Simulation is Decidable for One-counter Nets

(Extended Abstract)

Parosh Aziz Abdulla
Dept. of Computer Systems
P.O. Box 325
S-751 05 Uppsala, Sweden
parosh@docs.uu.se

Kārlis Čerāns
Institute of Mathematics
and Computer Science
University of Latvia
karlis@cclu.lv

Abstract

We prove that the simulation preorder is decidable for the class of one-counter nets. A *one-counter net* consists of a finite-state machine operating on a variable (counter) which ranges over the natural numbers. Each transition can increase or decrease the value of the counter. A transition may not be performed if this implies that the value of the counter becomes negative. The class of one-counter nets is computationally equivalent to the class of Petri nets with one unbounded place, and to the class of pushdown automata where the stack alphabet is restricted to one symbol. To our knowledge, this is the first result in the literature which gives a positive answer to the decidability of simulation preorder between pairs of processes in a class whose elements are neither finite-state nor allow finite partitioning of their state spaces.

1 Introduction

Several criteria for comparison of program behaviour have been introduced in the literature during the past few years. Examples are the notions of bisimulation equivalence and simulation preorder which are refinements of trace equivalence and trace inclusion in language theory. One area of interest has been to design algorithms to verify such properties both for finite-state systems and infinite-state systems, e.g. real-time automata [Čer92, LY93], data-independent systems [JP93], relational automata ([Čer94, AČJYK96]), and lossy channel systems [AK95, AČJYK96].

Classes of infinite-state systems to which a considerable research effort has been devoted are those of pushdown processes, context-free processes and Petri nets. The decidability of bisimulation has been shown for different variants of context-free processes [BBK93, CHS92, CHM93]. In [GH94] the simulation preorder is shown to be undecidable for context-free processes. In [Sti96] bisimulation was shown to be decidable for *normed* pushdown processes. Simulation and bisimulation properties have also been studied for Petri nets [Jan90, Jan95, JE96, Esp95, JM95, Jan97]. In [Jan95], both simulation and bisimulation are shown to be undecidable for Petri nets, assuming that the nets have at least two unbounded places. This result left open the problem of whether simulation and bisimulation are decidable for Petri nets with one unbounded place. This class of Petri nets is interesting also because it is equivalent to the class of pushdown automata where the stack alphabet is restricted to one symbol. In this paper, we show that the simulation preorder is decidable in this case. Jančar [Jan97] shows that bisimulation is decidable for nets with one unbounded place. Thus, our results contribute to delineating the border between decidable and undecidable problems in the context of infinite-state systems.

We consider the problem of deciding simulation for a class of infinite-state systems which we call *one-counter nets*. A *one-counter net* consists of a finite-state machine operating on a variable (counter) which ranges over the natural numbers. Each transition can increase or decrease the value of the counter. A transition may not be performed if this implies that the value of the counter becomes negative. A *configuration* γ of a net is a pair $\langle q, x\rangle$ where q is a state of the finite-state machine and x is the value of the counter. We show that given nets N_1 and N_2, and configurations γ_1 and γ_2 in N_1 and N_2, it is decidable whether γ_1 is simulated by γ_2. The class of one-counter nets is computationally equivalent to the class of Petri nets with one unbounded place, and the class of pushdown automata where the stack alphabet is restricted to one symbol[1].

To perform our decidability proof we notice that the simulation relation is upward closed with respect to the counter of N_2: if $\langle q, x\rangle$ is simulated by $\langle r, y\rangle$ and $y \leq y'$ then $\langle q, x\rangle$ is simulated[2] by $\langle r, y'\rangle$. In fact our decidability proof consists of two steps. The first step which is the crucial step and the most difficult one shows that the simulation relation satisfies a much stronger property than upward closedness, namely the following. Given nets N_1 and N_2 and states q and r in N_1 and N_2 respectively, there is a rational number ρ (which we call the *quality* of $\langle q, r\rangle$) which defines the border between simulation and non-simulation modulo some constant c. More precisely, we prove that $\langle q, x\rangle$ is simulated by $\langle r, y\rangle$ if $\rho \cdot x + c \leq y$ and $\langle q, x\rangle$ is not simulated by $\langle r, y\rangle$ if $\rho \cdot x - c \geq y$. The main idea in proving the existence of qualities is to "decompose" the nets into a finite number of deterministic subnets. The quality of a pair $\langle q, r\rangle$ is first computed in each of these deterministic subnets. We take the quality of $\langle q, r\rangle$ to be the minimum of its qualities in the deterministic subnets.

[1]The fact that the stack alphabet contains only one symbol implies implicitly that emptiness of the stack cannot be checked (corresponding to the fact that we do not perform zero-testing of the counter value). In most models of push-down automata, stack emptiness is implemented by having at least two symbols in the stack alphabet, using one symbol to represent the bottom of the stack.

[2]However, the simulation relation is obviously not upward closed with respect to the counter of N_1. Hence, the relation cannot be characterized by ideals (used e.g. in [AJ93]).

To prove the correctness of our definition of qualities, we describe the simulation relation as a game between two players A (representing N_1) and B (representing N_2). Different game-theoretic descriptions of simulation and bisimulation relations have been given e.g. in [Sti95, JM95]. We design a number of (quite complicated) strategies for the players A and B, which we use to show that qualities indeed satisfy the above property (i.e., they define borders separating counter values which give simulation and non-simulation respectively). In the second (simpler) step of the proof, we show that the existence of qualities implies that each pair of configurations which is related by simulation has a *semi-linear witness* (a semi-linear simulation relation containing the pair) which can be found and checked in a finite amount of time.

In [Jan97] it is shown that also bisimulation can be decided by providing semi-linear witnesses. In fact [Jan97] considers *one-counter machines* which are more general than one-counter nets, in the sense that they allow for zero-testing of the counter value. However, the method used in [Jan97] to prove the existence of bisimulation witnesses is quite different from ours. One crucial technique in [Jan97] is to define a finite set of "bottom configurations" (configurations with low counter values) which characterize the bisimulation relation. If two configurations are bisimilar then they have the same distance (minimum number of transitions) to the bottom configurations. This characterization implies that bisimulation has a "periodic behaviour": the relation between counter values of bisimilar configurations follow a pattern which is repeated at regular intervals. Semi-linear witnesses for bisimulation are derived in [Jan97] from this periodic relation. It is obvious that simulation does not exhibit such a periodic behaviour. In proving the correctness of the decomposition approach, we exploit another kind of regularity, namely the upward closedness property mentioned above.

It turns out that using decomposition to show the existence of simulation witnesses is more complicated than using periodicity to derive bisimulation witnesses. This is not surprising, since for many computation models checking simulation is a more difficult problem than checking bisimulation. For example in the case of finite-state systems, the best known algorithms check simulation in $O(n^2 \log(n))$ and bisimulation in $O(n \log(n))$. For deterministic processes, deciding bisimulation (language equivalence) is obviously reducible to deciding simulation (language inclusion). For context-free processes, bisimulation is decidable [BBK93, CHS92, CHM93], while simulation is undecidable [GH94]. In [Kin97] it is shown that it is decidable to check whether a finite-state system is weakly bisimilar to a lossy channel system, while it is undecidable to check whether a finite-state system is weakly simulated by a lossy channel system. In fact, to our knowledge, this is the first result in the literature where simulation is shown to be decidable for pairs of processes where both processes belong to a class of systems which are not essentially finite-state[3].

Outline In Section 2 we introduce one-counter nets. In Sections 3 we describe the simulation relation as a game between two players A and B. In Section 4 we show how to compute qualities. In Section 5 and Section 6 we define winning strategies for players B and A respectively. In Section 7 we study some properties of the simulation relation. Section 8 contains the main result.

2 One Counter Nets

A *(one-counter) net* consists of a finite labeled transition system operating on a counter which ranges over the natural numbers. Each transition may either increment the value of the counter by one, decrement it by one, or keep its value. In addition, each transition is labeled with an event which represents an observable interaction with the environment.

We use $\mathcal{N}$ and $\mathcal{Q}$ to denote the sets of natural numbers and rational numbers respectively.

[3]A system is *essentially finite-state* if either it is finite-state or it allows a finite partitioning of its states space into sets of states which are equivalent up to bisimulation, e.g. real-time automata.

By $\mathcal{Q}_{\geq 0}^{\infty}$ we mean the set of non-negative rational numbers including ∞.

Formally, a *net* N is a tuple $\langle Q, \Lambda, T\rangle$, where Q is a finite set of *states*, Λ is a finite set of *labels*, and $T \subseteq Q \times \Lambda \times \{-1, 0, +1\} \times Q$ is a finite set of *transitions*. For a transition $t = \langle q_1, \lambda, a, q_2\rangle$, we call q_1, λ, a, q_2 the *source*, *label*, *incrementation*, and *target* of t, and use $source(t)$, $L(t)$, $\Delta(t)$, and $target(t)$, to denote them respectively.

We say that N is *deterministic* if for each $q \in Q$ and $\lambda \in \Lambda$, there is at most one $t \in T$ such that $source(t) = q$ and $L(t) = \lambda$.

A *configuration* γ of N is of the form $\langle q, x\rangle$ where $q \in Q$ and $x \in \mathcal{N}$. We define a transition relation $\Longrightarrow$ as a set of triples $\langle \gamma_1, t, \gamma_2\rangle$, where $\gamma_1 = \langle q_1, x_1\rangle$ and $\gamma_2 = \langle q_2, x_2\rangle$ are configurations and $t \in T$, such that $\langle \gamma_1, t, \gamma_2\rangle \in\Longrightarrow$ if and only if $source(t) = q_1$, $target(t) = q_2$, and $x_2 = x_1 + \Delta(t)$. Notice that since we demand x_2 to be a natural number, the transition may be performed only if $x_1 + \Delta(t)$ is non-negative. We use $\gamma_1 \stackrel{t}{\Longrightarrow} \gamma_2$ to denote that $\langle \gamma_1, t, \gamma_2\rangle \in\Longrightarrow$, and $\gamma_1 \stackrel{\lambda}{\Longrightarrow} \gamma_2$ to denote that $\gamma_1 \stackrel{t}{\Longrightarrow} \gamma_2$ for some $t \in T$ with $L(t) = \lambda$. By $\gamma_1 \Longrightarrow \gamma_2$ we mean that $\gamma_1 \stackrel{t}{\Longrightarrow} \gamma_2$ for some $t \in T$ (or equivalently that $\gamma_1 \stackrel{\lambda}{\Longrightarrow} \gamma_2$ for some $\lambda \in \Lambda$).

For two nets N_1 and N_2, a *simulation relation* $\lhd$ between N_1 and N_2 is a set of pairs of the form $\langle \gamma_1, \gamma_2\rangle$, where γ_1 and γ_2 are configurations of N_1 and N_2 respectively, such that if there are λ and γ_3 with $\langle \gamma_1, \gamma_2\rangle \in \lhd$ and $\gamma_1 \stackrel{\lambda}{\Longrightarrow} \gamma_3$, then there is a γ_4 where $\gamma_2 \stackrel{\lambda}{\Longrightarrow} \gamma_4$ and $\langle \gamma_3, \gamma_4\rangle \in \lhd$. We use $\gamma_1 \lhd \gamma_2$ to denote that $\langle \gamma_1, \gamma_2\rangle \in \lhd$.

We say that γ_1 *is simulated by* γ_2 (written $\gamma_1 \sqsubseteq \gamma_2$) if there is a simulation $\lhd$ between N_1 and N_2 such that $\gamma_1 \lhd \gamma_2$. It is clear that $\sqsubseteq$ is the largest simulation relation between N_1 and N_2.

A restricted version of simulation For a net N, we say that N is *complete* if for each $q \in Q$ and $\lambda \in \Lambda$, there is at least one $t \in T$, such that $source(t) = q$ and $L(t) = \lambda$. We say that N is *partially complete* if for each $q \in Q$ there is at least one $t \in T$, such that $source(t) = q$.

Proposition 2.1 The general problem of deciding simulation between nets N_1 and N_2 can be reduced to a restricted version where N_1 is assumed to be deterministic and partially complete, and N_2 is assumed to be complete.

Proof. The proof can be found in the appendix. $\square$

In the rest of the paper, we consider only the restricted version of the problem.

3 Games

In this section we describe (Lemma 3.1) the problem of finding a simulation relation between a deterministic and partially complete net N_1, and a complete and (possibly) nondeterministic net N_2, as a *game* between two players A and B representing N_1 and N_2 respectively. A *play* of the game is started at two configurations γ_1 of N_1 and γ_2 of N_2. Player A starts by performing a transition, with a label (say λ) from[4] γ_1, while player B answers by performing a transition with the same label λ from γ_2. Then, player A performs the next transition, and so on. Player A wins the play, if we reach a point where B can no longer answer the moves of A. Since N_2 is complete, this can only happen if the counter of B is zero (and is about to become negative). On the other hand, B wins the play if either both players can continue for ever, or if we reach a point where A can not make any further moves. Since N_1 is partially complete, the latter case occurs only if the counter of A is zero (and is about to become negative).

[4]Since N_1 is deterministic, the transition is uniquely determined by λ.

Formally, a *(game) graph* G of a deterministic and partially complete net $N_1 = \langle Q_1, \Lambda, T_1\rangle$, and a complete and (possibly) nondeterministic net $N_2 = \langle Q_2, \Lambda, T_2\rangle$, is of the form $\langle V, \Lambda, T\rangle$, where $V = Q_1 \times Q_2$ is the set of *vertices* of G, and $T \subseteq T_1 \times T_2$ is the set of *transitions*. Furthermore, $\langle t_1, t_2\rangle \in T$ if and only if $L_1(t_1) = L(t_2)$. For a transition[5] $t = \langle t_1, t_2\rangle$, we define $source(t) = \langle source(t_1), source(t_2)\rangle$, $target(t) = \langle target(t_1), target(t_2)\rangle$, $L(t) = L(t_1)$, $\Delta_x(t) = \Delta(t_1)$, and $\Delta_y(t) = \Delta(t_2)$. A *configuration* γ of G is of the form $\langle\langle q, r\rangle, x, y\rangle$, where $\gamma_1 = \langle q, x\rangle$ and $\gamma_2 = \langle r, y\rangle$ are configurations of N_1 and N_2. Sometimes we write γ as $[\gamma_1, \gamma_2]$. We use $[\gamma_1, \gamma_2] \stackrel{t}{\Longrightarrow} [\gamma_1', \gamma_2']$ to denote that there are transitions $t_1 \in T_1$ and $t_2 \in T_2$ such that $t = \langle t_1, t_2\rangle$, $\gamma_1 \stackrel{t_1}{\Longrightarrow} \gamma_1'$, and $\gamma_2 \stackrel{t_2}{\Longrightarrow} \gamma_2'$. By $\gamma_1 \Longrightarrow \gamma_2$ we mean that $\gamma_1 \stackrel{t}{\Longrightarrow} \gamma_2$ for some $t \in T$.

A *path* π in G is a finite sequence $v_0 t_1 v_1 \cdots t_m v_m$ (sometimes, when $m \geq 1$, written simply as $t_1 \cdots t_m$), where $source(t_{i+1}) = target(t_i) = v_i$. We define two functions Δ_x and Δ_y on π, where $\Delta_x(\pi) = \sum_{i=1}^{m} \Delta_x(t_i)$ and $\Delta_y(\pi) = \sum_{i=1}^{m} \Delta_y(t_i)$. Intuitively, the function $\Delta_x(\pi)$ ($\Delta_y(\pi)$) gives the change in value of the counter of A (B) when the sequence $t_1, \ldots, t_m$ of transitions is performed. We use $|\pi|$ to denote m. We define *first*(π) to be v_0 and *last*(π) to be v_m. We say that π is a v-*path* if *first*$(\pi) = v$. We say that $\langle v, \lambda\rangle$ is *on* π, (denoted $\langle v, \lambda\rangle \in \pi$) if $v = v_i$ and $\lambda = L(t_{i+1})$ for some $i : 1 \leq i < m$. We say that v is *on* π (denoted $v \in \pi$), if $v = v_i$ for some $i : 1 \leq i \leq m$. We define $t \in \pi$ in a similar manner. Two paths π_1 and π_2 are said to be *compatible* if *last*$(\pi_1) =$ *first*(π_2). For compatible paths $\pi_1 = v_0 t_1 \cdots t_m v_m$ and $\pi_2 = v_m t_{m+1} \cdots t_n v_n$, we use $\pi_1 \bullet \pi_2$ to denote $v_0 t_1 \cdots t_m v_m t_{m+1} \cdots t_n v_n$. By $\pi' \preceq \pi$, we mean that there is a π'' such that $\pi = \pi' \bullet \pi''$.

A *play* P is an infinite sequence $v_0 t_1 v_1 t_2 \cdots$ such that $source(t_{i+1}) = target(t_i) = v_i$ for each $i \geq 0$. Notice that each prefix of P is a path. We use $P(i)$ to denote the path $v_0 t_1 v_1 \cdots t_i v_i$. We say that P is *winning for* A *from a configuration* $\gamma = \langle v, x, y\rangle$ if $v_0 = v$ and there is a $k \geq 0$ such that $y + \Delta_y(P(k)) < 0$ and $x + \Delta_x(P(j)) \geq 0$ for each $j : 0 \leq j \leq k$. Intuitively, this means that we start from γ and reach a configuration where B violates (at least one step before A) the condition of keeping a non-negative counter value. We say that P is *winning for* B *from* γ if we start from γ and P is not winning for A from γ. Notice that P is winning for B from γ if either (i) $x + \Delta_x(P(k) \geq 0$ and $y + \Delta_y(P(k) \geq 0$ for each $k \geq 0$; or (ii) there is a k such that $x + \Delta_x(P(k)) < 0$ and $y + \Delta_y(P(j)) \geq 0$ for each $j : 1 \leq j < k$. Intuitively, condition (i) means that both counters remain non-negative during the entire play, and condition (ii) means that the counter of A becomes negative (at the same time or before B). Observe[6] that each step (transition) of P describes a move from A (picking a label and performing a transition with that label in N_1) and an answer from B (picking and performing a transition in N_2 having the same label).

