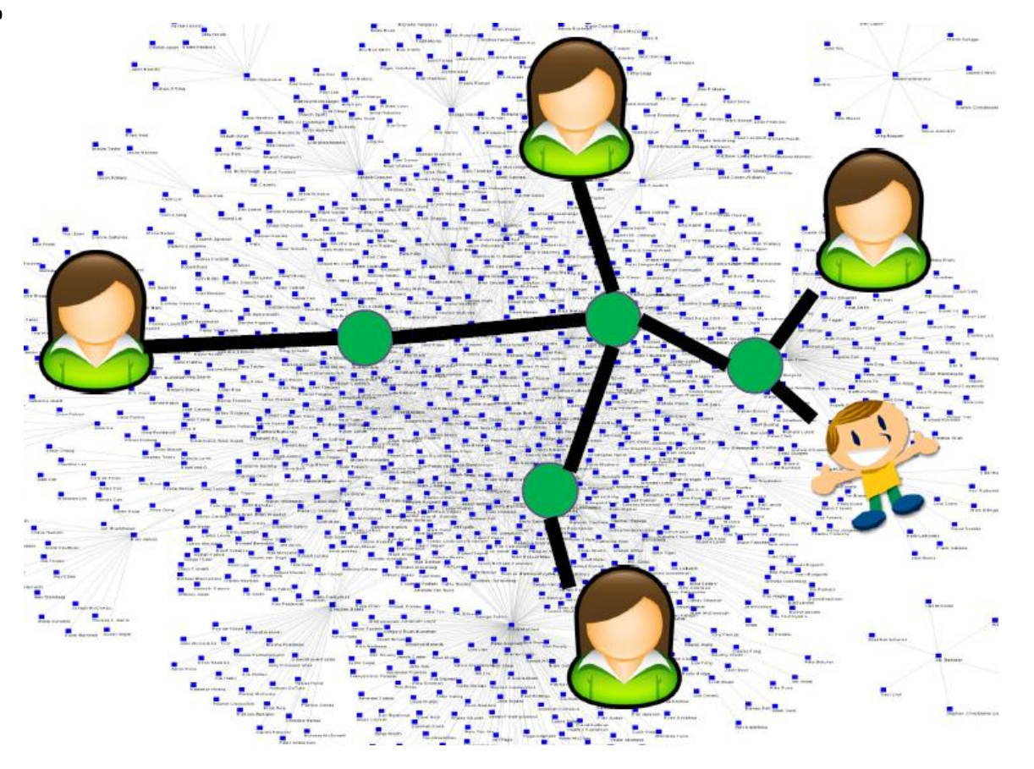
Memory-Efficient Fast Shortest Path Estimation in Large Social Networks

Volodymyr Floreskul, Konstantin Tretyakov (kt@ut.ee) and Marlon Dumas

Institute of Computer Science, University of Tartu, Estonia.

