Centrality of shortest paths: algorithms and complexity results

Dmytro Matsypura
with Johnson Phosavanh

Discipline of Business Analytics, The University of Sydney

WOMBAT 2025

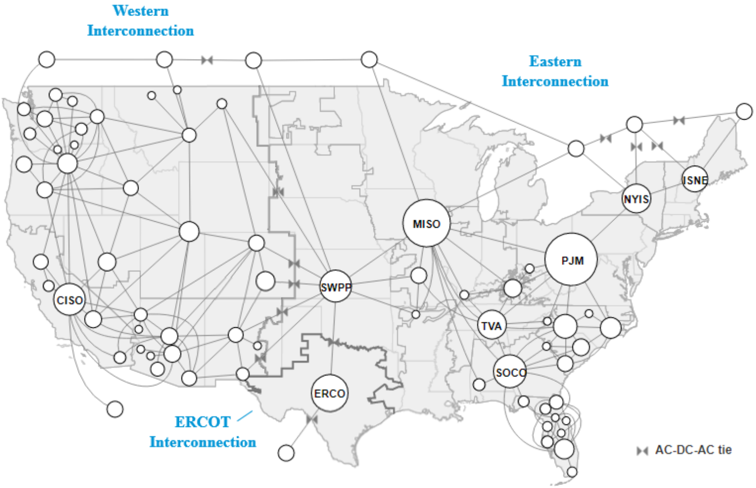
Outline

Motivation

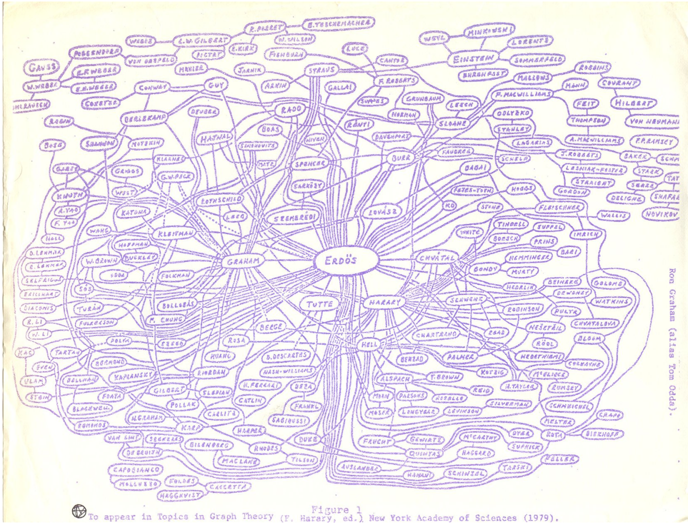
'Networks are present everywhere. All we need is an eye for them.'

Albert-Lásló Barabási, **Linked: The New Science of Networks**.

Network of electric power distribution: USA power grid

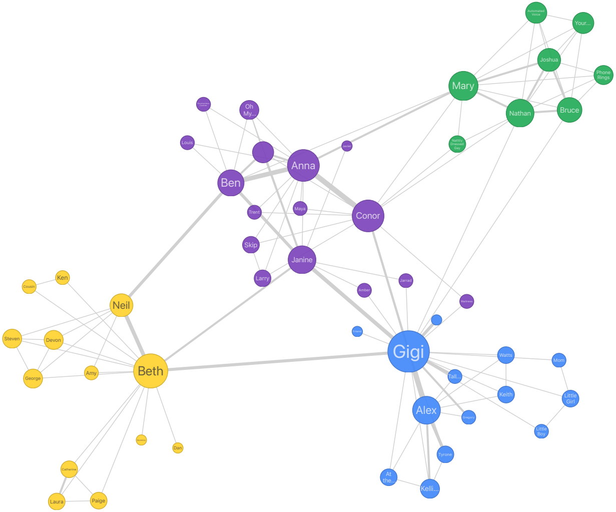


Source: hrefhttps://whyy.orghttps://whyy.org



Source: http://www.math.cmu.edu/∼ctsourak/amazing.html

Network of social interactions: He's just not that into you (2009)



Source: http://moviegalaxies.com

Centrality is a measure of importance

- **Centrality** is a property of a node's position in a network.

- It is the node's contribution to the structure of the network.

- Centrality helps us answer the question: **who or what is most important?**

- It is not one thing but a **family of concepts**:
  - degree
  - betweenness
  - closeness
  - k-step reach
  - eigenvector
  - ...

- Some centrality measures can be extended to **groups of nodes**.

- We focus on **degree**, **betweenness**, **closeness** and k-**step reach** centrality.

Definitions

- **Degree centrality** of a node is its **degree**:

$$C_{\mathsf{deg}}(i) = \mathsf{deg}(i)$$

- **Betweenness centrality** of node i is:

$$C_{\mathsf{btw}}(i) = \sum_{u<v:\ u,v\in V\setminus\{i\}} \frac{g_{uv}(i)}{g_{uv}},$$

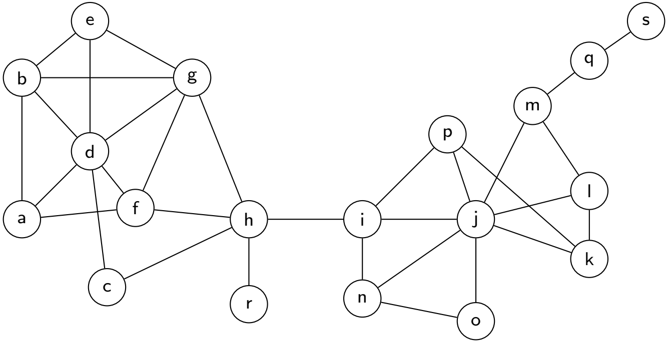
where $g_{uv}(i)$ is the number of shortest paths between nodes u and v that traverse through node i and g_{uv} is the total number of shortest paths between u and v.

- **Closeness centrality** of a node is:

$$C_{\mathsf{cls}}(i) = \max_{u\in V}\{d(i,u)\}.$$

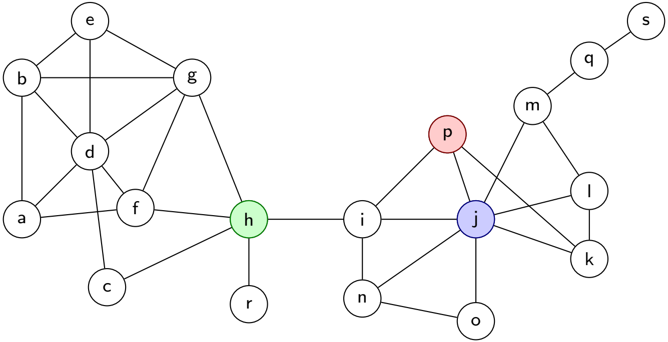
where $d(i,u)$ is the distance between nodes i and u.

