Distances for Weighted Transition Systems: Games and Properties

Uli Fahrenberg
IRISA/INRIA*
Rennes Cedex
France
ulrich.fahrenberg@irisa.fr

Claus Thrane
Department of Computer Science
Aalborg University
Denmark
{crt,kgl}@cs.aau.dk

Kim G. Larsen

We develop a general framework for reasoning about distances between transition systems with quantitative information. Taking as starting point an arbitrary distance on system traces, we show how this leads to natural definitions of a linear and a branching distance on states of such a transition system. We show that our framework generalizes and unifies a large variety of previously considered system distances, and we develop some general properties of our distances. We also show that if the trace distance admits a recursive characterization, then the corresponding branching distance can be obtained as a least fixed point to a similar recursive characterization. The central tool in our work is a theory of infinite path-building games with quantitative objectives.

1 Introduction

In verification of concurrent and reactive systems, one generally seeks to assert properties of systems expressed in terms of *sets of traces* (or languages) or in terms of *computation trees*. The language point of view leads to what is generally called *linear* semantics, whereas the tree point of view leads to *branching* semantics. These semantics are the extreme points in a spectrum containing a number of other useful notions; see [16] for an overview.

As emphasized in [20], working with applications in complex reactive systems or in embedded systems means that classical notions of linear and branching equivalence (or inclusion) of processes often need to be extended to accommodate *quantitative* information. This can be in relation to real-time behavior, resource usage, or can be probabilistic or stochastic information. In such a quantitative setting, equivalences and inclusions are replaced by symmetric or asymmetric *distances* between systems.

This approach of *quantitative analysis* has been taken in numerous papers by multiple authors, both in the real-time (or hybrid), in the probabilistic, and in general quantitative settings, see [2, 3, 8–12, 15, 19, 21, 26] for a (non-exhaustive) choice of references. Indeed, the quantitative approach is also useful in settings *without* quantitative information in the models, *e.g.* in [8] various distances related to implementation correctness of discrete systems are considered.

The above-mentioned dichotomy between languages and trees persists in the quantitative setting, where one hence encounters both notions of *linear* and of *branching* distances. To the best of our knowledge, the treatment of those distances, and of the relations between them, has so far been somewhat ad hoc. Indeed, the general approach appears to be to introduce some particular distances which are relevant for a particular application and then show some useful properties; in this paper, we try to unify and generalize some of these approaches.

*Most of this work was conducted while this author was still at Aalborg University.

Submitted to:
QAPL 2011

The present paper is in a sense a follow-up to previous papers [12, 26] by the same authors. In those papers, we introduce and investigate three different linear and branching distances. A paper similar in spirit to these is [2], which analyses properties of what we later will call the *point-wise* distance for weighted Kripke structures. The starting point for the present paper is then the observation of similarities between the constructions for different types of distances, which we here generalize to encompass all of them and to construct a coherent framework.

In this paper, we take the view that in practical applications, say in reactive systems, the *system distance* which measures adherence to the property which we want to verify, will be specific to the concrete domain of the application. Hence in a general framework like the one proposed here, its description must be given as an *input*. A method to obtain the actual system distances, for some desired level of interaction, is then prescribed by the framework.

In this paper we assume that this system distance input is given as a *distance on traces*: Given two sequences of executions, one needs only to define what it means for these sequences to be closely related to each other. We show that such a *trace distance* always gives rise to natural notions both of linear and of branching distance.

To relate linear and branching distances, we introduce a general notion of *simulation game with quantitative objectives*. The idea of using games for linear and branching equivalences is not new [24] and has been used in a quantitative setting *e.g.* in [9,11], but here we explore this idea in its full generality.

One interesting result which we can show in our general framework is that for all interesting trace distances, the corresponding linear and branching distance are *topologically inequivalent*. From an application point of view this means that corresponding linear and branching distances (essentially) always measure very different things and that results about one of them cannot generally be transferred to the other. This result – and indeed also its proof – is a generalization of the well-known fact that language inclusion does not imply simulation to a quantitative setting.

We also show that for the common special case that the trace distance has a recursive characterization, the associated branching distance can be obtained as a least fixed point to a similar recursive characterization. This is again a generalization of some standard facts about simulation, but shows that for a large class of branching distances, characterizations as least fixed points are available.

Acknowledgment

The authors acknowledge interesting and fruitful discussions on the topic of this work with Tom Henzinger, Pavol Černý and Arjun Radhakrishna of IST Austria.

2 From Trace Distances to System Distances

Our object of study in this work are general $\mathbb{K}$-weighted transition systems (to be defined below), where $\mathbb{K}$ is some set of weights. For applications, $\mathbb{K}$ may be further specified and admit some extra structure, but below we just assume $\mathbb{K}$ to be some finite or infinite set.

Definition 1. *A* trace *is an infinite sequence* $(\sigma_j)_{j=0}^{\infty}$ *of elements in* $\mathbb{K}$. *The set of all such traces is denoted* $\mathbb{K}^{\omega}$.

Note that we confine our study to *infinite* traces; this is mostly for convenience, to avoid issues with finite traces of different length. All our results are valid when also finite traces are allowed and the definitions changed accordingly. We write σ_j for the jth element in a trace σ, and σ^j for the trace obtained from σ by deleting elements σ_0 up to σ_{j-1}.

Definition 2. *A* $\mathbb{K}$-weighted transition system *(WTS) is a pair* $A = (S,T)$ *of sets* S,T *with* $T \subseteq S \times \mathbb{K} \times S$.

We use the familiar notation $s \xrightarrow{x} s'$ to indicate that $(s,x,s') \in T$. Note that S and T may indeed be infinite, also infinite branching. For simplicity's sake we shall follow the common assumption that all our WTS are *non-blocking*, *i.e.* that for any state $s \in S$ there is a transition $s \xrightarrow{x} s'$ in T.

A *path from* $s_0 \in S$ in a WTS (S,T) is an infinite sequence $\big(s_j \xrightarrow{x_j} s_{j+1}\big)_{j=0}^{\infty}$ of transitions in T. The set of such is denoted $\mathrm{Pa}(s_0)$. We will in some places also need *finite* paths, *i.e.* finite sequences $\big(s_j \xrightarrow{x_j} s_{j+1}\big)_{j=0}^{n}$ of transitions; the set of finite paths from s_0 is denoted $\mathrm{fPa}(s_0)$. For a finite path π as above, we let $\mathrm{len}(\pi) = n$ denote its length and $\mathrm{last}(\pi) = s_{n+1}$ its last state. We write $\pi_j = s_j$ for the $(j+1)$th state and $\mathrm{tr}(\pi)_j$ for the $(j+1)$th weight in a finite or infinite path.

