Preorder-Constrained Simulations for Program Refinement with Effects

Koko Muroya[1][0000−0003−0454−6900], Takahiro Sanada[1][0000−0003−3409−6963], and Natsuki Urabe[2][0000−0002−1554−6618]

[1] RIMS, Kyoto University, Japan
{kmuroya,tsanada}@kurims.kyoto-u.ac.jp
[2] National Institute of Informatics, Japan
urabenatsuki@nii.ac.jp

Abstract. We propose a notion of *preorder-constrained simulation*. It is parameterised by a preorder ("observation preorder") on traces, so that it can uniformly characterise quantitative notions of program refinement for different effects, such as exception, nondeterminism and I/O. Preorder-constrained simulation is additionally parameterised by a positive number ("look-ahead bound"), and forms a generative spectrum governed by the look-ahead bound. We analyse the complexity of determining preorder-constrained similarity, and show that preorder-constrained simulation can be enhanced by the so-called up-to technique.

1 Introduction

It is often important to have two programs behave the same in programming, for example, in code refactoring and compiler optimisation. *Observational equivalence* [26] is the standard notion that asserts the same observable behaviour between two programs.

Among known proof techniques for observational equivalence are *coinductive* techniques. These techniques reduce the problem of observational equivalence to *stepwise* comparison based on operational semantics. An important example is *applicative bisimilarity* [1]. It is tailored to the ordinary reduction semantics, syntactically constructing a binary relation that characterises observational equivalence. Together with its extension known as *environmental bisimilarity* [21], applicative bisimilarity is applicable to a wide range of side effects, such as general state [21], I/O [36] and continuation [38].

Other coinductive techniques that are for abstract machines have emerged recently. Examples are what we shall call *counting simulation* [27], and *improvement* [4] which can be seen as an instance of counting simulation. Notably, counting simulation can not only assert the same behaviour between programs, but also compare efficiency of two programs in terms of the number of execution steps. Despite this strength, it is only known to work on "deterministic" effects that yield at most one result per program.

$$\begin{array}{l}(1,\mathbf{Q})\text{-similarity}\\ \lesssim_{1,\mathbf{Q}}\end{array} \longrightarrow \begin{array}{l}(2,\mathbf{Q})\text{-similarity}\\ \lesssim_{2,\mathbf{Q}}\end{array} \longrightarrow \cdots \longrightarrow \begin{array}{l}\mathbf{Q}\text{-trace inclusion}\\ \sqsubseteq_{\mathbf{Q}}\end{array}$$

Fig. 1: A generative spectrum, parameterised by the observation preorder $\mathbf{Q}$

observation preorder $\mathbf{Q}$	$(1,\mathbf{Q})$-simulation	$\mathbf{Q}$-trace inclusion $\sqsubseteq_{\mathbf{Q}}$
$=$	standard simulation	finite trace inclusion
$=_{\mathsf{rem}_{\{\tau\}}}$	weak simulation	weak trace inclusion
$\dot{Q}$	(new instances)	refinement $\preceq^{Q}_{\mathsf{err}}$ for exception
$\dot{Q} \cap =_{\mathsf{rem}_{\{\tau\}\cup\overline{\Omega_{\mathsf{nd}}}}}$		refinement $\preceq^{Q}_{\mathsf{nd}}$ for nondeterminism
$\dot{Q} \cap =_{\mathsf{rem}_{\{\tau\}}}$		refinement $\preceq^{Q}_{\mathsf{io}}$ for I/O

Table 1: Instances of the two ends of the generative spectrum (see Sec. 4 for details)

It is desirable to obtain a coinductive proof technique for observational equivalence that is applicable to various effects and capable of quantitative comparison of efficiency. We have two kinds of techniques with different strengths: (1) applicative bisimilarity and environmental bisimilarity accommodating various effects, and (2) counting simulation and improvement which are capable of quantitative comparison. Our goal here is to enhance the techniques (2) so that they can accommodate a wider range of effects including nondeterminism.

One of the challenges of accommodating various effects is that a notion of observational equivalence, in particular that of observation, varies between effects. For example, nondeterministic choice is regarded as internal and unobservable. A program $\mathtt{or}(\underline{1},\underline{1})$ with a binary nondeterministic choice operator `or` would be identified with a program $\underline{1}$. The choice (with the same result) is *ignored*. On the other hand, the choice that is made according to a 1-bit input is regarded as external and observable. A program $\mathtt{in}(\underline{1},\underline{1})$, which results in $\underline{1}$ regardless of the value of the 1-bit input, would *not* be identified with the program $\underline{1}$. The received input value (0 or 1) and the induced choice (between $\underline{1}$ and $\underline{1}$) is *observed*. We need to be able to both ignore and observe effectful choices.

