Defining distances for all process semantics

David Romero Hernández, David de Frutos Escrig

Dpto. Sistemas Informáticos y Computación
Facultad CC. Matemáticas, Universidad Complutense de Madrid, Spain *
dromeroh@pdi.ucm.es, defrutos@sip.ucm.es

Abstract. Recently several authors have proposed some notions of distance between processes that try to quantify "how far away" is a process to be related with some other with respect to a certain semantics. These proposals are usually based on the simulation game, and therefore are mainly defined for simulation semantics or other semantics more or less close to these. These distances have a local character since only one of the successors of each state is taken into account in their computation. Here, we present an alternative proposal exploiting the fact that processes are trees. We define the distance between two of them as the cost of the transformations that we need to apply to get two processes related by the corresponding semantics. Our new distances can be uniformly defined for all the semantics in the ltbt-spectrum.

1 Introduction and Motivation

We are thirsty, but we hate those boring machines that only offer a few products. But we are very happy with the machine at our institution that offers a wide variety of beverages. So, each day we can go to the machine with our selected chosen item and get our bottle. But if some day the machine is out of that, then we have to choose another drink, and that day we are not so happy... Certainly, if it is only a single kind of drink that is missing we will probably stay very happy, but if something happens and the machine today offers only a single beverage, then we will be probably not so happy...

We have a collection of items in some numbered "collector desk". We look for a product by reminding its assigned number. But today, for some reason, somebody has interchanged two items and then if we look for one of them we will find the other, and we will have to make our job using it, obviously not so well as if we had found the desired item. But if one day the desk collapses and somebody has to put the items in the places without knowing their places, and he is wrong in all the cases, then for sure we will fail when looking for any of the items.

There is a lottery in the club and everybody expects that all the balls corresponding to the sold tickets will be in the bag. But for some reason the set of balls does not exactly corresponds with that of sold tickets. Certainly, the raffle

* Partially supported by the Spanish projects TESIS (TIN2009-14312-C02-01), DESAFIOS10 (TIN2009-14599-C03-01) and PROMETIDOS S2009 / TIC-1465.

is not fair, but how much unfair? An obvious reply will take into account the number of tickets that were not presented in the bag.

All these are simple "real life" situations, that we can easily model by means of a process with some kind of choice (either internal or external), where the number of choices in the initial model is large. This corresponds to the ideal situation, but if something is wrong, the choice is not the same and this would produce a process that does not fully satisfy our expectations. Then, we want to measure how far away we are from the desired behavior.

At the technical level, we want to define adequate distances between processes which measure, in a reasonable way, the gap between any behavior and the corresponding "expected" one. Of course, if we are talking about behaviors, then the first thing to fix is the reference semantics. There are plenty of proposals for process semantics, which have been presented in several versions of the linear-time branching-time (ltbt) spectrum [14, 5].

In the last few years we can find in the literature several proposals for distances between processes associated to a certain range of process semantics, but in all the cases far from being applicable to the whole spectrum [1]. Most of them, if not all, base their definitions on the (bi)simulation game that characterizes (bi)simulations between processes [11, 3, 2]. Although these are branching semantics, their co-inductive characterizations provide a (partially) local way to compare processes by considering, one by one, all the possible transitions from the compared states. The rules of these games state that any a-transition should be replicable by another a-transition of the other process; otherwise, we would have found a proof of non-bisimilarity (or that of non existence of a simulation) of the two compared processes.

Starting from them, the modified distance games allow the defender to reply an a-move by means of another b-move, where we could have $a \neq b$. Then he should pay to the attacker as the provided distance between these two actions, $d(b, a)$, states. Obviously, the attacker tries to maximize his profit by making his appropriate moves, while the defender tries to minimize them with his moves. Finally, the value of this game provides the (bi)simulation distance between the two compared processes w.r.t. the provided distance between actions, d.

Certainly, we could agree about the naturalness of these approaches, which in fact are proved to be correct, in the sense that the distance between two processes is 0, if and only if, they are (bi)similar. But, if we apply these distances to the formalizations of our three examples above, considering the discrete distance between processes (given by $d(a, a) = 0$ and $d(a, b) = 1$ if $a \neq b$) and taking p_n as the corresponding "ideal" behavior, where n is the desired number of choices, p_{n-1} the slightly "incorrect" approximation, and p_1 the poor approximation with a single choice, we obtain $d(p_n, p_{n-1}) = 1$, probably as expected, but a bit surprisingly, we also have $d(p_n, p_1) = 1$. In our opinion, it would be much more informative to get instead $d(p_n, p_1) = n - 1$, in such a way that if we consider the general approximation p_k of the ideal process p_n, which offers exactly k of the actions, then we have $d(p_n, p_k) = n - k$, and also $d(p_k, p_1) = k - 1$.

Why these known distances between processes fail to notice the quantity of choices that are lost? This is simple: just because the "local" character of the distance game. It certainly observes any of the lost actions, but this only happens at different plays of the game, each of them producing a profit $d(a_i, a_1) = 1$ to the attacker, so that the "final" profit (the value of the game, that generates $d(p_n, p_k)$) is always 1, when $k < n$, whatever the number of lost choices, $n - k$ was.

Even if we definitely advocate for a distance which will get $d(p_n, p_k) = n - k$, and in fact we will provide such a distance, we could still look for "justifications" of the distance produced by the game approaches: if we only study the computations of the processes "one by one" (certainly step by step, in order to get the characterizations of the branched semantics, instead of just the trace semantics) then we will never realize that several choices were lost at the same time (we only notice that "each one of them" was lost, but this is not enough).

What is the problem? (and then, how can we solve it?). Simulations define branched behaviors that are roughly trees which consider all computations of each process together [15]. These trees can be seen as "global" values (or full behaviors) of the process. Equality (resp. containment) of trees is defined (in a coalgebraic way) by bisimulation (resp. simulation), and then (in a partially local way) by the bisimulation (resp. simulation) game. We could say that this is the "magic" of (bi)simulation, but when we introduce distances between actions and we try to lift them up to the branched behaviors by means of the distance game, then we find that the obtained values are not able to capture the branched structure, because the value of the game is obtained by the minimax algorithm, which chooses the critical path generated by the application of the optimal strategies of both players, but is not able to "add" the differences observed at different branches. Indeed, we are using *max* instead of *add* when computing the value of the distance games, and then we cannot capture the "global" distances as required by the situations in our introductory examples.

