Repairing Multi-Player Games*

Shaull Almagor, Guy Avni, and Orna Kupferman

School of Computer Science and Engineering, The Hebrew University, Israel

Abstract

Synthesis is the automated construction of systems from their specifications. Modern systems often consist of interacting components, each having its own objective. The interaction among the components is modeled by a *multi-player game*. Strategies of the components induce a trace in the game, and the objective of each component is to force the game into a trace that satisfies its specification. This is modeled by augmenting the game with ω-regular winning conditions. Unlike traditional synthesis games, which are zero-sum, here the objectives of the components do not necessarily contradict each other. Accordingly, typical questions about these games concern their stability – whether the players reach an equilibrium, and their social welfare – maximizing the set of (possibly weighted) specifications that are satisfied.

We introduce and study *repair* of multi-player games. Given a game, we study the possibility of modifying the objectives of the players in order to obtain stability or to improve the social welfare. Specifically, we solve the problem of modifying the winning conditions in a given concurrent multi-player game in a way that guarantees the existence of a *Nash equilibrium*. Each modification has a value, reflecting both the cost of strengthening or weakening the underlying specifications, as well as the benefit of satisfying specifications in the obtained equilibrium. We seek optimal modifications, and we study the problem for various ω-regular objectives and various cost and benefit functions. We analyze the complexity of the problem in the general setting as well as in one with a fixed number of players. We also study two additional types of repair, namely redirection of transitions and control of a subset of the players.

1998 ACM Subject Classification J.4 Social and Behavioral Sciences, F.1.2 Modes of Computation

Keywords and phrases Nash Equilibrium, Concurrent games, Repair

Digital Object Identifier 10.4230/LIPIcs.CONCUR.2015.325

1 Introduction

Synthesis is the automated construction of systems from their specifications [19]. Modern systems often consist of interacting components, each having its own objective. The interaction among the components is modeled by a *multi-player game*. Each player in the game corresponds to a component in the interaction. In each round of the game, each of the players chooses an action, and the next vertex of the game depends on the current vertex and the vector of actions chosen. A strategy for a player is then a mapping from the history of the game so far to her next action.

The strategies of the players induce a trace in the game, and the goal of each player is to direct the game into a trace that satisfies her specification. This is modeled by augmenting the game with ω-regular winning conditions, describing the objectives of the players. Unlike traditional synthesis games, which are zero-sum, here the objectives of the

* The research has received funding from the European Research Council under the European Union's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no 278410.

26th International Conference on Concurrency Theory (CONCUR 2015).
Editors: Luca Aceto and David de Frutos Escrig; pp. 325–339

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

players do not necessarily contradict each other. Accordingly, typical questions about these games concern their stability – whether the players reach an equilibrium, and their social welfare – maximizing the set of (possibly weighted) specifications that are satisfied [23].

Different types of games can model different schemes of interaction among the components. In particular, we distinguish between *turn-based* and *concurrent* games. In the first, a single player chooses an action and determines the successor vertex in each step of the interaction. In the second, all players choose actions in all steps [1]. Another parameter is the way in which the winning conditions in the game are specified. Most common are *reachability*, *Büchi*, *co-Büchi*, and *parity* winning conditions [17], which are used to specify the set of winning traces. [1] As for stability and social welfare, here too, several types have been suggested and studied. The most common criterion for stability is the existence of a *Nash equilibrium* (NE) [18]. A profile of strategies, one for each player, is an NE if no (single) player can benefit from unilaterally changing her strategy. In the general setting of game theory, the outcome of a game fixes a reward to each of the players, thus "benefiting" stands for increasing the reward. In our setting here, the objective of a player is to satisfy her specification. Accordingly, "benefiting" amounts to moving from the set of losers – those players whose specifications are not satisfied, to the set of winners – those whose specifications are satisfied.

In [7, 22], the authors study the existence of an NE in games with Borel objectives. It turns out that while a turn-based game always has an NE [7, 22], this is not the case for concurrent games [2]. The problem of deciding whether a given concurrent game has an NE can be solved in polynomial time for Büchi games, but is NP-complete for reachability and co-Büchi games. Interestingly, this is one of the few examples in which reasoning about the Büchi acceptance condition is easier than reasoning about co-Büchi and reachability. The above results hold for a model with *nondeterministic* transition functions and with *imperfect monitoring*, where the players can observe the outcome of each transition and the vertex in which the game is, but cannot observe the actions taken by the other players [21]. In Remark 2.1 we elaborate on the difference between the two models. As we show in the paper, the results for reachability, co-Büchi, and Büchi stay valid also for our full-information model. For the parity condition, however, our model simplifies the setting and the problem of deciding the existence of an NE is NP-complete, as opposed to $\mathrm{P}_{\|}^{\mathrm{NP}}$ in the nondeterministic model with imperfect monitoring.

We introduce and study *repair* of multi-player games. We consider a setting with an authority (the designer) that aims to stabilize the interaction among the components and to increase the social welfare. In standard reactive synthesis [19], there are various ways to cope with situations when a specification is not realizable. Obviously, the specification has to be weakened, and this can be done either by adding assumptions on the behavior of the environment, fairness included, or by giving up some of the requirements on the system [6, 15]. In our setting, where the goal is to obtain stability, and the game is not zero-sum, a repair may both weaken and strengthen the specifications, which, in our main model, is modeled by modifications to the winning conditions.

The input to the *specification-repair problem* (SR problem, for short) is a game along with a *cost function*, describing the cost of each repair. For example, in Büchi games the cost function specifies, for each vertex v and player i, the cost of making v accepting for Player i and the cost of making v rejecting. The cost may be 0, reflecting the fact that v is accepting or rejecting in the original specification of Player i, or it may be ∞, reflecting the fact that

[1] A game may also involve incomplete information or stochastic transitions or strategies. The setting we consider here is not stochastic and players have full observability on the other players actions.

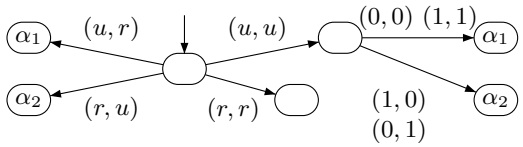

Figure 1 File sharing game. Initially, each player chooses to request either from the other user (action u) or from the repository (action r). In case both players choose u, the XOR game is played. The objective of Player i is to reach a vertex labeled α_i, in which case the other player sends her the file.

the original classification of v is a feature of the specification that the designer is not allowed to modify. We consider some useful classes of cost functions, like *uniform costs* – where all assignments cost 1, except for one that has cost 0 and stands for the original classification of the vertex, or *don't-care costs* – where several assignments have cost 0, reflecting a don't-care original classification, and all other assignments have cost ∞. In reachability, Büchi, and co-Büchi games, we also refer to one-way costs, where repair may only add or only remove vertices from the set of accepting vertices.

