The Complexity of Equilibria in Cost Sharing Games

Vasilis Syrgkanis *

Department of Computer Science, Cornell University, Ithaca, NY 14853
vasilis@cs.cornell.edu

Abstract. We study Congestion Games with non-increasing cost functions (Cost Sharing Games) from a complexity perspective and resolve their computational hardness, which has been an open question. Specifically we prove that when the cost functions have the form $f(x) = c_r/x$ (Fair Cost Allocation) then it is PLS-complete to compute a Pure Nash Equilibrium even in the case where strategies of the players are paths on a directed network. For cost functions of the form $f(x) = c_r(x)/x$, where $c_r(x)$ is a non-decreasing concave function we also prove PLS-completeness in undirected networks. Thus we extend the results of [7, 1] to the non-increasing case. For the case of Matroid Cost Sharing Games, where tractability of Pure Nash Equilibria is known by [1] we give a greedy polynomial time algorithm that computes a Pure Nash Equilibrium with social cost at most the potential of the optimal strategy profile. Hence, for this class of games we give a polynomial time version of the Potential Method introduced in [2] for bounding the Price of Stability.

Keywords: Cost Sharing, PLS-completeness, Price of Stability, Congestion Games

1 Introduction

The rapid and overwhelming expansion of the Internet has transformed it into a completely new economic arena where a large number of self-interested players interact. The lack of central coordination has rendered classic optimization problems insufficient to capture Internet interactions and has given rise to new game-theoretic models. In this work we study a general such model, namely Congestion Games with non-increasing cost functions (Cost Sharing Games), from a complexity perspective and we resolve the computational hardness of computing a Pure Nash Equilibrium (PNE) in such games, which has been an open problem. The computational hardness is an important aspect of an equilibrium concept since it indicates whether it is a reasonable outcome in real world settings.

We start with a motivating example: a group of Internet Service Providers (ISP) wants to create a new network on a set of nodes (possibly different set for each provider). Each ISP's goal is that any two of his nodes are connected by a path. For practical reasons, each provider can build edges only between two nodes in his set and his clients can use a link only if the ISP helped build it. Moreover, we assume that ISPs are clever enough and when they decide to build the same link as others then they all build one link and share the cost. The moment we add this last specification, the problem faced by an ISP is no longer an optimization problem and the setting becomes a game which from now on we will call the ISP Network Creation Game. ISP Network Creation Games can be easily modeled as Cost Sharing Games.

Congestion Games in general has been a widely studied game theoretic model. In Congestion Games a set of players allocate some set of shared resources. The cost incurred from using a resource is a function of the number of players that have allocated the resource and the total cost of a player is the sum of his costs on all the resources he has allocated. A reasonable outcome of such a setting is a Pure Nash Equilibrium (PNE): a strategy profile such that no player

* Supported in part by NSF grant CCF-0729006

can profit from deviating unilaterally. In a seminal paper, Rosenthal [13] gives a proof that Congestion Games always possess a PNE. To achieve this, he introduces a potential function and shows that the change in the potential induced by a unilateral move of some player is equal to the change of that player's utility. Several aspects of the PNE of Congestion Games have been studied in the literature.

An interesting research area has been the complexity of computing a PNE in Congestion Games. In a seminal paper Fabrikant et al. [7] proved that the above problem is PLS-complete even in the case where the strategies of the players are paths in a directed network. Later, Ackermann et al. [1] extended the above result to the case of undirected networks with linear cost functions. However, both results use cost functions on the resources that are non-decreasing (delays) and do not carry over to Cost Sharing Games. The complexity of computing a PNE in Cost Sharing Games has been an open question.

Another interesting line of research has been measuring the inefficiency that arises from selfishness. An important concept in that direction (especially in the case of Cost Sharing Games) has been the Price of Stability (PoS), which is the ratio of the quality (sum of players' costs) of the best PNE over the socially optimal outcome ([2]). One major motivation for the PoS is that it is the socially optimal solution subject to the constraint of unilateral stability. If there was a third-party that could propose to players a solution to their problem, then the optimal stable solution he could propose would be the best PNE. This motivation raises an interesting open question: Given an upper bound on the PoS for a class of games, is there a polynomial-time algorithm for computing a PNE with cost comparable to that bound?

In this work we make significant progress in both directions described above. We prove the first PLS-hardness results for Cost Sharing Games. Our results show that the non-increasing case is not easier than the non-decreasing. Moreover, we give the first polynomial-time algorithm that computes a PNE with quality equal to the known bound on the PoS for a significant class of Cost Sharing Games that contains, for example, the ISP Network Creation Game.

Results

- Our first main result is that a greedy approach leads to a polynomial time algorithm that computes a PNE of any Matroid Cost Sharing Game, with cost equal to the potential of the socially optimal solution. The quality of such a PNE is no worse than any bound on the PoS that can be proved via the Potential Method. Hence, for this class we give a polynomial time equivalent to the Potential Method. Matroid Cost Sharing Games are Cost Sharing Games where the strategy space of each player is exactly the set of bases of a player-specific matroid. The existence of algorithms like the one given here has been an interesting open question [18]. From previous work [5], we know that computing the global potential minimizer is NP-hard even for Singleton Cost Sharing Games. Also we note here that the same holds for the minimum social cost PNE. Hence it is surprising that we can achieve such an efficiency guarantee.
- The above result directly implies the logarithmic bound on the PoS for Matroid Cost Sharing Games with cost functions of the form $f(x) = c_r(x)/x$, where $c_r(x)$ is a nondecreasing concave function.
- For the case of Singleton Cost Sharing Games our algorithm does not output just a PNE but a Strong Nash Equilibrium. Hence this extends the results in [6] on the existence of Strong Nash Equilibria in Cost Sharing Games.
- Our second main result is that computing a PNE in the class of Network Cost Sharing Games where the cost functions come from the Shapley Cost Sharing Mechanism, $f(x) = c_r/x$ (Fair

Cost Allocation) is PLS-complete. The hardness results are based on a tight PLS-reduction from MAX CUT. The result is not restrictive to Fair Cost Allocation and holds for almost any reasonable set of decreasing functions.
- The tightness of our reduction also shows that there exist instances of Network Cost Sharing Games with Fair Cost Allocation, where best response dynamics will certainly need exponential time to reach a PNE. This gives a negative answer to an interesting open question proposed in [2] of whether there exist a scheduling of best response moves that lead to a PNE in polynomially many steps.
- For cost functions of the form $f(x) = c_r(x)/x$, where $c_r(x)$ is a nondecreasing concave function we also prove PLS-completeness for the case of Undirected Network Cost Sharing. This result is not restricted to the above class of functions but generalizes to any class of cost functions that contains almost constant functions.
- The new techniques that we introduce can be used to simplify the existing reductions for the non-decreasing case.

Techniques. To prove our main hardness result we introduce a new class of Congestion Games called k-Congestable Congestion Games. In a k-Congestable Congestion Game the resources of any two strategies of a player are disjoint and each resource is contained in some strategy of at most k players. Thus at most k players can share a resource in any strategy profile. No assumption on the cost functions is made. These games generalize k-Threshold Games introduced by Ackermann et al. [1].

We show how to reduce the computation of a PNE in a 2-Congestable Congestion Game with cost functions that satisfy mild assumptions, to the same problem in a Network Congestion Game with the same set of cost functions. If the class of cost functions is general enough such that it contains almost constant functions with arbitrary high cost, then we can reduce 2-Congestable Congestion Games to Undirected Network Games. We notice that the MAX CUT reduction of Fabrikant et al [7] constructs a 2-Congestable Congestion Game hence our construction can also be applied to simplify the PLS-completeness proofs for the non-decreasing case.

