Congestion Games with Multisets of Resources and Applications in Synthesis

Guy Avni[1], Orna Kupferman[1], and Tami Tamir[2]

1 School of Computer Science and Engineering, The Hebrew University, Jerusalem, Israel
2 School of Computer Science, The Interdisciplinary Center, Israel

Abstract

In classical congestion games, players' strategies are subsets of resources. We introduce and study *multiset congestion games*, where players' strategies are multisets of resources. Thus, in each strategy a player may need to use each resource a different number of times, and his cost for using the resource depends on the load that he and the other players generate on the resource.

Beyond the theoretical interest in examining the effect of a repeated use of resources, our study enables better understanding of non-cooperative systems and environments whose behavior is not covered by previously studied models. Indeed, congestion games with multiset-strategies arise, for example, in production planing and network formation with tasks that are more involved than reachability. We study in detail the application of synthesis from component libraries: different users synthesize systems by gluing together components from a component library. A component may be used in several systems and may be used several times in a system. The performance of a component and hence the system's quality depends on the load on it.

Our results reveal how the richer setting of multisets congestion games affects the stability and equilibrium efficiency compared to standard congestion games. In particular, while we present very simple instances with no pure Nash equilibrium and prove tighter and simpler lower bounds for equilibrium inefficiency, we are also able to show that some of the positive results known for affine and weighted congestion games apply to the richer setting of multisets.

Keywords and phrases Congestion games, Multiset strategies, Equilibrium existence and computation, Equilibrium inefficiency

Digital Object Identifier 10.4230/LIPIcs.xxx.yyy.p

1 Introduction

Congestion games model non-cooperative resource sharing among selfish players. Resources may be shared by the players and the cost of using a resource increases with the load on it. Such a cost paradigm models settings where high congestion corresponds to lower quality of service or higher delay. Formally, each resource e is associated with an increasing latency function $f_e : \mathbb{N} \to \mathbb{R}$, where $f_e(\ell)$ is the cost of a single use of e when it has load ℓ.

Previous work on congestion games assumes that players' strategies are subsets of resources, as is the case in many applications, most notably routing and network design. For example, in the setting of networks, players have reachability objectives and strategies are subsets of edges, each inducing a simple path from the source to the target [29, 3, 19]. We introduce and study multiset games, where players' strategies are multisets of resources. Thus, a player may need a resource multiple times – depending on the specific resource and strategy, and his cost for using the resource depends on the load that he and the other players generate on it. Formally, in *multiset congestion games* (MCGs, for short), a player that uses j times a resource e that is used ℓ times by all players together, pays $j \cdot f_e(\ell)$ for these uses.

Beyond the theoretical interest in examining the effect of multisets on the extensively-studied classical games, multiset congestion games arise naturally in many applications and environments.

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 1–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The use of multisets enables the specification of rich settings that cannot be specified by means of subsets. We give here several examples.

As a first example, consider *network formation*. In addition to reachability tasks, which involve simple paths (and hence, subsets of resources), researchers have studied tasks whose satisfaction may involve paths that are not simple. For example, a user may want to specify that each traversal of a low-security channel is followed by a visit to a check-sum node. A well-studied class of tasks that involve paths that need not be simple are these associated with a specific length, such as patrols in a geographical region. Several communication protocols are based on the fact that a message must pass a pre-defined length before reaching its destination, either for security reasons (e.g., in *Onion routing*, where the message is encrypted in layers [27]) or for marketing purposes (e.g., advertisement spread in social networks). In addition, tasks of a pre-defined length are the components of *proof-of-work* protocols that are used to deter denial of service attacks and other service abuses such as spam (e.g., [15]), and of several protocols for sensor networks [7]. The introduction of multiset corresponds to strategies that are not necessarily simple paths [5].

In *production systems* or in *planning*, a system is modeled by a network whose nodes correspond to configurations and whose edges correspond to actions performed by resources. Users have tasks, that need to be fulfilled by taking sequences of actions. This setting corresponds to an MCG in which the strategies of the players are multisets of actions that fulfill their tasks, which indeed often involve repeated execution of actions [13]; for example "once the arm is up, do not put it down until the block is placed". Also, multiset games can model *preemptive scheduling*, where the processing of a job may split in several feasible ways among a set of machines.

Our last example, which we are going to study in detail, is *synthesis form component libraries*. A central problem in formal methods is synthesis [26], namely the automated construction of a system from its specification. In real life, hardware and software systems are rarely constructed from scratch. Rather, a system is typically constructed from a library of components by *gluing* components from a library (allowing multiple uses) [23]. For example, when designing an internet browser, a designer does not implement the TCP protocol but uses existing implementations as black boxes. The library of components is used by multiple users simultaneously, and the usages are associated with costs. The usage cost can either decrease with load (e.g., when the cost of a component represents its construction price, the users of a component share this price) as was studied in [4], or increase with load (e.g., when the components are processors and a higher load means slower performance). The later scenario induces an instance of an MCG.

Let us demonstrate the intricacy of the multiset setting with the question of the existence of a *pure Nash equilibrium* (PNE). That is, whether each instance of the game has a profile of pure strategies that constitutes a PNE – a profile such that no player can decrease his cost by unilaterally deviating from his current strategy. By [28], classical congestion games are potential games and thus always have a PNE. Moreover, by [19], in a symmetric congestion game, a PNE can be found in polynomial time. As we show in Example 1 below, a PNE might not exist in an MCG even in a symmetric two-player game over identical resources.

Example 1: Consider the following symmetric MCG with two players and three resources: a, b, and c. The players' strategy space is $\{a, a, b\}$ or $\{b, b, c\}$ or $\{c, c, a\}$. That is, a player needs to access some resource twice and the (cyclically) consequent resource once. The latency function of all three resources is the same, specifically, $f_a(\ell) = f_b(\ell) = f_c(\ell) = \ell^2$. The players' costs in all possible profiles are given in Table 1. We show that no PNE exists in this game. Assume first that the two players select distinct strategies, w.l.o.g. $\{a, a, b\}$ and $\{b, b, c\}$. In this profile, a is accessed twice, b is accessed three times, and c is accessed once. Thus, every access of a, b and c costs $4, 9$ and 1 respectively. The cost of Player 1 is $8 + 9 = 17$, while the cost of Player 2 is $18 + 1 = 19$. By deviating to $\{c, c, a\}$, the cost of Player 2 will reduce to 17 (while the cost of Player 1 will increase to

19). Thus, no PNE in which the players select different strategies exists. If the player select the the same strategy, then one resource is accessed 4 times, and one resource is accessed twice, implying that the cost of both players is $2 \cdot 16 + 1 \cdot 4 = 36$, and any deviation is profitable. We conclude that no PNE exists in the game. ◀

	$\{a, a, b\}$	$\{b, b, c\}$	$\{c, c, a\}$
$\{a, a, b\}$	$36, 36$	$19, 17$	$17, 19$
$\{b, b, c\}$	$17, 19$	$36, 36$	$19, 17$
$\{c, c, a\}$	$19, 17$	$17, 19$	$36, 36$

Table 1 Players costs. Each entry describes the cost of Player 1 followed by the cost of Player 2.

We study and answer the following questions in general and for various classes of multiset congestion games (for formal definitions, see Section 2): (i) Existence of a PNE. (ii) An analysis of *equilibrium inefficiency*. A *social optimum* (SO) of the game is a profile that minimizes the total cost of the players; thus, the one obtained when the players obey some centralized authority. It is well known that decentralized decision-making may lead to solutions that are sub-optimal from the point of view of society as a whole. We quantify the inefficiency incurred due to selfish behavior according to the *price of anarchy* (PoA) [22] and *price of stability* (PoS) [3] measures. The PoA is the worst-case inefficiency of a PNE (that is, the ratio between the cost of a worst PNE and the SO). The PoS is the best-case inefficiency of a Nash equilibrium (that is, the ratio between the cost of a best PNE and the SO). (iii) *Computational complexity* of finding a PNE.

