Selfish Caching in Distributed Systems: A Game-Theoretic Analysis

Byung-Gon Chun[*]
bgchun@cs.berkeley.edu

Kamalika Chaudhuri[†]
kamalika@cs.berkeley.edu

Hoeteck Wee[‡]
hoeteck@cs.berkeley.edu

Marco Barreno[§]
barreno@cs.berkeley.edu

Christos H. Papadimitriou[†]
christos@cs.berkeley.edu

John Kubiatowicz[*]
kubitron@cs.berkeley.edu

Computer Science Division
University of California, Berkeley

ABSTRACT

We analyze replication of resources by server nodes that act selfishly, using a game-theoretic approach. We refer to this as the *selfish caching problem*. In our model, nodes incur either cost for replicating resources or cost for access to a remote replica. We show the existence of pure strategy Nash equilibria and investigate the price of anarchy, which is the relative cost of the lack of coordination. The price of anarchy can be high due to undersupply problems, but with certain network topologies it has better bounds. With a payment scheme the game can always implement the social optimum in the best case by giving servers incentive to replicate.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Algorithms, Economics, Theory, Performance

Keywords

Caching, Distributed Systems, Peer-to-Peer Systems, Game-theoretic Models, Nash Equilibria, Price of Anarchy

1. INTRODUCTION

Wide-area peer-to-peer file systems [2,5,22,32,33], peer-to-peer caches [15,16], and web caches [6,10] have become popular over the last few years. Caching[1] of files in selected servers is widely used to enhance the performance, availability, and reliability of these systems. However, most such systems assume that servers cooperate with one another by following protocols optimized for overall system performance, regardless of the costs incurred by each server.

In reality, servers may behave selfishly — seeking to maximize their own benefit. For example, parties in different administrative domains utilize their local resources (servers) to better support clients in their own domains. They have obvious incentives to cache objects[2] that maximize the benefit in their domains, possibly at the expense of globally optimum behavior. It has been an open question whether these caching scenarios and protocols maintain their desirable global properties (low total social cost, for example) in the face of selfish behavior.

In this paper, we take a game-theoretic approach to analyzing the problem of caching in networks of selfish servers through theoretical analysis and simulations. We model selfish caching as a non-cooperative game. In the *basic model*, the servers have two possible actions for each object. If a replica of a requested object is located at a nearby node, the server may be better off accessing the remote replica. On the other hand, if all replicas are located too far away, the server is better off caching the object itself. Decisions about caching the replicas locally are arrived at locally, taking into account only local costs. We also define a more elaborate *payment model*, in which each server bids for having an object replicated at another site. Each site now has the option of replicating an object and collecting the related bids. Once all servers have chosen a strategy, each game specifies a *configuration*, that is, the set of servers that replicate the object, and the corresponding costs for all servers.

Game theory predicts that such a situation will end up in a *Nash equilibrium*, that is, a set of (possibly randomized) strategies with the property that no player can benefit by changing its strategy while the other players keep their strategies unchanged [28]. Foundational considerations notwithstanding, it is not easy to accept randomized strategies as the behavior of rational agents in a distributed system (see [28] for an extensive discussion) — but this is what classical game theory can guarantee. In certain very fortunate situations, however (see [9]), the existence of *pure* (that is, deterministic) Nash equilibria can be predicted.

With or without randomization, however, the lack of coordination inherent in selfish decision-making may incur costs well beyond what would be globally optimum. This loss of efficiency is

[*]Research supported by NSF career award ANI-9985250, NFS ITR award CCR-0085899, and California MICRO award 00-049.

[†]Research supported by NSF ITR grant 0121555.

[‡]Research supported by NSF grant CCR 9984703 and a US-Israel Binational Science Foundation grant.

[§]Research supported by NSF grant EIA-0122599.

[1]We will use "caching" and "replication" interchangeably.

[2]We use the term "object" as an abstract entity that represents files and other data objects.

quantified by the *price of anarchy* [21]. The price of anarchy is the ratio of the social (total) cost of the worst possible Nash equilibrium to the cost of the social optimum. The price of anarchy bounds the worst possible behavior of a selfish system, when left completely on its own. However, in reality there are ways whereby the system can be guided, through "seeding" or incentives, to a preselected Nash equilibrium. This "optimistic" version of the price of anarchy [3] is captured by the smallest ratio between a Nash equilibrium and the social optimum.

In this paper we address the following questions :

- Do pure strategy Nash equilibria exist in the caching game?
- If pure strategy Nash equilibria do exist, how efficient are they (in terms of the price of anarchy, or its optimistic counterpart) under different placement costs, network topologies, and demand distributions?
- What is the effect of adopting payments? Will the Nash equilibria be improved?

We show that pure strategy Nash equilibria always exist in the caching game. The price of anarchy of the basic game model can be $O(n)$, where n is the number of servers; the intuitive reason is undersupply. Under certain topologies, the price of anarchy does have tighter bounds. For complete graphs and stars, it is $O(1)$. For D-dimensional grids, it is $O(n^{\frac{D}{D+1}})$. Even the optimistic price of anarchy can be $O(n)$. In the payment model, however, the game can always implement a Nash equilibrium that is same as the social optimum, so the optimistic price of anarchy is one.

Our simulation results show several interesting phases. As the placement cost increases from zero, the price of anarchy increases. When the placement cost first exceeds the maximum distance between servers, the price of anarchy is at its highest due to undersupply problems. As the placement cost further increases, the price of anarchy decreases, and the effect of replica misplacement dominates the price of anarchy.

The rest of the paper is organized as follows. In Section 2 we discuss related work. Section 3 discusses details of the basic game and analyzes the bounds of the price of anarchy. In Section 4 we discuss the payment game and analyze its price of anarchy. In Section 5 we describe our simulation methodology and study the properties of Nash equilibria observed. We discuss extensions of the game and directions for future work in Section 6.

2. RELATED WORK

There has been considerable research on wide-area peer-to-peer file systems such as OceanStore [22], CFS [5], PAST [32], FARSITE [2], and Pangaea [33], web caches such as NetCache [6] and SummaryCache [10], and peer-to-peer caches such as Squirrel [16]. Most of these systems use caching for performance, availability, and reliability. The caching protocols assume obedience to the protocol and ignore participants' incentives. Our work starts from the assumption that servers are selfish and quantifies the cost of the lack of coordination when servers behave selfishly.

The placement of replicas in the caching problem is the most important issue. There is much work on the placement of web replicas, instrumentation servers, and replicated resources. All protocols assume obedience and ignore participants' incentives. In [14], Gribble et al. discuss the data placement problem in peer-to-peer systems. Ko and Rubenstein propose a self-stabilizing, distributed graph coloring algorithm for the replicated resource placement [20]. Chen, Katz, and Kubiatowicz propose a dynamic replica placement algorithm exploiting underlying distributed hash tables [4]. Douceur and Wattenhofer describe a hill-climbing algorithm to exchange replicas for reliability in FARSITE [8]. RaDar is a system that replicates and migrates objects for an Internet hosting service [31]. Tang and Chanson propose a coordinated en-route web caching that caches objects along the routing path [34]. Centralized algorithms for the placement of objects, web proxies, mirrors, and instrumentation servers in the Internet have been studied extensively [18, 19, 23, 30].

The facility location problem has been widely studied as a centralized optimization problem in theoretical computer science and operations research [27]. Since the problem is NP-hard, approximation algorithms based on primal-dual techniques, greedy algorithms, and local search have been explored [17, 24, 26]. Our caching game is different from all of these in that the optimization process is performed among distributed selfish servers.

