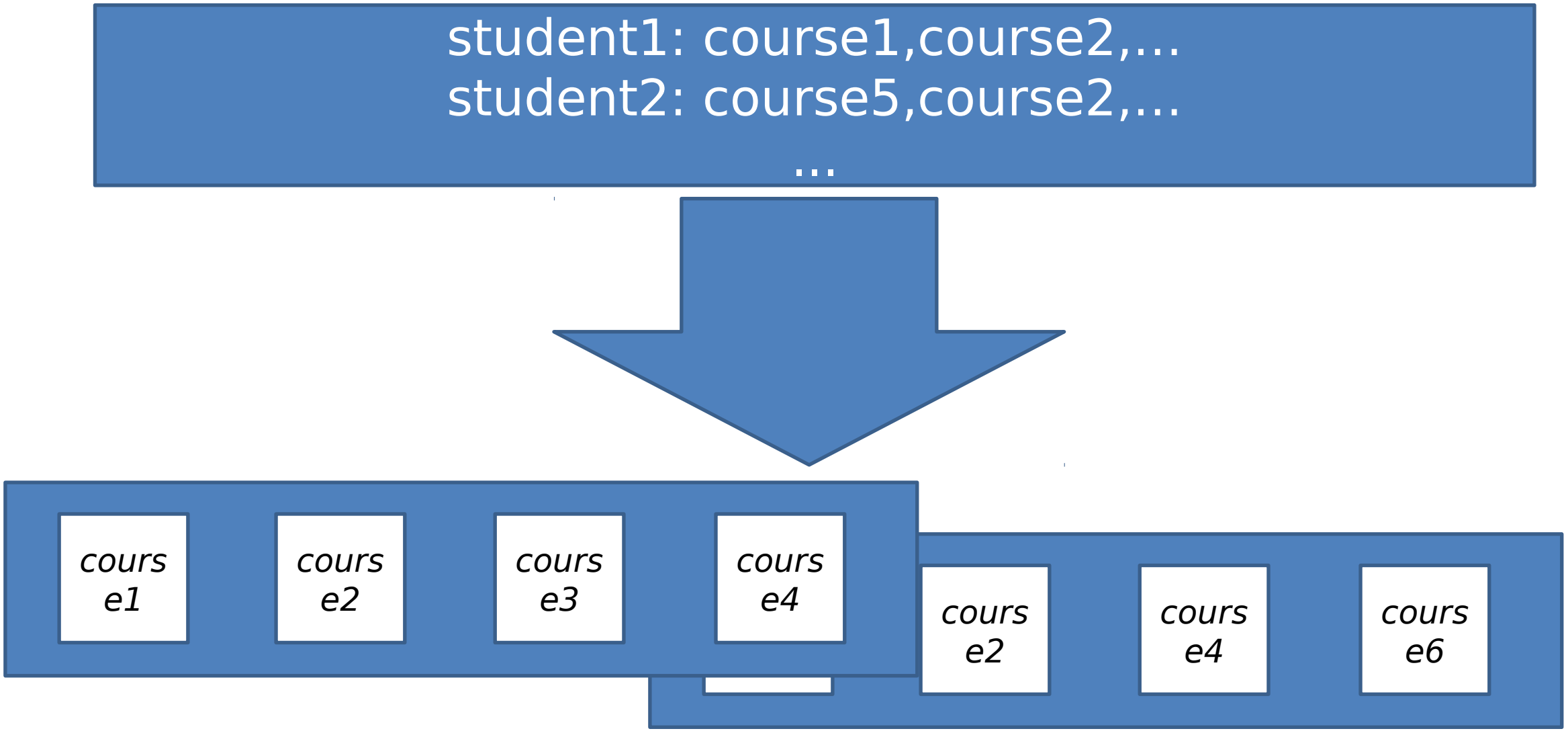
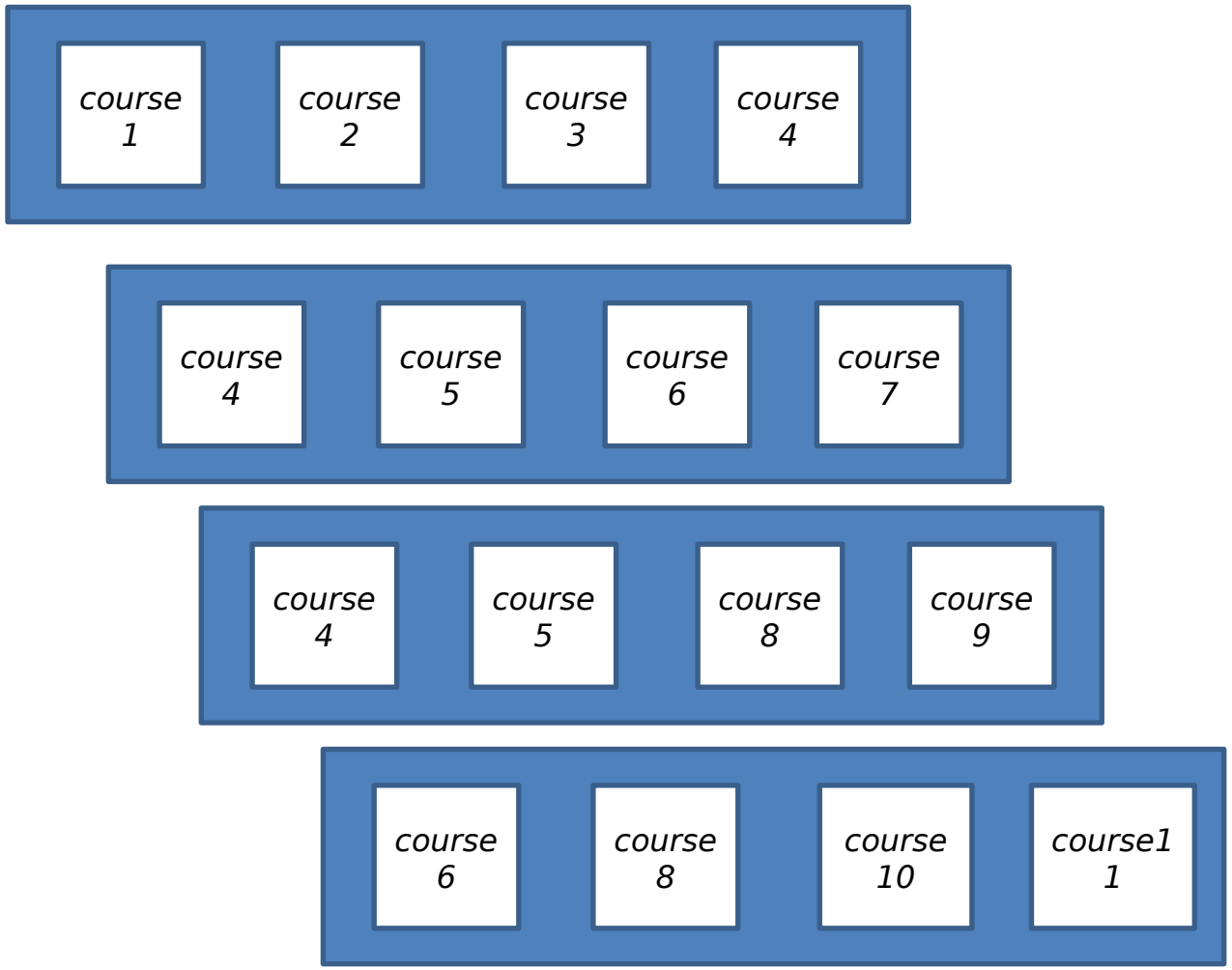
Peeling Set Systems

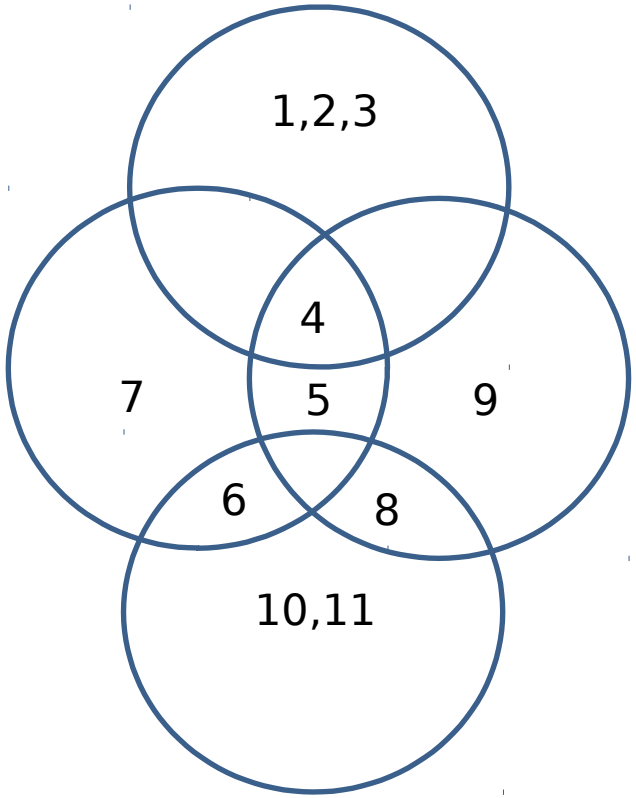
Daniel Nakhimovich
PI: James Abello

Data to Collection of Sets

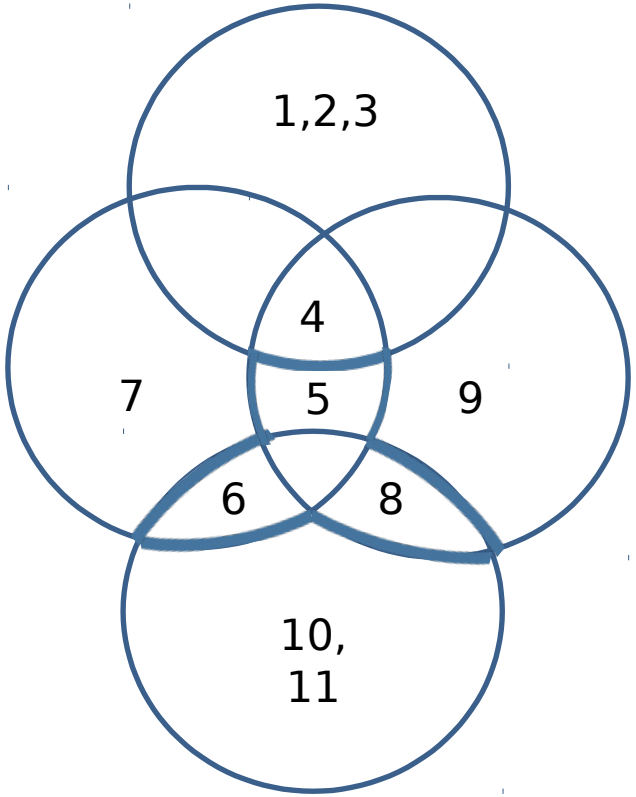


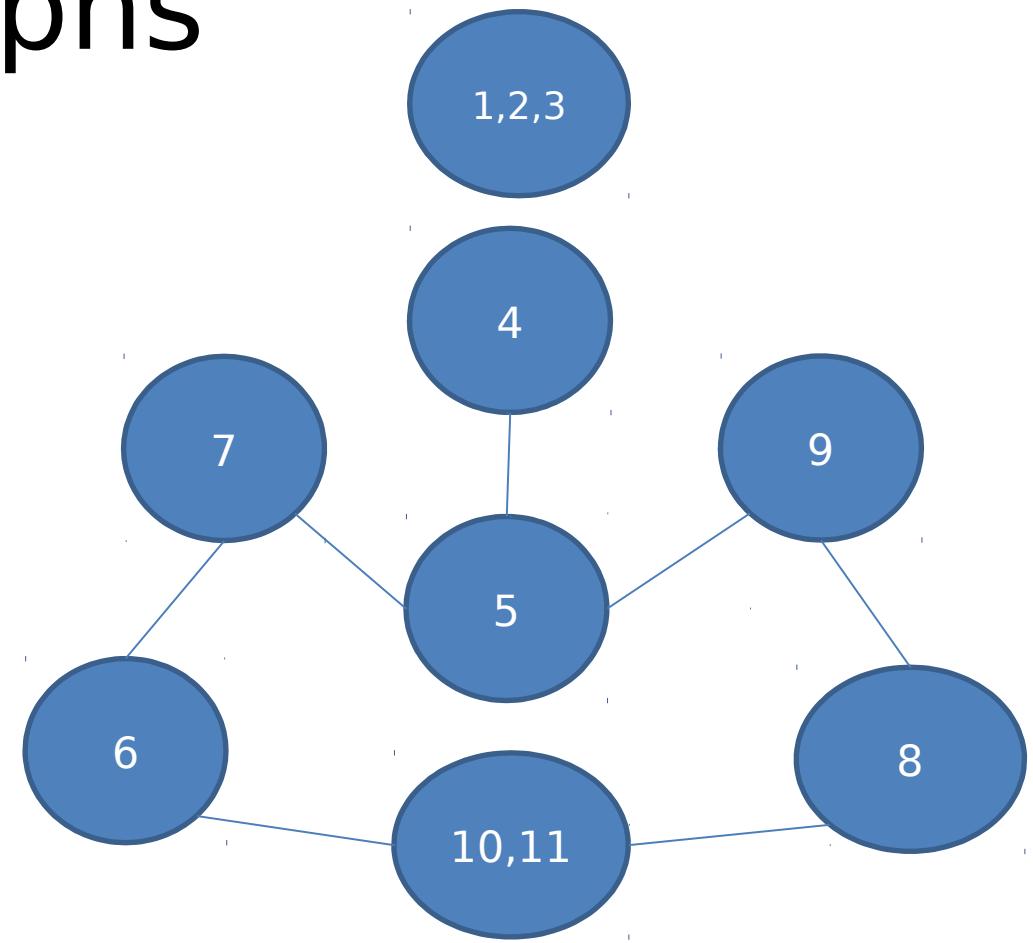
Sets to Region Graphs





Sets to Region Graphs

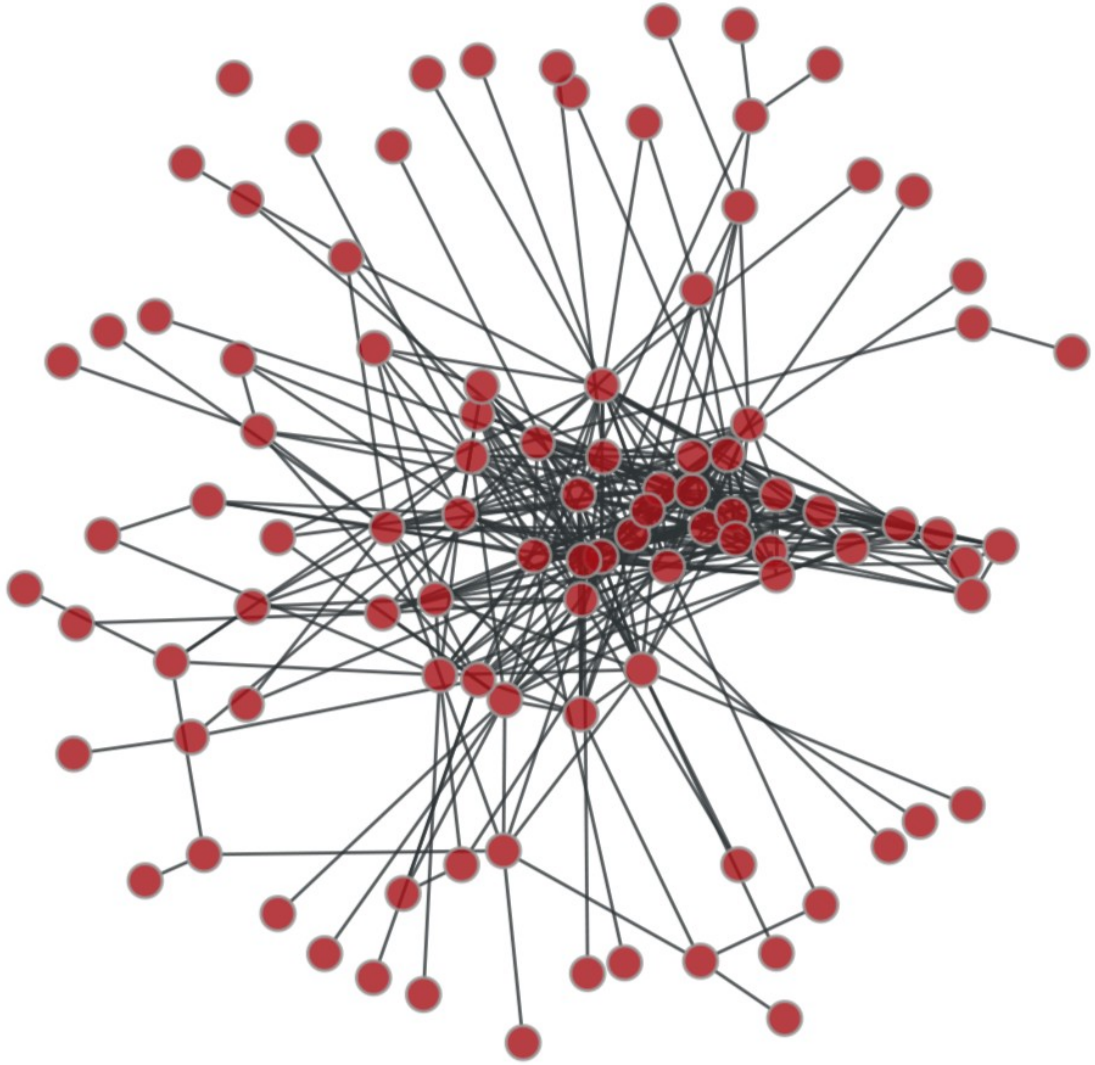




Problem?

