Analysis of the Indexing Gain When Searching in Spontaneous Wireless Networks

Michael Solomon Desta, Esa Hyytiä, Jörg Ott
Department of Communications and Networking
School of Electrical Engineering
Aalto University, Finland
Email: michael.desta@aalto.fi, {esa, jo}@comnet.tkk.fi

***Abstract*—Searching content in spontaneous wireless ad Hoc networks is a complex task due to the intermittent nature of the connections, energy constraints, the often present user mobility and most importantly the lack of centralized network administration. At the same time, the today's search engines (e.g., Google, Yahoo and Bing) are able to provide amazing results at the blink of an eye thanks to their sophisticated and extensive indexing of the content in the Web. Thereby, it is worthwhile to ask if searching content in (sometimes highly dynamic) spontaneous wireless networks could similarly benefit from indexing. In this paper, we provide an elementary analysis to this end. In particular, we compare the search performance without index to a system where full index is available (without taking into account the effort to build such an index). This allows us to characterize the premises under which indexing can be potentially useful, and vise versa. The concepts are illustrated with numerous examples and network topologies arising from different search strategies.**

I. Introduction

The number of smart phones and tablets keeps on increasing. Typically these devices are connected directly to the Internet via an access point or a base station. An alternative form of communication is a spontaneous wireless ad-hoc network (e.g., the opportunistic networking paradigm), where nodes establish direct links with their peers without any support from the infrastructure. This scenario can be extremely useful when the user for some reason cannot (cf. censorship) or does not want to (cf. high roaming costs) rely on the infrastructure [1]–[3]. Although the potential benefit of using such networks as an alternative to infrastructured networks is exciting, there are also a number of problems that arise due to the inherent lack of infrastructure, which would be easy to solve in the infrastructured world [4]. One such problem is *"How do we search for content in mobile opportunistic network?"*

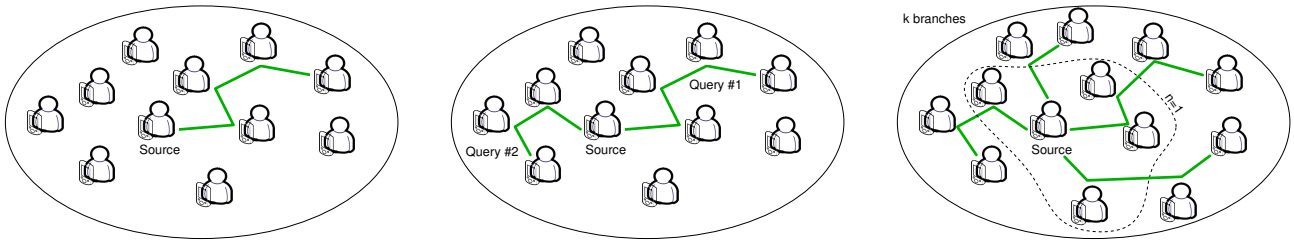
Searching for content in the infrastructured (server-client type) networks like the Internet is well-studied and understood, both in the academia and the industry. For example, today's search engines (e.g., Google, Yahoo and Bing) are able to provide amazing search results at the blink of an eye thanks to their sophisticated and extensive indexes of the content in the Web. Searching is also well studied in the less-structured networks such as Peer to Peer (P2P) networks [5], [6]. Unlike traditional server-client architecture, nodes in P2P networks act as peers with equal role which affects the network topology [6] (centralized, Hybrid or fully Distributed) and indexing scheme [7].

In *Centralized* P2P networks, there is a dedicated location/peer that acts as a server providing indexing function [7]. Peers can query this server to resolve the host address of the content they are searching whereas in *Hybrid* P2P networks, there is no centralized indexing server, rather many servers across the network that provide indexing function. The third one, *Fully Distributed* P2P network is where there is no separate indexing server. One prominent example of such network is the Internet overlay network *Gnutella* [8], [9].

With the complete lack of any sort of indexing server, a query for searching content in such P2P network is forwarded (based on some algorithm) until it is resolved by participating nodes. With the rapid interest and popularity P2P networks have attracted recently, there are quite many searching algorithms and query forwarding methods proposed; all with their pros and cons. Of the many solutions proposed, the two extremes that stand out are the traditional flooding (epidemic) search and direct delivery methods in which the latter one has the lowest search cost at the expense of success ratio. On the other end of the spectrum is the epidemic search algorithm with the highest cost by flooding the network, in exchange for a higher success ratio.

There are a number of solutions between these two extremes compromising between success ratio, response time and cost. Examples are limiting query forwarding (flooding) by TTL [10], hop-count or some for of its variants like [11], [12], Probabilistic Flooding [13], and Seeker Assisted Search (SAS) [14]. Most of the proposed solutions focus on resolving a search query only when it arises (e.g. see [15]), where as in [16], the authors have proposed a proactive system, local indices, where nodes proactively index contents in their locality for future use. In this paper, we consider the problem of searching content in spontaneous wireless ad-hoc networks, which is a non-trivial task due to the lack of infrastructure, the intermittent nature of the connections, energy constraints and the often present user mobility. We will not propose a new search algorithm, rather introduce a new dimension, *Mobile Opportunistic Indexing*, which can be used as an overlay solution to any of the search algorithms. Searching without an index can be seen as a *reactive* system, where a query for a given content is resolved only when the need arises. In contrast, search with index corresponds to a *proactive* system, where the nodes have in advance indexed the content in the network (either fully or partially) and also constantly update the index. A query can then be resolved with a minimal effort. Our aim is to gain understanding if searching content in such

ISBN 978-3-901882-80-7



(a) Linear network (b) Two linear networks (c) K linear networks

Fig. 1. Different logical topologies a query may follow.

networks could similarly benefit from indexing systems as in the Internet or P2P networks.

