Nash Equilibria in Symmetric Graph Games with Partial Observation

Patricia Bouyer[a], Nicolas Markey[a], Steen Vester[b,*]

[a] *LSV, CNRS & ENS Cachan, France*
[b] *Technical University of Denmark, Kgs. Lyngby, Denmark*

Abstract

We investigate a model for representing large multiplayer games, which satisfy strong symmetry properties. This model is made of multiple copies of an arena; each player plays in his own arena, and can partially observe what the other players do. Therefore, this game has partial information and symmetry constraints, which make the computation of Nash equilibria difficult. We show several undecidability results, and for bounded-memory strategies, we precisely characterize the complexity of computing pure Nash equilibria for qualitative objectives in this game model.

Keywords: Games on graphs, Network of systems, Symmetry, Nash equilibrium

1. Introduction

Multiplayer games. In the field of formal verification, games played on graphs extend the more classical Kripke structure with a way of modeling interactions between several components of a computerized system. Those types of games are intensively used as a tool to reason about and automatically synthesize (part of) reactive systems [1]. Consider a server granting access to a printer and connected to several clients. The clients may send requests to the server, and the server grants access to the printer depending on the requests

[☆] This work has benefited from support from projects FP7-ICT-601148 (Cassting) and ERC-StG-308087 (EQualIS).

[*] Corresponding author

Email addresses: bouyer@lsv.ens-cachan.fr (Patricia Bouyer), markey@lsv.ens-cachan.fr (Nicolas Markey), stve@dtu.dk (Steen Vester)

it receives. The server could have various strategies: for instance, never grant access to any client, or always immediately grant access upon request. However, it may also have constraints to satisfy which define its winning condition: for instance, that no two clients should access the printer at the same time, or that every request must eventually be granted. A strategy for the server is then a policy that it should apply in order to achieve these goals.

Until recently, more focus had been put on the study of purely antagonistic games (a.k.a. zero-sum games), which conveniently represent systems evolving in a hostile environment: the aim of one player is to prevent the other player from achieving his own objective.

Non-zero-sum games. Over the last ten years, computer scientists have started considering games with non-zero-sum objectives: they allow for conveniently modelling complex infrastructures where each individual system tries to fulfill its own objectives, while still being subject to uncontrollable actions of the surrounding systems. As an example, consider a wireless network in which several devices try to send data: each device can modulate its transmitting power, in order to maximize its bandwidth or reduce energy consumption as much as possible. In that setting, focusing only on optimal strategies for one single agent is too narrow. Game-theoreticians have defined and studied many other solution concepts for such settings, of which Nash equilibrium [2] is a prominent one. A Nash equilibrium is a strategy profile where no player can improve the outcome of the game by unilaterally changing his strategy.

Networks of identical devices. Our aim in this paper is to handle the special case where many of the interacting systems have identical abilities and objectives. This encompasses many situations involving computerized systems over a network. We propose a convenient way of modelling such situations, and develop algorithms for synthesizing a single strategy that, when followed by all the players, leads to a global Nash equilibrium. To be meaningful, this requires symmetry assumptions on the arena of the game. We also include *imperfect observation* of the players, which we believe is relevant in such a setting.

Our contributions. We propose a model for representing large interacting systems, which we call a *game network*. A game network is made of multiple copies of a single arena; each player plays on his own copy of the arena. As mentioned earlier, the players have imperfect information about the global state of the game. For instance, they may have a perfect view on some of their "neighbours", but may be blind to some other players. In *symmetric*

game networks, we additionally require that any two players are in similar situations: for every pair of players (A, B), we are able to map each player C to a corresponding player D with the informal meaning that 'player D is to B what player C is to A'. Of course, winning conditions and imperfect information should respect that symmetry. We present several examples illustrating the model, and argue why it is a relevant model. In these systems, we are interested in so-called symmetric pure Nash equilibria, which are special Nash equilibria where all players follow the same deterministic strategy.

We show several undecidability results, in particular that the parameterized synthesis problem (aiming to obtain one policy that forms a Nash equilibrium when applied to any number of participants) is undecidable. We then characterize the complexity of computing constrained pure symmetric Nash equilibria in symmetric game networks, when objectives are given as LTL formulas, and when restricting to memoryless and bounded-memory strategies. This problem with no memory bound is then proven undecidable.

Related work. Game theory has been a very active area since the 1940's, but its applications to computer science *via* graph games is quite recent. In that domain, until recently more focus had been put on zero-sum games [1]. Some recent works have considered multi-player non-zero-sum games, including the computation of constrained equilibria in turn-based and in concurrent games [3, 4, 5] or the development of temporal logics geared towards non-zero-sum objectives [6, 7, 8].

None of those works distinguish symmetry constraints in strategy profiles nor in game description. Still, symmetry has been studied in the context of normal-form games [9, 10]: in such a game, each player has the same set of actions, and the utility function of a player only depends on his own action and on the number of players who played each action (it is independent on 'who played what'). It will be further discussed in Appendix A.

Finally, let us mention that parameterized networks of identical systems are intensively studied in the context of model checking. Several techniques have been developed in this context for solving the parameterized model-checking problem ("*for a large number of copies of the system, does the property hold?*"): in some frameworks, one can prove the existence of a cutoff, that is the computation of a bound on the number of copies of the systems that is sufficient to prove the property for any number of participants, see for instance [11]; in other contexts, the model-checking problem can be reduced to analyzing the property of the network architecture, like in [12];

other techniques have been developed based on Vector Addition Systems with States [13], on algebraic techniques for quotienting the state space [14], on modular games [15], or on induction techniques for proving network invariants [16, 17]. Networks of *probabilistic* systems have been considered, on which qualitative (almost-sure) properties can be checked using game-based techniques [18]. However, none of these works solve the synthesis problem for (symmetric) strategies in networks of identical systems, which is the topic of the present paper.

2. Nash equilibria in Symmetric Games with Partial Observation

2.1. Definitions

Preliminary definitions. For every $k \in \mathbb{N} \cup \{\infty\}$, we write $[k]$ for the set $\{i \in \mathbb{N} \mid 0 \leq i < k\}$ and $[\infty] = \mathbb{N}$. Let $s = (p_i)_{i\in[n]}$ be a sequence, with $n \in \mathbb{N}\cup\{\infty\}$ being the length $|s|$ of s. Let $j \in \mathbb{N}$ such that $j-1 < n$. The jth element of s, denoted s_{j-1}, is the element p_{j-1} (so that a sequence $(p_i)_{i\in[n]}$ may be named p when no ambiguity arises). The jth prefix $s_{<j}$ of s is the finite sequence $(p_i)_{i\in[j]}$. If s is finite, we write $\mathsf{last}(s)$ for its last element $s_{|s|-1}$.

Given a mapping $f\colon A \to B$, $a \in A$, and $b \in B$, we write $f[a \mapsto b]$ for the mapping g such that $g(a) = b$ and $g(\alpha) = f(\alpha)$ for all $\alpha \in A \setminus \{a\}$. Given a mapping $f\colon A \to B$ and a mapping $g\colon B \to C$, the composition of g and f is the mapping $g \circ f\colon A \to C$ defined as $g \circ f(a) = g(f(a))$. Seeing a sequence $p = (p_i)_{i\in[n]}$ as a mapping with domain $[n]$, and given a mapping $f\colon [m] \to [n]$, we write $p \circ f$ for the sequence $(p_{f(j)})_{j\in[m]}$.

Concurrent games. Concurrent games played on graphs are used in the verification community as a tool to model, reason about and automatically synthesize interacting reactive systems. The model of concurrent games, originally given for two players in [19], is defined as follows:

Definition 1. *A* concurrent game *is a tuple* $\mathcal{G} = \langle \mathsf{States}, \mathsf{Agt}, \mathsf{Act}, \mathsf{Mov}, \mathsf{Tab}\rangle$ *where*

- States *is a finite set of states;*
- Agt *is a finite set of agents (also called players);*
- Act *is a finite set of actions;*

- $\mathit{Mov}\colon \mathit{States} \times \mathit{Agt} \to 2^{\mathit{Act}} \setminus \{\emptyset\}$ *is the set of actions available to a given player in a given state;*
- $\mathit{Tab}\colon \mathit{States} \times \mathit{Act}^{\mathit{Agt}} \to \mathit{States}$ *is a transition function that specifies the next state, given a state and an action of each player.*

The evolution of such a game is as follows: from a state s, each player A concurrently selects an available action $m_A \in \mathsf{Mov}(s, A)$. The successor state of s under the *move* $(m_A)_{A\in\mathsf{Agt}}$ is then looked up in Tab. A *path* in $\mathcal{G}$ from state s is a sequence $(s_i)_{i\geq 0}$ of states such that $s_0 = s$ and for all $i \geq 0$, there is a move $(m_A)_{A\in\mathsf{Agt}}$ such that $s_{i+1} \in \mathsf{Tab}(s_i, (m_A)_{A\in\mathsf{Agt}})$. Finite paths are called *histories*, while infinite paths are called *plays*. We write Path (resp. Hist, Play) for the set of paths (resp. histories, plays) in $\mathcal{G}$.

Let $A \in \mathsf{Agt}$. A *strategy* for A is a mapping $\sigma_A\colon \mathsf{Hist} \to \mathsf{Act}$ such that for every $\rho \in \mathsf{Hist}$, $\sigma_A(\rho) \in \mathsf{Mov}(\mathsf{last}(\rho), A)$. Given a set of players $C \subseteq \mathsf{Agt}$, a strategy for coalition C is a mapping σ assigning to each $A \in C$ a strategy for A (we will write σ_A instead of $\sigma(A)$ to alleviate notations). As a special case, a strategy for Agt is called a *strategy profile*.

A path ρ is *compatible* with a strategy σ of coalition C if, for every $i < |\rho|$, there exists a move $(m_A)_{A\in\mathsf{Agt}}$ such that $\mathsf{Tab}(\rho_{i-1}, (m_A)_{A\in\mathsf{Agt}}) = \rho_i$ and $m_A = \sigma_A(\rho_{<i})$ for all $A \in C$. The set of *outcomes* of σ from a state s, denoted $\mathsf{Out}(s, \sigma)$, is the set of plays from s that are compatible with σ.

Remark 2. *This work can be seen as extending the already vast literature of model checking networks of systems [16, 13, 17, 20, 11] towards synthesis. This also follows recent works about controller synthesis for parameterized discrete events systems [21, 22].*

*Games of infinite duration played on finite graphs are standard models in game theory, some classical examples are stochastic games [**?**] and mean-payoff games [**?**]. In particular, they are widely applied in the verification community as they are very natural for modelling (theoretically) infinite duration systems with finite state spaces [23, 19**?**]. An important advantage of this modelling choice compared to normal-form games with infinitely many actions or games played on infinite trees are that the finite graphs are finitely representable systems of infinite duration. As such, they can be used as input to algorithms for automatic verification and synthesis.*

Nash equilibria. Let $\mathcal{G}$ be a concurrent game. A *winning condition* for player A is a set Ω_A of plays of $\mathcal{G}$. We say that a play $\rho \in \Omega_A$ yields payoff 1

to A, and a play $\rho \notin \Omega_A$ yields payoff 0 to A. Winning conditions are usually infinite, but will most often be given symbolically as the set of plays satisfying a given property. For instance, given a formula ϕ of some logic (like LTL, see e.g. [25]) using States as atomic propositions, we write $\Omega(\phi)$ for the set of plays satisfying ϕ. A strategy σ of a coalition C is *winning* for A from a state s if $\mathsf{Out}(s, \sigma) \subseteq \Omega_A$. A strategy profile σ is a *pure Nash equilibrium* if, for every $A \in \mathsf{Agt}$ and every strategy σ'_A, if σ is losing for A, then so is $\sigma[A \mapsto \sigma'_A]$. In other terms, no player can individually improve his payoff.

Remark 3. *In this paper, we restrict to* pure *Nash equilibria, which correspond to deterministic programs for the players. Considering randomized strategies would clearly be of interest in presence of symmetry and would be a natural extension of this work. However some restrictions will be required since the existence of a randomized Nash equilibrium in concurrent games with reachability objectives is undecidable already with three players [26].*

Remark 4. *Even though Nash equilibria are some of the most well-studied solution concepts both in normal-form games and extensive-form games [24], there are situations where they are not sufficient to prescribe rational behavior, for instance in situations with non-credible threats. For a discussion, see e.g. [24]. To handle phenomena like this other kinds of equilibria, such as subgame-perfect equilibria [27], have been defined. Studying such equilibrium concepts in our setting would also be a natural continuation of our work. Though, despite the issues, Nash equilibria are still sufficient in many cases. As a practical example, most of the succesful programs in the AAAI Computer Poker Competition are based on Nash equilibrium computation [***?***].*

2.2. Symmetric Concurrent Games

As mentioned in the introduction, our aim is to propose a convenient way of modelling situations where all the interacting systems have identical abilities and objectives, and to develop algorithms for synthesizing symmetric strategy profiles in that setting. Intuitively, a symmetric strategy profile is a strategy profile where the same single strategy is played by all the players. The model we propose is made of a one-player arena, together with an observation relation. Intuitively, each player plays in his own copy of the one-player arena; the global system is the (synchronous) product of all the local copies, but each player observes the state of the global system only through an observation relation. This is in particular needed for representing large networks of systems, in which each player may only observe some of his neighbours.

Example 5. *Consider for instance a set of identical devices (e.g. cell phones) connected on a local area network. Each device can modulate its emitting power. In order to increase its bandwidth, a device tends to increase its emitting power; but besides consuming more energy, this also adds* noise *over the network, which decreases the other players' bandwidth and encourages them to in turn increase their power. We can model a device as an n-state arena where state i corresponds to some power p_i, with $p_0 = 0$ representing the device being off. A device would not know the exact state of the other devices, but would be able to evaluate the surrounding noise; this can be modelled using our observation relation, where all configurations with the same level of noise would be equivalent. Based on this information, the device can decide whether it should increase or decrease its emitting power, resulting in a good balance between bandwidth and energy consumption.*

Despite the global arena being described as a product of identical arenas, not all games described this way are symmetric: the observation relation should also be symmetric, and we have to impose extra conditions on that relation in order to capture an adequate notion of symmetry. Moreover, the observation relation relates global states of the system, and an explicit description of it will most often not be practical. We thus consider compact representations of this relation, as we now explain.

2.2.1. Formalization of the model

Game networks. We first define our notion of an n-player game network, which describes a complex game as a product of n identical one-player games.