Example: unweighted graph



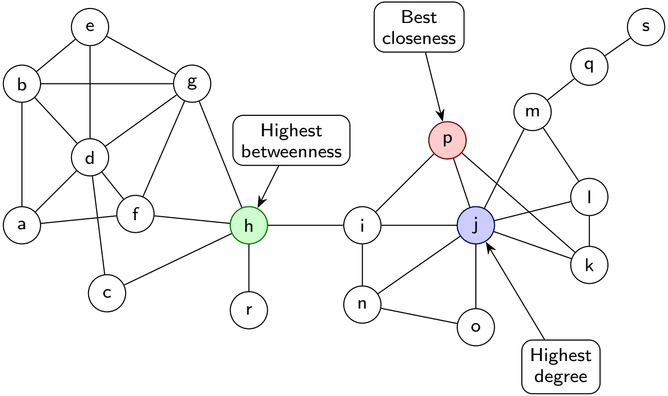
Nodes with best centrality scores:

Example: unweighted graph



Nodes with best centrality scores:

Example: unweighted graph



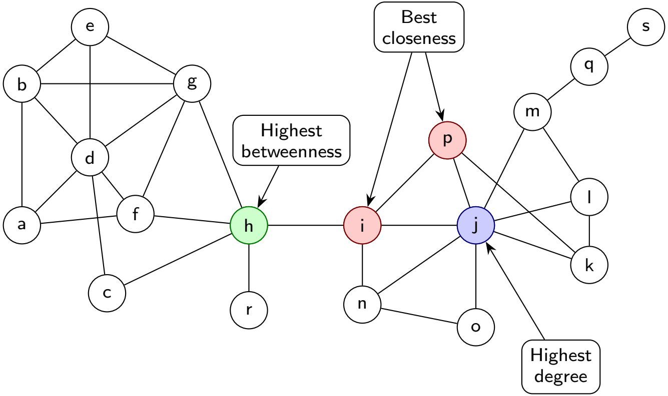
Nodes with best centrality scores:

j - highest degree centrality

h - highest betweenness centrality

p - best closeness centrality

Example: unweighted graph



Nodes with best centrality scores:

- j - highest degree centrality
- h - highest betweenness centrality
- p - best closeness centrality

The most central shortest path problem

- Let $G = (V, E)$ be an unweighted (possibly directed) graph.
- Let P be a path in G (a finite sequence of distinct adjacent vertices in G).

Problem (MCSP)

Given a graph G and a measure of centrality $C(P)$, solve the following problem:

$$\max\{C(P) : P \in \mathcal{SP}(G)\},$$

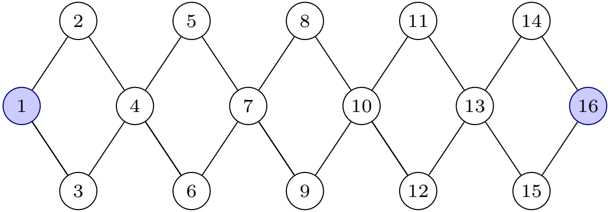
where $\mathcal{SP}(G)$ is the set of all shortest paths between all pairs of nodes in G.

In other words, we seek to find a path P with the largest centrality, provided that P is a shortest path between a pair of nodes in G.

Applications:

- Path-shaped facility location: segments of railroads, highways, pipelines
- Network design: routing air delivery service, subway, rail or bus service
- Defence: reconnaissance, recovery and aid delivery

Number of shortest paths can be exponential



- $s = 1$, $t = 16$
- number of shortest paths: $2^{(16-1)/3} = 2^5$
- shortest path length: $2(16-1)/3 = 10$
- can generalise by constructing graphs with $3n + 1$ nodes
  - $s = 1$, $t = 3n + 1$
  - number of shortest paths: 2^n
  - shortest path length: $2n$

Outline

Main results

Problem (MCSP)

Given a graph G and a measure of centrality $C(P)$, solve the following problem:

$$\max\{C(P) : P \in \mathcal{SP}(G)\},$$

where $\mathcal{SP}(G)$ is the set of all shortest paths between all pairs of nodes in G.

Centrality measure	Unweighted graph	Weighted graph
Betweenness	P	P
Degree	P	NP-hard
k-step reach	P	?
Closeness	NP-hard	NP-hard

Table: Complexity status of the problems considered.

Outline

Node degree centrality

- Let $G = (V, E)$ be an unweighted (possibly directed) graph
- The (open) **neighbourhood** of node u is the set of nodes adjacent to it:

$$\mathcal{N}(u) = \{v : (u, v) \in E\}$$

- **Degree** of node u is the size of its neighbourhood (i.e., number of adjacent nodes):

$$\text{deg}(u) = |\mathcal{N}(u)|$$

- **Degree centrality** of node u is its **degree**:

$$C_{\text{deg}}(u) = \text{deg}(u)$$

Path degree centrality

- Recall that P is a path in G (a finite sequence of distinct adjacent vertices).
- The (open) **neighbourhood** of path P is the set of nodes adjacent to it:

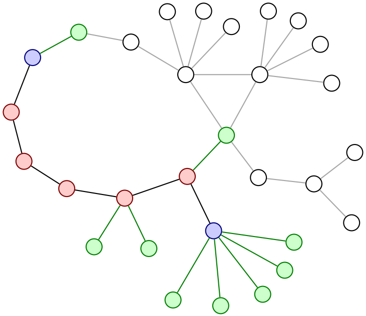
$$\begin{aligned}\mathcal{N}(P) &= \{v : (u, v) \in E,\ u \in P,\ v \notin P\} \\ &= \cup_{u \in P} \mathcal{N}(u) \setminus P\end{aligned}$$

- **Degree centrality** of path P is the size of its neighbourhood:

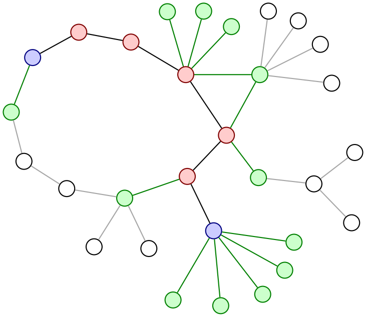
$$C_{\text{deg}}(P) = |\mathcal{N}(P)|$$

- Definition is consistent: if P is a singleton, i.e., $|P| = 1$, then $C_{\text{deg}}(P)$ reduces to **node degree centrality**.

Example



(a) path P (blue+red), $C_{\text{deg}}(P) = 9$.



(b) path $\tilde{P}$ (blue+red), $C_{\text{deg}}(\tilde{P}) = 12$.