A *strategy for player* A (or simply an A-*strategy*) str_A is a mapping such that, for natural numbers x and y and a path π, $str_A(x, y)(\pi) = \lambda$ for some label λ. Intuitively, str_A defines the moves of A: given counter values (x and y) from which the play is started and the progress of the play up to now (the path π), str_A provides the next move of A (the label λ chosen by A). A B-*strategy* str_B is a mapping such that, for natural numbers x and y, a path π, and label λ, $str_B(x, y)(\pi)(\lambda) = t$ for some transition t with $source(t) = last(\pi)$ and $L(t) = \lambda$. Intuitively, str_B defines the answers of B: given the initial counter values (x and y), the progress of play up to now (the path π), and the latest move of A (the label λ), str_B picks a transition with a label λ. For a play $P = v_0 t_1 v_1 t_2 \cdots$ and an A-strategy str_A, we say that P *follows* str_A from a configuration $\gamma = \langle v, x, y\rangle$ if we start from γ and all the

[5]Here, we abuse notation, in the sense that we use $t, t_1, t_2, \ldots$ to denote transitions both in nets and graphs.

[6]Often in the literature, a game graph is described as a bipartite graph with "A-nodes" and "B-nodes". The A-nodes (B-nodes) are those from which A (B) is about to make his next move. A transition leads either from an A-node to a B-node, or conversely. In our case, since A is deterministic and we use labels, a single transition in the game graph simulates two steps, namely A making a move and B giving an answer. Hence, we do not need to work with bipartite graphs.

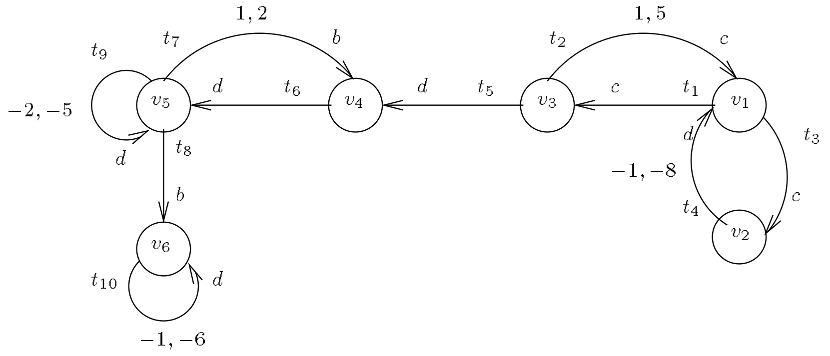

Figure 1: A game graph G

moves of A in P obey str_A. Formally, P *follows* str_A *from* γ if $P(0) = v$ and for each $k > 0$ we have $L(t_k) = str_A(x, y)(P(k-1))$. In a similar manner P *follows a* B*-strategy* str_B *from* γ if we start from γ and all the moves of B in P obey str_B, i.e., for each $k > 0$ we have $t_k = str_B(x, y)(P(k-1))(L(t_k))$. An A-strategy str_A is said to be *winning* from γ, if each play P is winning for A whenever P follows str_A from γ. A *winning* B*-strategy* is defined in a similar manner. We say that A (B) is *winning at* γ if there is a winning A-strategy (B-)strategy from γ.

We recall the following standard result which describes the simulation relation in terms of existence of winning A-strategies and winning B-strategies.

Lemma 3.1 Consider nets N_1 and N_2 and configurations γ_1 and γ_2 in N_1 and N_2, respectively. Let G be the graph of N_1 and N_2, and let $\gamma = [\gamma_1, \gamma_2]$. If B is winning at γ then $\gamma_1 \sqsubseteq \gamma_2$, and if A is winning at γ then $\gamma_1 \not\sqsubseteq \gamma_2$.

Now we characterize configurations from which there exists winning strategies for A or B. The crucial result is that we are able to define *qualities* for vertices. We assign to every vertex $\langle q, r\rangle$ its *quality* which is a ratio between counter values of A and B. This ratio provides a borderline (modulo some fixed constant) between the existence of a winning A-strategy or a winning B-strategy from v. More precisely we show the following lemma.

Lemma 3.2 There is a positive $c \in \mathcal{Q}$ such that for every vertex v in G there exists a $\rho \in \mathcal{Q}^{\infty}_{\geq 0}$, called the *quality* of v in G, where both:

1. if $\rho < \infty$ then B is winning at $\langle v, x, y\rangle$ whenever $\rho \cdot x + c \leq y$; and
2. if $\rho > 0$ then A is winning at $\langle v, x, y\rangle$ whenever either
   - $0 < \rho < \infty$ and $\rho \cdot x - c \geq y$; or
   - $\rho = \infty$ and $x \geq c$.

Proof. In Section 4.3 (Definition 4.1) we give a definition $Qual(v)$ of the quality of a vertex v. In Section 5 (Lemma 5.3) we show that $Qual$ satisfies the first condition above. In Section 6 (Lemma 6.5) we show that $Qual$ satisfies the second condition. □

Observe that if $\rho = 0$, then condition 1 implies that B is winning whenever $c \leq y$.

Note In the sequel we assume that we work with a given graph $G = \langle V, \Lambda, T\rangle$.

Example In Figure 1 we show a graphic representation of a game graph with 6 vertices and 10 transitions. The set of labels is $\{b, c, d\}$. We show only counter incrementations[7] which are different from 0, so, for example, t_4 decreases the first counter by 1 and decreases the second counter by 8, while t_5 does not change any of the counters. One path in G is $\pi = t_4 t_1 t_2 t_1 t_5$. We have $\Delta_x(\pi) = 0$ and $\Delta_y(\pi) = -3$.

4 Qualities

In this section we describe (Definition 4.1) a method for computing qualities of vertices. First, we introduce the notions of *deterministic subgraphs* and *loops*. A *deterministic subgraph D of G* generates a "deterministic behaviour" of player B. When the game is played according to D, then for each vertex v and label λ there is an *a priori* fixed transition t, with a source v and a label λ. Whenever the play reaches the vertex v and the next move of A is to pick λ, then B answers by picking the predefined choice t. The strategy of B will be to choose and play according to deterministic subgraphs which change his counter most favorably compared to the counter of A (perhaps playing according to different subgraphs during the different stages of the play). When player B moves according to a certain deterministic subgraph D, player A tries, through picking the "right" labels, to find paths through D which are most advantageous to him. A special type of paths which we show to be sufficient to define optimal strategies for A are *loops*. In fact, we will derive (Definition 4.1) the quality of each vertex v from the set of loops which are reachable from v in the different deterministic subgraphs of G. More precisely, first we compute qualities of vertices in the deterministic subgraphs of G by considering the sets of loops which are reachable from the vertex. Then, we define the quality of a vertex to be the minimum of its qualities in the subgraphs.

4.1 Deterministic Game Graphs

For a graph $G = \langle V, \Lambda, T \rangle$, a *deterministic subgraph D* of G is of the form $\langle V, \Lambda, T' \rangle$, such that $T' \subseteq T$ and, for each v and λ, it is the case that there is a $t \in T$, with $source(t) = v$ and $L(t) = \lambda$, if and only if there is exactly one such a transition in T'. Intuitively, we get a deterministic subgraph of G by removing, for each vertex v and label λ, all outgoing transitions from v labeled by λ, except one. We write $t \in D$ to denote that $t \in T'$. We let $\Box G$ denote $|Det|$, where Det is the set of deterministic subgraphs of G. We let $D[v, \lambda]$ denote the transition $t \in D$, such that $source(t) = v$ and $L(t) = \lambda$ (observe that $D[v, \lambda]$ may be undefined). We use $\Diamond G$ to denote the number of transitions in each deterministic subgraph of G. Observe that this number is the same in all deterministic subgraphs of G.

A path π of G is said to be in D (written $\pi \in D$) if $t \in D$ for each transition t in π. For $\pi \in D$, we use $v \leadsto_{\pi,D} v'$ to denote that $first(\pi) = v$ and $last(\pi) = v'$.

In Figure 2, we show the four deterministic subgraphs of the graph in Figure 1. Each of these subgraphs is obtained according to the definition by deleting one of t_1 and t_3, and one of t_7 and t_8.

In the sequel we assume that the set of deterministic subgraphs of G is given by Det.

4.2 Loops

For a vertex v_1, a v_1-*loop* is a path consisting of the concatenation of a *head* and a *tail*. The head leads from v_1 to a vertex v_2. The tail forms a cycle leading from v_2 back to v_2 (without

[7]To make Figure 1 more compact, we allow transitions to increment or decrement the counters by more than 1. Obviously, each such a transition can be simulated in our model by a sequence of transitions (each changing the counter value by at most one).

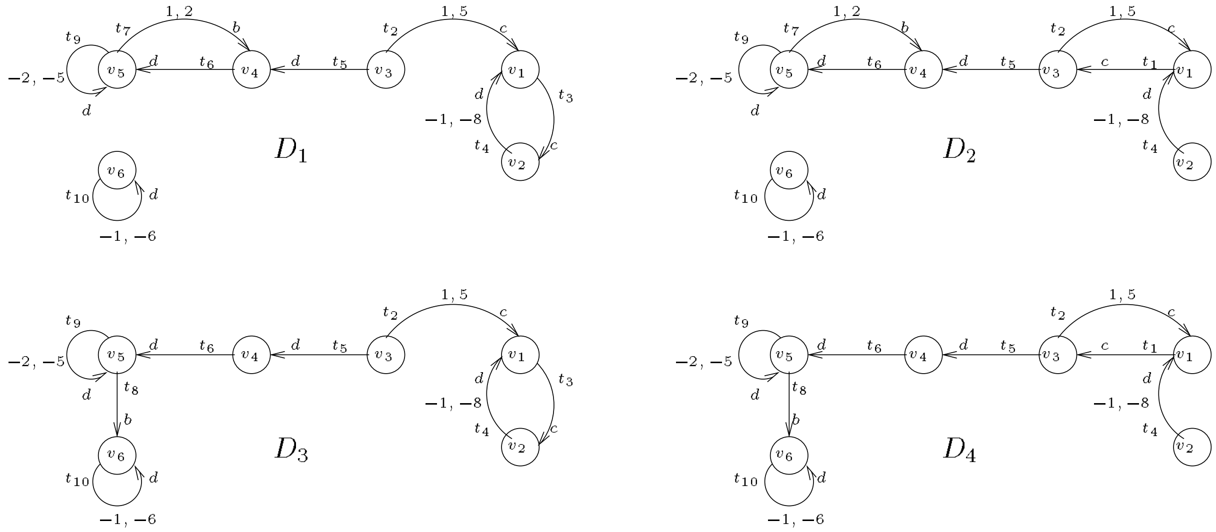

Figure 2: The deterministic subgraphs of the game in Figure 1

passing any other vertex more than once). Formally, we define loops as follows. A path ψ is said to be *simple* if all its vertices are different. A *loop* θ is a path of the form $\psi t v$ where ψ is simple and $v \in \psi$. A *simple loop* ϕ is a path of the form $\psi t v$ where ψ is simple and $v = \mathit{first}(\psi)$. Observe that each loop θ can be written as $\psi \bullet \phi$, where ψ is a simple path (the *head*), ϕ is a simple loop (the *tail*), $\mathit{last}(\psi) = \mathit{first}(\phi) = \mathit{last}(\phi)$, and all the other vertices in ψ and ϕ are different. Notice that there are finitely many loops in each game.

For a simple loop ϕ, we let ϕ^m denote the concatenation of m copies of ϕ. For a path π and a loop $\theta = \psi \bullet \phi$, we say that π is an *unfolding* of θ if $\pi \preceq \psi \bullet \phi^m$ for some $m \geq 0$. For $v \in \theta$, we define $\mathit{next}(\theta)(v) = \lambda$, where $\langle v, \lambda \rangle \in \theta$. Intuitively, an unfolding π of θ is a path generated by "iterating" θ, i.e., first traversing the head of θ and then traversing the tail of θ an arbitrary number of times. The function *next* defines the "next step" in the iteration of θ: if we are iterating θ and we have reached the vertex v, then the next step is to choose the label λ.

As mentioned above one crucial aspect in the definition of strategies for players A and B is the presence (or absence) of certain types of loops. The *type* of a loop is decided by the manner it changes the values of counters. We consider four types of loops, namely a loop θ is $\langle +,+ \rangle$ if $\Delta_x(\phi) > 0$ and $\Delta_y(\phi) \geq 0$; $\langle +,- \rangle$ if $\Delta_x(\phi) \geq 0$ and $\Delta_y(\phi) < 0$; $\langle -,+ \rangle$ if $\Delta_x(\phi) \leq 0$ and $\Delta_y(\phi) \geq 0$; and $\langle -,- \rangle$ if $\Delta_x(\phi) < 0$ and $\Delta_y(\phi) < 0$.

By the *slope* $\alpha(\theta)$ of a loop θ we mean the ratio $\Delta_y(\phi)/\Delta_x(\phi)$, where ϕ is the tail of θ.

Consider Figure 2. Examples of loops are $\theta_1 = t_1 t_5 t_6 t_7$ in D_2, and $\theta_2 = t_2 t_3 t_4$ in D_3. The head of θ_1 is $t_1 t_5$ while the tail is $t_6 t_7$. The loop θ_1 is $\langle +,+ \rangle$ with $\alpha(\theta_1) = 2$, and the loop θ_2 is $\langle -,- \rangle$, with $\alpha(\theta_2) = 8$. An unfolding of θ_1 is $t_1 t_5 t_6 t_7 t_6 t_7 t_6$.

As we show later, the quality of v_3 in D_3 is equal to 8. In fact, the quality of v_3 in D_3 is decided by slope of the loop θ_2 which is equal to 8. If a game is started from v_3 and is played inside D_3, then player A can first move to v_2 via the head of θ_2, and then start iterating the tail of θ_2. In this manner, the counter values are reduced by a ratio of 1:8. This means that A wins if the play is started from counter values with a ratio less than 1:8. On the other hand, the quality of v_3 in D_2 is ∞. Player A first iterates the tail of the loop $t_5 t_6 t_7$ increasing the counter values by a ratio of 1:2, and then moves to the loop t_9 iterating the tail t_9 and thus reducing the counter values by a ratio of 2:5. This implies that A wins from v_3 inside D_2 regardless of the initial counter values. For example suppose that the initial counters values are 10 and 200 respectively. Then, player A iterates the tail of $t_5 t_6 t_7$ say 400 times (obtaining counter values 410 and 1000), and then moves to and starts iterating

t_9 leading to the exhaustion of the counter of B before that of A.

In the latter example the quality of v_3 in D_2 was decided by the existence of a *sequence* of loops, namely $t_5t_6t_7$ and t_9, rather than one loop. In fact there are three types of loop sequences which are interesting when designing strategies for players A and B (and hence only these three types are considered in the definition of vertex qualities). Now, we identify these three types of loop sequences and call them *(quality) generators*. For a loop θ_1 with a tail ϕ_1 and a loop θ_2, we say that θ_1 and θ_2 are *matching* if $\mathit{first}(\theta_2) \in \phi_1$. Observe that if θ_1 and θ_2 are matching then we can reach θ_2 after iterating the tail ϕ_1 of θ_1. For $D \in Det$, a *(quality) generator* $\mathcal{G}$ in D is a tuple $\langle \theta_1, \theta_2, \ldots, \theta_m \rangle$, where $\theta_i \in D$ is a loop, and θ_i and θ_{i+1} are matching. We say that $\mathcal{G}$ is a *v-generator* if $v \in \theta_1$, We shall consider three types of generators, namely:

1. a generator of type 1 is of the form $\langle \theta \rangle$, where θ is a $\langle +, - \rangle$ loop.
2. a generator of type 2 is of the form $\langle \theta_1, \theta_2, \ldots, \theta_m, \theta_{m+1} \rangle$, where
   - $\theta_1, \theta_2, \ldots, \theta_m$ are $\langle +, + \rangle$ loops,
   - $\alpha(\theta_1) > \alpha(\theta_2) > \cdots > \alpha(\theta_m)$,
   - θ_{m+1} is either a $\langle -, - \rangle$ loop with $\alpha(\theta_{m+1}) > \alpha(\theta_m)$, or θ_{m+1} is a $\langle +, - \rangle$ loop.
3. a generator of type 3 is of the form $\langle \theta \rangle$, where θ is a $\langle -, - \rangle$ loop.

In Figure 2, $\langle t_2t_3t_4 \rangle$ is a v_3-generator of type 3 in D_3, and $\langle t_5t_6t_7, t_9 \rangle$ is a v_3-generator of type 2 in D_2. Observe that there are finitely many generators in G.

Loop Overheads One of the main ideas in designing a strategy for player A is iteration of loops. However, player A should take into consideration the effect of the "overhead" which may be added by a loop to the counter values. For example, consider a $\langle -, - \rangle$ loop θ with a slope 6 and a head ψ with $\Delta_x(\psi) = -4$ and $\Delta_y(\psi) = 3$. The effect of iterating θ is that the ratio of changes in counter values approaches 6. However, at any particular point this ratio is in fact less than 6. More precisely, if after a number of iterations the counter values have changed by x and y respectively, then $6x = y - 27$. The constant 27 reflects the "overhead" added to the counters by the head ψ of the loop. In general, iterating a loop may first introduce an overhead, before the effect of the iteration "stabilizes" converging towards the slope of the loop. Since the strategy of A consists mainly in iterating loops, it is essential that the effect of overheads is considered. This will be reflected when we build winning A-strategies in Section 6.

4.3 Computing Qualities

In this section (Definition 4.1) we suggest a definition for qualities of vertices. We prove the correctness of the definition in Section 5 and Section 6. First we propose a definition for qualities in the deterministic subgraphs of G.

For a vertex v and a deterministic subgraph D, we define $Qual(D)(v)$ to be the largest $\rho \in \mathcal{Q}^{\infty}_{\geq 0}$ such that (i) if there is a v-generator of type 1 or type 2 in D then $\rho = \infty$; and (ii) if there is a v-generator $\langle \theta \rangle$ of type 3 in D then $\rho = \alpha(\theta)$. If ρ is undefined, i.e., there are no v-generators in D then $Qual(D)(v) = 0$. Abusing notation we give the following definition.

Definition 4.1 $Qual(v) = \min_{D \in Det} Qual(D)(v)$. $\square$

Notice that, in the definition of $Qual$, we need only consider type 2 generators which contain 2 loops (i.e. $m = 1$ in the definition of a type 2 generator). However we need the more general definition to carry out some of the proofs in Section 6.

