Equilibria in Multiplayer Timed Games with Reachability Objectives

Shankara Narayanan Krishna, Samarth Mishra

Department of Computer Science and Engineering,
IIT Bombay, Powai, Mumbai-76, India.
{krishnas,samarth}@cse.iitb.ac.in

Abstract. Timed automata [1] are a well accepted formalism in modelling real time systems. In this paper, we study sequential multiplayer games on timed automata with costs attached to the locations and edges and try to answer the question of the existence of Nash Equilibrium (NE), Leader Equilibrium (LE) (or *Stackel Equilibrium*) [3] and Incentive Equilibrium(IE) [2]. Considering memoryless strategies, we show that with three clocks it is undecidable whether there exists a NE(LE or IE) where player 1, in a two-player game, has a cost bounded by a constant B.

1 Introduction

The concept of games on automata has been introduced with the central idea of multiple players making the automaton run in order to fulfil their interests. These games are classified into competitive and non-competitive games. In competitive games, one player wins the game while the others lose the game. In non-competetive games, there is no notion of winning or losing; each player plays the game in a way so that she gets a favourable outcome.

The games we consider in this paper are *non-competetive*, with 2 or more players on *weighted timed automata*. Costs are attached to the locations as well as edges. We show that 3 clocks are sufficient to obtain undecidability for the existence of NE (LE or IE). The definitions we use for Incentive Equilibrium and Leader Equilibrium are as mentioned in [2] and [3] respectively.

2 Preliminaries

2.1 Weighted Timed Automata (WTA)

We recall the definition of WTAs as in [5].

A *weighted timed automaton* is a tuple $\mathcal{A} = (L, L_0, X, Z, E, \eta, C)$ where L is a finite set of locations, $L_0 \subseteq L$ is a set of initial locations, X is a finite set of clocks, Z is a finite set of costs (let $|Z| = m$), $E \subseteq L \times \mathcal{C}(X) \times U_0(X) \times L$ is the set of transitions. A transition $e = (l, \varphi, \phi, l') \in E$ is a transition from l to l' with valuation $\nu \in \mathbb{T}^X$ satisfying the constraint φ, and ϕ gives the set of clocks to be reset. $\eta : L \to \mathcal{C}(X)$ defines the invariants of each location. $C : L \cup E \to \mathbb{N}^m$ is the cost function which gives the rate of growth of each cost. Note that the costs are called *stopwatches* if $C : L \cup E \to \{0,1\}^m$. From the nature of the costs and stopwatches, it is clear that stopwatches are restricted costs. WTA with stopwatches form a subclass of WTA with costs.

The semantics of a WTA $\mathcal{A} = (L, L_0, X, Z, E, \eta, C)$ is given by a labelled timed transition system $\mathcal{T}_\mathcal{A} = (S, \to)$ where $S = L \times \mathbb{T}^X \times \mathbb{T}^Z$. We refer to an element $l \in L$ of a WTA $\mathcal{A}$ as a *location* while we refer to an element $(l, \nu, \mu) \in S$ of $\mathcal{T}_\mathcal{A}$ as a *state*. The terms transition and edge are used interchangeably. $\to$ is composed of transitions

- Time elapse t in l: A state (l, ν, μ) after time elapse t evolves to (l', ν', μ'), where $l' = l$, $\nu' = \nu + t$, $\mu' = \mu + C(l) * t$ and for all $0 \le t' \le t$, $\nu + t' \models \eta(l)$.
- Location switch: $(l, \nu, \mu) \xrightarrow{(\varphi,\phi)} (l', \nu', \mu')$ if there exists $e = (l, \varphi, \phi, l') \in E$, such that $\nu \models \varphi$, $\nu' = \nu[\phi := 0]$ and $\mu' = \mu + C(e)$. Here, $\nu \models \eta(l)$, $\nu' \models \eta(l')$.

A path is a sequence of consecutive transitions in the transition system $\mathcal{T}_\mathcal{A}$. A path ρ starting at (l_0, ν'_0, μ'_0) is denoted as $\rho = (l_0, \nu'_0, \mu'_0) \xrightarrow{t_1} (l_0, \nu_1, \mu_1) \xrightarrow{(\varphi_1,\phi_1)} (l_1, \nu'_1, \mu'_1) \xrightarrow{t_2} (l_1, \nu_2, \mu_2) \xrightarrow{(\varphi_2,\phi_2)} (l_2, \nu'_2, \mu'_2) \cdots (l_n, \nu'_n, \mu'_n)$. Note that $\nu_i = \nu'_{i-1} + (t_i - t_{i-1}), \nu_i \models \varphi_i$, $\nu'_i = \nu_i[\phi := 0]$ and $\mu_i = \mu'_{i-1} + C(l_{i-1}) * (t_i - t_{i-1})$, $\mu'_i = \mu_i + C(l_{i-1}, \varphi_i, \phi_i, l_i)$.

2.2 Deterministic Two Counter Machines

A deterministic 2-counter machine $\mathcal{M}$ with counters c_1 and c_2 is described by a program formed by five basic instructions:

- l_m : goto l_j;
- l_m : if $c_i = 0$ then goto l_j else goto l_k; (check for zero)
- l_m : $c_i := c_i + 1$, goto l_j;(increment counter c_i)
- l_m : $c_i := c_i - 1$, goto l_j; (decrement counter c_i. A decrement instruction for c_i is always preceded by a check for zero for c_i so that $\mathcal{M}$ does not get stuck)
- l_m : HALT;

Without loss of generality, assume that the instructions are labelled $l_1, \dots, l_n$ where $l_n = HALT$ (a special instruction) and that to begin with, both counters have value zero. Let $L = \{l_i \mid 1 \le i \le n\}$ be the labels of the instructions in $\mathcal{M}$. A configuration of the two counter machine is a tuple $(l, n_1, n_2) \subseteq L \times \mathbb{N} \times \mathbb{N}$. A configuration tells us the current instruction of $\mathcal{M}$ as well as the values of c_1, c_2. The behavior of the machine is described by a possibly infinite sequence of configurations $(l_0, 0, 0), (l_1, C_1^1, C_2^1), \dots (l_k, C_1^k, C_2^k) \dots$ where C_1^k and C_2^k are the respective counter values and l_k is the label of the kth instruction. The halting problem of such a machine is known to be undecidable [**?**].

Let $V_\mathcal{M} = \{(c_1, c_2) \mid \exists l$ such that $\mathcal{M}$ visits $(l, c_1, c_2)\}$ be the set of all pairs of values of counters c_1, c_2 which result from $\mathcal{M}$. Two configurations (l, a, b) and (l', a', b') are distinct if $l \neq l'$ or $a \neq a'$ or $b \neq b'$. Clearly, $V_\mathcal{M}$ is finite iff $\mathcal{M}$ visits finitely many distinct configurations.

3 Timed Multiplayer Reachability Game

The timed multiplayer reachability game structure is defined as a tuple $\mathcal{G} = (L, L_0, X, Z, P, E, \eta, C, F)$ where, L is a finite set of locations, $L_0 \subseteq L$ is the singleton set containing the initial location, F is a set of final or target locations and X is a finite set of clocks; Z is a set of n cost variables, where n is the number of players. P is the set of n players $\{P_1, P_2 \dots P_n\}$; $E \subseteq L \times L \times \Theta(X) \times 2^X$ is the set of transitions, where $\Theta(X)$ is the set of all intervals for each of the clocks from which constraints on edge transitions are decided and 2^X corresponds to the set of clocks which are reset on taking that transition. $\eta : L \to \Theta(X)$ is a function assigning clock valuation invariants to each location; $C : L \to \mathbb{N}^n$ is a function associating cost growth rate to each of the players on the game locations. The game is then defined on the above structure as : The set of locations is partitioned into $L_1, L_2, \dots L_n$ where L_i is the set of locations which belong to the player P_i. At each location, the owner of that location chooses an edge $e_i \in E$ and a time delay t_i. Suppose from a state $q = (l, \nu, \mu), l \in L_i$, player P_i chooses edge e_i and time delay t_i, then there exist states q', q'' such that $q = (l, \nu, \mu) \xrightarrow{t_i} q' = (l, \nu + t_i, \mu') \xrightarrow{e_i} q'' = (l'', \nu'', \mu'')$. where the state transitions are as defined in Section 2.1, with a slight difference that as the edge $\xrightarrow{e_i}$ is taken, there is no further cost incurred for any of the players. Also note that here, $\nu \models \eta(l)$ and $\nu'' \models \eta(l'')$ and $\nu + t_i$ satisfies the transition edge constraints on clock valuations.