We propose a notion of *preorder-constrained simulation* that is applicable to effects such as nondeterminism and I/O, in addition to the "deterministic" effects such as exception. It is notably parameterised by an *observation preorder*, a preorder on traces (or words). By altering the observation preorder, we can characterise quantitative notions of observational refinement, which is the asymmetric version of observational equivalence, for both internal and ignored effects, and external and observed effects.

Preorder-constrained simulations are additionally parameterised by a positive number dubbed *look-ahead bound*. It determines the degree of awareness of branching. By altering the look-ahead bound, one can obtain a spectrum illustrated in Fig. 1. The "limit" of the spectrum is a novel generalisation of trace inclusion (i.e. $\mathbf{Q}$-trace inclusion $\sqsubseteq_{\mathbf{Q}}$) that is also parameterised by the observation preorder. The spectrum is generative in the sense that it yields various

concrete spectra by instantiating the observation preorder $\mathbf{Q}$. Some instances are shown in Tab. 1, whose details will be evident in Sec. 4.

We analyse the complexity of determining preorder-constrained similarity $\lesssim_{M,\mathbf{Q}}$ in the finite setting, and show that the complexity is polynomial time. The complexity analysis is via a game-theoretic characterisation of preorder-constrained simulation that is similar to *buffered simulation game* [17,16]. We also show that preorder-constrained simulation can be enhanced by the so-called *up-to* technique [32]. We discuss and identify sufficient conditions that make the up-to technique work, in terms of observation preorders.

Our contributions can be summarised as follows.

- (Sec. 4) The notion of preorder-constrained simulation, which is a new variant of simulation for characterising observational refinement between programs (Cor. 1). Preorder-constrained simulations enjoy soundness (Thm. 1) with respect to a novel generalisation of trace inclusion, and also monotonicity (Cor. 2) that yields a generative spectrum (Fig. 1).
- (Sec. 5) A game-theoretic characterisation of preorder-constrained simulation, with complexity analysis. The games also form a generative spectrum (Fig. 12).
- (Sec. 6) Integration of the so-called *up-to* technique [32], in terms of observation preorders.

Proofs of statements marked by (†) are formalised in Agda, and available at https://github.com/urabenatsuki/preorder-constr-sim-agda. Other proofs can be found in App. C.

2 Preliminaries

Let $\mathbb{N}$ be the set of natural numbers, and $\mathbb{N}_+$ be the set of positive numbers. For a set X, X^* denotes the set $\{x_1 \dots x_n \mid n \in \mathbb{N}; x_1, \dots, x_n \in X\}$ of finite words over X. We let ε denote the empty word, X^+ denote $X^* \setminus \{\varepsilon\}$, and $|w|$ denote the length of a finite word $w \in X^*$.

Given $X' \subseteq X$, the *filtered equality* $=_{\mathsf{rem}_{X'}}$ on the set X^* is defined by $w =_{\mathsf{rem}_{X'}} w'$ if w and w' are the same except for symbols in X'. For example, $a\tau b\sigma\sigma c =_{\mathsf{rem}_{\{\tau,\sigma\}}} \sigma ab\tau c\tau =_{\mathsf{rem}_{\{\tau,\sigma\}}} abc$.

A preorder $Q \subseteq \mathbb{N} \times \mathbb{N}$ is said to be *s-closed* if it is closed under summation, i.e. $kQl \wedge k'Ql' \implies (k+k')Q(l+l')$. For a set Σ, a preorder $\mathbf{Q} \subseteq \Sigma^* \times \Sigma^*$ is said to be *c-closed* if it is closed under concatenation, i.e. $w_1 \mathbf{Q} w_2 \wedge w_1' \mathbf{Q} w_2' \implies (w_1 w_1') \mathbf{Q} (w_2' w_2')$.

2.1 Nondeterministic Automata

A *nondeterministic automaton* (NA) is a quadruple $\mathcal{A} = (X, \Sigma, \rightsquigarrow, F)$ consisting of a set X called a *state space*, a set Σ called an *alphabet*, a *transition relation* $\rightsquigarrow \subseteq X \times \Sigma \times X$ and a set $F \subseteq X$ of *accepting states*. We write $x \overset{a}{\rightsquigarrow} x'$ when $(x, a, x') \in \rightsquigarrow$, and $x \overset{w}{\rightsquigarrow} x'$ for $w = a_1 \cdots a_n$ when there exist $x_0 \dots x_n \in X^+$

$$t ::= x \mid t\ t \mid \lambda x.t \mid t[x \leftarrow t] \mid \underline{n} \mid t + t \mid t \times t \mid f(t, \dots, t) \qquad \text{(Terms } \mathbf{T}_\Omega\text{)}$$
$$v ::= \lambda x.t \mid \underline{n} \qquad \text{(Values)}$$
$$L ::= \langle\,\rangle \mid L[x \leftarrow t] \qquad \text{(Answer contexts)}$$
$$E ::= \langle\,\rangle \mid E\ t \mid L\langle v\rangle\ E \mid E[x \leftarrow t] \mid E + t \mid L\langle v\rangle + E \mid E \times t \mid L\langle v\rangle \times E \qquad \text{(Evaluation contexts)}$$

