Theory of Computing Systems (2020) 64:272–310
https://doi.org/10.1007/s00224-019-09939-7

Cache Me if You Can: Capacitated Selfish Replication Games in Networks

Ragavendran Gopalakrishnan[1] · Dimitrios Kanoulas[2] · Naga Naresh Karuturi[1] · C. Pandu Rangan[1] · Rajmohan Rajaraman[3] · Ravi Sundaram[3]

Published online: 29 July 2019

Abstract

In Peer-to-Peer (P2P) network systems, content (object) delivery between nodes is often required. One way to study such a distributed system is by defining games, which involve selfish nodes that make strategic choices on replicating content in their local limited memory (cache) or accessing content from other nodes for a cost. These Selfish Replication games have been introduced in Chun et al. (2004) for nodes that

Gopalakrishnan and Karuturi were partially supported by a generous gift from Northeastern University alumnus Madhav Anand. This work was also partially supported by NSF grants CCF-0635119 and CNS-0915985. A preliminary version of this work appeared in *LATIN* 2012 [29].

✉ Dimitrios Kanoulas
Dimitrios.Kanoulas@iit.it

Ragavendran Gopalakrishnan
ragav@cse.iitm.ac.in

Naga Naresh Karuturi
nnaresh@cse.iitm.ac.in

C. Pandu Rangan
prangan@cse.iitm.ac.in

Rajmohan Rajaraman
rraj@ccs.neu.edu

Ravi Sundaram
koods@ccs.neu.edu

[1] Department of Computer Science and Engineering Chennai, Indian Institute of Technology Madras, Play Field Avenue, Indian Institute Of Technology, Chennai, TN 600036, India

[2] Humanoids and Human-Centered Mechatronics Lab, Istituto Italiano di Tecnologia (IIT), Via Morego, 30, Genoa, 16163, Italy

[3] School of Computer and Information Science, Northeastern University, 440 Huntington Avenue 240 West Village H, Boston, MA 02115, USA

do not have any capacity limits, leaving the capacitated problem, i.e. Capacitated Selfish Replication (CSR) games, open. In this work, we first form the model of the CSR games, for which we perform a Nash equilibria analysis. In particular, we focus on hierarchical networks, given their extensive use to model communication costs of content delivery in P2P systems. We present an exact polynomial-time algorithm for any hierarchical network, under two constraints on the utility functions: 1) “Nearer is better”, i.e. the closest the content is to the node the less its access cost is, and 2) “Independence of irrelevant alternatives”, i.e. aggregation of individual node preferences. This generalization represents a vast class of utilities and more interestingly allows each of the nodes to have simultaneously completely different functional forms of utility functions. In this general framework, we present CSR games results on arbitrary networks and outline the boundary between intractability and effective computability in terms of the network structure, object preferences, and the total number of objects. Moreover, we prove that the problem of equilibria existence becomes NP-hard for general CSR games. By adding some constraints in the number of objects and their preferences, we show that the equilibrium can be found in polynomial time. Finally, we introduce the fractional version of CSR games (F-CSR) to represent content distribution. We prove that equilibrium exists for every F-CSR game, but it is PPAD-complete.

Keywords Game theory · Caching · Distributed networks · Hierarchical networks · Preference orders and choice theory · Computational complexity

1 Introduction

Consider a P2P network for sharing movies (objects) among multiple users (nodes). Due to limited disk space, the movies can be stored either locally or obtained from other users in some cost. The storing decisions affect everyone that uses this service. A natural question is to predict the movie collection stability in your friends network (i.e. equilibrium) and your satisfaction from them (i.e. access cost), when users act selfishly. Similarly, in the new wireless 4G/5G services, users will not only consume different apps, but will also provide apps to their network through personal communications and computing devices. In such a network, the question is whether storing apps will lead to a situation of endless churn or could there be an equilibrium?

Content delivery and caching in P2P networks can be studied in a game-theoretic framework. In this work, we study Capacitated Selfish Replication (CSR) games as an abstraction of the above network scenarios. In CSR games the strategic agents, or players, are nodes in a network that act selfishly. The nodes have some object preferences and bounded storage space, i.e. caches, to store a limited number of content copies. Each node in cooperation with other nodes can serve access requests for the objects that are stored in its cache. However, the set of objects which a node chooses to store in its cache is from one side solely based on its own utility function (notice that this does not prevent the players to use the same utility function for the whole network) and from the other side based on where objects of interest have been stored in the network. Thus, each node in the model, has two potential actions for an object.

Either store a replica of the object in its limited cache, or access with some cost the object replica from a remote node.

Chun et al. [8] first introduced such a game-theoretic framework to analyse pure Nash equilibria in networks without cache capacities, but with some storage cost. They left the capacitated version of the problem open. The main interest of the CSR games in more recent works is on *hierarchical networks* that have been extensively used to model communication content delivery costs in P2P networks [25]. Ultrametric models for content delivery networks [38] and cooperative caching in hierarchical networks [41, 42, 48, 68] are just some examples. The best results on CSR games for hierarchical networks [46, 57] are about the existence of a Nash equilibrium for a generalized one-level hierarchical network, using the sum utility function for which each node is based on a weighted sum of the cost of accessing the objects.

1.1 Our Results

In this paper, we first introduce the basic model of the Capacitated Selfish Replication (CSR) games in Section 2. This includes the definition of the nodes (players) and objects, the formulation of the cost functions for a node accessing objects in the network, the object replication strategies among nodes, as well as the basic formulation of the network. The main focus is on the study of Nash equilibria existence and computability for a set of CSR games variants. In particular, we introduce a polynomial-time Nash equilibrium method for hierarchical networks, given their extensive use to model communication costs of content delivery in P2P systems. We address the following three problems, including their computational complexity:

- Does pure Nash equilibrium exist in a CSR game, for hierarchical networks?
- Does pure Nash equilibrium exist in a CSR game, for general undirected networks, setting specific restrictions on the number of objects and the utility/cost functions?
- Does pure Nash equilibrium exist, when the objects can be split in fractions, i.e. F-CSR games?

Note that in all the games, we assume that all the pieces of content, i.e. objects, have the same size, as considered in prior works [2, 8, 46, 57]. Otherwise, the problem becomes NP-hard even for computing the best response of a player (node) as a generalization of the well-known knapsack problem.

In Section 3 we present our main algorithm, which extends and resolves the open problem that was defined in [45, 47, 57]. In particular, it has been proved [46, 57] that CSR games for hierarchical networks have a Nash equilibrium in the case of a generalized 1-level hierarchy, when the utility function is a function of the costs sum of accessing replicated objects in the network. We introduce an exact polynomial-time algorithm for Nash Equilibrium computation in any hierarchical network. We use a novel technique which we name "fictional players[1]" method. We show that using this method we can extend to a general framework of natural preference orders

[1] not to be confused with "fictitious play" [24] which involves learning

that are entirely arbitrary, but follow two natural constraints: "Nearer is better", i.e. the closest the content is to the node the less its access cost is and "Independence of irrelevant alternatives", i.e. the aggregation of individual node preferences. This generalization represents a vast class of utility functions and more interestingly allows each of the nodes to have simultaneously completely different functional forms of utility functions. The method introduces and iteratively eliminates fictional players in a controlled fashion, maintaining a Nash equilibrium at each step. In the end, the desired equilibrium for the entire network is realized without any fictional players left in the network. Even though the analysis is specified in the context of the sum utility function to elucidate the technique of fictional players, we then abstract the central requirements for our proof technique. In particular, we develop a general framework of CSR games with ordinal preferences, for which a larger class of utility functions could be used as extension to the above result.

In Section 4, we present the general CSR games framework in terms of the utility preference relations and node preference orders. In particular, we consider the utility that is not just each node's specific numeric assignment for each objects placement, but a preference order each node has on object placements that satisfies two natural constraints: Monotonicity (or "Nearer is better") and Consistency (or "Independence of irrelevant alternatives"). In this way the method is extended to a vast class of utility functions, while nodes may simultaneously have utility functions of completely different functional forms.

After extending our hierarchical networks results to the broader class of utilities, in Sections 5 and 6 we study general CSR games that have various network structures (directed or undirected), forms of object preferences (binary or general). Intractability and effective computability of equilibria is delineated in terms of the network structure, object preferences, and the total number of objects. The results are summarized in Table 1. Most notable are the following results:

- equilibria existence for general undirected networks with two objects, using the technique of fictional players
- equilibria existence for general undirected networks when object preferences are binary

Table 1 Equilibria existence and computability in CSR games

Object preferences and count	Undirected networks	Directed networks
Binary, two objects	Yes, in P (5.3)	No, in P (6.2)
Binary, three or more objects	Yes, in PLS (5.2)	No, NP-complete (6.1)
General, two objects	Yes, in P (5.3)	No, NP-complete (6.1)
General, three or more objects	No, NP-complete (6.1) Hierarchical: Yes, in P (5.1)	No, NP-complete (6.1)

Each cell (other than in the first row/column) first indicates whether equilibria always exist for a particular CSR games sub-class. If so, the cell next indicates the complexity of determining an equilibrium; otherwise, it indicates the complexity of determining whether equilibria exist. The relevant subsection appears in parentheses

- the problem of equilibria existence becomes NP-hard for general CSR games
- equivalence of finding equilibria in polynomial time for CSR games in strongly connected networks with two objects and binary object preferences, via a reduction to the well-studied even-cycle problem [61].

Finally in Section 7, we introduce the fractional version of CSR games (F-CSR) to represent content distribution using erasure codes. In this framework, each node is allowed to store fractions of objects and can satisfy an object access request by retrieving any set of object fractions as long as these fractions sum to at least one. We present a natural implementation of this framework via erasure codes (e.g. using the Digital Fountain approach [5, 66]). We prove that F-CSR games always have equilibria and finding it is in PPAD. However, we also show finding equilibria is PPAD-hard even for a sum-of-distances utility function.

1.2 Related Work

Peer-to-Peer (P2P) networks have been used to model systems for sharing content and resources among the individual peers (such as the file systems [9, 44, 63, 64], web caches [10, 22], or P2P caches [33]). Even though P2P networks have been extensively studied from a theoretical point of view, there are several open problems when rational peers have diverse and selfish interests [23].

One of the most interesting problems is *caching*, i.e. holding copies of content in clients and servers. Several research studies have considered data storing [7, 30], self-stabilization [40], dynamic replication [13, 60, 67], and exchanging of content copies in a centralized manner [35, 36, 49, 58]. Research on capacitated caching has been also considerable as an optimization problem and various centralized and distributed algorithms have been presented for different networks in [2, 4, 41, 48, 71]. For instance, centralized optimization for the facility location problem has been studied in [62], including several approximations [34, 50, 52]. These frameworks usually ignore the fact that peers may make free choices that minimizes their content access cost, by not following usual instrumentation.

The caching problem that we study is in the intersection of game theory and computer science, that has been extensively studied the last decade [53, 69]. In [56] Papadimitriou laid the groundwork for algorithmic game theory by introducing syntactically defined sub-classes of FNP with complete problems, PPAD being a notable such subclass. Non-cooperative facility location games have attracted some small attention over the last decades. For instance, in [70], the problem of Nash equilibrium for games that allowed players build nodes in remote locations, whereas in our case nodes hold fixed spaces for storing objects/content. In [27], content distribution was studied, providing bounds on the approximated Nash equilibrium with respect to the price of anarchy and the convergence speed. The difference in the game design lies in the fact that each node had cost limits for storing objects without considering network latencies. The uncapacitated case of selfish caching games was introduced in [8], in which nodes could store more pieces of content by paying for the additional storage.

We focus on the capacitated version which was left open by [8], believing that limits on cache-capacity model an important real-world restriction. Special cases of the integral CSR games version have been studied. In [46], Nash equilibria were shown to exist in cases that nodes are equidistant from one another and a special centralized server holds all objects. In [57] the model is slightly extended to the case where special servers for different objects are at different distances. Our results generalize and completely subsume all these prior cases of CSR games. Market sharing games [28] also consider caches with capacity, but differ to cc games since they are special cases of congestion games. In this work we focus primarily on equilibria and our general framework of CSR games with ordinal preferences aligns more with the theory of social choice [3]; in this sense, we deviate from prior work [12, 21] that is focused on the price of anarchy [43].

Our work on CSR games in [29] has initiated various research lines and has been extended recently in different directions. For instance, in [31, 32] the selfish replication problem is studied for the case that nodes seek object placements with cache cooperation, and includes an experimental analysis. Etesami et al. have extended our model in a series of papers [18]. In [14] the Nash equilibrium algorithm for two resources is shown to converge faster and it is extended to arbitrary cache sizes for a polynomial time computation. This is extended in [15, 16, 19], where a quasi-polynomial algorithm is introduced to drive allocations whose total cost is within a constant factor of that in any pure-strategy Nash equilibrium, in games formed by undirected networks. The price of anarchy for CSR games with binary preferences over general undirected networks has been studied in [17, 20], showing an upper bound of 3. In [55], the caching problem is studied for operator-specific, non-linear, cost functions in games that form arbitrary peering graph topologies, while in [1] CSR games are studied for general undirected networks for which a randomized algorithm is introduced using a random tree search method to search for pure-strategy Nash equilibrium.

In related work, through a major breakthrough [6, 11] it has been proven that 2-player Nash Equilibrium is PPAD-complete. The PPAD-complete term is coming to occupy a role in algorithmic game theory analogous to NP-completeness in combinatorial optimization [26], and thus we study the fractional version of the problem, where nodes can store parts of objects, while accessing the remaining part from other nodes. In this setup we prove PPAD-completeness.

2 A Basic Model for CSR Games

We consider a set V of nodes (labeled 1 through $n = |V|$) to form a network in which they share a collection O of unit-size objects. We let d_{ij} denote i's cost for accessing an object at j, for $i, j \in V$; we refer to d as the access cost function. j is node's i *nearest* node in a set S of nodes, if $j \in S$ and $d_{ij} \leq d_{ik}$ for all $k \in S$. Moreover, a given network is *undirected* if d is symmetric, i.e. if $d_{ij} = d_{ji}$ for all $i, j \in V$. An undirected network is *hierarchical* if the access cost function forms an ultrametric, i.e. if $d_{ik} \leq \max\{d_{ij}, d_{jk}\}$ for all $i, j, k \in V$.

The cache of each node i is able to store a certain number of objects. Node's i placement is simply the set of objects stored at i. The strategy set of a given node is the set of all feasible placements at the node. A *global placement* is any tuple

$(P_i : i \in V)$, where $P_i \subseteq \mathrm{O}$ represents a feasible placement at node i. We are going to use P_{-i} to denote the collection $(P_j : j \in V \setminus \{i\})$, for convenience. We will also often use $P = (P_i, P_{-i})$ to refer to a global placement. Moreover, we also assume that V includes a *server* node that has the capacity to store all the objects. In this way it is ensured that at least one copy of every object is present in the system; this assumption is without loss of generality given that the access cost of every node to the server can be set an arbitrarily large number.

CSR Games Each node in our game-theoretic model, attaches a utility to each global placement. We assume that each node i has a weight $r_i(\alpha)$ for each object α representing the rate at which i accesses α. We define the *sum utility function* $U_s(i)$ as follows: $U_s(i)(P) = -\sum_{\alpha \in \mathrm{O}} r_i(\alpha) \cdot d_{i\sigma_i(P,\alpha)}$, where $\sigma_i(P, \alpha)$ is i's nearest node holding α in P. A CSR game is a tuple $(V, \mathrm{O}, d, \{r_i\})$. This work focuses on *pure Nash equilibria* (henceforth, simply *equilibria*) of the CSR games. Such a CSR game equilibrium instance is a global placement P such that for each $i \in V$ there is no placement Q_i such that $U_s(i)(P) < U_s(i)(Q)$.

Unit Cache Capacity In this work, we assume that all objects are of identical size. Under this assumption, we now argue that it is sufficient to consider the case where each node's cache holds exactly one object. Consider a set V of nodes in which the cache of node i can store c_i objects. Let V' denote a new set of nodes which contains, for each node i in V, new nodes $i_1, i_2, \ldots, i_{c_i}$, i.e., one new node for each unit of the cache capacity of i. We extend the access cost function as follows: $d_{j_\ell i_k} = d_{ji}$ for all $1 \leq \ell \leq c_j$, $1 \leq k \leq c_i$ and $d_{i_\ell i_k} = 0$ for all $1 \leq \ell < k \leq c_i$, for each node $i \in V$.