The goal of the designer is to suggest a repair to the winning conditions with which the game has an NE. One way to quantify the quality of a repair is its cost, and indeed the problem also gets as input a bound on the budget that can be used in the repairs. Another way, which has to do with the social welfare, considers the specifications that are satisfied in the obtained NE. Specifically, in the *rewarded specification-repair problem* (RSR problem, for short), the input also includes a *reward function* that maps subsets of specifications to rewards. When the suggested repair leads to an NE with a set W of winners, the designer gets a reward that corresponds to the specifications of the players in W. The quality of a solution then refers both to the budget it requires and to its reward. In particular, a reward function may prioritize the players and give a reward only to one player. Then, the question of finding an NE is similar to that of *rational synthesis*, where a winning strategy for the system can take into an account the objectives of the players that constitute the environment [9]. Thus, a special case of our contribution is repair of specifications in rational synthesis.

In [4], Brenguier describes several examples in which concurrent games and their stability model real-life scenarios. This includes peer-to-peer networks, wireless channel with a shared access, shared file systems, and more. The examples there also demonstrate the practicality of specification repair in these scenarios. We give an explicit example below.

▶ **Example 1.** Consider a file-sharing system serving two users. Each user requests a file from the other user or from a repository. Accessing the repository takes longer than transmitting between users, but the connection between the users can be used only in one direction at a time. If both users request the file from each other, they each choose a bit, and the file is transmitted to one of them according to the XOR of the bits. We model the interaction between the players as a reachability game, depicted in Fig. 1.

Observe that the game has no NE. Indeed, if w.l.o.g Player 1 does not reach α_1, then either Player 2 chose u and Player 1 lost in the XOR game, in which case Player 1 can deviate by choosing a different bit in the XOR, or Player 2 chose r, in which case Player 1 can deviate by playing u.

There are several ways to repair the game such that it has an NE. One is to break the symmetry between the players and make the vertex reached by playing (u,r) accepting for both players, and similarly for the vertex reached by playing (r,u). The cost involved in

this repair corresponds to the cost of communicating with the slower repository, and it is particularly useful when the reward function gives a priority to one of the players. Another possibility is to make the vertex reachable by playing (r, r) accepting for both players. Again, this involves a cost. ◀

Studying the SR and RSR problems, we distinguish between several classes, characterized by the type of winning conditions, cost functions, and reward functions. From a complexity point of view, we also distinguish between the case where the number of players is arbitrary and the one where it is constant. Recall that the problem of deciding whether an NE exists with an arbitrary number of players is NP-complete for reachability, co-Büchi, and parity games and can be solved in polynomial time for Büchi games. It is not too hard to lift the NP lower bound to the SR and RSR problems. The main challenge is the Büchi case, where one should find the cases where the polynomial complexity of deciding whether an NE exists can be lifted to the SR and RSR problems, and the cases where the need to find a repair shifts the complexity of the problem to NP. We show that the polynomial complexity can be maintained for don't-care costs, but the other settings are NP-complete. Our lower bounds make use of the fact that the unilateral change of a strategy that is examined in an NE can be linked to a change of the XOR of votes of all players, thus a single player can control the target of such transitions in a concurrent game.[2] We continue to study a setting with an arbitrary number of players. We check whether fixing the number of players can reduce the complexity of the SR and RSR problems, either by analyzing the complexity of the algorithms for an arbitrary number of players, or by introducing new algorithms. We show that in many cases, we can solve the problem in polynomial time, mainly thanks to the fact that it is possible to go over all possible subsets of players in search for a subset that can win in an NE.

In the context of verification, researchers have studied also other types of repairs (c.f., [11]). After focusing on the SR and RSR problems, we turn to study two other repair models. The first is *transition-repair*, in which a repair amounts to redirecting some of the transitions in the game. As with the SR problem, each redirection has a cost, and we seek repairs of minimal cost that would guarantee the existence of an NE. The transition-repair model is suitable in settings where the actions of the players do not induce a single successor state and we can choose between several alternatives. The second model we consider is that of *controlled-players*, in which we are allowed to dictate a strategy for some players. Also here, controlling players has a cost, and we want to minimize the cost and still guarantee the existence of an NE. We study several classes of the two types, and show that they are at least as difficult as specification repair.

Due to lack of space, most proofs can be found in the full version, in the authors' home pages.

2 Preliminaries

2.1 Concurrent games

A *concurrent game* is a tuple $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \{\alpha_i\}_{i\in\Omega}\rangle$, where Ω is a set of k players; V is a set of vertices; A is a set of actions, partitioned into sets A_i of actions for Player i, for

[2] We note that while the representation of our games is big, as the transition function specifies all vectors of actions, our complexity results hold also for games with a succinct representation of the transition function, in particular games with an arbitrary number of players in which only a constant number of players proceed in each vertex. The complexity of finding NE in succinctly represented games was studied in [10]. Succinctly represented games were studied in [16] the context of ATL model checking.

$i \in \Omega$; $v_0 \in V$ is an initial vertex; $\delta : V \times A_1 \times A_2 \times \cdots \times A_k \to V$ is a transition function, mapping a vertex and actions taken by the players to a successor vertex; and α_i, for $i \in \Omega$, specifies the objective for Player i. We describe several types of objectives in the sequel. For $v, v' \in V$ and $\overline{a} \in A_1 \times A_2 \times \cdots \times A_k$ with $\delta(v, \overline{a}) = v'$, we sometimes refer to $\langle v, \overline{a}, v' \rangle$ as a transition in $\mathcal{G}$.

A *strategy* for Player i is a function $\pi_i : (A_1 \times ... \times A_k)^* \to A_i$, which directs Player i which action to take, given the history of the game so far. Note that the history is given by means of the sequence of actions taken by all players so far.[3]

A *profile* is a tuple $P = \langle \pi_1, ..., \pi_k \rangle$ of strategies, one for each player. The profile P induces a sequence $\overline{a}_0, \overline{a}_1, \ldots \in (A_1 \times ... \times A_k)^\omega$ as follows: $\overline{a}_0 = \langle \pi_1(\epsilon), ..., \pi_k(\epsilon) \rangle$ and for every $i > 0$ we have $\overline{a}_i = \langle \pi_1(\overline{a}_0, ..., \overline{a}_{i-1}), ..., \pi_k(\overline{a}_0, ..., \overline{a}_{i-1}) \rangle$. For a profile P we define its *outcome* $\tau = \mathit{outcome}(P) \in V^\omega$ to be the path of vertices in $\mathcal{G}$ that is taken when all the players follow their strategies in P. Formally, $\tau = v_0, v_1, ...$ starts in v_0 and proceeds according to δ, thus $v_{i+1} = \delta(v_i, \overline{a}_i)$. The set of *winners* in P, denoted $W(P) \subseteq \Omega$, is the set of players whose objective is satisfied in $\mathit{outcome}(P)$. The set of *losers* in P, denote $L(P)$, is then $\Omega \setminus W(P)$, namely the set of players whose objective is not satisfied in $\mathit{outcome}(P)$.

