Content and Service Replication Strategies in Multi-hop Wireless Mesh Networks

Shudong Jin
Division of Computer Science, EECS
Case Western Reserve University
Cleveland, OH 44106
shudong.jin@case.edu

Limin Wang
Division of Computer Science, EECS
Case Western Reserve University
Cleveland, OH 44106
limin.wang@case.edu

ABSTRACT

Emerging multi-hop wireless mesh networks have much different characteristics than the Internet. They have low dimensionality and large diameters. Content and service replication can greatly improve their scalability. However, replication strategies in such networks have not been well studied. This paper studies the optimality of replication strategies and explores it in multi-hop wireless mesh networks for the first time. We start with the problem of determining the optimal numbers of replicas for a set of objects which have distinct probabilities of being requested in large 2-D mesh networks. We reveal the structure of the optimal replication strategy to minimize object access cost. To minimize average cost to access an object in 2-D mesh networks, the optimal strategy replicates an object such that the number of its replicas is proportional to $p^{0.667}$, where p is the access probability of the object. This result indicates the inefficiency of demand-driven content and service replication in 2-D mesh networks, where an object is replicated such that the number of its replicas is proportional to p. We further study practical, online algorithms to approximate the optimal strategy. Interestingly, the optimal replication can be approximated well by a localized replacement algorithm. The algorithm utilizes only handy information and incurs no communication overhead. The paper demonstrates a significant performance gain by the optimal strategy, and the effectiveness of the online replacement algorithm.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design—*Network topology, wireless communication*

General Terms

Algorithms, Performance

Keywords

Mesh networks, multi-hop communication, replication, cache replacement

1. INTRODUCTION

In multi-hop wireless networks, communication between two end hosts is carried out through a number of intermediate nodes whose function is to relay information. An increasing number of multi-hop wireless deployments and proprietary commercial solutions have focused on a class of networks termed *mesh networks*, which serve as access networks using multi-hop routing and forwarding techniques, but are less constrained by node mobility and power consumption. The scale of these networks can be large, as they are composed of thousands to millions of computing nodes to offer a huge aggregate capacity. On the other hand, an individual node has very limited capacity. Light-weight wireless devices have low processing speed; they are equipped with small memory; they can only transmit data and signals at relatively low rates and ranges.

These characteristics strongly influence application system designs in such wireless mesh networks. Application systems often choose decentralized implementations. Each computing node performs simple operations, but many of them cooperate to complete large and complex tasks. Application systems are also unstructured, as the networks cannot afford to maintain a fully-fledged directory system. With these design choices, a problem is how to efficiently share resources available in the networks, including both content and service.

Replication of content and service naturally becomes an attractive approach to providing efficient and more reliable access. In a large-scale mesh network, if we put just a few replicas of an object (an object represents any content or service that is in digital format and can be replicated.), then applications can expect a replica within a much shorter distance. Thus, we significantly reduce communication overhead. In addition, by strategically placing replicas in the vicinity of applications, applications are less impacted by lossy wireless links and long delays in a wide geographical area. Yet, a challenging problem is how to replicate content and service for access in an efficient fashion. This paper explores and develops optimal replication strategies in 2-D wireless mesh networks.

A commonly used replication strategy works in a demand-driven fashion: when an object is requested, it is pulled from the source of a replica and stored in a local node. Assumed that different objects have approximately the same lifetime, this approach leads to a property: the number of replicas of an object is *proportional* to the object's popularity. In this paper we show that this demand-driven proportional strategy is far from being optimal in 2-D mesh networks. To that end we start with the problem of determining the optimal numbers of replicas for a set of objects with distinct access probabilities, when the objective is to minimize object access cost (defined as the distance to the nearest replica). To minimize average access cost in 2-D mesh networks, the optimal strategy replicates an object such that the number of its replicas is proportional to $p^{0.667}$, where p is the access probability of the object.

An important goal of this work is to develop online algorithms to minimize the access cost, without any prior knowledge on the object popularity. Interestingly, we develop a localized replacement algorithm which works very well. When an object is pulled from another node, it is stored in the local node. However, its lifetime in the node is proportional to the access cost. To achieve this, we simply set this measure of cost as the cost function of a modified GreedyDual algorithm. We show via analysis and simulation that the replacement algorithm mimics the optimal replication strategies well. One of the advantages of the replacement algorithm is, it utilizes only handy local information and incur no communication overhead.

Several surprises from this work also underscore the importance of performance modeling and evaluation to gain insights on wireless mesh network algorithms and protocol designs. First, the paper shows the optimality of replication strategies in mesh networks. They are simple and intuitive, yet surprisingly they have not been identified before. Second, with an ideal wireless mesh network model, it is not difficult to find the optimal strategies. Third, once the optimal strategies are identified, we have very simple algorithms to approximate them with no additional network communication overhead.

The remainder of this paper is organized as follows. The next section describes a network model and the assumptions. In Section 3, we derive the optimal strategy to minimize access cost. In Section 4, we evaluate and compare several replication strategies under the assumption of Zipf-like popularity distribution. Section 5 presents an online replacement algorithm that mimics the optimal strategies. In Section 6, we describe our simulation of replacement algorithms and present the results. We briefly revisit related work and finally conclude the paper.

2. THE MODEL AND ASSUMPTIONS

Let us consider a 2-dimensional mesh network, where storage space is uniformly available at a large number of random locations (also called computing nodes). The nodes have network interfaces, but with limited transmission ranges. Physically neighboring nodes within a threshold distance can communicate with each other. Two well separated nodes may also exchange data and information, however, with help from other intermediate nodes. In other words, this is a multi-hop network. We do not assume that the nodes are static, although the mesh network is less constrained by node mobility.

