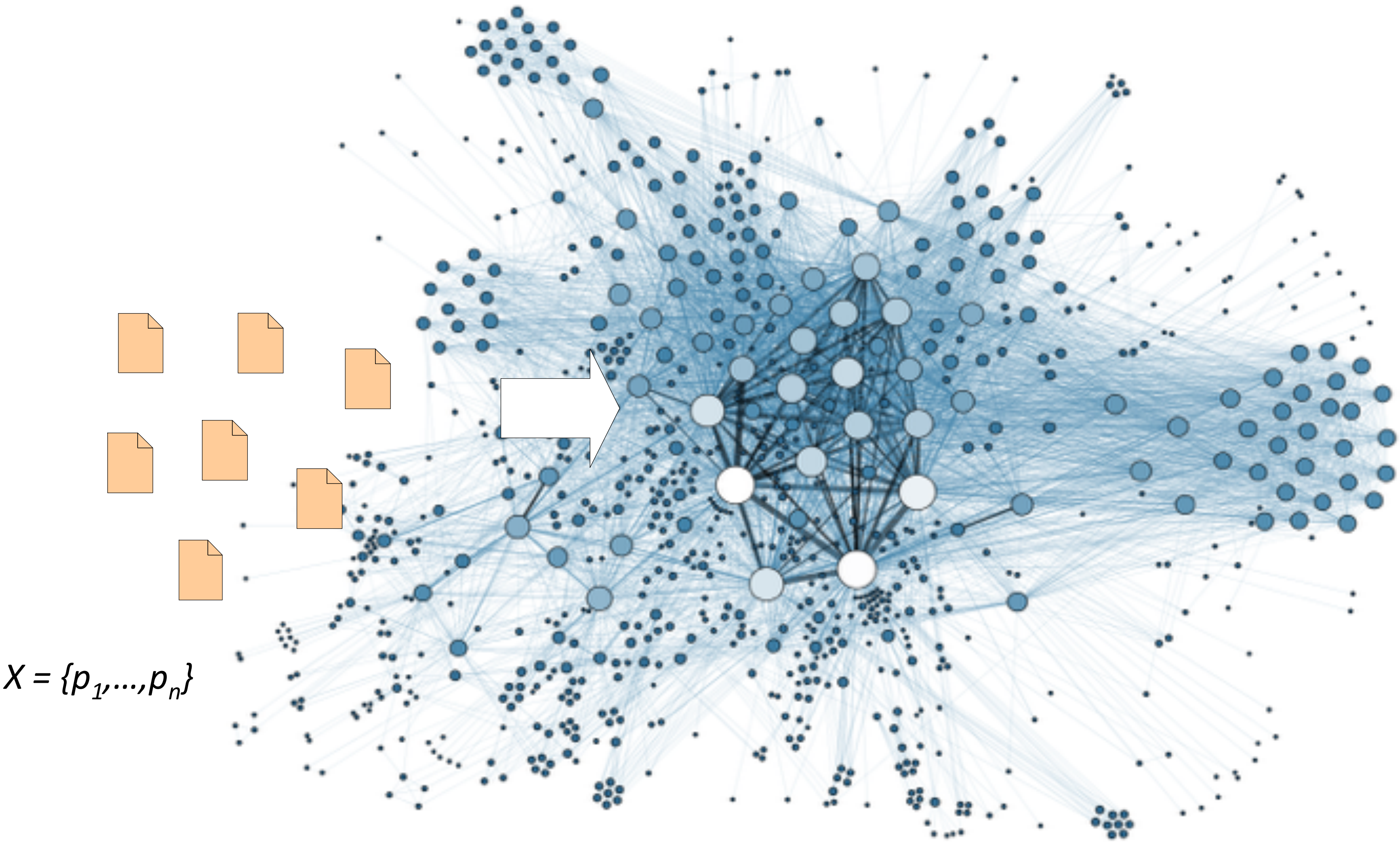
Optimal Network Structures for Decentralized Search

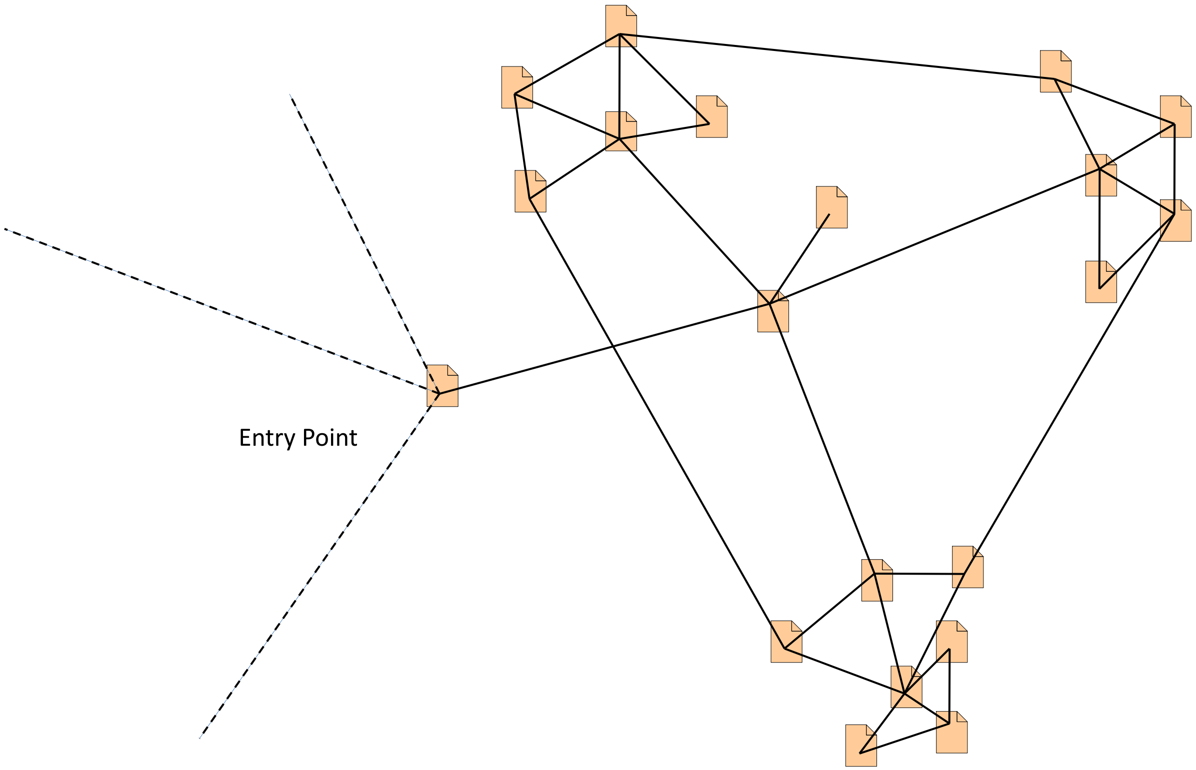
Alexander Ponomarenko, Irina Utkina, Mikhail Batsyn
National Research University Higher School of Economics
aponomarenko@hse.ru, iutkina@hse.ru, mbatsyn@hse.ru

Beauty is perfection. Perfection is optimality.

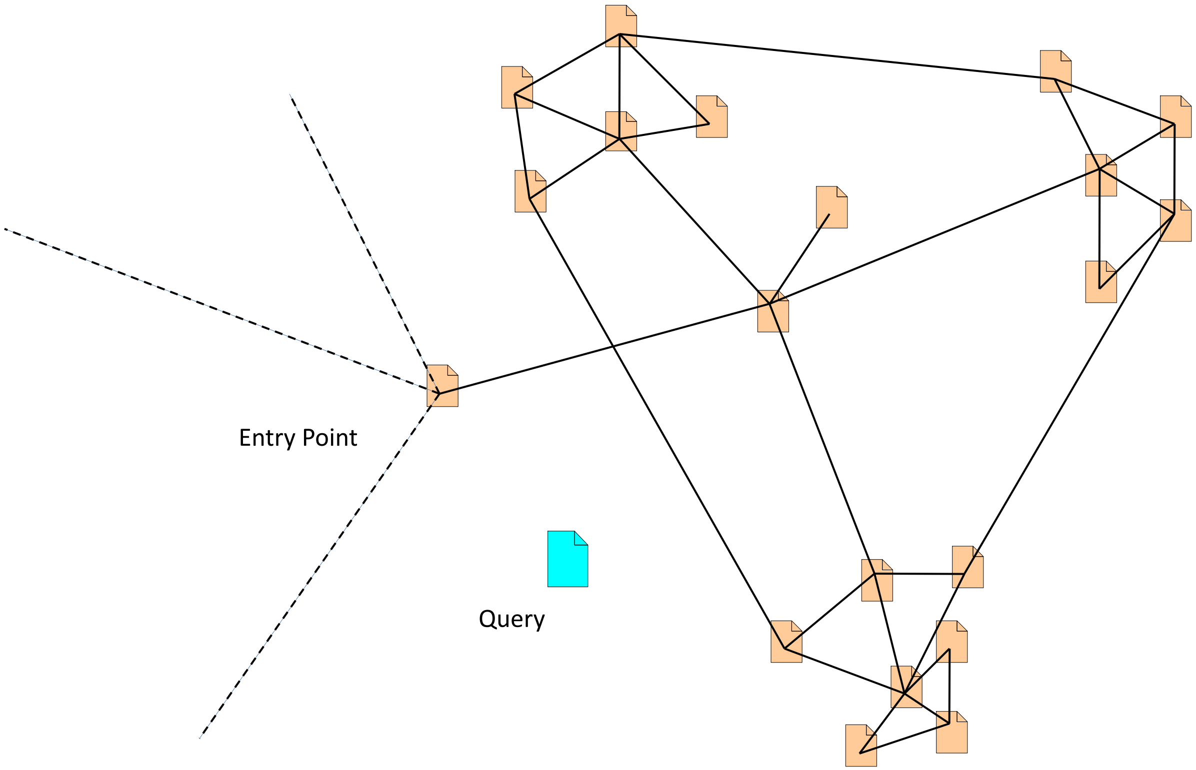
One approach for kNN indexing is to build a graph G(X,E) and use greedy walk as a base for search algorithm