Shortest path estimation

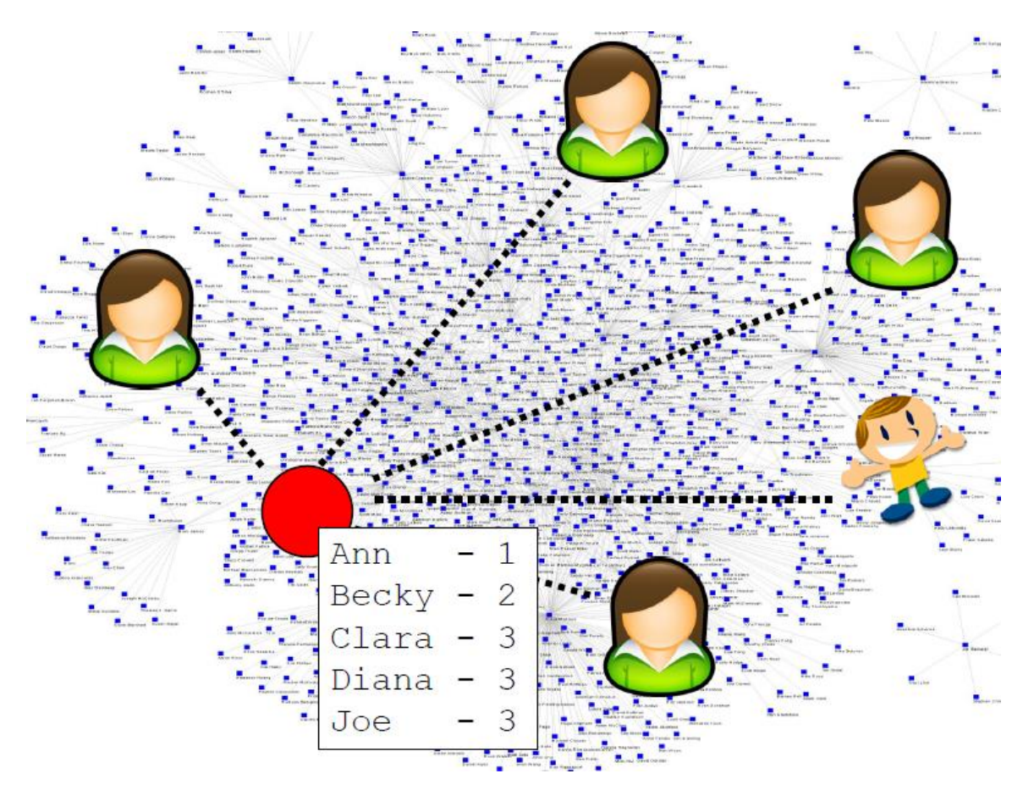
Is the President a friend of a friend of yours? Or perhaps a friend of a friend of a friend? How many social links, do you think, separate you from the author of this poster? Have you ever played the *Kevin Bacon game*? Do you know the *Erdős number* of Albert Einstein? Games aside, computing distances between nodes of social networks is a rather common problem both for sociological research as well as for various practical applications.



The problem of finding distances in graphs is well-studied. Unfortunately, all *exact* algorithms for solving it are a bit too slow to be useful in certain. Indeed, the naive shortest path search algorithm will have to traverse nearly the whole graph in order to answer even a single distance query. For a modern billion-user social network, this means performing billions of operations — way too much to spend on a single search request.

Landmark-based methods

Computation time may be significantly reduced if we resort to certain *approximate* algorithms. A popular family of such algorithms are *landmark-based methods*. The idea is simple: pick several (say, 100) random "landmark nodes" in the graph, then precompute and store shortest path distances from each landmark node to all other nodes.



Now suppose that we want to quickly find the approximate distance between "Joe" and "Mary". For that we simply check the lengths of all possible paths *through the landmark nodes*, i.e.

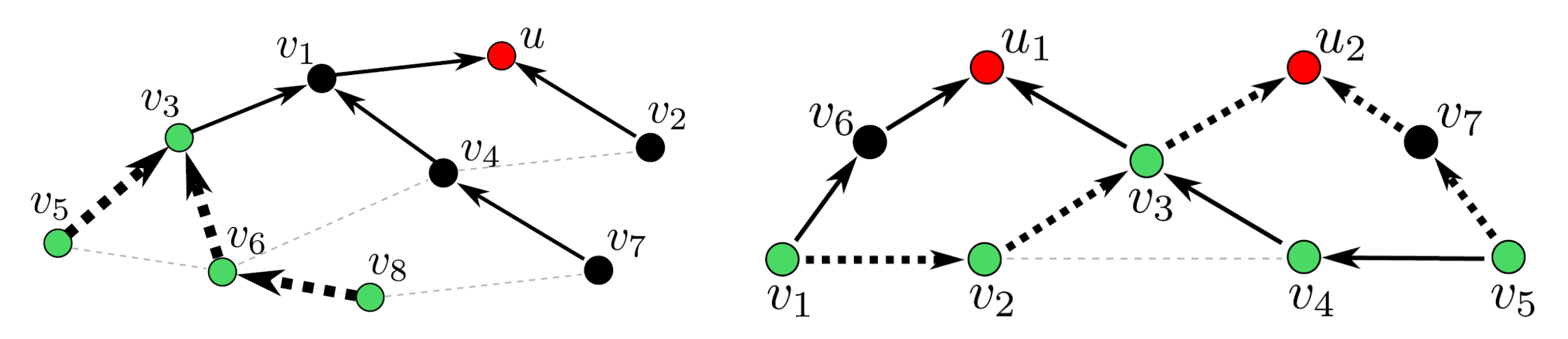
[Joe → Landmark1 → Mary],

[Joe → Landmark2 → Mary], etc,

and choose the smallest one of them. Recall that all the distances *[Joe → Landmark#]* and *[Landmark# → Mary]* have been precomputed and stored. We'll only need to retrieve and add them up, so the whole computation will take around a couple of hundred operations — orders of magnitude less than a billion.