Applications may frequently request the content or access the service available in the network. There are a number of objects and each object is replicated in the network to facilitate access. To satisfy a request, a replica of the requested object must be discovered and accessed. Although we focus on accessing a single replica, it is also possible to consider the case of accessing multiple replicas. To discover a replica, queries are sent to neighboring nodes, and if necessary, forwarded to further neighbors. In the context of this work, we focus on the cost to access the objects, but not the cost to discover the objects. In an unstructured system, queries can be flooded to discover replicas (or pointers to replicas). After discovering a set of replicas, the *nearest* one is accessed. Figure 1 gives an example. To satisfy a request originating at the center of the circle, several neighboring nodes (shown in the shaded area) are contacted (via flooding search for example), and eventually the nearest replica (stored in a solid node) is accessed.

We have a set of N independent objects. Requests originate in the mesh network uniformly and independently. For each request, the probability of requesting the i-th object is p_i, for $i = 1, 2, ..., N$. Without loss of generality, we assume $p_i \geq p_j$ if $i \leq j$. We do not consider requests for a non-existing object. In addition, we do not assume the objects have the same size.

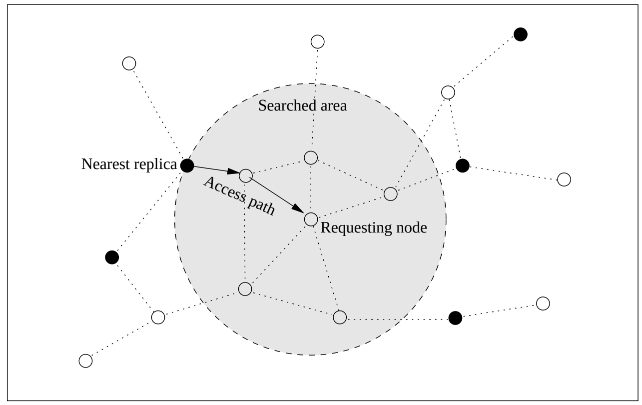

Figure 1: An illustration of the network model used in analysis.

Table 1: Notations Used in This Paper.

Notation	Explanation
N	Number of unique objects
p_i	Probability of requesting the i-th object
d_i	Density of replicas for the i-th object
C_i	Average cost to access the i-th object
C	Average cost to access an object

The problem we consider here is how to determine the density d_i of the replicas of the i-th object, under the aggregate capacity constraint $\sum_{i=1}^{N} d_i = 1$, such that we maximize the utility of available storage, or equivalently, minimize the access cost. Note we normalize the *density* of available storage. We define the access cost as follows.

DEFINITION 1. *Access cost is defined as the Euclidean distance between the nearest replica and the requesting node.*

Let C_i denote the average access cost of the i-th object and let C denote the average access cost of all object. All notations in this model are summarized in Table 1.

Ideally, we should put in the network as many replicas as possible, *i.e.* increase d_i for each object, such that its access cost is low. However, here a constraint is $\sum_{i=1}^{N} d_i = 1$. Before we proceed to solve the problem in the next section, we have an important observation which will be key to our derivation. In a 2-D mesh network, the access cost, defined as the Euclidean distance between the nearest replica and the requesting destination, satisfies the following proportional relationship.

$$C_i \quad \propto \quad 1/\sqrt{d_i}. \tag{1}$$

It is not hard to imagine, in a two-dimensional space, if we increase the density of the replicas of an object to four times, then we decrease the expected access cost by a half.

Remarks: In this section we explicitly or implicitly made several simplifying assumptions to ease analysis. (1) First, we assume the large 2-dimensional mesh network is unbounded. This allows us to ignore details risen from the boundary. This is reasonable since in a large network the number of nodes close to the boundary is limited. Nevertheless, in our simulation later the network will be bounded. (2) Second, we assume available storage is uniformly distributed in the network. However, this is not required for our algorithms to work well. (3) Third we assume the requests originate in the mesh network uniformly and independently, and the object size is uniform too. This assumption is for us to derive the optimal solutions. We consider geographical locality and non-uniform object size when we design our algorithms later. (4) Fourth, we do not consider the consistency problem arisen from replication since our objective is to study replication strategies only. (5) Finally, we define the access cost as the Euclidean distance between the requesting node and the replica. We should acknowledge that there are potentially many different (and plausible) definitions of the access cost. For example, an alternative definition is the number of hops between the requesting node and the replica. We note that in large mesh networks, the number of hops is usually proportional to the physical distance. It is worth noting that a more accurate definition of the access cost also relies on the routing algorithm. The routing algorithm decides via which intermediate nodes to deliver the object (if it is impossible to have a direct one-hop connection between the replica and the requesting node).

3. OPTIMAL REPLICATION STRATEGY

Our objective is to minimize the average access cost by determining the density d_i for each object. Intuitively, if an object is more frequently requested, then it should have higher density, so that likely a replica can be accessed at a lower cost. Here, the key question is, what is the quantitative relationship between d_i and p_i in the optimal strategy? We have the following result:

THEOREM 1. *To minimize average access cost, the density d_i for the i-th object should be proportional to $p_i^{2/3}$.*

To proof this, let us first examine the average access cost C, which is defined as:

$$C = \sum_{i=1}^{N} p_i C_i. \quad (2)$$

That is, the average access cost is the weighted sum of the access cost of all objects. From Equation (1), we have $C_i = \Omega/\sqrt{d_i}$, for $1 \le i \le N$, where Ω is some constant. Using this expression to substitute C_i in Equation (2) and eliminating the constant factor, we find we need to minimize

$$\sum_{i=1}^{N} p_i/\sqrt{d_i}, \quad (3)$$

under the constraint $\sum_{i=1}^{N} d_i = 1$.