Our premise is a semi-static scenario where the network is dynamic in longer time scale, whereas it can be assumed to be static in short time scale, i.e. for the duration of the search process. We believe that such a scenario is reasonable as searching in otherwise highly dynamic networks does not make sense as the network is not stable (links break up very fast), where a node's direct neighbor could be far away in a brief moment.

To this end, we provide an elementary analysis where we compare the search performance without an index to a system where a full index is available. *Such comparison will allow us to see when indexing (along with the over head cost of building and maintaining one) is better than not indexing. Such analysis is a very complicated task that due to space limitations, we focus here only to the analysis of indexing gain without taking into account the effort to build such an index, and leave the cost consideration as a future work.*

These analytical considerations allow us to characterize the premises under which indexing can be potentially useful, and vise versa. The concepts are illustrated with numerous examples and network topologies arising from different search strategies. To the best of our knowledge, this is the first work that studies the possible gain from an indexing system in spontaneous wireless ad-hoc networks, which in particular quantifies the gain and optimal search parameter functions with different search topologies. The study provides us the maximum performance bound an indexing system can provide, and hence gain an understanding of how much the cost of building an index should be, if it makes sense at all.

The rest of the paper is organized as follows. First, in Section II we introduce the basic model and the notation. Section III gives the theoretical results for a search strategy that forms a *linear* topology. This is followed by the analysis of *two* and k- directional searches in Section IV and V, respectively.In Section VI, we consider a fully connected network where the search strategy reduces to a *star* topology. Section VII concludes the paper.

II. Preliminaries

We consider searching for a content in spontaneous wireless ad-hoc network. In particular, our aim is to evaluate the gain from *indexing* the content *a priori*. The index has the information about the (likely) location of the content, but not its actual value (for the sake of efficiency, we do not want to replicate all the content to every device, whereas the index is assumed to be concise and "cheap" to communicate). For simplicity, we assume that *a priori* every node has a fixed probability p of having the searched content independently of each other. We let the random variable $C(\alpha)$ denote the cost of the search with search strategy α. A search strategy may follow different topologies depending on the physical (e.g. a node's communication range) and/or logical structure of the network. A simple search strategy could be to check if any immediate neighbor has the content. Typically we measure the cost in terms of the number of transmissions (hops). For example, if a query first travels i hops, and then a positive response is returned along the reverse paths, we have $C{=}2i$. Furthermore let a successfully resolved query yield a fixed revenue of v, which characterizes the *expected value* of the searched information. Our aim is to maximize the expected *net profit*,

$$\max_{\alpha} \mathrm{P}\{S(\alpha)\} \cdot v - \mathrm{E}[C(\alpha)],$$

where $S(\alpha)$ denotes the event of successfully resolving a search with strategy α. At this point, it is also wise to differentiate between successfully resolving a search for a content and retrieving it. Here, we assume that if the search is successful, content will also be retrieved at the same time. In the sequel, we consider two types of search strategies: (i) search without indexing, α, where a query (or queries) travel hop by hop until the searched content is found or the query is terminated, and (ii) search with index, α_i, where we assume the ideal case that a global knowledge of the content locations is available for the searching node.

Next we will consider several physical and logical network topologies that arise from different search strategies: simple linear network (L), two directional linear network (2D), and generalize for k-directional topology (KD). The topologies are illustrated in Figure 1. In other words, the search strategy α determines the *topology* that the query (or queries) will follow hop by hop until the search ends.

III. Linear Network

Let us first consider search in a simple one directional linear network as depicted in Figure 1a, before analyzing more general topologies.

In this case, the basic linear search strategy without index, $\mathrm{L}(n)$, is defined by the maximum search depth n: a search is carried up to a distance of n nodes, and if the content is found at node $i \leq n$, then the search ends there and a reply is sent. This costs $C^L(n){=}2i$ transmissions and yields a net profit of $v-2i$. Otherwise, when the first n nodes do not have

the content, the search is terminated at the nth node and no answer is returned. As a result, the *net profit* is negative, $-n$.

With the ideal index, the situation is similar. The search strategy $\mathrm{L_i}(n)$ is defined by the maximum search depth n. However, a query is send only if the content is known to be within one of the first n nodes, and otherwise the search ends already at the source node.

A. Mean Net Profit

Without Indexing: First we assume that there is no index that the searching node could rely on. Therefore, for every query the node has to initiate a search and see if the content is found. As the probability of the next node having the searched content is independent of the earlier nodes, the probability that the searched content is first found at the distance of i hops is

$$p_i = (1-p)^{i-1}p \tag{1}$$

The *expected net profit* with the search strategy $\mathrm{L}(n)$ that, without indexing, checks at most first n nodes is thus

$$w^{\mathrm{L}}(n) = \sum_{i=1}^{n}(1-p)^{i-1}p(v-2i) - (1-p)^n n, \tag{2}$$

where the last term takes into account the event that the search may not be resolved after searching the first n nodes. After some manipulation, (2) reduces to

$$w^{\mathrm{L}}(n) = \frac{(2+(n-v)p)(1-p)^n - 2 + pv}{p}, \tag{3}$$

so that unbounded search gives

$$\lim_{n\to\infty} w^{\mathrm{L}}(n) = v - \frac{2}{p}.$$

With Indexing: Next we assume that content has been *proactively* indexed and the searching node knows for each query how far the content resides. The basic linear search strategy with *indexing*, denoted by $\mathrm{L}_i(n)$, in this case was defined by the maximum retrieval distance n. If the content is further than n hops away, the search is terminated immediately at the origin. Otherwise the content is retrieved from the nearest node. With our assumption on the distribution of the content, the *expected net profit* in this case becomes

$$w_{\mathrm{i}}^{\mathrm{L}}(n) = \sum_{i=1}^{n}(1-p)^{i-1}p(v-2i), \tag{4}$$

i.e., the same as (2) but the last term has been omitted due to the availability of the index. After some manipulation, we again obtain

$$w_{\mathrm{i}}^{\mathrm{L}}(n) = \frac{(2+(2n-v)p)(1-p)^n - 2 + pv}{p}, \tag{5}$$

and for the unbounded search we obtain the same limit,

$$\lim_{n\to\infty} w_{\mathrm{i}}^{\mathrm{L}}(n) = v - \frac{2}{p},$$

as one would expect when the information given by the index is effectively ignored when $n \to \infty$.