Search by greedy algorithm

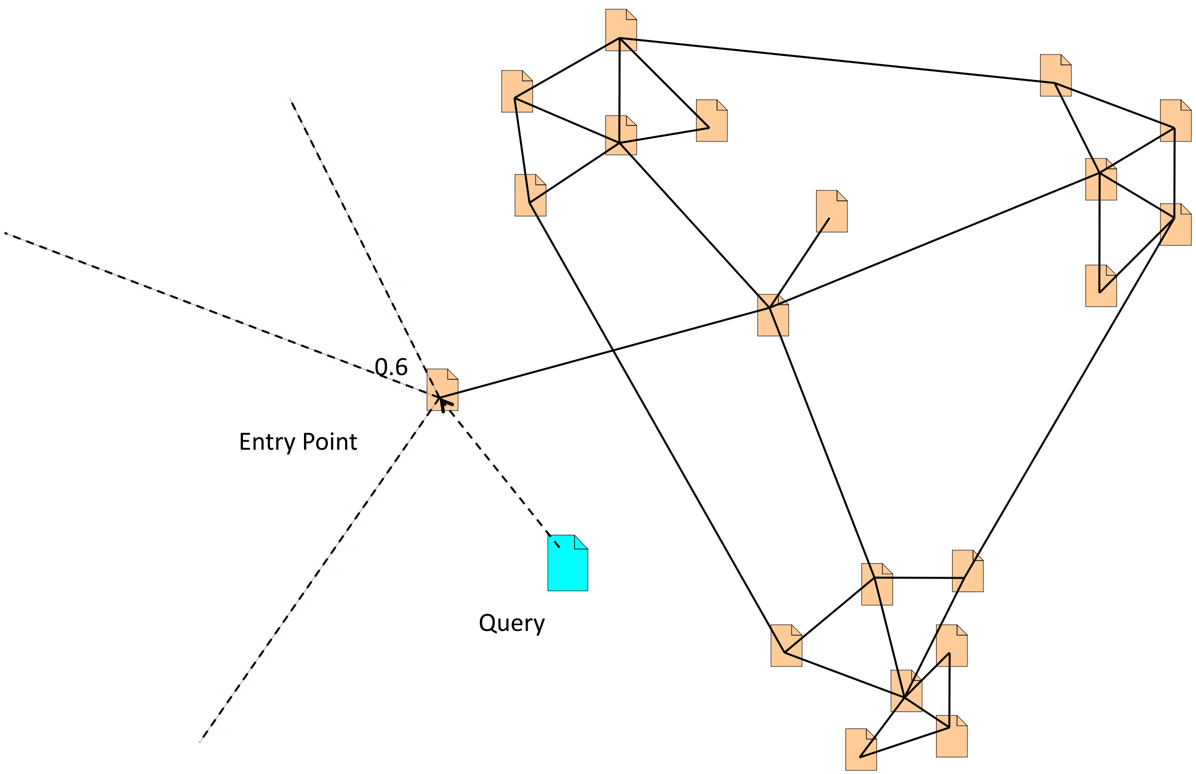


Search by greedy algorithm



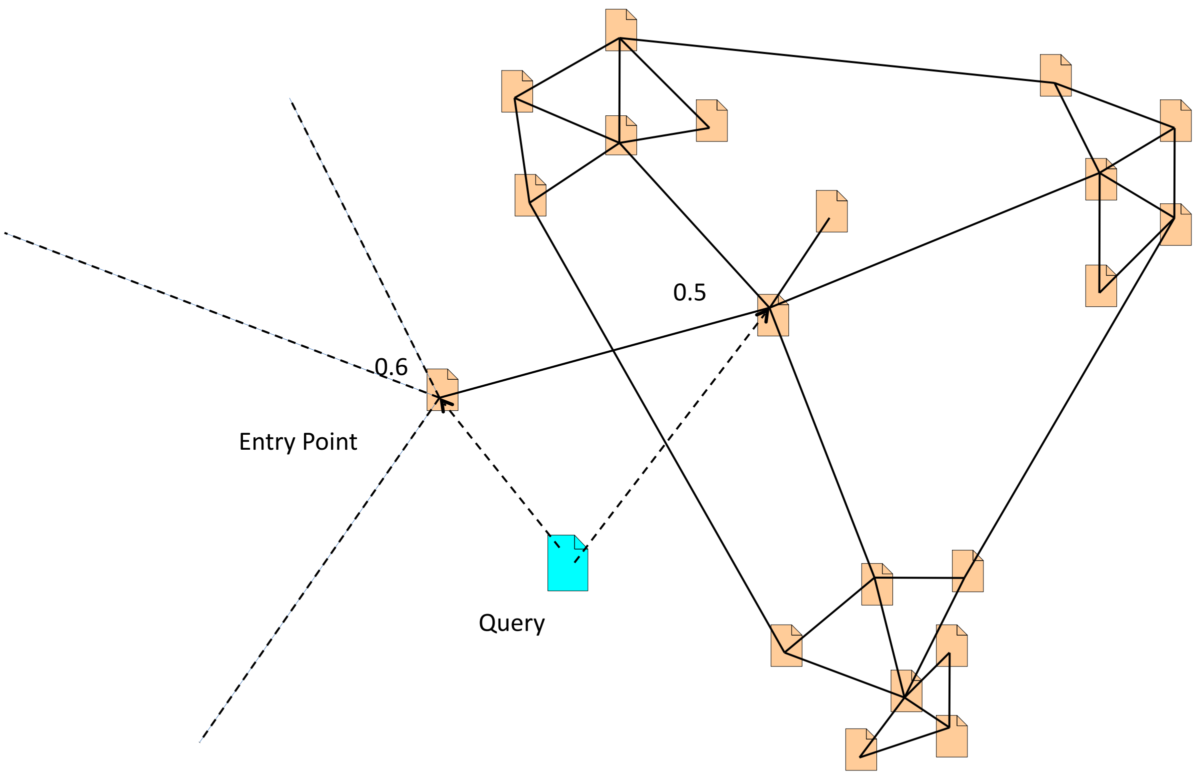
Search by greedy algorithm

Calculating distance to the entry point



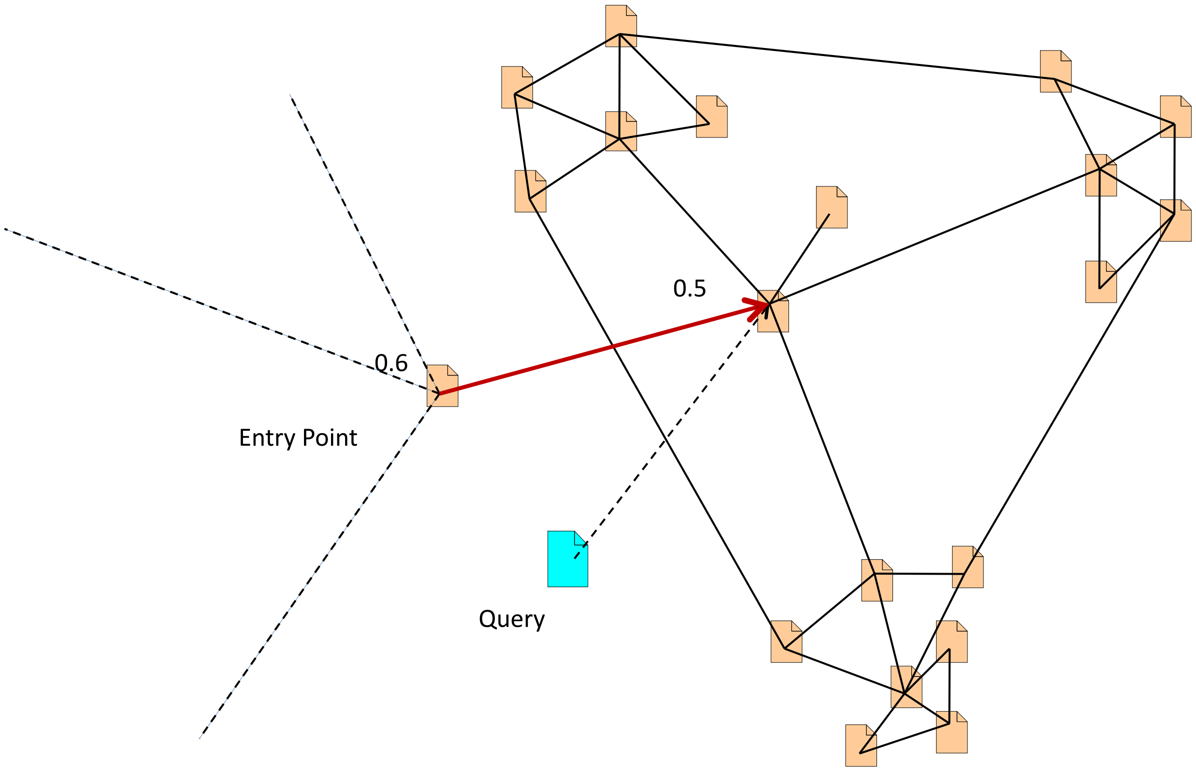
Search by greedy algorithm

Calculating metric for neighborhood



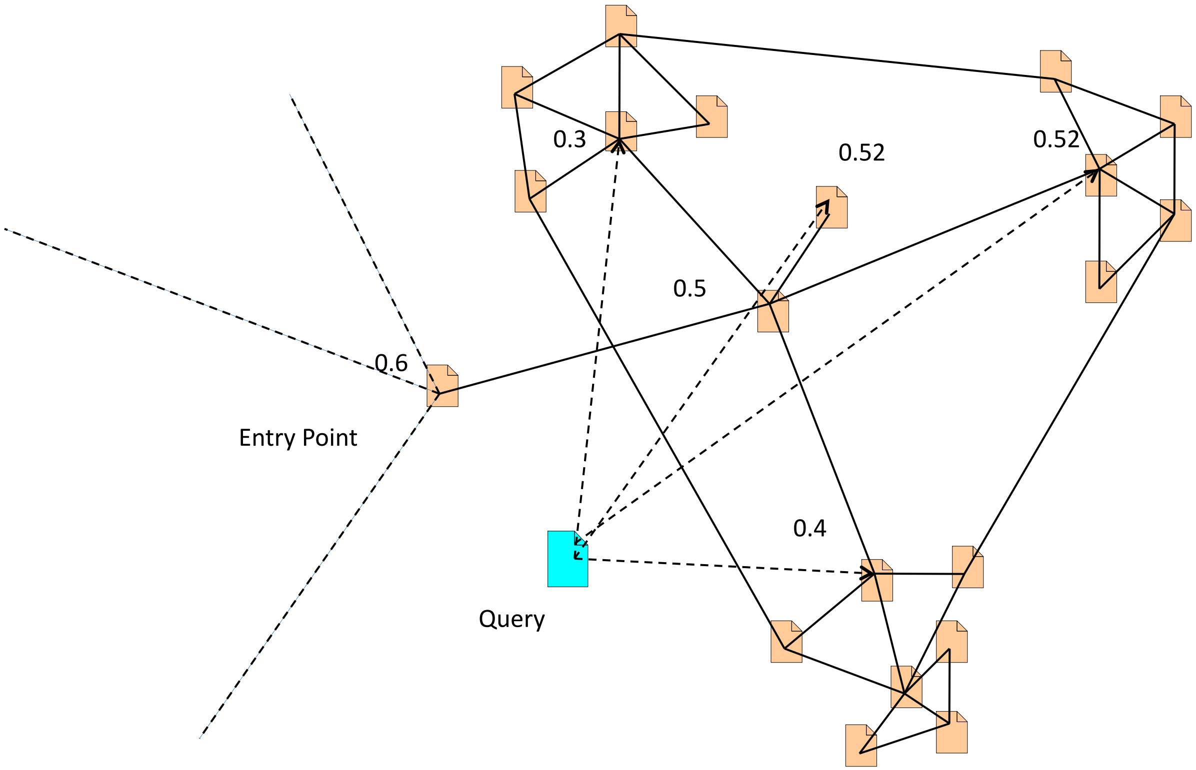
Search by greedy algorithm