A strategy for a player P_i is a function $\lambda_i : S \to (R^+ \times E_i) \cup \{\infty\}$ (where S is the set of all possible values of (l_i, ν, μ) where $\nu \models \eta(l_i)$ and $l_i \in Ł_i$, close to the definition in Section 2.1), such that if $\lambda_i(q) = (t_i, e_i)$, then it possible to incur a delay of t_i at a state q, and then take a discrete transition e_i. Note that a delay t_i of ∞ is allowed if $\eta(l_i)$ allows it. A strategy profile is an n-tuple $(\lambda_1, \lambda_2 \dots \lambda_n)$ of strategies where λ_i is a strategy for player P_i. A run ρ is said to be played according to a strategy profile if for each node l_i that the game visits, the outgoing edge e_i and the time delay t_i are chosen according to λ_i (assuming $l_i \in L_i$). ρ in the above case, is said to be the outcome of λ and is denoted as $outcome(\lambda)$. We also define the final accumulated costs for each of the players, as a function of the run, such that $u_i(\rho)$ is the final accumulated cost for player P_i. An n-tuple of all these costs for a given run, $u(\rho) = (u_1(\rho), u_2(\rho) \dots u_n(\rho))$. The strategies we consider are *memoryless*, since we only look at the current state to decide the next move. A *terminal history* is a run ρ starting from the initial location, ending in a target location and never passing through a target location in between. Given a terminal history ρ, the payoff of player i is $-u_i(\rho)$ where $u_i(\rho)$ is the cost accumulated along ρ. For a run ρ that does not end in a target location, the payoff for both players is ∞. The *objective* of each player is to reach a target location accumulating as small a cost as possible. Our games are therefore non-competetive, since neither player aims to increase the others' cost.

4 Examples

In the example figure 4.1, nodes 1,4 and 8 belong to player 1, node 2 belongs to player 2, node 3 belongs to player 3, and the remaining are target nodes. Note that the owner of the target nodes do not matter because there are no moves left after reaching one of those locations. In the example given, location i has been represented as l_i and the edge between locations l_i and l_j as e_{ij}. Note that in the examples below, the strategy we've mentioned is independent of the initial clock valuations when the node is arrived at (because the clock is being reset after each edge). Also, the strategy mentioned is independent of the cost accumulated by each of the players. Hence, in the examples given below, we change our notation for strategy λ slightly and is now represented as $\lambda(l)$ instead of $\lambda(l, \nu, \mu)$.

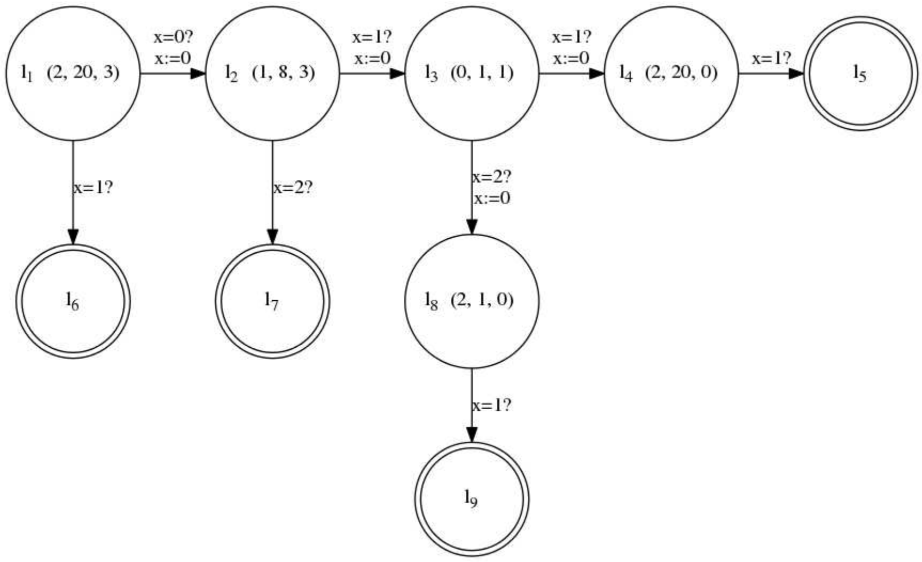

Fig. 4.1. A multiplayer timed reachability game

4.1 Nash Equilibrium

The strategy profile $\lambda = (\lambda_1, \lambda_2, \lambda_3)$ is a Nash Equilibrium, where $\lambda_1(l_1) = (1, e_{16})$, and λ_2 and λ_3 can be any strategies. Note that λ_2 and λ_3 need not be defined in the above strategy profile, given λ_1, because the game wouldn't progress to a point where the location belongs to Player 2 or 3. It can be seen that Player 1 can't unilaterally deviate from this strategy to (strictly) decrease his total cost incurred. In the above example, $u(outcome(\lambda)) = (2, 3, 3)$.

4.2 Leader Equilibrium

In the example under consideration, player 2 is the leader. The following strategy profile is a leader equilibrium : $\lambda = (\lambda_1, \lambda_2, \lambda_3)$, where $\lambda_1(l_1) = (1, e_{12})$, $\lambda_2(l_2) = (2, e_{27})$ and λ_3 need not be defined due to reasons similar to that mentioned in the previous section, i. e., section 4.1.

It can be seen that the above strategy profile is a leader strategy profile because player 1 can't strictly benefit by unilaterally deviating. Also, this is the best cost for player 2 among all leader strategy profiles, and hence is a leader equilibrium.

4.3 Incentive Equilibrium

In an incentive strategy profile, the leader can provide an incentive to each of the other players by offering to take some cost from the player if he/she chooses some edge as suggested by the leader. Here, the strategy profile λ, also consists of a function $\iota : \{P_1 \dots P_n\} - \{P_l\} \to \mathbb{R}_{\geq 0}$, where P_l is the leader and $\iota(P_i)$ gives the incentive given to (or cost taken from) P_i in that strategy profile.

An incentive strategy profile which corresponds to an incentive equilibrium in our example game graph is: $\lambda = (\lambda_1, \lambda_2, \lambda_3, \iota)$, where $\lambda_1(l_1) = (0, e_{12})$, $\lambda_1(l_8) = (1, e_{89})$, $\lambda_1(l_2) = (1, e_{23})$, $\lambda_3(l_3) = (2, e_{38})$, $\iota(P_1) = 1$, $\iota(P_3) = 1$. It can be seen that the above strategy profile is an incentive equilibrium and $u(outcome(\lambda)) = (2, 6, 4)$.

5 Existence of Nash Equilibrium

In this section, we show that the existence of a NE such that player P_1 has a payoff bounded by a constant B is undecidable. First, we show that the following question regarding two counter machines is undecidable, and then use it to prove our result. The proof for theorem 1 has been re-written exactly as has been given in [4].

Q1: Given a two counter machine $\mathcal{M}$, is it decidable whether, starting from a configuration $(q_0, 0, 0)$, $\mathcal{M}$ will visit finitely many distinct configurations?

Proposition 1. *Question Q1 is undecidable.*

Proof. The halting problem for two counter machines is known to be undecidable. We show a reduction of the halting problem to the problem in question *Q1*. Assume that there is an algorithm $\mathcal{A}_{\mathcal{M}}$ to detect whether $\mathcal{M}$ visits finitely many distinct configurations starting from $(q_0, 0, 0)$.

– If $\mathcal{M}$ indeed visits only finitely many distinct configurations, then we simulate $\mathcal{M}$ starting from the initial configuration, maintaining a list of visited configurations until it halts, or it revisits a previously visited configuration. In the latter case, we conclude that $\mathcal{M}$ does not halt. Note that since $\mathcal{M}$ visits only finitely many configurations, one of the above outcomes is bound to happen.

– If $\mathcal{M}$ visits infinitely many distinct configurations, then we conclude that it does not halt.