Vertices: 99

Edges: 408



Data from: Rutgers Masters Students' Studyplans

Iterative Edge Core Decomposition

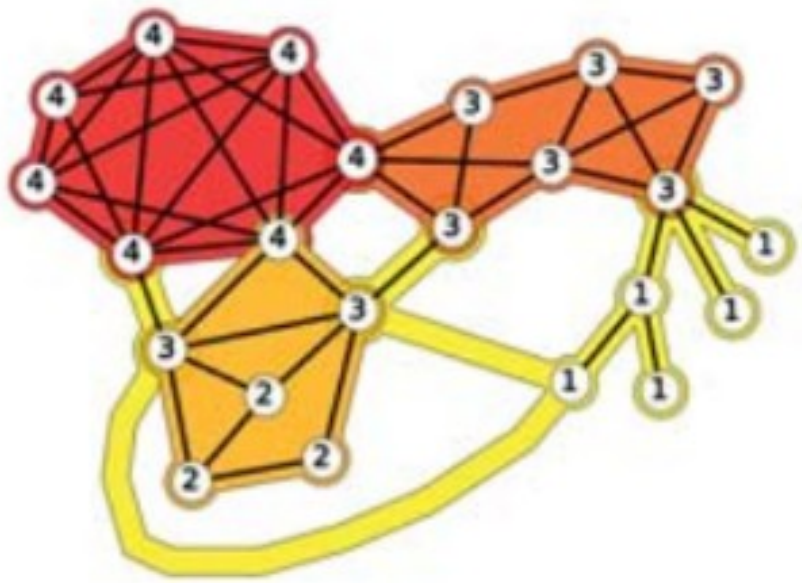
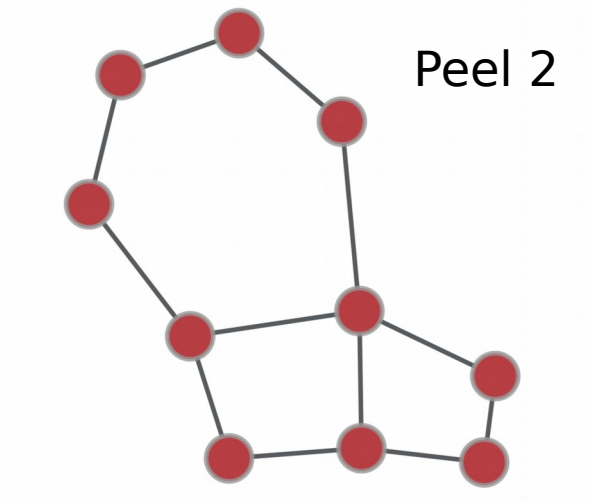
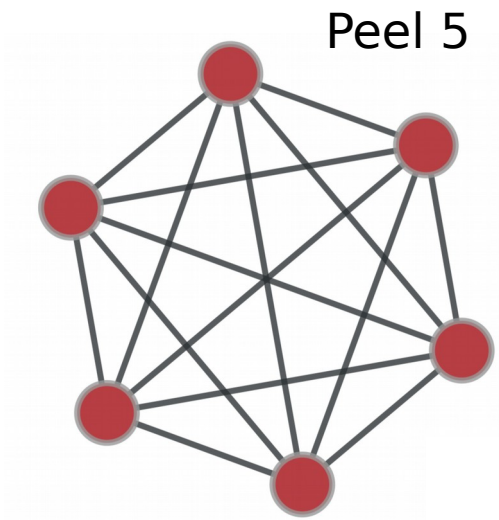
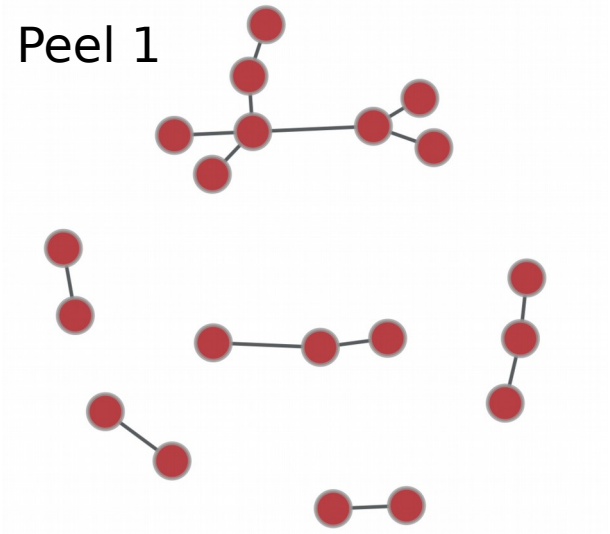


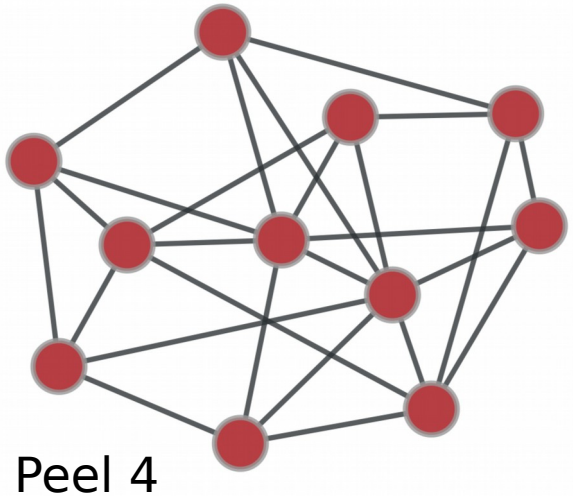
Fig. 3 (d) from: J. Abello, F. Quelroy, "Fixed Points of Degree Peeling", ASONAM 2013, Advances in Social Networks, IEEE/ACM International Conference, Niagara Falls, Canada, August 2013.

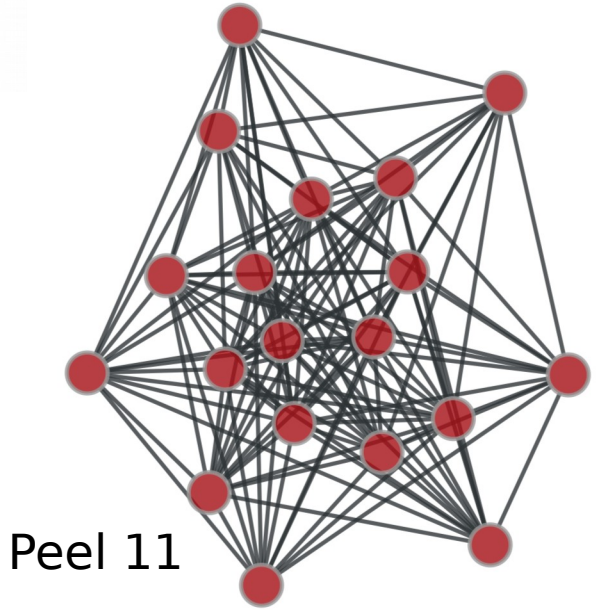
Iterative Edge Core Decomposition







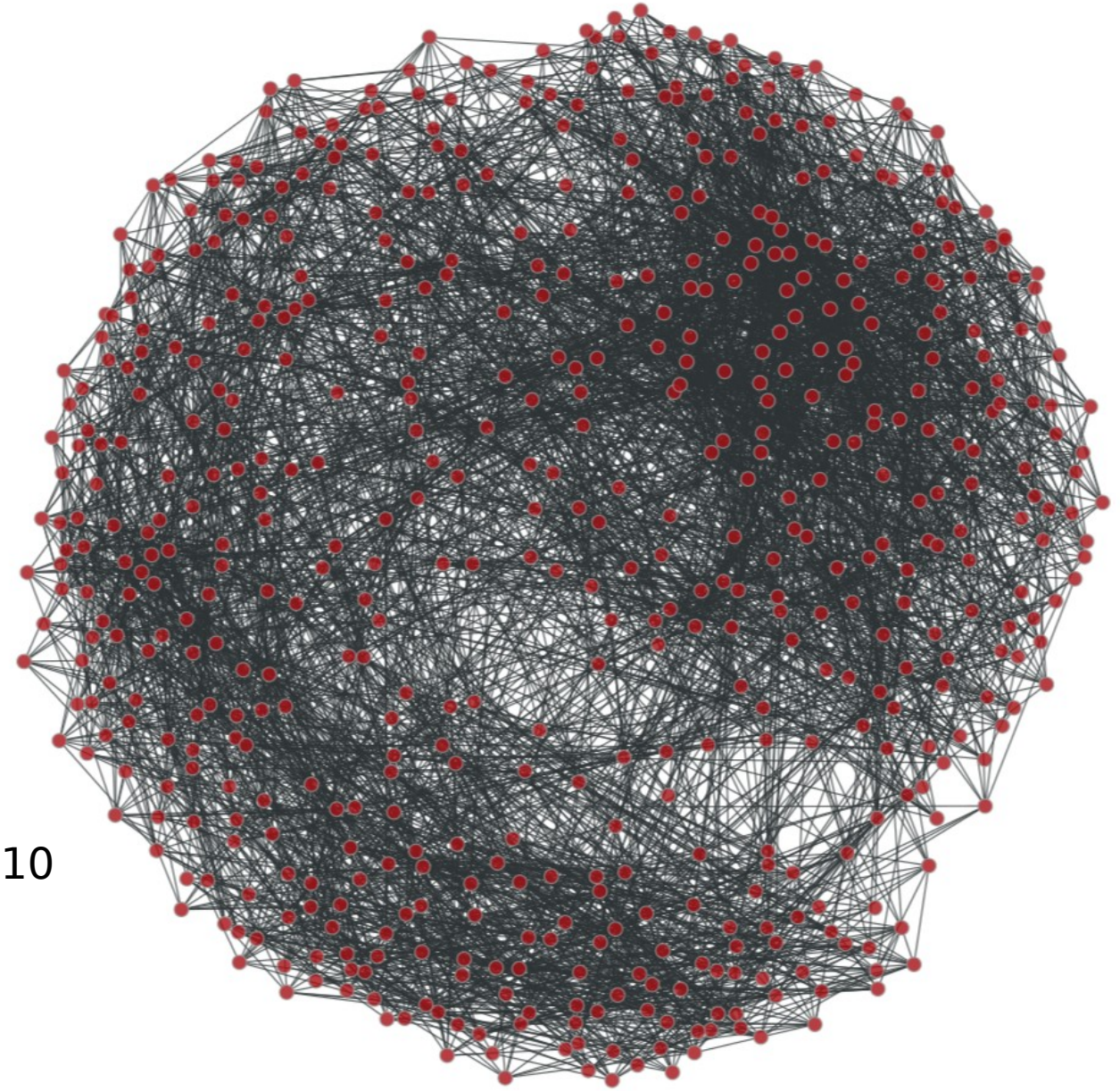




Problem?

Vertices: 572

Edges: 5237



Peel 10

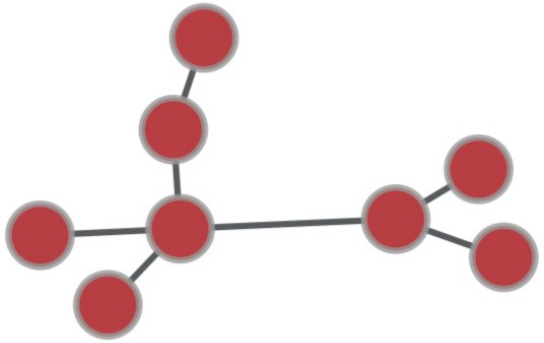
Data from: J. Leskovec, D. Huttenlocher, J. Kleinberg. Signed Networks in Social Media. CHI 2010.
J. Leskovec, D. Huttenlocher, J. Kleinberg. Predicting Positive and Negative Links in Online Social Networks. WWW 2010.

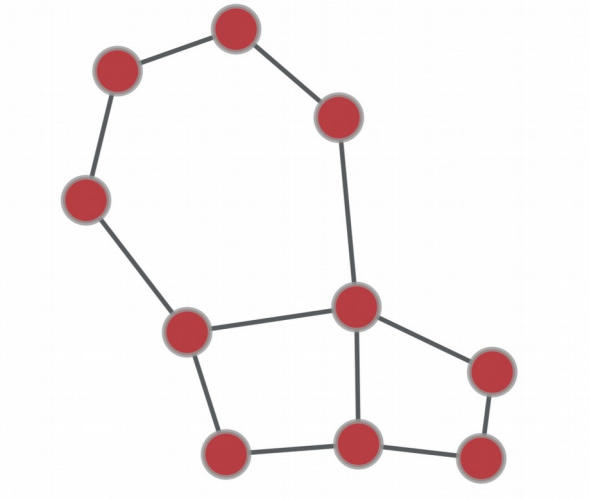
k-connectivity

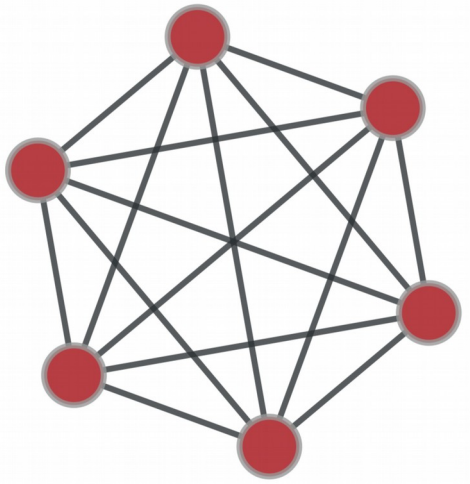
- A graph is **connected** when there is a path between every pair of vertices

- A **separating set** of a graph is a set of vertices that when removed from the graph cause the resulting induced sub-graph to be disconnected.

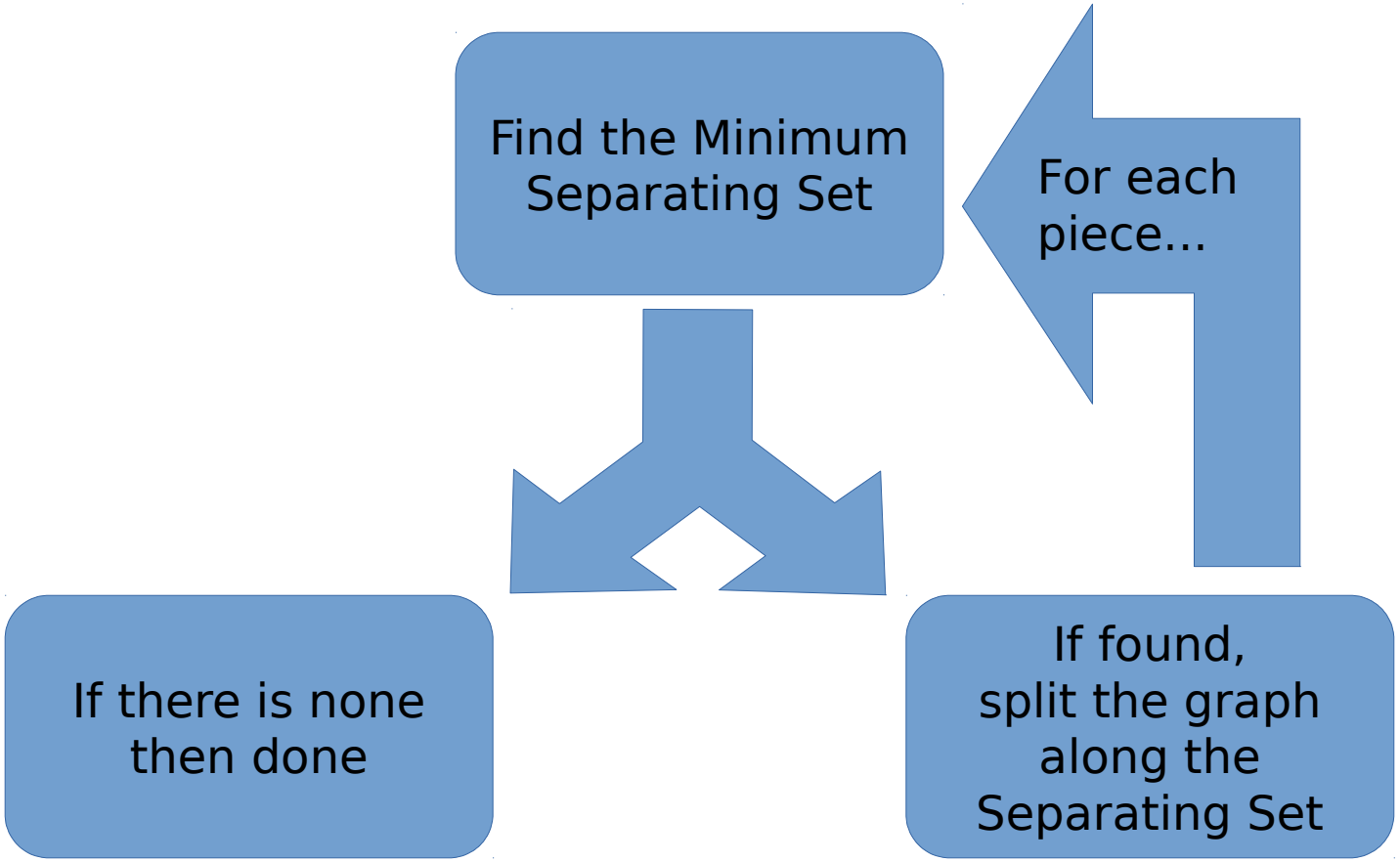
- A graph is **k-connected** if it has more than k vertices and does not have a separating set of less than k vertices.