As a matter of fact, the reason why the plain (bi)simulation game is able to capture a branched semantics is because we are interested in checking equality. This can be done by a boolean function which only considers boolean values, e.g. 0 for equal and 1 for unequal. Then, any move that the defender cannot match produces some 1, so the application of max would produce the value 1. But in this discrete domain, max can also be used to compute addition, which in fact coincides with disjunction. Instead, as soon as we have a more informative domain for the values of distances, then max and add become two different operations. It is clear that the first is only able to transmit a partial information about the branched behaviors, while addition collects all the "local" differences to compute a much more reasonable concept of global distance.

Once we have our mechanism to compute our global bisimulation distance, we will see that a quite simple customization, gives us a nice notion of distance for each of the semantics in the ltbt-spectrum. Roughly we just need to combine the preorder defining each of the other semantics in the ltbt-spectrum—see Fig.1—

(or equivalently, the inequalities that are included in their axiomatizations), with the rules which produce the values of our bisimulation distance, .

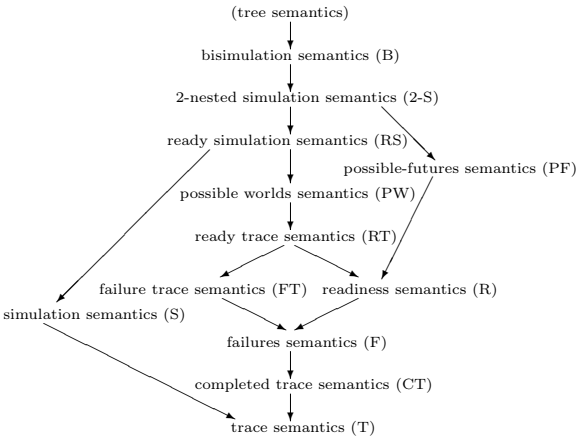

Fig. 1. The ltbt-spectrum

2 Preliminaries

All the semantics from the ltbt-spectrum [14, 5, 6] that we consider can be defined over arbitrary (possibly infinite) processes whose operational semantics is defined by means of a labelled transition system (lts) $\mathcal{P} = (Proc, Act, \rightarrow)$. We will use the classical notation $p \xrightarrow{a} p'$ to represent the transitions of processes. Moreover, it is also useful to have a syntactic notation for representing finite processes. We will use BCCSP [14, 5].

Definition 1. *Given a set of actions Act, the set BCCSP(Act) of processes is that defined by the BNF-grammar:* $p ::= \mathbf{0} \mid ap \mid p + q$. *The very well known operational semantics of BCCSP [14, 5] is defined by:*

$$(1)\ \frac{}{ap \xrightarrow{a} p} \qquad (2)\ \frac{p \xrightarrow{a} p'}{p + q \xrightarrow{a} p'} \qquad (3)\ \frac{q \xrightarrow{a} q'}{p + q \xrightarrow{a} q'}$$

In order to simplify the presentation, we start by considering a classic (symmetric) distance between actions $\overline{d} : Act \times Act \rightarrow \mathbb{N}$ with $\overline{d}(a, b) = \overline{d}(b, a)$ $\forall a, b \in Act$. Let us recall that any distance has besides to satisfy the following two properties: $\overline{d}(a, b) = 0 \Leftrightarrow a = b$, $\overline{d}(a, c) + \overline{d}(c, b) \leq \overline{d}(a, b)$ $\forall a, b, c \in Act$. Later, in Sect.6, we will discuss when an asymmetric quasi-distance could be used instead, and which is the intuitive meaning of the distances between processes that can be obtained using them.

We can represent any process as a tree (finite or infinite). Then a first approach to the definition of a distance measuring how far away is a process p of being equivalent to some other q, would study the differences between the trees which represent both processes, seeing what we have to change in order to turn them into two equivalent processes. Let us start by considering ordered trees, where we have a set of ordered sons for each node of the tree. We can present these trees as terms $\sum_{i=1}^{n} a_i p_i$, where $n = 0$ produces the empty tree $\mathbf{0}$.

Definition 2. *We say that an ordered tree p is at most at distance d from another tree q, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d_{\overline{d}}(p, q) \leq d$, if and only if:*

- $d \geq 0$ *and* $p = q = \boldsymbol{0}$*, or*
- $p = \sum_{i=1}^{n} a_i p_i$*,* $q = \sum_{i=1}^{n} b_i q_i$*, and* $d = \sum_{i=1}^{n} d_i + \sum_{i=1}^{n} \overline{d}(a_i, b_i)$ *with* $d_{\overline{d}}(p_i, q_i) \leq d_i \ \forall i = 1 \ldots n$*.*

It is clear that this definition only produces (finite) distances between trees which have exactly the same structure. For instance, for the processes $p = a + b$ and $q = c + d$ we obtain $d_{\overline{d}}(p, q) \leq \overline{d}(a, c) + \overline{d}(b, d)$. However, if we want to compare $r = a$ and $s = b + c$, we will get no finite value d for which $d_{\overline{d}}(r, s) \leq d$, and then we could say that $d_{\overline{d}}(r, s) = \infty$.

Moreover, when comparing two infinite trees we will only obtain a finite distance if the number of disagreements between them is finite. Certainly, this will be the expected result if we simply add the cost of all these mismatches. But it is important to notice that the simple approach here proposed will never been able to compute distances between infinite trees with infinitely many mismatches. Therefore, in the following we will restrict ourselves to the case of finite processes, leaving the case of infinite processes for our conclusions.

It is also true that in this simple scenario when we compare two trees with the same structure, we could directly obtain the distance between them. But we preferred to introduce this indirect presentation using bounds, because this will be later needed when considering more complicated scenarios. Certainly, the order between the summands is important in ordered trees. As a consequence, if we consider $p' = b + a$ and $\overline{d}(a, b) = 1$, we obtain $d_{\overline{d}}(p, p') \leq 2$, and definitely not $d_{\overline{d}}(p, p') \leq 0$.

But trees representing processes are unordered: each node has attached a set of subtrees, and this even implies that no identical sons are allowed. In fact, this corresponds to considering processes "up-to" bisimulation. Then, in order to define a reasonable and well behaved notion of (bound of the) distance between processes, we apply a push-out of the definition above and that of bisimulation. So we get a rewriting procedure where we try to change any of the two compared processes into the other: Either changing one of the actions in a tree by other, but then we need to pay for it, as stated by the function $\overline{d}$; or we simply apply for free to any subtree of them any of the bisimulation axioms:

$$(B1)\ x + y \simeq y + x \qquad (B2)\ x + x \simeq x$$

$$(B3)\ (x + y) + z \simeq x + (y + z) \qquad (B4)\ z + \mathbf{0} \simeq z$$

Obviously, this procedure is non-deterministic and different possible applications lead us to several (different) "distances", and this is why we need to talk about "bounds" of the distance between p and q.