This minimum cost problem is equivalent to a maximum utility problem. The optimal solution of this problem can be obtained by applying the *law of equi-marginal utility*[1]. More specifically, let utility function $U_i(d_i) = -p_i/\sqrt{d_i}$. We have marginal utility function $U_i'(d_i) = \frac{p_i}{2} d_i^{-3/2}$. Notice that U' is a monotone decreasing function of d_i, meaning a diminishing return of increasing the density d_i for the i-th object. This property allows us to apply the law of equi-marginal utility. We simply let $U_i'(d_i) = U_j'(d_j)$, for any $1 \le i, j \le N$. This results in $\frac{d_i}{d_j} = (\frac{p_i}{p_j})^{2/3}$. Equivalently, we can write it as:

$$d_i \propto p_i^{\frac{2}{3}}. \quad (4)$$

Let us call it *minimum-access* replication strategy. Recall that we are considering a boundless space. If the space is bounded, then this d_i may be translated to a fractional number of replicas for the i-th object. In other words, this optimal strategy may not exist in practice, where the network size is limited. However, this quantitative relationship gives us much insight on how to allocate storage space close to the optimal strategy. For example, in a two-dimensional mesh network if one object is 1000 times as popular as another, then this relationship states that the first object should have 100 times as many replicas as the second object, in order to minimize the average distance of the access path. This observation departs much from a simple *proportional* strategy. With the proportional strategy, the frequency at which an object is replicated is proportional to its popularity. Therefore, very popular objects will be over-replicated.

In the remainder of this section, we routinely present the cost functions of this optimal strategy. Since we have Equation (4), it is easy for us to compute the cost functions. First, from Equation (1) we have

$$C_i \propto p_i^{-\frac{1}{3}}. \quad (5)$$

That is, in a two-dimensional mesh network, if one object is 1000 times as popular as another, and we apply the minimum-access strategy, then the cost to access the first object is one-tenth of the cost to access the second object. On the contrary, if we have the the proportional strategy (which has d_i proportional to p_i), the access cost for the i-th object would be proportional to $p_i^{-1/2}$. That means, compared to our minimum-access strategy, proportional strategy results in even lower access cost for more popular objects, but higher access cost for less popular objects. Based on Equation (5), we can rewrite the average access cost of all objects:

$$C = \sum_{i=1}^{N} p_i C_{access,i} \propto \sum_{i=1}^{N} p_i^{\frac{2}{3}} \quad (6)$$

with a constant factor dependent of the exact form of p_i.

Notably, the proportional replication strategy was also shown far from being optimal in minimizing query cost in unstructured peer-to-peer networks [5]. Indeed, this strategy is as worse as the *uniform* replication strategy which gives equal opportunity to objects with much different access probabilities. Cohen and Shenkar [5] have also discovered the *square-root* rule: to minimize query cost, the optimal strategy replicates an object (a file in the context of peer-to-peer systems) such that d_i is proportional to $\sqrt{p_i}$. Let us call it *minimum-query* replication strategy. For comparison, we also give the cost functions of the *minimum-query* replication strategy. Similarly it is easy to compute the access cost for individual objects and the average access cost for all objects. They are

$$C_i \propto p_i^{-\frac{1}{4}}. \quad (7)$$

$$C = \sum_{i=1}^{N} p_i C_i \propto \sum_{i=1}^{N} p_i^{\frac{3}{4}} \quad (8)$$

[1]This law is an extension to the law of diminishing marginal utility and it is often used in economics. It explains the behavior of a consumer in distributing his/her limited income among various goods and services, in order to obtain maximum satisfaction.

Table 2: A Summary and Comparison of Four Strategies.

	$d_i \propto$	$C_i \propto$	$C \propto$
Proportional strategy	p_i	$p_i^{-\frac{1}{2}}$	$\sum_{i=1}^{N} p_i^{\frac{1}{2}}$
Minimum-access strategy	$p_i^{\frac{2}{3}}$	$p_i^{-\frac{1}{3}}$	$\sum_{i=1}^{N} p_i^{\frac{2}{3}}$
Minimum-query strategy	$p_i^{\frac{1}{2}}$	$p_i^{-\frac{1}{4}}$	$\sum_{i=1}^{N} p_i^{\frac{3}{4}}$
Uniform strategy	$O(1)$	$O(1)$	$O(1)$

3.1 Summary and Comparison

Table 2 summarizes the results from this section. For comparison, we also show the cost functions of the proportional strategy and the uniform strategy. Detailed but very similar calculations of cost functions are omitted here. The proportional strategy gives the strongest, linear preference to popular objects. As a result, access cost for more popular objects is the lowest, but this is certainly achieved by sacrificing less popular objects. The minimum-query strategy still gives preference to more popular objects, but among the three strategies, its preference is the weakest (square root of p_i). As a result, access cost and query cost for more popular and less popular objects are most balanced. The minimum-access strategy is in the middle and achieves the lowest average access cost.

4. EVALUATION WITH ZIPF-LIKE DISTRIBUTION

This section considers an example object popularity distribution: Zipf-like distribution. We obtain the exact expressions of replica density functions and cost functions for various replication strategies.

Zipf-like distribution is a frequently used distribution to model variable popularity in many networked systems, including Internet caching and content distribution [3]. With Zipf-like distribution, $p_i \propto i^{-\alpha}$, for $i = 1, ..., N$. More precisely, $p_i = \frac{i^{-\alpha}}{\zeta(\alpha,N)}$. Here, the incomplete Riemann zeta function $\zeta(s, N)$ is defined as $\sum_{i=1}^{N} i^{-s}$. In the remainder of this section, we simply use $\zeta(s)$ instead of $\zeta(s, N)$ when it is clear from the context.