Moving to object with minimal distance



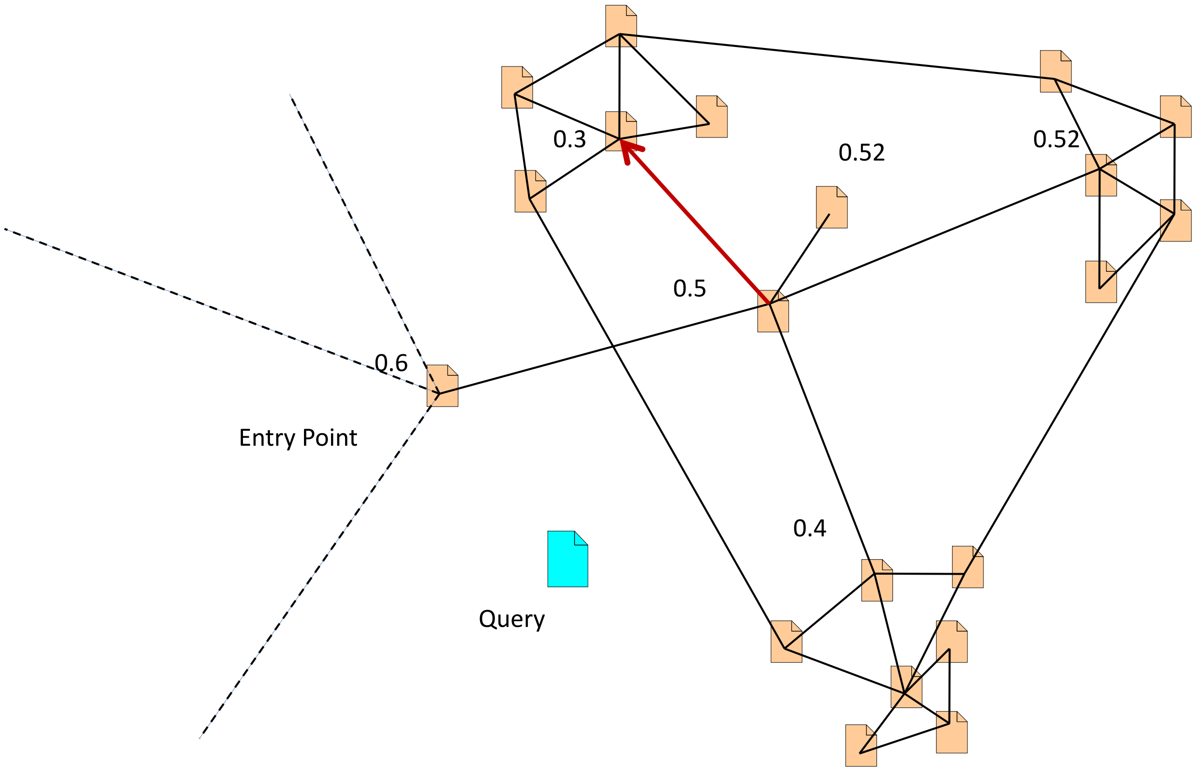
Search by greedy algorithm

Calculating metric for neighborhood



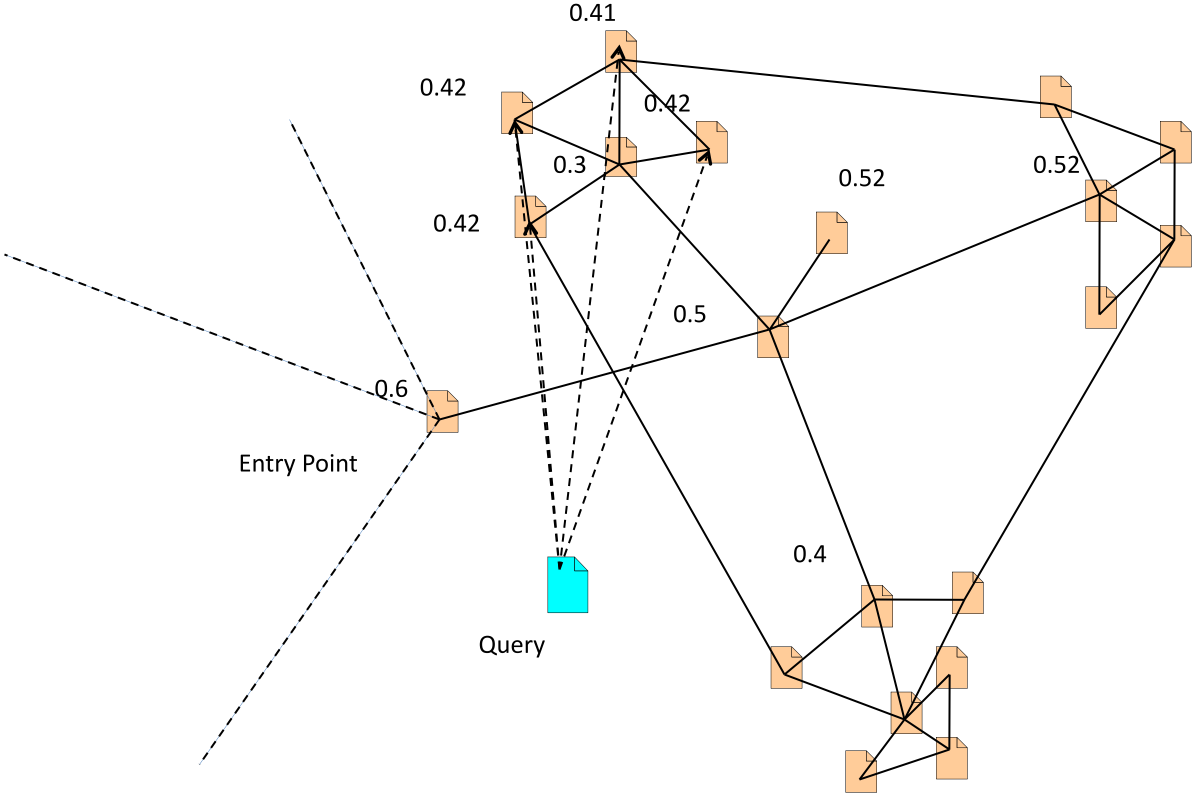
Search by greedy algorithm

Moving to object with minimal distance



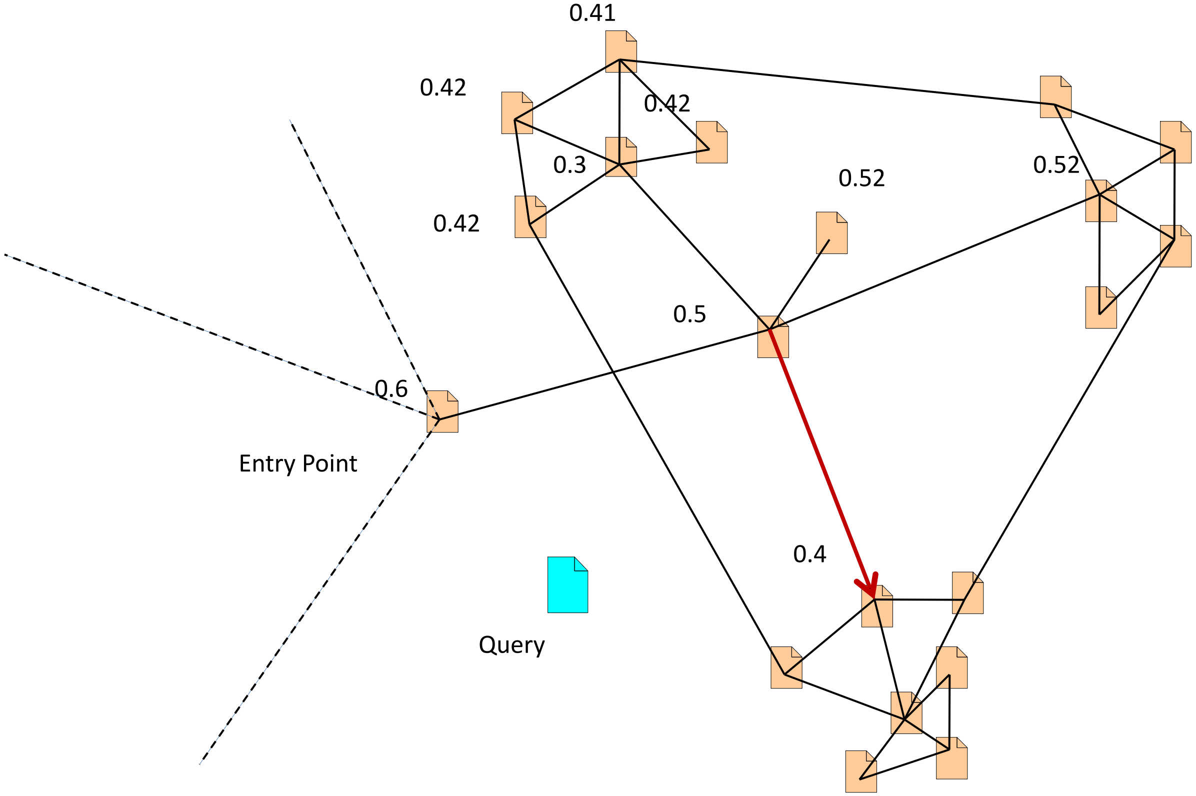
Search by greedy algorithm

Calculating metric for neighborhood



