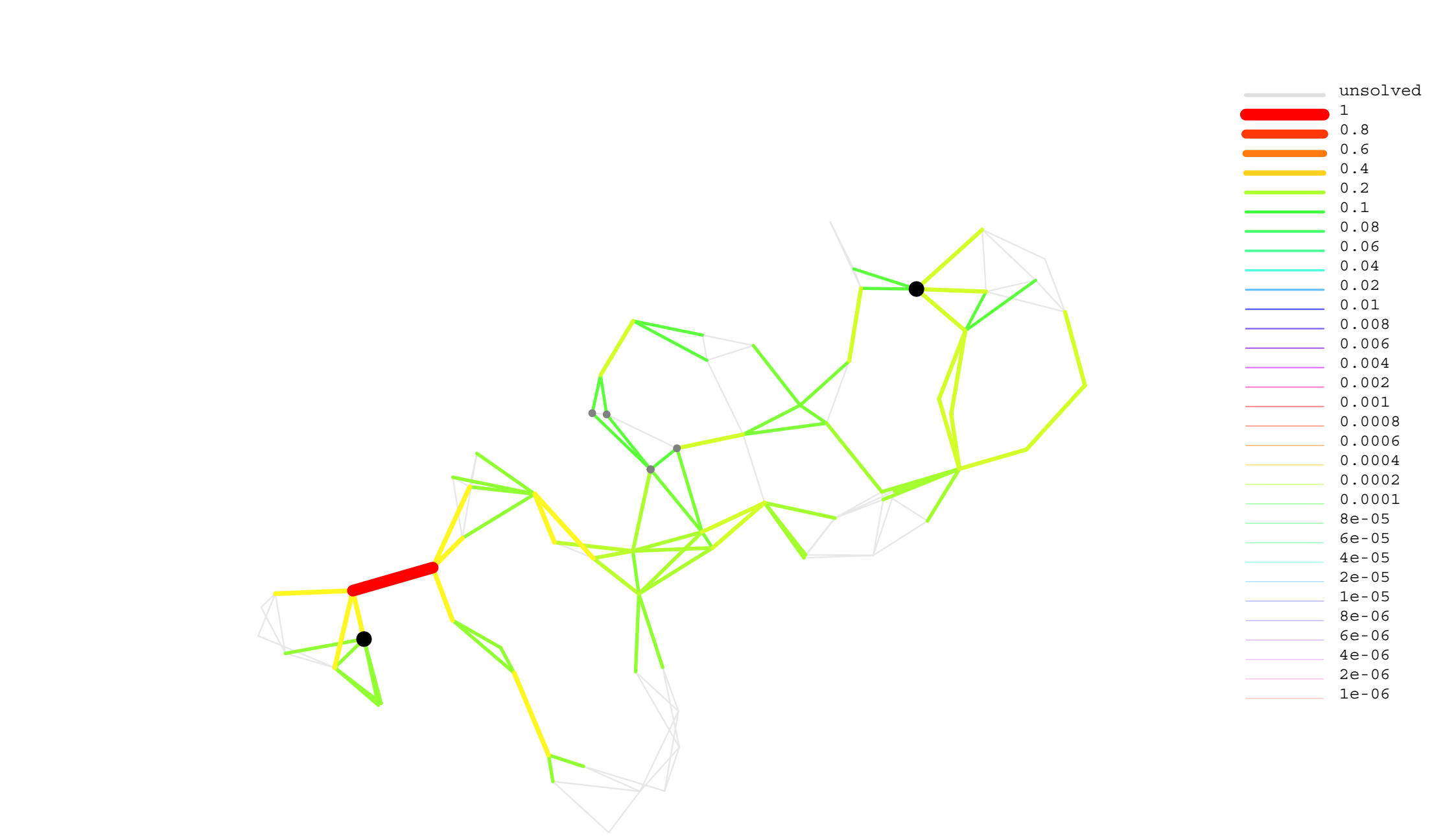
180 nodes, clock 23, layer 12, 3 bottlenecks at load 0.0833333333



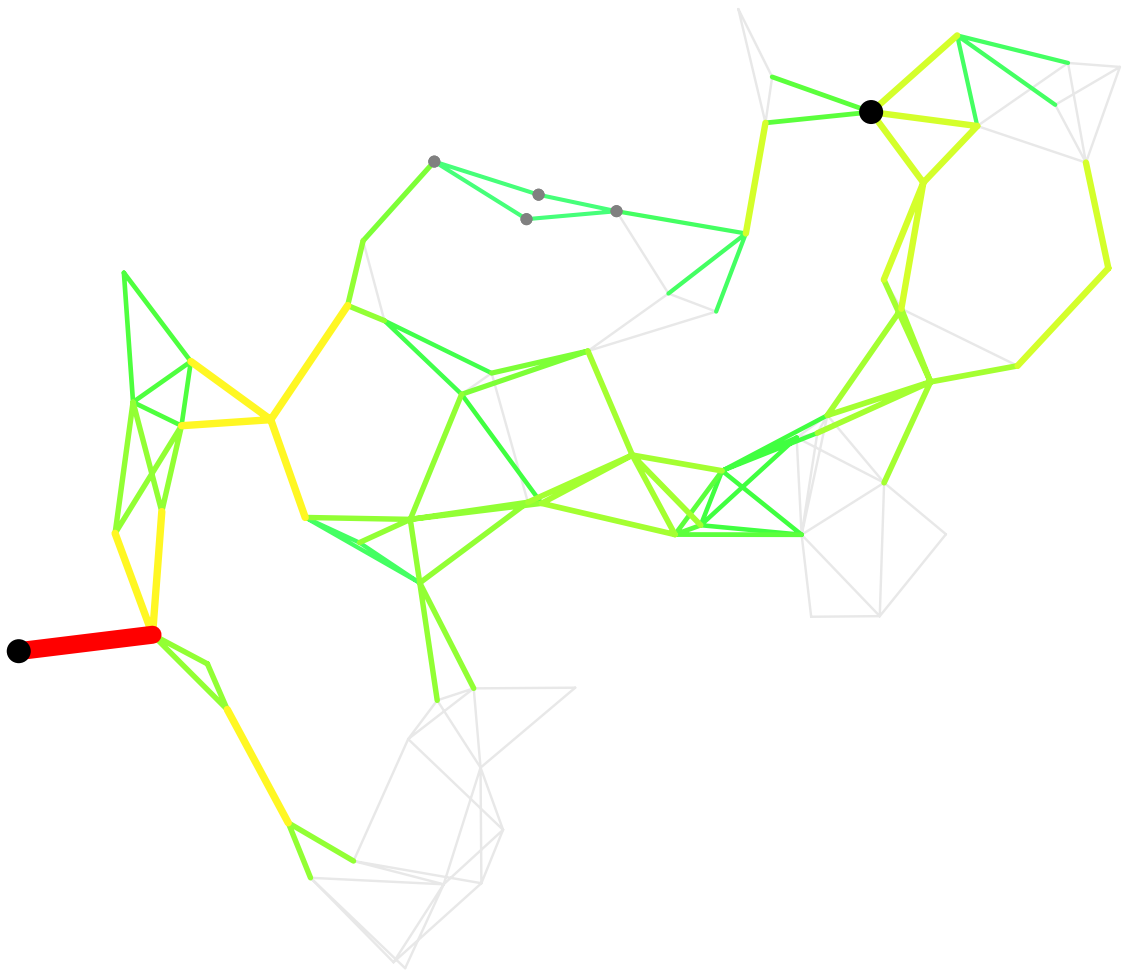
180 nodes, clock 24, layer 12, 5 bottlenecks at load 0.1



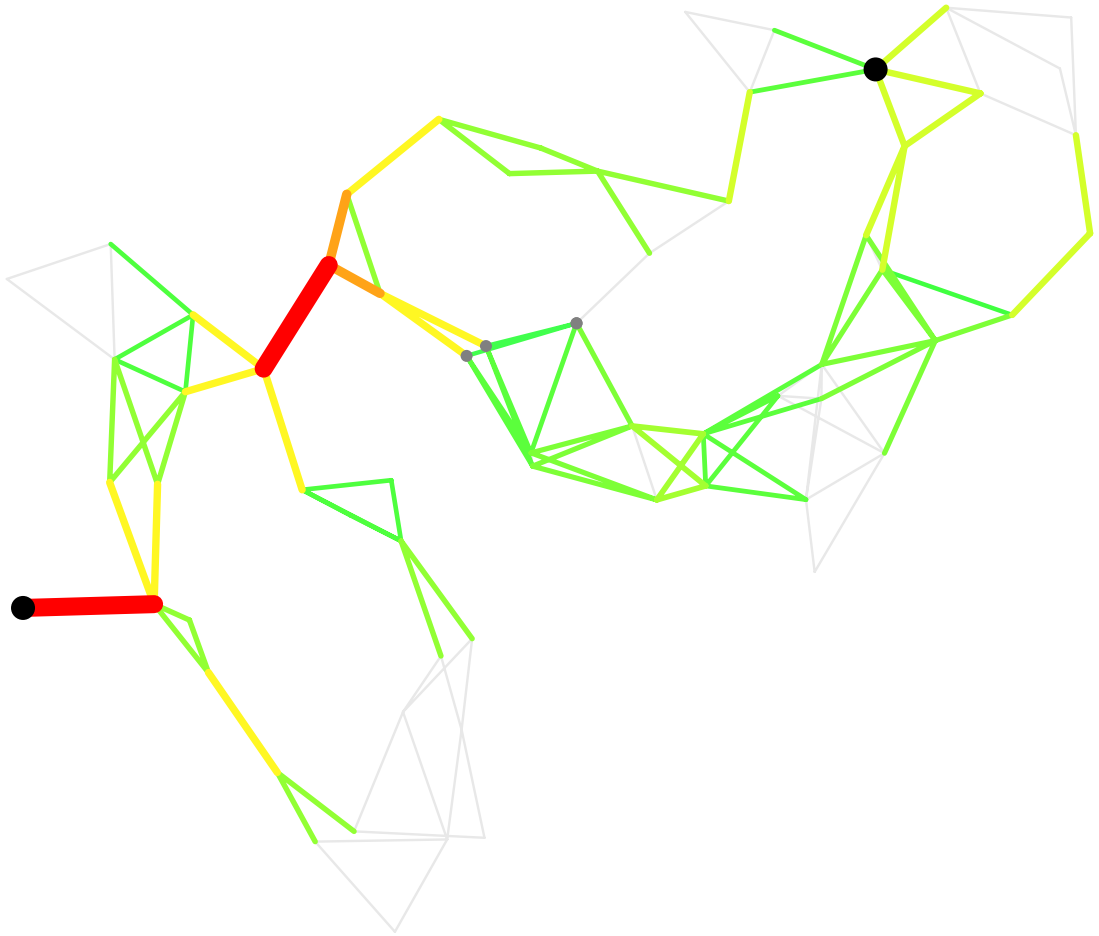
180 nodes, clock 25, layer 12, 8 bottlenecks at load 0.0526406036