Definition 3. *We say that an unordered tree p is at most at distance d from another tree q, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d_{\overline{d}}(p, q) \leq d$, if and only if:*

- *(C1) $p = ap'$, $q = bp'$, and $d \geq \overline{d}(a, b)$, or*
- *(C2) $p = p' + r$, $q = q' + r$, and $d \geq d_{\overline{d}}(p', q')$, or*
- *(C3) $p = ap'$, $q = aq'$, and $d \geq d_{\overline{d}}(p', q')$, or*
- *(C4) $d \geq 0$ and q can be obtained from p by application of (B1)-(B4), or*
- *(C5) There exist r, d' and d'' s.t. $d' \geq d_{\overline{d}}(p, r)$, $d'' \geq d_{\overline{d}}(r, q)$ and $d \geq d' + d''$.*

$(C1)$ corresponds to a single application of Def.1 producing a single change at the root of p. $(C2)$ and $(C3)$ allow the contextual application of $(C1)$ at any place, thus generating the possibility to change any action a in p by any other action b, paying $\overline{d}(a, b)$ for it. $(C4)$ introduces the possibility of transforming any process p into another bisimilar q, for free. Finally, $(C5)$ tells us that by adding the costs of the steps of any transformation that produces q from p, we obtain an upper bound of the distance between p and q.

We could obtain "the" distance between two trees by considering the minimal value d for which we have $d_{\overline{d}}(p, q) \leq d$. But unfortunately this corresponds to a global study of the set of derivations that produces the bounds. We prefer to avoid the explicit consideration of those "exact" distances, since it seems not possible to introduce the computation of these minimal values in our approach in a manageable way.

Moreover, it is easy to see that these distances would correspond to the shortest path in the graph whose nodes are processes, and the valued arcs correspond to the cost of the basic allowed transformations between them induced by rules $(C1) - (C4)$; $(C5)$ states somehow the Bellman's optimality principle. As a consequence we do not need "all the strength" of rule $(C5)$ which allows us to compose a path by concatenating two arbitrary paths, but it certainly includes the (needed) case in which the first path is a single step. However, by including this general rule we obtain a more symmetric definition, where those single steps do not need any separate treatment.

Now, by applying $(B1)$ we obtain $d_{\overline{d}}(a + b, b + a) \leq 0$. Moreover, we can compare trees that have not the same structure. For instance, we can transform for free $r = a$ into $r' = a + a$, and then we obtain $d_{\overline{d}}(r', s) \leq \overline{d}(a, b) + \overline{d}(a, c)$, from where we conclude $d_{\overline{d}}(r, s) \leq \overline{d}(a, b) + \overline{d}(a, c)$. Although it could be the case that we could obtain other "lower bounds" of this distance, as we will discuss later in Sect.3 (page 11).

Next, we present another equivalent definition of our bisimulation distance between processes. We consider processes up-to bisimulation, and following the coinductive approach, we will consider a collection of "distance relations" $\{G_m \mid m \in \mathbb{N}\}$, that are those generated by the SOS-rules below:

$$(1)\ \frac{}{p\ G_n\ p} \quad (2)\ \frac{p\ G_n q}{ap\ G_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\ G_n\ p'}{p+q\ G_n\ p'+q} \quad (4)\ \frac{p\ G_n\ q \quad q\ G_{n'} r}{p\ G_{n+n'}\ r}$$

Proposition 1. *For all $n \in \mathbb{N}$, we have $p\ G_n\ q$ if and only if $d_{\overline{d}}(p,q) \leq n$.*

Proof. It is clear the correspondence between the rules defining both collections of relations. We will only remark that $(C4)$ corresponds to working up-to bisimilarity, while rule (2) covers both $(C1)$ and $(C3)$ at the same time.

□

Remark 1. It would be possible to mix these rules in several ways, even reducing its total number. But we prefer this presentation, where basic transformations are shown in isolation. This definitely simplifies the rule-induction proofs in the following.

3 Simulation Distance

Starting from the bisimulation distance presented above, next we introduce the simulation distance. We start by recalling the definition of simulation.

Definition 4. *A simulation is a relation S between processes such that whenever we have pSq, for every $a \in Act$, if $p \xrightarrow{a} p'$ then, there exists some q', such that $q \xrightarrow{a} q'$ and $p'Sq'$. We say that process p is simulated by process q, or that q simulates p, written $p \sqsubseteq_S q$, if there exists a simulation S such that pSq.*

We want to define by means of rules the relations that indicate how far away is a process p of being simulated by another q. Of course, when q simulates p, the simulation distance between them (in this direction) will be 0. When this is not the case, we will need to change the tree that represents q, to get a process that simulates p, paying for each modification.

Definition 5. *Given two processes p and q, we say that the simulation distance from q to p is at most $m \in \mathbb{N}$, w.r.t. the symmetric distance between actions $\overline{d}$, and then we write $d^S_{\overline{d}}(p,q) \leq m$, if we can derive $p\ G^S_m\ q$ applying the following rules:*

$$(1)\ \frac{p\ \sqsubseteq_S\ q}{p\ G^S_n\ q} \quad (2)\ \frac{p\ G^S_n\ q}{ap\ G^S_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\ G^S_n\ p'}{p+q\ G^S_n\ p'+q} \quad (4)\ \frac{p\ G^S_n\ q \quad q\ G^S_{n'}\ r}{p\ G^S_{n+n'}\ r}$$

In other words, we can say that the simulation distance is obtained by computing the bisimulation distance up to the similarity relation. This can also be expressed in a transformational way: we look for the "minimal changes" that we need to make in q to get a process q' which simulates p.

Remark 2. Note that in this case we do not need to explicitly say that we work up-to bisimilarity, since when $q \sim q'$, we also have $q' \sqsubseteq_S q$, and then by applying (1) we can transmute q into q' for free, whenever this is needed.

Next we present a very simple example to illustrate how our definition works.