Example Consider Figure 2. Since there is a type 2 v_3-generator in D_2 (namely $\langle t_5t_6t_7, t_9\rangle$), we have $Qual(D_2)(v_3) = \infty$. Also, observe that there are no v_4-generators of type 1 or 2 in D_3, and that 6 is equal to the maximum rational number ρ such that there is v_4-generator $\langle\theta\rangle$ of type 3 in D_3 with $\alpha(\theta) = \rho$ (θ is given by $t_6t_8t_{10}$). This implies that $Qual(D_3)(v_4) = 6$.

In a similar manner we get $Qual(D_1)(v_4) = \infty$, $Qual(D_2)(v_4) = \infty$, and $Qual(D_4)(v_4) = 6$. This means that $Qual(v_4) = \min\{\infty, \infty, 6, 6\} = 6$.

5 A Winning Strategy for Player B

In this section we show (Lemma 5.3) that our definition of $Qual$ (Definition 4.1) satisfies the first condition in Lemma 3.2. We prove that there is a positive integer c (which can be derived from the syntactic description of the game) such that player B has a winning strategy, which we call win_B (Definition 5.1) from each configuration $\langle v, x, y\rangle$ where $x \cdot Qual(v) + c \leq y$. According to Definition 4.1, we know that there is a $D \in Det$ such that $Qual(v) = Qual(D)(v)$, and hence $x \cdot Qual(D)(v) + c \leq y$. The strategy of B is to consistently base his moves on D, i.e., regardless of the moves of A, player B always picks transitions within D.

Definition 5.1 For a vertex v, with $Qual(v) < \infty$, let $D \in Det$ be such that $Qual(D)(v) = Qual(v)$ (by Definition 4.1 we know that such a D exists). We define win_B to be the B-strategy where $win_B(x, y)(\pi)(\lambda) = D[last(\pi), \lambda]$. □

Notice that str_B is not dependent on the values of x and y, and is dependent only on the last vertex[8] of π. We prove the following preliminary lemma.

Lemma 5.2 There is a positive $c \in \mathcal{Q}$ such that the following holds. If $Qual(D)(v) < \infty$ then for every v-path $\pi \in D$ there is $\pi' \preceq \pi$ where $\Delta_y(\pi) \geq Qual(D)(v) \cdot \Delta_x(\pi') - c$.

Proof. The proof can be found in the appendix. The constant c is defined as follows. Let $c' = \max\{Qual(D)(v);\ Qual(D)(v) < \infty\}$. We define $c = (1 + c') \cdot (\Diamond G)$. □

Lemma 5.3 There is a positive $c \in \mathcal{Q}$ such that B is winning from $\langle v, x, y\rangle$ whenever $Qual(v) < \infty$ and $x \cdot Qual(v) + c \leq y$.

Proof. We take c to have the same value as in Lemma 5.2. We prove that win_B is winning from $\gamma = \langle v, x, y\rangle$. The result then follows immediately from the definitions. Consider any play P which follows win_B from γ. From Definition 4.1 and Definition 5.1 we know that there is a $D \in Det$ such that $Qual(v) = Qual(D)(v)$ and $P(j) \in D$ for each $j \geq 0$. Suppose that P is not winning for B from γ. By definition it follows that there is a $k \geq 0$ such that $y + \Delta_y(P(k)) < 0$ and $x + \Delta_x(P(j)) \geq 0$ for each $j : 0 \leq j \leq k$. Since $P(j) \preceq P(k)$ if $j \leq k$, it follows that $P(k) < Qual(D)(v) \cdot \Delta_x(\pi') - c$ for each $\pi' \preceq P(k)$, contradicting Lemma 5.2. □

6 Winning Strategy for A

In this section we show (Lemma 6.5) that Definition 4.1 satisfies the second condition in Lemma 3.2. We prove that there is a positive integer c (which can be derived from the syntactic description of the graph) such that player A has a winning strategy, which we call

[8]Such a strategy is said to be a *no-memory* strategy [Tho95]. To make the next move it needs only to remember the current vertex and the latest move of A.

win_A (Definition 6.4) from each configuration $\langle v, x, y\rangle$ where either $0 < Qual(v) < \infty$ and $x \cdot Qual(v) - c \geq y$, or $Qual(v) = \infty$ and $x \geq c$. This (together with Lemma 5.3) completes the proof of Lemma 3.2. The strategy of A is more complicated than that of B (described in Section 5), since A has to cope with the fact that B has nondeterministic choices, enabling B to change the deterministic subgraph within which the game is played at a given moment. An indication of the complexity of win_A is that the value of $win_A(x, y)(\pi)$ (in contrast to $win_B(x, y)(\pi)$) depends in general on both x, y, and the entire[9] path.

The strategy of A is controlled by a finite set $\{\omega_1, \omega_2, \ldots, \omega_{n-1}\}$ of constants, called the *coefficients (of G)*, defined as follows. Let $0 = \omega_0 < \omega_1 < \omega_2 < \cdots < \omega_n = \infty$ be such that $\omega \in \{\omega_1, \omega_2, \ldots, \omega_{n-1}\}$ if and only if there is $D \in Det$ and a $\langle -, -\rangle$-loop $\theta \in D$ such that $\alpha(\theta) = \omega$. In other words the set $\{\omega_1, \omega_2, \ldots, \omega_{n-1}\}$ contains all rational numbers which are slopes of $\langle -, -\rangle$-loops in the deterministic subgraphs of G. Observe that for each D and v, $Qual(D)(v) = \omega_j$ for some $j : 0 \leq j \leq n$, and consequently for each v, we have $Qual(v) = \omega_j$ for some $j : 0 \leq j \leq n$. In the graph shown in Figure 1, the set of coefficients is given by $\{0, 2.5, 6, 8, \infty\}$ since there are three $\langle -, -\rangle$-loops with slopes 2.5, 6, and 8, respectively.

If the play is started from a vertex v, where $Qual(v) \geq \omega_j$ and the ratio y/x of the counter values of B and A is ω_j, then the strategy of A is to systematically decrease this ratio over the scale $\omega_j, \omega_{j-1}, \ldots, \omega_1$ thus arriving at a point where the "counter resource" of B is exhausted. To achieve the above reduction, player A tries to find "optimal" loops whose iterations reduce the counter ratio along the above scale. The existence of such optimal loops is captured by Lemma 6.1.

6.1 Optimal Loops

We say that the vertices v_1 and v_2 are *competing* if there is a vertex v and transitions t_1 and t_2 such that $source(t_1) = source(t_2) = v$, $L(t_1) = L(t_2)$, $target(t_1) = v_1$ and $target(t_2) = v_2$. Intuitively, v_1 and v_2 are competing if we can reach a stage in a play (the vertex v) where player B can choose between going to v_1 (choosing t_1) or going to v_2 (choosing t_2).

Lemma 6.1 There are positive $c_1, c_2, c_3 \in \mathcal{Q}$ such that the following holds. Consider $D \in Det$, $v \in V$, and a coefficient ω_j, with $j > 1$ and $Qual(v) \geq \omega_j$. There is a loop $\theta^{opt}(D, \omega_j)(v)$, where each unfolding π of $\theta^{opt}(D, \omega_j)(v)$ satisfies the following properties.

1. $Qual(v') \geq \omega_j$, for every vertex v' competing with a vertex $v'' \in \pi$.
2. $\omega_{j-1} \cdot \Delta_x(\pi) \geq \Delta_y(\pi) + c_1 \cdot |\pi| - c_2$.
3. If $\omega_j < \infty$ then $\omega_j \cdot \Delta_x(\pi) \geq \Delta_y(\pi) - c_3$.
4. If $\omega_j = \infty$ then $\Delta_x(\pi) \geq -c_3$.

Proof. The proof (which is quite technical and highly nontrivial) can be found in the appendix. □

Properties 1-4 in Lemma 6.1 describe the effect of iterating (unfolding) the optimal loop $\theta^{opt}(D, \omega_j)(v)$. Property 1 tells us that B will never get the opportunity during the iteration to make a nondeterministic choice which enables him to reach a vertex with a lower quality than ω_j. Property 2 tells us that by taking a sufficiently long unfolding (i.e., by iterating the loop sufficiently many times) we will eventually decrease the ratio of counter values to ω_{j-1}. Properties 3 and 4 tell us that we will always maintain the ratio of at most ω_j (modulo the constant c_3). The constant c_3 covers the "overhead" which may be added by the loop (see Section 4.2 for a discussion of loop overheads).

[9] Such a strategy is said to be an *unbounded-memory* strategy. There is no finite limit on the amount of information which A must remember in order to decide the next move.

6.2 Informal Description of win_A

Suppose that we start the play from a vertex v_0, with $Qual(v_0) = \omega_j$. Player A chooses an arbitrary deterministic subgraph D_0, and assumes that B is playing according to D_0. Player A starts iterating $\theta^{opt}(D_0, \omega_j)(v_0)$ (this loops exists by Lemma 6.1). We say that the game at this point is being played according to the *mode* $\langle D_0, \omega_j \rangle$. We call D_0 the *graph mode* and ω_j the *coefficient mode* of the play. The play stays within the current mode until one of the following occurs

1. the counter of B becomes zero and B cannot answer the next move of A in which case A wins the play.
2. the ratio of the counter values becomes ω_{j-1} in which case A changes the coefficient mode to ω_{j-1}.
3. B chooses a transition which does not belong to D_0, in which case the graph mode is changed.

The main challenge in designing a winning strategy for A, is to define how A should react when the graph mode is changed (case 3 above). An obvious (and naive) strategy would be for A to start iterating a new optimal loop in the new subgraph. However, this would not work due to the loop overhead (Section 4.2). More precisely, if A adopted such a naive strategy then B would simply wait until the head of the loop is traversed (possibly adding some overhead), and then changes the graph mode by picking a transition outside the current subgraph. The procedure of changing subgraphs can be repeated by B, adding more and more overhead to the counters, without allowing A to iterate the tail of any loop. This may lead to the exhaustion of the counter of A, and hence causing A to lose the play. Consequently, A has to limit the amount of overhead added during the course of the entire play, in order to be able to control the ratio of counter values. One way to achieve this is the following. When B chooses a transition which does not belong to the current subgraph, the current graph mode is "suspended" by A and the graph mode is changed. The change of the graph mode corresponds either to "resuming" a play in a subgraph which has previously been suspended, or to starting an entirely new play in a new subgraph. In order for A to bound the overhead it is sufficient to bound the number of times A is forced to start a new play rather than resuming play in a suspended subgraph. This is accomplished by A keeping track of suspended plays in a data structure called a *mode tree* as follows.

Define the set $W \subseteq V \times \Lambda$ such that $\langle v, \lambda \rangle \in W$ if and only if $\langle v, \lambda \rangle$ represents a nondeterministic choice in G, i.e., there are at least two transitions in G whose sources are equal to v and whose labels are equal to λ. The mode tree is constructed from an arbitrary enumeration of the elements of W. The root of the tree is labeled by the element with smallest number, the children of the root are all labeled by the element with the next smallest number and so on until we reach the elements with the largest number. The children of elements with the largest number are the leaves of the tree. Each leaf has a label which is either undefined or equal to an optimal loop. From a node labeled with $\langle v, \lambda \rangle$, there is for each transition t with $source(t) = v$ and $L(t) = \lambda$ an edge, labeled by t, from the node to the next $\langle v, \lambda \rangle$ in the enumeration (or to a leaf). Thus, each path from the root to a leaf in the tree passes each $\langle v, \lambda \rangle \in W$ exactly once. Furthermore, each path from the root to a leaf represents (by collecting its transitions) a unique $D \in Det$. We call D the *subgraph* of the that leaf. Each edge in the tree has a flag with either of the values *on* or *off*. At each point in the play, for each node there is exactly one transition leaving the node which is *on*. We say that a leaf is *on* if all the transitions from the root to that leaf are *on*. It follows that at any point in the play, there is exactly one leaf which is *on*.

Suppose we start the play from a vertex v_0, with $Qual(v_0) = \omega_j$. Player A chooses an arbitrary assignment of the *on-off* flags to the nodes such that for each node there is exactly transition leaving the node which is *on*. Let D_0 be the graph mode of the unique leaf which is *on* initially. The leaf corresponding to D_0 is assigned the optimal loop $\theta^{opt}(D_0, \omega_j)(v_0)$. The labels of all other leaves are initially undefined. Now A starts to play according to the mode $\langle D_0, \omega_j \rangle$. Playing according to this mode consists of iterating the optimal loop $\theta^{opt}(D_0, \omega_j)(v_0)$.

If a leaf is *on* then its label (which is an optimal loop) is defined. If a leaf is not *on* then the corresponding graph mode is either currently suspended (in which case the label of the leaf is an optimal loop), or it has never been activated (in which case its label is undefined). If a leaf, with a label θ is *on* then A plays by iterating θ (which is always an optimal loop). Observe that Lemma 6.1 implies that we always maintain the invariant that $Qual(v') \geq \omega_j$ for every v' which is visited during the unfolding of θ.

If the game is played according to the mode $\langle D, \omega_j \rangle$, and B chooses a transition t', with a source v and a label λ, outside D, then A changes the graph mode and the mode tree as follows. For the unique node in the mode tree labeled by $\langle v, \lambda \rangle$ which is on the path from the root to the node which is currently *on*, A changes the label of the outgoing edge which is currently *on* to off, and changes the label of the outgoing edge which is labeled by t' to *on*. As a result, a new leaf has been turned on. Let the corresponding subgraph be D'. If play according to D' has been active before then we know that the label θ' of the corresponding leaf is defined. In such a case A resumes playing according to mode $\langle \omega_j, D_0 \rangle$ iterating θ'. On the other hand if D' has not been active before, let v' be the current vertex. Since $Qual(v') \geq \omega_j$ (according to the invariant mentioned above), we know that there is an optimal loop $\theta' = \theta^{opt}(D', \omega_j)(v')$. We assign θ' to the label of the leaf corresponding to D'. Now, player A starts iterating θ'.

The above implies that, during the entire play, at most $\Box G$ new plays will be started. This enables player A to bound his losses due to loop overheads. Furthermore, each play is the result of unfolding an optimal loop. By Lemma 6.1 we know that each of these unfoldings converges towards a counter ratio of ω_{j-1}. This means (Lemma 6.3) that the total effect which is the interleaving of the started plays also converges towards ω_{j-1}.

6.3 Formal Definition of win_A

In this section we give a formal definition of win_A. First, we formalize the notion of a *mode tree*.

6.3.1 Mode Trees

Let the set W be defined as above. Let $z = |W|$ and $\# : W \mapsto \{1 \ldots z\}$ be an enumeration of W, i.e., $\#(v, \lambda) \neq \#(v', \lambda')$ if $\langle v, \lambda \rangle \neq \langle v', \lambda' \rangle$. For a transition t, we define $\#t$ to be $\#(v, \lambda)$, where $source(t) = v$ and $L(t) = \lambda$. A *mode tree* φ is a partial function from tuples of transitions to transitions such that if $t_1, \ldots, t_n$ are transitions, $0 \leq n < z$, and $\#t_i = i$ for each $i : 1 \leq i \leq n$, then $\varphi(t_1, \ldots, t_n) = t$, where $\#t = n + 1$. Intuitively, each sequence $t_1, \ldots, t_n$ of transitions, with $\#t_i = i$, identifies a unique node in the mode tree, namely the node which we reach by starting from the root, and following the edges labeled by $t_1, \ldots, t_n$. The value of $\varphi(t_1, \ldots, t_n)$ defines the label of the edge which (i) has a flag *on* and (ii) originates from the node characterized by $t_1, \ldots, t_n$. For a mode tree φ and $i : 1 \leq i \leq z+1$, we define φ^i inductively to be a tuple of transitions, such that φ^1 is the empty list $\langle\rangle$, and φ^{i+1} is the result of appending $\varphi(\varphi^i)$ to the end of φ^i (i.e., $\varphi^{i+1} = append(\varphi^i, \varphi(\varphi^i)))$. The value of φ^i is the node we reach by starting from the root and following $i-1$ edges each of which has a flag value which is *on*. Observe that φ^{z+1} corresponds the leaf which is *on*. In other words, the transitions placed on φ^{z+1} define a deterministic subgraph $D_\varphi \in Det$. More precisely, $t' \in D_\varphi$ if and only if t' is in the tuple

φ^{z+1}.

For a mode tree φ and a transition t, we define the *update* $\varphi \odot t$ of φ to be the tree φ' where

$$\varphi'(t_1, \ldots, t_n) \;=\; \begin{cases} t & \text{if } \varphi^{n+1} = \langle t_1, \ldots, t_n \rangle \text{ and } \#t = n+1 \\ \varphi(t_1, \ldots, t_n) & \text{otherwise} \end{cases}$$

Intuitively, this operation corresponds to turning off the flag of one edge, and instead turning on the flag of another edge originating from the same node in the mode tree. This occurs when player B forces A to change the graph mode by choosing a transition outside the current graph mode. Observe that $(D_\varphi \odot t)[v, \lambda] = D_\varphi[v, \lambda]$ can be guaranteed only if $\#(v, \lambda) < \#t$.

Lemma 6.2 For a mode tree φ and transitions $t_1, \ldots, t_n$, define $\varphi_0 = \varphi$ and $\varphi_{i+1} = \varphi_i \odot t_{i+1}$. If $D(\varphi_k) = D(\varphi_\ell)$ and $D(\varphi_j) \neq D(\varphi_k)$, for all $j : j < k < \ell$, then $\langle source(t_{k+1}), L(t_{k+1}) \rangle = \langle source(t_\ell), L(t_\ell) \rangle$.
Proof. The proof can be found in the appendix. $\square$

6.3.2 Definition of win_A

Let $\omega_0, \omega_1, \ldots, \omega_n$ be the coefficients of the game. The definition of win_A uses a constant c given by $c = (c_3 + \omega_{n-1} + 1) \cdot (\Box G)$, where c_3 is defined as described by Lemma 6.1. The choice of c is motivated in the proof of Lemma 6.3. The strategy win_A is derived from a set $f_0, f_1, \ldots, f_n$ of functions, where f_j describes the behaviour of A when the coefficient mode of the play is ω_j. Each f_j represents a "generalized" A-strategy in the sense it returns, besides a label (which is the next move of A), some information about the the graph mode of the play. More precisely, given natural numbers x and y and a path π, the value of $f_j(x, y)(\pi)$ is a tuple of the form $\langle \lambda, \varphi, \Theta \rangle$, where $\lambda \in \Lambda$ is the next move of A, φ represents the mode tree, and Θ is a mapping from deterministic subgraphs to loops, representing the labels of the leafs. From each f_j we can derive an A-strategy, which we call win_A^j, where $win_A^j(x, y)(\pi) = \lambda$ if $f_j(x, y)(\pi) = \langle \lambda, \varphi, \Theta \rangle$ for some φ and Θ.