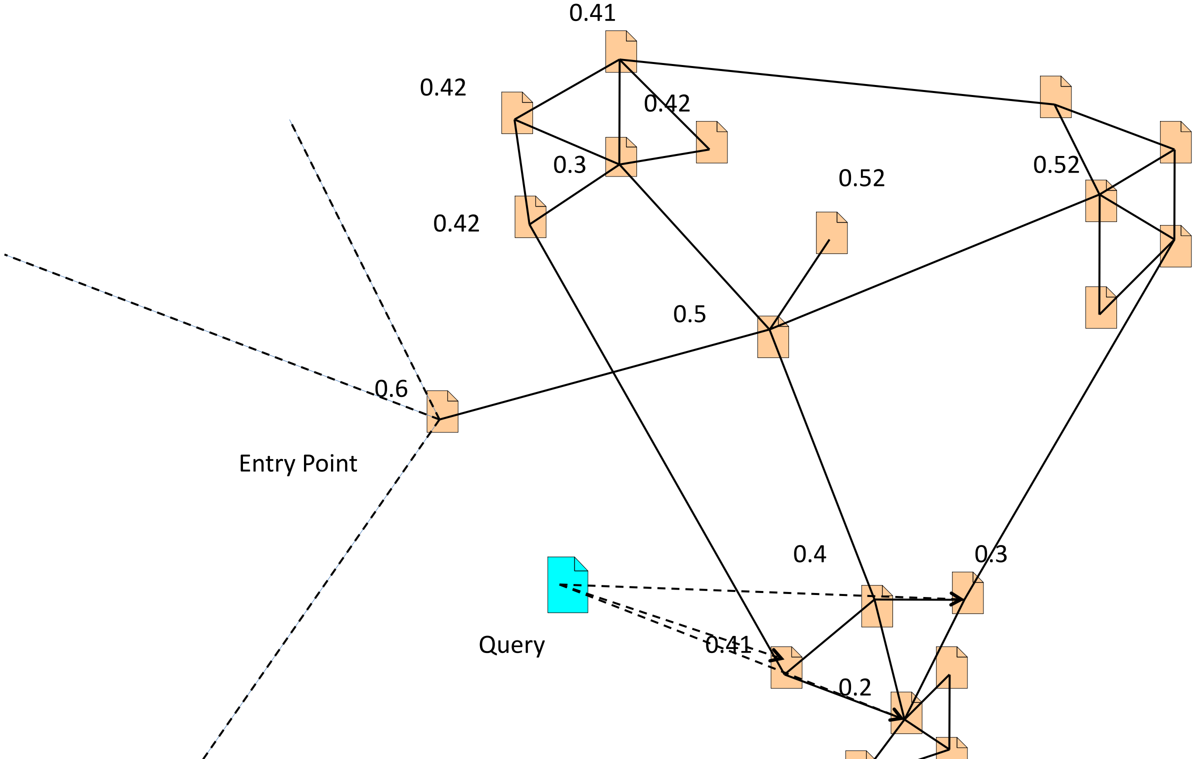
Search by greedy algorithm

Moving to object with minimal distance



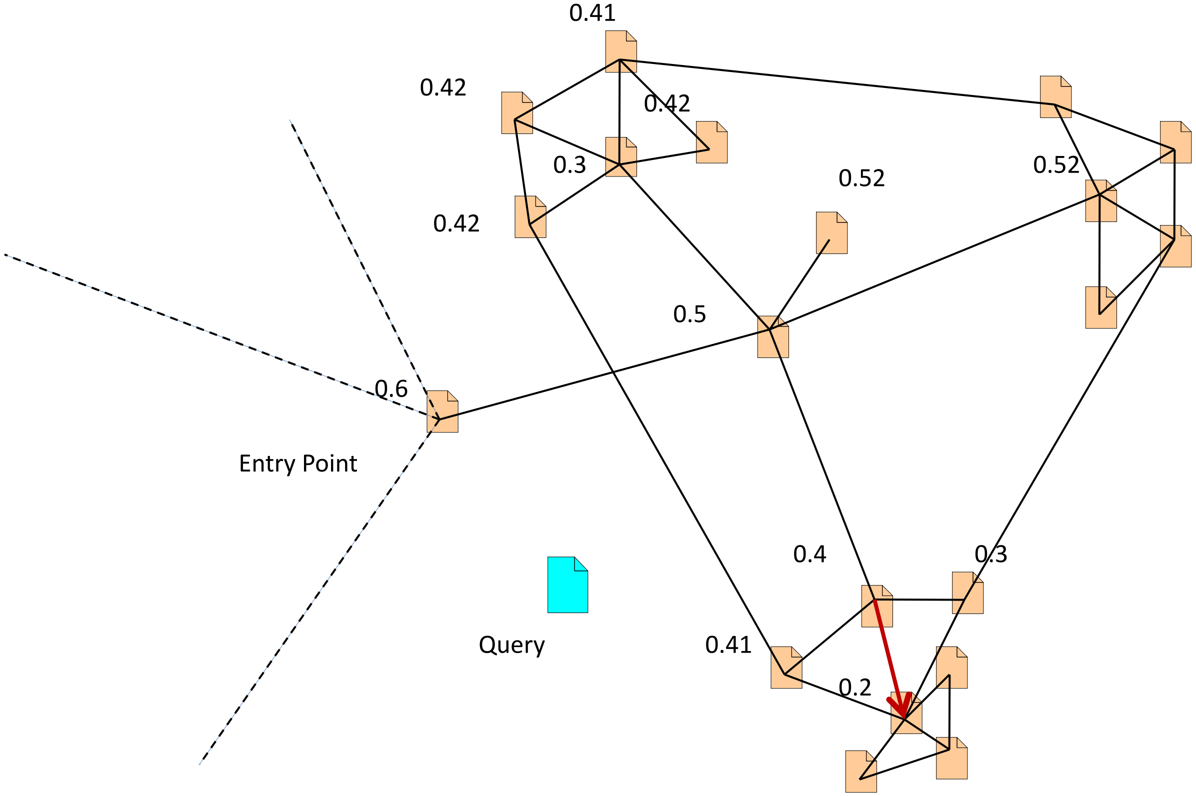
Search by greedy algorithm

Calculating metric for neighborhood



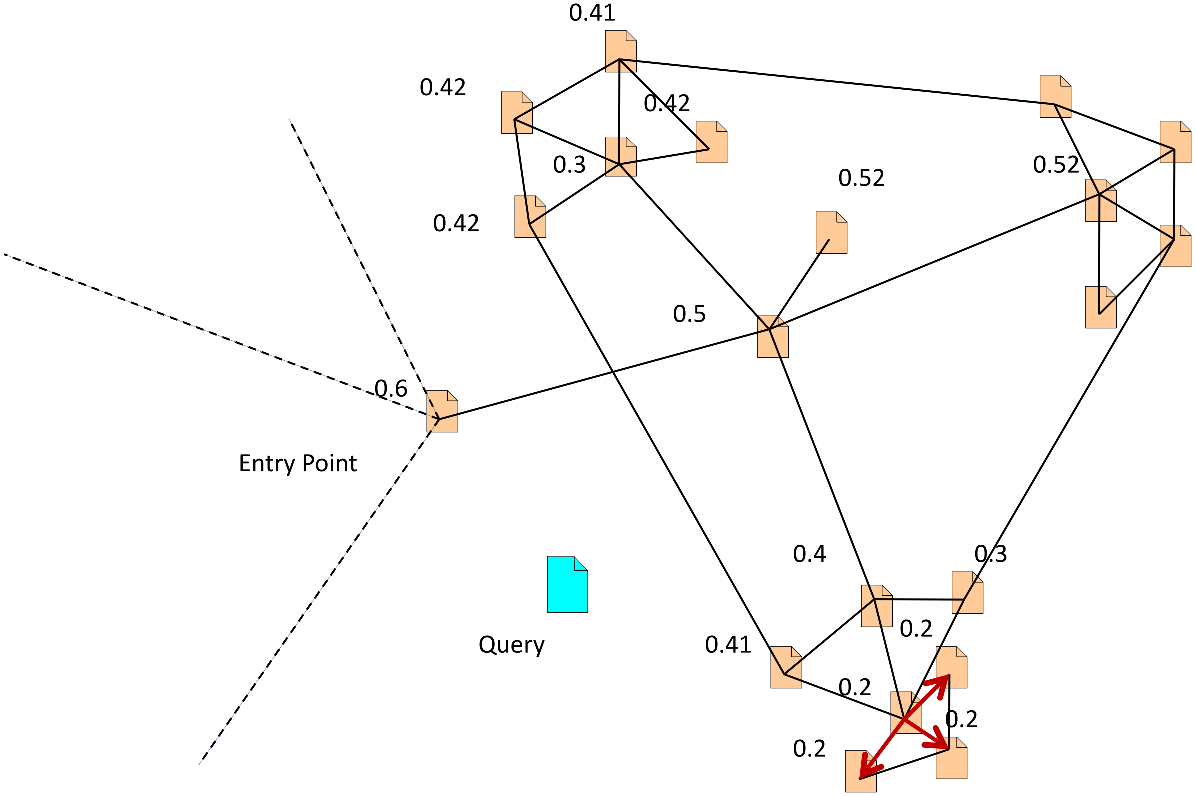
Search by greedy algorithm

Moving to object with minimal distance



Search by greedy algorithm

Return all matching objects as results