Before we turn to describe our results, let us review related work. *Weighted* congestion games (WCGs, for short), introduced in [25], are congestion games in which each player i has a *weight* $w_i \in \mathbb{N}$, and his contribution to the load of the resources he uses as well as his payments are multiplied by w_i. WCGs can be viewed as a special case of MCGs, where each resource in a strategy for Player i repeats w_i times. A different extension of WCGs in which players may use a resource more than once is *integer splittable WCGs* [24, 30]. These games model the setting in which a player has a number (integer) of tasks he needs to perform and can split them among the resources. For example, in the network setting, a player might need to send $\ell \in \mathbb{N}$ packets from vertex s to t. He can send the packets on different paths, but a packet cannot be split. MCGs are clearly more general than WCGs and integer splittable WCGs – the ability to repeat each resource a different number of times lead to a much more complex setting. Thus, it is interesting to compare our results with these known for these games.

It is shown in [17, 21] that the existence of a PNE in WCGs depends on the latency function: when the latency functions are either affine or exponential, WCGs are guaranteed to admit a PNE, whereas WCGs with a polynomial latency function need not have a PNE. In [24], the author shows that PNE always exists when the latency functions are linear using a *potential function* argument. This argument fails when the latency functions are convex, but [30] are still able to show that there is always a PNE in these games. We are able to show that the exact potential function of [17] applies also to (the much richer) affine MCGs (that is, MCGs with a affine latency function), and thus they always admit a PNE. As demonstrated in Example 1, very simple MCGs with quadratic latency functions might have no PNE.

We turn on to results in the front of equilibrium inefficiency. In congestion games with affine latency functions, both the PoA and PoS measures are well understood. It was shown in [12] that PoS $\geq 1 + \frac{1}{\sqrt{3}} \approx 1.577$ and is at most 1.6. A tight upper bound was later shown in [10]. Also, PoA $= \frac{5}{2}$ [12]. Going one step towards our setting to the study of affine WCGs, [6] shows that PoA $= 1 + \phi$, where $\phi \approx 1.618$ is the golden ratio. The PoS question is far from being settled. Only recently, [9] shows a first upper bound of 2 for PoS in linear WCGs, which is a subclass of affine

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 3–14
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

WCGs. As far as we know, the only lower bound that is known for affine WCGs is the lower bound from the unweighted setting. So there is a relatively large gap between the upper- and lower-bounds for the PoS in these games.

We bound the potential function in order to show that every affine MCG G has $\text{PoS}(G) < 2$. This improves and generalizes the result in [9]. Our most technically-challenging result is the PoS lower-bound proof, which involves the construction of a family $\mathcal{G}$ of linear MCGs. Essentially, the game $G_k \in \mathcal{G}$ is parameterized by the number of players and defined recursively. The use of multisets enables us to to define a game in which, although the sharing of resources dramatically changes between its profiles, the cost a player pays is equal in all of them. For $k = 17$ we obtain that $\text{PoS}(G_{17}) > 1.631$. This is the first lower bound in these models that exceeds the 1.577 lower bound in congestion games. Finally, the PNE in $\mathcal{G}$ is achieved with dominant strategies, so our bound holds for stronger equilibrium concepts.

As for the PoA, we show that MCGs with latency functions that are polynomials of degree at most d have PoA $= \Phi_d^{d+1}$, where Φ_d is the unique nonnegative real solution to $(x+1)^d = x^{d+1}$. Observe that Φ_d is a natural generalization of the golden ratio to higher degrees. Specifically, $\Phi_1 = \phi$. For the upper bound, we adjust the upper-bound proof of [2] to our setting. We show a simplified matching lower bound; a simple two-player MCG with only two resources and latency functions of the form $f(\ell) = \ell^d$. For general latency functions we show that the PoA can be arbitrarily high.

We turn to study the application of synthesis from component libraries by multiple players. Recall that in this application, different users synthesize systems from components. A component may be used in several systems and may be used several times in a system. The quality of a system depends on the load on its components. This gives rise to an MCG, which we term a *component library game* (CLG, for short). On the one hand, a CLG is a special case of MCG, so one could expect positive results about MCGs to apply to CLGs. On the other hand, while in MCGs the strategies of the players are given explicitly by means of multisets of resources, in CLGs the strategies of the players are given symbolically by means of a specification deterministic finite automaton – the one whose language has to be composed from the library's components.

We prove that every MCG has a corresponding CLG, implying that negative results for MCGs apply to CLGs. Moreover, we show that the succinctness of the presentation of the strategies makes decision problems about MCGs more complex in the setting of CLGs. We demonstrate this by studying the complexity of the *best-response* problem – deciding whether a player can benefit from a unilateral deviation from his strategy, and the problem of deciding whether a PNE exists in a given game. For the best-response problem, which is in P for MCGs, we prove NP-completeness for CLGs. The problem of deciding the existence of a PNE is known to be strongly NP-complete for weighted symmetric congestion games. For network congestion games with player specific cost functions, this problem is NP-complete for arbitrary networks, while a PNE can be found efficiently for constant size networks [1]. We provide a simpler hardness proof for MCGs, which is valid also for a constant number of resources, and we show that for CLGs the problem is Σ_2^P-complete. As good news, we are able to prove a "small-design property" for CLGs, which bounds the number of strategies that one needs to consider and enables us to lift to CLGs the positive results for MCGs with linear latency functions. Thus, such CLGs always have a PNE and their PoS is at most 2.

Due to the lack of space, some examples and proofs are omitted and can be found in the full version, in the authors' URLs.

2 Preliminaries

A *multiset* over a set E of elements is a generalization of a subset of E in which each element may appear more than once. For a multiset A over E and an element $e \in E$, we use $A(e)$ to denote the number of times e appears in A, and use $e \in A$ to indicate that $A(e) \geq 1$. When describing multisets,

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 4–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

we use e^m, for $m \in \mathbb{N}$, to denote m occurrences of e.

A *multiset congestion game* (MCG) is a tuple $G = \langle K, E, \{\Sigma_i\}_{i \in K}, \{f_e\}_{e \in E} \rangle$, where $K = \{1, \dots, k\}$ is a set of players, E is a set of *resources*, for every $1 \leq i \leq k$, the *strategy space* Σ_i of Player i is a collection of multisets over E, and for every resource $e \in E$, the *latency function* $f_e : \mathbb{N} \to \mathbb{R}$ is a non-decreasing function. The MCG G is an *affine* MCG if for every $e \in E$, the latency function f_e is affine, i.e., $f_e(x) = a_e x + b_e$, for non-negative constants a_e and b_e. Similarly, we say that G is a *linear* MCG if it is affine and for $e \in E$ we have $b_e = 0$. We assume w.l.o.g. that for $e \in E$ we have $a_e \geq 1$. Classical congestion games are a special case of MCGs where the players' strategies are sets of resources. Weighted congestion games can be viewed as a special case of MCGs, where for every $1 \leq i \leq k$, multiset $s_i \in \Sigma$ and $e \in s_i$, we have $s_i(e) = w_i$.

A profile of a game G is a tuple $P = \langle s_1, s_2, \dots, s_k \rangle \in (\Sigma_1 \times \Sigma_2 \times \dots \times \Sigma_k)$ of strategies selected by the players. For a resource $e \in E$, we use $L_{e,i}(P)$ to denote the number of times e is used in P by Player i. Note that $L_{e,i}(P) = s_i(e)$. We define the *load* on e in P, denoted $L_e(P)$, as the number of times it is used by all players, thus $L_e(P) = \sum_{1 \leq i \leq k} L_{e,i}(P)$ [1].