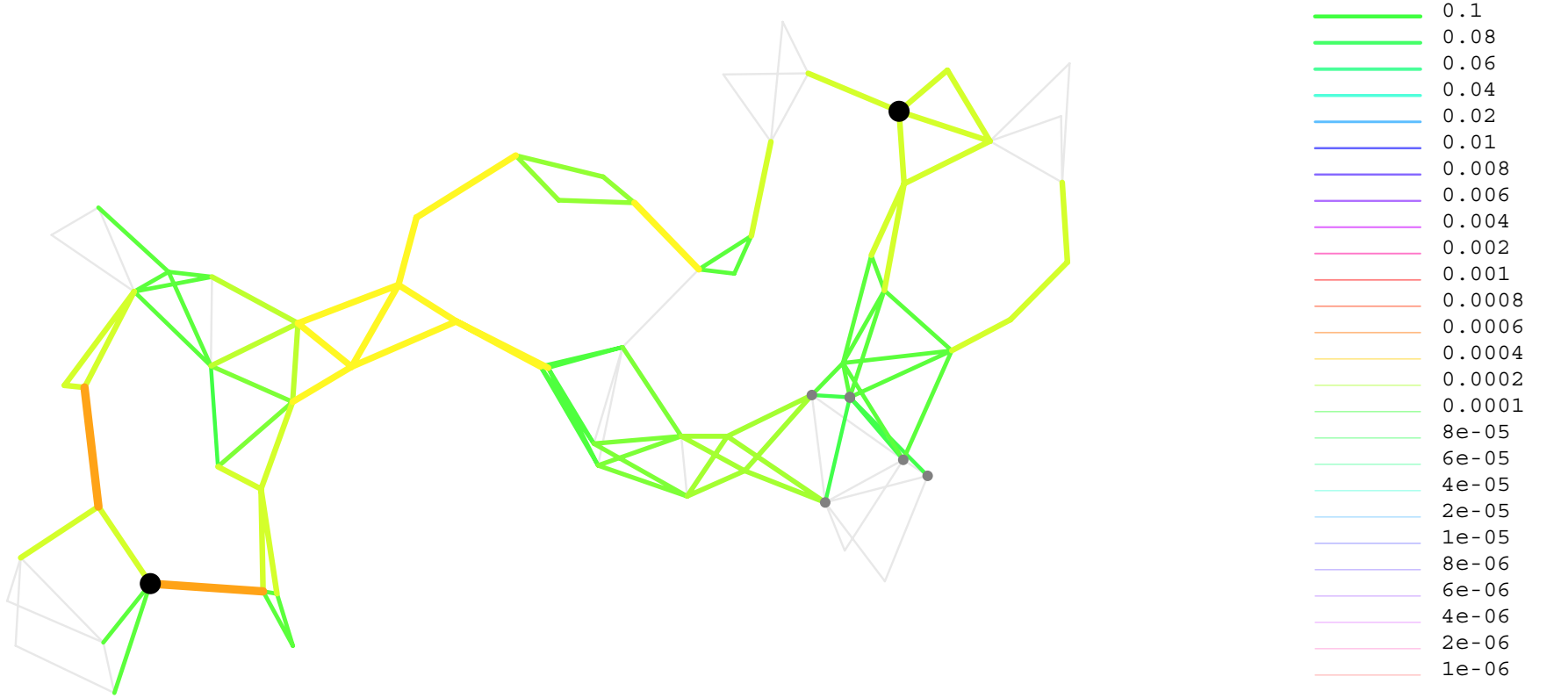
180 nodes, clock 26, layer 12, 3 bottlenecks at load 0.1166666667



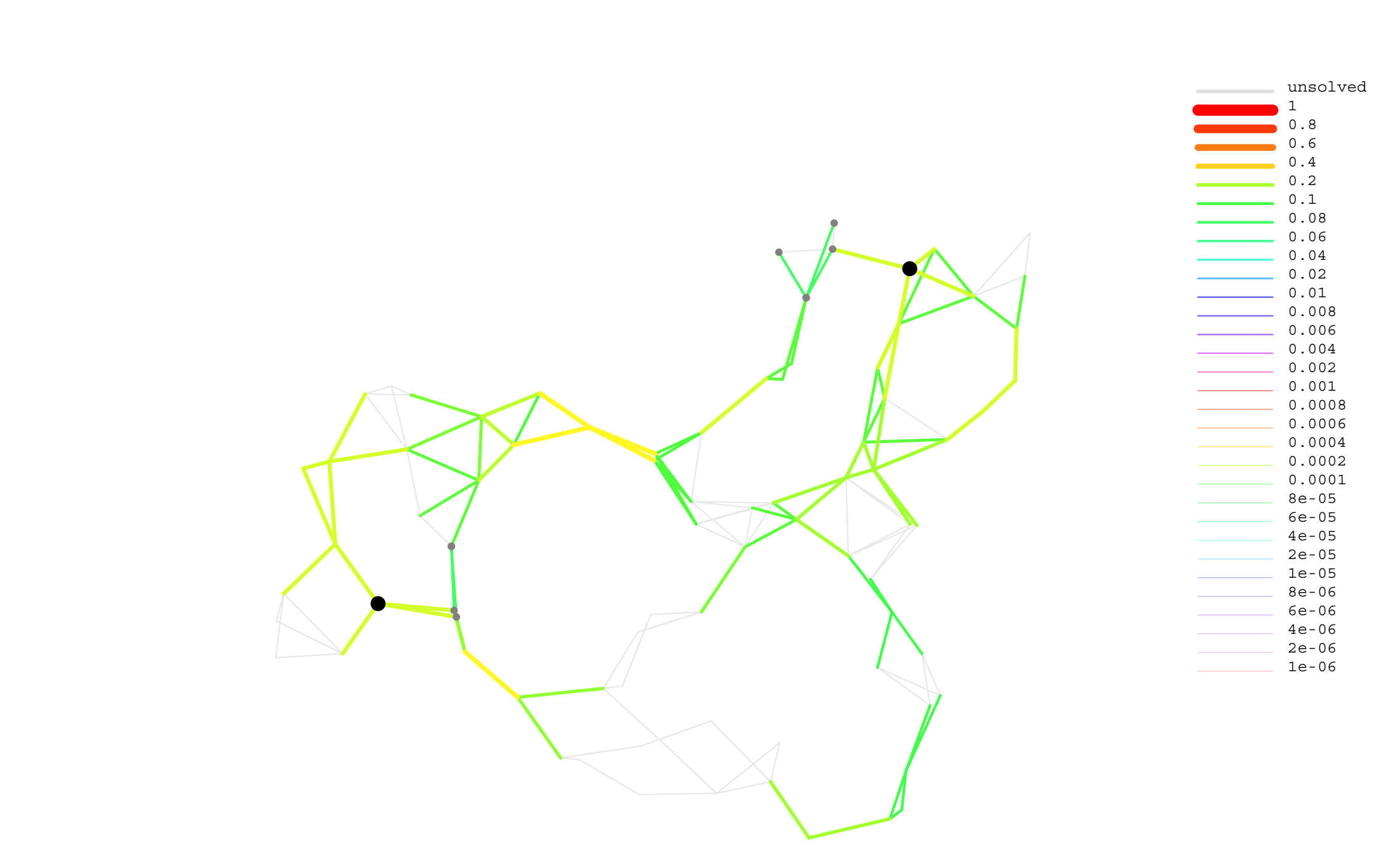
180 nodes, clock 27, layer 12, 4 bottlenecks at load 0.0729166667



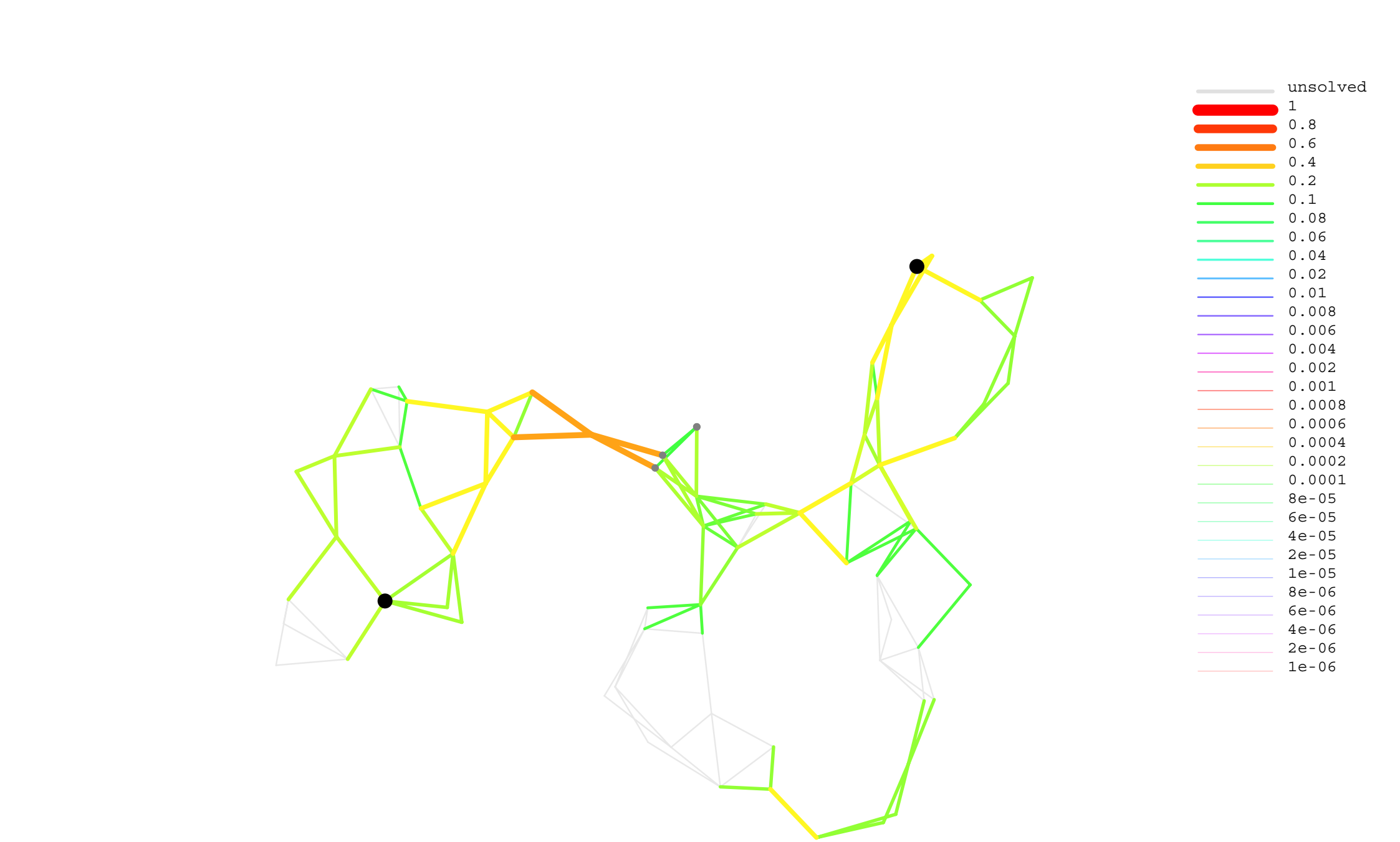
180 nodes, clock 28, layer 12, 2 bottlenecks at load 0.0933333333