We consider an obvious onto mapping f from placements in V' to those in V. Given placement P' for V', we set $f(P') = P$ where $P_i = \cup_{1 \leq k \leq c_i} P'_{i_k}$. This mapping ensures that $U_s(i)(P') = U_s(i)(P)$, giving us the desired reduction. Thus, in the remainder of the paper, we assume that every node in the network stores at most one object in its cache.

3 Hierarchical Networks

In this section, we present a polynomial-time equilibria construction for CSR games on hierarchical networks. We can represent any hierarchical network by a tree T in such a way that the node set V is the set of its leaves. Every internal node v has a label $\ell(v)$ such that:

1. if v is an ancestor[2] of w in T, then $\ell(v) \geq \ell(w)$
2. for any $i, j \in V$, d_{ij} is given by $\ell(\mathrm{lca}(i, j))$, where $\mathrm{lca}(i, j)$ is the least common ancestor of nodes i and j [38, 41].

Figure 1 illustrates a simple example for a hierarchical network tree with two internal nodes and three leaf nodes, with the corresponding label relationships, the least common ancestors, and the access costs.

[2] We let each node be both descendant and ancestor of itself.

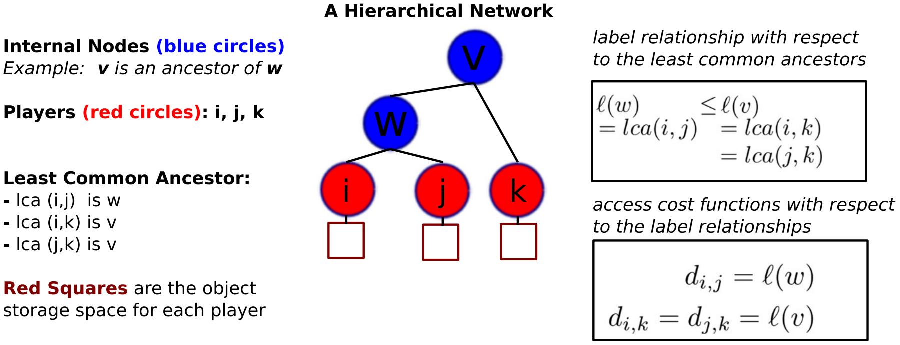
A Hierarchical Network
Internal Nodes (blue circles)
Example: v is an ancestor of w
Players (red circles): i, j, k
Least Common Ancestor:
- lca (i,j) is w
- lca (i,k) is v
- lca (j,k) is v
Red Squares are the object storage space for each player
V
W
i
j
k
label relationship with respect to the least common ancestors
$\ell(w) = lca(i,j) \leq \ell(v) = lca(i,k) = lca(j,k)$
access cost functions with respect to the label relationships
$d_{i,j} = \ell(w)$
$d_{i,k} = d_{j,k} = \ell(v)$

Fig. 1 A simple example of a hierarchical network tree with two internal nodes ($\ell(v)$ and $\ell(w)$) and three leaf nodes (i, j, and k). The label relationships, the least common ancestors, and the access costs are described

Fictional Players The proposed algorithm requires the introduction of the *fictional player* notion. A *fictional α-player* for an object α will be a new node which stores α in any equilibrium. In particular, for any fictional α-player ℓ, $r_\ell(\alpha)$ is 1 and $r_\ell(\beta)$ is 0 for any $\beta \neq \alpha$. In a particular hierarchy each fictional player is introduced as a leaf; our method determines the exact locations in the hierarchy. The access cost function for each fictional player is naturally extended using the hierarchy and the labels of the internal nodes. We let "node" denote both the elements of V and fictional players.

A preference relation The object weights for each node i in a hierarchical network induce a natural preorder $\sqsupset_i$ among elements of $\mathrm{O} \times A_i$, where A_i is the set of proper ancestors of i in T. In particular, we define $(\alpha, v) \sqsupset_i (\beta, w)$ whenever $r_i(\alpha) \cdot \ell(v) > r_i(\beta) \cdot \ell(w)$. In words, in hierarchical networks there is a total preorder in the objects-nodes preferences, which is used during the algorithm to define a potential function, when nodes are playing their best responses. For instance, $(\alpha, v) \sqsupset_i (\beta, w)$ means that if i needs to place either α or β in its cache, and the least common ancestor of i and the most i-preferred node in $V \setminus \{i\}$ holding α (resp., β) is v (resp., w), then i prefers to store α over β. Figure 2 illustrates an example of node i that will prefer to store object α that is stored further than object β and with a higher cost, due to the total preorder.

To express any player's best response in terms of these preference relations, we define $\mu_i(P) = (\alpha, v)$, where $P_i = \{\alpha\}$ and v is lca$(i, \sigma_i(P_{-i}, \alpha))$, where $\sigma_i(P_{-i}, \alpha)$ denotes i's nearest node in the set of nodes holding α in P_{-i}. For instance, in Fig. 2 $\sigma_i(P_{-i}, \alpha)$ is node k (the nearest node holding α), while $P_i = \{\alpha\}$ (node i is holding α) and the thus, these two information can be denoted as $\mu_i(P) = (\alpha, v)$, where v is the least common ancestor between nodes i and k.

Given, the aforementioned definitions, we can now express the best response of a player in terms of the preference relations in the following Lemma. This is needed in Lemma 2 to prove the existence of an equilibrium at each step of the algorithm.

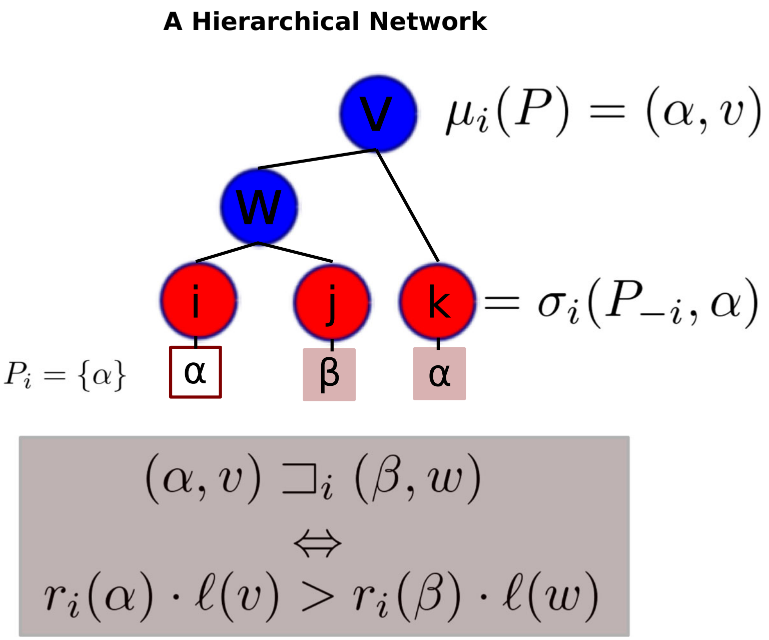
A Hierarchical Network
V
$\mu_i(P) = (\alpha, v)$
W
i
j
k
$= \sigma_i(P_{-i}, \alpha)$
$P_i = \{\alpha\}$
α
β
α
$(\alpha, v) \sqsupset_i (\beta, w)$
$\Leftrightarrow$
$r_i(\alpha) \cdot \ell(v) > r_i(\beta) \cdot \ell(w)$

Fig. 2 A simple example of a hierarchical network tree with two internal nodes ($\ell(v)$ and $\ell(w)$) and three leaf nodes (i, j, and k). The preference relation of node i is presented

Lemma 1 *A best response P_i of a node i for a placement P_{-i} of $V \setminus \{i\}$ is $\{\alpha\}$ where α maximizes $(\gamma, lca(i, \sigma_i(P_{-i}, \gamma)))$, over all objects γ, according to $\sqsupset_i$.*

Proof For a given placement P with $P_i = \{\alpha\}$, $U_s(i)(P)$ equals

$$-\sum_{\gamma \neq \alpha} r_i(\gamma)\ell(\mathrm{lca}(i, \sigma_i(P_{-i}, \gamma))),$$

which can be rewritten as

$$-\left(\sum_{\gamma \in \mathrm{O}} r_i(\gamma)\ell(\mathrm{lca}(i, \sigma_i(P_{-i}, \gamma)))\right) + r_i(\alpha) \cdot \ell(\mathrm{lca}(i, \sigma_i(P_{-i}, \alpha))).$$

Thus, $\{\alpha\}$ is a best response to P_{-i} if and only if α maximizes $r_i(\gamma) \cdot \ell(\mathrm{lca}(i, \sigma_i(P_{-i}, \gamma))$ over all objects γ, while the desired claim follows from the definition of $\sqsupset_i$. □

The Algorithm In the beginning of the algorithm we introduce a set of fictional players, maintaining in the same time the invariant that the current global placement in this hierarchy is an equilibrium. We then proceed by removing existing or adding new fictional players, tweaking in a particular way their set and locations, in such a way that at each step we guarantee an equilibrium. The algorithm terminates when all the fictional players are removed in the desired equilibrium state. Let W_t and P^t

denote the set of fictional players and equilibrium, respectively, at the start of step t of the algorithm.

Initialization We create an initial set W_0 by adding a fictional α-player as a leaf child of v, for each object α and internal node $v \in T$. In the initial equilibrium P^0 for each fictional α-player i we have $P_i^0 = \{\alpha\}$, i.e. each node $i \in V$ plays its best response. By definition, it is clear that each fictional player is in equilibrium. Moreover, for every α, every $i \in V$ has a sibling fictional α-player. Thus, the best response of every $i \in V$ does not depend on the placement of nodes in $V \setminus \{i\}$, which implies that P^0 is an equilibrium.

Algorithm's t step For the node set $V \cup W_t$ (the original nodes and the fictional ones) we fix an equilibrium P^t. If W_t is empty, i.e., no fictional player remained, we are done. Otherwise, we select a fictional node j in W_t. Let $P_j^t = \{\alpha\}$ and $\mu_j(P^t) = (\alpha, v)$, i.e. the fictional player j holds object $\{\alpha\}$ and the closest node that holds object $\{\alpha\}$ is through the internal node v. We let S be the set of all nodes $i \in V$ such that $(\alpha, v) \sqsupset_i \mu_i(P^t)$, i.e. the closest node that holds object $\{\alpha\}$ (except itself) is through the internal node v. We consider two cases for computing a new set of fictional players W_{t+1} and a new global placement P^{t+1} such that P^{t+1} is an equilibrium for $V \cup W_{t+1}$:

S is empty (there is a node holding the object closer than through the internal node v and thus the fictional node j is not affecting the strategy). We remove the j fictional player from W_t and the hierarchy. For the remaining nodes the placement remains as is. In this way $W_{t+1} = W_t - \{j\}$ (the fictional player is removed) and P^{t+1} is the same as P^t (since the fictional player j was not affecting any other node's best response strategy), but P_j^{t+1} is no longer defined, since j is removed.

S is nonempty (some nodes are accessing object α from the fictional player i). We select a node $i \in S$ such that $\mathrm{lca}(i, j)$ is lowest among all nodes in S (in this way no other node is affected from the change in the strategy of i) and we let $P_i^t = \{\beta\}$. We set $P_i^{t+1} = \{\alpha\}$, remove the fictional α-player j from W_t, and add a new fictional β-player ℓ as a leaf sibling of $i \in T$ (in this way the player will be in equilibrium by accessing β from the new fictional player). In this way $P_\ell^{t+1} = \{\beta\}$, while for every other node j we set $P_j^{t+1} = P_j^t$. Finally, we set $W_{t+1} = (W_t \cup \{\ell\}) \setminus \{j\}$, removing from the node set the removed fictional player and adding the new one.

An example of the steps is illustrated in Fig. 3. Next, in Lemma 2 we prove why at every step t, as described above, we have an equilibrium.

Lemma 2 *Consider step t of the algorithm. If P^t is an equilibrium for $V \cup W_t$, then the following statements hold.*

1. *For every node k in $V \cup W_{t+1}$, P_k^{t+1} is a best response to P_{-k}^{t+1}.*
2. *For every node k in $V \cup W_{t+1}$, $\mu_k\left(P^{t+1}\right) \sqsupseteq_k \mu_k(P^t)$.*
3. *We have $|W_{t+1}| \leq |W_t|$. Furthermore, either $|W_{t+1}| < |W_t|$ or there exists a node i in V such that $\mu_i(P^{t+1}) \sqsupset_i \mu_i(P^t)$.*

Proof Let α, v, S, i, and j be defined as described above in step t. We first prove the first two lemma's statements. We let k be any node in $V \cup W_{t+1}$. First, we consider

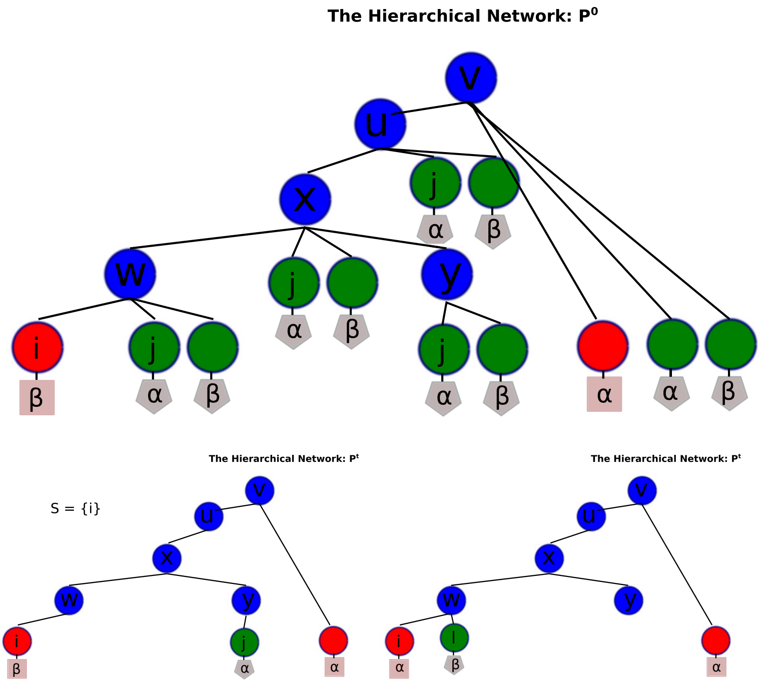
The Hierarchical Network: P⁰
v
u
x
j
w
y
i
α
β
The Hierarchical Network: Pᵗ
S = {i}
The Hierarchical Network: Pᵗ
l

Fig. 3 Illustrating the algorithm for a simple hierarchical network

the case that lca(k, j) is an ancestor of v. In this case k is not in the subtree rooted at the child u of v that contains j. For any object γ, $\sigma_k(P^{t+1}_{-k}, \gamma) = \sigma_k(P^t_{-k}, \gamma)$ and $P^{t+1}_k = P^t_k$. Statement 2 for k is implied thus from the fact that $\mu_k(P^{t+1}) = \mu_k(P^t)$. Since P^t is in equilibrium, statement 1 also holds for k. We then consider the case that lca(k, j) is a proper descendant of v. in this case k is in the subtree rooted at the child u of v that contains j. There are two cases.

In the case that S is empty, the fictional α-player j is removed. In this way j is not in W_{t+1}. Moreover, there is no copy of α in the subtree rooted at u. Given that no other object except α is created or removed, $\sigma_k(P^{t+1}_{-k}, \gamma) = \sigma_k(P^t_{-k}, \gamma)$ for $\gamma \neq \alpha$. The second statement is established for k by the fact that lca$(k, \sigma_k(P^{t+1}_{-k}, \alpha)) = v$ and $\mu_k(P^{t+1}) = \mu_k(P_t)$. Since S is empty, $\mu_k(P^t) \sqsupseteq_k (\alpha, v)$. The first statement for k follows from Lemma 1 and the fact that P^t_k is in equilibrium such that P^{t+1}_k is a best response against P^{t+1}_{-k}.

In the second case that S is not empty, we let i be a node in S such that lca(i, j) is lowest among all nodes in S, as defined above, x denote lca(i, j), and P^t_i be equal to $\{\beta\}$, where $\beta \neq \alpha$. From the algorithm it is true that $P^{t+1}_k = \{\alpha\}$. We let $k \neq i$ be a node in the subtree rooted at u. For any $\gamma \neq \alpha$, $\sigma_k(P^{t+1}_{-k}, \gamma) = \sigma_k(P^{t+1}_{-k}, \gamma)$. The second statement is established for k by the fact that since $P^{t+1}_k = P^t_k \neq \{\alpha\}$, we have $\mu_k\left(P^{t+1}\right) = \mu_k(P^t)$. Similarly for node i, we have $\mu_i\left(P^{t+1}\right) = (\alpha, v) \sqsupset_i \mu_i(P^t)$.