In classical congestion games, all players that use a resource e pay $f_e(\ell)$, where ℓ is the number of players that use e. As we formalize below, in MCGs, the payment of a player for using a resource e depends on the number of times he uses it. Given a profile P, a resource $e \in E$, and $1 \leq i \leq k$, the *cost of e for Player i in P* is $cost_{e,i}(P) = L_{e,i}(P) \cdot f_e(L_e(P))$. That is, for each of the $L_{e,i}(P)$ uses of e, Player i pays $f_e(L_e(P))$. The cost of Player i in the profile P is then $cost_i(P) = \sum_{e \in E} cost_{e,i}(P)$ and the cost of the profile P is $cost(P) = \sum_{1 \leq i \leq k} cost_i(P)$. We also refer to the cost of a resource e in P, namely $cost_e(P) = \sum_{i \in K} cost_{e,i}(P)$.

Consider a game G. For a profile P, player $i \in K$, and a strategy $s_i' \in \Sigma$ for Player i, let $P[i \leftarrow s_i']$ denote the profile obtained from P by replacing the strategy for Player i by s_i'. A profile P is a *pure Nash equilibrium* (PNE) if no Player i can benefit from unilaterally deviating from his strategy in P to another strategy; i.e., for every player i and every strategy $s_i' \in \Sigma$ it holds that $cost_i(P[i \leftarrow s_i']) \geq cost_i(P)$.

We denote by OPT the cost of a social-optimal solution; i.e., $OPT = \min_P cost(P)$. It is well known that decentralized decision-making may lead to sub-optimal solutions from the point of view of society as a whole. We quantify the inefficiency incurred due to self-interested behavior according to the *price of anarchy* (PoA) [22] and *price of stability* (PoS) [3] measures. The PoA is the worst-case inefficiency of a Nash equilibrium, while the PoS measures the best-case inefficiency of a Nash equilibrium. Formally,

▶ **Definition 1.** Let $\mathcal{G}$ be a family of games, and let G be a game in $\mathcal{G}$. Let $\Upsilon(G)$ be the set of Nash equilibria of the game G. Assume that $\Upsilon(G) \neq \emptyset$.

- The *price of anarchy* of G is the ratio between the *maximal* cost of a PNE and the social optimum of G. That is, $\text{PoA}(G) = \max_{P \in \Upsilon(G)} cost(P)/OPT(G)$. The *price of anarchy* of the family of games $\mathcal{G}$ is $\text{PoA}(\mathcal{G}) = sup_{G \in \mathcal{G}} PoA(G)$.
- The *price of stability* of G is the ratio between the *minimal* cost of a PNE and the social optimum of G. That is, $\text{PoS}(G) = \min_{P \in \Upsilon(G)} cost(P)/OPT(G)$. The *price of stability* of the family of games $\mathcal{G}$ is $\text{PoS}(\mathcal{G}) = sup_{G \in \mathcal{G}} PoS(G)$.

3 Existence of a Pure Nash Equilibrium

As demonstrated in Example 1, MCGs are less stable than weighted congestion games:

[1] Since our strategies are multisets, we have that $s_i(e)$, for all i and e, is an integer. Our considerations, however, are independent of this, thus all our results are valid also for games in which strategies might include fractional demands for resources. In non-splittable (atomic) games, the players must select a single strategy, even if fractional demands are allowed.

▶ **Theorem 2.** *There exists a symmetric two-player MCG with identical resources and quadratic latency function that has no PNE.*

On the positive side, we show that a PNE exists in all MCGs with affine latency functions. We do so by showing that an exact potential function exists, which is a generalization of the one in [9, 18].

▶ **Theorem 3.** *Affine MCGs are potential games.*

Proof. For a profile P and a resource $e \in E$, define

$$\Phi_e(P) = a_e \cdot \big(\sum_{i=1}^{k}\sum_{j=i}^{k} L_{e,i}(P) \cdot L_{e,j}(P)\big) + \big(b_e \cdot \sum_{i=1}^{k} L_{e,i}(P)\big).$$

Also, $\Phi(P) = \sum_{e\in E} \Phi_e(P)$. In the full version, we prove that Φ is an exact potential function. ◀

The negative result in Theorem 2 gives rise to the decision problem ∃PNE; given an MCG, decide whether it has a PNE. Being a generalization of WCGs, the hardness results known for WCGs imply that ∃PNE is NP-hard [14]. Using the richer definition of MCGs, we show below a much simpler hardness proof. We also show hardness for games with a constant number of resources, unlike congestion games with user-specific cost functions [1].

▶ **Theorem 4.** *Given an instance of an MCG, it is strongly NP-complete to decide whether the game has a PNE, as well as to find a PNE given that one exists. For games with a constant number of resources, the problems are NP-Complete.*

Remark 4.1: In *splittable (non-atomic) games*, each player can split his task among several strategies. This can be seen as if each player is replaced by $M \to \infty$ identical players all having the same strategy space scaled by $1/M$. This model suits several applications, in particular planning of preemptive production. Splittable games are well-understood in classical and weighted congestion games [29, 8]. In the full version we define the corresponding MCG and show that the positive PNE-existence result, known for weighted congestion games, carry over to games with multisets of resources. ◀

4 Equilibrium Inefficiency in MCGs

4.1 The Price of Stability

The PoS problem in affine congestion games is settled: [12, 10] show that PoS $= 1 + \frac{1}{\sqrt{3}} \approx 1.577$. For affine WCGs, the problem was open for a long time, and only recently progress was made by [9], who showed that PoS ≤ 2 for linear WCGs. As far as we know, there is no known lower bound for linear WCGs that exceeds the 1.577 bound for unweighted games. We show that every affine MCG G has PoS$(G) < 2$. Thus, we both improve the result to include affine functions, tighten the bound, and generalize it. For the lower bound, we show a family of linear MCGs $\mathcal{G}$ that has PoS$(\mathcal{G}) > 1.631$. We start with the upper bound.

▶ **Theorem 5.** *Every affine MCG G has PoS*$(G) < 2$.

Proof. Consider an affine MCG G and a profile P. It is not hard to see that for the potential function Φ that is presented in Theorem 3 we have $\Phi(P) \leq cost(P)$. Moreover, for $e \in E$ we have $2\Phi_e(P) = cost_e(P) + a_e \sum_{1\leq i\leq k} L^2_{e,i}(P) + b_e \sum_{1\leq i\leq k} L_{e,i}(P)$. Thus, $\Phi(P) > \frac{1}{2}cost(P)$. The theorem follows using standard techniques: $cost(O) \geq \Phi(O) \geq \Phi(N) > \frac{1}{2}cost(N)$, where O is the social optimum and N is a PNE that is reached from O by a sequence of best-respond moves of the players. Then, $PoS(G) \leq \frac{cost(N)}{cost(O)} < 2$. The details of the proof can be found in the full version. ◀

Note that while the PoS can get arbitrarily close to 2, it is strictly smaller than 2 for every game instance. The proof in [9], on the other hand, only shows PoS ≤ 2 for the family of affine MCGs, and our result does not improve this bound.

For the lower bound, we show a family of linear MCG $\mathcal{G} = \{G_k\}_{k\geq 2}$ that are parameterized by the number of players. Using a computerized simulation, we obtain that for the game with 17 players, we have $\text{PoS}(G_{17}) > 1.631$. We leave open the problem of calculating the exact value the PoS tends to as the number of players increases. In the full version we show a graph of the PoS as a function of k, which hints that the answer is only slightly higher than 1.631.