For the minimum-access strategy we have

$$d_i \propto i^{-\frac{2}{3}\alpha}. \quad (9)$$

Since we have also $\sum_{i=1}^{N} d_i = 1$, we can obtain:

$$d_i = \frac{i^{-\frac{2}{3}\alpha}}{\zeta(\frac{2}{3}\alpha)}. \quad (10)$$

From Section 2, in a 2-D mesh network the expected access cost for the i-th object is:

$$C_i = \frac{C_{base}}{d_i^{\frac{1}{2}}} = C_{base}\zeta^{\frac{1}{2}}(\frac{2\alpha}{3})i^{\frac{\alpha}{3}}, \quad (11)$$

where C_{base} denotes the base access cost. It is the the average access cost when the density of replicas is unit. Hence, the average access cost of all objects is calculated as:

$$C = \sum_{i=1}^{N} p_i C_i = C_{base}\frac{\zeta^{\frac{3}{2}}(\frac{2}{3}\alpha)}{\zeta(\alpha)}, \quad (12)$$

Similarly we have also computed the cost functions of other replication strategies under the assumption of Zipf-like distribution. We ignore the details but summarize the results in Table 3. For clarity we also ignore the common factor C_{base} in their cost functions.

Table 3: A Comparison of Replication Strategies with Zipf-like Distribution.

Strategy	d_i	C
Uniform strategy	$\frac{1}{N}$	$N^{\frac{1}{2}}$
Proportional strategy	$\frac{i^{-\alpha}}{\zeta(\alpha)}$	$\frac{\zeta(\frac{1}{2}\alpha)}{\zeta^{\frac{1}{2}}(\alpha)}$
Minimum-query strategy	$\frac{i^{-\frac{\alpha}{2}}}{\zeta(\frac{\alpha}{2})}$	$\frac{\zeta^{\frac{1}{2}}(\frac{\alpha}{2})\zeta(\frac{3}{4}\alpha)}{\zeta(\alpha)}$
Minimum-access strategy	$\frac{i^{-\frac{2}{3}\alpha}}{\zeta(\frac{2}{3}\alpha)}$	$\frac{\zeta^{\frac{3}{2}}(\frac{2}{3}\alpha)}{\zeta(\alpha)}$

4.1 Performance Comparison

We solve the cost functions in Table 3. Figure 2 shows the relative access cost of four replication strategies when the number of objects increases. Object popularity follows Zipf distribution with $\alpha = 1.5$. We have considered larger values of α and found the difference becomes more obvious. For all replication strategies, when N increases, access cost also increases. This is because when there are more objects, the density of each object decreases. This figure indicates that with the uniform strategy, the access cost is the highest. With the proportional strategy, the access cost is much lower (notice the logarithmic scale of the y-axis). Both the minimum-access strategy and the minimum-query strategy reduce the access cost by several times compared to the proportional strategy. The difference becomes more obvious when N is larger. For example, when $N = 1000000$, the minimum-access strategy reduces the cost of the proportional strategy by almost four times.

Figure 3 shows the performance when α varies in a range. The number of objects is fixed at one million. With the uniform strategy, the cost functions stay constant. It means, unfortunately this strategy does not exploit the skewed object popularity to reduce the access cost at all. Among the other strategies, the minimum-access strategy outperforms both the proportional strategy (by several times) and the minimum-query strategy (by a large fraction). The difference is more obvious when α is large (more skewed access probabilities).

5. REPLACEMENT ALGORITHMS

The last two sections have shown the existence of optimal strategies to minimize access cost. These strategies assume prior knowledge on the global popularity of the objects. This section describes an online algorithm that approximates the optimal strategy.

5.1 Algorithm Design

The optimal strategy can be pursued using distributed algorithms. For example, to minimize query cost in the context of unstructured peer-to-peer networks, several distributed algorithms were developed in [5] to approximate the minimum-query strategy. To minimize the access cost, a distributed algorithm can put replicas in the nodes along the path from the nearest replica to the requesting node. The primary drawback of this approach is that multiple replicas may be placed close to each other. In addition, the overhead required to frequently replace existing replicas of other objects cannot be simply ignored.

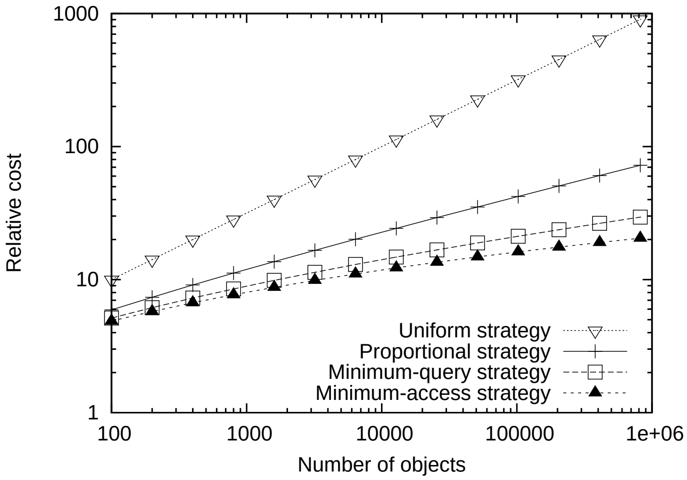

Figure 2: Performance of replication strategies as the number of objects varies, in 2-D mesh networks, Zipf-like distribution ($\alpha = 1.5$).

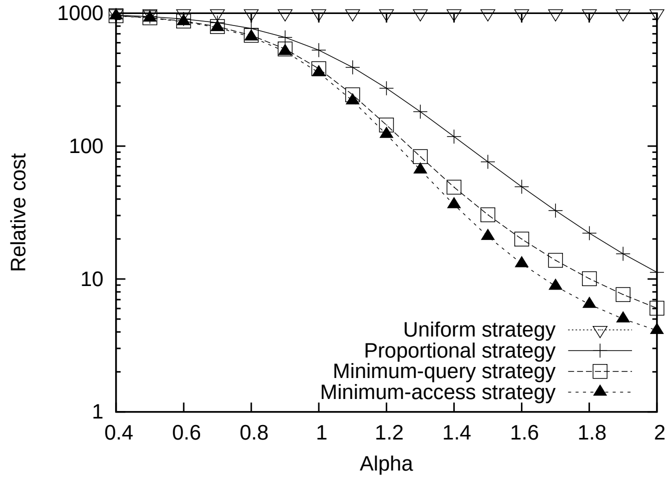

Figure 3: Performance of replication strategies as exponent α of Zipf-like distribution varies, in 2-D mesh networks, one million objects.