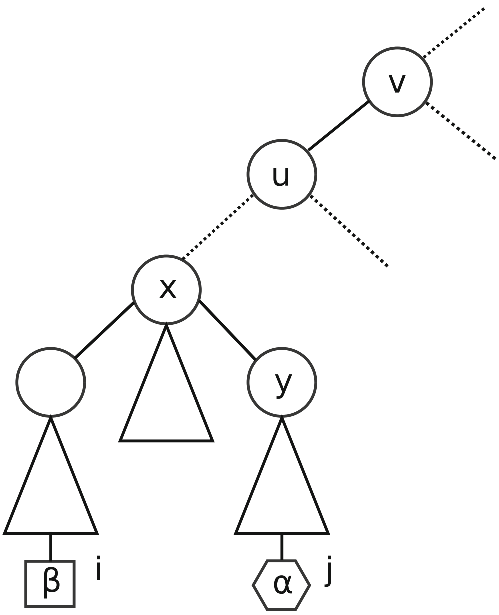

Fig. 4 Illustrating the analysis for hierarchical networks; referred to in the proof of Lemma 2. The square is a node i in V holding object β, and the hexagon is a fictional α-player

To establish the first statement or any node k in the subtree rooted at u we consider two cases. Let y be the child of x that is an ancestor of j (see Fig. 4). In the first case, we let k be in the subtree rooted at y. Then, by our choice of i, it is true that

$$\mu_k\left(P^{t+1}\right) \sqsupseteq_k (\alpha, v) \sqsupseteq_k (\alpha, x) = \left(\alpha, \sigma_k\left(P^{t+1}_{-k}, \alpha\right)\right),$$

which, by Lemma 1, implies that the first statement holds for k. In the second case, we let k be in the subtree rooted at u, but not in the subtree rooted at y. Again, $\sigma_k\left(P^{t+1}_{-k}, \gamma\right) = \sigma_k(P^t_{-k}, \gamma)$ for $\gamma \neq \alpha$. And for α it is true that

$$\left(\alpha, \text{lca}\left(k, \sigma_k\left(P^{t+1}_{-k}, \alpha\right)\right)\right) = (\alpha, \text{lca}(k, i)) \sqsupseteq_k (\alpha, x) \sqsupseteq_k \mu_k(P^t) = \mu_k\left(P^{t+1}\right),$$

which establishes the first statement for k using Lemma 1.

To establish the third statement we use the fact that $|W_{t+1}| \leq |W_t|$, which is immediate from the definition of the algorithm's t step. When S is empty, $|W_{t+1}| < |W_t|$ since a fictional player is deleted. When S is nonempty, we have proved above that $\mu_i\left(P^{t+1}\right) \sqsupset_i \mu_i(P^t)$. This concludes the proof of the third statement and of the whole lemma. □

Theorem 1 *For hierarchical node preferences, an equilibrium can be found in polynomial time.*

Proof From Lemma 2 and the definition of the algorithm it is straightforward that it returns a valid equilibrium at the termination. We should prove now that the termination is achieved in polynomial time. We consider the potential function given by the sum of $|W_t|$ and the sum of the $\mu_i(P^t)$ position in the preorder $\sqsupseteq_i$ over all i. We notice that $|W_0|$ is at most nm, where m is the number of objects and n is $|V|$ which is at least the number of internal nodes. Moreover, the initial potential is at most $nm + n^2m$ since $|\mathrm{O} \times I|$ is at most nm. From Lemma 2, the potential decreases by at least one in each step and thus the number of steps is at most $nm + n^2m$.

We need to also prove that each step can be implemented in polynomial time. In the initialization we add $O(nm)$ fictional players and compute the best response for each node $i \in V$. For the later process, we compare at most m placements for each $k \in V$, i.e. one for each object. During each subsequent step we select a fictional player j, we determine whether the set S is nonempty, and if so we compute node i and the updated placement. From this process we only need to explain the computation of S and i, where S is the set of all k nodes which are not in equilibrium when a fictional player j is deleted. S is computed as follows: for each node $k \in V$, we replace the current object in its cache by α and add k to S. According to the utility, this yields to a more preferable placement. Thus, S can be computed in time polynomial in n. To complete the proof of the theorem, we let node i simply be a node in S such that $\mathrm{lca}(i, j)$ is lowest among all nodes in S. This can also be computed in time polynomial in n. □

4 A General Framework for CSR Games with Ordinal Preferences

In this section, we present a new framework on CSR games with ordinal preferences, to generalize the results that were presented in Section 3 to a broad class of utility functions, and to also enable the study of the existence and complexity of equilibria in more general settings.

Node Preference Relations Among all the nodes in V, we assume that each node $i \in V$ has a total preorder $\geq_i$[3] and $\geq_i$ further satisfies $i \geq_i j$ for all $i, j \in V$. A node i *prefers* j over k if $j \geq_i k$, while a node j is the *most i-preferred* in a set S of nodes if $j \in S$ and $j \geq_i k$ for all $k \in S$. We let $j =_i k$ denote that $j \geq_i k$ and $k \geq_i j$, while when it is not the case that $k \geq_i j$, we denote it by $j >_i k$. Notice that $>_i$ is a strict weak order[4] and for any $i, j, k \in V$ exactly one of the following three relations hold: 1) $j >_i k$, 2) $k >_i j$, and 3) $k =_i j$. We also extend the $\sigma_i(P, \alpha)$ and $\sigma_i(P_{-i}, \alpha)$ notations such that they denote a most i-preferred node holding α in P and P_{-i} respectively, breaking ties arbitrarily.

The access cost function d introduced in Section 2 induces a natural node preference relation: $j >_i k$ if $d_{ij} < d_{ik}$, and $j =_i k$ if $d_{ij} = d_{ik}$. In fact, as we show in Lemma 3, undirected networks (i.e., when the access cost function is symmetric) are equivalent to acyclic node preference collections. Formally, the collection

[3] We define a total preorder as a binary relation that satisfies reflexivity, transitivity, and totality. By totality we mean that for any i, j, k, either $j \geq_i k$ or $k \geq_i j$.

[4] A strict weak order is a strict partial order $>$, i.e. a transitive relation that is irreflexive, in which the "neither $a > b$ nor $b > a$" relation is transitive. Strict weak orders and total preorders are widely used in the field of microeconomics.

$\{\geq_i : i \in V\}$ is an *acyclic node preference collection* if there does not exist a sequence of nodes $i_0, i_1, \ldots, i_{k-1}$ for an integer $k \geq 3$ such that $i_{(j-1) \bmod k} >_{i_j} i_{(j+1) \bmod k}$ for all $0 \leq j < k$.

Lemma 3 *Any undirected network yields an acyclic node preference collection. For any acyclic node preference collection, we can compute, in polynomial time, symmetric cost functions that are consistent with the node preferences.*

Proof Let d denote a symmetric access cost function over the set V of nodes. For a given node $i \in V$, we have $j \geq_i k$ iff $d_{ij} \leq d_{ik}$. We now argue that the collection $\{\geq_i : i \in V\}$ is acyclic. Suppose, for the sake of contradiction, that there exists a sequence of nodes $i_0, i_1, \ldots, i_{k-1}$ for an integer $k \geq 3$ such that $i_{(j-1) \bmod k} >_{i_j} i_{(j+1) \bmod k}$ for all $0 \leq j < k$. It then follows that:

$$d_{i_j i_{(j-1) \bmod k}} < d_{i_j i_{(j+1) \bmod k}} \text{ for } 0 \leq j < k.$$

Since d is symmetric, we obtain

$$d_{i_j i_{(j-1) \bmod k}} < d_{i_{(j+1) \bmod k} i_j} \text{ for } 0 \leq j < k,$$

which is a contradiction, since $d_{i_0 i_{(k-1)}} < d_{i_1 i_0} < \cdots < d_{i_{(k-1)} i_0} = d_{i_0 i_{(k-1)}}$.

Given an acyclic collection of node preferences, we compute an associated access cost function d in polynomial time as follows. We construct a directed graph G over the set U of all unordered pairs $(i, j) : i, j \in V, i \neq j$. There is a directed edge from node (i, j) to (i, k) if and only if $k \geq_i j$. Since the collection $\{\geq_i : i \in V\}$ is acyclic, G is a dag. We compute the topological ordering $\pi : U \to \mathbb{Z}$; thus, we have $\pi((i, j)) < \pi((k, \ell))$ whenever there is a directed path from (i, j) to (k, ℓ). Setting d_{ij} to be $\pi((i, j))$ gives us the desired undirected network. □

Utility Preference Relations Each node in our game-theoretic model attaches a utility to each global placement. In our general definition a large class of utility functions it is considered simultaneously. Instead of defining a numerical utility function, we let the utility at each node i be a total preorder $\succeq_i$ among the set of all global placements. The $\succ_i$ and $=_i$ notations over global placements are defined analogously. We require that $\succeq_i$, for each $i \in V$, satisfies the following two basic conditions:

- **Monotonicity**: If for any two global placements P and Q, for each object α, and each node q with $\alpha \in Q_q$, there exists a node p with $\alpha \in P_p$ and $p \geq_i q$, then $P \succeq_i Q$.
- **Consistency**: Let two global placements (P_i, P_{-i}) and (Q_i, Q_{-i}) such that for each object $\alpha \in P_i \cup Q_i$, if p (resp. q) is a most i-preferred node in $V \setminus \{i\}$ holding α, i.e. $\alpha \in P_p$ (resp. $\alpha \in Q_q$), then $p =_i q$. If $(P_i, P_{-i}) \succ_i (Q_i, P_{-i})$, then $(P_i, Q_{-i}) \succeq_i (Q_i, Q_{-i})$.

In words, the monotonicity condition says that for any node, if all the objects in a placement are placed at nodes that are at least as preferred as in another placement, then the node prefers the former placement at least as much as the latter. The consistency condition says that the preference for a node to store one set of objects instead of another is entirely a function of the set of most preferred other nodes that together hold these objects. For instance, if a node i with unit capacity prefers to store α over β in a scenario where the most i-preferred node (other than i) storing α (resp. β) is j (resp. k), then i prefers to store α at least as much as β in any other situation where the most i-preferred node (other than i) storing α (resp. β) is j (resp. k).

Generality of the Conditions We note that many standard utility functions defined for replica placement problems [8, 47, 57], including the sum and max functions, satisfy the monotonicity and consistency conditions. Indeed, any utility function that is an L_p norm, for any p, over the costs for the individual objects, also satisfies the conditions. Furthermore, since the monotonicity and consistency conditions apply to the individual utility functions, our model allows the different nodes to adopt different types of utilities, as long as each separately satisfies the two conditions.

Binary object Preferences One of the utility preference relations classes we study is based on binary object preferences. Assume that each node i is equally interested in an objects set S_i and it does not have any interest in the other objects. Then, $\tau_i(P)$ will denote the $|S_i|$-length sequence of the $\sigma_i(P, \alpha)$, such that $\alpha \in S_i$ and it is in non-increasing order based on the $\geq_i$ relation. In this setup the consistency condition can be further strengthened to the **binary consistency** term: for any placements $P = (P_i, P_{-i})$ and $Q = (Q_i, Q_{-i})$ with $P_{-i} = Q_{-i}$, we let $P \succeq_i Q$ if and only if for $1 \leq k \leq |S_i|$, the k^{th} component of $\tau_i(P)$ is at least as i-preferred as the k^{th} component of $\tau_i(Q)$.

CSR Games We let a CSR game be a tuple $(V, \mathrm{O}, \{\geq_i\}, \{\succeq_i\})$ in the general axiomatic framework. A pure Nash equilibrium in a CSR game instance is a global placement P such that there is no placement Q_i for which $(Q_i, P_{-i}) \succ_i (P_i, P_{-i})$, for each $i \in V$.

To further analyse the complexity results, a definition of a game instance specification is required. We first specify the set V, the node cache capacities, and an enumerated list of object names O. For each node $i \in V$, we specify i's preference relation $\geq_i$ succinctly by a set of at most $\binom{n}{2}$ bits. However, the utility preference relation $\succeq_i$ is over a potentially exponential number of placements in terms of n, m, and cache sizes. We further assume that the utility preference relations are specified by an efficient algorithm, which we denote as *utility preference oracle*, that takes as input a node i, and two global placements P and Q, and returns whether $P \succeq_i Q$. For the sum, max, and L_p-norm utilities, the utility preference oracle simply computes the relevant utility function. For binary object preferences, the binary consistency condition yields an oracle which is polynomial in the number of nodes, objects, and cache sizes.

Unit Cache Capacity We now argue that the unit cache capacity assumption of Section 2 continues to hold without loss of generality. Consider a set V of nodes in which the cache of node i can store c_i objects. Let V' denote a new set of nodes which contains, for each node i in V, new nodes $i_1, i_2, \ldots, i_{c_i}$, i.e., one new node for each unit of the cache capacity of i. We set the node preferences as follows: for all

$i, i', j \in V$, $1 \leq f, \ell \leq c_j$, $1 \leq k, k' \leq c_i$, we have $i_k \geq_{j_\ell} i'_{k'}$ whenever $i \geq_j i'$, and $j_f =_{i_k} j_\ell$.

We consider an obvious onto mapping f from placements in V' to those in V. Given placement P' for V', we set $f(P') = P$ where $P_i = \cup_{1 \leq k \leq c_i} P'_{i_k}$. This mapping naturally defines the utility preference relations for the node set V'. In particular, for any $i \in V$ and $1 \leq k \leq c_i$, $P' \succeq_{i_k} y$ whenever $f(P') \succeq_i f(Q')$. We also note that f is computable in time polynomial in the number of nodes and the sum of the cache capacities. It is easy to verify that the utility preference relation $\succeq_{i_k}$ for all $i_k \in V'$ satisfies the monotonicity and consistency conditions. Furthermore, P' is an equilibrium for V' if and only if $f(P')$ is an equilibrium for V; this together with the onto property of the mapping f gives us the desired reduction.

5 Existence of Equilibria in the General Framework

In this section, we establish the existence of equilibria for several CSR games under the general framework of CSR games with ordinal preferences that we introduced in Section 4. First, we extend the sum utility function results on hierarchical networks to the general framework (Section 5.1). Next, we show that CSR games on undirected networks and binary object preferences are potential games (Section 5.2). Finally, when there are only two objects in the system, we use the technique of fictional players to give a polynomial-time construction of equilibria for CSR games on undirected networks (Section 5.3).

5.1 Hierarchical Networks

We fist show that the polynomial time algorithm which was introduced in Section 3 holds also for the general framework of CSR games with ordinal preferences. A hierarchical network, as defined in the general framework, is a tree T whose leaves set is the node set V and the node preference relation $\geq_i$ is $j \geq_i k$ if $\mathrm{lca}(i, j)$ is a descendant of $\mathrm{lca}(i, k)$. This hierarchical network structure and each node's i pair-preference relations $\sqsupset_i$, determine completely the analysis of the algorithm introduced in Section 3. The latter were defined for the sum utility function. Extending our analysis to the general framework, requires a new preference relation derivation and the establishment of Lemma's 1 analogue, which we present next for arbitrary utility preference relations that satisfy the monotonicity and consistency properties.

Pair Preference Relations For any utility preference relation $\succeq_i$ that satisfies the monotonicity and consistency conditions, we define a strict weak order $\sqsupset_i$ on $\mathrm{O} \times A_i$, where A_i is the set of i's proper ancestors in T.

1. We let $(\alpha, v) \sqsupset_i (\alpha, w)$ hold whenever v is a proper ancestor of w, for each object α, node i, and proper i's ancestors v and w.
2. Considering distinct objects α, β and nodes i, j, k with $j, k \neq i$, we let $\mathcal{P}$ be the set of global placements P, such that j (resp. k) is a most i-preferred node in

$V \backslash \{i\}$ holding α (resp. β) in P_{-i}. If *there exist* global placements $P = (\{\alpha\}, P_{-i})$ and $Q = (\{\beta\}, P_{-i})$ in $\mathcal{P}$ with $P \succ_i Q$, then $(\alpha, \text{lca}(i, j)) \sqsupset_i (\beta, \text{lca}(i, k))$.