The PNE in the games in the family is achieved with dominant strategies, and thus it is resistant to stronger types of equilibria.

▶ **Theorem 6.** *There is a linear MCG G with PoS$(G) > 1.631$.*

Proof. We define a family of games $\{G_k\}_{k\geq 2}$ as follows. The game G_k is played by k players, thus $K_k = \{1, \dots, k\}$. For Player 1, all strategies $\Sigma_1^k = \{O_1^k\}$ consists of a single multiset. For ease of presentation we sometimes refer to O_1^k as N_1^k. For $i \geq 2$, the strategy space of Player i consists of two multisets, $\Sigma_i^k = \{O_i^k, N_i^k\}$. We define G_k so that for all $k \geq 2$, the profile $\bar{O}_k = \langle O_1^k, \dots, O_k^k\rangle$ is the social optimum and the profile $\bar{N}_k = \langle N_1^k, \dots, N_k^k\rangle$ is the only PNE.

When describing the games in the family, we partition the resources into *types* and describe a multiset as a collection of triples. A triple $\langle t, y, l\rangle$ stands for y different resources of type t, each appearing l times. For example, $\{\langle a, 2, 1\rangle, \langle b, 1, 3\rangle, \langle c, 2, 2\rangle\}$ stands for the multiset $\{a_1, a_2, b_1, b_1, b_1, c_1, c_1, c_2, c_2\}$. In all games and resources, there are two types of latency functions; the identity function, or identity *plus* epsilon, where the second type of function are linear functions of the form $f(x) = (1+\epsilon)\cdot x$, for some $\epsilon > 0$. The latency function of resources of the same type is the same, and we use the terms "a has identity latency" and "b has identity plus ϵ latency" to indicate that all the resources a' of type a have $f_{a'}(j) = j$ and all the resources b' of type b have $f_{b'}(j) = (1+\epsilon)\cdot j$, for all numbers j of uses.

The definition of G_k is complicated and we start by describing the idea in the construction of G_2 and G_3. In the full version we also describe G_4. We start by describing G_2. The game G_2 is defined with respect to two types of resources, a and b, with identity and identity plus ϵ latency, respectively. We define Player 1's strategy space $\Sigma_1^2 = \{O_1^2\}$ and Player 2's strategy space $\Sigma_2^2 = \{O_2^2, N_2^2\}$, with $O_1^2 = N_2^2 = \langle a, 2, 1\rangle$ and $O_2^2 = \langle b, 1, 2\rangle$. That is, $\Sigma_1^2 = \{\{a_1, a_2\}\}$ and $\Sigma_2^2 = \{\{a_1, a_2\}, \{b_1, b_1\}\}$. Clearly, the profile $\bar{N}_2 = \langle O_1^2, N_2^2\rangle$ is the only PNE in G_2.

We continue to describe G_3. The game G_3 is defined with respect to four types of resources, a, b, c^1 and c^2, where b has identity plus ϵ latency, c^1 has identity plus ϵ' latency, and the other resources have identity latency. Let $x_3 = 3! = 6$. We define $\Sigma_1^3 = \{O_1^3\}$, $\Sigma_2^2 = \{O_2^3, N_2^3\}$, and $\Sigma_3^3 = \{O_3^3, N_3^3\}$, with $O_1^3 = N_2^3 = \langle a, x_3, 1\rangle$, $O_2^3 = \langle b, \frac{x_3}{2}, 2\rangle$, $O_3^3 = \{\langle c^1, \frac{x_3}{3}, 3\rangle, \langle c^2, \frac{x_3}{2}, 1\rangle\}$, and $N_3^3 = \{\langle b, \frac{x_3}{2}, 1\rangle, \langle a, x_3, 1\rangle\}$. We claim that $\bar{N}_3 = \langle O_1^3, N_2^3, N_3^3\rangle$ is the only PNE. Our goal here is not to show a complete proof, but to demonstrate the idea of the construction. It is not hard to see that Player 2 deviates to N_2^3 from the profile $\bar{O}_3 = \langle O_1^3, O_2^3, O_3^3\rangle$, Player 3 deviates from the resulting profile $\bar{N}_3 = \langle O_1^3, N_2^3, N_3^3\rangle$. The crux of the construction is to keep Player 2 from deviating back from $\bar{N}_3$. Note that since Player 3 uses the b-type resources once in $\bar{N}_3$, when Player 2 deviates from N_2^3 to O_2^3, their load increases to 3. Thus, $cost_2(\bar{N}_3[2 \leftarrow O_2^3]) = 3(3\cdot 2\cdot(1+\epsilon)) > 6(3\cdot 1) = cost_2(\bar{N}_3)$ and the deviation is not beneficial.

We define the game G_k, for $k \geq 2$, as follows. Let $x_k = k!$. Player 1's strategy space consists of a single multiset $O_1^k = \langle e_{1,1}, x_k, 1\rangle$. For $2 \leq i \leq k$, assume we have defined the strategies and resources for players $1, \dots, i-1$. We define Player i's strategies as follows. We start with the multiset N_i^k, which does not introduce new resources. We define $N_i^k = \cup_{1\leq j\leq i-1}\{\langle t, x, 1\rangle : \langle t, x, l\rangle \subseteq O_i^k\}$. The definition of O_i^k is more involved, but the idea is simple. We define O_i^k so that it satisfies two

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 7–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

properties. First, O_i^k uses new resources. That is, for every $1 \le j \le i-1$, both $O_i^k \cap O_j^k = \emptyset$ and $O_i^k \cap N_j^k = \emptyset$. Consider the profile P_i in which, for every $1 \le j < i$, Player j uses N_j^k and, for every $i \le l \le k$, Player l uses O_l^k. We define O_i^k so that when all resources have identity latency, $cost_i(P_i) = cost_i(P_i[i \leftarrow N_i^k])$. For every multiset $\langle e_{j,a}, x_{j,a}, 1\rangle$ in N_i^k, which we have just defined, we introduce a multiset $\langle e_{i,b}, x_{i,b}, l_{i,b}\rangle$ in O_i^k that uses new resources, where b is a unique index that is arbitrarily chosen, and x_b^i and l_b^i are defined as follows. Let $l = |\{j : e_{j,a} \in N_j^k\}|$. We define $l_{i,b} = l+1$ and $x_{i,b} = x_{j,a}/l_{i,b}$. Since O_i^k uses new resources, showing the first property is easy. In the full version we show it satisfies a much stronger property.

Claim 6.1: Consider $k \in \mathbb{N}$, a profile P in G_k, and $1 < i \le k$. Assume Player i plays O_k^i in P. When the latency functions are identity, we have $cost_i(P) = cost_i(P[i \leftarrow N_k^i])$.

To complete the construction, we define the latency functions so that for every $2 \le i \le k$, we have that $e_{i,1}$-type resources have identity plus ϵ_i latency for $0 < \epsilon_2 < \ldots < \epsilon_k$. By Claim 6.1 there are such values that make N_i^k a dominant strategy for Player i. Thus, the only PNE in G_k, for $k \ge 2$, is the profile $\bar{N}_k = \langle O_1^k, N_2^k, \ldots, N_k^k\rangle$. Next, we identify the social optimum.

Claim 6.2: The profile $\bar{O}_k = \langle O_1^k, \ldots, O_k^k\rangle$ is the social optimum.