Shortest path trees

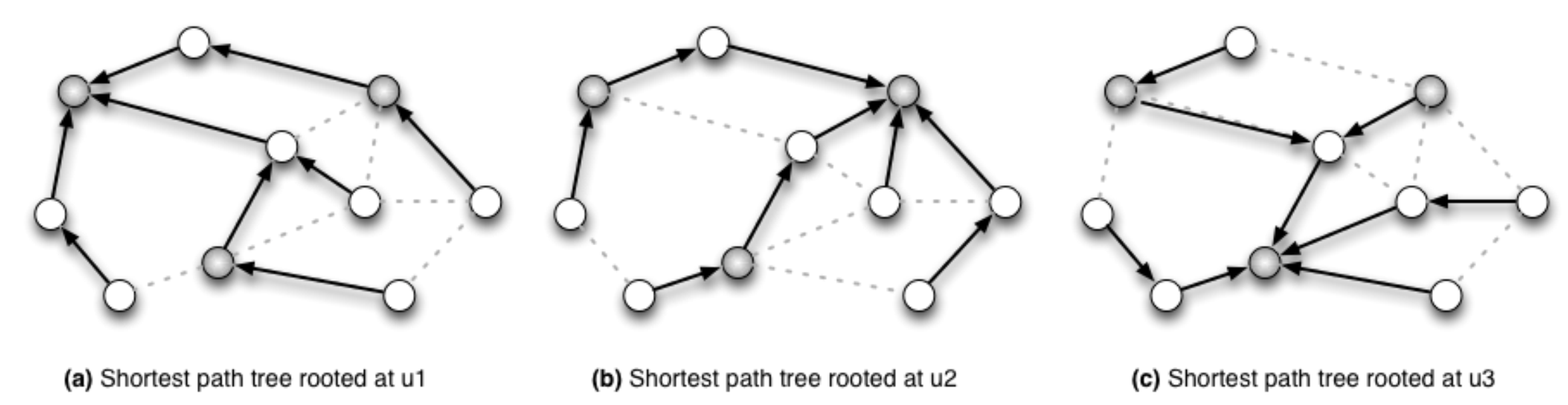
There are many variations of the landmark-based methods. Importantly, computing a complete *shortest path tree* (rather than just the distances) makes it possible to find much better distance approximations by locating *lowest common ancestors* of the query nodes or combining paths from several landmarks.