In words, item 1 says that i's preference for keeping α in its cache increases as the most i-preferred node holding α becomes less preferred (or "moves farther away"). In item 2, $(\alpha, v) \sqsupset_i (\beta, w)$ means that if i needs to place either α or β in its cache, and the least common ancestor of i and the most i-preferred node in $V \setminus \{i\}$ holding α (resp., β) is v (resp., w), then i prefers to store α over β. The strict weak order $\sqsupset_i$ induces a total preorder $\sqsupseteq_i$ as follows: $(\alpha, v) \sqsupseteq_i (\beta, w)$ if it is not the case that $(\beta, v) \sqsupset_i (\alpha, w)$. We similarly define $=_i$: $(\alpha, v) =_i (\beta, w)$ if $(\alpha, v) \sqsupseteq_i (\beta, w)$ and $(\beta, v) =_i (\alpha, w)$.

Lemma 4 *For each i, $\sqsupset_i$ as given above, is a well-defined strict weak order.*

Proof We need to ensure the well-definedness of part 2 of the definition of pair preference relations. That is, we need to show that for any placements P_{-i} and Q_{-i} such that a most i-preferred node in P_{-i} holding α (resp., β) is also a most i-preferred node in Q_{-i}, it is impossible that $(\{\alpha\}, P_{-i}) \succ_i (\{\beta\}, P_{-i})$ and $(\{\beta\}, Q_{-i}) \succ_i (\{\alpha\}, Q_{-i})$ both hold. This directly follows from the consistency condition for utility preference relations.

The reflexivity and transitivity of $\sqsupseteq_i$ are immediate from the definitions and the reflexivity and transitivity of $\succeq_i$. Finally, to ensure the well-definedness of the strict preorder $\sqsupset_i$, we also have to show that there is no collection of pairs (α_j, v_j), $0 \leq j < \ell$ for some integer $\ell > 1$, such that $(\alpha_j, v_j) \sqsupset_i (\alpha_{j+1 \bmod \ell}, v_{j+1 \bmod \ell})$ for $0 \leq j < \ell$. To see this, it is sufficient to note that if $(\alpha, v) \sqsupset_i (\alpha', v')$ then for all placements P and P' such that $P_{-i} = P'_{-i}$ and the least common ancestor of i and the most i-preferred node in $V \setminus \{i\}$ that holds α (resp. α') is v (resp. v') we have $P \succ_i P'$. So any cycle in the strict preorder $\sqsupset_i$ implies a cycle in $\succ_i$, yielding a contradiction. □

Analogous to Lemma 1, we can express the best response of any player in hierarchical networks as follows. For any global placement $P = (\{\alpha\}, P_{-i})$, assume that j (resp. k) is a most i-preferred node holding object α (resp. β) in P_{-i}, and $(\{\beta\}, P_{-i}) \succ_i (\{\alpha\}, P_{-i})$, i.e., for node i, storing β is a better response to P_{-i} than storing α. Then the following Lemma holds.

Lemma 5 *For any global placement $P = (\{\alpha\}, P_{-i})$, $(\beta, lca(i, k)) \sqsupset_i (\alpha, lca(i, j))$. Furthermore, $\{\alpha\}$ is a best response to P_{-i}, where α maximizes $(\gamma, lca(i, \sigma_i(P_{-i}, \gamma)))$, over all objects γ, according to $\sqsupseteq_i$.*

Proof The first statement of the lemma directly follows from item 2 of the definition of pair preference relations. We establish the second statement by contradiction. Suppose that for node i, $\{\beta\}$ is a better response to P_{-i} than $\{\alpha\}$. Then, we have $(\{\beta\}, P_{-i}) \succ_i (\{\alpha\}, P_{-i})$, which, by item 2 of the definition of pair preference relations, implies that $(\beta, \text{lca}(i, \sigma_i(P_{-i}, \beta))) \sqsupset_i (\alpha, \text{lca}(i, \sigma_i(P_{-i}, \alpha)))$, a contradiction to the choice of α. □

The remainder of the analysis for hierarchical networks (Lemma 2 and Theorem 1) follows as before, invoking Lemma 5 instead of Lemma 1.

5.2 Undirected Networks with Binary Object Preferences

Let d be a symmetric cost function for an undirected network over the node set V. From the binary object preferences definition for each node i we are given an object set S_i in which i is equally interested. We prove the existence of equilibria via a potential function argument. Given a placement P, we let $\Phi_i(P) = d_{ij}$, where j is the most i-preferred node in $V - \{i\}$ holding the object in P_i. We introduce the potential function Φ: $\Phi(P) = (\Phi_0, \Phi_{i_1}(P), \Phi_{i_2}(P), \ldots, \Phi_{i_n}(P))$, where Φ_0 is the number of nodes i such that $P_i \subseteq S_i$, and $\Phi_{i_j}(P) \leq \Phi_{i_{j+1}}(P), \forall j$, where $V = \{i_1, i_2, \ldots, i_n\}$. We prove that Φ is an increasing potential function, i.e. after any better response step, Φ increases in lexicographical order.

Let $P = (P_i, P_{-i})$ be an arbitrary global placement. Assume that $P_i = \{\alpha\}$ and j is the most i-preferred node in P_{-i} holding α. Consider any better response step, from placement P to $Q = (Q_i, P_{-i})$, where $Q_i = \{\beta\}$. Clearly $\beta \in S_i$. We consider two cases. First, suppose $\alpha \notin S_i$ and $\beta \in S_i$. Then, Φ_0 increases, and so does the potential. The second case is where $\alpha, \beta \in S_i$. Let k be the most i-preferred node in P_{-i} holding β. In this case, Φ_0 does not change. However, since this is a better response step of i, $j >_i k$, implying that $d_{ik} > d_{ij}$ and hence $\Phi_i(Q) > \Phi_i(P)$. Consider any other node j. If j holds any object γ other than β, since no new copy of γ has been added, $\Phi_j(Q) \geq \Phi_j(P)$. It remains to consider the case where j holds β. If S is the set of nodes in $V \setminus \{j\}$ holding β in P_{-j}, then $S \cup \{i\}$ is the set of nodes in $V \setminus \{j\}$ holding β. Thus, $\Phi_j(Q) = \min\{\Phi_j(P), d_{ji}\} \geq \min\{\Phi_j(P), \Phi_i(Q)\}$. This also means that $\Phi_j(P)$ appears later in the sorted order than $\Phi_i(P)$ and $\Phi_j(Q)$ appears no earlier in the sorted order than $\Phi_i(Q)$. Hence, $\Phi(Q)$ is lexicographically greater than $\Phi(P)$. This establishes that for undirected networks with binary object preferences, the resulting CSR game is a potential game, and hence also in PLS [37].

5.3 Undirected Networks with Two Objects

In the case of an undirected network with two objects we provide a polynomial-time algorithm to compute an equilibrium. We use the fictional player technique that was introduced in Section 5.1. In the beginning a set of fictional players are introduced to serve the two objects in the network at zero cost from each node. In each subsequent step, the fictional players are progressively moved "further" away, in a way that at each instance the equilibrium is ensured. The whole set of fictional players are completely removed when they are at the least preferred cost from all the nodes, yielding finally to an equilibrium for the original network.

Suppose we are given a undirected network with access cost function d. Also let $\mathcal{D}$ be the set $\{0, \ell_1, \ell_2, \ldots, \ell_r\}$ of all access costs between nodes in the system in increasing order; that is, $\ell_1 = \min_{i,j} d_{ij}$ and $\ell_r = \max_{i,j} d_{ij}$ and $\ell_i < \ell_{i+1}$ for all $1 \leq i < r$.

Fictional Player For an object α, a fictional α-player is a new node that will store α in every equilibrium; an fictional α-player prefers storing α over any other object. We denote by $srv_\alpha(\ell)$ the fictional α-player which is at access cost ℓ from every node in V.

The algorithm

Initialization. Assuming that there are two objects α and β in the system, we initially set up a fictional α-player $srv_\alpha(0)$ and β-player $srv_\beta(0)$ at access cost 0 from each node in V, which does not affect the actual distance between nodes. We let nodes replicate their most preferred object and access the other without any access cost from the corresponding fictional player. This placement is obviously an equilibrium.

Step t of algorithm. Fix an equilibrium P for the node set $V \cup \{srv_\alpha(\ell_t)\} \cup \{srv_\beta(\ell_t)\}$. We describe one step of the algorithm which computes a new set of fictional players $srv_\alpha(\ell_{t+1})$ and $srv_\beta(\ell_{t+1})$ and a new placement P' such that P' is an equilibrium for the node set $V \cup \{srv_\alpha(\ell_{t+1})\} \cup \{srv_\beta(\ell_{t+1})\}$. We first remove the α-player $srv_\alpha(\ell_t)$ from the system and instead we add $srv_\alpha(\ell_{t+1})$. If there do not exist nodes that want to deviate we are done. Otherwise, assume that there exists a node i that wants to deviate from its strategy. Since the most i-preferred node holding β in $V \cup \{srv_\alpha(\ell_t)\} \cup \{srv_\beta(\ell_t)\}$ remains the same in $V \cup srv_\alpha(\ell_{t+1}) \cup srv_\beta(\ell_t)$, i is not holding object α. Thus the only nodes that may want to deviate are those that are holding object β. We argue that if we let i to deviate from $\beta \in P_i$ to $\alpha \in P'_i$, there is no node $j \in V \setminus \{i\}$ that gets affected by i's deviation. Consider the following two cases:

- If a node j has access cost at most ℓ_t from i, then $\beta \in P_j$. Otherwise, if $\alpha \in P_j$, $srv_\alpha(\ell_t)$ would not be the most i-preferred node holding α and thus i would not be affected by any change of α-players. Thus there does not exist any node $j \in V \setminus \{i\}$ with access cost at most ℓ_t from i, such that $\alpha \in P_j$, and as we showed above $\alpha \in P'_j$.
- If a node j has access cost at least ℓ_{t+1} from i, then $P_j = P'_j$. Because of the α-player $srv_\alpha(\ell_{t+1})$ and the β-player $srv_{\ell_t}\beta$, i would never be the j-most preferred node in P'.

We then remove the β-player $srv_\beta(\ell_t)$ from the system and instead we add $srv_\beta(\ell_{t+1})$. Using a similar argument as above, we obtain a new equilibrium at the end of this step.

Theorem 2 *For undirected networks with two objects, an equilibrium can be found in polynomial time.*

Proof An initial placement P, where we have the set of fictional players $srv_\alpha(0)$ and $srv_\beta(0)$ in the system, is obviously an equilibrium. It is immediate from our argument above that at termination the algorithm returns a valid equilibrium.

The size of the set $\mathcal{D}$ is at most $\binom{n}{2}$ which is at most n^2. In each step t at most n nodes may want to deviate from their strategy, since we showed above that if a node deviates once in a step, it will not deviate again during the same step. Thus, the total number of deviations in the algorithm is at most n^3. □

6 Non-Existence of Equilibria in CSR Games and the Associated Decision Problem

In this section, we show that the classes of games studied in Section 5 are essentially the only games where equilibria are guaranteed to exist. We identify the most basic CSR games where equilibria may not exist, and study the complexity of the associated decision problem.

6.1 NP-Completeness

We first show that it is NP-hard to determine whether a given CSR game has an equilibrium even when the utility preference relations are based on the sum utility function and either the number of objects is small or the object preferences are binary. Some simple network examples appear in Fig. 5 (middle and right)–the networks are described in details after Theorem 3–showing that there does not exist an equilibrium in these configurations (proved in the second part of Theorems 4 and 5). The NP-hardness proof is by a polynomial-time reduction from 3SAT [26]. Each reduction is built on top of a gadget which has an equilibrium if and only if a specified node holds a certain object. Several copies of these gadgets are then put together to capture the given 3SAT formula.

Theorem 3 *The problem of determining whether a* CSR *instance has an equilibrium is in* NP *even if one of these three restrictions hold: (a) the number of objects is two; (b) the object preferences are binary and number of objects is three; (c) the network is undirected and the number of objects is three.*

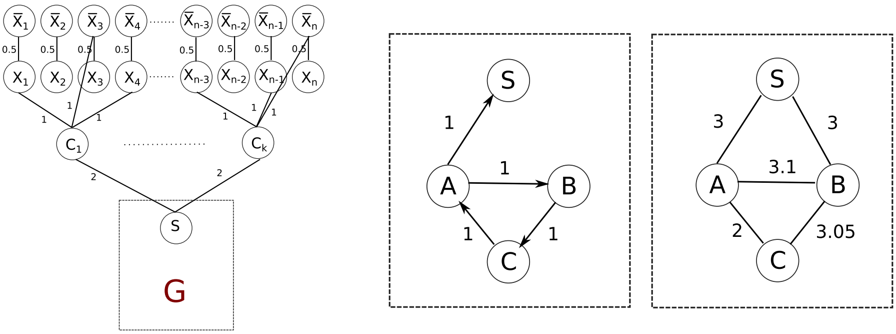

Fig. 5 Left: instance of the construction for the undirected case proof of NP-Hardness, where $\phi = (x_1 \vee \bar{x_3} \vee x_4) \wedge \ldots \wedge (x_{n-3} \vee x_{n-1} \vee \bar{x_n})$. Middle and right: gadget G for the directed and the undirected case

The membership in NP is immediate, since one can determine in polynomial time whether a given global placement is an equilibrium. The remainder of the proof focuses on the hardness reduction from 3SAT.

Given a 3SAT formula ϕ with n variables $x_1, x_2, \ldots, x_n$ and k clauses $c_1, c_2, \ldots, c_k$, we construct a CSR instance as follows. For each variable x_i in ϕ, we introduce two variable nodes X_i and $\bar{X}_i$. We set $d_{X_i\bar{X}_i}$ and the symmetric $d_{\bar{X}_iX_i}$ to be 0.5, where d is the underlying access cost function. For each clause c_j we introduce a clause node C_j. Assuming that $\ell_{j,r}$ for $r \in \{1, 2, 3\}$, are the three literals of the c_j clause in formula ϕ, we set $d_{C_jL_{j,r}}$ and $d_{L_{j,r}C_j}$ to be 1, where $L_{j,r}$ is the corresponding variable node. We also introduce a gadget G illustrated in Fig. 5 (middle and right), consisting of nodes S, A, B, and C. We set the access cost d_{SC_i} and the symmetric d_{C_iS}, for all $1 \leq i \leq k$ between node S and all clause nodes to be 2. The general construction is illustrated in Fig. 5 (left).

Directed Networks with Two Objects We set the access costs $d_{AS} = d_{AB} = d_{BC} = d_{CA} = 1$, and the server node, which stores a fixed copy of two objects α and β, at access cost $d_{srv} = 10$ from all nodes in V. We also set the weights of the variable nodes $r_{x_i}(\alpha) = r_{\bar{x}_i}(\alpha) = r_{x_i}(\beta) = r_{\bar{x}_i}(\beta) = 1$, the weights of the clause nodes $r_{C_i}(\alpha) = 0.85$ and $r_{C_i}(\beta) = 1$, for all $1 \leq i \leq k$. Finally, we set the weights of the nodes in the G gadget $r_S(\alpha) = 0.85$, $r_S(\beta) = r_A(\beta) = r_B(\alpha) = r_B(\beta) = r_C(\alpha) = r_C(\beta) = 1$ and $r_A(\alpha) = 0.7$. We refer to this CSR instance as I_1.

Undirected Networks with Three Objects We set the access costs $d_{AS} = d_{BS} = 3$, $d_{AB} = 3.1$, $d_{BC} = 3.05$, and $d_{CA} = 2$; while symmetry holds. The server node, which stores a fixed copy of three objects α, β, and γ, is at access cost $d_{srv} = 5$ from all nodes in V. We set the weights of the clause nodes $r_{x_i}(\alpha) = r_{\bar{x}_i}(\alpha) = r_{x_i}(\beta) = r_{\bar{x}_i}(\beta) = 1$, the weights of the clause nodes $r_{C_i}(\alpha) = 0.85$ and $r_{C_i}(\beta) = 1$, for all $1 \leq i \leq k$. Finally, we set the weight of the nodes in the G gadget $r_S(\alpha) = 0.85$, $r_S(\beta) = r_A(\alpha) = r_B(\beta) = r_C(\beta) = 1$, $r_A(\gamma) = 2$, $r_B(\gamma) = 0.9837$, and $r_C(\gamma) = 1.6$. All the remaining weights are set to 0. We refer to this CSR instance as I_2.