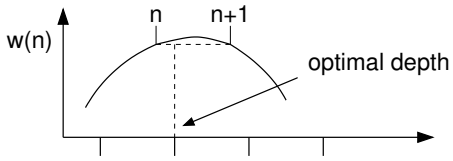


Fig. 2. Finding the optimal search depth n^*.

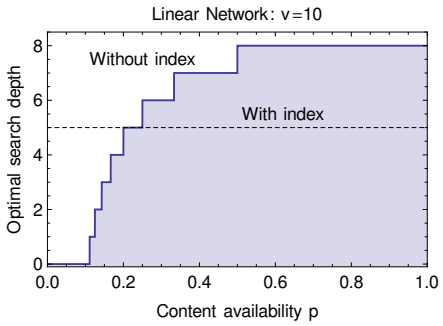


Fig. 3. Optimal search distance for $v = 10$ with and without index.

B. Optimum Search Depth

Let us next consider the optimal search depths denoted by $n_{\mathrm{opt}}^{\mathrm{L}}$. It follows from (2) and (4) that the expected net profit first increases as n increases, and then starts to decrease (or if v is too small, the expected net profit is negative for all values of $n > 0$). This is illustrated in Figure 2. Therefore, we can obtain the optimal search depth by first solving the following equation:

$$w(n+1) = w(n). \tag{6}$$

In our context, the feasible search depths are non-negative integer numbers, and therefore, referring to Figure 2, the optimal search depth is given by the integer when n is rounded up,

$$n_{\mathrm{opt}}^{\mathrm{L}} = \lceil n \rceil \tag{7}$$

Using (6), calculating the optimal search depth for the strategy $\mathrm{L}(n)$ yields[1],

$$n_{\mathrm{opt}}^{\mathrm{L}} = \begin{cases} 0, & v \le \frac{p+1}{p} \\ \left\lceil \frac{p(v-1)-1}{p} \right\rceil, & v > \frac{p+1}{p} \end{cases} \tag{8}$$

and for the search strategy $\mathrm{L}_i(n)$ (with index),

$$n_{\mathrm{opt,i}}^{\mathrm{L}} = \left\lfloor \frac{v}{2} \right\rfloor, \tag{9}$$

as any search going beyond the $\lfloor \frac{v}{2} \rfloor$ hops yields a negative net profit.

Note that when $p \to 1$, $n_{\mathrm{opt}}^{\mathrm{L}} \to v-2$, whereas $n_{\mathrm{opt,i}}^{\mathrm{L}} = \lfloor v/2 \rfloor$ (for all p), i.e., in the absence of index we (may) search further than with it. This is illustrated in Fig. 3 for $v = 10$.

[1]With some values of p and v, the solution to (6) is an integer number, in case of which both n and $n+1$ are optimal yielding the same net profit.

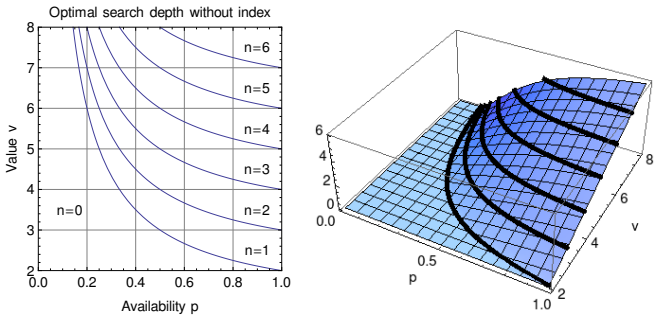


Fig. 4. The optimal search depth without index (left) and the corresponding mean net profit (right) as a function of p and v.

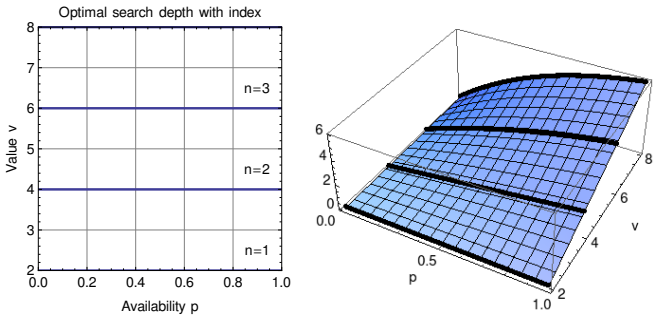


Fig. 5. Optimal search depth with index (left) and the corresponding mean net profit (right) as a function of p and v.