Example 1. We consider the lexicographic distance between actions induced by the lexicographic order, so we have $\overline{d}(a,b) = 1$, $\overline{d}(a,c) = 2$, and so on. Let us consider the processes $p = a(b+c)$ and $q = ab + ad$. Then, it is easy to see that $p \not\sqsubseteq_S q$ and $q \not\sqsubseteq_S p$. Let us start seeing how far away we are of having $q \sqsubseteq_S p$. It is clear that $q \sqsubseteq_S p'$, where p' is obtained from p by turning c into d, so that we define $p' = a(b+d)$. Therefore, we have $d^S_{\overline{d}}(p,q) \leq \overline{d}(c,d) = 1$. Next we see in detail how we can derive $q\, G^S_1\, p$ applying the rules in Def.5:

$$\dfrac{\dfrac{q \;\; \sqsubseteq_S \;\; p'}{q \;\; G^S_0 \;\; p'}(1) \qquad \dfrac{\dfrac{d \;\; G^S_{\overline{d}(c,d)} \;\; c}{b+d \;\; G^S_1 \;\; b+c}(3)}{p' \;\; G^S_{1+\overline{d}(a,a)} \;\; p}(2)}{q \;\; G^S_1 \;\; p}(4)$$

If we consider the opposite distance, which measures at which extent we have (not) $p \sqsubseteq_S q$, the shortest way to obtain some q' with $p \sqsubseteq_S q'$ is to duplicate (for free) the subtree below a, and then we change one of the b actions into c, paying for it $\overline{d}(b,c)$. So we obtain $q' = a(b+c)+ad$, which produces $d_{\overline{d}}(q,p) \leq \overline{d}(b,c) = 1$. This can be inferred applying our rules as follows:

$$\dfrac{\dfrac{p \;\; \sqsubseteq_S \;\; q'}{p \;\; G^S_0 \;\; q'}(1) \qquad \dfrac{\dfrac{\dfrac{c \;\; G^S_1 \;\; b}{b+c \;\; G^S_1 \;\; b+b}(3)}{b+c \;\; G^S_1 \;\; b}(B2)}{a(b+c) \;\; G^S_{1+\overline{d}(a,a)} \;\; ab}(2)}{q' \;\; G^S_{1+\overline{d}(a,a)} \;\; q}(3)}{p \;\; G^S_1 \;\; q}(4)$$

Next we compare the definitions of simulation distance based on the simulation game with ours.

Definition 6. *(Simulation game) Given two LTSs, L_1 and L_2, we call configurations the pairs (p,q), with $p \in L_1$ and $q \in L_2$. The simulation game is played by two players: the attacker $\mathbb{A}$ and the defender $\mathbb{D}$. The initial configuration of the game deciding if $p_0 \sqsubseteq_s q_0$, is just the pair (p_0, q_0). A round of the game, when the current configuration is (p,q), proceeds as follows:*

1. *$\mathbb{A}$ chooses a transition in L_1: $p \xrightarrow{a} p'$.*
2. *$\mathbb{D}$ must execute the same action at the other side of the board (L_2): $q \xrightarrow{a} q'$.*
3. *The game proceeds in the same way from the new configuration (p', q').*

The winner of the game is defined by the following rules: (1) Any infinite game is a win for $\mathbb{D}$. (2) $\mathbb{D}$ also wins if $\mathbb{A}$ cannot make any new move. (3) $\mathbb{A}$ wins when he makes a move, that $\mathbb{D}$ cannot reply with a transition from L_2.

Theorem 1. *$p \sqsubseteq_S q$ (resp. $p \not\sqsubseteq_S q$) if and only if $\mathbb{D}$ (resp. $\mathbb{A}$) has a winning strategy for the simulation game starting at (p,q).*

The simulation game can be turned into a (classical) simulation distance game by allowing to reply any a-move by some b-move with $b \neq a$, but then the defender should pay $\overline{d}(b,a)$ to the attacker for the mismatch. The value of the game provides the "classical" simulation distance between p and q [1]. We can obtain a coinductive characterization, which also provides a more general definition covering also infinite processes, as follows:

Definition 7. *A family of relations between processes $(S_n)_{n\in\mathbb{N}}$ is a classical simulation distance family (csdf), w.r.t. the symmetric distance between actions $\overline{d}$, when for each $(p,q) \in S_n$ we have the diagram:*

$$\begin{array}{ccc} p & S_n & q \\ \forall\ a\downarrow & \Longrightarrow & \downarrow\exists b \\ p' & S_{n-\overline{d}(b,a)} & q' \end{array}$$

We say that p and q are at most at classical simulation distance n, and then we write $d^S_{\overline{d}}(p,q) \leq n$, iff there is some csdf *$(S_n)_{n\in\mathbb{N}}$ such that pS_nq.*

Example 2. Using the distance relation $\overline{d}$ at Example 1, if we apply our Def.5, we get $d^S_{\overline{d}}(a+d, b+e) \leq 2$, but we cannot obtain $d^S_{\overline{d}}(a+d, b+e) \leq 1$. Instead, we can get a *csdf* taking $S_1 = \{(a+d, b+e)\}$ and $S_0 = \{(\mathbf{0},\mathbf{0})\}$, because $a+d \xrightarrow{a} \mathbf{0}$ can be replied by $b+e \xrightarrow{b} \mathbf{0}$ with cost 1. If we consider the discrete distance $\overline{d}$ defined by $\overline{d}(a,b) = 1 \Leftrightarrow a \neq b$, then we obtain $d_{\overline{d}}(\sum_{i=1}^n a_i, a_0) \leq n$, but $d_{\overline{d}}(\sum_{i=1}^n a_i, a_0) \not\leq n-1$, while using the classical simulation game approach we can take $S_1 = \{(\sum_{i=1}^n a_i, a_0) \mid n \in \mathbb{N}\}$ and $S_0 = \{\mathbf{0},\mathbf{0}\}$, because any move $\sum a_i \xrightarrow{a_i} \mathbf{0}$ can be replied by $a_0 \xrightarrow{a_0} \mathbf{0}$ with cost 1.

Even if we consider that our "global simulation distance", defined at Def.5, is the most adequate way to turn the simulation relation into a quantitative distance between processes, next we will show the flexibility of our approach showing that a simple variation of the system of rules defining it produces a characterization of the "classical" operational simulation distance, defined at Def.7. We only need to change rule (3), taking instead the new rule (3′), thus obtaining the revised system:

$$(1)\ \frac{p \sqsubseteq_S q}{p\ H^S_n\ q} \quad (2)\ \frac{p\,H^S_n q}{ap\ H^S_{n+\overline{d}(b,a)}\ bq} \quad (3')\ \frac{p\,H^S_n\,p'\ \ q\,H^S_{n'}\,q'}{p+q\ H^S_{max\{n,n'\}}\ p'+q'} \quad (4)\ \frac{p\,H^S_n\,q\ \ q\,H^S_{n'}r}{p\ H^S_{n+n'}\ r}$$

We will see that the use of max in this rule produces that only the cost of the simulation of the computation that is "harder to simulate" is taken into account when generating the relations H^S_n. As a consequence, the family $(H^S_n)_{n\in\mathbb{N}}$ is a *csdf* that accurately generates the classical simulation distance:

Theorem 2. *1. $(H^S_n)_{n\in\mathbb{N}}$ is a csdf.*
2. If $(S_n)_{n\in\mathbb{N}}$ is a csdf then $S_n \subseteq H^S_n$.