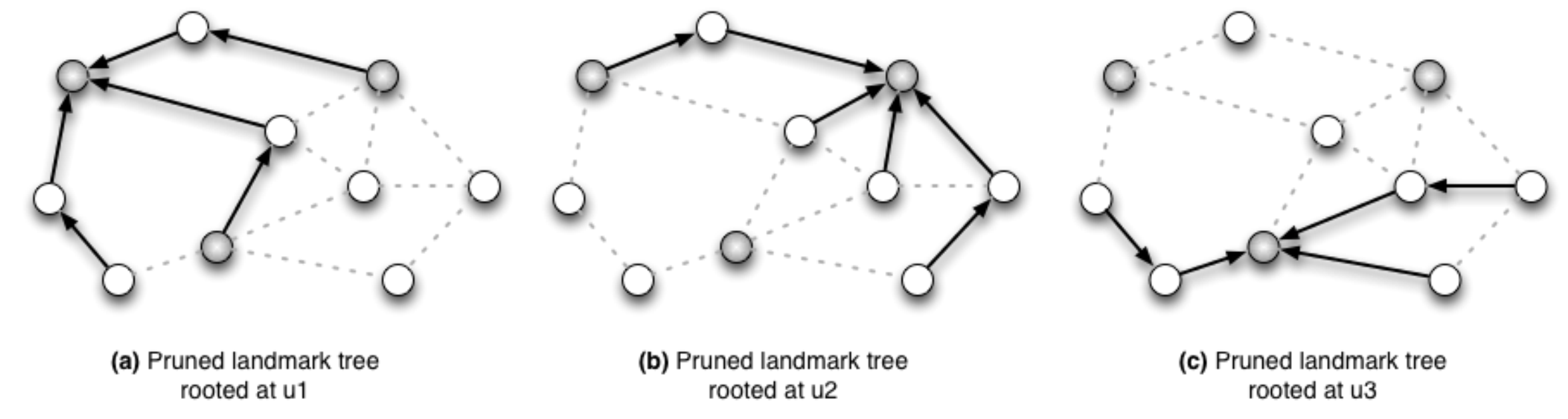
Pruned landmark trees

The problem with the landmark-based approach is that it may require quite a lot of memory to store the precomputed distances or shortest path trees. For a graph with a billion nodes, storing a shortest path tree (or a set of distances) for a single landmark requires no less than a gigabyte of disk space or RAM. With hundreds of landmarks the space consumption increases proportionally to hundreds of gigabytes.

To overcome this problem we propose to limit the set of nodes, spanned ("covered") by each landmark so that every node is present in exactly *k* shortest path trees (where *k* is considerably smaller than the number of landmarks). As a result, we need to store much smaller trees for each landmark. It turns out such *pruned landmark trees* can be computed efficiently using a rather straightforward algorithm.


(a) Shortest path tree rooted at u1
(b) Shortest path tree rooted at u2
(c) Shortest path tree rooted at u3

Complete shortest path trees for 3 landmarks in a graph


(a) Pruned landmark tree rooted at u1
(b) Pruned landmark tree rooted at u2
(c) Pruned landmark tree rooted at u3

Pruned shortest path trees for the same landmarks

Given a set of pruned landmark trees, the distance between any nodes *s* and *t* can be estimated by considering all possible combinations of the form

$$\mathrm{dist}(s,u) + \mathrm{dist}(u,v) + \mathrm{dist}(v,t)$$

where *u* and *v* are two landmark nodes that cover *s* and *t* respectively in their trees. Note that in order to quickly evaluate $\mathrm{dist}(u,v)$, the distances between all pairs of landmarks need to be precomputed in advance together with the landmark trees.