The optimal strategies can also be achieved using local replacement algorithms without any additional cost. This is our approach. The replacement algorithm keeps an object in the local cache (or in nearby nodes) when it is requested. However, if we do not distinguish different objects, we have the proportional strategy. This brings our attention to the GreedyDual approach [16, 4], which assigns different cost values to different objects. Figure 4 shows the pseudo code of an implementation. The algorithm associates a value H for each object. When a new object is brought in, the object with the lowest H value is evicted. This lowest H value is remembered as L. The new object is assigned a H value equal to its cost plus the L value. Here L is an inflation value. When L catches an object's H value, the object will be evicted. [2]

(1) Initialize $L \leftarrow 0$.
(2) for each request for object i do
(3) if i is in local cache
(4) then $H(i) \leftarrow L + C(i)$
(5) else while there is no space for i do
(6) $L \leftarrow min\{H(j) | j \text{ is in cache}\}$
(7) Evict j which satisfies $H(j) = L$
(8) Retrieve and store i
(9) $H(i) \leftarrow L + C(i)$

Figure 4: Pseudo code of GreedyDual Algorithm

Let us slightly modify this algorithm by removing line (4), *i.e.*, when an object is hit in a node, its H value is not restored.[3] In steady state, the L value increases at a constant rate and the expected lifetime of a replica in the local node is proportional to $C(i)$. Now let us set $C(i)$ to be a measure of access cost of the object, such that the mean value of $C(i)$ is proportional to $1/\sqrt{d_i}$.

The density d_i is a random variable $d_i(t)$ evolving with time. When the object's replicas are evicted from the networked system, $d_i(t)$ decreases, and when new replicas enter the system (with a lifetime value), $d_i(t)$ increases. We have a nonlinear dynamics system with differential equation:

$$\dot{d}_i(t) = -\gamma d_i(t) + \beta p_i / \sqrt{d_i(t)}, \quad (13)$$

where $0 < \gamma < 1$ is the rate at which the replicas of the object are evicted, and β is also a constant. Setting $\dot{d}_i(t) = 0$, we get the equilibrium point $\overline{d}_i = (\beta p_i/\gamma)^{2/3}$. This is consistent with the allocation Equation (4) of the minimum-access strategy. A remaining question is if the equilibrium point is stable. This is analyzed in the Appendix using the perturbation method.

5.2 Practical Consideration

In practice, this algorithm can be easily generalized to handle variable-size objects, just like [4]. The idea is to normalize the cost function $C(i)$ by the object size $S(i)$. The key value $H(i)$ is set to $L + C(i)/S(i)$ on each request for object i. In addition, the eviction process will be modified accordingly. If no ample space is available, we need to evict potentially multiple objects in order to make room for the newly requested object.

Another issue is related to the geographical locality of requests. In our analysis, we have assumed that the requests originate in the network uniformly. In practice, requests for a particular object may be concentrated in a small geographical area. We argue, without proof, that the proposed algorithm can handle such locality well. The algorithm is adaptive to the cost function. If an object is hot in a local area, even though it is not globally popular, the observed cost (the length of the access path) will be low since it is replicated more often in the vicinity. Therefore, the algorithm will not give preference to this object. On the contrary, a globally popular object can be cold locally, and its observed cost will be high. Hence the object received preference in replication. To interpret this in another way, our algorithm is *localized*, *i.e.* it utilizes only the object's local popularity and local access cost to make local replication decision. Thus it handles geographical locality well.

[2] The use of this inflation value L was proposed in [4]. By doing so, we can eliminate the need to update the key value of every object in the cache. Instead, we can have $O(\log n)$ computational complexity on each replacement if using a heap data structure. Although it first appears that the value of L is unbounded, it is trivial to handle this appropriately.

[3] We found it is necessary to remove this line to have the optimal solution. After the modification, the algorithm is similar to FIFO. It also implies the GreedyDual algorithm gives too strong preferences to more popular objects in distributed caches.

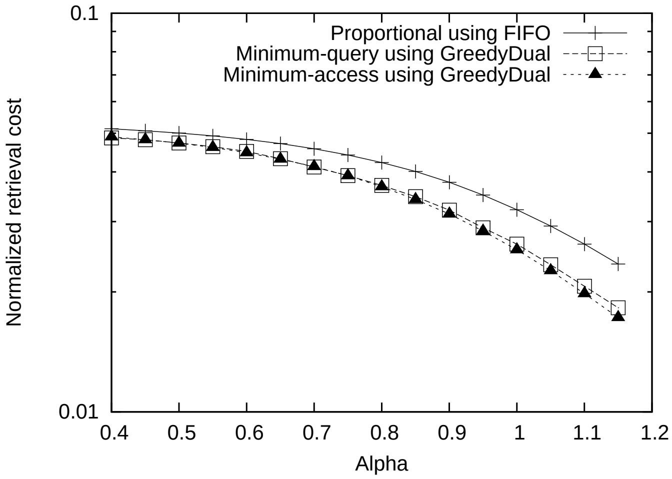

Figure 5: Performance of proportional strategy/FIFO, minimum-query strategy/GreedyDual, and minimum-access strategy/GreedyDual.

