MIPRO 2013, May 20-24, 2013, Opatija, Croatia

# Navigational Evaluation of Breadth First Search Spanning Trees

Denis Helic*, Markus Strohmaier*†,Weronika Wójcik*
* Knowledge Management Institute
Graz University of Technology
Graz, Austria
Email: {dhelic,markus.strohmaier}@tugraz.at, weronika.wojcik@student.tugraz.at
† Know-Center, Graz University of Technology, Graz, Austria

***Abstract*—Decentralized navigation is one of the most important functions of complex networks. A wide variety of different networks such as communication, social, or information networks possesses certain structural clues which allow navigational agents to efficiently navigate those networks even with local knowledge of the network only. Such structural clues include node degrees and their centralities, similarities between nodes, or node clustering coefficients. In practice, those properties may be combined and abstracted in the form of a distance metric on the network node set – a typical representation of such a distance metric is a hierarchy of network nodes where the most central nodes are situated in the upper levels of the hierarchy and the hierarchy links capture the node similarities or their clustering. Recently, a number of algorithms for extracting such hierarchies have been introduced. The majority of those algorithms is based on complex and computationally intensive methods such as hierarchical clustering. In this paper we analyze several simple spanning tree algorithms and their ability to extract sound hierarchies for network navigation. In particular, we are interested in correlation of structural node properties (such as node degree) and the navigational quality of spanning trees that are rooted at those nodes. Our work sheds light on the ability of spanning trees to serve as a distance metric for decentralized navigation in networks. Our results are relevant for scientists interested in the navigability of complex networks and for engineers who are interested in fast and simple extraction of hierarchies for supporting navigation in networks.**

## I. Introduction

Decentralized navigation in social or information networks represents a task where an agent navigates a network with limited knowledge. The agent's goal is to navigate from a starting node $s$ to another target node $t$ without having detailed knowledge about the full network topology. Informally, we say an agent uses *local* knowledge for navigation when the agent has knowledge about the nodes within the immediate one-hop neighborhood of the current location only, but no detailed knowledge of the network beyond that. We refer to these nodes as *candidate* nodes, because they are candidates for navigation. We say an agent uses *background* knowledge for navigation when the agent has some intuitions about the candidate nodes - for example which candidate nodes might bring her closer to a target. In previous research, Kleinberg [1]–[3] and others have introduced a simulation-based navigation framework called *decentralized search* that is formally capable of capturing these and other aspects of navigation.

In this paper, we turn our attention to spanning trees as one particular technique for constructing background knowledge. We define the problem formally, and investigate the usefulness of breadth first search constructed spanning trees for navigation. We prove a few navigational properties of spanning trees theoretically, and expand our analysis by conducting experiments. We conclude that breadth first search spanning trees represent an interesting new option for constructing efficient trees for navigation that can be used as background knowledge in decentralized navigation. However, relation between node properties and the efficiency of their corresponding breadth first search spanning trees is not immediately visible requiring further research that should analyze further algorithms for construction of spanning trees.

## II. Related Work

Research on decentralized search in social networks started with Milgram's seminal small world experiment [4] in which he aimed to study the connectedness of the US society. In his experiments he found that people are able to navigate large social networks (such as the population of the USA) efficiently. Specifically, he found that the average number of links between people in his experiment is around 6, hence a "small world". Later, Kleinberg analyzed an implicit result of the Milgram's experiment, the ability of humans to *find a short path* when there is such a path between two nodes [1]–[3]. Kleinberg concluded that social networks possess certain latent properties that humans are aware of. This *background knowledge* of network structure allows humans to find a short path between two arbitrary network nodes efficiently. Kleinberg defined an "efficiently" navigable network as a network for which a *decentralized search algorithm* exists, such that its delivery time (the number of nodes that the algorithm needs to visit before it reaches the destination node) is polynomial in $log(n)$, where $n$ is the number of nodes in the network.

Subsequent work has investigated the nature of background knowledge that is required for efficient decentralized search algorithms. In other words: What structural properties do efficiently navigable networks possess? To that end, Kleinberg designed a number of network models such as the 2D-grid model [1], hierarchical model [3], and group model [3]. Inde-

pendently, Watts [5] introduced the notion of social identity as a membership in a number of social groups organized in hierarchies and showed the existence of efficient decentralized search algorithms by simulation.