There is little research in non-cooperative facility location games, as far as we know. Vetta [35] considers a class of problems where the social utility is submodular (submodularity means decreasing marginal utility). In the case of competitive facility location among corporations he proves that any Nash equilibrium gives an expected social utility within a factor of 2 of optimal plus an additive term that depends on the facility opening cost. Their results are not directly applicable to our problem, however, because we consider each server to be tied to a particular location, while in their model an agent is able to open facilities in multiple locations. Note that in that paper the increase of the price of anarchy comes from oversupply problems due to the fact that competing corporations can open facilities at the same location. On the other hand, the significant problems in our game are undersupply and misplacement.

In a recent paper, Goemans et al. analyze content distribution on ad-hoc wireless networks using a game-theoretic approach [12]. As in our work, they provide monetary incentives to mobile users for caching data items, and provide tight bounds on the price of anarchy and speed of convergence to (approximate) Nash equilibria. However, their results are incomparable to ours because their payoff functions neglect network latencies between users, they consider multiple data items (markets), and each node has a limited budget to cache items.

Cost sharing in the facility location problem has been studied using *cooperative* game theory [7, 13, 29]. Goemans and Skutella show strong connections between fair cost allocations and linear programming relaxations for facility location problems [13]. Pál and Tardos develop a method for cost-sharing that is approximately budget-balanced and group strategyproof and show that the method recovers 1/3 of the total cost for the facility location game [29]. Devanur, Mihail, and Vazirani give a strategyproof cost allocation for the facility location problem, but cannot achieve group strategyproofness [7].

3. BASIC GAME

The caching problem we study is to find a configuration that meets certain objectives (e.g., minimum total cost). Figure 1 shows examples of caching among four servers. In network (a), A stores an object. Suppose B wants to access the object. If it is cheaper to access the remote replica than to cache it, B accesses the remote replica as shown in network (b). In network (c), C wants to access the object. If C is far from A, C caches the object instead of accessing the object from A. It is possible that in an optimal configuration it would be better to place replicas in A and B. Understanding the placement of replicas by selfish servers is the focus of our study.

The caching problem is abstracted as follows. There is a set N of n servers and a set M of m objects. The distance between servers can be represented as a distance matrix D (i.e., d_{ij} is the distance

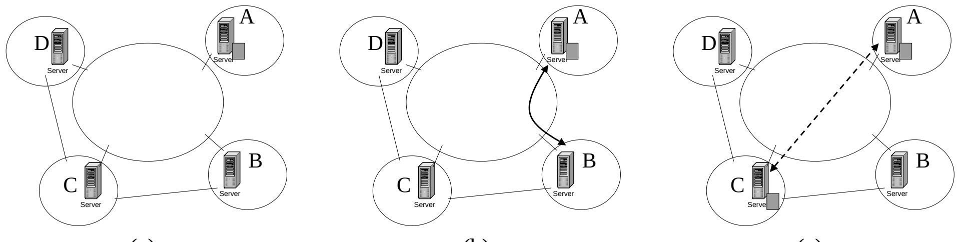


Figure 1: Caching. There are four servers labeled A, B, C, and D. The rectangles are object replicas. In (a), A stores an object. If B incurs less cost accessing A's replica than it would caching the object itself, it accesses the object from A as in (b). If the distance cost is too high, the server caches the object itself, as C does in (c). This figure is an example of our caching game model.

from server i to server j). D models an underlying network topology. For our analysis we assume that the distances are symmetric and the triangle inequality holds on the distances (for all servers i, j, k: $d_{ij} + d_{jk} \geq d_{ik}$). Each server has demand from clients that is represented by a demand matrix W (i.e., w_{ij} is the demand of server i for object j). When a server caches objects, the server incurs some placement cost that is represented by a matrix α (i.e., α_{ij} is a placement cost of server i for object j).

In this study, we assume that servers have no capacity limit. As we discuss in the next section, this fact means that the caching behavior with respect to each object can be examined separately. Consequently, we can talk about *configurations* of the system with respect to a given object:

DEFINITION 1. *A configuration X for some object O is the set of servers replicating this object.*

The goal of the basic game is to find configurations that are achieved when servers optimize their cost functions locally.

3.1 Game Model

We take a game-theoretic approach to analyzing the uncapacitated caching problem among networked selfish servers. We model the selfish caching problem as a non-cooperative game with n players (servers/nodes) whose strategies are sets of objects to cache. In the game, each server chooses a pure strategy that minimizes its cost. Our focus is to investigate the resulting configuration, which is the Nash equilibrium of the game. It should be emphasized that we consider only pure strategy Nash equilibria in this paper.

The cost model is an important part of the game. Let A_i be the set of feasible strategies for server i, and let $S_i \in A_i$ be the strategy chosen by server i. Given a strategy profile $S = (S_1, S_2, ..., S_n)$, the cost incurred by server i is defined as:

$$C_i(S) = \sum_{j \in S_i} \alpha_{ij} + \sum_{j \notin S_i} w_{ij} d_{i\ell(i,j)}. \tag{1}$$

where α_{ij} is the placement cost of object j, w_{ij} is the demand that server i has for object j, $\ell(i,j)$ is the closest server to i that caches object j, and d_{ik} is the distance between i and k. When no server caches the object, we define distance cost $d_{i\ell(i,j)}$ to be d_M—large enough that at least one server will choose to cache the object.

The placement cost can be further divided into first-time installation cost and maintenance cost:

$$\alpha_{ij} = k_{1i} + k_{2i} \frac{UpdateSize_j}{ObjectSize_j} \frac{1}{T} P_j \sum_k w_{kj}, \tag{2}$$

where k_{1i} is the installation cost, k_{2i} is the relative weight between the maintenance cost and the installation cost, P_j is the ratio of the number of writes over the number of reads and writes, $UpdateSize_j$ is the size of an update, $ObjectSize_j$ is the size of the object, and T is the update period. We see tradeoffs between different parameters in this equation. For example, placing replicas becomes more expensive as $UpdateSize_j$ increases, P_j increases, or T decreases. However, note that by varying α_{ij} itself we can capture the full range of behaviors in the game. For our analysis, we use only α_{ij}.

Since there is no capacity limit on servers, we can look at each single object as a separate game and combine the pure strategy equilibria of these games to obtain a pure strategy equilibrium of the multi-object game. Fabrikant, Papadimitriou, and Talwar discuss this existence argument: if two games are known to have pure equilibria, and their cost functions are cross-monotonic, then their union is also guaranteed to have pure Nash equilibria, by a continuity argument [9]. A Nash equilibrium for the multi-object game is the cross product of Nash equilibria for single-object games. Therefore, we can focus on the single object game in the rest of this paper.

For single object selfish caching, each server i has two strategies — to cache or not to cache. The object under consideration is j. We define S_i to be 1 when server i caches j and 0 otherwise. The cost incurred by server i is

$$C_i(S) = \alpha_{ij} S_i + w_{ij} d_{i\ell(i,j)} (1 - S_i). \tag{3}$$

We refer to this game as the *basic game*. The extent to which $C_i(S)$ represents actual cost incurred by server i is beyond the scope of this paper; we will assume that an appropriate cost function of the form of Equation 3 can be defined.

3.2 Nash Equilibrium Solutions

In principle, we can start with a random configuration and let this configuration evolve as each server alters its strategy and attempts to minimize its cost. Game theory is interested in stable solutions called *Nash equilibria*. A pure strategy Nash equilibrium is reached when no server can benefit by unilaterally changing its strategy. A Nash equilibrium[3] (S_i^*, S_{-i}^*) for the basic game specifies a configuration X such that $\forall i \in N, i \in X \Leftrightarrow S_i^* = 1$. Thus, we can consider a set $\mathcal{E}$ of all pure strategy Nash equilibrium configurations:

$$\begin{aligned} X \in \mathcal{E} \quad \Leftrightarrow \quad & \forall i \in N, \\ & \forall S_i \in A_i,\ C_i(S_i^*, S_{-i}^*) \leq C_i(S_i, S_{-i}^*) \end{aligned} \tag{4}$$

By this definition, no server has incentive to deviate in the configurations since it cannot reduce its cost.

