Games for Counting Abstractions*

J.-F. Raskin[1], M. Samuelides[2] and L. Van Begin[1†]
[1] Computer Science Department, University of Brussels, Belgium
[2] Ecole Normale Supérieure de Cachan, France

Abstract

It has been recently shown that monotonicity is not sufficient to obtain decidability results for two-player games. However, positive results can be obtained on restricted subclasses of monotonic two-player games. In this paper, we identify and study a subclass of monotonic two-player games that is useful for analysis of parametric (open-)systems that can be modeled by using counting abstractions. Although the reachability game problem is undecidable in general for that subclass, we identify two interesting and decidable problems and show how to apply those results in parametric system analysis.

1 Introduction

Model-checking methods were originally proposed for the automatic verification of critical systems that have natural finite-state abstractions. Nevertheless, much *recent* interest has concerned the application of model-checking methods to infinite-state systems. Several interesting classes of infinite state systems have been shown decidable. For example, Alur et al. [AD94] showed that timed automata have a decidable *reachability problem*. Finkel et al. in [FS01], and Abdulla et al. in [ACJT96] have shown that infinite, but *monotonic*, transition systems (also called well-structured transition systems) have a decidable *coverability problem*. For instance, Petri nets and broadcast protocols define monotonic transition systems.

Timed automata, Petri nets, and broadcast protocols are usually used to model *reactive systems* embedded in a critical environment. But those formalisms define transition systems that are semantics models for *closed systems*. In closed systems, we do not distinguish between the reactive system and its environment. So the properties that we can verify on transition systems are

*Supported by the FRFC project "Centre Fédéré en Vérification" funded by the Belgian National Science Fundation (FNRS) under grant nr 2.4530.02. This work was done partially when the second author was visiting the University of Brussels.

†This author was supported by a "First Europe" grant EPH3310300R0012 of the Walloon Region.

properties in which we cannot distinguish between the role of the reactive system and the role of the environment. If we want to distinguish the role of the reactive system and the environment in which it is embedded, we can use games played on state spaces.

Usual transition systems can be considered as one-player games on which only closed-system verification problems can be formulated. The *control* and *modular verification* problems of systems can be studied as two-player games played on state spaces, where *one player*, say player 1, represents the reactive system and the *other player*, say player 2, represents the environment. If the state space on which the game is played is infinite then we have to solve infinite-state games. Infinite-state games have not yet been studied as intensively as traditional verification problems on infinite-state transition systems. Nevertheless, recently there have been several interesting works in that direction. Here are some examples. In [MPS95], Maler et al. study how to solve games defined by timed automata. In [Wal96], Walukiewicz studies how to solve infinite games defined by pushdown automata. In [dAHM01], Henzinger et al. study symbolic algorithms to solve general infinite-state games.

In this paper, we study two-player games played on infinite but *monotonic game structures*. In particular, we study games that are useful to study *parametric systems*. Parametric systems are systems where the number of instances of component types is not fixed a priori. In general, we are interested to verify such systems for any number of instances. The notion of *counting abstraction* [GS92] is a powerful tool to reason on parametric systems and consists in only retaining, for each component type, the number of instances that are in each possible (local) configuration. Hence, if the number of component types and the number of (local) configurations for each component is finite, then states of parametric systems can be abstracted by integer vectors. In that context, parametric systems are modeled as vector addition systems with states (VASS for short). Here, we want to consider parametric systems as open systems over which game properties can be formulated and verified. For that, we identify an interesting *subclass* of monotonic game structures for which the *coverability game* and the *deadlock-avoidance game* problems are decidable. A restricted form of two-player VASS systems define game structures that fall into that class. We illustrate the interest of our decidable game structures with an example from parametric systems.

Related works Recently, Abdulla et al. [ABd03] have studied the class of monotonic game structures. On the negative side, they have shown that the reachability problem is undecidable on the general class of monotonic game structures. On the positive side, they have identified a subclass of monotonic game structures on which the reachability problem is decidable. The subclass that they have identified is called the class of *downward closed game structures*. Downward-closed game structures are game structures where at least one of the two players has downward closed behaviors, i.e. if one of the players can reach a configuration c' from a configuration c it can also reach c' from all the configurations than are greater or equal to c. This is relevant for example when one player may lose messages in a lossy channel system. The interest of this model is thus clear in the context of communication protocols. Unfortunately, the positive results of their paper has *no* direct and natural application in the context of counting abstractions. The subclass of monotonic game structures that we identify in this paper and for which we obtain (other) interesting decidability results is different from the class of downward closed game structures. Our class of monotonic game structure, contrary to the class of downward closed monotonic game structure, has direct and natural applications to the game analysis of counting abstractions. Our positive results are thus new and important in the context of automatic game-based analysis of parametric systems.

Structure of the paper In Section 2, we recall preliminaries, define monotonic game structures, reachability, coverability and deadlock avoidance games and their associated decision problems. In Section 3, we recall some well-known results for solving reachability games. In Section 4, we identify a subclass of monotonic game structures for which coverability game and deadlock-avoidance game problems are decidable. In section 5, we show how a subclass of open VASS systems can be used to define monotonic game structures that fall into the decidable subclass. We finish that section by illustrating the interest of this class for the game analysis of parametric systems abstracted using the counting abstraction.

2 Preliminaries

Well quasi-orderings A *well quasi ordering* $\preccurlyeq$ on the elements of a set S, wqo for short, is a *reflexive* and *transitive* relation such that for any infinite sequence $s_0 s_1 \dots s_n \dots$ of elements in S, there exist indices i and j, such that $i < j$ and $s_i \preccurlyeq s_j$. In the following, we note $s_i \prec s_j$ if $s_i \preccurlyeq s_j$ but $s_j \not\preccurlyeq s_i$. For example, it is well-known that the quasi order $\sqsubseteq \subseteq \mathbb{N}^k \times \mathbb{N}^k$ defined as $\langle m_1, m_2, \dots, m_k \rangle \sqsubseteq \langle m'_1, m'_2, \dots, m'_k \rangle$ if $m_i \leq m'_i$ for any $1 \leq i \leq k$ is a wqo. In this paper, we will concentrate on wqo. Given a wqo $\preccurlyeq$ over the elements of S, a set $U \subseteq S$ is called an *$\preccurlyeq$-upward closed set* if for any $s_1 \in U$, for any $s_2 \in S$ such that $s_1 \preccurlyeq s_2$, we have that $s_2 \in U$. We now recall one useful result from [Hig52]:

Lemma 1 *Let S be a set of elements, $\preccurlyeq \subseteq S \times S$ be a wqo, and $S_0 S_1 \dots S_n \dots$ be an infinite sequence of $\preccurlyeq$-upward closed subsets of S such that $S_i \subseteq S_{i+1}$ for any $i \geq 0$, then there exists $j \geq 0$ such that for any $k \geq j$, $S_j = S_k$.*

Two-player game structures A (two-player) *game structure* G is a tuple $\langle C, C_1, C_2, \rightarrow \rangle$ where C is a (potentially infinite) *set of configurations* partitioned into the set of player 1 (also called the *protagonist*) configurations C_1 and the set of player 2 (also called the *antagonist*) configurations C_2 (that is $C_1 \cap C_2 = \emptyset$ and $C = C_1 \cup C_2$), and $\rightarrow \subseteq (C_1 \times C_2) \cup (C_2 \times C_1)$ is the *transition relation*. In the following, we note $c \rightarrow c'$ when $(c, c') \in \rightarrow$. A *play* P in the game structure G from a configuration c is either an infinite sequence of configurations $c_0c_1 \ldots c_n \ldots$ such that $c_0 = c$ and $c_i \to c_{i+1}$ for all $i \geq 0$, or a finite maximal sequence of configurations $c_0c_1 \ldots c_n$ such that $c_0 = c$ and for all i, $0 \leq i < n$, we have that $c_i \to c_{i+1}$, and there does not exist $c \in C$ such that $c_n \to c$. We write $\lg(P)$ to denote the length of the play P, which is equal to the number of configurations in P, if P is finite, and is equal to $+\infty$ if P is infinite. Let G be a game structure and c be a configuration of G, we note $P(G, c)$ for the set of all plays in G starting from configuration c. A *winning condition* W for a game structure G and a configuration c is a subset $W \subseteq P(G, c)$, that is: a subset of plays starting in c. A *game* is a triple $\langle G, c, W\rangle$ where G is a game structure, c is a configuration of G, and W is the subset of plays starting in c. During a play, players apply strategies. Let C^* denotes finite sequences of configurations from set of configurations C. A *strategy* for player $i \in \{1, 2\}$ (i-strategy for short) is a partial function $\mathcal{S} : C^* \to C$ such that $\mathsf{dom}(S) = \{c_1c_2 \ldots c_n \mid c_n \in C_i\}$ and if we have $\mathcal{S}(c_1c_2 \ldots c_n) = c'$, $c_n \to c'$. A strategy is *memory free* if it is such that for any $c_1c_2 \ldots c_n \in C^*$, if $\mathcal{S}(c_1c_2 \ldots c_n)$ is defined then $\mathcal{S}(c_1c_2 \ldots c_n) = \mathcal{S}(c_n)$, that is the strategy only depends on the current configuration and not on the history of the play. The *outcome of a player-i strategy* is defined as follows ($i \in \{1, 2\}$). Let $\mathcal{S}$ be a i-strategy, the outcome of $\mathcal{S}$ in configuration c is the set of all plays $P = c_0c_1 \ldots c_n \ldots \in P(G, c)$, noted $\mathsf{Outcome_i}(G, c, \mathcal{S})$, such that: for any j, $0 \leq j < \lg(P)$, we have that if $c_j \in C_i$ and $\mathcal{S}(c_0c_1 \ldots c_j)$ is defined then $c_{j+1} = \mathcal{S}(c_0c_1 \ldots c_j)$. So, the outcome of a i-strategy S from a configuration c is the set of plays starting in c that are generated when player i plays with strategy $\mathcal{S}$. In the sequel of the paper, we always suppose that player 1 is the protagonist and tries to win the game while player 2 is the antagonist and tries to prevent player 1 from winning the game. We say that player 1 *has a winning strategy* for the game $\langle G, c, W\rangle$ if $\mathsf{Outcome}_1(G, c, \mathcal{S}) \subseteq W$ for some $\mathcal{S}$. We say that player 2 *has a spoiling strategy* for the game $\langle G, c, W\rangle$ if $\mathsf{Outcome}_2(G, c, \mathcal{S}) \cap W = \emptyset$ for some $\mathcal{S}$. Notice that player 1 has a winning strategy if and only if player 2 does not have a spoiling strategy for the games we will consider in this paper [GS53].

Monotonic game structures In this paper, we concentrate our attention on a class of infinite game structures, called monotonic game structures, introduced independently in [ABd03] and in [RSV03].

Definition 1 (Monotonicity) A *game structure* $\langle C, C_1, C_2, \to\rangle$ is *monotonic* for a wqo $\preccurlyeq\, \subseteq (C_1 \times C_1) \cup (C_2 \times C_2)$ if the following condition is verified: for any $c_1, c_2 \in C$, if $c_1 \to c_2$, then for all $c_3 \in C$, such that $c_1 \preccurlyeq c_3$, there exists $c_4 \in C$ with $c_3 \to c_4$ and $c_2 \preccurlyeq c_4$.

Given a game structure $G = \langle C, C_1, C_2, \to\rangle$ and a wqo $\preccurlyeq\, \subseteq (C_1 \times C_1) \cup (C_2 \times C_2)$ such that G is monotonic for $\preccurlyeq$, then we write $G = \langle C, C_1, C_2, \to, \preccurlyeq\rangle$ to underline that G is a monotonic game for $\preccurlyeq$.

Games and associated decision and synthesis problems In this paper, we concentrate on two important games: *reachability games* and *deadlock-avoidance games*. In a reachability game, player 1 tries to force the game into a given set of configurations (as a consequence player 2 tries to avoid that the game enters this set of configurations). In a deadlock-avoidance game, player 1 tries to avoid the set of deadlock configurations (as a consequence player 2 tries to force the game into the set of deadlock configurations). We now give a formal definition to those two games.

A *reachability game* is defined by a triple $\langle G, c, F\rangle$ where G is a game structure with set of configurations C, $c \in C$ is a configuration of G, and $F \subseteq C$ is a subset of configurations, called the *winning configurations*. The triple $\langle G, c, F\rangle$ defines the game $\langle G, c, W\rangle$ where W is the set of plays starting in c that contain at least one configuration of F. The *reachability game problem* is defined as follows: given a reachability game defined by a triple $\langle G, c, F\rangle$, does player 1 have a winning strategy for this game?

When the underlying game structure is monotonic for a given wqo $\preccurlyeq$, it is of interest to consider a special class of reachability games called coverability games. A *coverability game* is defined by a reachability game $\langle G, c, F\rangle$ where G is a monotonic game structure for a given wqo $\preccurlyeq$ on the configurations of G, c is a configuration of G, and F is an upward closed set of configurations for the wqo $\preccurlyeq$. The *coverability game problem* is defined as follows: given a coverability game defined by a triple $\langle G, c, F\rangle$, does player 1 have a winning strategy for this game ?

Besides reachability games, we study in this paper so-called deadlock-avoidance games. In a deadlock-avoidance game, player 1 tries to avoid that the play enters the set of *deadlock configurations*, i.e. configurations that have no outgoing transitions. Formally, a *deadlock-avoidance game* is a game $\langle G, c, W\rangle$ where W is the set of all the infinite plays of G starting in c. The *deadlock-avoidance problem* is defined as follows: Given a deadlock-avoidance game $\langle G, c, W\rangle$, does the player 1 have a winning strategy for this game?

Besides solving decision problems on infinite game structures, we are also interested in strategy synthesis problems. Given a game for which player 1 has a winning strategy, solving the *strategy synthesis problem* consists in constructing a winning strategy for this game.

3 Solving reachability and deadlock-avoidance games - Known results

Given a set of configurations $S \subseteq C$ of a game G, we define the following set: $\mathsf{CPre}_{1,G}(S)$ is the set of configurations where player 1 has a one step strategy to reach S, that is $\mathsf{CPre}_{1,G}(S)$ equals

$$\begin{gathered}\{c \in C_1 \mid \exists c' \in S : c \rightarrow c'\}\\ \cup\{c \in C_2 \mid \exists c' \in S : c \rightarrow c' \text{ and } \forall c' \in C : c \rightarrow c' \text{ implies } c' \in S\}.\end{gathered}$$

We define $\mathsf{CPre}^0_{1,G}(S)$ as S, for any $n \in \mathbb{N}$, $\mathsf{CPre}^{n+1}_{1,G}(S)$ as $\mathsf{CPre}_{1,G}(\mathsf{CPre}^n_{1,G}(S))$ and $\mathsf{CPre}^*_{1,G}(S) = \bigcup_{n\in\mathbb{N}} \mathsf{CPre}^n_{1,G}(S)$. It is well-known, see for example [dAHM01], that the following theorem holds:

Theorem 1 (From [dAHM01]) *For any reachability game $\langle G, c, F\rangle$, we have that player 1 has a winning strategy for the game $\langle G, c, F\rangle$ iff $c \in \mathsf{CPre}^*_{1,G}(F)$.*

Note that the operator $\mathsf{CPre}_{1,G}$ is monotonic w.r.t. the inclusion relation between sets of configurations. Unfortunately, this is not sufficient to ensure the decidability of reachability problems on monotonic game structures (the iteration of CPre may not stabilize). In [ABd03], Abdulla, Bouajjani et al. have shown the undecidability of both reachability and coverability problems on monotonic game structures. We also proved those results independently in [Sam03, RSV03].

Furthermore, by using a construction similar to those presented in [ABd03, RSV03, Sam03], we can easily reduce the termination problem for two counter machines to the deadlock-avoidance problem for (a subclass of) monotonic games. Since the termination problem is undecidable [Min67], we deduce the undecidability of the deadlock-avoidance problem for monotonic games.

Theorem 2 *The reachability, coverability and deadlock-avoidance game problems are undecidable for the class of monotonic games.*

4 B-game structures

As shown in the previous section, monotonic games are not satisfactory from a computational point of view. In this section, we identify a new interesting subclass of monotonic games that enjoys decidability results for the coverability and deadlock-avoidance game problems.

Remember that in practice, player 1 models the behaviours and decisions that the system can take and player 2 describes the environment in which the system is embedded. In a lot of practical cases, the whole set of behaviours of the environment can be modelled using a *finite state structure*. Based on this fact, we define here a subclass of monotonic game structures, called B-game structures[1], where the environment respects additional properties.

Definition 2 (B-game structures) A *B-game structure* G is a monotonic game structure $\langle C, C_1, C_2, \rightarrow, \preccurlyeq\rangle$ with the following additional property: for any $c_1, c_2 \in C_2$, $c_3 \in C_1$ if $c_1 \rightarrow c_3$ and $c_2 \preccurlyeq c_1$ then there exists $c_4 \in C_1$ such that $c_2 \rightarrow c_4$ and $c_4 \preccurlyeq c_3$.

As we will see in the following, particular B-game structures correspond to game structures where player 1 (the system) is equivalent to a VASS (counter system) and player 2 (the environment) is equivalent to a finite state automaton.

Since the reachability problem for monotonic systems is undecidable in general (for instance, the reachability between two markings is undecidable for several monotonic extensions of Petri nets, see [Duf98] for details), the following theorem holds.

[1] We call them B-game structure because one of the two players has access only to a bounded amount of information to make its decisions.

Theorem 3 *The reachability game problem is undecidable for B-games.*

Proof. *Sketch.* We reduce the reachability problem for monotonic extensions of Petri nets to the reachability game problem for B-games as follows. Given an extended Petri net $\mathcal{N}$, the configurations of the game are pairs $\langle \mathbf{m}, i \rangle$ where $\mathbf{m}$ is a marking of $\mathcal{N}$ and $i \in \{1, 2\}$ is the identity of the current player, transitions from configurations $\langle \mathbf{m}, 1 \rangle$ are $\langle \mathbf{m}, 1 \rangle \to \langle \mathbf{m}', 2 \rangle$ such that we have $\mathbf{m} \to \mathbf{m}'$ in $\mathcal{N}$, i.e. the behaviours of player 1 are defined by the whole set of transitions of $\mathcal{N}$. From the configurations $\langle \mathbf{m}, 2 \rangle$, we have the only transition $\langle \mathbf{m}, 2 \rangle \to \langle \mathbf{m}, 1 \rangle$. Hence, each play of this game corresponds to an execution of $\mathcal{N}$ and we conclude that $\mathbf{m}$ is reachable from $\mathbf{m}_0$ in $\mathcal{N}$ if and only if player 1 has a strategy to reach $\langle \mathbf{m}, 1 \rangle$ from $\langle \mathbf{m}_0, 1 \rangle$. Moreover, if we define $\preccurlyeq$ such that $\langle \mathbf{m}_1, i \rangle \preccurlyeq \langle \mathbf{m}_2, i \rangle$ if and only if $\mathbf{m}_1 \sqsubseteq \mathbf{m}_2$ where $\sqsubseteq$ is the classical wqo on markings of (extended) Petri nets, it is easy to show that the game is a B-game since player 2 leaves the configurations of $\mathcal{N}$ unchanged. Since the reachability problem is undecidable for several extensions of Petri nets, we conclude that the reachability game problem is undecidable for B-games. $\square$

On the other hand, we will show in the next two sub-sections that the coverability and deadlock-avoidance problems are decidable on B-game structures.

4.1 Coverability games are decidable on B-game structures

In order to apply lemma 1, we next prove that, contrary to (general) monotonic game structure, the CPre operator associated to B-game structures returns an upward-closed set of configurations when applied to an upward-closed set of configurations. This is the key to obtain the decidability of the coverability problem.

Lemma 2 *Let G be a B-game structure $\langle C, C_1, C_2, \to, \preccurlyeq \rangle$, let $U \subseteq C$ be any $\preccurlyeq$-upward-closed set of configurations of G, then $\mathsf{CPre}_{1,G}(U)$ is an $\preccurlyeq$-upward-closed set of configurations.*

Proof. Remember that $\mathsf{CPre}_{1,G}(U) = \{c \in C_1 \mid \exists c' \in U : c \to c'\} \cup \{c \in C_2 \mid \exists c' \in U : c \to c' \text{ and } \forall c' \in C : c \to c' \text{ implies } c' \in U\}$. Let $c \in \mathsf{CPre}_{1,G}(U)$ and c_0 a configuration such that $c \preccurlyeq c_0$. We will show that $c_0 \in \mathsf{CPre}_{1,G}(U)$. We study two cases. Case 1: $c \in C_1$. There exists $c' \in U$ such that $c \to c'$. Since G is monotonic and $c \preccurlyeq c_0$, there exists $c_0' \in C$ with $c_0 \to c_0'$ and $c' \preccurlyeq c_0'$. $c_0' \in U$ because U is upward closed and so $c_0 \in \mathsf{CPre}_{1,G}(U)$. Case 2: $c \in C_2$. There exists $c' \in U$ such that $c \to c'$ and for all $c' \in C$, $c \to c'$ implies $c' \in U$. Since G is monotonic there exists $c_0' \in C$ with $c_0 \to c_0'$ and $c' \preccurlyeq c_0'$. Let c_0' a configuration such that $c_0 \to c_0'$. Moreover, since G is a B-game and $c \preccurlyeq c_0$, there exists $c' \in C$ with $c \to c'$ and $c' \preccurlyeq c_0'$. $c \to c'$ implies $c' \in U$. U is upward closed, so $c_0' \in U$ and $c_0 \in \mathsf{CPre}_{1,G}(U)$. These two cases allow us to conclude that $\mathsf{CPre}_{1,G}(U)$ is upward closed. $\square$

The previous lemma together with lemma 1 allow us to state the following result about B-game structures:

Lemma 3 *Let G be a B-game structure $\langle C, C_1, C_2, \rightarrow, \preccurlyeq \rangle$ and let $F \subseteq C$ be an $\preccurlyeq$-upward-closed set, the sequence $S_0 S_1 \dots S_n \dots$ of sets of configurations defined by $S_0 = F$, and for any $i \geq 1$, $S_i = \bigcup_{l=0}^{l=i} \mathsf{CPre}^l_{1,G}(F)$, is such that there exists $j \geq 0$ such that for any $k \geq j$, $S_k = S_{k+1}$.*

This last lemma means that the iteration of $\mathsf{CPre}_{1,G}$ operator starting from an $\preccurlyeq$-upward closed set stabilizes after a finite number of steps. So, if $\mathsf{CPre}_{1,G}(U)$ can be effectively computed for any upward-closed U, the equality test between two upward closed sets and the inclusion test of a configuration into an upward closed set are decidable, then the coverability problem is decidable. In the next subsection, we define B-VASS game structures for which we can effectively compute $\mathsf{CPre}_{1,G}(U)$ for any upward-closed set U and decide the equality and inclusion tests.

Theorem 4 *The coverability problem is decidable for B-game structures – provided that $\mathsf{CPre}_{1,G}(U)$ is computable for any upward-closed set U, the equality test between two upward closed sets and the inclusion test of a configuration into an upward closed set are decidable.*

Strategy synthesis We have shown the decidability of the coverability problem for B-game structures G such that: (i) for any upward closed set U we can compute $\mathsf{CPre}_{1,G}(U)$; (ii) given two upward closed sets U_1 and U_2 we can decide if $U_1 = U_2$; and (iii) given an upward closed set U and a configuration c, we can decide if $c \in U$.

We now show that we can automatically construct winning strategies for those games, provided that for all configuration c and upward closed set U, if $\{c' \mid c \rightarrow c'\} \cap U \neq \emptyset$ then we can compute at least one $c' \in U$ such that $c \rightarrow c'$.

Let (G, c, U) be a coverability game defined by the B-game structure G with the wqo $\preccurlyeq$ as defined in Definition 2, and U be an $\preccurlyeq$-upward closed set of configurations. We assume that $c \in \mathsf{CPre}^*_{1,G}(U)$. Then we can construct a winning memory free 1-strategy $\mathcal{S}$ with the following algorithm. For any configuration c' of G, we define $\mathcal{S}(c')$ as follows. If $c' \in U$ we do not need to define $\mathcal{S}(c')$. If $c' \in \mathsf{CPre}^*_{1,G}(U) \setminus U$, we compute $\mathsf{CPre}^n_{1,G}(U)$ where n is the smallest integer such that $c' \in \mathsf{CPre}^n_{1,G}(U)$. We choose $\mathcal{S}(c')$ among $\{c'' \in \mathsf{CPre}^{n-1}_{1,G}(U) \mid c' \rightarrow c''\}$.

4.2 Deadlock-avoidance games are decidable on B-games

The *deadlock-avoidance tree* associated to a monotonic game structure G with initial configuration c_{init}, noted T_G, is the smallest finite prefix of the unfolding of G starting in c_{init} such that any leaf n of T_G is such that either (assume that $\mathsf{conf}(n)$ is the configuration associated to the node n)

1. the set $\{c \mid \mathsf{conf}(n) \rightarrow c\}$ is empty; or

2. there exists an ancestor n' of n in T_G such that $\mathsf{conf}(n') \preccurlyeq \mathsf{conf}(n)$.

Lemma 4 *A deadlock-avoidance tree exists for any finitely branching monotonic game structure.*

Proof. Suppose that such a tree does not exist for a monotonic finitely branching game structure G with initial configuration c_{init}. Then, there exists an infinite play $c_1c_2\ldots$ with $c_1 = c_{init}$ such that there is no $i < j$ with $c_i \preccurlyeq c_j$. However, since $\preccurlyeq$ is a wqo, such a play does not exist from which we derive a contradiction. □

Given a deadlock avoidance tree T_G with set of nodes N, transition relation $succ$ and labeling function $\mathsf{conf} : N \to C$, the function $\mathsf{label} : N \to \{1, 2\}$ is a labeling function that satisfies the following properties :

1. for each leaf n of T_G, $\mathsf{label}(n) = 2$ if $\mathsf{conf}(n)$ has no successor in G. Otherwise $\mathsf{label}(n) = 1$;
2. for each node n of T_G that is not a leaf, if $\mathsf{conf}(n) \in C_1$ then $\mathsf{label}(n) = 1$ if there exists $n' \in succ(n)$ such that $\mathsf{label}(n') = 1$, otherwise $\mathsf{label}(n) = 2$. If $\mathsf{conf}(n) \in C_2$, then $\mathsf{label}(n) = 1$ if for all $n' \in succ(n)$: $\mathsf{label}(n') = 1$, otherwise $\mathsf{label}(n) = 2$.

The reduced deadlock-avoidance tree of T_G is a tree constructed from T_G by removing all the sub-trees rooted by nodes n such that $\mathsf{label}(n) = 2$.

We now show that deadlock-avoidance trees are tools to reason about deadlock-avoidance B-games.

Lemma 5 *Given a finitely branching deadlock-avoidance B-game $\langle G, c_{init}, W\rangle$ and the deadlock-avoidance tree T_G with root node n_{init}, player 1 has a winning strategy if $\mathsf{label}(n_{init}) = 1$, otherwise player 2 has a spoiling strategy.*

Proof. First, let us show that there exists a winning strategy for player 1 if $\mathsf{label}(n_{init}) = 1$. The underlying idea consists in showing that player 1 can force plays to be such that we can associate to the configurations c of plays nodes n of the reduced deadlock-avoidance tree constructed from T_G such that $\mathsf{conf}(n) \preccurlyeq c$. Since for any node n of the reduced deadlock-avoidance tree we have that $\mathsf{conf}(n)$ has at least one successor, then by monotonicity, so are the configurations of the plays. We conclude that the plays are infinite, hence winning for player 1.

Note that $\mathsf{conf}(n_{init}) \preccurlyeq c_{init}$. Suppose a prefix of a play $c_1 \ldots c_k$ with $c_1 = c_{init}$ such that we can associate to each c_i a node n_i of the reduced deadlock-avoidance tree such that $\mathsf{conf}(n_i) \preccurlyeq c_i$ $(1 \leq i \leq k)$. Let us show that player 1 can force the game to go into a configuration c_{k+1} such that we can associate a node n_{k+1} with $\mathsf{conf}(n_{k+1}) \preccurlyeq c_{k+1}$. One of two cases hold:

- either $\mathsf{conf}(n_k) \in C_1$. Let $n = n_k$ if n_k is not a leaf of the deadlock-avoidance tree, otherwise let $n = n'$ where n' is an ancestor of n_k such that $\mathsf{conf}(n') \preccurlyeq \mathsf{conf}(n_k)$. Since $\mathsf{label}(n) = 1$, we know that there exists a successor n'' of n in the reduced deadlock avoidance tree such that

$\mathsf{conf}(n) \to \mathsf{conf}(n'')$ and by monotonicity there exists $c_{k+1} \succcurlyeq \mathsf{conf}(n'')$ such that $c_k \to c_{k+1}$. So, player 1 can choose c_{k+1} such that we can associate a node n'' of the reduced deadlock-avoidance tree with $\mathsf{conf}(n'') \preccurlyeq c_{k+1}$.

- or $\mathsf{conf}(n_k) \in C_2$. Let $n = n_k$ if n_k is not a leaf of the deadlock-avoidance tree, otherwise let $n = n'$ where n' is an ancestor of n_k such that $\mathsf{conf}(n') \preccurlyeq \mathsf{conf}(n_k)$. Since $\mathsf{conf}(n) \preccurlyeq c_k$ and $\mathsf{label}(n) = 1$, we have, by definition of B-games, that for every successor c' of c_k there exists a successor node n' of n in the deadlock avoidance tree such that $\mathsf{conf}(n') \preccurlyeq c'$ and the game is not in a deadlock state.

Hence, in all the cases either the player 1 can force the game to go into a configuration c_{k+1} such that there exists a node n of the reduced deadlock-avoidance tree with $\mathsf{conf}(n) \preccurlyeq c_{k+1}$ or player 2 is forced to go into such a configuration c_{k+1}.

Second, we prove that player 2 has a spoiling strategy if $\mathsf{label}(n_{init}) = 2$ in a similar manner. Indeed, by definition of the function label, player 2 can force the plays to follow paths of the deadlock-avoidance tree where all the nodes are labeled by 2. Since the leaf nodes of those paths have no successors, such plays are finite. $\square$

Theorem 5 *The deadlock-avoidance problem is decidable for finitely branching B-game structures – provided the successor relation and the wqo $\preccurlyeq$ are decidable.*

Strategy Synthesis We now show how to construct a winning strategy for deadlock avoidance games defined by a (finitely branching) B-game structure G with initial configuration c_{init} such that the deadlock-avoidance tree of G has a root node labeled by 1.

We construct a memory free deadlock-avoidance strategy $\mathcal{S}$ as follows: for all configuration $c \in C_1$ such that there exists a node n of the reduced deadlock-avoidance tree such that $l(n) \preccurlyeq c$, we define $\mathcal{S}(c) = c'$ such that $l(n) \to l(n')$, i.e. $n' \in succ(n)$, and $l(n') \preccurlyeq c'$. Notice that if n is a leaf, then there exists a predecessor node n'' in the deadlock-avoidance tree with $l(n'') \prec l(n)$, hence $l(n'') \prec c$, and the node n is replaced by n''.

5 B-VASS games

In this section, we show that the two general decidability results presented in the previous section have applications in parametric systems analysis where systems are abstracted with the so-called *counting abstraction* [GS92]. Parameterized systems are systems where the number of instances of component types is not known. Counting abstraction consists in only retaining, for each component type, the number of instances that are in each possible (local) configuration. Hence, if the number of component types and the number of (local) configurations for each component is finite, then states of parametric systems can be simply represented by integer vectors.

Before the introduction of B-VASS games, we recall the definition of VASS.

Given a set V of variables, $\mathcal{F}(V, D)$ with $D \in \{\mathbb{N}, \mathbb{Z}\}$ denotes the set of functions $f : V \to D$ that associate to each variable in V an element in D.

Definition 3 (VASS) A *VASS* is a tuple $A = \langle L, l_0, V, \to\rangle$ where L is a finite set of locations, $l_0 \in L$ is the initial location, V is a set of variables (also called counters) and $\to \subseteq L \times \mathcal{F}(V, \mathbb{N}) \times \mathcal{F}(V, \mathbb{Z}) \times L$ is the transition relation.

In the following, we assume that for any $\langle l_1, G, A, l_2\rangle \in \to$, if $A(v) < 0$ then $G(v) + A(v) \geq 0$. This last condition is a syntactic condition that ensures the counters of the VASS to be always non-negative.

Note that a VASS can be seen as a Petri net where places correspond to counters, and Petri net transitions and edges correspond to VASS transitions.

A valuation of the variables in V is a function $v \in \mathcal{F}(V, \mathbb{N})$. A state s of a VASS $\langle L, l_0, V, \to\rangle$ is a tuple $\langle l, v\rangle$ where $l \in L$ and v is a valuation of the variables in V. We need some additional notions. A transition $t = \langle l_1, G, A, l_2\rangle \in \to$ is firable from a state $s = \langle l, v\rangle$ if $l = l_1$ and $G(x) \leq v(x)$ for all $x \in V$. Firing t from $s = \langle l_1, v\rangle$ leads to the state $s' = \langle l_2, v'\rangle$ (noted $s \to^t s'$) such that $v'(x) = v(x) + A(x)$ for all $x \in V$. Given a set Σ of synchronization labels, the set $\mathcal{A}(\Sigma)$ of actions constructed from Σ is the set containing

- *internal actions:* a such that $a \in \Sigma$;
- *rendez-vous:* $a!$ (send) and $a?$ (receive) such that $a \in \Sigma$.

Definition 4 (Communicating VASS) A *pair of communicating VASS* is a tuple $\langle A_1, A_2, V, \Sigma, \mathcal{L}\rangle$ where $A_1 = \langle L_1, l_1, V, \to_1\rangle$ and $A_2 = \langle L_2, l_2, V, \to_2\rangle$ are VASS, Σ is a set of synchronization labels, $\mathcal{L} : (\to_1 \cup \to_2) \to \mathcal{A}(\Sigma)$ is a function that labels the transitions of A_1 and A_2 by actions, and V is the set of counters shared by A_1 and A_2.

Given a pair of communicating VASS $\langle A_1, A_2, V, \Sigma, \mathcal{L}\rangle$, the underlying game structure $\langle C, C_1, C_2, \to\rangle$, called VASS game structure, is defined as follows: $C_1 = \{\langle l_1, l_2, v, 1\rangle \mid l_1 \in L_1, l_2 \in L_2,\ v \in \mathcal{F}(V, \mathbb{N})$ is a valuation $\}$, $C_2 = \{\langle l_1, l_2, v, 2\rangle \mid l_1 \in L_1, l_2 \in L_2,\ v \in \mathcal{F}(V, \mathbb{N})$ is a valuation $\}$, and $\to = \to_i \cup \to_r$. We define $\to_i$ and $\to_r$ as follows:

- $\to_i = \{\langle\langle l_1, l_2, v, 1\rangle, \langle l_1', l_2', v', 2\rangle\rangle \mid \langle l_1, v\rangle \to^t \langle l_1', v'\rangle, \mathcal{L}(t)$ is an internal action, $l_2' = l_2\} \cup \{\langle\langle l_1, l_2, v, 2\rangle, \langle l_1', l_2', v', 1\rangle\rangle \mid \langle l_2, v\rangle \to^t \langle l_2', v'\rangle, \mathcal{L}(t)$ is an internal action, $l_1' = l_1\}$;
- $\to_r = \{\langle\langle l_1, l_2, v, 1\rangle, \langle l_1', l_2', v', 2\rangle\rangle \mid \langle l_1, v\rangle \to^{t_1} \langle l_1', v'\rangle, \langle l_2, v\rangle, \to^{t_2} \langle l_2', v'\rangle$ and either $\mathcal{L}(t_1) = a!$ and $\mathcal{L}(t_2) = a?$, or $\mathcal{L}(t_1) = a?$ and $\mathcal{L}(t_2) = a!\}$

We define the partial order $\overline{\sqsubseteq}$ on the configurations of a pair of communicating VASS $\langle A_1, A_2, V, \Sigma, \mathcal{L}\rangle$ as follows: $\langle l_1, l_2, v, i\rangle \overline{\sqsubseteq} \langle l_1', l_2', v', i'\rangle$ if and only if $i = i'$, $l_1 = l_1'$, $l_2 = l_2'$ and $v \preccurlyeq v'$ where $v \preccurlyeq v'$ if and only if $v(x) \leq v'(x)$ for all $x \in V$. It is easy to see that $\overline{\sqsubseteq}$ is a wqo.

Lemma 6 *Any VASS game structure is monotonic for $\overline{\sqsubseteq}$.*

Let us now define a subset of pairs of communicating VASS that defines B-game structures.

Definition 5 (B-Communicating VASS) A *B-pair of communicating VASS* is a pair of communicating VASS $\langle A_1, A_2, V, \Sigma, \mathcal{L}\rangle$ where

1. A_1 is a VASS with a transition relation $\rightarrow_1$ such that for all $\langle l_1, G, A, l_2\rangle \in\rightarrow_1$ labeled by $a?$, we have that G associates 0 to each variables in V; and

2. A_2 is a VASS with a transition relation $\rightarrow_2$ such that for all $\langle l_1, G, A, l_2\rangle \in\rightarrow_2$, we have $A \in \mathcal{F}(V, \mathbb{N})$ and G associates 0 to each variable $x \in V$.

The underlying intuition behind B-pairs of communicating VASS is that the first player is a VASS, i.e. equivalent to a Petri net, and the second player is a finite automata that communicates with player 1 with synchronization labels (rendez-vous) or by incrementing the counters of player 1 (but all the choices of player 2 are independent from the value of the unbounded counters). This is usually acceptable in practice because the environment can in most practical cases be abstracted as a finite state automaton.

Proposition 1 *B-pairs of communicating VASS define finitely branching B-game structures.*

Proof. Lemma 6 states that VASS games structures, hence B-VASS game structures, are monotonic for $\overline{\sqsubseteq}$.

So, given a B-pair of communicating VASS $G = \langle A_1, A_2, V, \Sigma, \mathcal{L}\rangle$ that defines the game structure $\langle C, C_1, C_2, \rightarrow, \overline{\sqsubseteq}\rangle$, it remains to prove that for all the configurations $c = \langle l_1, l_2, v, 2\rangle, c' = \langle l'_1, l'_2, v', 2\rangle \in C_2$ with $c' \overline{\sqsubseteq} c$, if we have $c \rightarrow c''$, then there exists $c''' \overline{\sqsubseteq} c''$ with $c' \rightarrow c'''$. In both cases of internal action and rendez-vous, only increments are applied and no test on the variables are evaluated when passing from c to c''. Since $c' \overline{\sqsubseteq} c$, $l_1 = l'_1$ and $l_2 = l'_2$, hence the same transition(s) can be fired from c' and leads to a state c''' with $c''' \overline{\sqsubseteq} c''$. $\Box$

Using Proposition 1 we next show that the coverability problem and the deadlock-avoidance problem are decidable for B-VASS game structure.

Theorem 6 *The coverability and deadlock-avoidance game problems are decidable for the class of B-VASS game structures. Moreover, we can solve the strategy synthesis problem for coverability and deadlock-avoidance games.*

Proof. It is easy to show that $\overline{\sqsubseteq}$ is decidable and the successor relation is computable for pairs of communicating VASS. Hence we conclude that the deadlock-avoidance game problem is decidable and we can solve the strategy synthesis problem for deadlock-avoidance games.

Moreover, upward-closed sets are classically represented by their set of minimal elements which are guaranteed to be finite when the considered order is a wqo. It is easy to show that using such a representation for upward closed sets of configurations of pairs of communicating VASS, we can compute $\mathsf{CPre}_{1,G}(U)$ for any upward closed set U, the equality test between upward closed sets and the inclusion test of a configuration into an upward closed set are decidable (see for instance [Van03] for more details). Hence, the coverability game problem is decidable for B-VASS game structures. Moreover, since the successor relation is computable, we can solve the strategy synthesis problem for coverability games. □

In [Sam03], we showed the undecidability of game problems for a class equivalent to B-pairs of communicating VASS where accepting conditions are inspired by classical problems on VASS (and Petri nets). More precisely, we showed that we cannot decide if player 1 has a winning strategy for games where the accepting condition is defined either by a set of configurations to avoid, a LTL formula, the counters that must remain bounded all along the plays or one counter that must be bounded all along the plays.

Let us now show how Theorem 6 can be used in practice to reason about parametric systems.

6 An application in parametric systems analysis

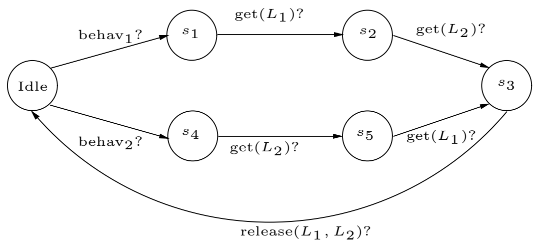

Figure 1: A process.

We present here a system embedded in a simple environment that can be naturally modeled with a B-pair of communicating VASS. That system is composed of processes that use two resources L_1 and L_2 with a mutually exclusive policy, as shown in Figure 1. Processes have two possible behaviours: either they take L_1 and then L_2, or they take first L_2 and then L_1. A resource manager decides to which processes the resources are given. As the system can be composed of any number of processes, we model the entire system by applying a so-called *counting abstraction*, i.e. counters are used to count the number of processes that are in each of their possible states: I counts the number of processes that are in an idle state and x_i counts the number of processes that are in the state s_i for i such that $1 \leq i \leq 5$.

The behaviour of the entire system is described by the VASS obtained by composing the *resource manager* VASS and the *Process* VASS shown in Figure 2.

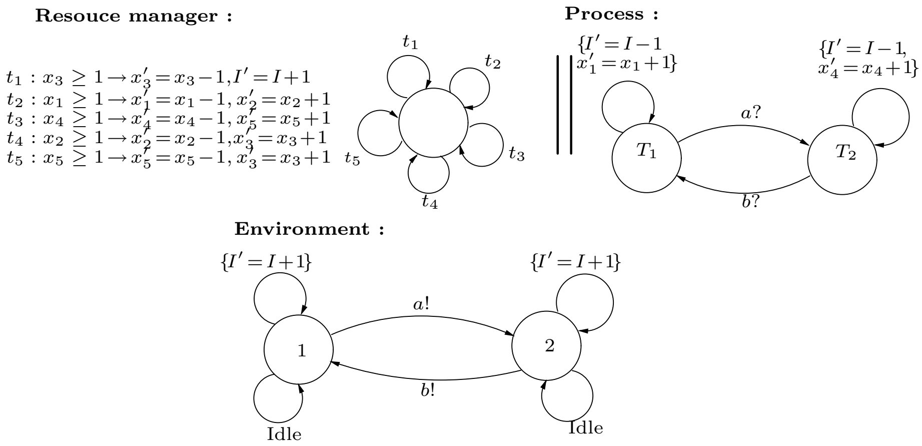

Figure 2: An example.

The *Resource Manager* VASS manages the access to L_1 and L_2 and the *Process* VASS manages the behaviours of processes; when it is in the state T_1 processes access first to L_1 and then to L_2, the converse otherwise. The initial state of the system is $\langle\langle L_1 L_2\rangle, T_1\rangle$ with all the counters equal to 0 except I which is equal to 1. As shown in Figure 2, in each state the resource manager can chose non-deterministically between the subset of transitions $\{t_1, \ldots, t_5\}$ that are enabled. Those transitions give accesses to shared resources.

The *environment* VASS describes a simple environment interacting with the system: either it adds a new process to the system, send a signal (a or b) to the system that changes the behaviour of the processes (those that have started some treatments finish them first), or does nothing. It is easy to be convinced that the pair of communicating VASS composed of the VASS describing the system together with the environment VASS is a B-pair of communicating VASS. Indeed, the environment VASS only increments counters and transitions that synchronize with other ones do not test or modify counters.

The question that we ask is as follows. Can the resource manager resolve its choices (when several transitions are enabled in the set $\{t_1, \ldots, t_5\}$) and ensure that no deadlock can occur no matter how the environment is behaving?

To answer this question and compute a strategy that will ensure this property, we construct a deadlock avoidance tree as defined in section 4.2. By lemma 4.6, this construction is guaranteed to terminate and by lemma 4.7 we know that this tree contains enough information to decide if the transition relation of the resource manager can be restricted to ensure that the resulting system is deadlock free no matter how the environment behaves. From the deadlock avoidance tree, we can extract the strategy, as explained at the end of section 4.2, that is summarized in figure 3. For instance, when the resource manager has granted a process the resource L_1 (resp. L_2), then it must grant this process the resource L_2 before granting the resource L_2 (resp. L_1) to another process. This strategy is thus synthesized automatically using the results of this paper.

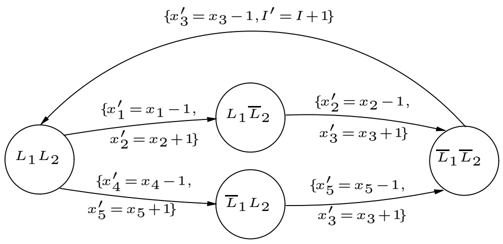

Figure 3: Strategy for the resource manager to avoid deadlocks.

References

[ABd03] P. A. Abdulla, A. Bouajjani, and J. d'Orso. Deciding monotonic games. In *in Proc. Intern. Workshop Computer Science Logic (CSL'03)*, volume 2803, Vienna (Austria), 2003. Springer Verlag.

[ACJT96] P. A. Abdulla, K. Cerans, B. Jonsson, and Y.-K. Tsay. General Decidability Theorems for Infinite-state Systems. In *Proceedings of the 11th Annual Symposium on Logic in Comuter Science (LICS'96)*, pages 313–321. IEEE Computer Society Press, 1996.

[AD94] R. Alur, D.L. Dill. A theory of timed automata. *Theoretical Computer Science* 126:183–235, 1994.

[dAHM01] L. de Alfaro, T.A. Henzinger, and R. Majumdar. Symbolic algorithms for infinite-state games. In *CONCUR 01: Concurrency Theory*, Lecture Notes in Computer Science 2154, pages 536–550. Springer-Verlag, 2001.

[Duf98] C. Dufourd. *Réseaux de Petri avec reset/transfert : Décidabilité et indécidabilité*. PhD thesis, ENS de Cachan, 1998.

[FS01] A. Finkel and P. Schnoebelen. Well-structured transition systems everywhere! *Theoretical Computer Science*, 256(1-2):63–92, 2001.

[GS53] D. Gale and F.M. Stewart. Infinite Games with Perfect Information. In *Contribution of the Theory Game II*, Annals of Mathematics Studies (28), pages 245–266, 1953.

[GS92] S. M. German and A. P. Sistla. Reasoning about Systems with Many Processes. *Journal of ACM*, 39(3):675–735, 1992.

[Hig52] G. Higman. Ordering by divisibility in abstract algebras. *Proc. London Math. Soc.*, 3(2):326–336, 1952.

[Min67] N.M. Minsky. *Finite and Infinite Machines.* Englewood Cliffs, N.J., Prentice-Hall, 1967.

[MPS95] O. Maler, A. Pnueli, J. Sifakis. On the synthesis of discrete controllers for timed systems. In *Proc. 12th Symposium on Theoretical Aspects of Computer Science (STACS'95)*, Lecture Notes in Computer Science 900, pages 229-242, 1995.

[Sam03] M. Samuelides. Jeux dans un réseau de Petri. *Mémoire de DEA*, Ecole Normale Suprieure de Cachan, France, 2003.

[RSV03] J.-F. Raskin, M. Samuelides, L. Van Begin. Petri games are monotonic but difficult to decide. *Technical report 508*, Université Libre de Bruxelles, Belgium.

[Van03] L. Van Begin Efficient Verification of Counting Abstractions for Parametric Systems. Ph.D. thesis, Université Libre de Bruxelles, 2003.

[Wal96] I. Walukiewicz. Pushdown processes: games and model checking. In *Proceeding of the 8th International Conference on Computer-Aided Verification (CAV'96)*, Lecture Notes in Computer Science 1102, pages 62–75, 1996.