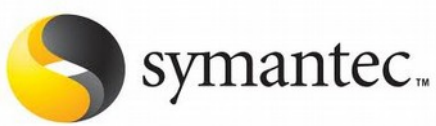
Fast distributed k-nn graph update

Thibault Debatty, Fabio Pulvirenti,
Pietro Michiardi & Wim Mees



The context : Visual SPAM analysis

- Large, distributed k-NN graph
- E.g. *Scalable k-NN based text clustering* A Lulli, T Debatty M Dell'Amico, P Michiardi, L Ricci
- Subject similarity: Jaro-Winkler (not a metric)

The problem

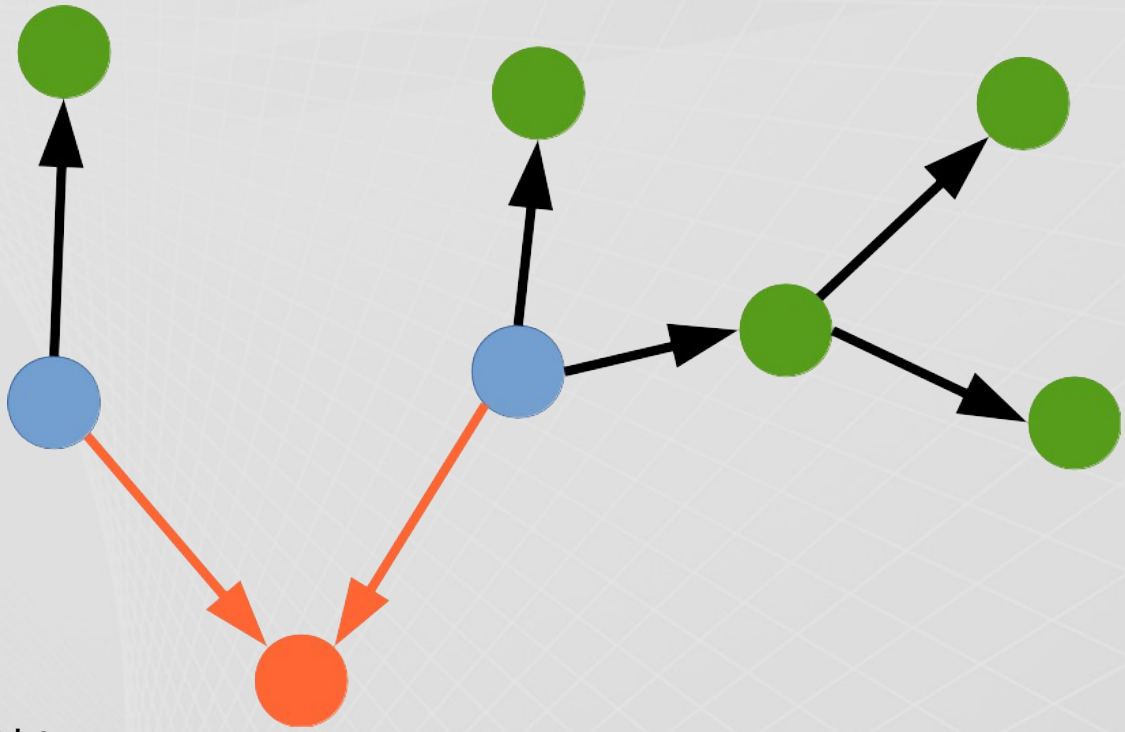
How to efficiently add or remove nodes?

Naive algorithm:

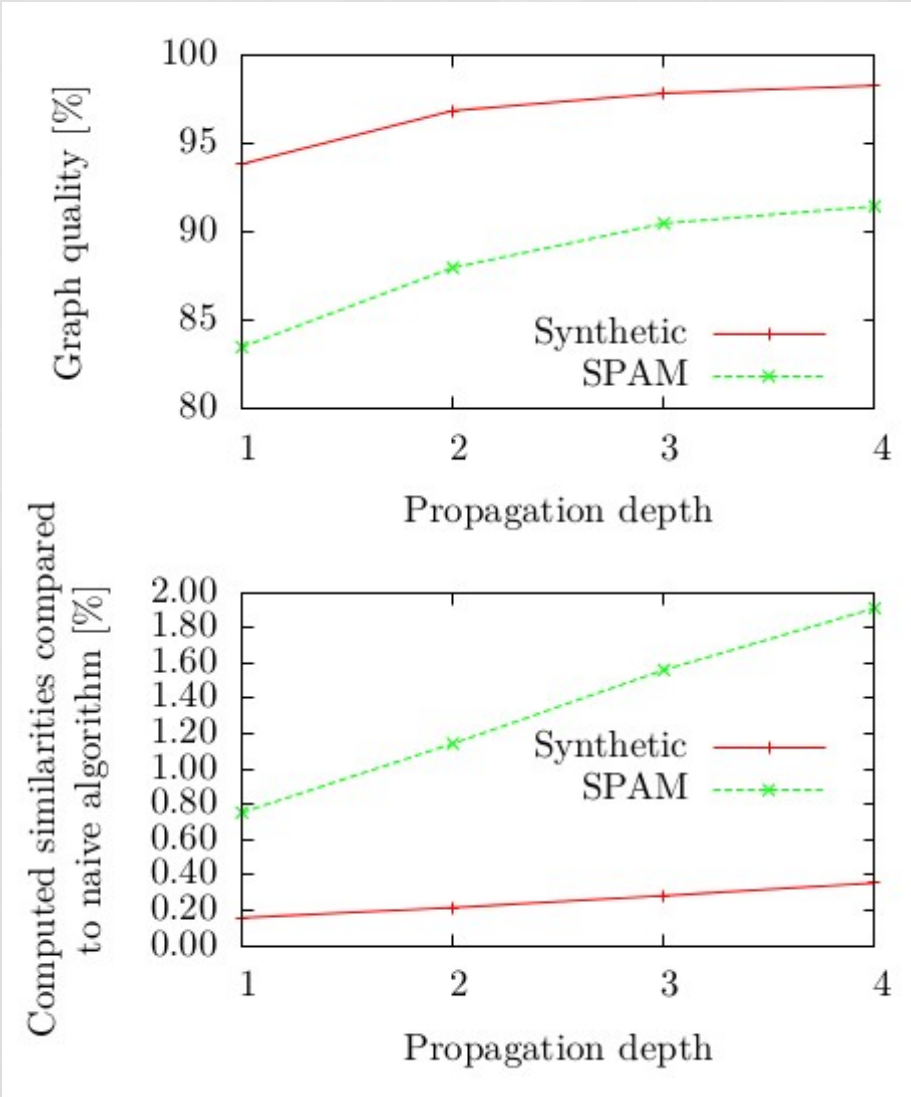
- Add: O(n) similarities
- Remove: O(kn) similarities

Remove a node

- Use propagation to identify candidates $O[(k+1)^{DEPTH+1}]$
- Find new neighbor $O[k^{DEPTH+2}]$



Remove a node

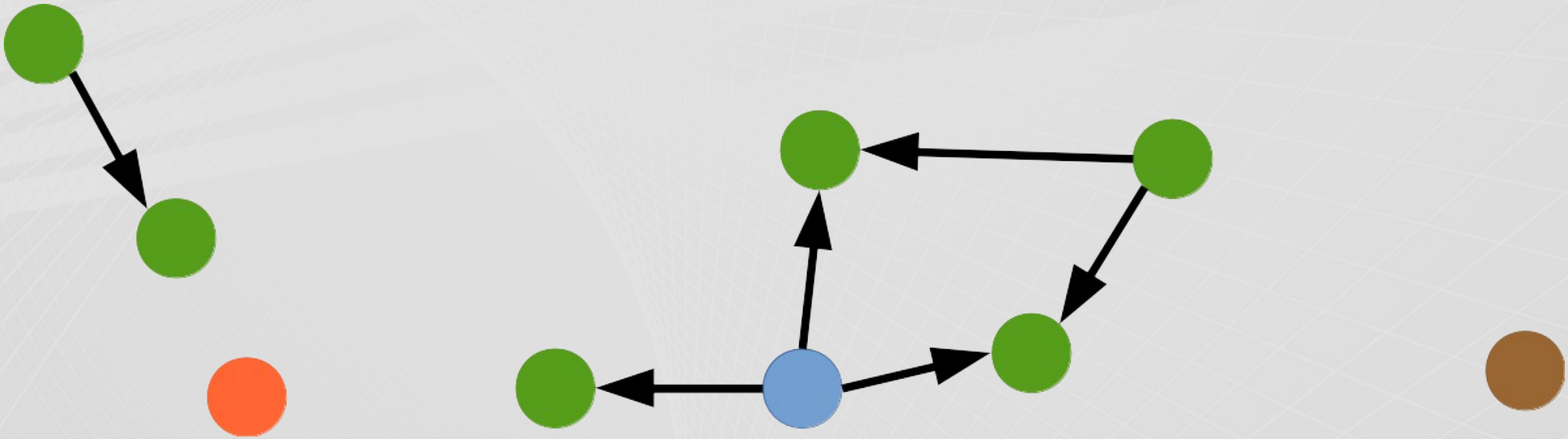


Add a node

- Search neighbors of new node
  - Distributed graph based NN search
  - Graph partitioning:
    Distributed balanced k-medoids clustering
- Use propagation to update existing nodes