A path $\pi = \big(s_j \xrightarrow{x_j} s_{j+1}\big)_{j=0}^{\infty}$ gives rise to a trace $\mathrm{tr}(\pi) = \big(x_j\big)_{j=0}^{\infty}$. The set of (infinite) *traces from* $s_0 \in S$ is denoted $\mathrm{Tr}(s_0) = \big\{\, \mathrm{tr}(\pi) \;\big|\; \pi \in \mathrm{Pa}(s_0) \big\}$.

2.1 Interlude: Hemimetrics

Before we proceed, we recall some of the notions regarding asymmetric metrics which we will be using. First, a *hemimetric* on a set X is a function $d : X \times X \to [0,\infty]$ which satisfies $d(x,x) = 0$ for all $x \in X$ and the triangle inequality $d(x,y) + d(y,z) \geq d(x,z)$ for all $x,y,z \in X$.

We will have reason to consider two different notions of equivalence of hemimetrics. Two hemimetrics d_1, d_2 on X are said to be *Lipschitz equivalent* if there are constants $m,M \in \mathbb{R}$ such that

$$m\,d_1(x,y) \leq d_2(x,y) \leq M\,d_1(x,y)$$

for all $x,y \in X$. Lipschitz equivalent hemimetrics are hence dependent on each other; intuitively, a property using one hemimetric can always be approximated using the other.

Another, weaker, notion of equivalence of hemimetrics is the following: Two hemimetrics d_1, d_2 on X are said to be *topologically equivalent* if the topologies on X generated by the open balls $B_i(x;r) = \{y \in X \mid d_i(x,y) < r\}$, for $i = 1,2$, $x \in X$, and $r > 0$, coincide. Topological equivalence hence preserves topological notions such as convergence of sequences: If a sequence (x_j) of points in X converges in one hemimetric, then it also converges in the other.

It is a standard fact that Lipschitz equivalence implies topological equivalence. From an application point-of-view, topological equivalence is interesting for showing *negative* results; proving that two hemimetrics are not topologically equivalent can be comparatively easy, and implies that intuitively, the two hemimetrics measure very different properties.

2.2 Examples of Trace Distances

The framework we are proposing in this article takes as starting point a *trace distance* defined on executions of a weighted automaton, *i.e.* a hemimetric $d_T : \mathbb{K}^\omega \times \mathbb{K}^\omega \to [0,\infty]$. In this section we introduce a number of different such trace distances, to show that the framework is applicable to a variety of interesting examples.

Discrete trace distances. The discrete trace distance on $\mathbb{K}^\omega$ is defined by $d_T(\sigma,\tau) = 0$ if $\sigma = \tau$ and $d_T(\sigma,\tau) = \infty$ otherwise. Hence only equality or inequality of traces is measured; we shall see below that this distance exactly recovers the usual Boolean framework of trace inclusion and simulation.

If $\mathbb{K}$ comes equipped with a preorder $\sqsubseteq \subseteq \mathbb{K} \times \mathbb{K}$ indicating that a label $x \in \mathbb{K}$ may be replaced by any $y \in \mathbb{K}$ with $x \sqsubseteq y$, as *e.g.* in [25], then we may refine the above distance by instead letting $d_T(\sigma,\tau) = 0$ if $\sigma_j \sqsubseteq \tau_j$ for all j and $d_T(\sigma,\tau) = \infty$ otherwise. We will see later that using this trace distance, we exactly recover the *extended simulation* of [25]; note that something similar is done in [22].

Hamming distance. If one defines a metric d on $\mathbb{K}$ by $d(x,y) = 0$ if $x = y$ and $d(x,y) = 1$ otherwise, then the sum $\sum_j d(\sigma_j,\tau_j)$ for any pair of finite traces σ, τ of equal length is precisely the well-known Hamming distance [18]. For infinite traces, some technique can be used for providing finite values for infinite sums; two such techniques are to use *limit average* or *discounting*. We can hence define the limit-average Hamming distance by $d_T(\sigma,\tau) = \liminf_{j\to\infty} \frac{1}{j}\sum_j d(\sigma_j,\tau_j)$, and for a fixed discounting factor $0 \leq \lambda < 1$, the discounted Hamming distance by $d_T(\sigma,\tau) = \sum_j \lambda^j d(\sigma_j,\tau_j)$.

Note that this approach can easily be generalized to other (hemi)metrics d on $\mathbb{K}$; indeed the discrete trace distances from above can be recovered using $d(x,y) = 0$ if $x \sqsubseteq y$ and $d(x,y) = \infty$ otherwise.

Labeled weighted transition systems. A common example of weighted systems [6–9, 26] has $\mathbb{K} = \Sigma \times \mathbb{R}$ where Σ is a discrete set of labels. Hence $x = (x^\ell, x^w) \in \mathbb{K}$ has $x^\ell \in \Sigma$ as discrete component and $x^w \in \mathbb{R}$ as real weight. A useful trace distance for this type of systems is the *point-wise distance*, see [2, 26], given by $d_T(\sigma,\tau) = \sup_j |\sigma_j^w - \tau_j^w|$ if $\sigma_j^\ell = \tau_j^\ell$ for all j and $d_T(\sigma,\tau) = \infty$ otherwise. This measures the biggest individual difference between σ and τ.

Another interesting trace distance in this setting is the *accumulated distance* [26], where individual differences in weights are added up. Again one can use limit average or discounting for infinite sums; limit-average accumulating distance is defined by

$$d_T(\sigma,\tau) = \begin{cases} \liminf\limits_{j\to\infty} \frac{1}{j}\sum_j |\sigma_j^w - \tau_j^w| & \text{if } \sigma_j^\ell = \tau_j^\ell \text{ for all } j \\ \infty & \text{otherwise} \end{cases}$$

and discounted accumulating distance, for a fixed $\lambda < 1$, by

$$d_T(\sigma,\tau) = \begin{cases} \sum_j \lambda^j |\sigma_j^w - \tau_j^w| & \text{if } \sigma_j^\ell = \tau_j^\ell \text{ for all } j \\ \infty & \text{otherwise} \end{cases}$$

This is indeed a generalization of the Hamming distance above, setting $d\big((x,w),(y,v)\big) = |w-v|$ if $x = y$ and $d\big((x,w),(y,v)\big) = \infty$ otherwise.

Also of interest is the *maximum-lead distance* from [19], where the individual weights are added up and one is concerned with the maximal difference between the accumulated weights. The definition is

$$d_T(\sigma,\tau) = \begin{cases} \sup_j \big|\sum_{i=0}^j \sigma_i^w - \sum_{i=0}^j \tau_i^w\big| & \text{if } \sigma_j^\ell = \tau_j^\ell \text{ for all } j \\ \infty & \text{otherwise} \end{cases}$$

2.3 Simulation Games

In this central section we introduce the game which we will use to define both the linear and the branching distances. We shall use some standard terminology and constructions from game theory here; for a good introduction to the subject see *e.g.* [13].