$$\frac{}{L\langle\lambda x.t\rangle\ L'\langle v\rangle \xrightarrow{\tau} L\langle t[x \leftarrow L\langle v\rangle]\rangle} \qquad \frac{}{E\langle x\rangle[x \leftarrow L\langle v\rangle] \xrightarrow{\tau} L\langle E\langle v\rangle[x \leftarrow v]\rangle}$$

$$\frac{\bullet \in \{+, \times\}}{L\langle\underline{n}\rangle \bullet L'\langle\underline{m}\rangle \xrightarrow{\tau} \underline{n \bullet m}} \qquad \frac{}{f(t_0, \dots, t_{\mathrm{ar}(f)-1}) \xrightarrow{f_i} t_i} \qquad \frac{t \xrightarrow{\ell} u \quad \ell \in \{\tau\} \cup \overline{\Omega}}{E\langle t\rangle \xrightarrow{\ell} E\langle u\rangle} \qquad \frac{}{L\langle\underline{n}\rangle \xrightarrow{n} \checkmark}$$

Fig. 2: Reduction semantics as an NA $\mathcal{A}_\Omega = (\mathbf{T}_\Omega \cup \{\checkmark\},\ \{\tau\} \cup \mathbb{N} \cup \overline{\Omega},\ \to,\ \{\checkmark\})$

such that $x_0 = x$, $x_n = x'$ and $x_{i-1} \overset{a_i}{\rightsquigarrow} x_i$ for each $i \in \{1, \dots, n\}$. In particular, $x \overset{\varepsilon}{\rightsquigarrow\!\!\!\!\rightarrow} x$. We write $x \not\rightsquigarrow$ and say x is *stuck*, if there exist no $x' \in X$ and $a \in \Sigma$ such that $x \overset{a}{\rightsquigarrow} x'$. An NA is said to be *branching-free* if its transition relation $\rightsquigarrow \subseteq X \times \Sigma \times X$ satisfies the following: for any $x \in X$, if there exist two pairs $(a_1, x_1), (a_2, x_2) \in \Sigma \times X$ such that $x \overset{a_1}{\rightsquigarrow} x_1$ and $x \overset{a_2}{\rightsquigarrow} x_2$, then $(a_1, x_1) = (a_2, x_2)$.

The *(finite) language* of an NA $\mathcal{A}$ is a function $L^*_{\mathcal{A}} : X \to \Sigma^*$ defined by $L^*_{\mathcal{A}}(x) := \{w \in \Sigma^* \mid \exists x' \in F.\ x \overset{w}{\rightsquigarrow\!\!\!\!\rightarrow} x'\}$. In particular, $\varepsilon \in L^*_{\mathcal{A}}(x)$ when $x \in F$. We omit the subscript and write $L^*(x)$ for $L^*_{\mathcal{A}}(x)$ when no confusion is likely.

2.2 A Linear Substitution Calculus with Algebraic Effects

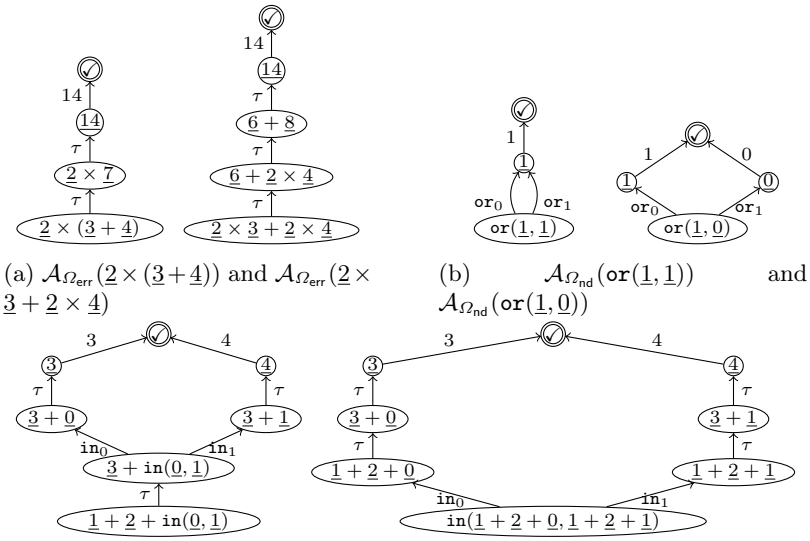
We recall *algebraic effects* [29] and present their reduction semantics as an NA. Algebraic effects are specified by a *signature* Ω. It is a set of *algebraic operations* f, each of which comes with an *arity* $\mathrm{ar}(f) \in \mathbb{N}$. We write $f\colon n$ when $\mathrm{ar}(f) = n$.