Thus, $\mathcal{A}_{\mathcal{M}}$ yields a solution to the halting problem of $\mathcal{M}$. Since we know that this is not possible, the question of whether $\mathcal{M}$ visits finitely many distinct configurations starting from a configuration $(q_0, 0, 0)$ is undecidable. □

Theorem 1. *Given a timed 2-player reachability game structure $\mathcal{G}$ with three clocks and stopwatch costs, it is undecidable if there exists an NE (λ_1, λ_2) for the corresponding game, in the outcome of which P_1 incurs a cost bounded by a constant B; that is, $u_1(outcome(\lambda_1, \lambda_2)) < B$.*

Proof. We construct a timed game structure $\mathcal{G}$ with 3 clocks x_1, x_2, x_3 that simulates a two counter machine $\mathcal{M}$. The clocks x_1, x_2, x_3 encode the counter values c_1, c_2 as follows: At the end of each module in $\mathcal{G}$,

$$\nu(x_1) = \frac{1}{2^{c_1}3^{c_2}}, \nu(x_2) = 0.$$

The clock x_3 is used to do calculations and is used for rough work. We show that $\mathcal{G}$ has an NE (λ_1, λ_2) such that $u_1(outcome(\lambda_1, \lambda_2)) < 6$ iff $\mathcal{M}$ visits only finitely many distinct configurations.

Construction of $\mathcal{G}$

In all locations of $\mathcal{G}$, we add a loop with constraint $x_i = 1, i \in \{1, 2, 3\}$ and reset it when x_i reaches 1. For convenience, we will not draw this in any of the locations in the various modules. We would also omit this loop from the player strategy when mentioned. The loop is as depicted below. This loop ensures that the values of x_1, x_2 are always related to each other in one of the following ways:

$$\nu(x_1) - \nu(x_2) = \frac{1}{2^{c_1}3^{c_2}}, 0 \le x_2 \le x_1 \le 1, \text{ or}$$
$$\nu(x_2) - \nu(x_1) = 1 - \frac{1}{2^{c_1}3^{c_2}}, 0 \le x_1 \le x_2 \le 1.$$

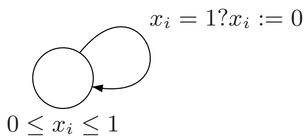

Fig. 5.1. Loop on all locations to ensure that $0 \le x_i \le 1$ for $i \in \{1, 2, 3\}$

$\mathcal{G}$ is constructed by connecting the modules simulating the various increment, decrement and zero check instructions according to $\mathcal{M}$. For example, if $\mathcal{M}$ contains the instructions l_1: Increment c_1, goto l_2, l_2: if $c_1 > 0$, goto l_3, else HALT, and l_3: decrement c_1, goto l_1, then $\mathcal{G}$ is obtained by connecting the modules for incrementing c_1, checking if c_1 is zero, then decrementing c_1 in a round robin fashion. We describe below, the modules for increment, decrement as well as zero check. In each of the modules, a rectangular node represents a location belonging to player P_2, and the oval node represents a a player P_1 node. In the following text, we will use "player 1" and "player P_1" interchangeably. For ease of notation, we have used in the locations costs $\in \mathbb{N}$ instead of just 0 and 1. It must be noted that a location which uses a cost other than 0 and 1 can be replaced with a sequence of locations, each of which have costs over $\{0,1\}^2$. We have illustrated this in the case of Figure 5.3. This can be done for all the figures we have drawn.

Simulation of an increment instruction l_i: increment c_1, goto l_j:

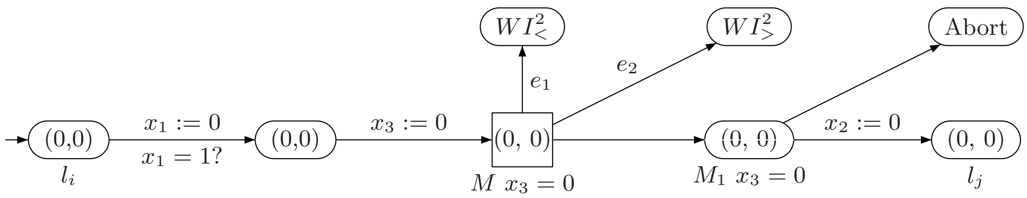

Fig. 5.2. Module for incrementing c_1

Figure 5.2 is the module for incrementing c_1. On entry into this module, the values of clocks x_2, x_3 is 0. If the first instruction of $\mathcal{M}$ is an increment instruction, then on entry, $\nu(x_1) = \nu(x_2) = \nu(x_3) = 0$. Note that in the beginning $c_1 = c_2 = 0$, so that $\nu(x_2) - \nu(x_1) = 1 - \frac{1}{2^{c_1}3^{c_2}}$.

Assume that the clock valuations of x_1, x_2, x_3 on entry is given by the tuple $(x_{old}, 0, 0)$ where x_{old} is of the form $\frac{1}{2^{c_1}3^{c_2}}$. On leaving l_i, $x_1 = 0, x_2 = x_3 = 1 - x_{old}$. A non-deterministic amount of time x_{new} is spent in the next location. On coming out of this location, x_3 is reset to zero, so that $x_1 = x_{new}, x_2 = 1 - x_{old} + x_{new}$ or $x_{new} - x_{old}$ and $x_3 = 0$ (Note that if $x_{new} < x_{old}$, then $x_2 = 1 - x_{old} + x_{new}$ and if $x_{new} > x_{old}$, then $x_2 = x_{new} - x_{old}$ due to the self loop on all locations). If $x_{new} = x_{old}$, then $x_2 = 0$ (note that when $x_2 = 1$, the self loop resets x_2 to zero). No time delay can happen at location M or M_1 due to the invariant $x_3 = 0$. Player 1, at M_1, has two choices of going to the Abort widget or continuing with the simulation of $\mathcal{M}$, while player 2, at M, can check whether x_{new} is indeed $\frac{x_{old}}{2}$ by going to the widgets $WI^2_<$ or $WI^2_>$. We now consider the three cases when $x_{new} = \frac{x_{old}}{2}, x_{new} > \frac{x_{old}}{2}$ and $x_{new} < \frac{x_{old}}{2}$.

Let us look at the widgets $WI^2_<$ and $WI^2_>$ with respect to this. We explain in detail the widget $WI^2_<$. On entry into $WI^2_<$, $x_1 = x_{new}$, $x_2 = 1 - x_{old} + x_{new}$ (or $x_2 = x_{new} - x_{old}$). We walk through the locations of $WI^2_<$ and list down the values of the clocks as well as the costs incurred by the players. Note that till the entry into widgets $WI^2_<, WI^2_>$ and Abort, no cost is incurred by either player.

As given by Table 1, the costs incurred by P_1 and P_2 at the end of the widget $WI^2_<$ are respectively $3(x_{old} - 2x_{new} + 2)$ and $3(2x_{new} - x_{old} + 2)$. If $x_{new} = \frac{x_{old}}{2}$, then the costs are both 6, while if $x_{new} < \frac{x_{old}}{2}$ then P_1 incurs a cost > 6 while P_2 incurs a cost < 6. If $x_{new} > \frac{x_{old}}{2}$, then the cost incurred by P_1 is < 6 while that

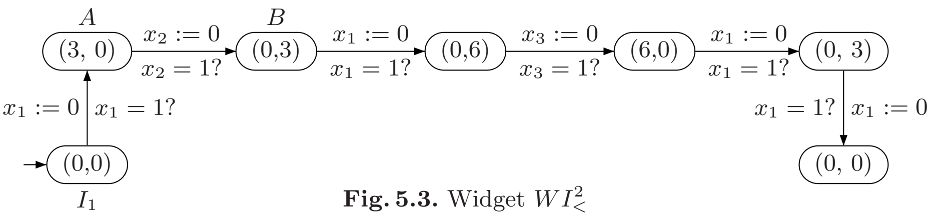

Fig. 5.3. Widget $WI^2_<$

Location of $WI^2_<$	$\nu(x_1)$ on entry	$\nu(x_2)$ on entry	$\nu(x_3)$ on entry	Accumulated cost of P_1 on entry	Accumulated cost of P_2 on entry
Initial	x_{new}	$1 - x_{old} + x_{new}$ or $x_{new} - x_{old}$	0	0	0
2	0	$1 - x_{old}$	$1 - x_{new}$	0	0
3	x_{old}	0	$1 - x_{new} + x_{old}$	$3x_{old}$	0
4	0	1-x_{old}	1- x_{new}	$3x_{old}$	3 (1-x_{old})
5	x_{new}	1-$(x_{old} - x_{new})$	0	$3x_{old}$	3 (1-x_{old}) $+ 6x_{new}$
6	0	1-x_{old}	1-x_{new}	$3x_{old} + 6(1 - x_{new})$	3 (1-x_{old}) $+ 6x_{new}$
7	0	1-x_{old}	1-x_{new}	$3x_{old} + 6(1 - x_{new})$	3 (1-x_{old}) $+ 6x_{new} + 3$

Table 1. Clock valuations and costs incurred in $WI^2_<$

of P_2 is > 6. The widget $WI^2_>$ is obtained by switching the costs in all locations of $WI^2_<$.