Lemma 6 *A variable node X_i holds object α (resp., β) if and only if node $\bar{X}_i$ holds object β (resp., α).*

Proof The proof is immediate, since $\bar{X}_i$ (resp., X_i) is X_i's (resp., $\bar{X}_i$'s) nearest node, and both X_i and $\bar{X}_i$ are interested equally in α and β. □

Lemma 7 *Clause node C_i holds object α if and only if its variable nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold object β.*

Proof First, assume that $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold β. These nodes are C_i's nearest nodes holding β. By Lemma 6 we know that nodes $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$ hold α, and they are C_i's nearest nodes holding α. Node's C_i cost for holding α and accessing β from $L_{i,j}$, for $j \in \{1, 2, 3\}$, is $r_{C_i}(\beta)d_{C_iL_{i,j}} = 1$; while the cost for holding β and accessing α from $\bar{L}_{i,j}$, for $j \in \{1,2,3\}$, is $r_{C_i}(\alpha)d_{C_i\bar{L}_{ij}} = 1.275$. Obviously, node C_i prefers to replicate α.

Now assume that at least one of the nodes $\bar{L}_{i,j}$, for $j \in \{1,2,3\}$ holds α. These nodes are C_i's nearest nodes holding α. Also, by Lemma 6, C_i's nearest nodes holding β are all the remaining nodes from the set $L_{i,j}$, $\bar{L_{i,j}}$, for $j \in \{1,2,3\}$, that don't hold α. Node's C_i cost for holding β and accessing α from $L_{i,j}$, for $j \in \{1,2,3\}$, is $r_{C_i}(\alpha)d_{C_iL_{i,j}} = 0.85$; while the cost for holding α and accessing β from node $\bar{L}_{i,j}$ (resp., $L_{i,j}$), is $r_{C_i}(\beta)d_{C_i\bar{L}_{i,j}} = 1.5$ (resp., $r_{C_i}(\beta)d_{C_iL_{i,j}} = 1$). Obviously, in any case node C_i prefers to replicate β. □

Lemma 8 *Node S holds object α if and only if all clause nodes $C_1, \ldots, C_k$ hold object β.*

Proof First, assume that $C_1, \ldots, C_k$ are holding β. These nodes are S's nearest nodes holding β. Also by Lemma 7, S's nearest node holding α is at least one of $L_{i,j}$ nodes, where $i \in [1,k]$, $j \in \{1,2,3\}$. The cost for S holding α and accessing β from a node $C_i, i \in [1,k]$, is $r_S(\beta)d_{SC_i} = 2$; while the cost for holding β and accessing α from $L_{i,j}$, where $i \in [1,k]$, $j \in \{1,2,3\}$, is $r_S(\alpha)d_{SL_{i,j}} = 2.55$. Obviously, node S prefers to replicate α.

Now assume that at least one of $C_1, \ldots, C_k$ holds α. These nodes are S's nearest node holding α. Also S's nearest node holding β, due to Lemma 7 is one of $L_{i,j}$, where $i \in [1,k]$, $j \in \{1,2,3\}$. The cost for holding β and accessing α from a node C_i, is $r_S(\alpha)d_{SC_i} = 1.7$; while the cost for holding α and accessing β from a node L_{ij}, where $j \in \{1,2,3\}$, is $r_S(\beta)d_{SL_{i,j}} = 3$. Obviously, in any case node S prefers to replicate β. □

Theorem 4 *The* CSR *instance I_1 has an equilibrium if and only if node S holds object α.*

Proof First, assume that S is holding α. By Lemma 8 nodes $C_1, \ldots, C_k$ hold object β, and by Lemma 7 at least one of nodes $L_{i,j}$, for $j \in \{1,2,3\}$ for each node $C_i, i \in [1,k]$, holds object α, and the corresponding $\bar{L}_{i,j}$ is holding object β. We claim that the placement where A holds β, B holds β, and C holds α, is a pure Nash equilibrium. We prove this by showing that none of these nodes wants to deviate from their strategy.

Node A does not want to deviate since its cost for holding object β and accessing α from A's nearest node S, is $r_A(\alpha)d_{AS} = 0.7$; while the cost for holding object α and accessing β from A's nearest node B, is $r_A(\alpha)d_{AB} = 1$. Node B does not want to deviate since its cost for holding object β and accessing α from B's nearest node C, is $r_B(\alpha)d_{BC} = 1$; while the cost for holding object α and accessing β from B's nearest node A, is $r_B(\beta)d_{BA} = 2$. Node C does not want to deviate since its cost for holding object α and accessing β from C's nearest node A, is $r_C(\beta)d_{CA} = 1$; while the cost for holding object β and accessing α from C's nearest node S, is $r_C(\alpha)d_{CS} = 2$. Also note that none of $S, C_1, \ldots, C_k, L_{ij}, \bar{L}_{ij}$ for $i \in [1,k]$, $j \in \{1,2,3\}$ is getting affected of the objects been held by the gadget nodes.

Now assume that node S holds object β. We are going to prove that for every possible placement over nodes A, B, and C, at least one node wants to deviate from its strategy. Consider the following cases:

- Nodes A, B, and C hold object α: Node B (resp., C) wants to deviate, since the cost for holding object α and accessing β from B's (resp., C's) nearest node S, is $r_B(\beta)d_{BS} = 3$ (resp., $r_C(\beta)d_{CS} = 2$); while the cost for holding object β and accessing α from B's nearest node A, is $r_B(\beta)d_{BA} = 2$ (resp., $r_C(\beta)d_{CA} = 1$).
- Two nodes hold object α and the third holds β: In the case where A and B hold α, A wants to deviate since the cost while holding α and accessing β from A's nearest node S is $r_A(\beta)d_{AS} = 1$; while the cost for holding β and accessing α from A's nearest node B is $r_A(\alpha)d_{AB} = 0.7$. In the case where A and C hold α, then C wants to deviate since the cost while holding α and accessing β from C's nearest node B is $r_C(\beta)d_{CB} = 2$; while the cost for holding β and accessing α from C's nearest node A is $r_C(\alpha)d_{CA} = 1$. In the case where B and C hold α, B wants to deviate since the cost while holding α and accessing β from B's nearest node A is $r_B(\beta)d_{BA} = 2$; while the cost for holding β and accessing α from B's nearest node C is $r_B(\alpha)d_{BC} = 1$.
- One node holds α: If A (resp., B, or C) holds α, B (resp., C, A) wants to deviate since the cost while holding β and accessing α from B's (resp., C's, or A's) nearest node A (resp., B, or C) is $r_B(\alpha)d_{BA} = 2$ (resp., $r_C(\alpha)d_{CB} = 2$, or $r_A(\alpha)d_{AC} = 1.4$); while the cost for holding α and accessing β from B's (resp., C's, or A's) nearest node C (resp., A, or B), is $r_B(\beta)d_{BC} = 1$ (resp., $r_C(\beta)d_{CA} = 1$, or $r_A(\beta)d_{AB} = 1$).
- Nodes A, B, and C hold β: All of them want to deviate. Node A wants to deviate since the cost while holding β and accessing α from A's nearest node C_i, for some $i \in [1, k]$, is $r_A(\beta)r_{AC_i} = 3$; while the cost for holding β and accessing α from A's nearest node S is $r_A(\alpha)d_{AS} = 0.7$. Similar proof holds for nodes B and C.

Obviously the system does not have a pure Nash equilibrium, which completes the proof. □

Theorem 5 *The* CSR *instance I_2 has an equilibrium if and only if node S holds object α.*

Proof First, assume that S is holding α. By Lemma 8 nodes $C_1, \ldots, C_k$ hold object β, and by Lemma 7 at least one of nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ for each node $C_i, i \in [1, k]$, holds object α, and the corresponding $\bar{L}_{i,j}$ is holding object β. We claim that the placement where A holds γ, node B holds β, and C holds γ is a pure Nash equilibrium. We prove this by showing that none of these nodes wants to deviate from their strategy. Node A doesn't want to deviate since the cost for holding object γ and accessing object α from node S is $r_A(\alpha)d_{AS} = 3$; while the cost for holding α and accessing γ from node C increases to $r_A(\gamma)d_{AC} = 4$. Node B doesn't want to deviate since the cost for holding object β and accessing object γ from node C is $r_B(\gamma)d_{BC} = 3.000285$; while the cost for holding object β and accessing γ from the server increases to $r_B(\gamma)d_{srv} = 5$. Node C doesn't want to deviate since the cost for holding object γ and accessing β from node B is $r_C(\beta)d_{CB} = 3.05$; while the cost for holding object β and accessing γ from node A increases to $r_C(\beta)d_{CA} = 3.2$.

Now assume that node S holds object β. We are going to prove that for every possible placement over nodes A, B, and C, at least one node wants to deviate from its strategy. Consider the following cases:

- Node A holds α, node B holds γ, and node C holds β: Node A wants to deviate since the cost while it is holding object α and accessing object γ from node B is ($r_A(\gamma)d_{AB} = 6.2$); while the cost for holding object γ and accessing α from the server decreases to $r_A(\alpha)d_{srv} = 5$.
- Node A holds γ, node B holds γ, and node C holds β: Node B wants to deviate since the cost while it is holding object γ and accessing object β from node C is ($r_B(\beta)d_{BC} = 3.05$); while the cost for holding object β and accessing γ from node A decreases to $r_B(\gamma)d_{BA} = 3.04947$.
- Node A holds γ, node B holds β, and node C holds β: Node C wants to deviate since the cost while it is holding object β and accessing object γ from node A is ($r_C(\gamma)d_{CA} = 3.2$); while the cost for holding object γ and accessing β from node B decreases to $r_C(\beta)d_{CB} = 3.05$.
- Node A holds γ, node B holds β, and node C holds γ: Node A wants to deviate since the cost while it is holding object γ and accessing object α from the server is ($r_A(\alpha)d_{srv} = 5$); while the cost for holding object α and accessing γ from node C decreases to $r_A(\gamma)d_{AC} = 4$.
- Node A holds α, node B holds β, and node C holds γ: Node B wants to deviate since the cost while it is holding object β and accessing object γ from node C is ($r_B(\gamma)d_{BC} = 3.000285$); while the cost for holding object γ and accessing β from node S decreases to $r_B(\beta)d_{BS} = 3$.
- Node A holds α, node B holds γ, and node C holds γ: Node C wants to deviate since the cost while it is holding object γ and accessing object β from the server is ($r_C(\beta)d_{srv} = 5$); while the cost for holding object β and accessing γ from B decreases to $r_C(\gamma)d_{BC} = 4.88$.
- Node A holds α, node B holds β, and node C holds β: Node C wants to deviate since the cost while it is holding object β and accessing object γ from the server is ($r_C(\gamma)d_{srv} = 4.9185$); while the cost for holding object γ and accessing β from node B decreases to $r_B(\beta)d_{CB} = 3.05$.
- Node A holds γ, node B holds γ, and node C holds γ: Node A wants to deviate since the cost while it is holding object γ and accessing object α from the server is ($r_A(\alpha)d_{srv} = 5$); while the cost for holding object α and accessing γ from C decreases to ($r_A(\gamma)d_{AC} = 4$.

The remaining placements where A holds α, B holds α, and C holds α, obviously are not stable since none of the nodes are interested in these objects. Since there does not exist a stable placement, an equilibrium does not exist. □

Binary Object Preferences Over Three Objects For the binary object preferences, we introduce two extra nodes K and L. We set d_{C_iK}, for $i \in [1, k]$, between clause nodes and K to be 1.4, d_{SL} to be 2.1, and d_{AS}, d_{AB}, d_{BC}, d_{CA} to be 1. The server node, which is at access cost $d_{srv} = 10$ from all nodes in V, stores a fixed copy of three objects α, β, and γ. Each node i has a set S_i of objects in which it is equally interested. For nodes X_i, $\bar{X}_i$, for $i \in [1, n]$, we set $S_{X_i} = \{\alpha, \beta\}$ and $S_{\bar{X}_i} = \{\alpha, \beta\}$. For nodes C_i, for $i \in [1, k]$, we set $S_{C_i} = \{\alpha, \gamma\}$. For node K we set $S_K = \{\gamma\}$; while for node L we set $S_L = \{\beta\}$. For node S we set $S_S = \{\alpha, \beta\}$. For nodes A, B, and C we set S_A, S_B, and S_C correspondingly to be the set $\{\alpha, \gamma\}$. As we mentioned in the binary object preference definition for our utility function $U_s(i)$, equally interested means weight 1 for all objects in S_i, and 0 for the remaining. We refer to this instance as I_3.

Lemma 6 holds as it is for the binary object preferences directed case.

Lemma 9 *Clause node C_i holds object α if and only if its variable nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold object β.*

Proof First, assume that $L_{i,j}$, for $j \in \{1, 2, 3\}$ hold β. By Lemma 6 we know that nodes $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$ hold α, and they are C_i's nearest nodes holding α; while C_i's nearest node holding γ is node K. Node's C_i cost for holding α and accessing γ from K is $d_{C_iK} = 1.4$; while the cost for holding γ and accessing α from $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$, is $d_{C_i\bar{L}_{ij}} = 1.5$. Obviously, node C_i prefers to replicate α.

Now assume that at least one of the nodes $\bar{L}_{i,j}$, for $j \in \{1, 2, 3\}$ holds α. These nodes are C_i's nearest nodes holding α; while again C_i's nearest node holding γ is node K. Node's C_i cost for holding γ and accessing α from $L_{i,j}$, for $j \in \{1, 2, 3\}$, is $d_{C_iL_{i,j}} = 1$; while the cost for holding α and accessing γ from node K is $d_{C_iK} = 1.4$. Obviously, node C_i prefers to replicate γ. □

Lemma 10 *Node S holds object α if and only if all clause nodes $C_1, \ldots, C_k$ hold object γ.*

Proof First, assume that $C_1, \ldots, C_k$ are holding γ. By Lemma 9, S's nearest node holding α is at least one of $L_{i,j}$ nodes, where $i \in [1, k]$, $j \in \{1, 2, 3\}$; while S's nearest nodes holding β is node L. The cost for S holding α and accessing β from node L, is $d_{SL} = 2.1$; while the cost for holding β and accessing α from $L_{i,j}$, where $i \in [1, k]$, $j \in \{1, 2, 3\}$, is $d_{SL_{i,j}} = 3$. Obviously, node S prefers to replicate α.

Now assume that at least one of $C_1, \ldots, C_k$ holds α. These nodes are S's nearest node holding α; while again S's nearest node holding β is L. The cost for holding β and accessing α from a node C_i, is $d_{SC_i} = 2$; while the cost for holding α and accessing β from a node L is $d_{SL} = 2.1$. Obviously, node S prefers to replicate β. □

Theorem 6 *There exists an equilibrium for the* CSR *instance I_3 if and only if node S holds object α.*

Proof First, assume that S is holding α. By Lemma 10 nodes $C_1, \ldots, C_k$ hold object γ, and by Lemma 9 at least one of nodes $L_{i,j}$, for $j \in \{1, 2, 3\}$ for each node

$C_i, i \in [1, k]$, holds object α, and the corresponding $\bar{L}_{i,j}$ is holding object β. We claim that the placement where A holds γ, B holds γ, and C holds α, is a pure Nash equilibrium. We prove this by showing that none of these nodes wants to deviate from their strategy.

Node A does not want to deviate since its cost for holding object γ and accessing α from A's nearest node S, is $d_{AS} = 1$; while the cost for holding object α and accessing γ from A's nearest node B, is still $d_{AB} = 1$. Node B does not want to deviate since its cost for holding object γ and accessing α from B's nearest node C, is $d_{BC} = 1$; while the cost for holding object α and accessing γ from B's nearest node A, is still $d_{BA} = 1$. Node C does not want to deviate since its cost for holding object α and accessing γ from C's nearest node A, is $d_{CA} = 1$; while the cost for holding object γ and accessing α from C's nearest node S, is still $d_{CS} = 1$. Also note that none of $S, C_1, \ldots, C_k, L_{ij}, \bar{L}_{ij}$ for $i \in [1, k]$, $j \in \{1, 2, 3\}$ is getting affected of the objects been holded by the gadget nodes.