A profile $P = \langle \pi_1, ..., \pi_k \rangle$ is a *Nash equilibrium* (NE, for short) if, intuitively, no (single) player can benefit from unilaterally changing her strategy. In the general setting, the outcome of P associates a reward with each of the players, thus "benefiting" stands for increasing the reward. In our setting here, the objective of Player i is binary – either α_i is satisfied or not. Accordingly, "benefiting" amounts to moving from the set of losers to the set of winners. Formally, for $i \in \Omega$ and some strategy π'_i for Player i, let $P[i \leftarrow \pi'_i] = \langle \pi_1, \ldots, \pi_{i-1}, \pi'_i, \pi_{i+1}, ..., \pi_k \rangle$ be the profile in which Player i *deviates* to the strategy π'_i. We say that P is an NE if for every $i \in \Omega$, if $i \in L(P)$, then for every strategy π'_i we have $i \in L(P[i \leftarrow \pi'_i])$.

We consider the following types of objectives. Let $\tau \in V^\omega$ be an infinite path.

- In *reachability* games, $\alpha_i \subseteq V$, and τ satisfies α_i if τ reaches α_i.
- In *Büchi* games, $\alpha_i \subseteq V$, and τ satisfies α_i if τ visits α_i infinitely often.
- In *co-Büchi* games, $\alpha_i \subseteq V$, and τ satisfies α_i if τ visits $V \setminus \alpha_i$ only finitely often.
- In *parity* games, $\alpha_i : V \to \{1, ..., d\}$, for the *index* d of the game, and τ satisfies α_i if the maximal rank that is visited by τ infinitely often is even. Formally, let $\tau = v_0, v_1, ...$, then τ satisfies α_i if $\max\{j \in \{1, ..., d\} : \alpha_i(v_l) = j \text{ for infinitely many } l \geq 0\}$ is even.

Note that Büchi and co-Büchi games are special cases of parity games, with ranks $\{1, 2\}$ and $\{2, 3\}$, respectively. We sometimes refer to a winning condition $\alpha_i \subseteq V$ also as a function $\alpha_i : V \to \{\top, \bot\}$, with $\alpha_i(v) = \top$ iff $v \in \alpha_i$.

▶ Remark. Our definition of strategy is based on the history of actions played. This is different from the setting in [4], where strategies are based on the history of visited vertices. Our setting reflects the fact that players have full knowledge of the actions played by other players, and not only the outcome of these actions. As we now demonstrate, our setting is different as it enables the players to make use of this full knowledge to obtain an NE. In Section 2.4 we elaborate on the algorithmic differences between the settings.

Consider the concurrent three-player Büchi game $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \{\alpha_i\}_{i \in \Omega} \rangle$, where $\Omega = \{1, 2, 3\}$, $V = \{v_0, v_1, a, b, c\}$, $A_i = \{0, 1\}$ for $i \in \Omega$, $\alpha_1 = \{a\}, \alpha_2 = \{b\}, \alpha_3 = \{c\}$, and the transition function is as follows. In v_0, if 1 and 2 play $(0, 0)$, the game moves to c, and

[3] Note that strategies observe the history of actions, rather than the history of vertices. In Remark 2.1 we elaborate on this aspect.

otherwise to v_1. In v_1, Player 3 can choose to go to a or to b. The vertices a, b, and c are sinks.

There is an NE in $\mathcal{G}$, whose outcome is the path v_0, c^ω. That is, players 1 and 2 play $(0, 0)$. However, in order for this to be an NE profile, Player 3 needs to be able to "punish" either Player 1 or Player 2 if they deviate to v_1. For that, Player 3 needs to know the action that leads to v_1: if Player 1 deviates, then Player 3 chooses to proceed to b, and if Player 2 deviates, then Player 3 chooses to proceed to a. If we consider strategies that refer to histories of vertices, then there is no NE in the game. ◀

2.2 Partial games with costs and rewards

Let $\mathbb{N}_\infty = \mathbb{N} \cup \{\infty\}$. A *partial concurrent parity game* $\mathcal{G}$ is a concurrent parity game in which the winning conditions are replaced by a cost function that describes the cost of augmenting $\mathcal{G}$ with different winning conditions. Formally, $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \mathit{cost} \rangle$, where the cost function $\mathit{cost} : V \times \Omega \times \{1, ..., d\} \to \mathbb{N}_\infty$ states, for each vertex $v \in V$, player $i \in \Omega$, and rank $j \in \{1, ..., d\}$, what the cost of setting $\alpha_i(v)$ to be j. We can think of a concrete game (one with fully specified winning conditions α_i, for $i \in \Omega$) as a partial game in which the cost function is such that $\mathit{cost}(v, i, j) = 0$ if $\alpha_i(v) = j$ and $\mathit{cost}(v, i, j) = \infty$ otherwise. Intuitively, leaving $\alpha_i(v)$ as specified is free of charge, and changing $\alpha_i(v)$ is impossible, as it costs ∞. Partial games enable us to model settings where a designer can play with the definition of the winning conditions, subject to some cost function and a given budget.

Consider a partial parity game $\mathcal{G}$. A *winning-condition assignment* for $\mathcal{G}$ is $f : V \times \Omega \to \{1, ..., d\}$. The parity game induced by $\mathcal{G}$ and f, denoted $\mathcal{G}^f$, has $\alpha_i(v) = f(v, i)$, for all $v \in V$ and $i \in \Omega$. The *cost* of f is $\mathit{cost}(f) = \sum_{i \in \Omega} \sum_{v \in V} \mathit{cost}(v, i, f(v, i))$.

In the case of reachability, Büchi, and co-Büchi, the cost function is $\mathit{cost} : V \times \Omega \times \{\top, \bot\} \to \mathbb{N}_\infty$, and the winning-condition assignment is of the form $f : V \times \Omega \to \{\top, \bot\}$.

Consider a partial game $\mathcal{G}$ and a cost function *cost*. For every player $i \in \Omega$ and vertex $v \in V$, we define the set $\mathit{free}_{\mathit{cost}}(v, i) \subseteq \{1, ..., d\}$ as the set of ranks we can assign to $\alpha_i(v)$ free of charge. Formally, $\mathit{free}_{\mathit{cost}}(v, i) = \{j : \mathit{cost}(v, i, j) = 0\}$. We consider the following two classes of cost functions in parity games.

- *Uniform costs:* For every $i \in \Omega$ and $v \in V$, we have $|\mathit{free}_{\mathit{cost}}(v, i)| = 1$ and for every $j \notin \mathit{free}_{\mathit{cost}}(v, i)$, we have $\mathit{cost}(v, i, j) = 1$. Thus, a partial game with a uniform cost function corresponds to a concrete game in which we can modify the winning condition with a uniform cost of 1 for each modification.
- *Don't cares:* For every $i \in \Omega$, $v \in V$, and $j \in \{1, ..., d\}$, we have $\mathit{cost}(v, i, j) \in \{0, \infty\}$, and $|\mathit{free}_{\mathit{cost}}(v, i)| \geq 1$. Thus, as in concrete games, we cannot modify the rank of vertices that are not in $\mathit{free}_{\mathit{cost}}(v, i)$, but unlike concrete games, here $\mathit{free}_{\mathit{cost}}(v, i)$ need not be a singleton, reflecting a situation with "don't cares", where a designer can choose among several possible ranks free of charge.