The costs incurred by P_1, P_2 at the end of $WI^2_>$ are respectively $3(2x_{new} - x_{old} + 2)$ and $3(x_{old} - 2x_{new} + 2)$. The widgets $WI^2_<, WI^2_>$ respectively are player 2's opportunities to catch player 1 when $x_{new} < \frac{x_{old}}{2}$ and $x_{new} > \frac{x_{old}}{2}$. Note that if $x_{new} < \frac{x_{old}}{2}$, then P_2 can move into $WI^2_<$, by a strategy $\lambda_2(M, (x_{new}, 1 - x_{old} + x_{new}, 0), (0, 0)) = (0, e_1)$ and can move into $WI^2_>$ by a strategy $\lambda_2(M, (x_{new}, 1 - x_{old} + x_{new}, 0), (0, 0)) = (0, e_2)$ if $x_{new} > \frac{x_{old}}{2}$.
Note that for incrementing c_2, widgets $WI^3_<$ and $WI^3_>$ can be constructed which will check if $x_{new} = \frac{x_{old}}{3}$.

Simulation of a decrement instruction l_i: decrement c_1, goto l_j:

Since we have assumed that a decrement instruction is preceded by a zero check instruction, the above module starts with $x_1 = \frac{1}{2^{c_1} 3^{c_2}}$ and ends with $x_1 = \frac{1}{2^{c_1 - 1} 3^{c_2}}$, with $c_1 - 1 \geq 0$. This is similar to the module in Figure 5.2. On entry, $x_1 = x_{old}$, where x_{old} is of the form $\frac{1}{2^{c_1} 3^{c_2}}, c_1 > 0$ and $x_2 = x_3 = 0$. A non-deterministic amount of time x_{new} is spent in the second location in Figure 5.5. Player 2 can check if $x_{new} = 2x_{old}$ by entering either of the widgets $WD^2_<$ or $WD^2_>$. $WD^2_<, WD^2_>$ respectively are player 2's chances to catch player 1 respectively when $x_{new} < 2x_{old}$ and $x_{new} > 2x_{old}$ by choosing strategies $\lambda_2(N, (x_{new}, x_{new} - x_{old}, 0), (0, 0)) = (0, e'_1)$ if $x_{new} < 2x_{old}$ and $\lambda_2(N, (x_{new}, x_{new} - x_{old}, 0), (0, 0)) = (0, e'_2)$ if $x_{new} > 2x_{old}$. In these cases, player 1 incurs a cost > 6, while if $x_{new} = 2x_{old}$, the cost is exactly 6. Table 2 runs us through the clock valuations and accumulated costs incurred by the players in $WD^2_>$. Note that till the time a widget is entered, no cost is incurred by either player. The widget $WD^2_<$ is obtained by switching the costs in all locations of $WD^2_>$.

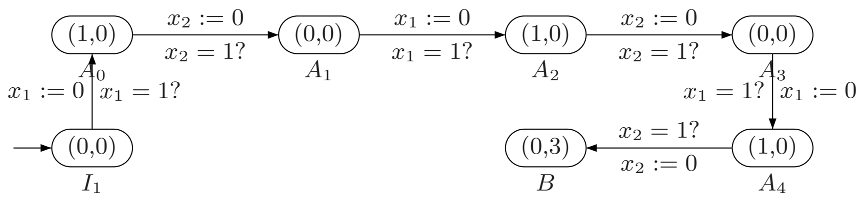

Fig. 5.4. The location A in Figure 5.3 can be replaced by the path consisting of the locations A_0, A_1, A_2, A_3, A_4. The cost accumulated between A_0 and B is $(3x_{old}, 0)$ which is the same as the cost accumulated between A and B. This can be done for all locations with costs (i, j) where i or $j \notin \{0, 1\}$ in a way that the accumulated costs are same.

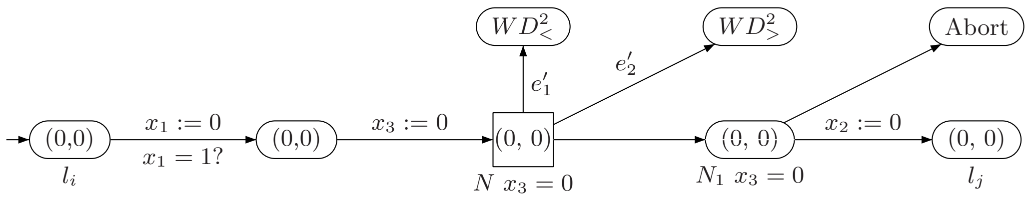

Fig. 5.5. Module for decrementing c_1

The costs incurred by P_1, P_2 respectively at the end of the widget $WD^2_>$ are $3(x_{new} - 2x_{old} + 2)$ and $3(2x_{old} - x_{new} + 2)$. Clearly, if $x_{new} > 2x_{old}$, P_1 incurs a cost > 6, while P_2 incurs a cost > 6 when $x_{new} < 2x_{old}$. The widget $WD^2_<$ is similar and is given below. The costs incurred by P_1, P_2 at the end of this widget respectively are $3(2x_{old} - x_{new} + 2)$ and $3(x_{new} - 2x_{old} + 2)$.

Note that for decrementing c_2, widgets $WD^3_<$ and $WD^3_>$ can be constructed which will check if $x_{new} = 3x_{old}$.

Simulation zero check l_i: if $c_2 = 0$, goto l_j, else goto l_k:

Figure 5.7 is the module for simulating the instruction for zero check of c_2. The invariant $x_3 = 0$ enforces no time be spent at locations l_i, Z and NZ. Player 1 can non-deterministically choose to goto Z or NZ. In a correct simulation, player 1 must goto Z when $c_2 = 0$ and to NZ when $c_2 \neq 0$. Otherwise, player 2 can move into widgets *Check* $c_2 = 0$ and *Check* $c_2 > 0$. We now explain the functionality of the widgets *Check* $c_2 = 0$ and *Check* $c_2 > 0$. *Check* $c_2 = 0$ is the widget for ensuring that c_2 is zero, while the widget *Check* $c_2 > 0$ ensures that c_2 is non-zero. The locations J, K, L in the widget *Check* $c_2 = 0$ form a loop that repeatedly multiplies x_1 by 2 until x_1 becomes 1. Note that this is possible only if $c_2 = 0$. The widgets $WD^2_<$ and $WD^2_>$ can be invoked by player 2 to check that this multiplication goes on correctly in each round (that is, $x_{new} = 2x_{old}$). The location $T1$ in widget *Check* $c_2 = 0$ can be reached only when x_1 becomes 1, which is possible only if $c_2 = 0$. *Check* $c_2 > 0$ is similar to *Check* $c_2 = 0$. The upper loop CDE repeatedly multiplies x_1 by 2, while the lower loop CDG multiplies x_1 by 3. This continues till $x_1 = \frac{1}{3}$. The location $T2$ can be reached

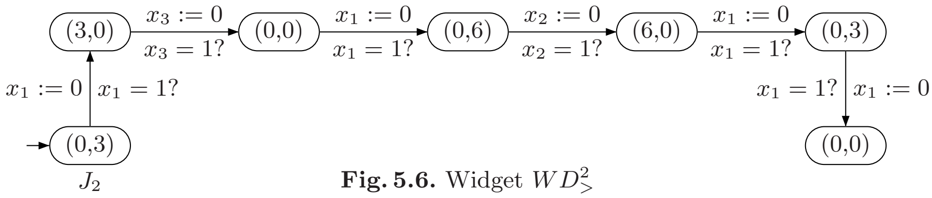

Fig. 5.6. Widget $WD^2_>$

Location of $WD^2_>$	$\nu(x_1)$ on entry	$\nu(x_2)$ on entry	$\nu(x_3)$ on entry	Accumulated cost of P_1 on entry	Accumulated cost of P_2 on entry
Initial	x_{new}	$1 - x_{old} + x_{new}$ or $x_{new} - x_{old}$	0	0	0
2	0	$1 - x_{old}$	$1 - x_{new}$	0	$3(1 - x_{new})$
3	x_{new}	$x_{new} - x_{old}$	0	$3x_{new}$	$3(1 - x_{new})$
4	0	1-x_{old}	1- x_{new}	$3x_{new}$	$3(1 - x_{new})$
5	x_{old}	0	1-$(x_{new} - x_{old})$	$3x_{new}$	$3(1 - x_{new}) + 6x_{old}$
6	0	1-x_{old}	1-x_{new}	$3x_{new} + 6(1 - x_{old})$	$3(1 - x_{new}) + 6x_{old}$
7	0	1-x_{old}	1-x_{new}	$3x_{new} + 6(1 - x_{old})$	$3(1 - x_{new}) + 6x_{old} + 3$

Table 2. Clock valuations and costs incurred in $WD^2_>$

only in this case, which can happen only when $c_2 > 0$. Player 2 can invoke the widgets $WD^2_>$ or $WD^2_<$ as part of the upper loop to check if multiplication by 2 is happening correctly and widgets $WD^3_>$ or $WD^3_<$ with respect to the lower loop to check if multiplication by 3 is happening correctly.