Related Work

Complexity of Equilibria. Apart from the results mentioned in the introduction [7, 1], there has been several works on the relation between PLS and PNE. Skopalik et al [16] proves that even computing an approximate PNE is PLS complete for Congestion Games. Their techniques can also be used to prove PLS-completeness of approximate equilibria in Bottleneck Games (player cost is maximum of cost of allocated resources) as noted independently by Syrgkanis [17] and Harks et al [9].

For Cost Sharing Games Chekuri et al [5] prove that it is NP-hard to compute the global potential minimizer for Multicast Games with Fair Cost Allocation. Hansen et al [8] give an exponential sequence of best response moves for the case of Metric Facility Location Games and provide a polynomial time algorithm for computing approximate equilibria in that class.

On the positive side, Ieong et al. [10] give a dynamic programming algorithm for computing the optimal PNE in the class of Symmetric Singleton Games with arbitrary cost functions. Moreover, Ackermann et al [1] introduce Matroid Congestion Games as a class of games where best response dynamics converge in polynomially many steps.

Quality of Equilibria. Cost Sharing Games in the form studied in this work were introduced by Anshelevich et al. [2]. One of their main results is that the PoS is $O(\log(n))$ (where n is

the number of players) for Cost Sharing Games where the cost functions have the form $f(x) = c_r(x)/x$, where $c_r(x)$ is a nondecreasing concave function. Their proof introduces the Potential Method, a way of bounding the PoS by showing that the global minimizer of Rosenthal's potential (which is a PNE) has cost close to the optimal.

Several other works have dealt with Cost Sharing Games from the perspective of bounding the inefficiency that arises from selfishness. Chekuri et al [5] study Multicast and Facility Location Games when players arrive sequentially and then perform best response. They prove that the quality of the resulting PNE is at most $O(\sqrt{n}\log^2 n)OPT$. Later Charikar et al. [4] improve this bound to $O(\log^3 n)OPT$ and also make progress for the case when best response and sequential arriving is interleaved. Epstein et al [6] study the quality of Strong Nash Equilibria of Cost Sharing Games with Fair Cost Allocation. Strong Nash Equilibria allow for group moves of players, therefore they are a solution concept robust to collusion. However, they do not always exist in Cost Sharing Games. When they exist Epstein et al [6] show that their worst case quality matches the PoS bound of H_n. Balcan et al [3] study Cost Sharing Games with Fair Cost Allocation under the perspective of Learning Agents. They prove that if players perform best response but at each step with a small fixed probability chose their strategy in a nearly optimal outcome, then the expected quality of the resulting PNE is $O(\log(n)\log(n|F|))OPT$, where $|F|$ is the number of resources.

2 Definitions and Notation

We will now give the formal definitions and notation for all the models that we cope with.

Definition 1. *A **Congestion Game**, denoted by $\langle N, F, (S_i)_{i\in N}, (r_f)_{f\in F}\rangle$, consists of: A set of N players and a set of facilities F. For each player i a set of strategies $S_i \subseteq 2^F$. For each facility f a cost function $r_f(x)$.*

The cost of a player i at strategy profile s is $C_i(s_i, s_{-i}) = \sum_{f\in s_i} r_f(n_f(s))$, where $n_f(s)$ (congestion) is the number of players using facility f in strategy profile s.

Given a strategy profile s, we define the Social Cost of s as: $SC(s) = \sum_{i\in[N]} C_i(s)$ and the potential of s as: $\Phi(s) = \sum_{f\in F}\sum_{k=1}^{n_f(s)} r_f(k)$.

Definition 2. *A **Cost Sharing Game** is a Congestion Game where the facility cost functions $r_f(x)$ are non-increasing. Any Cost Sharing Game may also be augmented by the property of **Fair Cost Allocation** which means that the cost functions have the specific form of $r_f(x) = \frac{c_f}{x}$.*

Definition 3. *A **Network Cost Sharing Game** is a Cost Sharing Game, where the strategy space of each player i is the set of all possible paths between two nodes (s_i, t_i) on a directed network $G = (V, E)$. If we assume an undirected network then we have the class of **Undirected Network Cost Sharing Games**. If all players share a common sink then we have the case of a **Multicast Cost Sharing Game** either on a directed or undirected network.*

Definition 4. *A **Matroid Cost Sharing Game** is a Cost Sharing Game, where for each player $i \in [N]$, S_i is the set of bases of a matroid $\mathcal{M}_i = (F, l_i)$, where l_i is the set of independent sets [15]. Additionally we denote by $rk(G) = max_{i\in[N]} rk(\mathcal{M}_i)$ the rank of the game G, where $rk(\mathcal{M}_i)$ is the rank of matroid $rk(\mathcal{M}_i)$.*

Definition 5. *A **Singleton Cost Sharing Game** is a Cost Sharing Game, where for each player $i \in [N]$, S_i consists of singleton sets. For this class of games we will use an equivalent model that consists of: n jobs and m machines and an arbitrary bipartite graph $\mathcal{G}$ on nodes $[n] \cup [m]$. The jobs are the players of the game and their strategies is to choose one of the machines they are connected to in $\mathcal{G}$.*

We denote with M_j the set of neighbors of job j, i.e. the set of possible machines job j can choose from. We denote with N_i the set of neighbors of machine i, i.e. the set of jobs machine i can be picked by.

3 Computing a Good Pure Nash Equilibrium

Matroid Cost Sharing Games is a subclass of Matroid Congestion Games, hence by Ackermann et al. [1] we have that best response dynamics converge to a PNE in at most $n^2 m\ rk(G)$ steps. Thus one polynomial algorithm that gives a PNE starts from an arbitrary configuration and performs best response in each step.

However, one interesting question is whether we can find a good quality PNE, for example the one that globally minimizes the potential function or at least a PNE with good social cost characteristics. Chekuri et al. [5] give a negative answer for the first question through an NP-hardness result which in our notation implies the following theorem:

Theorem 1 ([5]). *Computing the global potential minimizer for the class of Singleton Cost Sharing Games with cost functions of the form $f(x) = 1/x$ is NP-hard.*

The proof is based on a gap introducing reduction by Lund and Yannakakis [12]. We remark here that this reduction can also be used to prove that computing the socially optimal PNE is NP-hard. For completeness purposes we give the proof in the Appendix.

However, these hardness results do not exclude the possibility of computing a PNE with social cost comparable with the bound on the PoS produced by the Potential Method. Specifically the Potential Method works as follows: Suppose that for any profile s: $SC(s) \leq \Phi(s) \leq \alpha SC(s)$. Then if we find the global potential minimizer $\hat{s}$ and denoting with s^* the optimal, we get that: $SC(\hat{s}) \leq \Phi(\hat{s}) \leq \Phi(s^*) \leq \alpha SC(s^*)$, hence the PoS is at most α. Now we know that computing the global potential or social cost minimizer is NP-hard, however if we manage to find a PNE $\hat{s}'$ such that: $SC(\hat{s}') \leq \Phi(s^*)$, then we would have: $SC(\hat{s}') \leq \alpha SC(s)$ and we would get the same upper bound. This is the guarantee that we will achieve for the algorithms that follow.

3.1 Singleton Cost Sharing Games

In this section we present the polynomial time algorithm that computes a good PNE for the class of Singleton Cost Sharing Games. We start from Singleton Cost Sharing Games to give a clear intuition for the case of Matroid Cost Sharing Games that will be a generalization of the results presented in this section.