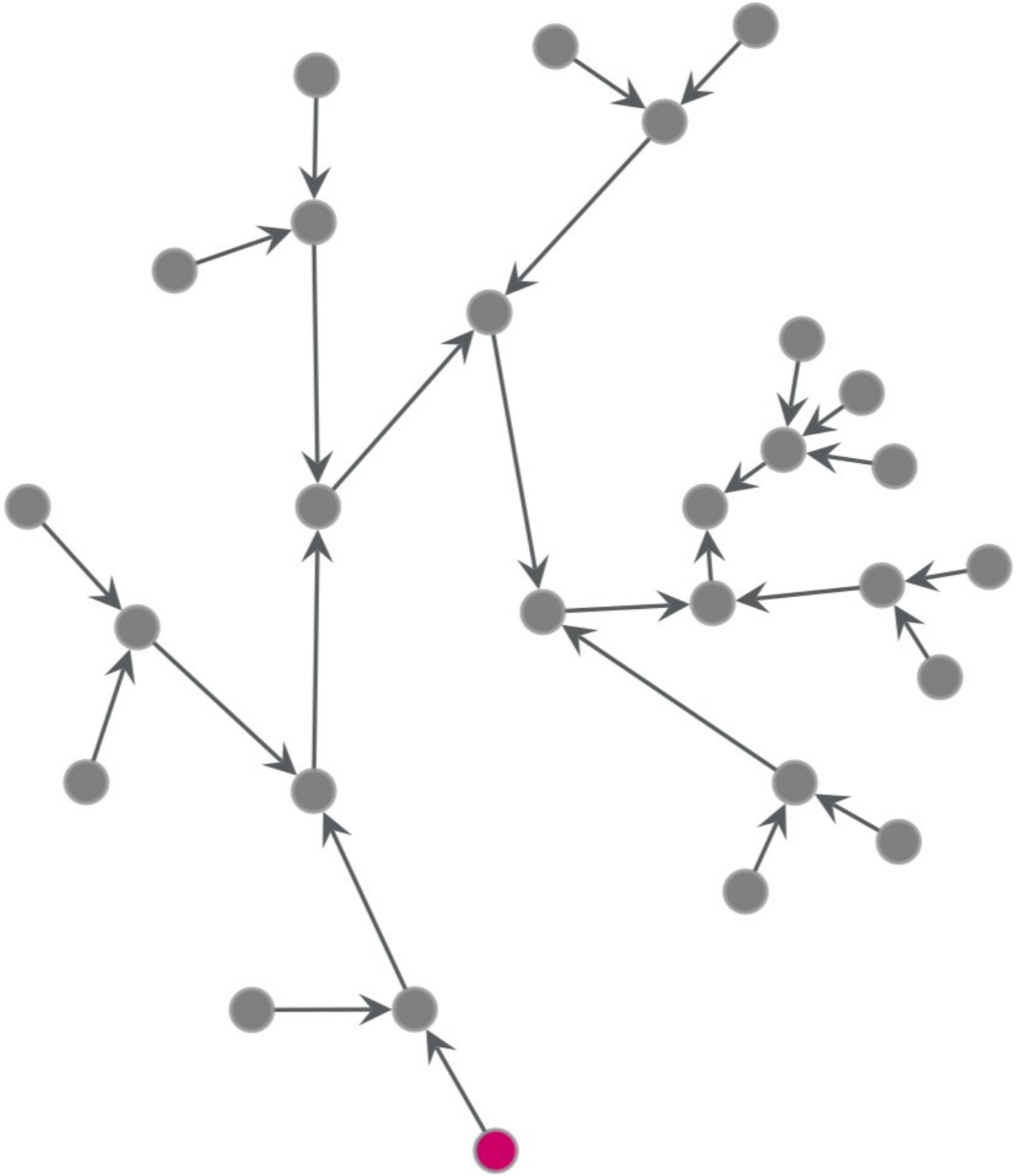


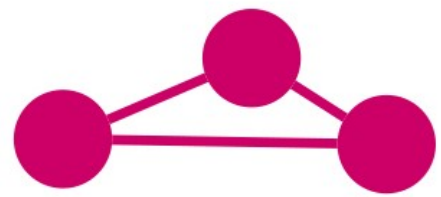


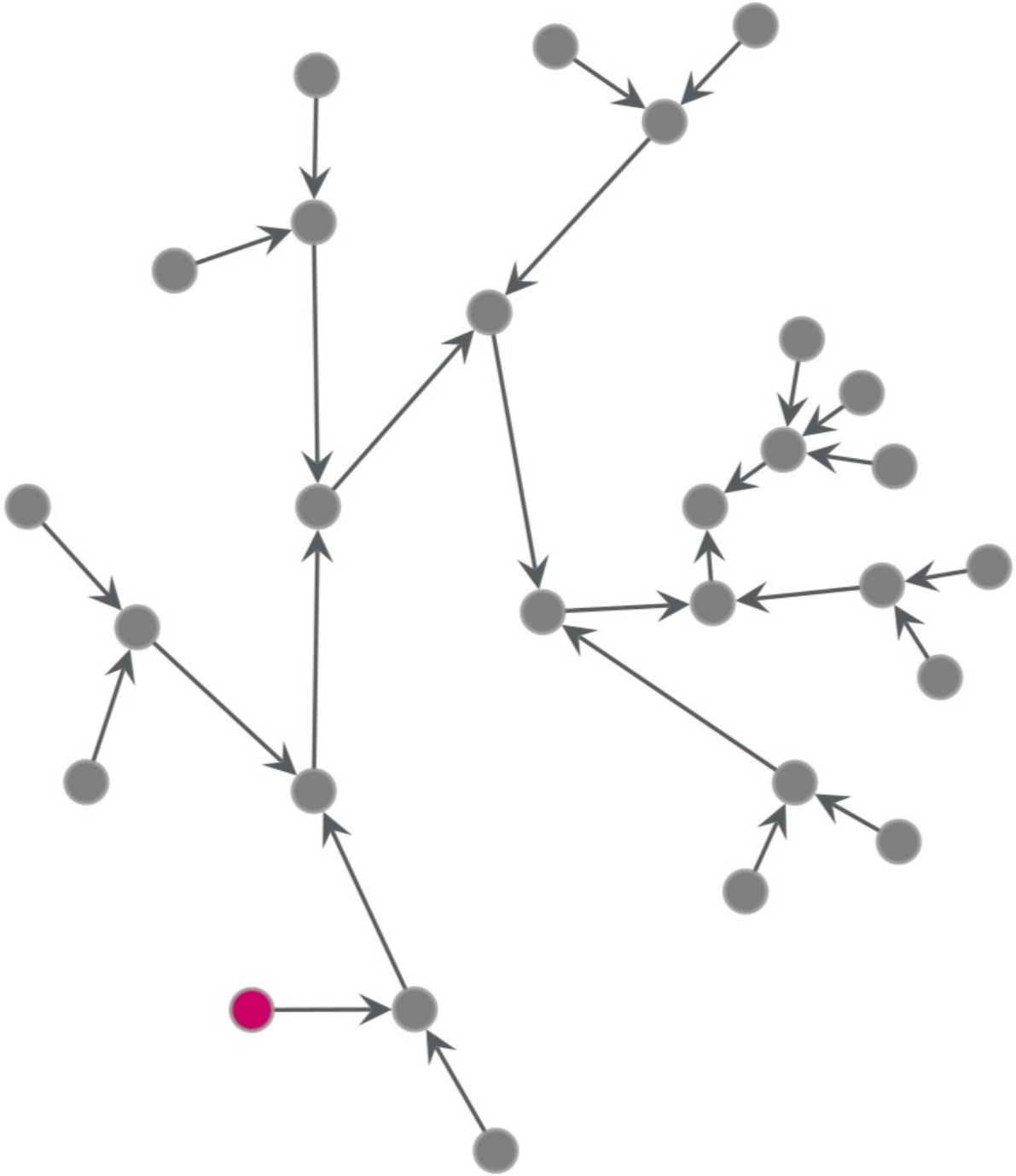
k-connected component decomposition

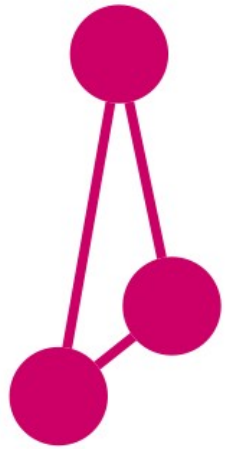


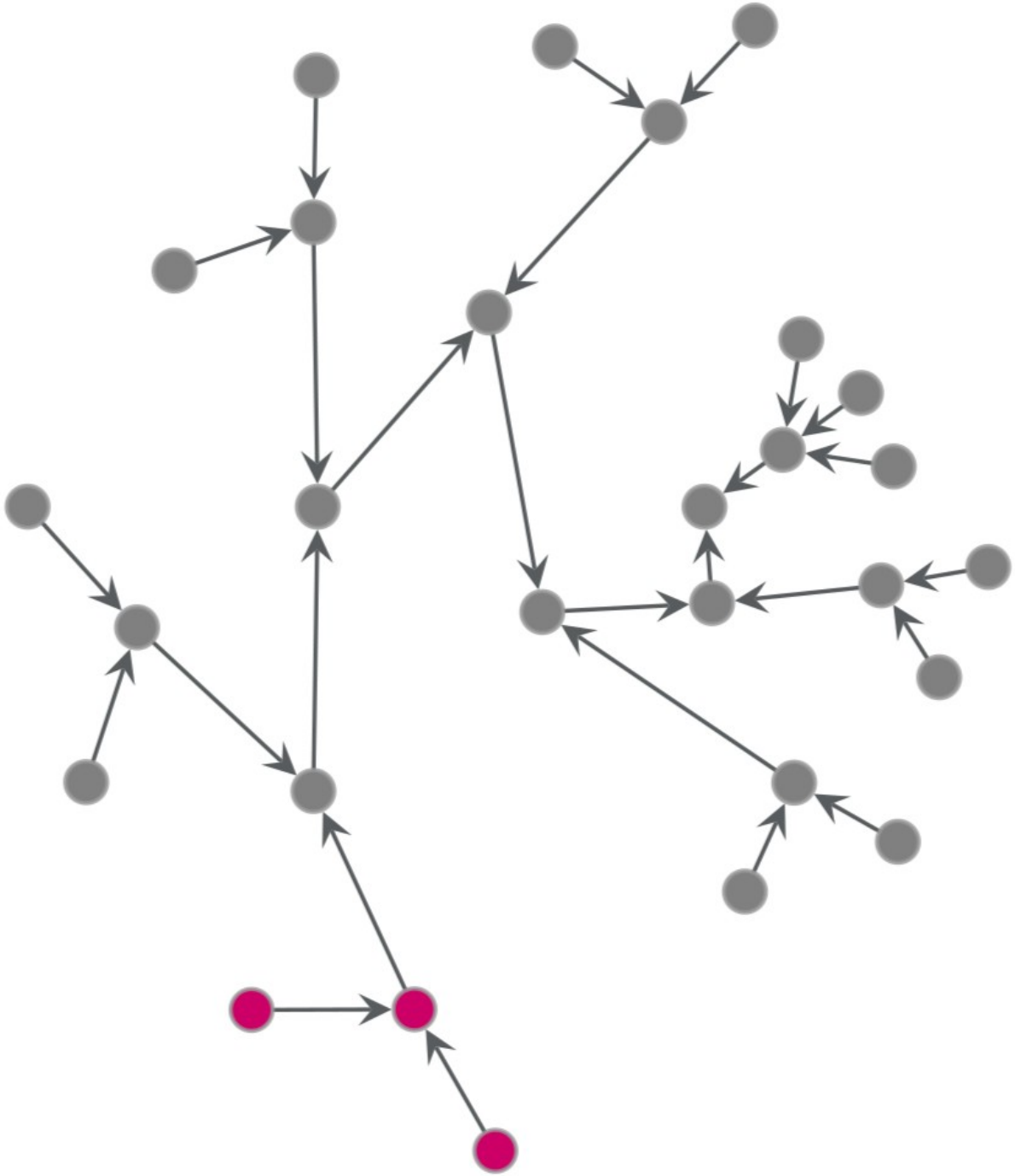
Demo

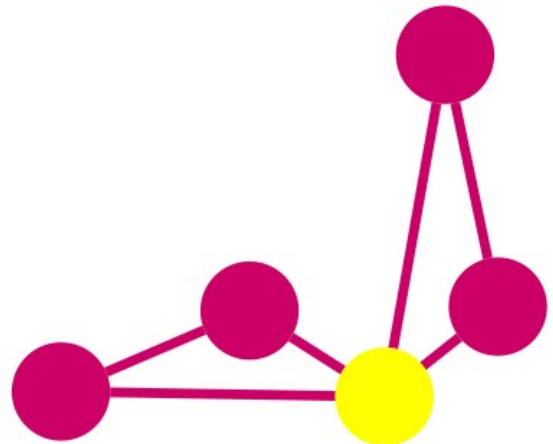


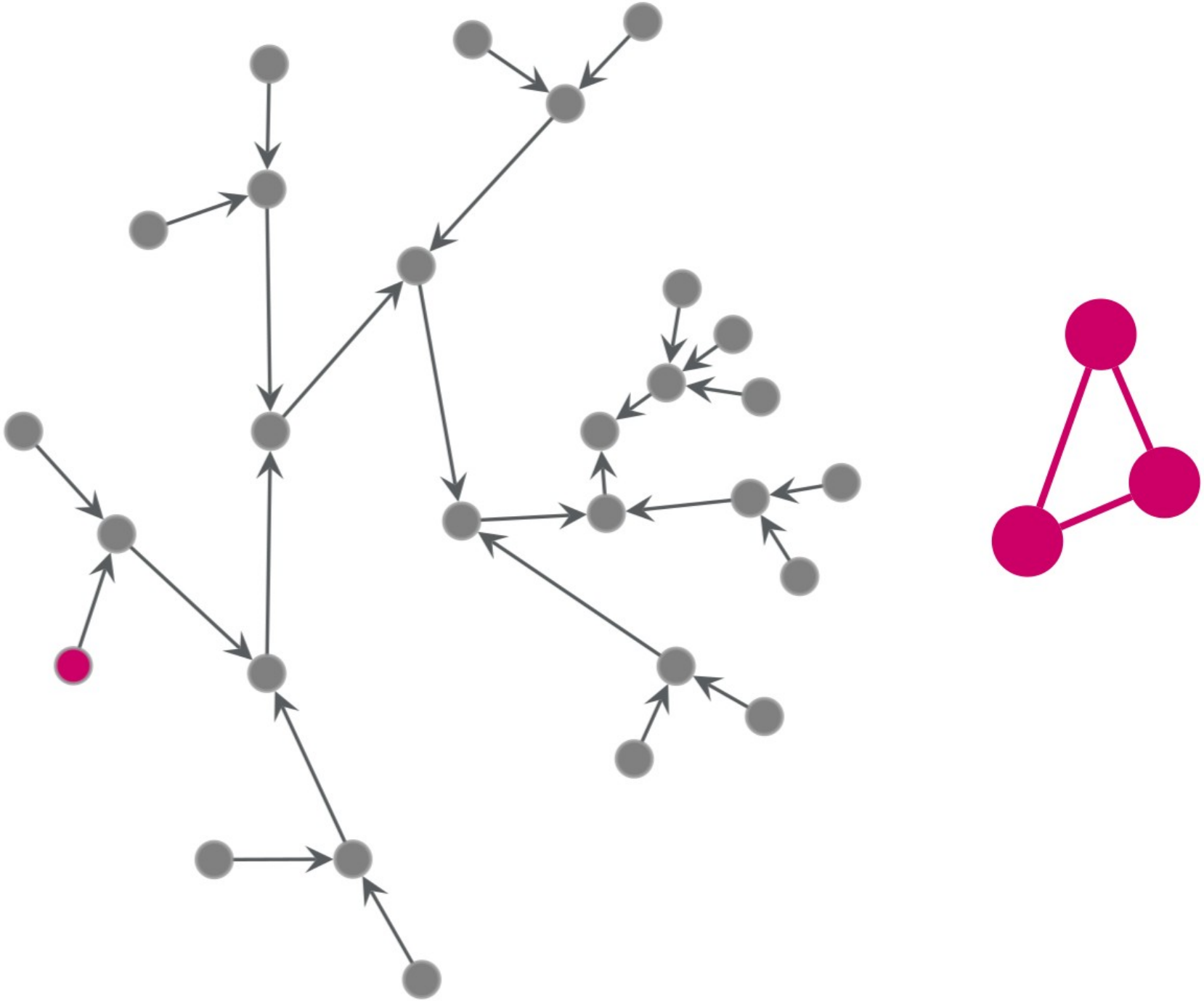


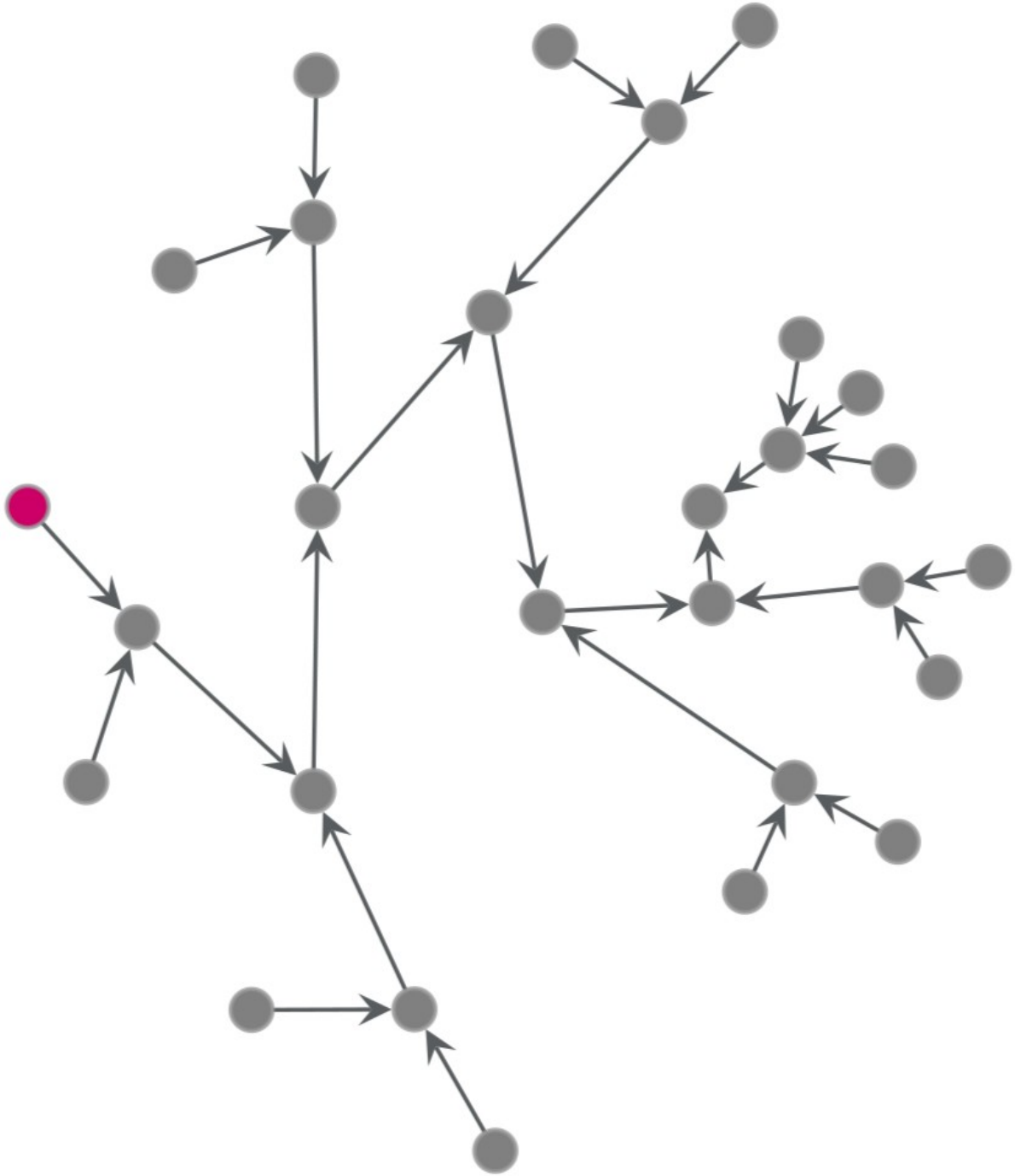


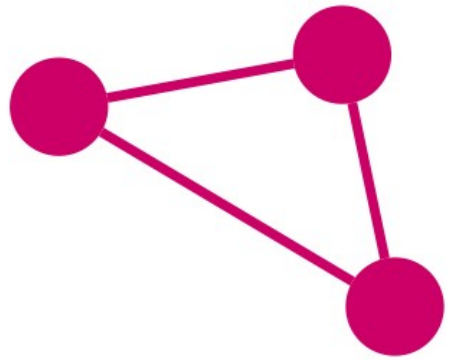


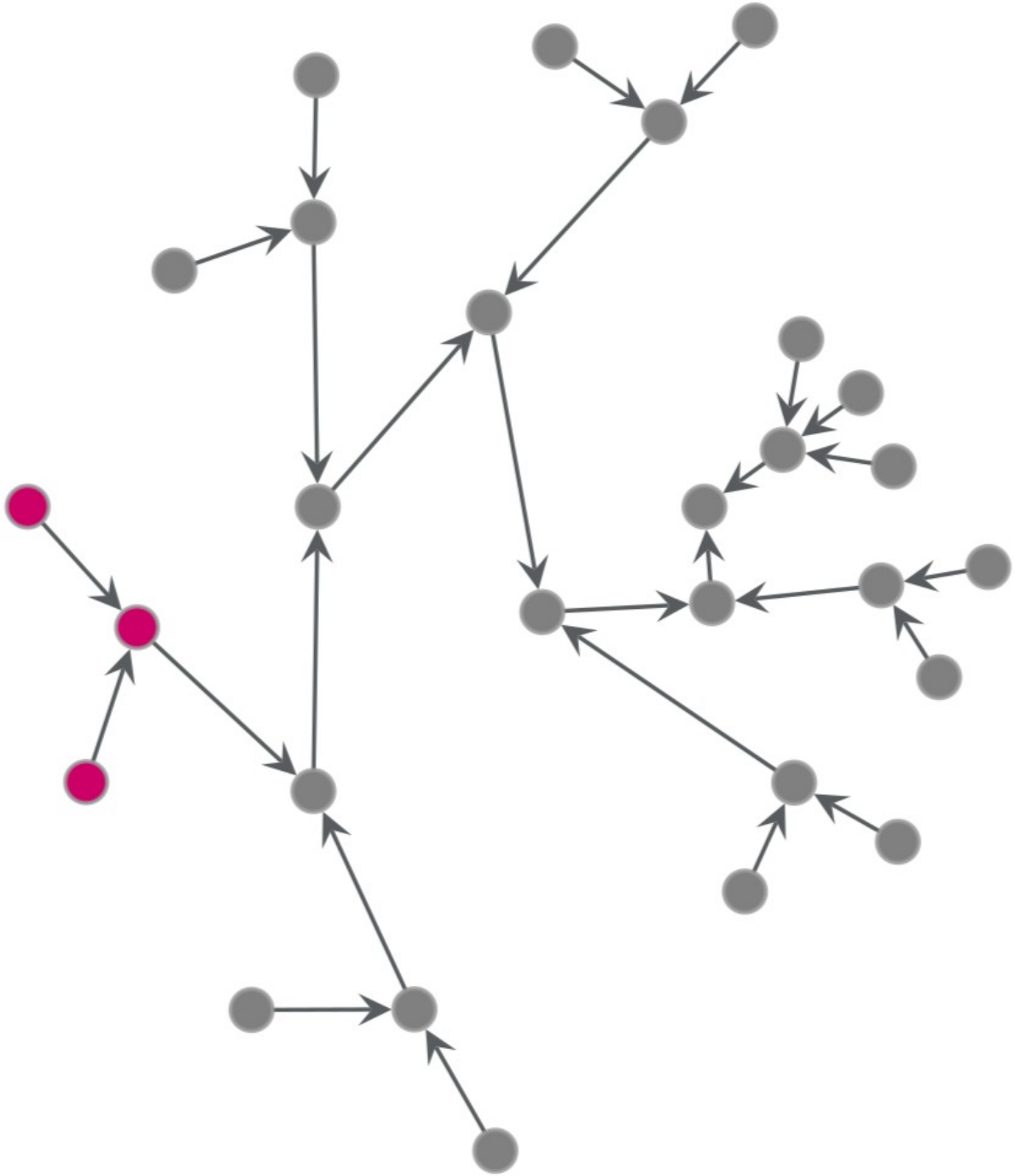


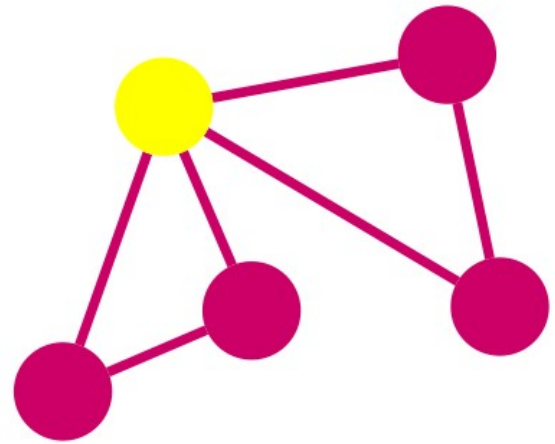


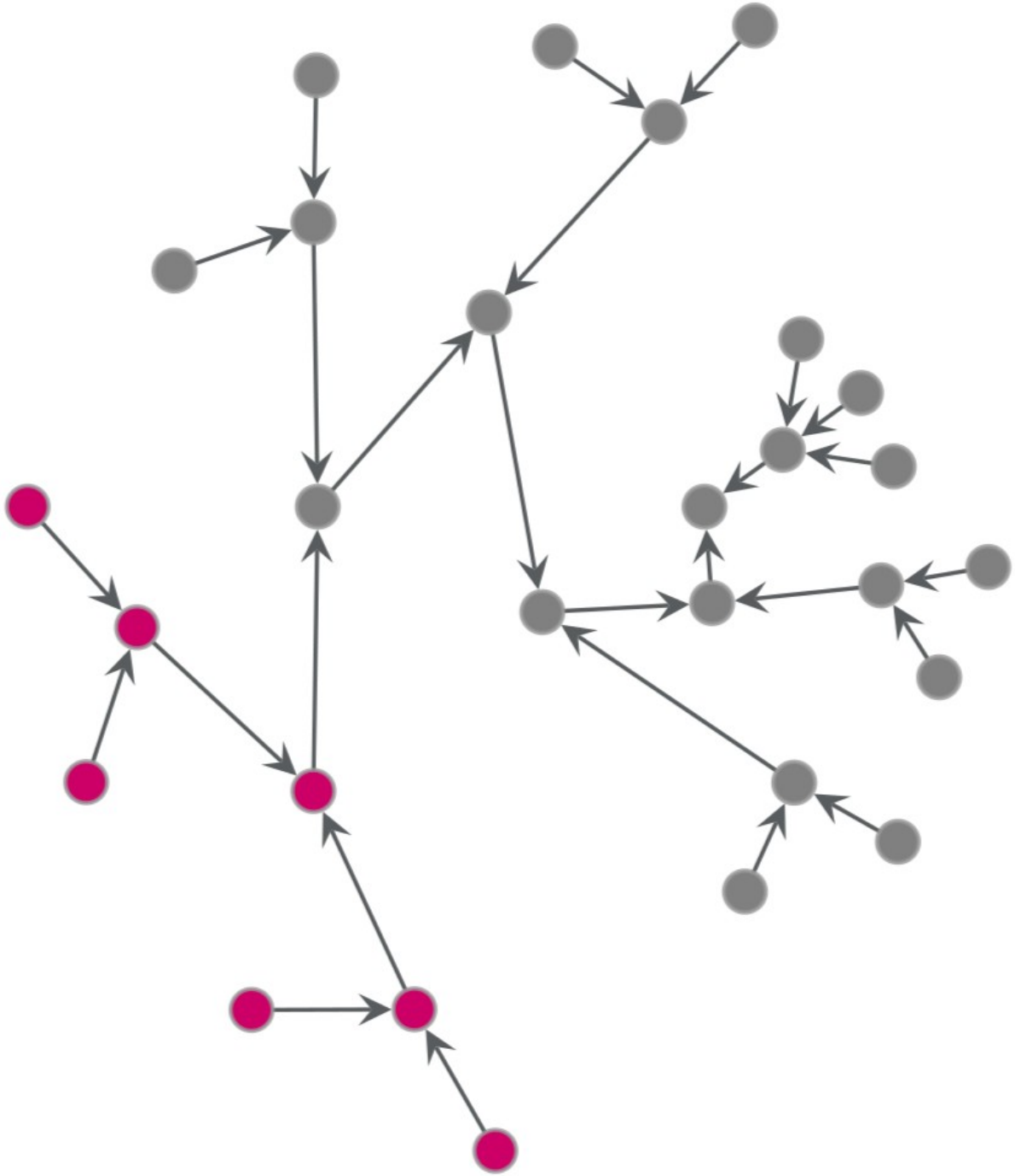


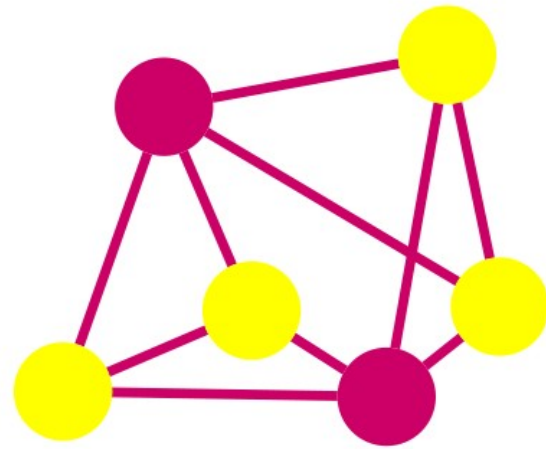


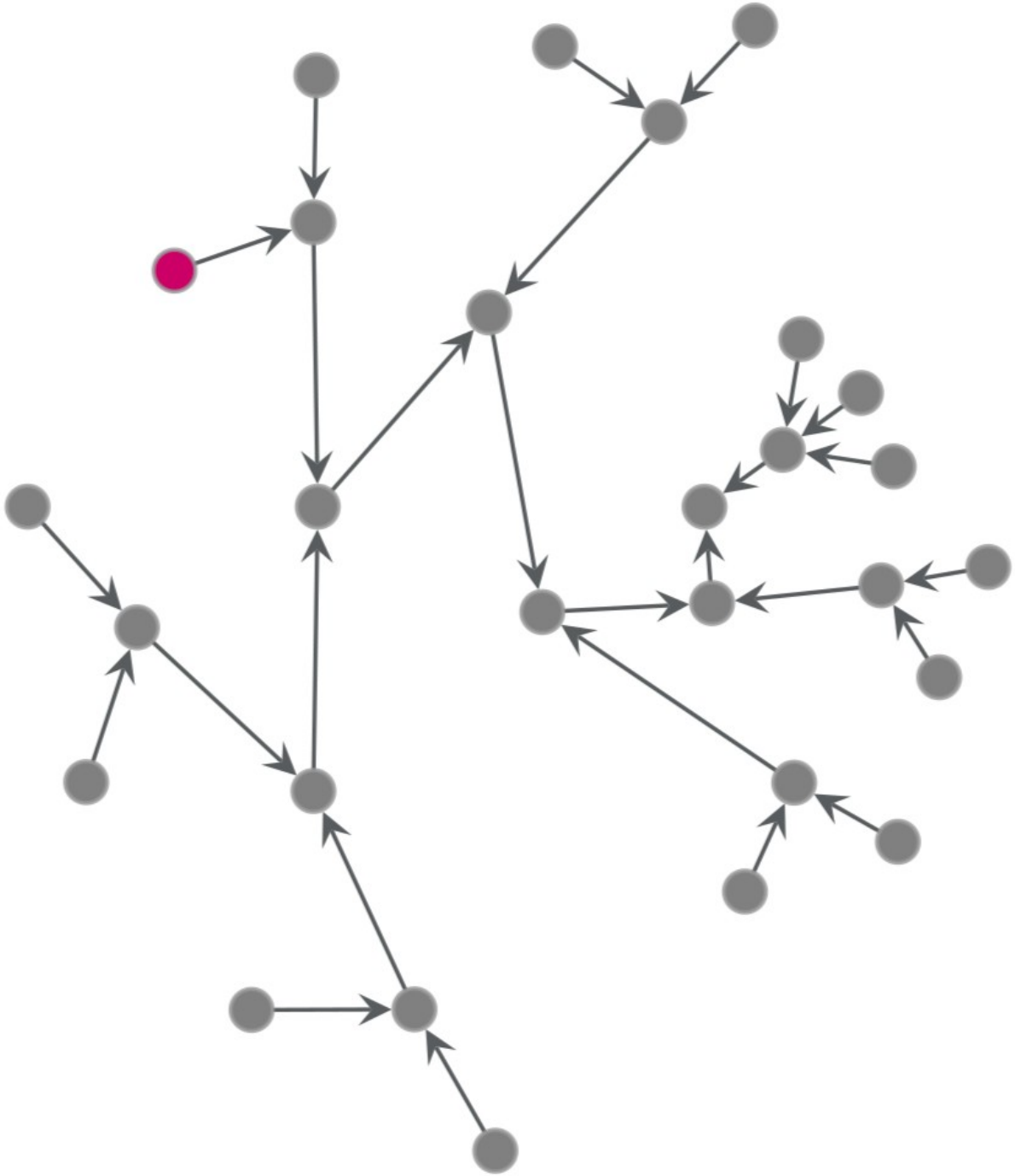