Note that if player 1 enters $Z(NZ)$ when $c_2 > 0 (c_2 = 0)$, then the locations $T1, T2$ in the widgets *Check* $c_2 = 0$ and *Check* $c_2 > 0$ can never be reached. Further, if player 1 enters Z when $c_2 > 0$, the transition from J to K cannot be taken. Likewise, if NZ is entered when $c_2 = 0$, the transition from C to D cannot be taken. The only way then to reach a target location in widgets *Check* $c_2 = 0$ and *Check* $c_2 > 0$ is when player 2 forces a move into one of the widgets $WD^i_>$ or $WD^i_<$ $(i \in \{2, 3\})$. This can make player 1 incur a cost ≥ 6. If the simulation is correct and player 1 enters $Z(NZ)$ diligently (by having a strategy $\lambda_1(l_i, (x_{old}, 0, 0), (0, 0)) = (0, e_z)$ if x_{old} is of the form $\frac{1}{2^m}$ and $\lambda_1(l_i, (x_{old}, 0, 0), (0, 0)) = (0, e_{nz})$ if x_{old} is of the form $\frac{1}{2^m 3^n}$, $n > 0$), then l_j (l_k) is reached).

Player 1 can enter the widget Abort after the simulation of any instruction. On entering this module, c_2 is decremented and c_1 is incremented until c_2 becomes zero. This is followed by incrementing c_1 once more, so that starting with $x_1 = \frac{1}{2^{c_1} 3^{c_2}}, x_2 = x_3 = 0$ in Abort, we obtain $x_1 = \frac{1}{2^{c_1+c_2+1}}, x_2 = x_3 = 0$ at location H. The costs incurred by P_1 and P_2 if all increment and decrement instructions are executed correctly on reaching location F in Abort are $5 + \frac{1}{2^{c_1+c_2+1}}$ and 6 respectively. We will use this in the proof below.

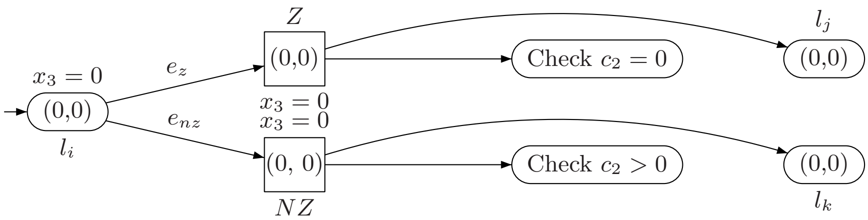

Fig. 5.7. Zero Check for c_2

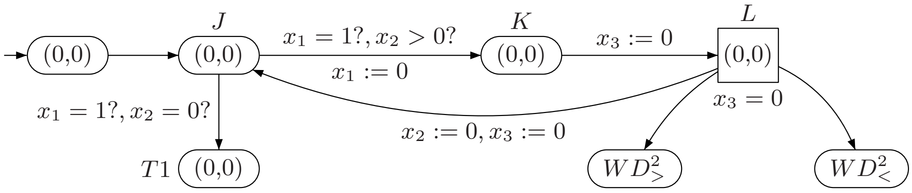

Fig. 5.8. Check $c_2 = 0$

Existence of bounded NE in $\mathcal{G} \Leftrightarrow$ finiteness of $V_{\mathcal{M}}$

Having finished the details on the construction of $\mathcal{G}$, we now prove that

$\mathcal{M}$ visits finitely many distinct configurations *iff* there exists a NE in the outcome of which player 1 has a cost bounded above by 6.

1. Assume that $\mathcal{M}$ visits finitely many distinct configurations. Then the number of distinct pairs of values (c_1, c_2) is finite. Recall from Section 2.2 that $V_{\mathcal{M}} = \{(c_1, c_2) \mid \exists q$ such that $\mathcal{M}$ visits $(q, c_1, c_2)\}$, the set of all pairs of values of counters c_1, c_2 which result from $\mathcal{M}$. Clearly, $V_{\mathcal{M}}$ is finite iff $\mathcal{M}$ visits finitely many distinct configurations. If $\mathcal{M}$ visits finitely many distinct configurations, let $c^{max} = max\{c_1 + c_2 \mid (c_1, c_2) \in V_{\mathcal{M}}\}$.
   Consider the strategy profile $(\lambda_1^*, \lambda_2^*)$ given as follows:
   - λ_1^* is the strategy for P_1 which suggests it to correctly simulate $\mathcal{M}$ until the counters attain values summing up to c^{max}, and then to enter the widget Abort. In Abort, correctly simulate widgets *Increment* c_1 and *Decrement* c_2 until $c_2 = 0$.
   - λ_2^* is the strategy for P_2 which suggests it to enter any of the widgets $WD^2_<$, $WD^3_<$, $WD^2_>$, $WD^3_>$, $WI^2_>$, $WI^3_>$, $WI^2_<$, $WI^3_<$, *Check* $c_2 = 0$, *Check* $c_2 > 0$, *Check* $c_1 = 0$ or *Check* $c_1 > 0$ when P_1 makes a simulation error. Precisely, λ_2^* is the strategy for P_2 such that it enters $WI^2_>(WI^2_<)$ when $x_{new} > \frac{x_{old}}{2}(x_{new} < \frac{x_{old}}{2})$, $WI^3_>(WI^3_<)$ when $x_{new} > \frac{x_{old}}{3}(x_{new} < \frac{x_{old}}{3})$, $WD^2_>(WD^2_<)$ when $x_{new} > 2x_{old}(x_{new} < 2x_{old})$, and $WD^3_>(WD^3_<)$ when $x_{new} > 3x_{old}(x_{new} < 3x_{old})$.

   The outcome of $(\lambda_1^*, \lambda_2^*)$ is a run in which instructions of $\mathcal{M}$ are simulated correctly until counter values sum up to c^{max}, followed by the widget Abort. P_2 does not execute any transitions in this run. Hence,

   $$u_1(outcome(\lambda_1^*, \lambda_2^*)) = 5 + \frac{1}{2^{c^{max}+1}},\ u_2(outcome(\lambda_1^*, \lambda_2^*)) = 6$$

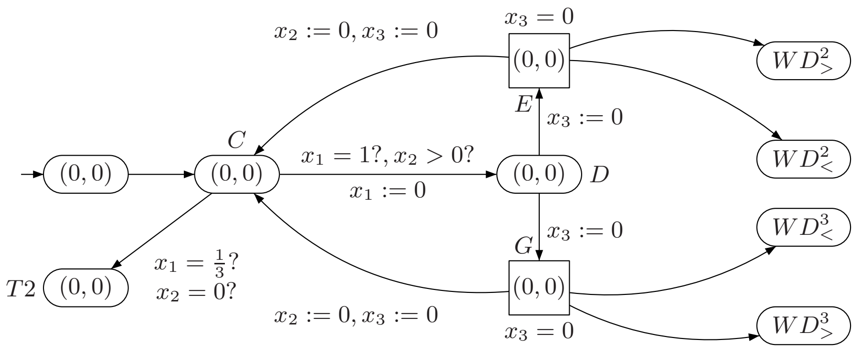

Fig. 5.9. Check $c_2 > 0$

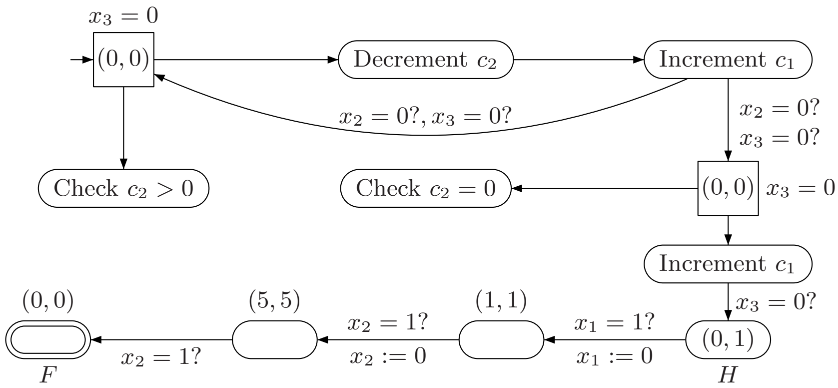

Fig. 5.10. Abort

We prove that $(\lambda_1^*, \lambda_2^*)$ is a NE. Clearly, the cost incurred by P_1 in this strategy profile is < 6. To prove that $(\lambda_1^*, \lambda_2^*)$ is an NE, we have to show that