Example 1 (algebraic operations).

1. $\Omega_{\mathsf{err}} = \{\mathtt{err}\colon 0\}$ for raising an error.
2. $\Omega_{\mathsf{nd}} = \{\mathtt{or}\colon 2\}$ for nondeterministic choice between two operands.
3. $\Omega_{\mathsf{io}} = \{\mathtt{in}\colon 2, \mathtt{out}^0\colon 1, \mathtt{out}^1\colon 1\}$ for I/O with a single bit. We focus on a single-bit I/O for simplicity. Evaluation of a term $\mathtt{in}(t_0, t_1)$ proceeds with t_i if the input value is i, for each $i \in \{0, 1\}$. Evaluation of a term $\mathtt{out}^i(t)$ outputs the value $i \in \{0, 1\}$ and proceeds with t.

We use the (left-to-right) call-by-value *linear substitution calculus* (LSC) [5] equipped with algebraic effects Ω and arithmetic. The LSC has been used as a cost model of various abstract machines for the λ-calculus [3]. The LSC exploits explicit substitutions $[x \leftarrow t]$ to disclose cost of the traditional β-reduction.

We present reduction semantics of the LSC, with a signature Ω, as an NA defined by Fig. 2. Transitions are labelled in a similar way as the original reduction semantics for algebraic effects [29]; labels consist of τ representing *silent*



(a) $\mathcal{A}_{\Omega_{\mathsf{err}}}(\underline{2} \times (\underline{3}+\underline{4}))$ and $\mathcal{A}_{\Omega_{\mathsf{err}}}(\underline{2} \times \underline{3} + \underline{2} \times \underline{4})$

(b) $\mathcal{A}_{\Omega_{\mathsf{nd}}}(\mathtt{or}(\underline{1},\underline{1}))$ and $\mathcal{A}_{\Omega_{\mathsf{nd}}}(\mathtt{or}(\underline{1},\underline{0}))$

(c) $\mathcal{A}_{\Omega_{\mathsf{io}}}(\underline{1} + \underline{2} + \mathtt{in}(\underline{0},\underline{1}))$ and $\mathcal{A}_{\Omega_{\mathsf{io}}}(\mathtt{in}(\underline{1} + \underline{2} + \underline{0}, \underline{1} + \underline{2} + \underline{1}))$

Fig. 3: Example pairs of NAs

transitions, $\mathbb{N}$ representing result values, and a set $\overline{\Omega} = \{f_i \mid f \in \Omega, i \in \{0,\dots,\mathrm{ar}(f)-1\}\}$ labelling *effect* transitions. We add a *return* transition $\underline{n} \overset{n}{\rightsquigarrow} \checkmark$ from each ground value to a sole accepting state $\checkmark$. This extra transition enables us to observe an evaluation result as a transition label. For $t \in \mathbf{T}_\Omega$, an NA $\mathcal{A}_\Omega(t)$ is the restriction of $\mathcal{A}_\Omega$ to the states that are reachable from t.

A successful evaluation of a term t is given by a sequence $t \overset{w}{\twoheadrightarrow} L\langle\underline{n}\rangle \overset{n}{\rightarrow} \checkmark$ of silent or effect transitions followed by a return transition. We refer to the word wn as a *trace* of t, and to n as a *result* of the execution. By removing the silent transitions $\overset{\tau}{\rightarrow}$, we obtain a word w' from w. We refer to w' as an *effect trace* of t. Note that the NA $\mathcal{A}_{\Omega_{\mathsf{err}}}$ is in fact branching-free.

Example 2 (pairs of NAs).

1. In Fig. 3a, $\underline{2} \times (\underline{3}+\underline{4})$ has a trace $\tau\tau 14$, and $\underline{2} \times \underline{3} + \underline{2} \times \underline{4}$ has a trace $\tau\tau\tau 14$.
2. In Fig. 3b, terms $\mathtt{or}(\underline{1},\underline{1})$ and $\mathtt{or}(\underline{1},\underline{0})$ have the same effect traces $\{\mathtt{or}_0, \mathtt{or}_1\}$.
3. In Fig. 3c, terms $\underline{1} + \underline{2} + \mathtt{in}(\underline{0},\underline{1})$ and $\mathtt{in}(\underline{1} + \underline{2} + \underline{0}, \underline{1} + \underline{2} + \underline{1})$ have the same results $\{\underline{3},\underline{4}\}$ and the same effect traces $\{\mathtt{in}_0, \mathtt{in}_1\}$, but not the same traces.

We can now formalise program refinement, using NAs $\mathcal{A}_\Omega$ for the signatures in Ex. 1. The notion is quantitative, in the sense that it is parameterised by a *length preorder* $Q \subseteq \mathbb{N} \times \mathbb{N}$.