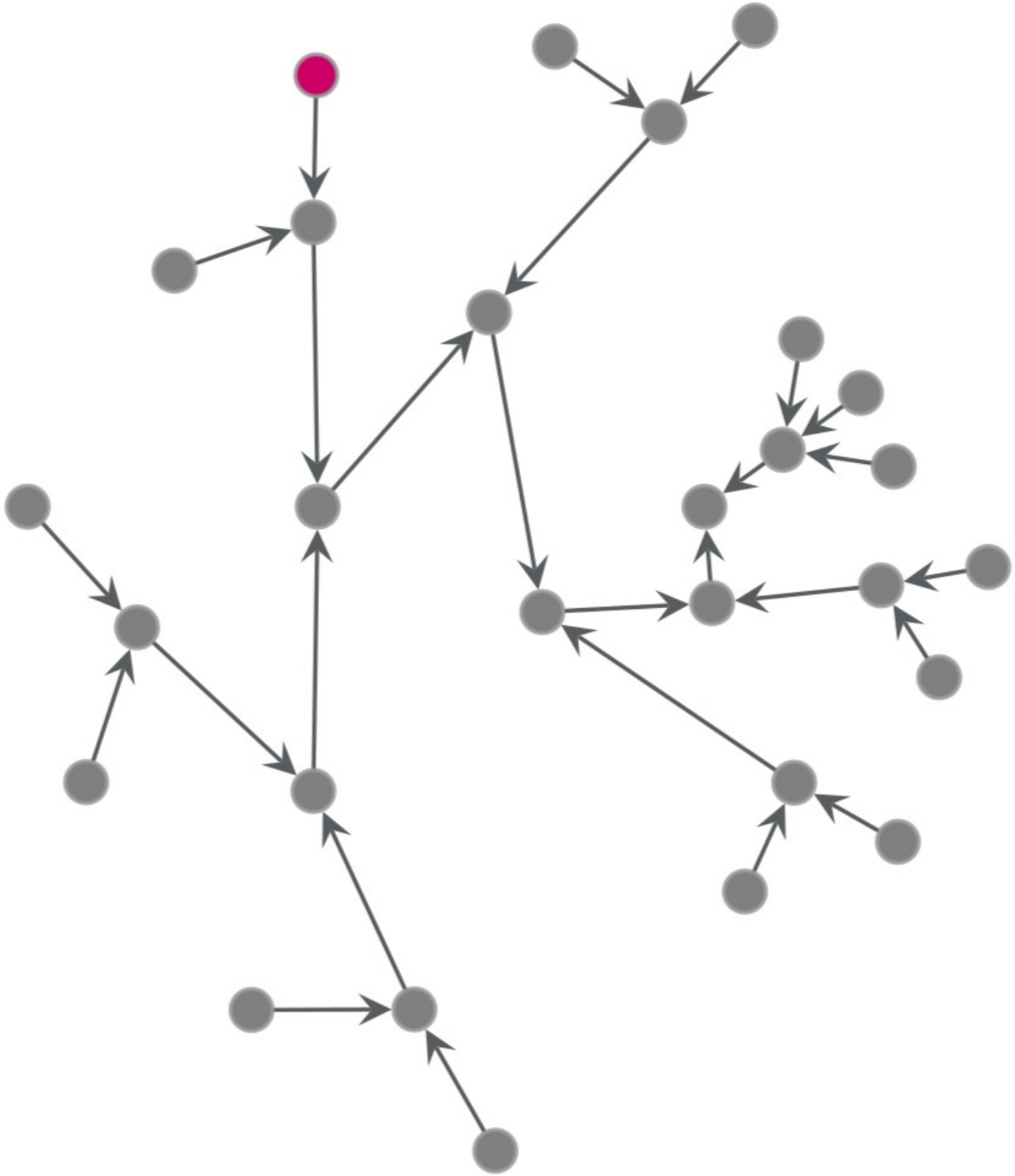


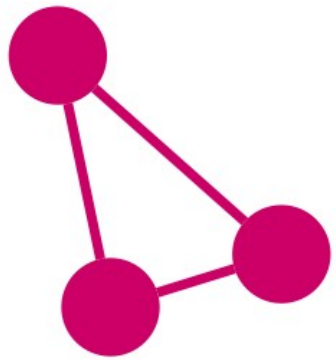


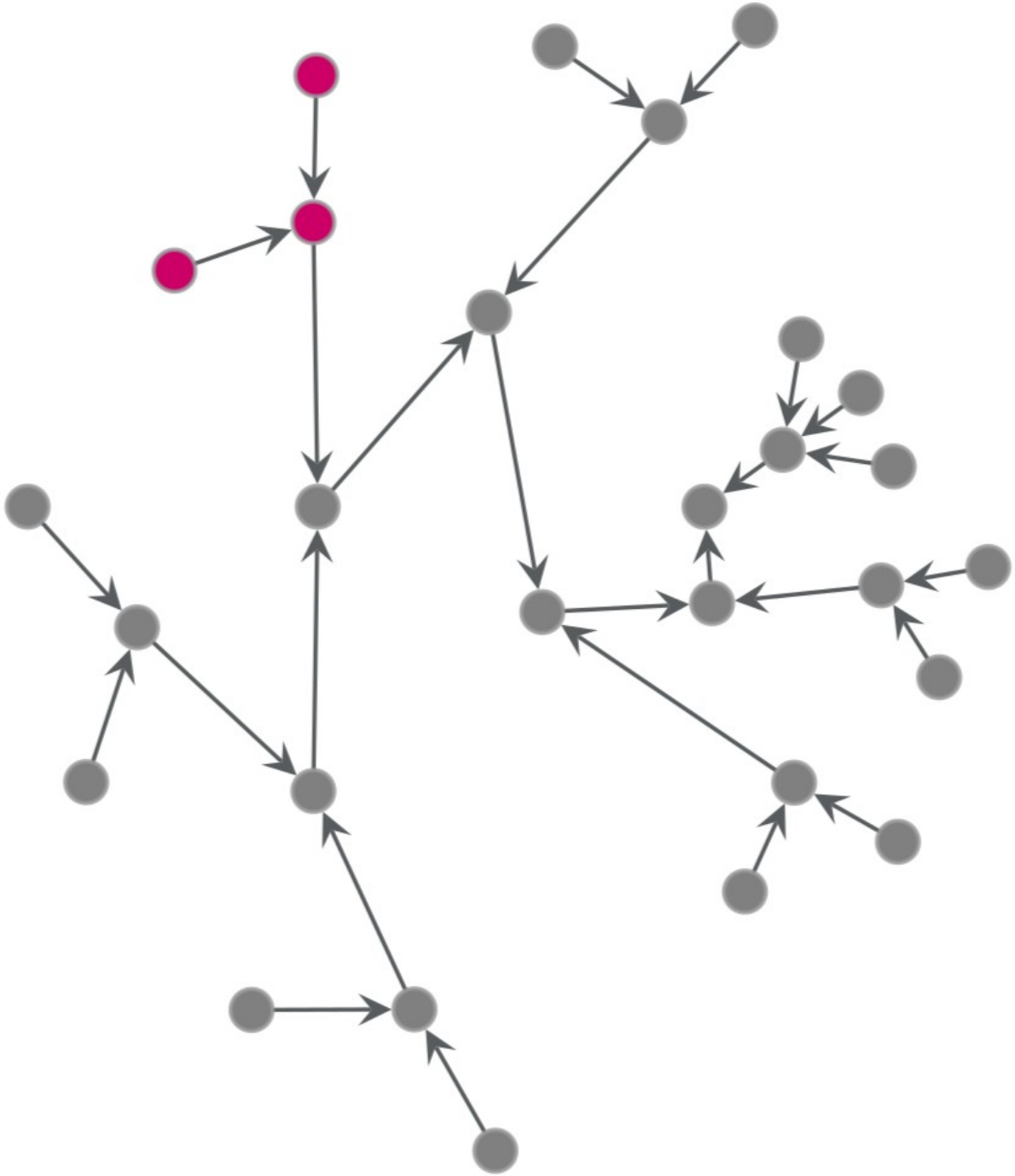


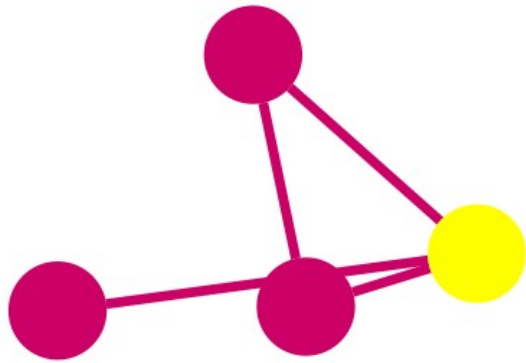


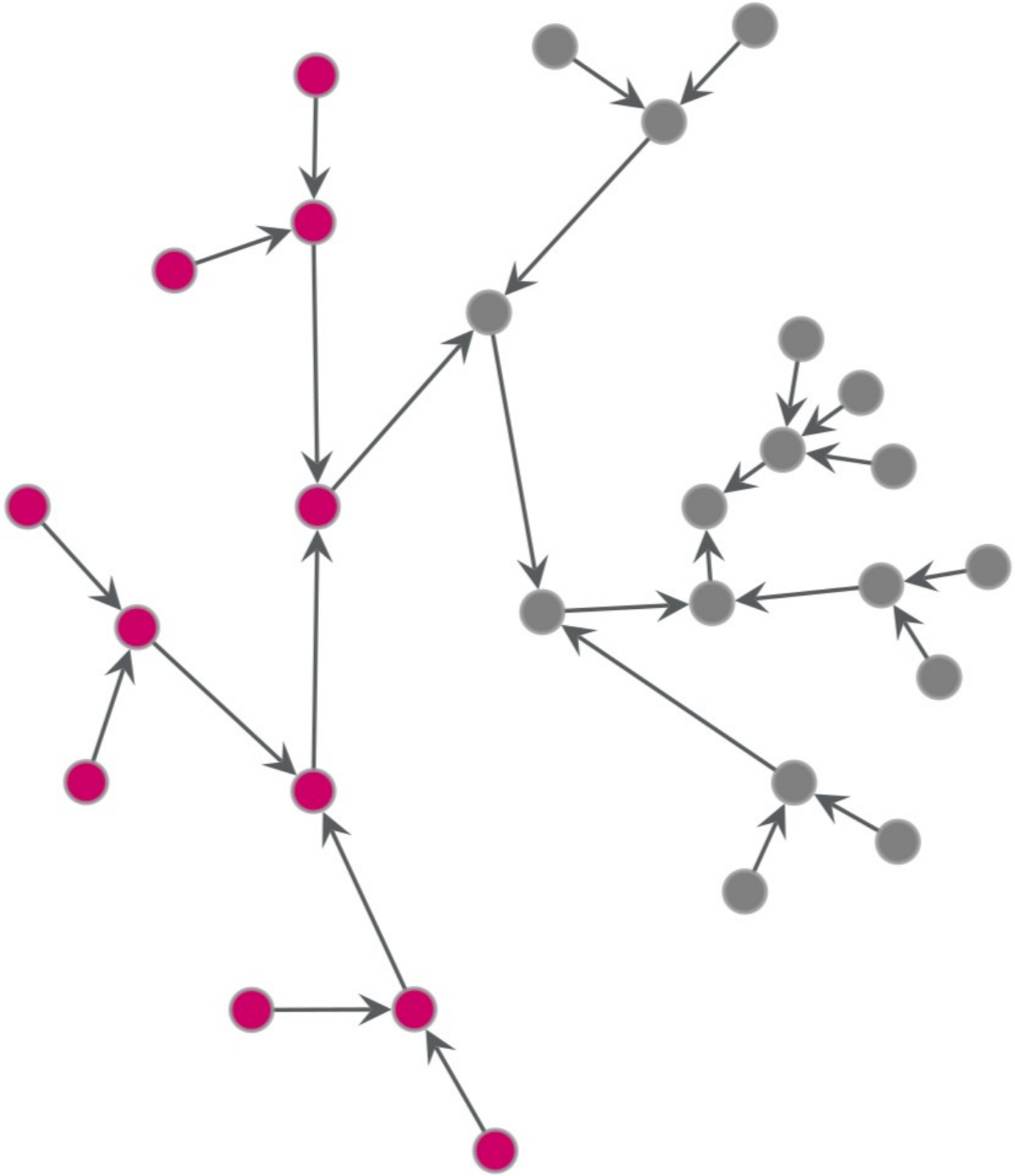


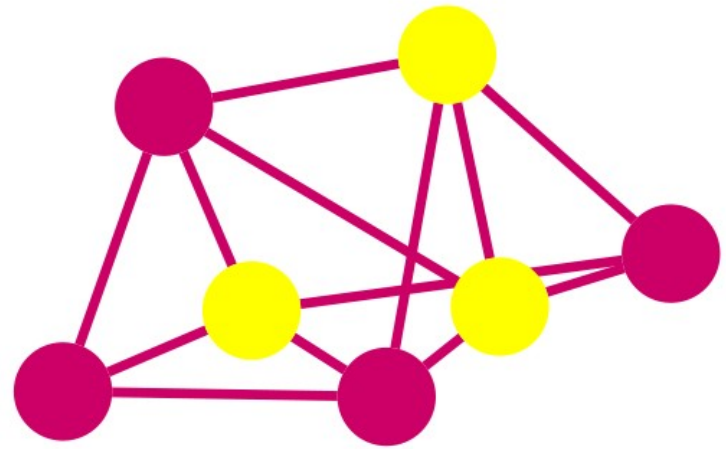


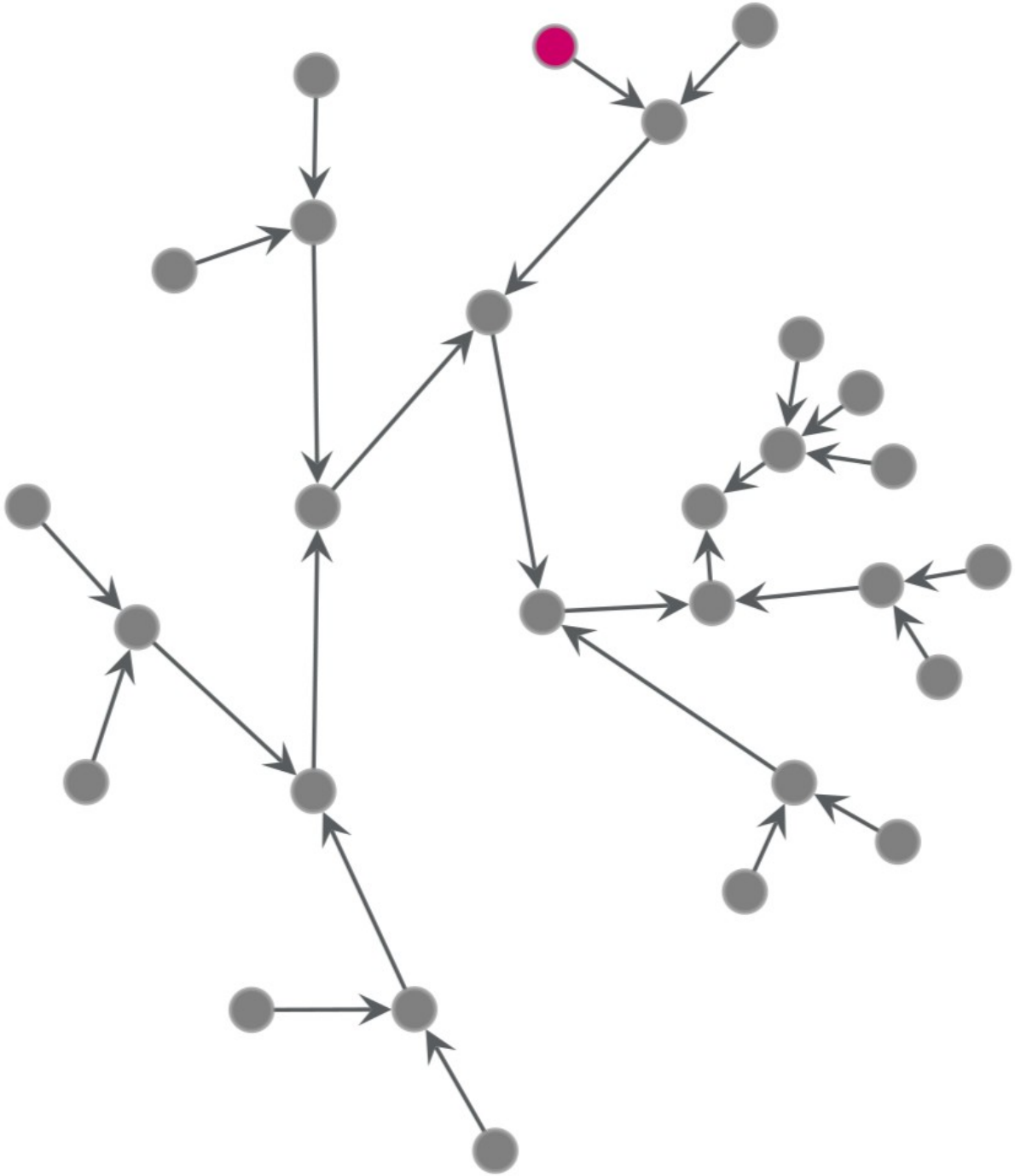


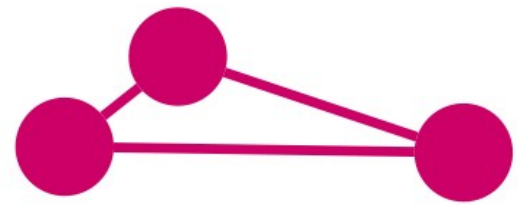


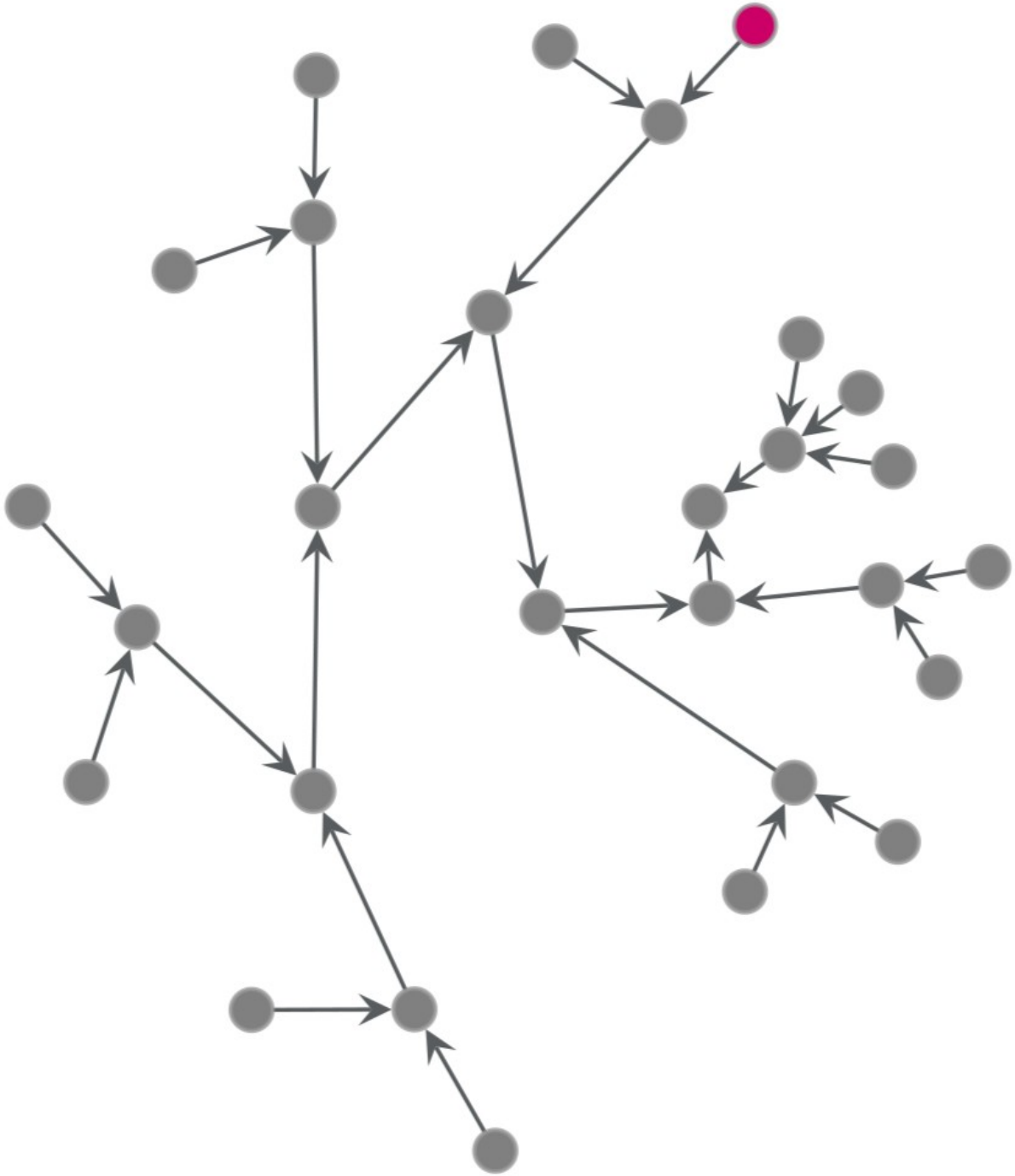


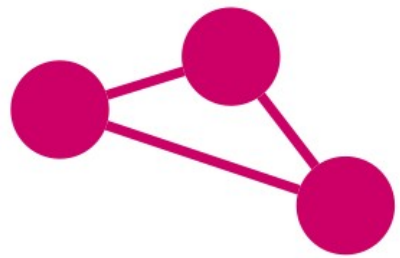


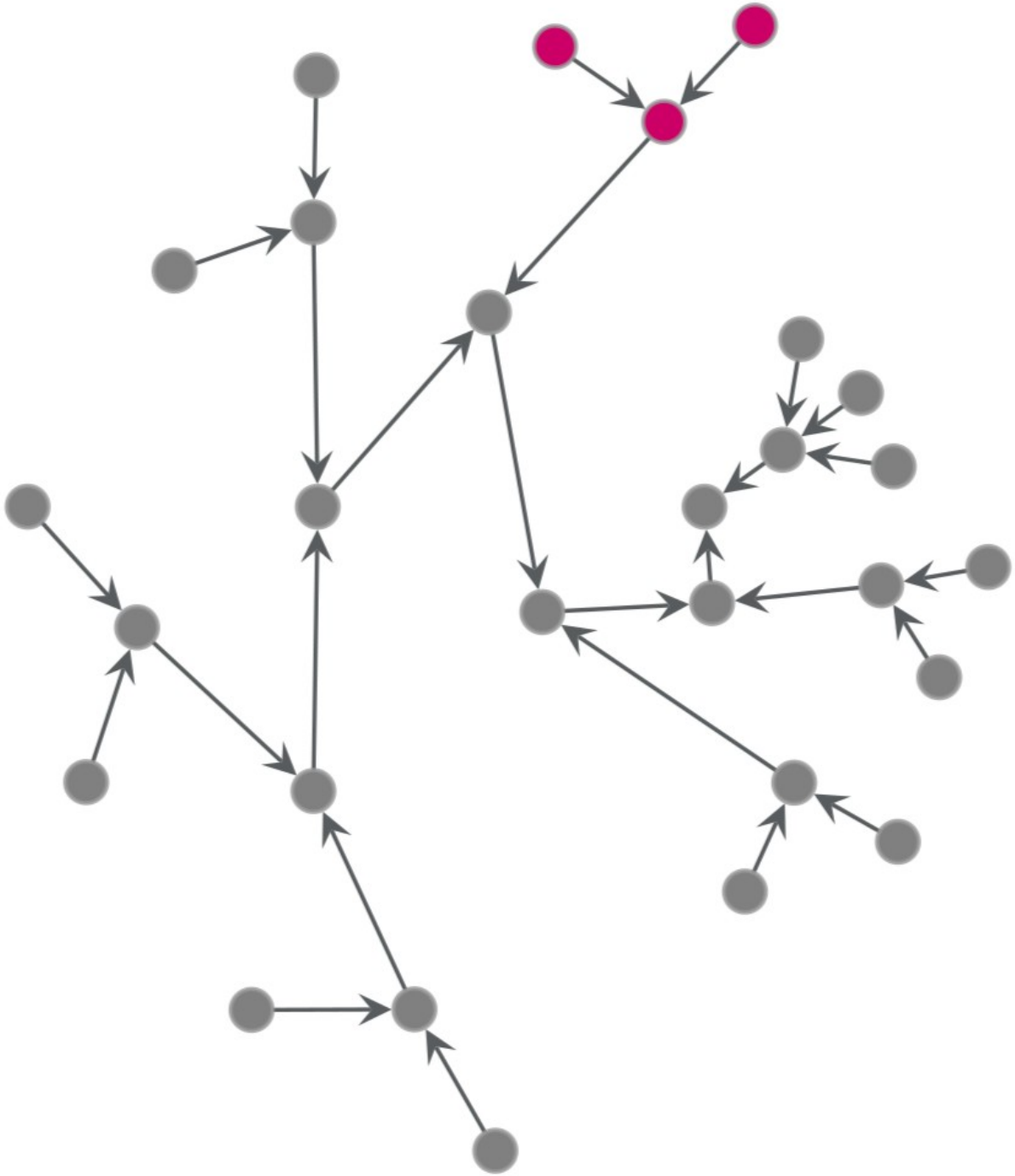


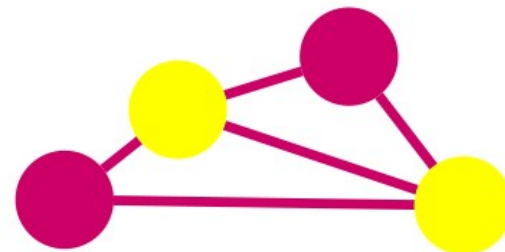


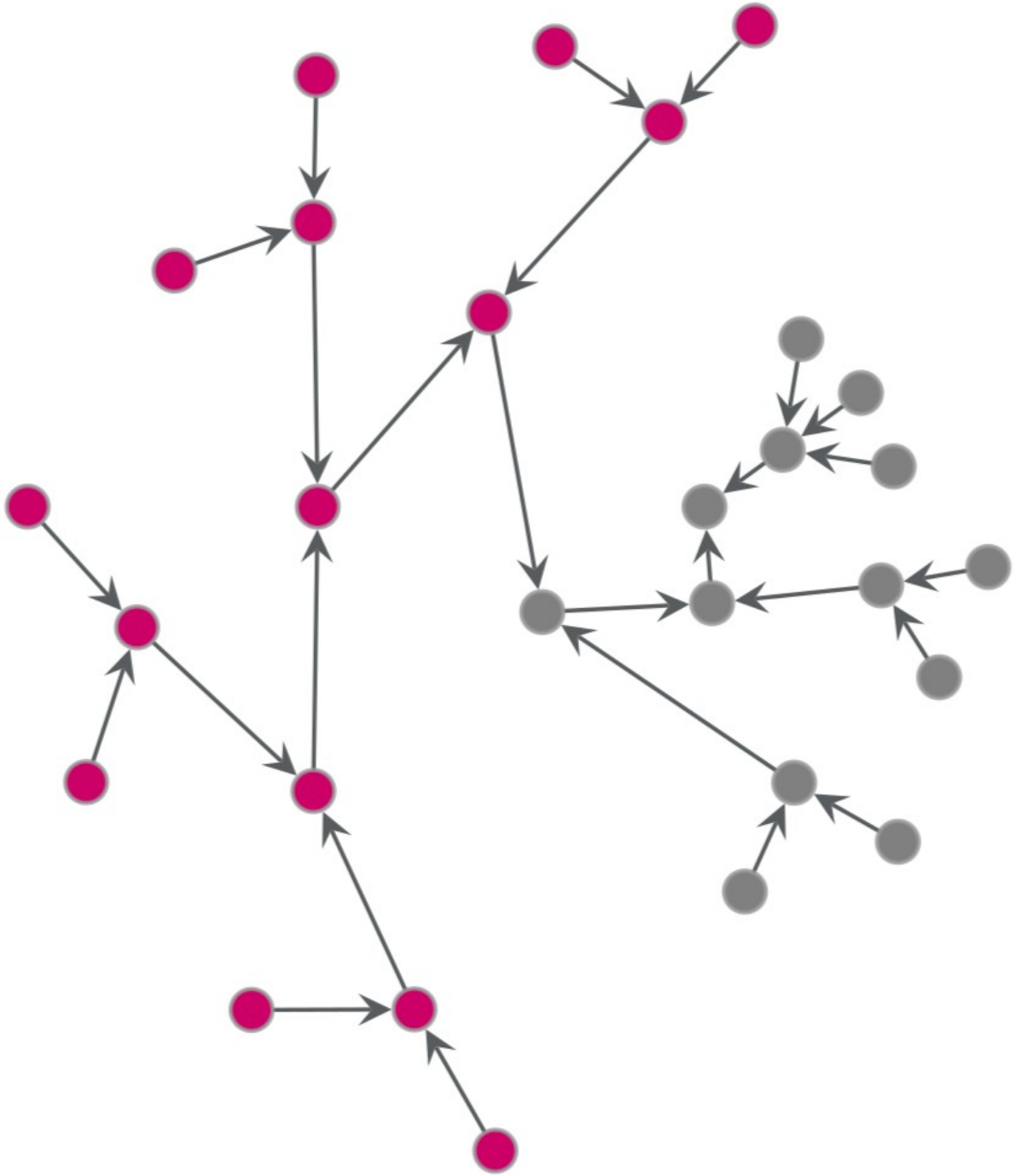


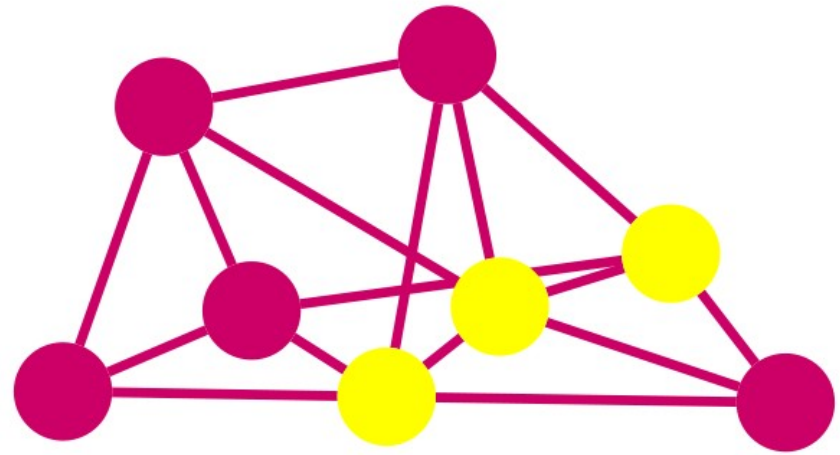