Sequential graph based NN search

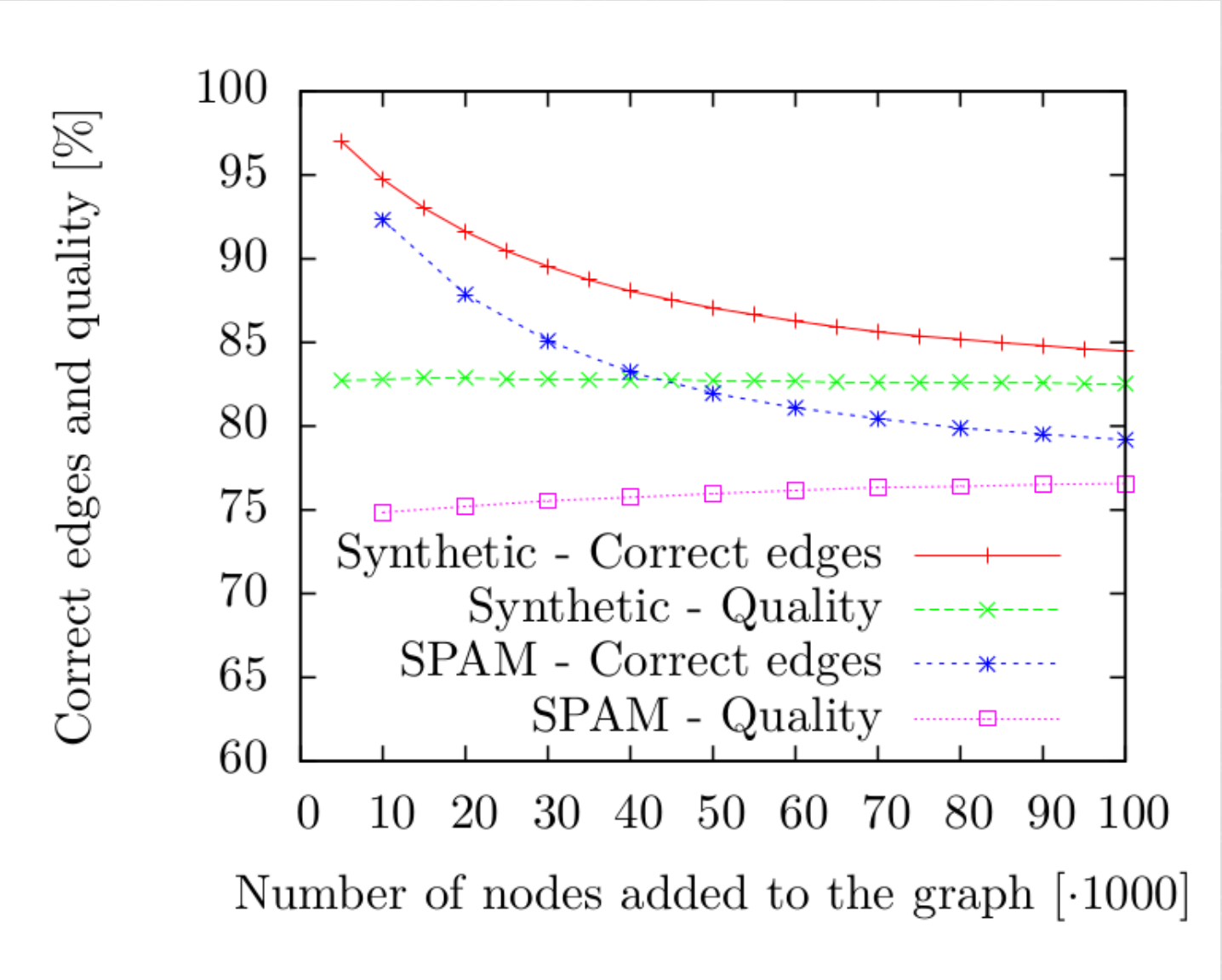
- Hill climbing with restart
- Eager iteration
- Smart starting node selection