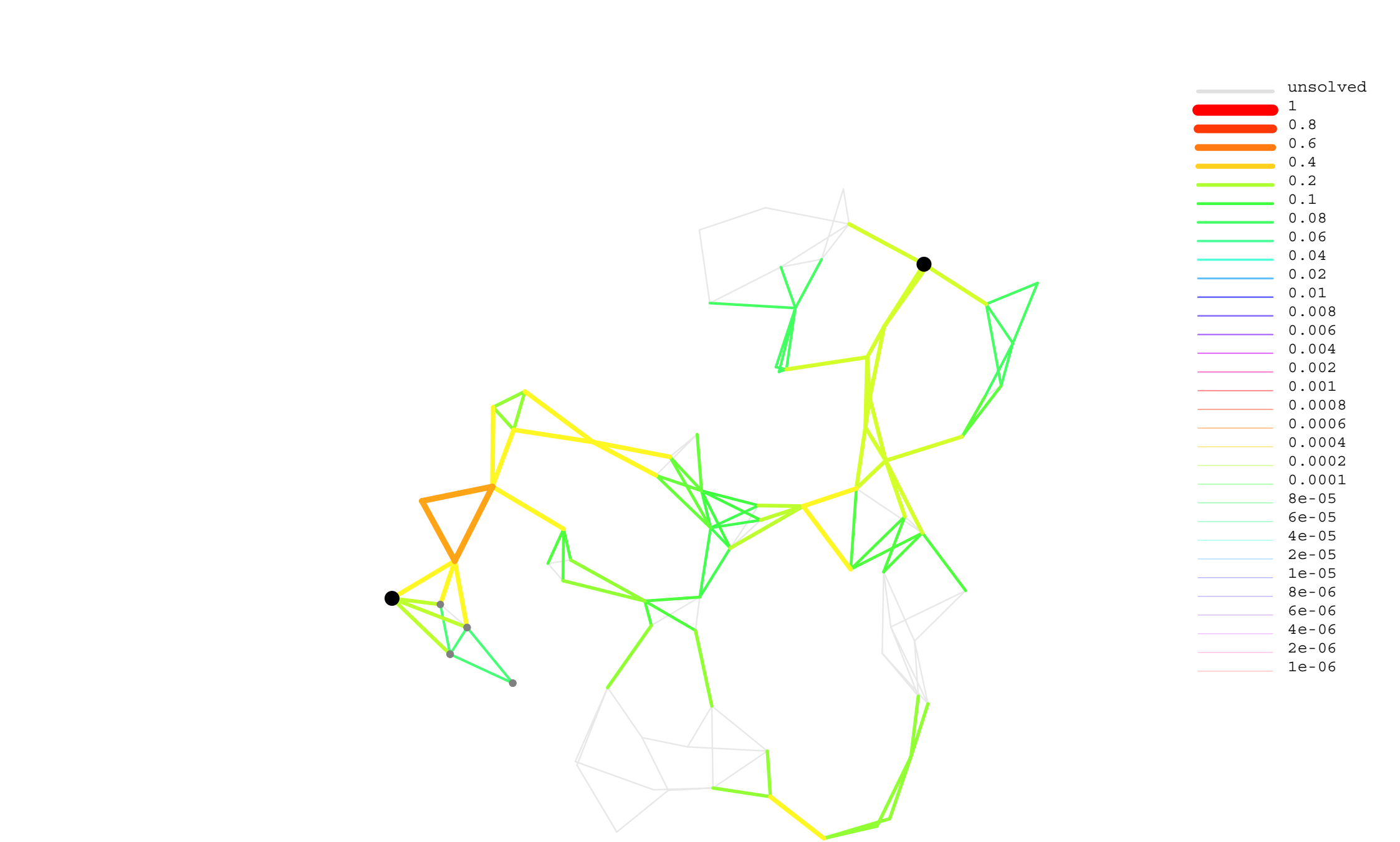
180 nodes, clock 29, layer 12, 4 bottlenecks at load 0.09375



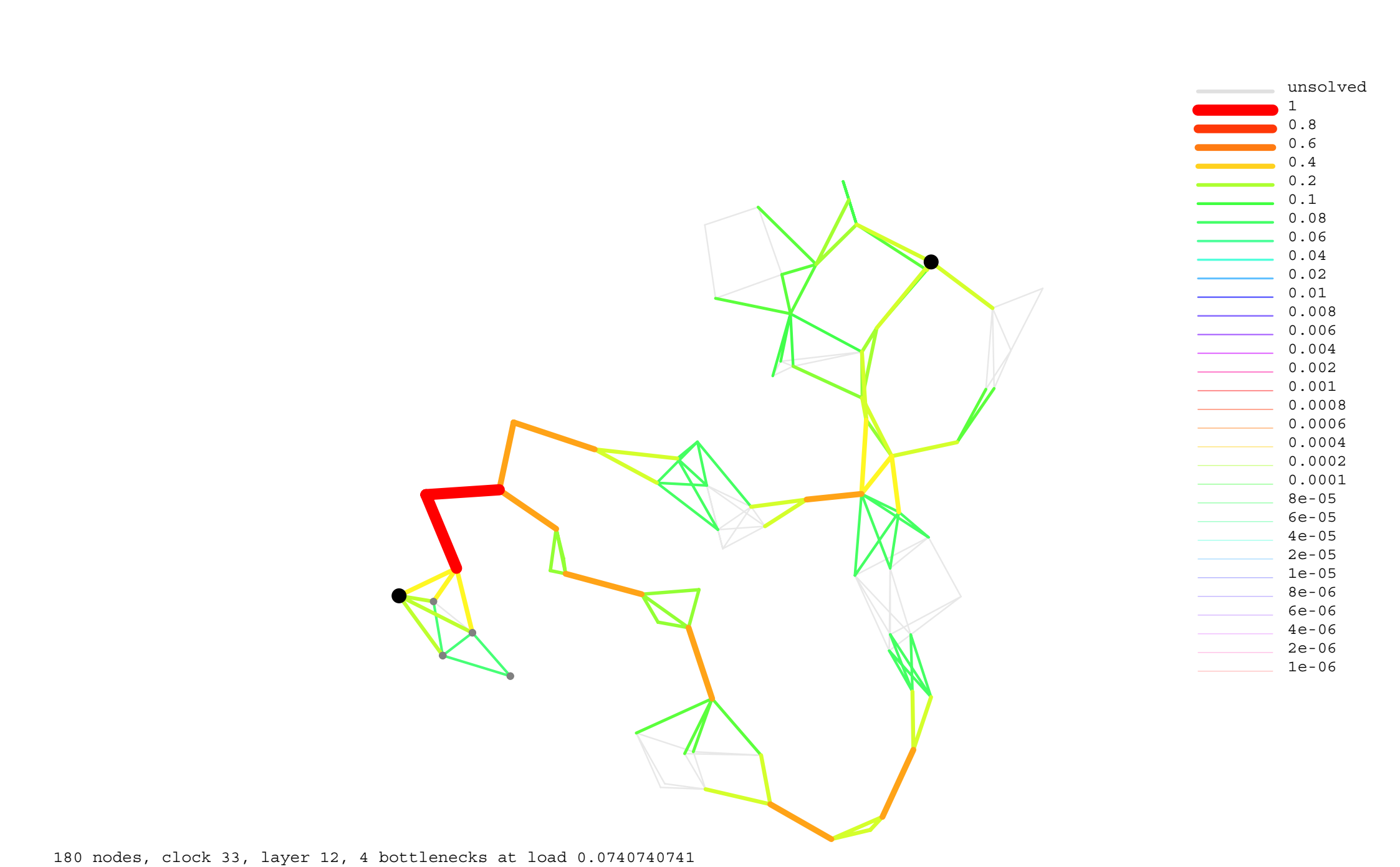
180 nodes, clock 30, layer 12, 5 bottlenecks at load 0.0833333333



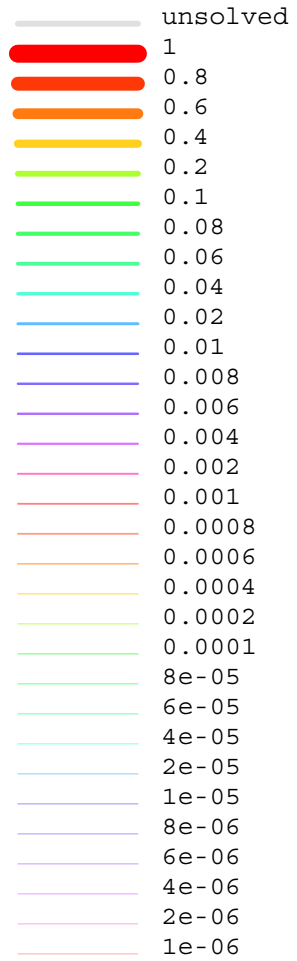
180 nodes, clock 31, layer 12, 2 bottlenecks at load 0.1