Decentralized search has been used to model navigation in social and information networks. For example, Adamic and Adar [6] have used organizational hierarchies as background knowledge to model navigation in e-mail networks. In our own previous work, we have used decentralized search to evaluate the usefulness of topical hierarchies for navigating information networks such as social tagging systems [7], [8]. Decentralized search has also been used to model navigational trails left by users on Wikipedia [9]. However, little is known about the nature and impact of background knowledge on decentralized search. For example, we don't know how different kinds of background knowledge influence the ability of a decentralized search model to efficiently navigate networks with local information. This issue is what we want to explore in this work. In particular, we turn our attention to the efficiency of spanning trees as background knowledge for decentralized search.

## III. Theoretical Framework

### A. Preliminaries

Let us first define some basic terms that we use throughout the paper. $G(V, E)$ is a network with nodes from a non-empty set $V$ and links from set $E$. We denote the number of nodes in a network with $n$ and the number of links with $m$. In this paper we focus on the analysis of undirected networks – the analysis of the directed networks is more complicated and we leave it for the future work. A *path* in a network is a sequence of different nodes that are connected by links – it is a *route* through the network. The path length is the number of links in that path. A *shortest path* between nodes $u$ and $v$ is a path between $u$ and $v$ with the minimal length. We define distance $d(u, v)$ between $u$ and $v$ as the length of the shortest path connecting $u$ and $v$. We call a network connected if there is a path from every node to every other node in the network. A *cycle* is a path that starts and ends at the same node. A *tree* is a connected network that has no cycles. It is easy to show that a tree with $n$ nodes has exactly $n - 1$ links. A network can also consist of a number of disconnected parts, all of which contain no cycles. In that case the individual parts are all trees. We call then the complete network a *forest*.

A *spanning tree* of a connected graph $G$ is a tree that spans every node from the network. A special type of a spanning tree is a *shortest path spanning tree rooted at node* $r$, which is a spanning tree such that for any node $u$, the distance between $r$ and $u$ is the same as in the network.

In this paper we will denote the distance between $u$ and $v$ in the network with $d^G(u, v)$ and their distance in a forest with $d^F(u, v)$.

### B. Global vs. local knowledge

Finding a shortest path between any two nodes in a network is easy if we know about all the nodes and links; that is if we have *global knowledge of the network*. Asymptotically, global knowledge is of size $O(n^2)$, since we might have a link between any two nodes to denote distances in the network. In practice, we often do not possess such global knowledge of the network, but *local knowledge of the network* only. For instance, in a social network we typically know about our one hop neighborhood; that is we know about our friends, but not about the friends of their friends, and so on. Analogously, in an information network such as Wikipedia, we can see only the links emanating from the page that we are currently viewing. That means that in practice, our knowledge about the network structure is much smaller than $O(n^2)$ – we only have intuitions about parts of the network.

Yet, people are still able to find shortest paths in social networks even with such limited knowledge. For example, in the famous "small world" experiment by Milgram [4], randomly selected persons from Nebraska and Massachusetts were required to pass a letter to a target person in Boston, Massachusetts in a decentralized manner through their social contacts, thereby requiring a group of people to search in a very large social network by utilizing local knowledge only. One of the most important results of the experiment was the finding that people are able to efficiently find short chains from the network in a *decentralized* manner. For example, Kleinberg concluded that humans possess *background knowledge* of the network structure and that this background knowledge allows humans to efficiently find short paths [1]–[3]. Kleinberg represented such background knowledge as a hierarchy (tree) of nodes, where more similar nodes are situated closer to each other in the tree. Thus, people consult the tree and base their navigation decisions on this background knowledge, which is encoded in a tree.

It is important to note that the size of background knowledge (represented as a tree) is considerably smaller than the size of global knowledge. Specifically, there are $n-1$ links in the tree, which means asymptotically the size of knowledge encoded in the tree is $O(n)$ as compared to the $O(n^2)$ links encoded in global knowledge of the network.