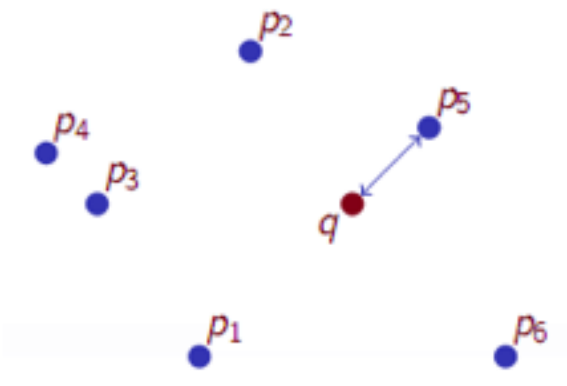
Nearest Neighbor Search

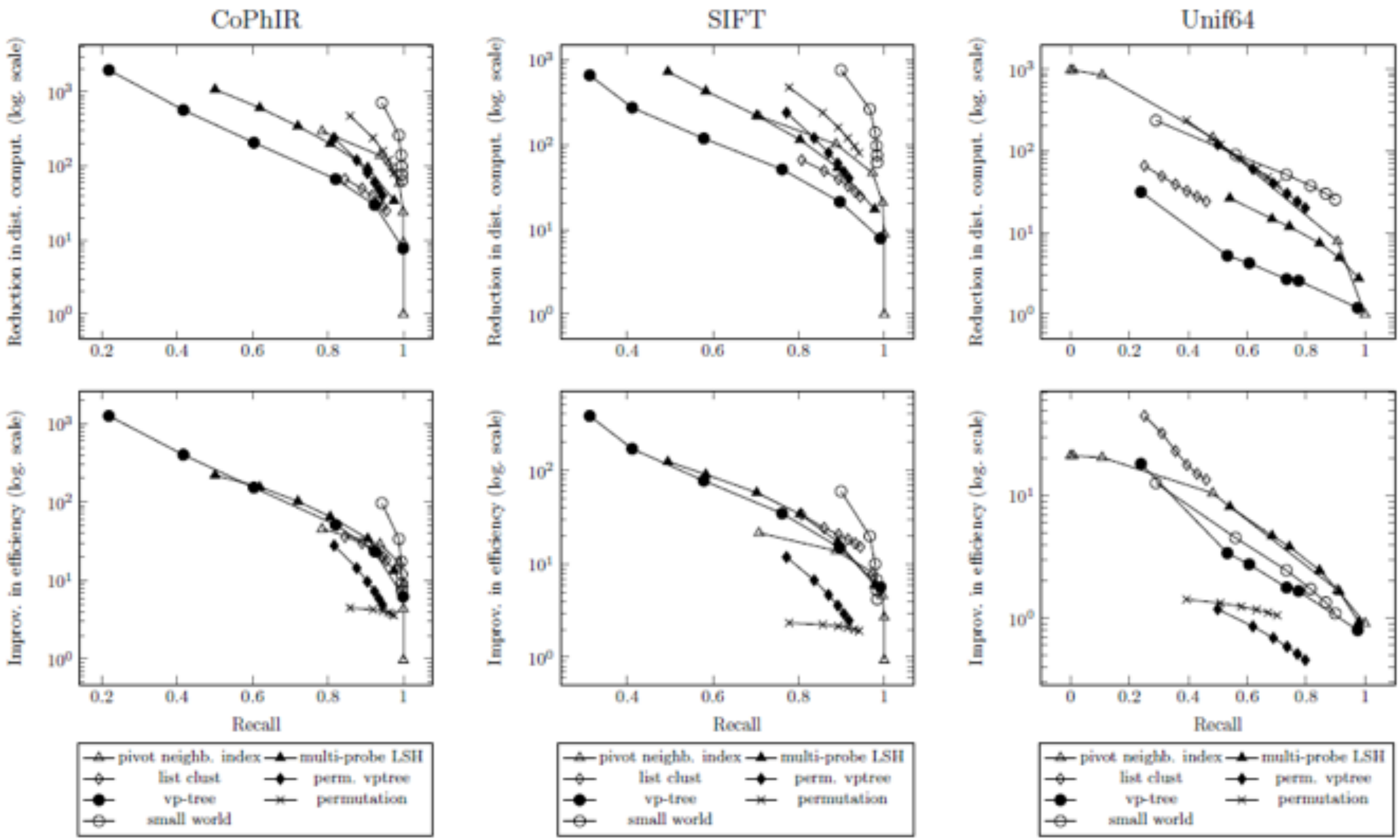
Let D – domain

$d: D \times D \to R_{[0;+\infty)}$ - distance function which satisfies properties:

- strict positiveness: d(x, y) > 0 ⇔ x ≠ y,
- symmetry: d(x, y) = d(y, x),
- reflexivity: d(x, x) = 0,
- triangle inequality: d(x, y)+d(y, z) ≥ d(x, z).