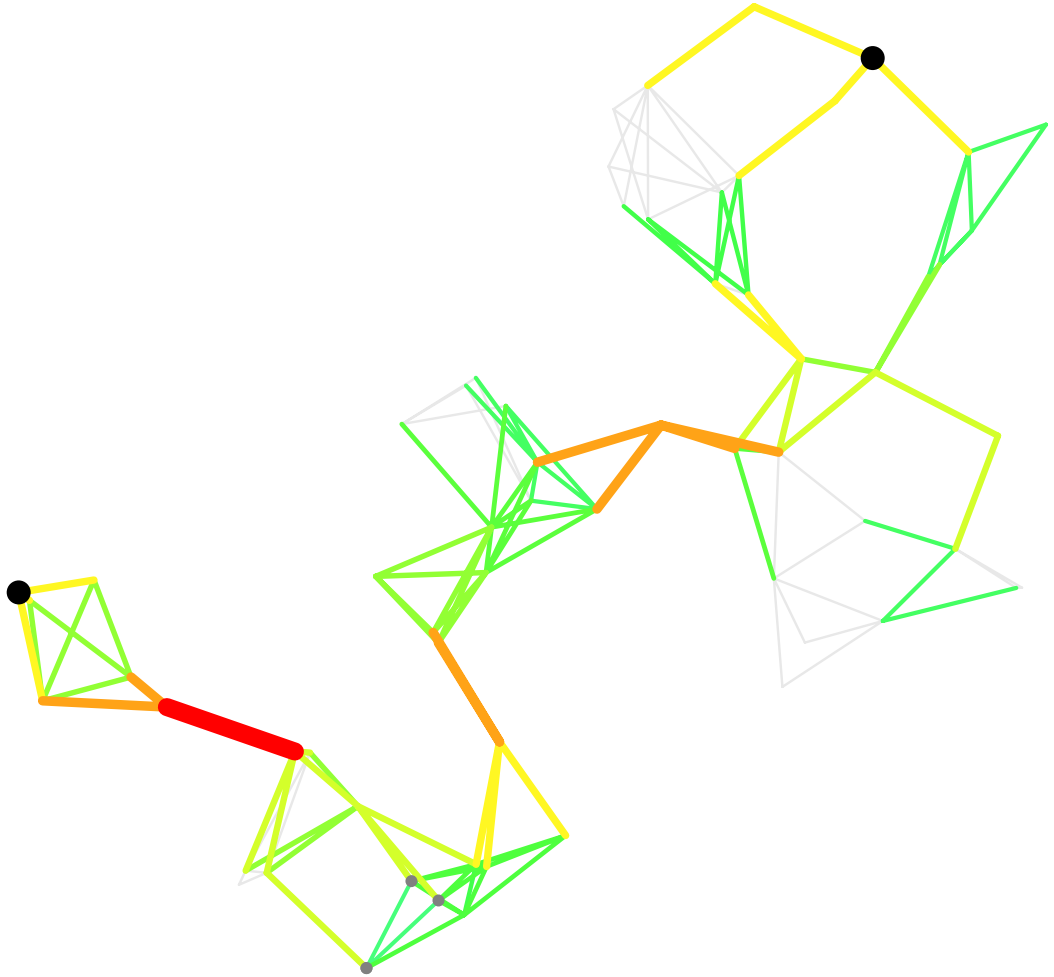
180 nodes, clock 32, layer 12, 4 bottlenecks at load 0.0740740741



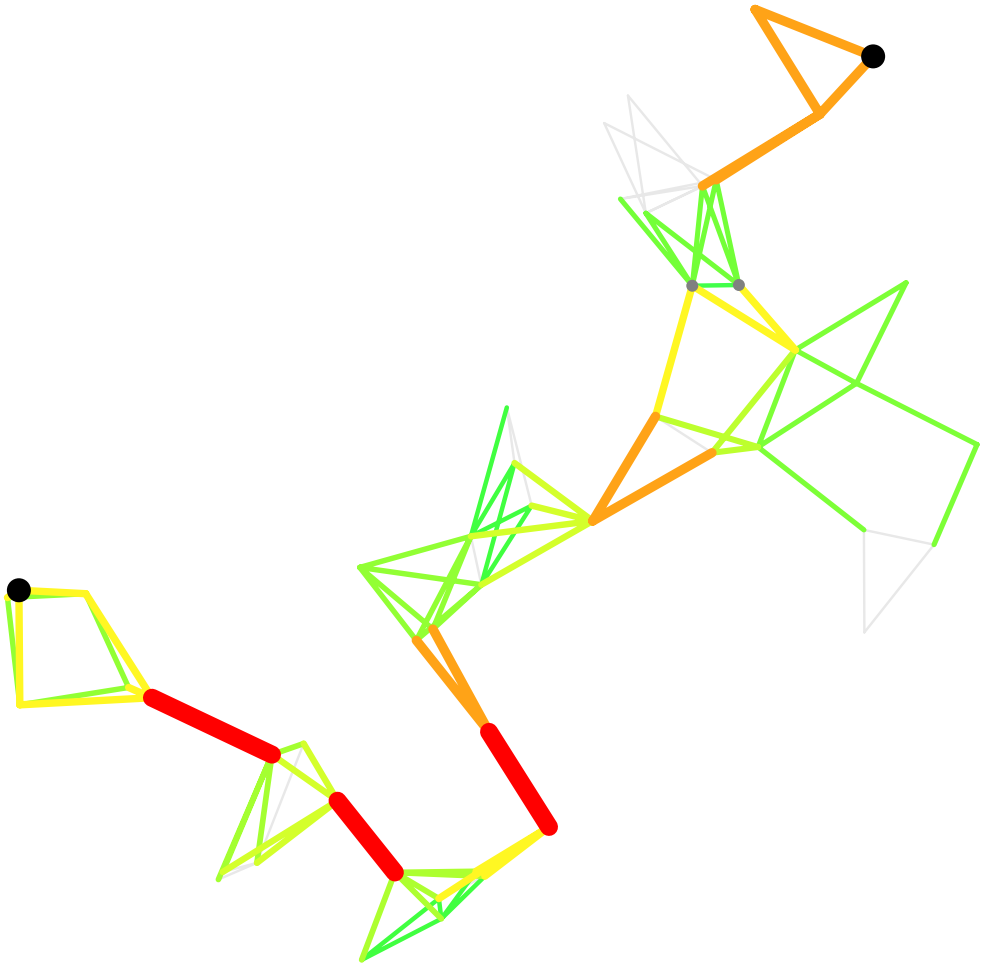
180 nodes, clock 33, layer 12, 4 bottlenecks at load 0.0740740741



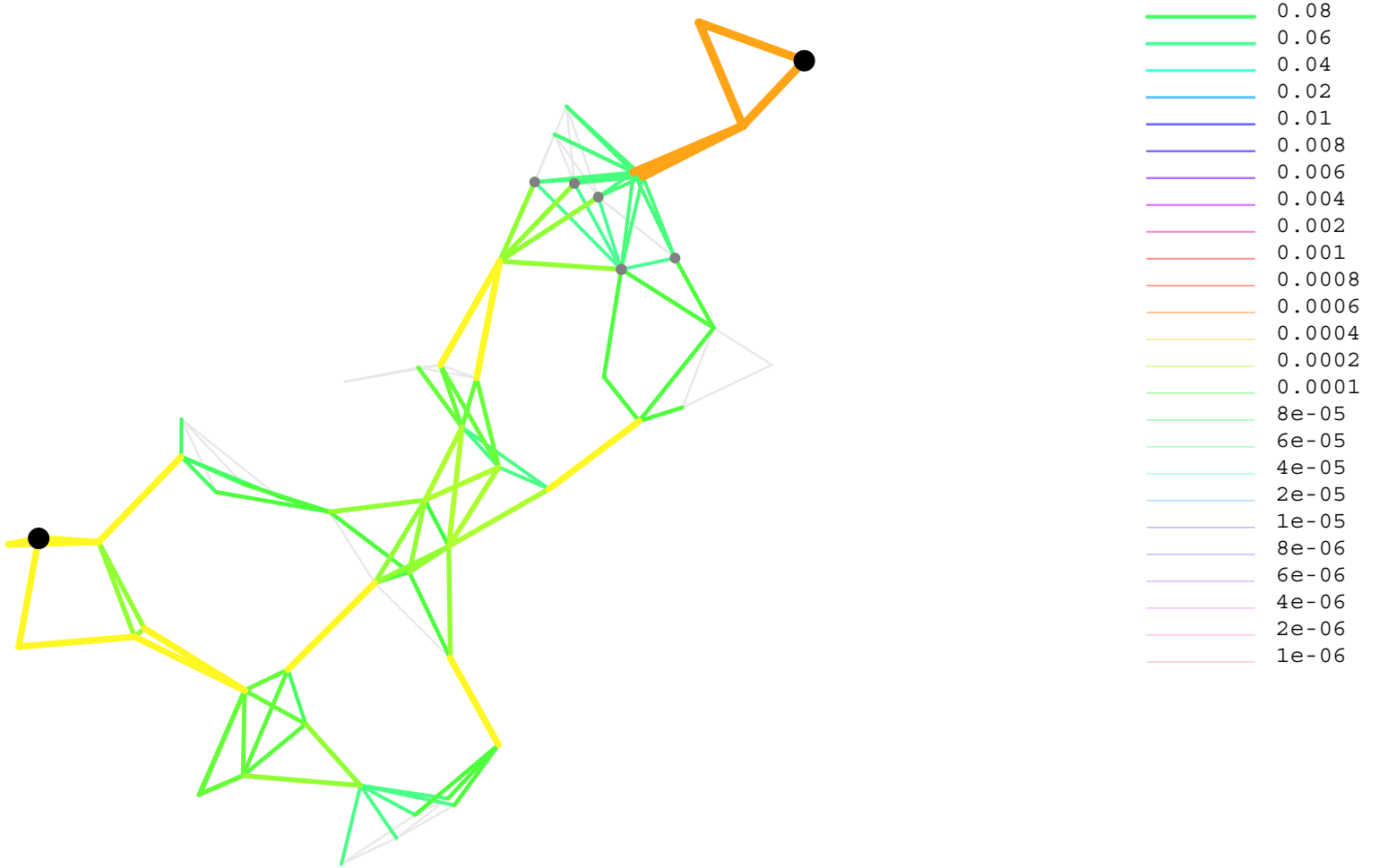
180 nodes, clock 34, layer 0, 0 bottlenecks at load N/A