Observations:

- P (left) and $\tilde{P}$ (right) are both shortest paths.
- Neighbourhoods of P and $\tilde{P}$ are depicted in green.
- $C_{\text{deg}}(P) = 9$ while $C_{\text{deg}}(\tilde{P}) = 12$, so $\tilde{P}$ is more central than P.

Most degree-central shortest path problem

Problem (2)

Given a graph $G = (V, E)$ *and measure of centrality* $C_{deg}(P) = |\mathcal{N}(P)|$, *solve the following problem:*

$$\max\{C_{deg}(P) : P \in \mathcal{SP}(G)\},$$

where $\mathcal{SP}(G)$ *is the set of all shortest paths between all pairs of nodes in* G.

Previous result:

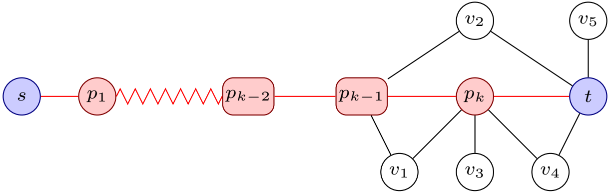
- Matsypura et al. (2023)[1] proved that the problem is polynomial.
- They developed the MVP algorithm with the worst-case running time of $O(k|V|^6)$.
- k is the diameter of G (length of the longest shortest path).
- Can we do better?

[1] M, Veremyev, Pasiliao, Prokopyev (2023). Finding the most degree-central walks and paths in a graph: Exact and heuristic approaches.

Finding most degree-central shortest path

Lemma (most degree-central shortest path)

If $\langle s, p_1, \ldots, p_{k-1}, p_k, t\rangle$ *is a most degree-central shortest path from* s *to* t*, then* $\langle s, p_1, \ldots, p_{k-1}\rangle$ *is a most degree-central shortest path from* s *to* p_{k-1}*.*



Proof (sketch).

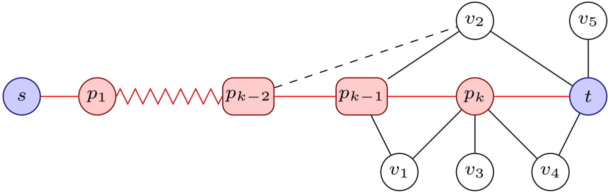
- t and p_{k-2} cannot have common neighbours

□

Finding most degree-central shortest path

Lemma (most degree-central shortest path)

If $\langle s, p_1, \ldots, p_{k-1}, p_k, t\rangle$ *is a most degree-central shortest path from* s *to* t*, then* $\langle s, p_1, \ldots, p_{k-1}\rangle$ *is a most degree-central shortest path from* s *to* p_{k-1}*.*



Proof (sketch).

- t and p_{k-2} cannot have common neighbours
- because if they did, there would have to be a shortcut

□

Algorithm for the most degree-central shortest path (Algorithm 1)