For the special case of reachability, Büchi, and co-Büchi games, we also consider the following classes.

- *Negative one-way costs:* For every $i \in \Omega$ and $v \in V$, either $\mathit{cost}(v, i, \top) = 0$ and $\mathit{cost}(v, i, \bot) = 1$, or $\mathit{cost}(v, i, \bot) = 0$ and $\mathit{cost}(v, i, \top) = \infty$. Intuitively, we are allowed only to modify $\top$ vertices to $\bot$ ones, thus we are only allowed to make satisfaction harder by removing vertices from α_i.
- *Positive One-way costs:* For every $i \in \Omega$ and $v \in V$, either $\mathit{cost}(v, i, \top) = 0$ and $\mathit{cost}(v, i, \bot) = \infty$, or $\mathit{cost}(v, i, \bot) = 0$ and $\mathit{cost}(v, i, \top) = 1$. Intuitively, we are allowed only to modify $\bot$ vertices to $\top$ ones, thus we are only allowed to make satisfaction easier by adding vertices to α_i.

Reward function. Consider a game $\mathcal{G}$. A *reward function* for $\mathcal{G}$ is $\zeta : 2^\Omega \to \mathbb{N}$. Intuitively, if the players follow a profile P of strategies, then the reward to the designer is $\zeta(W(P))$. Thus, a designer has an incentive to suggest to the players a stable profile of strategies that maximizes her reward. We assume that ζ is monotone w.r.t. containment.

2.3 The specification-repair problem

Given a partial game $\mathcal{G}$ and a threshold $p \in \mathbb{N}$, the *specification-repair problem* (SR problem, for short) is to find a winning-condition assignment f such that $cost(f) \leq p$ and $\mathcal{G}^f$ has an NE. Thus, we are willing to invest at most p in order to be able to suggest to the players a stable profile of strategies.

In the *rewarded specification-repair problem* (RSR problem, for short) we are also given a reward function ζ and a threshold q, and the goal is to find a winning-condition assignment f such that $cost(f) \leq p$ and $\mathcal{G}^f$ has an NE with a winning set of players W for which $\zeta(W) \geq q$.

▶ Remark. An alternative definition to the RSR problem would have required all NEs in $\mathcal{G}^f$ to have a reward greater than q. This is similar to the cooperative vs. non-cooperative definitions of rational synthesis [9, 14]. In the cooperative setting, which we follow here, we assume that the authority can suggest a profile of strategies to the players, and if this profile is an NE, then they would follow it. In the non-cooperative one, the authority cannot count on the players to follow its suggested profile even if it is an NE. We find the cooperative setting more realistic, especially in the context of repairs, which assumes rational cooperative agents (indeed, they are willing to apply a repair for a cost). Moreover, all existing work in Algorithmic Game Theory follow the cooperative setting in games that are similar to the ones we study.

Also, rather than including in the input to the RSR problem two thresholds, one could require that $\zeta(W) \geq cost(f)$ or to compare $\zeta(W)$ with $cost(f)$ in some other way. Our results hold also for such definitions. ◀

We distinguish between several classes of the SR and RSR problems, characterized by the type of winning conditions, cost function, and reward function. From a complexity point of view, we also distinguish between the case where the number of players is arbitrary and the one where it is constant.

▶ Remark. Another complexity issue has to do with the size of the representation of the game. Recall that, specifying $\mathcal{G}$, we need to specify the transition $\delta(v, \overline{a})$ for every vertex $v \in V$ and action vector $\overline{a} \in A^{|\Omega|}$. Thus, the description of $\mathcal{G}$ is exponential in the size of Ω. While this may make the lower bounds more challenging, it may also makes polynomial upper bounds easy. In Remark 3.1 we argue that our complexity results hold also in a settings with a succinct representation of $\mathcal{G}$. For example, when $\mathcal{G}$ is *c-concurrent* for some $c \geq 1$, meaning that in each vertex, only c players *control* the vertex. That is, in each vertex only c players choose actions and determine the successor vertex. Then, the size of δ is bounded by $|V \times A^c|$, for a constant c. ◀

2.4 Deciding the existence of an NE

The problem of deciding the existence of an NE, which is strongly related to the SR problem was studied in [4]. The model there subsumes our model. First, as discussed in Remark 2.1, our strategies have full knowledge of actions, whereas the strategies in [4] only observe vertices. Second, the transition function in [4] is *nondeterministic*, thus a vertex and a vector of actions are mapped to a set of possible successors. We can efficiently convert a game in

our model into an "equivalent" game in the model of [4] (in the sense that the existence of a NE is preserved). Thus, algorithmic upper bounds from [4] apply to our setting as well. Conversely, however, lower bounds from [4] do not apply to our model, and indeed the lower bounds we show differ from those of [4].

Specifically, it is shown in [4, 3] that the problem of deciding whether a given game has an NE is $\mathrm{P}_{\|}^{\mathrm{NP}}$-complete for parity objectives; that is, it can be solved in polynomial time with parallel queries to an NP oracle. The problem is NP-complete for reachability and co-Büchi objectives, and can be solved in polynomial time for Büchi games. We show that in our model, while the complexity of the problem for reachability, Büchi, and co-Büchi objectives coincides with that of [4], the complexity for parity objectives is NP-complete. In Section 3.1 we present Theorem 5, which entails an explicit algorithm for deciding the existence of an NE in Büchi games. Our algorithm is significantly simpler than the one in [4] as it considers a deterministic model.

Additionally, we emphasize that the main contribution of this work is the introduction of repairs, and our choice of model is in part for its clarity. Indeed, repair can similarly be defined in the model of [4], as partial observation is an orthogonal notion.

In the full version we prove the following theorem.

▶ **Theorem 2.** *The problem of deciding whether a concurrent reachability, co-Büchi, or parity game has an NE is NP-complete.*

In particular, we note that the problem of *verifying* the existence of an NE can be solved in polynomial time, using an appropriate witness. See the full version for details.

3 Solving the SR and RSR Problems

3.1 An Arbitrary Number of Players

In this section we consider the SR problem for an arbitrary number of players. Recall that the problem of deciding whether an NE exists is NP-hard for reachability, co-Büchi, and parity games and is in P for Büchi games. It is not too hard to lift the NP lower bound to the SR problem. The main challenge is the Büchi case, where one should find the cases where the polynomial complexity of deciding whether an NE exists can be lifted to the SR problem, and the cases where the need to find a repair shifts the complexity of the problem to NP.