- For any strategy λ_1 of P_1 such that $\lambda_1 \neq \lambda_1^*$, $u_1(outcome(\lambda_1^*, \lambda_2^*)) \leq u_1(outcome(\lambda_1, \lambda_2^*))$, and
- For any strategy λ_2 of P_2 such that $\lambda_2 \neq \lambda_2^*$, $u_2(outcome(\lambda_1^*, \lambda_2^*)) \leq u_2(outcome(\lambda_1^*, \lambda_2))$.

Note that the game reaches a target location only if one of the widgets $WD^2_<$, $WD^3_<$, $WD^2_>$, $WD^3_>$, $WI^2_>$, $WI^3_>$, $WI^2_<$, $WI^3_<$, *Check* $c_2 = 0$, *Check* $c_2 > 0$, *Check* $c_1 = 0$, *Check* $c_1 > 0$ or Abort is invoked.

(a) Assume that $u_1(outcome(\lambda_1, \lambda_2^*)) < u_1(outcome(\lambda_1^*, \lambda_2^*))$. That means, $u_1(outcome(\lambda_1, \lambda_2^*)) < 5 + \frac{1}{2^{c^{max}+1}}$. Since $u_1(outcome(\lambda_1, \lambda_2^*))$ is bounded above by a finite value, it must be that the outcome of (λ_1, λ_2^*) is a run ending in a target location. We consider various cases for this target.

   i. Assume the target location is in $WI^2_>$. Since λ_2^* is such that it enters $WI^2_>$ when P_1 has committed a simulation error, P_2 will enter $WI^2_>$ on a clock valuation $(x_{new}, 1 - x_{old} + x_{new}, 0)$ such that $x_{new} > \frac{x_{old}}{2}$. Then the cost incurred by P_1 is $u_1(outcome(\lambda_1, \lambda_2^*)) > 6 > 5 + \frac{1}{2^{c^{max}+1}}$, which is a contradiction to our assumption.

ii. Assume the target location is in $WI^2_<$. Again, since λ_2^* is such that it enters $WI^2_<$ when P_1 has committed a simulation error, P_2 will enter $WI^2_<$ on a clock valuation $(x_{new}, 1 - x_{old} + x_{new}, 0)$ such that $x_{new} < \frac{x_{old}}{2}$. Then the cost incurred by P_1 is $u_1(outcome(\lambda_1, \lambda_2^*)) > 6 > 5 + \frac{1}{2^{c^{max}+1}}$, which is a contradiction to our assumption.

iii. Assume the target location is in $WD^2_>$. λ_2^* is such that it enters $WD^2_>$ from a clock valuation $(x_{new}, 1 - x_{old} + x_{new}, 0)$ such that $x_{new} > 2x_{old}$, the cost incurred by P_1 is $u_1(outcome(\lambda_1, \lambda_2^*)) > 6 > 5 + \frac{1}{2^{c^{max}+1}}$, which is a contradiction to our assumption.

iv. The cases of $WD^2_<$, $WD^3_>$, $WD^3_<$, $WI^3_<$ and $WI^3_>$ are similar.

v. The location is in *Check* $c_2 = 0$ or *Check* $c_2 > 0$. In this case, the target location must be in one of $WD^2_>, WD^2_<$, $WD^3_>$, $WD^3_<$, $WI^2_<$, $WI^2_>$, $WI^3_<$ or $WI^3_>$. The cases considered above apply.

vi. The target location is F in the widget Abort. Since $\lambda_1 \neq \lambda_1^*$, P_1 must have entered Abort before c^{max} is attained as the sum of the counter values (of course, P_1 simulates all the way correctly till it enters Abort and in Abort; if not, then the earlier cases apply). Then, $u_1(outcome(\lambda_1, \lambda_2^*)) = 5 + \frac{1}{2^{c_1+c_2+1}}$ where $c_1 + c_2 < c^{max}$. Then $u_1(outcome(\lambda_1, \lambda_2^*)) > 5 + \frac{1}{2^{c^{max}+1}}$, contradicting our assumption.

Thus, in all cases, we have $u_1(outcome(\lambda_1^*, \lambda_2^*)) \leq u_1(outcome(\lambda_1, \lambda_2^*))$ for all strategies $\lambda_1 \neq \lambda_1^*$.

(b) Assume that $u_2(outcome(\lambda_1^*, \lambda_2)) < u_2(outcome(\lambda_1^*, \lambda_2^*))$. That means, $u_2(outcome(\lambda_1^*, \lambda_2)) < 6$. Since $u_2(outcome(\lambda_1^*, \lambda_2))$ is bounded above by a finite value, it must be that the outcome of (λ_1^*, λ_2) is a run ending in a target location. We consider various cases for this target. The target location cannot be F in Abort, since P_2 incurs a cost 6 in this widget.

i. The target location is in $WD^2_<$. Since λ_1^* suggests to P_1 to correctly simulate instructions till c^{max} is attained and also inside Abort, $WD^2_<$ must have been entered from a valuation $(x_{new}, 1 - x_{old} + x_{new}, 0)$ such that $x_{old} = 2x_{new}$. In this case, P_2 incurs a cost 6, which means $u_2(outcome(\lambda_1^*, \lambda_2)) = 6$ contradicting the assumption.

ii. The target location is in $WD^2_>$ or $WD^3_<$ or $WD^3_>$ or $WI^3_<$ or $WI^3_>$ or $WI^2_<$ or $WI^2_>$. In all these cases, by choice of λ_1^*, P_1 correctly simulates the instructions and hence P_2 incurs a cost of 6, which contradicts the assumption.

Therefore, $u_2(outcome(\lambda_1^*, \lambda_2^*)) \leq u_2(outcome(\lambda_1^*, \lambda_2))$ for all strategies $\lambda_2 \neq \lambda_2^*$.

2. Assume that $\mathcal{M}$ visits infinitely many distinct configurations. That is, $V_{\mathcal{M}}$ is infinite. Assume further that there exists a NE (λ_1', λ_2') in the outcome of which P_1 incurs a cost bounded above by 6; that is, $u_1(outcome(\lambda_1', \lambda_2')) < 6$. The cost being bounded, the run which is in the outcome of (λ_1', λ_2') must end in a target location. We do a case analysis on the various target locations and in each case, prove that (λ_1', λ_2') cannot be a NE such that $u_1(outcome(\lambda_1', \lambda_2')) < 6$.

(a) The run ends in a target location of $WI^2_>$. The cost incurred by P_1 is $3(2x_{new} - x_{old} + 2)$ which by assumption is < 6. Then, $2x_{new} < x_{old}$.

This implies that the cost incurred by P_2 is $3(x_{old} - 2x_{new} + 2) > 6$. Now consider a strategy λ_2 for P_2 ($\lambda_2 \neq \lambda_2'$) which suggests that P_2 enter $WI^2_<$ instead of $WI^2_>$. Then the cost incurred by P_2 is $3(2x_{new} - x_{old} + 2)$ which is < 6 by the condition $2x_{new} < x_{old}$. Then, $u_1(outcome(\lambda_1', \lambda_2)) < u_1(outcome(\lambda_1', \lambda_2'))$ which means that (λ_1', λ_2') is not an NE.