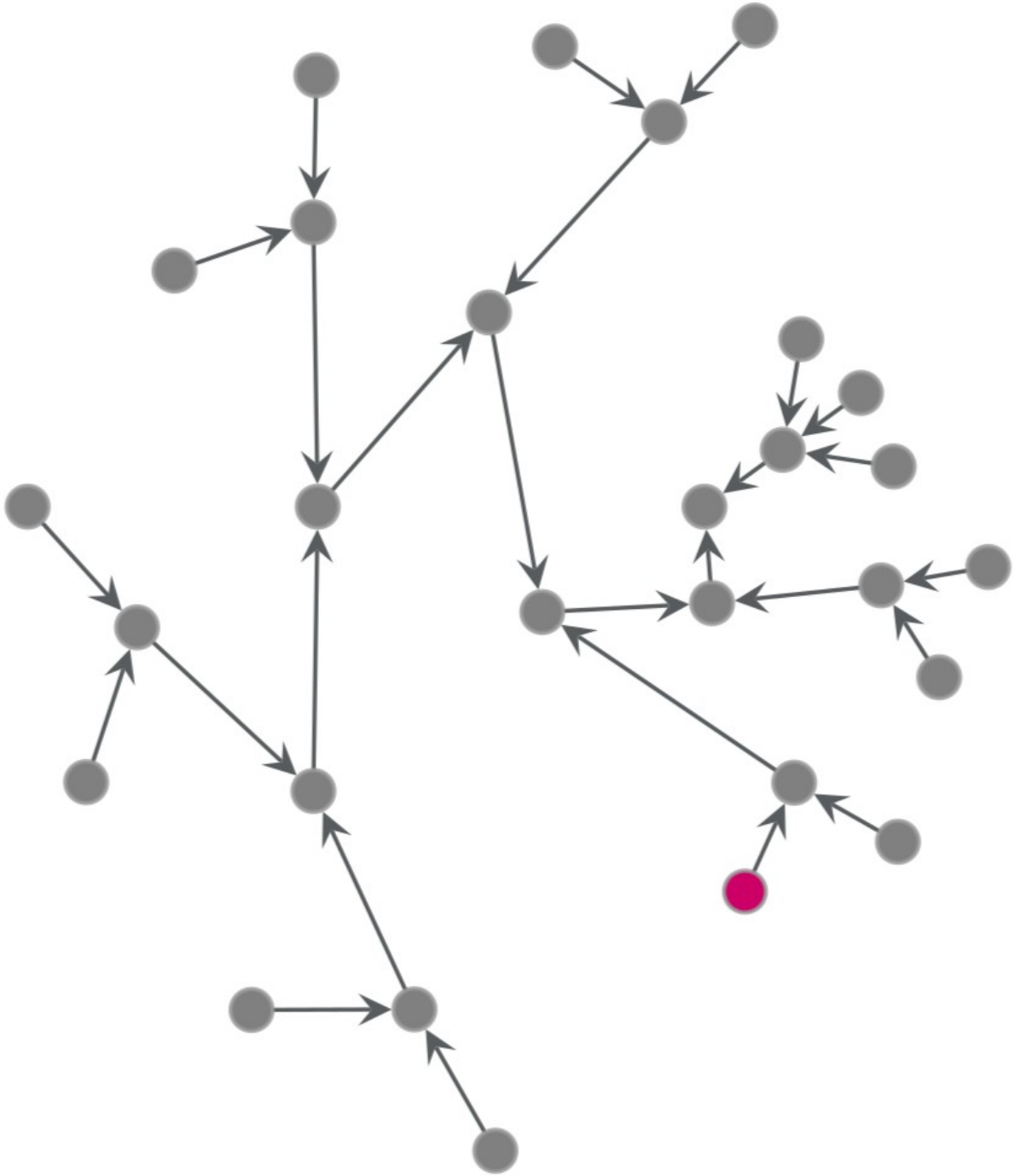


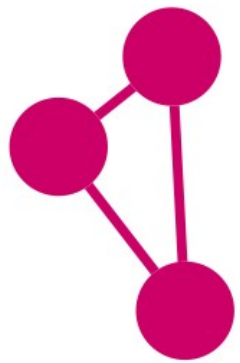


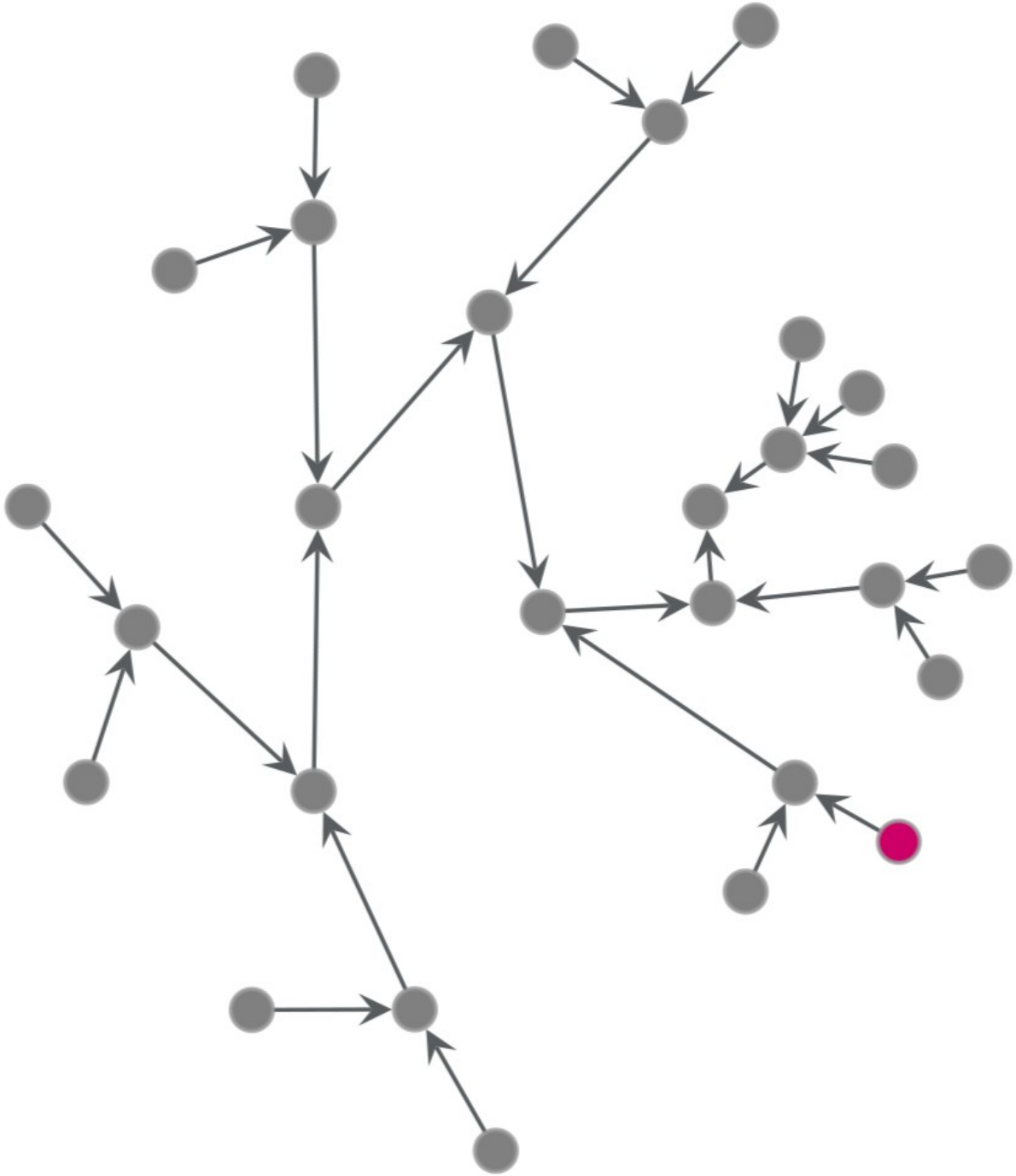


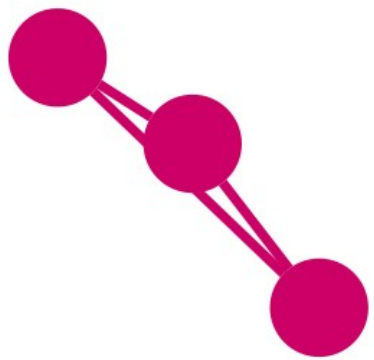


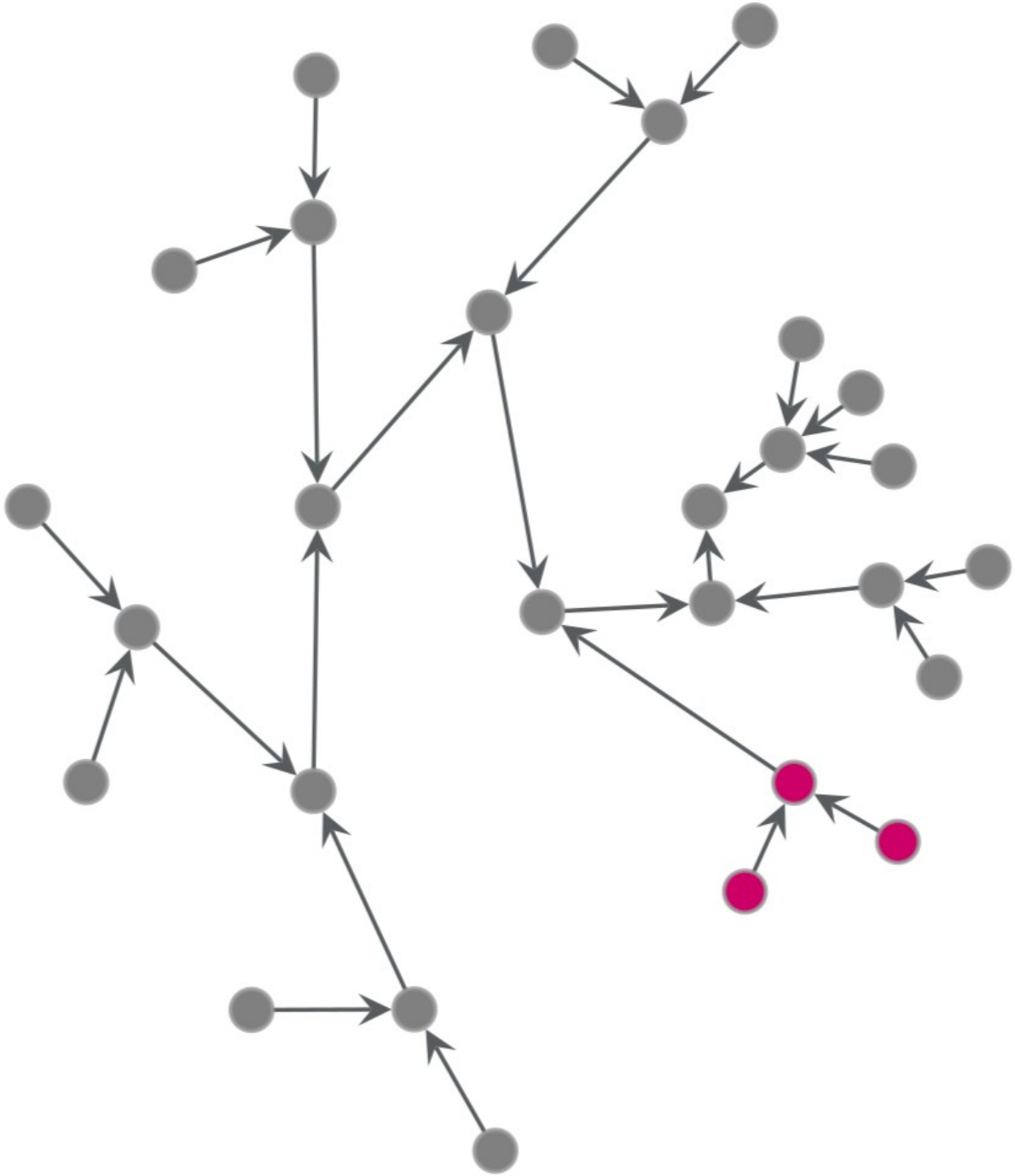


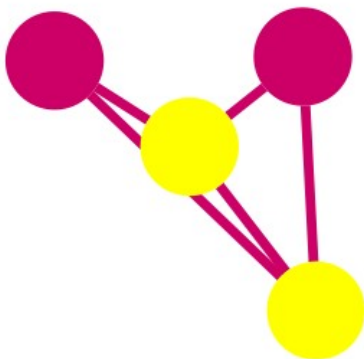


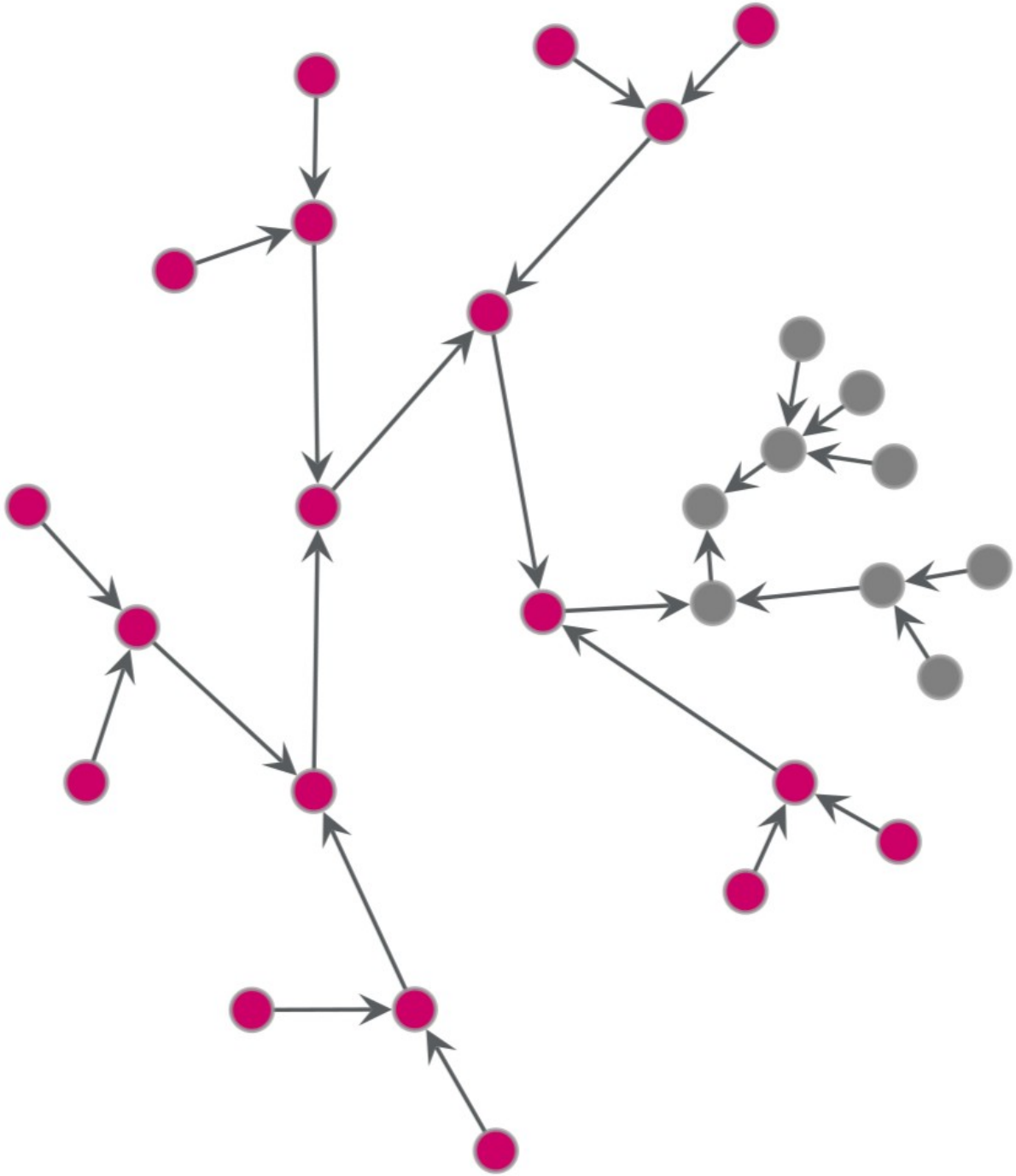


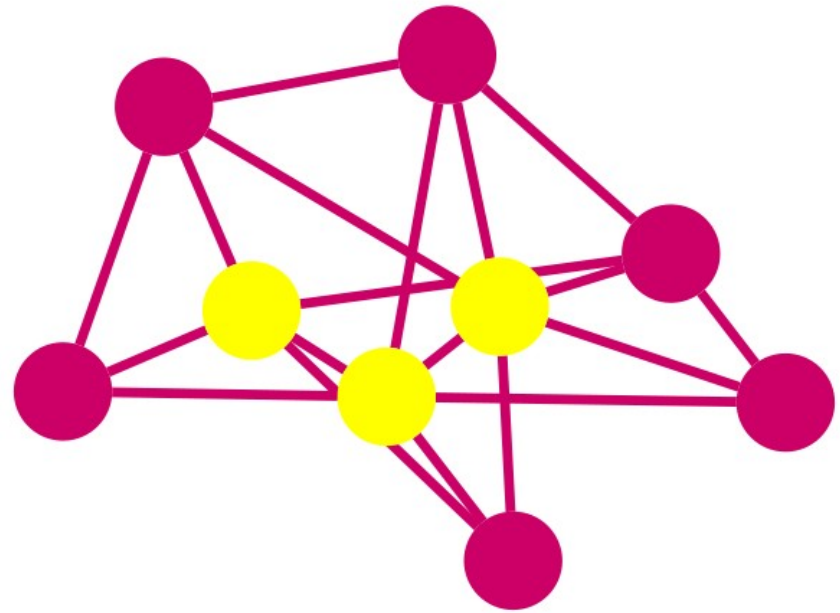


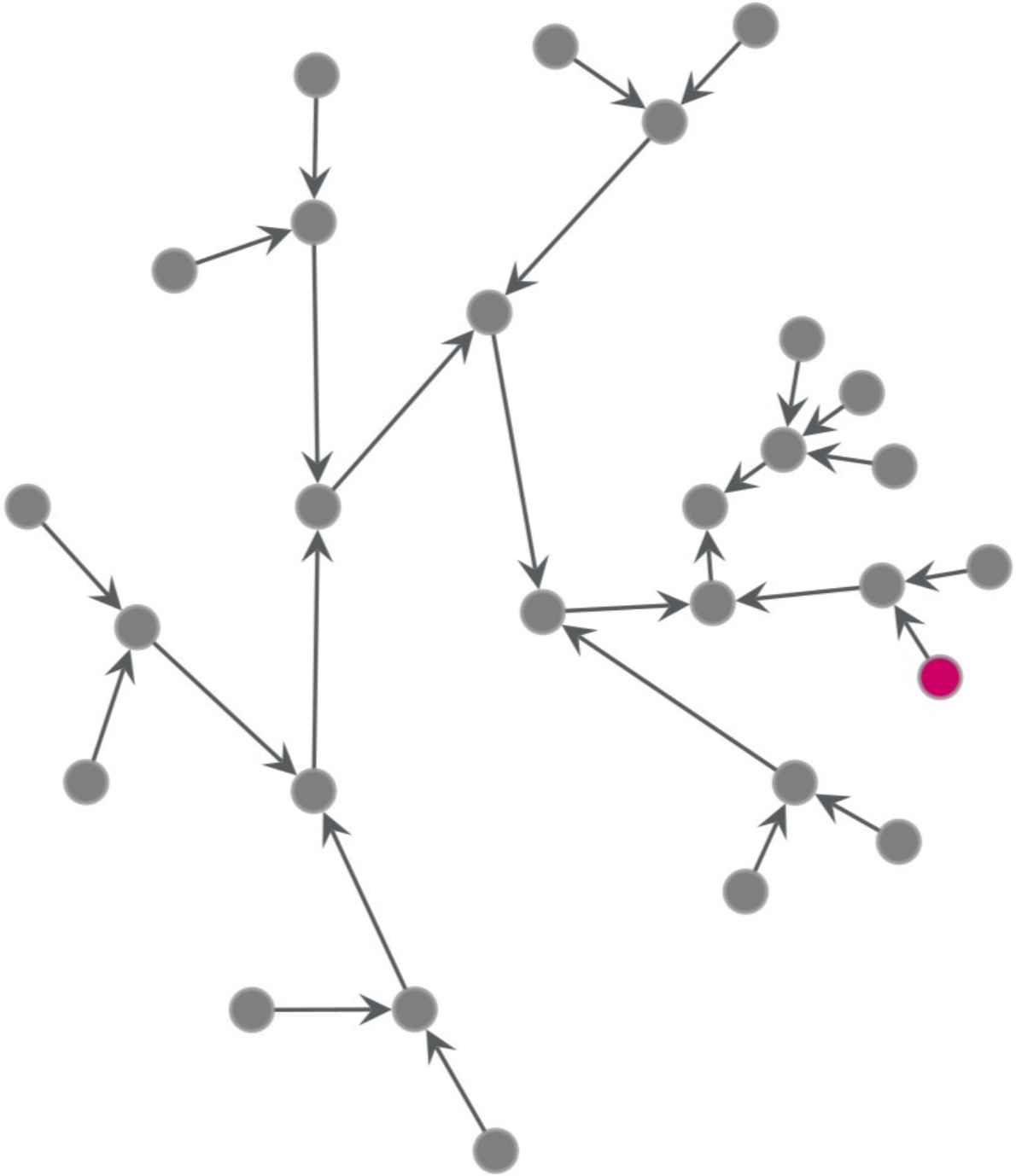


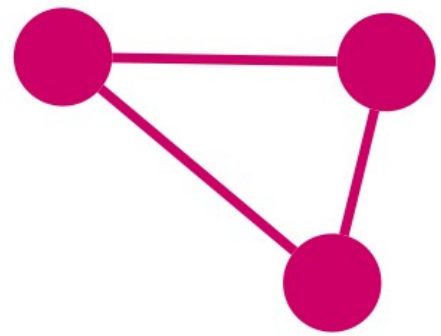


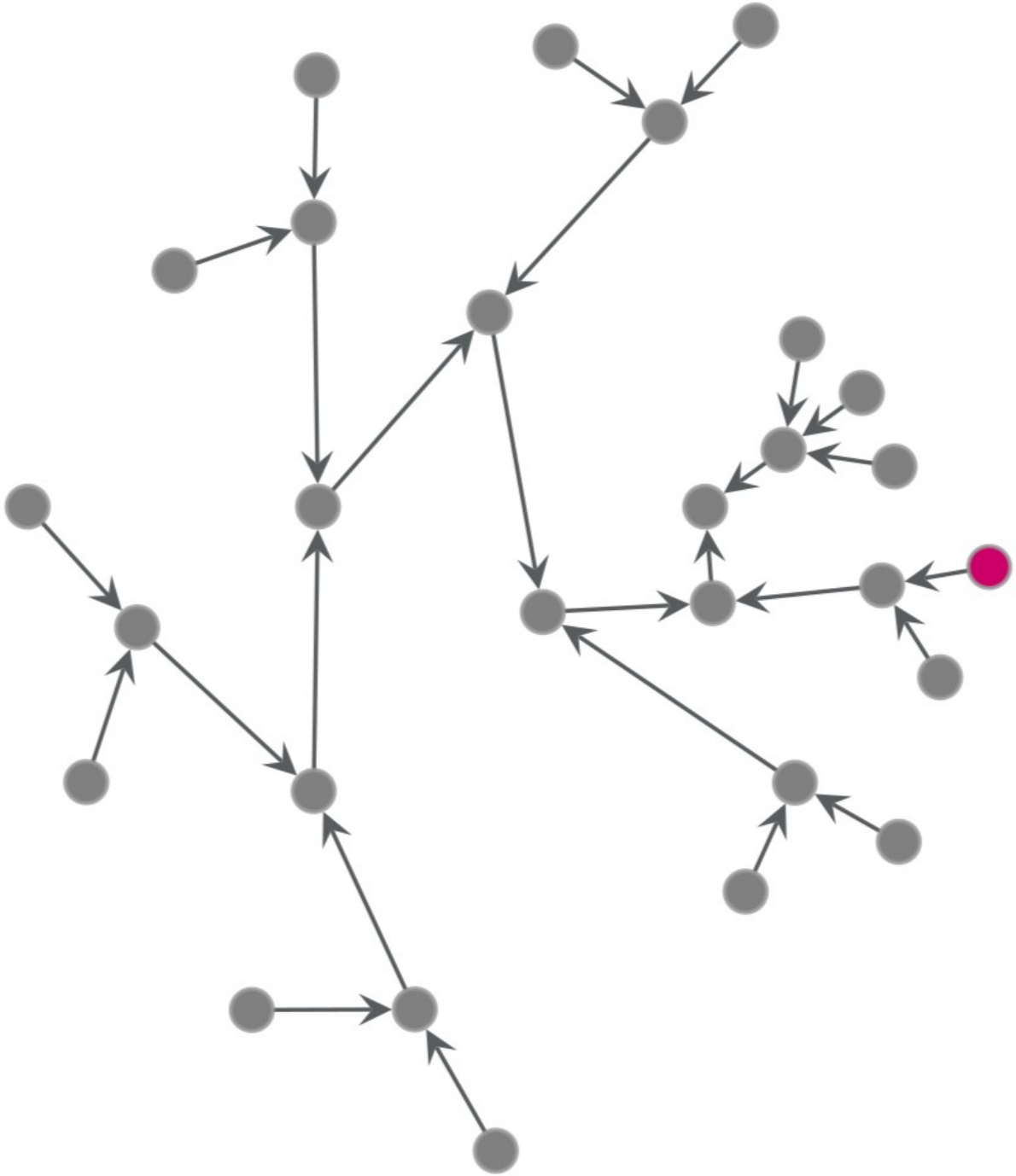


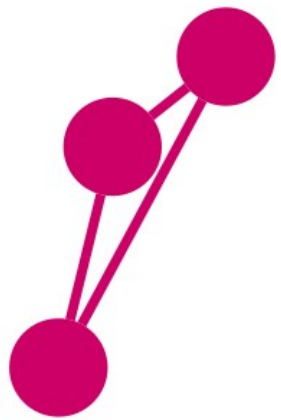


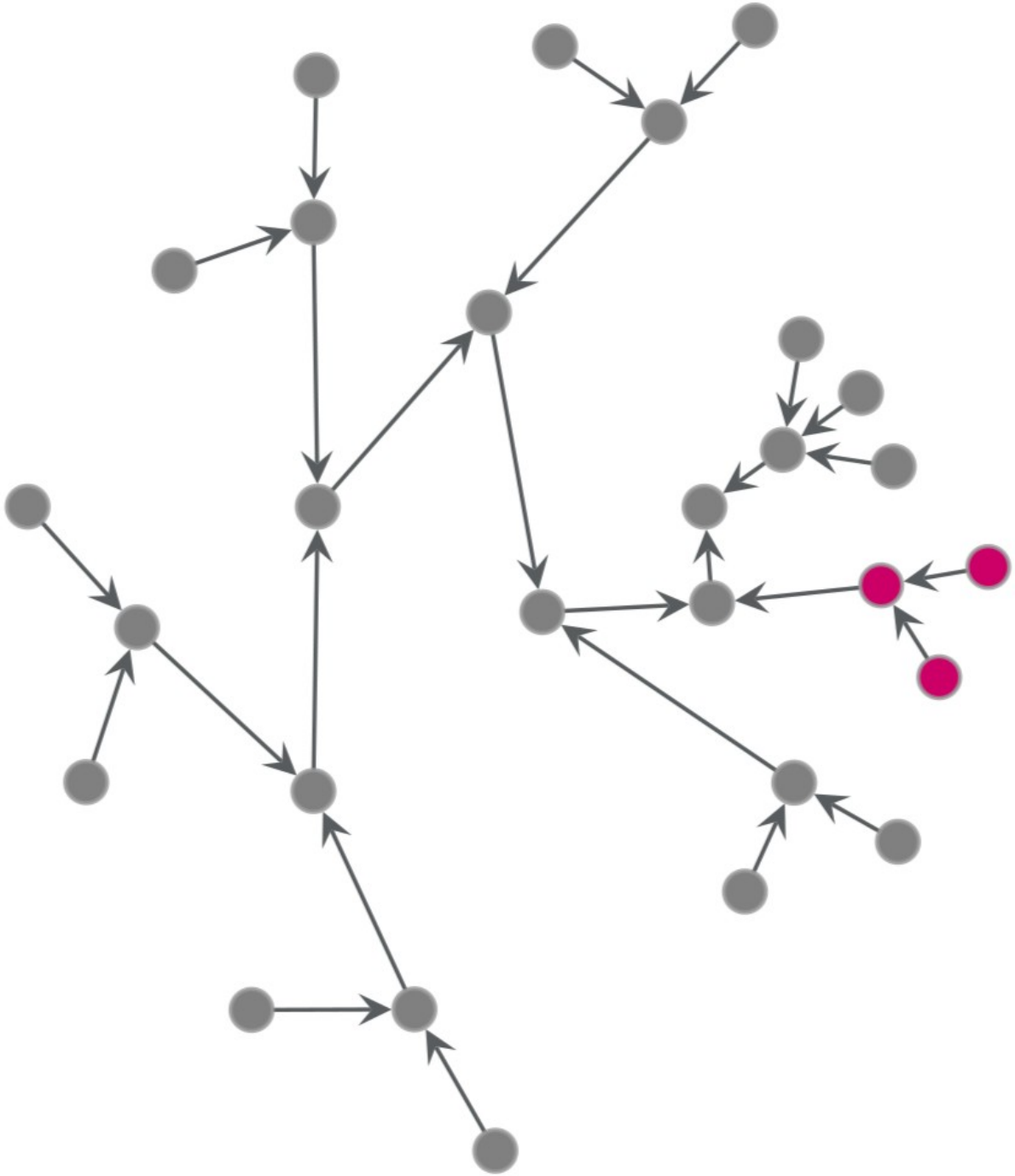


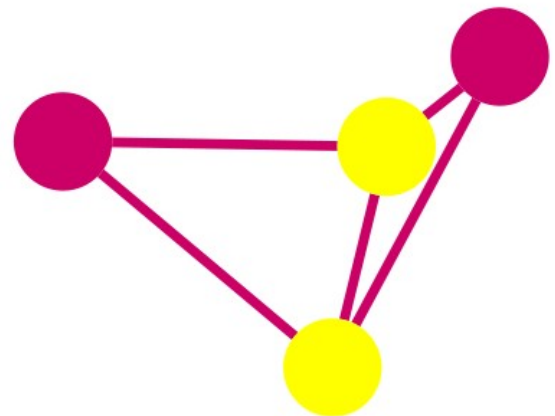