▶ **Theorem 3.** *The SR problem for reachability, co-Büchi, and parity games with uniform, don't cares, positive one-way, or negative one-way costs is NP-complete.*

Proof. Membership in NP is easy, as given a game $\mathcal{G}$, a cost function *cost*, and a threshold p, we can guess a winning-condition assignment f, and then proceed to nondeterministically check whether there exists an NE in $\mathcal{G}^f$ as described in Section 2.4.

For the lower bound, we describe a reduction from the problem of deciding whether an NE exists in a given co-Büchi or reachability game is NP-complete, proved to be NP-hard in Theorem 2.

Consider a game $\mathcal{G}$, and let *cost* be the cost function induced naturally by it. That is, for every $v \in V$ and $i \in \Omega$, we have $cost(v, i, \alpha_i(v)) = 0$, and the rest of the cost function is defined to involve a positive cost and respect the definition of uniform, don't cares, positive one-way, or negative one-way cost function. With this cost function, the only assignment with cost 0 is such that $f(v, i) = \alpha_i(v)$ for every $i \in \Omega$ and $v \in V$. Thus, $\mathcal{G}$ has an NE iff there is a winning-condition assignment f such that $cost(f) \leq 0$ and $\mathcal{G}^f$ has an NE. ◀

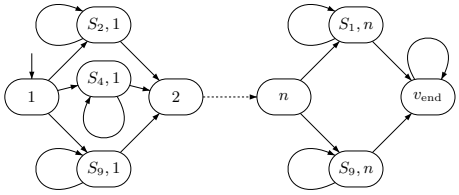

Figure 2 Reduction in the uniform costs setting in Theorem 4. Here, $1 \in S_2 \cap S_4 \cap S_9$, and $n \in S_1 \cap S_9$.

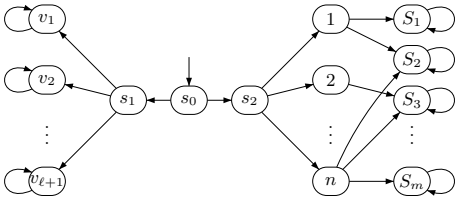

Figure 3 Reduction of the negative one-way costs setting in Theorem 4. Here, $1 \in S_1 \cap S_2$, $2 \in S_3$ and $n \in S_2 \cap S_3 \cap S_m$.

We turn to Büchi games, where the goal is to find the cases where the polynomial complexity of deciding the existence of an NE can be maintained. We start with the negative cases.

▶ **Theorem 4.** *The SR problem for Büchi games with uniform, positive one-way, or negative one-way costs is NP-complete.*

Proof. Membership in NP is easy, as given a game $\mathcal{G}$, a cost function *cost*, and a threshold p, we can guess a winning-condition assignment f, and then check in polynomial time whether there exists an NE in $\mathcal{G}^f$ [4]. For the lower bounds we describe reductions from SET-COVER, which is well known to be NP-complete [12]. We bring its definition here for completeness. Consider a set $U = \{1, \ldots, n\}$ of elements and a set $S = \{S_1, ..., S_m\}$ of subsets of U, thus $S_i \subseteq U$ for every $1 \leq j \leq m$. A set-cover of size ℓ is $\{S_{j_1}, ..., S_{j_\ell}\} \subseteq S$ such that for every $i \in U$ there exists $1 \leq l \leq \ell$ such that $i \in S_{j_l}$. The SET-COVER problem is to decide, given U, S, and ℓ, whether there exists a set-cover of size ℓ. We assume w.l.o.g that $\ell < \min\{n, m\}$.

Uniform costs. Consider an input $\langle U, S, \ell\rangle$ for SET-COVER. We construct a partial concurrent game $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \mathit{cost}\rangle$ such that there is a set cover of U of size ℓ iff there exists a winning-condition assignment f with $\mathit{cost}(f) \leq \ell$ such that $\mathcal{G}^f$ has an NE.

The players in $\mathcal{G}$ are $\Omega = U \cup S$. That is, there is one player, referred to as Player i, for every $i \in U$, and one player, referred to as Player S_j, for every $S_j \in S$. The set of vertices in $\mathcal{G}$ is $V = U \cup \{\langle S_j, i\rangle : i \in S_j \in S\} \cup \{v_{\text{end}}\}$. The initial vertex is $1 \in U$. We now describe the transitions and actions (see Fig. 2).

At vertex $i \in U$, Player i alone has control, in the sense that only her action is taken into an account in deciding the successor. Player i can choose to move to a vertex $\langle S_j, i\rangle$ for which $i \in S_j$. At vertex $\langle S_j, i\rangle$, all players in $U \cup \{S_j\}$ have control on the choice of the successor vertex and can choose either to stay at $\langle S_j, i\rangle$, or to proceed, either to vertex $i + 1$, if $i < n$, or to v_{end}, if $i = n$. This choice is made as follows. The actions of the players are $\{0, 1\}$, and the transition depends on the XOR of the actions. If the XOR is 0, then the game stays in $\langle S_j, i\rangle$ and if the XOR is 1, the game proceeds to $i + 1$ or to v_{end}. Finally, v_{end} has a self loop.

We now describe the cost function. Intuitively, we define *cost* so that the default for v_{end} is to be accepting for all players $i \in U$ and rejecting for all players $S_j \in S$. Thus, $\mathit{cost}(v_{\text{end}}, i, \top) = 0$ for all $i \in U$ and $\mathit{cost}(v_{\text{end}}, S_j, \bot) = 0$ for all $S_j \in S$. Also, for every $S_j \in S$ and $i \in S_j$, we have $\mathit{cost}(\langle S_j, i\rangle, S_j, \top) = 0$ and $\mathit{cost}(\langle S_j, i\rangle, i, \bot) = 0$. Thus, $\langle S_j, i\rangle$ is accepting for Player S_j and is rejecting for Player i. All other costs are set to 1, as required by a uniform cost.

We claim that $\langle U, S, \ell\rangle \in$ SET-COVER iff $\mathcal{G}$ has a winning-condition assignment f with cost at most ℓ such that $\mathcal{G}^f$ has an NE. In the full version we formally prove the correctness of the reduction. Intuitively, every assignment f of cost at most ℓ must set $f(v_{\text{end}}, i) = \top$

and $f(v,i) = \bot$ for $v \neq v_{\text{end}}$, for some $i \in U$. Thus, an NE must end in v_{end}, as otherwise Player i uses the XOR transitions in order to deviate to a strategy whose outcome reaches v_{end}. Hence, an assignment must set $f(v_{\text{end}}, S_j) = \top$ for at most ℓ players $S_{j_1}, ..., S_{j_\ell}$, such that it is possible to get from 1 to v_{end} by going only through vertices $\langle S_{j_k}, i\rangle$ for $1 \leq k \leq \ell$. These ℓ players induce a set cover. The other direction is easy.

Positive one-way costs. In the correctness proof of the reduction above we show that in fact, the only assignments that need to be considered are positive one-way. Thus, the same reduction, in fact with a simpler correctness argument, can be used to show NP-hardness of the setting with positive one-way costs.