The value of f_j is defined using induction on j as follows. The value of f_0 is defined arbitrarily (the game will never be played according to the coefficient mode ω_0). Suppose that we are given natural numbers x and y and a path $\pi = v_0 t_1 v_1 \cdots t_m v_m$, where $Qual(v_0) = \omega_j$. Let $x_i = x + \Delta_x(v_0 t_1 \cdots v_i)$ and $y_i = y + \Delta_y(v_0 t_1 \cdots v_i)$. There are two cases in the definition of $f_j(x, y)(\pi)$, depending to whether the play has reached a point where the coefficient mode needs to be changed.

1. If there is no $k : 1 \leq k \leq m$ such that $\omega_{j-1} \cdot x_k - c \geq y_k$. This means that the ratio of counter values has not yet reduced to ω_{j-1}. In such a case the game continues to be played according to the coefficient mode ω_j. There are two subcases in the definition of $f_j(x, y)(\pi)$ depending on the length of π. In both cases it can be observed that each vertex v which is visited during the play, is included in the unfolding of an optimal loop of the form $\theta^{opt}(D, \omega_j)(v')$. Hence the invariant $Qual(v) \geq \omega_j$ holds.

   1a. if $m = 0$ then we define $f_j(x, y)(\pi) = \langle \lambda_0, \varphi_0, \Theta_0 \rangle$, where λ_0, φ_0, and Θ_0 are given as follows

   - we choose φ_0 arbitrarily, i.e., when A starts playing according to a certain coefficient mode, the graph mode of the play is chosen arbitrarily.
   - we define $\Theta_0(D_{\varphi_0}) = \theta^{opt}(D_{\varphi_0}, \omega_j)(v_0)$, while $\Theta_0(D)$ is undefined if $D \neq D_{\varphi_0}$. In other words, initially the label of exactly one leaf is defined, namely the leaf corresponding to the current active subgraph D_{φ_0}. This label is given by the optimal loop $\theta^{opt}(D_{\varphi_0}, \omega_j)(v_0)$. The existence of such an optimal loop is guaranteed

by Lemma 6.1 and the fact that $Qual(v_0) \geq \omega_j$. The labels of all other leafs are undefined (the corresponding subgraphs have not been active previously in the current coefficient mode).

- we take $\lambda_0 = next(\Theta_0(D_{\varphi_0}), v_0)$. This means that player A starts iterating the loop $\theta^{opt}(D_{\varphi_0}, \omega_j)(v_0)$, choosing λ_0 which corresponds to the first step in the iteration of the loop.

1b. If $m > 0$, suppose that $f_j(x,y)(v_0 t_1 v_1 \cdots t_{m-1} v_{m-1}) = \langle \lambda_{m-1}, \varphi_{m-1}, \Theta_{m-1} \rangle$. We define $f_j(x,y)(\pi) = \langle \lambda_m, \varphi_m, \Theta_m \rangle$, where λ_m, φ_m, and Θ_m are given as follows

- $\varphi_m = \varphi_{m-1} \odot t_m$.
- if $\Theta(D_{\varphi_m})$ is defined then $\Theta_m = \Theta_{m-1}$. If $\Theta_m(D_{\varphi_m})$ is undefined then $\Theta_m(D_{\varphi_m}) = \theta^{opt}(D(\varphi_m), \omega_j)(v_m)$, and $\Theta_m(D) = \Theta_{m-1}(D)$ if $D \neq D_{\varphi_m}$. The existence of $\theta^{opt}(D(\varphi_m), \omega_j)(v_m)$ follows from Lemma 6.1 and the fact that $Qual(v_m) \geq \omega_j$ according to the invariant mentioned above. Intuitively, if the current graph mode has been active previously, then the labels of the leafs are not changed. In such a case player A continues (if $\varphi_m = \varphi_{m-1}$) or resumes (if $\varphi_m \neq \varphi_{m-1}$) iterating the loop $\Theta(D_{\varphi_m})$. On the other hand, if the graph mode is activated for the first time then an optimal loop is assigned to the label of the corresponding leaf (that label has been undefined up to now), and A starts iterating the new loop.
- we define $\lambda = next(\Theta(D_{\varphi_m}), v_m)$, i.e., we perform another step in the iteration of the currently active loop.

2. if there is a $k : 1 \leq k \leq m$ such that $\omega_{j-1} \cdot x_k \geq y_k + c$. then[10] let k be the minimum integer satisfying this property. We define $f_j(x,y)(\pi) = f_{j-1}(x_k, y_k)(v_k t_{k+1} v_{k+1} \cdots t_m v_m)$. Intuitively, this means that as soon as the ratio of counter values is reduced to ω_{j-1}, the coefficient mode of the play is changed to ω_{j-1}, and the behaviour of A will henceforth be defined by f_{j-1}.

Lemma 6.3 For each $0 < j$, the A-strategy win_A^j is winning from $\langle v, x, y \rangle$, whenever either (i) $\omega_j \leq Qual(v) < \infty$ and $\omega_j \cdot x - c \geq y$, or (ii) $Qual(v) = \omega_n = \infty$ and $x \geq c$.

Proof. *(Sketch)* The details of the proof can be found in the appendix. We use induction on j. The base case is trivial. Suppose that $j > 0$. Let P be any play which follows win_A from γ. We show that P is winning for A from γ. Let $f_j(x,y)(P(i)) = \langle \lambda_i, \varphi_i, \Theta_i \rangle$, and let $P(i)$ be of the form $\pi = v_0 t_1 v_1 \cdots t_i v_i$. We define $x_i = x + \Delta_x(P(i))$ and $y_i = y + \Delta_y(P(i))$. Let $D \in Det$. Consider the sequence $0 \leq i_1 < i_2 < \cdots < i_p \leq m$, where $k \in \{i_1, i_2, \ldots, i_p\}$ if and only if $D_{\varphi_k} = D$. By Lemma 6.2 it follows that $\langle v_{i_j}, \lambda_{i_j} \rangle = \langle v_{i_{j+1}-1}, \lambda_{i_{j+1}-1} \rangle$, and hence $source(t_{i_{j+1}}) = v_{i_j}$. This means that for any $k \leq \ell$, the sequence $v_k t_{k+1} v_{k+1} \cdots t_\ell v_\ell$ is a path. We define the path

$$\pi(k, D) \quad = \quad \begin{cases} v_{i_1} t_{i_2} v_{i_2} \cdots t_{i_j} v_{i_j} & \text{if } i_1 = 0 \\ v_{i_1 - 1} t_{i_1} v_{i_1} t_{i_2} v_{i_2} \cdots t_{i_j} v_{i_j} & \text{if } i_1 > 0 \end{cases}$$

We let $\pi'(k, D)$ be the path $v_{i_1} t_{i_2} v_{i_2} \cdots t_{i_j} v_{i_j}$. We observe that $\pi'(k, D)$ is an unfolding[11] of the optimal loop $\theta^{opt}(\omega_j, D)(v_{i_1})$. It follows by Lemma 6.1 (where π is instantiated by $\pi'(k, D)$) that there are positive integers c_1, c_2, c_3 satisfying the conditions of Lemma 6.1. Using the facts above, we define $c = (c_3 + \omega_{n-1} + 1) \cdot (\Box G)$. and show in the appendix the following two claims (which together with the induction hypothesis imply the result).

---

[10]Observe that if the condition is satisfied then $j > 1$. Otherwise the value of y would be negative.

[11]Notice that t_{i_1} need not be part of the unfolding of any loop. However, since only a bounded number of graph modes are activated, the losses of A due to such transitions are bounded.

1. there is a k such that $x_k \cdot \omega_{j-1} \geq y_k + c$.

2. For each $j \geq 0$ it is the case that $x_j \leq 0$ implies $y_j < 0$. $\square$

Definition 6.4 We define win_A to be the A-strategy where $win_A(x,y)(\pi) = win_A^j(x,y)(\pi)$ if $Qual(first(\pi)) = \omega_j$. $\square$

Lemma 6.5 There is a positive $c \in \mathcal{Q}$ such that A is winning from $\langle v, x, y \rangle$ whenever either (i) $0 < Qual(v) < \infty$ and $x \cdot Qual(v) - c \geq y$, or (ii) $Qual(v) = \infty$ and $x \geq c$.

Proof. From Lemma 6.3 and Definition 6.4 we know that win_A is winning from $\langle v, x, y \rangle$. The result follows immediately from the definitions. Notice that the constant c is given in the definition of win_A^j. $\square$

7 Properties of Simulation

A simulation $\lhd$ is said to be *upward closed* if $\langle q, x \rangle \lhd \langle r, y \rangle$ and $y \leq y'$ imply $\langle q, x \rangle \lhd \langle r, y' \rangle$. We observe that $\sqsubseteq$ is upward-closed.

For two nets N_1 and N_2, and a natural number z, a *z-partial simulation* $\trianglelefteq$ between N_1 and N_2 is a set of pairs of the form $\langle \gamma_1, \gamma_2 \rangle$, where $\gamma_1 = \langle q, x \rangle$ and $\gamma_2 = \langle r, y \rangle$ are configurations of N_1 and N_2 respectively, $0 \leq x \leq z$, such that if $\langle q, x \rangle \trianglelefteq \langle r, y \rangle$ and $\langle q, x \rangle \stackrel{\lambda}{\Longrightarrow} \langle q', x' \rangle$, where $0 \leq x, x' \leq z$, then there are r' and y' such that $\langle r, y \rangle \stackrel{\lambda}{\Longrightarrow} \langle r', y' \rangle$ and $\langle q', x' \rangle \trianglelefteq \langle r', y' \rangle$. We say that $\trianglelefteq$ is a *partial simulation* between N_1 and N_2 if $\trianglelefteq$ is a z-partial simulation for some z. For a simulation $\lhd$ we define $\lhd|_z$ to be the set $\{\langle \langle q, x \rangle, \langle r, y \rangle \rangle;\ x \leq z\}$.

For an upward closed simulation $\lhd$, *the simulation table of* $\lhd$, is a two-dimensional array with $|Q_1| \cdot |Q_2|$ rows and infinitely many columns. The table has a column for each natural number x and a row for each pair $\langle q, r \rangle$, where $q \in Q_1$ and $r \in Q_2$. Intuitively the value of the entry corresponding to column x and row $\langle q, r \rangle$ is the minimum y such that $\langle q, x \rangle \lhd \langle r, y \rangle$. Notice that the simulation table gives a complete characterization of $\lhd$. Formally, we represent the simulation table by a function $SimTab_{\lhd} : \mathcal{N} \mapsto Q_1 \times Q_2 \mapsto \mathcal{N} \cup \{\infty\}$, where $SimTab_{\lhd}(x)(q,r)$ is the value in column x and row $\langle q, r \rangle$. By $SimTab_{\lhd}(x)(q,r) = \infty$ we mean that there is no y such that $\langle q, x \rangle \lhd \langle r, y \rangle$.

An upward-closed z-partial simulation can in a similar manner be represented by a simulation table where the number of columns is equal to $z+1$.

For a simulation $\lhd$, we define a function $D_{\lhd}$ to describe the difference between any pair of elements in a row in the simulation table, that is $D_{\lhd}(x)(q',r')(q,r) = SimTab_{\lhd}(x)(q',r') - SimTab_{\lhd}(x)(q,r)$.

The function $D_{\trianglelefteq}$, for a partial simulation $\trianglelefteq$, is defined in a similar manner.

For a z-partial simulation $\trianglelefteq$ and a simulation $\lhd$, we say that $\lhd$ is consistent with $\trianglelefteq$ if $SimTab_{\lhd}(x) = SimTab_{\trianglelefteq}(x)$ for $0 \leq x \leq z$.

Consider a total preorder $\preceq$ on the set $Q_1 \times Q_2$. We let $\equiv$ denote the equivalence relation induced by $\preceq$, and let $\prec$ denote the set $\preceq \setminus \equiv$.

For a simulation $\lhd$ and $w < z$, we say that $SimTab_{\lhd}(w)$ *is covered by* $SimTab_{\lhd}(z)$ if

- $\langle q, r \rangle \prec \langle q', r' \rangle \implies ((SimTab_{\trianglelefteq}(x)(q',r') - SimTab_{\trianglelefteq}(x+1)(q,r)) \geq 2)$, for all x : $w \leq x < z$, and
- $\langle q, r \rangle \preceq \langle q', r' \rangle \implies (D_{\trianglelefteq}(w)(q',r')(q,r) \leq D_{\trianglelefteq}(z)(q',r')(q,r))$

The definition of the relation is extended naturally to partial simulations.

Lemma 7.1 For a z-partial simulation $\trianglelefteq$ and $w < z$, if $SimTab_{\trianglelefteq}(w)$ is covered by $SimTab_{\trianglelefteq}(z)$, then there exists a simulation $\triangleleft$ consistent with $\trianglelefteq$.

Proof. We define $E_{\trianglelefteq}(x', x) = SimTab_{\trianglelefteq}(x') - SimTab_{\trianglelefteq}(x)$. We define $SimTab_{\triangleleft}(x) = SimTab_{\trianglelefteq}(x)$ if $0 \leq x \leq z$, and $SimTab_{\triangleleft}(x) = k * E_{\trianglelefteq}(z, w) + SimTab_{\trianglelefteq}(w + i)$ if $x = w + i + k * (z - w)$ for some $1 \leq k$ and $0 \leq i < z - w$. See the appendix for the details of the proof. □

Observe that $\triangleleft$ is a semi-linear simulation relation.

Corollary 7.2 For a z-partial simulation $\trianglelefteq$ and $w < z$, if $SimTab_{\trianglelefteq}(w)$ is covered by $SimTab_{\trianglelefteq}(z)$, then $\trianglelefteq \subseteq \sqsubseteq$.

Lemma 7.3 For a simulation $\triangleleft$, $\triangleleft|_z$ is a z-partial simulation.

Proof. Let $\trianglelefteq = \triangleleft|_z$. Suppose that $\langle q, x\rangle \trianglelefteq \langle r, y\rangle$ and $\langle q, x\rangle \stackrel{\lambda}{\Longrightarrow} \langle q', x'\rangle$, where $x' \leq z$. We know that $\langle q, x\rangle \triangleleft \langle r, y\rangle$, so there are r' and y' such that $\langle r, y\rangle \stackrel{\lambda}{\Longrightarrow} \langle r', y'\rangle$ and $\langle q', x'\rangle \triangleleft \langle r', y'\rangle$. This implies that $\langle q', x'\rangle \trianglelefteq \langle r', y'\rangle$. □

8 The Main Result

We take our preorder on $Q_1 \times Q_2$ (Section 7) as $v \preceq v'$ if and only if $Qual(v) \leq Qual(v')$.

Lemma 8.1 There is an infinite increasing sequence $w_1, w_2, w_3, \ldots$ such that if $i < j$ then $SimTab_{\sqsubseteq}(w_i)$ is covered by $SimTab_{\sqsubseteq}(w_j)$.

Proof. From Lemma 3.1, Lemma 3.2 (with ρ instantiated to $Qual(v)$ according to Lemma 6.5 and Lemma 5.3), it is clear that there is a positive $c \in \mathcal{Q}$ such that $Qual(v) \cdot x - c \leq SimTab_{\sqsubseteq}(x)(v) \leq Qual(v) \cdot x + c$ if $Qual(v) < \infty$, and $SimTab_{\sqsubseteq}(x) = \infty$ if $Qual(v) = \infty$ and $x \geq c$. The result follows immediately. □

Lemma 8.2 Simulation is semi-decidable for one-counter nets.

Proof. Given nets N_1 and N_2, with sets of states Q_1 and Q_2, and configurations $\langle q, x\rangle$ in N_1 and $\langle r, y\rangle$ in N_2, we generate systematically all finite tables of width $|Q_1| \cdot |Q_2|$, where the elements of a table belong to $\mathcal{Q}^{\infty}_{\geq 0}$. For a generated table of length z, we check whether

- it represents an upward closed z-partial simulation $\trianglelefteq$ between N_1 and N_2, and
- there is $w < z$ such that $SimTab_{\trianglelefteq}(w)$ is covered by $SimTab_{\trianglelefteq}(z)$, and
- $SimTab_{\trianglelefteq}(x)(q, r) \leq y$.

If this is the case we declare $\langle q, x\rangle \sqsubseteq \langle r, y\rangle$ to hold. The correctness of this procedure follows from Corollary 7.2.

Now, suppose that $\langle q, x\rangle \sqsubseteq \langle r, y\rangle$. We show that this will eventually be detected. From Lemma 8.1 it follows that there are w and z such that $x < w < z$ and $SimTab_{\sqsubseteq}(w)$ is covered by $SimTab_{\sqsubseteq}(z)$. By Lemma 7.3 we know that $\trianglelefteq = \triangleleft|_z$ is a z-partial simulation. This means that when we generate the table $SimTab_{\trianglelefteq}$, we will find that the three conditions mentioned above all hold. □

Note Observe that if $\langle q, x\rangle \sqsubseteq \langle r, y\rangle$, then there is a semi-linear witness, i.e., a semi-linear simulation relation which contains the pair $\langle q, x\rangle$ and $\langle r, y\rangle$. This semi-linear relation is defined by Lemma 7.1.

Theorem 8.3 Simulation is decidable for one-counter nets.

Proof. We observe that one-counter nets are finitely-branching transition systems. The result follows from Lemma 8.2 and the standard result that non-simulation is semi-decidable for finitely-branching transition systems. □

References

[AČJYK96] Parosh Aziz Abdulla, Karlis Čerāns, Bengt Jonsson, and Tsay Yih-Kuen. General decidability theorems for infinite-state systems. In *Proc. 11^{th} IEEE Int. Symp. on Logic in Computer Science*, pages 313–321, 1996.