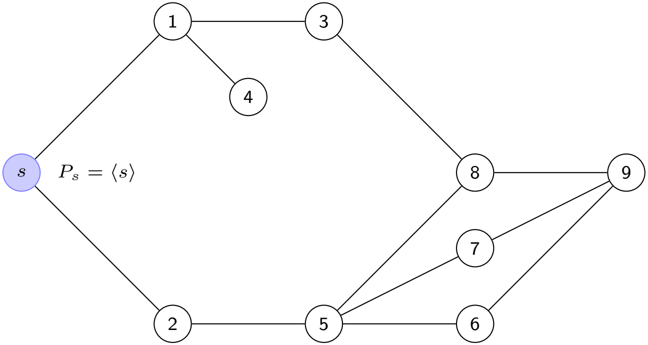


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

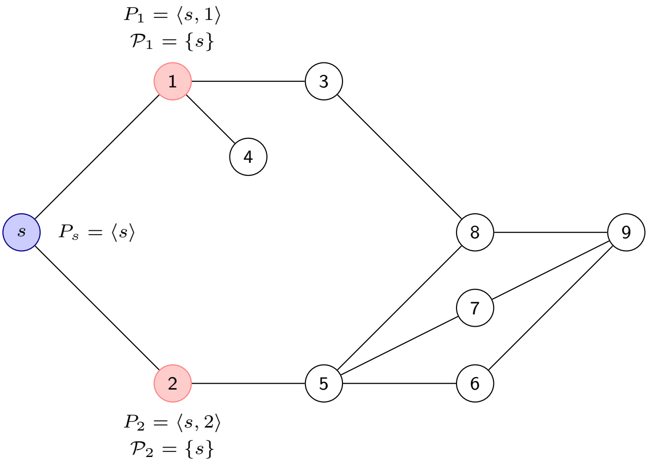


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

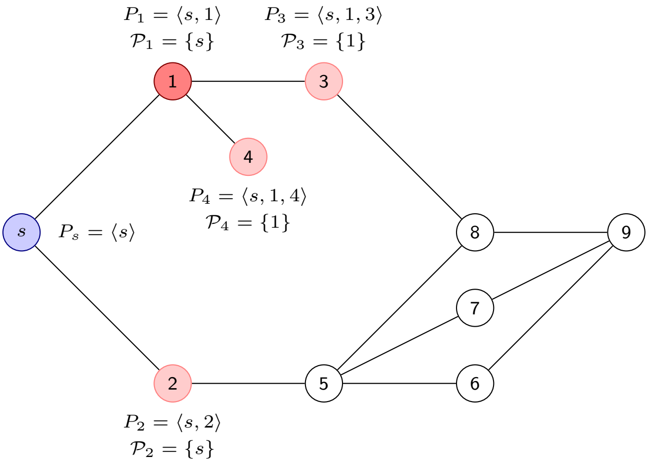


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

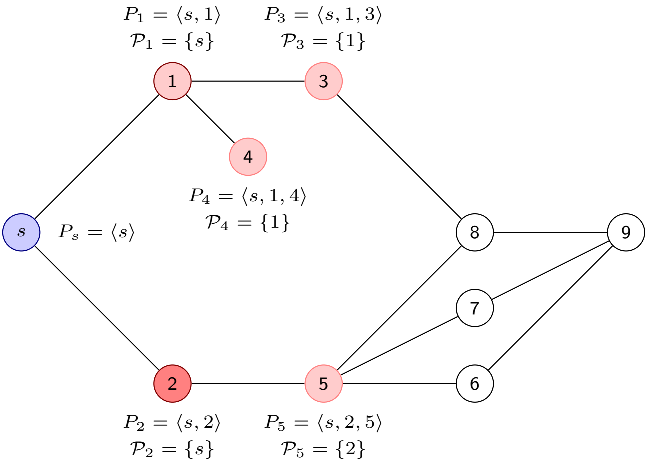


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

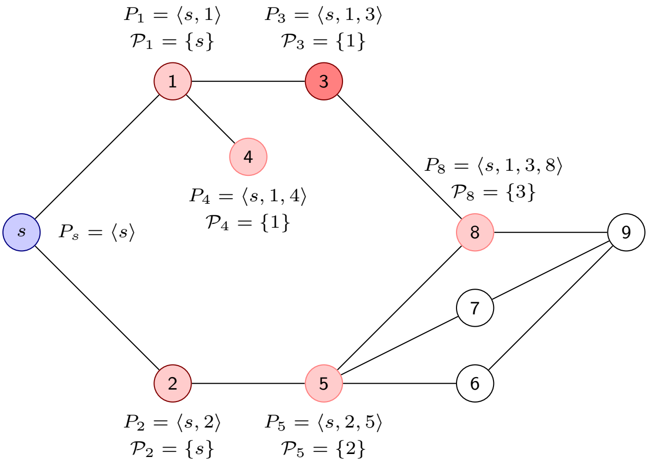


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

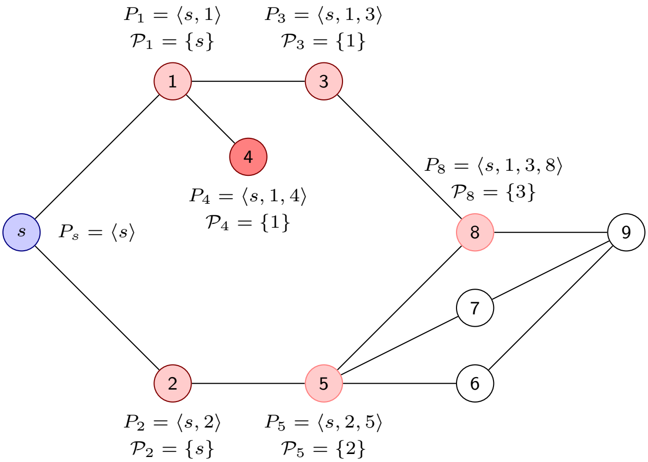


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

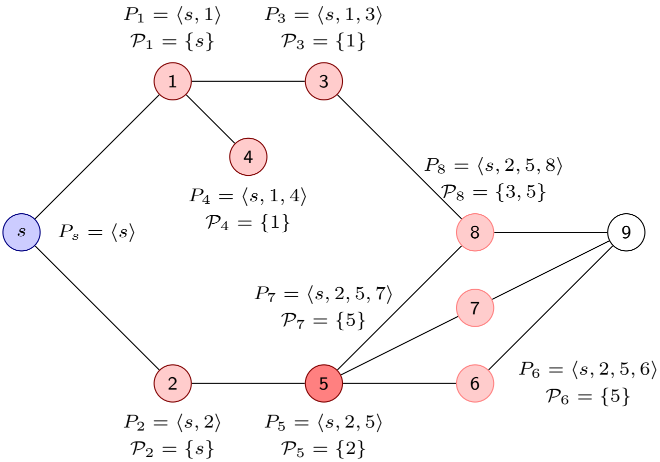


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

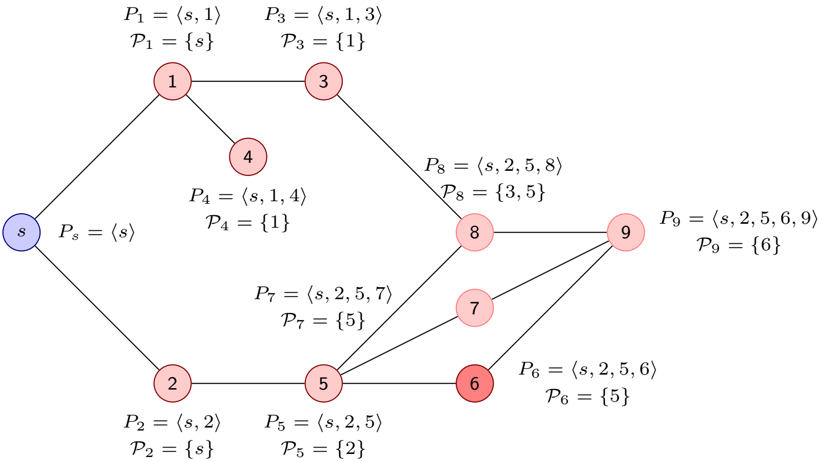


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

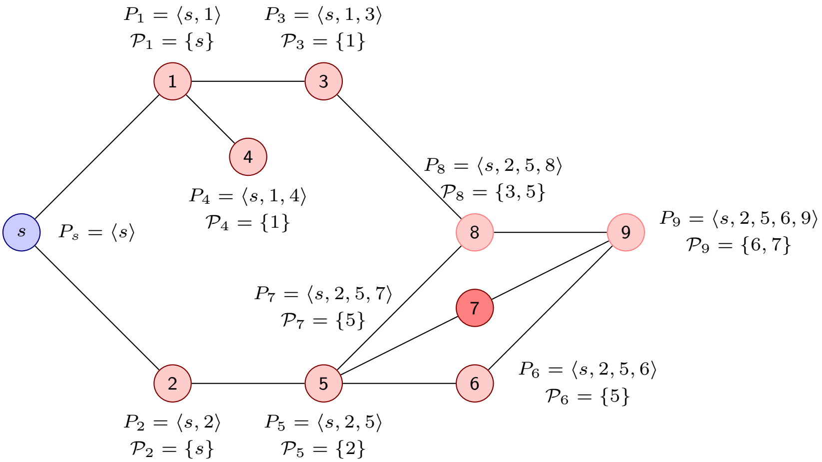


Figure: Example with starting node s

Algorithm for the most degree-central shortest path (Algorithm 1)