Negative one-way costs. Finally, we consider the setting of negative one-way costs. Again, we describe a reduction from SET-COVER. Consider an input $\langle U, S, \ell\rangle$ for SET-COVER. We construct a partial two-player game $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \mathit{cost}\rangle$ such that there is a set cover of U of size ℓ iff there exists a winning-condition assignment f with $\mathit{cost}(f) \leq \ell$ such that $\mathcal{G}^f$ has an NE. The game $\mathcal{G}$ is constructed as follows. The players are $\Omega = \{1, 2\}$. The vertices are $V = U \cup S \cup \{s_0, s_1, s_2\} \cup \{v_1, ..., v_{\ell+1}\}$. The game starts in s_0, where the actions for the players are $\{0, 1\}$. If the XOR of the actions is 0, the game moves to vertex s_1, where Player 1 chooses a vertex from $v_1, ..., v_{\ell+1}$, all of which have self loops. We set $\mathit{cost}(v_i, 1, \top) = 0$ for $1 \leq i \leq \ell + 1$. Intuitively, if the game proceeds to s_1, then Player 1 can choose a winning vertex, and the play gets stuck there. If the XOR in s_0 was 1, the game proceeds to vertex s_2 from which player 2 chooses a vertex $i \in U$. In vertex i, player 1 chooses a vertex S_j such that $i \in S_j$. For every $1 \leq j \leq m$, the vertex S_j has only a self loop. We set $\mathit{cost}(S_j, 2, \top) = 0$ for $1 \leq j \leq m$. Intuitively, if the game proceeds to s_2, then Player 2 "challenges" Player 1 with a value $i \in U$, and Player 1 has to respond with some set S_j such that $i \in S_j$, and then the play gets stuck in S_j. See Fig. 3 for an illustration.

The rest of the *cost* function is set to give $\bot$ cost 0, and is completed to be a negative one-way cost. That is, we set $\mathit{cost}(v_j, 2, \bot) = 0$ for every $1 \leq j \leq \ell + 1$, $\mathit{cost}(S_j, 2, \bot) = 0$ for every $1 \leq j \leq m$, and $\mathit{cost}(x, 1, \bot) = \mathit{cost}(x, 2, \bot) = 0$ for $x \in U \cup \{s_0, s_1, s_2\}$. Finally, $\mathit{cost}(v, i, \bot) = 1$ if $\mathit{cost}(v, i, \top) = 0$ and $\mathit{cost}(v, i, \top) = \infty$ if $\mathit{cost}(v, i, \bot) = 0$, for every $i \in \{1, 2\}$ and $v \in V$, as per the definition of a negative one-way cost.

In the full version we formally prove the correctness of the reduction. Intuitively, every assignment f of cost at most ℓ must set $f(v_i, 1) = \top$ for some $1 \leq i \leq \ell + 1$. Thus, Player 1 is guaranteed to be able to deviate and win in any profile. In order to have an NE, we must be able to set $f(S_j, 2) = \bot$ for at most ℓ vertices $S_{j_1}, ..., S_{j_\ell}$, such that for every $i \in U$ that Player 2 chooses, there exists $1 \leq k \leq \ell$ such that $i \in S_{j_k}$, and so there is a set-cover. The other direction is again, easy. ◀

We now turn to consider the positive case, where the polynomial complexity of deciding whether an NE exists can be lifted to the SR problem.

▶ **Theorem 5.** *The SR problem for Büchi games and don't-cares can be solved in polynomial time.*

Proof. Consider a partial Büchi game $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \mathit{cost}\rangle$ with don't-cares. For every $i \in \Omega$, the set of vertices V can be partitioned into three sets:

1. The set $F_i = \{v : \mathit{cost}(v, i, \top) = 0 \wedge \mathit{cost}(v, i, \bot) = \infty\}$, of *accepting* vertices.
2. The set $R_i = \{v : \mathit{cost}(v, i, \bot) = 0 \wedge \mathit{cost}(v, i, \top) = \infty\}$, of *rejecting* vertices.
3. The set $DC_i = \{v : \mathit{cost}(v, i, \bot) = \mathit{cost}(v, i, \top) = 0\}$, of *don't-care* vertices.

The SR problem then amounts to deciding whether there is an assignment $f : \bigcup_{i\in\Omega} DC_i \to \{\top, \bot\}$ such that $\mathcal{G}^f$ has an NE. Note the cost of every such assignment is 0.

For a set $S \subseteq V$, let $W_S \subseteq \Omega$ be the set of *potential winners* in S: players that either have an accepting or don't-care vertex in S. Formally, $W_S = \{i \in \Omega : (F_i \cup DC_i) \cap S \neq \emptyset\}$. The set of *losers* in S is then $L_S = \Omega \setminus W_S$, thus $i \in L_S$ iff $S \subseteq R_i$.

We describe the intuition behind our algorithm. An outcome of a profile is an infinite path in $\mathcal{G}$, which gets stuck in a SCC S. We distinguish between the case S is an ergodic SCC – one that has no outgoing edges to other SCCs in $\mathcal{G}$, and the case S is not ergodic. Our algorithm tries to find a *witness* ergodic SCC S: one for which there is an assignment f such that $\mathcal{G}^f$ has an NE whose outcome gets stuck in S. When an ergodic SCC cannot serve as a witness, it is removed from $\mathcal{G}$ along with transitions that guarantee the soundness of such a removal, and the search for a witness ergodic SCC in the new game continues. When all SCCs are removed, the algorithm concludes that no NE exists.

In order to examine whether an ergodic SCC S can serve as a witness, the algorithm checks whether the players in W_S can force the game to reach S. Once the game reaches S, every outcome would not satisfy the objective of the players in L_S. Moreover, consider the assignment f that sets, for $i \in W_S$, every vertex in DC_i to $\top$. The profile whose outcome visits all the vertices in S is an NE in G^f. Checking whether the players W_S can force the game to reach S is not straightforward, as it should take into an account possible collaboration from players in L_S that are doomed to lose anyway and thus have no incentive to deviate from a strategy in which they collaborate with the players in W_S.

Formalizing this intuition involves the following definitions. Consider a player $i \in \Omega$. We define the *game against* i to be a two-player zero-sum concurrent game, where the players are Player i and the coalition $\Omega \setminus \{i\}$. The game is played on $\mathcal{G}$, where the objective of player i is α_i, and the objective of $\Omega \setminus \{i\}$ is to prevent i from satisfying α_i. The *cage* for player i is the set of vertices $C_i \subseteq V$ that consists of all vertices from which the coalition wins the game against i.

Deciding whether a vertex v is in C_i amounts to solving a two-player zero-sum concurrent game. These games can be solved in polynomial time for reachability, Büchi, and co-Büchi objectives [8].

Next, consider a transition $t = \langle v, \overline{a}, v' \rangle$ with $v, v' \in C_i$, thus $\delta(v, \overline{a}) = v'$. We say that t is *doomed for Player* i if Player i cannot alter her action in $\overline{a}$ and escape the cage C_i. We denote by doomed(B) the set of transitions that are doomed for all players in B.