Definition 1 ((quantitative) refinement). *Let Q be a preorder on $\mathbb{N}$ (dubbed* length preorder*).*

1. *For* Ω_{err}, $t \preceq^{Q}_{\mathsf{err}} u$ *is defined by* $\forall w.(t \overset{w}{\twoheadrightarrow} \checkmark \implies \exists w'.u \overset{w'}{\twoheadrightarrow} \checkmark \wedge |w|Q|w'|)$.
2. *For* Ω_{nd}, $t \preceq^{Q}_{\mathsf{nd}} u$ *is defined by* $\forall w.(t \overset{w}{\twoheadrightarrow} \checkmark \implies \exists w'.u \overset{w'}{\twoheadrightarrow} \checkmark \wedge |w|Q|w'| \wedge w =_{\mathsf{rem}_{\{\tau\}\cup\overline{\Omega_{\mathsf{nd}}}}} w')$.
3. *For* Ω_{io}, $t \preceq^{Q}_{\mathsf{io}} u$ *is defined by* $\forall w.(t \overset{w}{\twoheadrightarrow} \checkmark \implies \exists w'.u \overset{w'}{\twoheadrightarrow} \checkmark \wedge |w|Q|w'| \wedge w =_{\mathsf{rem}_{\{\tau\}}} w')$.

The refinement $t \preceq^{Q}_{\mathsf{err}} u$ focuses on successful termination, and additionally, compares the numbers of steps using Q. It asserts that when evaluation of t terminates in m steps, evaluation of u also terminates in n steps, and moreover, mQn. An example of Q is the total relation $\mathbb{N} \times \mathbb{N}$. This yields the asymmetric version of the basic notion of *observational equivalence*. Another example of Q is the greater-than-equal relation $\geq$, with which refinement $t \preceq^{\geq} u$ can assert that u terminates in a fewer steps.

Def. 1(2) and Def. 1(3) use the filtered equality differently to deal with different observations made for nondeterministic choice and I/O. The refinement $t \preceq_{\mathsf{nd}} u$ for nondeterministic choice asserts that when evaluation of t terminates, evaluation of u also terminates with the same result. The filtered equality $w =_{\mathsf{rem}_{\{\tau\}\cup\overline{\Omega_{\mathsf{nd}}}}} w'$ in Def. 1(2) *ignores* effect traces, and only ensures the coincidence of results. We note that Def. 1(2) corresponds to program refinement for angelic nondeterminism [35]. In contrast, the refinement $t \preceq_{\mathsf{io}} u$ for I/O asserts that when evaluation of t terminates, evaluation of u also terminates with the same effect trace and result. The filtered equality $w =_{\mathsf{rem}_{\{\tau\}}} w'$ in Def. 1(3) *observes* effect traces.

The three pairs of NAs in Fig. 3 all exhibit refinement. We have $\underline{2} \times (\underline{3} + \underline{4}) \preceq^{\leq}_{\mathsf{err}} \underline{2} \times \underline{3} + \underline{2} \times \underline{4}$, $\mathtt{or}(\underline{1}, \underline{1}) \preceq^{=}_{\mathsf{nd}} \mathtt{or}(\underline{1}, \underline{0})$, and $\underline{1} + \underline{2} + \mathtt{in}(\underline{0}, \underline{1}) \preceq^{=}_{\mathsf{io}} \mathtt{in}(\underline{1} + \underline{2} + \underline{0}, \underline{1} + \underline{2} + \underline{1})$. We do not always have the opposite: i.e. $\mathtt{or}(\underline{1}, \underline{0}) \not\preceq^{=}_{\mathsf{nd}} \mathtt{or}(\underline{1}, \underline{1})$.

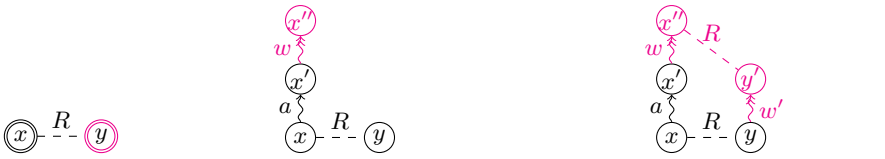
3 Counting Simulation and its Deficiencies

We recall the simulation notion that was introduced for an abstract machine for programs with at most one result [27]. We call it *counting simulation*, albeit with no connection to counter automata. It is parameterised by a preorder Q on natural numbers, dubbed *length preorder*, to *count* and compare the number of steps. The original presentation [27, Def. 4.4.1] of the counting simulation is for an unlabelled transition system, and it is equipped with an up-to technique. We here present its naive extension to NAs, but without any up-to technique.

In this section, we let $\mathcal{A}_i = (X_i, \Sigma, \rightsquigarrow_i, F_i)$ $(i \in \{1, 2\})$ be two NAs with the same alphabet, and $Q \subseteq \mathbb{N} \times \mathbb{N}$ be an s-closed preorder.