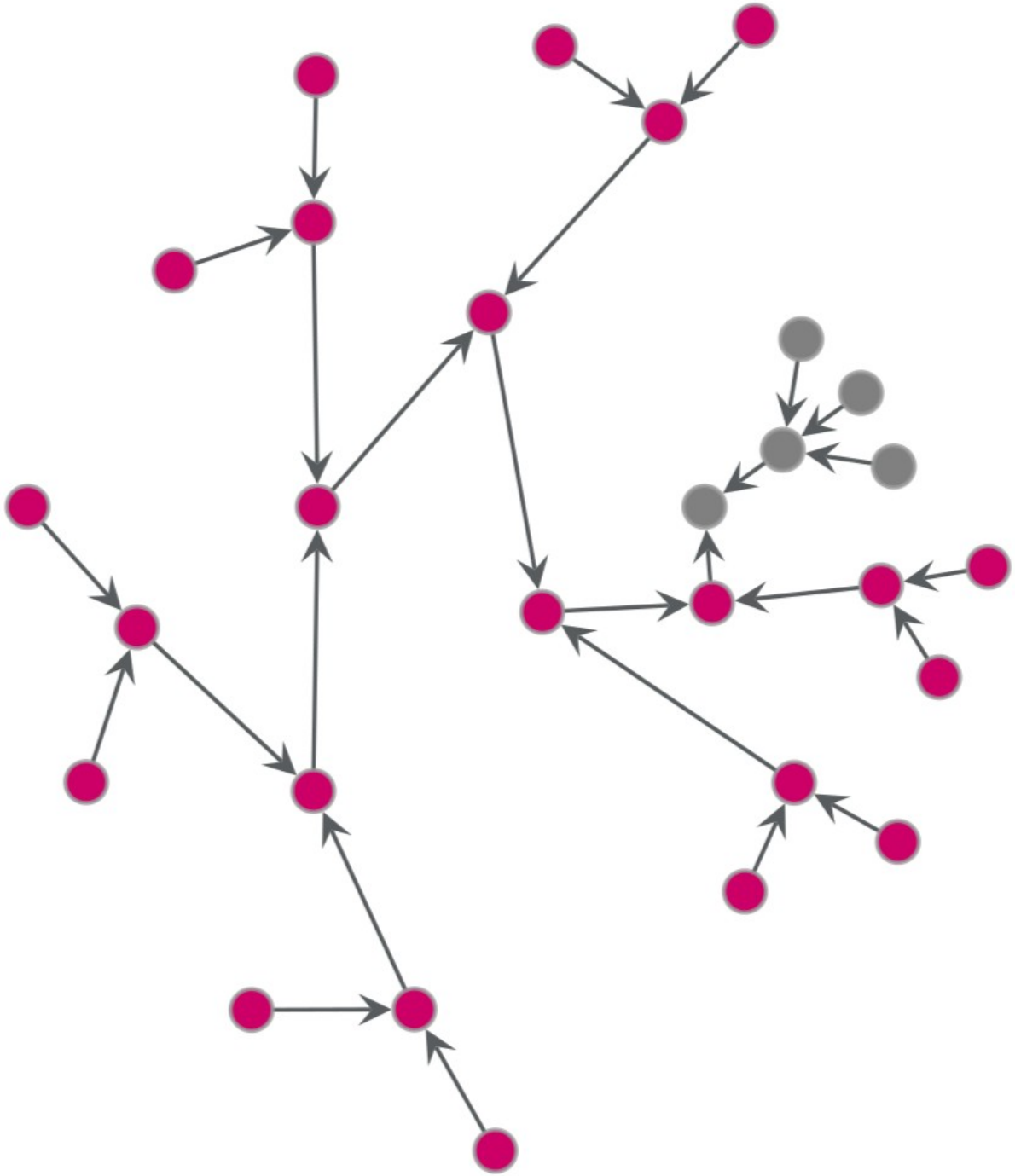


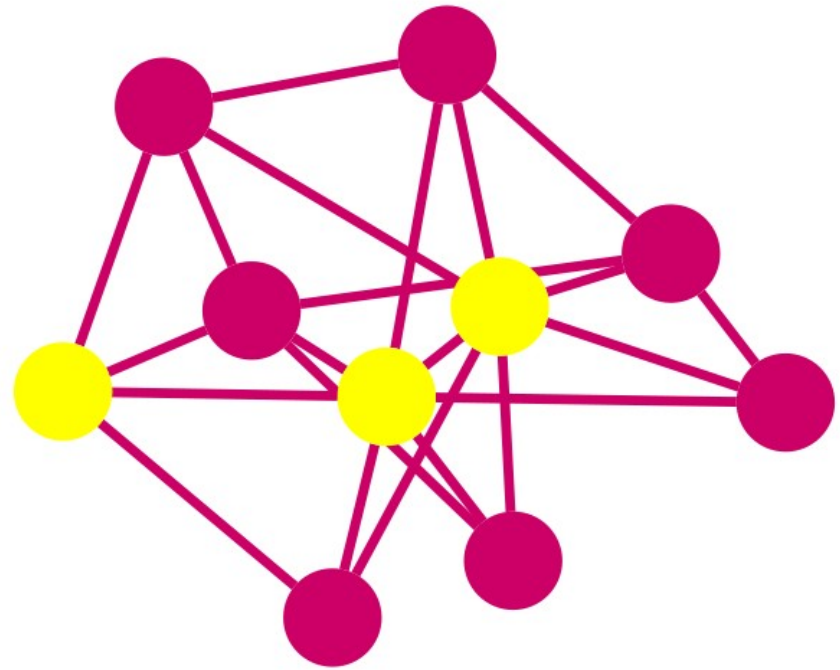


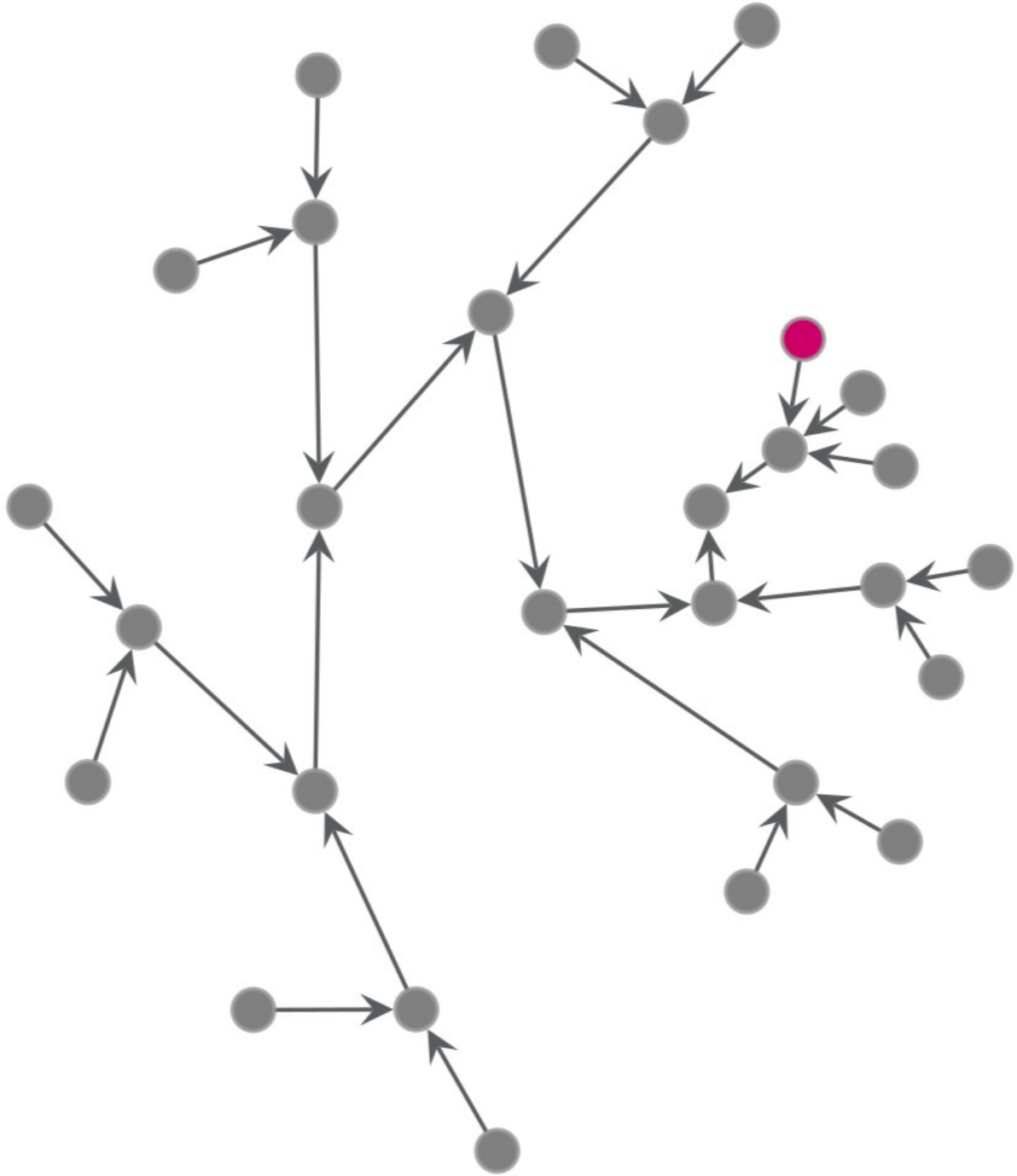


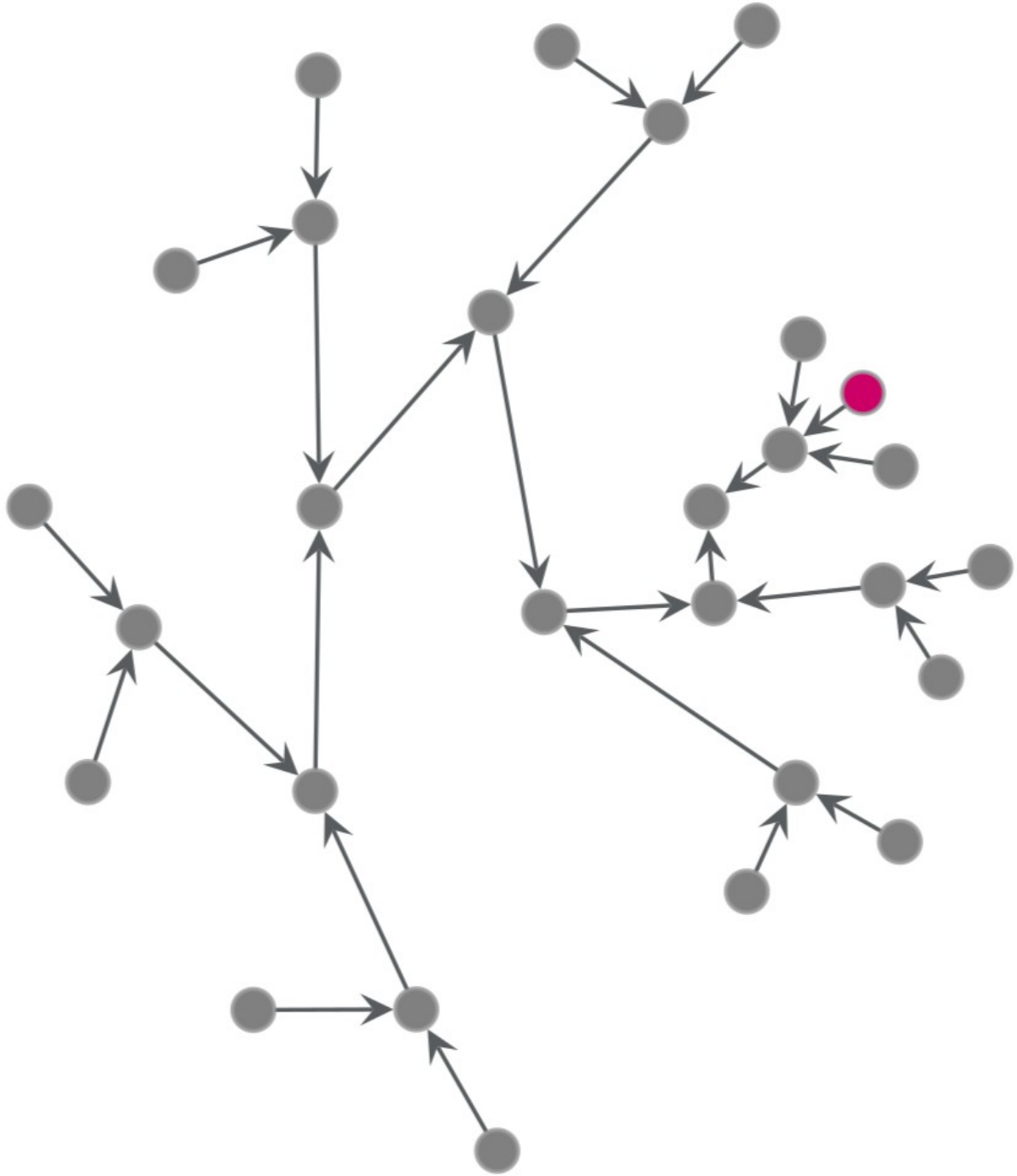


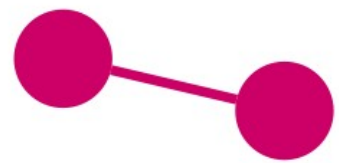


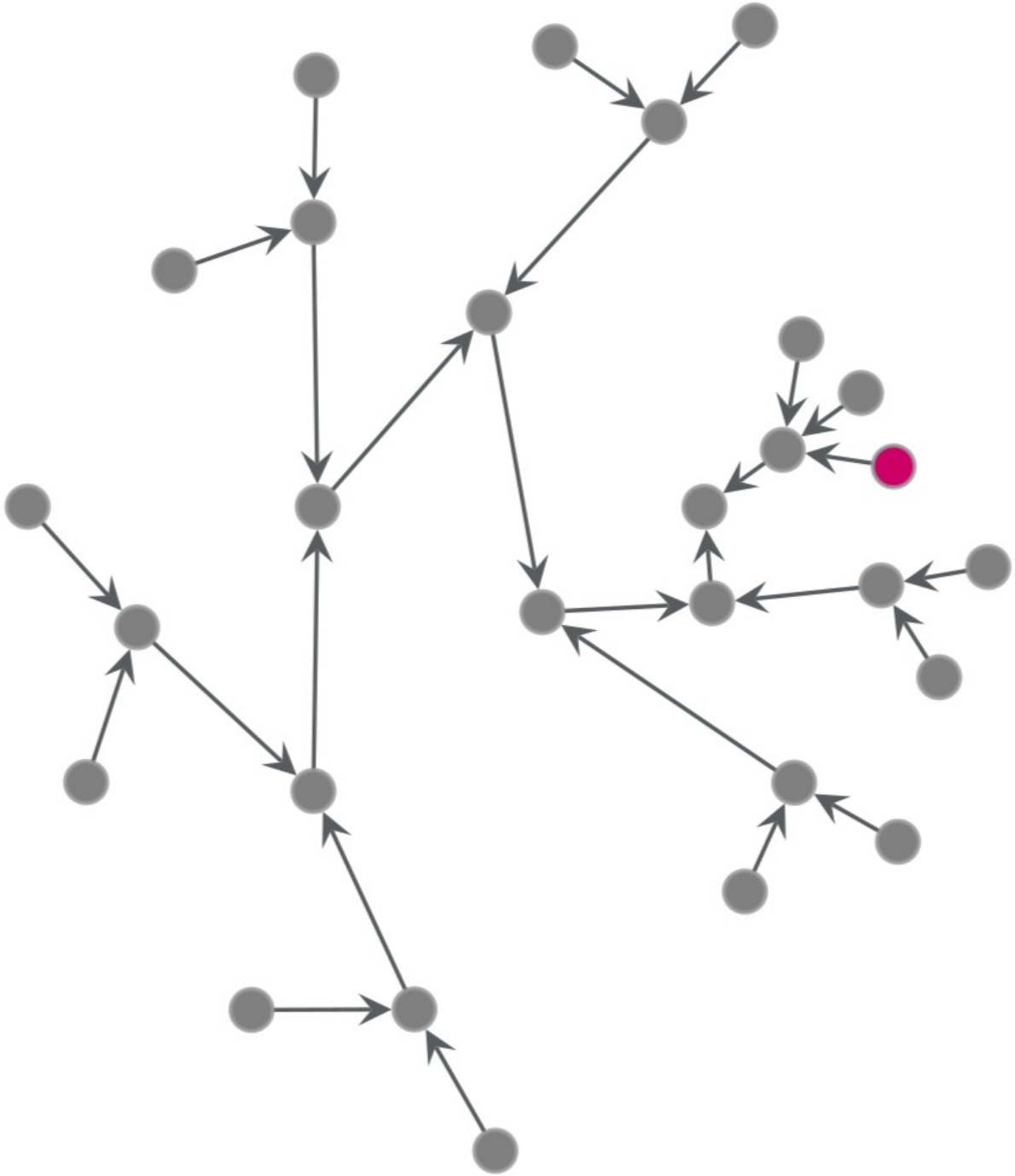


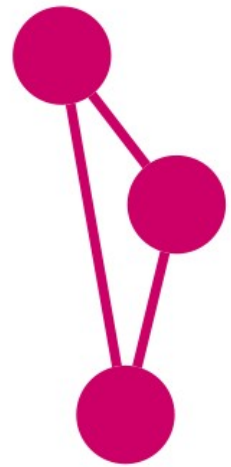


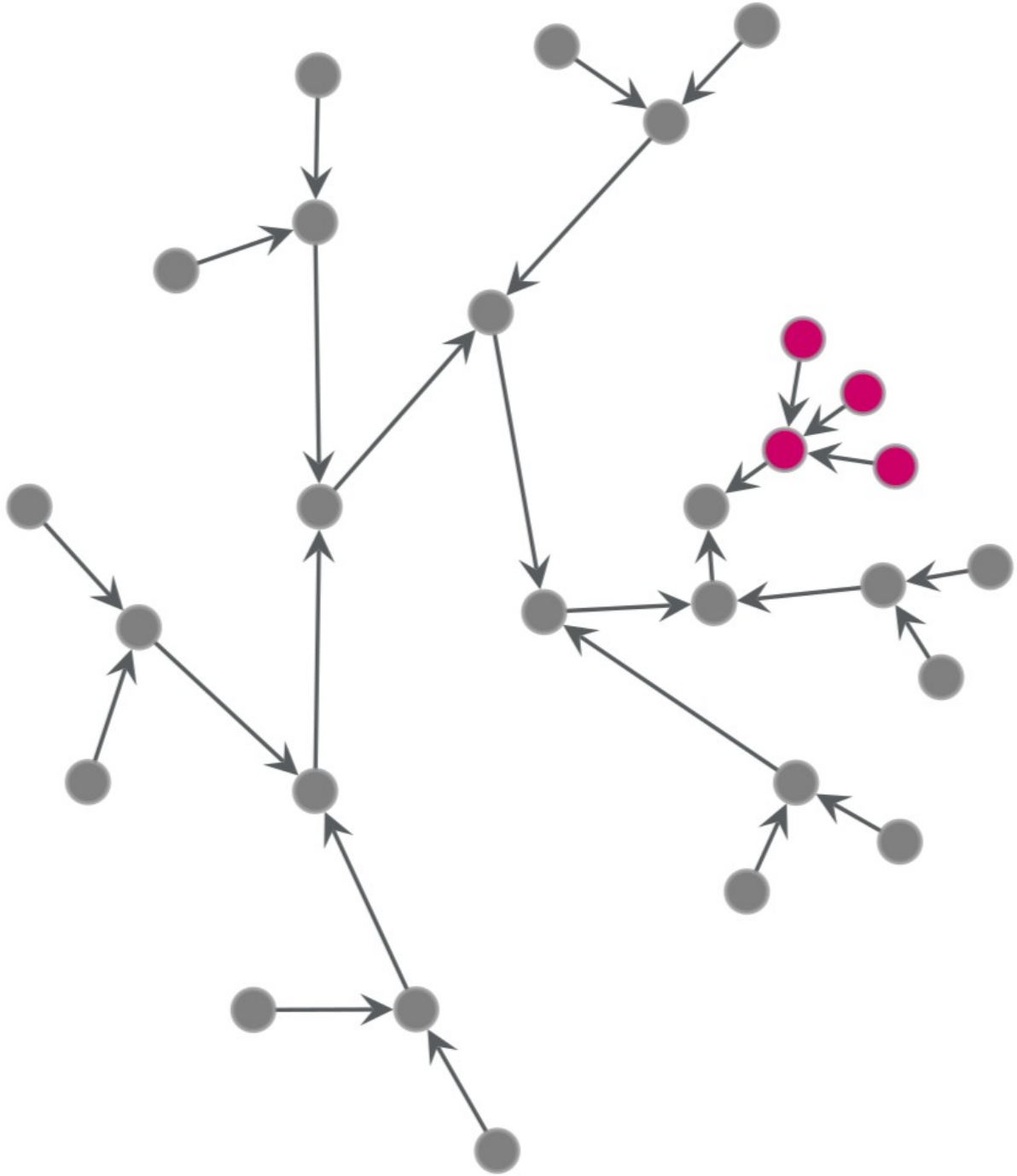


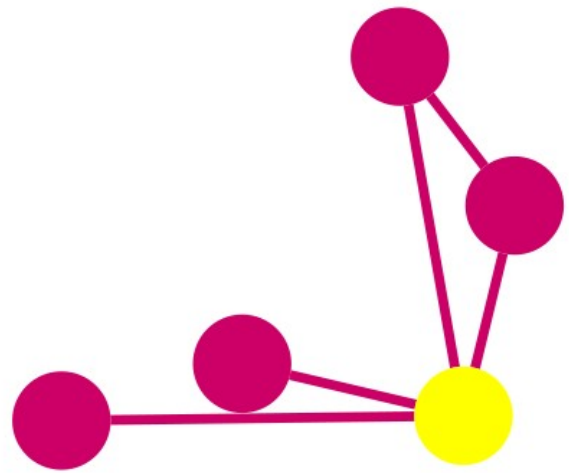


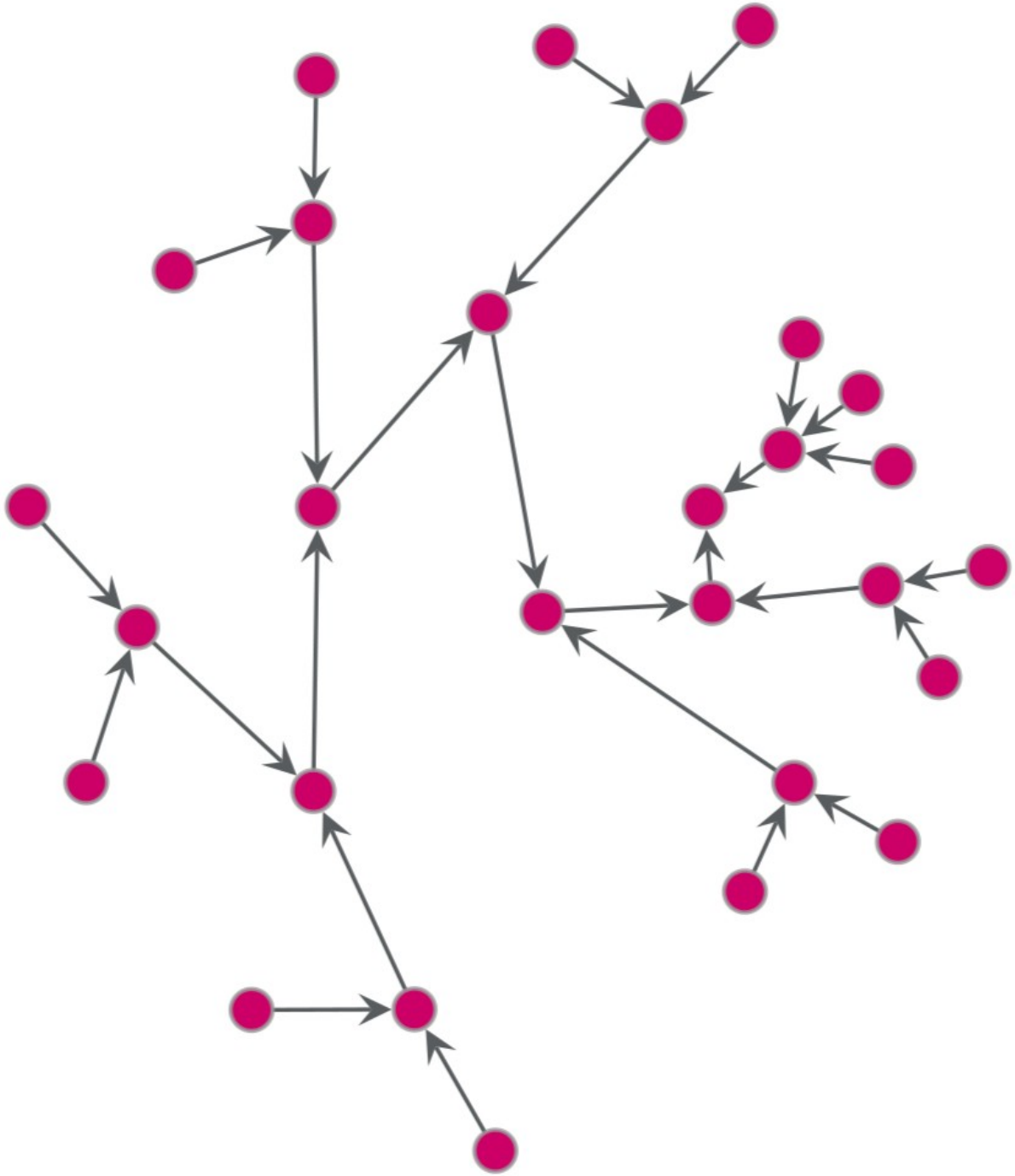


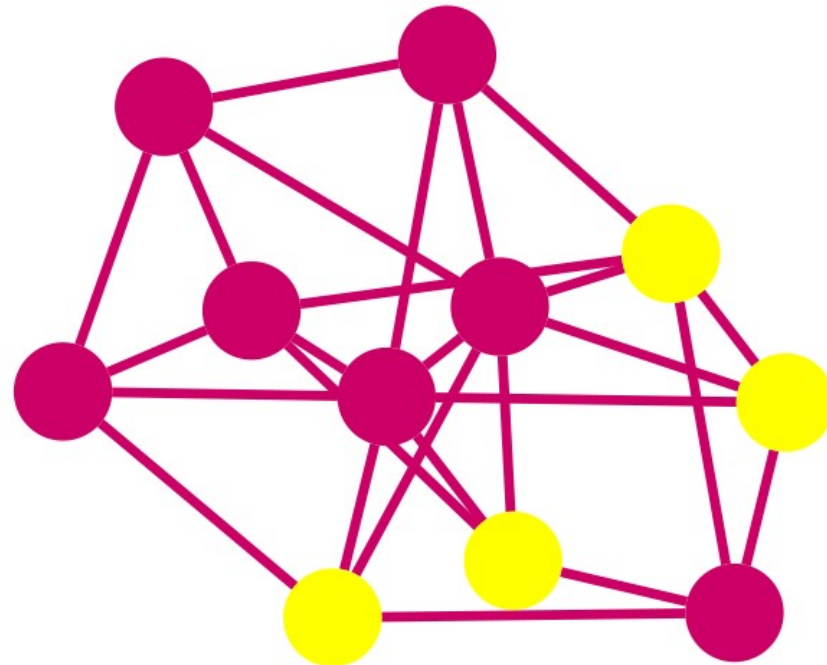






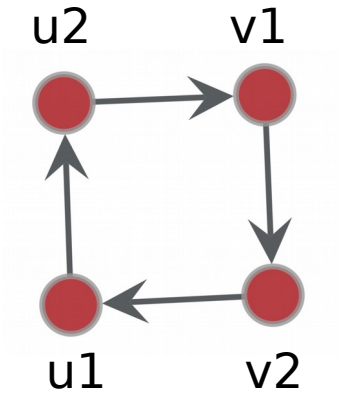






Minimum Separating Set

- Flow?
    - yes, but we need a transformation:



    - Remark: this only gives us minimum separating set between 2 vertices, not overall.
    - O(VE)

Minimum Separating Set

- Approximations:
    1. Don’t check every pair of vertices, just check the pair of vertices furthest apart. (actually lets relax that to just far apart, O(V+E) vs O(V^3))
    2. Instead of finding min cut with flow algorithms on auxiliary graph, just compare a few minimum separating set candidates.

- Candidate Finding Procedure:
    1. Run a breadth first searches starting from each of those vertices furthest apart from each other. (O(V+E))
    2. Note that each depth level in the BFS trees is a valid separating set and thus a candidate.
    3. Find the intersection of the middle depth level of both BFS trees and if it is a valid separating set include it as a candidate. (O(V+E))

Next

1. Decompose more data
    - Dreams
    - Drugs
    - Wiki-Votes
    - etc.
2. Look for structures that arise
3. Find more minimum separating set candidates

Special Thanks

- Grant: Computer-Human Graph TeleDiscovery (IIS-1563971)
- Principal Investigator: James Abello
- Region Graph Visuals: Monica Bansal
- Code Library used in Demo: Tiago P. Peixoto, “The graph-tool python library”, figshare. (2014) DOI: 10.6084/m9.figshare.1164194 [sci-hub, @tor]