Once we identify $\bar{O}_k$ as the social optimum and $\bar{N}_k$ as the only PNE, the calculation of the PoS boils down to calculating their costs, which we do using a computer. Specifically, we have PoS$(G_{17}) = 1.6316$, and we depict the values of G_k, for $2 \le k \le 17$, in the full version. ◀

Remark 6.1: We conjecture that the correct value for the PoS is closer to our lower bound of 1.631 rather than to the upper bound of 2. In the full version we show a more careful analysis of the potential function than the one in Theorem 5 that shows that for every linear MCG G we have $PoS(G) \le 2 - \frac{\sum_{e\in E}\sqrt{cost_e(N_G)}}{cost(O_G)}$, where N_G and O_G denote the cheapest PNE and the social optimum of G, respectively. Also, we show that for every $n \ge 2$, for the MCG G_n that is described in Theorem 6, the inequality in the expression is essentially an equality. ◀

Remark 6.2: We can alter the family in Theorem 6 to have quadratic latency functions instead of identity functions. Although Claim 6.1 does not hold in the altered family, a computerized simulation shows that the N strategies are still dominant strategies. Also, using a computerized simulation, we show that the PoS for G_{15} is 2.399, higher than the upper bound of 2.362 for congestion games, which is shown in [9, 11]. ◀

4.2 The Price of Anarchy

In this section we study the PoA for MCGs. We start with MCGs with polynomial latency functions and show that the upper bound proven in [2] for WCGs can be adjusted to our setting. Being a special case of MCGs, the matching lower bound for WCGs applies too. Still, we present a different and much simpler lower-bound example, which uses a two-player singleton MCG. In a singleton game, each strategy consists of (multiple accesses to) a single resource. Finally, when the latency functions are not restricted to be polynomials, we show that the PoA is unbounded, and it is unbounded already in a singleton MCG with only two players.

We start by showing that the PoA in polynomial MCGs is not higher than in polynomial WCGs. The proof adjusts the one known for WCGs [2] to our setting. For $d \in \mathbb{N}$, we denote by $\mathcal{P}_d$ the set of polynomials of degree at most d.

► **Theorem 7.** *The PoA in MCGs with latency functions in $\mathcal{P}_d$ is at most Φ_d^{d+1}, where Φ_d is the unique nonnegative real solution to $(x+1)^d = x^{d+1}$.*

Next, we show a matching lower bound that is stronger and simpler than the one in [2].

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 8–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

▶ **Theorem 8.** *For $d \in \mathbb{N}$, the PoA in two-player singleton MCG with latency functions in $\mathcal{P}_d$ is at least Φ_d^{d+1}.*

Proof. Let $d \in \mathbb{N}$. Consider the two-player singleton MCG G with resources $E = \{e_1, e_2\}$, strategy spaces $\Sigma_1 = \{e_1^x, e_2^y\}$ and $\Sigma_2 = \{e_1^y, e_2^x\}$, and for $\ell \in \mathbb{R}$, we define the latency functions $f_{e_1}(\ell) = f_{e_2}(\ell) = \ell^d$. We define $x = \Phi_d$ and $y = 1$. Since $x > y$ the social optimum is attained in the profile $\langle e_1^y, e_2^y \rangle$ and its cost is $2y^d = 2$. Recall that in MCGs, the players' strategies are multisets. In particular, x should be a natural number. To fix this, we consider a family of MCGs in which the ratio between x and y tends to the ratio above.

We claim that the profile $N = \langle e_1^x, e_2^x \rangle$ is a PNE. This would imply that $\text{PoA}(G) = \frac{2x^{d+1}}{2} = \Phi_d^{d+1}$, which would conclude the proof. We continue to prove the claim. The cost of a player in N is $x \cdot x^d = x^{d+1}$ and by deviating, the cost changes to $y \cdot (x+y)^d = (x+1)^d$. Our definition of x implies that $x^{d+1} = (x+1)^d$. Thus, the cost does not change after deviating. Since the players are symmetric, we conclude that the profile N is a PNE, and we are done. ◀

Finally, by taking variants with factorial latency functions to the game described in Theorem 8, we are able to increase the PoA in an unbounded manner.

▶ **Theorem 9.** *The PoA in two-player MCGs is unbounded.*

5 Synthesis from Component Libraries

In this section we describe the application of MCGs in synthesis from component libraries. As briefly explained in Section 1, in this application, different users synthesize systems by gluing together components from a component library. A component may be used in several systems and may be used several times in a system. The performance of a component and hence the system's quality depends on the load on it. We describe the setting in more detail, formalize it by means of MCGs, and relate to the results studied in earlier sections.

Today's rapid development of complex and safety-critical systems requires reliable verification methods. In *formal methods*, we reason about systems and their specifications by solving mathematical questions about their models. A central problem in formal methods is synthesis, namely the automated construction of a system from its specification. In real life, systems are rarely constructed from scratch. Rather, a system is typically constructed from a library of components by *gluing* components from the library [23]. In this setting, the input to the synthesis problem is a specification and a library of components, and the goal is to construct from the components a system that exhibits exactly the behaviors specified in the specification.

Remark 9.1: The above setting corresponds to *closed systems*, whose behavior is independent of their environment. It is possible to generalize the definitions to *open systems*, which interact with their environment. In [4], we studied both the closed and open settings in the context of cost-sharing (rather than congestion) games. The technical challenges that have to do with the system being open are orthogonal to these that arise from the congestion effects, and on which we focus in this work. ◀

In our setting, we use deterministic finite automata (DFAs, for short) to model the specification and use *box-DFAs* to model the components in the library. Formally, a DFA is $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$, where Σ is an alphabet, Q is a set of states, $\delta : Q \times \Sigma \to Q$ is a partial transition function, $q_0 \in Q$ is an initial states, and $F \subseteq Q$ is a set of accepting states. The *run* of $\mathcal{A}$ on a word $w = w_1, \ldots w_n \in \Sigma^*$ is the sequence of states $r = r_0, r_1, \ldots, r_n$ such that $r_0 = q_0$ and for every $0 \leq i \leq n-1$, we have $r_{i+1} = \delta(r_i, w_{i+1})$. Now, a box-DFA $\mathcal{B}$ is a DFA augmented with a set of *exit states*. When a run of $\mathcal{B}$ reaches an exit state, it moves to another box-DFA, as we formalize below.

The input to the synthesis from component libraries problem is a specification DFA $\mathcal{S}$ over an alphabet Σ and a library of box-DFAs components $\mathcal{L} = \{\mathcal{B}_1, \ldots, \mathcal{B}_n\}$. The goal is to produce a

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 9–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

design, which is a recipe to compose the components from $\mathcal{L}$ to a DFA. A design is correct if the language of the system it induces coincides with that of the specification.

Intuitively, the design can be thought of as a scheduler; it passes control between the different components in $\mathcal{L}$. When a component $\mathcal{B}_i$ is in *control*, it reads letters in Σ, visits the states of $\mathcal{B}_i$, follows its transition function, and if the run terminates, it is accepting iff it terminates in one of $\mathcal{B}_i$'s accepting states. A component relinquishes control when the run reaches one of its exit states. It is then the design's duty to choose the next component, which gains control through its initial state.

Formally (see an example in Figure 1), a *transducer* is a DFA that has, in addition to the input alphabet that labels the transitions, also an output alphabet that labels the states. Also, a transducer has no rejecting states. Let $[n] = \{1, \ldots, n\}$. A design is a transducer $\mathcal{D}$ whose input alphabet is the set $\mathcal{E}$ of all exit states of all the components in $\mathcal{L}$ and whose output alphabet is $[n]$. We can think of $\mathcal{D}$ as running beside the components. When a component reaches an exit state e, then $\mathcal{D}$ reads the input letter e, proceeds to its next state, and outputs the index of the component to gain control next. Note that the components in the library are black boxes: the design $\mathcal{D}$ does not read the alphabet Σ of the components and has no information about the states that the component visits. It only sees which exit state have been reached. Given a library $\mathcal{L}$ and a design $\mathcal{D}$, their *composition* is a DFA $\mathcal{A}_{\mathcal{L},\mathcal{D}}$ obtained by composing the components in $\mathcal{L}$ according to $\mathcal{D}$. We say that a design $\mathcal{D}$ is *correct* with respect to a specification DFA $\mathcal{S}$ iff $L(\mathcal{A}_{\mathcal{L},\mathcal{D}}) = L(\mathcal{S})$. In the full version we construct $\mathcal{A}_{\mathcal{L},\mathcal{D}}$ formally.