Proof. • $\underline{1|}$ We prove that $(H^S_n)_{n\in\mathbb{N}}$ satisfies the definition of *csdf*, by rule induction on the definition of H^S_n:

$$\begin{array}{lcccl} (1): & p & H^S_n & q & \\ & \forall a\downarrow & & \downarrow\exists\ b{=}a & (\overset{df}{\Leftarrow} p \sqsubseteq_S q) \\ & p' & H^S_n & q' & (\overset{(1)}{\Leftarrow} p' \sqsubseteq_S q') \end{array}$$

$$\begin{array}{lcccc} (2): & ap & H^S_{n+\overline{d}(b,a)} & bq & \\ & a\downarrow & & \downarrow b & \\ & p & H^S_{n+\overline{d}(b,a)-\overline{d}(b,a)} & q & \\ & & \Downarrow & & \\ & p & H^S_n & q & \end{array} \qquad \begin{array}{lcccl} (3'): & p+q & H^S_n & p'+q' & \\ & a\downarrow & & \downarrow b & \\ & p'' & H^S_{n-\overline{d}(b,a)} & p''' & \\ & & & & \Uparrow pH^S_n p' \wedge qH^S_{n'}q' \text{ with } n{\geq}n' \\ & p & H^S_n & p' & \\ & a\downarrow & & \downarrow b & \textit{(by i.h.)} \\ & p'' & H^S_{n-\overline{d}(b,a)} & p''' & \end{array}$$

$$\begin{array}{lcccl} (4): & p & H^S_{n+n'} & r & \\ & a\downarrow & & \downarrow c & \\ & p' & H^S_{n+n'-(\overline{d}(c,b)+\overline{d}(b,a))} & r' & \\ & & \Downarrow \overline{d}(c,a)\leq\overline{d}(c,b)+\overline{d}(b,c) & & \\ & p' & H^S_{n+n'-\overline{d}(c,a)} & r' & \end{array} \quad (\Leftarrow^{i.h.(4)}) \quad \begin{array}{lcccccl} & p & H^S_n & q & q & H^S_{n'} & r \\ & a\downarrow & & \downarrow b & b\downarrow & & \downarrow c \\ & p' & H^S_{n-\overline{d}(b,a)} & q' & q' & H^S_{n'-\overline{d}(c,b)} & r' \end{array}$$

• $\underline{2|}$ We use complete induction on the depth of p:

$$0\ S_n\ q\ \Rightarrow\ 0\ \sqsubseteq_s\ q\ \Rightarrow\ 0\ H^S_n\ q$$

Let $p = ap'_a + r$ and $q = bq'_b + q''$ such that

$$\begin{array}{ccccc} p = ap'_a + r & S_n & q = bq'_b + q'' \\ \forall\ a\downarrow & \Longrightarrow & \downarrow\exists\ b \\ p_a & S_{n-d(b,a)} & q'_b \end{array}$$

Then we have:

$$p'_a\ S_{n-d(b,a)}\ q'_b\ \Rightarrow\ p'_a\ H^S_{n-d(b,a)}\ q'_b\ \Rightarrow\ ap'_a\ H^S_n\ bq'_b\,.$$

This happens for all the summands of p, which means that up-to idempotence of $+$, we can assume that $p = \sum a_i p'_{a_i}$ and $q = \sum b_i q'_{b_i} + r$, where for all $i \in I$ we have $a_i p'_{a_i}\ H^S_n\ b_i q'_{b_i}$; and finally we conclude $p\ H^S_n\ q$, by applying repeatedly the rule (3), and (1) to get $\mathbf{0}\, H^S_n\, r$.

□

It is interesting to note that we have not used the transitivity rule (4) at all in the previous proof, which means that we can obtain the following corollary:

Corollary 1. *If we define $H_n^{S'}$ as H_n^S, but removing the transitivity rule (4), we have that $H_n^{S'}$ is equivalent to H_n^S.*

Proof. From the fact that H_n^S is a *csdf* we immediately obtain that $H_n^{S'}$ is too. But since in the proof of Th.2 we do not use the transitivity rule (4), we have also proved there that for any *csdf* $(S_n)_{n\in\mathbb{N}}$ we have $S_n \subseteq H_n^{S'}$. Then we have $H_n^S \subseteq H_n^{S'}$ and from their definitions we immediately obtain $H_n^{S'} \subseteq H_n^S$, from where we can conclude that H_n^S is equivalent to $H_n^{S'}$.

□

Note however, that when we consider the sum between branches in rule (3) instead of the maximum, as done in Def.5, we need indeed the transitivity rule, because in this case it cannot be "derived" from the rest of the rules. The following example shows the necessity of this rule.

Example 3. Consider the processes $p = a$ and $q = b + c$, if we want to simulate q by p, we need to change action a into both b and c. However, it is possible that it would be better to transform first a into some a', and then this a' into b and c. Without the transitivity rule we cannot generate this elaborated transformation, and then we would not get the "desired" global simulation distance. Instead, when we consider the classical simulation distance, by the triangular inequality, it is not useful to transform first a into some a' and then a' into b, because that will be always worse than transforming directly a into b.

This example also illustrates the possible interest of such an elaborated procedure in order to efficiently simulate several branches of the simulated process by a common branch of the simulating one. The cost of the transformation of a into a' is shared by the two branches, and then we only pay once for it. Note that the use (for free) of idempotence allows this double use of a common branch.

4 Bisimulation Distance

Using the bisimulation game, we can define a "classical" bisimulation distance as done in [7]. It measures how far away are two processes of being bisimilar.

Theorem 3 ([10, 12]). *$p \sim q$ (resp. $p \not\sim q$) if and only if $\mathbb{D}$ (resp. $\mathbb{A}$) has a winning strategy for the bisimulation game starting at (p, q).*

Definition 8. *A family $(R_n)_{n\in\mathbb{N}}$ is a classical bisimulation distance family (cbdf), w.r.t. the symmetric distance relation between actions $\overline{d}$, when it satisfies*

$$\begin{array}{ccc} p & R_n & q \\ \forall\, a\downarrow & \Longrightarrow & \downarrow\exists b \\ p' & R_{n-\overline{d}(b,a)} & q' \end{array} \qquad \wedge \qquad \begin{array}{ccc} p & R_n & q \\ \exists a\downarrow & \Longleftarrow & \downarrow\forall\, b \\ p' & R_{n-\overline{d}(a,b)} & q' \end{array}$$

We say that p and q are at most at classical bisimulation distance n, and then we write $d_{\overline{d}}^B(p, q) \leq n$, iff there is some cbdf $(R_n)_{n\in\mathbb{N}}$ *such that pR_nq.*

From the symmetric definition of bisimulation we immediately obtain that our classical bisimulation distance is also symmetric.