[AJ93] Parosh Aziz Abdulla and Bengt Jonsson. Verifying programs with unreliable channels. In *Proc. 8^{th} IEEE Int. Symp. on Logic in Computer Science*, pages 160–170, 1993.

[AK95] Parosh Aziz Abdulla and Mats Kindahl. Decidability of simulation and bisimulation between lossy channel systems and finite state systems. In Lee and Smolka, editors, *Proc. CONCUR '95, 6^{th} Int. Conf. on Concurrency Theory*, volume 962 of *Lecture Notes in Computer Science*, pages 333 – 347. Springer Verlag, 1995.

[BBK93] J.C.M. Baeten, J.A. Bergstra, and J.W. Klop. Decidability of bisimulation equivalence for processes generating context-free languages. *Journal of the ACM*, (40):653–682, 1993.

[Čer92] K. Čerāns. Decidability of bisimulation equivalence for parallel timer processes. In *Proc. Workshop on Computer Aided Verification*, volume 663 of *Lecture Notes in Computer Science*, pages 302–315, 1992.

[Čer94] K. Čerāns. Deciding properties of integral relational automata. In Abiteboul and Shamir, editors, *Proc. ICALP '94*, volume 820 of *Lecture Notes in Computer Science*, pages 35–46. Springer Verlag, 1994.

[CHM93] S. Christensen, Y. Hirshfeld, and F. Moller. Bisimulation equivalence is decidable for basic parallel processes. In *Proc. CONCUR '93, Theories of Concurrency: Unification and Extension*, pages 143–157, 1993.

[CHS92] S. Christensen, H. Hüttel, and C. Stirling. Bisimulation equivalence is decidable for all context-free processes. In W. R. Cleaveland, editor, *Proc. CONCUR '92, Theories of Concurrency: Unification and Extension*, pages 138–147, 1992.

[Esp95] J. Esparza. Petri nets, commutative context-free grammers, and basic parallel processes. In *Proc. Fundementals of Computation Theory*, number 965 in Lecture Notes in Computer Science, pages 221–232, 1995.

[GH94] J.F. Groote and H. Hüttel. Undecidable equivelences for basic process algebra. *Information and Computation*, 1994.

[Jan90] P. Jančar. Decidability of a temporal logic problem for Petri nets. *Theoretical Computer Science*, 74:71–93, 1990.

[Jan95] P. Jančar. Undecidability of bisimilarity for Petri nets and related problem. *Theoretical Computer Science*, (148):281–301, 1995.

[Jan97] P. Jančar. Bisimulation equivalence is decidable for one-counter processes. In *Proc. ICALP '97*, pages 549–559, 1997.

[JE96] P. Jančar and J. Esparza. Deciding finiteness of petri nets up to bisimulation. In *Proc. ICALP '96*, number 1099 in Lecture Notes in Computer Science, pages 478–489, 1996.

[JM95] P. Jančar and F. Moller. Checking regular properties of Petri nets. In *Proc. CONCUR '95, 6^{th} Int. Conf. on Concurrency Theory*, pages 348–362, 1995.

[JP93] B. Jonsson and J. Parrow. Deciding bisimulation equivalences for a class of non-finite-state programs. *Information and Computation*, 107(2):272–302, Dec. 1993.

[Kin97] Mats Kindahl. Results on decidability of simulation and bisimulation for lossy channel systems. Technical Report 91, Department of Computer Systems, Uppsala University, May 1997. Licentiate Thesis.

[LY93] K. Larsen and W. Yi. Time-abstracted bisimulation: Implicit specification and decidability. In *Mathematical Foundations of Programming Semantics*, Lecture Notes in Computer Science, New Orleans, April 1993. Springer Verlag.

[Sti95] C. Stirling. Local model checking games. In *Proc. CONCUR '95, 6^{th} Int. Conf. on Concurrency Theory*, volume 962 of *Lecture Notes in Computer Science*, pages 1–11. Springer Verlag, 1995.

[Sti96] C. Stirling. Decidability of bisimulation equivalence for normed pushdown processes. In *Proc. CONCUR '96, 7^{th} Int. Conf. on Concurrency Theory*, volume 1119 of *Lecture Notes in Computer Science*, pages 217–232. Springer Verlag, 1996.

[Tho95] W. Thomas. On the synthesis of strategies in infinite games. In *STACS 95*, volume 900 of *Lect. Notes in Comput. Sci.*, pages 1–13. Springer-Verlag, 1995.

A Appendix- Proofs of some lemmas

A.1 Proof of Proposition 2.1

(Sketch) Given nets N and N', we produce a deterministic and partially complete net N_3, and a complete net N_3' such that for any configurations γ of N, and γ' of N', there is a simulation between N and N' containing $\langle \gamma, \gamma' \rangle$ if and only if there is a simulation between N_3 and N_3' containing $\langle \gamma, \gamma' \rangle$. We perform the proof in three steps.

1. From N and N' we generate N_1 and N_1' where N_1 is deterministic. Let h be any enumeration of the transitions in N.

To derive N_1 from N, we replace the label λ of each transition t by a unique new label λ_i where i is the number of t (as defined by h). Since each transition in N_1 gets a unique label, N_1 is trivially deterministic.

To derive N_1' from N', we replace each transition t by a set of transitions, each with the same source, target, and incrementation as t, and with a label of the form λ_i where λ is the label of t and i is in the range of h.

2. From N_1 and N_1' we generate N_2 and N_2' where N_2 is deterministic and partially complete. We add a new symbol $\$_1$ to the set of labels.

To derive N_2 from N_1 we add a new state r_1 to the set of states in N_1. For each state from which there are no outgoing transitions, we add a transition to r_1 with a label $\$_1$ and an incrementation -1. From r_1 there is only one transition, leading back to r_1, labeled with $\$_1$, and with an incrementation -1.

To derive N_2' from N_1', we add to each state a transition leading back to the state, labeled with $\$_1$, and with an incrementation 0.

3. From N_2 and N_2' we generate N_3 and N_3' where N_3 is deterministic and partially complete, and N_3' is complete. We add a new symbol $\$_2$ to the set of labels.

To derive N_3 from N_2, we add to each state a transition leading back to the state, labeled with $\$_2$, and with an incrementation 0.

To derive N_3' from N_2', we add a new state r_2 to the set of states in N_2'. We add to each state in N_2' a transition leading back to the state, labeled with $\$_2$, and with an incrementation 0. For each state q and λ where there is no outgoing transition from q labeled with λ, we add a transition from q to r_2, labeled with λ, and with an incrementation -1. For each label λ, there is a transition from r_2 to itself labeled with λ and with an incrementation -1.

A.2 Proof of Lemma 5.2

First, some preliminaries.

Proposition A.1 If $v_1 \rightsquigarrow_{\pi,D} v_2$ then $Qual(D)(v_1) \geq Qual(D)(v_2)$.

Proof. The proof follows directly from the definitions.

For a simple loop ϕ and path $\pi = \pi_1 \bullet \phi \bullet \pi_2$, we use π/ϕ to denote $\pi_1 \bullet \pi_2$. A *decomposition* $\mathcal{D}$ of a path π is of the form $\langle \phi_1, \ldots, \phi_m \rangle$, where there are paths $\pi_1, \ldots, \pi_m$ such that $\pi_1 = \pi$, $\pi_{i+1} = \pi_i/\phi_i$ for $1 \leq i \leq m$, and π_{m+1} is a simple path (which we call *the residue* of π). By

$\Delta_x(\mathcal{D})$ ($\Delta_y(\mathcal{D})$) we mean $\sum_{i=1}^{m} \Delta_x(\phi_i)$ ($\sum_{i=1}^{m} \Delta_y(\phi_i)$). For decompositions $\mathcal{D}_1$ and $\mathcal{D}_2$ we use $\mathcal{D}_1 \cdot \mathcal{D}_2$ to denote the result of appending the elements of $\mathcal{D}_1$ with those of $\mathcal{D}_2$. Observe that if θ is a loop with a head ψ and a tail ϕ then there is only one decomposition $\mathcal{D}$ of θ, where $\mathcal{D}$ is of the form $\langle \phi \rangle$ and has the residue of $\mathcal{D}$ is ψ. Also, if ϕ is a simple path then the only decomposition of ϕ is $\langle\rangle$ with a residue ψ.

Proposition A.2 The following properties hold for decompositions.

- If $\mathcal{D}$ is a decomposition of a path π with a residue ψ then $\mathit{first}(\pi) = \mathit{first}(\psi)$, and $\mathit{last}(\pi) = \mathit{last}(\psi)$.
- For a path $\pi = \pi_1 \bullet \pi_2$ if $\mathcal{D}_1$ is a decomposition of π_1 with a residue ψ_1 and $\mathcal{D}_2$ is a decomposition of $\psi_1 \bullet \pi_2$ then $\mathcal{D}_1 \bullet \mathcal{D}_2$ is a decomposition of π.
- If $\mathcal{D}$ is a decomposition of π then $\Delta_x(\mathcal{D}) - \Diamond G \leq \Delta_x(\pi) \leq \Delta_x(\mathcal{D}) + \Diamond G$. The same applies for Δ_y.

Proof. The proofs follow directly from the definitions.

Proposition A.3 Consider $v \in V$ and $D \in \mathit{Det}$, with $\mathit{Qual}(D)(v) < \infty$. For each v-path $\pi \in D$, there are π' and π'', with $\pi = \pi' \bullet \pi''$, and a decomposition $\mathcal{D}'$ of π' with a residue ψ', such that both

- $\Delta_y(\mathcal{D}') \geq \mathit{Qual}(D)(v) \cdot \Delta_x(\mathcal{D}')$, and
- there is a decomposition $\mathcal{D}''$ of $\psi' \bullet \pi_2$ where
  - $\Delta_x(\mathcal{D}'') \geq 0$, and
  - $\Delta_y(\mathcal{D}'') \geq \mathit{Qual}(D)(\mathit{last}(\pi)) \cdot \Delta_x(\mathcal{D}'')$

Proof. By induction on $|\pi|$. The base case (with $|\pi| = 0$) holds trivially.

For the induction step, suppose that π is of the form $\pi_1 v_1 t v_2$. Let $\pi_0 = v_1 t v_2$. By the induction hypothesis it follows that there are π_2 and π_3, with $\pi_1 = \pi_2 \bullet \pi_3$, and a decomposition $\mathcal{D}_2$ of π_2 with a residue ψ_2, such that both

- $\Delta_y(\mathcal{D}_2) \geq \mathit{Qual}(D)(v) \cdot \Delta_x(\mathcal{D}_2)$, and
- there is a decomposition $\mathcal{D}_3$ of $\psi_2 \bullet \pi_3$ where
  - $\Delta_x(\mathcal{D}_3) \geq 0$, and
  - $\Delta_y(\mathcal{D}_3) \geq \mathit{Qual}(D)(\mathit{last}(\pi_1)) \cdot \Delta_x(\mathcal{D}_3)$

Let ψ_3 be the residue of π_3. By Proposition A.2 it follows that ψ_3 and π_0 are compatible. It is obvious that $\psi_3 \bullet \pi_0$ is either a simple path or a loop. We consider both cases.

1. If $\psi_3 \bullet \pi_0$ is a simple path. By Proposition A.2 it follows that $\mathcal{D}_3$ is a decomposition of π. We take $\pi' = \pi_2$, $\pi'' = \pi_3 \bullet \pi_0$, $\mathcal{D}' = \mathcal{D}_2$, and $\mathcal{D}'' = \mathcal{D}_3$.

2. If $\psi_3 \bullet \pi_0$ is a loop. Let ϕ be the head of the loop. By Proposition A.2 it follows that $\mathcal{D}_4 = \mathcal{D}_3 \cdot \langle \phi \rangle$ is a decomposition of π. Notice that $\mathit{last}(\pi) \in \pi_1$ and hence by Proposition A.1 we get $\mathit{Qual}(D)(\mathit{last}(\pi)) = \mathit{Qual}(D)(\mathit{last}(\pi_1))$.

First, we show that $\Delta_y(\phi) \geq Qual(D)(last(\pi)) \cdot \Delta_x(\phi)$. If ϕ is $\langle -, + \rangle$ then the claim is satisfied trivially. If π is $\langle +, + \rangle$ then, since $Qual(D)(v) < \infty$, we know that $\alpha(\phi) \geq Qual(D)(last(\pi))$ (otherwise there exists a type 2 v-generator in D), and hence $\Delta_y(\phi) = \alpha(\phi) \cdot \Delta_x(\phi) \geq Qual(D)(last(\pi)) \cdot \Delta_x(\phi)$. If π is $\langle -, - \rangle$ then it follows from the definitions that $Qual(D)(last(\pi)) \geq \alpha(\phi)$ and hence $\Delta_y(\phi) = \alpha(\phi) \cdot \Delta_x(\phi) \geq Qual(D)(last(\pi)) \cdot \Delta_x(\phi)$. Also since $Qual(D)(v) < \infty$ we know from the definitions that ϕ cannot be $\langle +, - \rangle$.

This implies that

$$\begin{aligned}
&\Delta_y(\mathcal{D}_4) \ \geq \ \Delta_y(\mathcal{D}_3) + \Delta_y(\phi) \ \geq \\
&Qual(D)(last(\pi_1)) \cdot \Delta_x(\mathcal{D}_3) + Qual(D)(last(\pi)) \cdot \Delta_x(\phi) \ \geq \\
&Qual(D)(last(\pi)) \cdot \Delta_x(\mathcal{D}_3) + Qual(D)(last(\pi)) \cdot \Delta_x(\phi) \ \geq \\
&Qual(D)(last(\pi)) \cdot \Delta_x(\mathcal{D}_4)
\end{aligned}$$

There are two cases:

- If $\Delta_x(\mathcal{D}_4) \geq 0$. We take $\pi' = \pi_2$, $\pi'' = \pi_3 \bullet \pi_0$, $\mathcal{D}' = \mathcal{D}_2$, and $\mathcal{D}'' = \mathcal{D}_4$.
- If $\Delta_x(\mathcal{D}_4) \leq 0$. By Proposition A.1 we know that $Qual(D)(v) \geq Qual(D)(last(\pi))$. It follows that $\Delta_y(\mathcal{D}_4) \geq Qual(D)(v) \cdot \Delta_x(\mathcal{D}_4)$. Define $\mathcal{D}_5 = \mathcal{D}_2 \bullet \mathcal{D}_4$. By Proposition A.2 it follows that $\mathcal{D}_5$ is a decomposition of π. Furthermore

  $$\begin{aligned}
  &\Delta_y(\mathcal{D}_5) \ \geq \ \Delta_y(\mathcal{D}_2) + \Delta_y(\mathcal{D}_4) \ \geq \ Qual(D)(v) \cdot \Delta_x(\mathcal{D}_2) + Qual(D)(v) \cdot \Delta_x(\mathcal{D}_4) \ \geq \\
  &Qual(D)(v) \cdot \Delta_x(\mathcal{D}_5)
  \end{aligned}$$

We take $\pi' = \pi$, $\pi'' = last(\pi)$, $\mathcal{D}' = \mathcal{D}_5$, and $\mathcal{D}'' = \langle \rangle$.

Lemma 5.2 Proof. We define c as follows. Let

$c' = \max\{Qual(D)(v);\ Qual(D)(v) < \infty\}$.

We define $c = (1 + c') \cdot (\Diamond G)$. Let $\pi \in D$ be a v-path. By Proposition A.3 there are π_1 and π_2, with $\pi = \pi_1 \bullet \pi_2$, and a decomposition $\mathcal{D}_1$ of π_1 with a residue ψ_1, such that both

- $\Delta_y(\mathcal{D}_1) \geq Qual(D)(v) \cdot \Delta_x(\mathcal{D}_1)$, and
- there is a decomposition $\mathcal{D}_2$ of $\psi_1 \bullet \pi_2$ where
  - $\Delta_x(\mathcal{D}_2) \geq 0$, and
  - $\Delta_y(\mathcal{D}_2) \geq Qual(D)(last(\pi)) \cdot \Delta_x(\mathcal{D}_2)$

Observe that $\Delta_y(\mathcal{D}_2) \geq 0$. By Proposition A.2 it follows that $\mathcal{D} = \mathcal{D}_1 \bullet \mathcal{D}_2$ is a decomposition of π. Furthermore:

$$\begin{aligned}
&\Delta_y(\mathcal{D}) \ \geq \ \Delta_y(\mathcal{D}_1) + \Delta_y(\mathcal{D}_2) \ \geq \ Qual(D)(v) \cdot \Delta_x(\mathcal{D}_1) + \Delta_y(\mathcal{D}_2) \ \geq \\
&Qual(D)(v) \cdot \Delta_x(\mathcal{D}_1)
\end{aligned}$$

By Proposition A.2 it follows that $\Delta_y(\pi) \geq \Delta_y(\mathcal{D}) - \Diamond G$ and $\Delta_x(\mathcal{D}_1) \geq \Delta_x(\pi_1) - \Diamond G$. It follows that

$$\begin{aligned}
&\Delta_y(\pi) \ \geq \ Qual(D)(v) \cdot \Delta_x(\pi_1) - (1 + Qual(D)(v)) \cdot (\Diamond G) \\
&Qual(D)(v) \cdot \Delta_x(\pi_1) - c
\end{aligned}$$

We take $\pi' = \pi_1$.

A.3 Proof of Lemma 6.1

Let $\pi = v_0 t_1 v_1 \cdots t_m v_m$ be a path. We say that π is $\langle v, \lambda\rangle$*-path* if $m > 1$, $\mathit{first}(\pi) = v$, and $L(t_1) = \lambda$. For a generator $\mathcal{G} = \langle \theta_1, \theta_2, \ldots, \theta_n\rangle$, we say that $\mathcal{G}$ is a $\langle v, \lambda\rangle$*-generator*, if $\langle v, \lambda\rangle \in \theta_1$.