For the basic game, we can easily see that:

$$\begin{aligned} X \in \mathcal{E} \quad \Leftrightarrow \quad & \forall i \in N, \quad \exists j \in X \quad s.t.\ d_{ji} \leq \alpha \\ and \quad & \forall j \in X, \quad \neg \exists k \in X \quad s.t.\ d_{kj} < \alpha \end{aligned} \tag{5}$$

The first condition guarantees that there is a server that places the replica within distance α of each server i. If the replica is not placed

[3]The notation for strategy profile (S_i^*, S_{-i}^*) separates node $i's$ strategy (S_i^*) from the strategies of other nodes (S_{-i}^*).

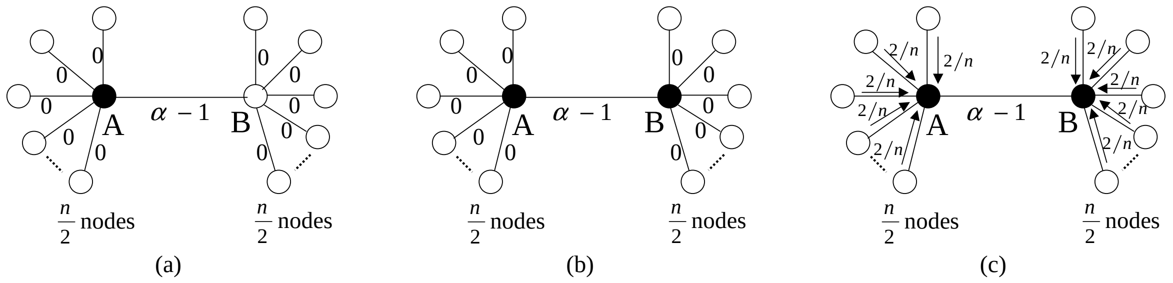


Figure 2: Potential inefficiency of Nash equilibria illustrated by two clusters of $\frac{n}{2}$ servers. The intra-cluster distances are all zero and the distance between clusters is $\alpha - 1$, where α is the placement cost. The dark nodes replicate the object. Network (a) shows a Nash equilibrium in the basic game, where one server in a cluster caches the object. Network (b) shows the social optimum where two replicas, one for each cluster, are placed. The price of anarchy is $O(n)$ and even the optimistic price of anarchy is $O(n)$. This high price of anarchy comes from the undersupply of replicas due to the selfish nature of servers. Network (c) shows a Nash equilibrium in the payment game, where two replicas, one for each cluster, are placed. Each light node in each cluster pays $2/n$ to the dark node, and the dark node replicates the object. Here, the optimistic price of anarchy is one.

at i, then it is placed at another server within distance α of i, so i has no incentive to cache. If the replica is placed at i, then the second condition ensures there is no incentive to drop the replica because no two servers separated by distance less than α both place replicas.

3.3 Social Optimum

The *social cost* of a given strategy profile is defined as the total cost incurred by all servers, namely:

$$C(S) = \sum_{i=0}^{n-1} C_i(S) \tag{6}$$

where $C_i(S)$ is the cost incurred by server i given by Equation 1.

The social optimum cost, referred to as $C(S_O)$ for the remainder of the paper, is the minimum social cost. The social optimum cost will serve as an important base case against which to measure the cost of selfish caching. We define $C(S_O)$ as:

$$C(S_O) = \min_S C(S) \tag{7}$$

where S varies over all possible strategy profiles. Note that in the basic game, this means varying configuration X over all possible configurations. In some sense, $C(S_O)$ represents the best possible caching behavior — if only nodes could be convinced to cooperate with one another.

The social optimum configuration is a solution of a mini-sum facility location problem, which is NP-hard [11]. To find such configurations, we formulate an integer programming problem:

$$\begin{array}{ll} \text{minimize} \sum_i \sum_j \left[\alpha_{ij} x_{ij} + \sum_k w_{ij} d_{ik} y_{ijk}\right] & \\ \quad \text{subject to} & \\ \quad\quad \forall i,j \quad \sum_k y_{ijk} = I(w_{ij}) & \\ \quad\quad \forall i,j,k \quad x_{ij} - y_{kji} \geq 0 & \\ \quad\quad \forall i,j \quad x_{ij} \in \{0,1\} & \\ \quad\quad \forall i,j,k \quad y_{ijk} \in \{0,1\} & \end{array} \tag{8}$$

Here, x_{ij} is 1 if server i replicates object j and 0 otherwise; y_{ijk} is 1 if server i accesses object j from server k and 0 otherwise; $I(w)$ returns 1 if w is nonzero and 0 otherwise. The first constraint specifies that if server i has demand for object j, then it must access j from exactly one server. The second constraint ensures that server i replicates object j if any other server accesses j from i.

3.4 Analysis

To analyze the basic game, we first give a proof of the existence of pure strategy Nash equilibria. We discuss the price of anarchy in general and then on specific underlying topologies. In this analysis we use simply α in place of α_{ij}, since we deal with a single object and we assume placement cost is the same for all servers. In addition, when we compute the price of anarchy, we assume that all nodes have the same demand (i.e., $\forall i \in N \; w_{ij} = 1$).

THEOREM 1. *Pure strategy Nash equilibria exist in the basic game.*

PROOF. We show a constructive proof. First, initialize the set V to N. Then, remove all nodes with zero demand from V. Each node x defines β_x, where $\beta_x = \frac{\alpha}{w_{xj}}$. Furthermore, let $Z(y) = \{z : d_{zy} \leq \beta_z, z \in V\}$; $Z(y)$ represents all nodes z for which y lies within β_z from z.

Pick a node $y \in V$ such that $\beta_y \leq \beta_x$ for all $x \in V$. Place a replica at y and then remove y and all $z \in Z(y)$ from V. No such z can have incentive to replicate the object because it can access y's replica at lower (or equal) cost. Iterate this process of placing replicas until V is empty. Because at each iteration y is the remaining node with minimum β, no replica will be placed within distance β_y of any such y by this process. The resulting configuration is a pure-strategy Nash equilibrium of the basic game. □

The Price of Anarchy (POA): To quantify the cost of lack of coordination, we use the price of anarchy [21] and the optimistic price of anarchy [3]. The price of anarchy is the ratio of the social costs of the worst-case Nash equilibrium and the social optimum, and the optimistic price of anarchy is the ratio of the social costs of the best-case Nash equilibrium and the social optimum.

We show general bounds on the price of anarchy. Throughout our discussion, we use $C(S_W)$ to represent the cost of worst case Nash equilibrium, $C(S_O)$ to represent the cost of social optimum, and PoA to represent the price of anarchy, which is $\frac{C(S_W)}{C(S_O)}$.

The worst case Nash equilibrium maximizes the total cost under the constraint that the configuration meets the Nash condition. Formally, we can define $C(S_W)$ as follows.

$$C(S_W) = \max_{X \in \mathcal{E}} (\alpha |X| + \sum_i \min_{j \in X} d_{ij}) \tag{9}$$

where $\min_{j \in X} d_{ij}$ is the distance to the closest replica (including i itself) from node i and X varies through Nash equilibrium configurations.