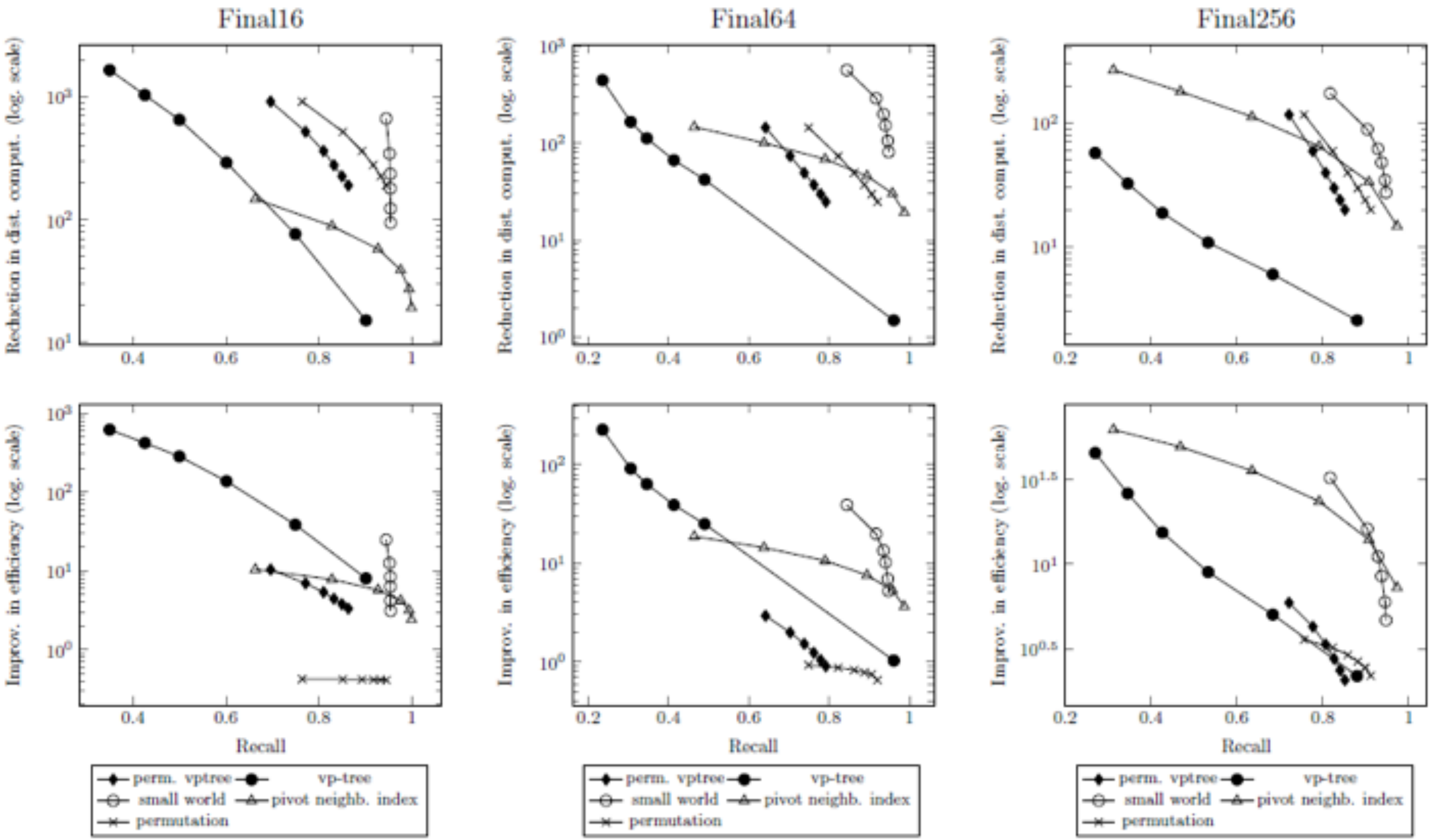
Given a finite set $X = \{p_1,\dots,p_n\}$ of n points in some metric space *(D, d),* need to build a data structure on X so that for a given query point $q \in D$ one can find a point $p \in X$ which minimizes *d(p, q) with as few distance computations as possible*





Performance of a 10-NN search for L_2: plots in the same column correspond to the same data set

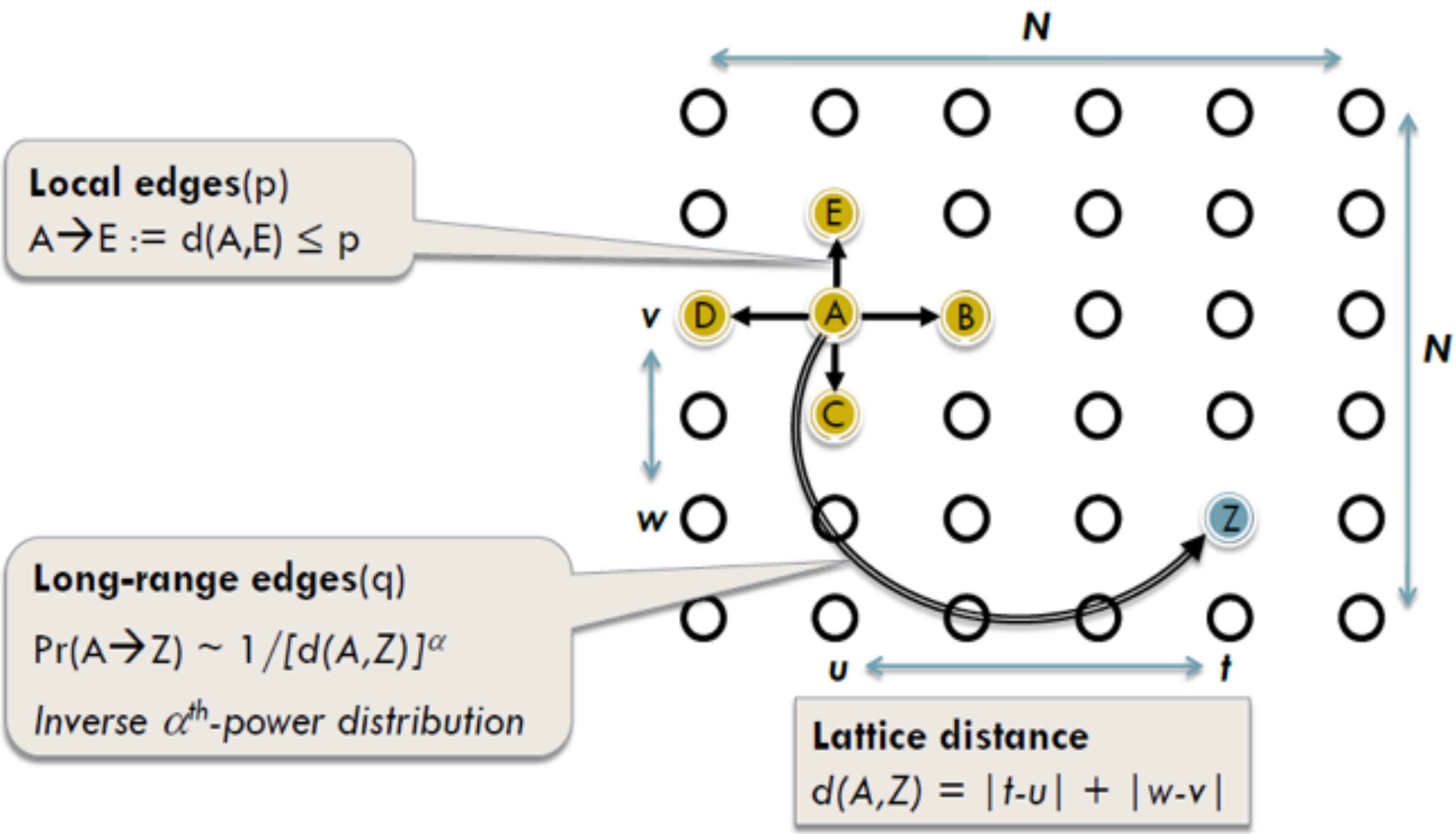
[Ponomarenko A. et al. Comparative Analysis of Data Structures for Approximate Nearest Neighbor Search //DATA ANALYTICS 2014, The Third International Conference on Data Analytics. – 2014. – C. 125-130.]



KL-divergence: $d(x,y) = \sum x_i \log \frac{x_i}{y_i}$

Final16, Final64, and Final256: are sets of 0.5 million topic histograms generated using the Latent Dirichlet Allocation (LDA).

Kleinberg’s Navigable Small World



[Kleinberg J. The small-world phenomenon: An algorithmic perspective //Proceedings of the thirty-second annual ACM symposium on Theory of computing. – ACM, 2000. – C. 163-170.]