Based on this difference between the size of background knowledge vs. global knowledge, an interesting research question to ask is the extent to which different kinds of background knowledge exhibit different kinds of navigational properties. In other words, how do different trees or forests differ with regard to their ability to encode the knowledge about a network for navigational purposes? To answer this question, we next formalize some of the terms that are needed for analysis.

### C. Navigation

*Definition 1:* **Navigation (routing)**. Let $G(V_G, E_G)$ be a connected undirected network, and let $F(V_F, E_F)$ be a forest such that $V_F = V_G$, and $|E_F| \leq n - 1$. Let $s$ be a start node and $t$ a target node. A navigation path between $s$ and $t$ is defined as a path $(s, u_1, u_2, ..., u_{k-2}, u_{k-1}, t)$ from $G$ such that $d^F(u_i, t) > d^F(u_j, t)$ for $i < j$.

With that, we can say that in a navigation path, the subsequent node in a path has a smaller *forest distance* to the target node than its predecessor. We call a network $G$ *navigationally*

*connected* with respect to a forest $F$ (that is the forest $F$ provides the distances for the navigational paths in $G$) *if and only if* there exists a navigation path from every node to every other node in the network. This allows us to connect the structural description of a network and a forest in the form of navigational paths with the dynamics of an agent navigating the network. Thus, at each navigation step towards a target node, an agent consults the forest about the distances of the candidate nodes to the target node and selects a node with a smaller distance to the target node than that of the current node. We denote the current node with $u$, and the set of its neighbors; that is the set of the candidate nodes with $\Gamma(u)$.

Next, we demonstrate a number of ways in which forests relate to their corresponding networks from a navigational perspective.

*Theorem 1:* Let $G$ be a connected undirected network, and let $F$ be a forest such that $V_F = V_G$, and $|E_F| \leq n-1$. Then:

(a) If $F$ is disconnected then $G$ is navigationally disconnected with respect to $F$.
(b) If $F$ is connected and $E_F \backslash E_G \neq \emptyset$ then it is possible that $G$ is navigationally disconnected with respect to $F$.
(c) If $F$ is connected, and $F$ is a spanning tree of $G$; that is $E_F \subseteq E_G$ then $G$ is navigationally connected with respect to $F$.

*Proof:*

(a) Since $F$ is disconnected, there are at least two trees in $F$. Let us denote these two trees with $T_1$ and $T_2$, respectively. Let $t_1 \in T_1$ and $t_2 \in T_2$. Then by definition of the length of the shortest path $d^F(t_1, t_2) = \infty$, and thus we can not create a navigation path between those two nodes with decreasing distances.
(b) We need to construct one configuration without a navigation path between two nodes $u$ and $v$. Let $t$ be a node with a single link $(v, t)$ in $G$ and a single link $(u, t)$ in $F$. In that case $(u, t) \in E_F \backslash E_G$ with $d^F(u, t) = 1$. There is no link between $v$ and $t$ in the forest, and thus $d^F(v, t) \geq 2$. Now suppose also that there is a link $(u, v)$ between $u$ and $v$ in $G$. A path from $u$ to $t$ in the network must go through $v$ and so we have $(u, v, t)$ with the forest distances $(1, \geq 2, 0)$ (see also Figure 1).
(c) Since $F$ is a spanning tree of $G$, we have a path in $F$ from every node $s$ to every other node $t$ that is also a path in the network. Each subsequent node from this path has a smaller distance to target node $t$, and the forest distance is decreased by 1. Thus, $d^F(u_i, t) > d^F(u_j, t)$ holds for every $i < j$ and $G$ is navigationally connected with respect to $F$ (see also Figure 2).

■

The concept of navigational connectivity is closely related to the success rate of an agent navigating a network with the forest as the background knowledge. The success rate captures the extent to which an agent is successful in finding paths between two nodes with limited (local & background) knowledge only.