Let $A = (S, T)$ be a weighted transition system with $s, t \in S$ and $d_T : \mathbb{K}^\omega \times \mathbb{K}^\omega \to [0, \infty]$ a trace distance. Using A as a game graph, the *simulation game* played on A from (s,t) is an infinite turn-based two-player game, where we denote the strategy space of Player i by Θ_i and the utility function of Player 1 by $u : \Theta_1 \times \Theta_2 \to [0, \infty]$. As usual u determines the pay-off of Player 1; we will not use pay-offs for Player 2 here.

The game moves along transitions in A while building a pair of paths extending from s and t, according to the strategies of the players. In the terminology of [5, 14] we are playing a *partisan path-forming game*.

A *configuration* of the game is a pair of finite paths $(\pi_1, \pi_2) \in \mathrm{fPa}(s) \times \mathrm{fPa}(t)$ (*i.e.* the history) which are consecutively updated by the players as the game progresses. The players must play according to a strategy of the following types:

- $\Theta_1 = T^{\mathrm{fPa}(s) \times \mathrm{fPa}(t)}$, the set of mappings from pairs of finite paths to transitions, with the additional requirement that for all $\theta_1 \in \Theta_1$ and $(\pi_1, \pi_2) \in \mathrm{fPa}(s) \times \mathrm{fPa}(t)$, $\theta_1(\pi_1, \pi_2) = (\mathrm{last}(\pi_1), x, s')$ for some $x \in \mathbb{K}$, $s' \in S$. This is the set of Player-1 strategies which observe the complete configuration.
- Similarly, $\Theta_2 = T^{\mathrm{fPa}(s) \times \mathrm{fPa}(t)}$ with the additional requirement that for all $\theta_2 \in \Theta_2$ and $(\pi_1, \pi_2) \in \mathrm{fPa}(s) \times \mathrm{fPa}(t)$, $\theta_2(\pi_1, \pi_2) = (\mathrm{last}(\pi_2), y, t')$ for some $y \in \mathbb{K}$, $t' \in S$.
- $\tilde{\Theta}_1 = T^{\mathrm{fPa}(s)}$, the set of *blind* Player-1 strategies which cannot observe the moves of Player 2. (Blind Player-2 strategies can be defined similarly, but we will not need those here.) It is convenient to identify $\tilde{\Theta}_1$ with the subset of Θ_1 of all strategies θ_1 which satisfy $\theta_1(\pi_1, \pi_2) = \theta_1(\pi_1, \pi_2')$ for all $\pi_1 \in \mathrm{fPa}(s)$, $\pi_2, \pi_2' \in \mathrm{fPa}(t)$.
- In the proof of Proposition 4 we will also need Player-2 strategies with additional memory. Such a strategy is a mapping $\mathrm{fPa}(s) \times \mathrm{fPa}(t) \times M \to T \times M$, where M is a set used as memory.

Given a game with configuration (π_1, π_2), a *round* is played, according to a *strategy profile* (*i.e.* a pair of strategies) $(\theta_1, \theta_2) \in \Theta_1 \times \Theta_2$, by first updating π_1 according to θ_1 and then updating the resulting configuration according to θ_2. Hence we define

$$\mathrm{Round}_{(\theta_1,\theta_2)}(\pi_1, \pi_2) = \big(\pi_1 \cdot \theta_1(\pi_1, \pi_2), \pi_2 \cdot \theta_2(\pi_1 \cdot \theta_1(\pi_1, \pi_2), \pi_2)\big)$$

where $\cdot$ denotes sequence concatenation.

A strategy profile $(\theta_1, \theta_2) \in \Theta_1 \times \Theta_2$ inductively determines an infinite sequence $\big((\pi_1^j, \pi_2^j)\big)_{j=0}^\infty$ of configurations given by $(\pi_1^0, \pi_2^0) = (s,t)$ and $(\pi_1^j, \pi_2^j) = \mathrm{Round}_{(\theta_1,\theta_2)}(\pi_1^{j-1}, \pi_2^{j-1})$ for $j \geq 1$. The paths in this sequence satisfy $\pi_i^j \sqsubseteq \pi_i^{j+1}$, where $\sqsubseteq$ denotes prefix ordering, hence the limits $\pi_1(\theta_1, \theta_2) = \lim_{j\to\infty} \pi_1^j \in \mathrm{Pa}(s)$, $\pi_2(\theta_1, \theta_2) = \lim_{j\to\infty} \pi_2^j \in \mathrm{Pa}(t)$ exist (as infinite paths). We define the utility function u as

$$u(\theta_1, \theta_2) = d_T\big(\mathrm{tr}(\pi_1(\theta_1, \theta_2)), \mathrm{tr}(\pi_2(\theta_1, \theta_2))\big)$$

This determines the pay-off to Player 1 when the game is played according to strategies θ_1, θ_2. Note again that the utility function for Player 2 is left undefined; especially we make no claim as to the game being zero-sum.

The *value* of game on A from (s,t) is defined to be the optimal Player-1 pay-off

$$v(s,t) = \sup_{\theta_1 \in \Theta_1} \inf_{\theta_2 \in \Theta_2} u(\theta_1, \theta_2)$$

Observe that the game is *asymmetric*; in general, $v(s,t) \neq v(t,s)$.

A strategy $\hat{\theta}_1$ for Player 1 is said to be *optimal* if it realizes the supremum above, *i.e.* whenever $\inf_{\theta_2\in\Theta_2} u(\hat{\theta}_1,\theta_2) = \sup_{\theta_1\in\Theta_1}\inf_{\theta_2\in\Theta_2} u(\theta_1,\theta_2)$. The strategy is called ε-*optimal* for some $\varepsilon > 0$ provided that $\inf_{\theta_2\in\Theta_2} u(\hat{\theta}_1,\theta_2) \geq \sup_{\theta_1\in\Theta_1}\inf_{\theta_2\in\Theta_2} u(\theta_1,\theta_2) - \varepsilon$. Note that ε-optimal strategies always exist for any $\varepsilon > 0$, whereas optimal strategies may not.

We also recall that the game is said to be *determined* if the sup and inf above can be interchanged, *i.e.* if $v(s,t) = \inf_{\theta_2\in\Theta_2}\sup_{\theta_1\in\Theta_1} u(\theta_1,\theta_2)$. Intuitively, the game is determined if there also exist ε-optimal Player-2 strategies for any $\varepsilon > 0$ which realize the value of the game (up to ε) independent of the strategy Player 1 might choose.