For example, consider the library $\mathcal{L} = \{\mathcal{B}_1, \mathcal{B}_2\}$ over the alphabet $\Sigma = \{a, b, c\}$, and the design $\mathcal{D}$ that are depicted in Figure 1. We describe the run on the word bc. The component that gains initial control is $\mathcal{B}_1$ as the initial state of $\mathcal{D}$ outputs 1. The run in $\mathcal{B}_1$ proceeds with the letter b to the exit state e_1 and relinquishes control. Intuitively, control is passed to the design that advances with the letter e_1 to the state that outputs 2. Thus, the component $\mathcal{B}_2$ gains control, and it gains it through its initial state. Then, the letter c is read, $\mathcal{B}_2$ proceeds to the exit state e_3 and relinquishes control. The design advances with the letter e_3 to a state that outputs 1, and control is assigned to $\mathcal{B}_1$. Since the initial state of $\mathcal{B}_1$ is rejecting, the word ab is rejected. As a second example, consider the word ab. Again, $\mathcal{B}_1$ gains initial control. After visiting the exit state e_2, control is reassigned to $\mathcal{B}_1$. Finally, after visiting the state e_1, control is assigned to $\mathcal{B}_2$, where the run ends. Since the initial state of $\mathcal{B}_2$ is accepting, the run is accepting.

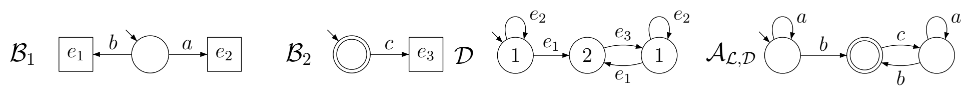

Figure 1 An example of a library $\mathcal{L} = \{\mathcal{B}_1, \mathcal{B}_2\}$, a design $\mathcal{D}$, and the resulting composition $\mathcal{A}_{\mathcal{L},\mathcal{D}}$.

The synthesis problem defined above is aimed at synthesizing correct designs. We now add costs to the setting. A *component library game* (CLG, for short) is a tuple $\langle K, \mathcal{L}, \{\mathcal{S}_i\}_{i \in K}, \{f_\mathcal{B}\}_{\mathcal{B} \in \mathcal{L}} \rangle$, where $K = \{1, \ldots, k\}$ is a set of players, $\mathcal{L}$ is a collection of box-DFAs, the objective of Player $i \in K$ is given by means of a specification DFA $\mathcal{S}_i$, and, as in MCGs, the latency function $f_\mathcal{B}$ for a component $\mathcal{B} \in \mathcal{L}$ maps the load on $\mathcal{B}$ to its cost with this load. For $i \in K$, the set of strategies for Player i is the set of designs that are correct with respect to $\mathcal{S}_i$. A CLG corresponds to an MCG with a slight difference that there might be infinitely many correct designs. Consider a profile $P = \langle \mathcal{D}_1, \ldots, \mathcal{D}_k \rangle$. For a component $\mathcal{B} \in \mathcal{L}$, we use $L_{\mathcal{B},i}(P)$ to denote the number of times Player i uses $\mathcal{B}$ in P. Recall that each state in the transducer $\mathcal{D}_i$ is labeled by a component in $\mathcal{L}$. We define $L_{\mathcal{B},i}(P)$ to be the number of states in $\mathcal{D}_i$ that are labeled with $\mathcal{B}$. The rest of the definitions are as in MCGs.

We first show that every MCG can be translated to a CLG:

▶ **Theorem 10.** *Consider a k-player MCG G. There is a CLG G' between k players with corresponding profiles. Formally, there is a one-to-one and onto function f from profiles of G to profiles of G' such that for every profiles P in G and Player $i \in [k]$, we have that $cost_i(P) = cost_i(f(P))$.*

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 10–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Proof. Consider an MCG $\langle K, E, \{\Sigma_i\}_{i \in K}, \{f_e\}_{e \in E}\rangle$. Recall that Σ_i is the set of strategies for Player i that consists of multisets over E. We construct a CLG with alphabet $E \cup \bigcup_{i \in K} \Sigma_i$. For $i \in K$, the specification $\mathcal{S}_i$ for Player i consists of $|\Sigma_i|$ words. Every strategy $s = \{e_1, \ldots, e_n\}$ (allowing duplicates) in Σ_i contributes to $L(S)$ the word $s \cdot e_1 \cdot e_2 \cdot \ldots \cdot e_n$. We construct a library $\mathcal{L}$ with $|E| + \sum_{i \in K} |\Sigma_i|$ components of two types: a *strategy component* $\mathcal{B}_s$ for each $s \in \Sigma_i$ and a *resource component* $\mathcal{B}_e$ for each $e \in E$. In addition, $\mathcal{L}$ contains the component $\mathcal{B}_{acc}$ that is depicted in Figure 2. Intuitively, a correct design must choose one strategy component $\mathcal{B}_s$ and then use the component $\mathcal{B}_e$ the same number of times e appears in s. We continue to describe the components. For $s \in \Sigma_i$, the component $\mathcal{B}_s$ relinquishes control only if the letter s is read. It accepts every word in $L(\mathcal{S}_i)$ that does not start with s. For $e \in E$, the resource component $\mathcal{B}_e$ has an initial state with an e-labeled transition to an exit state. Finally, the latency function for the resource components coincides with latency functions of the resources in the MCG, thus for $e \in E$, we have $f_{\mathcal{B}_e} = f_e$. The other latency functions are $f \equiv 0$. In the full version we prove that there is a cost-preserving one-to-one and onto correspondence between correct designs with respect to $\mathcal{S}_i$ and strategies in Σ_i, implying the existence of the required function between the profiles. ◀

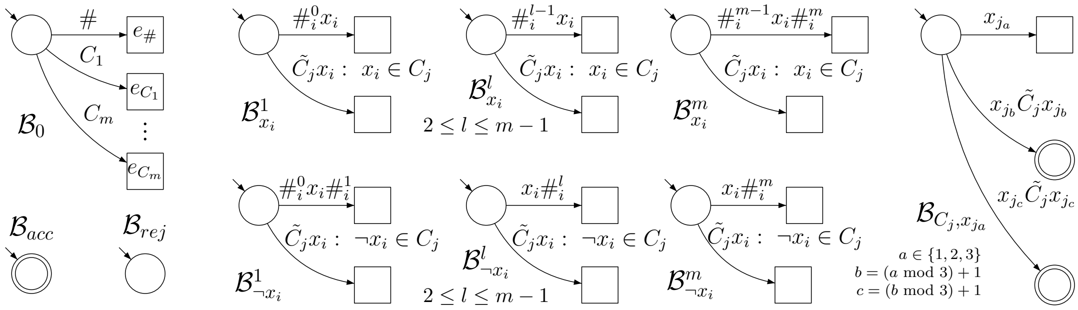

Figure 2 The components in the library $\mathcal{L}$.