(b) The run ends in a target location of $WI^2_<$. The cost incurred by P_1 is $3(x_{old} - 2x_{new} + 2)$ which by assumption is < 6. Then, $x_{old} < 2x_{new}$. This implies that the cost incurred by P_2 is $3(2x_{new} - x_{old} + 2) > 6$. Now consider a strategy λ_2 for P_2 ($\lambda_2 \neq \lambda_2'$) which suggests that P_2 enter $WI^2_>$ instead of $WI^2_<$. Then the cost incurred by P_2 is $3(x_{old} - 2x_{new} + 2)$ which is < 6 by the condition $x_{old} < 2x_{new}$. Then, $u_1(outcome(\lambda_1', \lambda_2)) < u_1(outcome(\lambda_1', \lambda_2'))$ which means that (λ_1', λ_2') is not an NE.

(c) The cases of $WI^3_<$ and $WI^3_>$, $WD^2_<$, $WD^2_>$, $WD^3_<$ and $WD^3_>$ are similar.

(d) The target location is F in Abort. Then we know that $u_2(outcome(\lambda_1', \lambda_2'))$ $= 6$.

i. Assume that λ_1' is a strategy by which P_1 does not execute all instructions of $\mathcal{M}$ correctly. Let λ_2 be a strategy which asks P_2 to enter a widget ($WI^2_<$, $WI^2_>$, $WI^3_<$, $WI^3_>$, $WD^2_<$, $WD^2_>$, $WD^3_<$ or $WD^3_>$) after the first increment/decrement that P_1 has made an error on (based on $x_{new} < \frac{x_{old}}{2}$ for $WI^2_<$ and so on for each widget). Then the cost incurred by P_2 is < 6 (For example, if P_2 entered $WI^2_>$, its cost is $3(x_{old} - 2x_{new} + 2)$ which is less than 6 since $x_{new} > \frac{x_{old}}{2}$). Thus, $u_2(outcome(\lambda_1', \lambda_2)) < 6 < u_2(outcome(\lambda_1', \lambda_2'))$, which implies that (λ_1', λ_2') is not an NE.

ii. Assume that λ_1' is a strategy by which P_1 executes all instructions of $\mathcal{M}$ correctly. Let c_1, c_2 be the counter values when P_1 enters Abort. On reaching F, $u_1(outcome(\lambda_1', \lambda_2')) = 5 + \frac{1}{2^{c_1+c_2+1}}$. As $V_{\mathcal{M}}$ is infinite, there exists $(c_1', c_2') \in V_{\mathcal{M}}$ such that $c_1' + c_2' > c_1 + c_2$. Let λ_1 be a strategy which suggests P_1 enter Abort after correctly simulating instructions of $\mathcal{M}$ until the counter values sum up to $c_1' + c_2'$, and then to correctly simulate increments/decrements inside Abort. Then $u_1(outcome(\lambda_1, \lambda_2')) = 5 + \frac{1}{2^{c_1'+c_2'+1}} < u_1(outcome(\lambda_1', \lambda_2')) = 5 + \frac{1}{2^{c_1+c_2+1}}$, which implies that (λ_1', λ_2') is not an NE.

Thus, we have shown that for a given constant $B = 6$, $\mathcal{M}$ visits finitely many distinct configurations iff there exists a NE in the outcome of which P_1 has a cost bounded above by B. □

Existence of Leader Equilibrium

We use the same game graph construction $\mathcal{G}$ and try to prove this undecidability using 3 clocks using the same simulation as done in the previous section. Note that the only change in the theorem statement is that we prove that there exists a Leader Equilibrium in the outcome of which P_1 has a cost bounded above by 6 (a constant positive integer) iff the two counter machine $\mathcal{M}$, that the game simulates visits only finite number of distinct configurations.

Note that there are 2 cases to work out here, the case when P1 is the leader and the case where he isn't.

1. Assume that $\mathcal{M}$ visits finitely many distinct configurations. We saw in the previous section that there exists a strategy profile $\lambda = (\lambda_1^*, \lambda_2^*)$ such that

$$u_1(outcome(\lambda_1^*, \lambda_2^*)) = 5 + \frac{1}{2^{c^{max}+1}},\ u_2(outcome(\lambda_1^*, \lambda_2^*)) = 6$$

   - **If P_1 is the leader**

     Given the above strategy, it can be seen that P_2 can't improve his cost by unilaterally deviating from the strategy, because the strategy is a Nash Equilibrium. Hence, the strategy is a leader strategy profile.
     It can also be seen that P_1 can have a cost of less than 6 iff he goes to a final state in an *Abort* module. And among all the *Abort* modules, his best cost would be in the one which has $c_1 + c_2 = c^{max}$. Hence, the above strategy is a leader equilibrium as well.

   - **If P_2 is the leader**

     In the strategy profile mentioned, which is a Nash Equilibrium, P_1 can't unilaterally deviate to improve his cost. Hence, the strategy is a leader strategy profile. Also, since P_1 simulates everything correctly, the best cost that P_2 can get is 6, and hence the strategy given is a leader equilibrium. Note that P_2 can't suggest P_1 to perform a wrong simulation and then catch him as this will be strictly lossy for P_1 and he won't agree to such a strategy.

2. Assume that $\mathcal{M}$ visits infinitely many distinct configurations. Also assume that there exists a Leader Equilibrium $\lambda = (\lambda_1', \lambda_2')$ in the outcome of which, the cost of P_1 is bounded above by 6; that is, $u_1(outcome(\lambda_1', \lambda_2')) < 6$. In both the cases, when P_1 is a leader and when he is not, the cost of either P_1 or P_2 can be reduced by a unilateral deviation by the respective player, as was seen in the previous section. Hence, (λ_1', λ_2') can't be a leader equilibrium.

Thus, we have shown that for a given constant $B = 6$, $\mathcal{M}$ visits finitely many distinct configurations iff there exists a Leader Equilibrium in the outcome of which P_1 has a cost bounded above by B.

□

Existence of Incentive Equilibrium

Again, the same game construction is used, and we seek to prove that there exists an Incentive Equilibrium in the outcome of which, P_1 has a cost bounded above by 6 iff the two counter machine $\mathcal{M}$, that the game simulates visits only finite number of distinct configurations.

1. Assume that $\mathcal{M}$ visits finitely many distinct configurations. We saw in the previous section that there exists a strategy profile $\lambda = (\lambda_1^*, \lambda_2^*)$ such that

$$u_1(outcome(\lambda_1^*, \lambda_2^*)) = 5 + \frac{1}{2^{c^{max}+1}},\ u_2(outcome(\lambda_1^*, \lambda_2^*)) = 6$$

The incentives given in the strategy profile will be mentioned below, according to the case when P_1 is the leader, or when he isn't.

   – **If P_1 is the leader**

     Assume that the strategy has $\iota(P_2) = 0$. Given the above strategy, it can be seen that P_2 can't improve his cost by unilaterally deviating from the strategy, because the strategy is a Nash Equilibrium. Hence, the strategy is an incentive strategy profile.
     It can also be seen that P_1 can have a cost of less than 6 iff he goes to a final state in an *Abort* module. And among all the *Abort* modules, his best cost would be in the one which has $c_1 + c_2 = c^{max}$. He can't improve his cost by changing the strategy of P_2 and hence, any incentive given for the current strategy will only add to his cost, and hence, the given strategy profile is an incentive equilibrium.

   – **If P_2 is the leader**

     Assume that the strategy has $\iota(P_1) = 0$. In the strategy profile mentioned, which is a Nash Equilibrium, P_1 can't unilaterally deviate to improve his cost. Hence, the strategy is an incentive strategy profile. Note that for P_2 to improve his cost, he would have to make P_1 deviate from his correct simulation strategy, and would have to catch him at some wrong simulation to drive the cost of P_1 beyond 6 and his own cost below. It can be seen that if the cost of P_1 is $6 + \delta$, then the cost of P_2 will be $6 - \delta$. To make P_1 follow this strategy, an incentive has to be given so that the cost of P_1 is at most the same as the cost he got in the previous strategy of correct simulation. This incentive equals $\delta + 1 - \frac{1}{2^{c^{max}+1}}$. This, when added to P_2's cost of $6 - \delta$ will be greater than 6 and hence, won't benefit P_2. Hence, the strategy $(\lambda_1^*, \lambda_2^*)$ with the incentive $\iota(P_1) = 0$, is an incentive equilibrium.

2. Assume that $\mathcal{M}$ visits infinitely many distinct configurations. Also assume that there exists an Incentive Equilibrium $\lambda = (\lambda_1', \lambda_2')$ in the outcome of which, the cost of P_1 is bounded above by 6; that is, $u_1(outcome(\lambda_1', \lambda_2')) < 6$.