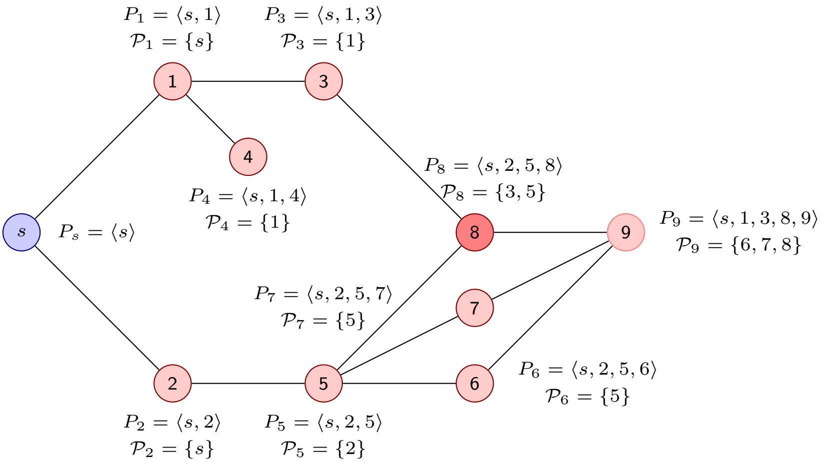


Figure: Example with starting node s

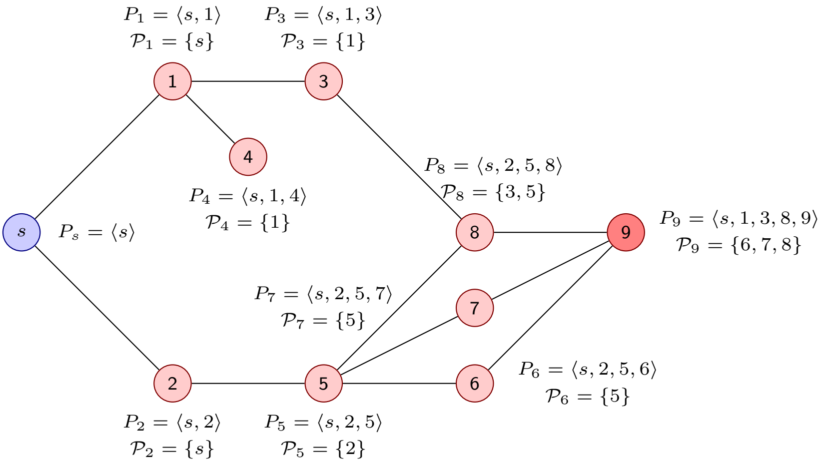


Figure: Example with starting node s

Pseudocode for Algorithm 1

Algorithm 1: Finding the most degree-central shortest path

Input: Graph $G = (V, E)$, starting vertex s.
Output: Most degree-central shortest path P_v from s to v for every $v \in V$.

```
for $v \in V$ do
    $d_v \leftarrow +\infty$, $P_v \leftarrow$ undefined, $\mathcal{P}_v = \emptyset$.
end for
$P_s \leftarrow \langle s \rangle$.
Insert $(0, s)$ to the queue $Q$.
while $Q$ not empty do
    $(d_u, u) \leftarrow$ pop next element of $Q$.
    for $v \in \{v' \in V : (u, v') \in E\}$ do
        $d_{\text{new}} \leftarrow d_u + 1$.
        if $d_{\text{new}} = 1$ then
            Insert $(d_{\text{new}}, v)$ to the queue $Q$.
            $d_v \leftarrow d_{\text{new}}$, $P_v = \langle s, v \rangle$, $\mathcal{P}_v \leftarrow \{s\}$.
        else if $d_{\text{new}} < d_v$ then
            Insert $(d_{\text{new}}, v)$ to the queue $Q$.
            $d_v \leftarrow d_{\text{new}}$, $\mathcal{P}_v \leftarrow \mathcal{P}_v \cup \{u\}$.
            for $w \in \mathcal{P}_u$ do
                if $C_{\text{deg}}(\langle P_w, u, v \rangle) > C_{\text{deg}}(P_v)$ then
                    $P_v \leftarrow \langle P_w, u, v \rangle$.
                end if
            end for
        else if $d_{\text{new}} = d_v$ then
            $\mathcal{P}_v \leftarrow \mathcal{P}_v \cup \{u\}$.
            for $w \in \mathcal{P}_u$ do
                if $C_{\text{deg}}(\langle P_w, u, v \rangle) > C_{\text{deg}}(P_v)$ then
                    $P_v \leftarrow \langle P_w, u, v \rangle$.
                end if
            end for
        end if
    end for
```

Complexity of Algorithm 1

Theorem

Given a graph $G = (V, E)$*, the worst-case running time of Algorithm 1 is* $O(|E||V|\Delta(G))$*.*

Corollary

The most degree-central shortest path problem can be solved in $O(|E||V|^2\Delta(G))$ *time.*

- This is (roughly) $|V|^2$-more efficient than the MVP algorithm of Matsypura et al. (2023), which has the worst-case running time of $O(k|V|^6)$.
- $\Delta(G)$ is the degree of G (the largest degree of G's vertices).
- k is the diameter of G (length of the longest shortest path).

The case of weighted graphs

Theorem

The most degree-central shortest path problem on a weighted graph is NP-hard.

Proof (sketch).

Reduction from the Maximum Satisfiability (MaxSAT) problem. □

Special cases:

- If edge weights are positive and integer-valued, Algorithm 1 runs in pseudo-polynomial time $O(w_{\mathsf{sum}}|V|^2\Delta(G))$, where w_{sum} is the sum of all edge weights.
- If the edge weights are generated from some positive continuous distribution, the shortest path between each pair of nodes will be unique with probability 1. Hence, we can solve the problem in polynomial time.

Numerical results: synthetic instances

	Barabási-Albert				
$\|V\|$	100	500	1000	5000	10000
$\|E\|$	196	996	1996	9996	19996
$\Delta(G)$	25.27	53.83	76.53	188.20	257.40
$\|\mathcal{SP}(G)\|/U(G)$	2.13	2.67	2.87	3.34	3.53
diam	5.57	7.03	7.27	8.53	9.00
Degree centrality					
diam centrality	46.17	120.10	179.30	382.57	553.27
path length	3.87	4.67	5.07	5.80	6.03
path centrality	51.87	138.33	199.83	483.93	676.47
MVP runtime	0.08	7.50	58.44	7532.65	-
Alg. 1 runtime	0.07	2.31	13.46	713.06	3792.77

- Figures are averages over 30 instances for each size.
- Runtimes are in seconds.
- Row **path length** gives the length of the optimal shortest path.
- Row **path centrality** gives the centrality of the optimal shortest path.

Numerical results: real-world instances

	IEEE Bus	Santa Fe	US Air 97	Bus	Email	Cerevisiae
$\lvert V \rvert$	118	118	332	662	1133	1458
$\lvert E \rvert$	179	200	2126	906	5451	1948
$\Delta(G)$	9	29	139	9	71	56
$\lvert \mathcal{SP}(G) \rvert / U(G)$	2.26	1.51	5.55	2.44	6.73	2.57
diam	14	12	6	25	8	19
Degree centrality						
diam centrality	32	90	167	45	159	57
path length	8	10	3	20	4	7
path centrality	33	92	206	50	187	156
MVP runtime	0.26	0.20	3.43	40.58	107.14	246.92
Alg. 1 runtime	0.09	0.09	2.17	3.88	26.53	22.89

- Runtimes are in seconds.
- Row **path length** gives the length of the optimal shortest path.
- Row **path centrality** gives the centrality of the optimal shortest path.