Theorem 10 implies that the negative results we show for MCGs apply to CLGs:

▶ **Corollary 11.** *There is a CLG with quadratic latency functions with no PNE; for CLGs with affine latency functions, we have $PoS(CLG) > 1.631$; for $d \in \mathbb{N}$, the PoA in a two-player singleton MCG with latency functions in $\mathcal{P}_d$ is at least Φ_d^{d+1}.*

Remark 11.1: We note that the positive results for CLGs with linear latency functions, namely existence of PNE and $PoS(CLG) \leq 2$, do not follow immediately from Theorem 3, as its proof relies on the fact that an MCG has only finitely many profiles. Since the strategy space of a player might have infinitely many strategies, a CLG might have infinitely many profiles. In order to show that CLGs with linear latency functions have a PNE we need Lemma 12 below, which implies that even in games with infinitely many profiles, there is a best response dynamics that only traverses profiles with "small" designs. Such a traversal is guaranteed to reach a PNE as there are only finitely many such profiles. ◀

Computational complexity We turn to study two computational problems for CLGs: finding a *best-response* and deciding the existence of a PNE. We show that the succinctness of the representation of the objectives of the players in CLGs makes these problems much harder than for MCGs. Our upper bounds rely on the following lemma. The lemma is proven in [4] for cost-sharing games, and the considerations in the proof there applies also for congestion games.

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 11–14
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

▶ **Lemma 12.** *Consider a library $\mathcal{L}$, a specification $\mathcal{S}$, and a correct design $\mathcal{D}$. There is a correct design $\mathcal{D}'$ with at most $|\mathcal{S}| \cdot |\mathcal{L}|$ states, where $|\mathcal{L}|$ is the number of states in the components of $\mathcal{L}$, such that for every component $\mathcal{B} \in \mathcal{L}$, the number of times $\mathcal{D}'$ uses $\mathcal{B}$ is at most the number of times $\mathcal{D}$ uses $\mathcal{B}$.*

We start with the best-response problem (BR problem, for short): Given an MCG $\mathcal{G}$ between k players, a profile P, an index $i \in K$, and $\mu \in \mathbb{R}$, decide whether Player i has a strategy S'_i such that $cost_i(P[i \leftarrow S'_i]) \leq \mu$.

▶ **Theorem 13.** *The BR problem for MCGs is in P. For CLGs it is NP-complete, and NP-hardness holds already for games with one player and linear latency functions.*

Proof. Showing that the BR problem is in P for MCGs follows easily from the fact the set of strategies for Player i is given implicitly and calculating the cost for a player in a profile can be done in polynomial time.

The upper bound for CLGs follows from Lemma 12, which implies an upper bound on the size of the cheapest correct designs. Since checking whether a design is correct and calculating its cost can both be done in polynomial time, membership in NP follows.

We continue to the lower bound. We describe the intuition of the reduction and the formal definition along with the correctness proof can be found in the full version. Given a 3SAT formula φ with clauses $C_1, \ldots, C_m$ and variables $x_1, \ldots, x_n$, we construct a library $\mathcal{L}$ and a specification $\mathcal{S}$ such that there is a design $\mathcal{D}$ that costs at most $\mu = nm + m$ iff φ is satisfiable. The library $\mathcal{L}$ consists of an *initial component* $\mathcal{B}_0$, *variable components* $\mathcal{B}^j_{x_i}$ and $\mathcal{B}^j_{\neg x_i}$ for $j \in [m]$ and $i \in [n]$, *clause components* $\mathcal{B}_{C_j, x_{j_k}}$ for $j \in [m]$ and $k \in \{1, 2, 3\}$, and component $\mathcal{B}_{acc}$ and $\mathcal{B}_{rej}$. The components of the library are depicted in Figure 2. The latency function of the variable components is the identity function $f(x) = x$, thus using such a component once costs 1. The latency functions of the other components is the constant function $f \equiv 0$, thus using such components any number of times is free.

Intuitively, a correct design corresponds to an assignment to the variable and must use nm variable components as follows. For $i \in [n]$, either use all the components $\mathcal{B}^1_{x_i}, \ldots, \mathcal{B}^m_{x_i}$ or all the components $\mathcal{B}^1_{\neg x_i}, \ldots, \mathcal{B}^m_{\neg x_i}$ with a single use each. Thus, a correct design implies an assignment $\eta : \{x_1, \ldots, x_n\} \to \{T, F\}$. Choosing $\mathcal{B}^1_{x_i}, \ldots, \mathcal{B}^m_{x_i}$ corresponds to $\eta(x_i) = F$ and choosing $\mathcal{B}^1_{\neg x_i}, \ldots, \mathcal{B}^m_{\neg x_i}$ corresponds to $\eta(x_i) = T$.

Additionally, in order to verify that a correct design corresponds to a satisfying assignment, it must use m clause components and m more variable components as follows. Consider a correct design $\mathcal{D}$, and let $\eta : \{x_1, \ldots, x_n\} \to \{T, F\}$ be the corresponding assignment as described above. For every $j \in [m]$, $\mathcal{D}$ must use a clause component $\mathcal{B}_{C_j, x_i}$, where recall that the clause C_j includes a literal $\ell \in \{x_i, \neg x_i\}$. Using the component $\mathcal{B}_{C_j, x_i}$ requires $\mathcal{D}$ to use a variable component $\mathcal{B}^t_\ell$, for some $t \in [m]$. So, a correct design uses a total of $nm + m$ components with identity latency. If $\eta(\ell) = F$, then $\mathcal{B}^t_\ell$ is already in use and a second use will cost more than 1, implying that the design costs more than $nm + m$. ◀

The next problem we study is deciding the existence of a PNE. As we show in Theorem 4, the problem is NP-complete for MCGs. As we show below, the succinctness of the representation makes this problem harder for CLGs.

▶ **Theorem 14.** *The $\exists$PNE problem for CLGs is Σ^P_2-complete.*

Proof. The upper bound is easy and follows from Lemma 12. For the lower bound we show a reduction from the complement of *not all equal* $\forall\exists$ *3SAT* (NAE, for short), which is known to be Σ^P_2-complete [16]. An input to NAE is a 3CNF formula φ over variables $x_1, \ldots, x_n, y_1, \ldots, y_n$. It is in NAE if for every assignment $\eta : \{x_1, \ldots, x_n\} \to \{T, F\}$ there is an assignment $\rho : \{y_1, \ldots, y_n\} \to$

$\{T, F\}$ such that every clause in φ has a literal that gets value truth and a literal that gets value false (in η or ρ, according to whether the variable is an x or a y variable). We say that such a pair of assignments $\langle \eta, \rho \rangle$ is *legal* for φ.

Given a 3CNF formula φ, we construct a CLG G with three players such that $\varphi \in$ NAE iff G does not have a PNE. We describe the intuition of the reduction. The details can be found in the full version. There is a one-to-one correspondence between Player 3 correct designs and assignments to the variables $\{x_1, \ldots, x_n\}$. For an assignment $\eta : \{x_1, \ldots, x_n\} \to \{T, F\}$ we refer to the corresponding correct design by $\mathcal{D}_\eta$. Consider a legal pair of assignments $\langle \eta, \rho \rangle$, and assume Player 3 chooses the design $\mathcal{D}_\eta$. Similarly to the proof of Theorem 13, the library contains variable components with identity latency function. We construct the library and the players' objectives so that there is a correct design $\mathcal{D}_\rho$ for Player 1 that uses $mn + 2m$ variable components each with load 1 iff $\langle \eta, \rho \rangle$ is a legal pair for φ. More technically, both $\mathcal{D}_\eta$ and $\mathcal{D}_\rho$ use mn variable components that correspond to the variables $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$, respectively. For every $j \in [m]$, assuming the j-th clause is $\ell_j^1 \vee \ell_j^2 \vee \ell_j^3$, the design $\mathcal{D}_\rho$ must use two additional variable components $\mathcal{B}^{t_1}_{\ell_j^a}$ and $\mathcal{B}^{t_2}_{\ell_j^b}$, for $a \neq b \in \{1, 2, 3\}$ and $t_1, t_2 \in [m]$, which corresponds to η or ρ assigning value true to ℓ_j^a and value false to ℓ_j^b.