Family of network models with parameter α

$\alpha = 0$

Long-range contacts chosen independently of their position (~Watts-Strogatz model)

$\alpha > 0$

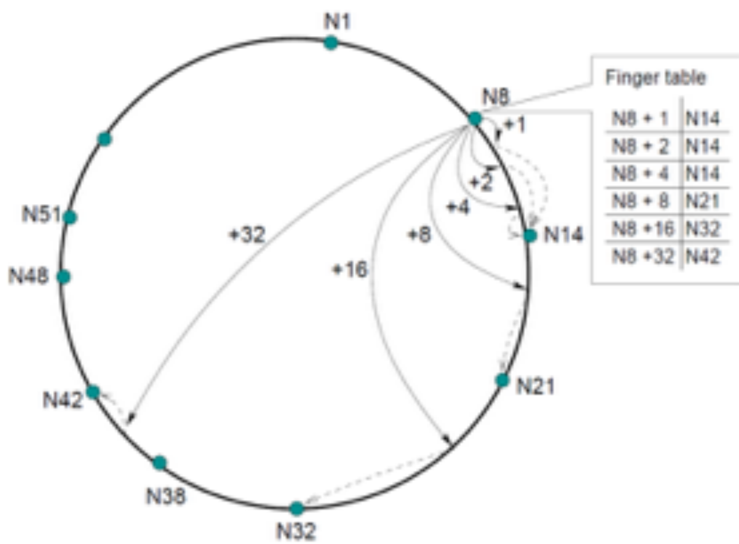
Long-range contacts tend to cluster in the nodes' vecinity

Which α *yields an* effectively *navigable network?*

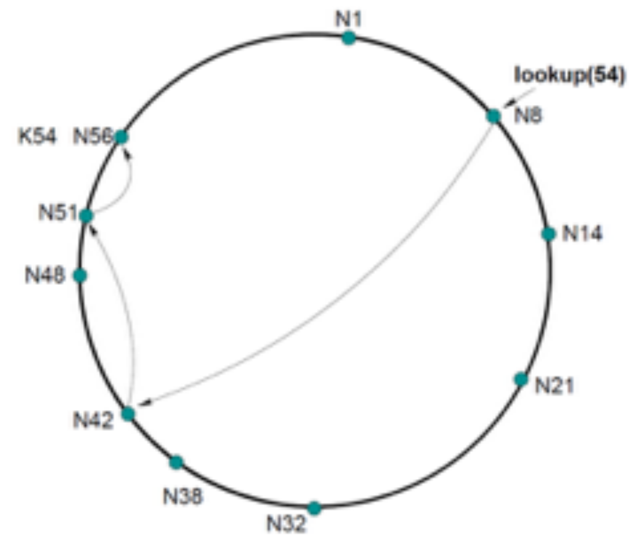
Expected delivery time T

- Expected number of steps to reach the destination
- Shortness (small T) of paths is defined as ***polylogarithmic***

Structured Peer-to-Peer Networks: Chord Protocol



Routing table of node «N8»

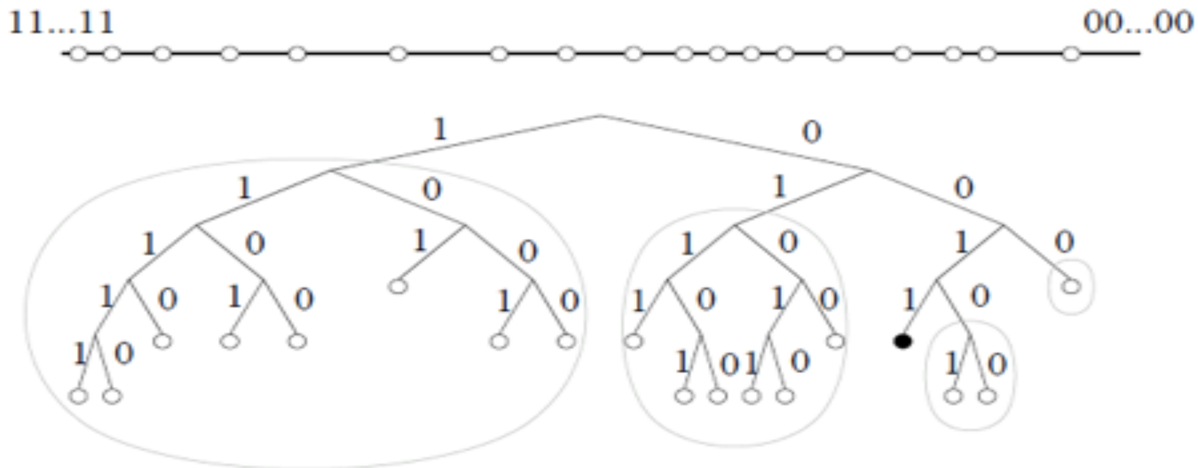


Searching of key 54 staring from «N8».

Distance function: *d(x,y)= (y – x) mod 2^m*

Each node, *n*, maintains a routing table with (at most) *m* entries, called the *finger table*. The i-th entry in the table at node *n* contains the identity of the first node, s, that succeeds *n* by at least 2^(*i*-1) on the identifier circle, i.e., s = successor(*n*+2^(*i*-1)), where 1<= *i*<= *m*

Structured Peer-to-Peer Networks: Kademlia



Identifier space of Kademlia

Distance function: *d(x,y)= x xor y*

Maymounkov P., Mazieres D. Kademlia: A peer-to-peer information system based on the xor metric // Peer-to-Peer Systems. – Springer Berlin Heidelberg, 2002. – C. 53-65.

DHT protocols and implementations

- Aeropike
- Apache Cassandra
- BATON Overlay
- Mainline DHT - Standard DHT used by BitTorrent (based on Kademlia as provided by Khashmir[16])
- CAN (Content Addressable Network)
- Chord
- Koorde
- Kademlia
- Pastry
- P-Grid
- Riak
- Tapestry
- TomP2P
- Voldemort_(distributed_data_store)

Applications employing DHTs

- BTDigg: BitTorrent DHT search engine
- cjdns: routing engine for mesh-based networks
- CloudSNAP: a decentralized web application deployment platform
- Codeen: web caching
- Coral Content Distribution Network
- FAROO: peer-to-peer Web search engine
- Freenet: a censorship-resistant anonymous network
- GlusterFS: a distributed file system used for storage virtualization
- GNUnet: Freenet-like distribution network including a DHT implementation
- Hazelcast: Open-source in-memory data grid
- I2P: An open-source anonymous peer-to-peer network.
- I2P-Bote: serverless secure anonymous e-mail.
- JXTA: open-source P2P platform
- Oracle Coherence: an in-memory data grid built on top of a Java DHT implementation
- Retroshare: a Friend-to-friend network[17]
- YaCy: a distributed search engine
- Tox: an instant messaging system intended to function as a Skype replacement
- Twister: a microblogging peer-to-peer platform
- Perfect Dark: a peer-to-peer file-sharing application from Japan

Boolean non-linear programming formulation for optimal graph structure

Decision variables

$$x_{ij} = \begin{cases} 1, & \text{if edge } (i, j) \text{ belongs to the solution} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

$$y_{ij}^{k} = \begin{cases} 1, & \text{if vertex } k \text{ belongs to the greedy walk from } i \text{ to } j \\ 0, & \text{otherwise} \end{cases} \quad (2)$$

Objective function

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} O(i, j) \quad (2)$$

$$O(i, j) = \left| \left\{ l \in V : \exists k \; x_{lk} = 1 \text{ and } y_{ij}^{k} = 1 \right\} \right| \quad (4)$$

Constraints

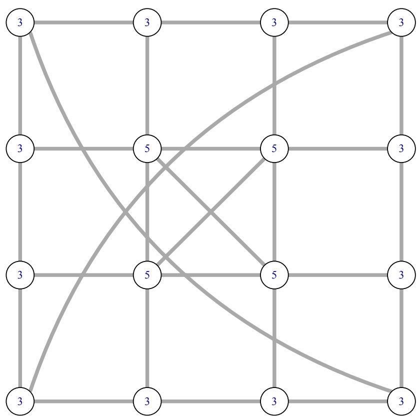
$$x_{ii} = 0 \quad \forall i \in V \quad (5)$$

$$y_{ij}^{i} = y_{ij}^{j} = 1 \quad \forall i, j \in V \quad (6)$$

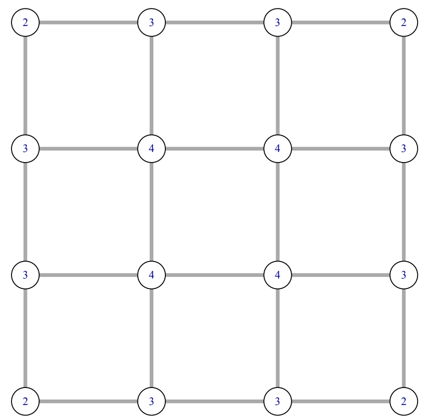
$$\sum_{k=1}^{n} x_{lk} y_{ij}^{k} \geq y_{ij}^{l} \quad \forall i, j, l \in V \quad (7)$$

$$l^{*} = \underset{l \in V : x_{kl} = 1}{\arg\min}(d(l, j)) \Rightarrow y_{ij}^{l^{*}} \geq y_{ij}^{k} \quad \forall i, j, k \in V, j \neq i, k \neq j \quad (8)$$

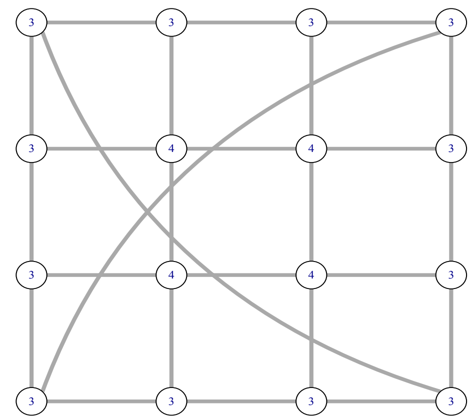
The exact solutions founded by branch and bound algorithm for regular lattice 4x4