For a game with don't-cares, whenever we consider the game against i, we refer to the concrete two-player games obtained from $\mathcal{G}$ with the assignment that assigns $\bot$ to the vertices in DC_i.

Our algorithm checks whether there is a path τ to S that traverses only transitions in doomed(L_S). If so, it concludes that $\mathcal{G}$ can be repaired to have an NE with the assignment f that is defined as follows. For every $v \in V$, for $j \in L_S$, we have $f(v, j) = \bot$ and, for $j \in W_S$, we have $f(v, j) = \top$. Indeed, the profile whose outcome is τ followed by a path that visits all the vertices in S infinitely often, and which punishes a player that deviates from her expected action in τ is an NE in $\mathcal{G}^f$. ◀

▶ Remark. As discussed in Remark 2.3, our results stay valid when the games are c-concurrent for a constant $c \geq 2$. In particular, the running time of the algorithm described in Theorem 5 is polynomial in the representation size of $\mathcal{G}$. As for lower bounds, the second reduction described in the proof of Theorem 4 generates a game with only two players. In addition, the first reduction there can be slightly modified to capture 2-concurrent games. For that, we replace the vertices $S \times U$ in $\mathcal{G}$ by a cycle of n vertices, where $\langle S_j, i \rangle$ is the first vertex in the cycle $\langle S_j, i \rangle_1, ..., \langle S_j, i \rangle_n$. The players that control the l-th vertex, for $1 \leq l \leq n$, are S_j and l. Both players have two possible actions $\{0, 1\}$. If the XOR of their choices is 0, the

Table 1 Complexity results for the setting with a constant number of players.

Problem ╲ Game	Büchi	co-Büchi	Reachability	Parity
NE Existence	P [3]	P	P [2]	NP∩coNP
Uniform	P	NP-C	P	NP-C
Don't care	P	P	P	NP∩coNP
Negative One-way	NP-C			–
Positive One-way	P	P	P	–

game continues to $(l+1) \mod n$, the next vertex in the cycle, and if it is 1, then the game exits the cycle and proceeds to vertex $i+1$. Clearly, each player in $U \cup \{S_j\}$ can force the game to stay in the gadget or exit it assuming the other players fix a strategy.

3.2 A Constant Number of Players

In this section, we consider the SR problem for a constant number of players. The algorithms presented in Section 3.1 can be clearly applied in this setting. For example, Theorem 5 implies that the SR problem for Büchi games with don't cares can be solved in polynomial time, and in particular this holds when the number of players is fixed. For NP-complete problems, however, the upper bounds in Section 3.1 only imply exponential time algorithms. In this section, we check whether fixing the number of players can reduce the complexity, either by analyzing the complexity of the algorithms from Section 3.1, or by introducing new algorithms.

The results are summarized in Table 1, and the proofs appear in the full version with the exception of Theorem 6 below.

▶ **Theorem 6.** *The SR problem for co-Büchi games with positive one-way costs and a constant number of players can be solved in polynomial time.*

Proof. We solve the problem by presenting a polynomial time algorithm for checking, given a game $\mathcal{G}$, a bound $p \in \mathbb{N}$ on the budget for the repair, and a set $W \subseteq \Omega$, whether there is a positive one-way assignment f with cost at most p, for which $\mathcal{G}^f$ has an NE profile P with $W \subseteq W(P)$. We then iterate over all subsets $W \subseteq \Omega$ to obtain a polynomial time algorithm.

Under the definitions used in the proof of Theorem 5, let $L = \Omega \setminus W$, and let $\mathcal{G}_W = \mathcal{G}|_{\text{doomed}(L)}$. Consider a vertex $v \in V$ that is reachable from v_0 in $\mathcal{G}_W$. We look for an assignment f for which there is a cycle that contains v and traverses only vertices in $\bigcap_{i \in W} \alpha_i^f$. Such a cycle satisfies the objectives of the players in W. In order to do so, we add weights to $\mathcal{G}_W$ as follows. The weight of an edge $\langle u, u' \rangle$ in $\mathcal{G}_W$ is the repair budget that is needed in order to make u' accepting for all players in W. That is, $\langle u, u' \rangle$ gets the weight $\sum_{i \in W} cost(u', i, \top)$. Then, we run Dijkstra's shortest-path algorithm from v to find the minimal-weight cycle that contains v. If the weight of the cycle is at most p, we return "yes". If there is no such cycle for every $v \in V$ and $W \subseteq \Omega$, we return "no". We then repeat this process for every $W \subseteq \Omega$.

In the full version we analyze the runtime and prove the correctness of the algorithm. ◀

3.3 Solving the RSR problem

Recall that in the RSR problem we are given, in addition to $\mathcal{G}$, *cost*, and $p \in \mathbb{N}$, a reward function $\zeta : 2^\Omega \to \mathbb{N}$ and a threshold q, and we need to decide whether we can repair $\mathcal{G}$ with cost at most p in a way that the set of winners W in the obtained NE is such that $\zeta(W) \geq q$.

In the full version we argue that the additional requirement about the reward maintains the complexity of the problem.

▶ **Theorem 7.** *The complexity of the SR and RSR problems coincide for all classes of objectives and cost functions, for both an arbitrary and a constant number of players.*

4 Other Types of Repairs

So far, we studied repairs that modify the winning conditions of the players. Other types of repairs can be considered. In this section, we examine two such types: *transition repair*, which modifies the transitions of the game, and *controlled-players repair*, where we can control (that is, force a strategy) on a subset of the players. The later is related to the *Stackelberg model*, which has been extensively studied in economics and more recently in Algorithmic Game Theory [13, 20], and in which some of the players are selfish whereas others are controllable.

4.1 Transition repair

In the *transition repair* model, we are allowed to redirect the transitions of a game. This is suitable in cases where a system is composed of several concurrent components, and we have some control on the flow of the entire composition. For example, consider a system in which several threads request a lock and granting a lock to a certain thread is modeled by a transition. Redirecting this transition can correspond to the lock being given to a different thread. Typically, not all repairs are possible, which is going to be modeled by an ∞ cost to impossible repairs. Finally, the games we study are sometimes obtained from LTL specifications of the players. Repairs in the winning conditions then have the flavor of switching between "until" and "weak-until" in the LTL specification. In this setting, one may find transition-repair to be more appropriate. First, it enables more elaborate changes in the specifications. Secondly, changes in the acceptance condition of the nondeterministic Büchi automata for the specifications induce transition changes in their deterministic parity automata, which compose the game.

In the full version we formalize this model, and define the *transition-repair problem* (TR problem, for short) similarly to the SR problem, with the goal being to find a cheap transition-repair that guarantees the existence of an NE. We prove the following results.

