Capacitated Caching Games

Ragavendran Gopalakrishnan
California Institute of Technology
ragad3@caltech.edu

Dimitrios Kanoulas
Northeastern University
dkanou@ccs.neu.edu

Naga Naresh Karuturi
Google India
naganareshk@google.com

C. Pandu Rangan
IIT Madras, India
rangan@iitm.ernet.in

Rajmohan Rajaraman
Northeastern University
rraj@ccs.neu.edu

Ravi Sundaram
Northeastern University
koods@ccs.neu.edu

July 16, 2010

Abstract

Capacitated Caching (CC) Games are motivated by P2P and web caching applications, and involve nodes on a network making strategic choices regarding the content to replicate in their caches. Caching games were introduced by Chun et al [6], who analyzed the uncapacitated case leaving the capacitated version as an open direction.

In this work, we study pure Nash equilibria of both fractional (FCC) and integral (ICC) versions of CC games. Using erasure codes we present a compelling realization of FCC games in content distribution. We show that every FCC game instance possesses an equilibrium. We also show, however, that finding an equilibrium in an FCC game is PPAD-complete.

For ICC games we delineate the boundary between intractability and effective computability in terms of the network structure, object preferences, and the total number of objects. A central result of this paper is the existence (and poly-time computability) of equilibria for hierarchical networks. Using a potential function argument, we also show the existence of equilibria for general symmetric networks when object preferences are binary. For general ICC games, however, equilibria may not exist. In fact, we show that it is NP-hard to determine whether equilibria exist, even when either the network is symmetric and there are only three objects in the system, or the network is arbitrary and there are only two objects in the system. Finally, we show that the existence of equilibria in strongly connected networks with two objects and binary object preferences can be determined in polynomial time, via a reduction to the even-cycle problem.

A significant feature of our work is the development of the above results in a general framework of natural preference orders, orders that are entirely arbitrary except for two benign constraints - “Nearer is better” and “Irrelevance of alternatives”. A novel aspect of our framework is that the results apply even to the situation where different nodes have utility functions of completely different functional forms (e.g. one node’s utility could depend on the sum of distances to the objects of interest whereas another’s could depend on the maximum distance).

1 Introduction

You have a computer with a limited amount of disk space at home. You need to decide what movies to store on your hard drive. Your decision is influenced by your social network and their collections of movies as well. If a friend who is nearby and has good connectivity to you is storing *Avatar* then you can borrow it from him when needed; on the other hand if no proximate friend shares your interest in the classics then you may be better off storing a local copy of *Casablanca*. Note that your decisions affect those of your friends, who in turn take actions that affect you. A natural question arises: what is the prognosis for you and your network of friends in terms of the stability of your movie collections and the contentment you will derive from them?

In the brave new wireless world of 4G you will not only be a consumer of different apps you will also be a provider of apps to others around you. Your personal computing device will not only act as an end-device but also as an intermediate relay for others. You will pay others for the privilege of using apps hosted by them while getting paid for apps that you host. Each of you will independently attempt to optimize your utility function but could it lead to a situation of endless churn or could there be an equilibrium?

In this paper, we study Capacitated Caching (CC) Games, which provide an abstraction of the above scenarios. These are games in which the strategic agents, or players, are nodes in a network. The nodes have object preferences as well as bounded storage space – caches – in which they can store copies of the content. Each node has a utility for each placement of objects in the caches based on where objects of interest have been placed. As with the earlier works [6, 14, 20, 1], we focus on the case where all pieces of content have the same size, for otherwise the problem is a generalization of the well-known knapsack problem and even computing the best response of a player(node) is NP-hard.

Several variants of CC games can be formulated by considering different kinds of networks (e.g., directed, symmetric, hierarchical) and various forms of object preferences (e.g., uniform or weighted). Our paper subsumes all the earlier works in the capacitated case by developing a general framework that allows us to characterize the computational complexity of the different variants of CC Games.

1.1 Our results

We consider two classes of CC games: integral (ICC) and fractional (FCC). In the integral case, nodes are only allowed to store an object entirely or not at all, whereas in the fractional case we allow nodes to store fractions of objects rather than entire objects.

- We define ICC and FCC games under a general framework of utility preference relations and node preference orders. Rather than specifying a numerical utility assigned by each node to each placement of objects, we only require that the preference order each node has on object placements satisfy two natural constraints of Monotonicity (or "Nearer is Better") and Consistency (or "Irrelevance of Alternatives"). Section 2 presents a detailed description of our framework.

A node's preference order among different placements may depend on the underlying network structure and its preferences among different objects. We consider symmetric networks (where pairwise distances are symmetric), hierarchical networks, and general networks. We consider general object preferences as well as binary object preferences, where each node is interested in a subset of objects but is neutral between objects in the subset. Our first set of results concerns ICC games.

- Table 1.1 completely characterizes the computational complexity of the different classes of ICC games we study. We wish to highlight three results in the table: (1) the existence (and polynomial-time computability) of equilibria for hierarchical networks, (2) the existence of equilibria for symmetric networks when object preferences are binary, and (3) the equivalence of finding equilibria in ICC games with two objects and binary object preferences to the even-cycle problem [22].

Object preferences and count	Symmetric networks	General networks
Binary, two objects	Yes, in P (4.3)	No, in P (3.2)
Binary, three or more objects	Yes, in PLS (4.2)	No, NP-complete (3.1)
General, two objects	Yes, in P (4.3)	No, NP-complete (3.1)
General, three or more objects	No, NP-complete (3.1) Hierarchical: Yes, in P (4.1)	No, NP-complete (3.1)

Table 1: Existence and computability of equilibria in ICC games . Each cell (other than in the first row or the first column) first indicates whether equilibria always exist in the particular sub-class of ICC games. If equilibria always exist, then the cell next indicates the complexity of determining an equilibrium; otherwise, the cell indicates the complexity of determining whether equilibria exist for a given instance. The numbers in parentheses denote the relevant subsection.

Our second set of results concerns FCC games, where each node is allowed to store fractions of objects. In our framework, a node can satisfy an object access request by retrieving any set of fractions of the object as long as these fractions sum to at least one. A natural implementation of this framework is via erasure codes; in fact, the Digital Fountain approach to data dissemination provides precisely this property [4, 24] (see Section 5 for more details).

- We show that FCC games always have equilibria, and the problem of finding an equilibrium is in PPAD. We also show, however, finding equilibria is PPAD-hard even when using a simple numerical sum-of-distances utility function. Our reduction is from the PPAD-hard problem Preference Games [11].

1.2 Related work

In the last decade there has been a tremendous flowering of research at the intersection of game theory and computer science [17]. In a seminal paper [19] Papadimitriou laid the groundwork for algorithmic game theory by defining syntactically defined subclasses of FP with complete problems, PPAD being a notable such subclass. Recently, in a major breakthrough 2-player Nash Equilibrium was shown to be PPAD-complete [7, 5]. The term PPAD-complete is coming to occupy a role in algorithmic game theory analogous to the term NP-complete in combinatorial optimization [8].