Definition 2 (counting simulations). *A binary relation* $R \subseteq X_1 \times X_2$ *is a* Q-counting simulation *from* $\mathcal{A}_1$ *to* $\mathcal{A}_2$ *if, for any* $(x, y) \in R$*, the following* ***C-Final*** *and* ***C-Step*** *hold.*

C-Final *If* $x \in F_1$*, then* $y \in F_2$.



(a) ***C-Final*** (b) ***C-Step*** (1) where $x'' \not\rightarrow$ (c) ***C-Step*** (2) where $|aw|Q|w'|$

Fig. 4: Conditions of Def. 2. Black parts are universally quantified, and magenta parts are existentially quantified.

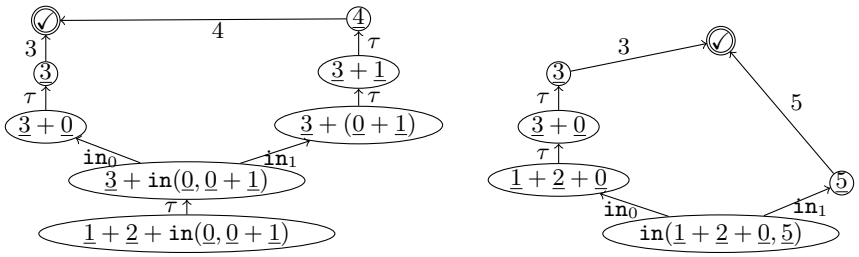


Fig. 5: Unsoundness of counting simulation

C-Step *For each $x' \in X_1$ and $a \in \Sigma$ such that $x \overset{a}{\rightsquigarrow}_1 x'$, either of the following holds.*

1. *There exist $x'' \in X_1$ and $w \in \Sigma^*$ such that $x' \overset{w}{\rightsquigarrow}_1 x'' \not\rightarrow_1$ and $x'' \notin F_1$.*
2. *There exist $x'' \in X_1$, $y' \in X_2$ and $w, w' \in \Sigma^*$, such that $x' \overset{w}{\rightsquigarrow}_1 x''$, $y \overset{w'}{\rightsquigarrow}_2 y'$, $|aw|Q|w'|$, and $x''Ry'$.*

Fig. 4 illustrates the conditions of Def. 2. The condition ***C-Final*** is standard. The condition ***C-Step*** (1) deals with a stuck and non-accepting state $x'' \in X_1$ (i.e. $x'' \not\rightarrow$ and $x'' \notin F_1$). A term $\mathtt{err}() \in \mathbf{T}_{\Omega_{\mathsf{err}}}$ is an example of such a state in $\mathcal{A}_{\Omega_{\mathsf{err}}}$. Lastly, ***C-Step*** (2) is the key condition of Def. 2. It asserts that any transition $x \overset{a}{\rightsquigarrow} x'$ followed by *some* transitions $x' \overset{w}{\rightsquigarrow} x''$ in $\mathcal{A}_1$ can be simulated by some transitions $y \overset{w'}{\rightsquigarrow} y'$ in $\mathcal{A}_2$.

Thanks to ***C-Step***, which compares not just single steps but numbers of steps, counting simulation can witness quantitative refinement for exception.

Proposition 1 (correctness wrt. refinement). *If R is a Q-counting simulation from $\mathcal{A}_{\Omega_{\mathsf{err}}}$ to $\mathcal{A}_{\Omega_{\mathsf{err}}}$, then $tRu \implies t \preceq^{Q}_{\mathsf{err}} u$ holds for any $t, u \in \mathbf{T}_{\Omega_{\mathsf{err}}}$.* □

Example 3. For the pair of branching-free NAs in Fig. 3a, a relation $R_{\mathsf{distr}} = \{(\underline{2} \times (\underline{3} + \underline{4}), \underline{2} \times \underline{3} + \underline{2} \times \underline{4}), (✓, ✓)\}$ is a $\leq$-counting simulation. The length preorder $\leq$ asserts that $\underline{2} \times (\underline{3} + \underline{4})$ has better efficiency. The relation R_{distr} represents the distributive law, without relating any intermediate states.

Nevertheless, counting simulation cannot witness refinement for effects such as nondeterminism and I/O. This is due to two challenges. The first challenge

is varying observation. While we ignore effect traces for nondeterminism, we observe effect traces for I/O. However, counting simulation can neither ignore nor observe effect traces correctly. It simply compares the lengths of traces using the length preorder Q. The second challenge is branching. It is branching effects that counting simulation becomes unsound for.