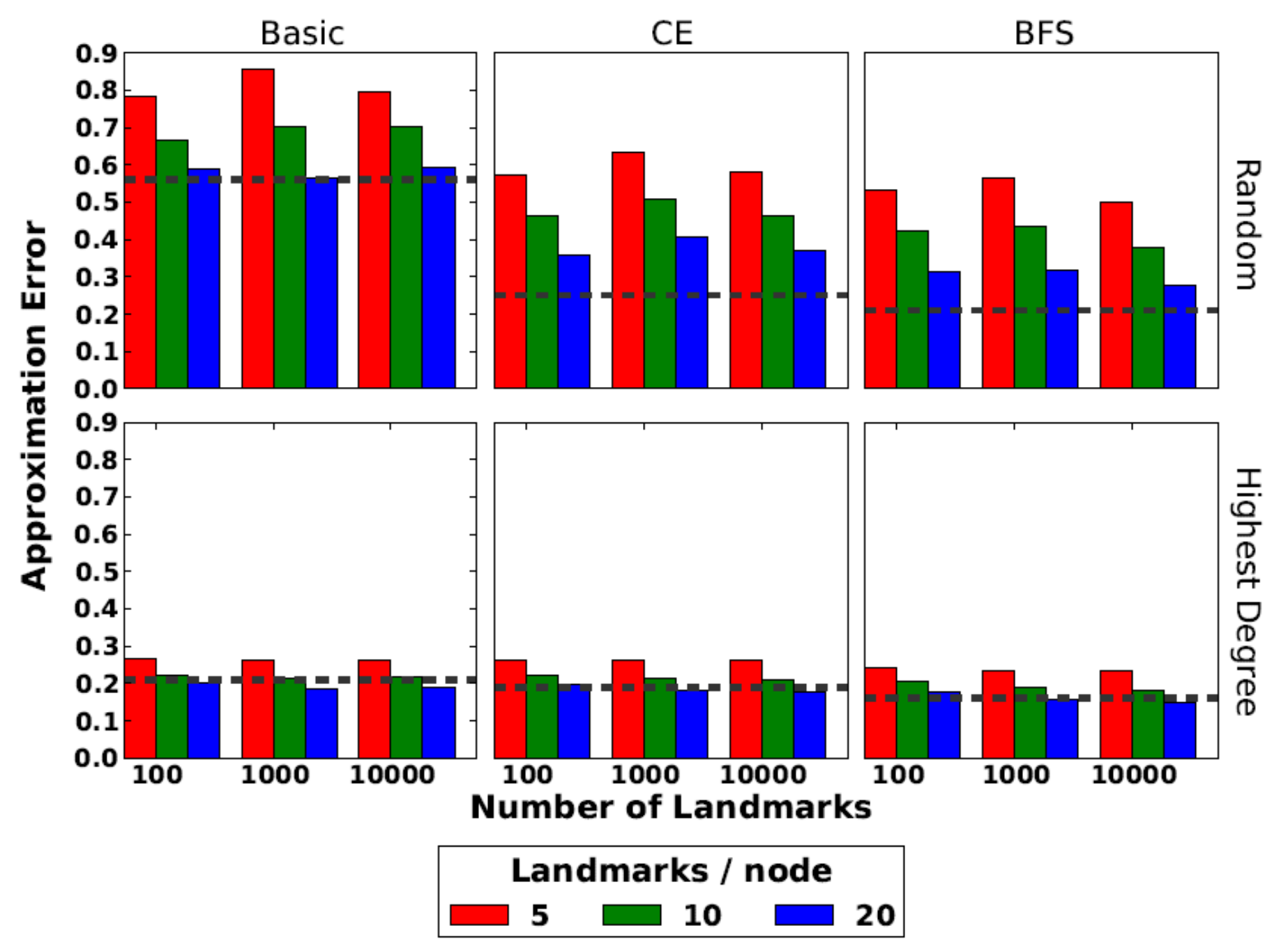
Results

We evaluated the use of pruned landmarks trees within three different landmark-based estimation techniques using 100, 1000 and 10000 landmarks on four social networks, spanning several magnitudes in terms of size.