Now assume that node S holds object β. We are going to prove that for every possible placement over nodes A, B, and C, at least one node wants to deviate from its strategy. Consider the following cases:

- Nodes A, B, and C hold object α: Node B (resp., C) wants to deviate, since the cost for holding object α and accessing γ from B's (resp., C's) nearest node C_i, for some $i \in [1, k]$ or from node K, is $d_{BC_i} = 5$ or $d_{BK} = 6.4$ (resp., $d_{CC_i} = 4$ or $d_{CK} = 5.4$); while the cost for holding object γ and accessing α from B's nearest node A, is $d_{BA} = 2$ (resp., $d_{CA} = 1$).
- Two nodes hold object α and the third holds γ: In the case where A and B hold α, A wants to deviate since the cost while holding α and accessing γ from A's nearest node C is $d_{AC} = 2$; while the cost for holding γ and accessing α from A's nearest node B is $d_{AB} = 1$. The other cases are symmetric.
- One node holds α: If A holds α, B wants to deviate since the cost while holding γ and accessing α from B's nearest node A is $d_{BA} = 2$; while the cost for holding α and accessing γ from B's nearest node C, is $d_{BC} = 1$. The other cases are symmetric.
- Nodes A, B, and C hold γ: All of them want to deviate. Node A wants to deviate since the cost while holding γ and accessing α from A's nearest node C_i, for some $i \in [1, k]$, is $d_{AC_i} = 3$; while the cost for holding α and accessing γ from A's nearest node B is $d_{AB} = 1$. The other cases are symmetric.

Obviously the system does not have a pure Nash equilibrium, which completes the proof. □

We now show that ϕ is satisfiable if and only if the above CSR games (both undirected and directed cases) (resp., for the binary object preferences, directed case) has a pure Nash equilibrium. Suppose that ϕ is satisfiable and consider a satisfying assignment for ϕ. If the assignment of a variable x_i is True, then we replicate object α in cache of variable node X_i; otherwise, we replicate object β. By Lemma 6 we know that a variable node X_i holds object α (resp., β) if and only if node $\bar{X}_i$ holds object β (resp., α). In this way we keep the consistency between truth assignment of a variable and its negation. By Lemma 7 (resp., Lemma 9) we know that a clause node C_i, will replicate object β (resp., γ) if and only if at least one of its variable nodes, holds object α. From above, any clause node C_i will hold object β (resp., γ) only if at least one of clause c_i literals is True. By Lemma 8 (resp., Lemma 10), we know that node S, will replicate object α if and only if all clause nodes $C_1, \ldots, C_k$ are holding object β (resp., γ). Thus, node S replicates object α only if all clauses $c_1, \ldots, c_k$ are True. By Theorems 4 and 5 (resp., 6), we know that there exists a pure Nash Equilibrium if and only if object β is stored to node S; thus, there exists a pure Nash Equilibrium if and only if all clauses are True. This gives our proof.

6.2 Binary Preferences Over Two Objects

Consider the problem 2BIN: does a given CSR instance with two objects and binary preferences possess an equilibrium? We prove that 2BIN is polynomial-time equivalent to the notorious EVEN-CYCLE problem [72]: does a given digraph contain an even cycle? Despite intensive efforts, the complexity of the problem EVEN-CYCLE was open until [51, 61] provided a tour de force polynomial-time algorithm. Our result thus also places 2BIN in P.

Theorem 7 EVEN-CYCLE *is polynomial-time equivalent to* 2BIN.

We prove the polynomial-time equivalence of 2BIN and EVEN-CYCLE by a series of reductions. We first show the equivalence between 2BIN and 2DIR-BIN, which is the sub-class of 2BIN instances in which the node preferences are specified by an unweighted directed graph (henceforth *digraph*); in a 2DIR-BIN instance, we are given a digraph, and the preference of a node for the other nodes increases with decreasing distance in the graph.

Lemma 11 2BIN *is polynomial-time equivalent to* 2DIR-BIN.

Proof Given a 2BIN instance I with node set V, two objects, node preference relations $\{\geq_i : i \in V\}$, and interest sets $\{S_i : i \in V\}$, we construct a 2DIR-BIN instance I' with the same node set, objects, and interest sets, but with the node preference relations specified by an unweighted digraph G. Our construction will ensure that any equilibrium in I is an equilibrium in I' and vice-versa. For distinct nodes i and j, we have an edge from i to j if and only if j is a most i-preferred node in $V \setminus \{i\}$. We now argue that I has an equilibrium if and only if I' has an equilibrium. A placement for I is an equilibrium if and only if the following holds for each node i: (a) if $|S_i| = 1$, then i holds the lone object in S_i; (b) if $|S_i| = 2$, then the object not held by i is at an i-most preferred node. Similarly, any equilibrium placement for I' satisfies the following condition for each i: (a) if $|S_i| = 1$, then i holds the lone object in S_i; (b) if $|S_i| = 2$, then the object not held by i is at a neighbor of i. By our construction of the instances, equilibria of I are equilibria of I' and vice-versa. □

We next define EXACT-2DIR-BIN, which is the subclass of 2DIR-BIN games where each node is interested in both objects; thus, an EXACT-2DIR-BIN instance is completely specified by a digraph G. We say that a node i is *stable* in a given placement P if P_i is a best response to P_{-i}. We say that an EXACT-2DIR-BIN instance G is *stable* (resp., *1-critical*) if there exists a placement in which all nodes (resp., all nodes except at most one) are stable. Since each node has unit cache capacity, each placement is a 2-coloring of the nodes: think of a node as colored by the object it holds in its cache. Given a placement, an arc is said to be bichromatic if its head and tail have different colors. Note that for any EXACT-2DIR-BIN instance, a node is stable in a placement iff it has a bichromatic outgoing arc.

Lemma 12 2DIR-BIN *and* EXACT-2DIR-BIN *are polynomial-time equivalent on general digraphs.*

Proof Since EXACT-2DIR-BIN games are a special subclass of 2DIR-BIN games, we only need to show that 2DIR-BIN games reduce to EXACT-2DIR-BIN games. Given an instance of a 2DIR-BIN game, we need to handle the nodes that are interested in at most one object. First, note that we can remove the outgoing arcs from all such nodes. Let V_0 consist of the nodes with no objects of interest. For each node u in V_0 we add a new node u_0 to V_0 along with arcs (u, u_0) and (u_0, u). Let red and blue denote the two objects. Let V_r and V_b denote the set of nodes interested in red and blue, respectively. Without loss of generality, let $|V_r| \geq |V_b|$. Add $|V_r| - |V_b|$ additional nodes to the set V_b (so that $|V_r| = |V_b|$) and connect all the nodes in $V_r \bigcup V_b$ with a directed cycle that alternates strictly between V_r nodes and V_b nodes. The rest of the network is kept the same and all the nodes are set to have interest in both objects. Now, if the original instance is stable then we can stabilize the new instance by having each node in V_r (resp., V_b) cache the red (resp., blue) object, the nodes in V_0 cache any object (so long as an original node u and its associated node u_0 store complementary objects) and the other nodes cache the same object as in the placement that made the original instance stable. And in the other direction, if the transformed instance is stable then in an equilibrium placement, the nodes in V_r must each store an object of one color while each node in V_b stores the object of the other color. By renaming the colors, if necessary, we get a stable coloring (placement) for the original instance. □

For completeness, we next present some standard graph-theoretic terminology that we will use in our proof. A digraph is said to be *weakly* connected if it is possible to get from a node to any other by following arcs without paying heed to the direction of the arcs. A digraph is said to be *strongly* connected if it is possible to get from a node to any other by a directed path. We will use the following well-known structure result about digraphs: a general digraph that is weakly connected is a directed acyclic graph on the unique set of maximal strongly connected (node-disjoint) components. We will also use the following strengthening of the folklore ear-decomposition of strongly connected digraphs [65]:

Lemma 13 *An ear-decomposition can be obtained starting with any cycle of a strongly connected digraph.*

Proof The proof is by contradiction. Suppose not, then consider a subgraph with a maximal ear-decomposition obtainable from the cycle in question. If it is not the entire digraph then consider any arc leaving the subgraph. Note that the digraph is strongly connected and hence such an arc must exist. Further, note that every arc in a digraph is contained in a cycle since there is a directed path from the head of the arc to the tail. Starting from the arc follow this cycle until it intersects the subgraph again, as it must because it ends at the tail which lies in the subgraph. This forms an ear that contradicts the maximality of the decomposition. □

Lemma 14 EVEN-CYCLE *on strongly connected digraphs and* EVEN-CYCLE *on general digraphs are polynomial-time equivalent.*

Proof Since strongly connected digraphs are a special subclass of general digraphs it suffices to show that EVEN-CYCLE on general digraphs can be reduced to EVEN-CYCLE on strongly connected digraphs. Remember that a general digraph has a unique set of maximal strongly connected components that are disjoint and computable in polynomial-time. Further any cycle, including even cycles, must lie entirely within a strongly connected component. Thus a digraph possesses an even cycle iff one of its strongly connected components does. Hence it follows that EVEN-CYCLE on general digraphs reduces to EVEN-CYCLE on strongly connected digraphs. □

Lemma 15 EVEN-CYCLE *and* EXACT-2DIR-BIN *games are polynomial-time equivalent on strongly connected digraphs.*

Proof To show the polynomial-time equivalence, we show that a strongly connected digraph is stable iff it has an even cycle. One direction is easy. If the digraph is stable then consider the placement in which every node is stable. So every node has a bichromatic outgoing arc; by starting at any node and following outgoing bichromatic edges we will eventually loop back on ourselves. The loop so obtained is the required even cycle; it is even because it is composed of bichromatic arcs. In the other direction, if there is an even cycle then we take the ear-decomposition starting with that cycle (Lemma 13), stabilize that cycle (by making each arc bichromatic since it is of even cardinality) and then stabilize each node in each ear by working backwards along the ear. □

Lemma 16 *Any* EXACT-2DIR-BIN *game on a strongly connected digraph is 1-critical.*

Proof Consider an ear-decomposition of the strongly connected digraph starting with a cycle. Observe that all but at most one node of the cycle can be stabilized by arbitrarily assigning one color to a node, and then assigning alternate colors to the nodes as we progress along the cycle. Every node in the cycle, other than possibly the initial node, is stable. The rest of the digraph can be stabilized ear by ear, stabilizing each ear by working backwards from the point of attachment. Hence, all but one node of the digraph can be stabilized. □

Lemma 17 EXACT-2DIR-BIN *on general digraphs is polynomial-time equivalent to* EXACT-2DIR-BIN *on strongly connected digraphs.*

Proof Since strongly connected digraphs are a subclass of general digraphs we need only show that the problem EXACT-2DIR-BIN on general digraphs reduces to EXACT-2DIR-BIN on strongly connected digraphs. A general digraph is stable iff all of its weakly connected components are. A weakly connected component is a directed acyclic graph (dag) on the strongly connected components. It is clear that a weakly connected component cannot be stabilized if any one of the strongly connected components that is a minimal element of the directed acyclic graph cannot be stabilized. Interestingly, the converse is also true. If all of the strongly connected components that are minimal elements of the dag can be stabilized then the entire weakly connected component can be stabilized because each of the other strongly connected components has at least one outgoing arc which is used to stabilize its tail while the rest of the strongly connected component can be stabilized because strongly connected components are 1-critical by Lemma 16. We can determine such a stable placement by processing the strongly connected components in topologically sorted order (according to the dag) starting from the minimal elements. Thus a digraph is stable iff every strongly connected component that is a minimal element is stable. Hence, EXACT-2DIR-BIN on general digraphs is reducible in polynomial-time to strongly connected digraphs. □

7 Fractional Replication Games

We introduce a new class of capacitated replication games where nodes can store fractions of objects, as opposed to whole objects, and satisfy an object access request by retrieving enough fractions that make up the whole object. Rather than associate different identities with different fractions of a given object, we view each portion of an object as being fungible, thus allowing any set of fractions of an object, adding up to at least one, to constitute the whole object. Such fractional replication scenarios naturally arise when objects are encoded and distributed within a network to permit both efficient and reliable access.

Several implementations of fractional replication, in fact, already exist. For instance, fountain codes [5, 66] and the information dispersal algorithm [59] present two ways of encoding an object as a number of smaller pieces – of size, say $1/m$ fraction of the full object size, where m is an integer – such that the full object may be reconstructed from any m of the pieces. A natural formalization is to view each object as a polynomial of high degree, and consider each piece of the object as the evaluation of the polynomial on a random point in a suitable large field. Then, accessing an object is equivalent (with very high probability) to accessing a sufficient number of pieces of the object.

We now present fractional capacitated selfish replication (F-CSR) games, which are an adaptation of the game-theoretic framework developed in Section 4 to fractional replication. We have a set V of nodes sharing a set O of objects. In an F-CSR game, the strategies are *fractional placements*; a fractional placement $\widetilde{P}$ is a $|V|$-tuple

$\{\widetilde{P}_i : i \in V\}$ where $\widetilde{P}_i : \text{O} \rightarrow \Re$ under the constraint that sum of $\widetilde{P}_i(\alpha)$, over all α in O, is at most the cache size of i.

We begin by presenting F-CSR games in the special case of sum utilities, where the generalization from the integral to the fractional setting is most natural. For sum utilities, recall that we are given a cost function d and node-object weights $r_i(\alpha)$, $i \in V$, $\alpha \in \text{O}$. Given a fractional global placement $\widetilde{P}$, we define the cost incurred by i for accessing object α as the minimum value of $x_j d_{ij}$ under the constraints that $\sum_j x_j = 1$ and $x_j \leq \widetilde{P}_j(\alpha)$ for all j. Then, the total cost incurred by i is the sum, over all objects α, of $r_i(\alpha)$ times the cost incurred by i for accessing α. For a given fractional global placement $\widetilde{P}$, the utility of i is the negative of the total cost incurred by i under $\widetilde{P}$.

We now consider F-CSR games under the more general setting of utility preference relations. As before, each node i has a node preference relation $\geq_i$ and a preference relation $\succeq_i$ among global (integral) placements. Recall that the node and placement preference relations of each node i induce a preorder $\sqsupseteq_i$ among the elements of $\text{O} \times (V \setminus \{i\})$ (see Section 4). For F-CSR games, we require the existence of a *total* preorder $\sqsupseteq_i$, for all i. We now specify the best response function for each player for a given fractional global placement $\widetilde{P}$. For each node i and object α, we determine the assignment $\mu_{i,\widetilde{P},\alpha} : V \setminus \{i\} \rightarrow \Re$ that is lexicographically minimal under the node preference relation $\geq_i$ subject to the condition that $\mu_{i,\widetilde{P},\alpha} \leq \widetilde{P}_k(\alpha)$ for each k and $\sum_k \mu_{i,\widetilde{P},\alpha}(k) = 1$. We next compute $b_{i,\widetilde{P}} : \text{O} \times (V \setminus \{i\}) \rightarrow \Re$ to be the lexicographically maximal assignment under $\sqsupseteq_i$ subject to the condition that $b_{i,\widetilde{P}}(\alpha, k) \leq \mu_{i,\widetilde{P},\alpha}(k)$ for all k and $\sum_{\alpha,k} b_{i,\widetilde{P}}(\alpha, k)$ is at most the size of i's cache. The best response of a player i is then to store $\sum_k b_{i,\widetilde{P}}(\alpha, k)$ of α in their cache. This completes the definition of F-CSR games.

Using standard fixed-point machinery, we show that every F-CSR game has an equilibrium. We also show that finding equilibria in F-CSR games is PPAD-complete.

Theorem 8 *Every* F-CSR *instance has a pure Nash equilibrium. Finding an equilibrium in an* F-CSR *game is* PPAD-*complete.*

We prove Theorem 8 by establishing separately the existence of equilibria, membership in PPAD, and the PPAD-hardness of finding equilibria.

7.1 Existence of Equilibria

Theorem 9 *Every* F-CSR *instance has a pure Nash equilibrium.*

Proof By [54] (Proposition 20.3, based on Kakutani's fixed-point theorem), a game has a pure Nash equilibrium if the strategy space of each player is a compact, non-empty, convex space, and the payoff function of each player is continuous on the strategy space of all players and quasi-concave in the strategy space of the player. In an F-CSR instance, the strategy space of each player i is simply the set of all its fractional placements: that is, the set of functions $f : \text{O} \rightarrow [0, 1]$ subject to condition that $\sum_{\alpha \in \text{O}} f(\alpha) \leq c_i$, where c_i is the cache size of the node (player). The strategy set thus is clearly convex, non-empty, and compact. Furthermore, as defined above, the payoff for any player i under fractional placement $\widetilde{P}$ is simply the solution to the following linear program:

$$\begin{aligned} \max - \sum_{\alpha \in \mathrm{O}} r_i(\alpha) \left(\sum_{j \in V} x_{ij}(\alpha) d_{ij} \right) & \\ \sum_{j \in V} x_{ij}(\alpha) = 1 & \quad \text{for all } i \in V, \alpha \in \mathrm{O} \\ x_{ij}(\alpha) \leq \widetilde{P}_j(\alpha) & \quad \text{for all } i, j \in V, \alpha \in \mathrm{O} \\ x_{ij}(\alpha) \geq 0 & \quad \text{for all } i, j \in V, \alpha \in \mathrm{O} \end{aligned}$$

It is easy to see that the payoff function is both continuous in the placements of all players, and quasi-concave in the strategy space of player i, thus completing the proof of the theorem. □

7.2 Membership in PPAD

Theorem 10 *Finding an equilibrium in an* F-CSR *game is in* PPAD.