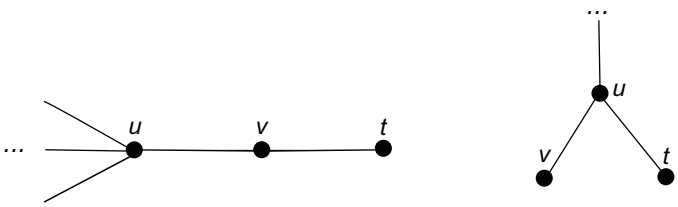
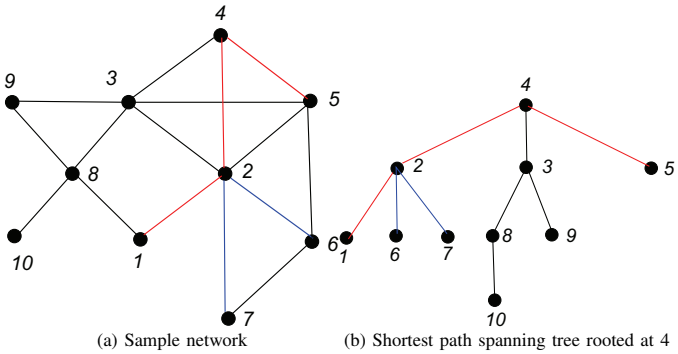



Fig. 1. A particular configuration of $G$ (left) and $F$ (right), which is navigationally disconnected. There exist links in $F$ that do not exist in $G$ – the link between $u$ and $t$. The path from $u$ to $t$ passes through $v$ – the distances in $F$ are not monotonously decreasing: $(1, 2, 0)$.




(a) Sample network (b) Shortest path spanning tree rooted at 4

Fig. 2. Paths from 1 to 5 (red) and 6 and 7 (blue) exist both in the network (left) and in the spanning tree (right). The distance $d^F(u, v)$ along the paths is monotonously decreasing. Thus, the network is navigationally connected with respect to the spanning tree.

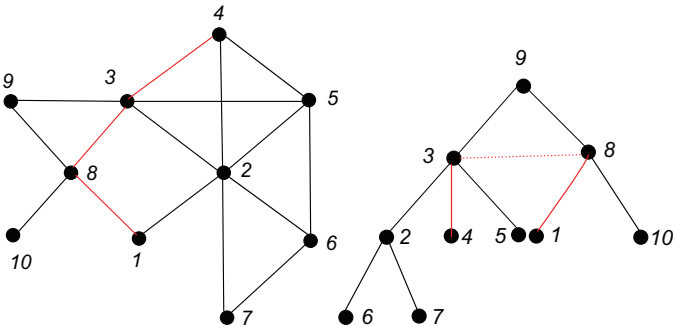
Thus, if a network $G$ is navigationally connected with respect to the forest $F$, then the navigation agent will be able to find any target node starting from any other start node. However, if the network is not navigationally connected then for certain configurations of the network $G$ and forest $F$ the success rate will drop as for some combinations of start and target node the navigational path does not exist. This might happen in cases where $F$ contains links that do not exist in $G$ – those links might be exogenous to the network – for instance, two nodes are similar in some external properties but they do not share a link in the network.

### D. Greedy navigation

Next, we are interested in the efficiency of the navigational agent, or in other words we are interested in the length of the navigational paths.

*Definition 2:* **Greedy navigation**. Let $G(V_G, E_G)$ be a connected undirected network, and let $F(V_F, E_F)$ be a forest such that $V_F = V_G$, and $|E_F| \leq n-1$. Let $s$ be a start node and $t$ a target node. A greedy navigation path is a navigation path between $s$ and $t$ defined as a path $(s, u_1, u_2, ..., u_{k-2}, u_{k-1}, t)$ from $G$ such that $u_{i+1} \in M := \text{argmin}_{u_j \in \Gamma(u_i)} d^F(u_j, t)$ is the argument of the minimum of the distance function of candidate nodes at node $u_i$.

Thus, a greedy navigation path is a path where each node in the path has the smallest distance to the target node among all candidate nodes at that position. Dynamically, for each node $u$ a navigation agent selects an adjacent node $v$ which has




(a) Sample network (b) Shortest path spanning tree rooted at 9

Fig. 3. Path from 4 to 1 (red) is a greedy navigational path. There is a leap in the distance reduction along the path (the dashed link from 3 to 8) – this link exists in the network but not in the tree – this is an example of the greedy navigator taking a shortcut in the network. The greedy navigational path $(4, 3, 8, 1)$ is longer than the shortest path in the network $(4, 2, 1)$.

the smallest tree distance to the target node $t$. In cases where $|M| > 1$ we have two or more candidate nodes that all have the minimal distance to the target node we rely on a total order defined over the node set $V$. For example, we might count nodes and use $<$ relation over the integer representation of the nodes to select a node with the minimal encoding.