Example 4 (unsoundness for I/O). For the pair of NAs in Fig. 5, refinement does not hold, i.e. $\underline{1}+\underline{2}+\mathtt{in}(\underline{0},\underline{0}+\underline{1}) \not\preceq_{\mathsf{io}} \mathtt{in}(\underline{1}+\underline{2}+\underline{0},\underline{5})$, because right branches have traces $\tau\mathtt{in}_1\tau\tau 4 \not=_{\mathsf{rem}_\tau} \mathtt{in}_1 5$. However, the relation $\{(\underline{1}+\underline{2}+\mathtt{in}(\underline{0},\underline{0}+\underline{1}), \mathtt{in}(\underline{1}+\underline{2}+\underline{0},\underline{5})), (\checkmark,\checkmark)\}$ is an $=$-counting simulation. It only asserts that left branches have "identical" traces (i.e. $\tau\mathtt{in}_0\tau 3$ and $\mathtt{in}_0\tau\tau 3$), and it does not inspect the right branches with distinct traces.

This unsoundness is because counting simulation does not necessarily inspect all possibilities of branching. Technically, this is due to the existential quantification on $x' \overset{w}{\rightsquigarrow\!\!\!\!\rightarrow} x''$ in ***C-Step*** (2).

4 Preorder-Constrained Simulation

4.1 Definition

We present our main contribution, the notion of *preorder-constrained simulation*. It generalises counting simulation from branching-free NAs to general NAs, and hence characterises a notion of observational refinement for a wider class of effects (Cor. 1 below). The generalisation is technically two-fold, dealing with the two challenges we discussed in Sec. 3.

Firstly, preorder-constrained simulation is parameterised by a c-closed preorder $\mathbf{Q}$ on traces Σ^*, dubbed *observation preorder*, instead of the length preorder $Q \subseteq \mathbb{N}\times\mathbb{N}$ that parameterises counting simulation. Using preorders like the filtered equality $=_{\mathsf{rem}_X}$, preorder-constrained simulation can flexibly *observe* and compare traces, adapting to varying observations.

Example 5 (observation preorders). Note that the following preorders are all c-closed.

1. The equality $=$ on words.
2. Each preorder $Q \subseteq \mathbb{N}\times\mathbb{N}$ that is s-closed induces a preorder $\dot{Q}$ on words such that $w\dot{Q}w' \iff |w|Q|w'|$, which is c-closed.
3. Each subset $\Sigma' \subseteq \Sigma$ induces the preorder $=_{\mathsf{rem}_{\Sigma'}} \subseteq \Sigma^*\times\Sigma^*$ of filtered equality.
4. The *substring* preorder $\subseteq_{\mathsf{sub}}$, where $w \subseteq_{\mathsf{sub}} w'$ means that w is a substring of w'.
5. Assume Σ is the powerset 2^{AP} of some set AP. Let $\subseteq^*$ be a preorder where $a_1\dots a_k \subseteq^* a'_1\dots a'_{k'}$ means $k=k'$ and $\forall i.\ a_i \subseteq a'_i$.
6. Let $\mathbb{R}$ be the set of real numbers, and $\bullet$ be a binary operation on $\mathbb{R}$ such as summation $+$ and multiplication $\times$. When $\Sigma=\mathbb{R}$, let $\leq_\bullet$ be a preorder where $a_1\dots a_n \leq_\bullet a'_1\dots a'_{n'}$ means $a_1\bullet\dots\bullet a_n \leq a'_1\bullet\dots\bullet a'_{n'}$. Its inverse $\geq_\bullet$ is also a preorder.

(a) ***Final***^{M} where $|w| < M \ \wedge \ w\mathbf{Q}w'$ (b) ***Step***^{M} where $a_1 \cdots a_k \mathbf{Q} w'$

Fig. 6: Conditions of Def. 3. Black parts are universally quantified, and magenta parts are existentially quantified.

Secondly, preorder-constrained simulation limits the existential quantification, namely that on $x' \overset{w}{\rightsquigarrow\!\!\!\!\rightarrow} x''$ in ***C-Step*** (2), to overcome the unsoundness due to incomplete inspection of branching. The idea is to put the existential quantification inside universal quantification, so that every branch gets inspected by the resultant simulation notion.

Additionally, preorder-constrained simulation is parameterised by a positive number M dubbed *look-ahead bound*. It determines the degree of awareness of branching; we leave the study of the look-ahead bound to Sec. 4.3. We let $\mathcal{A}_i = (X_i, \Sigma, \rightsquigarrow_i, F_i)$ $(i \in \{1,2\})$ be two NAs with the same alphabet, $Q \subseteq \mathbb{N} \times \mathbb{N}$ be an s-closed preorder, and $\mathbf{Q} \subseteq \Sigma^* \times \Sigma^*$ be a c-closed preorder.