Singleton Cost Sharing Games can also be viewed as a Multicast Cost Sharing Game on a directed network. Given an instance of our model we create a multicast game as follows: Set a common sink t. Create a machine node v_i for each machine i and connect it with t with an edge of cost r_i. Create one source node s_j for each job j. Then for each $j \in N_i$ create an edge of cost 0 from source node s_j to machine node v_i.

It could also be viewed as a Multicast Cost Sharing Game on an undirected network. Instead of setting the edge costs of edges (s_j, v_i) to 0 we set it to some huge number q. This number has no strategic consequence on the game: (1) players will use only one such edge because otherwise they would incur a huge cost, (2) all strategies of the players will have this additive cost of q and therefore players will choose only according to the cost on the final edge (v_i, t). However, the Price of Anarchy and PoS bounds do not carry over in this case.

In the special case of Fair Cost Allocation, the social cost is the sum of the costs of the machines used and the optimization problem of computing a strategy profile with minimum social cost is a problem equivalent to SET COVER.

Algorithm 1 Poly-time algorithm for good PNE

Require: An instance of $G = \langle \mathcal{G} = ([n] \cup [m], E), (r_i)_{i \in [m]} \rangle$
  $\mathcal{G}^1 = \mathcal{G}$
  for $t = 1$ to m **do**
    Let $d_i^t = |N_i^t|$ be the degree of machine i in $\mathcal{G}^t$
    Let $i^t = \arg\min_{i \in [m]} r_i(d_i^t)$
    For all $j \in N_{i^t}^t$ set $s_j = i^t$
    Remove nodes $N_{i^t}^t \cup i^t$ from $\mathcal{G}^t$ to obtain $\mathcal{G}^{t+1}$
  end for
  return Strategy profile s

Our algorithm works as follows: Each time pick the machine that incurs the minimum player cost if it is assigned all the possible unassigned jobs and assign to that machine all possible jobs. We iterate until all jobs are assigned. A pseudocode of this is depicted in Algorithm 1.

Theorem 2. *Algorithm 1 outputs a PNE of* $G = \langle \mathcal{G} = ([n] \cup [m], E), (r_i)_{i \in [m]} \rangle$

Proof. Suppose in the end of the algorithm, some job j wants to move from his current machine i to some i'. Assume j was assigned to i at time step t_0, i.e. $i = i^{t_0}$. Since i was assigned at time step t_0 it was not connected to any machine i^t for $t < t_0$. Thus i' must correspond to some machine i^{t_1} for $t_1 > t_0$. Since j was not assigned to i' it means that at t_0, the degree of i' dropped by at least 1. Moreover, at each subsequent time step the degree of machine i' can only drop. Thus $d_{i'}^{t_1} \leq d_{i'}^{t_0} - 1$. Moreover, since i was selected at t_0 it means that: $r_i(d_i^{t_0}) \leq r_{i'}(d_{i'}^{t_0}) \leq r_{i'}(d_{i'}^{t_1} + 1)$, where the last inequality holds from the fact that $d_{i'}^{t_0} \geq d_{i'}^{t_1} + 1$ and r_i are non-increasing functions. Now we just need to observe that in the end of the algorithm $n_i = d_i^{t_0}$ and $n_{i'} = d_{i'}^{t_1}$. Combining with the previous inequality we get: $r_i(n_i) \leq r_{i'}(n_{i'} + 1)$. Thus player j does not want to switch to i' which is a contradiction. □

For the case of Fair Cost Allocation, Algorithm (1) is exactly the greedy H_n-approximation for SET COVER. Hence we immediately get a good efficiency guarantee. In addition it is interesting to notice that the tight example for the H_n-approximation given in Example 2.5 of [19] for the greedy approximation algorithm for SET COVER is the identical analogue of the tight example for the PoS given in [2].

Theorem 3. *For any instance of Singleton Cost Sharing Games, Algorithm (1) computes a PNE that is as good as the potential of the optimal allocation.*

Sketch of Proof. We work with a price scheme. Consider a machine i in the optimum solution and assume d players are assigned to it in OPT. Order these players in the order that the

algorithm assigns them to some other machine. When the j-th player was assigned to another machine i' we know that at least $d - j + 1$ players can still be assigned to i. Since we assign all possible players to i' and we choose the machine with the smallest possible player cost we know that j-th player pays a cost of at most $r_i(d - j + 1)$ for being assigned to i'. Iterating this for all j and for all machines in the optimum solution we get the desired result. □

We note here that the above algorithm actually computes a Strong Nash Equilibrium. The proof is simple and is given in the appendix. For Matroid Cost Sharing Games such a guarantee cannot be achieved since the example of [6] that possesses no Strong Nash Equilibrium can be easily transformed into a Matroid Game.

3.2 Matroid Cost Sharing Games

In this section we present a generalization of Algorithm (1) that computes a good PNE for the class of Matroid Cost Sharing Games.

Algorithm 2 Poly-time algorithm for good PNE in Matroid Cost sharing Games

Require: An instance of $\langle N, F, (S_i)_{i\in[N]}, (r_f)_{f\in F}\rangle$
  $\forall i \in [N] : s_i^0 = \emptyset;\ \ \forall f \in F : N_f^0 = \{i \in [N] \mid f \in l_i\},\ d_f^0 = |N_f^0|$
  $t = 0$
  while $\exists f \in F : d_f^t > 0$ **do**
    $f^t = \arg\min_{f\in F} r_f(d_f^t)$
    $\forall i \in N_{f^t}^t$ set $s_i^t = s_i^{t-1} \cup \{f^t\}$
    $t = t + 1$
    $N_f^t = \{i \in [N] \mid s_i^{t-1} \cup \{f\} \in l_i\},\ d_f^t = |N_f^t|$
  end while
  return Strategy profile s

The algorithm works as follows: At each point we keep a temporary strategy for each player, starting from the empty strategy. At each iteration t we compute for each resource to how many players' strategy it could be added (d_f^t). Then we choose the resource that has minimum player cost if added to the strategy of all possible players ($\min_{f\in F} r_f(d_f^t)$) and we perform this addition. This happens until no player's strategy can be further augmented.

Assuming that checking whether some set is in l_i takes polynomial time in the size of the input, then it is clear that the above algorithm runs in polynomial time since the while loop is executed at most $n \cdot rk(G)$ times and during each time step we go over all the resources. For example the above property is true for the case where the strategy space of each player is the set of spanning trees on a set of nodes, like the ISP Network Creation Game.

The following theorem shows that Algorithm (2) behaves in the same way as in the case of Singleton Cost Sharing Games (the proof is given in the Appendix).

Theorem 4. *For any Matroid Cost Sharing Game, Algorithm (2) computes a PNE with social cost at most the potential of the optimal allocation.*

Sketch of Proof. To prove that the resulting allocation is a PNE we use extensively the matroid property that a base is minimum if and only if there is no (1,1) exchange of a facility that results to a better strategy. Then we argue that no profitable (1,1) exchange can exist in a player's

strategy due to the way the algorithm works. To prove the efficiency guarantee we construct for every player a 1-1 mapping of the facilities in the algorithm's allocation and those in the optimal allocation with the following property: whenever a facility is assigned to the player then its mapping in the optimal allocation is also still an option at that time step. In this way we are able to simulate the same logic that we used in the proof of Theorem 3. □

4 Intractability of Cost Sharing Games

In this section we provide the PLS-hardness results. For a more detailed exposition of the class PLS and definitions and properties of PLS reductions the reader is redirected to the initial papers on PLS [11, 14] and to previous work on PLS hardness of congestion games [7, 1].

4.1 General Cost Sharing Games

We will prove that finding a PNE in the class of Cost Sharing Games is PLS-complete via a reduction from MAX CUT under the flip neighborhood.