We denote the length of a greedy navigation path with $h(s,t)$ – it is the number of links in a greedy navigator path between $s$ and $t$. Note that depending on the particular structure of the forest, a greedy navigational path might take shortcuts (larger leaps than 1) in decreasing the distance in a forest. On the other hand, a greedy navigation path is not necessarily a shortest path in $G$, as can be seen in Figure 3.

To measure the navigational efficiency of a forest $F$, we can compare $h(s,t)$ to $d^G(s,t)$ for every $s$ and every $t$. We introduce stretch $\tau(s,t)$ as the ratio between $h(s,t)$ to $d^G(s,t)$:

$$\tau(s,t) = \frac{h(s,t)}{d^G(s,t)}, s \neq t. \tag{1}$$

Stretch $\tau(s,t)$ is equal to one if the greedy navigation path is equal to the length of the shortest path between $s$ and $t$ in $G$. Otherwise if the greedy navigation path is longer than the shortest path, we obtain stretch values larger than one. For example, a stretch value of 2 and $d^G(s,t) = 2$ means that the greedy navigation path between $s$ and $t$ is twice as long as the global shortest path – it is 4. To assess the navigational efficiency of a spanning tree, we will analyze the stretch in more details.

We also introduce the global stretch $\tau$. For a navigationally connected network $G$ with respect to a spanning tree $F$, we calculate the global stretch:

$$\tau = \frac{1}{n(n-1)} \sum_{s \neq t} \frac{h(s,t)}{d^G(s,t)}. \tag{2}$$

### *E. Navigation efficiency*

Next, we turn our attention to the analysis of the navigational properties of different spanning trees. In this paper we focus on shortest path spanning trees, which can be obtained, for example, by performing breadth first search. Such spanning trees are rooted at the node from which we start breadth first search and span a tree with distances from the root node to every other node being the same as its distances to every other node in the network. It is interesting to investigate the relation between node properties in the network and the navigational efficiency of spanning trees rooted at those nodes. This would allow us to better identify spanning trees with useful navigational properties.

First, we construct a shortest path spanning tree rooted at a node $v$. Obviously, the efficiency of greedy navigation from any starting node $s$ towards $v$ is maximal, since the tree contains shortest paths to $v$ from every other node. In other words, the stretch $\tau(s,v)$ for all $s$ is equal to 1.

However, we are interested in exploring what happens if we navigate towards arbitrary nodes, not only towards $v$. Can we measure stretch induced by a shortest path spanning tree rooted at $v$ over all search pairs by taking arbitrary $s$ and $t$ as the starting and target node respectively?

Let us start our analysis with a conservative estimation (upper bound) of the stretch of a given shortest path spanning tree rooted at $v$. For a starting node $s$ and a target node $t$ we can always take a path towards $v$ first and then after we reach $v$ we can navigate from $v$ towards $t$. The total number of steps is:

$$d^G(s,v) + d^G(v,t). \tag{3}$$

If we iterate over all starting nodes fixing $t$ as the target node, the total distance that the navigator travels is given by:

$$\sum_s (d^G(s,v) + d^G(v,t)) = \sum_s d^G(s,v) + n d^G(v,t). \tag{4}$$

If we now iterate over all target nodes $t$ we obtain:

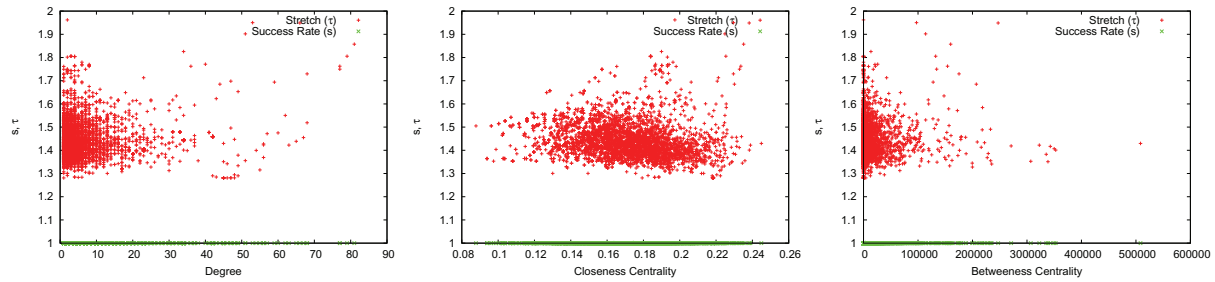
$$\sum_t (\sum_s d^G(s,v) + n d^G(v,t)) = 2n \sum_s d^G(s,v). \tag{5}$$