▶ **Theorem 8.** *The TR problem is NP-complete for the following cases:*

- *A constant number of players, for all objectives.*
- *Uniform costs with an arbitrary number of players, for all objectives.*
- *Uniform costs with a constant number of players, and co-Büchi and parity objectives.*

and can be solved in polynomial time for uniform costs with a constant number of players and Büchi and reachability objectives. The TR problem with uniform costs can be solved in polynomial time for Büchi and reachability objectives, and is NP complete

4.2 Controlled-players repair

The underlying assumption in game theory is that players are selfish and rational. In particular, they would follow a suggested strategy only if it is in their interest. In the *controlled-players repair* model, we assume that we can control some of the players and guarantee they would follow the strategy we assign them. The other players cooperate only if the profile is an NE. Controlling a player has a cost and our goal is to reach such a profile

with a minimal cost. This model is a type of Stackelberg model, where there is a *leader* player whose goal is to increase the social welfare. She moves first, selects a fraction α of the players, and assigns strategies to them. The rest of the players are selfish and choose strategies to maximize their revenue. Previous works in Algorithmic Game Theory study how the parameter α affects the social welfare in an NE. Clearly, when α is high, the social welfare increases.

Formally, given a game $\mathcal{G} = \langle \Omega, V, A, v_0, \delta, \{\alpha_i\}_{i\in\Omega}\rangle$ and a *control cost* function $cost : \Omega \to \mathbb{N}_\infty$, which maps each agent to the cost of controlling him, the *controlled-player repair* problem (the CR problem, for short), is to find a set of players of minimal cost such that if we are allowed to fully control these players, then the game has an NE. By *controlling* we mean that the players are not allowed to deviate from their strategies in the suggested profile.

Controlled-players repair arises in settings where an unstable system can be stabilized by restricting the environment, but this involves a cost. For example, controlling players is possible in settings where players accept an outside payment. As another example, taken from [20], the players are customers who can either pay a full price for using a system, and then their choices are unlimited, or they can pay a "bargain" price, and then their choices are limited, and hence their quality of service is not guaranteed. As a third example, consider a system that receives messages from the environment. We may want to require that messages arrive chronologically, otherwise our system is unstable. We can require this, but it involves a latency cost, and is effectively translated to asking the message dispatching thread to work in a non-optimal way, which is not the best strategy for the message dispatch server.

In the decision version of the problem, we are given a threshold p, and we need to determine if there exists a set $S \subseteq \Omega$ such that $cost(S) = \sum_{i\in S} cost(i) \leq p$ and controlling the players in S ensures the existence of an NE.

We start by studying the general case. In order to solve the CR problem, we observe that controlling Player i can be modeled by setting α_i to be the most permissive, thus for reachability, Büchi, and co-Büchi objectives, we set $\alpha_i = V$, and for parity objectives $\alpha_i(v)$ is the maximal even index. Indeed, if there is an NE profile P in $\mathcal{G}$ in which we control Player i, then P is also an NE when we set α_i as in the above (without controlling player i). Clearly, Player i has no incentive to deviate. Conversely, if there is an NE profile P after setting α_i to be the most permissive, then the same profile P is an NE in a game in which we control Player i and force it to play his strategy in P.

Theorem 9 below summarizes our results, and is proved in the full version.

▶ **Theorem 9.** *The CR problem is NP-complete for reachability, co-Büchi, and parity objectives, as well as for c-concurrent Büchi games, and is in P for general Büchi games, and for all objectives with a constant number of players.*

▶ Remark. In the future, we plan to investigate *scheduling repairs*, where a repair controls the set of players that proceed in a vertex, as well as *disabling repairs*, in which some actions of some players are disabled in some vertices.

References

1 R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

2 P. Bouyer, R. Brenguier, and N. Markey. Nash equilibria for reachability objectives in multi-player timed games. In *Proc. 21st CONCUR*, pages 192–206, 2010.

3 P. Bouyer, R. Brenguier, N. Markey, and M. Ummels. Nash equilibria in concurrent games with Büchi objectives. In *Proc. 31st FSTTCS 2011*, pages 375–386, 2011.

4 R. Brenguier. Nash Equilibria in Concurrent Games – Application to Timed Games. PhD thesis, École normale supérieure, 2012.

5 K. Chatterjee, L. de Alfaro, and T.A. Henzinger. Qualitative concurrent parity games. *ACM Trans. Comput. Log.*, 12(4):28, 2011.

6 K. Chatterjee, T. Henzinger, and B. Jobstmann. Environment assumptions for synthesis. In *Proc. 19th CONCUR*, LNCS 5201, pages 147–161. Springer, 2008.

7 K. Chatterjee, R. Majumdar, and M. Jurdzinski. On nash equilibria in stochastic games. In *Proc. 13th CSL*, LNCS 3210, pages 26–40. Springer, 2004.

8 L. de Alfaro and T.A. Henzinger. Concurrent ω-regular games. In *15th LICS*, pages 141–154, 2000.

9 D. Fisman, O. Kupferman, and Y. Lustig. Rational synthesis. In *Proc. 16th TACAS*, LNCS 6015, pages 190–204. Springer, 2010.

10 G. Gottlob, G. Greco, and F. Scarcello. Pure Nash equilibria: hard and easy games. In *Proc. 9th TARK*, pages 215–230, 2003.

11 B. Jobstmann, A. Griesmayer, and R. Bloem. Program repair as a game. In *Proc. 17th CAV*, LNCS 3576, pages 226–238, Springer 2005,

12 R.M. Karp. Reducibility among combinatorial problems. In *Complexity of Computer Computations*, pages 85–103, 1972.

13 A. Korilis, A. Lazar and A. Orda. Achieving Network Optima Using Stackelberg Routing Strategies. In *IEEE/ACM Trans. Netw.*, 5(1):161–173, 1997.

14 O. Kupferman, G. Perelli, and M.Y. Vardi. Synthesis with rational environments. In *Proc. 12th EUMAS*. Springer, 2014.

15 W. Li, L. Dworkin, and S. A. Seshia. Mining assumptions for synthesis. In *Proc. 9th MEMOCODE*, pages 43–50, 2011.

16 F. Laroussinie, N. Markey, and G. Oreiby. On the Expressiveness and Complexity of ATL. *LMCS*, 4(2), 2008.

17 D.A. Martin. Borel determinacy. *Annals of Mathematics*, 65:363–371, 1975.

18 J.F. Nash. Equilibrium points in n-person games. In *Proceedings of the National Academy of Sciences of the United States of America*, 1950.

19 A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Proc. 16th POPL*, pages 179–190, 1989.

20 D. Rosenberg, E. Solan, and N. Vieille. The maxmin value of stochastic games with imperfect monitoring. *Int. J. Game Theory*, 32(1):133–150, 2003.

21 T. Roughgarden. Stackelberg Scheduling Strategies. *SIAM J. Comput.*, 33(2):332–350, 2004.

22 M. Ummels. The complexity of nash equilibria in infinite multiplayer games. In *Proc. 11th FOSSACS*, pages 20–34, 2008.

23 J. von Neumann and O. Morgenstern. Theory of games and economic behavior.