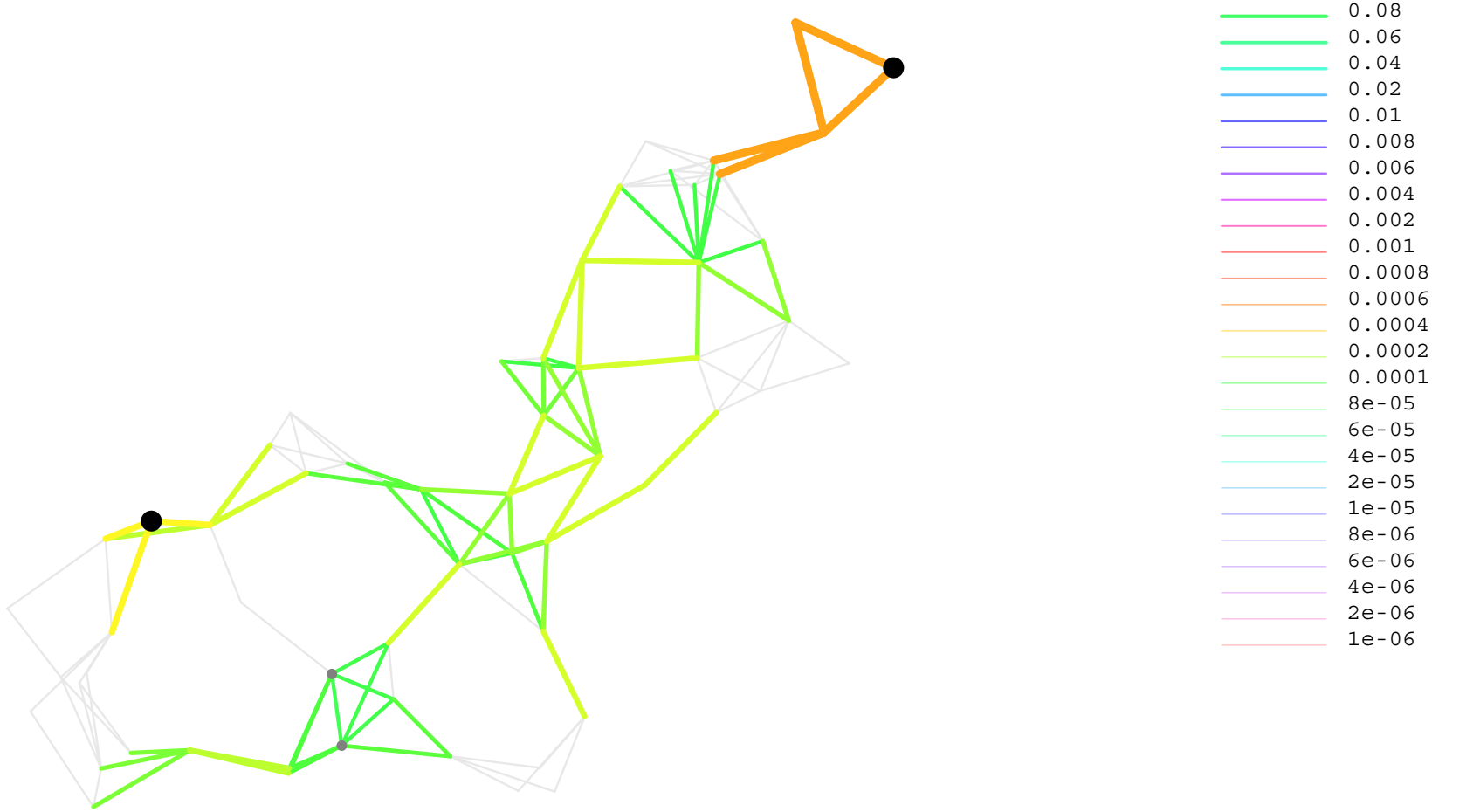
180 nodes, clock 35, layer 12, 2 bottlenecks at load 0.0722222222



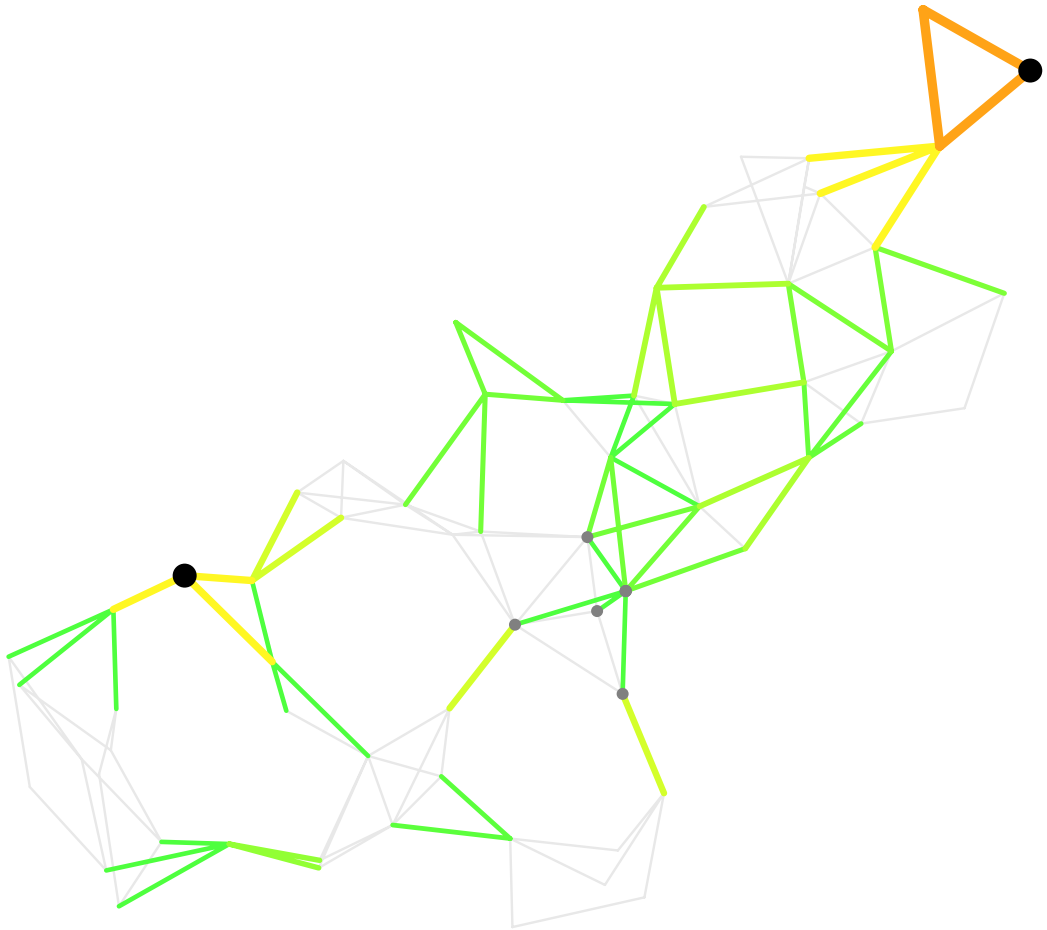
180 nodes, clock 36, layer 12, 1 bottlenecks at load 0.0952380952



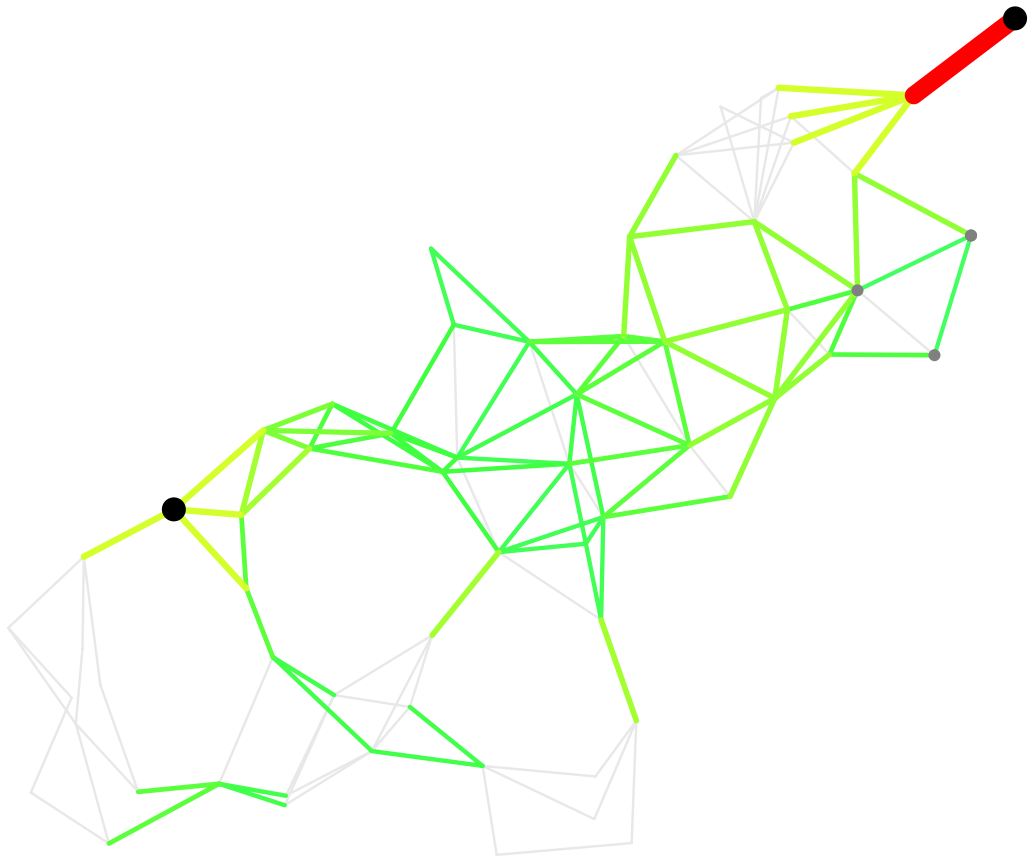
180 nodes, clock 37, layer 12, 4 bottlenecks at load 0.0615079365



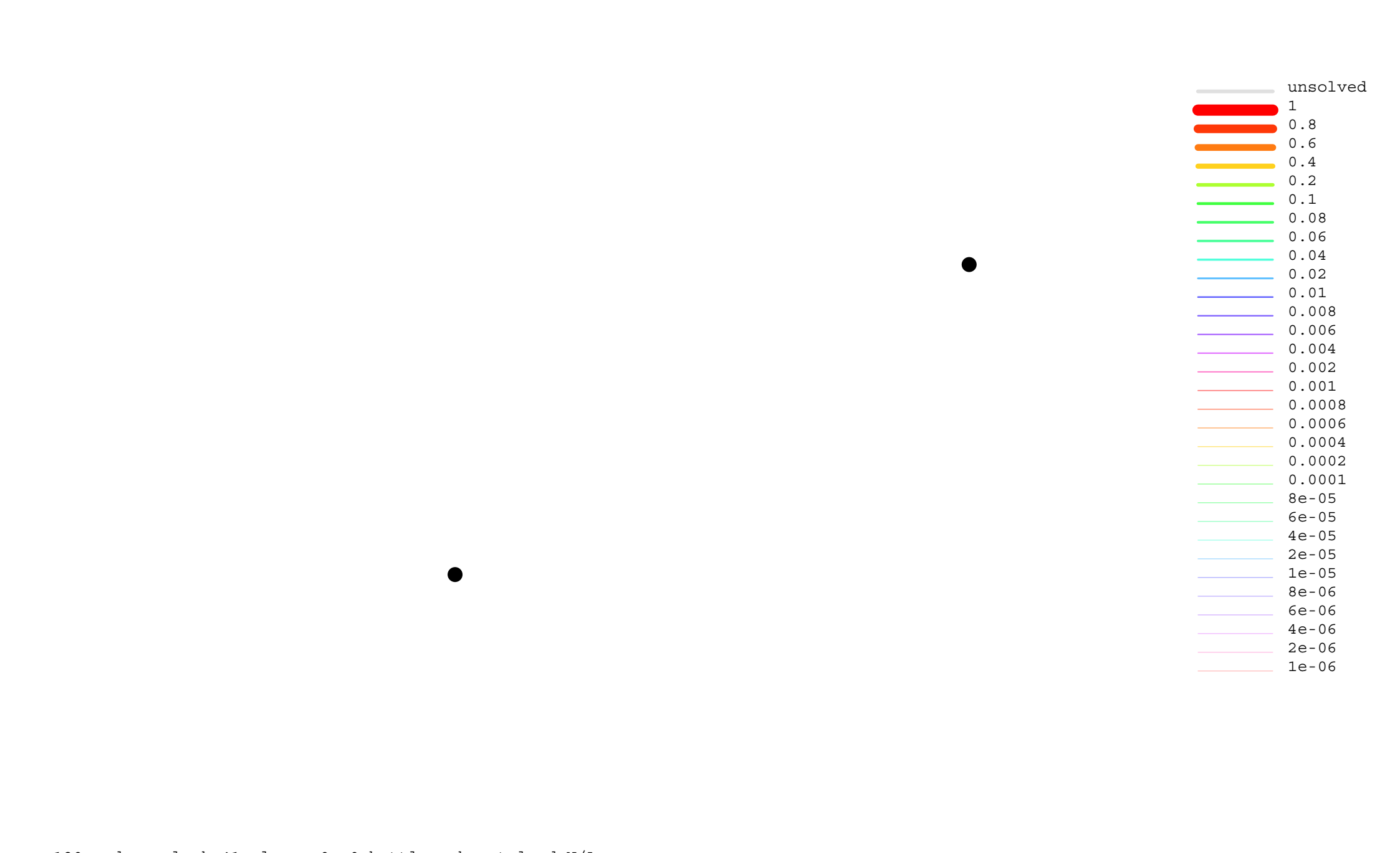
180 nodes, clock 38, layer 12, 1 bottlenecks at load 0.0902777778