The *1-blind value* of the game is defined to be

$$\tilde{v}(s,t) = \sup_{\theta_1\in\tilde{\Theta}_1}\inf_{\theta_2\in\Theta_2} u(\theta_1,\theta_2)$$

2.4 Example: Discrete Trace Distance

It may be instructive to apply the above simulation game in the context of the discrete trace distance $d_T(\sigma,\tau) = 0$ if $\sigma = \tau$, $d_T(\sigma,\tau) = \infty$ otherwise, from Section 2.2. In this case, the game has value $v(s,t) = 0$ if and only if, for every $\theta_1 \in \Theta_1$ there exist a $\theta_2 \in \Theta_2$ which in each round $i \geq 0$ of the game, in configuration (π_1^i,π_2^i), maps $\theta_2(\pi_1^i,\pi_2^i) = (\mathrm{last}(\pi_2^i),x,t_{i+1})$ whenever $\theta_1(\pi_1^i,\pi_2^i) = (\mathrm{last}(\pi_1^i),x,s_{i+1})$. Otherwise, $v(s,t) = \infty$.

Hence we have $v(s,t) = 0$ if t simulates s in the sense of [23], and $v(s,t) = \infty$ otherwise. In other words, for discrete trace distance the game reduces to the standard simulation game of [24].

Likewise, the blind value $\tilde{v}(s,t) = 0$ if and only if every $\theta_1 \in \tilde{\Theta}_1$ and corresponding path π_1 has a match $\theta_2 \in \Theta_2$ where configuration (π_1^i,π_2^i), of round $i \geq 0$ facilitates $\theta_2(\pi_1^i,\pi_2^i) = (\mathrm{last}(\pi_2^i),x,t_{i+1})$ whenever $\theta_1(\pi_1^i,\pi_2^i) = (\mathrm{last}(\pi_1^i),x,s_{i+1})$. Hence $\tilde{v}(s,t) = 0$ if $\mathrm{Tr}(s) \subseteq \mathrm{Tr}(t)$; we recover standard trace inclusion.

2.5 Linear and Branching Distance

We can now use the game introduced in Section 2.3 to define linear and branching distance:

Definition 3. *Let $A = (S,T)$ be a WTS and $s,t \in S$.*

- *The* linear distance *from s to t is the 1-blind value $d_L(s,t) = \tilde{v}(s,t)$.*
- *The* branching distance *from s to t is the value $d_B(s,t) = v(s,t)$.*

We proceed to show that the distances so defined are hemimetrics on S, *cf.* the proof of Theorem 1 in [8]. For linear distance, this also follows from Theorem 6 below, and we only include the proof for reasons of exposition. For branching distance, we have to assume in the proof below that the simulation game is *determined*; currently we do not know whether this assumption can be lifted.

Proposition 4. *Linear distance d_L is a hemimetric on S, and if the simulation game is determined, so is d_B.*

Proof. Non-negativity of d_L and d_B follow directly from the non-negativity of d_T. To prove that $d_L(s,s) = d_B(s,s) = 0$ for all $s \in S$, given any strategy $\theta_1 \in \Theta_1$, we construct a Player-2 strategy $\theta_2 \in \Theta_2$ that mimics θ_1 and attains the game value of 0, as follows:

$$\theta_2(\pi',\pi) = \begin{cases} \theta_1(\pi,\pi) & \text{if } \pi' = \pi\cdot\theta_1(\pi,\pi) \\ (\mathrm{last}(\pi),y,s') & \text{for some } (\mathrm{last}(\pi),y,s') \in T \text{ otherwise} \end{cases}$$

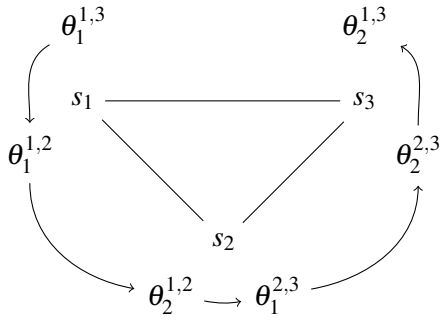

Figure 1: Construction of strategies in the proof of Proposition 4

It can be seen easily that the paths constructed by both players are the same, *i.e.* $u(\pi_1,\pi_2) = d_T(\tau,\tau)$ for some $\tau \in \mathrm{Pa}(s_0)$. Therefore, $u(\pi_1,\pi_2) = 0$ as d_T is a hemimetric, whence $d_L(s,s) = d_B(s,s) = 0$.

We are left with showing that d_L and d_B obey the triangle inequality. For linear distance, let $s_1,s_2,s_3 \in S$ and write $\Theta_k^{i,j}$ ($\tilde{\Theta}_k^{i,j}$) for the set of (blind) Player-k strategies in the simulation game computing $d_L(s_i,s_j)$, for $i,j \in \{1,2,3\}$ and $k \in \{1,2\}$. Let $\varepsilon > 0$. It might be beneficial to look at Figure 1 to see the "chase of strategies" we will be conducting.

Choose a blind Player-1 strategy $\theta_1^{1,3} \in \tilde{\Theta}_1^{1,3}$. Blind strategies correspond to choosing a path, so let $\pi_1 \in \mathrm{Pa}(s_1)$ be the path chosen by $\theta_1^{1,3}$. This path in turn corresponds to a blind Player-1 strategy $\theta_1^{1,2} \in \tilde{\Theta}_1^{1,2}$.

Let $\theta_2^{1,2} \in \Theta_2^{1,2}$ be a Player-2 strategy for which $u(\theta_1^{1,2},\theta_2^{1,2}) < d_L(s_1,s_2) + \frac{\varepsilon}{2}$. Write $\pi_2 \in \mathrm{Pa}(s_2)$ for the path constructed by the strategy profile $(\theta_1^{1,2},\theta_2^{1,2})$, and let $\theta_1^{2,3} \in \tilde{\Theta}_1^{2,3}$ be a blind Player-1 strategy which constructs π_2.

Let $\theta_2^{2,3} \in \Theta_2^{2,3}$ ensure $u(\theta_1^{2,3},\theta_2^{2,3}) < d_L(s_2,s_3) + \frac{\varepsilon}{2}$. Write $\pi_3 \in \mathrm{Pa}(s_3)$ for the path constructed by the strategy profile $(\theta_1^{2,3},\theta_2^{2,3})$, and let $\theta_2^{1,3} \in \Theta_2^{1,3}$ be a strategy which constructs π_3. For the strategy profile $(\theta_1^{1,3},\theta_2^{1,3})$ in $G_{1,3}$, the paths constructed are $\pi_1 \in \mathrm{Pa}(s_1)$ and $\pi_3 \in \mathrm{Pa}(s_3)$. Hence we have

$$\begin{aligned} \inf_{\theta_2 \in \Theta_2^{1,3}} u(\theta_1^{1,3},\theta_2) \le u(\theta_1^{1,3},\theta_2^{1,3}) &= d_T(\pi_1,\pi_3) \\ &\le d_T(\pi_1,\pi_2) + d_T(\pi_2,\pi_3) \\ &= u(\theta_1^{1,2},\theta_2^{1,2}) + u(\theta_1^{2,3},\theta_2^{2,3}) \\ &\le d_L(s_1,s_2) + d_L(s_2,s_3) + \varepsilon \end{aligned} \tag{1}$$

As $\theta_1^{1,3} \in \tilde{\Theta}_1^{1,3}$ was chosen arbitrarily, we have

$$\sup_{\theta_1 \in \tilde{\Theta}_1^{1,3}} \inf_{\theta_2 \in \Theta_2^{1,3}} u(\theta_1,\theta_2) \le d_L(s_1,s_2) + d_L(s_2,s_3) + \varepsilon$$

and as also ε was chosen arbitrarily, $d_L(s_1,s_3) \le d_L(s_1,s_2) + d_L(s_2,s_3)$.