Definition 6. *We say that a class of functions has the property $\mathcal{P}_1$ if for arbitrary $a > 0$ it contains a function $f(x)$ such that $f(1) = f(2) + a$.*

Theorem 5. *Computing a PNE for the class of Cost Sharing Games with a class of cost functions that has property $\mathcal{P}_1$ is PLS-complete.*

Proof. Assume an instance of MAX CUT on weighted graph $G = (V, E, (w_e)_{e\in E})$. Assume that $(i, j) \notin E \Rightarrow w_{ij} = 0$. We will create an instance of a Cost Sharing Game $\langle N, F, (S_i)_{i\in[N]}, (r_f)_{f\in F}\rangle$ such that from every PNE of the game we can construct in polynomial time a local maximum cut of the MAX CUT instance.

For each node $i \in V$ we add a player $P_i \in [N]$. We assume an ordering of the players $P_1, \ldots, P_N$. For each unordered pair of players $\{i, j\}$ $(i < j)$ we add two facilities f^1_{ij} and f^2_{ij} in the set F, each with cost function r_{ij}, such that $r_{ij}(1) = r_{ij}(2) + w_{ij}$, which can be achieved due to the $\mathcal{P}_1$ property.

$$s^A_i = \{f^2_{ji} \mid j \in \{1 \ldots i-1\}\} \cup \{f^1_{ij} \mid j \in \{i+1 \ldots N\}\}$$
$$s^B_i = \{f^1_{ji} \mid j \in \{1 \ldots i-1\}\} \cup \{f^2_{ij} \mid j \in \{i+1 \ldots N\}\}$$

In other words for each pair of players $\{i, j\}$ if player i has facility f^1_{ij} in his s^A_i strategy then player j has facility f^2_{ij} in his s^A_j strategy and correspondingly for the B strategies.

Now given any strategy profile s we consider the following partition of the initial graph: If player $P_i \in N$ is playing s^A_i then place node i in partition $V_A(s)$ and to partition $V_B(s)$ otherwise. For every node $i \in V$ denote with $w_i = \sum_{j\in V} w_{ij}$, $w(i, V_A) = \sum_{j\in V_A} w_{ij}$, $w(i, V_B) = \sum_{j\in V_B} w_{ij}$.

Given any strategy profile s the cost of player P_i for playing each strategy is:

$$C_i(s^A_i, s_{-i}) = \sum_{j\in V_A(s)} r_{ij}(1) + \sum_{j\in V_B(s)} r_{ij}(2) = w(i, V_A(s)) + \sum_{j\neq i} r_{ij}(2)$$
$$C_i(s^B_i, s_{-i}) = \sum_{j\in V_A(s)} r_{ij}(2) + \sum_{j\in V_B(s)} r_{ij}(1) = w(i, V_B(s)) + \sum_{j\neq i} r_{ij}(2)$$

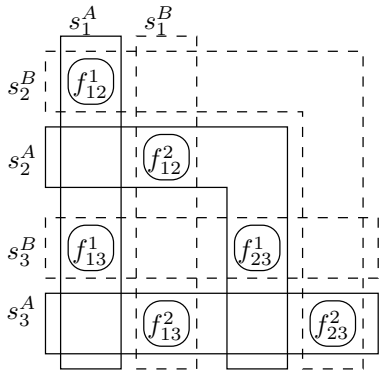

Fig. 1. Example of a reduction from MAX CUT on three node graph to computing a PNE in Cost Sharing Games.

Thus if s is a PNE and P_i is playing s_i^A then: $C_i(s_i^A, s_{-i}) \leq C_i(s_i^B, s_{-i}) \implies w(i, V_A(s)) \leq w(i, V_B(s))$. Hence, switching node i from partition $V_A(s)$ to $V_B(s)$ will not increase the weight of the cut. Similarly if P_i is playing s_i^B we get the opposite inequality. Therefore, for any PNE s the corresponding partition $(V_A(s), V_B(s))$ is a local maximum of the initial MAX CUT instance. □

An example of the above reduction is depicted in Fig. 1.

Corollary 1. *Computing a PNE for the class of Cost Sharing Games with Fair Cost Allocation is PLS-complete.*

Proof. We just need to show that the class of cost functions $f(x) = c_r/x$ for arbitrary c_r has property $\mathcal{P}_1$. It is easy to see that for arbitrary $a > 0$ if we set $c_r = 2a$ then we get the desired property that $f(1) = f(2) + a$. □

4.2 Extending to Network Games

We will introduce k-Congestable Games. We will then notice that the game instance created in the MAX CUT reduction belongs to the class of 2-Congestable Cost Sharing Games with Fair Cost Allocation. We will then show how from any instance of a 2-Congestable Game we can create a Network Congestion Game that preserves the PNE.

Definition 7. *A k-Congestable Congestion Game is a Congestion Game where: (1) Any facility is used by at most k players. (2) The facilities on the different strategies of a player are disjoint. There is no restriction on the cost functions.*

Definition 8. *A class of cost functions has property $\mathcal{P}_2$ if for arbitrary $H > 0$ it contains a function $f(x)$ such that $\min_{k \in [1..N]} f(k) > H$ and any member of the class has bounded maximum in a finite integer range $[1..N]$.*

Theorem 6. *Given an instance of a 2-Congestable Congestion Game with cost functions from a class with property $\mathcal{P}_2$, we can create an instance of a Network Congestion Game on a directed network, where any PNE of the latter corresponds to a PNE of the former and the conversion can be computed in polynomial time. Moreover, the reduced game contains the same set of cost functions as the initial. (proof in appendix)*

Sketch of Proof. First we create gadgets that force two players to share an edge in the network. We add one such gadget in our construction for each facility in the initial game and the shared edge has exactly the same cost function as the initial. Then we interconnect these gadgets with external edges. The interconnection and the costs of the external edges is such that for each player the only undominated paths in the network are those that correspond to strategies in the initial game. □

Corollary 2. *Computing a PNE in the class of Network Cost-Sharing Games with Fair Cost Allocation is PLS complete.*

Proof. The Cost Sharing Game with Fair Cost Allocation constructed in the MAX CUT reduction is a 2-Congestable Cost Sharing Game. Moreover, functions of the form $f(x) = c_r/x$ have property $\mathcal{P}_2$, since given some H we can satisfy the desired property by setting $c_r = NH$. □

In Fig. 4 of the appendix we depict the reduction described above for the game created in the MAX CUT reduction in Fig. 1. Now we show for which classes of functions we can have PLS-completeness in undirected networks too.