In both the cases, when P_1 is a leader and when he is not, the cost of either P_1 or P_2 can be reduced by a unilateral deviation by the respective player, as was seen in the previous section. Hence, (λ_1', λ_2') can't be an incentive equilibrium.

Thus, we have shown that for a given constant $B = 6$, $\mathcal{M}$ visits finitely many distinct configurations iff there exists an Incentive Equilibrium in the outcome of which P_1 has a cost bounded above by B.

$\square$

6 Timed Multiplayer Mean Pay-off Games : a digression

In this section, we discuss another kind of multiplayer timed game which might be interesting to discuss.

The timed multiplayer mean pay-off game structure is defined as a tuple $\mathcal{G} = (L, L_0, X, Z, P, E, \eta, C)$ where, L is a finite set of locations, $L_0 \subseteq L$ is the singleton set containing the initial location; X is a finite set of clocks; P is the set of n players $\{P_1, P_2 \ldots P_n\}$ Z is a set of n mean-cost variables, where n is the number of players, such that Z_i store the $mean\,cost = \frac{cost\,accumulated}{total\,time\,elapsed}$ by P_i; $E \subseteq L \times L \times \Theta(X) \times 2^X$ is the set of transitions, where $\Theta(X)$ is the set of all intervals for each of the clocks from which constraints on edge transitions are decided and 2^X corresponds to the set of clocks which are reset on taking that transition. $\eta : L \rightarrow \Theta(X)$ is a function assigning clock valuation invariants to each location; $C : L \rightarrow \mathbb{N}^n$ is a function associating cost growth rate to each of the players on the game locations. The game is then defined on the above structure as : The set of locations is partitioned into $L_1, L_2, \ldots L_n$ where L_i is the set of locations which belong to the player P_i. At each location, the owner of that location chooses an edge $e_i \in E$ and a time delay t_i. Suppose from a state $q = (l, \nu, \mu), l \in L_i$, player P_i chooses edge e_i and time delay t_i, then there exist states q', q'' such that $q = (l, \nu, \mu) \xrightarrow{t_i} q' = (l, \nu + t_i, \mu') \xrightarrow{e_i} q'' = (l'', \nu'', \mu'')$. where the state transitions are as defined in Section 2.1, with a slight difference that as the edge $\xrightarrow{e_i}$ is taken, there is no further cost incurred for any of the players. Also note that here, $\nu \models \eta(l)$ and $\nu'' \models \eta(l'')$ and $\nu + t_i$ satisfies the transition edge constraints on clock valuations.

A strategy for a player P_i is a function $\lambda_i : S \rightarrow (R^+ \times E_i) \cup \{\infty\}$ (where S is the set of all possible values of (l_i, ν, μ) where $\nu \models \eta(l_i)$ and $l_i \in \text{Ł}_i$, close to the definition in Section 2.1), such that if $\lambda_i(q) = (t_i, e_i)$, then it possible to incur a delay of t_i at a state q, and then take a discrete transition e_i. Note that a delay t_i of ∞ is allowed if $\eta(l_i)$ allows it. A strategy profile is an n-tuple $(\lambda_1, \lambda_2 \ldots \lambda_n)$ of strategies where λ_i is a strategy for player P_i. A run ρ is said to be played according to a strategy profile if for each node l_i that the game visits, the outgoing edge e_i and the time delay t_i are chosen according to λ_i (assuming $l_i \in L_i$). ρ in the above case, is said to be the outcome of λ and is denoted as $outcome(\lambda)$. Note that any run ρ according to strategy λ will be of infinite length (in time). We also define the final accumulated costs for each of the players, as a function of the run, such that $u_i(\rho)$ is the final accumulated cost for player P_i. An n-tuple of all these costs for a given run, $u(\rho) = (u_1(\rho), u_2(\rho) \ldots u_n(\rho))$. The strategies we consider are *memoryless*, since we only look at the current state to decide the next move. A *terminal history* is a run ρ starting from the initial location, ending in a target location and never passing through a target location in between. Given a terminal history ρ, the payoff of player i is $-u_i(\rho)$ where $u_i(\rho)$ is the cost accumulated along ρ. For a run ρ that does not end in a target location, the payoff for both players is ∞. The *objective* of each player is to minimize his mean-cost over the run, that is, in the lim $t \to \infty$. These games are therefore non-competetive, since neither player aims to increase the others' cost.

Examples

In the example figure 6.1 , player 1 owns node l_1, player 2 owns node l_2 and player 3 owns nodes l_3, l_4 and l_5. In the example given, location i has been represented as l_i and the edge between locations l_i and l_j as e_{ij}. Note that in the examples below, the strategy we've mentioned is independent of the initial clock valuations when the node is arrived at (because the clock is being reset after each edge). Also, the strategy mentioned is independent of the cost accumulated by each of the players. Hence, in the examples given below, we change our notation for strategy λ slightly and is now represented as $\lambda(l)$ instead of $\lambda(l, \nu, \mu)$.

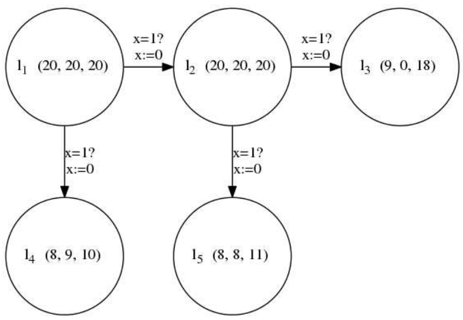

Fig. 6.1. A multiplayer timed mean pay-off game

Nash Equilibrium

The following strategy is a Nash Equilibrium.

$$\lambda_1(l_1) = (1, e_{14}),\ \lambda_3(l_4) = \infty,\ \lambda_2 \text{ need not be specified.}$$

This gives us:

$$u(outcome(\lambda)) = (8, 9, 10)$$

The values are calculated as(an example of $u_1(outcome(\lambda))$ is shown below:

$$u_1(outcome(\lambda)) = \lim_{t \to \infty} \frac{20 + 8(t-1)}{t} = 8$$

Leader Equilibrium

If player 2 is the leader, the following strategy is a Leader Equilibrium:

$$\lambda_1(l_1) = (1, e_{12}),\ \lambda_2 = (1, e_{25}),\ \lambda_3(l_5) = \infty$$

This gives us

$$u(outcome(\lambda)) = (8, 8, 11)$$

Incentive Equilibrium

Note that for an incentive strategy profile, the function $\iota : P - P_l \to R$, where P_l is the leader, is added to the strategy profile. Here, the leader offers to take a certain amount of mean cost from a player if the player follows the suggested strategy each time he visits some node. This cost is given by $\iota(P_i)$ for a player P_i.

Again assume that player 2 is the leader. The following incentive strategy profile is an incentive equilibrium:

$$\lambda_1(l_1) = (1, e_{12}),\ \lambda_2(l_2) = (1, e_{12}),\ \lambda_3(l_3) = \infty$$
$$\iota(P_1) = 1\ ,\ \iota(P_3) = 0$$

This gives us:

$$u(outcome(\lambda)) = (8, 1, 18)$$

7 Future Work

The two player timed reachability game with one clock has a solution for finding equilibria. This is an ongoing work, but the existence of the solution has been determined. We believe that the multiplayer reachability game can also be solved to find the different kinds of equilibria. We hope to find the solution of this using the strategy used in [2] for incentive equilibrium and [3] for leader equilibrium. We also hope to build an implementation of the solution, which could take in an input game and output strategy profiles which are NE, LE or IE.

References

1. R. Alur and D. L. Dill. A Theory of Timed Automata. *Theoretical Computer Science*, 126(2), 183-235, 1994.
2. Anshul Gupta , M. S. Krishna Deepak , Bharath Kumar Padarthi, Sven Schewe , and Ashutosh Trivedi. Incentive Equilibria in Mean-payoff Games *LIPICS*
3. Anshul Gupta and Sven Schewe. Quantitative Verification in Rational Environments. *21st International Symposium on Temporal Representation and Reasoning, TIME 2014* , 123-131, 2014
4. Shankara Narayanan Krishna, Lakshmi Manasa, Ashish Chiplunkar. Nash Equilibrium in Weighted Concurrent Timed Games with Reachability Objectives *Distributed Computing and Internet Technology - 8th International Conference, ICDCIT 2012* , 117-128, 2012
5. T. Brihaye, V. Bruyère, and J. Raskin. Model-Checking for Weighted Timed Automata. *Proceedings of FORMATS/FTRTFT'04*, LNCS 3253, pp 277-292, 2004.