The estimation is proportional to the sum of shortest paths of node $v$. For nodes which are on average "closer" to all other nodes in the network this sum will be smaller. In fact, the sum is reciprocal of the node closeness centrality – for nodes with high closeness centrality we can expect the sum of the length of navigational paths in the network to be small.

Following the same argument we can also calculate the estimation (upper bound) of the global stretch of a shortest path spanning tree rooted at $v$:

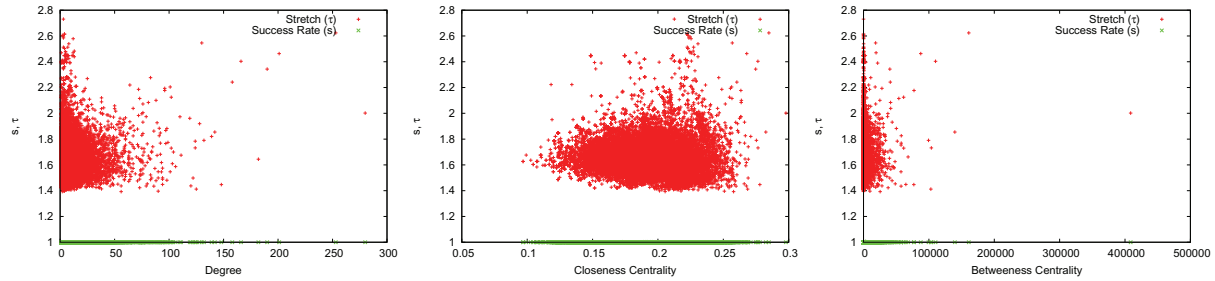
$$\tau = \frac{1}{n(n-1)} \sum_{s \neq t} \frac{d^G(s,v) + d^G(v,t)}{d^G(s,t)}. \tag{6}$$

The global stretch will become smaller for node $v$ which lies frequently on a shortest path between other nodes in the network. In that case we have: $d^G(s,v) + d^G(v,t) = d^G(s,t)$ and therefore $\tau(s,t) = 1$. Therefore, we can expect that nodes with high betweeness centrality produce spanning trees that are more efficient than the nodes with lower betweeness centrality.




(a) Degree (b) Closeness centrality (c) Betweeness centrality

Fig. 4. Success rate (green) and stretch (red) for each shortest path spanning tree of the GR-QC network. Our searches are always producing a successful result with the stretch between 1.2 and 2, however it is impossible to notice, whether is it more advantageous, to choose shortest path spanning tree starting at the root with a greater or with a lower centrality property.




(a) Degree (b) Closeness centrality (c) Betweeness centrality

Fig. 5. Success rate (green) and stretch (red) for each shortest path spanning tree of the GR-CondMat network. Again, we observe maximal success rates (as expected from the theoretical analysis) with the stretch between 1.4 and 2.8. Consistent with the experiments with the GR-QC dataset we can not deduce a simple relation between stretch and node centrality properties.

## IV. Experiments

We perform our experiments on two datasets: GR-QC (General Relativity and Quantum Cosmology)[1] and GR-CondMat (Condense Matter Physics)[2] collaboration network. The first dataset covers scientific collaborations between authors of papers submitted to General Relativity and Quantum Cosmology category, whereas the second dataset includes the collaborations between authors publishing in the field of Condense Matter. If an author $i$ co-authored a paper with author $j$, the graph contains an undirected edge from $i$ to $j$. If the paper is co-authored by $k$ authors, this generates a completely connected (sub)graph on $k$ nodes. The data covers papers in the period from January 1993 to April 2003.