L_2 $f = 1800$

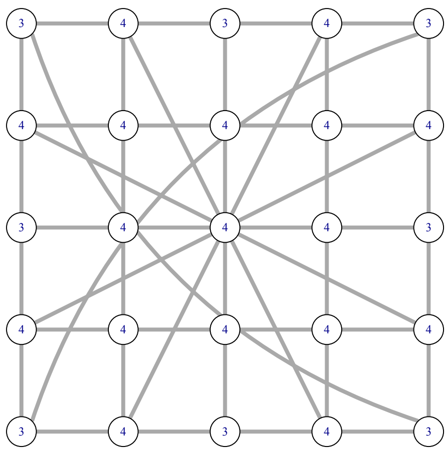


L_1 $f = 1728$

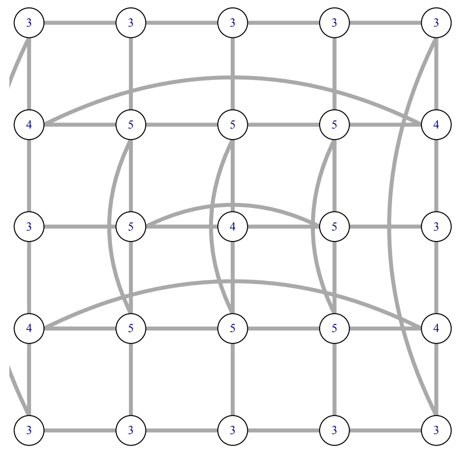


L_∞ $f = 1816$

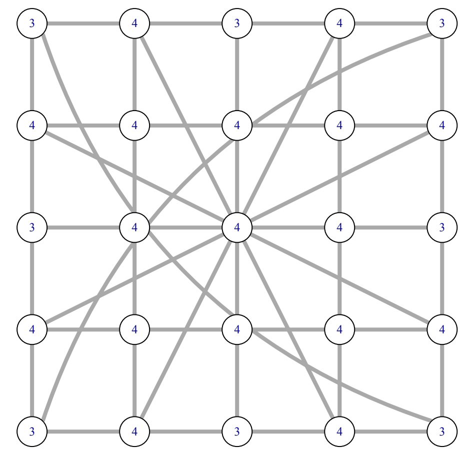
The solutions founded by heuristic for regular lattice 5x5



L_2 $\quad f = 5593$

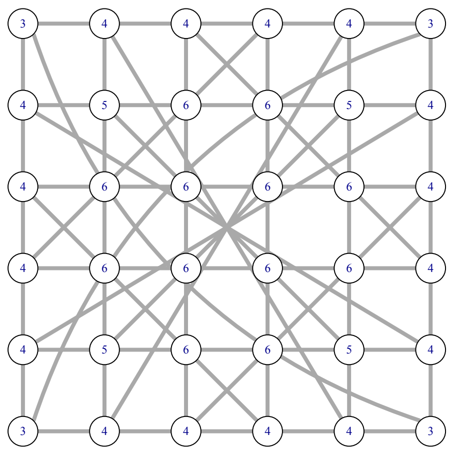


L_1 $\quad f = 5429$

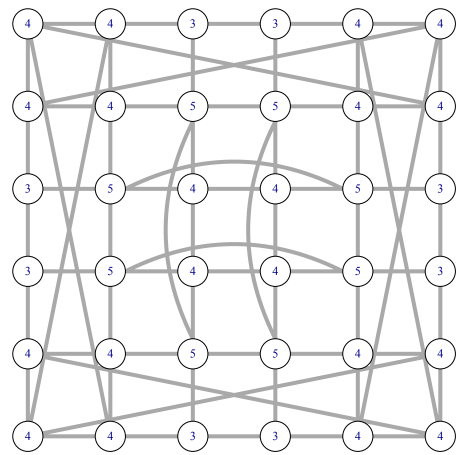


L_∞ $\quad f = 5619$

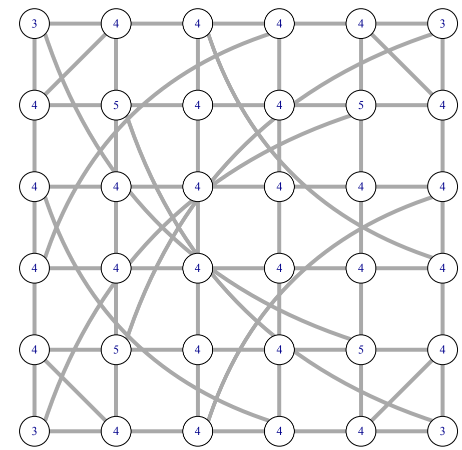
The solutions founded by heuristic for regular lattice 6x6



L_2 $f = 13452$

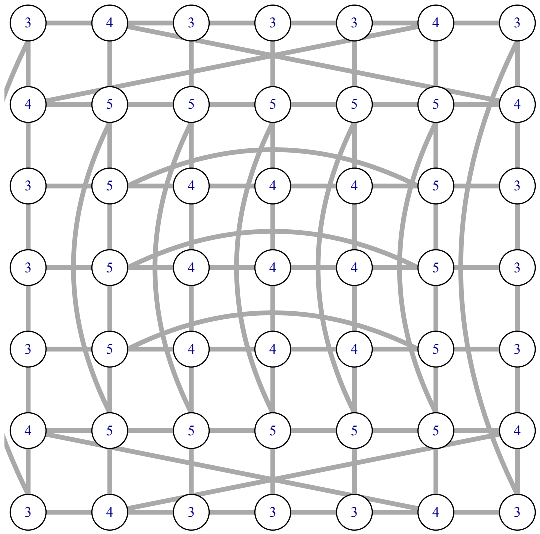


L_1 $f = 13356$

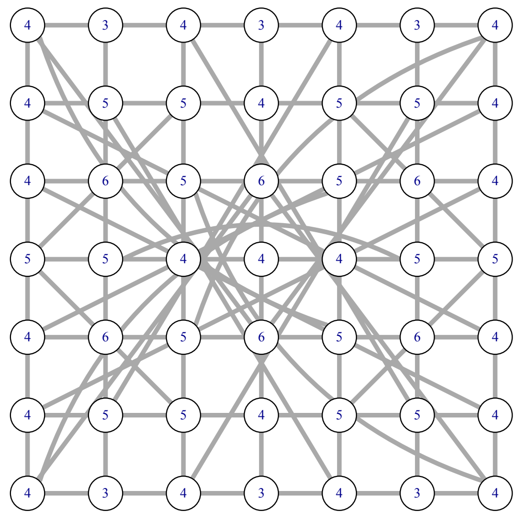


L_∞ $f = 13882$

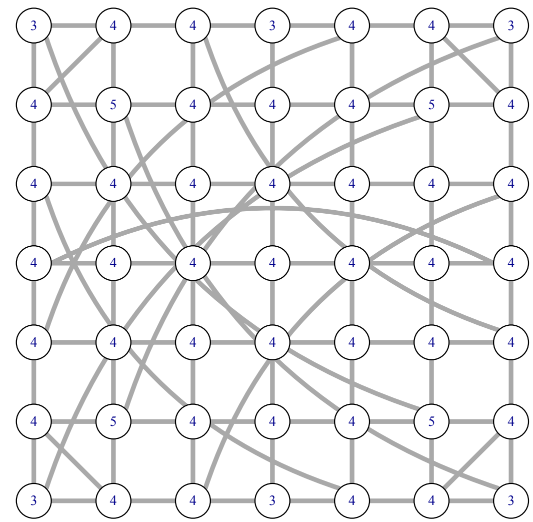
The solutions founded by heuristic for regular lattice 7x7



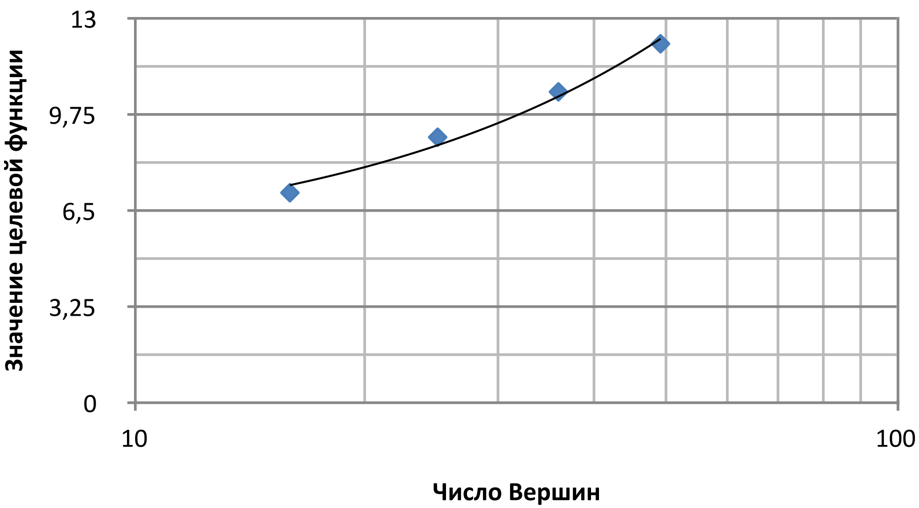
L_1 $\boldsymbol{f} = 28177$



L_2 $\boldsymbol{f} = 29095$



L_∞ $\boldsymbol{f} = 29489$


Значение целевой функции
13
9,75
6,5
3,25
0
10
100
Число Вершин