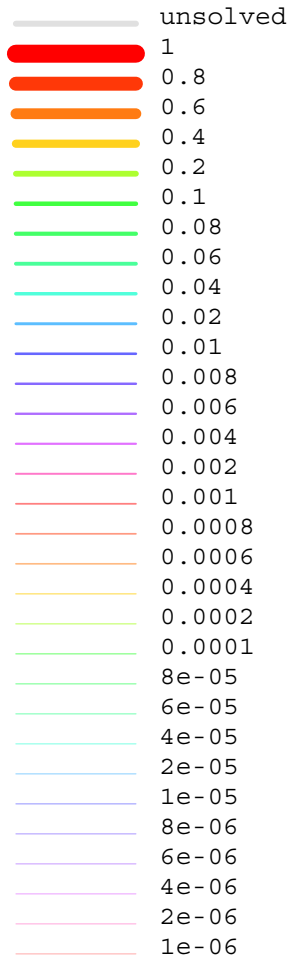
180 nodes, clock 39, layer 12, 4 bottlenecks at load 0.1071428571



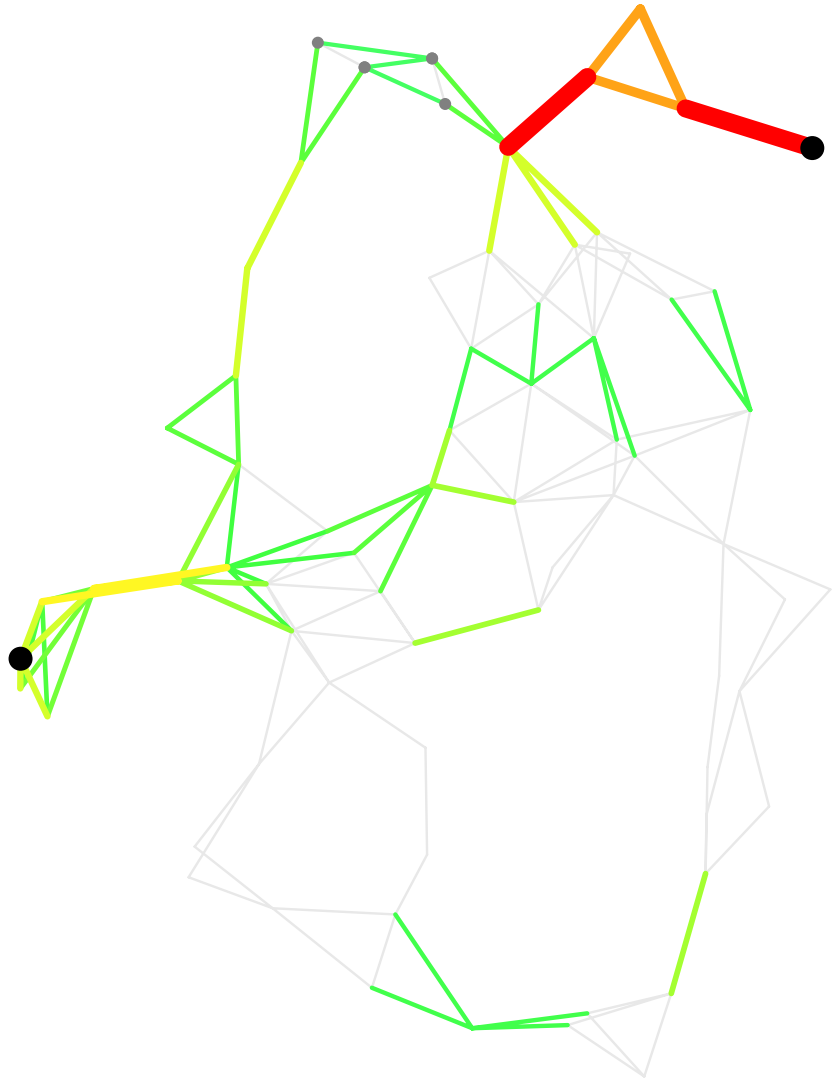
180 nodes, clock 40, layer 12, 2 bottlenecks at load 0.0833333333



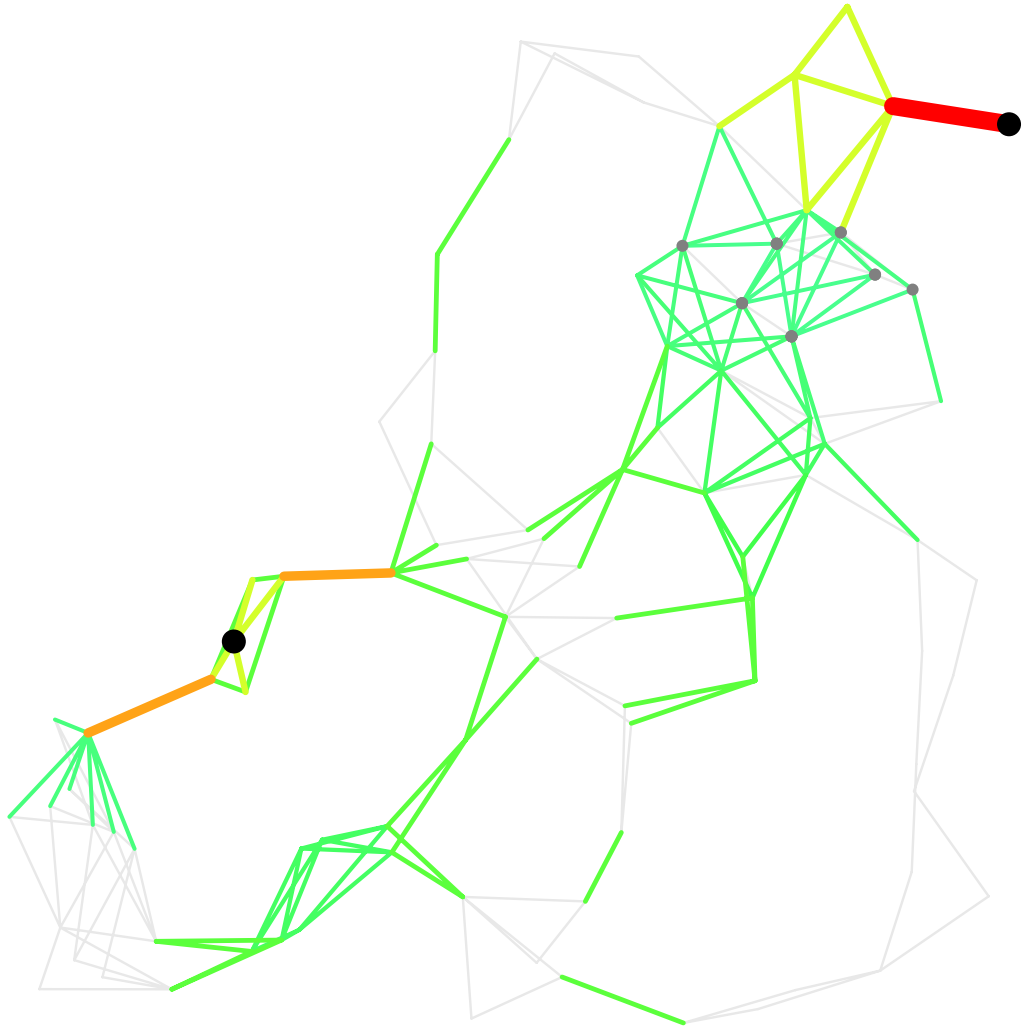
180 nodes, clock 41, layer 0, 0 bottlenecks at load N/A