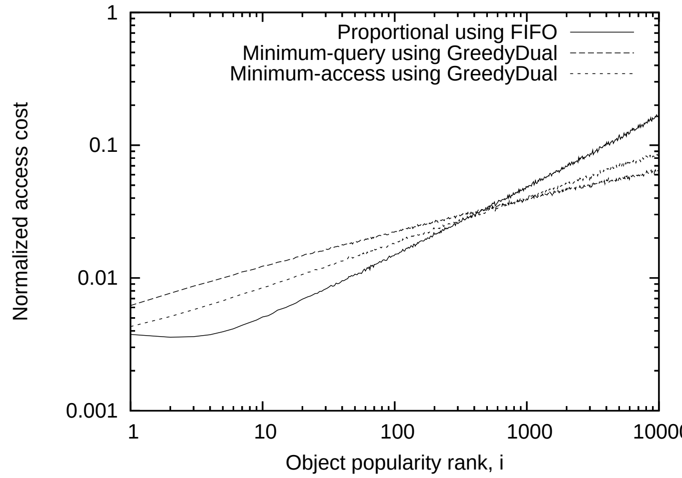

Figure 6: Access cost for objects with different access probabilities, three strategies.

6. EVALUATION OF REPLACEMENT ALGORITHMS

6.1 Evaluation Methodology

We have implemented a simulator. A number of nodes with uniform storage capacity are randomly distributed in a bounded square area. We have also simulated systems with variable-capacity nodes. The proposed algorithm can handle it well. However, since this is not the focus of this paper, we do not present those similar results here. Except one simulation study (where requests exhibit geographical locality), requests originate at random points in the square area. A request will be directed to a nearby node. If the requested object is not found in the local node, this node contacts other nodes and accesses the nearest replica of the object. We ignore the cost to discover the replicas of the object.

Accessing a remote replica usually incurs multi-hop communication. Therefore, we construct a multi-hop wireless mesh network. We set a threshold distance, such that two nodes within this threshold can communicate each other directly. In our simulation, on average a node in the wireless mesh network has more than 5 directly connected neighbors. A few disjoint nodes are ignored. To route traffic between two arbitrary nodes, Dijkstra's shortest path algorithm is implemented with the physical distance as the link cost. We acknowledge that in practice, the link cost can be determined in different ways, for example link quality metrics, such as packet loss rate and delay.

Replacement algorithms determine how to utilize the limited storage space as there will be contention from much more objects. In our simulation study, we have 40000 nodes in the wireless mesh network. Each node can store up to 25 objects, as we assume the nodes are not powerful. The number of unique objects $N = 10000$ and their popularity follows a Zipf-like distribution. Initially, there is only one replica of each object randomly placed in the entire system. The objects are replicated when requests originate, and eventually replacement takes place. After a long run of the simulator, the system reaches a steady state. All results reported in this section are obtained when the system is in such a steady state. Due to the time complexity of the simulation system, we have not been able to simulate much larger networks with much more objects.

Three replacement algorithms are considered. The first is FIFO to achieve the effect of the proportional replication strategy. Let us call it *proportional/FIFO*. The second is our proposed algorithm whose cost function is the distance between the nearest replica and the requesting node. As we described in Section 5, it tries to replicate the i-th object such that the density d_i of its replicas is proportional to $p_i^{2/3}$, in a two-dimensional mesh network. Let us call it *minimum-access/GreedyDual*. The third algorithm is similar to the second one except that its cost function is the square of the distance. It tries to replicate an object such that the density of its replicas is proportional to the square root of p_i. Let us call it *minimum-query/GreedyDual*. We still use access cost as the performance metric to compare the algorithms.

6.2 Results and Discussions

Figure 5 shows the performance of the algorithms when α varies. Here, the access cost is the average distance divided by the diagonal length of the square area. Notice that this figure shows results that are similar to those in Figure 3 when α is small. We observe that with either minimum-query/GreedyDual or minimum-access/GreedyDual replacement, the access cost is much lower than that with proportional/FIFO replacement. The difference is more obvious when α is larger. Unfortunately, we have not been able to set even larger α since the number of objects and the number of nodes is limited by the time complexity of the simulation. This is also the reason why we cannot show a more obvious difference between the two optimal strategies.

We have also sampled the access cost of the objects with different popularity values when $\alpha = 1.0$. We first group objects with comparable popularity together, say, with difference less than 1%. Then we compute the average access cost of the objects in the same group, and plot the cost as a function of the object popularity rank in Figure 6. This figure shows a trade-off between more popular and less popular objects. With proportional/FIFO replacement, there is a lower cost to access few most popular objects, but a much higher cost for many more less popular objects. However, the average cost is predominantly contributed by less popular objects, due to the heavy-tailed nature of Zipf-like distribution (*i.e.*, there are much more less popular objects). Notice that both x-axis and y-axis are in logarithmic scale. On the contrary, minimum-

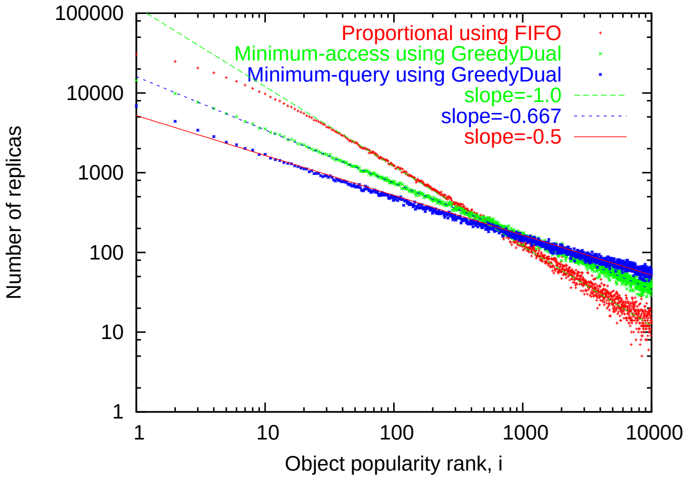

Figure 7: Number of replicas of objects with different access probabilities in steady-state, proportional strategy/FIFO, minimum-query strategy/GreedyDual, and minimum-access strategy/GreedyDual.

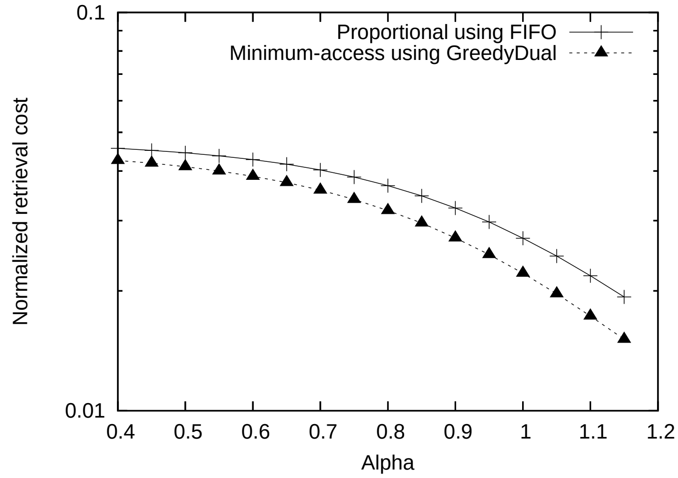

Figure 8: Performance of the proportional strategy/FIFO and the minimum-access strategy/GreedyDual when α varies and geographical locality exists.

access/GreedyDual and minimum-query/GreedyDual obtain a good balance, and therefore, better overall performance.

To verify that these replacement algorithms indeed have their expected replica density functions. We obtained a snapshot of the simulator in steady-state. In Figure 7, we plot the number of replicas for each object at the time, as a function of its popularity rank. With minimum-query/GreedyDual replacement, the figure shows that the number of replicas is proportional to $i^{-0.5}$. With proportional/FIFO replacement, the number of replicas fits $1/i$ well. With minimum-access/GreedyDual, the line is between those of the others. We have found the number of replicas fits $i^{-0.667}$ very well.

It is worth noting that, in Figure 7 the line levels off for few most popular objects (when the value of i is small). We found this is more obvious when we use larger values for α. This again can be explained by the limited scale of our simulation. We have only 40000 nodes in our simulation and the number of replicas for popular objects cannot be larger than 40000. This also explains back in Figure 6, why the access cost line levels off for proportional/FIFO replacement, and further back in Figure 5, why the access cost for proportional/FIFO is slightly lower than expected when α is large.

Finally, as we discussed in Section 5, the proposed replacement algorithm can handle geographical locality of access. To support this, we conducted an additional simulation. Here the requests for each object are disproportionally distributed in the two halves of the geographical area. For each object, one random half of the area is hit by 90% of the requests and the other half is hit by 10%. We have repeated the simulation runs to produce Figure 5, and show the new results in Figure 8. We do not include minimum-query/GreedyDual as it is close to minimum-access/GreedyDual. The results here are similar to those in Figure 5. It suggests that with geographical locality, the relative performance of the algorithms does not change much. A more careful examination of the results also reveals the follows. With geographical locality, the access cost of both algorithms decreases slightly. Potentially it is because in some local areas, object popularity are more skewed. All the algorithms exploit such skewness (locality) to improve the performance.

To summarize, these above simulation results demonstrate that the online replacement algorithms achieve the goals of approximating the optimal strategies.

7. RELATED WORK

Replication strategies have been well studied in the context of Internet content distribution and Web caching system. For example, Qiu, Padmanabhan, and Voelker [13] developed several server replica placement algorithms that use client access latency and request rates to make informed placement decisions. Kangasharju *el al.* [8] studied how to optimally replicate objects in content delivery network (CDN) servers, such that when clients fetch objects from the nearest cache with the requested object, the average number of autonomous systems traversed is minimized. The problem of determining optimal cache placement in a general network is related to the facility location problem and the K-median problem. It has been proved NP-hard [14, 1, 10]. The optimal replication strategies considered in this paper may be extended to the Internet. However, the problem is more difficult due to the complexity of the Internet topology.

Recently, there were several studies on content and service placement in ad hoc wireless networks (and sensor networks). Bhattacharya *et al.* [2] presented a distributed service for data placement, aiming at power conservation. Their strategy is to minimize data access and update traffic. They also described an asynchronous multicast mechanism to propagate updates from sensors. They show a little bookkeeping and local processing is required. Hara [6, 7] considered the optimal placement of data replicas around the network that achieves high data accessibility in the event of network partitioning. Nuggehalli *el al.* [12] studied caching strategies that optimally trade-off between energy consumption and access latency. They designed a polynomial time algorithm which can be applied arbitrary network topology and can be implemented in a distributed manner. Yin and Cao [15] designed cooperative caching techniques to efficiently support data access in ad hoc networks. They proposed that intermediate nodes can cache data to anticipate future requests. Recently, Ko and Rubenstein [9] considered the placement of replicated resources as a graph coloring problem, and presented a distributed and self-stabilizing asynchronous protocol. They found the protocol is close to optimal in convergence time and message overhead. In wireless mesh networks, the requirements on replication strategies are different as they are less constrained by energy consumption and node mobility. Nevertheless, the replication strategies studied in this paper can also be applied to wireless ad hoc networks. One difference is that wireless ad hoc networks are often mobile. This makes the optimal replication problem more difficult.

A closely related work has been done by Cohen and Shenkar, see reference [5] and also [11] for more details. They revealed that to minimize query cost in unstructured peer-to-peer networks, the optimal replication strategy is to replicate objects in proportional to the square root of their popularity. Therefore our work complements their work in the context of unstructured peer-to-peer networks. However, this paper is novel in several aspects. First, this paper has derived the optimal strategy to minimize access cost. This general result is not limited to unstructured networks. Indeed, this general result can be used in any networks (including wireless mesh networks) that use replication techniques to improve performance. Second, this paper considers mainly wireless mesh networks with low dimensionality (in particular 2-D mesh networks). On the other hand, unstructured peer-to-peer networks exhibit more randomness and have high dimensionality. We have shown that in 2-D mesh networks, the optimal replication strategy is much different from the replication strategies in other networks. Third, this paper has revealed that local replacement algorithms attain the optimal strategies without incur any communication overhead. Localized algorithm is important for large wireless mesh networks.

8. CONCLUSION

This paper has focused on content and service replication strategies in multi-hop wireless mesh networks. We considered the problem of determining the optimal numbers of replicas for a set of objects with distinct access probabilities. The objective is to minimize object access cost in large decentralized and unstructured 2-D mesh networks. Different replication strategies are evaluated in networks. The paper shows that the optimal *minimum-access* strategy departs significantly from the commonly used proportional replication strategy, and it has huge performance gains. Furthermore, the paper shows that they can be implemented using a local replacement algorithm modified from the GreedyDual approach.

We conclude that content and service replication strategies should consider the unique characteristics of wireless mesh networks. Modeling such networks helps us explore and develop novel replication techniques that are much different from those in other networks, for example the Internet. Our future work includes exploring replication strategies to minimize other cost functions, and implementing the replication strategies in a distributed fashion, especially in the context of cooperative caches in wireless mesh networks. Finally, given the general applicability of the minimum-access strategy, we are considering to exploit it in other networked systems, *e.g.*, Internet content distribution and grid networks.

9. REFERENCES

[1] I. D. Baev and R. Rajaraman. Approximation algorithms for data placement in arbitrary networks. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2001.

[2] S. Bhattacharya, H. Kim, S. Prabh, and T. Abdelzaher. Energy-conserving data placement and asynchronous multicast in wireless sensor networks. In *Proceedings of the International Conference on Mobile Systems, Applications, and Services (MobiSys)*, May 2003.

[3] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker. Web caching and Zipf-like distributions: Evidence and implications. In *Proceedings of IEEE INFOCOM*, April 1999.

[4] P. Cao and S. Irani. Cost-aware WWW proxy caching algorithms. In *Proceedings of USENIX Symposium on Internet Technologies and Systems (USITS)*, December 1997.

[5] E. Cohen and S. Shenker. Replication strategies in unstructured peer-to-peer networks. In *Proceedings of ACM SIGCOMM*, pages 177–190, August 2002.

[6] T. Hara. Effective replica allocation in ad hoc networks for improving data accessibility. In *Proceedings of IEEE INFOCOM*, April 2001.

[7] T. Hara. Replica allocation methods in ad hoc networks with data update. *ACM-Kluwer Journal on Mobile Networks and Applications (MONET)*, 8(4):343–354, 2003.

[8] K. Kangasharju, J. Roberts, and K. Ross. Object replication strategies in content distribution networks. In *Proceedings of Web Caching Workshop*, June 2001.

[9] B.-J. Ko and D. Rubenstein. A distributed, self-stabilizing protocol for placement of replicated resources in emerging networks. In *Proceedings of IEEE ICNP*, November 2003.

[10] P. Krishnan, D. Raz, and Y. Shavitt. The cache location problem. *IEEE/ACM Transactions on Networking*, 8(5):568–582, October 2000.

[11] Q. Lv, P. Cao, E. Cohen, K. Li, and S. Shenker. Search and replication in unstructured peer-to-peer networks. In *Proceedings of International Conference on Supercomputing*, November 2002.

[12] P. Nuggehalli, V. Srinivasan, and C.-F. Chiasserini. Energy-efficient caching strategies in ad hoc wireless networks. In *Proceedings of ACM MobiHoc*, June 2003.

[13] L. Qiu, V. N. Padmanabhan, and G. M. Voelker. On the placement of Web server replicas. In *Proceedings of IEEE INFOCOM*, April 2001.

[14] D. B. Shmoys, Éva Tardos, and K. Aardal. Approximation algorithms for facility location problems (extended abstract). In *Proceedings of ACM symposium on Theory of computing (STOC)*, May 1997.

[15] L. Yin and G. Cao. Supporting cooperative caching in ad hoc networks. In *Proceedings of IEEE INFOCOM*, March 2004.

[16] N. E. Young. On-line caching as cache size varies. In *Proceedings of ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1991.

APPENDIX

Nonlinear dynamics system (13) is stable

In this appendix we use the perturbation method to study the behavior of the system under the perturbation off the equilibrium point $\overline{d}_i = (\beta p_i/\gamma)^{k/(k+1)}$. Assumed at time t there is a small perturbation δ, then from Equation (13),

$$\dot{d}_i(t) = -\gamma(\overline{d}_i + \delta) + \beta p_i(\overline{d}_i + \delta)^{-1/k}. \tag{14}$$

Applying Taylor expansion, we have

$$\begin{aligned} \dot{d}_i(t) &\approx -\gamma(\overline{d}_i + \delta) + \beta p_i \overline{d}_i^{-1/k}(1 - \frac{\delta}{k\overline{d}_i}) \\ &= -(\gamma + \frac{\gamma}{k})\delta. \end{aligned} \tag{15}$$

Therefore, $d_i(t) + \dot{d}_i(t) = \overline{d}_i + (1 - \gamma - \frac{\gamma}{k})\delta$. For any k, $-1 < (1 - \gamma - \frac{\gamma}{k}) < 1$. Clearly this perturbation decays exponentially fast, and the system is stable.