Numerical results: real-world instances

	Graph 1		Graph 2		Graph 3	
weighted	no	yes	no	yes	no	yes
$\lvert V\rvert$	3783	281050	5881	397821	6539	167369
$\lvert E\rvert$	24186	301453	35592	427532	51127	211957
$\Delta(G)$	511	511	795	795	805	805
$\lvert\mathcal{SP}(G)\rvert/U(G)$	9.50	2.68	10.74	19.18	8.54	1.83
diam	10	107	11	107	11	35
Degree centrality						
diam nodes traversed	11	12	12	12	12	7
diam centrality	689	266	971	299	318	846
path length	5	39	4	31	4	11
path nodes traversed	6	9	5	12	5	4
path centrality	865	891	1349	1433	1318	1165
Alg. 1 runtime	1134	20996	4189	58050	1345	24927

- Runtimes are in seconds.
- $|E| = w_{\text{sum}}$ (sum of all edge weights) for weighted graphs.
- Row **path length** gives the length of the optimal shortest path.
- Row **path centrality** gives the centrality of the optimal shortest path.

Outline

k-step reach centrality

- Let $d(u, v)$ denote the distance between u and v (length of the shortest path)
- For unweighted graphs, the k-**step reach neighbourhood** of path P is

$$\mathcal{N}_k(P) = \{v : d(u, v) \leq k,\ u \in P,\ v \notin P\}.$$

- Then, the k-**step reach centrality** for path P is

$$C_k = |\mathcal{N}_k(P)|$$

Theorem

The most 2-step reach shortest path problem is polynomial.

Proposition

The most k-step reach shortest path problem is polynomial.

Outline

- **Classic** definition by Everett and Borgatti (1999):

$$C_{\text{btw}}(P) = \sum_{u<v:\ u,v \in V \setminus P} \frac{g_{uv}(P)}{g_{uv}},$$

where $g_{uv}(P)$ is the number of shortest paths between nodes u and v that traverse through **at least one node** in P.

- **Alternative** definition by Puzis et al. (2007):

$$C_{\text{btw}}(P) = \sum_{u<v:\ u,v \in V \setminus P} \frac{\tilde{g}_{uv}(P)}{\tilde{g}_{uv}},$$

where $\tilde{g}_{uv}(P)$ is the number of shortest paths between nodes u and v that traverse through **all nodes** in P.

- **Our** definition (aka **stress centrality** of Shimbel (1953)):

$$C_{\text{btw}}(P) = \sum_{u<v:\ u,v \in V \setminus P} g_{uv}(P).$$

Most betweenness-central shortest path problem

Path betweenness centrality $C_{\text{btw}}(P)$ is the number of shortest paths between all pairs of nodes not on P that traverse through at least one node in P:

$$C_{\text{btw}}(P) = \sum_{u<v:u,v\in V\setminus P} g_{uv}(P),$$

where $g_{uv}(P)$ is the number of shortest paths between u and v that traverse through at least one node in P.

Problem (3)

Given a graph $G = (V, E)$ and measure of centrality C_{btw}, solve the following problem:

$$\max \{C_{\textit{btw}}(P) : P \in \mathcal{SP}(G)\}.$$

Theorem

Problem (3) is solvable in $O(|E|^2|V|^2)$ time.

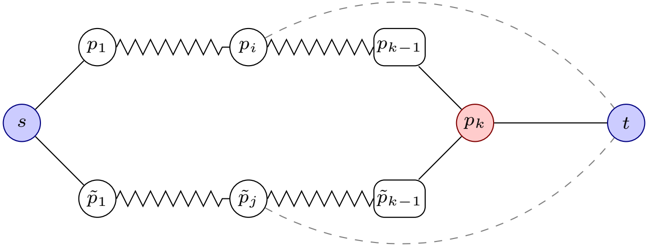
Corollary

Problem (3) on a graph with positively weighted edges is solvable in polynomial time.

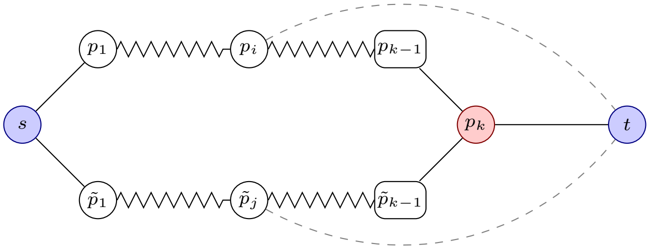
Finding the most betweenness-central shortest path

Lemma (most betweenness-central shortest path)

If $\langle s, p_1, \ldots, p_k, t \rangle$ is a most betweenness-central shortest path from s to t, then $\langle s, p_1, \ldots, p_k \rangle$ is a most betweenness-central shortest path from s to p_k.



Finding the most betweenness-central shortest path



Proof (sketch).

- If $P = \langle s, p_1, \ldots, p_k, t\rangle$ is shortest from s to t then $P_{-t} = \langle s, p_1, \ldots, p_k\rangle$ is shortest from s to p_k.
- Let P be the most betweenness-central shortest path between s and t and $\tilde{P} = \langle s, \tilde{p}_1, \ldots, \tilde{p}_{k-1}, p_k, t\rangle$ be an alternative path.
- Removing t from P updates the betweenness centrality score of P_{-t} by adding paths that end at t and traverse through at least one node in P, and subtracting paths that only traverse through t and none of the nodes in P_{-t}.
- The number of paths subtracted is fixed regardless of whether P or $\tilde{P}$ was the most betweenness-central.
- $\Rightarrow$ for P to be most betweenness-central, P_{-t} must be most betweenness-central between s and p_k.

□

Pseudocode for Algorithm 2

Algorithm 2: Finding the most betweenness-central shortest path

Input: Graph $G = (V, E)$, starting vertex s.
Output: Most betweenness-central shortest path P_v from s to v for every $v \in V$.

for $v \in V$ **do**
    $d_v \leftarrow +\infty$, $P_v \leftarrow$ undefined.
end for
$P_s \leftarrow \langle s \rangle$.
Insert $(0, s)$ to the queue Q.
while Q not empty **do**
    $(d_u, u) \leftarrow$ pop next element of Q.
    for $v \in \{v' \in V : (u, v') \in E\}$ **do**
        $d_{\text{new}} \leftarrow d_u + 1$.
        if $d_{\text{new}} < d_v$ **then**
            Insert (d_{new}, v) to the queue Q.
            $d_v \leftarrow d_{\text{new}}$, $P_v = \langle s, v \rangle$.
        else if $d_{\text{new}} = d_v$ **then**
            if $C_{\text{btw}}(\langle P_u, v \rangle) > C_{\text{btw}}(P_v)$ **then**
                $P_v \leftarrow \langle P_u, v \rangle$.
            end if
        end if
    end for
end while