Proposition 2. *For any two processes p, q and any $n \in \mathbb{N}$, we have $d_{\bar{d}}^B(p,q) \leq n$ if and only if $d_{\bar{d}}^B(q,p) \leq n$.*

Following the same ideas that we used in Sect.3, we can obtain a rule system that produces the biggest relations H_n^B that state that the related processes are at most at distance n to be bisimilar.

Definition 9. *We consider the family of relations $(H_n^B)_{n\in\mathbb{N}}$ which are generated by applying the following rules, modulo bisimulation:*

$$(1)\ \frac{}{p\ H_n^B\ p} \quad (2)\ \frac{p\,H_n^B\,q}{ap\ H_{n+\bar{d}(b,a)}^B\ bq} \quad (3')\ \frac{p\,H_n^B\,p' \quad q\,H_{n'}^B\,q'}{p+q\,H_{max\{n,n'\}}^B\,p'+q'} \quad (4)\ \frac{p\,H_n^B\,q \quad q\,H_{n'}^B\,r}{p\ H_{n+n'}^B\ r}$$

It is nice to observe the close similarity between the rules defining this classical bisimulation distance and our previous bisimulation distance in Sect.2: in fact, if we change the max operator in $(3')$ by addition, then it is easy to check that the obtained definition is equivalent to our original one.

Remark 3. It is clear that we can remove the "up-to" bisimulation at the definition above if we explicitly introduce the bisimilarity relation in the definition, by replacing rule (1) by the following rule:

$$(1')\ \frac{p \sim q}{p\ H_n^B\ q}$$

However, we prefer our first presentation in order to stress the fact that the system of rules that defines the classical simulation distance is obtained from the one above simply adding the similarity relation to produce pairs that are "0-far" away.

We can prove the relationship between the family H_n^B defined above and the "classical" bisimulation distance relations defined at Def.8, exactly as we made for the simulation case.

Theorem 4. *1. $(H_n^B)_{n\in\mathbb{N}}$ is a cbdf.*
2. If $(R_n)_{n\in\mathbb{N}}$ is a cbdf then $R_n \subseteq H_n^B$.

Once again, we do not use rule (4) at the proof above, which allows to derive the following corollary, that is analogous to Cor.1 in Sect.3.

Corollary 2. *If we define $H_n^{B'}$ as H_n^B in Def.9, but removing the transitivity rule (4), then we obtain the same family of relations, that is $H_n^{B'} = H_n^B$, $\forall n \in \mathbb{N}$.*

5 Distances for all the semantics in the ltbt-spectrum

Inspired by the connection between the bisimulation and the simulation distances, next we define a general notion of distance between processes. It can be instantiated by any of the different semantics in the ltbt-spectrum. These distances will measure how far away is any process q of being greater than p with respect to each of the semantic preorders defining the semantics in Fig.1. Roughly speaking, to obtain these distances, we compute the cost of changing some actions in both p and q in order to obtain two new processes p' and q' which are related under the considered semantics.

We could try to base our general definitions on the "classical" simulation distance. It is defined in a similar way as the "classical" bisimulation distance. The only difference between those two definitions was the use of $\sqsubseteq_S$ at rule (1). This immediately suggests us to define the semantic distances, corresponding to any semantics defined by an order $\sqsubseteq_{\mathcal{L}}$, by means of the following system of rules:

$$(1)\ \frac{p \sqsubseteq_{\mathcal{L}} q}{p\ H_n^{\mathcal{L}}\ q} \quad (2)\ \frac{p\ H_n^{\mathcal{L}} q}{ap\ H^{\mathcal{L}}_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\ H_n^{\mathcal{L}}\ p' \quad q\ H_{n'}^{\mathcal{L}}\ q'}{p+q\ H^{\mathcal{L}}_{max\{n,n'\}}\ p'+q'} \quad (4)\ \frac{p\ H_n^{\mathcal{L}}\ q \quad q\ H_{n'}^{\mathcal{L}}\ r}{p\ H^{\mathcal{L}}_{n+n'}\ r}$$

However, when checking some simple examples we see that this "local" approach (based on max) does not produce a "reasonable" distance for some of the most popular semantics in the ltbt-spectrum. Next, we consider the case of ready simulation (RS).

Example 4. Let us consider the processes $p = b + c$ and $q = d + f$. As distance relation $\overline{d}$ between actions, we consider again the lexicographic distance. We can check that the definition above produces

$$\dfrac{\dfrac{b\ H_2^{RS}\ d \quad c\ H_3^{RS}\ f}{b+c\ H^{RS}_{max\{2,3\}}\ d+f}(3)}{p\ \ H_3^{RS}\ \ q}(df)$$

We infer $p\ H_3^{RS}\ q$, that is the result of the necessary change in the branch which needs the most expensive change. However, this is, by no means, consistent with the definition of ready simulation: In order to have $p \sqsubseteq_{RS} q$, we need that the two processes have the same initial offer. Therefore, we would need to transform the offer $\{d, f\}$ into $\{b, c\}$. We would need changes whose aggregated cost would be (at least) 4—see Example 5—, and not just 3.
Note that this problem does not appear in the simulation case, because the definition of simulation does not contain any "global" factor. But, most of the rest of the semantics, take somehow into account some "global" information that could only be obtained by combining the information taken from several separated computations. This is the case of ready sets at readiness semantics, or even the case of failures defining the failure semantics.