For our purposes, we choose the largest connected component of the aforementioned networks containing 4158 nodes and 26850 links (GR-QC dataset), and 21363 nodes and 182628 links (GR-CondMat dataset). From the chosen component, we create shortest path spanning trees using breadth first search, where each spanning tree has the root node in a different node of the component. For evaluation, we randomly select 1000 pairs of nodes and simulate navigation using decentralized search. Each of those pairs represents a start node and a target node for a greedy navigation path.

For every tree, we perform greedy navigation on the selected pairs. Search is considered successful *iff* a navigation agent finds a navigation path between the start and the target node. Moreover for each root, we calculate degree, closeness and betweenness centrality. This allows us to observe the extent to which stretch changes according to the aforementioned properties of a node. Figures 4 and 5 depict the relationship between stretch and the various properties of the root.

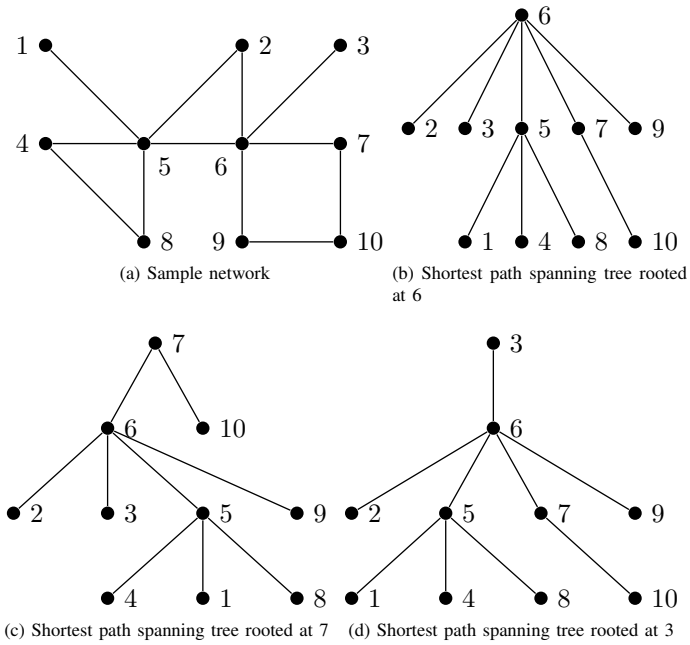
[1]http://snap.stanford.edu/data/ca-GrQc.html
[2]http://snap.stanford.edu/data/ca-CondMat.html

## V. Results and Discussion

Figures 4 and 5 show that for every shortest spanning tree, our navigation agent is able to find a navigation path between the start and the target node - as expected from our theoretical discussion of the problem. The stretch for both datasets is low: for GR-QC it is less than 2, and for GR-CondMat less than 2.8.

When we focus on just one property of the root, for example degree, we cannot deduce whether the stretch is going to be better for a greater or for a lower degree of the root of a shortest path spanning tree. A simple explanation for this somewhat unexpected behavior might be easily found through a closer investigation of the structure of the spanning trees constructed by breadth first search.

Breadth first search algorithm produces in many cases structurally similar or even identical shortest path spanning trees regardless of the node from which we start the search. For example, suppose that we start breadth first search at a so-called hub $i$; that is a node which has a lot of links to other nodes. Now, suppose that among the nodes that are linked to that hub $i$, we have a node $j$ with a single link $(i, j)$.




(a) Sample network

(b) Shortest path spanning tree rooted at 6

(c) Shortest path spanning tree rooted at 7

(d) Shortest path spanning tree rooted at 3

Fig. 6. A sample network and three breadth first search spanning trees rooted at nodes 6, 7, and 3. Although these nodes have different degrees their corresponding spanning trees are identical. Therefore, the trees have identical navigational properties, and we are not able to relate the navigational efficiency of the trees with the structural properties of nodes at which the trees are rooted.

Obviously, starting breadth first search at node $j$ produces an identical spanning tree to the spanning tree produced by breadth first search started at node $i$. In the first step of the algorithm we immediately reach node $i$ and proceed from there in the same way as previously. For an illustration of this situation please consult Figure 6.