Selfish caching games were introduced in [6] who considered the uncapacitated case where nodes could store more pieces of content by paying for the additional storage. We believe that limits on cache-capacity model an important real-world restriction and hence our focus on the capacitated version which was left as an open direction by [6]. Special cases of the integral version of CC Caching Games have been studied. In [14], Nash equilibria were shown to exist for weight-based object preferences where nodes are equidistant from one another and a special server holds all objects. [20] slightly extends [14] to the case where special servers for different objects are at different distances. Our results generalize and completely subsume all these prior cases of CC Games. The Market sharing games defined by[9] also consider caches with capacity, but are of a very special kind; unlike CC games, market sharing games are a special case of congestion games.

There has been considerable research on capacitated caching, viewed as an optimization problem. Various centralized and distributed algorithms have been presented for different classes of networks in [1, 2, 15, 12, 25].

2 Framework for CC games

We consider a network consisting of a set V of nodes labeled 1 through $n = |V|$ sharing a collection O of unit-size objects. Each node i has a total preorder $\geq_i$ among all the nodes in V[1]; $\geq_i$ further satisfies $i \geq_i j$ for all $i, j \in V$. We say that a node i *prefers* j over k if $j \geq_i k$, and call a node j *most i-preferred* in a set S of nodes if $j \geq_i k$ for all k in S. We also use the notation $j =_i k$ if $j \geq_i k$ and $k \geq_i j$, and $j >_i k$ if $j \geq_i k$ and it is not the case that $k \geq_i j$.

Each node i has a cache to store a certain number of objects. The placement at a node i 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 $P_i \subseteq$ O represents a feasible placement at node i. For notational convenience, we use P_{-i} to denote the collection $(P_j : j \in V \setminus \{i\})$, thus often using $P = (P_i, P_{-i})$ to refer to a global placement. We also assume that V includes a (server) node that has the capacity to store all objects. This ensures that at least one copy of every object is present in the system; also, this assumption can be made without loss of generality since the transformation of setting this server as the minimum element in each node's preference order leaves the problem unchanged.

Utility preference relations. In our game-theoretic model, each node attaches a utility to each global placement. We present a general definition that allows us to consider a large class of utility functions simultaneously. Rather than define a numerical utility function, we present the utility at each node i as a total preorder $\succeq_i$ among the set of all global placements. (The notation $\succ_i$ and $=_i$ over global placements are defined analogously.) We require that $\succeq_i$, for each $i \in V$, satisfies the following two basic conditions.

- **Monotonicity**: Let P and Q be two global placements such that for each object α and any 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 (P_i, P_{-i}) and (Q_i, Q_{-i}) denote two global placements such that for each object $\alpha \in P_i \cup Q_i$, if p (resp., q) is the most i-preferred node in $V \setminus \{i\}$ for which $\alpha \in P_p$ (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 only a function of the set of most preferred other nodes that are together holding these objects. Thus, for instance, if a node i prefers to store α over β in a scenario with the property that 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 with the same property.

ICC **Games.** An ICC game is a tuple $(V, \mathrm{O}, \{\geq_i\}, \{\succeq_i\})$. The focus of this paper is on the existence and computation of *pure Nash equilibrium* (henceforth, referred to simply as *equilibrium*) of the CC games we define. An equilibrium for an ICC game instance is a global placement P such that for each $i \in V$ there is no placement Q_i such that $(Q_i, P_{-i}) \succeq_i (P_i, P_{-i})$.

Generality of the conditions. We note that many standard utility functions defined for replica placement problems [6, 13, 20] satisfy the monotonicity and consistency conditions. For instance, suppose there is a cost function d such that d_{ij} is the cost incurred at i for accessing an object at j. Furthermore, suppose each node i has a weight $r_i(\alpha)$ for each object α. Consider the natural sum utility function $U_s(i)$ for each node i: $U_s(i)(P) = \sum_{\alpha \in \mathrm{O}} r_i(\alpha) \cdot d_{i\sigma_i(P,\alpha)}$, where $\sigma_i(P, \alpha)$ is the most i-preferred node holding α in P. Another natural utility function is the max utility function $U_m(i)$ given by $U_m(i)(P) = \max_{\alpha \in \mathrm{O}} r_i(\alpha) \cdot d_{i\sigma_i(P,\alpha)}$.

[1] A total preorder is a binary relation that satisfies reflexivity, transitivity, and totality. Totality means that for any i, j, k, either $j \geq_i k$ or $k \geq_i j$.

It is easy to verify that the above utility functions satisfy both 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 class of utility preference relations that we would like to highlight is the ones based on binary object preferences. Suppose each node i has a set S_i of objects in which it is equally interested, and it has no interest in the other objects. For any placement P, let $\sigma_i(P, \alpha)$ denote the most i-preferred node holding α in P. Let $\tau_i(P)$ denote the $|S_i|$-length sequence consisting of the $\sigma_i(P, \alpha)$, for $\alpha \in S_i$, in nonincreasing order according to the relation $\geq_i$. Then, the consistency condition can be further strengthened to the following.

- **Binary Consistency:** For any two placements $P = (P_i, P_{-i})$ and $Q = (Q_i, Q_{-i})$ with $P_{-i} = Q_{-i}$, $P \succeq_i Q$ if for $1 \leq k \leq |S_i|$, the kth component of $\tau_i(P)$ is at least as i-preferred as the kth component of $\tau_i(Q)$, then $P \succeq_i Q$.

Symmetric networks and acyclic node preference collections. As the sum and max utility functions suggest, node preference relations arise most naturally out of an underlying cost function d over ordered pairs of nodes: $j >_i k$ if $d_{ij} < d_{ik}$, and $j =_i k$ if $d_{ij} = d_{ik}$. We say that d is *symmetric* if $d_{ij} = d_{ji}$ for all i, j in V. We refer to the network over V as *symmetric* if the node preference relations are defined by a symmetric cost function. In the following lemma, whose proof is in Appendix A, we show that symmetric networks are equivalent to acyclic node preference collections. Formally, the collection $\{\geq_i : i \in V\}$ 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 1. *Any symmetric network yields an acyclic collection of node preferences. For any acyclic collection of node preferences, we can define symmetric access cost functions that are consistent with the node preferences.* □

Pair preference relations. Given any utility preference relation $\succeq_i$ that satisfies the monotonicity and consistency conditions, we now present a partial preorder $\sqsupseteq_i$ on $\mathrm{O} \times V \setminus \{i\}$, for each i. These partial preorders, referred to as pair preference relations, are very useful in our algorithm design and analyses.

1. For each object α and $j, k \neq i$, we have $(\alpha, j) \sqsupset_i (\alpha, k)$ whenever $k >_i j$, and $(\alpha, j) =_i (\alpha, k)$ whenever $k =_i j$.
2. Consider distinct objects α, β and nodes i, j, k with $j, k \neq i$. If *there exist* global placements $P = (P_i, P_{-i})$ and $Q = (Q_i, Q_{-i})$ such that $P \succ_i Q$, $P_i = \{\alpha\}$, $Q_i = \{\beta\}$, $P_{-i} = Q_{-i}$ and the most i-preferred node in $V \setminus \{i\}$ holding α (resp., β) in P_{-i} is j (resp., k), then we have $(\alpha, j) \sqsupset_i (\beta, k)$. If for *all* global placements $P = (P_i, P_{-i})$ and $Q = (Q_i, Q_{-i})$ such that $P \succeq_i Q$, $P_i = \{\alpha\}$, $Q_i = \{\beta\}$, $P_{-i} = Q_{-i}$ and the most i-preferred node in $V \setminus \{i\}$ holding α (resp., β) in P_{-i} is j (resp., k), then we have $(\alpha, j) \sqsupset_i (\beta, k)$.