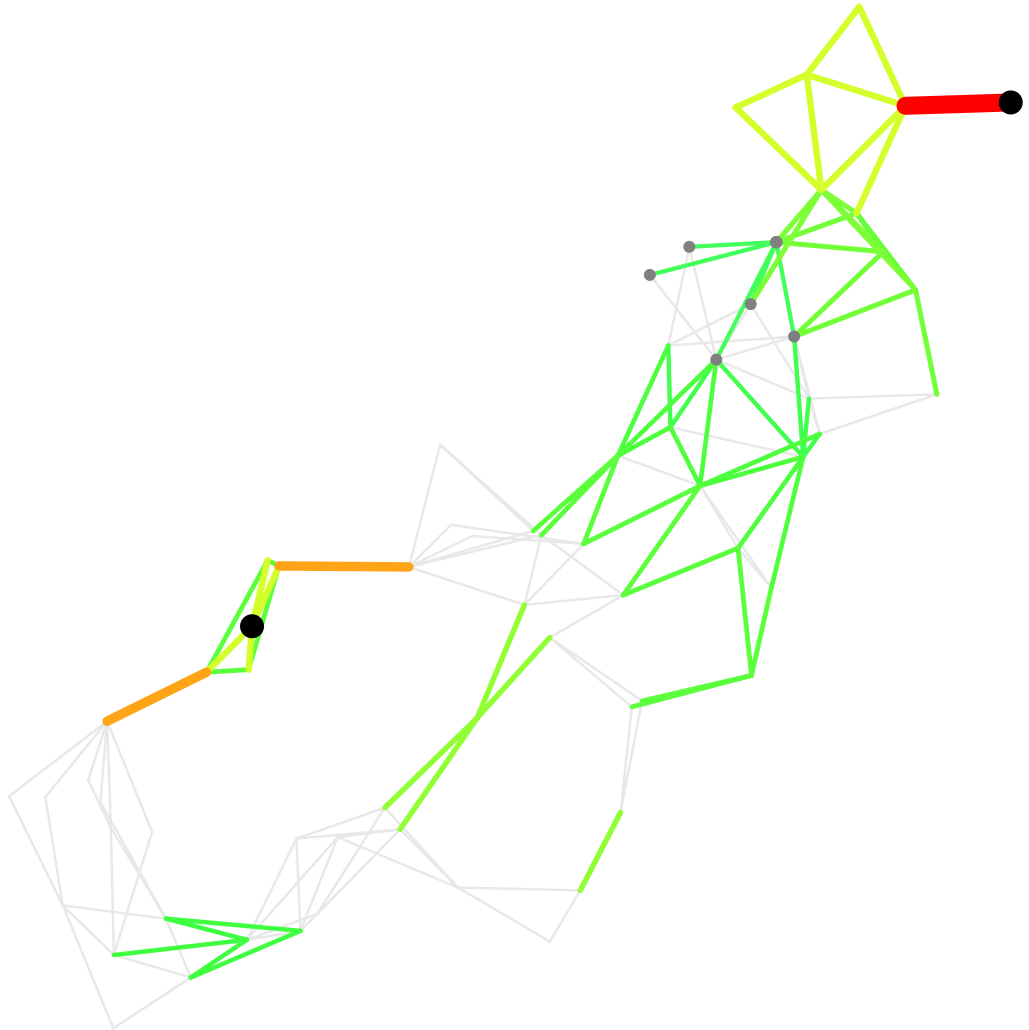
180 nodes, clock 42, layer 0, 0 bottlenecks at load N/A



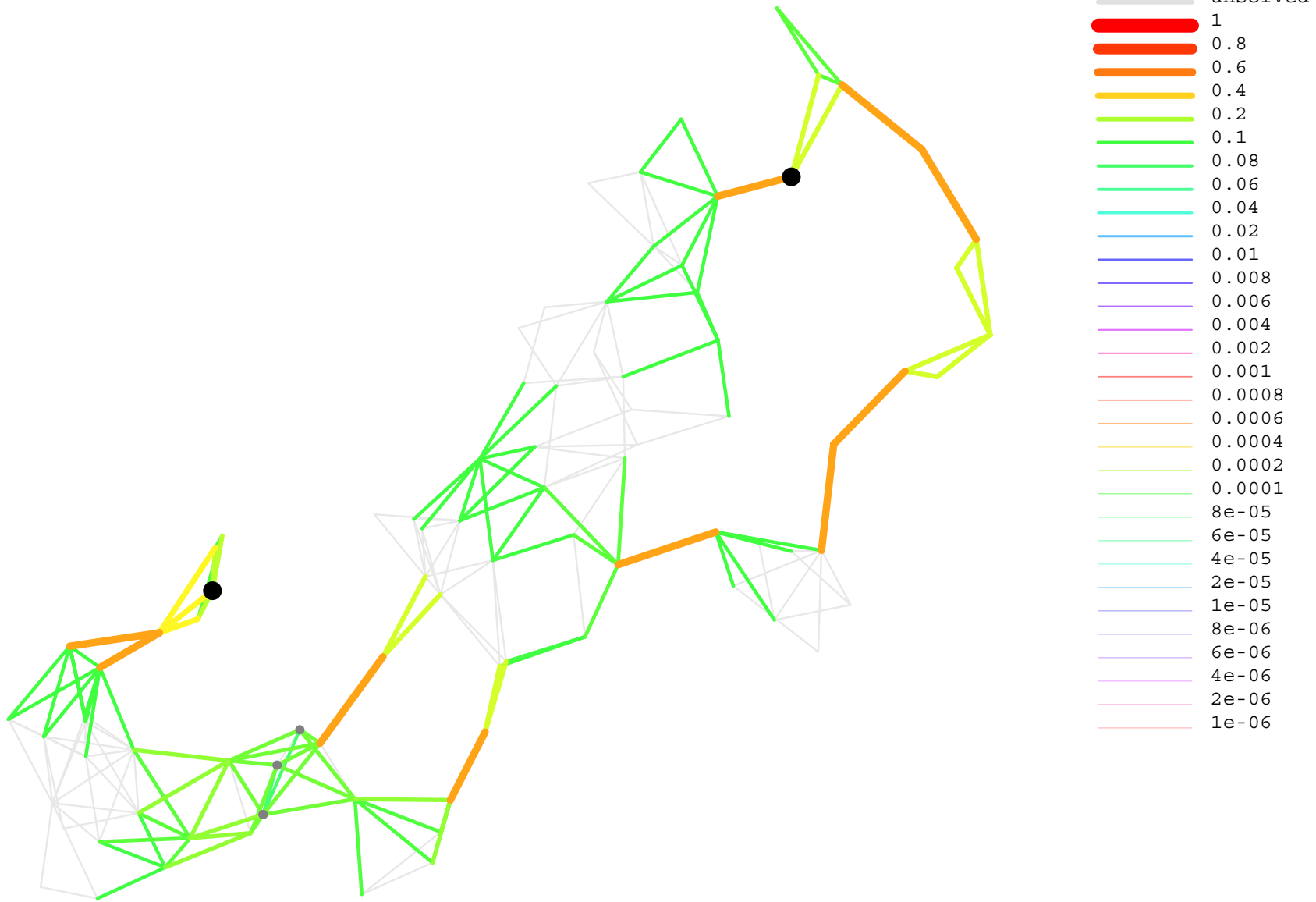
180 nodes, clock 43, layer 12, 3 bottlenecks at load 0.0833333333



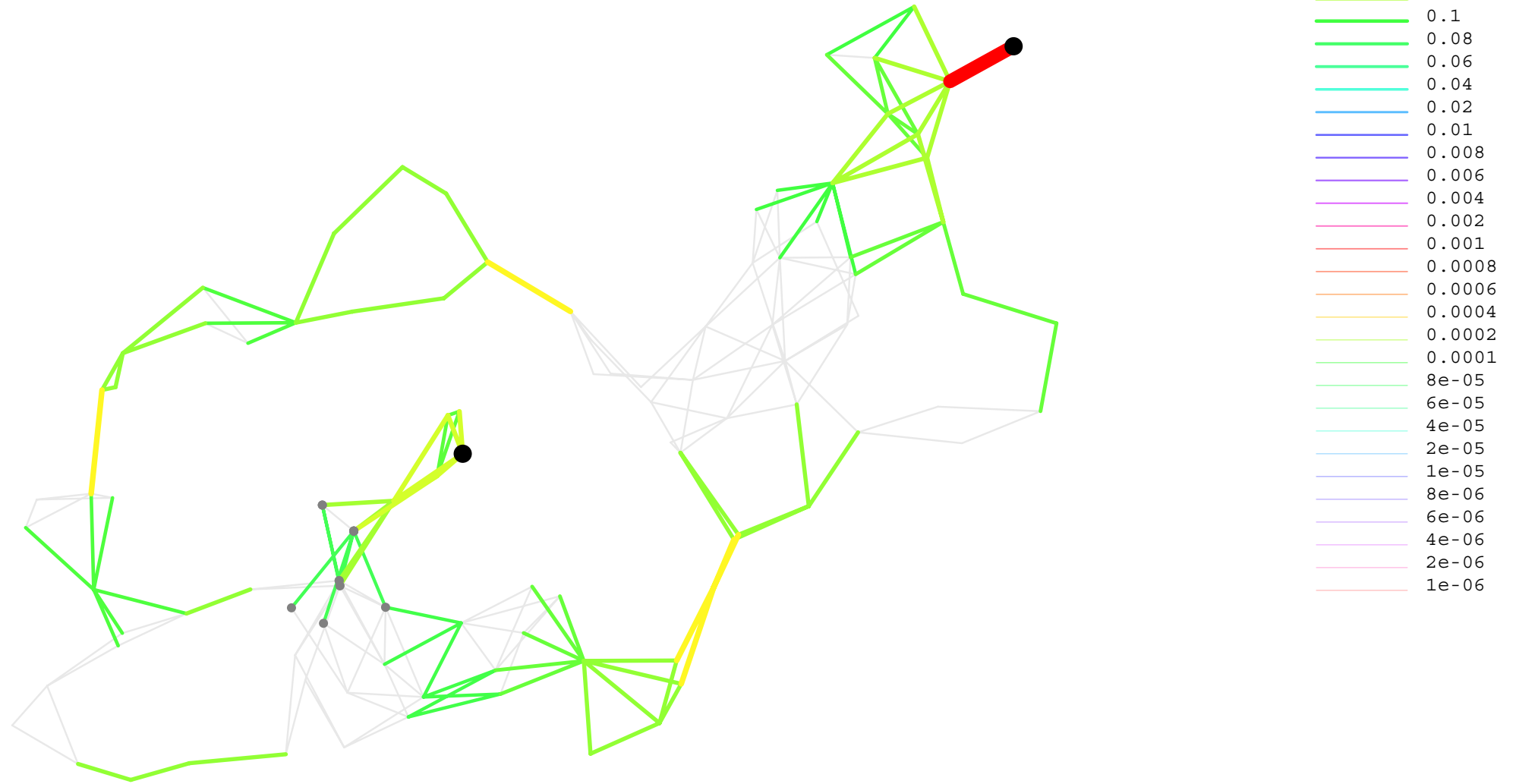
180 nodes, clock 44, layer 12, 8 bottlenecks at load 0.0653935185