For branching distance, we cannot construct the paths in a one-shot manner as above, as the transitions chosen by Player 1 may depend on the history of the play. Let again $\varepsilon > 0$; assuming that the simulation game is determined, we can choose Player-2 strategies $\theta_2^{1,2} \in \Theta_2^{1,2}$, $\theta_2^{2,3} \in \Theta_2^{2,3}$ for which $\sup_{\theta_1 \in \Theta_1^{1,2}} u(\theta_1,\theta_2^{1,2}) < d_B(s_1,s_2) + \frac{\varepsilon}{2}$ and $\sup_{\theta_1 \in \Theta_1^{2,3}} u(\theta_1,\theta_2^{2,3}) < d_B(s_2,s_3) + \frac{\varepsilon}{2}$. Intuitively, we will use these strategies to allow Player 2 to find replying moves to Player-1 moves in the game computing

$d_B(s_1,s_2)$ by using the reply given by $\theta_2^{2,3}$ to the reply given by $\theta_2^{1,2}$. Hence we still follow the proof strategy depicted in Figure 1, but now for individual moves.

The strategy $\theta_2^{1,3}$ uses a finite path $m = \pi_2 \in \mathrm{fPa}(s_2)$ as memory and is defined by

$$\theta_2^{1,3}(\pi_1,\pi_3)(\pi_2) = \theta_2^{2,3}\big(\pi_2 \cdot \theta_2^{1,2}(\pi_1,\pi_2),\pi_3\big)$$

with memory update $m(\pi_1,\pi_3)(\pi_2) = \pi_2 \cdot \theta_2^{1,2}(\pi_1,\pi_2)$. The initial memory for $\theta_2^{1,3}$ is set to be the empty path, hence as the game progresses, a path $\pi_2 \in \mathrm{Pa}(s_2)$ is constructed.

Now choose some $\theta_1^{1,3} \in \Theta_1^{1,3}$, and let $\pi_1 \in \mathrm{Pa}(s_1)$ and $\pi_3 \in \mathrm{Pa}(s_3)$ be the paths constructed by the strategy profile $(\theta_1^{1,3},\theta_2^{1,3})$. If $\pi_2 \in \mathrm{Pa}(s_2)$ is the corresponding memory path, then the pair (π_1,π_2) is constructed by the strategy profile $(\theta_1^{1,3},\theta_2^{1,2})$ and the pair (π_2,π_3) by the profile $(\theta_2^{1,2},\theta_2^{2,3})$. Hence we can use the exact same reasoning as in (1) to conclude that

$$\inf_{\theta_2 \in \Theta_2^{1,3}} u(\theta_1^{1,3},\theta_2) \le d_B(s_1,s_2) + d_B(s_2,s_3) + \varepsilon$$

and hence $d_B(s_1,s_3) \le d_B(s_1,s_2) + d_B(s_2,s_3)$. □

2.6 Properties

The following general result confirms that, regardless of the trace distance chosen, the linear distance is always bounded above by the branching distance. In the context of the *discrete* trace distance from Section 2.2, this specializes to the well-known fact that simulation implies language inclusion.

Theorem 5. *For any $s,t \in S$, we have $d_L(s,t) \le d_B(s,t)$.*

Proof. Any Player-1 strategy in $\tilde{\Theta}_1$ is also in Θ_1, hence

$$\sup_{\theta_1 \in \tilde{\Theta}_1} \inf_{\theta_2 \in \Theta_2} u(\theta_1,\theta_2) \le \sup_{\theta_1 \in \Theta_1} \inf_{\theta_2 \in \Theta_2} u(\theta_1,\theta_2)$$ □

The game definition of linear distance yields the following explicit formula. Note the resemblance of this to the well-known Hausdorff construction for lifting a metric on a set to its set of subsets.

Theorem 6. *For all $s,t \in S$ we have*

$$d_L(s,t) = \sup_{\sigma \in \mathrm{Tr}(s)} \inf_{\tau \in \mathrm{Tr}(t)} d_T(\sigma,\tau)$$

Proof. The definition of $\tilde{v}(s,t)$ immediately entails the fact that for any $\pi_1 \in \mathrm{Pa}(s)$ there exists $\pi_2 \in \mathrm{Pa}(t)$ such that $d_T(\mathrm{tr}(\pi_1),\mathrm{tr}(\pi_2)) \le \tilde{v}(s,t)$. It remains to show that $\tilde{v}(s,t) \le \sup_{\sigma \in \mathrm{Tr}(s)} \inf_{\tau \in \mathrm{Tr}(t)} d_T(\sigma,\tau)$. By blindness, any $\theta_1 \in \tilde{\Theta}_1$ produces a unique path $\pi_1 \in \mathrm{Pa}(s)$ independent of the opponent strategy $\theta_2 \in \Theta_2$. Hence we need only consider strategies θ_2 which define a single path π_2 from t, and the result follows.□

We finish this section by exposing two properties regarding *equivalence* of the introduced hemimetrics. Transferring (in)equivalence of distances from one setting to another is an important subject, and we hope to show other results of the below kind, especially relating trace distance to branching distance, in future work.

Proposition 7 (*cf.* [4, Thm. 3.87]). *If trace distances d_T^1 and d_T^2 are Lipschitz equivalent, then the corresponding linear distances d_L^1 and d_L^2 are topologically equivalent.*

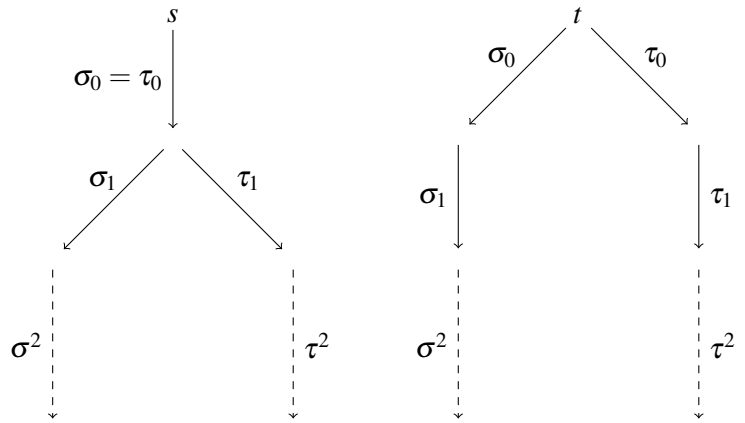

Figure 2: Weighted transition system for the proof of Proposition 9

Proof. This follows immediately from Theorem 6 and Theorem 3.87 in [4]. Note that the theorem in [4] actually is stronger; it is enough to assume d_T^1 and d_T^2 to be *uniformly equivalent*. □

The next theorem shows that if a trace distance can be used for measuring trace differences beyond the first symbol (which will be the case except for some especially trivial trace distances), then the corresponding linear and branching distances are topologically inequivalent. The proof is an easy adaption of the standard argument for the fact that language inclusion does not imply simulation.