C. Examples

Figure 4 (left) depicts the optimal search depth $n^{\mathrm{L}}_{\mathrm{opt}}$ as a function of the availability p and the value of the searched content v when an index is not available. The figure on the right then illustrates the *mean net profit* per search with this optimal search depth. We note that the flat area at bottom left corner corresponds to case where the value of the searched content is too small to justify sending of any queries. Let us next consider our example in the ideal case where a global index is available. In this case, only those searches where the content can be found within the distance of $\lceil v/2 \rceil$ are actually carried out. The *expected net profit* in this case is $w^{\mathrm{L}}_{\mathrm{i}}(\lceil v/2 \rceil)$. Figure 5 (left) depicts the optimal search depth $n^{\mathrm{L}}_{\mathrm{opt,i}}$ as a function of the availability p and the value of the searched content v when the index is available. First we note that $n^{\mathrm{L}}_{\mathrm{opt,i}}$ does not depend on p, obviously. Then also the expected net profit behaves more smoothly and is clearly higher than without the index when $p < 1$.

Figure 6 illustrates the expected net profit when the value of the searched content is $v = \{3, 5, 10\}$ with the optimal search depths. On the x-axis is the availability probability p, and the y-axis corresponds to the mean net profit (per search). We can see that as $p \to 1$, the gain from indexing the content vanishes, as expected. However, when p is small, say $p < 0.5$, the index is very useful and leads to a significant improvement of the average search result.

Figure 7 depicts the mean net profit in logarithmic scale. The left figure corresponds to a system without an index, and the right figure to a system with an index. The dotted curves on left figure are with a fixed search depth of $n = \lfloor v/2 \rfloor$. We can see that when p is too small, searching content when one

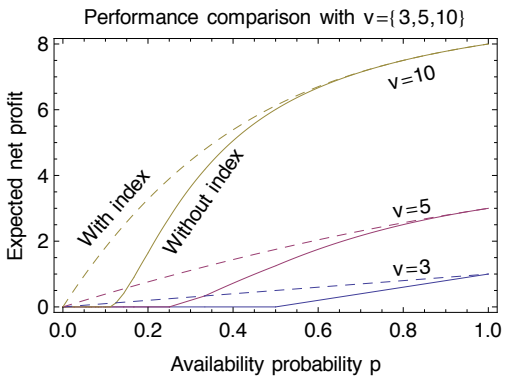


Fig. 6. Expected net profit with and without indexing with the optimal search depths for $v = \{3, 5, 10\}$.

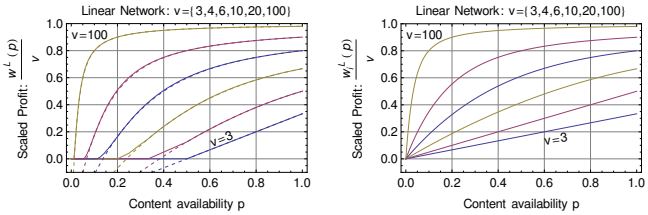


Fig. 7. Performance without (left) and with (right) index in linear network in logarithmic scale

should not can be very costly and yield a negative mean net profit. However, as soon as searching content makes sense, the difference between the optimal depth, given by (8), and the maximum sensible one, $n = \lfloor v/2 \rfloor$, is not very significant. Comparing the two graphs, we also observe that as $p \to 1$, the gain from index becomes negligible, as expected.

D. Indexing Gain

We define **indexing gain** as

$$\gamma := \frac{w^{\mathrm{L}}_{\mathrm{i}}(p) - w^{\mathrm{L}}(p)}{v}. \tag{10}$$

which is the difference between the expected net gain *with* and *without index* normalized by the value of the content v. This is illustrated in Figure 8, where we can see that the gain is high when p is smallish and v increases. At the limit $p \to 1$, the gain converges to zero, naturally. We can also see that for large values of v, there is a small content availability p at which the gain remains about $(0.2, 0.25)$. That is, $w^{\mathrm{L}}_{\mathrm{i}}(p) - w^{\mathrm{L}}(p) \to \infty$, i.e., the absolute performance with index can be better by an arbitrarily high amount.

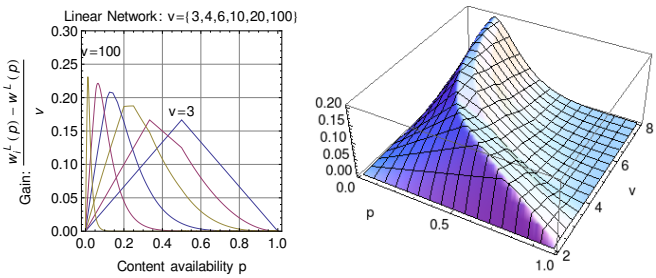


Fig. 8. Indexing gain γ illustrated as a function of p and v.

IV. Two Linear Networks

Next, we consider search in a two directional linear network of nodes. A node searching for a content in a two directional linear network nodes has two sets of linear network to search from. Figure 1b illustrates a search in two directional linear networks.

A. Mean Net Profit

Without Indexing: The basic search strategy in two linear networks without *indexing*, denoted by $2\mathrm{D}(n)$, is defined by a search depth of n, where the search is first carried up to a depth of n_1 hops in one direction. If the search is not successful after some time t, the search is further carried out up to a depth of n_2 hops in the second direction. In terms of cost, switching to the other direction after n_1 depth is always better as there is no return cost incurred for the first n_1 nodes. But this comes at the expense of latency t, the time the searching node has to wait before deciding to issue a new query in the other direction.

When a search is successful in the first direction, the total cost of the search, $C^{2\mathrm{D}}(n)$, will be $2i_1$ for $i_1 \leq n_1$ and $n_2=0$, (the same as that of a linear network). For $i_2>0$, meaning search is also carried out in the second direction, the cost will become $2i_2 + n_1$. This is because there is no return cost incurred for the first n_1 nodes as the content is not found in the first direction. Otherwise, the search is terminated at the n^{th} node. This will cost $n=n_1 + n_2$.