Definition 9. *A class of functions has property $\mathcal{P}_3$ if for any $a, \epsilon > 0$ it contains a function $f(x)$ such that $f(1) = a$ and $\max_{k\in[1..N]} f(k) - \min_{k\in[1..N]} f(k) \leq \epsilon$.*

Theorem 7. *Given an instance of a 2-Congestable Congestion Game with cost functions from a class that has properties $\mathcal{P}_2$ and $\mathcal{P}_3$, we can create an instance of an Undirected Network Congestion Game, where any PNE of the latter corresponds to a PNE of the former and the conversion can be computed in polynomial time. Moreover, the reduced game contains the same set of cost functions as the initial. (proof in appendix)*

Corollary 3. *Computing a PNE in the class of Undirected Network Cost-Sharing Games with functions of the form $f(x) = c_r(x)/x$, where $c_r(x)$ is a non-decreasing concave function, is PLS-complete.*

Proof. We need to show that the above class of functions has properties $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$. Since it contains all functions of the form $f(x) = c_r/x$, we know from previous discussion that it has properties $\mathcal{P}_1$ and $\mathcal{P}_2$. For property $\mathcal{P}_3$ we assume that $c_r(x) = cx^{1-\delta}$. By setting $c = a$ we have that $f(1) = a$. Now we need $\max_{k\in[1..N]} f(k) - \min_{k\in[1..N]} f(k) \leq \epsilon$. Since the cost functions are decreasing the above translates to $f(1) \leq f(N) + \epsilon$. If $\epsilon \geq a - \frac{a}{N}$ then setting $\delta = 1$ yields the desired property. Otherwise, we have $a \leq \frac{a}{N^\delta} + \epsilon \Rightarrow \delta \leq \frac{\log(\frac{a}{a-\epsilon})}{\log(N)}$. Since $\epsilon < a - \frac{a}{N}$, $0 < \delta < 1$, hence $c_r(x)$ is an increasing concave function. □

4.3 Tightness of PLS-reductions

It is easy to observe that all the PLS reductions used in the previous sections are tight reductions as defined in [14]. From the initial works on PLS [11, 14] we know that tight reductions do not decrease the distance of an initial solution from a local optimum through local improvement steps. Moreover, we know that there exist instances of MAX CUT with initial configurations that have exponential distance from any local maximum. This directly implies that there exist

instances of the class of games for which we prove PLS-completeness, with strategy profiles such that any sequence of best response moves needs exponential time to reach a PNE.

Moreover, the tightness of our reductions shows that for the class of games we cope with it is PSPACE-complete to compute a PNE that is reachable from a specific initial strategy profile through best response moves.

5 Discussion and Further Results

Another interesting fact that might be useful in other reductions is the following:

Theorem 8. *Computing a PNE in General Congestion Games where all players have two strategies and each facility is used by at most two players can be reduced to computing a PNE of a 2-Congestable Congestion Game. If the initial game contains a cost function $r_f(x)$ then the reduced game might contain cost functions of the form $r_f(x+k)$ for arbitrary k.*

Last, we observe that our reductions from 2-congestable games also show how one can conclude PLS completeness of Undirected Network Congestion Games with linear cost functions, directly from the MAX CUT reduction of [7] without even introducing 2-threshold congestion games. It is easy to observe that the Congestion Game created in the reduction of [7] is a 2-Congestable Game and linear functions is a class that satisfies properties $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$.

Acknowledgements. I would like to deeply thank Eva Tardos for the many fruitful and insightful discussions on the subject. I would also like to thank the reviewers for the helpful comments.

References

1. Heiner Ackermann, Heiko Röglin, and Berthold Vöcking, *On the impact of combinatorial structure on congestion games*, J. ACM **55** (2008), no. 6.
2. Elliot Anshelevich, Anirban Dasgupta, Jon M. Kleinberg, Éva Tardos, Tom Wexler, and Tim Roughgarden, *The price of stability for network design with fair cost allocation*, FOCS, IEEE Computer Society, 2004, pp. 295–304.
3. Maria-Florina Balcan, Avrim Blum, and Yishay Mansour, *Circumventing the price of anarchy: Leading dynamics to good behavior*, ICS, 2010, pp. 200–213.
4. Moses Charikar, Howard J. Karloff, Claire Mathieu, Joseph Naor, and Michael E. Saks, *Online multicast with egalitarian cost sharing*, SPAA, ACM, 2008, pp. 70–76.
5. Chandra Chekuri, Julia Chuzhoy, Liane Lewin-Eytan, Joseph Naor, and Ariel Orda, *Non-cooperative multicast and facility location games*, IEEE Journal on Selected Areas in Communications **25** (2007), no. 6, 1193–1206.
6. Amir Epstein, Michal Feldman, and Yishay Mansour, *Strong equilibrium in cost sharing connection games*, EC, ACM, 2007, pp. 84–92.
7. Alex Fabrikant, Christos H. Papadimitriou, and Kunal Talwar, *The complexity of pure nash equilibria*, STOC, ACM, 2004, pp. 604–612.
8. Thomas Dueholm Hansen and Orestis Telelis, *Improved bounds for facility location games with fair cost allocation*, COCOA, Lecture Notes in Computer Science, vol. 5573, Springer, 2009, pp. 174–185.
9. Tobias Harks, Martin Hoefer, Max Klimm, and Alexander Skopalik, *Computing pure nash and strong equilibria in bottleneck congestion games*, ESA (2), Lecture Notes in Computer Science, vol. 6347, Springer, 2010, pp. 29–38.
10. Samuel Ieong, Robert McGrew, Eugene Nudelman, Yoav Shoham, and Qixiang Sun, *Fast and compact: A simple class of congestion games*, AAAI, 2005, pp. 489–494.
11. David S. Johnson, Christos H. Papadimitriou, and Mihalis Yannakakis, *How easy is local search?*, J. Comput. Syst. Sci. **37** (1988), no. 1, 79–100.

12. Carsten Lund and Mihalis Yannakakis, *On the hardness of approximating minimization problems*, J. ACM **41** (1994), no. 5, 960–981.
13. Robert W. Rosenthal, *A class of games possessing pure-strategy nash equilibria*, International Journal of Game Theory **2** (1973), no. 1, 65–67.
14. Alejandro A. Schäffer and Mihalis Yannakakis, *Simple local search problems that are hard to solve*, SIAM J. Comput. **20** (1991), no. 1, 56–87.
15. Alexander Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency, Volume B, Matroids, Trees, Stable Sets*, 2003, pp. 39–69.
16. Alexander Skopalik and Berthold Vöcking, *Inapproximability of pure nash equilibria*, STOC, ACM, 2008, pp. 355–364.
17. Vasileios Syrgkanis, *Equilibria in Congestion Game Models*, Undergraduate Diploma Thesis, http://www.cs.cornell.edu/~vasilis/thesis.pdf (2009).
18. Eva Tardos and Jon Kleinberg, *Algorithm Design*, ch. 12, Addison Wesley, 2005.
19. Vijay V. Vazirani, *Approximation Algorithms*, Springer, 2001.

A Computing a Good PNE

Theorem 9. *It is NP-hard to compute the PNE that minimizes the Social Cost even in the class of Singleton Cost Sharing Games with cost functions of the form $f(x) = 1/x$.*

Proof. We give a reduction from 3SAT. A variation of a theorem by Lund and Yannakakis [12] is the following:

Theorem 10. *Given a 3SAT formula ϕ, an instance of the unweighted set cover problem can be constructed in polynomial time, such that:*

- *All sets have equal size (denoted by s).*
- *If ϕ is satisfiable (yes-instance), then there is a solution to the set cover instance that uses $X = n/s$ sets (n is the number of elements), and each element is covered by exactly one set in this solution.*
- *If ϕ is not satisfiable (no-instance), then the size of any solution to the set cover instance is at least aX, where $a > 1$ is some constant.*

The reduction is as follows: First construct the instance of Set Cover that corresponds to formula ϕ of 3SAT as indicated by Theorem 10. From this construct the corresponding instance of our game, using the connection between Set Cover and our game used so far.

If ϕ is satisfiable then by theorem (10) we know that there exists a strategy profile where exactly $X = n/s$ machines are used, each machine is connected with exactly s jobs and all jobs are connected with only one machine currently used by some other player. Hence this strategy profile is clearly a PNE and has social cost X.