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, $(\alpha, j) \sqsupset_i (\beta, k)$ means that if i needs to place either α or β in its cache, and j and k are most i-preferred nodes in $V \setminus \{i\}$ holding α and β, respectively, then i prefers to have α over β.

Lemma 2. *For each i, $\sqsupseteq_i$ as given above, is a well-defined preorder.*

Proof. We need to ensure the well-definedness of part 2 of the above definition. 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 $\succeq_i$. 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, k_j), $0 \le j < \ell$ for some integer $\ell > 1$, such that $(\alpha_j, k_j) \sqsupset_i (\alpha_{j+1 \bmod \ell}, k_{j+1 \bmod \ell})$ for $0 \le j < \ell$. To see this, it is sufficient to note that if $(\alpha, k) \sqsupset_i (\alpha', k')$ then for all placements P and P' such that $P_{-i} = P'_{-i}$ and the most i-preferred node in $V \setminus \{i\}$ that holds α (resp., α') is k (resp., k') we have $P \succ_i P'$. So any cycle in the strict preorder $\sqsupset_i$ implies a cycle in $\succ_i$, yielding a contradiction. □

For many utility preference relations, the pair preference relations are easy to derive directly from input parameters. For instance, for sum and max utilities, we have $(\alpha, j) \sqsupset_i (\beta, k)$ if $r_i(\alpha)d_{ij} > r_i(\beta)d_{ik}$, where $r_i(\alpha)$ is the weight of node i for object α. Note that this definition, in fact, extends $\sqsupseteq_i$ to be a total preorder.

Using the pair preference relation, there is a natural way to describe the best response P_i of a node i in response to a placement P_{-i} of $V \setminus \{i\}$: P_i is simply the object α that maximizes $(\alpha, \sigma_i(P_{-i}, \alpha))$ according to $\sqsupseteq_i$ (here $\sigma_i(P_{-i}, \alpha)$ is a most i-preferred node holding α in P_{-i}).

Unit cache capacity. In this paper, 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 $i_1, i_2, \ldots, i_{c_i}$. We set the node preferences as follows: $j_k =_{i_k} j_\ell$ for all $i, j \in V$, $1 \le k, \ell \le c_i$; $i_{k_1} \ge_{j_k} i'_{k_2}$ whenever $i \ge_j i'$.

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 $P_i = \cup_{1 \le k \le c_i} P'_{i_k}$. This mapping naturally defines the utility preference relations for the node set V'. In particular, for any $i \in V$ and $1 \le k \le c_i$, $P' \succeq_{i_k} Q'$ whenever $f(P') \succeq_i f(Q')$. 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. Thus, in the remainder of the paper, we assume that every node in the network stores at most one object in its cache.

3 Non-Existence of equilibria in ICC games and the associated decision problem

It is relatively easy to show that general ICC games may not have equilibria. In this section, we identify the most basic ICC games where equilibria may not exist, and study the complexity of the associated decision problem (i.e. of determining the existence of equilibria).

3.1 NP-Completeness

The main result of this section is the following theorem, which establishes the NP-hardness of ICC games even when the number of objects is small or the object preferences are binary. The hardness proofs are by a polynomial-time reduction from 3SAT [8]. 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 so as to capture the given 3SAT formula. The full proof, which involves several case analyses, is tedious and is deferred to Appendix B.

Theorem 3. *The problem of determining whether a given* ICC *instance has an equilibrium is in* NP. *It is NP-hard to determine whether an* ICC *intance has an equilibrium even if one of these three restrictions* *hold: (a) number of objects is two; (b) object preferences are binary and number of objects is three; (c) network is symmetric and number of objects is three.*

3.2 Binary preferences over two objects

Consider an arbitrary ICC game with two objects and binary preferences. The question is: given an instance of such a game does it possess an equilibrium? Let us call this problem 2BIN. We prove by a series of reductions that 2BIN is polynomial-time equivalent to the notorious EVEN-CYCLE problem [26]: does a given digraph contain an even cycle? The proof is presented in a series of lemmas in Appendix C.

Theorem 4. EVEN-CYCLE *on general digraphs is polynomial-time equivalent to* 2BIN. □

Despite intensive efforts over a long time the status of EVEN-CYCLE was in limbo [3] – it had neither been shown to be in P nor NP-complete – until [16, 22] provided a tour de force polynomial-time algorithm. This result thus also places 2BIN in P.

4 Existence and computation of equilibria in ICC games

In this section, we establish the existence of equilibria for several versions of ICC games. Our main result is a polynomial time construction of equilibria for hierarchical networks (Section 4.1). We next show that for symmetric networks and binary object preferences, the resulting ICC game is a potential game (Section 4.2). Finally, when there are only two objects in the system, we give a polynomial-time construction of equilibria for ICC games for symmetric networks (Section 4.3).

4.1 Hierarchical networks

In this section, we show that equilibria always exist for ICC games on hierarchical networks. Our proof is by means of a polynomial-time construction of an equilibrium. In particular, this implies that if the distances between nodes form an ultrametric (that is, $d_{ik} \leq \max\{d_{ij}, d_{jk}\}$ for all $i, j, k \in V$) then equilibria always exist and can be found in polynomial time.

Hierarchical networks. We formally define hierarchical networks and associated node preference relations. Let T denote a tree whose set of leaves is the node set V. The node preference relation $\geq_i$ is then given as follows: $j \geq_i k$ if the least common ancestor of i and j is a descendant of the least common ancestor of i and k. We use lca(i, j) to denote the least common ancestor of nodes i and j.

Recall that for each node i, its node preference relation $\geq_i$ and placement preference relation $\succeq_i$ induces a preorder $\sqsupseteq_i$ over object-node pairs. For the proof, it is more convenient to work with a preorder $\sqsupseteq_i$ among elements of O $\times$ I, where I is the set of internal nodes in T. Specifically, we define $(\alpha, v) \sqsupset_i (\beta, w)$ if there exist nodes j and k in V such that v is lca(i, j) and k is lca(i, k) and $(\alpha, j) \sqsupset_i (\beta, k)$. Since $j =_i \ell$ whenever lca$(i, j) =$ lca(i, ℓ), the preceding definition of $\sqsupseteq_i$ is well-defined.

Auxiliary servers. In order to present our algorithm, we introduce the notion of an *auxiliary server*. For an object α, an *auxiliary α-server* is a new node that will store α in every equilibrium; an auxiliary α-server prefers storing α over any other object. Like the nodes in V, each auxiliary server is introduced as a leaf in the hierarchy T. The node preference relations for all nodes in V are extended, using the hierarchy T, to include the auxiliary servers.