Numerical results: synthetic instances

	Barabási-Albert				
$\|V\|$	100	500	1000	5000	10000
$\|E\|$	196	996	1996	9996	19996
$\Delta(G)$	25.27	53.83	76.53	188.20	257.40
$\|\mathcal{SP}(G)\|/U(G)$	2.13	2.67	2.87	3.34	3.53
diam	5.57	7.03	7.27	8.53	9.00
Degree centrality					
diam centrality	46.17	120.10	179.30	382.57	553.27
path length	3.87	4.67	5.07	5.80	6.03
path centrality	51.87	138.33	199.83	483.93	676.47
MVP runtime	0.08	7.50	58.44	7532.65	-
Alg. 1 runtime	0.07	2.31	13.46	713.06	3792.77
Betweenness centrality					
diam centrality	12879.93	346136.47	-	-	-
path length	4.37	5.37	-	-	-
path centrality	14788.67	402991.40	-	-	-
preprocessing time	0.01	0.18	-	-	-
Alg. 2 runtime	34.66	23966.26	-	-	-

Table: Results for Barabási-Albert graphs averaged over 30 instances. Runtimes and preprocessing times are in seconds. Rows **path length** and **path centrality** give the length and the centrality of the optimal shortest path, respectively.

Numerical results: synthetic instances

	Watts-Strogatz (4, 0.1)				
$\|V\|$	100	500	1000	5000	10000
$\|E\|$	200	1000	2000	10000	20000
$\Delta(G)$	5.87	6.40	6.70	7.27	7.33
$\|\mathcal{SP}(G)\|/U(G)$	2.45	3.18	3.53	4.37	4.80
diam	10.43	15.40	17.73	22.33	24.50
Degree centrality					
diam centrality	22.23	31.20	35.37	42.53	45.20
path length	9.03	12.97	14.73	18.63	19.37
path centrality	23.17	33.87	37.60	46.90	50.40
MVP runtime	0.14	13.24	106.47	15056.90	-
Alg. 1 runtime	0.07	2.08	10.26	366.88	1712.52
Betweenness centrality					
diam centrality	11823.33	172901.27	-	-	-
path length	8.27	11.73	-	-	-
path centrality	13130.13	213512.87	-	-	-
preprocessing time	0.01	0.18	-	-	-
Alg. 2 runtime	38.61	28597.39	-	-	-

Table: Results for Watts-Strogatz graphs with rewiring probability set to 0.1, averaged over 30 instances. Runtimes and preprocessing times are in seconds. Rows **path length** and **path centrality** give the length and the centrality of the optimal shortest path, respectively.

Numerical results: synthetic instances

	Watts-Strogatz (4, 0.2)				
$\lvert V\rvert$	100	500	1000	5000	10000
$\lvert E\rvert$	200	1000	2000	10000	20000
$\Delta(G)$	6.60	7.40	7.43	8.03	8.20
$\lvert\mathcal{SP}(G)\rvert/U(G)$	2.04	2.25	2.35	2.57	2.66
diam	8.43	11.50	12.97	16.13	17.47
Degree centrality					
diam centrality	21.67	28.80	30.60	37.73	40.17
path length	7.03	9.33	10.17	12.53	13.27
path centrality	23.33	31.73	35.17	43.77	47.07
MVP runtime	0.12	10.54	83.88	-	-
Alg. 1 runtime	0.06	2.08	9.75	359.95	1704.02
Betweenness centrality					
diam centrality	7343.40	81079.20	-	-	-
path length	6.83	8.83	-	-	-
path centrality	8283.87	100763.80	-	-	-
preprocessing time	0.01	0.18	-	-	-
Alg. 2 runtime	36.89	26936.55	-	-	-

Table: Results for Watts-Strogatz graphs with a rewiring probability of 0.2, averaged over 30 instances. Runtimes and preprocessing times are in seconds. Rows **path length** and **path centrality** give the length and the centrality of the optimal shortest path, respectively.

Numerical results: real-world instances

	IEEE Bus	Santa Fe	US Air 97	Bus	Email	Cerevisiae
$\lvert V\rvert$	118	118	332	662	1133	1458
$\lvert E\rvert$	179	200	2126	906	5451	1948
$\Delta(G)$	9	29	139	9	71	56
$\lvert\mathcal{SP}(G)\rvert/U(G)$	2.26	1.51	5.55	2.44	6.73	2.57
diam	14	12	6	25	8	19
Degree centrality						
diam centrality	32	90	167	45	159	57
path length	8	10	3	20	4	7
path centrality	33	92	206	50	187	156
MVP runtime	0.26	0.20	3.43	40.58	107.14	246.92
Alg. 1 runtime	0.09	0.09	2.17	3.88	26.53	22.89
Betweenness centrality						
diam centrality	26530	20422	180104	650164	-	-
path length	13	11	5	18	-	-
path centrality	26734	20422	254286	705878	-	-
preprocessing time	0.01	0.01	0.15	0.29	-	-
Alg. 2 runtime	61.18	47.08	10612.16	76869.96	-	-

Table: Results for real-world instances. Runtimes and preprocessing times are in seconds. Rows **path length** and **path centrality** give the length and the centrality of the optimal shortest path, respectively.

Numerical results: real-world instances

	Copenhagen calls			
weighted	no	yes	no	yes
directed	yes	yes	no	no
$\lvert V \rvert$	536	536	536	536
$\lvert E \rvert$	924	924	621	621
$\Delta(G)$	18	18	18	18
$\lvert \mathcal{SP}(G) \rvert / U(G)$	1.43	1.35	1.79	1.50
diam	21	75	22	197
Betweenness centrality				
diam nodes traversed	22	25	23	15
diam centrality	54084	35826	133280	140648
path length	9	17	7	32
path nodes traversed	10	11	8	16
path centrality	59959	57692	177592	154253
preprocessing time	0.06	0.11	0.10	0.26
Alg. 2 runtime	1118.62	1182.93	5320.75	5741.69

Table: Results for betweenness centrality on Copenhagen Calls graph instances. Runtimes and preprocessing times are in seconds. Rows **path length** and **path centrality** give the length and the centrality of the optimal shortest path, respectively.

Numerical results: real-world instances

	Copenhagen SMS			
weighted	no	yes	no	yes
directed	yes	yes	no	no
$\lvert V \rvert$	568	568	568	568
$\lvert E \rvert$	1303	1303	697	697
$\Delta(G)$	11	11	11	11
$\lvert \mathcal{SP}(G) \rvert / U(G)$	1.99	1.30	2.12	1.32
diam	22	1783	20	3781
Betweenness centrality				
diam nodes traversed	23	10	21	12
diam centrality	127129	81092	224086	118298
path length	8	81	8	106
path nodes traversed	9	16	9	23
path centrality	228071	156347	280440	159783
preprocessing time	0.14	0.45	0.15	0.55
Alg. 2 runtime	12492.22	13737.50	15609.16	17860.28

Table: Results for betweenness centrality on Copenhagen SMS graph instances. Runtimes and preprocessing times are in seconds. Rows **path length** and **path centrality** give the length and the centrality of the optimal shortest path, respectively.