Definition 8. *A trace distance $d_T : \mathbb{K}^\omega \times \mathbb{K}^\omega$ is said to be* one-step indiscriminate *if $\sigma_0 = \tau_0$ implies $d_T(\sigma,\tau) = 0$ for all $\sigma,\tau \in \mathbb{K}^\omega$.*

Proposition 9. *If d_T is not one-step indiscriminate, then there exists a weighted transition system A on which the corresponding distances d_L and d_B are topologically inequivalent.*

Proof. Let $\sigma,\tau \in \mathbb{K}^\omega$ such that $\sigma_0 = \tau_0$, $d_T(\sigma,\tau) > 0$, and $d_T(\tau,\sigma) > 0$. A is depicted in Figure 2.

We have $\mathrm{Tr}(s) = \mathrm{Tr}(t)$, hence $d_L(s,t) = 0$. On the other hand, $d_B(s,t) = \min\big(d_T(\sigma,\tau), d_T(\tau,\sigma)\big) > 0$. As two equivalent hemimetrics have value 0 at the same set of pairs of points, this finishes the proof. □

Also note that if σ and τ are cyclic, the construction can be adapted to yield a *finite* WTS A.

3 Recursively Defined Distances

The game definition of branching distance in Definition 3 gives a useful framework, but it is not very operational. In this section we show that if the given trace distance has a recursive characterization, then the corresponding branching distance can be obtained as the least fixed point of a similar recursive formula.

We give the fixed-point theorem first and show in Section 3.1 below that the theorem covers all examples of distances introduced earlier.

Theorem 10. *Let L be a complete lattice and $f : \mathbb{K}^\omega \times \mathbb{K}^\omega \to L$, $g : L \to [0,\infty]$, $F : \mathbb{K} \times \mathbb{K} \times L \to L$ such that $d_T = g \circ f$, g is monotone, $F(x,y,\cdot) : L \to L$ is monotone for all $x,y \in \mathbb{K}$, and*

$$f(\sigma,\tau) = F\big(\sigma_0,\tau_0,f(\sigma^1,\tau^1)\big) \tag{2}$$

for all $\sigma,\tau \in \mathbb{K}^\omega$. *Define* $I : L^{S\times S} \to L^{S\times S}$ *by*

$$I(h)(s,t) = \sup_{s \xrightarrow{x} s'\, t \xrightarrow{y} t'} \inf F\big(x,y,h(s',t')\big)$$

Then I *has a least fixed point* $h^* : S\times S \to L$, *and* $d_B = g\circ h^*$.

Let us give some intuition about the theorem before we prove it. Note first the composition $d_T = g\circ f$, where f maps pairs of traces to the lattice L which will act as *memory* in the applications below. Equation (2) then expresses that F acts as a *distance iterator function* which, within the lattice domain, computes the trace distance by looking at the first elements in the traces and then iterating over the rest of the trace. Under the premises of the theorem then, branching distance is the projection by g of the least fixed of a similar recursive function involving F.

Proof. It is not difficult to show that I indeed has a least fixed point: The lattice $L^{S\times S}$ with partial order defined point-wise by $h_1 \le h_2$ iff $h_1(s,t) \le h_2(s,t)$ for all $s,t \in S$ is complete, and I is monotone because of the monotonicity condition on F, hence Tarski's Fixed-point Theorem can be applied.

To show that $d_B = g\circ h^*$, we pull back d_B along g: With the notation for the simulation game from Section 2.3, define

$$f_B(s,t) = \sup_{\theta_1\in\Theta_1} \inf_{\theta_2\in\Theta_2} f\big(\mathrm{tr}(\pi_1(\theta_1,\theta_2)),\mathrm{tr}(\pi_2(\theta_1,\theta_2))\big)$$

We have $d_B = g\circ f_B$ by monotonicity of g, so it will suffice to show that $f_B = h^*$.

Let us first prove that f_B is a fixed point for I: Let $s,t \in S$, then

$$\begin{aligned} I(f_B)(s,t) &= \sup_{s \xrightarrow{x} s'\, t \xrightarrow{y} t'} \inf F\big(x,y,f_B(s',t')\big) \\ &= \sup_{s \xrightarrow{x} s'\, t \xrightarrow{y} t'} \inf F\Big(x,y,\sup_{\theta_1'\in\Theta_1'} \inf_{\theta_2'\in\Theta_2'} f(\mathrm{tr}(\pi_1(\theta_1',\theta_2')),\mathrm{tr}(\pi_2(\theta_1',\theta_2')))\Big) \\ &= \sup_{s \xrightarrow{x} s'\, t \xrightarrow{y} t'} \inf \sup_{\theta_1'\in\Theta_1'} \inf_{\theta_2'\in\Theta_2'} F\big(x,y,f(\mathrm{tr}(\pi_1(\theta_1',\theta_2')),\mathrm{tr}(\pi_2(\theta_1',\theta_2')))\big) \end{aligned}$$

(the last step by the monotonicity assumption on F; note that in the second sup-inf pair, strategies from s' and t' are considered). By the recursion formula (2) for F, we end up with

$$I(f_B)(s,t) = \sup_{s \xrightarrow{x} s'\, t \xrightarrow{y} t'} \inf \sup_{\theta_1'\in\Theta_1'} \inf_{\theta_2'\in\Theta_2'} f(x\cdot\mathrm{tr}(\pi_1(\theta_1',\theta_2')),y\cdot\mathrm{tr}(\pi_2(\theta_1',\theta_2')))$$

Now because of independence of choices, we can rewrite this to

$$I(f_B)(s,t) = \sup_{s \xrightarrow{x} s'} \sup_{\theta_1'\in\Theta_1'} \inf_{t \xrightarrow{y} t'} \inf_{\theta_2'\in\Theta_2'} f(x\cdot\mathrm{tr}(\pi_1(\theta_1',\theta_2')),y\cdot\mathrm{tr}(\pi_2(\theta_1',\theta_2')))$$

and collapsing the sup-sup and inf-inf into one sup and inf, respectively, conclude $I(f_B) = f_B$.