The same phenomenon can be observed when concentrating on closeness or betweenness centrality. We expected that trees with higher betweenness centrality of the root should have lower stretch. However with breadth first search two shortest path spanning trees constructed from roots with different betweenness centralities, can be identical and therefore can have the same stretch.

The estimations of the upper bound on stretch are in the range of 1.4 to 4.0 for the GR-QC dataset. In our experiments, the estimations turn out very precise. The stretch measured by simulation is smaller than the conservative estimation in 99.8% of cases. The small amount of cases where stretch is greater than the estimated upper bound is due to a small sample size. We sample 1,000 search pairs out of 17,000,000 possible search pairs.

## VI. Conclusions

In this work, we analyzed spanning trees and their usefulness to act as background knowledge for informing decentralized search in networks. We found that the navigational quality of spanning trees rooted at certain nodes is mostly independent from the node's centrality in the network. On one hand, this can be explained by the fact that different roots can produce similar or identical spanning trees. On the other hand, we found - through inspection - that even if different spanning trees are produced, they do not necessarily differ significantly. For example, the majority of the structure of a spanning tree rooted at a high degree node can be similar to the structure of a spanning tree rooted at a low degree node. These results are interesting as they suggest that spanning trees rooted at arbitrary nodes in a network can be used efficiently as background knowledge for decentralized search. Our results are relevant for designers of user interfaces aiming to build hierarchical structures that aid users in navigating information networks (such as Wikipedia) or for organizational researchers interested in identifying organizational hierarchies that optimize navigability of social networks within organizations. For future research, we consider an expansion of our analysis to directed networks, or the investigation of other spanning tree algorithms (such as depth-first-search), as interesting routes to take. Another promising approach would be to construct spanning trees with a small, or even minimal diameter [10]. The diameter of a graph is the longest shortest path in a network, and as such it is a structural upper bound on the length of the navigational paths. One can expect a tree with a shorter diameter to be more navigationally efficient than a tree with a longer diameter.

## Acknowledgment

This research was in part funded by the FWF Austrian Science Fund research project "Navigability of Decentralized Information Networks" (P 24866-N15).

## References

[1] J. Kleinberg, "The small-world phenomenon: an algorithm perspective," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*, ser. STOC '00. New York, NY, USA: ACM, 2000, pp. 163–170.

[2] J. M. Kleinberg, "Navigation in a small world," *Nature*, vol. 406, no. 6798, p. 845, August 2000.

[3] J. Kleinberg, "Small-world phenomena and the dynamics of information," in *Advances in Neural Information Processing Systems (NIPS) 14*. Cambridge, MA, USA: MIT Press, 2001, p. 2001.

[4] S. Milgram, "The small world problem," *Psychology Today*, vol. 1, pp. 60–67, 1967.

[5] D. J. Watts, P. S. Dodds, and M. E. J. Newman, "Identity and search in social networks," *Science*, vol. 296, pp. 1302–1305, 2002.

[6] L. Adamic and E. Adar, "How to search a social network," *Social Networks*, vol. 27, no. 3, pp. 187 – 203, 2005.

[7] D. Helic, M. Strohmaier, C. Trattner, M. Muhr, and K. Lerman, "Pragmatic evaluation of folksonomies," in *Proceedings of the 20th international conference on World wide web*, ser. WWW '11. New York, NY, USA: ACM, 2011, pp. 417–426.

[8] M. Strohmaier, D. Helic, D. Benz, C. Körner, and R. Kern, "Evaluation of folksonomy induction algorithms," *ACM Trans. Intell. Syst. Technol.*, vol. 3, no. 4, pp. 74:1–74:22, Sep. 2012.

[9] C. Trattner, P. Singer, D. Helic, and M. Strohmaier, "Exploring the differences and similarities between hierarchical decentralized search and human navigation in information networks," in *Proceedings of the 12th International Conference on Knowledge Management and Knowledge Technologies*, ser. i-KNOW '12. New York, NY, USA: ACM, 2012, pp. 14:1–14:8.

[10] R. Hassin and A. Tamir, "On the minimum diameter spanning tree problem," *Information Processing Letters*, vol. 53, no. 2, pp. 109 – 111, 1995.