Certainly, we also had $p\ H_3^B\ q$ for the (classical) bisimulation distance, and then we should also expect $p\ H_3^{\mathcal{L}}\ q$ for any semantics coarser than bisimulation. But as we discussed at the end of our introduction, plain bisimilarity is able to

check the equality of the offers of two processes even if working in a local way. However, once we need to compare two unequal offers, this local procedure proves to be quite limited. Therefore, we need to recover our first proposal at Sect.2 that measures the distance between processes by adding the cost of all the changes that we have to do at all the branches of the tree that represents a process. We already saw that it provides two reasonable "global" notions of simulation and bisimulation distances. Based on it, we obtain our general definition of "global" semantic distance between processes:

Definition 10. *Given a semantics $\mathcal{L}$, defined by a preorder $\sqsubseteq_{\mathcal{L}}$, we say that a process q is at global distance at most $m \in \mathbb{N}$ of being better than some other p, w.r.t. the semantics $\mathcal{L}$ and the distance between actions $\overline{d}$, and then we write $gd^{\mathcal{L}}_{\overline{d}}(p,q) \leq n$, if we can infer $p\ G^{\mathcal{L}}_n\ q$, by applying the following rules:*

$$(1)\ \frac{p \sqsubseteq_{\mathcal{L}} q}{p\ G^{\mathcal{L}}_n\ q} \quad (2)\ \frac{p\, G^{\mathcal{L}}_n\, q}{ap\ G^{\mathcal{L}}_{n+\overline{d}(b,a)}\ bq} \quad (3)\ \frac{p\, G^{\mathcal{L}}_n\, p'}{p+q\ G^{\mathcal{L}}_n\ p'+q} \quad (4)\ \frac{p\, G^{\mathcal{L}}_n\, q \quad q\, G^{\mathcal{L}}_{n'}\, r}{p\ G^{\mathcal{L}}_{n+n'}\ r}$$

Example 5. It is easy to check that for the processes in Example 4 and the ready simulation semantics RS, we obtain now the desired distance $gd^{RS}_{\overline{d}}(p,q) \leq 4$, since we can infer applying the rules for $\mathcal{L} = RS$ that:

$$\dfrac{\dfrac{\dfrac{\dfrac{b\, G^{RS}_1\, c}{b+c\, G^{RS}_1\, c+c}(3) \quad \dfrac{\dfrac{c+c \sqsubseteq_{RS} c}{c+c\, G^{RS}_0\, c}(1) \quad \dfrac{\dfrac{c\, G^{RS}_1\, d \quad \dfrac{d \sqsubseteq_{RS} d+d}{d\, G^{RS}_0\, d+d}(1)}{c\, G^{RS}_{1+0}\, d+d}(4)}{c+c\, G^{RS}_{0+1}\, d+d}(4)}{b+c\, G^{RS}_{1+1}\, d+d}(4) \quad \dfrac{d\, G^{RS}_2\, f}{d+d\, G^{RS}_2\, d+f}(3)}{b+c\, G^{RS}_{2+2}\, d+f}(4)}{p\ G^{RS}_4\ q}(df)$$

Remark 4. As a matter of fact, we have only used rule (1) in the partial case of "idempotence". This means that the computed (bound of the) distance will also be valid for the bisimulation semantics and in fact for any other semantics in the spectrum. Of course, if we consider a coarser semantics, it could be the case that we could obtain a smaller distance by applying (1) in some other way. For instance, for the simulation semantics (S) we will easily obtain $gd^{S}_{\overline{d}(p,q)} \leq 2$.

Generally, we immediately obtain the following result that asserts that our family of distances reflects exactly the hierarchy in the ltbt-spectrum.

Proposition 3. *Whenever we have two semantics $\mathcal{L}_1$ and $\mathcal{L}_2$ and the first is finer than the latter ($\sqsubseteq_{\mathcal{L}_1} \subseteq \sqsubseteq_{\mathcal{L}_2}$), then we have $gd^{\mathcal{L}_1}_{\overline{d}}(p,q) \leq n \Rightarrow gd^{\mathcal{L}_2}_{\overline{d}}(p,q) \leq n$, for all processes p, q and any value $n \in \mathbb{N}$.*

6 Generalizations, Applications and some Conclusions

In the developments above we have preferred to consider symmetric distances between actions because in particular we wanted to apply all the notions and

technical definitions to the case of bisimulation, that is an equivalence relation and therefore symmetric. However, the rest of the semantics are typically defined by means of a preorder, instead of by an equivalence relation. This is why the consideration of asymmetric quasi-distances opens a new and quite interesting space for developments and applications of our theory.

Let us consider the case of the simulation semantics: when we have $p \sqsubseteq_S q$, this reflects that q has all the capabilities of p and possibly some others. The simulation distances presented above reflect how many changes we need to make in q in order to get a process that really simulates p. But it could be the case that q instead of directly offering the same actions offered by p, offers some others that we consider that "do perfectly the work". This situation is formally covered simply by replacing the symmetric distance between actions by an asymmetric quasi-distance, defined as follows:

Definition 11. *An asymmetric quasi-distance in a set of actions Act is a function* $d : Act \times Act \to \mathbb{N}$ *which satisfies* $d(a, a) = 0\ \forall a \in Act$*, and the triangular inequality* $d(a, b) + d(b, c) \geq d(a, c)\ \forall a, b, c \in Act$*. We will say that* $d(a, b)$ *expresses "how far away" is action* a *of covering the expectation to have a* b*.*

Remark 5. Now we can have $d(b, a) = 0$ even if $b \neq a$, and this would reflect the fact that b totally "simulates" a. Then we could replace without "cost" any occurrence of an action a in the simulated process p using the action b. Of course, now we can have $d(a, b) \neq d(b, a)$, because the cost of replacing a by b could be very different from that of replacing b by a. Finally, any asymmetric quasi-distance induces a symmetric quasi-distance, simply taking $\overline{d}(a, b) = max\{d(a, b), d(b, a)\}$. This becomes a distance if we impose that $a \neq b \Rightarrow \overline{d}(a, b) \neq 0$.

Example 6. If we consider a simple vending machine that returns no change, and a product costs 1€, then from the machine point of view a payment of 2€ for it, could be perfectly assumed. Instead, if the situation is the other way around and we pay 1€ for a product whose cost is 2€, then the company loses 1€. This would be reflected by the asymmetric quasi-distance defined by d(1€,2€) = 0 and d(2€,1€) = 1. Using it we obtain that the process where we pay 2€ instead of 1€ is at distance 0 of simulating the specification, while when we pay 1€ when a 2€ cost is specified, we would be at distance 1 of satisfying the specification.

Using the fact that all the semantics in the (extended) ltbt-spectrum are connected to some constrained simulation, we could justify the consideration of the corresponding "biased" distances. Instead, it seems not possible to define a reasonable bisimulation distance really based on an asymmetric quasi-distance. Of course, we could always do the task using the induced distance $\overline{d}$, but in this way we are "loosing" the asymmetric information in the original distance d.

We have defined our distances with natural values just to simplify the presentation, but there is no problem at all on using any other totally ordered set, such as $\mathbb{R}^+$. Moreover, if we use fixed values for the weight of any discordance along a computation (or at any place of the trees when considering "global" distances) then the distance between two (infinite) processes would become infinite

as soon as the number of discordances between them is also infinite. This would be certainly a problem, for instance, when comparing cyclic programs where any discordance will appear again at any iteration of the compared processes. Of course, the solution to this problem would consist (as proposed, e.g. in [4, 13]) on defining weighted distances. For them the weight of any disagreement at the n-th step of a computation (or at the n-th level of the unfolded processes) will decrease fast enough (for instance, the classical weights used at the literature are those defined by the exponential sequence $\frac{1}{2^n}$).

It is true, however, that we have not discussed how to obtain in a precise way the (bounds for the) distances between two infinite processes, when they "disagree" at infinitely many places. This could be done by using either finite approximations or recursion-induction rules, for the case of finite state processes. But certainly the details need a careful work.

A simpler extension solves the problem of unexpected termination. If we consider for instance our Def.3, we could extend it by adding a fixed payment f, for unexpected termination, taking $d(p, \mathbf{0}) \leq f$ and $d(\mathbf{0}, p) \leq f$, $\forall p \neq \mathbf{0}$. Instead, we could pay for each of the lost actions a quantity q_a, taking $\overline{d}(a\mathbf{0}, \mathbf{0}) \leq q_a$ and $\overline{d}(\mathbf{0}, a\mathbf{0}) \leq q_a$ $\forall a \in Act$. Of course, this second possibility would produce infinite distances if the terminated process was infinite, but weights can be also introduced here if we want to follow this approach.

We consider that starting from the basic (but quite flexible) definitions introduced in this paper we are plenty of more elaborated possibilities, which could be developed by adapting the ideas in our general theory to them. Next, we give a list of interesting directions that we expect to explore in the near future. First, we are working in a definition of *approximated testing*, where we indicate "at which extent" a process passes a test. Using this notion we can quantify the testing procedure by formalizing the quite frequent situation in practice where the specification states the *ideal* behavior of the desired implementations, but some small disagreements are tolerated by the *quality* standards. A dual application of our distances would also provide for free a nice quantification of the notion of *robustness*: given some specification p we would say that a given implementation q is n-robust w.r.t. some semantics $\mathcal{L}$ when any "n-wrong" behavior of q, that is, any q' such that $d_x^{\mathcal{L}}(q', q) \leq n$, satisfies $d_x^{\mathcal{L}}(p, q') = 0$. We can combine our approximated correctness and the quantified robustness proposed above, to define a notion of approximated robustness, where we also allow some small disagreement between p and the n-wrong behaviors of q.

Another generalization would use "contextually defined" distances between actions, that take into account the fact that several occurrences of the same action in a specification could play totally different roles. In such a case, we could specify at each state of the specification which is the distance between actions that we should use locally at each place. The distances between *pure trees*, where the application of the idempotence law is not allowed, will also capture *redundancy*, and then when investigating fault tolerance the previously discussed ideas on approximated robustness could be used to define *approximated fault tolerance*.

Finally, we could also allow negative values at the distances between actions, that would state that whenever we have $d(a, b) = -n$ then using b to simulate a we would be "improving" the quality of the system. This could amortize some other steps where we have the opposite situation. A typical application would appear when comparing two transmission protocols, and is clearly related with the previous work by Vogler and Lüttgen in [9], where "faster than" preorders where studied, and those by Kiehn and Arun-Kumar [8] on *amortized bisimulation*.

References

[1] P. Cerný, T. A. Henzinger, and A. Radhakrishna. Quantitative simulation games. In Z. Manna and D. Peled, editors, *Essays in Memory of Amir Pnueli*, volume 6200 of *LNCS*, pages 42–60. Springer, 2010.

[2] P. Cerný, T. A. Henzinger, and A. Radhakrishna. Simulation distances. In P. Gastin and F. Laroussinie, editors, *CONCUR*, volume 6269 of *LNCS*, pages 253–268. Springer, 2010.

[3] X. Chen and Y. Deng. Game characterizations of process equivalences. In G. Ramalingam, editor, *APLAS*, volume 5356 of *LNCS*, pages 107–121. Springer, 2008.

[4] L. de Alfaro, M. Faella, and M. Stoelinga. Linear and branching system metrics. *IEEE Trans. Software Eng.*, 35(2):258–273, 2009.

[5] D. de Frutos-Escrig, C. Gregorio-Rodríguez, and M. Palomino. On the unification of process semantics: equational semantics. *ENTCS*, 249:243–267, 2009.

[6] D. de Frutos-Escrig, C. Gregorio-Rodríguez, and M. Palomino. On the unification of process semantics: observational semantics. In *SOFSEM 2009: TPCS*, volume 5404/2009, pages 279–290. Springer Berlin / Heidelberg, 2009.

[7] U. Fahrenberg, A. Legay, and C. R. Thrane. The quantitative linear-time–branching-time spectrum. In S. Chakraborty and A. Kumar, editors, *FSTTCS*, volume 13 of *LIPIcs*, pages 103–114. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2011.

[8] A. Kiehn and S. Arun-Kumar. Amortised bisimulations. In F. Wang, editor, *FORTE*, volume 3731 of *LNCS*, pages 320–334. Springer, 2005.

[9] G. Lüttgen and W. Vogler. Safe reasoning with logic lts. In M. Nielsen, A. Kucera, P. B. Miltersen, C. Palamidessi, P. Tuma, and F. D. Valencia, editors, *SOFSEM*, volume 5404 of *LNCS*, pages 376–387. Springer, 2009.

[10] M. Nielsen and C. Clausen. Bisimulation, games, and logic. In J. Karhumäki, H. A. Maurer, and G. Rozenberg, editors, *Results and Trends in Theoretical Computer Science*, volume 812 of *LNCS*, pages 289–306. Springer, 1994.

[11] C. Stirling. Modal and temporal logics for processes. In F. Moller and G. M. Birtwistle, editors, *Banff Higher Order Workshop*, volume 1043 of *LNCS*, pages 149–237. Springer, 1995.

[12] C. Stirling. Bisimulation, modal logic and model checking games. *Logic Journal of the IGPL*, 7(1):103–124, 1999.

[13] C. R. Thrane, U. Fahrenberg, and K. G. Larsen. Quantitative analysis of weighted transition systems. *J. Log. Algebr. Program.*, 79(7):689–703, 2010.

[14] R. van Glabbeek. The linear time-branching time spectrum I: the semantics of concrete, sequential processes. In J. A. Bergstra, A. Ponse, and S. A. Smolka, editors, *Handbook of Process Algebra, chapter 1*, pages 3–99. Elsevier, 2001.

[15] G. Winskel. Synchronisation trees. In J. Díaz, editor, *ICALP*, volume 154 of *LNCS*, pages 695–711. Springer, 1983.