We can represent the mean *net profit* with search strategy $2\mathrm{D}(n)$, i.e. without an index, as

$$w^{2\mathrm{D}}(n_1,n_2) = \sum_{i=1}^{n_1+n_2} [(1-p)^{i-1}p(v - C^{2\mathrm{D}}(n_1,n_2))] - (n_1+n_2)(1-p)^{n_1+n_2}, \quad (11)$$

where,

$$C^{2\mathrm{D}}(i_1,i_2) = \begin{cases} 2i_1, \text{if} i_2 = 0, \\ 2i_2 + n_1, \text{otherwise} \end{cases} \quad (12)$$

We note that for $i_2 = 0$, Eq.(11) reduces to the simple linear network discussed in Section (III).

With Indexing: Searching in two linear networks with *index*, $2\mathrm{D}_i(n)$, follows the physical topology of the network as the index helps to effectively ignore one of the directions that is known not to host the content (or at a higher cost than the other). With the availability of the index, a searching node knows in which segment of the network the content is located, if it exists. Similarly, the search strategy in two linear networks with indexing, $2\mathrm{D}_\mathrm{i}(n)$, is defined again by the maximum search (or retrieval depth) n. However, a query is sent only if the content is known to be located in two linear branches of the network. Otherwise the search ends already at the source node. The *expected net profit* with the search strategy $2\mathrm{D}_\mathrm{i}(n)$, i.e. with *index*, will therefore be:

$$w_\mathrm{i}^{2\mathrm{D}}(n_1,n_2) = w_\mathrm{i}^{2\mathrm{D}}(n_1,0) = \sum_{i=1}^{n}(1-p)^{i-1}p(v-2i), \quad (13)$$

which after some manipulation also reduces to

$$w_\mathrm{i}^{2\mathrm{D}}(n_1,0) = \frac{(2+(2n_1-v)p)(1-p)^{n_1} - 2 + pv}{p}. \quad (14)$$

For the unbounded search, we obtain the limit,

$$\lim_{n_1\to\infty} w_\mathrm{i}^{2\mathrm{D}}(n_1,0) = v - \frac{2}{p},$$

as expected.

B. Optimum Search Depth

We mentioned above that when searching in two linear networks, switching the search direction is better. Since the searching node can not search one node at a time and switch back and forth repeatedly, there should be some optimal depth $n_{1,\mathrm{opt}}^{2\mathrm{D}}$ to search in the first direction before deciding to search the other direction, again up to an optimal depth of $n_{2,\mathrm{opt}}^{2\mathrm{D}}$, such that Eq.(11) is maximized at $(n_{1,\mathrm{opt}}^{2\mathrm{D}}, n_{2,\mathrm{opt}}^{2\mathrm{D}})$. In colloquial words, it is always better to search for some depth, then switch direction and search a little more, and hence the following theorem.

Theorem 1: The optimal search depths $n_{1,\mathrm{opt}}^{2\mathrm{D}}$ and $n_{2,\mathrm{opt}}^{2\mathrm{D}}$, maximizing $w^{2\mathrm{D}}(n_1,n_2)$, are such that $n_{1,\mathrm{opt}}^{2\mathrm{D}} \leq n_{2,\mathrm{opt}}^{2\mathrm{D}}$.

Proof: We proof this by comparing $w^2(n_1,n_2)$ and $w^2(n_2,n_1)$. The case $n_1 = n_2$ is trivial, and hence we can assume that $n_1 < n_2 < v/2$.

First we break the search process into two parts: First part is the going "forward" until the content is found or the search is dropped, and the second part is the going "backwards" when the content found is returned. The expected cost of the forward part, denoted by F, can be written as:

$$\mathrm{F}(\mathrm{n}_1,\mathrm{n}_2) = \sum_{i=1}^{n_1+n_2}(1-p)^{i-1}pi + (n_1+n_2)(1-p)^{n_1+n_2}.$$

We note that exchanging n_1 and n_2 makes no difference to $F(\cdot)$. For the backward part, occurring when the searched content is found, the expected net gain, $w^{2\mathrm{D}}(n_1,n_2)$, is

$$\begin{aligned} w^{2\mathrm{D}}(n_1,n_2) &= \sum_{i=1}^{n_1}(1-p)^{i-1}p(v-i) + \\ &\quad (1-p)^{n_1}\sum_{i=1}^{n_2}(1-p)^{i-1}p(v-i), \end{aligned}$$

which reduces to

$$\begin{aligned} w^{2\mathrm{D}}(n_1,n_2) &= [n_1(1-p)^{n_1}p + n_2(1-p)^{n_1+n_2}p \\ &\quad -(pv-1)[(1-p)^{a+b}-1]]/p. \end{aligned}$$

We then compare the gain by exchanging n_1 and n_2, such that $w^{2\mathrm{D}}(n_1,n_2) - w^{2\mathrm{D}}(n_2,n_1)$ becomes

$$\begin{aligned} w^{2\mathrm{D}}(n_1,n_2) - w^{2\mathrm{D}}(n_2,n_1) &= n_2(1-p)^{n_2}[(1-p)^{n_1}-1] \\ &\quad -n_1(1-p)^{n_1}[(1-p)^{n_2}-1]. \end{aligned}$$

But, for $n_1 < n_2$, changing the multiplier n_2 in the first term of the above by n_1 gives us

$$\begin{aligned} w^{2\mathrm{D}}(n_1,n_2) - w^{2\mathrm{D}}(n_2,n_1) &> n_1(1-p)^{n_2}[(1-p)^{n_1}-1] \\ &\quad -n_1(1-p)^{n_1}[(1-p)^{n_2}-1], \end{aligned}$$

which, after some manipulation, reduces to

$$w^{2\mathrm{D}}(n_1,n_2) - w^{2\mathrm{D}}(n_2,n_1) > n_1[(1-p)^{n_1} - (1-p)^{n_2}].$$