Bounds on the Price of Anarchy: We show bounds of the price of anarchy varying α. Let $d_{min} = \min_{(i,j) \in N \times N, i \neq j} d_{ij}$ and $d_{max} = \max_{(i,j) \in N \times N} d_{ij}$. We see that if $\alpha \leq d_{min}$, $PoA = 1$

Topology	***PoA***
Complete graph	1
Star	≤ 2
Line	$O(\sqrt{n})$
D-dimensional grid	$O(n^{\frac{D}{D+1}})$

Table 1: PoA in the basic game for specific topologies

trivially, since every server caches the object for both Nash equilibrium and social optimum. When $\alpha > d_{max}$, there is a transition in Nash equilibria: since the placement cost is greater than any distance cost, only one server caches the object and other servers access it remotely. However, the social optimum may still place multiple replicas. Since $\alpha \leq C(S_O) \leq \alpha + \min_{j \in N} \sum_i d_{ij}$ when $\alpha > d_{max}$, we obtain $\frac{\alpha + \max_{j \in N} \sum_i d_{ij}}{\alpha + \min_{j \in N} \sum_i d_{ij}} \leq PoA \leq \frac{\alpha + \max_{j \in N} \sum_i d_{ij}}{\alpha}$. Note that depending on the underlying topology, even the lower bound of PoA can be $O(n)$. Finally, there is a transition when $\alpha > \max_{j \in N} \sum_i d_{ij}$. In this case, $PoA = \frac{\alpha + \max_{j \in N} \sum_i d_{ij}}{\alpha + \min_{j \in N} \sum_i d_{ij}}$ and it is upper bounded by 2.

Figure 2 shows an example of the inefficiency of a Nash equilibrium. In the network there are two clusters of servers whose size is $\frac{n}{2}$. The distance between two clusters is $\alpha - 1$ where α is the placement cost. Figure 2(a) shows a Nash equilibrium where one server in a cluster caches the object. In this case, $C(S_W) = \alpha + (\alpha - 1)\frac{n}{2}$, since all servers in the other cluster accesses the remote replica. However, the social optimum places two replicas, one for each cluster, as shown in Figure 2(b). Therefore, $C(S_O) = 2\alpha$. $PoA = \frac{\alpha + (\alpha-1)\frac{n}{2}}{2\alpha}$, which is $O(n)$. This bad price of anarchy comes from an undersupply of replicas due to the selfish nature of the servers. Note that all Nash equilibria have the same cost; thus even the optimistic price of anarchy is $O(n)$.

In Appendix A, we analyze the price of anarchy with specific underlying topologies and show that PoA can have tighter bounds than $O(n)$ for the complete graph, star, line, and D-dimensional grid. In these topologies, we set the distance between directly connected nodes to one. We describe the case where $\alpha > 1$, since $PoA = 1$ trivially when $\alpha \leq 1$. A summary of the results is shown in Table 1.

4. PAYMENT GAME

In this section, we present an extension to the basic game with payments and analyze the price of anarchy and the optimistic price of anarchy of the game.

4.1 Game Model

The new game, which we refer to as the *payment game*, allows each player to offer a payment to another player to give the latter incentive to replicate the object. The cost of replication is shared among the nodes paying the server that replicates the object.

The strategy for each player i is specified by a triplet $(v_i, b_i, t_i) \in \{N, \mathbb{R}_+, \mathbb{R}_+\}$. v_i specifies the player to whom i makes a bid, $b_i \geq 0$ is the value of the bid, and $t_i \geq 0$ denotes a threshold for payments beyond which i will replicate the object. In addition, we use R_i to denote the total amount of bids received by a node i ($R_i = \sum_{j:v_j=i} b_j$).

A node i replicates the object if and only if $R_i \geq t_i$, that is, the amount of bids it receives is greater than or equal to its threshold. Let I_i denote the corresponding indicator variable, that is, I_i equals 1 if i replicates the object, and 0 otherwise. We make the rule that if a node i makes a bid to another node j and j replicates the object, then i must pay j the amount b_i. If j does not replicate the object, i does not pay j.

Given a strategy profile, the outcome of the game is the set of tuples $\{(I_i, v_i, b_i, R_i)\}$. I_i tells us whether player i replicates the object or not, b_i is the payment player i makes to player v_i, and R_i is the total amount of bids received by player i. To compute the payoffs given the outcome, we must now take into account the payments a node makes, in addition to the placement costs and access costs of the basic game.

By our rules, a server node i pays b_i to node v_i if v_i replicates the object, and receives a payment of R_i if it replicates the object itself. Its net payment is $b_i I_{v_i} - R_i I_i$. The total cost incurred by each node is the sum of its placement cost, access cost, and net payment. It is defined as

$$C_i(S) = \alpha_{ij} I_i + w_{ij} d_{i\ell(i,j)}(1 - I_i) + b_i I_{v_i} - R_i I_i. \qquad (10)$$

The cost of social optimum for the payment game is same as that for the basic game, since the net payments made cancel out.

4.2 Analysis

In analyzing the payment model, we first show that a Nash equilibrium in the basic game is also a Nash equilibrium in the payment game. We then present an important positive result — in the payment game the socially optimal configuration can always be implemented by a Nash equilibrium. We know from the counterexample in Figure 2 that this is not guaranteed in the the basic game. In this analysis we use α to represent α_{ij}.

THEOREM 2. *Any configuration that is a pure strategy Nash equilibrium in the basic game is also a pure strategy Nash equilibrium in the payment game. Therefore, the price of anarchy of the payment game is at least that of the basic game.*

PROOF. Consider any Nash equilibrium configuration in the basic game. For each node i replicating the object, set its threshold t_i to 0; everyone else has threshold α. Also, for all i, $b_i = 0$.

A node that replicates the object does not have incentive to change its strategy: changing the threshold does not decrease its cost, and it would have to pay at least α to access a remote replica or incentivize a nearby node to cache. Therefore it is better off keeping its threshold and bid at 0 and replicating the object.

A node that is not replicating the object can access the object remotely at a cost less than or equal to α. Lowering its threshold does not decrease its cost, since all b_i are zero. The payment necessary for another server to place a replica is at least α.

No player has incentive to deviate, so the current configuration is a Nash equilibrium. □

In fact, Appendix B shows that the PoA of the payment game can be more than that of the basic game in a given topology.

Now let us look at what happens to the example shown in Figure 2 in the best case. Suppose node B's neighbors each decide to pay node B an amount $2/n$. B does not have an incentive to deviate, since accessing the remote replica does not decrease its cost. The same argument holds for A because of symmetry in the graph. Since no one has an incentive to deviate, the configuration is a Nash equilibrium. Its total cost is 2α, the same as in the socially optimal configuration shown in Figure 2(b). Next we prove that indeed the payment game always has a strategy profile that implements the socially optimal configuration as a Nash equilibrium. We first present the following observation, which is used in the proof, about thresholds in the payment game.

OBSERVATION 1. *If node i replicates the object, j is the nearest node to i among the other nodes that replicate the object, and $d_{ij} < \alpha$ in a Nash equilibrium, then i should have a threshold at* *least ($\alpha - d_{ij}$). Otherwise, it cannot collect enough payment to compensate for the cost of replicating the object and is better off accessing the replica at j.*

THEOREM 3. *In the payment game, there is always a pure strategy Nash equilibrium that implements the social optimum configuration. The optimistic price of anarchy in the payment game is therefore always one.*

PROOF. Consider the socially optimal configuration ϕ_{opt}. Let N_o be the set of nodes that replicate the object and $N_c = N - N_o$ be the rest of the nodes. Also, for each i in N_o, let Q_i denote the set of nodes that access the object from i, not including i itself. In the socially optimal configuration, $d_{ij} \leq \alpha$ for all j in Q_i.

We want to find a set of payments and thresholds that makes this configuration implementable. The idea is to look at each node i in N_o and distribute the minimum payment needed to make i replicate the object among the nodes that access the object from i. For each i in N_o, and for each j in Q_i, we define

$$\delta_j = \min\{\alpha, \min_{k \in N_o - \{i\}} d_{jk}\} - d_{ji} \tag{11}$$

Note that δ_j is the difference between j's cost for accessing the replica at i and j's next best option among replicating the object and accessing some replica other than i. It is clear that $\delta_j \geq 0$.

CLAIM 1. *For each $i \in N_o$, let ℓ be the nearest node to i in N_o. Then, $\sum_{j \in Q_i} \delta_j \geq \alpha - d_{i\ell}$.*

PROOF. (of claim) Assume the contrary, that is, $\sum_{j \in Q_i} \delta_j < \alpha - d_{i\ell}$. Consider the new configuration ϕ_{new} wherein i does not replicate and each node in Q_i chooses its next best strategy (either replicating or accessing the replica at some node in $N_o - \{i\}$). In addition, we still place replicas at each node in $N_o - \{i\}$. It is easy to see that cost of ϕ_{opt} minus cost of ϕ_{new} is at least:

$$\begin{aligned} &(\alpha + \sum_{j \in Q_i} d_{ij}) - (d_{i\ell} + \sum_{j \in Q_i} \min\{\alpha, \min_{k \in N_o - \{i\}} d_{ik}\}) \\ = \quad & \alpha - d_{i\ell} - \sum_{j \in Q_i} \delta_j > 0, \end{aligned}$$

which contradicts the optimality of ϕ_{opt}. □

We set bids as follows. For each i in N_o, $b_i = 0$ and for each j in Q_i, j bids to i (i.e., $v_j = i$) the amount:

$$b_j = \max\{0, \delta_j - \epsilon_i/(|Q_i| + 1)\}, \quad j \in Q_i \tag{12}$$

where $\epsilon_i = \sum_{j \in Q_i} \delta_j - \alpha + d_{i\ell} \geq 0$ and $|Q_i|$ is the cardinality of Q_i. For the thresholds, we have:

$$t_i = \begin{cases} \alpha & \text{if } i \in N_c; \\ \sum_{j \in Q_i} b_j & \text{if } i \in N_o. \end{cases} \tag{13}$$

This fully specifies the strategy profile of the nodes, and it is easy to see that the outcome is indeed the socially optimal configuration.

Next, we verify that the strategies stipulated constitute a Nash equilibrium. Having set t_i to α for i in N_c means that any node in N is at least as well off lowering its threshold and replicating as bidding α to some node in N_c to make it replicate, so we may disregard the latter as a profitable strategy. By observation 1, to ensure that each i in N_o does not deviate, we require that if ℓ is the nearest node to i in N_o, then $\sum_{j \in Q_i} b_j$ is at least $(\alpha - d_{i\ell})$. Otherwise, i will raise t_i above $\sum_{j \in Q_i} b_j$ so that it does not replicate and instead accesses the replica at ℓ. We can easily check that

$$\sum_{j \in Q_i} b_j \geq \sum_{j \in Q_i} \delta_j - \frac{|Q_i|\epsilon_i}{|Q_i| + 1} = \alpha - d_{i\ell} + \frac{\epsilon_i}{|Q_i| + 1} \geq \alpha - d_{i\ell}.$$

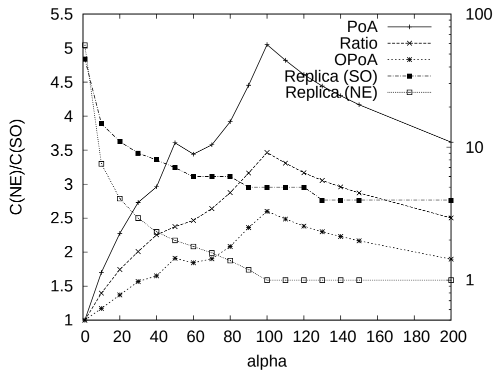


Figure 3: We present PoA, $Ratio$, and $OPoA$ results for the basic game, varying α on a 100-node line topology, and we show number of replicas placed by the Nash equilibria and by the optimal solution. We see large peaks in PoA and $OPoA$ at $\alpha = 100$, where a phase transition causes an abrupt transition in the lines.

Therefore, each node $i \in N_o$ does not have incentive to change t_i since i loses its payments received or there is no change, and i does not have incentive to b_i since it replicates the object. Each node j in N_c has no incentive to change t_j since changing t_j does not reduce its cost. It also does not have incentive to reduce b_j since the node where j accesses does not replicate and j has to replicate the object or to access the next closest replica, which costs at least the same from the definition of b_j. No player has incentive to deviate, so this strategy profile is a Nash equilibrium. □

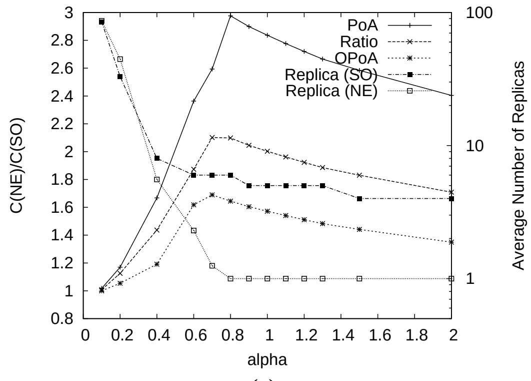
5. SIMULATION

We run simulations to compare Nash equilibria for the single-object caching game with the social optimum computed by solving the integer linear program described in Equation 8 using Mosek [1]. We examine price of anarchy (PoA), optimistic price of anarchy ($OPoA$), and the average ratio of the costs of Nash equilibria and social optima ($Ratio$), and when relevant we also show the average numbers of replicas placed by the Nash equilibrium (*Replica(NE)*) and the social optimum (*Replica(SO)*). The PoA and $OPoA$ are taken from the worst and best Nash equilibria, respectively, that we observe over the runs. Each data point in our figures is based on 1000 runs, randomly varying the initial strategy profile and player order. The details of the simulations including protocols and a discussion of convergence are presented in Appendix C.

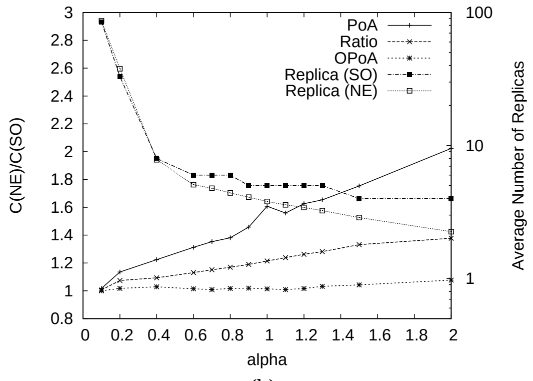
In our evaluation, we study the effects of variation in four categories: placement cost, underlying topology, demand distribution, and payments. As we vary the placement cost α, we directly influence the tradeoff between caching and not caching. In order to get a clear picture of the dependency of PoA on α in a simple case, we first analyze the basic game with a 100-node line topology whose edge distance is one.

We also explore transit-stub topologies generated using the GT-ITM library [36] and power-law topologies (Router-level Barabasi-Albert model) generated using the BRITE topology generator [25]. For these topologies, we generate an underlying physical graph of 3050 physical nodes. Both topologies have similar minimum, average, and maximum physical node distances. The average distance is 0.42. We create an overlay of 100 server nodes and use the same overlay for all experiments with the given topology.

In the game, each server has a demand whose distribution is Bernoulli(p), where p is the probability of having demand for the object; the default unless otherwise specified is $p = 1.0$.

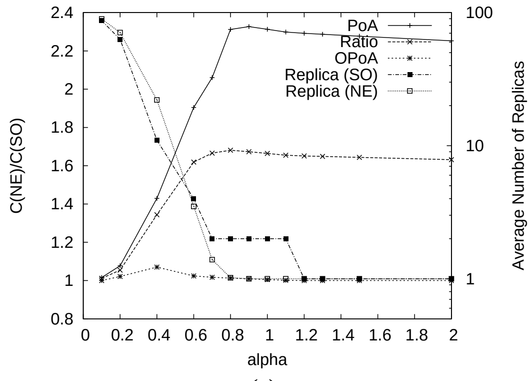


(a)

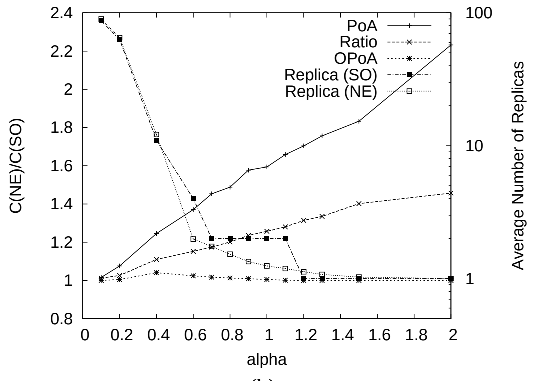


(b)

Figure 4: Transit-stub topology: (a) basic game, (b) payment game. We show the PoA, $Ratio$, $OPoA$, and the number of replicas placed while varying α between 0 and 2 with 100 servers on a 3050-physical-node transit-stub topology.



(a)



(b)

Figure 5: Power-law topology: (a) basic game, (b) payment game. We show the PoA, $Ratio$, $OPoA$, and the number of replicas placed while varying α between 0 and 2 with 100 servers on a 3050-physical-node power-law topology.

5.1 Varying Placement Cost

Figure 3 shows PoA, $OPoA$, and $Ratio$, as well as number of replicas placed, for the line topology as α varies. We observe two phases. As α increases the PoA rises quickly to a peak at 100. After 100, there is a gradual decline. $OPoA$ and $Ratio$ show behavior similar to PoA.

These behaviors can be explained by examining the number of replicas placed by Nash equilibria and by optimal solutions. We see that when α is above one, Nash equilibrium solutions place fewer replicas than optimal on average. For example, when α is 100, the social optimum places four replicas, but the Nash equilibrium places only one. The peak in PoA at $\alpha = 100$ occurs at the point for a 100-node line where the worst-case cost of accessing a remote replica is slightly less than the cost of placing a new replica, so selfish servers will never place a second replica. The optimal solution, however, places multiple replicas to decrease the high global cost of access. As α continues to increase, the undersupply problem lessens as the optimal solution places fewer replicas.

5.2 Different Underlying Topologies

In Figure 4(a) we examine an overlay graph on the more realistic transit-stub topology. The trends for the PoA, $OPoA$, and $Ratio$ are similar to the results for the line topology, with a peak in PoA at α = 0.8 due to maximal undersupply.

In Figure 5(a) we examine an overlay graph on the power-law topology. We observe several interesting differences between the power-law and transit-stub results. First, the PoA peaks at a lower level in the power-law graph, around 2.3 (at $\alpha = 0.9$) while the peak PoA in the transit-stub topology is almost 3.0 (at $\alpha = 0.8$). After the peak, PoA and $Ratio$ decrease more slowly as α increases. $OPoA$ is close to one for the whole range of α values. This can be explained by the observation in Figure 5(a) that there is no significant undersupply problem here like there was in the transit-stub graph. Indeed the high PoA is due mostly to misplacement problems when α is from 0.7 to 2.0, since there is little decrease in PoA when the number of replicas in social optimum changes from two to one. The $OPoA$ is equal to one in the figure when the same number of replicas are placed.

5.3 Varying Demand Distribution

Now we examine the effects of varying the demand distribution. The set of servers with demand is random for $p < 1$, so we calculate the expected PoA by averaging over 5 trials (each data point is based on 5000 runs). We run simulations for demand levels of $p \in \{0.2, 0.6, 1.0\}$ as α is varied on the 100 servers on top of the transit-stub graph. We observe that as demand falls, so does expected PoA. As p decreases, the number of replicas placed in the social optimum decreases, but the number in Nash equilibria changes little. Furthermore, when α exceeds the overlay diameter, the number in Nash equilibria stays constant when p varies. Therefore, lower p leads to a lesser undersupply problem, agreeing with intuition. We do not present the graph due to space limitations and redundancy; the PoA for p = 1.0 is identical to PoA in Figure 4(a), and the lines for p = 0.6 and p = 0.2 are similar but lower and flatter.

5.4 Effects of Payment

Finally, we discuss the effects of payments on the efficiency of Nash equilibria. The results are presented in Figure 4(b) and Figure 5(b). As shown in the analysis, the simulations achieve $OPoA$ close to one (it is not exactly one because of randomness in the simulations). The $Ratio$ for the payment game is much lower than the $Ratio$ for the basic game, since the protocol for the payment game tends to explore good regions in the space of Nash equilibria. We observe in Figure 4 that for $\alpha \geq 0.4$, the average number of replicas of Nash equilibria gets closer with payments to that of the social optimum than it does without. We observe in Figure 5 that more replicas are placed with payments than without when α is between 0.7 and 1.3, the only range of significant undersupply in the power-law case. The results confirm that payments give servers incentive to replicate the object and this leads to better equilibria.

6. DISCUSSION AND FUTURE WORK

We suggest several interesting extensions and directions. One extension is to consider multiple objects in the capacitated caching game, in which servers have capacity limits when placing objects. Since caching one object affects the ability to cache another, there is no separability of a multi-object game into multiple single object games. As studied in [12], one way to formulate this problem is to find the best response of a server by solving a knapsack problem and to compute Nash equilibria.

In our analyses, we assume that all nodes have the same demand. However, nodes could have different demand depending on objects. We intend to examine the effects of heterogeneous demands (or heterogeneous placement costs) analytically. We also want to look at the following “aggregation effect”. Suppose there are $n - 1$ clustered nodes with distance of $\alpha - 1$ from a node hosting a replica. All nodes have demands of one. In that case, the price of anarchy is $O(n)$. However, if we aggregate $n - 1$ nodes into one node with demand $n - 1$, the price of anarchy becomes $O(1)$, since α should be greater than $(n - 1)(\alpha - 1)$ to replicate only one object. Such aggregation can reduce the inefficiency of Nash equilibria.

We intend to compute the bounds of the price of anarchy under different underlying topologies such as random graphs or growth-restricted metrics. We want to investigate whether there are certain distance constraints that guarantee $O(1)$ price of anarchy. In addition, we want to run large-scale simulations to observe the change in the price of anarchy as the network size increases.

Another extension is to consider server congestion. Suppose the distance is the network distance plus $\gamma \times (number\ of\ accesses)$ where γ is an extra delay when an additional server accesses the replica. Then, when $\alpha > \gamma$, it can be shown that PoA is bounded by $\frac{\alpha}{\gamma}$. As γ increases, the price of anarchy bound decreases, since the load of accesses is balanced across servers.

While exploring the caching problem, we made several observations that seem counterintuitive. First, the PoA in the payment game can be worse than the PoA in the basic game. Another observation we made was that the number of replicas in a Nash equilibrium can be more than the number of replicas in the social optimum even without payments. For example, a graph with diameter slightly more than α may have a Nash equilibrium configuration with two replicas at the two ends. However, the social optimum may place one replica at the center. We leave the investigation of more examples as an open issue.

7. CONCLUSIONS

In this work we introduce a novel non-cooperative game model to characterize the caching problem among selfish servers without any central coordination. We show that pure strategy Nash equilibria exist in the game and that the price of anarchy can be $O(n)$ in general, where n is the number of servers, due to undersupply problems. With specific topologies, we show that the price of anarchy can have tighter bounds. More importantly, with payments, servers are incentivized to replicate and the optimistic price of anarchy is always one. Non-cooperative caching is a more realistic model than cooperative caching in the competitive Internet, hence this work is an important step toward viable federated caching systems.

8. ACKNOWLEDGMENTS

We thank Kunal Talwar for enlightening discussions regarding this work.

9. REFERENCES

[1] http://www.mosek.com.

[2] A. Adya et al. FARSITE: Federated, Available, and Reliable Storage for an Incompletely Trusted Environment. In *Proc. of USENIX OSDI*, 2002.

[3] E. Anshelevich, A. Dasgupta, E. Tardos, and T. Wexler. Near-optimal Network Design with Selfish Agents. In *Proc. of ACM STOC*, 2003.

[4] Y. Chen, R. H. Katz, and J. D. Kubiatowicz. SCAN: A Dynamic, Scalable, and Efficient Content Distribution Network. In *Proc. of Intl. Conf. on Pervasive Computing*, 2002.

[5] F. Dabek et al. Wide-area Cooperative Storage with CFS. In *Proc. of ACM SOSP*, Oct. 2001.

[6] P. B. Danzig. NetCache Architecture and Deploment. In *Computer Networks and ISDN Systems*, 1998.

[7] N. Devanur, M. Mihail, and V. Vazirani. Strategyproof cost-sharing Mechanisms for Set Cover and Facility Location Games. In *Proc. of ACM EC*, 2003.

[8] J. R. Douceur and R. P. Wattenhofer. Large-Scale Simulation of Replica Placement Algorithms for a Serverless Distributed File System. In *Proc. of MASCOTS*, 2001.

[9] A. Fabrikant, C. H. Papadimitriou, and K. Talwar. The Complexity of Pure Nash Equilibria. In *Proc. of ACM STOC*, 2004.

[10] L. Fan, P. Cao, J. Almeida, and A. Z. Broder. Summary Cache: A Scalable Wide-area Web Cache Sharing Protocol. *IEEE/ACM Trans. on Networking*, 8(3):281–293, 2000.

[11] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Co., 1979.

[12] M. X. Goemans, L. Li, V. S. Mirrokni, and M. Thottan. Market Sharing Games Applied to Content Distribution in Ad-Hoc Networks. In *Proc. of ACM MOBIHOC*, 2004.

[13] M. X. Goemans and M. Skutella. Cooperative Facility Location Games. In *Proc. of ACM-SIAM SODA*, 2000.

[14] S. Gribble et al. What Can Databases Do for Peer-to-Peer? In *WebDB Workshop on Databases and the Web*, June 2001.

[15] K. P. Gummadi et al. Measurement, Modeling, and Analysis of a Peer-to-Peer File-Sharing Workload. In *Proc. of ACM SOSP*, October 2003.

[16] S. Iyer, A. Rowstron, and P. Druschel. Squirrel: A Decentralized Peer-to-Peer Web Cache. In *Proc. of ACM PODC*, 2002.

[17] K. Jain and V. V. Vazirani. Primal-Dual Approximation Algorithms for Metric Facility Location and k-Median Problems. In *Proc. of IEEE FOCS*, 1999.

[18] S. Jamin et al. On the Placement of Internet Instrumentation. In *Proc. of IEEE INFOCOM*, pages 295–304, 2000.

[19] S. Jamin et al. Constrained Mirror Placement on the Internet. In *Proc. of IEEE INFOCOM*, pages 31–40, 2001.

[20] B.-J. Ko and D. Rubenstein. A Distributed, Self-stabilizing Protocol for Placement of Replicated Resources in Emerging Networks. In *Proc. of IEEE ICNP*, 2003.

[21] E. Koutsoupias and C. Papadimitriou. Worst-Case Equilibria. In *STACS*, 1999.

[22] J. Kubiatowicz et al. OceanStore: An Architecture for Global-scale Persistent Storage. In *Proc. of ACM ASPLOS*. ACM, November 2000.

[23] B. Li, M. J. Golin, G. F. Italiano, and X. Deng. On the Optimal Placement of Web Proxies in the Internet. In *Proc. of IEEE INFOCOM*, 1999.

[24] M. Mahdian, Y. Ye, and J. Zhang. Improved Approximation Algorithms for Metric Facility Location Problems. In *Proc. of Intl. Workshop on Approximation Algorithms for Combinatorial Optimization Problems*, 2002.

[25] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: Universal Topology Generation from a User's Perspective. Technical Report 2001-003, 1 2001.

[26] R. R. Mettu and C. G. Plaxton. The Online Median Problem. In *Proc. of IEEE FOCS*, 2000.

[27] P. B. Mirchandani and R. L. Francis. *Discrete Location Theory*. Wiley-Interscience Series in Discrete Mathematics and Optimization, 1990.

[28] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

[29] M. Pal and E. Tardos. Group Strategyproof Mechanisms via Primal-Dual Algorithms. In *Proc. of IEEE FOCS*, 2003.

[30] L. Qiu, V. N. Padmanabhan, and G. M. Voelker. On the Placement of Web Server Replicas. In *Proc. of IEEE INFOCOM*, 2001.

[31] M. Rabinovich, I. Rabinovich, R. Rajaraman, and A. Aggarwal. A Dynamic Object Replication and Migration Protocol for an Internet Hosting Service. In *Proc. of IEEE ICDCS*, 1999.

[32] A. Rowstron and P. Druschel. Storage Management and Caching in PAST, A Large-scale, Persistent Peer-to-peer Storage Utility. In *Proc. of ACM SOSP*, October 2001.

[33] Y. Saito, C. Karamanolis, M. Karlsson, and M. Mahalingam. Taming Aggressive Replication in the Pangaea Wide-Area File System. In *Proc. of USENIX OSDI*, 2002.

[34] X. Tang and S. T. Chanson. Coordinated En-route Web Caching. In *IEEE Trans. Computers*, 2002.

[35] A. Vetta. Nash Equilibria in Competitive Societies, with Applications to Facility Location, Traffic Routing, and Auctions. In *Proc. of IEEE FOCS*, 2002.

[36] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee. How to Model an Internetwork. In *Proc. of IEEE INFOCOM*, 1996.

APPENDIX

A. ANALYZING SPECIFIC TOPOLOGIES

We now analyze the price of anarchy (PoA) for the basic game with specific underlying topologies and show that PoA can have better bounds. We look at complete graph, star, line, and D-dimensional grid. In all these topologies, we set the distance between two directly connected nodes to one. We describe the case where $\alpha > 1$, since $PoA = 1$ trivially when $\alpha \leq 1$.

For a complete graph, $PoA = 1$, and for a star, $PoA \leq 2$. For a complete graph, when $\alpha > 1$, both Nash equilibria and social optima place one replica at one server, so $PoA = 1$. For star, when $1 < \alpha < 2$, the worst case Nash equilibrium places replicas at all leaf nodes. However, the social optimum places one replica at the center node. Therefore, $PoA = \frac{(n-1)\alpha+1}{\alpha+(n-1)} \leq \frac{2(n-1)+1}{1+(n-1)} \leq 2$. When $\alpha > 2$, the worst case Nash equilibrium places one replica at a leaf node and the other nodes access the remote replica, and the social optimum places one replica at the center. $PoA = \frac{\alpha+1+2(n-2)}{\alpha+(n-1)} = 1 + \frac{n}{\alpha+(n-1)} \leq 2$.