We introduce a number of auxiliary servers at the start of the algorithm. At each step of the algorithm, we maintain the invariant that the current global placement is an equilibrium in this enhanced hierarchy. As the algorithm proceeds, the set of auxiliary servers and their locations changes as we remove existing auxiliary servers or add new ones. On termination, there are no auxiliary servers leaving us with a desired equilibrium. Let W_t denote the set of auxiliary servers at the start of step t of the algorithm.

The algorithm. To distinguish between the placements on the nodes and the auxiliary servers, we represent a global placement as a pair $\langle P, Q\rangle$, where P (resp., Q) denotes the collection of placements at the nodes in V (resp., auxiliary servers). For any i (node in V or auxiliary server), placement $\langle P, Q\rangle$ and object α, let $\sigma_i(\langle P, Q\rangle, \alpha)$ denote lca(i, j) where j is a most i-preferred node holding α in $\langle P, Q\rangle$; in other words, $\sigma_i(\langle P, Q\rangle, \alpha)$ is the lowest node v in T such that i is a descendant of v and there exists another descendant j of v that holds α in $\langle P, Q\rangle$. Let $\mu_i(\langle P, Q\rangle) = (\alpha, v)$ where α is the object held by i in $\langle P, Q\rangle$ and v is $\sigma_i(\langle P, Q\rangle, \alpha)$.

Initialization. We add, for each object α and for each internal node v of T, an auxiliary α-server as a leaf child of v; this constitutes the set W_0.

Step t of algorithm. Fix an equilibrium $\langle P, Q\rangle$ for the node set $V \cup W_t$. If W_t is empty, then we are done. Otherwise, select a node j in W_t. Let $Q_j = \{\alpha\}$, and let $\mu_j(\langle P, Q\rangle) = (\alpha, v)$. Let S denote the set of all nodes $i \in V$ such that $(\alpha, v) \sqsupset_i \mu_i(\langle P, Q\rangle)$. We now describe one step of the algorithm which computes a new set of auxiliary servers W_{t+1} and a new placement $\langle P', Q'\rangle$ such that $\langle P', Q'\rangle$ is an equilibrium for $V \cup W_{t+1}$. We consider two cases.

- S is empty: In this case, we set P' to P. We remove the auxiliary server j from W_t and leave the placement in the remaining auxiliary servers as before. Thus $W_{t+1} = W_t - \{j\}$ and Q' is the same as Q except that Q'_j is no longer defined.
- S is nonempty: Let i denote a node in S such that lca(i, j) is lowest among all nodes in S. We set $P'_i = Q_j$ and $P'_{-i} = P_{-i}$. We remove the auxiliary α-server j from W_t and instead add a new auxiliary P_i-server k as a leaf sibling of i in T; $W_{t+1} = W_t + \{k\} \setminus \{j\}$.

Lemma 5. *Consider step t of the algorithm. Let object α and node v be as defined above. If $\langle P, Q\rangle$ is an equilibrium for $V \cup W_t$, then the following hold.*

1. *Every node plays their best response in $\langle P', Q'\rangle$; that is, $\langle P', Q'\rangle$ is an equilibrium for $V \cup W_{t+1}$.*
2. *For all $i \in V$, $\mu_i(\langle P', Q'\rangle) \sqsupseteq_i \mu_i(\langle P, Q\rangle)$.*
3. *Either $|W_{t+1}| < |W_t|$ or there exists a node i in V such that $\mu_i(\langle P', Q'\rangle) \sqsupset_i \mu_i(\langle P, Q\rangle)$.*

Proof. We first note that for any node k not in subtree rooted at the child of v that has j as a descendant $P'_k = P_k$, and for every object α, $\sigma_k(\langle P', Q'\rangle, \alpha) = \sigma_k(\langle P, Q\rangle, \alpha)$. This establishes statement 1 of the lemma for the node. Furthermore, if P_k is a best response of k in $\langle P, Q\rangle$, then P'_k is a best response of k in $\langle P', Q'\rangle$. This establishes statement 2 of the lemma for the node.

It remains to consider nodes rooted at the child of v that has j as a descendant. Let k be such a node. Note that for any $\beta \neq \alpha$, $\sigma_k(\langle P', Q'\rangle, \beta) = \sigma_k(\langle P, Q\rangle, \beta)$, since there is no change in location of β. If $P_k = \{\beta\}$ and $\beta \neq \alpha$, then we have $P'_k = \{\beta\}$ and $\mu_k(\langle P', Q'\rangle) =_i \mu_k(\langle P, Q\rangle)$ since there is no change in location of β. Thus, statement 2 of the lemma holds for k.

We consider two cases, according to whether S is empty. If S is not empty and hence the node i exists, let x denote lca(i, j). If k is not in the subtree rooted at the child of x that has j as descendant, the most preferred node holding α has not changed. Then, $\sigma_k(\langle P', Q'\rangle, \alpha) = \sigma_k(\langle P, Q\rangle, \alpha)$, implying that both statements 1 and 2 of the lemma hold for k. If k is in the subtree rooted at the child of x that has j as descendant, we have

$$\mu_k(\langle P, Q\rangle) \sqsupseteq_i (\alpha, v) \sqsupseteq_i (\alpha, x).$$

This implies that $\mu_k(\langle P', Q'\rangle) \sqsupseteq_k \mu_k(\langle P, Q\rangle)$ and that k is playing its best response. Thus, $\langle P', Q'\rangle$ is an equilibrium. Furthermore, for node i, $\mu_i(\langle P', Q'\rangle) \sqsupset_i \mu_i(\langle P, Q\rangle)$, establishing statement 3 of the lemma.

If S is empty, then the placement on the nodes in V remains the same as before. Since there is no node i such that $(\alpha, v) \sqsupseteq_i \mu_i(\langle P, Q\rangle)$ and the location of none of the objects has changed, other than the removal of α from v, it follows that for all $k \in V$, $\mu_k(\langle P', Q'\rangle) \sqsupseteq_k \mu_k(\langle P, Q\rangle)$, establishing statement 1 of the lemma. Since $\langle P, Q\rangle$ is an equilibrium, so is $\langle P', Q'\rangle$, statement 2 of the lemma holds. Since $|W_{t+1}| = |W_t| - 1$, the statement 3 of the lemma holds. This completes the proof of the lemma. □

Theorem 6. *For hierarchical node preferences, an equilibrium can be found in polynomial time.*

Proof. It is immediate from the definition of the algorithm and Lemma 5 that at termination, the algorithm returns a valid equilibrium. It remains to show that our algorithm terminates in polynomial time. Consider the potential given by the sum of $|W_t|$ and the sum, over all i, of the position of $\mu_i(\langle P, Q, \rangle)$ in the total preorder $\sqsupseteq_i$. The term $|W_0|$ is at most nm, where n is $|V|$ (which is at least the number of internal nodes) and m is the number of objects. Furthermore, since $|\mathrm{O} \times I|$ is at most nm, the initial potential is initially at most $nm + n^2m$. By Lemma 5, the potential decreases by at least one in each step of the algorithm. Thus, the number of steps of the algorithm is at most $nm + n^2m$. Furthermore, each step can be implemented in time linear in nm. This completes the proof of the theorem. □

4.2 Symmetric networks with binary object preferences

Let d be a symmetric cost function for a network over the node set V. Recall that for binary object preferences, we are given, for each node i a set S_i of objects in which i is equally interested. Our proof of existence of equilibria is via a potential function argument. Given a placement P, 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 following 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: 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 symmetric networks with binary object preferences, the resulting ICC game is a potential game, and hence also in PLS.