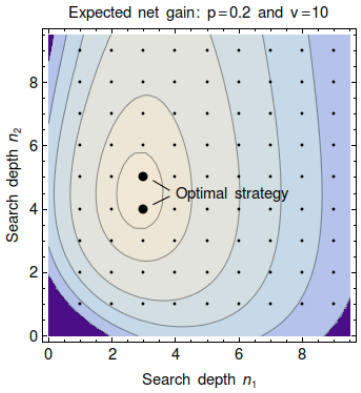


Fig. 9. Search in a two directional Linear Network

Since $p \leq 1$, and $n_1[(1-p)^{n_1}-(1-p)^{n_2}] \geq 0$ for all $n_1 < n_2$, it follows for $n_1 < n_2$,

$$w^2(n_1, n_2) > w^2(n_2, n_1),$$

which concludes the proof. ■

Alternatively, the optimal search depths $n^{2D}_{1,\text{opt}}$ and $n^{2D}_{2,\text{opt}}$ can also be computed numerically by maximizing Eq. (11) or by using the following greedy algorithm.

$n_1 \leftarrow 0$
while Max_Gain(n_1+1) > Max_Gain(n_1) **do**
  $n_1 \leftarrow n_1 + 1$
end while
return (n_1)

The **Max_Gain**(n_1) is a subroutine that returns the maximum of Eq. (11) for a fixed search depth n_1 in the first linear network. For n_2, the algorithm is as follows:

$n_2 \leftarrow 0$
while $w^{2D}(n_1, n_2+1) > w^{2D}(n_1, n_2)$ **do**
  $n_2 \leftarrow n_2 + 1$
end while
return $w^{2D}(n_1, n_2)$

C. Example

Suppose next that the availability of the content is $p = 0.2$, and the value of the content is $v = 10$. Figure 9 depicts the equi-value contours for the search strategy with depths (n_1, n_2). We can see that it is beneficial to first search only up to a short distance, and then if that fails, the second query may travel a bit longer. In this case, the optimal policy $(n^{2D}_{1,\text{opt}}, n^{2D}_{2,\text{opt}})$ that maximize the *mean net profit* is $(3, 4)$ or $(3, 5)$, both yielding the same expected net profit of about 2.37.

On the other hand, when an index is available, the searching node will not issue any query in the direction where the content is further away, but chooses the other direction immediately, given the content lies within a distance of at most $\lfloor v/2 \rfloor$ hops. In this case, the optimal policy $(n^{2D}_{1,\text{opt}}, n^{2D}_{2,\text{opt}})$ that maximizes the *mean net profit* becomes $(0, \lfloor v/2 \rfloor)$.

V. K Linear Networks

In this section, we generalize search in k-directional linear network. In this type of topology, the searching node has k-directly connected linear network of nodes forming a star topology. Figure 1c shows an example of a k-directional network for k=4.

A. Mean Net Profit

Without Indexing: The basic search strategy in k-linear networks without *indexing*, denoted by KD(n), is defined by search depths $\{n_j\}$ with $n = \sum_{j=1}^{k} n_j$; where the search is carried up to a depth of n_j hops in the j^{th} direction before a new search is issued at the $(j+1)^{th}$ direction, until the search is resolved, or $j = k$ and the search fails.

A search that is terminated (successful or failed) in the first direction is similar to search in linear network and results in a *mean net profit* similar to Eq.(2). If the search is further carried out in the second direction, the *mean net profit* will become

$$w^{\text{KD}}(n)=(1-p)^{n_1}\sum_{i=1}^{n_2}[(1-p)^{i-1}p(v-2i-n_1)]-n(1-p)^n,$$

and

$$w^{\text{KD}}(n) = (1-p)^{n_1+n_2}\sum_{i=1}^{n_3}[(1-p)^{i-1}p(v-2i-(n_1+n_2))] - n(1-p)^n,$$

for when the search has progressed up to the third direction.

Now if we let $\tilde{n}_j = \sum_{i=1}^{j-1} n_i$, and $\tilde{v}_j = v - \tilde{n}_j$, we can generalize the mean *net profit* for a search in k linear networks without index, as

$$w^{\text{KD}}(n) = \sum_{j=1}^{k}(1-p)^{\tilde{n}_j}\sum_{i=1}^{n_j}[(1-p)^{i-1}p(\tilde{v}_j-2i)]-n(1-p)^n. \quad (15)$$

With Indexing: Searching in K-directional linear network with index, a node will search k branches and at most n nodes. Since the node knows where the content is, it will forward its queryto the dimension that has the content at the shortest distance, if any.

The *expected net profit* per query in k-directional linear network with indexing can thus be expressed as

$$w_i^{\text{KD}}(n) = \sum_{i=1}^{n} P\{\tilde{D}=i\}(v-2i), \quad (16)$$

where $P\{\tilde{D} = i\}$ is the probability of finding a content at a distance of i hops in any of the k-directions in the network. But the probability of a content being located at a distance of i hops in a linear network is $(1-p)^{i-1}p$, which can also be generalized for k directions by replacing p with $p^{(k)}=1-(1-p)^k$, yielding

$$P\{\tilde{D} = i\} = [1-p^{(k)}]^{i-1}p^{(k)}. \quad (17)$$

Putting (17) in (16), gives the *mean net profit* with index,

$$w_i^{\text{KD}}(n) = \left(v - \frac{2}{1-(1-p)^k}\right)\left(1-(1-p)^{kn}\right) + 2n(1-p)^{kn}. \quad (18)$$

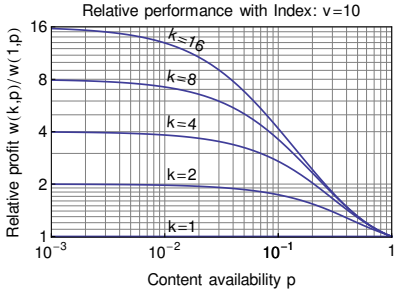


Fig. 10. Effect of topology with index.