To show that f_B is the least fixed point for I, let $\bar h : S\times S \to L$ such that $I(\bar h) = \bar h$; we want to prove $f_B \le \bar h$. Note first that for all $s,t \in S$ and all $s \xrightarrow{x} s'$, there is $t \xrightarrow{y} t'$ such that $F(x,y,\bar h(s',t')) \le I(\bar h)(s,t)$. Now fix $s,t \in S$ and let $\theta_1 \in \Theta_1$; we will be done once we can construct a Player-2 strategy $\theta_2 \in \Theta_2$ for which $f(\mathrm{tr}(\pi_1(\theta_1,\theta_2)),\mathrm{tr}(\pi_2(\theta_1,\theta_2))) \le \bar h(s,t)$.

We have to define θ_2 for configurations $(\pi_1',\pi_2) \in \mathrm{fPa}(s)\times\mathrm{fPa}(t)$ in which $\pi_1' = \pi_1\cdot(s_j,x,s_{j+1})$ and $\mathrm{len}(\pi_1) = \mathrm{len}(\pi_2)$. Write $\mathrm{last}(\pi_2) = t_j$; by the note above, we can choose a transition $t_j \xrightarrow{y} t_{j+1}$ for which $F(x,y,\bar h(s',t')) \le I(\bar h)(s,t)$, so we let $\theta_2(\pi_1',\pi_2) = (t_j,y,t_{j+1})$. The so-defined strategy has

$$f(\mathrm{tr}(\pi_1(\theta_1,\theta_2)),\mathrm{tr}(\pi_2(\theta_1,\theta_2))) \le \sup_{s \xrightarrow{x} s'\, t \xrightarrow{y} t'} \inf F(x,y,\bar h(s',t')) = \bar h(s,t) \qquad \square$$

3.1 Applications

We reconsider here the example trace distances from Section 2.2 and exhibit the corresponding linear and branching distances.

Discrete trace distances. For the discrete trace distance on $\mathbb{K}^\omega$ given by $d_T(\sigma,\tau)=0$ if $\sigma=\tau$ and $d_T(\sigma,\tau)=\infty$ otherwise, we saw already in Section 2.4 that we recover ordinary trace inclusion and simulation. For linear distance, we can also use Theorem 6 to show that $d_L(s,t)=0$ if $\mathrm{Tr}(s)\subseteq\mathrm{Tr}(t)$ and $d_L(s,t)=\infty$ otherwise.

For the branching distance, we can now also apply Theorem 10 with $L=[0,\infty]$, g the identity mapping, and $F(x,y,z)=z$ if $x=y$, $F(x,y,z)=\infty$ otherwise. Then the branching distance is the least fixed point of the equations $d_B(s,t)=\sup_{s\xrightarrow{x}s'}\inf_{t\xrightarrow{x}t'}d_B(s',t')$, hence $d_B(s,t)=0$ if t simulates s in the standard sense [23], and $d_B(s,t)=\infty$ otherwise.

For the refined discrete trace distance $d_T(\sigma,\tau)=0$ if $\sigma_j\sqsubseteq\tau_j$ for all j, $d_T(\sigma,\tau)=\infty$ otherwise, we analogously get $d_L(s,t)=0$ if all $\sigma\in\mathrm{Tr}(s)$ can be refined by a $\tau\in\mathrm{Tr}(t)$ (*i.e.* $\sigma_j\sqsubseteq\tau_j$ for all j) and $d_L(s,t)=\infty$ otherwise. Also, $d_B(s,t)=0$ if there is a relation $R\subseteq S\times S$ for which $(s,t)\in R$ and whenever $(s',t')\in R$ and $s'\xrightarrow{x}s''$, then also $t'\xrightarrow{y}t''$ with $x\sqsubseteq y$ and $(s'',t'')\in R$ (the *extended simulation* of [25]), and $d_B(s,t)=\infty$ otherwise.

Hamming distance. For Hamming distance induced by the metric $d(x,y)=0$ if $x=y$ and $d(x,y)=1$ otherwise on $\mathbb{K}$, linear distance is given by $d_L(s,t)\le k$ if and only if any trace $\sigma\in\mathrm{Tr}(s)$ can be matched by a trace $\tau\in\mathrm{Tr}(t)$ with Hamming distance at most k, both for the limit-average and the discounting interpretation. The branching distance associated with the discounting version is precisely the (discounted) *correctness distance* of [8]: $d_B(s,t)$ measures "how often [the system starting in s_2] can be forced to cheat", *i.e.* to take a transition different from the one the system starting in s_1 takes.

Labeled weighted transition systems. For the trace distances on labeled weighted transition systems, let us for simplicity assume that $|\Sigma|=1$, hence $\mathbb{K}=\mathbb{R}$. For the point-wise trace distance $d_T(\sigma,\tau)=\sup_j|\sigma_j-\tau_j|$ we can derive a recursive formula for the corresponding branching distance by applying Theorem 10 with $L=[0,\infty]$, g the identity mapping, and $F(x,y,z)=\max\big(|x-y|,z\big)$. Then d_B is the least fixed point to the equations

$$d_B(s,t)=\sup_{s\xrightarrow{x}s'}\inf_{t\xrightarrow{y}t'}\max\big(|x-y|,d_B(s',t')\big)$$

For discounted accumulated trace distance $d_T(\sigma,\tau)=\sum_j\lambda^j|\sigma_j-\tau_j|$, we can similarly let $F(x,y,z)=|x-y|+\lambda z$, then the corresponding branching distance is the least fixed point to the equations

$$d_B(s,t)=\sup_{s\xrightarrow{x}s'}\inf_{t\xrightarrow{y}t'}|x-y|+\lambda d_B(s',t')$$

Note that these two branching distances are exactly the ones the authors define in [26].

For the maximum-lead distance $d_T(\sigma,\tau)=\sup_j\big|\sum_{i=0}^j\sigma_i-\sum_{i=0}^j\tau_i\big|$, we need to do more work. Intuitively, a recursive formulation needs to keep track of the accumulated delay, hence needs (infinite) memory. This can be accomplished by letting $L=[0,\infty]^{[-\infty,\infty]}$; the set of functions from leads to dis-

tances. We can then define

$$f(\sigma,\tau)(\delta) = \max\left(|\delta|, \sup_{j=0}^{\infty} |\delta + \sum_{i=0}^{j} \sigma_j - \sum_{i=0}^{j} \tau_j|\right)$$

and $g(h) = h(0)$. Now with $F(x,y,h)(\delta) = \max\left(|\delta + x - y|, h(\delta + x - y)\right)$, we indeed have that $f(\sigma,\tau) = F\left(\sigma_0, \tau_0, f(\sigma^1, \tau^1)\right)$, hence we can apply Theorem 10 to conclude that $d_B(s,t) = h^*(s,t)(0)$, where h^* is the least fixed point to the equations

$$h(s,t)(\delta) = \sup_{s \xrightarrow{x} s'} \inf_{t \xrightarrow{y} t'} \max\left(|\delta + x - y|, h(s',t')(\delta + x - y)\right)$$

This is precisely the formulation of branching maximum-lead distance given in [19].

4 Conclusion and Future Work

We have shown that simulation games with quantitative objectives provide a general framework for studying linear and branching distances for quantitative systems. Specifically, that our framework covers and unifies a number of previously distinct approaches, and that certain common special cases lead to useful recursive characterizations of branching distance.

Already we have seen that one very general property, topological inequivalence of linear and branching distance, follows almost immediately from the game characterization. Also this general approach permits the conclusion that independent of the trace distance, the branching distance provides an upper bound on the linear distance, a property which is useful for applications such as analysis of real-time systems, where linear distances are known to be uncomputable [26].

It seems likely that by permitting a broader range of strategies, we may encompass more advanced levels of system interaction and observations, and hence capture quantitative extensions of other well-known system relations such as 2-nested simulation [1, 17] or bisimulation [23]. Thus, we expect our framework to be of great use for reasoning about, and applying quantitative verification.

The game perspective on linear and branching distances also suggests that several interesting results and properties of games with quantitative objectives are transferable to our setting. As an example, one may consider computability and complexity results: For a concrete setting such as finite weighted labeled automata, discounted or limit average accumulating distances can be computed using discounted and mean-payoff games, respectively. Hence the complexity of computing these branching distances is in $\mathsf{NP} \cap \mathsf{coNP}$. Similarly, results concerning strategy iteration or value iteration for games with quantitative objectives may be transferred to the distance setting.

References

[1] Luca Aceto, Wan Fokkink & Anna Ingólfsdóttir (2001): *2-Nested Simulation Is Not Finitely Equationally Axiomatizable*. In Afonso Ferreira & Horst Reichel, editors: *STACS. Lecture Notes in Computer Science* 2010, Springer, pp. 39–50.

[2] Luca de Alfaro, Marco Faella & Mariëlle Stoelinga (2009): *Linear and Branching System Metrics*. *IEEE Trans. Software Eng.* 35(2), pp. 258–273.

[3] Luca de Alfaro, Thomas A. Henzinger & Rupak Majumdar (2003): *Discounting the Future in Systems Theory*. In: *Proc. ICALP'03. Lecture Notes in Computer Science* 2719, Springer-Verlag, pp. 1022–1037.

[4] Charalambos D. Aliprantis & Kim C. Border (2007): *Infinite Dimensional Analysis: A Hitchhiker's Guide*, 3rd edition. Springer-Verlag.

[5] Hans L. Bodlaender (1993): *Complexity of Path-Forming Games*. Theoretical Computer Science 110(1), pp. 215–245.

[6] Patricia Bouyer, Uli Fahrenberg, Kim G. Larsen, Nicolas Markey & Jiří Srba (2008): *Infinite Runs in Weighted Timed Automata with Energy Constraints*. In: Proc. FORMATS'08. Lecture Notes in Computer Science 5215, Springer-Verlag, pp. 33–47.

[7] Franck van Breugel (2005): *A Behavioural Pseudometric for Metric Labelled Transition Systems*. In: Proc. CONCUR'05. Lecture Notes in Computer Science 3653, Springer-Verlag, pp. 141–155.

[8] Pavol Černý, Thomas A. Henzinger & Arjun Radhakrishna (2010): *Simulation Distances*. In: Proc. CONCUR'10. Lecture Notes in Computer Science 6269, Springer-Verlag, pp. 253–268.

[9] Krishnendu Chatterjee, Laurent Doyen & Thomas A. Henzinger (2010): *Quantitative languages*. ACM Trans. Comput. Log. 11(4).

[10] Josee Desharnais, Vineet Gupta, Radha Jagadeesan & Prakash Panangaden (2004): *Metrics for labelled Markov processes*. Theoretical Computer Science 318(3), pp. 323–354.

[11] Josée Desharnais, François Laviolette & Mathieu Tracol (2008): *Approximate Analysis of Probabilistic Processes: Logic, Simulation and Games*. In: QEST. IEEE Computer Society, pp. 264–273.

[12] Uli Fahrenberg, Kim G. Larsen & Claus Thrane (2010): *A Quantitative Characterization of Weighted Kripke Structures in Temporal Logic*. Computing and Informatics 29.

[13] Thomas S. Ferguson: *Game Theory*. `http://www.math.ucla.edu/~tom/Game_Theory/Contents.html`.

[14] Aviezri S. Fraenkel & Shai Simonson (1993): *Geography*. Theoretical Computer Science 110(1), pp. 197–214.

[15] Alessandro Giacalone, Chi-chang Jou & Scott A. Smolka (1990): *Algebraic Reasoning for Probabilistic Concurrent Systems*. In: Proc. IFIP TC2 Working Conference on Programming Concepts and Methods. North-Holland, pp. 443–458.

[16] Rob J. van Glabbeek (2001): *The Linear Time – Branching Time Spectrum I*. In Jan A. Bergstra, Alban Ponse & Scott A. Smolka, editors: Handbook of Process Algebra, Chapter 1. Elsevier, pp. 3–99.

[17] Jan Friso Groote & Frits W. Vaandrager (1992): *Structured Operational Semantics and Bisimulation as a Congruence*. Inf. Comput. 100(2), pp. 202–260.

[18] Richard W. Hamming (1950): *Error Detecting and Error Correcting Codes*. Bell System Technical Journal 29, pp. 147–160.

[19] Thomas A. Henzinger, Rupak Majumdar & Vinayak Prabhu (2005): *Quantifying Similarities Between Timed Systems*. In: Proc. FORMATS'05. Lecture Notes in Computer Science 3829, Springer-Verlag, pp. 226–241.

[20] Thomas A. Henzinger & Joseph Sifakis (2006): *The Embedded Systems Design Challenge*. In: Proc. FM'06. Lecture Notes in Computer Science 4085, Springer-Verlag, pp. 1–15.

[21] Dexter Kozen (1983): *A Probabilistic PDL*. In: STOC. ACM, pp. 291–297.

[22] Ana Karla Alves de Medeiros, Wil M. P. van der Aalst & A. J. M. M. Weijters (2008): *Quantifying process equivalence based on observed behavior*. Data & Knowledge Engineering 64(1), pp. 55–74.

[23] Robin Milner (1989): *Communication and Concurrency*. Prentice Hall.

[24] Colin Stirling (1995): *Modal and Temporal Logics for Processes*. In: Proc. Banff Higher Order Workshop. Lecture Notes in Computer Science 1043, Springer-Verlag, pp. 149–237.

[25] Bent Thomsen (1987): *An Extended Bisimulation Induced by a Preorder on Actions*. Master's thesis, Aalborg University Centre.

[26] Claus Thrane, Uli Fahrenberg & Kim G. Larsen (2010): *Quantitative analysis of weighted transition systems*. Journal of Logic and Algebraic Programming 79(7), pp. 689–703.