For a line, the price of anarchy is $O(\sqrt{n})$. When $1 < \alpha < n$, the worst case Nash equilibrium places replicas every 2α so that there is no overlap between areas covered by two adjacent servers that cache the object. The social optimum places replicas at least every $\sqrt{2\alpha}$. The placement of replicas for the social optimum is as follows. Suppose there are two replicas separated by distance d. By placing an additional replica in the middle, we want to have the reduction of distance to be at least α. The distance reduction is $d/2 + 2\{((d/2-1)-1) + ((d/2-2)-2) + ... + ((d/2-d/4)-d/4)\} \geq d^2/8$. d should be at most $2\sqrt{2\alpha}$. Therefore, the distance between replicas in the social optimum is at most $\sqrt{2\alpha}$. $C(S_W) = \alpha\frac{(n-1)}{2\alpha} + \frac{\alpha(\alpha+1)}{2}\frac{(n-1)}{2\alpha} = \Theta(\alpha n)$. $C(S_O) \geq \alpha\frac{n-1}{\sqrt{2\alpha}} + 2\frac{\sqrt{2\alpha}/2(\sqrt{2\alpha}/2+1)}{2}\frac{n-1}{\sqrt{2\alpha}}$. $C(S_O) = \Omega(\sqrt{\alpha}n)$. Therefore, $PoA = O(\sqrt{\alpha})$. When $\alpha > n-1$, the worst case Nash equilibrium places one replica at a leaf node and $C(S_W) = \alpha + \frac{(n-1)n}{2}$. However, the social optimum still places replicas every $\sqrt{2\alpha}$. If we view PoA as a continuous function of α and compute a derivative of PoA, the derivative becomes 0 when α is $\Theta(n^2)$, which means the function decreases as α increases from n. Therefore, PoA is maximum when α is n, and $PoA = \frac{\Theta(n^2)}{\Omega(\sqrt{n}n)} = O(\sqrt{n})$. When $\alpha > \frac{(n-1)n}{2}$, the social optimum also places only one replica, and PoA is trivially bounded by 2. This result holds for the ring and it can be generalized to the D-dimensional grid. As the dimension in the grid increases, the distance reduction of additional replica placement becomes $\Omega(d^{D+1})$ where d is the distance between two adjacent replicas. Therefore, $PoA = \frac{\Theta(n^2)}{\Omega(n^{\frac{1}{D+1}}n)} = O(n^{\frac{D}{D+1}})$.

B. PAYMENT CAN DO WORSE

Consider the network in Figure 6 where $\alpha > 1 + \alpha/3$. Any Nash equilibrium in the basic game model would have exactly two replicas - one in the left cluster, and one in the right. It is easy to verify that the worst placement (in terms of social cost) of two replicas occurs when they are placed at nodes A and B. This placement can be achieved as a Nash equilibrium in the payment game, but not in the basic game since A and B are a distance $3\alpha/4$ apart.

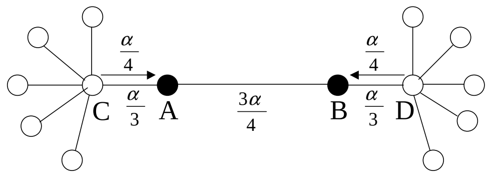


Figure 6: Example where the payment game has a Nash equilibrium which is worse than any Nash equilibrium in the basic game. The unlabeled distances between the nodes in the cluster are all 1. The thresholds of white nodes are all α and the thresholds of dark nodes are all $\alpha/4$. The two dark nodes replicate the object in this payment game Nash equilibrium.

Algorithm 1 Initialization for the Basic Game

```
$L_1$ = a random subset of servers
for each node $i$ in $N$ do
    if $i \in L_1$ then
        $S_i$ = 1 ; replicate the object
    else
        $S_i$ = 0
```

Algorithm 2 Move Selection of i for the Basic Game

```
$Cost_1 = \alpha$
$Cost_2 = \min_{j \in X-\{i\}} d_{ij}$ ; $X$ is the current configuration
$Cost_{min} = \min\{Cost_1, Cost_2\}$
if $Cost_{now} > Cost_{min}$ then
    if $Cost_{min} == Cost_1$ then
        $S_i = 1$
    else
        $S_i = 0$
```

C. NASH DYNAMICS PROTOCOLS

The simulator initializes the game according to the given parameters and a random initial strategy profile and then iterates through rounds. Initially the order of player actions is chosen randomly. In each round, each server performs the Nash dynamics protocol that adjusts its strategies greedily in the chosen order. When a round passes without any server changing its strategy, the simulation ends and a Nash equilibrium is reached.

In the basic game, we pick a random initial subset of servers to replicate the object as shown in Algorithm 1. After the initialization, each player runs the move selection procedure described in Algorithm 2 (in algorithms 2 and 4, $Cost_{now}$ represents the current cost for node i). This procedure chooses greedily between replication and non-replication. It is not hard to see that this Nash dynamics protocol converges in two rounds.

In the payment game, we pick a random initial subset of servers to replicate the object by setting their thresholds to 0. In addition, we initialize a second random subset of servers to replicate the object with payments from other servers. The details are shown in Algorithm 3. After the initialization, each player runs the move selection procedure described in Algorithm 4. This procedure chooses greedily between replication and accessing a remote replica, with the possibilities of receiving and making payments, respectively. In the protocol, each node increases its threshold value by *incr* if it does not replicate the object. By this ramp up procedure, the cost of replicating an object is shared fairly among the nodes that access a replica from a server that does cache. If *incr* is small, cost is shared more fairly, and the game tends to reach equilibria that encourages more servers to store replicas, though the convergence takes longer. If *incr* is large, the protocol converges quickly, but it may miss efficient equilibria. In the simulations we set *incr* to 0.1. Most of our simulation runs converged, but there were a very few cases where the simulation did not converge due to the cycles of dynamics. The protocol does not guarantee convergence within a certain number of rounds like the protocol for the basic game.

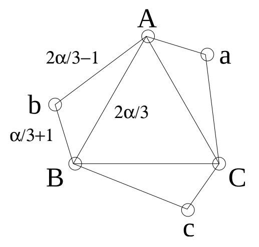


Figure 7: **An example where the Nash dynamics protocol does not converge in the payment game.**

Algorithm 3 Initialization for the Payment Game

```
$L_1$ = a random subset of servers
for each node $i$ in $N$ do
    $b_i$ = 0
    if $i \in L_1$ then
        $t_i$ = 0 ; replicate the object
    else
        $t_i$ = $\alpha$

$L_2 = \{\}$
for each node $i$ in $N$ do
    if coin toss == head then
        $M_i = \{j : d(j,i) < \min_{k \in L_1 \cup L_2} d(j,k)\}$
        if $M_i$ != $\emptyset$ then
            for each node $j \in M_i$ do
                $b_j = max\{\frac{\alpha + \sum_{k \in M_i} d(i,k)}{|M_i|} - d(i,j), 0\}$
            $L_2 = L_2 \cup \{i\}$
```

Algorithm 4 Move Selection of i for the Payment Game

```
$Cost_1 = \alpha - R_i$
$Cost_2 = \min_{j \in N-\{i\}} \{t_j - R_j + d_{ij}\}$
$Cost_{min} = \min\{Cost_1, Cost_2\}$
if $Cost_{now} > Cost_{min}$ then
    if $Cost_{min} == Cost_1$ then
        $t_i = R_i$
    else
        $t_i = R_i + incr$
        $v_i = argmin_j\{t_j - R_j + d_{ij}\}$
        $b_i = t_{v_i} - R_{v_i}$
```

We provide an example graph and an initial condition such that the Nash dynamics protocol does not converge in the payment game if started from this initial condition. The graph is represented by a shortest path metric on the network shown in Figure 7. In the starting configuration, only A replicates the object, and a pays it an amount $\alpha/3$ to do so. The thresholds for A, B and C are $\alpha/3$ each, and the thresholds for a, b and c are $2\alpha/3$. It is not hard to verify that the Nash dynamics protocol will never converge if we start with this condition.

The Nash dynamics protocol for the payment game needs further investigation. The dynamics protocol for the payment game should avoid cycles of actions to achieve stabilization of the protocol. Finding a self-stabilizing dynamics protocol is an interesting problem. In addition, a fixed value of *incr* cannot adapt to changing environments. A small value of *incr* can lead to efficient equilibria, but it can take long time to converge. An important area for future research is looking at adaptively changing *incr*.