B. Optimum Search Depth

Similar to the optimum search strategy in two-directional linear networks discussed in Section IV-B, the optimum search policy when searching in k-directional topology without an index is to search in one direction and if that fails, then search a bit more in the next direction[2] and so on, i.e., $\{n_j\}$ with $n_1 \leq n_2 \leq \ldots \leq n_k$. For large k or when $k \to n$, the searched content will be more closer to the searching node and the network will reduce to a star-topology. Therefore, it is beneficial to switch direction than search deeper, and the optimum search depth becomes 1.

Obviously, with an index the optimal search depth is $\lfloor v/2 \rfloor$.

C. Example: Effect of Directions with Index

Let us next vary the number of directions k in the case with index. Figure 10 depicts the gain from having more than one direction as a function of availability p in logarithmic scale for $v = 10$,

$$\beta_i(k) := \frac{w_i^{\mathrm{KD}}(k)}{w_i^{\mathrm{KD}}(1)}.$$

We observe that the gain behaves linearly as a function of k when p is small, in case of which a multi-directional search is useful. In fact, it is easy to show that

$$\lim_{p\to 0} \beta_i(k) = k, \qquad \text{and} \qquad \lim_{p\to 1} \beta_i(k) = 1,$$

for all $v > 2$.

VI. Star Network

Next, we consider search in a star network of nodes, where all k other nodes are directly connected to the searching node forming a star topology. This topology follows as a special case of K linear networks when we restrict the search perimeter to a single hop (see Figure 1c).

A. Mean Net Profit

Without Indexing: The basic search strategy in a star network without index, $\mathrm{ST}(n)$, is defined by the maximum search depth n: a search is carried up to n neighboring (directly connected) nodes, and if the content is found at node $i \leq n$, then the search ends there and the reply is sent. Search strategy $\mathrm{ST}(n)$ is in fact a special case of $\mathrm{KD}(n)$, where n=k, and n_j=1.

[2]Due to space limitation, we omit the formal proof for k>2, but the result itself is self-evident from the case k=2.

The *mean net profit* with the search strategy $\mathrm{ST}(n)$ that checks at most n directly connected nodes is

$$w^{\mathrm{ST}}(n) = \sum_{i=1}^{n} (1-p)^{i-1} p(v-i-1) - (1-p)^n n,$$

which reduces to

$$w^{\mathrm{ST}}(n) = \frac{(1-(1-p)^n)(p(v-1)-1)}{p}. \tag{19}$$

We note that (19) is negative if $p(v-1) < 1$, in case of which the optimal search depth is zero, $n = 0$. Otherwise, (19) is maximized when $n \to k$. Consequently, the optimal search depth in star topology without index is

$$n_{\mathrm{opt}}^{\mathrm{ST}} = \begin{cases} 0, & \text{if } p(v-1) \leq 1 \\ k, & \text{otherwise.} \end{cases} \tag{20}$$

With Indexing: With an ideal index, the situation becomes a bit different. The search strategy in physical star networks with indexing, a query will be sent only if the content is known to be carried by one of the neighboring nodes, and otherwise the search ends already at the source node. Hence, the *expected net profit* with the search strategy $\mathrm{ST_i}$ with an index is

$$w_{\mathrm{i}}^{\mathrm{st}} = [1-(1-p)^k](v-2), \tag{21}$$

which follows from Eq.(18) as a special case.

B. Asymptotic case $k \to \infty$

Let us next assume a dense network with an infinite number of nodes, $k \to \infty$. In this case, the net profit without index reduces to

$$\lim_{k\to\infty} w^{\mathrm{ST}}(k) = v - 1 - 1/p, \qquad (p(v-1) > 1)$$

and with an index,

$$\lim_{k\to\infty} w_{\mathrm{i}}^{\mathrm{st}}(k) = v - 2.$$

The absolute difference is thus $(1-p)/p$.

C. Indexing gain in star network

Let us again study the indexing gain defined in (10). For $p(v-1) > 1$, we obtain

$$\gamma^{\mathrm{ST}} = \left(1-(1-p)^k\right) \frac{1-p}{pv},$$

which is a decreasing function of p. For $p(v-1) \leq 1$ (and $v > 2$), it is not worth it to search content if there is no index, and the indexing gain reduces to $\gamma^{\mathrm{ST}} = w_{\mathrm{i}}^{\mathrm{st}}(n)/v$, yielding

$$\gamma^{\mathrm{ST}} = \left(1-(1-p)^k\right) \frac{v-2}{v}.$$

The above is an increasing function of p. Therefore, the maximum gain from having an index is obtained at $p = (v-1)^{-1}$,

$$\gamma_{\max}^{\mathrm{ST}} = \left(\left(\frac{v-2}{v-1}\right)^k - 1\right) \frac{v-2}{v},$$

which achieves maximum when $k, v \to \infty$, yielding $\gamma_{\max}^{\mathrm{ST}} \to 1$. The same limit is in fact obtained for any $0 < p \leq (v-1)^{-1}$. In other words, in star topology, the absolute gain from having an index can be as high as it in theory can be.

VII. Conclusion

In this paper, we provided the basic analysis of indexing gain when searching content in spontaneous wireless networks that lack the infrastructure to utilize an indexing server. We compared search performance of a system without an *index* to that of a system with a global index of available contents.