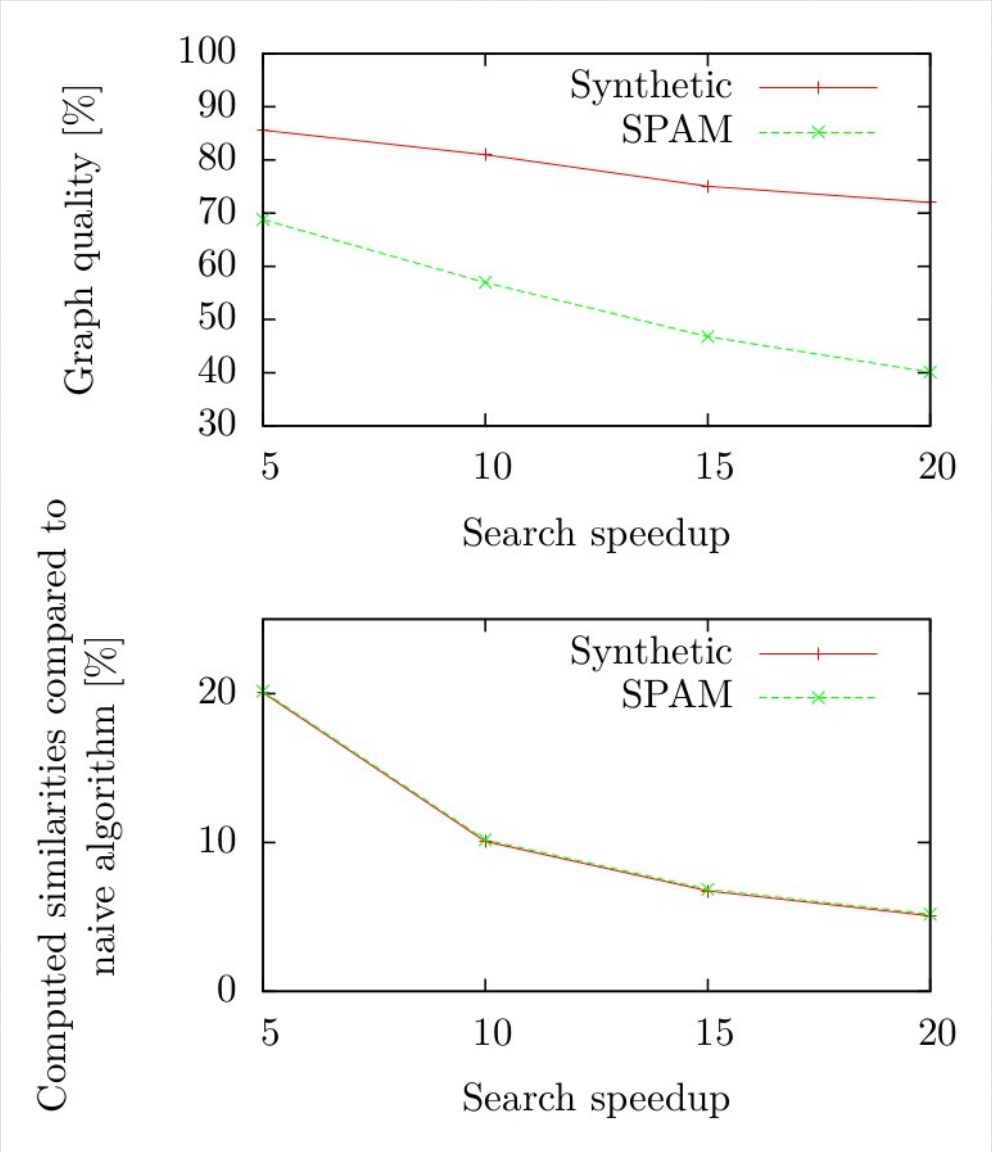
Distributed balanced k-medoids clustering

- Voronoï iteration
- Balanced: weight = 1 - size / capacity
- Distributed: randomized dataset

Add node


Correct edges and quality [%]
100
95
90
85
80
75
70
65
60
0
10
20
30
40
50
60
70
80
90
100
Number of nodes added to the graph [·1000]
Synthetic - Correct edges
Synthetic - Quality
SPAM - Correct edges
SPAM - Quality

Add node


Graph quality [%]
100
90
80
70
60
50
40
30
5
10
15
20
Search speedup
Synthetic
SPAM
Computed similarities compared to naive algorithm [%]
20
10
0
5
10
15
20
Search speedup
Synthetic
SPAM

Conclusions & future work...

- Fast add remove nodes
- Future:
  - Online algorithm and streaming framework
  - Simulated annealing based k-medoids clustering

Thank you!