Now if ϕ is not satisfiable then by theorem (10) any solution to the Set Cover instance must have cost at least aX. Hence, any strategy profile of the corresponding instance and hence any PNE must have social cost at least aX. Hence, if we could find the PNE with the optimal social cost, we could decide whether formula ϕ is satisfiable. □

Proof (Proof of Theorem 3). We work with a price scheme. We introduce the following notation: If s is an assignment then $N_i(s)$ is the set of jobs assigned to machine i in s, $n_i(s) = |N_i(s)|$ and $M(s)$ is the set of machines used in s. If the contexts are clear then we might sometimes drop the s from the above notations. The cost of an arbitrary allocation is:

$$\sum_{j\in[n]} r_{s_j}(n_{s_j}(s)) = \sum_{i\in M(s)} n_i(s) r_i(n_i(s)) \tag{1}$$

Thus, each used machine in s accounts for a cost of $n_i(s)r_i(n_i(s))$. Therefore, if we distribute that cost equally among all jobs in $N_i(s)$ then the cost per job is $r_i(n_i(s))$.

Let a be the allocation output by the algorithm and o the optimal. During the algorithm each time we pick a machine and assign $N_i(a)$ to it we charge each job in $N_i(a)$ with the price per job for machine i, which is $r_i(n_i(a))$. In the end the total charge of the elements will be exactly the cost of allocation a. In the above definition we also use the fact that each machine is allocated jobs only once and each job is allocated to exactly one machine.

Now, consider an arbitrary machine $i \in M(o)$. Let $d = n_i(o)$, $N_i(o) = \{j_d, \ldots, j_1\}$ and assume wlog that the jobs in $N_i(o)$ are ordered in the order assigned by the algorithm (again the worst case is when all jobs are assigned in different time steps). Consider the time step t when job j_k was assigned by the algorithm. We know that $d_i^t \geq k$ and therefore $r_i(d_i^t) \leq r_i(k)$. Since the algorithm chooses the machine i' with the smallest $r_{i'}(d_{i'}^t)$ we get that job j_k was charged at most $r_i(k)$. Thus the total charge of the jobs in $N_i(o)$ is at most $\sum_{k=1}^{d} r_i(k)$. Summing over all machines in $M(o)$ and since all jobs are assigned to exactly one machine in the optimal solution, we get that the total charge given to the jobs by the algorithm is: $\Phi(o) = \sum_{i \in M(o)} \sum_{k=1}^{n_i(o)} r_i(k)$. The above completes the proof. □

Theorem 11. *Algorithm 1 outputs a Strong Nash Equilibrium of the game* $G = \langle \mathcal{G} = ([n] \cup [m], E), (r_i)_{i \in [m]} \rangle$

Proof. We will prove it by induction on the time step that a player was assigned. We will prove that if no player that is assigned to a machine at any time $t' \leq t$ wants to change strategy no matter who he colludes with, then no player who is assigned at iteration $t+1$ wants to make any group deviation.

Base Case: Consider a player assigned at time step 1. Suppose that he colludes with a group of players and switches from i^1 to some other machine i^k. In the best case he will collude with all the $d_{i^k}^1$ players that can be assigned to i^k and they will move there. But from the way the algorithm works we know that $r_{i^1}(n_{i^1}) = r_{i^1}(d_{i^1}^1) \leq r_{i^k}(d_{i^k}^1)$. Hence, this player has no incentive to collude and change strategy.

Inductive Step: Consider a player p assigned at time step $t+1$. Suppose that he colludes with a group of players and switches from i^{t+1} to some other machine i^k. We know that $k > t+1$ since such a player does not have the option to move to a machine i^k with $k < t+1$. Now from the induction hypothesis no player assigned earlier has incentive to make a coalitional move. Hence, the only players that p can collude with are those assigned at time step $t+1$ and later on. In the best case he will collude with all the at most $d_{i^k}^{t+1}$ players and switch to i^k. But from the algorithm we know that $r_{i^{t+1}}(n_{i^{t+1}}) = r_{i^{t+1}}(d_{i^{t+1}}^{t+1}) \leq r_{i^k}(d_{i^k}^{t+1})$. Hence, p has no incentive deviate. □

Proof (Proof of Theorem 4). We will first prove that the strategy profile is a PNE:

Lemma 1. *Algorithm 2 outputs a PNE*

Proof. First we just need to notice that the algorithm outputs a valid strategy profile. In other words the strategies for each player are bases of their matroids. Assume that some players strategy is not a base. Therefore there must exist some facility f that if added to his strategy will lead to a set still in l_i. However, this means that at the last iteration of the while loop $d_f^t > 0$ which is a contradiction. Lets assume that the strategy profile s output by Algorithm (2) is not a PNE. Hence, there must exist some player i with profitable deviating strategy.

From the properties of matroids [15] we know that given a matroid $M = (F, l)$ with weights $F \mapsto \mathbb{N}$, a basis $B \in l$ is a minimum weight basis of M if and only if there exists no basis $B^* \in l$ with $|B \setminus B^*| = 1$ and $w(B^*) < w(B)$.

This property leads to the fact since s is not a PNE player i can decrease his cost by switching one facility f^* in his strategy with a facility f that he does not use. Let t^* be the iteration at which f^* was added to player i's strategy. Since f is not in player i's strategy it means that it was considered in a later iteration $t > t^*$. If it was considered previous to t^* and it was not added to player i's strategy then it means that an (f, f^*) switch is not valid since all resources that prevented f from being added to player i's strategy are still there after the exchange.

Hence, at time step t^* it holds: $r_{f^*}(d_{f^*}^{t^*}) \leq r_f(d_f^{t^*})$. Moreover, since f was never placed in player i's strategy but was accounted during computation of $d_f^{t^*}$, it means that in the final strategy profile $n_f(s) \leq d_f^{t^*} - 1 \implies d_f^{t^*} \geq n_f(s) + 1$. Moreover, it is clear that $n_{f^*}(s) = d_{f^*}^{t^*}$. Hence:

$$r_{f^*}(n_{f^*}(s)) = r_{f^*}(d_{f^*}^{t^*}) \leq r_f(d_f^{t^*}) \leq r_f(n_f(s) + 1) \tag{2}$$

where the last inequality follows from the fact that r_f are non-increasing delay functions. The last inequality show that (f, f^*) is not a profitable deviation for player i and hence we have a contradiction. □

Now we give the quality guarantee of the strategy profile:

Lemma 2. *Algorithm 2 outputs a strategy profile with social cost at most the potential of the optimal profile*

Proof. Again we work with the price scheme. Some notation again: If s is a strategy profile then $N_f(s)$ is the set of players that use facility f in s, $n_f(s) = |N_f(s)|$ and $F(s)$ is the set of facilities used in s. If the contexts are clear then we might sometimes drop the s from the above notations. The cost of an arbitrary allocation is:

$$\sum_{i \in [N]} \sum_{f \in s_i} r_f(n_f(s)) = \sum_{i \in F(s)} n_f(s) r_f(n_f(s)) \tag{3}$$

Thus, each used facility in s accounts for a cost of $n_f(s) r_f(n_f(s))$. Therefore, if we distribute that cost equally among all players in $N_f(s)$ then the cost per player on a facility is $r_f(n_f(s))$.

Let a be the strategy profile output by the algorithm and o the optimal. During the algorithm each time we pick a facility and allocate it to $N_f(a)$ we charge each player in $N_f(a)$ with the price per player for facility f, which is $r_f(n_f(a))$. In the end the total charge of the players will be exactly the cost of allocation a.

In the above definition we also use the fact that each facility is allocated to players only once and the cost of a player is the sum of his costs on the facilities he uses.

Let a_i, o_i be the strategies of player i in the algorithm's output and in the optimal. We will create a $1-1$ mapping μ from a_i to o_i such that when a facility $f \in a_i$ is added to player i's strategy then the option of adding $f^* = \mu(f)$ was also available.