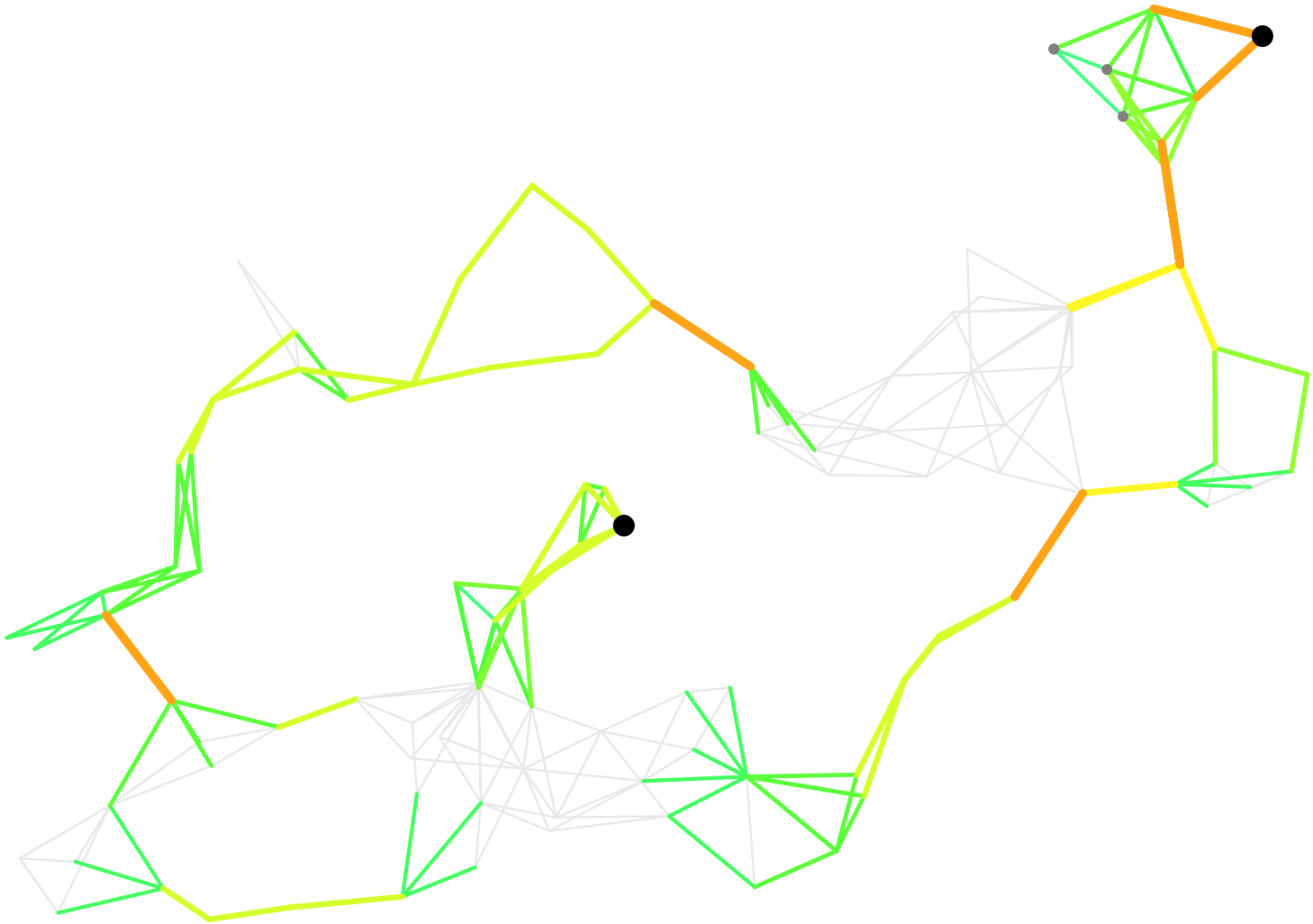
180 nodes, clock 45, layer 12, 5 bottlenecks at load 0.086



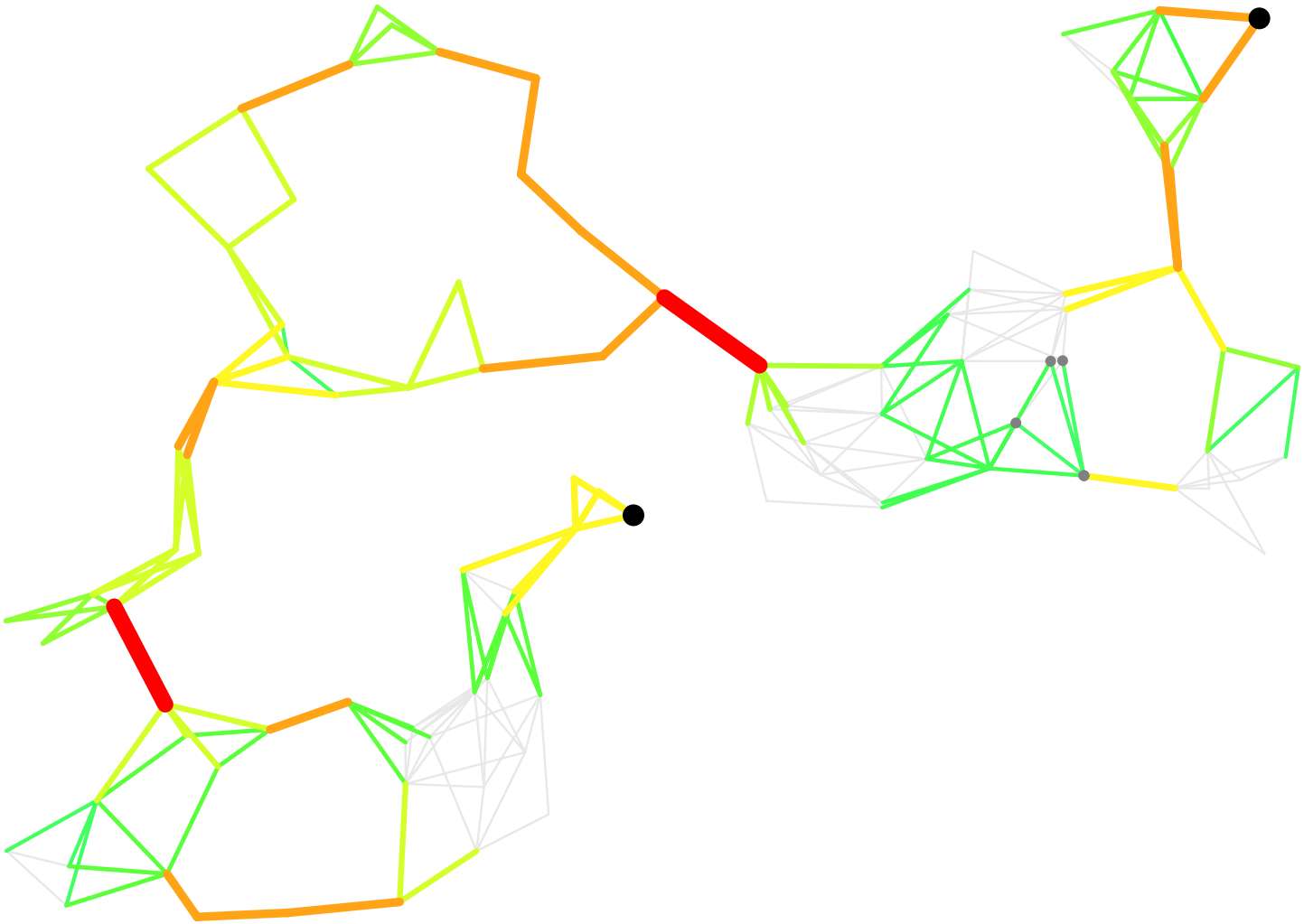
180 nodes, clock 46, layer 12, 2 bottlenecks at load 0.0785714286



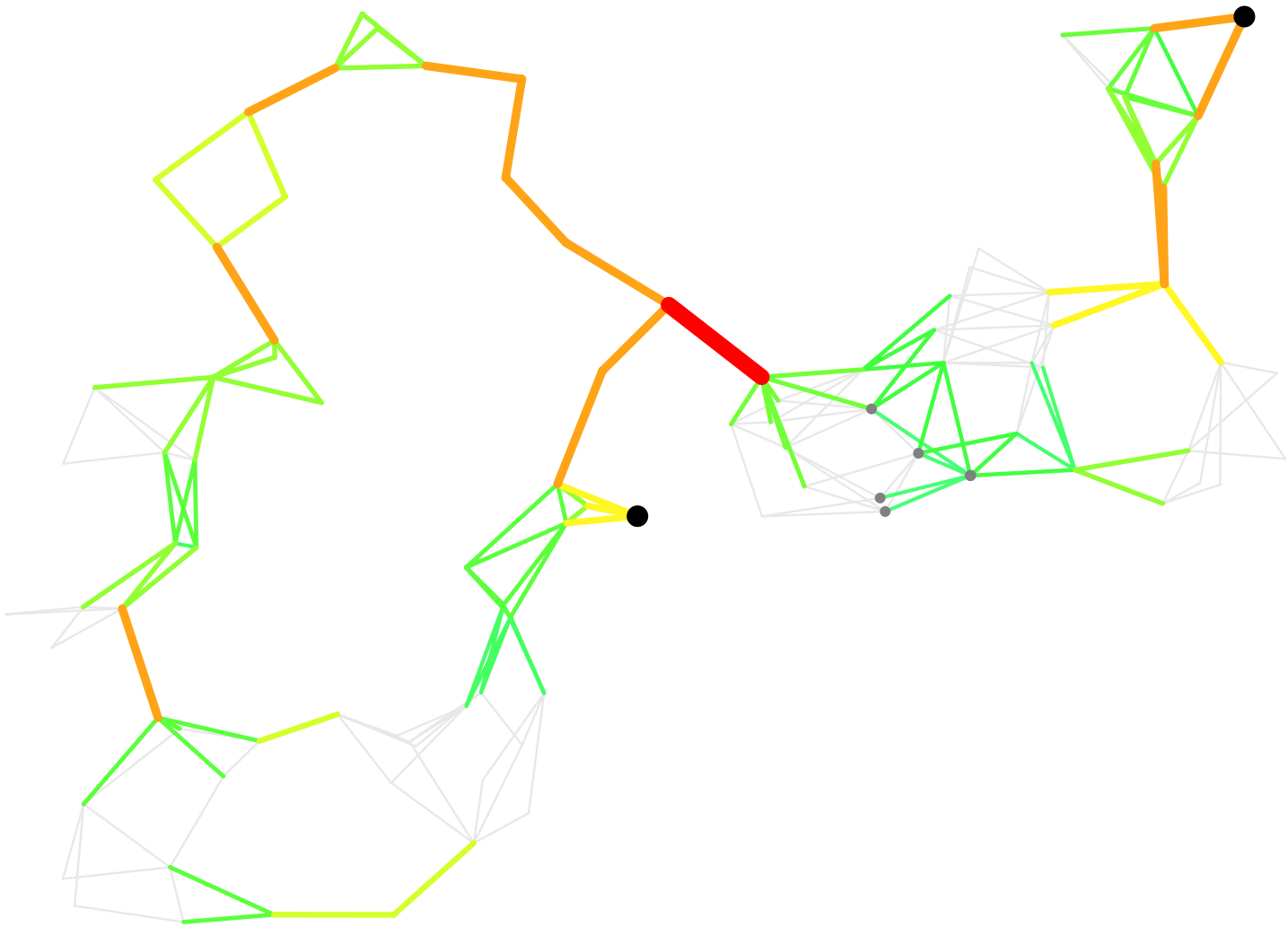
180 nodes, clock 47, layer 12, 7 bottlenecks at load 0.0892857143



180 nodes, clock 48, layer 12, 2 bottlenecks at load 0.0666666667



180 nodes, clock 49, layer 12, 3 bottlenecks at load 0.0808080808



180 nodes, clock 50, layer 12, 4 bottlenecks at load 0.075

Thank you!

Emin.Gabrielyan@epfl.ch