Definition 6. *An* n-player game network $\mathcal{G}$ *is a tuple* $\langle G, (\equiv_i)_{i\in[n]}, (\Omega_i)_{i\in[n]}\rangle$ *such that*

- $G = \langle \mathsf{States}, \{A\}, \mathsf{Act}, \mathsf{Mov}, \mathsf{Tab}\rangle$ *is a one-player arena;*
- *for each* $i \in [n]$, $\equiv_i$ *is an equivalence relation on* States^n *extended in a natural way to sequences of states of* States^n*. Two* $\equiv_i$*-equivalent elements of* States^n *are indistinguishable to player* i*. This models* imperfect information *for player* i*. If* $\equiv_i$ *is the identity, then we say player* i *has perfect information;*
- *for each* $i \in [n]$, $\Omega_i \subseteq (\mathsf{States}^n)^\omega$ *is the objective of player* i*. We require that for all* $\rho, \rho' \in (\mathsf{States}^n)^\omega$*, if* $\rho \equiv_i \rho'$ *then* ρ *and* ρ' *are equivalently in* Ω_i*.*

We make the synchronous hypothesis and we define the semantics of this game as the "product game" $\mathcal{G}' = \langle \mathsf{States}', [n], \mathsf{Act}, \mathsf{Mov}', \mathsf{Tab}', (\Omega_i)_{i\in[n]}\rangle$ where $\mathsf{States}' = \mathsf{States}^n$, $\mathsf{Mov}'((s_0,\ldots,s_{n-1}),i) = \mathsf{Mov}(s_i,i)$, and the transition table is defined as

$$\mathsf{Tab}'((s_0,\ldots,s_{n-1}),(m_i)_{i\in[n]}) = (\mathsf{Tab}(s_0,m_0),\ldots,\mathsf{Tab}(s_{n-1},m_{n-1})).$$

Notice that we do not fix an initial state for the one-player arena as we want to be able to model cases where the players start in different states. For example, this makes us capable of modelling settings where not all players start playing at the same time.

Example 7. *Consider the cell-phone game again. It can be modelled as a game network where each player observes everything. That is, the equivalence relations $\equiv_i$ are the identity. A more realistic model for the system can be obtained by assuming that each player only gets precise information about his close neighbours, and less precise information, or no information at all, about the devices that are far away.*

We now give some further useful definitions. An element of States^n is called a *configuration* of $\mathcal{G}$. The equivalence relation $\equiv_i$ induces equivalence classes of configurations that player i cannot distinguish. We call these equivalence classes *information sets* and denote $\mathcal{I}_i$ the set of information sets for player i. Strategies should respect these information sets: a strategy σ_i for player i is *$\equiv_i$-realizable* whenever $\rho \equiv_i \rho'$ implies $\sigma_i(\rho) = \sigma_i(\rho')$. A strategy profile $\sigma = (\sigma_i)_{1\leq i\leq n}$ is said to be *realizable* whenever σ_i is $\equiv_i$-realizable for every $i \in [n]$.

Symmetric game networks. If we impose no restriction on the observation relation, n-player game networks do not fully capture symmetries in a system. Besides playing on similar arenas, we will add the extra requirement that all the players are in similar situations w.r.t. the other players.

Given a permutation π of $[n]$, for a configuration $t = (s_i)_{i\in[n]}$ we define $t(\pi) = (s_{\pi(i)})_{i\in[n]}$; similarly, for a path $\rho = (t_j)_{j\in\mathbb{N}}$, we define $\rho(\pi) = (t_j(\pi))_{j\in\mathbb{N}}$.

We now refine the previous definition for a game network to capture symmetries in the system.

Definition 8. *A game network $\mathcal{G} = \langle G, (\equiv_i)_{i\in[n]}, (\Omega_i)_{i\in[n]}\rangle$ is* symmetric *whenever for every two players $i, j \in [n]$, there is a permutation $\pi_{i,j}$ of $[n]$ such that $\pi_{i,j}(i) = j$ and satisfying the following conditions: for every $i, j, k \in [n]$,*

1. $\pi_{i,i}$ *is the identity, and* $\pi_{k,j} \circ \pi_{i,k} = \pi_{i,j}$*; hence* $\pi_{i,j}^{-1} = \pi_{j,i}$.
2. *the observation made by the players is compatible with the symmetry of the game: for all configurations* t *and* t', $t \equiv_i t'$ *if, and only if,* $t(\pi_{i,j}^{-1}) \equiv_j t'(\pi_{i,j}^{-1})$*;*
3. *objectives are compatible with the symmetry of the game: for every play* ρ, $\rho \in \Omega_i$ *if, and only if,* $\rho(\pi_{i,j}^{-1}) \in \Omega_j$.

In that case, $\pi = (\pi_{i,j})_{i,j\in[n]}$ *is called a* symmetric representation *of* $\mathcal{G}$.

The mappings $\pi_{i,j}$ define the symmetry of the game: $\pi_{i,j}(k) = l$ means that player l plays vis-à-vis player j the role that player k plays vis-à-vis player i. We give the intuition why we apply $\pi_{i,j}^{-1}$ in the definition above, and not $\pi_{i,j}$. Assume configuration $t = (s_0, \ldots, s_{n-1})$ is observed by player i. The corresponding configuration for player j is $t' = (s'_0, \ldots, s'_{n-1})$ where player-$\pi_{i,j}(k)$ state should be that of player k in t. That is, $s'_{\pi_{i,j}(k)} = s_k$, so that $t' = t(\pi_{i,j}^{-1})$. As mentioned in the introduction, we discuss such subtleties in the context of normal-form games in Appendix A.

These mappings define how symmetry must be used in strategies: let $\mathcal{G}$ be a symmetric n-player game network with symmetric representation π. We say that a strategy profile $\sigma = (\sigma_i)_{i\in[n]}$ is *symmetric* for the representation π if it is realizable (*i.e.*, each player only plays according to what he can observe) and if for all $i, j \in [n]$ and every history ρ, it holds $\sigma_i(\rho) = \sigma_j(\rho(\pi_{i,j}^{-1}))$.

Symmetric Nash equilibria are the special kinds of Nash equilibria which are also symmetric strategy profiles. This means that a symmetric Nash equilibrium is a Nash equilibrium where all players apply the same strategy.

Example 9. *Consider a card game tournament with six players, three on each table. Here each player has a left neighbour, a right neighbour, and three opponents at a different table. To model this, one could assume player* 0 *knows everything about himself, and has some information about his right neighbour (player* 1*) and his left neighbour (player* 2*). But he knows nothing about players* 3, 4 *and* 5.

Now, the role of player 2 *vis-à-vis player* 1 *is that of player* 1 *vis-à-vis player* 0 *(he is his right neighbour). Hence, we can define the symmetry as* $\pi_{0,1}(0) = 1$, $\pi_{0,1}(1) = 2$, $\pi_{0,1}(2) = 0$, *and* $\pi_{0,1}(3, 4, 5) = (3, 4, 5)$ *(any choice is fine here). As an example, the observation relation in this setting could be that player* 0 *has perfect knowledge of his set of cards, but only knows the number of cards of players* 1 *and* 2*, and has no information about the other*

three players. Notice that other observation relations would have been possible, for instance, giving more information about the right player.

Example 10. *Consider again the cell-phone example. In this model, the noise depends on the relative positions of the devices, and in that sense this game is not symmetric. The model of the cell-phone could include information about the relative positions of the other devices, by including several disjoints copies of the model, in which the neighbour devices have different influences over the noise. The initial state for each player would then depend on the topology of the network.*

Example 11. *Finally, let us mention that even though it is not fully symmetric, it is possible to model a client-server architecture in our framework. Let S be a model for the server and C be a model for the client. The game arena G will then be the disjoint union of S and C, and the equivalence $\equiv_i$ will look like: "if player i is in part S, then he has perfect information on all players, and if player i is in part C, then he sees his own states and the state of player 0", having in mind that player 0 will be the server and all other players will be clients. Such a game is not symmetric since the server observes more players than the clients do. In order to make the game fully symmetric, we would add extra players, trapped in a sink state, and observable by the clients. See Appendix B for a way to model this in our framework.*

Discussion on the model. First note that symmetric representations are not unique in general. We discuss here the impact of the representation on Nash equilibria.

We first define for each player a partitioning of the set of all the players, which will define what the players cannot distinguish. We then prove various indistinguishability properties, give examples and give conditions over the representations which ensure identical Nash equilibria.

For every player $i \in [n]$, we let $\cong_i$ be the following equivalence relations on the set of players $[n]$: $j \cong_i k$ if, and only if, for every configuration t, $t \equiv_i t(\pi_{j \leftrightarrow k})$, where $\pi_{j \leftrightarrow k}$ is the permutation of j and k. It means that player i cannot distinguish between the players j and k. We then define for every $i \in [n]$, the partition $\mathcal{P}_i$ of $[n]$ which is induced by $\cong_i$. We call $\mathcal{P} = (\mathcal{P}_i)_{i \in [n]}$ the canonical partitioning for $\mathcal{G}$.

Lemma 12. *For every symmetric representation π of $\mathcal{G}$, for every $i, j \in [n]$, for every $P_i \in \mathcal{P}_i$, it holds $\pi_{i,j}(P_i) \in \mathcal{P}_j$.*

Proof. Assume that it is not the case. There are two cases:

- First, suppose $\pi_{i,j}(P_i) \subsetneq P_j$ for some $P_j \in \mathcal{P}_j$. Take $k_j \in P_j \subseteq \pi_{i,j}(P_i)$ and $p_j \in \pi_{i,j}(P_i)$. For every configuration t, we have that $t \equiv_j t(\pi_{k_j \leftrightarrow p_j})$. We define $p_i = \pi_{i,j}^{-1}(p_j)$ (which is then in P_i) and $k_i = \pi_{i,j}^{-1}(k_j)$ (which is then not in P_i). As π is a symmetric representation of $\mathcal{G}$, we have that $t(\pi_{i,j}^{-1}) \equiv_i t(\pi_{i,j}^{-1} \circ \pi_{k_j \leftrightarrow p_j})$. We can now notice that $t(\pi_{i,j}^{-1} \circ \pi_{k_j \leftrightarrow p_j}) = t(\pi_{k_i \leftrightarrow p_i} \circ \pi_{i,j}^{-1})$, which then implies $t(\pi_{i,j}^{-1}) \equiv_i t(\pi_{k_i \leftrightarrow p_i} \circ \pi_{i,j}^{-1})$. For every t', we therefore get $t' \equiv_i t'(\pi_{k_i \leftrightarrow p_i})$. This contradicts the fact that $p_i \in P_i$ and $k_i \notin P_i$. This case is not possible.

- Second, suppose there exists two sets P_j and P'_j such that $P_j \neq P'_j \in \mathcal{P}_j$, and $\pi_{i,j}(P_i) \cap P_j \neq \emptyset$ and $\pi_{i,j}(P_i) \cap P'_j \neq \emptyset$. The reasoning is similar to above. □

Example 13. *For a configuration $t = (s_i)_{i \in [n]}$ and a subset P of players, we define t_P as the subsequence $(s_i)_{i \in P}$, and its Parikh image $\mathsf{Parikh}(t_P)$ as the function mapping each state s to its number of occurrences in t_P. Now, we define the observation relation $\mathsf{Parikh}(P)$ as follows:*

$$(t, t') \in \mathsf{Parikh}(P) \quad \textit{if, and only if,} \quad \mathsf{Parikh}(t_P) = \mathsf{Parikh}(t'_P).$$

Similarly, we define the observation relation $\mathsf{Id}(P)$ as

$$(t, t') \in \mathsf{Id}(P) \quad \textit{if, and only if,} \quad t[i] = t'[i] \textit{ for all } i \in P.$$

Using these relations and the fact that the intersection of two equivalence relations is an equivalence relation, we can define various observation relations, for instance the relation $\equiv_i$ defined by

$$\mathsf{Id}(\{i\}) \wedge \mathsf{Parikh}(\{i+1, i+2, i+3\}) \wedge \mathsf{Parikh}(\{i+3, i+4, i+5\})$$

where all indices are taken modulo n. With such an observation, player i has perfect information about his own state, and knows the Parikh images for players $i+1$, $i+2$ and $i+3$ and for players $i+3$, $i+4$ and $i+5$. One can check that the partition $\mathcal{P}_i$ is then $(\{i\}, \{i+1, i+2\}, \{i+3\}, \{i+4, i+5\})$ where player $i+3$ plays a special role as he appears in the two Parikh conditions.

There are two reasons why a symmetric game network may admit several symmetric representations. For instance, in a three-player game where each player only observes the Parikh image of the other two players, mappings π can either be defined as $\pi_{0,1}(1) = 2$ and $\pi_{0,1}(2) = 0$, or $\pi_{0,1}(1) = 0$ and $\pi_{0,1}(2) = 2$. Such distinctions are harmless in general, and those will generate the same symmetric behaviours. More precisely:

Lemma 14. *Let $\mathcal{G}$ be a symmetric n-player game network, and assume $\mathcal{P}$ is the canonical partitioning of $\mathcal{G}$. Take two symmetric representations π and $\widetilde{\pi}$ for $\mathcal{G}$. Assume that for every $i \in [n]$, for every piece $P \in \mathcal{P}_i$, $\pi_{i,j}(P) = \widetilde{\pi}_{i,j}(P)$. Then, a strategy profile σ is symmetric for π if, and only if, it is symmetric for $\widetilde{\pi}$.*

Proof. It is sufficient to show that for every configuration t, $t(\pi_{i,j}^{-1}) \equiv_j t(\widetilde{\pi}_{i,j}^{-1})$.

Let π be a permutation of $[n]$ that preserves partition $\mathcal{P}_i$ such that $\widetilde{\pi}_{i,j} = \pi_{i,j} \circ \pi$. Let t be a configuration. As π preserves $\mathcal{P}_i$, $t \equiv_i t(\pi^{-1})$. This implies, if we apply the symmetry condition for $\pi_{i,j}$: $t(\pi_{i,j}^{-1}) \equiv_j t(\pi^{-1} \circ \pi_{i,j}^{-1})$, that is, $t(\pi_{i,j}^{-1}) \equiv_j t(\widetilde{\pi}_{i,j}^{-1})$. Therefore the symmetry condition for the strategy profile does not depend on the choice of the symmetry mappings. □

Symmetric representations might however differ more 'dramatically'. Assume for instance that $n = 6$, and that $\equiv_i$ is defined for every $i \in [6]$ as '$\mathsf{Id}(\{i\}) \wedge \mathsf{Parikh}(\{i+1, i+2\}) \wedge \mathsf{Parikh}(\{i+3, i+4\})$' (taken modulo 6). Then the canonical partition $\mathcal{P}_i$ is equal to $(\{i\}, \{i+1, i+2\}, \{i+3, i+4\}, \{i+5\})$, and the mappings $\pi_{i,j}(i+k) = j+k \bmod 6$ properly define the symmetry. But there are other mappings that define the symmetry, for instance:

$$\pi'_{2i,2i+1}: \begin{array}{rl} 2i & \mapsto 2i+1 \\ 2i+1 & \mapsto 2i+4 \\ 2i+2 & \mapsto 2i+5 \\ 2i+3 & \mapsto 2i+2 \\ 2i+4 & \mapsto 2i+3 \\ 2i+5 & \mapsto 2i \end{array} \qquad \pi'_{2i+1,2i+2}: \begin{array}{rl} 2i & \mapsto 2i+1 \\ 2i+1 & \mapsto 2i+2 \\ 2i+2 & \mapsto 2i+5 \\ 2i+3 & \mapsto 2i \\ 2i+4 & \mapsto 2i+3 \\ 2i+5 & \mapsto 2i+4 \end{array}$$

The other mappings are obtained by composition. This also properly represents the symmetry, but generates different symmetric strategy profiles. Under additional technical conditions, we can prove that Nash equilibria coincide for two symmetric representations of a given symmetric game network. First we realize that a symmetric strategy profile is fully determined by an $\equiv_0$-realizable strategy for player 0.

Lemma 15. *Fix a symmetric representation π for $\mathcal{G}$. If σ_0 is an $\equiv_0$-realizable strategy for player 0, then the strategy profile σ defined by $\sigma_i(\rho) = \sigma_0(\rho(\pi_{i,0}^{-1}))$ defines a realizable and symmetric strategy profile.*

Proof. Symmetry is straightforward:

$$\sigma_j(\rho(\pi_{i,j}^{-1})) = \sigma_0(\rho(\pi_{j,0}^{-1} \circ \pi_{i,j}^{-1})) = \sigma_0(\rho(\pi_{i,0}^{-1})) = \sigma_i(\rho).$$

Assume that σ_i is not $\equiv_i$-realizable: this means that there are two runs $\rho \equiv_i \rho'$ such that $\sigma_i(\rho) \neq \sigma_i(\rho')$. By the symmetry of the game, it holds that $\rho(\pi_{i,0}^{-1}) \equiv_0 \rho'(\pi_{i,0}^{-1})$, which implies that $\sigma_0(\rho(\pi_{i,0}^{-1})) = \sigma_0(\rho'(\pi_{i,0}^{-1}))$. However this precisely means $\sigma_i(\rho) = \sigma_i(\rho')$. Hence strategy σ_i is $\equiv_i$-realizable. □

We can now show the following result.

Lemma 16. *Assume that a symmetric representation π of game network $\mathcal{G} = (G, (\equiv_i)_{i\in[n]}, (\Omega_i)_{i\in[n]})$ has to satisfy the following additional constraint: if there exist permutations $(\kappa_i)_{i\in[n]}$ of $[n]$ such that:*

(i) *$\kappa_i(i) = i$ for every i*

(ii) *for every two configurations t and t',*

$$t \equiv_i t' \;\Leftrightarrow\; t(\kappa_i \circ \pi_{j,i} \circ \kappa_j^{-1} \circ \pi_{i,j}) \equiv_i t'(\kappa_i \circ \pi_{j,i} \circ \kappa_j^{-1} \circ \pi_{i,j})$$

(iii) *for every run ρ, we have $\rho \in \Omega_i$ if, and only if, $\rho(\kappa_i \circ \pi_{j,i} \circ \kappa_j^{-1} \circ \pi_{i,j}) \in \Omega_i$*

then for every configuration t, $t \equiv_i t(\kappa_i \circ \pi_{j,i} \circ \kappa_j^{-1} \circ \pi_{i,j})$. Under that additional constraint, the choice of the representations does not affect Nash equilibria. More precisely: if π and $\widetilde{\pi}$ are two symmetric representations of game $\mathcal{G}$ that satisfy the above hypothesis, then a realizable strategy profile σ is a symmetric Nash equilibrium from t in $\mathcal{G}$ for representation π if, and only if, it is a symmetric Nash equilibrium from t in $\mathcal{G}$ for representation $\widetilde{\pi}$.

Proof. Let $(\pi_{i,j})_{i,j}$ and $(\widetilde{\pi}_{i,j})_{i,j}$ be two different representations (for the pieces of the canonical partitioning). Following Def. 8, those mappings are uniquely characterized by $(\pi_{0,i})_i$ and $(\widetilde{\pi}_{0,i})_i$. Assume κ_i is the permutation of $[n]$ such that $\widetilde{\pi}_{0,i} = \kappa_i \circ \pi_{0,i}$ (in particular w.l.o.g. κ_i swaps pieces of $\mathcal{P}_i$). We notice that $\widetilde{\pi}_{i,j} = \kappa_j \circ \pi_{i,j} \circ \kappa_i^{-1}$, again applying the properties listed in Def. 8.

It is not difficult to prove all the conditions for the κ_i's:

- using $\widetilde{\pi}_{0,i} = \kappa_i \circ \pi_{0,i}$, we get $\widetilde{\pi}_{0,i}(0) = \kappa_i \circ \pi_{0,i}(0)$, which entails $\kappa_i(i) = i$.

- it holds $t \equiv_i t'$ if, and only if, $t(\widetilde{\pi}_{i,j}^{-1}) \equiv_j t'(\widetilde{\pi}_{i,j}^{-1})$, which in turn is equivalent to $t(\widetilde{\pi}_{i,j}^{-1} \circ \pi_{j,i}^{-1}) \equiv_i t'(\widetilde{\pi}_{i,j}^{-1} \circ \pi_{j,i}^{-1})$. This directly entails the second property of the lemma.

- the third property is proven by applying the same argument: $\rho \in \Omega_i$ is equivalent to $\rho(\widetilde{\pi}_{i,j}^{-1} \circ \pi_{j,i}^{-1}) \in \Omega_i$, from which the property follows.

We therefore get that $t \equiv_i t(\kappa_i \circ \pi_{j,i} \circ \kappa_j^{-1} \circ \pi_{i,j})$ for every i and j, and in particular, taking $j = 0$, we get that $t \equiv_i t(\kappa_i)$ (since κ_0 is the identity).

Fix a strategy σ_0 for player 0 (which is $\equiv_0$-realizable). It defines two strategy profiles σ and $\widetilde{\sigma}$. For every i, we compute:

$$\widetilde{\sigma}_i(\rho) = \sigma_0(\rho(\widetilde{\pi}_{i,0}^{-1})) = \sigma_0(\rho(\kappa_i \circ \pi_{i,0}^{-1})) = \sigma_i(\rho(\kappa_i)) = \sigma_i(\rho) \quad \text{(since } \rho \equiv_i \rho(\kappa_i)\text{)}$$

In particular, $\widetilde{\sigma}$ and σ have the same outcome, yielding the same payoff to all players. Now if one of the players can improve his payoff, say player i can use strategy σ_i' to get better payoff than with his strategy σ_i, then he can also improve his payoff by playing the same strategy σ_i' in place of strategy $\widetilde{\sigma}_i$. □

In the sequel, we always assume that the symmetric representation is given. Note however that a symmetric representation can be computed in space polynomial in the number of players, by just enumerating the permutations and checking that they satisfy the constraints. As we show later, the problems we consider have higher complexity (when decidable), so that this assumption does not alter our results.

Discussion on the encoding of symmetric game networks. One motivation for the definition of this model is to represent large networks of identical systems in a rather compact way. To this aim, we need a succinct representation of game networks, in particular for the relations $\equiv_i$. Notice that representing those equivalence relations explicitly as $|\mathsf{States}|^n \times |\mathsf{States}|^n$ tables is not practicable. We therefore allow equivalence relations to be given *symbolically*, for instance as a polynomial-time program (or Turing machine) taking two integers $i \leq n$ and two states t and t' in States^n, and returning 1 if, and only if, $t \equiv_i t'$. Examples of such functions are $\mathsf{Parikh}(P)$ and $\mathsf{Id}(P)$ defined previously.

3. Problems considered and relationships between them

In this paper we are interested in the computation of Nash equilibria and symmetric Nash equilibria in symmetric game networks. More precisely, we are interested in the following three problems:

Problem 1 (Existence of a symmetric NE). *Given a symmetric game network $\mathcal{G}$, a symmetric representation π, and a configuration t, the* existence problem *asks whether there is a symmetric Nash equilibrium in $\mathcal{G}$ from t for the representation π.*

Remark 17. *There might not exist a pure Nash equilibrium in a symmetric game network. Figure 2 shows how one can simulate the* matching penny game*, which is known not to have pure Nash equilibria. We assume there are two players, and they both have perfect information. Player* 0 *starts from p_0 whereas player* 1 *starts from q_0. The objective is the same for player* 0 *and for player* 1 *and is written:*

$$\Big(p_0 \Rightarrow (\mathbf{F}\,((p_+ \wedge q_+) \vee (p_- \wedge q_-)))\Big) \wedge \Big(q_0 \Rightarrow (\mathbf{F}\,((p_+ \wedge q_-) \vee (p_- \wedge q_+)))\Big).$$

This reads as follows: "if you are in p_0, then you have to eventually visit both p_+ and q_+, or both p_- and q_-, and if you are in q_0, you have to eventually visit both p_+ and q_-, or both p_- and q_+". It is not hard to be convinced that it is symmetric, and that there is no pure Nash equilibrium from (p_0, q_0) in that game network.

Figure 1: Matching pennies as a symmetric game network

Problem 2 (Constrained existence of a symmetric NE). *Given a symmetric game network $\mathcal{G}$, a symmetric representation π, a configuration t, a set $L \subseteq [n]$ of losing players, and a set $W \subseteq [n]$ of winning players, the* constrained existence problem *asks whether there is a symmetric Nash equilibrium σ in $\mathcal{G}$ from t for the representation π, such that all players in L lose and all players in W win. If $W = [n]$, the problem is called the* positive existence problem.

Note that for the constrained problem, the input sets L and W do not need to cover the entire set of players. Thus, L and W constitute a partial specification of which players should win and lose.

3.1. From Nash equilibria to symmetric Nash equilibria

In this section we show that even though symmetric Nash equilibria are Nash equilibria satisfying special properties, they are in some sense at least as hard to find as Nash equilibria. This unfortunately means that we cannot in general hope to have an algorithm with better complexity for the symmetric problem by using properties of symmetry. Furthermore, it allows us to infer hardness results from the framework with standard Nash equilibria to the framework with symmetric Nash equilibria. The result is formalized in the following proposition.

Proposition 18. *From a symmetric game network $\mathcal{G}$, we can construct in polynomial time a symmetric game network $\mathcal{H}$ such that there exists a symmetric Nash equilibrium in $\mathcal{H}$ if, and only if, there exists a Nash equilibrium in $\mathcal{G}$. Furthermore the construction only changes the arena, but does not change the number of players nor the objectives or the resulting payoffs.*

Proof. Let $\mathcal{G} = (G, (\equiv_i)_{i\in[n]}, (\Omega_i)_{i\in[n]})$ be a symmetric game network and $(s_{0,0}, ..., s_{n-1,0})$ be a configuration as an input to the existence problem. We build a symmetric game network $\mathcal{H}$ which has a symmetric Nash equilibrium from some particular configuration if, and only if, $\mathcal{G}$ has a Nash equilibrium from $(s_{0,0}, ..., s_{n-1,0})$. We generate $\mathcal{H}$ as follows.

Let $\mathcal{H} = (H, (\sim_i)_{i\in[n]}, (\Theta_i)_{i\in[n]})$ be a symmetric game network with n players as $\mathcal{G}$. We design H as n disconnected copies of G. These copies will be denoted $H^0, ..., H^{n-1}$. We write $H^j(s)$ to denote the state of H^j corresponding to some state s of G. We introduce the mapping λ_1 such that $\lambda_1(H^j(s)) = s$ for all states s and all players j. Then we let the initial configuration in $\mathcal{H}$ be $(H^0(s_{0,0}), H^1(s_{1,0}), ..., H^{n-1}(s_{n-1,0}))$. In other words, the players in $\mathcal{H}$ start in their assigned initial state, but in different copies of G. We define $\sim_i$ such that for all i, for all states $s_0, ..., s_{n-1}, s'_0, ..., s'_{n-1}$ in G and for all $m_0, ..., m_{n-1}, j_0, ..., j_{n-1} \in [n]$, it holds:

$$(H^{m_0}(t_0), H^{m_1}(t_1), ..., H^{m_{n-1}}(t_{n-1})) \sim_i (H^{j_0}(v_0), H^{j_1}(v_1), ..., H^{j_{n-1}}(v_{n-1}))$$

if, and only if,

$$(t_0, ..., t_{n-1}) \equiv_i (v_0, ..., v_{n-1}) \wedge m_i = j_i.$$

Finally, for every player i and every infinite play $\rho = (s_0^1, ..., s_{n-1}^1)(s_0^2, ..., s_{n-1}^2)...$ in H, the objectives of the players in $\mathcal{H}$ are such that for all $j_0, ..., j_{n-1} \in [n]$,

$$(H^{j_0}(s_0^1), ..., H^{j_{n-1}}(s_{n-1}^1))(H^{j_0}(s_0^2), ..., H^{j_{n-1}}(s_{n-1}^2))... \in \Theta_i \quad \Leftrightarrow \quad \rho \in \Omega_i$$

We first show that $\mathcal{H}$ as defined here is indeed a symmetric game network. The proof is in two steps.

Lemma 19. *The relation $\sim_i$ is an equivalence relation for all i.*

Proof. Every state in H can be written as $H^j(s)$ for a state s in G and an index $j \in [n]$ in a unique way. Now $\sim_i$ is reflexive for all i since for all states $s_0, ..., s_{n-1}$ in G and all $j_0, ..., j_{n-1} \in [n]$, we have

$$\begin{aligned}(s_0, ..., s_{n-1}) \equiv_i (s_0, ..., s_{n-1}) \wedge j_i = j_i \Rightarrow \\ (H^{j_0}(s_0), ..., H^{j_{n-1}}(s_{n-1})) \sim_i (H^{j_0}(s_0), ..., H^{j_{n-1}}(s_{n-1}))\end{aligned}$$

It is symmetric for all i, since for all states $s_0, ..., s_{n-1}, s'_0, ..., s'_{n-1}$ in G and all $j_0, ..., j_{n-1}, m_0, ..., m_{n-1} \in [n]$, it holds

$$\begin{aligned}(H^{j_0}(s_0), ..., H^{j_{n-1}}(s_{n-1})) \sim_i (H^{m_0}(s'_0), ..., H^{m_{n-1}}(s'_{n-1})) \\ \Rightarrow (s_0, ..., s_{n-1}) \equiv_i (s'_0, ..., s'_{n-1}) \wedge j_i = m_i \\ \Rightarrow (s'_0, ..., s'_{n-1}) \equiv_i (s_0, ..., s_{n-1}) \wedge j_i = m_i \\ \Rightarrow (H^{m_0}(s'_0), ..., H^{m_{n-1}}(s'_{n-1})) \sim_i (H^{j_0}(s_0), ..., H^{j_{n-1}}(s_{n-1})).\end{aligned}$$

It is transitive for all i since for all states $s_0, ..., s_{n-1}, s'_0, ..., s'_{n-1}, s''_0, ..., s''_{n-1}$ in G, and $j_0, ..., j_{n-1}, k_0, ..., k_{n-1}, m_0, ..., m_{n-1} \in [n]$, we have

$$\begin{cases}(H^{j_0}(s_0), ..., H^{j_{n-1}}(s_{n-1})) \sim_i (H^{k_0}(s'_0), ..., H^{k_{n-1}}(s'_{n-1})) \wedge \\ (H^{k_0}(s'_0), ..., H^{k_{n-1}}(s'_{n-1})) \sim_i (H^{m_0}(s''_0), ..., H^{m_{n-1}}(s''_{n-1}))\end{cases}$$
$$\Rightarrow \begin{cases}(s_0, ..., s_{n-1}) \equiv_i (s'_0, ..., s'_{n-1}) \wedge (s'_0, ..., s'_{n-1}) \equiv_i (s''_0, ..., s''_{n-1}) \\ \wedge j_i = k_i \wedge k_i = m_i\end{cases}$$
$$\Rightarrow (s_0, ..., s_{n-1}) \equiv_i (s''_0, ..., s''_{n-1}) \wedge j_i = m_i$$
$$\Rightarrow (H^{j_0}(s_0), ..., H^{j_{n-1}}(s_{n-1})) \sim_i (H^{m_0}(s''_0), ..., H^{m_{n-1}}(s''_{n-1}))$$

This means that $\sim_i$ is an equivalence relation for all i. □

Lemma 20. *Let $\pi = (\pi_{i,j})_{i,j\in[n]}$ be a symmetric representation of $\mathcal{G}$. Then it is also a symmetric representation of $\mathcal{H}$.*

Proof. The first property of symmetric representations does not depend on the underlying game network, so π satisfies it. Secondly, for any two configurations $(H^{k_0}(s_0), ..., H^{k_{n-1}}(s_{n-1}))$ and $(H^{k'_0}(s'_0), ..., H^{k'_{n-1}}(s'_{n-1}))$ of $\mathcal{H}$ and every $i, j \in [n]$, we have

$$\begin{aligned}
&(H^{k_0}(s_0), ..., H^{k_{n-1}}(s_{n-1})) \sim_i (H^{k'_0}(s'_0), ..., H^{k'_{n-1}}(s'_{n-1}))\\
&\quad\Leftrightarrow (s_0, ..., s_{n-1}) \equiv_i (s'_0, ..., s'_{n-1}) \wedge s_i = s'_i\\
&\quad\Leftrightarrow (s_0, ..., s_{n-1})(\pi_{i,j}^{-1}) \equiv_j (s'_0, ..., s'_{n-1})(\pi_{i,j}^{-1}) \wedge s_{\pi_{i,j}^{-1}(j)} = s'_{\pi_{i,j}^{-1}(j)}\\
&\quad\Leftrightarrow (H^{k_0}(s_0), ..., H^{k_{n-1}}(s_{n-1}))(\pi_{i,j}^{-1}) \sim_j (H^{k'_0}(s'_0), ..., H^{k'_{n-1}}(s'_{n-1}))(\pi_{i,j}^{-1})
\end{aligned}$$

which means it satisfies the second requirement. For the third point, for every play $\rho = (H^{i_0}(s^0_0), ..., H^{i_{n-1}}(s^0_{n-1}))(H^{i_0}(s^1_0), ..., H^{i_{n-1}}(s^1_{n-1}))...$ and every $i, j \in [n]$, we have

$$\begin{aligned}
&(H^{i_0}(s^0_0), ..., H^{i_{n-1}}(s^0_{n-1}))(H^{i_0}(s^1_0), ..., H^{i_{n-1}}(s^1_{n-1}))... \in \Theta_i\\
&\quad\Leftrightarrow (s^0_0, ..., s^0_{n-1})(s^1_0, ..., s^1_{n-1})... \in \Omega_i\\
&\quad\Leftrightarrow (s^0_{\pi_{i,j}^{-1}(0)}, ..., s^0_{\pi_{i,j}^{-1}(n-1)})(s^1_{\pi_{i,j}^{-1}(0)}, ..., s^1_{\pi_{i,j}^{-1}(n-1)})... \in \Omega_j\\
&\quad\Leftrightarrow \begin{cases} (H^{i_{\pi_{i,j}^{-1}(0)}}(s^0_{\pi_{i,j}^{-1}(0)}), ..., H^{i_{\pi_{i,j}^{-1}(n-1)}}(s^0_{\pi_{i,j}^{-1}(n-1)}))\\ \qquad (H^{i_{\pi_{i,j}^{-1}(0)}}(s^1_{\pi_{i,j}^{-1}(0)}), ..., H^{i_{\pi_{i,j}^{-1}(n-1)}}(s^1_{\pi_{i,j}^{-1}(n-1)}))... \in \Theta_j \end{cases}
\end{aligned}$$

which is the final step showing that $(\pi_{i,j})_{i,j\in[n]}$ is also a symmetric representation for $\mathcal{H}$ which is therefore a symmetric game network. □

It remains to prove that there is a Nash equilibrium in $\mathcal{G}$ from configuration $(s_{0,0}, ..., s_{n-1,0})$ if, and only if, there is a symmetric Nash equilibrium in $\mathcal{H}$ from configuration $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$. First we introduce a bit of notation. The way we define the equivalence relations $(\sim_i)_{i\in[n]}$ the information sets of a player in $\mathcal{H}$ depends on which copy of G he is in as well as which information set in $\mathcal{G}$ the current configuration corresponds to. We denote the information sets for every player i, every copy j of G and every information set I of player i in $\mathcal{G}$ as follows

$$H_i^j(I) = \{(H^{m_0}(s_0), ..., H^{m_{n-1}}(s_{n-1})) \mid (s_0, ..., s_{n-1}) \in I \wedge m_i = j\}.$$

We now start with the first direction and assume there is a Nash equilibrium σ in $\mathcal{G}$ from $(s_{0,0}, ..., s_{n-1,0})$. Then we create the strategy profile σ' in $\mathcal{H}$ such that for all players i, j and all sequences of information sets $I_0, ..., I_{k-1}$ of player i in $\mathcal{G}$,

$$\sigma'_i(H_i^j(I_0)...H_i^j(I_{k-1})) = \sigma_j(\pi_{i,j}^{-1}(I_0)...\pi_{i,j}^{-1}(I_{k-1}))$$

which we will prove is a symmetric Nash equilibrium. Note again that in all legal sequences of information sets, a player will stay in the same copy of G and therefore this is a full definition of a strategy for each player.

Lemma 21. *If σ is a Nash equilibrium in $\mathcal{G}$ from $(s_{0,0}, ..., s_{n-1,0})$, then σ' is a symmetric Nash equilibrium in $\mathcal{H}$ from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$.*

Proof. To prove that it is symmetric, we need the following result, stating that for all i, j and all information sets I of player i in $\mathcal{G}$, it holds

$$\begin{aligned}
\pi_{i,j}^{-1}(H_i^j(I)) &= \{(H^{m_{\pi_{i,j}^{-1}(0)}}(s_{\pi_{i,j}^{-1}(0)}), ..., H^{m_{\pi_{i,j}^{-1}(n-1)}}(s_{\pi_{i,j}^{-1}(n-1)})) \mid \\
&\qquad\qquad (H^{m_0}(s_0), ..., H^{m_{n-1}}(s_{n-1})) \in H_i^j(I)\} \\
&= \{(H^{m_{\pi_{i,j}^{-1}(0)}}(s_{\pi_{i,j}^{-1}(0)}), ...H^{m_{\pi_{i,j}^{-1}(n-1)}}(s_{\pi_{i,j}^{-1}(n-1)})) \mid \\
&\qquad\qquad (s_0, ..., s_{n-1}) \in I \wedge m_i = j\} \\
&= \{(H^{m_{\pi_{i,j}^{-1}(0)}}(s_{\pi_{i,j}^{-1}(0)}), ...H^{m_{\pi_{i,j}^{-1}(n-1)}}(s_{\pi_{i,j}^{-1}(n-1)})) \mid \\
&\qquad\qquad (s_{\pi_{i,j}^{-1}(0)}, ..., s_{\pi_{i,j}^{-1}(n-1)}) \in \pi_{i,j}^{-1}(I) \wedge m_{\pi_{i,j}^{-1}(j)} = j\} \\
&= H_j^j(\pi_{i,j}^{-1}(I))
\end{aligned}$$

The requirement for σ' to be symmetric can now be reformulated as follows. For all players i, j and all information sets $I_0, ...I_{k-1}$ of player i in $\mathcal{G}$ we have

$$\begin{aligned}
\sigma'_i(H_i^j(I_0)...H_i^j(I_{k-1})) &= \sigma'_j(\pi_{i,j}^{-1}(H_i^j(I_0))...\pi_{i,j}^{-1}(H_i^j(I_{k-1}))) \\
&\Leftrightarrow \sigma_j(\pi_{i,j}^{-1}(I_0)...\pi_{i,j}^{-1}(I_{k-1})) = \sigma'_j(H_j^j(\pi_{i,j}^{-1}(I_0))...H_j^j(\pi_{i,j}^{-1}(I_{k-1}))) \\
&\Leftrightarrow \sigma_j(\pi_{i,j}^{-1}(I_0)...\pi_{i,j}^{-1}(I_{k-1})) = \sigma_j(\pi_{j,j}^{-1}(\pi_{i,j}^{-1}(I_0))...\pi_{j,j}^{-1}(\pi_{i,j}^{-1}(I_{k-1})))
\end{aligned}$$

Since $\pi_{j,j} = \pi_{j,j}^{-1}$ and $\pi_{j,j}$ is the identity, it follows that the bottom equality is true and therefore σ' is symmetric for representation $(\pi_{i,j})_{i,j\in[n]}$.

To see that σ' is also a Nash equilibrium from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$ consider the deviation of a player p from σ'_p to $\sigma'_{p,\text{dev}}$. We then look at a

corresponding deviation of p from σ_p to $\sigma_{p,\text{dev}}$ in $\mathcal{G}$ where for all sequences of information set tuples

$$\sigma_{p,\text{dev}}(I_0, ..., I_{k-1}) = \sigma'_{p,\text{dev}}(H^p_p(I_0), ..., H^p_p(I_{k-1})).$$

We consider the outcomes of the two profiles in the two games, denoted $\rho_{\mathcal{G}}$ and $\rho_{\mathcal{H}}$ respectively. We wish to show that $\rho_{\mathcal{G}} = \lambda_1(\rho_{\mathcal{H}})$ by induction. For the base case we have

$$\rho_{\mathcal{G},=0} = (s_{0,0}, ..., s_{n-1,0}) = \lambda_1(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0})) = \lambda_1(\rho_{\mathcal{H},=0}).$$

As induction hypothesis suppose it holds for prefixes of outcomes with length at most v. Further, let $\rho_{\mathcal{G},\leq v+1} = (s^0_0, ..., s^0_{n-1})...(s^{v+1}_0, ..., s^{v+1}_{n+1})$ and let I^j_i be the information set for player i in $\mathcal{G}$ which contains s^j_i. We will need that for a move m of all the players we have for all states $s_0, ..., s_{n-1}$ in $\mathcal{G}$ that

$$\text{Tab}((s_0, ..., s_{n-1}), m) = \lambda_1(\text{Tab}((H^0(s_0), ..., H^{n-1}(s_{n-1})), m)).$$

Then we get

$$\begin{aligned}
\rho_{\mathcal{G},\leq v+1} &= \rho_{\mathcal{G},\leq v} \cdot \text{Tab}(\rho_{\mathcal{G},=v}, \sigma[\sigma_p \mapsto \sigma_{p,\text{dev}}](I(\rho_{\mathcal{G},\leq v}))) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \text{Tab}((s^v_0, ..., s^v_{n-1}), \sigma[\sigma_p \mapsto \sigma_{p,\text{dev}}](I(\rho_{\mathcal{G},\leq v}))) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \lambda_1(\text{Tab}((H^0(s^v_0), ..., H^{n-1}(s^v_{n-1})), \\
&\qquad\qquad \sigma[\sigma_p \mapsto \sigma_{p,\text{dev}}](I(\rho_{\mathcal{G},\leq v})))) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \lambda_1(\text{Tab}(\rho_{\mathcal{H},=v}, \sigma[\sigma_p \mapsto \sigma_{p,\text{dev}}](I(\rho_{\mathcal{G},\leq v}))))
\end{aligned}$$

Since for all players i and all information sets $I_0, ..., I_{k-1}$ of i in $\mathcal{G}$ it holds that $\sigma[\sigma_p \mapsto \sigma_{p,\text{dev}}]_i(I_0, ..., I_{k-1}) = \sigma'[\sigma'_p \mapsto \sigma'_{p,\text{dev}}]_i(H^i_i(I_0), ..., H^i_i(I_{k-1}))$ we get

$$\begin{aligned}
\rho_{\mathcal{G},\leq v+1} &= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \lambda_1(\text{Tab}(\rho_{\mathcal{H},=v}, \sigma[\sigma_p \mapsto \sigma_{p,\text{dev}}](I(\rho_{\mathcal{G},\leq v})))) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \lambda_1(\text{Tab}(\rho_{\mathcal{H},=v}, \sigma'[\sigma'_p \mapsto \sigma'_{p,\text{dev}}]((H^0_0(I^0_0), ..., \\
&\qquad\qquad H^{n-1}_{n-1}(I^0_{n-1}))...(H^0_0(I^v_0), ..., H^{n-1}_{n-1}(I^v_{n-1}))))) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \lambda_1(\text{Tab}(\rho_{\mathcal{H},=v}, \sigma'[\sigma'_p \mapsto \sigma'_{p,\text{dev}}](\rho_{\mathcal{H},\leq v}))) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v}) \cdot \lambda_1(\rho_{H,=v+1}) \\
&= \lambda_1(\rho_{\mathcal{H},\leq v+1})
\end{aligned}$$

This means that if a player p can deviate from σ' in $\mathcal{H}$ to obtain outcome $\rho_{\mathcal{H}}$, then he can deviate from σ in $\mathcal{G}$ to obtain an outcome $\rho_{\mathcal{G}}$ with $\lambda_1(\rho_{\mathcal{H}}) = \rho_{\mathcal{G}}$.

Given the way we have defined objectives in $\mathcal{H}$, every player will get the same payoff from a play ρ in $\mathcal{H}$ as in $\lambda_1(\rho)$ in $\mathcal{G}$ for every play ρ. Since σ is a Nash equilibrium in $\mathcal{G}$ from $(s_{0,0}, ..., s_{n-1,0})$ where no player can deviate to improve his payoff, then no player can deviate to improve his payoff from σ' in $\mathcal{H}$ from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$ because otherwise that player would be able to deviate from σ in $\mathcal{G}$ to improve his payoff. Thus, σ' is a symmetric Nash equilibrium from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$. □

For the other direction we assume there is a symmetric Nash equilibrium σ' in $\mathcal{H}$ from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$. We now define σ in $\mathcal{G}$ for all information sets $I_0, ..., I_{k-1}$ of player i by letting

$$\sigma_i(I_0, ..., I_{k-1}) = \sigma'_i(H^i_i(I_0), ..., H^i_i(I_{k-1})).$$

Lemma 22. *If σ' is a symmetric Nash equilibrium in $\mathcal{H}$ from global state $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$, then σ is a Nash equilibrium in $\mathcal{G}$ from $(s_{0,0}, ..., s_{n-1,0})$.*

Proof. Contrary to the previous case we consider a deviation from σ in $\mathcal{G}$ by player p from σ_p to $\sigma_{p,\text{dev}}$ and consider a corresponding deviation in $\mathcal{H}$ from σ' defined by

$$\sigma'_{p,\text{dev}}(H^i_i(I_0), ..., H^i_i(I_{k-1})) = \sigma_{p,\text{dev}}(I_0, ..., I_{k-1})$$

for all information sets $I_0, ..., I_{k-1}$ of player p in $\mathcal{G}$. Let the outcomes of the profiles with the deviations be $\rho_{\mathcal{G}}$ and $\rho_{\mathcal{H}}$. As in the previous case we can show that $\lambda_1(\rho_{\mathcal{H}}) = \rho_{\mathcal{G}}$ which means that when a player p deviates in $\mathcal{G}$ from σ he can do a deviation in $\mathcal{H}$ from σ' which gives him the same payoff. Since σ' is a Nash equilibrium from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$ no player can deviate to improve his payoff from σ'. This means that no player can deviate to improve his payoff from σ in $\mathcal{G}$ from $(s_{0,0}, ..., s_{n-1,0})$ and therefore it is a Nash equilibrium from this configuration. □

This concludes the proof that there is a symmetric Nash equilibrium from $(H^0(s_{0,0}), ..., H^{n-1}(s_{n-1,0}))$ in $\mathcal{H}$ if, and only if, there is a Nash equilibrium from $(s_{0,0}, ..., s_{n-1,0})$ in $\mathcal{G}$.

There are n times as many states in the arena H as in the arena G. In addition, there are n times as many equivalence classes, which implies that the size of $\mathcal{H}$ is polynomial in the size of $\mathcal{G}$.

3.2. From positive existence to existence

Before turning to our decidability and undecidability results, we begin by showing that positive existence of Nash equilibria is not harder than existence. This is quite natural as all players have to win and there is no need to look for improvements: positive existence is equivalent to finding a path along which all objectives are fulfilled.

Proposition 23. *Deciding the symmetric existence problem in symmetric game networks is at least as hard as deciding the positive symmetric existence problem. The reduction doubles the number of players and uses* LTL *objectives, but does not change the nature of the strategies (memoryless, bounded-memory, or general).*

This result is a consequence of the following lemma, which we prove below.

Lemma 24. *Let $\mathcal{G}$ be an n-player symmetric game network and t_0 be an initial configuration in $\mathcal{G}$. We can construct in polynomial time a $(2n)$-player symmetric game network $\mathcal{G}'$ and a configuration t'_0 in $\mathcal{G}'$ such that there is a Nash equilibrium from t_0 along which all players win in $\mathcal{G}$ if, and only if, there is a Nash equilibrium from t'_0 in $\mathcal{G}'$. Moreover, this equivalence also holds for memoryless and bounded-memory equilibria.*

Proof. Let $\mathcal{G} = (G, (\equiv_i)_{i\in[n]}, (\Omega_i)_{i\in[n]})$ be a symmetric game network with n players. Assume the symmetric representation $(\pi_{i,j})_{i,j\in[n]}$. The game $\mathcal{G}'$ will consist of playing a matching pennies game between new players before entering the arena G. The new arena for the game is depicted in Figure 4. While one player will play in the left part (containing G), a new player will play in the right part. The former player will aim at matching the pennies or reaching his objective in G, while the second player will try not to match the pennies. If there is a Nash equilibrium in the resulting game, both players must win, since otherwise they can change their strategy and improve their payoff. In that equilibrium, it must be the case that the former player satisfies his objective in G. This is formalised below.

We define the new game network $\mathcal{G}'$ with $2n$ players as $\mathcal{G}' = \langle G', (\equiv'_i)_{i\in[2n]}, (\Omega'_i)_{i\in[2n]}\rangle$ defined below:

- $G' = (\mathsf{States}', \{A\}, \mathsf{Act}', \mathsf{Mov}', \mathsf{Tab}')$ where:

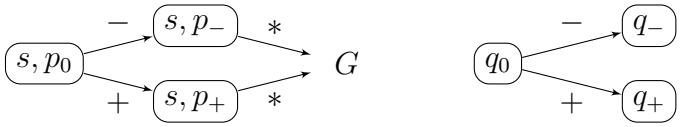

Figure 2: The player arena G'

- $\mathsf{States}' = \mathsf{States} \cup (\mathsf{States} \times \{p_0, p_+, p_-\}) \cup \{q_0, q_+, q_-\}$. The last three states constitute the *isolated part* of the arena, while the other states are in the *main part*;
- $\mathsf{Act}' = \mathsf{Act} \cup \{+, -, *\}$;
- for every $s \in \mathsf{States}$, $\mathsf{Mov}'(s) = \mathsf{Mov}(s)$ $\quad \mathsf{Mov}'(s, p_+) = \{*\}$
  $\mathsf{Mov}'(s, p_0) = \{+, -\}$ $\quad \mathsf{Mov}'(s, p_-) = \{*\}$
  $\mathsf{Mov}'(q_0) = \{+, -\}$ and $\mathsf{Mov}'(q_+) = \mathsf{Mov}'(q_-) = \emptyset$;
- for every $s \in \mathsf{States}$,
  $\mathsf{Tab}'(s, a) = \mathsf{Tab}(s, a)$ for every $a \in \mathsf{Mov}'(s)$
  $\mathsf{Tab}'((s, p_0), +) = (s, p_+)$
  $\mathsf{Tab}'((s, p_0), -) = (s, p_-)$
  $\mathsf{Tab}'((s, p_+), *) = \mathsf{Tab}'((s, p_-), *) = s$
  Finally, $\mathsf{Tab}'(q_0, +) = q_+$ and $\mathsf{Tab}'(q_0, -) = q_-$.

- For every configuration $t = (s_0, \ldots, s_{2n-1})$, we define

$$\mathit{main}(t) = \{j \in [2n] \mid s_j \text{ is in the main part of the arena}\}.$$

  We extend *main* to runs in a straightforward way. For every $i \in [2n]$, we define $\eta(i) = n + i \mod 2n$. Then, for every $i \in [2n]$, $t \equiv'_i t'$ if and only if the following conditions are satisfied:

  - $\mathit{main}(t) = \mathit{main}(t')$;
  - if the above conditions hold, there is a bijection $\nu \colon \mathit{main}(t) \to [n]$ such that $\nu(j) \in \{j, \eta(j)\}$ for every $j \in \mathit{main}(t)$ such that $t(\nu^{-1}) \equiv_i t'(\nu^{-1})$. Note that this only holds if $|\mathit{main}(t)| = |\mathit{main}(t')| = n$ and either $i \in \mathit{main}(t)$ or $\eta(i) \in \mathit{main}(t)$.

- Let $i \in [2n]$ and Γ_i^{isol} be the set of runs ϱ such that:
  - player i plays in the isolated part (that is, $i \notin \mathit{main}(\varrho)$)
  - player $\eta(i)$ plays in the main part (that is, $\eta(i) \in \mathit{main}(\varrho)$)

– players i and $\eta(i)$ realise their matching penny (that is, they visit states (s, p_+) and q_+, or states (s, p_-) and q_-, along ϱ)

Let $\Gamma_i^{\text{main}} = \Gamma_{\eta(i)}^{\text{isol}}$ be the counterpart in the main component.

Then, a run ϱ of $\mathcal{G}'$ will belong to $\widetilde{\Omega}_i$ whenever there exists a bijection $\nu\colon \mathit{main}(\varrho) \to [n]$ such that $\nu(j) \in \{j, \eta(j)\}$ for every $j \in \mathit{main}(\varrho)$, and $\varrho[\nu]$ belongs to Ω_i, where $\varrho[\nu]$ is the run obtained after having projected ϱ on the players $\nu(\mathit{main}(\varrho))$ and having removed the first two states of ϱ.

Finally, for every $i \in [2n]$, we define the objective Ω'_i for player i as $\widetilde{\Omega}_i \cup \overline{\Gamma_i^{\text{main}}} \cup \Gamma_i^{\text{isol}}$

We now define for every $i, j, k \in [n]$, the permutations $\pi'_{i,j}$ as follows:

- $\pi'_{i,j}(k) = \pi_{i,j}(k)$ and $\pi'_{i,j}(\eta(k)) = \eta(\pi_{i,j}(k))$,
- $\pi'_{i,\eta(j)}(k) = \eta(\pi_{i,j}(k))$ and $\pi'_{i,\eta(j)}(\eta(k)) = \pi_{i,j}(k)$,
- $\pi'_{\eta(i),j}(k) = \eta(\pi_{i,j}(k))$ and $\pi'_{\eta(i),j}(\eta(k)) = \pi_{i,j}(k)$
- $\pi'_{\eta(i),\eta(j)}(k) = \eta(\pi_{i,j}(k))$ and $\pi'_{\eta(i),\eta(j)}(\eta(k)) = \eta(\pi_{i,j}(k))$

Since $(\pi_{i,j})_{i,j\in[2n]}$ is a symmetric representation for $\mathcal{G}$, so is $(\pi'_{i,j})_{i,j\in[2n]}$ for $\mathcal{G}'$. We should also note that if $\mathcal{G}$ has a compact representation, then so has $\mathcal{G}'$. Furthermore, if objectives $(\Omega_i)_{i\in[n]}$ are given by LTL formulas, then objectives $(\Omega'_i)_{i\in[2n]}$ can also be given by LTL formulas of the same size. The result now follows from Lemma 25 below. □

Lemma 25. *There is a symmetric Nash equilibrium in $\mathcal{G}$ from configuration $t_0 = (s_0^0, \ldots, s_0^{n-1})$ where every player wins if, and only if, there is a symmetric Nash equilibrium in $\mathcal{G}'$ from configuration $t'_0 = ((s_0^0, p_0), \ldots, (s_0^{n-1}, p_0), q_0, \ldots, q_0)$.*

Proof. Assume there is a Nash equilibrium σ in $\mathcal{G}$ from t_0 where every player wins. We define the strategy profile σ' where players $i \in [n]$ play action $+$, then $*$, and finally play in G following their strategy in σ, while players $n + i$ for $i \in [n]$ play $-$. Note that if σ is symmetric, then this new profile is also symmetric. Under this new strategy profile, all the players achieve their objectives; this is therefore a Nash equilibrium.

Conversely, assume σ' is a Nash equilibrium in $\mathcal{G}'$. Each player $n + i$ can easily ensure $\Gamma_{n+i}^{\text{isol}}$ by swapping its action-choice (between $+$ and $-$).

Player $n+i$ must therefore be winning in a Nash equilibrium. Similarly, if player i is not winning, he can swap his first action and make the outcome belong to $\overline{\Gamma_i^{\text{main}}}$; this means that player i must be winning along the outcome of σ'. But since player $n+i$ is winning, the outcome of σ' does not belong to $\overline{\Gamma_i^{\text{main}}} \cup \Gamma_i^{\text{isol}}$, hence it must belong to $\widetilde{\Omega}_i$. The strategy profile σ is then just the part of σ' after the matching-penny part, and restricted to the first n players. The outcome of σ from t_0 in $\mathcal{G}$ then fulfills all the objectives of the players in $[n]$, hence it is a positive Nash equilibrium. Finally, we note that if σ' is symmetric, then so is σ. □

4. Existence in Symmetric Game Networks

Recent works have considered the computation of Nash equilibria in standard concurrent or turn-based games. In particular, the abstraction of suspect games described in [28] has allowed the development of efficient algorithms for computing Nash equilibria in concurrent games, for various classes of objectives. However those algorithms cannot be applied to our framework for the following reasons:

- each player has only *partial information* on the state-space of the game;
- the *symmetry requirement* induces non-local constraints in the concurrent game resulting from the product of the one-player arenas.

Notice that even in the case of symmetric games with perfect information, an approach using *Strategy Logic* [29], which can express Nash equilibria and impose several players to play the same strategy, would not work out-of-the-box, as in our setting strategies are equal *up to a permutation of the states*.

We now list the results we have obtained about computing Nash equilibria in symmetric game networks. We begin with undecidability results, for the following cases:

- non-regular objectives (for two players, perfect observation and recall);
- partial observation (for three players, LTL objectives, perfect recall).
- parametrized number of players (LTL objectives, incomplete observation, memoryless strategies);

We prove decidability when the number of players is given in the input and there is a restriction to bounded memory strategies.

4.1. Undecidability with non-regular objectives

Our games allow for arbitrary Boolean objectives, defined for each player as a set of winning plays. We prove that it is too general to get decidability of our problems even with perfect information.

Theorem 26. *The existence of a symmetric Nash equilibrium for non-regular objectives in two-player symmetric game networks is undecidable (even with perfect information).*

Proof. We do a reduction of the halting problem for a deterministic two-counter machine, which is well-known to be undecidable. A two-counter machine M is a 3-tuple $M = \langle Q, \Delta, q_F \rangle$ where

- Q is a finite set of control states, and $q_F \in Q$ is a halting state;
- $\Delta \colon Q \setminus \{q_F\} \to \{\text{inc}\} \times \{c, d\} \times Q \cup \{\text{dec}\} \times \{c, d\} \times Q^2$ is an instruction function which assigns an instruction to each state.

A configuration of M is a 3-tuple in $Q \times \mathbb{N} \times \mathbb{N}$. A run of M is a sequence of configurations $\rho = (q_0, c_0, d_0)(q_1, c_1, d_1)...$ where (q_0, c_0, d_0) is the initial configuration (usually assuming $c_0 = d_0 = 0$), and for two consecutive configurations we are in one of the following situations:

- $\Delta(q_i) = (\text{inc}, c, q_{i+1})$, $c_{i+1} = c_i + 1$ and $d_{i+1} = d_i$;
- $\Delta(q_i) = (\text{inc}, d, q_{i+1})$, $d_{i+1} = d_i + 1$ and $c_{i+1} = c_i$;
- $\Delta(q_i) = (\text{dec}, c, q_{i+1}, q)$ for some q, $c_{i+1} = c_i - 1 \geq 0$ and $d_{i+1} = d_i$;
- $\Delta(q_i) = (\text{dec}, d, q_{i+1}, q)$ for some q, $d_{i+1} = d_i - 1 \geq 0$ and $c_{i+1} = c_i$;
- $\Delta(q_i) = (\text{dec}, c, q, q_{i+1})$ for some q, $c_{i+1} = c_i = 0$ and $d_{i+1} = d_i$;
- $\Delta(q_i) = (\text{dec}, d, q, q_{i+1})$ for some q, $d_{i+1} = d_i = 0$ and $c_{i+1} = c_i$.

The run is infinite if there is no i so $q_i = q_F$ and otherwise it is finite with q_F being the halting state in the final configuration of ρ. The problem of deciding if the run of a two-counter machine has a halting run from a configuration (q_0, c_0, c_d) is undecidable and we wish to reduce an instance of this problem to the existence problem in symmetric game networks.

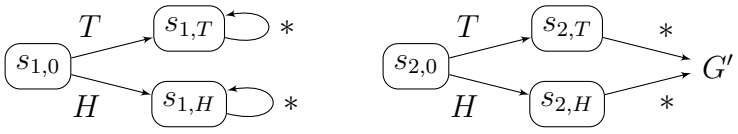

Figure 3: Illustration of G

(a) The incrementation module

(b) The decrementation module

Figure 4: Constructions of the incrementation and decrementation modules

Let M be a deterministic two-counter machine and let (q_0, c_0, d_0) be an initial configuration. From this we create a symmetric game network with two players $\mathcal{G} = \langle G, (\equiv_i)_{i\in[2]}, (\Omega_i)_{i\in[2]}\rangle$ where $(s_1, s_2) \equiv_i (s'_1, s'_2)$ if, and only if, $s_1 = s'_1$ and $s_2 = s'_2$ for $i = 1, 2$. The arena G consists of two disconnected parts. It is shown in Fig. 5, but without G'.

The idea is that player 1 starts in $s_{1,0}$ and player 2 starts in $s_{2,0}$. They first play a matching pennies game, and then player 2 plays in G', this simulating the counter machine M. We design the objectives so that player 2 wins if he acts according to the rules of the counter machine and reaches a halting state. If he does not reach a halting state, he wins if he chose an action different from that of player 1 in the initial matching pennies game; otherwise player 1 wins. This way, if there is a legal, halting run of the counter machine, then there is a Nash equilibrium where player 2 wins and player 1 loses. If there is no legal halting run then the game is essentially reduced to a matching pennies game which has no Nash equilibrium.

Formally, we do this by letting G' consist of the control states of M with the state connected to $s_{2,H}$ and $s_{2,T}$ being q_0. Then for all i, j there is an action C^+ taking the play from q_i through an intermediate state C^+_{ij} to q_j if $\Delta(q_i) = (inc, C, q_j)$ for some counter $C \in \{c, d\}$ as illustrated in Fig. 6a.

In addition, for all i, j, k there is an action C^- and an action C^0, respectively taking the play from q_i through intermediate states C^-_{ij} to q_j and C^0_{ik} to q_k if $\Delta(q_i) = (dec, C, q_j, q_k)$ for some counter $C \in \{c, d\}$, as shown in Fig. 6b.

Additionally, we add a self-loop to the halting state q_F. For a finite path ρ we now define

$$C_\rho = \big|\{k \mid \rho_k = C_{ij}^+ \text{ for some } i,j\}\big| - \big|\{k \mid \rho_k = C_{ij}^- \text{ for some } i,j\}\big|$$

When given an initial value c_0 and d_0 of the counters we then define the objectives such that player 2 loses in all plays ρ that contains a prefix $\rho_{\leq k}$ such that state $\rho_k = C_{ij}^-$ and $C_0 - C_{\rho_{\leq k}} < 0$ for some C, i and j to make sure player 2 plays according to the rules of the counter machine and does not subtract from a counter with value zero. In addition, he loses in all plays ρ that contains a prefix $\rho_{\leq k}$ such that $\rho_k = C_{ij}^0$ and $C_0 - C_{\rho_{\leq k}} \neq 0$ to make sure player 2 does not follow the true branch of a zero test when the value of the counter being tested is not zero. Finally, player 2 wins if he does not violate any of these restrictions and reaches q_F. He also wins if he wins the matching pennies game, no matter whether he violates the restrictions or not. Player 1 simply wins whenever player 2 does not win.

In total this means that there is a Nash equilibrium where player 2 wins and player 1 loses if M halts with initial counter values c_0 and d_0. If M does not halt with initial values c_0 and d_0 the game is reduced to a matching pennies game which has no Nash equilibrium. Thus, there is a Nash equilibrium in $\mathcal{G}$ if, and only if, M halts, implying that the existence problem is undecidable.

The partially defined strategies specified for the two players in the reduction can trivially be extended to symmetric strategies which makes the symmetric existence problem undecidable as well. □

4.2. Undecidability with partial information

We already mentioned an undecidability proof in Theorem 26 for two players, perfect observation and perfect recall. However, the objectives used for achieving the reduction are quite complex. We explain here how partial observation also leads to undecidability, but for LTL objectives, and with only three players. To show this, we can slightly alter a proof from [30]. There, synthesis of distributed reactive systems with LTL objectives is shown undecidable in the presence of partial observation. The situation used in that proof, where two processes (players 1 and 2) with an LTL objective φ play against a hostile environment (player 3), can be modelled in our framework. The idea is that φ is built from a deterministic Turing machine $\mathcal{M}$ in such a way that the processes can win if, and only if, $\mathcal{M}$ halts on the empty input tape.

On top of this reduction, we add an initial matching-pennies module between player 1 and 2, and slightly change the LTL objectives as follows:

players 1 and 2 still win if φ is true, but each player can also win by winning the initial matching-pennies game. Player 3 still wins if φ is not true. Now, if $\mathcal{M}$ halts on the empty input tape, then there is a Nash equilibrium where players 1 and 2 play in such a way that φ is true; they both win, while player 3 loses and has no winning deviation.

On the other hand, suppose $\mathcal{M}$ does not halt on the empty input tape. Let $\sigma = (\sigma_1, \sigma_2, \sigma_3)$ be a given strategy profile. If player 3 is losing along the outcome of σ, then he can change his strategy and improve, since $\mathcal{M}$ does not halt on the empty input tape; thus σ is not a Nash equilibrium. On the other hand, if player 3 is winning along the outcome of σ, then one of players 1 and 2 is losing. But then, this player can improve by changing his strategy in order to win the initial matching pennies game. Thus, σ is not a Nash equilibrium in this case either. This implies that there exists a Nash equilibrium if, and only if, $\mathcal{M}$ halts on the empty input tape.

Theorem 27. *Deciding the existence of a symmetric Nash equilibrium for* LTL *objectives in symmetric game networks is undecidable for $n \geq 3$ players.*

4.3. Decidability for memoryless strategies

In this section we prove that the existence of a memoryless symmetric Nash equilibrium is decidable, and that it is PSPACE-complete. Notice here that the input of the observation relations $\equiv_i$ are already of size $|\mathsf{States}|^n \times |\mathsf{States}|^n$. In the next section we consider more succinct encodings for these relations.

We first observe that PSPACE-hardness is a direct consequence of the proof of PSPACE-hardness of model-checking of LTL in finite-state transition systems [**?**].

We now explain our algorithms for deciding the (constrained) existence of symmetric Nash equilibria restricted to memoryless strategies. The algorithm is as follows: it first guesses a memoryless strategy for one player, from which it deduces the strategies to be played by the other players. It then looks for the players that are losing, and checks if they alone can improve their payoff.

More formally, we fix a symmetric game network $\mathcal{G} = \langle G, (\equiv_i)_{i\in[n]}, (\Omega_i)_{i\in[n]}\rangle$ with symmetric representation $\pi = (\pi_{i,j})_{i.j\in[n]}$. We assume that each objective Ω_i is given by an LTL formula ϕ_i.

The first step is to guess and store an $\equiv_0$-realizable memoryless strategy σ_0 for player 0. Such a strategy is a mapping from States^n to Act; following our remark above, such a strategy has size polynomial in the size of the

input. We intend player 0 to play according to σ_0, and every player i to play according to $\sigma_0(\pi_{i,0}^{-1}(s_0, ..., s_{n-1}))$ in state $(s_0, ..., s_{n-1})$. From Lemma 15, we know that all symmetric memoryless strategy profiles can be characterized by such an $\equiv_0$-realizable memoryless strategy for player 0.

The algorithm then checks that no player can improve his payoff (and checks the constraint, if any, on the sets of winning and losing players). To this aim, for each player, the algorithm builds the outcome of the strategy profile on-the-fly, and checks that it fulfills the objective of the considered player; if not, it checks whether this player can play differently and satisfy its objective.

Such an algorithm requires storing the memoryless strategy, and building ultimately-periodic paths. This can be performed on-the-fly: having strategy σ_0 stored on the tape, the algorithm computes the moves of all the players, and can then apply the resulting transition. The state space States^n being polynomial in the size of the input, this can be performed in polynomial space.

Theorem 28. *The constrained existence of a memoryless symmetric Nash equilibrium for* LTL *objectives in symmetric game networks is* PSPACE*-complete.*

Remark 29. *Notice that this algorithm can be extended to bounded-memory strategies. The algorithm would then require exponential space if the bound on the memory is given in binary, and still polynomial space otherwise.*

Notice also that the algorithm above could be adapted to handle non-symmetric bounded-memory equilibria in non-symmetric game networks: it would just guess all the strategies, and check the satisfaction of the LTL *objectives in the product automaton obtained by applying the strategies.*

The algorithm could also be adapted, still with the same complexity, to handle richer objectives as in the semi-quantitative setting of [28], where players have several preordered objectives. Instead of guessing the winners, the algorithm would guess, for each player, which objectives are satisfied, and check that no individual improvement is possible. This can be done by listing all possible improvements and checking that none of them can be reached.

5. Succinct Game Networks

As our goal is to represent large networks of components it is not feasible to store the entire observation relation explicitly for all players since this can be very large. In this section we investigate a succinct representation for symmetric game networks that can be represented symbolically in a succinct way.

A succinct symmetric game network is a tuple $\mathcal{P} = \langle G, (\alpha_j)_{j\in[k]}, \equiv, \phi\rangle$ where G is a one-player arena, $\alpha_j \colon \mathbb{N} \to (\mathbb{N} \to \mathbb{N})$ indicate the k neighbours of each player, and $\equiv$ and ϕ are *templates* for defining $\equiv_i$ and ϕ_i for each player. We now explain how a symmetric game network $\mathcal{G} = \mathcal{P}^n$ can be obtained from a succinct symmetric game network $\mathcal{P}$ and an integer $n \geq k$.

For a given n, the state space of $\mathcal{P}^n$ is States^n. Then each $\alpha_j(n)$ is a mapping $[n] \to [n]$; the integer $\alpha_j(n)(i)$ represents the j-th neighbour of player i. We require that the mappings α_j be represented symbolically, e.g. as arithmetic expressions involving j and the arguments n and i. We explain below how this partially defines the symmetric representation for $\mathcal{P}^n$.

The equivalence relation $\equiv$ is a relation over $(\mathsf{States}^k \times \mathbb{N}^{\mathsf{States}})$: the first component deals with the k neighbours of each state, while the second component compares the Parikh image of the configurations[1]. For any $i \in [n]$, and for any two configurations t and t' in States^n, we let

$$t \equiv_i t' \Leftrightarrow (t_{\alpha_j(i)})_{j\in[k]} \times (\#_s(t))_{s\in\mathsf{States}} \equiv (t'_{\alpha_j(i)})_{j\in[k]} \times (\#_s(t'))_{s\in\mathsf{States}}.$$

where $\#_s(t)$ is the number of elements in the configuration t that are given by state s.

For instance, in order to define exact observation of the left- and right neighbours, we would define $\alpha_1(n)(i) = i - 1 \pmod n$, $\alpha_2(n)(i) = i + 1 \pmod n$, and let $\equiv$ relate any two tuples as soon as their first two items (t_1, t_2) and (t'_1, t'_2) match.

Similarly, ϕ is an LTL formula from which the objectives of the players can be derived: the formula is built on two types of atomic propositions:

- for each atomic proposition p appearing in G, and for any $j \in [k]$, p_k is an atomic proposition;
- for any two states s and s', formulas of the form $\#_s \sim c$ and $\#_s - \#_{s'} \sim c$, with $\sim \in \{<, \leq, =, \geq, >\}$ and $c \in \mathbb{N}$, are atomic propositions.

For each $i \in [n]$, formula ϕ_i is then obtained by replacing p_k with $p_{\alpha_k(n)(i)}$. The semantics of these atomic propositions is defined as follows:

- $p_{\alpha_k(n)(i)}$ holds true in configuration t if the label of state $t(\alpha_k(n)(i))$ contains p;

[1]Notice that this slightly differs from the Parikh condition we used in Example **??**: there several conditions would be imposed on Parik images of different subsets of neighbours. The setting defined here could easily be extended to this case.

- $\#_s - \#_{s'} \sim c$ holds true in t if, writing n_s for the number of occurrences of s in t and $n_{s'}$ for the number of occurrences of s' in t, it holds $n_s - n_{s'} \sim c$. Similarly for $\#_s \sim c$.

It remains to see under which conditions the resulting game network $\langle G, (\equiv_i)_{i\in[n]}, (\phi_i)_{i\in[n]}\rangle$ is a symmetric game network: for this, we need to prove the existence of a symmetric representation π. This puts contraints on $(\alpha_j)_{j\in[k]}$, depending on $\equiv$ and ϕ. In the general case (omitting trivial cases where e.g. $\equiv$ is the identity relation, or ϕ is always true), the condition $t \equiv_i t' \Leftrightarrow t(\pi_{i,j}^{-1}) \equiv_j t'(\pi_{i,j}^{-1})$ might give rise to conditions $\pi_{i,j}(\alpha_l(n)(i)) = \alpha_l(n)(j)$ on the symmetric representation. This corresponds to our intuition that the role of player $\alpha_l(n)(j)$ w.r.t. j (namely, being his l-th neighbour) is the same as the role of $\alpha_l(n)(i)$ w.r.t. i. In particular, this in general implies that if $\alpha_l(n)(i) = \alpha_{l'}(n)(i)$ for some i, then $\alpha_l(n)(j) = \alpha_{l'}(n)(j)$ for all $j \in [n]$.

Finally, the initial configuration of a succinct game network is given as a function mapping each integer $n \geq k$ to a configuration in States^n. This can for instance be given as a sequence of pairs (s_j, ϕ_j) where $s_j \in \mathsf{States}$ and ϕ_j is a boolean function taking n and i as argument. Then, in $\mathcal{P}^n$, player i would have initial state s_j for the smallest j for which $\phi_j(n, i)$ is true (requiring that $\phi_l \equiv \top$ for some l, so that such a j always exists).

5.1. Undecidability of parameterized existence

The synthesis of symmetric Nash equilibria for an arbitrary number of players was one of our target applications in this work: we study the problem whether a succinct symmetric game network admits a symmetric Nash equilibrium when the number of players is large enough. More precisely, we aim at deciding the existence of a one-player strategy σ_0, and of an integer n_0, such that the strategy profile obtained by making all n_0 players follow strategy σ_0 (each player having its own observation) is a Nash equilibrium. We show that deciding the existence of such an equilibrium is undecidable, even when considering only memoryless strategies.

Theorem 30. *The existence of a parameterized symmetric Nash equilibrium for LTL objectives in succinct symmetric game networks is undecidable (even for memoryless strategies).*

We first give a proof for existence of a positive symmetric Nash equilibrium and then describe how to do a similar construction without the positivity constraint.

Let $\mathcal{M} = (Q, q_0, \Sigma, \delta, \mathsf{Halt})$ be a deterministic Turing machine ($\delta \colon Q \times \Sigma \to Q \times \Sigma \times \{-1, +1\}$). We assume Halt is a sink state. We build a succinct symmetric game network $\mathcal{P}$ that captures the behaviour of $\mathcal{M}$. We intend to enforce that $\mathcal{M}$ halts if, and only if, there exists n such that $\mathcal{P}^n$ has a positive symmetric Nash equilibrium from some initial configuration. Moreover, we will show that there is a positive Nash equilibrium in $\mathcal{P}^n$ if, and only if, there is a memoryless one.

We first define the one-player arena $G = \langle \mathsf{States}, \{A\}, \mathsf{Act}, \mathsf{Mov}, \mathsf{Tab} \rangle$, which is depicted on Fig. 7, as follows:

- $\mathsf{States} = ((Q \times \Sigma) \cup \Sigma \cup \{\mathsf{Halt}\}) \times \{L, \#, R\}$
- $\mathsf{Act} = \Sigma \cup Q$
- $\mathsf{Mov}(((q, a), \bullet), A) = \Sigma$ if $q \neq \mathsf{Halt}$
  $\mathsf{Mov}((a, \bullet), A) = Q \cup \{a\}$
  $\mathsf{Mov}(((\mathsf{Halt}, a), \bullet), A) = \mathsf{Mov}(\mathsf{Halt}) = \{\mathsf{Halt}\}$
- $\mathsf{Tab}(((q, a), \bullet), b) = (b, \bullet)$ if $q \neq \mathsf{Halt}$
  $\mathsf{Tab}((a, \bullet), q) = ((q, a), \bullet)$, $\mathsf{Tab}((a, \bullet), a) = (a, \bullet)$
  $\mathsf{Tab}(((\mathsf{Halt}, \bullet), a), \mathsf{Halt}) = \mathsf{Tab}((\mathsf{Halt}, \bullet), \mathsf{Halt}) = (\mathsf{Halt}, \bullet)$

Each state is marked with a special symbol in $\{L, \#, R\}$: letters L and R are used to indicate the left-most and right-most cells of the tape, while $\#$ identifies all other cells.

In this reduction, we let $k = 3$ (each player observes two neighbours plus himself), with $\alpha_1(n)(i) = i - 1 \pmod n$, $\alpha_2(n)(i) = i$, and $\alpha_3(n)(i) = i + 1 \pmod n$. The observation relation $\equiv$ is the identity on States^3, with no condition on the Parikh images. This defines a ring topology where each player has perfect observation of himself and of his left and right neighbours.

We now define the objectives of the players, by describing an LTL formula φ. For the sake of readability, we use atomic propositions p_{-1}, p and p_{+1} (instead of p_1, p_2 and p_3, respectively), representing the value of atomic proposition p for players $\alpha_1(n)(i)$, $\alpha_2(n)(i)$ and $\alpha_3(n)(i)$. The LTL formula φ is given in Fig. 8.

Assuming (w.l.o.g.) that all the cells of the tape initially contains a special symbol $\flat$, we set the initial configuration of the network to be $(\flat, \#)$ for all players, except for players 0, 1 and 2, starting respectively in states $(\flat, R)$, $(\flat, L)$ and $((q_0, \flat), \#)$. We write γ^n for this initial configuration.

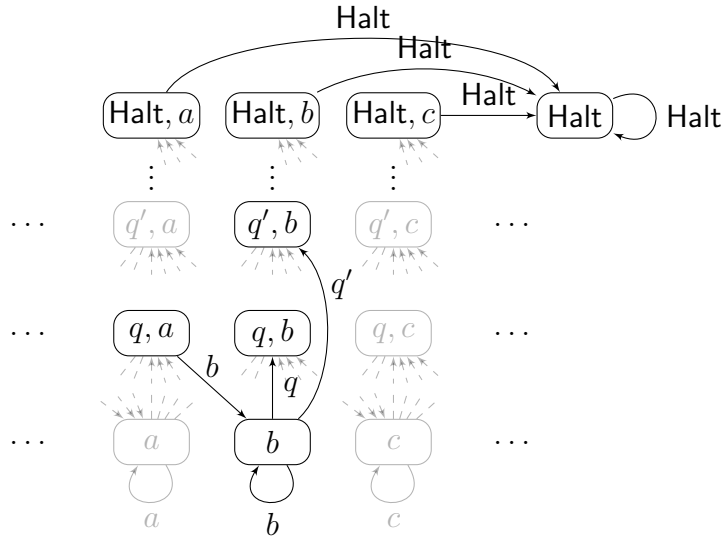

Figure 5: The one-player arena (with transitions for one 4-tuple (q, q', a, b)). Note that the second component $\bullet \in \{L, \#, R\}$ of the states are omitted.

Lemma 31. *If $\mathcal{M}$ halts, then there exists n and a memoryless strategy σ that induces a positive symmetric memoryless Nash equilibrium in $\mathcal{P}^n$ from γ^n.*

Proof. Suppose $\mathcal{M}$ halts. Then the unique finite run ρ of $\mathcal{M}$ uses only a finite number of tape cells. Let this number be n_1, and consider the game $\mathcal{P}^n$, for any $n = n_1 + 2$.

In this game, define a memoryless strategy such that action $\flat$ is always chosen from states $(\flat, L)$ and $(\flat, R)$ (since the tape head never points to these positions). For states with second component $\#$, let the choice of action for player $i \in \{1, ..., n_1\}$ in round k of the game correspond to the content of cell i in the k-th step of the run ρ. It also means playing the control state of $\mathcal{M}$ when the tape head in ρ moves to cell i. As player i can see the contents of the two cells $i + 1$ and $i - 1$, as well as cell i, he can derive from the current state what to play next in this strategy profile. Thus, this can be done using a memoryless strategy. As every player can use this strategy, this induces a symmetric memoryless strategy profile.

As φ_i expresses that player i plays exactly according to the rules of the Turing machine, that some player eventually reaches the Halt state (because it requires all players to eventually stay in the same state), and that the state

$$\varphi = \bigwedge_{\substack{(q,a)\in Q\times\Sigma \text{ s.t.} \\ \delta(q,a)=(q',b,r),\ c,d\in\Sigma \\ \gamma_1,\gamma_2,\gamma_3\in\{L,\#,R\}}} \left[\begin{array}{l} \mathbf{G}\left[\begin{array}{l} \Big(((q,a),\gamma_1)\wedge(c,\gamma_2)_r\wedge(d,\gamma_3)_{-r}\Big)\Rightarrow \\ \qquad \mathbf{X}\Big((b,\gamma_1)\wedge((q',c),\gamma_2)_r\wedge(d,\gamma_3)_{-r}\Big) \end{array} \right] \\ \wedge \\ \mathbf{G}\left[\begin{array}{l} \Big((c,\gamma_1)\wedge(d,\gamma_2)_r\wedge((q,a),\gamma_3)_{-r}\Big)\Rightarrow \\ \qquad \mathbf{X}\Big(((q',c),\gamma_1)\wedge(d,\gamma_2)_r\wedge(b,\gamma_3)_{-r}\Big) \end{array} \right] \end{array} \right]$$

$$\wedge \bigwedge_{\substack{a,b,c\in\Sigma \\ \gamma_1,\gamma_2,\gamma_3\in\{L,\#,R\}}} \mathbf{G}\left[\Big((a,\gamma_1)_{-1}\wedge(b,\gamma_2)\wedge(c,\gamma_3)_{+1}\Big)\Rightarrow \mathbf{X}\,(b,\alpha_2)\right]$$

$$\wedge\ \mathbf{F} \bigvee_{s\in\mathsf{States}} \mathbf{G}\,s\ \wedge \bigwedge_{\gamma\in\{L,R\}} \Big((\flat,\gamma)\Rightarrow \mathbf{G}\,(\flat,\gamma)\Big)$$

Figure 6: Formula φ

of players beginning in $(\flat, L)$ and $(\flat, R)$ never change states, this strategy profile ensures that every player wins as ρ is a halting run.

Thus, the strategy profile defined is a memoryless positive symmetric Nash equilibrium in $\mathcal{P}^n$. Note that this strategy also induces a memoryless positive symmetric Nash equilibrium in $\mathcal{P}^{n'}$ for all $n' \geq n$. □

Lemma 32. *If $\mathcal{M}$ does not halt, then there exists no n for which there is a positive Nash equilibrium in $\mathcal{P}^n$ from γ^n.*

Proof. We do the proof by contraposition. Suppose that there exists n such that there is a positive symmetric Nash equilibrium σ in $\mathcal{P}^n$ from γ^n. Then the unique outcome ϱ of the associated strategy profile from γ^n satisfies φ_i for all $0 \leq i \leq n+1$. In particular, player 0 and player $n_0 - 1$ always choose action $\flat$ and stay in the states $(\flat, L)$ and $(\flat, R)$ respectively.

Further, as φ_i is satisfied in ϱ for all other players, the topmost conjuncts in the definitions of the formulas imply that the players must play according to the unique run ρ of $\mathcal{M}$. The truth of the formula also implies that one of the players eventually plays the halting state as all players eventually keep staying in the same state. This means that $\mathcal{M}$ does in fact halt. □

Theorem 28 now follows from Lemmas 29 and 30. In particular, note that they imply undecidability both with and without the restriction to memoryless strategies. Note also that the proof above is only for the restriction to positive equilibria. However, using techniques similar to the proof of Proposition 23, the proof can be adapted to handle unconstrained Nash equilibria.

5.2. Decidability with bounded memory

In this section, we keep the setting of succinct representations for the observation relation and for the LTL objectives, but fix the number of players. We prove that the existence of a memoryless (or even bounded-memory) symmetric Nash equilibrium is decidable, and that it is EXPSPACE-complete. Notice that we assume that the number of players is given in binary, so that the state space States^n is actually doubly-exponential in the size of the input.

We first notice that EXPSPACE-hardness is a direct consequence of the proof of Theorem 28; the only difference is that we have to consider exponential-space Turing machines. The reason that the size of the tape is exponential in this reduction is that there is one cell for each player. Here, the crucial point is that the number n of players is given in binary in the succinct representation.

The algorithm follows the same line as in Section **??**: it guesses a memoryless strategy to be stored on the tape, and checks that no player has a profitable deviation by guessing paths step-by-step. The strategy maps each information set to an action. The number of information sets is the number of different equivalence classes in $\equiv$: the number of different Parikh images of size n over States is bounded by n^{States}, and the number of different configurations for the k neighbours is States^k. Here k can be assumed to be given in unary, since the input contains one function α_j for each $0 \leq j \leq k-1$. Hence the number of information sets is exponential, and the strategy can be guessed and stored using exponential space.

Checking whether a player meets his objective or has an incentive to deviate from the guessed strategy can be achieved in exponential space, following the same ideas as in Section **??**.

Theorem 33. *Deciding the constrained existence of a memoryless symmetric Nash equilibrium for LTL objectives in succinct symmetric game networks is* EXPSPACE*-complete.*

Remark 34. *As for the case of non-succinct symmetric game networks, this algorithm can be lifted to handle finite-memory strategies. Here, the problem remains* EXPSPACE*-complete, even when the memory bound is given in binary.*

The algorithm can also be adapted to handle non-symmetric equilibria, by guessing and storing exponentially many memoryless strategies (one for each player).

6. Conclusion

In this paper, we have proposed a model of games for large networks of identical devices. This model of games is composed of a single arena, which is duplicated (one copy for each player), and each player has only a partial information on the whole state-space of the system. To fully represent large networks of identical devices, we added symmetry constraints, which yields non-local constraints in the system.

For this model, we have studied several problems related to the computation of symmetric pure Nash equilibria. We have fully characterized the complexity of the (constrained) existence problem for bounded-memory strategies, and we have proven several undecidability results when the memory of the strategies is unbounded.

This work opens many interesting directions of research. Besides solving the questions left open in this paper, these directions include the study of *mixed* Nash equilibria in such networks of games. Other possibilities for further work include extended quantitative objectives, or stronger solution concepts, like sub-game perfect equilibria [27] or secure equilibria [31]. Restriction to interesting subclasses of observation relations and network topologies is also important to find meaningful special cases with lower complexity.

References

[1] T. A. Henzinger, Games in system design and verification, in: Proc. 10th Conference on Theoretical Aspects of Rationality and Knowledge (TARK'05), 2005, pp. 1–4.

[2] J. F. Nash, Jr., Equilibrium points in n-person games, Proc. National Academy of Sciences 36 (1) (1950) 48–49.

[3] K. Chatterjee, R. Majumdar, M. Jurdziński, On Nash equilibria in stochastic games, in: Proc. 18th International Workshop on Computer Science Logic (CSL'04), 2004, pp. 26–40.

[4] M. Ummels, D. Wojtczak, The complexity of Nash equilibria in stochastic multiplayer games, Logical Methods in Computer Science 7 (3:20).

[5] P. Bouyer, R. Brenguier, N. Markey, M. Ummels, Pure Nash equilibria in concurrent games, Logical Methods in Computer Science 11 (2:9).

[6] K. Chatterjee, T. A. Henzinger, N. Piterman, Strategy logic, Information and Computation 208 (6) (2010) 677–693.

[7] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, Reasoning about strategies: On the model-checking problem, ACM Transactions on Computational Logic 15 (4) (2014) 34:1–34:47.

[8] F. Laroussinie, N. Markey, Augmenting ATL with strategy contexts, Information and Computation(To appear).

[9] J. F. Nash, Jr., Non-cooperative games, Annals of Mathematics 54 (2) (1951) 286–295.

[10] P. Dasgupta, E. Maskin, The existence of equilibrium in discontinuous economic games, 1: theory, The Review of Economic Studies 53 (1) (1986) 1–26.

[11] B. Aminof, S. Jacobs, A. Khalimov, S. Rubin, Parametrized model checking of token-passing systems, in: Proceedings of the 15th International Workshop on Verification, Model Checking, and Abstract Interpretation (VMCAI'14), 2014, pp. 262–281.

[12] B. Aminof, T. Kotek, S. Rubin, F. Spegni, H. Veith, Parameterized model checking of rendezvous systems, in: CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings, 2014, pp. 109–124.

[13] S. M. German, A. P. Sistla, Reasoning about systems with many processes, Journal of the ACM 39 (3) (1992) 675–735.

[14] E. A. Emerson, A. P. Sistla, Symmetry and model checking, Formal Methods in System Design 9 (1-2) (1996) 105–131.

[15] R. Alur, S. La Torre, P. Madhusudan, Modular strategies for recursive game graphs, Theor. Comput. Sci. 354 (2) (2006) 230–249.

[16] P. Wolper, V. Lovinfosse, Verifying properties of large sets of processes with network invariants, in: Proceedings of the 1st International Workshop on Automatic Verification Methods for Finite State Systems (CAV'89), 1990, pp. 68–80.

[17] P. A. Abdulla, B. Jonsson, On the existence of network invariants for verifying parameterized systems, in: Correct System Design, Recent Insight and Advances, 1999, pp. 180–197.

[18] N. Bertrand, P. Fournier, A. Sangnier, Playing with probabilities in reconfigurable broadcast networks, in: Proceedings of the 17th International Conference on Foundations of Software Science and Computation Structure (FoSSaCS'14), 2014, pp. 134–148.

[19] R. Alur, Th. A. Henzinger, O. Kupferman, Alternating-time temporal logic, Journal of the ACM 49 (2002) 672–713.

[20] P. A. Abdulla, B. Jonsson, M. Nilsson, M. Saksena, A survey of regular model checking, in: Proceedings of the 15th International Conference on Concurrency Theory (CONCUR'04), 2004, pp. 35–48.

[21] H. Bherer, J. Desharnais, R. St-Denis, Control of parameterized discrete event systems, Discrete Event Dynamic Systems 19 (2) (2009) 213–165.

[22] H. Bherer, Controller synthesis for parameterized discrete event systems, Ph.D. thesis, Université Laval, Québec, Canada (2009).

[23] W. Thomas, Infinite games and verification, in: Proc. 14th International Conference on Computer Aided Verification (CAV'02), 2002, pp. 58–64.

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

[25] C. Baier, J.-P. Katoen, Principles of Model-Checking, MIT Press, 2008.

[26] P. Bouyer, N. Markey, D. Stan, Mixed Nash equilibria in concurrent games, in: Proceedings of the 34th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'14), 2014, pp. 351–363.

[27] M. J. Osborne, An Introduction to Game Theory, Oxford University Press, 2004.

[28] P. Bouyer, R. Brenguier, N. Markey, M. Ummels, Concurrent games with ordered objectives, in: Proc. 15th International Conference on Foundations of Software Science and Computation Structure (FoSSaCS'12), 2012, pp. 301–315.

[29] F. Mogavero, A. Murano, M. Y. Vardi, Reasoning about strategies, in: Proceedings of the 30th Conferentce on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'10), Leibniz-Zentrum für Informatik, 2010, pp. 133–144.

[30] A. Pnueli, R. Rosner, Distributed reactive systems are hard to synthesize, in: Proc. 31st Annual Symposium on Foundations of Computer Science (FOCS'90), IEEE Computer Society Press, 1990, pp. 746–757.

[31] K. Chatterjee, T. A. Henzinger, M. Jurdzinski, Games with secure equilibria, Theor. Comput. Sci. 365 (1-2) (2006) 67–82.

[32] F. Brandt, F. A. Fischer, M. Holzer, Symmetries and the complexity of pure nash equilibrium, J. Comput. Syst. Sci. 75 (3) (2009) 163–177.

[33] S. fen Cheng, D. M. Reeves, Y. Vorobeychik, M. P. Wellman, Notes on equilibria in symmetric games, in: Proceedings of the 6th International Workshop On Game Theoretic And Decision Theoretic Agents (GTDT), 2004, pp. 71–78.

[34] C. T. Ryan, A. X. Jiang, K. Leyton-Brown, Computing pure strategy nash equilibria in compact symmetric games., in: EC, 2010, pp. 63–72.

[35] C. Papadimitriou, The complexity of finding Nash equilibria, in: N. Nisan, T. Roughgarden, É. Tardos, V. V. Vazirani (Eds.), Algorithmic Game Theory, Cambridge University Press, 2007, pp. 29–51.

[36] Y. Shoham, K. Leyton-Brown, Multiagent Systems - Algorithmic, Game-Theoretic, and Logical Foundations, Cambridge University Press, 2009.

Appendix A. Symmetric Normal-Form Games

In this section, we question the definition of symmetric normal-form games introduced in [10] which is as follows:

Definition 35. *A symmetric normal-form game is a tuple* $\langle [n], S, (u_i)_{i\in[n]}\rangle$ *where* $[n]$ *is the set of players,* S *is a finite set of strategies and* $u_i \colon S^n \to \mathbb{R}$ *are utility functions such that for all strategy vectors* $(a_0, ..., a_{n-1}) \in S^n$*, all permutations* π *of* $[n]$ *and all* i *it holds that*

$$u_i(a_0, ..., a_{n-1}) = u_{\pi(i)}(a_{\pi(0)}, ..., a_{\pi(n-1)}).$$

In [10] and other sources where symmetric games are defined [32, 33, 34, 24, 35, 36] the basic intuition of what a symmetric normal-form game should be is a game where all players have the same set of actions and the same utility functions in the sense that the utility of player should depend exactly on which action he chooses himself and on the number of other players choosing any particular action. However, the definition above does not seem to capture the scenario at hand and might even be erroneous. For two players we agree with the definition, and indeed in [24, 36] it is only defined for two players. In [32, 33, 34, 35] however it is defined such that $u_i(a_i, a_{-i}) = u_j(a_j, a_{-j})$ whenever $a_i = a_j$ and $a_{-i} = a_{-j}$. Here, a_i means the action taken by player i and a_{-i} means the set of actions taken by players other than player i. This definition seems to agree with the intuition about the games which we wish to represent. One argument against Definition 34 is that it has the following consequence:

Theorem 36. *Using Definition 34 for games with more than two players, for all pairs of players* i, j *and all strategy profiles* $(a_0, ..., a_{n-1})$ *we have*

$$u_i(a_0, ..., a_{n-1}) = u_j(a_0, ..., a_{n-1}).$$

In particular, for zero-sum games, we have $u_i(a_0, ..., a_{n-1}) = 0$ *for every player* i *and all strategy profiles* $(a_0, ..., a_{n-1})$*.*

Proof. We start by looking at games with $n \geq 3$. First, using the permutation[2] $(1, 2, 0, ...)$, we get

$$u_0(a_0, a_1, a_2, ...) = u_1(a_1, a_2, a_0, ...).$$

[2]We write $(p_0, p_1, p_2, ...)$ to denote the permutation $(i \mapsto p_i)$ for all i, so that permutation $(1, 2, 0)$ is $(0 \mapsto 1, 1 \mapsto 2, 2 \mapsto 0, ...)$.

Then, using $(1, 0, 2, ...)$ (which exchanges the first two players), we get

$$u_1(a_1, a_2, a_0, ...) = u_0(a_2, a_1, a_0, ...).$$

Putting these equations together, we get $u_0(a_0, a_1, a_2, ...) = u_0(a_2, a_1, a_0, ...)$. This means that player 0 can switch action with player 2 without changing his utility. This can be done for arbitrary opponents by symmetry and since we can switch actions of all other players without changing the utility using permutations where $\pi(0) = 0$, we have that $u_0(a_0, ..., a_{n-1}) = u_0(a_{\pi(0)}, ..., a_{\pi(n-1)})$ for every permutation π. By symmetry, for every i and every permutation π, we get

$$u_i(a_0, ..., a_{n-1}) = u_i(a_{\pi(0)}, ..., a_{\pi(n-1)}).$$

Now consider any two players i and j as well as a permutation π such that $\pi(i) = j$. Then we have

$$\begin{aligned} u_i(a_0, ..., a_{n-1}) &= u_{\pi(i)}(a_{\pi(0)}, ..., a_{\pi(n-1)}) \\ &= u_j(a_{\pi(0)}, ..., a_{\pi(n-1)}) \\ &= u_j(a_0, ..., a_{n-1}) \end{aligned}$$

using the result above. This means that for every choice of actions of the players, every player gets the same utility. □

One could define a class of games with this property, however for the definition at hand the property is not true for games with two players, which is a quite strange property of a class of games.

In the following definition we propose a fix to the above definition. The resulting definition is equivalent to the definitions in [32, 33, 34, 35], but it is given in a form resembling Definition 34. Note that this formulation has similarities with our own definition of a symmetric game network (Definition 8).

Definition 37. *A symmetric normal-form game is a tuple* $\langle [n], S, (u_i)_{i \in [n]} \rangle$ *where* $[n]$ *is the set of players,* S *is a finite set of strategies, and* $u_i \colon S^n \to \mathbb{R}$ *are utility functions such that for all strategy vectors* $(a_0, ..., a_{n-1}) \in S^n$*, all permutations* π *of* $[n]$ *and all* i*, it holds that*

$$u_i(a_0, ..., a_{n-1}) = u_{\pi^{-1}(i)}(a_{\pi(0)}, ..., a_{\pi(n-1)}).$$

The intuition behind this definition is as follows. Suppose we have a strategy profile $\sigma = (a_0, ..., a_{n-1})$ and a strategy profile where the actions of

the players have been rearranged by permutation π, $\sigma_\pi = (a_{\pi(0)}, ..., a_{\pi(n-1)})$. We would prefer that player j, using the same action in σ_π as player i does in σ, gets the same utility. Since j uses $a_{\pi(j)}$, this means that $\pi(j) = i \Rightarrow j = \pi^{-1}(i)$. Now, from this intuition we have that $u_i(a_0, ..., a_{n-1}) = u_j(a_{\pi(0)}, ..., a_{\pi(n-1)}) = u_{\pi^{-1}(i)}(a_{\pi(0)}, ..., a_{\pi(n-1)})$. Apart from this intuition, the new definition can be shown to be equivalent to the one from [32, 33, 34, 35]. The reason that we agree with Definition 34 for two-player games is that $\pi = \pi^{-1}$ for all permutations π of two elements.

Appendix B. A server-client architecture

We show how to model the server-client architecture in a symmetric way. Assume S represents the arena for the server, and C for the client. The arena G is composed of three disconnected components: S, C, and an extra state sink with a selfloop on it. We assume there are n clients. The game $\mathcal{G}$ is defined as the $(2n+1)$-players game network $\mathcal{G} = (G, (\equiv_i)_{i\in[2n+1]}, (\Omega_i)_{i\in[2n+1]})$ where:

- we define $N_S(0) = \{0, 1, \ldots, n\}$, for every $1 \leq i \leq n$, we define $N_S(i) = \{0, i, n+1 \ldots, 2n-1\}$, and for every $n+1 \leq i \leq 2n$, we define $N_S(i) = \{0, n+1, \ldots, 2n\}$
- we define $N_C(0) = \{0, n+1\}$, and for every $1 \leq i \leq 2n$, we define $N_C(i) = \{0, i\}$
- $(s_0, \ldots, s_{2n-2}) \equiv_i (s'_0, \ldots, s'_{2n-2})$ if, and only if, the following conditions hold:
  - s_i is an S-state if, and only if, s'_i is an S-state
  - s_i is a C-state if, and only if, s'_i is a C-state
  - if s_i is an S-state, then for every $j \in N_S(i)$, $s_j = s'_j$
  - if s_i is a C-state, then for every $j \in N_C(i)$, $s_j = s'_j$

This is a symmetric game network. A possible symmetric representation is: $\pi_{0,i}(0) = i$, $\pi_{0,i}(1) = i$ and $\pi_{0,i}(j) = n - j + 1$ if $1 \leq i \leq n$; $\pi_{0,i}(0) = i$, $\pi_{0,i}(1) = i$ and $\pi_{0,i}(2) = n + 1$... $\pi_{0,i}(n) = 2n$ if $n + 1 \leq i \leq 2n$.

Assuming that player 0 will play in S, players $1, \ldots, n$ will play in C, and all other players are trapped in sink, this properly model the server-client architecture since in that case:

- $\equiv_0$ reduces to $\mathsf{Id}([n])$
- $\equiv_i$ reduces to $\mathsf{Id}(\{0, i\})$ for every $1 \leq i \leq n$

With this modeling, we will be interested in symmetric Nash equilibria starting at configuration $(s_0, c_0, \ldots, c_0, \mathsf{sink}, \ldots, \mathsf{sink})$ where s_0 is the initial state of S and c_0 the initial state of C.