Outline

Most closeness-central shortest path problem

Recall that $d(u,v)$ denotes the distance between nodes u and v.
We overload notation and use $d(u,P)$ for the distance from node u to path P:

$$d(u,P) = \min_{v \in P}\{d(u,v)\}$$

We define **path closeness centrality** for path P as

$$C_{\text{cls}}(P) = \max_{u \in V \setminus P}\{d(u,P)\}.$$

Problem (4)

Given a graph $G=(V,E)$ and measure of centrality C_{cls}, solve the following problem:

$$\min\left\{C_{cls}(P) : P \in \mathcal{SP}(G)\right\}.$$

Theorem

Problem (4) is NP-hard.

Outline

Concluding remarks

- Finding central shortest paths in networks is an interesting problem.
- The problem is challenging because the number of shortest paths between a pair of nodes can be exponential.
- Computational complexity depends on the measure of centrality and whether the edges are weighted or not:

Centrality measure	Unweighted graph	Weighted graph
Betweenness	P	P
Degree	P	NP-hard
k-step reach	P	?
Closeness	NP-hard	NP-hard

- The worst-case runtime for the most degree-central shortest path problem on unweighted graphs is $O(|E||V|^2\Delta(G))$.
- The worst-case runtime for the most betweenness-central shortest path problem is
  - $O(|E|^2|V|^2)$ on unweighted graphs
  - $O(|E|^2|V|^2 + |V|^2\log(|V|))$ on graphs with positively weighted edges
- Both algorithms are easy to parallelise.

Extensions

- MIP formulations for NP-hard problems + approximation schemes
- Relax the constraint: generalisation to **almost** shortest path

The end!

Questions? Comments?

Appendix

Undirected graphs

Definition: an **undirected graph** G consists of a set V of **nodes** (or vertices) and a set E of **edges** (or undirected arcs), where an edge is an **unordered pair** of distinct nodes. We write $G = (V, E)$.

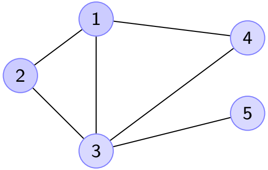


Figure: a graph with $V = \{1, \ldots, 5\}$ and $E = \{(1,2), (1,3), (1,4), (2,3), (3,4), (3,5)\}$.

assumptions:

- there is at most one edge from node i to node j
- edges (i, j) and (j, i) are one and the same
- there are no loops (no edges (i, i))

Networks

- In scientific literature, the terms **network** and **graph** are often used interchangeably
- However, typically **network** is more than just a graph
- **Definition**: a **network** is a graph $G = (V, E)$ together with additional node and edge features
  - external **supply** to each node
  - edge **capacity**
  - **cost** per unit of flow along an edge
  - ...

Data sources

Links to large real-life instances:

- https://networks.skewed.de/net/advogato
- https://networks.skewed.de/net/arxiv_collab
- https://networks.skewed.de/net/bitcoin_alpha
- https://networks.skewed.de/net/bitcoin_trust
- https://networks.skewed.de/net/copenhagen

Numerical results: synthetic instances

	Watts-Strogatz (4, 0.1)				
$\|V\|$	100	500	1000	5000	10000
$\|E\|$	200	1000	2000	10000	20000
$\Delta(G)$	5.87	6.40	6.70	7.27	7.33
$\|\mathcal{SP}(G)\|/U(G)$	2.45	3.18	3.53	4.37	4.80
diam	10.43	15.40	17.73	22.33	24.50
diam centrality	22.23	31.20	35.37	42.53	45.20
path length	9.03	12.97	14.73	18.63	19.37
path centrality	23.17	33.87	37.60	46.90	50.40
MVP runtime	0.41	59.73	513.84	-	-
Alg 1 runtime	0.07	2.08	10.26	366.88	1712.52

	Watts-Strogatz (4, 0.2)				
$\|V\|$	100	500	1000	5000	10000
$\|E\|$	200	1000	2000	10000	20000
$\Delta(G)$	6.60	7.40	7.43	8.03	8.20
$\|\mathcal{SP}(G)\|/U(G)$	2.04	2.25	2.35	2.57	2.66
diam	8.43	11.50	12.97	16.13	17.47
diam centrality	21.67	28.80	30.60	37.73	40.17
path length	7.03	9.33	10.17	12.53	13.27
path centrality	23.33	31.73	35.17	43.77	47.07
MVP runtime	0.41	55.73	482.51	-	-
Alg 1 runtime	0.06	2.08	9.75	359.95	1704.02

Table: Results for Watts-Strogatz graphs averaged over 30 instances. Runtimes and preprocessing times are in seconds.

Distance-based group centrality measures

- **Harmonic centrality**:

$$C_1(S,G) = \sum_{i \in V \setminus S} \frac{1}{d_G(i,S)}$$

- **Decay centrality**:

$$C_2(S,G) = \sum_{i \in V \setminus S} \delta^{d_G(i,S)},$$

  where $0 < \delta < 1$

- k**-step reach centrality**:

$$C_3(S,G) = \sum_{i \in V \setminus S} \mathbb{1}_{\{d_G(i,S) \leq k\}},$$

  where $\mathbb{1}_{\{\}}$ denotes an indicator function

Scale-free networks

A scale-free network is a network whose degree distribution follows a power law, at least asymptotically. That is, the fraction $P(k)$ of nodes in the network having k connections to other nodes goes for large values of k as

$$P(k) \sim k^{-\gamma}$$

where γ is a parameter whose value is typically in the range $2 < \gamma < 3$, although occasionally, it may lie outside these bounds.

- the first moment (location) of $k^{-\gamma}$ is finite
- the second moment (scale parameter) of $k^{-\gamma}$ is infinite, hence the name.

References

Everett, M. G. and S. P. Borgatti (1999). The centrality of groups and classes. *The Journal of Mathematical Sociology 23*(3), 181–201.

Matsypura, D., A. Veremyev, E. L. Pasiliao, and O. A. Prokopyev (2023). Finding the most degree-central walks and paths in a graph: Exact and heuristic approaches. *European Journal of Operational Research 308*(3), 1021–1036.

Puzis, R., Y. Elovici, and S. Dolev (2007). Fast algorithm for successive computation of group betweenness centrality. *Physical Review. E 76*(5 Pt 2), 056709.

Shimbel, A. (1953). Structural parameters of communication networks. *The Bulletin of Mathematical Biophysics 15*(4), 501–507.