A *quality approximation* is a function $\kappa : Det \mapsto V \times \Lambda \mapsto \mathcal{Q}^{\infty}_{\geq 0}$. We define $\kappa_{\min} : V \times \Lambda \mapsto \mathcal{Q}^{\infty}_{\geq 0}$ by $\kappa_{\min}(v, \lambda) = \min\{\kappa(D)(v, \lambda) | D \in Det\}$, and $\kappa_{\max} : Det \times V \mapsto \mathcal{Q}^{\infty}_{\geq 0}$ by $\kappa_{\max}(D)(v) = \max\{\kappa(D)(v, \lambda) | \lambda \in \Lambda\}$.

A (v, λ)-generator $\mathcal{G}$ is said to be in $Gen(D, \rho, \kappa, v, \lambda)$ if and only if either

- $\mathcal{G}$ is a type 1 generator $\langle\theta\rangle$ such that $\kappa_{\min}(v', \lambda') \geq \rho$ for every $\langle v', \lambda'\rangle$ on θ,
- $\mathcal{G}$ is a type 3 generator $\langle\theta\rangle$ such that $\alpha(\theta) \geq \rho$ and $\kappa_{\min}(v', \lambda') \geq \rho$ for every $\langle v', \lambda'\rangle$ on θ, or
- $\mathcal{G}$ is a type 2 generator $\langle\theta_1, \ldots, \theta_{m+1}\rangle$ such that
  - for every $\langle v', \lambda'\rangle$ on θ_1, $\kappa_{\min}(v', \lambda') \geq \rho$ and $\kappa_{\min}(v', \lambda') > \alpha(\theta_1)$, and
  - for each $i : 1 \leq i \leq m$ and $\langle v', \lambda'\rangle$ on θ_{i+1}, $\kappa_{\min}(v', \lambda') > \alpha(\theta_i)$.

Let $\Gamma(\kappa)(D)(v, \lambda)$ be the largest $\rho \in \mathcal{Q}^{\infty}_{\geq 0}$ such that the set $Gen(D, \rho, \kappa, v, \lambda)$ is not empty. If $Gen(D, \rho, \kappa, v, \lambda) = \emptyset$ for all ρ, let $\Gamma(\kappa)(D)(v, \lambda) = 0$.

Let κ^0 be a quality approximation such that $\kappa^0(D)(v, \lambda) = \infty$ for all D and (v, λ). Let $\kappa^{i+1} = \Gamma(\kappa)$, for $i \geq 0$, so, e.g. , $\kappa^1(D)(v, \lambda) = \Gamma(\kappa^0)(D)(v, \lambda)$ and $\kappa^2(D)(v, \lambda) = \Gamma(\kappa^1)(D)(v, \lambda)$.

Notice that $Qual(D)(v) = \kappa^1_{\max}(D)(v)$.

As an example, consider Figure 1 and Figure 2. We observe that $\langle t_8 t_{10}\rangle$ is a $\langle v_5, b\rangle$-generator of type 3 in D_3, and $\langle t_4 t_1 t_2, t_5 t_6 t_7, t_9\rangle$ is a $\langle v_2, d\rangle$-generator of type 2 in D_2. The set $Gen(D_2, 1, \kappa^0, v_3, d) = \{\langle t_5 t_6 t_9\rangle, \langle t_5 t_6 t_7, t_9\rangle\}$ while $Gen(D_2, 4, \kappa^0, v_3, d) = \{\langle t_5 t_6 t_7, t_9\rangle\}$, since $\alpha(t_5 t_6 t_9) = 2.5 < 4$. In fact, since there is a type 2 $\langle v_3, d\rangle$-generator in D_2 (namely $\langle t_5 t_6 t_7, t_9\rangle$), we have $Gen(D_2, \infty, \kappa^0, v_3, d) \neq \emptyset$, and hence $\kappa^1(D_2)(v_3, d) = \infty$.

Also, we have $Gen(D_3, 8, \kappa^0, v_3, c) = \{\langle t_2 t_3 t_4\rangle\}$, while $Gen(D_3, j, \kappa^0, v_3, c) = \emptyset$, for any $j > 8$. Consequently, we have $\kappa^1(D_3)(v_3, c) = 8$. In a similar manner we get $\kappa^1(D_1)(v_3, c) = 8$, $\kappa^1(D_2)(v_3, c) = \infty$, $\kappa^1(D_4)(v_3, c) = \infty$, and hence ${\kappa^1}_{\min}(v_3, c) = 8$.

Although $Gen(D_2, \infty, \kappa^0, v_3, d) \neq \emptyset$ we have $Gen(D_2, \infty, \kappa^1, v_3, d) = \emptyset$.
Furthermore, $Gen(D_2, 8, \kappa^1, v_3, d) = \{\langle t_2 t_1, t_5 t_6 t_7, t_9\rangle\}$. Observe that e.g. , $\langle t_5 t_6 t_7, t_9\rangle \notin Gen(D_2, 8, \kappa^1, v_3, d)$. This is due to the fact that ${\kappa^1}_{\min}(v_4, d) = 6 < 8$. In fact $\kappa^2(D_2)(v_3, d) = 8$.

Computing $Qual$ according to Definition 4.1, we obtain $Qual(v_1) = Qual(v_2) = Qual(v_3) = 8$ and $Qual(v_4) = Qual(v_5) = Qual(v_6) = 6$.

The proof of Lemma 6.1 is carried out in several steps:

- In Lemma A.6 we show that there is a deterministic subgraph D^{opt} which is optimal for player B, in the sense that the quality of each vertex in D^{opt} is equal to its minimum value over all elements of Det.
- Using the existence of D^{opt}, we show in Lemma A.7 a surprising stability property of Γ, namely that the equation $\kappa^{i+1} = \Gamma(\kappa^i)$ always reaches a fixpoint after at most two steps.

- In Lemma A.8, we use the stability property of Γ to give a characterization of optimal loops. We derive the proof of Lemma 6.1 from this characterization.

Let $\preceq_{Det}$ be an arbitrary ordering on Det. For every $\langle v, \lambda \rangle$ let $D^{\min}_{v,\lambda}$ be the smallest element with respect to $\preceq_{Det}$ in the set $\{D;\ \kappa(D)(v,\lambda) = \kappa^1{}_{\min}(v,\lambda)\}$. We define our optimal game D^{opt} such that for each $\langle v, \lambda \rangle$, we have $D^{opt}[v,\lambda] = D^{\min}_{v,\lambda}[v,\lambda]$.

We define an ordering $\preceq_{Q\times Det}$ on $\mathcal{Q}^{\infty}_{\geq 0} \times Det$ where $\langle \rho_1, D_1 \rangle \preceq_{Q\times Det} \langle \rho_2, D_2 \rangle$ if and only if either

- $\rho_1 < \rho_2$, or
- $\rho_1 = \rho_2$ and $D_1 \preceq_{Det} D_2$.

By $\beta(v,\lambda)$ we mean the pair $\left\langle \kappa^1{}_{\min}(v,\lambda), D^{\min}_{v,\lambda} \right\rangle$.

Proposition A.4 For a transition $t \in D^{opt}$, if $source(t) = v$, $target(t) = v'$, and $L(t) = \lambda$, then $\beta(v',\lambda') \preceq_{Q\times Det} \beta(v,\lambda)$, for each $\lambda' \in \Lambda$.

Proof. Since $t \in D^{opt}$, we know by definition of D^{opt} that $t \in D^{\min}_{v,\lambda}$. By definition of $D^{\min}$ it follows that $\kappa^1(D^{\min}_{v,\lambda})(v,\lambda) = \kappa^1{}_{\min}(v,\lambda)$. We know that $\kappa^1(D^{\min}_{v,\lambda})(v',\lambda') \leq \kappa^1(D^{\min}_{v,\lambda})(v,\lambda)$. By definition of $\kappa^1{}_{\min}$, $\kappa^1{}_{\min}(\lambda', v') \leq \kappa^1(D^{\min}_{v,\lambda})(v',\lambda')$. Consequently,
$\kappa^1{}_{\min}(\lambda', v') \leq \kappa^1(D^{\min}_{v,\lambda})(v',\lambda') \leq \kappa^1(D^{\min}_{v,\lambda})(v,\lambda) \leq \kappa^1{}_{\min}(v,\lambda)$.

Suppose that $\kappa^1{}_{\min}(v',\lambda') = \kappa^1{}_{\min}(v,\lambda)$. It follows that $\kappa^1(D^{\min}_{v,\lambda})(v',\lambda') = \kappa^1{}_{\min}(v',\lambda')$, and consequently $D^{\min}_{v',\lambda'} \preceq_{Det} D^{\min}_{v,\lambda}$.

Proposition A.5 For every $\langle v, \lambda \rangle$, if there is no type 2 $\langle v, \lambda \rangle$-generators in D^{opt}, then $\kappa^1(D^{opt})(v,\lambda) \leq \kappa^1{}_{\min}(v,\lambda)$.

Proof. We show that $\kappa^1(D^{opt}))(v,\lambda) \geq \rho$ implies $\kappa^1{}_{\min}(v,\lambda) \geq \rho$. We know there is a v-generator $\mathcal{G} = \langle \theta \rangle$ in D^{opt}, where either θ is of type 1, or θ is of type 3 with $\alpha(\theta) \geq \rho$. Let ψ and ϕ be the head and tail of θ respectively.

By Proposition A.4 we know that $\kappa^1{}_{\min}(v',\lambda') = \kappa^1{}_{\min}(v'',\lambda'')$ for each $\langle v', \lambda' \rangle, \langle v'', \lambda'' \rangle \in \phi$, and that there is a particular $D \in Det$ such that $D^{\min}_{v',\lambda'} = D$ for each $\langle v', \lambda' \rangle \in \phi$. By definition of D^{opt} we know $\phi \in D$. By definitions of D, κ^1, and $\kappa^1{}_{\min}$ it follows that for each $\langle v', \lambda' \rangle \in \phi$

$$\kappa^1{}_{\min}(v',\lambda') = \kappa^1(D^{\min}_{v',\lambda'})(v',\lambda') = \kappa^1(D)(v',\lambda') \geq \rho$$

To show the claim holds for each $\langle v', \lambda' \rangle \in \psi$, we use induction on $|\psi|$. The base case follows from above. The induction step follows from the induction hypothesis and the definition of D^{opt}.

Lemma A.6 For every $\langle v, \lambda \rangle$, we have $\kappa^1(D^{opt})(v,\lambda) = \kappa^1{}_{\min}(v,\lambda)$.

Proof. We prove that $\kappa^1(D^{opt})(v,\lambda) \geq \rho$ implies $\kappa^1{}_{\min}(v,\lambda) \geq \rho$. We know there is a v-generator $\mathcal{G}$ in D^{opt}, which is either of type 1, of type 3 with $\alpha(\theta) \geq \rho$, or of type 2. In the first two cases the result follows from Proposition A.5. In the second case we can assume (by definition of type 2 generators and κ^1) that $\mathcal{G}$ is of the form $\langle \theta_1, \theta_2 \rangle$. Without loss of generality we assume that $\langle v, \lambda \rangle$ is such that $\beta(v, \lambda)$ is minimal (with respect to $\preceq_{Q\times Det}$) among all elements in $V \times \Lambda$ which have type 2 generators in D^{opt}. From this assumption it follows that $\beta(v, \lambda) \preceq_{Q\times Det} \beta(v', \lambda')$, and from Proposition A.4 it follows that $\beta(v', \lambda') \preceq_{Q\times Det} \beta(v, \lambda)$ for each $\langle v', \lambda' \rangle \in \theta_1$. Consequently $\beta(v, \lambda) = \beta(v', \lambda')$ for each $\langle v', \lambda' \rangle \in \theta_1$. There are two cases:

1. If $\beta(v', \lambda') = \beta(v, \lambda)$ for each $\langle v', \lambda' \rangle \in \theta_2$. Using a proof similar to that of Proposition A.5 we can show that there is a $\langle v, \lambda \rangle$-generator of type 2 in D^{opt} and hence ${\kappa^1}_{\min}(v, \lambda) = \infty \geq \kappa^1(D^{opt})(v, \lambda)$.

2. If there is a $\langle v_0, \lambda_0 \rangle \in \theta_2$ where $\beta(v_0, \lambda_0) \neq \beta(v, \lambda)$, we assume without loss of generality that $\langle v_0, \lambda_0 \rangle$ is the first such a pair in θ_2. This means that there is $\theta_2' \preceq \theta_2$ where $last(\theta_2') = v_0$ and for each $\langle v', \lambda' \rangle \in \theta_2'$, it is the case that $\beta(v', \lambda') = \beta(v, \lambda)$. In a similar manner to the proof of Proposition A.5 there is a particular $D \in Det$ such that $\theta_1 \bullet \theta_2' \in D$ and $D^{\min}_{v', \lambda'} = D$ for each $\langle v', \lambda' \rangle \in \theta_1 \bullet \theta_2'$. It follows from Proposition A.4 that $\beta(v_0, \lambda_0) \prec_{Q\times Det} \beta(v, \lambda)$. By the assumption on $\langle v, \lambda \rangle$, it follows that there are no type 2 $\langle v_0, \lambda_0 \rangle$-generators in D^{opt}. Proposition A.5 implies that ${\kappa^1}_{\min}(v_0, \lambda_0) = \kappa^1(D^{opt})(v_0, \lambda_0)$. By definition of type 2 generators we know that $\alpha(\theta_1) < \alpha(\theta_2)$. We get

$$\alpha(\theta_1) < \alpha(\theta_2) \leq \kappa^1(D^{opt})(v_0, \lambda_0) = {\kappa^1}_{\min}(v_0, \lambda_0) \leq \kappa^1(D)(v_0, \lambda_0)$$

It follows that there exists a $\langle v, \lambda \rangle$-generator of type 2 in D and hence ${\kappa^1}_{\min}(v, \lambda) = \infty \geq \kappa^1(D^{opt})(v, \lambda)$.

Let ψ be a simple computation and θ be a loop such that there is a $v \in V$ where $v \in \psi$ and $v \in \theta$. We define $\psi \oplus \theta$ to be a loop with a head ψ' and a tail ϕ' given as follows. Let $\psi_1 \preceq \psi$ be the shortest simple computation such that $last(\psi_1) \in \theta$. We know that $\psi = \psi_1 \bullet \psi_2$ for some ψ_2. Let ψ_3 and ϕ_1 be respectively the head and tail of θ. There are two cases.

- If $last(\psi_1) \in \psi_3$, we know that $\psi_3 = \psi_4 \bullet \psi_5$ where $last(\psi_4) = last(\psi_1)$. We define $\psi' = \psi_1 \bullet \psi_5$, and $\phi' = \phi_1$.
- If $last(\psi_1) \in \phi_1$, we know that $\phi_1 = \psi_6 \bullet \psi_7$ where $last(\psi_1) = first(\psi_7) = last(\psi_6)$. We define $\psi' = \psi_1$, and $\phi' = \psi_7 \bullet \psi_6$.

Let θ_1 and θ_2 be loops such that there is a $v \in V$ where $v \in \theta_1$ and $v \in \theta_2$. We define $\theta_1 \otimes \theta_2$ as follows. Let $\psi_1 \preceq \theta_1$ be the shortest computation such that $last(\psi_1) \in \theta_2$. We know that $\theta_1 = \psi_1 \bullet \pi$ where $|\pi| > 0$ and ψ_1 is a simple computation. We define $\theta_1 \otimes \theta_2 = \psi_1 \oplus \theta_2$. We assume $\otimes$ to be right-associative.

Lemma A.7 $\kappa^2_{\min} = {\kappa^1}_{\min}$

Notice that Lemma A.7 implies $\kappa^3 = \kappa^2$ (the fact that $\kappa^3 = \kappa^2$ is not used later in the proof.)

Proof. We work by contradiction. Let ρ be the smallest rational (or ∞) such that $\kappa^2(D)(v, \lambda) < {\kappa^1}_{\min}(v, \lambda) = \rho$, for some D and $\langle v, \lambda \rangle$. We fix particular D and $\langle v, \lambda \rangle$ with this property. Define a deterministic game D^* by

- $D^*[v', \lambda'] = D^{opt}[v', \lambda']$, if ${\kappa^1}_{\min}(D^{opt})(v', \lambda') < \rho$ (in such a case we say that $\langle v', \lambda' \rangle$ is in the D^{opt}-zone), and
- $D^*[v', \lambda'] = D[v', \lambda']$ otherwise (in this case we say that $\langle v', \lambda' \rangle$ is in the D-zone).

Since $\kappa^1(D^*)(v,\lambda) \geq \kappa^1{}_{\min}(v,\lambda) \geq \rho$ there exists a generator $\mathcal{G} \in Gen(D^*, \rho, \kappa^0, v, \lambda)$. We show that there is $\mathcal{G}' \in Gen(D, \rho, \kappa^1, v, \lambda)$, and (consequently) $\kappa^2(D)(v,\lambda) \geq \rho$, which proves the result.

During the proof we make use of the following observation

observation If $v' \xrightarrow{\lambda'} v''$ and $\langle v', \lambda' \rangle$ is in the D^{opt}-zone then $\langle v'', \lambda'' \rangle$ is in the D^{opt}-zone for each λ''. **end of observation**

There are two cases:

1. If $\mathcal{G} = \langle \theta \rangle$ is of type 1 or type 3.

First, we show that θ lies entirely in the D-zone. Assume that this is not the case. By definition there is $\langle v_0, \lambda_0 \rangle \in \theta$ such that $\kappa^1(D^{opt})(v_0, \lambda_0) < \rho$. Furthermore, by the observation it follows that there is a simple computation ψ and a loop θ', where $\theta = \psi \bullet \theta'$, $\mathit{first}(\theta') = v_0$, and θ' lies entirely in the D^{opt}-zone. This means that either $\kappa^1(D^{opt})(v_0, \lambda_0) = \infty$ (if θ is $\langle +, - \rangle$) or $\kappa^1(D^{opt})(v_0, \lambda_0) \geq \alpha(\theta') = \alpha(\theta) \geq \rho$ (if θ is $\langle -, - \rangle$), which is a contradiction since $\kappa^1(D^{opt})(v_0, \lambda_0) < \rho$.

By definition it follows that $\kappa^1(D^{opt})(v', \lambda') \geq \rho$ for each $\langle v', \lambda' \rangle \in \theta$. By definition of κ^2 it follows that $\kappa^2(D)(v,\lambda) \geq \rho$.