4.3 Symmetric networks with two objects

We give a polynomial-time algorithm for computing an equilibrium in any symmetric network with two objects. Our algorithm starts by introducing auxiliary servers for both the objects in the network at zero cost from each node. In each subsequent step, we move the auxiliary servers progressively "further" away, ensuring that each instant, we have an equilibrium. Finally, when the auxiliary servers are at least preferred cost from all the nodes, they can be removed yielding an equilibrium for the original network. We refer to Appendix D for details.

5 Fractional caching games

We introduce a new class of caching games where the nodes can store fractions of objects, as opposed to whole objects, and a node can 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 to constitute the whole object as long as the fractions add up to at least one. Such fractional caching scenarios naturally arise when objects are encoded and potentially distributed within a network to permit both efficient and reliable access.

Several implementations of fractional caching, in fact, already exist. For instance, fountain codes [4, 24] and the information dispersal algorithm [21] 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 caching (FCC) games, which are an adaptation of the game-theoretic framework developed in Section 2 to fractional caching. We have a set V of nodes sharing a set O of objects. In the fractional caching game, the strategies are *fractional placements*; a fractional placement $\widetilde{P}$ is a $|V|$-tuple $\{\widetilde{P}_i : i \in V\}$ where $\widetilde{P}_i : \mathrm{O} \to \Re$ under the constraint that sum of $\widetilde{P}_i(\alpha) \cdot \mathrm{size}(\alpha)$, over all α in O, is at most the cache size of i (here size(α) is the size of object α).

We begin by presenting FCC 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 a node-object weights $r_i(\alpha)$, $i \in V$, $\alpha \in \mathrm{O}$. Given a fractional global placement $\widetilde{P}$, we define the cost incurred by i for accessing object α as the minimum value of $x_j d_{ij}$ subject to 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 FCC 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 $\mathrm{O}\times(V\setminus\{i\})$ (see Section 2). For FCC 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 $\mu_{i,\widetilde{P},\alpha} : V \setminus \{i\} \to \Re$ 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 $b_{i,\widetilde{P}} : \mathrm{O} \times (V \setminus \{i\}) \to \Re$ 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 FCC games.

5.1 Existence of equilibria in FCC games

Using standard fixed-point machinery, we show that every FCC game has an equilibrium (Theorem 20). We also show that finding equilibria in FCC games is in PPAD (Theorem 21). Our proof is by a reduction to the Fractional Stable Paths Problem (FSPP) which was introduced in [10] and shown to be PPAD-complete in [11]. Due to space constraints, both these proofs have been placed in Appendix E.

5.2 PPAD-Hardness

We now show that finding an equilibrium in FCC games is PPAD-hard even when the underlying cost function d is a metric; our proof is by a reduction from preference games, introduced in [11]. Given a preference game G with n players $1, 2, \ldots, n$ and their preferences given by $\geq_i$, we construct a fractional caching 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: $w_{ij} = P_i(\alpha_j)$. We refer to Appendix E for a proof of the following lemma.

Lemma 7. *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.*

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 FCC implying that the latter is PPAD-hard.

6 Concluding remarks

In this paper we have defined integral and fractional caching games in networks, where the cache capacity of each node is bounded. Under the assumption that all objects are of uniform size, we have almost completely characterized the complexity of integral capacitated caching games: For what classes of games do equilibria exist? Can we determine efficiently whether they exist? When they do exist, can we efficiently find them? One complexity question that is still open is the case of symmetric networks with binary preferences. We conjecture that finding equilbria in such games (which we prove are potential games) is PLS-hard.

We showed that fractional capacitated caching 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.

References

[1] E. Angel, E. Bampis, G. G. Pollatos, and V. Zissimopoulos. Optimal data placement on networks with constant number of clients. *CoRR*, abs/1004.4420, 2010.

[2] I. D. Baev, R. Rajaraman, and C. Swamy. Approximation algorithms for data placement problems. *SIAM J. Comput.*, 38(4):1411–1429, 2008.

[3] M. Brundage. From the even cycle mystery to the l-matrix problem and beyond. Thesis, 1996.

[4] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. In *SIGCOMM '98: Proceedings of the ACM SIGCOMM '98 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 56–67, New York, NY, USA, 1998. ACM.

[5] X. Chen, X. Deng, and S.-H. Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM (JACM)*, 56(3), 2009.

[6] B.-G. Chun, K. Chaudhuri, H. Wee, M. Barreno, C. H. Papadimitriou, and J. Kubiatowicz. Selfish caching in distributed systems: a game-theoretic analysis. In *PODC*, pages 21–30, 2004.

[7] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *In Proceedings of the thirty-eighth annual ACM Symposium on Theory of Computing*, pages 71–78, 2006.

[8] M. R. Garey and D. S. Johnson. *Computers and intractability.* Freeman Press, 1979.

[9] M. X. Goemans, L. Li, V. S. Mirrokni, and M. Thottan. Market sharing games applied to content distribution in ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 24(5):1020–1033, 2006.

[10] P. E. Haxell and G. T. Wilfong. A fractional model of the Border Gateway Protocol (BGP). *SODA*, pages 193–199, 2008.

[11] S. Kintali, L. J. Poplawski, R. Rajaraman, R. Sundaram, and S.-H. Teng. Reducibility among fractional stability problems. *FOCS*, 2009.

[12] M. Korupolu, C. G. Plaxton, and R. Rajaraman. Placement algorithms for hierarchical cooperative caching. *Journal of Algorithms*, 38:260–302, 2001. Special issue devoted to selected papers from SODA 1999.

[13] N. Laoutaris, G. Smaragdakis, K. Oikonomou, I. Stavrakakis, and A. Bestavros. Distributed placement of service facilities in large-scale networks. In *INFOCOM*, pages 2144–2152, 2007.

[14] N. Laoutaris, O. Telelis, V. Zissimopoulos, and I. Stavrakakis. Distributed selfish replication. *IEEE Trans. Parallel Distrib. Syst.*, 17(12):1401–1413, 2006.

[15] A. Leff, J. L. Wolf, and P. S. Yu. Replication algorithms in a remote caching architecture. *IEEE Transactions on Parallel and Distributed Systems*, 4:1185–1204, 1993.

[16] W. McCuaig, N. Robertson, P. D. Seymour, and R. Thomas. Permanents, pfaffian orientations, and even directed circuits (extended abstract). In *STOC*, pages 402–405, 1997.

[17] N. Nisan, T. Roughgarden, É. Tardos, and V. V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.

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

[19] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994.

[20] G. G. Pollatos, O. Telelis, and V. Zissimopoulos. On the social cost of distributed selfish content replication. In *Networking*, pages 195–206, 2008.

[21] M. O. Rabin. Efficient dispersal of information for security, load balancing and fault tolerance. *Journal of the ACM*, 36:335–348, 1989.