Proof Our proof is by a reduction from FSPP (Fractional Stable Paths Problem), which is defined as follows [39]. Let G be a graph with a distinguished destination node d. Each node $v \neq d$ has a list $\pi(v)$ of simple paths from v to d and a preference relation $\geq_v$ among the paths in $\pi(v)$. For a path S, we also define $\pi(v, S)$ to be the set of paths in $\pi(v)$ that have S as a suffix. A *proper suffix* S of P is a suffix of P such that $S \neq P$ and $S \neq \emptyset$. A *feasible fractional paths solution* is a set $w = \{w_v : v \neq d\}$ of assignments $w_v : \pi(v) \to [0, 1]$ satisfying: (1) **Unity condition**: for each node v, $\sum_{P \in \pi(v)} w_v(P) \leq 1$, and (2) **Tree condition**: for each node v, and each path S with start node u, $\sum_{P \in \pi(v,S)} w_v(P) \leq w_u(S)$.

In other words, a feasible solution is one in which each node chooses at most 1 unit of flow to d such that no suffix is filled by more than the amount of flow placed on that suffix by its starting node. A feasible solution w is *stable* if for any node v and path Q starting at v, one of the following holds: **(S1)** $\sum_{P \in \pi(v)} w_v(P) = 1$, and for each P in $\pi(v)$ with $w_v(P) > 0$, $P \geq_v Q$; or **(S2)** There exists a proper suffix S of Q such that $\sum_{P \in \pi(v,S)} w_v(P) = w_u(S)$, where u is the start node of S, and for each $P \in \pi(v, S)$ with $w_v(P) > 0$, $P \geq_v Q$.

Given an F-CSR G with node set V, object set O, node preference relations $\geq_i$ for $i \in V$, and utility preference relations $\succeq_i$ for $i \in V$, we construct an instance $\mathcal{I}$ of FSPP as follows. For nodes $i, j \in V$ and object $\alpha \in \mathrm{O}$, we introduce the following FSPP vertices.

- hold(i, α) representing the amount of α that node i will store in its cache.
- serve(i, j, α) representing the amount of α that node j will serve for i given a placement for $V \setminus \{i\}$.
- serve'(i, j, α), an auxiliary vertex needed for serve(i, j, α).

- serve(i, α), representing the amount of α that other nodes will serve for i given a placement for $V \setminus \{i\}$.
- hold(i), representing the best response of i give the placement of other nodes.
- hold'(i, α), an auxiliary vertex needed for hold(i, α).

We now present the path sets and preferences for each vertex of the FSPP instance.

- serve(i, α): the path set includes all paths of the form $\langle$serve(i, α), hold$(j, \alpha), d\rangle$, and serve(i, α) prefers $\langle$serve(i, α), hold$(j, \alpha), d\rangle$ over $\langle$serve(i, α), hold(k, α), $d\rangle$ if $j \geq_i k$.
- serve'(i, j, α): the path set includes all paths of the form $\langle$ serve'(i, j, α), serve($i, \alpha)$, hold$(j, \alpha), d\rangle$ and the direct path $\langle$serve'$(i, j, \alpha), d\rangle$. For the preference order, serve'(i, j, α) prefers all paths $\langle$serve'(i, j, α), serve(i, α), hold$(j, \alpha), d\rangle$ equally, and all of them over the direct path.
- serve(i, j, α): the path set includes the path $\langle$serve'$(i, j, \alpha), d\rangle$ and the direct path $\langle$serve$(i, j, \alpha), d\rangle$ with a higher preference for the former path.
- hold(i): the path set includes paths of the form $\langle$hold(i), serve$(i, j, \alpha), d\rangle$, and hold(i) prefers the path $\langle$hold(i), serve$(i, j, \alpha), d\rangle$ over $\langle$hold(i), serve(i, k, β), $d\rangle$ if $(j, \alpha) \sqsupseteq_i (k, \beta)$.
- hold'(i, α): the path set includes paths of the form $\langle$ hold'(i, α), hold(i), serve($i, j, \alpha), d\rangle$ all of which are preferred equally, and the direct path $\langle$hold'$(i, \alpha), d\rangle$ which is preferred the least.
- hold(i, α): the path set includes two paths $\langle$hold'$(i, \alpha), d\rangle$ and the direct path with a higher preference for the former path.

We now show that the F-CSR instance has an equilibrium if and only if the FSPP instance has an equilibrium. Our proof is by giving a mapping f from global fractional placements in the F-CSR instance to feasible solutions in the FSPP instance such that (a) if $\widetilde{P}$ is an equilibrium for the F-CSR instance, then $f(\widetilde{P})$ is an equilibrium for the FSPP instance, and (b) if w is an equilibrium for the FSPP instance, then $f^{-1}(w)$ is an equilibrium for the F-CSR instance.

Let $\widetilde{P}$ denote any fractional placement of the F-CSR instance. We now define the solution $f(\widetilde{P})$ of the FSPP instance. In $f(\widetilde{P})$ vertex hold(i, α) plays $\widetilde{P}_i(\alpha)$ on the direct path and $1 - \widetilde{P}_i(\alpha)$ on the other path in its path set, for every i in V and α in O. The remaining vertices play their best responses, considered in the following order. First, consider vertices of the form serve(i, α). In the best response, the amount played by serve(i, α) on the path $\langle$serve(i, α), hold$(j, \alpha), d\rangle$, equals $\mu_{i,\widetilde{P},\alpha}(j)$; recall that $\mu_{i,\widetilde{P},\alpha}(j)$ is the assignment that is lexicographically minimal under the node preference relation $\geq_i$ subject to the condition that $\mu_{i,\widetilde{P},\alpha} \leq \widetilde{P}_k(\alpha)$ for each k and $\sum_k \mu_{i,\widetilde{P},\alpha}(k) = 1$. We next consider the vertices of the form serve'(i, j, α). In its best response, vertex serve'(i, j, α) plays $\mu_{i,\widetilde{P},\alpha}(j)$ on the path $\langle$serve'(i, j, α), serve(i, α), hold$(j, \alpha), d\rangle$. Next, in its best response, vertex serve(i, j, α) plays $\mu_{i,\widetilde{P},\alpha}(j)$ on its direct path and $1 - \mu_{i,\widetilde{P},\alpha}(j)$ on its remaining path. We now consider the best response of vertex hold(i); it distributes its unit among paths of the form $\langle$hold(i), serve(i, j, α), $d\rangle$ (for all j in $V \setminus \{i\}$ and α in O) lexicographically maximally under the total preorder $\sqsupseteq_i$ over node-object pairs. That is, hold(i) plays $b_{i,\widetilde{P}}(\alpha, j)$ on the path $\langle$hold(i), serve$(i, j, \alpha), d\rangle$. We next consider the best response of the vertex hold'(i, α); it plays $1 - \sum_j b_{i,\widetilde{P}}(\alpha, j)$ on its direct path and $b_{i,\widetilde{P}}(\alpha, j)$ on the path $\langle$hold'(i, α), hold(i), serve$(i, j, \alpha), d\rangle$. This completes the definition of the solution $f(\widetilde{P})$.

We now argue that if $\widetilde{P}$ is an equilibrium so is $f(\widetilde{P})$. By construction, every vertex other than of the form hold(i, α) play their best responses in $f(\widetilde{P})$. We next show that i plays a best response in $\widetilde{P}$ if and only if the vertices hold(i, α) play their best response in $f(\widetilde{P})$. The best response of hold(i, α) is to play $1 - \sum_j b_{i,\widetilde{P}}(\alpha, j)$ on the path $\langle$hold(i, α), hold'$(i, \alpha), d\rangle$ and the $\sum_j b_{i,\widetilde{P}}(\alpha, j)$ on its direct path. The best response of i in $\widetilde{P}$ is to set $\widetilde{P}_i(\alpha)$ to $\sum_j b_{i,\widetilde{P}}(\alpha, j)$. Thus if $\widetilde{P}$ is an equilibrium, then so is $f(\widetilde{P})$. Furthermore, if w is an equilibrium, by definition of f, $\widetilde{P} = f^{-1}(w)$ is well-defined. Since the best responses of i and the vertices hold(i, α) are consistent, $\widetilde{P}$ is also an equilibrium. This completes the reduction from F-CSR to FSPP, placing F-CSR in PPAD. □

7.3 PPAD-Hardness

This section is devoted to the proof of the following theorem.

Theorem 11 *The problem of finding an equilibrium in* F-CSR *games is PPAD-hard even when the underlying cost function d is a metric.*

Our reduction is from preference games [39]. Given a preference game G with n players $1, 2, \ldots, n$ and their preferences given by $\geq_i$, we construct an F-CSR game $\widehat{G}$ as follows. The game $\widehat{G}$ has a set V of n^2+3n players numbered 1 through n^2+3n, and a set O of $2n$ objects $\alpha_1, \ldots, \alpha_{2n}$. We set the utility function for each node to be the sum utility function, thus ensuring that the desired monotonicity and consistency conditions are satisfied.

We next present the metric cost function d over the nodes. We group the players into four sets $V_1 = \{i : 1 \leq i \leq n\}$, $V_2 = \{i \cdot n + j : 1 \leq i \leq n, 1 \leq j \leq n\}$, $V_3 = \{n^2 + n + i : 1 \leq i \leq n\}$, and $V_4 = \{n^2 + 2n + i : 1 \leq i \leq n\}$. For each node i in V_1 and j in V_3, we set $d_{ii} = 2$ and $d_{ij} = 4$. We set $d_{n^2+n+i, n^2+2n+i} = 3$. For each node i in V_1 and $k = i \cdot n + j$, we set d_{ik} as follows: if $j >_i i$ then d_{ij} equals $6 - \ell/n$ when j is the ℓth most preferred player for i; if $i \geq_i j$, then d_{ij} equals 1. All the other distances are obtained by using metric properties.

We finally specify the object weights. For $k \in V_1$, we set $r_k(\alpha_i) = 1$ for all $i \neq k$ such that $i \geq_k k$; we set $r_k(\alpha_k) = 2.5$ such that $4 < 2r_k(\alpha_k) \leq 5$. For node $k = i \cdot n + j$ in V_2, we set $r_k(\alpha_j) = 1$. For node $k = n^2 + n + i$ in V_3, we set $r_k(\alpha_i) = r_k(\alpha_{i+n}) = 1$. Finally, for node $k = n^2+2n+i$ in V_4, we set $r_k(\alpha_{i+n}) = 1$.

Given a placement P for $\widehat{G}$, we define a solution $\omega(P) = \{w_{ij}\}$ for the preference game G: $w_{ij} = P_i(\alpha_j)$. The following lemma immediately follows from the definition of $\widehat{G}$.

Lemma 18 *The following statements hold for any placement P for $\widehat{G}$.*

- *For $k = i \cdot n + j$, $1 \leq j \leq n$, P_k is a best response to P_{-k} if and only if $P_k(\alpha_j) = 1$.*

- *For $k = n^2 + n + i$, $1 \leq i \leq n$, P_k is a best response to P_{-k} if and only if $P_k(\alpha_{n+i}) = 1$.*
- *For $k = n^2+n+i$, P_k is a best response to P_{-k} if and only if $P_k(\alpha_i) = 1 - P_i(\alpha_i)$ and $P_k(\alpha_{n+i} = P_i(\alpha_i)$.*

Lemma 19 *Let P be a placement for $\widehat{G}$ in which every node not in V_1 plays their best response. Then, the best response of a node i in V_1 is the lexicographically maximum $(P_i(\alpha_{j_1}), P_i(\alpha_{j_2}), \ldots, P_i(\alpha_{j_n}))$, where $j_1 \geq_i j_2 \geq_i \cdots \geq_i j_n$, subject to the constraint that $P_i(\alpha_j) \leq P_j(\alpha_j)$ for $j \neq i$.*

Proof Fix a node i in V_1. By Lemma 18, node $i \cdot n + j$ holds object j, for $1 \leq j \leq n$; each of these nodes is at distance at least 5 and at most 6 away from i. By Lemma 18, for every node $k = n^2 + n + j$, $1 \leq j \leq n$, $P_k(\alpha_j) = 1 - P_j(\alpha_j)$ and $P_k(\alpha_{n+j}) = P_j(\alpha_j)$.

We now consider the best response of node i. We first note that for any $j \in \{1, \ldots, n\} \setminus \{i\}$ such that $i \geq_i j$, $P_i(\alpha_j) = 0$ since the nearest full copy of α_j is nearer than the nearest node holding any fraction of object α_i. Let S denote the set of j such that $j \geq_i i$. For any j in $S \setminus \{i\}$, $P_i(\alpha_j) \leq P_j(\alpha_j)$ since node $n^2 + n + j$ at distance 5 holds $1 - P_j(\alpha_j)$ fraction of α_j, the nearest node holding any fraction of α_i is at distance 4, and $4r_i(\alpha_i) > 5r_i(\alpha_j)$. Furthermore, for any j, k in S if $j >_i k$, then the farthest $P_j(\alpha_j)$ fraction of α_j is farther than the farthest $P_k(\alpha_k)$ fraction of α_k, implying that in the best response, if $P_i(\alpha_j) < P_j(\alpha_j)$ then $P_i(\alpha_k) = 0$. Thus, the best response of i is the unique lexicographically maximum solution $(P_i(\alpha_{j_1}), P_i(\alpha_{j_2}), \ldots, P_i(\alpha_{j_n}))$, where $j_1 \geq_i j_2 \geq_i \cdots \geq_i j_n$, subject to the constraint that $P_i(\alpha_j) \leq P_j(\alpha_j)$ for $j \neq i$. □

Lemma 20 *A placement P is an equilibrium for $\widehat{G}$ if and only if $\omega(P)$ is a equilibrium for G and every node not in V_1 plays their best response in P.*

Proof Consider an equilibrium placement P for $\widehat{G}$ Clearly, every node plays their best response. We now prove that $\omega(P)$ is an equilibrium for G. Fix a node i in V_1. By Lemma 19, the best response of i is the unique lexicographically maximum solution $(P_i(\alpha_{j_1}), P_i(\alpha_{j_2}), \ldots, P_i(\alpha_{j_n}))$, where $j_1 \geq_i j_2 \geq_i \cdots \geq_i j_n$, subject to the constraint that $P_i(\alpha_j) \leq P_j(\alpha_j)$ for $j \neq i$. Since this applies to every node i, it is immediate from the definitions of $\omega(P)$ and preference games that if P is an equilibrium for $\widehat{G}$ then $\omega(P)$ is an equilibrium for G.

We now consider the reverse direction. Suppose we have a placement P in which every player not in V_1 plays their best response and $\omega(P)$ is an equilibrium for the preference game G. By Lemma 19 and the definition of $\omega(P)$, the best response of i in G matches that in the F-CSR game; hence every player in V_1 also plays their best response in P, implying that P is an equilibrium for $\widehat{G}$. □

The construction of $\widehat{G}$ from G is clearly polynomial time. Furthermore, given any equilibrium for $\widehat{G}$, an equilibrium for G can be constructed in linear time. We thus have a reduction from a PPAD-complete problem to F-CSR implying that the latter is PPAD-hard, thus completing the proof of Theorem 11.

8 Concluding Remarks

In this paper, we first define the integral and fractional selfish replication games (CSR and F-CSR) in networks. In our setup each node has a bounded cache capacity for uniform size objects. We prove that every hierarchical network has a pure Nash equilibrium, introducing the notion of fictional players. We almost completely characterize the complexity of CSR games, i.e. which classes have an equilibrium, the complexity of determine whether it exists, and if so, how efficiently it can be found. For the open complexity question about undirected networks with binary preferences (proved to be potential games), we conjecture that finding equilibria is PLS-hard. For the cases of games where equilibria exist, we study the convergence of the best response process. The main focus of this work is in equilibria, leaving the problem of estimating the price of anarchy for all the configurations as future work, extending the work of [20].