2. If $\mathcal{G}$ is of type 2, by definitions of κ^1 and type 2 generators we can assume that $\mathcal{G}$ is of the form $\langle \theta_1, \theta_2 \rangle$. Notice that $\rho = \infty$.

First, we show that θ_1 lies entirely in the D-zone. Assume that this not the case. By definition there is $\langle v_0, \lambda_0 \rangle \in \theta_1$ such that $\kappa^1(D^{opt})(v_0, \lambda_0) < \infty$. By the observation we know that there is a simple computation ψ_1 and a loop θ_1' where $\theta_1 = \psi_1 \bullet \theta_1'$, $v_0 = \mathit{first}(\theta_1')$, and both θ_1' and θ_2 lie entirely in the D^{opt}-zone. This means that there exists a $\langle v_0, \lambda_0 \rangle$-generator (namely $\langle \theta_1', \theta_2 \rangle$) of type 2 in D^{opt}, and hence $\kappa^1(D^{opt})(v_0, \lambda_0) = \infty$ which is a contradiction since $\kappa^1(D^{opt})(v_0, \lambda_0) < \infty$ by assumption.

It follows that $\kappa^1(D^{opt})(v', \lambda') = \infty$ and (by Lemma A.6) that $\kappa^1{}_{\min}(v', \lambda') = \infty$, for each $\langle v', \lambda' \rangle \in \theta_1$.

There are two cases.

1a. if θ_2 lies entirely in the D-zone. By definition we have $\kappa^1(D^{opt})(v', \lambda') = \infty$ for each $\langle v', \lambda' \rangle \in \theta_2$. By Lemma A.6 we get $\kappa^1{}_{\min}(v', \lambda') = \infty$, for each $\langle v', \lambda' \rangle \in \theta_2$. By definition of κ^2 it follows that $\kappa^2(D)(v,\lambda) = \infty$.

1b. If there is a $\langle v_0, \lambda_0 \rangle \in \theta_2$ such that $\langle v_0, \lambda_0 \rangle$ is in the D^{opt}-zone (i.e. $\kappa^1(D^{opt})(v_0, \lambda_0) = \rho_0 < \infty = \rho$) for some ρ_0. By definition of ρ it follows that $\kappa^2(D)(v_0, \lambda_0) \geq \rho_0$. Without loss of generality we can assume that $\langle v_0, \lambda_0 \rangle$ is the earliest such a pair on θ_2. It follows that there is a simple computation ψ_2 and a loop θ_2' where $\theta_2 = \psi_2' \bullet \theta_2'$, $v_0 = \mathit{first}(\theta_2')$, and θ_2' lies entirely in the D^{opt}-zone. This means that $\rho_0 > \alpha(\theta_1)$ since

$$\rho_0 \;=\; \kappa^1(D^{opt})(v_0, \lambda_0) \;\geq\; \alpha(\theta_2') \;=\; \alpha(\theta_2) \;>\; \alpha(\theta_1)$$

Also, since $\kappa^2(D)(v_0, \lambda_0) = \rho_0$, there is a $\langle v_0, \lambda_0 \rangle$-generator $\mathcal{G}_0 \in Gen(D, v_0, \lambda_0, \rho_0, \kappa^1)$.

There are two cases:

- If $\mathcal{G}_0 = \langle \theta_0 \rangle$ is of type 1 or type 3, we take $\mathcal{G}' = \langle \theta_1, \psi_2' \oplus \theta_0 \rangle$.
- If $\mathcal{G}_0 = \left\langle \theta_0^{(1)}, \ldots, \theta_0^{(m+1)} \right\rangle$ is of type 3. Let p be the minimum i such that $\alpha(\theta_1) > \alpha(\theta_0^{(i)})$, or $p = m+1$ if no such an i exists. We define

$$\mathcal{G}_0 = \left\langle \theta_1, \psi_2' \oplus \left(\theta_0^{(1)} \otimes \cdots \otimes \theta_0^{(p)}\right), \theta_0^{(p+1)}, \ldots, \theta_0^{(m+1)} \right\rangle.$$

Lemma A.8 For $v \in V$, $\lambda \in \Lambda$, $D \in Det$, and a coefficient ω_j, where $j \geq 1$, if $\kappa^1(D)(v,\lambda) \geq \omega_j > 0$, then there exists a $\langle v,\lambda\rangle$-loop $\hat{\theta}(D,\omega_j)(v,\lambda)$ (or simply $\hat{\theta}$) in D such that both

- ${\kappa^1}_{\min}(v',\lambda') \geq \omega_j$ for every $\langle v',\lambda'\rangle \in \hat{\theta}$, and
- one of the following holds:
  - $\hat{\theta}$ is a $\langle +,-\rangle$-loop,
  - $\hat{\theta}$ is a $\langle -,-\rangle$-loop and $\alpha(\hat{\theta}) \geq \omega_j$, or
  - $\hat{\theta}$ is a $\langle +,+\rangle$-loop and $\alpha(\hat{\theta}) < \omega_{j-1}$

Proof. Since $\kappa^1(D)(v,\lambda) \geq \omega_j$, it follows by Lemma A.7 that $\kappa^2(D)(v,\lambda) \geq \omega_j$, and hence by definition of κ^2 there is $\mathcal{G} \in Gen(D,\omega_j,\kappa^1,v,\lambda)$.

If $\mathcal{G} = \langle\theta\rangle$ is of type 1, or type 3, we can take $\hat{\theta} = \theta$.

If $\mathcal{G} = \langle\theta_1,\ldots,\theta_{m+1}\rangle$ is of type 2, we consider three cases.

1. If for some $p : 1 \leq p \leq m$ there is $\langle v',\lambda'\rangle \in \theta_{p+1}$ such that ${\kappa^1}_{\min}(v',\lambda') < \omega_j$, we take the least p with this property. Observe that ${\kappa^1}_{\min}(v',\lambda') \leq \omega_{j-1}$, and ${\kappa^1}_{\min}(v'',\lambda'') \geq \omega_j$ for each $\langle v'',\lambda''\rangle \in \theta_j$ where $1 \leq j \leq p$. By definition of κ^2 it follows that $\alpha(\theta_p) < \omega_{j-1}$, We take $\hat{\theta} = \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_p$.

2. If there is no p satisfying the condition in **1**, and θ_{m+1} is either a $\langle -,-\rangle$-loop with $\alpha(\theta_{m+1}) \geq \omega_j$, or a $\langle +,-\rangle$-loop. We take $\hat{\theta} = \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_{m+1}$.

3. If there is no p satisfying the condition in **1**, and θ_{m+1} is a $\langle -,-\rangle$-loop with $\alpha(\theta_{m+1}) < \omega_j$. This means that $\alpha(\theta_{m+1}) \leq \omega_{j-1}$. By definition of type 3 generators we know that $\alpha(\theta_m) < \omega_{j-1}$. We take $\hat{\theta} = \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_m$.

Lemma 6.1 **Proof.** We define c_1, c_2, and c_3 as follows. Let $\omega_0,\omega_1,\ldots,\omega_n$ be the game coefficients. First we define

$$\begin{aligned} c_1' &= \min\{\omega_j - \omega_{j-1};\ 0 < j < n\} \\ c_1'' &= \min\{\omega_j - \alpha(\theta);\ 0 < j < n \text{ and } \theta \text{ is a } \langle +,+\rangle\text{-loop with } \alpha(\theta) < \omega_j\} \end{aligned}$$

If c_1'' is undefined according to the above equation then we define $c_1'' = \infty$. We take

$$\begin{aligned} c_1 &= \frac{\min\{c_1', c_1'', 1\}}{\Diamond G} \\ c_2 &= c_3 = (1+\omega_{n-1}) \cdot \Diamond G \end{aligned}$$

Since $Qual(v) \geq \omega_j$, we know from Definition 4.1 and the definition of $\kappa^1_{\max}$, that there is a $\lambda \in \Lambda$ such that $\kappa^1(D)(v,\lambda) \geq \omega_j$. We define $\theta^{opt}(D,\omega_j)(v) = \hat{\theta}(D,\omega_j)(v,\lambda)$.

Claim 1 follows from Lemma A.8. By Lemma A.8 we know that π is the unfolding of a loop θ, where θ is either $\langle +,-\rangle$, $\langle -,-\rangle$ with $\alpha(\theta) \geq \omega_j$, or $\langle +,+\rangle$ with $\alpha(\theta) < \omega_{j-1}$. Let ϕ and ψ be the head and tail of θ respectively. This means that π is of the form $\psi_1 \bullet \phi^k \bullet \psi_2$, where ψ_1 and ψ_2 are simple paths, ϕ is a simple loop and $k \geq 0$. Let $\psi = \psi_1 \bullet \psi_2$. By Lemma A.8 and the definitions of unfoldings, $\hat{\theta}$, and θ^{opt} the following holds.

- $\alpha(\theta) = \alpha(\phi)$.
- $-(\Diamond G) \leq \Delta_x(\psi), \Delta_y(\psi) \leq (\Diamond G)$.
- If θ is $\langle -,- \rangle$ then $\alpha(\phi) \geq \omega_j$ and $\Delta_x(\phi) \leq -1$.
- If θ is $\langle +,+ \rangle$ then $\alpha(\phi) < \omega_{j-1}$ and $\Delta_x(\phi) \geq 1$.
- If θ is $\langle +,- \rangle$ then $\Delta_y(\phi) - \omega_{j-1} \cdot \Delta_x(\phi) \leq -1$.
- $k \geq \frac{|\pi|}{\Diamond G}$

First we show claim 2. There are three cases, namely if θ is $\langle -,- \rangle$, $\langle +,+ \rangle$, or $\langle +,- \rangle$. If θ is $\langle -,- \rangle$ then (notice that $0 = \omega_0 < \alpha(\theta) < \omega_n = \infty$)

$$\begin{aligned}
&\Delta_y(\pi) \ \leq \ k \cdot \Delta_y(\phi) + \Delta_y(\psi) \ \leq \ k \cdot \Delta_y(\phi) + (\Diamond G) \ \leq \ k \cdot \alpha(\phi) \cdot \Delta_x(\phi) + (\Diamond G) \ \leq \\
&k \cdot \omega_j \cdot \Delta_x(\phi) + (\Diamond G) \ \leq \ k \cdot \omega_{j-1} \cdot \Delta_x(\phi) + k \cdot (\omega_j - \omega_{j-1}) \cdot \Delta_x(\phi) + (\Diamond G) \ \leq \\
&k \cdot \omega_{j-1} \cdot \Delta_x(\phi) - k \cdot (\omega_j - \omega_{j-1}) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot (k \cdot \Delta_x(\phi) + \Delta_x(\psi)) - \omega_{j-1} \cdot \Delta_x(\psi) - k \cdot (\omega_j - \omega_{j-1}) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) + \omega_{j-1} \cdot (\Diamond G) - k \cdot (\omega_j - \omega_{j-1}) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - k \cdot (\omega_j - \omega_{j-1}) + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - |\pi| \cdot \tfrac{(\omega_j - \omega_{j-1})}{\Diamond G} + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - c_1 \cdot |\pi| + c_2
\end{aligned}$$

If θ is $\langle +,+ \rangle$ then (notice that $\alpha(\theta) < \omega_{j-1}$)

$$\begin{aligned}
&\Delta_y(\pi) \ \leq \ k \cdot \Delta_y(\phi) + \Delta_y(\psi) \ \leq \ k \cdot \Delta_y(\phi) + (\Diamond G) \ \leq \ k \cdot \alpha(\phi) \cdot \Delta_x(\phi) + (\Diamond G) \ \leq \\
&k \cdot \omega_{j-1} \cdot \Delta_x(\phi) - k \cdot (\omega_{j-1} - \alpha(\phi)) \cdot \Delta_x(\phi) + (\Diamond G) \ \leq \\
&k \cdot \omega_{j-1} \cdot \Delta_x(\phi) - k \cdot (\omega_{j-1} - \alpha(\phi)) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot (k \cdot \Delta_x(\phi) + \Delta_x(\psi)) - \omega_{j-1} \cdot \Delta_x(\psi) - k \cdot (\omega_{j-1} - \alpha(\phi)) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) + \omega_{j-1} \cdot (\Diamond G) - k \cdot (\omega_{j-1} - \alpha(\phi)) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - k \cdot (\omega_{j-1} - \alpha(\phi)) + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - |\pi| \cdot \tfrac{(\omega_{j-1} - \alpha(\phi))}{\Diamond G} + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - c_1 \cdot |\pi| + c_2
\end{aligned}$$

If θ is $\langle +,- \rangle$ then

$$\begin{aligned}
&\Delta_y(\pi) \ \leq \ k \cdot \Delta_y(\phi) + \Delta_y(\psi) \ \leq \ k \cdot \Delta_y(\phi) + (\Diamond G) \ \leq \\
&k \cdot \omega_{j-1} \cdot \Delta_x(\phi) + k \cdot (\Delta_y(\phi) - \omega_{j-1} \cdot \Delta_x(\phi) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot (k \cdot \Delta_x(\phi) + \Delta_x(\psi)) - \omega_{j-1} \cdot \Delta_x(\psi) + k \cdot (\Delta_y(\phi) - \omega_{j-1} \cdot \Delta_x(\phi)) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) + \omega_{j-1}(\Diamond G) + k \cdot (\Delta_y(\phi) - \omega_{j-1} \cdot \Delta_x(\phi)) + (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) + k \cdot (\Delta_y(\phi) - \omega_{j-1} \cdot \Delta_x(\phi)) + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - k + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \\
&\omega_{j-1} \cdot \Delta_x(\pi) - |\pi| \cdot \tfrac{1}{\Diamond G} + (1 + \omega_{j-1}) \cdot (\Diamond G) \ \leq \ \omega_{j-1} \cdot \Delta_x(\pi) - c_1 \cdot |\pi| + c_2
\end{aligned}$$

Now we show 3. We observe that $\alpha(\theta) < \omega_n = \infty$ and hence $\alpha(\theta) \leq \omega_{n-1}$. There are three cases.

If θ is $\langle -,- \rangle$ then

$$\begin{aligned}
&\Delta_y(\pi) \;\leq\; k \cdot \Delta_y(\phi) + \Delta_y(\psi) \;\leq\; k \cdot \Delta_y(\phi) + (\Diamond G) \;\leq \\
&k \cdot \alpha(\phi) \cdot \Delta_x(\phi) + (\Diamond G) \;\leq\; k \cdot \omega_j \cdot \Delta_x(\phi) + (\Diamond G) \;\leq \\
&\omega_j \cdot (k \cdot \Delta_x(\phi) + \Delta_x(\psi)) - \omega_j \cdot \Delta_x(\psi) + (\Diamond G) \;\leq \\
&\omega_j \cdot \Delta_x(\pi) - \omega_j \cdot \Delta_x(\psi) + (\Diamond G) \;\leq\; \omega_j \cdot \Delta_x(\pi) + \omega_j \cdot (\Diamond G) + (\Diamond G) \;\leq \\
&\omega_j \cdot \Delta_x(\pi) + (1 + \omega_j) \cdot (\Diamond G) \;\leq\; \omega_j \cdot \Delta_x(\pi) + c_3
\end{aligned}$$

If θ is $\langle +,+ \rangle$ then we observe that $\alpha(\phi) \leq \omega_{j-1} \leq \omega_j$ and the proof can be carried out in the same manner as for $\langle -,- \rangle$ above.

If θ is $\langle +,- \rangle$ then observe that $\Delta_y(\phi) \leq 0 \leq \omega_j \cdot \Delta_x(\phi)$. We get

$$\begin{aligned}
&\Delta_y(\pi) \;\leq\; k \cdot \Delta_y(\phi) + \Delta_y(\psi) \;\leq\; k \cdot \omega_j \cdot \Delta_x(\phi) + (\Diamond G) \;\leq \\
&\omega_j \cdot (k \cdot \Delta_x(\phi) + \Delta_x(\psi)) - \omega_j \cdot \Delta_x(\psi) + (\Diamond G) \;\leq\; \omega_j \cdot \Delta_x(\pi) - \omega_j \cdot \Delta_x(\psi) + (\Diamond G) \;\leq \\
&\omega_j \cdot \Delta_x(\pi) + \omega_j \cdot (\Diamond G) + (\Diamond G) \;\leq\; \omega_j \cdot \Delta_x(\pi) + (1 + \omega_j) \cdot (\Diamond G) \;\leq\; \omega_j \cdot \Delta_x(\pi) + c_3
\end{aligned}$$

To prove 4, we observe that θ is either $\langle +,+ \rangle$ or $\langle +,- \rangle$. In each case we have

$$\Delta_x(\pi) \;\geq\; k \cdot \Delta_x(\phi) + \Delta_x(\psi) \;\geq\; \Delta_x(\psi) \;\geq\; -\Diamond G \;\geq\; -c_3$$

A.4 Proof of Lemma 6.2

Proof. Let v_i and λ_i denote $source(t_i)$ and $L(t_i)$, respectively, for each $i : 1 \leq i \leq n$.

If $k = \ell - 1$, the result follows trivially.