[22] N. Robertson, P. D. Seymour, and R. Thomas. Permanents, pfaffian orientations, and even directed circuits. *Annals of Mathematics*, pages 929–975, 1999.

[23] A. Schrijver. *Combinatorial Optimization (3 Volume)*. Springer-Verlag Berlin Heidelberg, 2003.

[24] A. Shokrollahi. Raptor codes. In *IEEE Transactions on Information Theory*, pages 2551–2567, 2006.

[25] O. Wolfson, S. Jajodia, and Y. Huang. An adaptive data replication algorithm. *ACM Transactions on Database Systems*, 22:255–314, 1997.

[26] D. H. Younger. Graphs with interlinked directed circuits. In *Proceedings of Midwestern Symposium on Circuit Theory*, volume 2, pages XVI2.1–XVI2.7, 1973.

A Proof for model formulation

Proof of Lemma 1: 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.

Given an acyclic collection of node preferences, 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 symmetric network. □

B NP-hardness proofs

Theorem 8. *The problem of determining whether a given* ICC *instance has an equilibrium is in* NP. *It is NP-hard to determine whether an* ICC *intance has an equilibrium even if one of these three restrictions hold: (a) number of objects is two; (b) object preferences are binary and number of objects is three; (c) network is symmetric and number of objects is three.*

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 an ICC instance as follows. For each variable x_i in ϕ, we introduce two variable nodes: node X_i and $\bar{X}_i$. We set $d_{X_i,\bar{X}_i}$ and $d_{\bar{X}_i,X_i}$ to be 0.5. 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 clause c_j in formula ϕ, we set $d_{C_j,L_{j,r}}$ and $d_{L_{j,r},C_j}$ to be 1 for $r \in \{1,2,3\}$. Note that each $L_{j,s}$, for $j \in [1,k]$, and $s \in \{1,2,3\}$ is in fact some variable node X_h or $\bar{X}_h$ for some $h \in [1,n]$. We also introduce a gadget G illustrated in Figure 2, consisting of nodes S, A, B, and C. We set the distances d_{S,C_i} and the symmetric $d_{C_i,S}$, for all $1 \leq i \leq k$ between node S and all clause nodes to be 2. The general construction is illustrated in Figure 1.

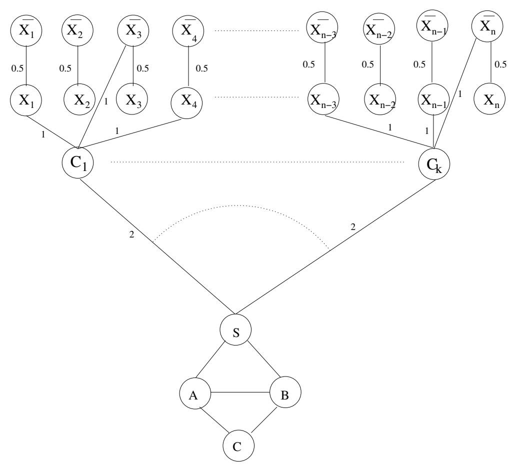

Figure 1: Instance of the construction for the metric case proof of NP-Hardness, where $\phi = (x_1 \vee \bar{x_3} \vee x_4) \wedge \ldots \wedge (x_{n-3} \vee x_{n-1} \vee \bar{x_n})$

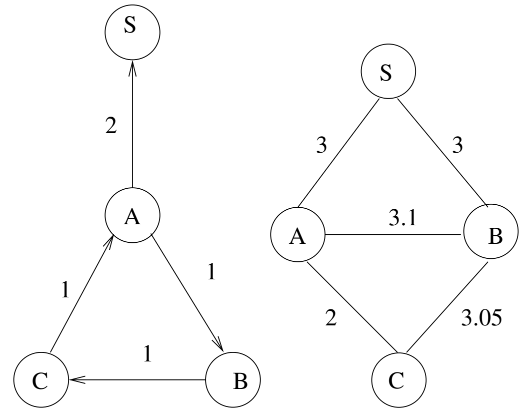

Figure 2: Gadget for the non-metric (left) and the metric (right) case

Asymmetric networks with two objects. We set $d_{A,S}$, $d_{A,B}$, $d_{B,C}$, $d_{C,A}$ to be 1. The server node at access cost $d_{srv} = 10$ from all nodes in V, stores a fixed copy of two objects α and β. We set the weights $r_{x_i}(\alpha)$, $r_{\bar{x_i}}(\alpha)$, $r_{x_i}(\beta)$, $r_{\bar{x_i}}(\beta)$ of each variable node on objects α and β, to be 1. For each clause node we set the weight $r_{C_i}(\alpha)$ on object α to be 1, and the weight $r_{C_i}(\beta)$ on object β to be 0.7, for all $1 \leq i \leq k$. We set the weight $r_S(\alpha)$ on object α to be 1, and the weight $r_S(\beta)$ on object β to be 0.7. We set the weight $r_A(\alpha)$ of node A on object α to be 0.7 and the weight $r_A(\beta)$ of node A on object β to be 1. Finally, we set the weight of nodes B and C on both objects α, and β to be 1. We refer to this ICC instance as I_1.

Symmetric networks with three objects. For the metric case, we set $d_{A,S}$ and $d_{B,S}$ to be 3, $d_{A,B}$ to be 3.1, $d_{B,C}$ to be 3.05, and $d_{C,A}$ to be 2; while symmetry holds. The server node, which is at access cost $d_{srv} = 5$ from all nodes in V, stores a fixed copy of three objects α, β, and γ. We set the weights $r_{x_i}(\alpha)$, $r_{\bar{x_i}}(\alpha)$, $r_{x_i}(\beta)$, $r_{\bar{x_i}}(\beta)$ of each variable node on objects α and β, to be 1. Also, for each clause node we set the weight $r_{C_i}(\alpha)$ on object α to be 0.85, and the weight $r_{C_i}(\beta)$ on object β to be 1, for all $1 \leq i \leq k$. We set the weight $r_S(\alpha)$ on object α to be 0.85, and the weight $r_S(\beta)$ on object β to be 1. Also we set the weight $r_A(\alpha) = 1$, $r_A(\gamma) = 2$, $r_B(\beta) = 1$, $r_B(\gamma) = 0.9837$, $r_C(\beta) = 1$, and $r_C(\gamma) = 1.6$. We set all the remaining weights to be 0. We refer to this ICC instance as I_2.