To create the above mapping we will heavily use the matroid property of the strategies of player i. Let $a_i = \{f_1, f_2, \ldots, f_r\}$ and $o_i = \{f_1^*, \ldots, f_r^*\}$, where r is the rank of player i's matroid. Now, assume the time step that f_r was added to player i's strategy. From the augmentation property of matroids we know that adding a strategy from o_i was also an option. Wlog we can

call that strategy f_r^*. Now, let's argue about the time step that f_{r-1} was added to a_i. From the properties of matroids we know that if a set is in the matroid then every subset is also. Thus, $o_i^{r-1} = \{f_1^*, \ldots, f_{r-1}^*\} \in l$. At the moment f_{r-1} was added to a_i we know that the current set for player i was $a_i^{r-2} = \{f_1, \ldots, f_{r-2}\}$. Since $|o_i^{r-1}| > |a_i^{r-2}|$, there must exist some facility in $f^* \in o_i^{r-1}$ such that $a_i^{r-1} \cup f^* \in l$. Wlog we can call that facility f_{r-1}^*. Continuing in the same manner we see that we can create the $1-1$ mapping that we described in the previous paragraph.

We say that a player i is excluded from facility f^* if $\mu^{-1}(f^*)$ is allocated to him. Now consider an arbitrary facility $f \in F(o)$. Let $d = n_f(o)$, $N_f(o) = \{i_d, \ldots, i_1\}$ and assume wlog that the players in $N_f(o)$ are ordered in the order at which the algorithm excludes them from f.

Consider the time step t when player i_k was excluded from f by the algorithm. We know that $d_f^t \geq k$ and therefore $r_f(d_f^t) \leq r_i(k)$. Since the algorithm chooses the facility f' with the smallest $r_{f'}(d_{f'}^t)$ we get that player i_k was charged at most $r_f(k)$. Thus the total charge of the players in $N_f(o)$ is at most $\sum_{k=1}^{d} r_f(k)$. Summing over all facilities in $F(o)$, and because the mapping that we created was a bijection, we get that the total charge given to the players by the algorithm is: $\Phi(o) = \sum_{f \in F(o)} \sum_{k=1}^{n_f(o)} r_f(k)$. □

The above two lemmata give the theorem. □

B Intractability of Cost sharing Games

Proof (Proof of Theorem 6). Wlog we may assume that a k-Congestable Congestion Game is of the following form: Each strategy $s_i^{k_i}$ of player i consists of a facility $f_{ij}^{k_i,k_j}$ for each other player j and some strategy k_j of j and of a facility $f_i^{k_i}$. We assume an ordering on the players and whenever we denote a facility $f_{ij}^{k_i,k_j}$ we assume $i < j$.

If the initial description of the Congestion Game is not of the above form then we can simply introduce new facilities with zero cost so as to bring it in the above form. Hence we only need to work on the above restricted form.

Each resource $f_{ij}^{k_i,k_j}$ is represented in the Network Congestion Game by the gadget depicted in Fig. 2, which is the same as the one used in [1] for reducing threshold games to network games.

Players are forced, in equilibrium, to exit from the designated edges because of the structure of the rest of the game as will be seen later on. We will denote a gadget associated with facility $f_{ij}^{k_i,k_j}$, with $G_{ij}^{k_i,k_j}$.

For non shared facilities $f_i^{k_i}$ the gadget $G_i^{k_i}$ that will represent them has a single input and single output and contains a single edge $f_i^{k_i}$ that has the same cost function as facility $f_i^{k_i}$ (sometimes we might overload notation and denote the above gadget with $G_{ii}^{k_i,k_i}$.

Now we need to describe how to interconnect the gadgets and how to set the start and end nodes of each player.

Each player i has a separate start and end node (s_i, t_i).

The facilities are placed as follows (Fig. 3): We order the player $[1..N]$. We place facilities in cells of a lower triangular matrix of size $N-1 \times N-1$. Each cell $(i,j), i < j$ contains all gadgets $G_{ij}^{k_i,k_j}$ for any k_i, k_j, each cell (i,i) contains all gadgets $G_i^{k_i}$ and all cells $(i,j), i > j$ are empty.

We connect the i output of gadget $G_{ij}^{k_i,k_j}$ with the i input of gadget $G_{i,j+1}^{k_i}$. Moreover:

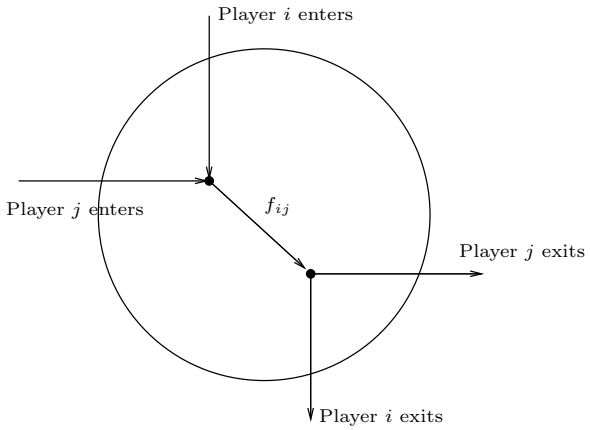

Fig. 2. Gadget for representing a shared facility $f_{ij}^{k_i,k_j}$

- If $i+1<j$ we connect the j output of $G_{ij}^{k_i,k_j}$ to the j input of gadget $G_{i+1,j}^{k_{i+1},k_j}$, where k_{i+1} is the strategy of player $i+1$ with which strategy k_j of j overlaps.
- If $i+1=j$ we connect the j output of $G_{ij}^{k_i,k_j}$ to the input of $G_j^{k_j}$.

Moreover, we connect the output of each gadget $G_i^{k_i}$ to the i input of gadget $G_{i,i+1}^{k_i,k_{i+1}}$, where k_{i+1} is the strategy of player $i+1$ with which strategy k_i overlaps.

All the connecting edges of the form $e_i=(G_{i,j}^{k_i,k_j},G_{i,j+1}^{k_i,k_{j+1}})$ have cost 0. All the connecting edges of the form $e_j=(G_{i,j}^{k_i,k_j},G_{i+1,j}^{k_{i+1},k_j})$ have cost functions r_j that satisfy the property:

$$\min_{i\in[N]} r_j(i) \geq \max_{i\in[N]} r_{j-1} + D$$

where D is some huge constant larger than any cost that could be incurred in the initial k-Congestable Game.

Last we describe how to connect the start and end node for each player i. We connect each s_i to the i inputs of all gadgets $G_{1,i}^{k_i,k_1}$ with an edge of cost 0. Moreover, we connect the i output of all gadgets $G_{i,N}^{k_i,k_N}$ to the end node t_i.

A pictorial representation of the above construction is depicted in Fig. 3.

Now we see that each player i enters in column 1 and has to exit from column i. Thus he has to follow exactly $i-1$ edges e_j to reach that column no matter how he travels. However, the property that the edges e_j satisfy ensures that each player e_j prefers to travel on his row j than on any other row (the rows above him he does not have the option due to the direction of the edges and the rows below him contain edges that will give him and additive cost larger than any cost incurred by the gadget edges.