Our analysis is very important in two ways. First, it gives us a preliminary understanding of indexing in infrastructure-less networks to see if the success of indexing contents in the Internet can also be replicated. To this end, we have shown with numerous search topologies and examples that *indexing* indeed provides a clear gain. Without *indexing*, the critical decision is to *decide if it is worth to trigger a search at all*, which requires a good understanding of both *availability* and *value*of content.

For *linear networks*, we have shown in Figure.6 that *indexing* is especially useful when the content sought after is rare, i.e. p is small, and its value v is high enough. Figure 8 on the other hand gives us insight as to how much the cost of building and maintaining such an index can be, as the gain with indexing in linear networks seems to be less than $v/4$ for all (p, v).

For search in more complicated networks, such as *two* and k directional linear networks, the basic search strategy is to search a bit more whenever a search switches to a new direction, i.e., the *optimal search depth* without an index is of form $n_1 \leq n_2 \leq \ldots \leq n_k$ which we have shown for k=2. With the availability of an index, we have found closed form results (14) and (18) that quantify the *expected mean gain* with an indexing system.

Search in *star* networks is interesting in its own, as a node without index will search infinitely, or *not* at all. This is because in star topology, the additional cost for searching one more node is minimal, and a node should keep searching until the content is found. In contrast, with an index, the optimal search depth is always *one*.

We have considered specific regular topologies in this work as it gives us insight into more general cases, where topologies can be, e.g. *random* or *change more dynamically*. In this paper, we did not look at the *cost of building an index*, but rather analyzed the gain an index can provide so that we can characterize the premises under which indexing can be useful. *Accounting the effort of building and maintaining an index provides a better understanding of indexing gain in mobile opportunistic networks*, and hence it will be one of our topics for future study. Another question that can be further investigated is the gain of partial indexing, if global indexing in the network is difficult to achieve. It would also be interesting to study more realistic scenarios which for the assumed mobile opportunistic network setup, would probably entail asymmetry between forward and return paths (in terms of query/content forwarding costs) in a single linear search path, cost asymmetry among different paths, and so on.

Furthermore, it is also important to study and adopt existing systems on how a "global index" becomes available in spontaneous wireless networks, such as mobile opportunistic networks which are inherently infrastructure-less. To this end, we propose "floating index" as another potential future work, where schemes like *floating content* [1] could be used to provide an indexing functionality in spontaneous wireless networks.

VIII. Acknowledgments

This work was supported by the Academy of Finland in the PDP project with grant number 260389.

References

[1] J. Kangasharju, J. Ott, and O. Karkulahti, "Floating Content: Information Availability in Urban Environments," in *Proc. of IEEE Percom 2010, Work in Progress session*, March 2010.

[2] J. Ott, E. Hyytiä, P. Lassila, T. Vaegs, and J. Kangasharju, "Floating Content: Information Sharing in Urban Areas," in *IEEE Percom*, 2011.

[3] E. Hyytiä, J. Virtamo, P. Lassila, J. Kangasharju, and J. Ott, "When does content float? characterizing availability of anchored information in opportunistic content sharing," in *IEEE Infocom*, Shanghai, China, Apr. 2011.

[4] C.-M. Huang, K. chan Lan, and C.-Z. Tsai, "A survey of opportunistic networks," in *Advanced Information Networking and Applications - Workshops, 2008. AINAW 2008. 22nd International Conference on*, March 2008, pp. 1672–1677.

[5] R. Steinmetz and K. Wehrle, *Peer-to-peer systems and applications*. Springer, 2005.

[6] S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Computing Surveys (CSUR)*, vol. 36, no. 4, pp. 335–371, 2004.

[7] X. Li and J. Wu, "Searching techniques in peer-to-peer networks," *Handbook of Theoretical and Algorithmic Aspects of Ad Hoc, Sensor, and Peer-to-Peer Networks*, pp. 613–642, 2006.

[8] E. Adar and B. A. Huberman, "Free riding on gnutella," *First Monday*, vol. 5, no. 10, 2000.

[9] M. Ripeanu, "Peer-to-peer architecture case study: Gnutella network," in *Peer-to-Peer Computing, 2001. Proceedings. First International Conference on*. IEEE, 2001, pp. 99–100.

[10] N. Chang and M. Liu, "Controlled flooding search in a large network," *Networking, IEEE/ACM Transactions on*, vol. 15, no. 2, pp. 436–449, April 2007.

[11] C. Avin and C. Brito, "Efficient and robust query processing in dynamic environments using random walk techniques," in *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks*, ser. IPSN '04. New York, NY, USA: ACM, 2004, pp. 277–286. [Online]. Available: http://doi.acm.org/10.1145/984622.984663

[12] S. D. Servetto and G. Barrenechea, "Constrained random walks on random graphs: Routing algorithms for large scale wireless sensor networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, ser. WSNA '02. New York, NY, USA: ACM, 2002, pp. 12–21. [Online]. Available: http://doi.acm.org/10.1145/570738.570741

[13] W. W. Terpstra, J. Kangasharju, C. Leng, and A. P. Buchmann, "Bubblestorm: Resilient, probabilistic, and exhaustive peer-to-peer search," in *Proceedings of the 2007 ACM SIGCOMM Conference*. New York, NY, USA: ACM Press, Aug 2007, pp. 49–60.

[14] S. Bayhan, E. Hyytiä, J. Kangasharju, and J. Ott, "Seeker-assisted information search in mobile clouds," in *Proceedings of the second ACM SIGCOMM workshop on Mobile cloud computing*. ACM, 2013, pp. 9–14.

[15] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*. IEEE, 2002, pp. 5–14.

[16] W. Shi, S. Li, G. Peng, T. Ma, and X. Lin, "Local index tree for mobile peer-to-peer networks," in *Grid and Cooperative Computing-GCC 2004*. Springer, 2004, pp. 650–656.