Claim 1. *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 the most X_i-preferred (resp., $\bar{X}_i$-preferred) node, and both X_i and $\bar{X}_i$ are interested equally in α and β. □

Claim 2. *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 the most C_i-preferred nodes holding β. By Claim 1 we know that nodes $\bar{L_{i,j}}$, for $j \in \{1, 2, 3\}$ hold α, and they are the most C_i-preferred 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 $r_{C_i}(\beta)d_{C_i,L_{i,j}} = 1 \times 1 = 1$; while the cost for holding β and accessing α from $\bar{L_{i,j}}$, for $j \in \{1, 2, 3\}$, is $r_S(\alpha)d_{C_i,\bar{L_{ij}}} = 0.85 \cdot 1.5 = 1.275$. 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 the most C_i-preferred nodes holding α. Also, by Claim 1, the most C_i-preferred 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 $r_{C_i}(\alpha)d_{C_i,L_{i,j}} = 0.85 \cdot 1 = 0.85$; while the cost for holding α and accessing β from node $\bar{L_{i,j}}$ (resp., $L_{i,j}$), is $r_{C_i}(\beta)d_{C_i,\bar{L_{i,j}}} = 1 \cdot 1.5 = 1.5$ (resp., $r_{C_i}(\beta)d_{C_i,L_{i,j}} = 1 \cdot 1 = 1$). Obviously, in any case node C_i prefers to replicate β. □

Claim 3. *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 the most S-preferred nodes holding β. Also by Claim 2, the most S-preferred 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 $r_S(\beta)d_{S,C_i} = 1 \cdot 2 = 2$; while the cost for holding β and accessing α from $L_{i,j}$, where $i \in [1, k], j \in \{1, 2, 3\}$, is $r_S(\alpha)d_{S,L_{i,j}} = 0.85 \cdot 3 = 2.55$. Obviously, node S prefers to replicate α.

Now assume that at least one of $C_1, \ldots, C_k$ holds α. These nodes are the most S-preferred node holding α. Also the most S-preferred node holding β, due to Claim 2 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 $r_S(\alpha)d_{S,C_i} = 0.85 \cdot 2 =$; while the cost for holding α and accessing β from a node L_{ij}, where $j \in \{1, 2, 3\}$, is $r_S(\beta)d_{S,L_{i,j}} = 1 \cdot 3 = 3$. Obviously, in any case node S prefers to replicate β. □

Theorem 9. *The* ICC *instance I_1 has an equilibrium if and only if node S holds object α.*

Proof. First, assume that S is holding α. By Claim 3 nodes $C_1, \ldots, C_k$ hold object β, and by Claim 2 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 the most A-preferred node S, is $r_A(\alpha)d_{A,S} = 0.7 \cdot 1 = 0.7$; while the cost for holding object α and accessing β from the most A-preferred node B, is $r_A(\alpha)d_{A,B} = 1 \cdot 1 = 1$. Node B does not want to deviate since its cost for holding object β and accessing α from the most B-preferred node C, is $r_B(\alpha)d_{B,C} = 1 \cdot 1 = 1$; while the cost for holding object α and accessing β from the most B-preferred node A, is $r_B(\beta)d_{B,A} = 1 \cdot 2 = 2$. Node C does not want to deviate since its cost for holding object α and accessing β from the most C-preferred node A, is $r_C(\beta)d_{C,A} = 1 \cdot 1 = 1$; while the cost for holding object β and accessing α from the most C-preferred node S, is $r_C(\alpha)d_{C,S} = 1 \cdot 2 = 2$. 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 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 β from the most B-preferred (resp., C-preferred) node S, is $r_B(\beta)d_{B,S} = 1 \cdot 3 = 3$ (resp., $r_C(\beta)d_{C,S} = 1 \cdot 2 = 2$); while the cost for holding object β and accessing α from the most B-preferred node A, is $r_B(\beta)d_{B,A} = 1 \cdot 2 = 2$ (resp., $r_C(\beta)d_{C,A} = 1 \cdot 1 = 1$).

- 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 the most A-preferred node S is $r_A(\beta)d_{A,S} = 1 \cdot 1 = 1$; while the cost for holding β and accessing α from the most A-preferred node B is $r_A(\alpha)d_{A,B} = 0.7 \cdot 1 = 0.7$. In the case where A and C hold α, then C wants to deviate since the cost while holding α and accessing β from the most C-preferred node B is $r_C(\beta)d_{C,B} = 1 \cdot 2 = 2$; while the cost for holding β and accessing α from the most C-preferred node A is $r_C(\alpha)d_{C,A} = 1 \cdot 1 = 1$. In the case where B and C hold α, B wants to deviate since the cost while holding α and accessing β from the most B-preferred node A is $r_B(\beta)d_{B,A} = 1 \cdot 2 = 2$; while the cost for holding β and accessing α from the most B-preferred node C is $r_B(\alpha)d_{B,C} = 1 \cdot 1 = 1$.

- 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 the most B-preferred (resp., C-preferred, or A-preferred) node A (resp., B, or C) is $r_B(\alpha)d_{B,A} = 1 \cdot 2 = 2$ (resp., $r_C(\alpha)d_{C,B} = 1 \cdot 2 = 2$, or $r_A(\alpha)d_{A,C} = 0.7 \cdot 2 = 1.4$); while the cost for holding α and accessing β from the most B-preferred (resp., C-preferred, or A-preferred) node C (resp., A, or B), is $r_B(\beta)d_{B,C} = 1 \cdot 1 = 1$ (resp., $r_C(\beta)d_{C,A} = 1 \cdot 1 = 1$, or $r_A(\beta)d_{A,B} = 1 \cdot 1 = 1$).

- 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 the most A-preferred node C_i, for some $i \in [1, k]$, is $r_A(\beta)d_{A,C_i} = 1 \cdot 3 = 3$; while the cost for holding β and accessing α from the most A-preferred node S is $r_A(\alpha)d_{A,S} = 0.7 \cdot 1 = 0.7$. Similar proof holds for nodes B and C.

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

Theorem 10. *The* ICC *instance I_2 has an equilibrium if and only if node S holds object α.*

Proof. First, assume that S is holding α. By Claim 3 nodes $C_1, \ldots, C_k$ hold object β, and by Claim 2 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 $r_A(\alpha)d_{A,S} = 3$; while the cost for holding α and accessing γ from node C increases to $r_A(\gamma)d_{A,C} = 4$. Node B doesn't want to deviate since the cost for holding object β and accessing object γ from node C is $r_B(\gamma)d_{B,C} = 0.9837 \cdot 3.05 = 3.000285$; 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 $r_C(\beta)d_{C,B} = 3.05$; while the cost for holding object β and accessing γ from node A increases to $r_C(\beta)d_{C,A} = 3.2$.

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 $(r_A(\gamma)d_{A,B} = 6.2)$; 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 $(r_B(\beta)d_{B,C} = 3.05)$; while the cost for holding object β and accessing γ from node A decreases to $r_B(\gamma)d_{B,A} = 3.04947$.

- 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 $(r_C(\gamma)d_{C,A} = 3.2)$; while the cost for holding object γ and accessing β from node B decreases to $r_C(\beta)d_{C,B} = 3.05$.

- 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 $r_A(\gamma)d_{A,C} = 4$.

- 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 $(r_B(\gamma)d_{B,C} = 3.000285)$; while the cost for holding object γ and accessing β from node S decreases to $r_B(\beta)d_{B,S} = 3$.

- 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 $r_C(\gamma)d_{B,C} = 4.88$.

- 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 $r_B(\beta)d_{C,B} = 3.05$.

- 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 $r_A(\gamma)d_{A,C} = 4$.

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 $d_{C_i,K}$, for $i \in [1,k]$, between clause nodes and K to be 1.4, $d_{S,L}$ to be 2.1, and $d_{A,S}$, $d_{A,B}$, $d_{B,C}$, $d_{C,A}$ 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 $\{\alpha, \gamma\}$. 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.