Player 1 has an additional correct design $\mathcal{D}_{ALL}$ in which he does not share any components regardless of the other players' choices. Player 2 has two possible designs $\mathcal{D}_A$ and $\mathcal{D}_B$. Assume Player 3 chooses a design $\mathcal{D}_\eta$. We describe the interaction between Player 1 and Player 2. We define the library and the players' objectives so that when Player 1 chooses some design $\mathcal{D}_\rho$, Player 2 prefers $\mathcal{D}_B$ over $\mathcal{D}_A$, thus $cost_2(\langle \mathcal{D}_\rho, \mathcal{D}_A, \mathcal{D}_\eta \rangle) > cost_2(\langle \mathcal{D}_\rho, \mathcal{D}_B, \mathcal{D}_\eta \rangle)$. When Player 2 plays $\mathcal{D}_B$, Player 1 prefers $\mathcal{D}_{ALL}$ over every design $\mathcal{D}_\rho$, thus $cost_1(\langle \mathcal{D}_\rho, \mathcal{D}_B, \mathcal{D}_\eta \rangle) > cost_1(\langle \mathcal{D}_{ALL}, \mathcal{D}_B, \mathcal{D}_\eta \rangle)$. When Player 1 chooses $\mathcal{D}_{ALL}$, Player 2 prefers $\mathcal{D}_A$ over $\mathcal{D}_B$, thus $cost_2(\langle \mathcal{D}_{ALL}, \mathcal{D}_B, \mathcal{D}_\eta \rangle) > cost_2(\langle \mathcal{D}_\eta, \mathcal{D}_A, \mathcal{D}_\eta \rangle)$. Finally, when Player 2 chooses $\mathcal{D}_A$, Player 1 prefers the design $\mathcal{D}_\rho$ iff the pair $\langle \eta, \rho \rangle$ is legal for φ, thus $cost_1(\langle \mathcal{D}_{ALL}, \mathcal{D}_A, \mathcal{D}_\eta \rangle) > cost_1(\langle \mathcal{D}_\rho, \mathcal{D}_A, \mathcal{D}_\eta \rangle)$, for a legal pair $\langle \eta, \rho \rangle$.

Thus, if $\varphi \in$ NAE, then for every assignment η, there is an assignment ρ such that $\langle \eta, \rho \rangle$ is a legal pair. Then, assuming Player 3 chooses a design $\mathcal{D}_\eta$, Player 1 prefers either choosing $\mathcal{D}_{ALL}$ or $\mathcal{D}_\rho$ over every other design, where $\langle \eta, \rho \rangle$ is a legal pair. By the above, there is no PNE in the game. If $\varphi \notin$ NAE, then there is an assignment η such that for every assignment ρ, the pair $\langle \eta, \rho \rangle$ is illegal. Then, the profile $\langle \mathcal{D}_{ALL}, \mathcal{D}_A, \mathcal{D}_\eta \rangle$ is a PNE, and we are done. ◀

References

1 H. Ackermann and A. Skopalik. Complexity of pure Nash equilibria in player-specific network congestion games. *Internet Mathematics*, 5(4):321–515, 2008.

2 S. Aland, D. Dumrauf, M. Gairing, B. Monien, and F. Schoppmann. Exact price of anarchy for polynomial congestion games. *SIAM J. Comput.*, 40(5):1211–1233, 2011.

3 E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. *SIAM J. Comput.*, 38(4):1602–1623, 2008.

4 G. Avni and O. Kupferman. Synthesis from component libraries with costs. In *Proc. 25th CONCUR*, LNCS 8704, pages 156–172. Springer, 2014.

5 G. Avni, O. Kupferman, and T. Tamir. Network-formation games with regular objectives. In *Proc. 17th FoSSaCS*, LNCS 8412, pages 119–133. Springer, 2014.

6 B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. *SIAM J. Comput.*, 42(1):160–177, 2013.

7 N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proc. 8th AAMAS*, 2009.

8 K. Bhawalkar, M. Gairing, and T. Roughgarden. Weighted congestion games: Price of anarchy, universal worst-case examples, and tightness. In *ESA (2)*, pages 17–28, 2010.

9 V. Bilò. A unifying tool for bounding the quality of non-cooperative solutions in weighted congestion games. In *WAOA*, pages 215–228, 2012.
10 I. Caragiannis, M. Flammini, C. Kaklamanis, P. Kanellopoulos, and L. Moscardelli. Tight bounds for selfish and greedy load balancing. *Algorithmica*, 61(3):606–637, 2011.
11 G. Christodoulou and M. Gairing. Price of stability in polynomial congestion games. In *Proc. 40th ICALP*, pages 496–507, 2013.
12 G. Christodoulou and E. Koutsoupias. On the price of anarchy and stability of correlated equilibria of linear congestion games. In *ESA*, pages 59–70, 2005.
13 N. Daniele, F. Guinchiglia, and M.Y. Vardi. Improved automata generation for linear temporal logic. In *Proc. 11th CAV*, LNCS 1633, pages 249–260. Springer, 1999.
14 J. Dunkel and A.S. Schulz. On the complexity of pure-strategy nash equilibria in congestion and local-effect games. *Mathematics of Operations Research*, 33(4):851–868, 2008.
15 C. Dwork and M. Naor. Pricing via processing, or, combatting junk mail. In *Proc. CRYPTO*, pages 139–177, 2009.
16 T. Eiter and G. Gottlob Note on the complexity of some eigenvector problems. Technical Report CD-TR 95/89, Christian Doppler Laboratory for Expert Systems, TU Vienna, 1995.
17 D. Fotakis, S. Kontogiannis, and P. Spirakis. Selfish unsplittable flows. *Theoretical Computer Science*, 348(2-3):226–239, 2005.
18 D. Fotakis, S. Kontogiannis, and P. Spirakis. Symmetry in Network Congestion Games: Pure Equilibria and Anarchy Cost. In *Proc. WAOA*, pages 161-175, 2005.
19 A. Fabrikant, C. Papadimitriou, and K. Talwar. The complexity of pure Nash equilibria. In *Proc. 36th STOC*, pages 604–612, 2004.
20 M. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. Freeman and Co., 1979.
21 T. Harks and M. Klimm. On the existence of pure Nash equilibria in weighted congestion games. *Math. Oper. Res.*, 37(3):419–436, 2012.
22 E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. *Computer Science Review*, 3(2):65–69, 2009.
23 Y. Lustig and M.Y. Vardi. Synthesis from component libraries. *STTT*, 15(5-6):603–618, 2013.
24 C. Meyers. *Network flow problems and congestion games: complexity and approximation results*. PhD thesis, MIT, 2006.
25 I. Milchtaich. Congestion games with player-specific payoff functions. *Games and Economic Behavior*, 13(1):111 – 124, 1996.
26 A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *POPL*, pages 179–190, 1989.
27 M.G. Reed, P.F. Syverson, and D.M. Goldschlag. Anonymous connections and onion routing. *IEEE J. on Selected Areas in Communication*, 1998. Issue on Copyright and Privacy Protection.
28 R.W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2:65–67, 1973.
29 T. Roughgarden and E. Tardos. How bad is selfish routing? *JACM*, 49(2):236–259, 2002.
30 L. Tran-Thanh, M. Polukarov, A. C. Chapman, A. Rogers, and N. R. Jennings. On the existence of pure strategy nash equilibria in integer-splittable weighted congestion games. In *SAGT*, pages 236–253, 2011.

Conference title on which this volume is based on.
Editors: Billy Editor, Bill Editors; pp. 14–14
LIPICS Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany