Simple Priced Timed Games Are Not That Simple

Thomas Brihaye, Gilles Geeraerts, Axel Haddad, Engel Lefaucheux, Benjamin Monmege

Simple Priced Timed Games Are Not That Simple*

Thomas Brihaye[1], Gilles Geeraerts[2], Axel Haddad[1], Engel Lefaucheux[3], and Benjamin Monmege[4]

1 **Université de Mons, Belgium,** {thomas.brihaye,axel.haddad}@umons.ac.be
2 **Université libre de Bruxelles, Belgium,** gigeerae@ulb.ac.be
3 **LSV, ENS Cachan, Inria Rennes, France,** engel.lefaucheux@ens-cachan.fr
4 **LIF, Aix-Marseille Université, CNRS, France,** benjamin.monmege@lif.univ-mrs.fr

Abstract

Priced timed games are two-player zero-sum games played on priced timed automata (whose locations and transitions are labeled by weights modeling the costs of spending time in a state and executing an action, respectively). The goals of the players are to minimise and maximise the cost to reach a target location, respectively. We consider priced timed games with one clock and arbitrary (positive and negative) weights and show that, for an important subclass of theirs (the so-called *simple* priced timed games), one can compute, in exponential time, the optimal values that the players can achieve, with their associated optimal strategies. As side results, we also show that one-clock priced timed games are determined and that we can use our result on simple priced timed games to solve the more general class of so-called reset-acyclic priced timed games (with arbitrary weights and one-clock).

1998 ACM Subject Classification D.2.4 Software/Program Verification, F.3.1 Specifying and Verifying and Reasoning about Programs

Keywords and phrases Priced timed games; Real-time systems; Game theory

Digital Object Identifier 10.4230/LIPIcs.FSTTCS.2015.p

1 Introduction

The importance of models inspired from the field of game theory is nowadays well-established in theoretical computer science. They allow to describe and analyse the possible interactions of antagonistic agents (or players) as in the *controller synthesis* problem, for instance. This problem asks, given a model of the environment of a system, and of the possible actions of a controller, to compute a controller that constraints the environment to respect a given specification. Clearly, one can not, in general, assume that the two players (the environment and the controller) will collaborate, hence the need to find a *controller strategy* that enforces the specification *whatever the environment does*. This question thus reduces to computing a so-called winning strategy for the corresponding player in the game model.

In order to describe precisely the features of complex computer systems, several game models have been considered in the literature. In this work, we focus on the model of Priced Timed Games [17] (PTGs for short), which can be regarded as an extension (in several directions) of classical finite automata. First, like timed automata [2], PTGs have *clocks*,

* The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under Grant Agreement n°601148 (CASSTING).

35th International Conference on Foundation of Software Technology and Theoretical Computer Science (FSTTCS'15).
Editors: Prahladh Harsha and Ganesan Ramalingam; pp. 1–14

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

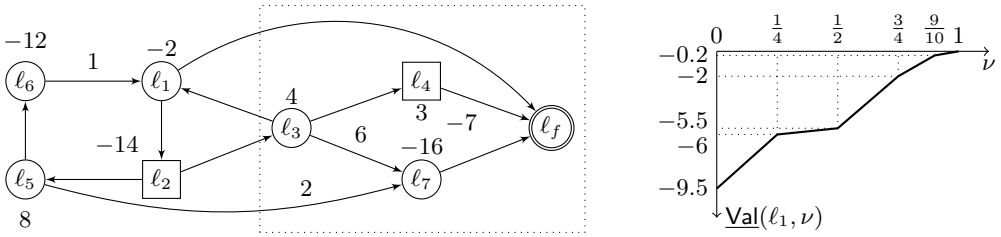

Figure 1 A simple priced timed game (left) and the lower value function of location ℓ_1 (right).

which are real-valued variables whose values evolve with time elapsing, and which can be tested and reset along the transitions. Second, the locations are associated with price-rates and transitions are labeled by discrete prices, as in priced timed automata [4, 3, 6]. These prices allow one to associate a *cost* with each run (or play), which depends on the sequence of transitions traversed by the run, and on the time spent in each visited location. Finally, a PTG is played by two players, called Min and Max, and each location of the game is owned by either of them (we consider a turn-based version of the game). The player who controls the current location decides how long to wait, and which transition to take.

In this setting, the goal of Min is to reach a given set of target locations, following a play whose cost is as small as possible. Player Max has an antagonistic objective: he tries to avoid the target locations, and, if not possible, to maximise the accumulated cost up to the first visit of a target location. To reflect these objectives, we define the upper value $\overline{\mathsf{Val}}$ of the game as a mapping of the configurations of the PTG to the least cost that Min can guarantee while reaching the target, whatever the choices of Max. Similarly, the lower value $\underline{\mathsf{Val}}$ returns the greatest cost that Max can ensure (letting the cost being $+\infty$ in case the target locations are not reached).

An example of PTG is given in Figure 1, where the locations of Min (respectively, Max) are represented by circles (respectively, rectangles), and the integers next to the locations are their price-rates, i.e., the cost of spending one time unit in the location. Moreover, there is only one clock x in the game, which is never reset and all guards on transitions are $x \in [0,1]$ (hence this guard is not displayed and transitions are only labeled by their respective discrete cost): this is an example of *simple priced timed game* (we will define them properly later). It is easy to check that Min can force reaching the target location ℓ_f from all configurations (ℓ, ν) of the game, where ℓ is a location and ν is a real valuation of the clock in $[0,1]$. Let us comment on the optimal strategies for both players. From a configuration (ℓ_4, ν), with $\nu \in [0,1]$, Max better waits until the clock takes value 1, before taking the transition to ℓ_f (he is forced to move, by the rules of the game). Hence, Max's optimal value is $3(1-\nu) - 7 = -3\nu - 4$ from all configurations (ℓ_4, ν). Symmetrically, it is easy to check that Min better waits as long as possible in ℓ_7, hence his optimal value is $-16(1-\nu)$ from all configurations (ℓ_7, ν). However, optimal value functions are not always *that simple*, see for instance the lower value function of ℓ_1 on the right of Figure 1, which is a piecewise affine function. To understand why value functions can be piecewise affine, consider the sub-game enclosed in the dotted rectangle in Figure 1, and consider the value that Min can guarantee from a configuration of the form (ℓ_3, ν) in this sub-game. Clearly, Min must decide how long he will spend in ℓ_3 and whether he will go to ℓ_4 or ℓ_7. His optimal value from all (ℓ_3, ν) is thus $\inf_{0 \leqslant t \leqslant 1-\nu} \min\big(4t + (-3(\nu+t) - 4), 4t + 6 - 16(1-(\nu+t))\big) = \min(-3\nu - 4, 16\nu - 10)$. Since $16\nu - 10 \geqslant -3\nu - 4$ if and only if $\nu \leqslant 6/19$, the best choice of Min is to move instantaneously

to ℓ_7 if $\nu \in [0, 6/19]$ and to move instantaneously to ℓ_4 if $\nu \in (6/19, 1]$, hence the value function of ℓ_3 (in the subgame) is a piecewise affine function with two pieces.

Related work. PTGs were independently investigated in [8] and [1]. For (non-necessarily turn-based) PTGs with *non-negative* prices, semi-algorithms are given to decide the *value problem* that is to say, whether the upper value of a location (the best cost that Min can guarantee in valuation 0), is below a given threshold. They have also shown that, under the *strongly non-Zeno assumption* on prices (asking the existence of $\kappa > 0$ such that every cycle in the underlying region graph has a cost at least κ), the proposed semi-algorithms always terminate. This assumption was justified in [11, 7] by showing that, without it, the *existence problem*, that is to decide whether Min has a strategy guaranteeing to reach a target location with a cost below a given threshold, is indeed undecidable for PTGs with non-negative prices and three or more clocks. This result was recently extended in [9] to show that the *value problem* is also undecidable for PTGs with non-negative prices and four or more clocks. In [5], the undecidability of the existence problem has also been shown for PTGs with arbitrary price-rates (without prices on transitions), and two or more clocks. On a positive side, the value problem was shown decidable by [10] for PTGs with one clock when the prices are non-negative: a 3-exponential time algorithm was first proposed, further refined in [18, 16] into an exponential time algorithm. The key point of those algorithms is to reduce the problem to the computation of optimal values in a restricted family of PTGs called *Simple Priced Timed Games* (SPTGs for short), where the underlying automata contain no guard, no reset, and the play is forced to stop after one time unit. More precisely, the PTG is decomposed into a sequence of SPTGs whose value functions are computed and re-assembled to yield the value function of the original PTG. Alternatively, and with radically different techniques, a pseudo-polynomial time algorithm to solve one-clock PTGs with arbitrary prices on transitions, and price-rates restricted to two values amongst $\{-d, 0, +d\}$ (with $d \in \mathbf{N}$) was given in [14].

Contributions. Following the decidability results sketched above, we consider PTGs with one clock. We extend those results by considering arbitrary (positive and negative) prices. Indeed, all previous works on PTGs with only one clock (except [14]) have considered non-negative weights only, and the status of the more general case with arbitrary weights has so far remained elusive. Yet, arbitrary weights are an important modeling feature. Consider, for instance, a system which can consume but also produce energy at different rates. In this case, energy consumption could be modeled as a positive price-rate, and production by a negative price-rate. We propose an *exponential time algorithm to compute the value of one-clock* SPTG*s with arbitrary weights*. While this result might sound limited due to the restricted class of simple PTGs we can handle, we recall that the previous works mentioned above [10, 18, 16] have demonstrated that solving SPTGs is a key result towards solving more general PTGs. Moreover, this algorithm is, as far as we know, the first to handle the full class of SPTGs with arbitrary weights, and we note that the solutions (either the algorithms or the proofs) known so far do not generalise to this case. Finally, as a side result, this algorithm allows us to solve the more general class of *reset-acyclic* one-clock PTGs that we introduce. Thus, although we can not (yet) solve the whole class of PTGs with arbitrary weights, our result may be seen as a potentially important milestone towards this goal.

Due to lack of space most proofs and technical details may be found in a detailed version [12].

2 Priced timed games: syntax, semantics, and preliminary results

Notations and definitions. Let x denote a positive real-valued variable called *clock*. A *guard* (or *clock constraint*) is an interval with endpoints in $\mathbf{N} \cup \{+\infty\}$. We often abbreviate guards, for instance $x \leqslant 5$ instead of $[0,5]$. Let $S \subseteq \mathsf{Guard}(x)$ be a finite set of guards. We let $[\![S]\!] = \bigcup_{I \in S} I$. Assuming $M_0 = 0 < M_1 < \cdots < M_k$ are all the endpoints of the intervals in S (to which we add 0), we let $\mathsf{Reg}_S = \{(M_i, M_{i+1}) \mid 0 \leqslant i \leqslant k-1\} \cup \{\{M_i\} \mid 0 \leqslant i \leqslant k\}$ be the set of *regions* of S. Observe that Reg_S is also a set of guards.

We rely on the notion of *cost function* to formalise the notion of optimal value function sketched in the introduction. Formally, for a set of guards $S \subseteq \mathsf{Guard}(x)$, a *cost function* over S is a function $f\colon [\![\mathsf{Reg}_S]\!] \to \overline{\mathbf{R}} = \mathbf{R} \cup \{+\infty, -\infty\}$ such that over all regions $r \in \mathsf{Reg}_S$, f is either infinite or a continuous piecewise affine function, with a finite set of cutpoints (points where the first derivative is not defined) $\{\kappa_1, \ldots, \kappa_p\} \subseteq \mathbf{Q}$, and with $f(\kappa_i) \in \mathbf{Q}$ for all $1 \leqslant i \leqslant p$. In particular, if $f(r) = \{f(\nu) \mid \nu \in r\}$ contains $+\infty$ (respectively, $-\infty$) for some region r, then $f(r) = \{+\infty\}$ ($f(r) = \{-\infty\}$). We denote by CF_S the set of all cost functions over S. In our algorithm to solve SPTGs, we will need to combine cost functions thanks to the $\triangleright$ operator. Let $f \in \mathsf{CF}_S$ and $f' \in \mathsf{CF}_{S'}$ be two costs functions on set of guards $S, S' \subseteq \mathsf{Guard}(x)$, such that $[\![S]\!] \cap [\![S']\!]$ is a singleton. We let $f \triangleright f'$ be the cost function in $\mathsf{CF}_{S \cup S'}$ such that $(f \triangleright f')(\nu) = f(\nu)$ for all $\nu \in [\![\mathsf{Reg}_S]\!]$, and $(f \triangleright f')(\nu) = f'(\nu)$ for all $\nu \in [\![\mathsf{Reg}_{S'}]\!] \setminus [\![\mathsf{Reg}_S]\!]$.

We consider an extended notion of one-clock priced timed games (PTGs for short) allowing for the use of *urgent locations*, where only a zero delay can be spent, and *final cost functions* which are associated with each final location and incur an extra cost to be paid when ending the game in this location. Formally, a PTG $\mathcal{G}$ is a tuple $(L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \boldsymbol{\varphi}, \Delta, \pi)$ where L_{Min} (respectively, L_{Max}) is a finite set of *locations* for player Min (respectively, Max), with $L_{\mathsf{Min}} \cap L_{\mathsf{Max}} = \emptyset$; L_f is a finite set of *final* locations, and we let $L = L_{\mathsf{Min}} \cup L_{\mathsf{Max}} \cup L_f$ be the whole location space; $L_u \subseteq L \setminus L_f$ indicates *urgent* locations[1]; $\Delta \subseteq (L \setminus L_f) \times \mathsf{Guard}(x) \times \{\top, \bot\} \times L$ is a finite set of *transitions*; $\boldsymbol{\varphi} = (\varphi_\ell)_{\ell \in L_f}$ associates to each $\ell \in L_f$ its *final cost function*, that is an affine[2] cost function φ_ℓ over $S_{\mathcal{G}} = \{I \mid \exists \ell, R, \ell' : (\ell, I, R, \ell') \in \Delta\}$; $\pi\colon L \cup \Delta \to \mathbf{Z}$ mapping an integer *price* to each location—its *price-rate*—and transition.

Intuitively, a transition (ℓ, I, R, ℓ') changes the current location from ℓ to ℓ' if the clock has value in I and the clock is reset according to the Boolean R. We assume that, in all PTGs, the clock x is *bounded*, i.e., there is $M \in \mathbf{N}$ such that for all guards $I \in S_{\mathcal{G}}$, $I \subseteq [0, M]$.[3] We denote by $\mathsf{Reg}_{\mathcal{G}}$ the set $\mathsf{Reg}_{S_{\mathcal{G}}}$ of *regions of* $\mathcal{G}$. We further denote[4] by $\Pi^{\mathrm{tr}}_{\mathcal{G}}$, $\Pi^{\mathrm{loc}}_{\mathcal{G}}$ and $\Pi^{\mathrm{fin}}_{\mathcal{G}}$ respectively the values $\max_{\delta \in \Delta} |\pi(\delta)|$, $\max_{\ell \in L} |\pi(\ell)|$ and $\sup_{\nu \in [0,M]} \max_{\ell \in L} |\varphi_\ell(\nu)| = \max_{\ell \in L} \max(|\varphi_\ell(0)|, |\varphi_\ell(M)|)$. That is, $\Pi^{\mathrm{tr}}_{\mathcal{G}}$, $\Pi^{\mathrm{loc}}_{\mathcal{G}}$ and $\Pi^{\mathrm{fin}}_{\mathcal{G}}$ are the largest absolute values of the location prices, transition prices and final cost functions.

Let $\mathcal{G} = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \boldsymbol{\varphi}, \Delta, \pi)$ be a PTG. A *configuration* of $\mathcal{G}$ is a pair $s = (\ell, \nu) \in L \times \mathbf{R}^+$. We denote by $\mathsf{Conf}_{\mathcal{G}}$ the set of configurations of $\mathcal{G}$. Let (ℓ, ν) and (ℓ', ν') be two configurations. Let $\delta = (\ell, I, R, \ell') \in \Delta$ be a transition of $\mathcal{G}$ and $t \in \mathbf{R}^+$ be a delay. Then,

---

[1] Here we differ from [10] where $L_u \subseteq L_{\mathsf{Max}}$.

[2] The affine restriction on final cost function is to simplify our further arguments, though we do believe that all of our results could be adapted to cope with general cost functions.

[3] Observe that this last restriction is *not* without loss of generality in the case of PTGs. While all timed automata $\mathcal{A}$ can be turned into an equivalent (with respect to reachability properties) $\mathcal{A}'$ whose clocks are bounded [4], this technique can not be applied to PTGs, in particular with arbitrary prices.

[4] Throughout the paper, we often drop the $\mathcal{G}$ in the subscript of several notations when the game is clear from the context.

there is a (t,δ)-transition from (ℓ,ν) to (ℓ',ν') with cost c, denoted by $(\ell,\nu) \xrightarrow{t,\delta,c} (\ell',\nu')$, if (i) $\ell \in L_u$ implies $t = 0$; (ii) $\nu + t \in I$; (iii) $R = \top$ implies $\nu' = 0$; (iv) $R = \bot$ implies $\nu' = \nu + t$; (v) $c = \pi(\delta) + t \times \pi(\ell)$. Observe that the cost of (t,δ) takes into account the price-rate of ℓ, the delay spent in ℓ, and the price of δ. We assume that the game has no deadlock: for all $s \in \mathsf{Conf}_\mathcal{G}$, there are (t,δ,c) and $s' \in \mathsf{Conf}_\mathcal{G}$ such that $s \xrightarrow{t,\delta,c} s'$. Finally, we write $s \xrightarrow{c} s'$ whenever there are t and δ such that $s \xrightarrow{t,\delta,c} s'$. A *play* of $\mathcal{G}$ is a finite or infinite path $\rho = (\ell_0,\nu_0) \xrightarrow{c_0} (\ell_1,\nu_1) \xrightarrow{c_1} (\ell_2,\nu_2) \cdots$. For a finite play $\rho = (\ell_0,\nu_0) \xrightarrow{c_0} (\ell_1,\nu_1) \xrightarrow{c_1} (\ell_2,\nu_2) \cdots \xrightarrow{c_{n-1}} (\ell_n,\nu_n)$, we let $|\rho| = n$. For an infinite play $\rho = (\ell_0,\nu_0) \xrightarrow{c_0} (\ell_1,\nu_1) \xrightarrow{c_1} (\ell_2,\nu_2) \cdots$, we let $|\rho|$ be the least position i such that $\ell_i \in L_f$ if such a position exists, and $|\rho| = +\infty$ otherwise. Then, we let $\mathsf{Cost}_\mathcal{G}(\rho)$ be the *cost* of ρ, with $\mathsf{Cost}_\mathcal{G}(\rho) = +\infty$ if $|\rho| = +\infty$, and $\mathsf{Cost}_\mathcal{G}(\rho) = \sum_{i=0}^{|\rho|-1} c_i + \varphi_{\ell_{|\rho|}}(\nu_{|\rho|})$ otherwise.

A *strategy* for player Min is a function σ_Min mapping every finite play ending in location of Min to a pair $(t,\delta) \in \mathbf{R}^+ \times \Delta$, indicating what Min should play. We also request that the strategy proposes only valid pairs (t,δ), i.e., that for all runs ρ ending in (ℓ,ν), $\sigma_\mathsf{Min}(\rho) = (t,(\ell,I,R,\ell'))$ implies that $\nu + t \in I$. Strategies σ_Max of player Max are defined accordingly. We let $\mathsf{Strat}_\mathsf{Min}(\mathcal{G})$ and $\mathsf{Strat}_\mathsf{Max}(\mathcal{G})$ be the sets of strategies of Min and Max, respectively. A pair of strategies $(\sigma_\mathsf{Min},\sigma_\mathsf{Max}) \in \mathsf{Strat}_\mathsf{Min}(\mathcal{G}) \times \mathsf{Strat}_\mathsf{Max}(\mathcal{G})$ is called a *profile of strategies*. Together with an initial configuration $s_0 = (\ell_0,\nu_0)$, it defines a unique play $\mathsf{Play}(s_0,\sigma_\mathsf{Min},\sigma_\mathsf{Max}) = s_0 \xrightarrow{c_0} s_1 \xrightarrow{c_1} s_2 \cdots s_k \xrightarrow{c_k} \cdots$ where for all $j \geqslant 0$, s_{j+1} is the unique configuration such that $s_j \xrightarrow{t_j,\delta_j,c_j} s_{j+1}$ with $(t_j,\delta_j) = \sigma_\mathsf{Min}(s_0 \xrightarrow{c_0} s_1 \cdots s_{j-1} \xrightarrow{c_{j-1}} s_j)$ if $\ell_j \in L_\mathsf{Min}$; and $(t_j,\delta_j) = \sigma_\mathsf{Max}(s_0 \xrightarrow{c_0} s_1 \cdots s_{j-1} \xrightarrow{c_{j-1}} s_j)$ if $\ell_j \in L_\mathsf{Max}$. We let $\mathsf{Play}(\sigma_\mathsf{Min})$ (respectively, $\mathsf{Play}(s_0,\sigma_\mathsf{Min})$) be the set of plays that conform with σ_Min (and start in s_0).

As sketched in the introduction, we consider optimal reachability-cost games on PTGs, where the aim of player Min is to reach a location of L_f while minimising the cost. To formalise this objective, we let the value of a strategy σ_Min for Min be the function $\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Min}} \colon \mathsf{Conf}_\mathcal{G} \to \overline{\mathbf{R}}$ such that for all $s \in \mathsf{Conf}_\mathcal{G}$: $\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Min}}(s) = \sup_{\sigma_\mathsf{Max} \in \mathsf{Strat}_\mathsf{Max}} \mathsf{Cost}(\mathsf{Play}(s,\sigma_\mathsf{Min},\sigma_\mathsf{Max}))$. Intuitively, $\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Min}}(s)$ is the largest value that Max can achieve when playing against strategy σ_Min of Min (it is thus a worst case from the point of view of Min). Symmetrically, for $\sigma_\mathsf{Max} \in \mathsf{Strat}_\mathsf{Max}$, $\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Max}}(s) = \inf_{\sigma_\mathsf{Min} \in \mathsf{Strat}_\mathsf{Min}} \mathsf{Cost}(\mathsf{Play}(s,\sigma_\mathsf{Min},\sigma_\mathsf{Max}))$, for all $s \in \mathsf{Conf}_\mathcal{G}$. Then, the *upper and lower values* of $\mathcal{G}$ are respectively the functions $\overline{\mathsf{Val}}_\mathcal{G} \colon \mathsf{Conf}_\mathcal{G} \to \overline{\mathbf{R}}$ and $\underline{\mathsf{Val}}_\mathcal{G} \colon \mathsf{Conf}_\mathcal{G} \to \overline{\mathbf{R}}$ where, for all $s \in \mathsf{Conf}_\mathcal{G}$, $\overline{\mathsf{Val}}_\mathcal{G}(s) = \inf_{\sigma_\mathsf{Min} \in \mathsf{Strat}_\mathsf{Min}} \mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Min}}(s)$ and $\underline{\mathsf{Val}}_\mathcal{G}(s) = \sup_{\sigma_\mathsf{Max} \in \mathsf{Strat}_\mathsf{Max}} \mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Max}}(s)$. We say that a game is *determined* if the lower and the upper values match for every configuration s, and in this case, we say that the optimal value $\mathsf{Val}_\mathcal{G}$ of the game $\mathcal{G}$ exists, defined by $\mathsf{Val}_\mathcal{G} = \underline{\mathsf{Val}}_\mathcal{G} = \overline{\mathsf{Val}}_\mathcal{G}$. A strategy σ_Min of Min is *optimal* (respectively, *ε-optimal*) if $\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Min}} = \overline{\mathsf{Val}}_\mathcal{G}$ ($\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Min}} \leqslant \overline{\mathsf{Val}}_\mathcal{G} + \varepsilon$), i.e., σ_Min ensures that the cost of the plays will be at most $\overline{\mathsf{Val}}_\mathcal{G}$ ($\overline{\mathsf{Val}}_\mathcal{G} + \varepsilon$). Symmetrically, a strategy σ_Max of Max is *optimal* (respectively, *ε-optimal*) if $\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Max}} = \underline{\mathsf{Val}}_\mathcal{G}$ ($\mathsf{Val}_\mathcal{G}^{\sigma_\mathsf{Max}} \geqslant \underline{\mathsf{Val}}_\mathcal{G} - \varepsilon$).

Properties of the value. Let us now prove useful preliminary properties of the value function of PTGs, that—as far as we know—had hitherto never been established. Using a general determinacy result by Gale and Stewart [15], we can show that PTGs (with one clock) are *determined*. Hence, the value function $\mathsf{Val}_\mathcal{G}$ exists for all PTG $\mathcal{G}$. We can further show that, for all locations ℓ, $\mathsf{Val}_\mathcal{G}(\ell)$ is a *piecewise continuous function* that might exhibit discontinuities *only on the borders of the regions* of $\mathsf{Reg}_\mathcal{G}$ (where $\mathsf{Val}_\mathcal{G}(\ell)$ is the function such that $\mathsf{Val}_\mathcal{G}(\ell)(\nu) = \mathsf{Val}_\mathcal{G}(\ell,\nu)$ for all $\nu \in \mathbf{R}^+$).

▶ **Theorem 1.** *For all (one-clock)* PTG*s $\mathcal{G}$: (i) $\overline{\mathsf{Val}}_\mathcal{G} = \underline{\mathsf{Val}}_\mathcal{G}$, i.e.,* PTG*s are* determined*; and (ii) for all $r \in \mathsf{Reg}_\mathcal{G}$, for all $\ell \in L$, $\mathsf{Val}_\mathcal{G}(\ell)$ is either infinite or continuous over r.*

Simple priced timed games. As sketched in the introduction, our main contribution is to solve the special case of simple one-clock priced timed games with arbitrary costs. Formally, an r-SPTG, with $r \in \mathbf{Q}^+ \cap [0,1]$, is a PTG $\mathcal{G} = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \boldsymbol{\varphi}, \Delta, \pi)$ such that for all transitions $(\ell, I, R, \ell') \in \Delta$, $I = [0, r]$ and $R = \bot$. Hence, transitions of r-SPTGs are henceforth denoted by (ℓ, ℓ'), dropping the guard and the reset. Then, an SPTG is a 1-SPTG. This paper is devoted mainly to proving the following theorem on SPTGs:

▶ **Theorem 2.** *Let $\mathcal{G}$ be an* SPTG. *Then, for all locations $\ell \in L$, the function $\mathsf{Val}_{\mathcal{G}}(\ell)$ is either infinite, or continuous and piecewise-affine with at most an exponential number of cutpoints. The value functions for all locations, as well as a pair of optimal strategies $(\sigma_{\mathsf{Min}}, \sigma_{\mathsf{Max}})$ (that always exist if no values are infinite) can be computed in exponential time.*

Before sketching the proof of this theorem, we discuss a class of (simple) strategies that are sufficient to play optimally. Roughly speaking, Max has always a *memoryless* optimal strategy, while Min might need *(finite) memory* to play optimally—it is already the case in untimed quantitative reachability games with arbitrary weights [13]. Moreover, these strategies are finitely representable (recall that even a memoryless strategy depends on the current *configuration* and that there are infinitely many in our time setting).

We formalise Max's strategies with the notion of *finite positional strategies* (FP-strategies): they are memoryless strategies σ (i.e., for all finite plays $\rho_1 = \rho_1' \xrightarrow{c_1} s$ and $\rho_2 = \rho_2' \xrightarrow{c_2} s$ ending in the same configuration, we have $\sigma(\rho_1) = \sigma(\rho_2)$), such that for all locations ℓ, there exists a finite sequence of rationals $0 \leqslant \nu_1^\ell < \nu_2^\ell < \cdots < \nu_k^\ell = 1$ and a finite sequence of transitions $\delta_1, \ldots, \delta_k \in \Delta$ such that (i) for all $1 \leqslant i \leqslant k$, for all $\nu \in (\nu_{i-1}^\ell, \nu_i^\ell]$, either $\sigma(\ell, \nu) = (0, \delta_i)$, or $\sigma(q, \nu) = (\nu_i^\ell - \nu, \delta_i)$ (assuming $\nu_0^\ell = \min(0, \nu_1^\ell)$); and (ii) if $\nu_1^\ell > 0$, then $\sigma(\ell, 0) = (\nu_1^\ell, \delta_1)$. We let $\mathsf{pts}(\sigma)$ be the set of ν_i^ℓ for all ℓ and i, and $\mathsf{int}(\sigma)$ be the set of all successive intervals generated by $\mathsf{pts}(\sigma)$. Finally, we let $|\sigma| = |\mathsf{int}(\sigma)|$ be the size of σ. Intuitively, in an interval $(\nu_{i-1}^\ell, \nu_i^\ell]$, σ always returns the same move: either to take *immediately* δ_i or to wait until the clock reaches the endpoint ν_i^ℓ and then take δ_i.

Min, however may require memory to play optimally. Informally, we will compute optimal *switching* strategies, as introduced in [13] (in the untimed setting). A switching strategy is described by a pair $(\sigma^1_{\mathsf{Min}}, \sigma^2_{\mathsf{Min}})$ of FP-strategies and a switch threshold K, and consists in playing σ^1_{Min} until the total accumulated cost of the discrete transitions is below K; and then to *switch* to strategy σ^2_{Min}. The role of σ^2_{Min} is to ensure reaching a final location: it is thus a (classical) attractor strategy. The role of σ^1_{Min}, on the other hand, is to allow Min to decrease the cost low enough (possibly by forcing negative cycles) to secure a cost below K, and the computation of σ^1_{Min} is thus the critical point in the computation of an optimal switching strategy. To characterise σ^1_{Min}, we introduce the notion of negative cycle strategy (NC-strategy). Formally, an NC-strategy σ_{Min} of Min is an FP-strategy such that for all runs $\rho = (\ell_1, \nu) \xrightarrow{c_1} \cdots \xrightarrow{c_{k-1}} (\ell_k, \nu') \in \mathsf{Play}(\sigma_{\mathsf{Min}})$ with $\ell_1 = \ell_k$, and ν, ν' in the same interval of $\mathsf{int}(\sigma_{\mathsf{Min}})$, the sum of prices of *discrete transitions* is at most -1, i.e., $\pi(\ell_1, \ell_2) + \cdots + \pi(\ell_{k-1}, \ell_k) \leqslant -1$. To characterise the fact that σ^1_{Min} must allow Min to reach a cost which is *small enough, without necessarily reaching a target state*, we define the *fake value* of an NC-strategy σ_{Min} from a configuration s as $\mathsf{fake}^{\sigma_{\mathsf{Min}}}_{\mathcal{G}}(s) = \sup\{\mathsf{Cost}(\rho) \mid \rho \in \mathsf{Play}(s, \sigma_{\mathsf{Min}}), \rho \text{ reaches a target}\}$, i.e., the value obtained when *ignoring* the σ_{Min}-induced plays that *do not* reach the target. Thus, clearly, $\mathsf{fake}^{\sigma_{\mathsf{Min}}}_{\mathcal{G}}(s) \leqslant \mathsf{Val}^{\sigma_{\mathsf{Min}}}(s)$. We say that an NC-strategy is *fake-optimal* if its fake value, in every configuration, is equal to the optimal value of the configuration in the game. This is justified by the following result whose proof relies on the switching strategies described before:

Algorithm 1: `solveInstant`$(\mathcal{G},\nu)$

Input: r-SPTG $\mathcal{G} = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \varphi, \Delta, \pi)$, a valuation $\nu \in [0, r]$

```
foreach ℓ ∈ L do
    if ℓ ∈ L_f then X(ℓ) := φ_ℓ(ν) else X(ℓ) := +∞
repeat
    X_pre := X
    foreach ℓ ∈ L_Max do X(ℓ) := max_{(ℓ,ℓ')∈Δ} (π(ℓ,ℓ') + X_pre(ℓ'))
    foreach ℓ ∈ L_Min do X(ℓ) := min_{(ℓ,ℓ')∈Δ} (π(ℓ,ℓ') + X_pre(ℓ'))
    foreach ℓ ∈ L such that X(ℓ) < −(|L| − 1)Π^tr − Π^fin do X(ℓ) := −∞
until X = X_pre
return X
```

▶ **Lemma 3.** *If* $\mathsf{Val}_{\mathcal{G}}(\ell,\nu) \neq +\infty$*, for all* ℓ *and* ν*, then for all NC-strategies* σ_{Min}*, there is a strategy* σ'_{Min} *such that* $\mathsf{Val}_{\mathcal{G}}^{\sigma'_{\mathsf{Min}}}(s) \leqslant \mathsf{fake}_{\mathcal{G}}^{\sigma_{\mathsf{Min}}}(s)$ *for all configurations* s*. In particular, if* σ_{Min} *is a fake-optimal NC-strategy, then* σ'_{Min} *is an optimal (switching) strategy of the* SPTG*.*

Then, an SPTG is called *finitely optimal* if (*i*) Min has a fake-optimal NC-strategy; (*ii*) Max has an optimal FP-strategy; and (*iii*) $\mathsf{Val}_{\mathcal{G}}(\ell)$ is a cost function, for all locations ℓ. The central point in establishing Theorem 2 will thus be to prove that **all SPTGs are finitely optimal**, as this guarantees the existence of well-behaved optimal strategies and value functions. We will also show that they can be computed in exponential time. The proof is by induction on the number of urgent locations of the SPTG. In Section 3, we address the base case of SPTGs with urgent locations only (where no time can elapse). Since these SPTGs are very close to the *untimed* min-cost reachability games of [13], we adapt the algorithm in this work and obtain the `solveInstant` function (Algorithm 1). This function can also compute $\mathsf{Val}_{\mathcal{G}}(\ell, 1)$ for all ℓ and all games $\mathcal{G}$ (even with non-urgent locations) since time can not elapse anymore when the clock has valuation 1. Next, using the continuity result of Theorem 1, we can detect locations ℓ where $\mathsf{Val}_{\mathcal{G}}(\ell,\nu) \in \{+\infty, -\infty\}$, for all $\nu \in [0, 1]$, and remove them from the game. Finally, in Section 4 we handle SPTGs with non-urgent locations by refining the technique of [10, 18] (that work only on SPTGs with non-negative costs). Compared to [10, 18], our algorithm is simpler, being iterative, instead of recursive.

3 SPTGs with only urgent locations

Throughout this section, we consider an r-SPTG $\mathcal{G} = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \varphi, \Delta, \pi)$ where all locations are urgent, i.e., $L_u = L_{\mathsf{Min}} \cup L_{\mathsf{Max}}$. We first explain briefly how we can compute the value function of the game for a *fixed* clock valuation $\nu \in [0, r]$ (more precisely, we can compute the vector $(\mathsf{Val}_{\mathcal{G}}(\ell,\nu))_{\ell\in L}$). Since no time can elapse, we can adapt the techniques developed in [13] to solve (untimed) *min-cost reachability games*. The adaptation consists in taking into account the final cost functions. This yields the function `solveInstant` (Algorithm 1), that computes the vector $(\mathsf{Val}_{\mathcal{G}}(\ell,\nu))_{\ell\in L}$ for a fixed ν. The results of [13] also allow us to compute associated optimal strategies: when $\mathsf{Val}(\ell,\nu) \notin \{-\infty, +\infty\}$ the optimal strategy for Max is memoryless, and the optimal strategy for Min is a switching strategy $(\sigma^1_{\mathsf{Min}}, \sigma^2_{\mathsf{Min}})$ with a threshold K (as described in the previous section).

Now let us explain how we can reduce the computation of $\mathsf{Val}_{\mathcal{G}}(\ell)\colon \nu \in [0, r] \mapsto \mathsf{Val}(\ell,\nu)$ (for all ℓ) to a *finite number of calls* to `solveInstant`. Let $\mathsf{F}_{\mathcal{G}}$ be the set of affine functions over $[0, r]$ such that $\mathsf{F}_{\mathcal{G}} = \{k + \varphi_\ell \mid \ell \in L_f \wedge k \in \mathcal{I}\}$, where $\mathcal{I} = [-(|L| - 1)\Pi^{\mathrm{tr}}, |L|\Pi^{\mathrm{tr}}] \cap \mathbf{Z}$.

Observe that $\mathsf{F}_{\mathcal{G}}$ has cardinality $2|L|^2\Pi^{\mathrm{tr}}$, i.e., pseudo-polynomial in the size of $\mathcal{G}$. From [13], we conclude that the functions in $\mathsf{F}_{\mathcal{G}}$ are sufficient to characterise $\mathsf{Val}_{\mathcal{G}}$, in the following sense: for all $\ell \in L$ and $\nu \in [0,r]$ such that $\mathsf{Val}(\ell,\nu) \notin \{-\infty,+\infty\}$, there is $f \in \mathsf{F}_{\mathcal{G}}$ with $\mathsf{Val}(\ell,\nu) = f(\nu)$. Using the continuity of $\mathsf{Val}_{\mathcal{G}}$ (Theorem 1), we show that all the cutpoints of $\mathsf{Val}_{\mathcal{G}}$ are intersections of functions from $\mathsf{F}_{\mathcal{G}}$, i.e., belong to the set of *possible cutpoints* $\mathsf{PossCP}_{\mathcal{G}} = \{\nu \in [0,r] \mid \exists f_1, f_2 \in \mathsf{F}_{\mathcal{G}} \quad f_1 \neq f_2 \wedge f_1(\nu) = f_2(\nu)\}$. Observe that $\mathsf{PossCP}_{\mathcal{G}}$ contains at most $|\mathsf{F}_{\mathcal{G}}|^2 = 4|L_f|^4(\Pi^{\mathrm{tr}})^2$ points (also a pseudo-polynomial in the size of $\mathcal{G}$) since all functions in $\mathsf{F}_{\mathcal{G}}$ are affine, and can thus intersect at most once with every other function. Moreover, $\mathsf{PossCP}_{\mathcal{G}} \subseteq \mathbf{Q}$, since all functions of $\mathsf{F}_{\mathcal{G}}$ take rational values in 0 and $r \in \mathbf{Q}$. Thus, for all ℓ, $\mathsf{Val}_{\mathcal{G}}(\ell)$ is a cost function (with cutpoints in $\mathsf{PossCP}_{\mathcal{G}}$ and pieces from $\mathsf{F}_{\mathcal{G}}$). Since $\mathsf{Val}_{\mathcal{G}}(\ell)$ is a piecewise affine function, we can characterise it completely by computing only its value on its cutpoints. Hence, we can reconstruct $\mathsf{Val}_{\mathcal{G}}(\ell)$ by calling `solveInstant` on each rational valuation $\nu \in \mathsf{PossCP}_{\mathcal{G}}$. From the optimal strategies computed along `solveInstant` [13], we can also reconstruct a fake-optimal NC-strategy for Min and an optimal FP-strategy for Max, hence:

▶ **Proposition 4.** *Every r-SPTG $\mathcal{G}$ with only urgent locations is finitely optimal. Moreover, for all locations ℓ, the piecewise affine function $\mathsf{Val}_{\mathcal{G}}(\ell)$ has cutpoints in $\mathsf{PossCP}_{\mathcal{G}}$ of cardinality $4|L_f|^4(\Pi^{\mathrm{tr}})^2$, pseudo-polynomial in the size of $\mathcal{G}$.*

4 Solving general SPTGs

In this section, we consider SPTGs with possibly non-urgent locations. We first prove that all such SPTGs are finitely optimal. Then, we introduce Algorithm 2 to compute optimal values and strategies of SPTGs. To the best of our knowledge, this is the first algorithm to solve SPTGs with arbitrary weights. Throughout the section, we fix an SPTG $\mathcal{G} = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \varphi, \Delta, \pi)$ with possibly non-urgent locations. Before presenting our core contributions, let us explain how we can detect locations with infinite values. As already argued, we can compute $\mathsf{Val}(\ell,1)$ for all ℓ assuming all locations are urgent, since time can not elapse anymore when the clock has valuation 1. This can be done with `solveInstant`. Then, by continuity, $\mathsf{Val}(\ell,1) = +\infty$ (respectively, $\mathsf{Val}(\ell,1) = -\infty$), for some ℓ if and only if $\mathsf{Val}(\ell,\nu) = +\infty$ (respectively, $\mathsf{Val}(\ell,\nu) = -\infty$) for all $\nu \in [0,1]$. We remove from the game all locations with infinite value without changing the values of other locations (as justified in [13]). Thus, we henceforth assume that $\mathsf{Val}(\ell,\nu) \in \mathbf{R}$ for all (ℓ,ν).

The $\mathcal{G}_{L',r}$ construction. To prove finite optimality of SPTGs and to establish correctness of our algorithm, we rely in both cases on a construction that consists in decomposing $\mathcal{G}$ into a sequence of SPTGs with *more urgent locations*. Intuitively, a game with more urgent locations is easier to solve since it is closer to an untimed game (in particular, when all locations are urgent, we can apply the techniques of Section 3). More precisely, given a set L' of non-urgent locations, and a valuation $r_0 \in [0,1]$, we will define a (possibly infinite) sequence of valuations $1 = r_0 > r_1 > \cdots$ and a sequence $\mathcal{G}_{L',r_0}, \mathcal{G}_{L',r_1}, \ldots$ of SPTGs such that (i) all locations of $\mathcal{G}$ are also present in each $\mathcal{G}_{L',r_i}$, except that the locations of L' are now urgent; and (ii) for all $i \geqslant 0$, the value function of $\mathcal{G}_{L',r_i}$ is equal to $\mathsf{Val}_{\mathcal{G}}$ on the interval $[r_{i+1}, r_i]$. Hence, we can re-construct $\mathsf{Val}_{\mathcal{G}}$ by assembling well-chosen parts of the values functions of the $\mathcal{G}_{L',r_i}$ (assuming $\inf_i r_i = 0$). This basic result will be exploited in two directions. First, we prove by induction on the number of urgent locations that all SPTGs are finitely optimal, by re-constructing $\mathsf{Val}_{\mathcal{G}}$ (as well as optimal strategies) as a $\triangleright$-concatenation of the value functions of a finite sequence of SPTGs with one more urgent locations. The

base case, with only urgent locations, is solved by Proposition 4. This construction suggests a *recursive* algorithm in the spirit of [10, 18] (for non-negative prices). Second, we show that this recursion can be *avoided* (see Algorithm 2). Instead of turning locations urgent one at a time, this algorithm makes them all urgent and computes directly the sequence of SPTGs with only urgent locations. Its proof of correctness relies on the finite optimality of SPTGs and, again, on our basic result linking the values functions of $\mathcal{G}$ and games $\mathcal{G}_{L',r_i}$.

Let us formalise these constructions. Let $\mathcal{G}$ be an SPTG, let $r \in [0,1]$ be an endpoint, and let $\boldsymbol{x} = (x_\ell)_{\ell \in L}$ be a vector of rational values. Then, $\mathtt{wait}(\mathcal{G}, r, \boldsymbol{x})$ is an r-SPTG in which both players may now decide, in all non-urgent locations ℓ, to *wait* until the clock takes value r, and then to stop the game, adding the cost x_ℓ to the current cost of the play. Formally, $\mathtt{wait}(\mathcal{G}, r, \boldsymbol{x}) = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L'_f, L_u, \boldsymbol{\varphi}', T', \pi')$ is such that $L'_f = L_f \uplus \{\ell^f \mid \ell \in L \setminus L_u\}$; for all $\ell' \in L_f$ and $\nu \in [0,r]$, $\varphi'_{\ell'}(\nu) = \varphi_{\ell'}(\nu)$, for all $\ell \in L \setminus L_u$, $\varphi'_{\ell f}(\nu) = (r-\nu)\cdot\pi(\ell) + x_\ell$; $T' = T \cup \{(\ell, [0,r], \bot, \ell^f) \mid \ell \in L \setminus L_u\}$; for all $\delta \in T'$, $\pi'(\delta) = \pi(\delta)$ if $\delta \in T$, and $\pi'(\delta) = 0$ otherwise. Then, we let $\mathcal{G}_r = \mathtt{wait}\big(\mathcal{G}, r, (\mathsf{Val}_{\mathcal{G}}(\ell, r))_{\ell \in L}\big)$, i.e., the game obtained thanks to $\mathtt{wait}$ by letting $\boldsymbol{x}$ be the value of $\mathcal{G}$ in r. One can check that this first transformation does not alter the value of the game, for valuations before r: $\mathsf{Val}_{\mathcal{G}}(\ell,\nu) = \mathsf{Val}_{\mathcal{G}_r}(\ell,\nu)$ for all $\nu \leqslant r$.

Next, we make locations urgent. For a set $L' \subseteq L \setminus L_u$ of non-urgent locations, we let $\mathcal{G}_{L',r}$ be the SPTG obtained from $\mathcal{G}_r$ by making urgent every location ℓ of L'. Observe that, although all locations $\ell \in L'$ are now urgent in $\mathcal{G}_{L',r}$, their clones ℓ^f allow the players to wait until r. When L' is a singleton $\{\ell\}$, we write $\mathcal{G}_{\ell,r}$ instead of $\mathcal{G}_{\{\ell\},r}$. While the construction of $\mathcal{G}_r$ does not change the value of the game, introducing urgent locations *does*. Yet, we can characterise an interval $[a,r]$ on which the value functions of $\mathcal{H} = \mathcal{G}_{L',r}$ and $\mathcal{H}^+ = \mathcal{G}_{L'\cup\{\ell\},r}$ coincide, as stated by the next proposition. The interval $[a,r]$ depends on the *slopes* of the pieces of $\mathsf{Val}_{\mathcal{H}^+}$ as depicted in Figure 2: for each location ℓ of Min, the slopes of the pieces of $\mathsf{Val}_{\mathcal{H}^+}$ contained in $[a,r]$ should be $\leqslant -\pi(\ell)$ (and $\geqslant -\pi(\ell)$ when ℓ belongs to Max). It is proved by lifting optimal strategies of $\mathcal{H}^+$ into $\mathcal{H}$, and strongly relies on the determinacy result of Theorem 1:

▶ **Proposition 5.** *Let $0 \leqslant a < r \leqslant 1$, $L' \subseteq L \setminus L_u$ and $\ell \notin L' \cup L_u$ a non-urgent location of* Min *(respectively,* Max*). Assume that $\mathcal{G}_{L'\cup\{\ell\},r}$ is finitely optimal, and for all $a \leqslant \nu_1 < \nu_2 \leqslant r$*

$$\frac{\mathsf{Val}_{\mathcal{G}_{L'\cup\{\ell\},r}}(\ell,\nu_2) - \mathsf{Val}_{\mathcal{G}_{L'\cup\{\ell\},r}}(\ell,\nu_1)}{\nu_2 - \nu_1} \geqslant -\pi(\ell) \quad (\textit{respectively, } \leqslant -\pi(\ell))\,. \tag{1}$$

Then, for all $\nu \in [a,r]$ and $\ell' \in L$, $\mathsf{Val}_{\mathcal{G}_{L'\cup\{\ell\},r}}(\ell',\nu) = \mathsf{Val}_{\mathcal{G}_{L',r}}(\ell',\nu)$. Furthermore, fake-optimal NC-strategies and optimal FP-strategies in $\mathcal{G}_{L'\cup\{\ell\},r}$ are also fake-optimal and optimal over $[a,r]$ in $\mathcal{G}_{L',r}$.

Given an SPTG $\mathcal{G}$ and some *finitely optimal* $\mathcal{G}_{L',r}$, we now characterise precisely the left endpoint of the maximal interval ending in r where the value functions of $\mathcal{G}$ and $\mathcal{G}_{L',r}$ coincide, with the operator $\mathsf{left}_{L'} : (0,1] \to [0,1]$ (or simply left, if L' is clear) defined as:

$$\mathsf{left}_{L'}(r) = \sup\{r' \leqslant r \mid \forall \ell \in L\ \forall \nu \in [r',r]\ \mathsf{Val}_{\mathcal{G}_{L',r}}(\ell,\nu) = \mathsf{Val}_{\mathcal{G}}(\ell,\nu)\}\,.$$

By continuity of the value (Theorem 1), this supremum exists and $\mathsf{Val}_{\mathcal{G}}(\ell, \mathsf{left}_{L'}(r)) = \mathsf{Val}_{\mathcal{G}_{L',r}}(\ell, \mathsf{left}_{L'}(r))$. Moreover, $\mathsf{Val}_{\mathcal{G}}(\ell)$ is a cost function on $[\mathsf{left}(r), r]$, since $\mathcal{G}_{L',r}$ is finitely optimal. However, this definition of $\mathsf{left}(r)$ is semantical. Yet, building on the ideas of Proposition 5, we can effectively compute $\mathsf{left}(r)$, given $\mathsf{Val}_{\mathcal{G}_{L',r}}$. We claim that $\mathsf{left}_{L'}(r)$ is the *minimal valuation* such that for all locations $\ell \in L' \cap L_{\mathsf{Min}}$ (respectively, $\ell \in L' \cap L_{\mathsf{Max}}$), the slopes of the affine sections of the cost function $\mathsf{Val}_{\mathcal{G}_{L',r}}(\ell)$ on $[\mathsf{left}(r), r]$ are at least (at most) $-\pi(\ell)$. Hence, $\mathsf{left}(r)$ can be obtained (see Figure 3), by inspecting iteratively, for

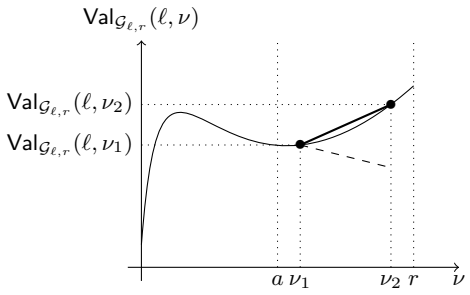

Figure 2 The condition (1) (in the case $L' = \emptyset$ and $\ell \in L_{\mathsf{Min}}$): graphically, it means that the slope between every two points of the plot in $[a, r]$ (represented with a thick line) is greater than or equal to $-\pi(\ell)$ (represented with dashed line).

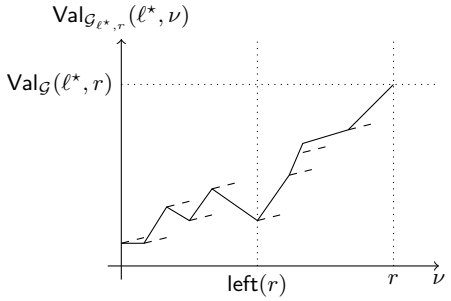

Figure 3 In this example $L' = \{\ell^\star\}$ and $\ell^\star \in L_{\mathsf{Min}}$. $\mathsf{left}(r)$ is the leftmost point such that all slopes on its right are smaller than or equal to $-\pi(\ell^\star)$ in the graph of $\mathsf{Val}_{\mathcal{G}_{\ell^\star,r}}(\ell^\star,\nu)$. Dashed lines have slope $-\pi(\ell^\star)$.

all ℓ of Min (respectively, Max), the slopes of $\mathsf{Val}_{\mathcal{G}_{L',r}}(\ell)$, by decreasing valuations, until we find a piece with a slope $> -\pi(\ell)$ (respectively, $< -\pi(\ell)$). This enumeration of the slopes is effective as $\mathsf{Val}_{\mathcal{G}_{L',r}}$ has finitely many pieces, by hypothesis. Moreover, this guarantees that $\mathsf{left}(r) < r$. Thus, one can reconstruct $\mathsf{Val}_{\mathcal{G}}$ on $[\inf_i r_i, r_0]$ from the value functions of the (potentially infinite) sequence of games $\mathcal{G}_{L',r_0}$, $\mathcal{G}_{L',r_1}, \dots$ where $r_{i+1} = \mathsf{left}(r_i)$ for all i such that $r_i > 0$, for all possible choices of non-urgent locations L'. Next, we will define two different ways of choosing L': the former to prove finite optimality of all SPTGs, the latter to obtain an algorithm to solve them.

SPTGs are finitely optimal. To prove finite optimality of all SPTGs we reason by induction on the number of non-urgent locations and instantiate the previous results to the case where $L' = \{\ell^\star\}$ where $\ell^\star$ is a non-urgent location of *minimum price-rate* (i.e., for all $\ell \in L$, $\pi(\ell^\star) \leqslant \pi(\ell)$). Given $r_0 \in [0, 1]$, we let $r_0 > r_1 > \cdots$ be the decreasing sequence of valuations such that $r_i = \mathsf{left}_{\ell^\star}(r_{i-1})$ for all $i > 0$. As explained before, we will build $\mathsf{Val}_{\mathcal{G}}$ on $[\inf_i r_i, r_0]$ from the value functions of games $\mathcal{G}_{\ell^\star,r_i}$. Assuming finite optimality of those games, this will prove that $\mathcal{G}$ is finitely optimal *under the condition* that $r_0 > r_1 > \cdots$ eventually stops, i.e., $r_i = 0$ for some i. This property is given by the next lemma, which ensures that, for all i, the owner of $\ell^\star$ has a strictly better strategy in configuration $(\ell^\star, r_{i+1})$ than waiting until r_i in location $\ell^\star$.

▶ **Lemma 6.** *If $\mathcal{G}_{\ell^\star,r_i}$ is finitely optimal for all $i \geqslant 0$, then (i) if $\ell^\star \in L_{\mathsf{Min}}$ (respectively, L_{Max}), $\mathsf{Val}_{\mathcal{G}}(\ell^\star, r_{i+1}) < \mathsf{Val}_{\mathcal{G}}(\ell^\star, r_i) + (r_i - r_{i+1})\pi(\ell^\star)$ (respectively, $\mathsf{Val}_{\mathcal{G}}(\ell^\star, r_{i+1}) > \mathsf{Val}_{\mathcal{G}}(\ell^\star, r_i) + (r_i - r_{i+1})\pi(\ell^\star)$), for all i; and (ii) there is $i \leqslant |\mathsf{F}_{\mathcal{G}}|^2 + 2$ such that $r_i = 0$.*

By iterating this construction, we make all locations urgent iteratively, and obtain:

▶ **Proposition 7.** *Every SPTG $\mathcal{G}$ is finitely optimal and for all locations ℓ, $\mathsf{Val}_{\mathcal{G}}(\ell)$ has at most $O\left((\Pi^{\mathrm{tr}}|L|^2)^{2|L|+2}\right)$ cutpoints.*

Proof. As announced, we show by induction on $n \geqslant 0$ that every r-SPTG $\mathcal{G}$ with n non-urgent locations is finitely optimal, and that the number of cutpoints of $\mathsf{Val}_{\mathcal{G}}(\ell)$ is at most $O\left((\Pi^{\mathrm{tr}}(|L_f| + n^2))^{2n+2}\right)$, which suffices to show the above bound, since $|L_f| + n^2 \leqslant |L|^2$.

The base case $n = 0$ is given by Proposition 4. Now, assume that $\mathcal{G}$ has at least one non-urgent location, and consider $\ell^\star$ one with minimum price. By induction hypothesis, all r'-SPTGs $\mathcal{G}_{\ell^\star,r'}$ are finitely optimal for all $r' \in [0,r]$. Let $r_0 > r_1 > \cdots$ be the decreasing sequence defined by $r_0 = r$ and $r_i = \mathsf{left}_{\ell^\star}(r_{i-1})$ for all $i \geqslant 1$. By Lemma 6, there exists $j \leqslant |\mathsf{F}_\mathcal{G}|^2 + 2$ such that $r_j = 0$. Moreover, for all $0 < i \leqslant j$, $\mathsf{Val}_\mathcal{G} = \mathsf{Val}_{\mathcal{G}_{\ell^\star,r_{i-1}}}$ on $[r_i, r_{i-1}]$ by definition of $r_i = \mathsf{left}_{\ell^\star}(r_{i-1})$, so that $\mathsf{Val}_\mathcal{G}(\ell)$ is a cost function on this interval, for all ℓ, and the number of cutpoints on this interval is bounded by $O\left((\Pi^{\mathrm{tr}}(|L_f| + (n-1)^2 + n))^{2(n-1)+2}\right) = O\left((\Pi^{\mathrm{tr}}(|L_f| + n^2))^{2(n-1)+2}\right)$ by induction hypothesis (notice that maximal transition prices are the same in $\mathcal{G}$ and $\mathcal{G}_{\ell^\star,r_{i-1}}$, but that we add n more final locations in $\mathcal{G}_{\ell^\star,r_{i-1}}$). Adding the cutpoint 1, summing over i from 0 to $j \leqslant |\mathsf{F}_\mathcal{G}|^2 + 2$, and observing that $|\mathsf{F}_\mathcal{G}| \leqslant 2\Pi^{\mathrm{tr}}|L_f|$, we bound the number of cutpoints of $\mathsf{Val}_\mathcal{G}(\ell)$ by $O\left((\Pi^{\mathrm{tr}}(|L_f| + n^2))^{2n+2}\right)$. Finally, we can reconstruct fake-optimal and optimal strategies in $\mathcal{G}$ from the from fake-optimal and optimal strategies of $\mathcal{G}_{\ell^\star,r_i}$. ◀

Computing the value functions. The finite optimality of SPTGs allows us to compute the value functions. The proof of Proposition 7 suggests a *recursive* algorithm to do so: from an SPTG $\mathcal{G}$ with minimal non-urgent location $\ell^\star$, solve recursively $\mathcal{G}_{\ell^\star,1}$, $\mathcal{G}_{\ell^\star,\mathsf{left}(1)}$, $\mathcal{G}_{\ell^\star,\mathsf{left}(\mathsf{left}(1))}$, *etc.* handling the base case where all locations are urgent with Algorithm 1. While our results above show that this is correct and terminates, we propose instead to solve—without the need for recursion—the sequence of games $\mathcal{G}_{L\setminus L_u,1}$, $\mathcal{G}_{L\setminus L_u,\mathsf{left}(1)}, \ldots$ i.e., *making all locations urgent at once*. Again, the arguments given above prove that this scheme is *correct*, but the key argument of Lemma 6 that ensures *termination* can not be applied in this case. Instead, we rely on the following lemma, stating, that there will be at least one cutpoint of $\mathsf{Val}_\mathcal{G}$ in each interval $[\mathsf{left}(r), r]$. Observe that this lemma relies on the fact that $\mathcal{G}$ is finitely optimal, hence the need to first prove this fact independently with the sequence $\mathcal{G}_{\ell^\star,1}$, $\mathcal{G}_{\ell^\star,\mathsf{left}(1)}$, $\mathcal{G}_{\ell^\star,\mathsf{left}(\mathsf{left}(1))}, \ldots$ Termination then follows from the fact that $\mathsf{Val}_\mathcal{G}$ has finitely many cutpoints by finite optimality.

▶ **Lemma 8.** *Let $r_0 \in (0,1]$ such that $\mathcal{G}_{L',r_0}$ is finitely optimal. Suppose that $r_1 = \mathsf{left}_{L'}(r_0) > 0$, and let $r_2 = \mathsf{left}_{L'}(r_1)$. There exists $r' \in [r_2, r_1)$ and $\ell \in L'$ such that (i) $\mathsf{Val}_\mathcal{G}(\ell)$ is affine on $[r', r_1]$, of slope equal to $-\pi(\ell)$, and (ii) $\mathsf{Val}_\mathcal{G}(\ell, r_1) \neq \mathsf{Val}_\mathcal{G}(\ell, r_0) + \pi(\ell)(r_0 - r_1)$. As a consequence, $\mathsf{Val}_\mathcal{G}(\ell)$ has a cutpoint in $[r_1, r_0)$.*

Algorithm 2 implements these ideas. Each iteration of the **while** loop computes a new game in the sequence $\mathcal{G}_{L\setminus L_u,1}$, $\mathcal{G}_{L\setminus L_u,\mathsf{left}(1)}, \ldots$ described above; solves it thanks to `solveInstant`; and thus computes a new portion of $\mathsf{Val}_\mathcal{G}$ on an interval on the left of the current point $r \in [0,1]$. More precisely, the vector $(\mathsf{Val}_\mathcal{G}(\ell, 1))_{\ell \in L}$ is first computed in line 1. Then, the algorithm enters the **while** loop, and the game $\mathcal{G}'$ obtained when reaching line 6 is $\mathcal{G}_{L\setminus L_u,1}$. Then, the algorithm enters the **repeat** loop to analyse this game. Instead of building the whole value function of $\mathcal{G}'$, Algorithm 2 builds only the parts of $\mathsf{Val}_{\mathcal{G}'}$ that coincide with $\mathsf{Val}_\mathcal{G}$. It proceeds by enumerating the possible cutpoints a of $\mathsf{Val}_{\mathcal{G}'}$, starting in r, by decreasing valuations (line 8), and computes the value of $\mathsf{Val}_{\mathcal{G}'}$ in each cutpoint thanks to `solveInstant` (line 9), which yields a new piece of $\mathsf{Val}_{\mathcal{G}'}$. Then, the **if** in line 10 checks whether this new piece coincides with $\mathsf{Val}_\mathcal{G}$, using the condition given by Proposition 5. If it is the case, the piece of $\mathsf{Val}_{\mathcal{G}'}$ is added to f_ℓ (line 11); **repeat** is stopped otherwise. When exiting the **repeat** loop, variable b has value $\mathsf{left}(1)$. Hence, at the next iteration of the **while** loop, $\mathcal{G}' = \mathcal{G}_{L\setminus L_u,\mathsf{left}(1)}$ when reaching line 6. By continuing this reasoning inductively, one concludes that the successive iterations of the **while** loop compute the sequence $\mathcal{G}_{L\setminus L_u,1}$, $\mathcal{G}_{L\setminus L_u,\mathsf{left}(1)}, \ldots$ as announced, and rebuilds $\mathsf{Val}_\mathcal{G}$ from them. Termination in exponential

Algorithm 2: $\texttt{solve}(\mathcal{G})$

Input: SPTG $\mathcal{G} = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L_f, L_u, \boldsymbol{\varphi}, \Delta, \pi)$

1 $\boldsymbol{f} = (f_\ell)_{\ell \in L} := \texttt{solveInstant}(\mathcal{G}, 1)$ /* $f_\ell \colon \{1\} \to \overline{\mathbf{R}}$ */
2 $r := 1$
3 **while** $0 < r$ **do** /* Invariant: $f_\ell \colon [r,1] \to \overline{\mathbf{R}}$ */
4 $\quad \mathcal{G}' := \texttt{wait}(\mathcal{G}, r, \boldsymbol{f}(r))$ /* r-SPTG $\mathcal{G}' = (L_{\mathsf{Min}}, L_{\mathsf{Max}}, L'_f, L'_u, \boldsymbol{\varphi}', T', \pi')$ */
5 $\quad L'_u := L'_u \cup L$ /* every location is made urgent */
6 $\quad b := r$
7 $\quad$ **repeat** /* Invariant: $f_\ell \colon [b,1] \to \overline{\mathbf{R}}$ */
8 $\quad\quad a := \max(\mathsf{PossCP}_{\mathcal{G}'} \cap [0,b))$
9 $\quad\quad \boldsymbol{x} = (x_\ell)_{\ell \in L} := \texttt{solveInstant}(\mathcal{G}', a)$ /* $x_\ell = \mathsf{Val}_{\mathcal{G}'}(\ell, a)$ */
10 $\quad\quad$ **if** $\forall \ell \in L_{\mathsf{Min}}\ \frac{f_\ell(b)-x_\ell}{b-a} \leqslant -\pi(\ell) \wedge \forall \ell \in L_{\mathsf{Max}}\ \frac{f_\ell(b)-x_\ell}{b-a} \geqslant -\pi(\ell)$ **then**
11 $\quad\quad\quad$ **foreach** $\ell \in L$ **do** $f_\ell := \left(\nu \in [a,b] \mapsto f_\ell(b) + (\nu - b)\frac{f_\ell(b)-x_\ell}{b-a}\right) \rhd f_\ell$
12 $\quad\quad\quad b := a\,;\ stop := false$
13 $\quad\quad$ **else** $stop := true$
14 $\quad$ **until** $b = 0$ *or* $stop$
15 $\quad r := b$
16 **return** $\boldsymbol{f}$

time is ensured by Lemma 8: each iteration of the **while** loop discovers at least one new cutpoint of $\mathsf{Val}_{\mathcal{G}}$, and there are at most exponentially many (note that a tighter bound on this number of cutpoints would entail a better complexity of our algorithm).

▶ **Example 9.** Let us briefly sketch the execution of Algorithm 2 on the SPTG in Figure 1. During the first iteration of the **while** loop, the algorithm computes the correct value functions until the cutpoint $\frac{3}{4}$: in the *repeat* loop, at first $a = 9/10$ but the slope in ℓ_1 is smaller than the slope that would be granted by waiting, as depicted in Figure 1. Then, $a = 3/4$ where the algorithm gives a slope of value -16 in ℓ_2 while the cost of this location of Max is -14. During the first iteration of the **while** loop, the inner **repeat** loop thus ends with $r = 3/4$. The next iterations of the **while** loop end with $r = \frac{1}{2}$ (because ℓ_1 does not pass the test in line 10); $r = \frac{1}{4}$ (because of ℓ_2) and finally with $r = 0$, giving us the value functions on the entire interval $[0,1]$.

5 Beyond SPTGs

In [10, 18, 16], *general* PTGs with *non-negative prices* are solved by reducing them to a finite sequence of SPTGs, by eliminating guards and resets. It is thus natural to try and adapt these techniques to our general case, in which case Algorithm 2 would allow us to solve *general* PTG*s with arbitrary costs*. Let us explain why it is not (completely) the case. The technique used to remove guards from PTGs consists in enhancing the locations with regions while keeping an equivalent game. This technique *can* be adapted to arbitrary weights.

The technique to handle resets, however, consists in *bounding* the number of clock resets that can occur in each play following an optimal strategy of Min or Max. Then, the PTG can be *unfolded* into a *reset-acyclic* PTG with the same value. By reset-acyclic, we mean that no cycle in the configuration graph visits a transition with a reset. This reset-acyclic PTG can be decomposed into a finite number of components that contain no reset and are

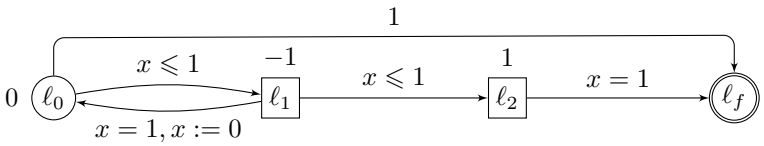

Figure 4 A PTG where the number of resets in optimal plays can not be bounded a priori.

linked by transitions with resets. These components can be solved iteratively, from the bottom to the top, turning them into SPTGs. Thus, if we *assume* that the PTGs we are given as input *are* reset-acyclic, we can solve them in *exponential time*, and show that their value functions are cost functions with at most exponentially many cutpoints, using our techniques. Unfortunately, the arguments to bound the number of resets do not hold for arbitrary costs, as shown by the PTG in Figure 4. We claim that $\mathsf{Val}(\ell_0) = 0$; that Min has no optimal strategy, but a family of ε-optimal strategies $\sigma^\varepsilon_{\mathsf{Min}}$ each with value ε; and that each $\sigma^\varepsilon_{\mathsf{Min}}$ requires *memory whose size depends on* ε and might *yield a play visiting at least* $1/\varepsilon$ *times the reset* between ℓ_0 and ℓ_1 (hence the number of resets can not be bounded). For all $\varepsilon > 0$, $\sigma^\varepsilon_{\mathsf{Min}}$ consists in: waiting $1-\varepsilon$ time units in ℓ_0, then going to ℓ_1 during the $\lceil 1/\varepsilon \rceil$ first visits to ℓ_0; and to go directly to ℓ_f afterwards. Against $\sigma^\varepsilon_{\mathsf{Min}}$, Max has two possible choices: (*i*) either wait 0 time unit in ℓ_1, wait ε time units in ℓ_2, then reach ℓ_f; or (*ii*) wait ε time unit in ℓ_1 then force the cycle by going back to ℓ_0 and wait for Min's next move. Thus, all plays according to $\sigma^\varepsilon_{\mathsf{Min}}$ will visit a sequence of locations which is either of the form $\ell_0(\ell_1\ell_0)^k\ell_1\ell_2\ell_f$, with $0 \leqslant k < \lceil 1/\varepsilon \rceil$; or of the form $\ell_0(\ell_1\ell_0)^{\lceil \frac{1}{\varepsilon} \rceil}\ell_f$. In the former case, the cost of the play will be $-k\varepsilon + 0 + \varepsilon = -(k-1)\varepsilon \leqslant \varepsilon$; in the latter, $-\varepsilon(\lceil 1/\varepsilon \rceil) + 1 \leqslant 0$. This shows that $\mathsf{Val}(\ell_0) = 0$, but there is no optimal strategy as none of these strategies allow one to guarantee a cost of 0 (neither does the strategy that waits 1 time unit in ℓ_0).

However, we may apply the result on reset-acyclic PTGs to obtain:

▶ **Theorem 10.** *The value functions of all one-clock* PTG*s are cost functions with at most exponentially many cutpoints.*

Proof. Let $\mathcal{G}$ be a one-clock PTG. Let us replace all transitions $(\ell, g, \top, \ell')$ resetting the clock by $(\ell, g, \bot, \ell'')$, where ℓ'' is a new final location with $\varphi_{\ell''} = \mathsf{Val}_{\mathcal{G}}(\ell, 0)$—observe that $\mathsf{Val}_{\mathcal{G}}(\ell, 0)$ exists even if we can not compute it, so this transformation is well-defined. This yields a reset-acyclic PTG $\mathcal{G}'$ such that $\mathsf{Val}_{\mathcal{G}'} = \mathsf{Val}_{\mathcal{G}}$. ◀

References

1 Rajeev Alur, Mikhail Bernadsky, and P. Madhusudan. Optimal reachability for weighted timed games. In *Proceedings of the 31st International Colloquium on Automata, Languages and Programming (ICALP'04)*, volume 3142 of *Lecture Notes in Computer Science*, pages 122–133. Springer, 2004.

2 Rajeev Alur and David L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

3 Rajeev Alur, Salvatore La Torre, and George J. Pappas. Optimal paths in weighted timed automata. *Theoretical Computer Science*, 318(3):297–322, 2004.

4 Gerd Behrmann, Ansgar Fehnker, Thomas Hune, Kim G. Larsen, Judi Romijn, and Frits W. Vaandrager. Minimum-cost reachability for priced timed automata. In *Proceedings of the 4th International Workshop on Hybrid Systems: Computation and Control (HSCC'01)*, volume 2034 of *Lecture Notes in Computer Science*, pages 147–161. Springer, 2001.

5 J. Berendsen, T. Chen, and D. Jansen. Undecidability of cost-bounded reachability in priced probabilistic timed automata. In *Theory and Applications of Models of Computation*, volume 5532 of *LNCS*, pages 128–137. Springer, 2009.

6 Patricia Bouyer, Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On the optimal reachability problem of weighted timed automata. *Formal Methods in System Design*, 31(2):135–175, 2007.

7 Patricia Bouyer, Thomas Brihaye, and Nicolas Markey. Improved undecidability results on weighted timed automata. *Information Processing Letters*, 98(5):188–194, 2006.

8 Patricia Bouyer, Franck Cassez, Emmanuel Fleury, and Kim G. Larsen. Optimal strategies in priced timed game automata. In *Proceedings of the 24th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'04)*, volume 3328 of *Lecture Notes in Computer Science*, pages 148–160. Springer, 2004.

9 Patricia Bouyer, Samy Jaziri, and Nicolas Markey. On the value problem in weighted timed games. Research Report LSV-14-12, Laboratoire Spécification et Vérification, ENS Cachan, France, October 2014. 24 pages.

10 Patricia Bouyer, Kim G. Larsen, Nicolas Markey, and Jacob Illum Rasmussen. Almost optimal strategies in one-clock priced timed games. In *Proceedings of the 26th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'06)*, volume 4337 of *Lecture Notes in Computer Science*, pages 345–356. Springer, 2006.

11 Thomas Brihaye, Véronique Bruyère, and Jean-François Raskin. On optimal timed strategies. In *Proceedings of the Third international conference on Formal Modeling and Analysis of Timed Systems (FORMATS'05)*, volume 3829 of *Lecture Notes in Computer Science*, pages 49–64. Springer, 2005.

12 Thomas Brihaye, Gilles Geeraerts, Axel Haddad, Engel Lefaucheux, and Benjamin Monmege. Simple priced timed games are not that simple. Research Report 1507.03786, arXiv, July 2015.

13 Thomas Brihaye, Gilles Geeraerts, Axel Haddad, and Benjamin Monmege. To reach or not to reach? Efficient algorithms for total-payoff games. In Luca Aceto and David de Frutos Escrig, editors, *Proceedings of the 26th International Conference on Concurrency Theory (CONCUR'15)*, volume 42 of *LIPIcs*, pages 297–310. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, September 2015.

14 Thomas Brihaye, Gilles Geeraerts, Shankara Narayanan Krishna, Lakshmi Manasa, Benjamin Monmege, and Ashutosh Trivedi. Adding Negative Prices to Priced Timed Games. In *Proceedings of the 25th International Conference on Concurrency Theory (CONCUR'13)*, volume 8704 of *Lecture Notes in Computer Science*, pages 560–575. Springer, 2014.

15 D. Gale and F. M. Stewart. Infinite games with perfect information. In *Contributions to the theory of games, vol. 2. Annals of Mathematical Studies*, volume 28 of *Lecture Notes in Computer Science*, pages 245–266. Princeton University Press., 1953.

16 Thomas Dueholm Hansen, Rasmus Ibsen-Jensen, and Peter Bro Miltersen. A faster algorithm for solving one-clock priced timed games. In *Proceedings of the 24th International Conference on Concurrency Theory (CONCUR'13)*, volume 8052 of *Lecture Notes in Computer Science*, pages 531–545. Springer, 2013.

17 Peter J. Ramadge and W. Murray Wonham. The control of discrete event systems. In *Proceedings of the IEEE*, volume 77(1), pages 81–98, 1989.

18 Michał Rutkowski. Two-player reachability-price games on single-clock timed automata. In *Proceedings of the 9th Workshop on Quantitative Aspects of Programming Languages (QAPL'11)*, volume 57 of *Electronic Proceedings in Theoretical Computer Science*, pages 31–46, 2011.