Thus the only undominated options of player i any path of the form:

$$s_i \to G_{1,i}^{k_1,k_i} \to G_{2,i}^{k_2,k_i} \to \ldots \to G_{i,i}^{k_i,k_i} \to G_{i,i+1}^{k_i,k_{i+1}} \to \ldots \to G_{i,N}^{k_i,k_N} \to t_i$$

Each such option corresponds to strategy $s_i^{k_i}$ of the initial 2-Congestable Congestion Game. Denote with $C_i(s)$ the cost of a player on the network game in strategy profile s. When s contains no undominated strategy we denote with $\tilde{s}$ the corresponding strategy profile in the

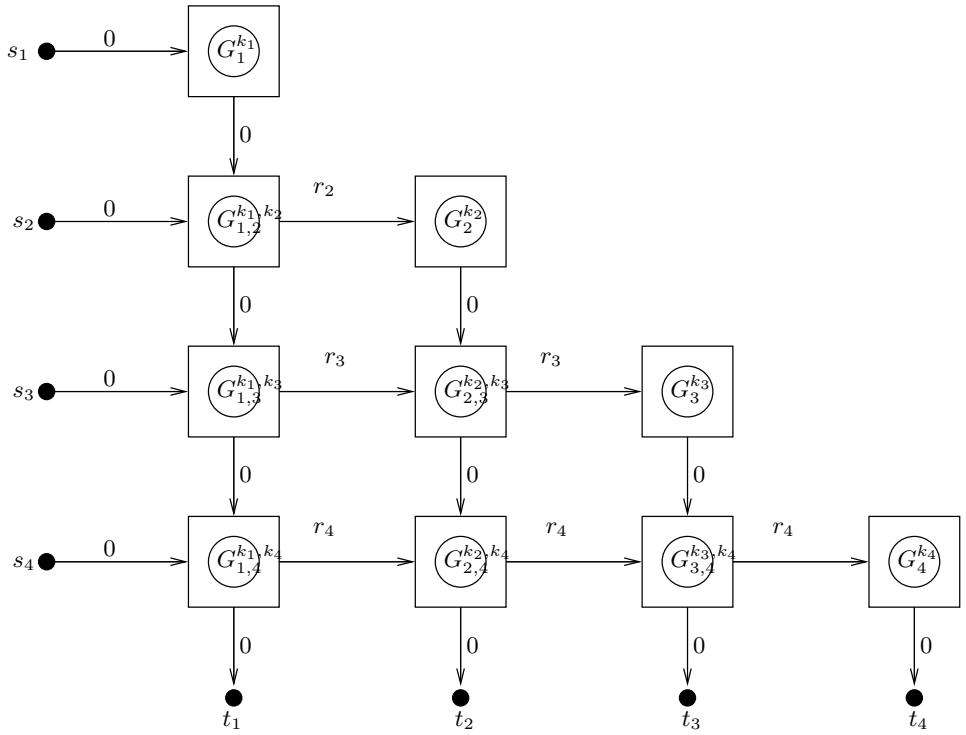

Fig. 3. Network Game instance.

initial 2-Congestable Game. Also denote with $\tilde{C}_i(\tilde{s})$ the cost of a player i in the 2-Congestable Game. For any undominated strategy profile s we observe the following property:

$$\forall k_i \in S_i : C_i(s_i^{k_i}, s_{-i}) = \tilde{C}_i(\tilde{s}_i^{k_i}, \tilde{s}_{-i}) + (i-1)r_i(1)$$

Since the additive term of $(i-1)r_i(1)$ is the same in all of player i's strategies it has no strategic consequences.

Hence, we conclude that if s is a PNE in the Network Game then $\tilde{s}$ is a PNE in the initial game.

□

Proof (Proof of Theorem 7). We make a small alteration to the construction in the proof of Theorem 6. Instead of zero cost the vertical edges have all cost function $v(x)$ such that $v(1) = H^1$ and $v(N) = H^N$, where H^1, H^N is a number much greater than any cost imposed by the horizontal edges or the facility edges and $H^1 = H^N + \epsilon$, where ϵ is a very small number. Moreover, we add a cost of $g(x)$, such that $\min_{k\in[N]} g(k) = H_2 >> H^1, H^N$ to all the edges that connect the starting nodes with the whole structure. Having done this we remove the directedness of the edges.

No player wants to visit the starting node of another player since he would incur delay at least H_2 which is much greater than any cost on his canonical paths. Now we examine whether a player would want to deviate from his canonical path, which to travel horizontal until the diagonal and then vertically.

If some player goes up at some point during the horizontal part then he has to pay at least two H^N more than his canonical path and the gain he gets from using a less costly horizontal edge

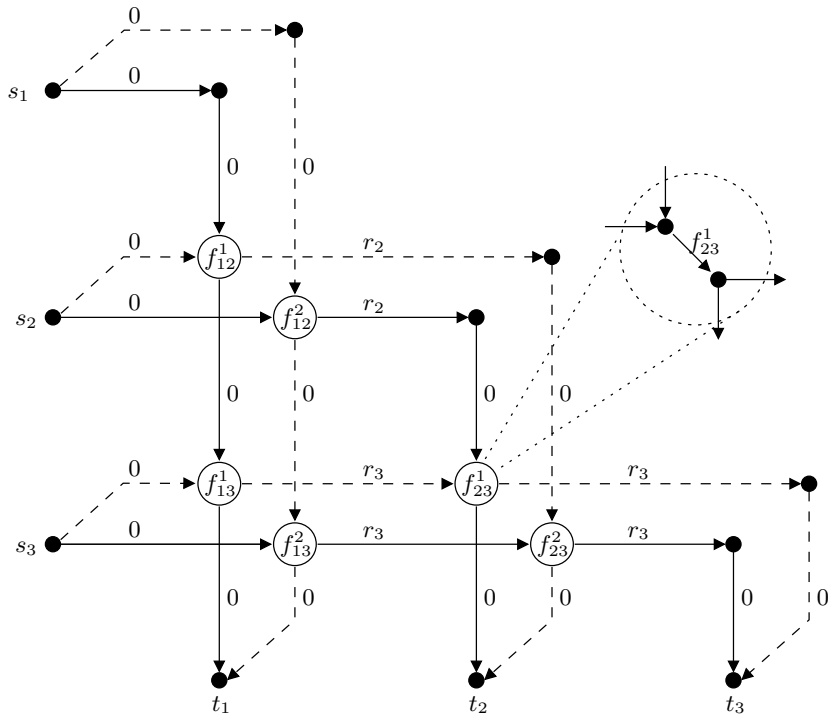

Fig. 4. Reduction from General Cost Sharing to Network Cost Sharing.

cannot compensate for that. If a player goes down at a point other than the diagonal then he has to use some horizontal edge that is more costly than his own. The only gain he might get from not going down at the diagonal would be a small ϵ, which cannot compensate for the much larger cost of the horizontal edge. Last, suppose that some player deviates during his vertical part. By doing that he can only gain at most ϵ from sharing a vertical edge with some other player, but he has to pay at least two more horizontal edges which have cost much more than ϵ. Hence, no matter what the rest of the players do, a player has incentive to use his canonical path. □

C Further Results

Proof (Proof of Theorem 8). We observe that the only difference between this class and 2-Congestable Games is that the facilities of two different strategies of a player are not disjoint. However,since each player i has only 2 strategies, the facilities in the intersection will always be used by that player. Moreover, the cost incurred on those facilities will have no strategic meaning for the choice of the player. Hence, we can simply do the following two modifications and still preserve the PNE:

- For each player i, remove the intersection of the two strategies from each one making the strategies disjoint.
- For any facility f let $k_f = |\{i \in N \mid f \in s_i^1 \cap s_i^2\}|$. Update the costs of all the facilities to $r'_f(n_f) = r_f(n_f + k_f)$.

Now if $r'_f(n_f)$ still remains in the class of cost functions considered initially then the 2-Congestable Game is also in the same class of games in terms of cost functions. For example this is true for linear functions. □