Claim 1 holds as it is for the binary object preferences non-metric case.

Claim 4. *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 Claim 1 we know that nodes $\bar{L_{i,j}}$, for $j \in \{1,2,3\}$ hold α, and they are the most C_i-preferred nodes holding α; while the most C_i-preferred node holding γ is node K. Node's C_i cost for holding α and accessing γ from K is $d_{C_i,K} = 1.4$; 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 the most C_i-preferred nodes holding α; while again the most C_i-preferred 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 $d_{C_i,L_{i,j}} = 1$; while the cost for holding α and accessing γ from node K is $d_{C_i,K} = 1.4$. Obviously, node C_i prefers to replicate γ. □

Claim 5. *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 Claim 4, the most S-preferred node holding α is at least one of $L_{i,j}$ nodes, where $i \in [1,k], j \in \{1,2,3\}$; while the most S-preferred nodes holding β is node L. The cost for S holding α and accessing β from node L, is $d_{S,L} = 2.1$; while the cost for holding β and accessing α from $L_{i,j}$, where $i \in [1,k], j \in \{1,2,3\}$, is $d_{S,L_{i,j}} = 3$. Obviously, node S prefers to replicate α.

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

Theorem 11. *There exists an equilibrium for the* ICC *instance I_3 if and only if node S holds object α.*

Proof. First, assume that S is holding α. By Claim 5 nodes $C_1, \ldots, C_k$ hold object γ, and by Claim 4 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 the most A-preferred node S, is $d_{A,S} = 1$; while the cost for holding object α and accessing γ from the most A-preferred node B, is still $d_{A,B} = 1$. Node B does not want to deviate since its cost for holding object γ and accessing α from the most B-preferred node C, is $d_{B,C} = 1$; while the cost for holding object α and accessing γ from the most B-preferred node A, is still $d_{B,A} = 1$. Node C does not want to deviate since its cost for holding object α and accessing γ from the most C-preferred node A, is $d_{C,A} = 1$; while the cost for holding object γ and accessing α from the most C-preferred node S, is still $d_{C,S} = 1$. 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 the most B-preferred (resp., C-preferred) node C_i, for some $i \in [1,k]$ or from node K, is $d_{B,C_i} = 5$ or $d_{B,K} = 6.4$ (resp., $d_{C,C_i} = 4$ or $d_{C,K} = 5.4$); while the cost for holding object γ and accessing α from the most B-preferred node A, is $d_{B,A} = 2$ (resp., $d_{C,A} = 1$).

- 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 the most A-preferred node C is $d_{A,C} = 2$; while the cost for holding γ and accessing α from the most A-preferred node B is $d_{A,B} = 1$. 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 the most B-preferred node A is $d_{B,A} = 2$; while the cost for holding α and accessing γ from the most B-preferred node C, is $d_{B,C} = 1$. 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 the most A-preferred node C_i, for some $i \in [1, k]$, is $d_{A,C_i} = 3$; while the cost for holding α and accessing γ from the most A-preferred node B is $d_{A,B} = 1$. 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 ICC games (both metric and non-metric cases) (resp., for the binary object preferences, non-metric 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 Claim 1 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 Claim 2 (resp., Claim 4) 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 Claim 3 (resp., Claim 5), 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 9 and 10 (resp., 11), 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.

C Binary object preferences over two objects

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 12. 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 13. 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 [23]:

Lemma 14. *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 15. 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 16. 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 14), 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 17. *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 18. 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 17. We can determine such a stabel 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. □

D Proof for symmetric networks with two objects

Suppose we are given a symmetric 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 $1 \leq i < r$. In order to present our algorithm, we use the notion of the *auxiliary server* introduced in Section 4.1.

Auxiliary servers. For an object α, an auxiliary α-server is a new node that will store α in every equilibrium; an auxiliary α-server prefers storing α over any other object. We denote by $srv_\alpha(\ell)$ the auxiliary α-server 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 an auxiliary α-server $srv_\alpha(0)$ and β-server $srv_\beta(0)$ at access cost 0 from each node in V. We let nodes replicate their most preferred object and access the other without any access cost from the corresponding auxiliary server. 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 auxiliary servers $srv_\alpha(\ell_{t+1})$ 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 auxiliary α-server $srv_\alpha(\ell_t)$ 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 $\alpha \in P_j$, $srv_\alpha(\ell_t)$ whould not be the most i-preferred node holding α and thus i would not be affected by any change of auxiliary α-server. 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 auxiliary α-server $srv_\alpha(\ell_{t+1})$ and the auxiliary β-server $srv_\beta(\ell_t)$, i whould never be the j-most preferred node in P'.

We then remove the auxiliary β-server $srv_\beta(\ell_t)$ 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 19. *For symmetric networks with two objects, an equilibrium can be found in polynomial time.*

Proof. An initial placement P, where we have the set of auxiliary servers $srv_\alpha(0)$ 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. $\square$

E Proofs for FCC games

Theorem 20. *Every* FCC *instance has a pure Nash equilibrium.*

Proof. By [18], 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 FCC instance, the strategy space of each player i is simply the set of all its fractional placements: that is, the set of functions $f : \mathrm{O} \to [0, 1]$ subject to condition that $\sum_{\alpha \in \mathrm{O}} f(\alpha) \le c_i$, 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:

$$\max -\sum_{\alpha\in \mathrm{O}} r_i(\alpha)(\sum_{j\in V} x_{ij}(\alpha)d_{ij})$$

$$\sum_{j\in V} x_{ij}(\alpha) = 1 \qquad \text{for all } i\in V, \alpha\in \mathrm{O}$$

$$x_{ij}(\alpha) \le \widetilde{P}_j(\alpha) \quad \text{for all } i,j\in V, \alpha\in \mathrm{O}$$

$$x_{ij}(\alpha) \ge 0 \quad \text{for all } i,j\in V, \alpha\in \mathrm{O}$$

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. □

Theorem 21. *Finding an equilibrium in an* FCC *game is in* PPAD.

Proof. Our proof is by a reduction from FSPP (Fractional Stable Paths Problem), which is defined as follows [11]. 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, $\sum_{P\in\pi(v)} w_v(P) \le 1$, and (2) **Tree condition**: for each node v, and each path S with start node u, $\sum_{P\in\pi(v,S)} w_v(P) \le w_u(S)$.

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 a fractional caching game 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 $\alpha \in$ 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$(k, \alpha), d\rangle$ 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, α), 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$(i,k,\beta), d\rangle$ 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 FCC 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 FCC instance to feasible solutions in the FSPP instance such that (a) if $\widetilde{P}$ is an equilibrium for the FCC 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 FCC instance.

Let $\widetilde{P}$ denote any fractional placement of the FCC 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 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$(i,j,\alpha), d\rangle$ (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 FCC to FSPP, placing FCC in PPAD. □

Lemma 22 is immediate.

Lemma 22. *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 23. *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 22, 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 22, 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 $j \neq i$. □

Proof of Lemma 7: 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 23, 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 23 and the definition of $\omega(P)$, the best response of i in G matches that in the caching game; hence every player in V_1 also plays their best response in P, implying that P is an equilibrium for $\widehat{G}$. □