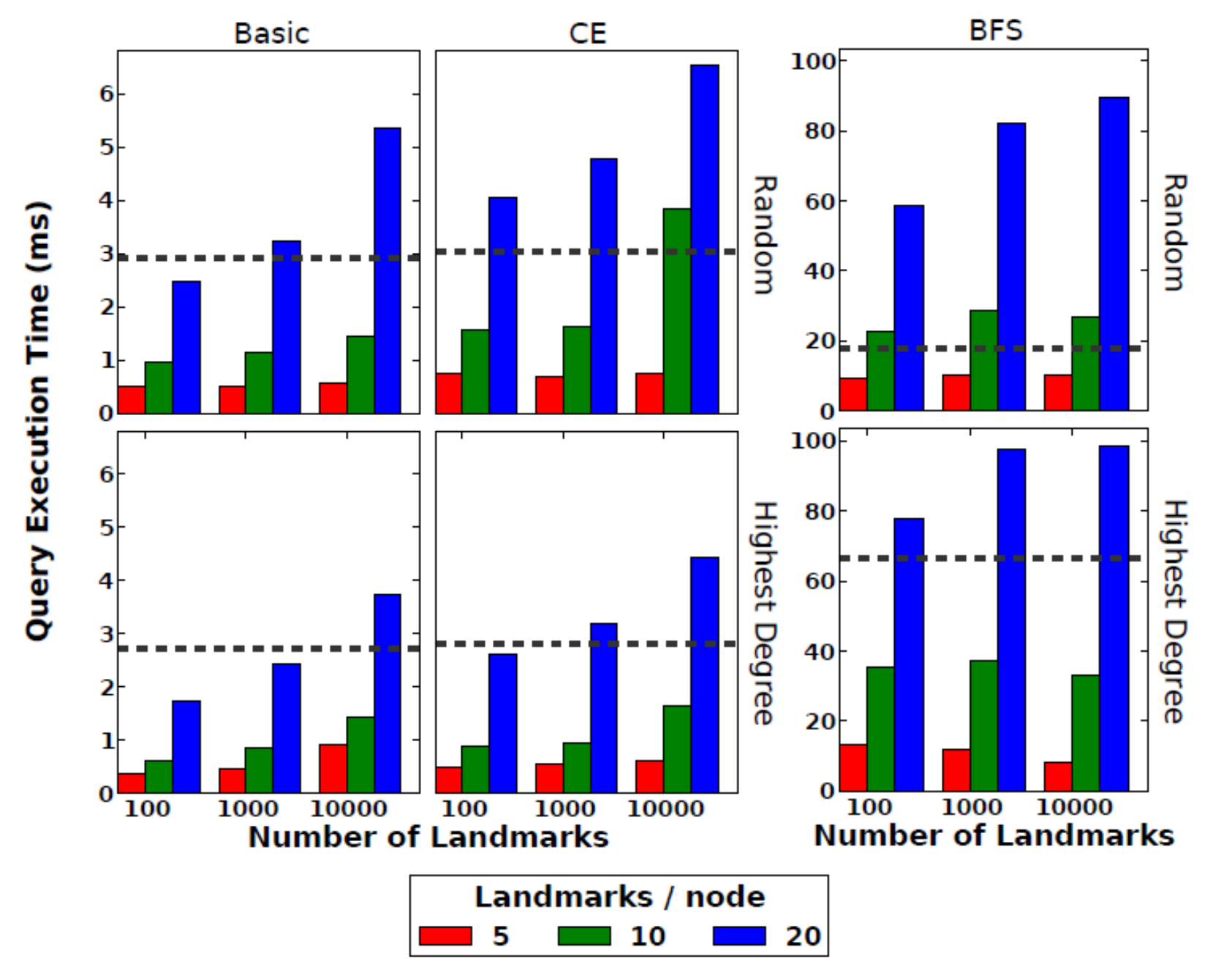
Dataset	DBLP	Orkut	Twitter	Skype
$\lvert V\rvert$	770K	3.1M	41.7M	454M
$\lvert E\rvert$	2.6M	117M	1.2B	3.1B
$\bar{d}$	6.25	5.70	4.17	6.7
Δ	23	10	24	60
$\lvert S\rvert/\lvert V\rvert$	85%	100%	100%	85%
t_{BFS}	343 ms	25.4 sec	11 min	33 min

We compared the results to the baseline approach (with full landmark trees) and discovered the following:

- The approximation quality with pruned landmark trees is similar to the baseline approach.
- The query execution time is also similar.
- The amount of memory required to store the landmark data structures is, however**, reduced up to 10 times.**



Approximation error with pruned landmarks on the Skype graph. The dashed line denotes the baseline (no pruning). The upper panels correspond to the case when landmarks are selected randomly. The lower panels correspond to the case when nodes with the highest degree are used as landmarks.



Query execution time on the Skype graph. Dashed line denotes the baseline (no pruning).

Graph	Landmarks	Landmarks / Node 5	Landmarks / Node 10	Landmarks / Node 20	Baseline (100 SPTs)
DBLP	100	30M	59M	117M	300M
	1000	34M	63M	121M	
	10000	411M	441M	499M	
Orkut	100	118M	235M	469M	1.2G
	1000	122M	239M	473M	
	10000	499M	616M	851M	
Twitter	100	1.6G	3.2G	6.3G	16G
	1000	1.6G	3.2G	6.3G	
	10000	2.0G	3.5G	6.6G	
Skype	100	17G	34G	68G	170G
	1000	17G	34G	68G	
	10000	18G	35G	69G	

Table 3: Total PLT index memory usage

Memory, consumed by the indexing data structures with and without pruning.

Acknowledgments

The authors acknowledge the feedback from Ando Saabas from Skype/Microsoft Labs. This research is funded by ERDF via the Estonian Competence Center Programme and Microsoft/Skype Labs.

STACC Software Technology and Applications Competence Center

European Union Regional Development Fund Investing in your future