Suppose that $k < \ell - 1$. Since $D(\varphi_k) = D(\varphi_\ell)$, we know that $\varphi_\ell(\varphi_k^{\# t_{k+1}}) = \varphi_k(\varphi_k^{\# t_{k+1}})$. Let $r : k < r \leq \ell$ be the smallest with the property

$$\varphi_r(\varphi_k^{\# t_{k+1}}) \;=\; \varphi_k(\varphi_k^{\# t_{k+1}}) \tag{1}$$

We observe that

$$\varphi_{r-1}^{\# t_r} \;=\; \varphi_r^{\# t_r} \tag{2}$$

and that $\varphi_r(\varphi_k^{\# t_{k+1}}) = (\varphi_{r-1} \odot t_r)(\varphi_k^{\# t_{k+1}}) \neq \varphi_{r-1}(\varphi_k^{\# t_{k+1}})$. By definition of $\odot$ it follows that

$$\# t_r \;=\; \# t_{k+1} \tag{3}$$

and

$$\varphi_k^{\# t_{k+1}} \;=\; \varphi_{r-1}^{\# t_{k+1}} \tag{4}$$

Since $\varphi_k^{(\# t_{k+1})+1} = append(\varphi_k^{\# t_{k+1}}, \varphi_k(\varphi_k^{\# t_{k+1}}))$, it follows by (1) and (4) that $\varphi_k^{(\# t_{k+1})+1} = append(\varphi_{r-1}^{\# t_{k+1}}, \varphi_r(\varphi_{r-1}^{\# t_{k+1}}))$. By (2) and (3) we get

$$\varphi_k^{(\# t_{k+1})+1} \;=\; append(\varphi_r^{\# t_r}, \varphi_r(\varphi_r^{\# t_r})) \;=\; \varphi_r^{(\# t_r)+1} \tag{5}$$

Since $\varphi_{k+1} = \varphi_k \odot t_{k+1}$, we know that $\varphi_{k+1}(\varphi_k^n) = \varphi_k(\varphi_k^n)$ for any $n : n > \#t_{k+1}$. By definition of r we know that, for each $s : k < s < r$, it is the case that $\varphi_s(\varphi_k^{\#t_{k+1}}) \neq \varphi_k(\varphi_k^{\#t_{k+1}})$. This means that $\varphi_s(\varphi_k^n) = \varphi_{s+1}(\varphi_k^n)$, for each $s : k < s < r-1$ and $n : n > \#t_{k+1}$. Furthermore, since $\varphi_r = \varphi_{r-1} \odot t_r$ and $\#t_r = t_{k+1}$, it follows that $\varphi_{r_1}(\varphi_k^n) = \varphi_r(\varphi_k^n)$, for each $n : n > \#t_{k+1}$. This implies

$$\varphi_k(\varphi_k^n) \quad = \quad \varphi_r(\varphi_k^n) \text{ for each } n : n > \#t_{k+1} \tag{6}$$

From (5) and (6) we get $D(\varphi_k) = D(\varphi_r)$, which implies that $r = \ell$.

A.5 Proof of Lemma 6.3

Let c_1, c_2, c_3 be defined as described in Lemma 6.1.

1. Take any $k \geq 0$. We know that

$$\begin{array}{l}
\Delta_y(\pi(D,k)) \ \leq \ \Delta_y(\pi'(D,k)) + 1 \ \leq \\
\omega_{j-1} \cdot \Delta_x(\pi'(D,k)) - c_1 \cdot |\pi'(D,k)| + c_2 + 1 \ \leq \\
\omega_{j-1} \cdot \Delta_x(\pi(D,k)) - c_1 \cdot |\pi(D,k)| + c_2 + 1 + \omega_{j-1} + c_1
\end{array}$$

Let $d = c_2 + 1 + \omega_{j-1} + c_1$. We get

$$\begin{array}{l}
y_k \ \leq \ y + \Delta_y(P(k)) \ \leq \ y + \sum_{D \in Det} \Delta_y(\pi(D,k)) \ \leq \\
y + \sum_{D \in Det} (\omega_{j-1} \cdot \Delta_x(\pi(D,k)) - c_1 \cdot |\pi(D,k)| + d) \ \leq \\
y + \omega_{j-1} \cdot \sum_{D \in Det} \Delta_x(\pi(D,k)) - c_1 \cdot \sum_{D \in Det} |\pi(D,k)| + \sum_{D \in Det} d \ \leq \\
y + \omega_{j-1} \cdot \Delta_x(P(k)) - c_1 \cdot |P(k)| + d \cdot (\Box G) \ \leq \\
\omega_{j-1} \cdot x + y - \omega_{j-1} \cdot x + \omega_{j-1} \cdot \Delta_x(P(k)) - c_1 \cdot k + d \cdot (\Box G) \ \leq \\
\omega_{j-1} \cdot (x + \Delta_x(P(k))) - c_1 \cdot k + y - \omega_{j-1} \cdot x + d \cdot (\Box G) \ \leq \\
\omega_{j-1} \cdot x_k - c_1 \cdot k + y - \omega_{j-1} \cdot x + d \cdot (\Box G)
\end{array}$$

Since $c_1 > 0$, the claim will be satisfied if we take k to be any natural number such that

$$k \quad \geq \quad \frac{c - y + \omega_{j-1} \cdot x - d \cdot (\Diamond G)}{c_1}$$

2. There are two cases. If $\omega_j < \infty$ we know that

$$\begin{array}{l}
\Delta_y(\pi(D,k)) \ \leq \ \Delta_y(\pi'(D,k)) + 1 \ \leq \\
\omega_j \cdot \Delta_x(\pi'(D,k)) + c_3 + 1 \ \leq \\
\omega_j \cdot \Delta_x(\pi(D,k)) + c_3 + 1 + \omega_j
\end{array}$$

We get

$$\begin{array}{l}
y_j \ \leq \ y + \Delta_y(P(j)) \ \leq \ \omega_j \cdot x - c + \sum_{D \in Det} \Delta_y(\pi(n,D)) \ \leq \\
\omega_j \cdot x - c + \sum_{D \in Det} (\omega_j \cdot \Delta_x(\pi(D,k)) + c_3 + 1 + \omega_j) \ \leq \\
\omega_j \cdot x - c + \omega_j \cdot \sum_{D \in Det} \Delta_x(\pi(D,k)) + \sum_{D \in Det} (c_3 + 1 + \omega_j) \ \leq \\
\omega_j \cdot x - c + \omega_j \cdot \Delta_x(P(j)) + (c_3 + 1 + \omega_j) \cdot (\Box G) \ < \\
\omega_j \cdot (x + \Delta_x(P(j))) \ \leq \\
\omega_j \cdot x_j
\end{array}$$

If $\omega_j = \infty$ then

$$\begin{array}{l}
x + \Delta_x(P(j)) \ \geq \ c + \Delta_x(P(j)) \ \geq \ c + \sum_{D \in Det} \Delta_x(\pi(D,k)) \ \geq \\
c - \sum_{D \in Det} c_3 \ \geq \ c - c_3 \cdot (\Box G) \ \geq \ 0
\end{array}$$

A.6 Proof of Lemma 7.1

Proof. We define a function $E_{\trianglelefteq}$ to describe the pointwise difference between two columns in a partial simulation table, that is

$$E_{\trianglelefteq}(x', x) \;=\; SimTab_{\trianglelefteq}(x') - SimTab_{\trianglelefteq}(x)$$

We observe if $\langle q, r\rangle \preceq \langle q', r'\rangle$ then $E_{\trianglelefteq}(z, w)(q, r) \leq E_{\trianglelefteq}(z, w)(q', r')$, since

$$\begin{array}{l}
E_{\trianglelefteq}(z, w)(q, r) = \\
SimTab_{\trianglelefteq}(z)(q, r) - SimTab_{\trianglelefteq}(w)(q, r) = \\
SimTab_{\trianglelefteq}(z)(q', r') - D_{\trianglelefteq}(z)(q', r')(q, r) - SimTab_{\trianglelefteq}(w)(q', r') + D_{\trianglelefteq}(w)(q', r')(q, r) \leq \\
SimTab_{\trianglelefteq}(z)(q', r') - SimTab_{\trianglelefteq}(w)(q', r) = \\
E_{\trianglelefteq}(z, w)(q', r')
\end{array}$$

We define $SimTab_{\triangleleft}$ as follows

$$SimTab_{\triangleleft}(x) \;=\; \begin{cases} SimTab_{\trianglelefteq}(x) & \text{if } 0 \leq x \leq z \\[2ex] \begin{pmatrix} k * E_{\trianglelefteq}(z, w) \\ + \\ SimTab_{\trianglelefteq}(w + i) \end{pmatrix} & \begin{array}{l} \text{if } x = w + i + k * (z - w) \\ \text{for some } 1 \leq k \text{ and } 0 \leq i < z - w \end{array} \end{cases}$$

Suppose that $\langle q, x\rangle \triangleleft \langle r, y\rangle$ and $\langle q, x\rangle \stackrel{\lambda}{\Longrightarrow} \langle q', x'\rangle$. We show that there are r' and y' such that $\langle r, y\rangle \stackrel{\lambda}{\Longrightarrow} \langle r', y'\rangle$ and $\langle q', x'\rangle \triangleleft \langle r', y'\rangle$. There are five cases:

1. if $x \leq z$ and $x' \leq z$. By definitions of $\triangleleft$ and $\trianglelefteq$, we know that there are r_1 and y_1 such that $\langle r, y\rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_1\rangle$ and $\langle q', x'\rangle \triangleleft \langle r_1, y_1\rangle$. We take $\langle r', y'\rangle = \langle r_1, y_1\rangle$.

2. if $z \leq x = x'$. We know that $x = x' = w + i + k * (z - w)$ for some $1 \leq k$ and $0 \leq i < z - w$, and that $\langle q, w + i\rangle \stackrel{\lambda}{\Longrightarrow} \langle q', w + i\rangle$. Let $y_1 = SimTab_{\trianglelefteq}(w + i)(q, r)$. By definition of $\triangleleft$ it follows that $y \geq k * E_{\trianglelefteq}(z, w)(q, r) + y_1$ and that there are r_1 and y_2 such that $\langle r, y_1\rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_2\rangle$ and $y_2 \geq SimTab_{\trianglelefteq}(w + i)(q', r_1)$. Observe that $\langle q', r_1\rangle \preceq \langle q, r\rangle$, otherwise

$$\begin{array}{l}
SimTab_{\trianglelefteq}(w + i)(q', r_1) \geq \\
SimTab_{\trianglelefteq}(w + i + 1)(q, r) + 2 \geq \\
SimTab_{\trianglelefteq}(w + i)(q, r) + 2 = \\
y_1 + 2 > \\
y_2
\end{array}$$

which is a contradiction. Consequently $E_{\trianglelefteq}(z, w)(q', r_1) \leq E_{\trianglelefteq}(z, w)(q, r)$. This means that $\langle r, y\rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_3\rangle$, where

$$\begin{array}{l}
y_3 = \\
y + y_2 - y_1 \geq \\
k * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w + i)(q', r_1) \geq \\
k * E_{\trianglelefteq}(z, w)(q', r_1) + SimTab_{\trianglelefteq}(w + i)(q', r_1) = \\
SimTab_{\trianglelefteq}(x')(q', r)
\end{array}$$

From definition of $\triangleleft$ we get $\langle q', x'\rangle \triangleleft \langle r_1, y_3\rangle$. We take $\langle r', y'\rangle = \langle r_1, y_3\rangle$.

3. if $x \geq z$, $x' \geq z$, $x' < x$, and $x = w + k * (z - w)$ for some $k \geq 2$. We know that $x' = w + i + (k-1) * (z - w)$ where $0 \leq i < z - w$, and that $\langle q, z \rangle \stackrel{\lambda}{\Longrightarrow} \langle q', w + i \rangle$. Let $y_1 = SimTab_{\trianglelefteq}(z)(q, r)$. By definition of $\lhd$ it follows that $y \geq k * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w)(q, r)$ and that there are r_1 and y_2 such that $\langle r, y_1 \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_2 \rangle$ and $y_2 \geq SimTab_{\trianglelefteq}(w + i)(q', r_1)$. Observe that $\langle q', r_1 \rangle \preceq \langle q, r \rangle$, otherwise

$$\begin{array}{l} SimTab_{\trianglelefteq}(w+i)(q', r_1) \geq \\ SimTab_{\trianglelefteq}(z)(q, r) + 2 = \\ y_1 + 2 > \\ y_2 \end{array}$$

which is a contradiction. Consequently $E_{\trianglelefteq}(z, w)(q', r_1) \leq E_{\trianglelefteq}(z, w)(q, r)$. This means that $\langle r, y \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_3 \rangle$, where

$$\begin{array}{l} y_3 = \\ y + y_2 - y_1 \geq \\ k * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w)(q, r) + SimTab_{\trianglelefteq}(w+i)(q', r_1) - SimTab_{\trianglelefteq}(z)(q, r) = \\ (k-1) * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w+i)(q', r_1) \geq \\ (k-1) * E_{\trianglelefteq}(z, w)(q', r_1) + SimTab_{\trianglelefteq}(w+i)(q', r_1) = \\ SimTab_{\trianglelefteq}(x')(q', r_1) \end{array}$$

From definition of $\lhd$ we get $\langle q', x' \rangle \lhd \langle r_1, y_3 \rangle$. We take $\langle r', y' \rangle = \langle r_1, y_3 \rangle$.

4. if $x \geq z$, $x' \geq z$, $x' < x$, and $x = w + i + k * (z - w)$ for some $k \geq 1$ and $0 < i < z - w$. We know that $x' = w + i - 1 + k * (z - w)$ where $0 \leq i - 1 < z - w$, and that $\langle q, w + i \rangle \stackrel{\lambda}{\Longrightarrow} \langle q', w + i - 1 \rangle$. Let $y_1 = SimTab_{\trianglelefteq}(w + i)(q, r)$. By definition of $\lhd$ it follows that $y \geq k * E_{\trianglelefteq}(z, w)(q, r) + y_1$ and that there are r_1 and y_2 such that $\langle r, y_1 \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_2 \rangle$ and $y_2 \geq SimTab_{\trianglelefteq}(w + i - 1)(q', r_1)$. Observe that $\langle q', r_1 \rangle \preceq \langle q, r \rangle$, otherwise

$$\begin{array}{l} SimTab_{\trianglelefteq}(w+i-1)(q', r_1) \geq \\ SimTab_{\trianglelefteq}(w+i)(q, r) + 2 = \\ y_1 + 2 > \\ y_2 \end{array}$$

which is a contradiction. Consequently $E_{\trianglelefteq}(z, w)(q', r_1) \leq E_{\trianglelefteq}(z, w)(q, r)$. This means that $\langle r, y \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_3 \rangle$, where

$$\begin{array}{l} y_3 = \\ y + y_2 - y_1 \geq \\ k * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w+i)(q', r_1) \geq \\ k * E_{\trianglelefteq}(z, w)(q', r_1) + SimTab_{\trianglelefteq}(w+i)(q', r_1) = \\ SimTab_{\trianglelefteq}(x')(q', r_1) \end{array}$$

From definition of $\lhd$ we get $\langle q', x' \rangle \lhd \langle r_1, y_3 \rangle$. We take $\langle r', y' \rangle = \langle r_1, y_3 \rangle$.

5. if $x \geq z$, $x' \geq z$, $x < x'$, and $x' = w + k * (z - w)$ for some $k \geq 2$. We know that $x = w + i + (k-1) * (z - w)$ where $0 \leq i < z - w$, and that $\langle q, w + i \rangle \stackrel{\lambda}{\Longrightarrow} \langle q', z \rangle$. Let $y_1 = SimTab_{\trianglelefteq}(w+i)(q, r)$. By definition of $\lhd$ it follows that $y \geq (k-1) * E_{\trianglelefteq}(z, w)(q, r) + y_1$, and that there are r_1 and y_2 such that $\langle r, y_1 \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_2 \rangle$ and $y_2 \geq SimTab_{\trianglelefteq}(z)(q', r_1)$.

Observe that $\langle q', r_1 \rangle \preceq \langle q, r \rangle$, otherwise

$$\begin{array}{l} SimTab_{\trianglelefteq}(z)(q', r_1) \geq \\ SimTab_{\trianglelefteq}(w+i)(q', r_1) \geq \\ SimTab_{\trianglelefteq}(z)(q, r) + 2 \geq \\ SimTab_{\trianglelefteq}(w+i)(q, r) + 2 = \\ y_1 + 2 > \\ y_2 \end{array}$$

which is a contradiction. Consequently $E_{\trianglelefteq}(z, w)(q', r_1) \leq E_{\trianglelefteq}(z, w)(q, r)$. This means that $\langle r, y \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_3 \rangle$, where

$$\begin{array}{l} y_3 = \\ y + y_2 - y_1 \geq \\ (k-1) * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(z)(q', r_1) = \\ k * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w)(q', r_1) \geq \\ k * E_{\trianglelefteq}(z, w)(q', r_1) + SimTab_{\trianglelefteq}(z)(q', r_1) = \\ SimTab_{\trianglelefteq}(x')(q', r_1) \end{array}$$

From definition of $\lhd$ we get $\langle q', x' \rangle \lhd \langle r_1, y_3 \rangle$. We take $\langle r', y' \rangle = \langle r_1, y_3 \rangle$.

6. if $x \geq z$, $x' \geq z$, $x < x'$, and $x' = w + i + k * (z - w)$ for some $k \geq 1$ and $0 < i < z - w$. We know that $x = w + i - 1 + k * (z - w)$ where $0 \leq i - 1 < z - w$, and that $\langle q, w + i - 1 \rangle \stackrel{\lambda}{\Longrightarrow} \langle q', w + i \rangle$. Let $y_1 = SimTab_{\trianglelefteq}(w + i - 1)(q, r)$. By definition of $\lhd$ it follows that $y \geq k * E_{\trianglelefteq}(z, w + i - 1)(q, r) + y_1$, and that there are r_1 and y_2 such that $\langle r, y_1 \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_2 \rangle$ and $y_2 \geq SimTab_{\trianglelefteq}(w+i)(q', r_1)$. Observe that $\langle q', r_1 \rangle \preceq \langle q, r \rangle$, otherwise

$$\begin{array}{l} SimTab_{\trianglelefteq}(w+i)(q', r_1) \geq \\ SimTab_{\trianglelefteq}(w+i-1)(q', r_1) \geq \\ SimTab_{\trianglelefteq}(w+i)(q, r) + 2 \geq \\ SimTab_{\trianglelefteq}(w+i-1)(q, r) + 2 = \\ y_1 + 2 > \\ y_2 \end{array}$$

which is a contradiction. Consequently $E_{\trianglelefteq}(z, w)(q', r_1) \leq E_{\trianglelefteq}(z, w)(q, r)$. This means that $\langle r, y \rangle \stackrel{\lambda}{\Longrightarrow} \langle r_1, y_3 \rangle$, where

$$\begin{array}{l} y_3 = \\ y + y_2 - y_1 \geq \\ k * E_{\trianglelefteq}(z, w)(q, r) + SimTab_{\trianglelefteq}(w+i-1)(q', r_1) \geq \\ k * E_{\trianglelefteq}(z, w)(q', r_1) + SimTab_{\trianglelefteq}(w+i-1)(q', r_1) = \\ SimTab_{\trianglelefteq}(x')(q', r_1) \end{array}$$

From definition of $\lhd$ we get $\langle q', x' \rangle \lhd \langle r_1, y_3 \rangle$. We take $\langle r', y' \rangle = \langle r_1, y_3 \rangle$.