We also show that F-CSR games always have equilibria, though they may be hard to find. It is not hard to argue that an equilibrium in the corresponding integral variant is an equilibrium in the fractional instance. So whenever an "integral" equilibrium can be determined efficiently, so can a "fractional" equilibrium. An interesting direction of research is to identify other special cases of fractional games where equilibria may be efficiently determined. We also note that our proof of existence of equilibria in F-CSR games, currently presented for the case of unit-size objects, extends to arbitrary object sizes.

Finally, even though our proofs work for a model that the sets of nodes, objects, and preference relations are all static, we believe that our results will be meaningful for dynamically changing environments. Developing better models for addressing infrequently changes is a very important practical research direction.

References

1. Ahmadyan, S.N., Etesami, S.R., Poor, H.V.: A random tree search algorithm for Nash equilibrium in capacitated selfish replication games. In: IEEE 55th conference on decision and control (CDC), pp 4439–4444. https://doi.org/10.1109/CDC.2016.7798943 (2016)
2. Angel, E., Bampis, E., Pollatos, G.G., Zissimopoulos, V.: Optimal data placement on networks with constant number of clients. Theoretical Computer Science (2013)
3. Arrow, K.: Social choice and individual values. Yale University Press (1951)
4. Baev, I.D., Rajaraman, R., Swamy, C.: Approximation Algorithms for Data Placement Problems. SIAM J. Comput. **38**(4), 1411–1429 (2008)
5. Byers, J.W., Luby, M., Mitzenmacher, M., Rege, A.: A digital fountain approach to reliable distribution of bulk data. In: SIGCOMM '98, pp 56–67 (1998)
6. Chen, X., Deng, X., Teng, S.H.: Settling the complexity of computing two-player Nash equilibria. Journal of the ACM (JACM), 56(3) (2009)
7. Chen, Y., Katz, R.H., Kubiatowicz, J.D.: Scan: A dynamic, scalable, and efficient content distribution network. In: Mattern, F., Naghshineh, M. (eds.) Pervasive computing, pp. 282–296. Springer, Berlin (2002)
8. Chun, B.G., Chaudhuri, K., Wee, H., Barreno, M., Papadimitriou, C.H., Kubiatowicz, J.: Selfish caching in distributed systems: A game-theoretic analysis. In: ACM symposium on principles of distributed computing (PODC), pp 21–30 (2004)

9. Dabek, F., Kaashoek, M.F., Karger, D., Morris, R., Stoica, I.: Wide-area cooperative storage with cfs. In: Proceedings of the 8th ACM symposium on operating systems principles, SOSP '01, pp. 202–215. ACM, New York (2001). https://doi.org/10.1145/502034.502054
10. Danzig, P.: Netcache architecture and deployment. Comput. Netw. ISDN Syst. **30**(22-23), 2081–2091 (1998). https://doi.org/10.1016/S0169-7552(98)00250-5
11. Daskalakis, C., Goldberg, P.W., Papadimitriou, C.H.: The complexity of computing a Nash equilibrium. STOC ACM, 71–78 (2006)
12. Devanur, N.R., Garg, N., Khandekar, R., Pandit, V., Saberi, A., Vazirani, V.V.: Price of anarchy, locality gap, and a network service provider game. In: WINE, pp 1046–1055 (2005)
13. Douceur, J.R., Wattenhofer, R.P.: Large-scale simulation of replica placement algorithms for a serverless distributed file system. In: MASCOTS 2001 proceedings 9th international symposium on modeling, analysis and simulation of computer and telecommunication systems, pp. 311–319. https://doi.org/10.1109/MASCOT.2001.948882 (2001)
14. Etesami, S.R., Basar, T.: Pure Nash equilibrium in capacitated selfish replication (CSR) game. arXiv:1404.3442 (2014)
15. Etesami, S.R., Basar, T.: Approximation algorithm for the binary-preference capacitated selfish replication game and a tight bound on its price of anarchy. arXiv:1506.04047v2 (2016)
16. Etesami, S.R., Basar, T.: Pure Nash equilibrium in a capacitated resource allocation game with binary preferences. arXiv:1404.3442v3 (2016)
17. Etesami, S.R., Basar, T.: Pure Nash equilibrium in a capacitated selfish resource allocation game. IEEE Transactions on Control of Network Systems **PP**(99), 1–1 (2016)
18. Etesami, S.R., Başar, T.: Network games, pp. 1–46. Springer International Publishing, Cham (2017). https://doi.org/10.1007/978-3-319-27335-8_10-1
19. Etesami, S.R., Başar, T.: An approximation algorithm and price of anarchy for the binary-preference capacitated selfish replication game. In: 54th IEEE conference on decision and control (CDC), pp. 3568–3573. https://doi.org/10.1109/CDC.2015.7402771 (2015)
20. Etesami, S.R., Başar, T.: Price of anarchy and an approximation algorithm for the binary-preference capacitated selfish replication game. Automatica **76**(Supplement C), 153–163 (2017). https://doi.org/10.1016/j.automatica.2016.10.002
21. Fabrikant, A., Luthra, A., Maneva, E., Papadimitriou, C.H., Shenker, S.: On a Network Creation Game. In: PODC '03: Proceedings of the 22nd annual symposium on principles of distributed computing, pp. 347–351. ACM Press, New York (2003). https://doi.org/10.1145/872035.872088
22. Fan, L., Cao, P., Almeida, J., Broder, A.Z.: Summary cache: A scalable wide-area web cache sharing protocol. In: Proceedings of the ACM SIGCOMM '98 conference on applications, technologies, architectures, and protocols for computer communication, SIGCOMM '98, pp. 254–265. ACM, New York (1998). https://doi.org/10.1145/285237.285287
23. Feldman, M., Chuang, J.: Overcoming free-riding behavior in peer-to-peer systems. ACM Sigecom Exchanges **5**(4), 41–50 (2005)
24. Fudenberg, D., Levine, D.: The theory of learning in games. MIT Press (1998)
25. Garces-Erice, L., Biersack, E.W., Felber, P.A., Ross, K.W., Urvoy-Keller, G.: Hierarchical peer-to-peer systems. Parallel Process. Lett. **13**(04), 643–657 (2003). https://doi.org/10.1142/S0129626403001574
26. Garey, M., Johnson, D.: Computers and intractability. Freeman Press (1979)
27. Goemans, M.X., Li, L., Mirrokni, V.S., Thottan, M.: Market sharing games applied to content distribution in ad hoc networks. IEEE J. Sel. Areas Commun. **24**(5), 1020–1033 (2006). https://doi.org/10.1109/JSAC.2006.872884
28. Goemans, M.X., Li, L., Mirrokni, V.S., Thottan, M.: Market sharing games applied to content distribution in Ad Hoc networks. IEEE J. Sel. Areas Commun. **24**(5), 1020–1033 (2006)
29. Gopalakrishnan, R., Kanoulas, D., Karuturi, N., Pandu Rangan, C., Rajaraman, R., Sundaram, R.: Cache me if you can: Capacitated selfish replication games. In: Latin American symposium on theoretical informatics (LATIN), vol. 7256, pp. 420-432 (2012)
30. Gribble, S.D., Halevy, A.Y., Ives, Z.G., Rodrig, M., Suciu, D.: What can database do for peer-to-peer?. In: WebDB workshop on databases and the web (2001)
31. Hu, X., Gong, J.: PhD forum: Not so cooperative caching. In: 21st IEEE international conference on network protocols (ICNP), pp. 1–3. https://doi.org/10.1109/ICNP.2013.6733656 (2013)

32. Hu, X.Y., Gong, J.: Study on the theoretical framework of not so cooperative caching. J. Internet Technol. **15**(3), 351–362 (2014). https://doi.org/10.6138/JIT.2014.15.3.04
33. Iyer, S., Rowstron, A.I.T., Druschel, P.: Squirrel: a decentralized peer-to-peer web cache. In: PODC (2002)
34. Jain, K., Vazirani, V.V.: Primal-dual approximation algorithms for metric facility location and k-median problems. In: 40th annual symposium on foundations of computer science (Cat. No.99CB37039), pp. 2–13. https://doi.org/10.1109/SFFCS.1999.814571 (1999)
35. Jamin, S., Jin, C., Jin, Y., Raz, D., Shavitt, Y., Zhang, L.: On the placement of internet instrumentation. In: Proceedings IEEE INFOCOM 2000. Conference on computer communications. 19th annual joint conference of the ieee computer and communications societies (Cat. No.00CH37064), vol. 1, pp. 295-304. https://doi.org/10.1109/INFCOM.2000.832199 (2000)
36. Jamin, S., Jin, C., Kurc, A.R., Raz, D., Shavitt, Y.: Constrained mirror placement on the internet. In: Proceedings IEEE INFOCOM 2001. Conference on computer communications. 20th annual joint conference of the ieee computer and communications society (Cat. No.01CH37213), vol. 1, pp. 31–40. https://doi.org/10.1109/INFCOM.2001.916684 (2001)
37. Johnson, D.S., Papadimitriou, C.H., Yannakakis, M.: How Easy is Local Search? J. Comput. Syst. Sci. **37**(1), 79–100 (1988)
38. Karger, D., Lehman, E., Leighton, T., Panigrahy, R., Levine, M., Lewin, D.: Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In: Proceedings of the 29th annual ACM symposium on theory of computing (STOC), pp. 654–663 (1997)
39. Kintali, S., Poplawski, L.J., Rajaraman, R., Sundaram, R., Teng, S.H.: Reducibility among fractional stability problems. Proceedings of the 50th annual IEEE symposium on foundations of computer science (FOCS) (2009)
40. Ko, B.J., Rubenstein, D.: Distributed self-stabilizing placement of replicated resources in emerging networks. IEEE/ACM Trans. Netw. **13**(3), 476–487 (2005). https://doi.org/10.1109/TNET.2005.850196
41. Korupolu, M., Plaxton, C.G., Rajaraman, R.: Placement algorithms for hierarchical cooperative caching. J. Algorithms **38**, 260–302 (2001)
42. Korupolu, M.R., Dahlin, M.: Coordinated placement and replacement for large-scale distributed caches. IEEE Trans. Knowl. Data Eng. **14**(6), 1317–1329 (2002)
43. Koutsoupias, E., Papadimitriou, C.: Worst-case equilibria. In: STACS, pp. 404–413 (1999)
44. Kubiatowicz, J., Bindel, D., Chen, Y., Czerwinski, S., Eaton, P., Geels, D., Gummadi, R., Rhea, S., Weatherspoon, H., Weimer, W., Wells, C., Zhao, B.: Oceanstore: An architecture for global-scale persistent storage. ACM SIGPLAN Not. **35**(11), 190–201 (2000). https://doi.org/10.1145/356989.357007
45. Laoutaris, N., Smaragdakis, G., Bestavros, A., Stavrakakis, I.: Mistreatment in distributed caching groups: Causes and implications. In: INFOCOM (2006)
46. Laoutaris, N., Telelis, O., Zissimopoulos, V., Stavrakakis, I.: Distributed selfish replication. IEEE Trans. Parallel Distrib. Syst. **17**(12), 1401–1413 (2006)
47. Laoutaris, N., Smaragdakis, G., Oikonomou, K., Stavrakakis, I., Bestavros, A.: Distributed placement of service facilities in large-scale networks. In: INFOCOM, pp. 2144–2152 (2007)
48. Leff, A., Wolf, J.L., Yu, P.S.: Replication algorithms in a remote caching architecture. IEEE Trans. Parallel Distrib. Syst. **4**(11), 1185–1204 (1993)
49. Li, B., Golin, M.J., Italiano, G.F., Deng, X., Sohraby, K.: On the optimal placement of web proxies in the internet. In: IEEE INFOCOM '99. Conference on computer communications. Proceedings. 18th annual joint conference of the IEEE computer and communications societies. The future is now (Cat. No.99CH36320), vol. 3, pp. 1282–1290. https://doi.org/10.1109/INFCOM.1999.752146 (1999)
50. Mahdian, M., Ye, Y., Zhang, J.: Improved approximation algorithms for metric facility location problems. In: Jansen, K., Leonardi, S., Vazirani, V. (eds.) Approximation algorithms for combinatorial optimization, pp. 229-242. Springer, Berlin (2002)
51. McCuaig, W., Robertson, N., Seymour, P.D., Thomas, R.: Permanents, Pfaffian orientations, and even directed circuits. In: STOC, pp. 402–405 (1997)
52. Mettu, R.R., Plaxton, C.G.: The online median problem. In: Proceedings of the 41st annual symposium on foundations of computer science, p. 339. IEEE Computer Society, Washington (2000). FOCS '00

53. Nisan, N., Roughgarden, T., Tardos, É., Vazirani VV: Algorithmic game theory. Cambridge University Press, Cambridge (2007)
54. Osborne, M.J., Rubinstein, A.: A course in game theory. MIT Press (1994)
55. Pacifici, V.: Resource allocation in operator-owned content delivery systems. KTH School of Electrical Engineering, PhD thesis (2016)
56. Papadimitriou, C.H.: On the complexity of the parity argument and other inefficient proofs of existence. JCSS **48**(3), 498–532 (1994)
57. Pollatos, G.G., Telelis, O., Zissimopoulos, V.: On the social cost of distributed selfish content replication. In: Networking, pp. 195–206 (2008)
58. Qiu, L., Padmanabhan, V.N., Voelker, G.M.: On the placement of web server replicas. In: Proceedings IEEE INFOCOM 2001. Conference on computer communications. 20th annual joint conference of the IEEE computer and communications society (Cat. No.01CH37213), vol. 3, pp. 1587-1596. https://doi.org/10.1109/INFCOM.2001.916655 (2001)
59. Rabin, M.: Efficient dispersal of information for security, load balancing and fault tolerance. J. ACM **36**, 335–348 (1989)
60. Rabinovich, M., Rabinovich, I., Rajaraman, R., Aggarwal, A.: A dynamic object replication and migration protocol for an internet hosting service. In: Proceedings. 19th IEEE international conference on distributed computing systems (Cat. No.99CB37003), pp. 101–113. https://doi.org/10.1109/ICDCS.1999.776511 (1999)
61. Robertson, N., Seymour, P.D., Thomas, R.: Permanents, Pfaffian orientations, and even directed circuits. Annals of Mathematics, pp. 929–975 (1999)
62. Rosenwein, M.B.: Discrete location theory. In: Mirchandani, P.B., Francis, R.L. (eds.), p. 555. Wiley, New York (1990). Networks 24(2):124–125 https://doi.org/10.1002/net.3230240212
63. Rowstron, A., Druschel, P.: Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility. In: Proceedings of the 18th ACM symposium on operating systems principles, SOSP '01, pp. 188–201. ACM, New York (2001). https://doi.org/10.1145/502034.502053
64. Saito, Y., Karamanolis, C., Karlsson, M., Mahalingam, M.: Taming aggressive replication in the pangaea wide-area file system. SIGOPS Oper. Syst. Rev. **36**(SI), 15–30 (2002). https://doi.org/10.1145/844128.844131
65. Schrijver, A.: Combinatorial optimization (3 Vol.) Springer-Verlag, Berlin (2003)
66. Shokrollahi, A.: Raptor codes. In: IEEE Trans Inf Theory, pp. 2551–2567 (2006)
67. Tang, X., Chanson, S.T.: Coordinated en-route web caching. IEEE Trans. Comput. **51**(6), 595–607 (2002). https://doi.org/10.1109/TC.2002.1009146
68. Tewari, R., Dahlin, M., Vin, H.M., Kay, J.S.: Design Considerations for Distributed Caching on the Internet. In: ICDCS, pp 273–284 (1999)
69. Tsaknakis, H., Spirakis, P.G., Kanoulas, D.: Performance evaluation of a descent algorithm for bi-matrix games. In: Internet and network economics, 4th international workshop, WINE 2008, Shanghai, China, December 17-20, 2008. Proceedings, pp 222–230. https://doi.org/10.1007/978-3-540-92185-1_29 (2008)
70. Vetta, A.: Nash equilibria in competitive societies, with applications to facility location traffic routing and auctions. In: FOCS (2002)
71. Wolfson, O., Jajodia, S., Huang, Y.: An Adaptive Data Replication Algorithm. ACM Trans. Database Syst. **22**, 255–314 (1997)
72. Younger, D.H.: Graphs with interlinked directed circuits. In: Proceedings of midwestern symposium on circuit theory, vol. 2, pp. XVI2.1–XVI2.7 (1973)