Definition 3 ($(M, \mathbf{Q})$-simulations). *For each $M \in \mathbb{N}_+$, a binary relation $R \subseteq X_1 \times X_2$ is an M-*bounded **Q**-constrained simulation *($(M, \mathbf{Q})$-simulation in short) from $\mathcal{A}_1$ to $\mathcal{A}_2$ if, for any $(x, y) \in R$, the following* ***Final***^{M} *and* ***Step***^{M} *hold.*

Final^M *For each $w = a_1 \dots a_n \in \Sigma^*$ and $x_1 \dots x_n \in X_1^*$ such that $n < M$, $x \overset{a_1}{\rightsquigarrow}_1 x_1 \cdots \overset{a_n}{\rightsquigarrow}_1 x_n$ and $x_n \in F_1$, there exist $w' \in \Sigma^*$ and $y' \in X_2$ such that $w\mathbf{Q}w'$, $y \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y'$ and $y' \in F_2$.*

Step^M *For each $a_1 \dots a_M \in \Sigma^M$ and $x_1 \dots x_M \in X_1^M$ such that $x \overset{a_1}{\rightsquigarrow}_1 x_1 \cdots \overset{a_M}{\rightsquigarrow}_1 x_M$, there exist $k \in \{1, \dots, M\}$, $w' \in \Sigma^*$ and $y' \in X_2$ such that $a_1 \cdots a_k \mathbf{Q} w'$, $y \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y'$ and $x_k R y'$.*

When an $(M, \mathbf{Q})$-simulation relates x and y, we say x is $(M, \mathbf{Q})$-*similar* to y and write $x \lesssim_{M,\mathbf{Q}} y$. When R is an $(M, \mathbf{Q})$-simulation from $\mathcal{A}$ to $\mathcal{A}$, we say it is *on* $\mathcal{A}$.

Fig. 6 illustrates the conditions of Def. 3. The difference between Fig. 4c and Fig. 6b is crucial to overcome the incomplete inspection of branching that counting simulation suffers from. In Fig. 6b, existential quantification is limited to x_k, which is an intermediate state of the sequence $x \rightsquigarrow\!\!\!\!\rightarrow x_M$ that is universally quantified.

4.2 Soundness

Thanks to the observation preorder **Q** and the limited existential quantification, preorder-constrained simulation can characterise notions of quantitative refine-

ment $\preceq^Q$ for all the signatures Ω in Ex. 1. Namely, $(M, \mathbf{Q})$-simulations provide a sufficient condition for refinement.

Corollary 1 (correctness of $(M, \mathbf{Q})$-simulations wrt. refinement).

1. *For any* $M \in \mathbb{N}_+$ *and* $t, u \in \mathbf{T}_{\Omega_{\mathsf{err}}}$, $t \lesssim_{M,\dot{Q}} u \implies t \preceq^Q_{\mathsf{err}} u$.
2. *For any* $M \in \mathbb{N}_+$ *and* $t, u \in \mathbf{T}_{\Omega_{\mathsf{nd}}}$, $t \lesssim_{M,\dot{Q} \cap =_{\mathsf{rem}_{\{\tau\} \cup \overline{\Omega_{\mathsf{nd}}}}}} u \implies t \preceq^Q_{\mathsf{nd}} u$.
3. *For any* $M \in \mathbb{N}_+$ *and* $t, u \in \mathbf{T}_{\Omega_{\mathsf{io}}}$, $t \lesssim_{M,\dot{Q} \cap =_{\mathsf{rem}_{\{\tau\}}}} u \implies t \preceq^Q_{\mathsf{io}} u$. □

Example 6 (Fig. 3 revisited).

1. For the NAs in Fig. 3a, $\{(\underline{2} \times (\underline{3} + \underline{4}), \underline{2} \times \underline{3} + \underline{2} \times \underline{4}), (\underline{14}, \underline{14})\}$ is a $(2, \dot{\leq})$-simulation. This simulation represents the distributive law.
2. For Fig. 3b, $\{(\mathtt{or}(\underline{1}, \underline{1}), \mathtt{or}(\underline{1}, \underline{0})), (\underline{1}, \underline{1}), (\checkmark, \checkmark)\}$ is a $(1, \dot{=} \cup =_{\mathsf{rem}_{\{\tau\} \cup \Omega_{\mathsf{nd}}}})$-simulation.
3. For Fig. 3c, let $t_1 \equiv \underline{1} + \underline{2} + \mathtt{in}(\underline{0}, \underline{1})$ and $t_2 \equiv \mathtt{in}(\underline{1} + \underline{2} + \underline{0}, \underline{1} + \underline{2} + \underline{1})$. The filtered equality $=_{\mathsf{rem}_{\{\tau\}}}$ distinguishes traces of length 1 from t_1, t_2, namely: $\tau \neq_{\mathsf{rem}_{\{\tau\}}} \mathtt{in}_i$ $(i \in \{0, 1\})$. This leads to non-existence of any $(1, \dot{=} \cap =_{\mathsf{rem}_{\{\tau\}}})$-simulation that includes the pair (t_1, t_2). In contrast, the filtered equality can identify traces of length 2 from t_1, t_2, namely: $\tau \mathtt{in}_i =_{\mathsf{rem}_{\{\tau\}}} \mathtt{in}_i \tau$ $(i \in \{0, 1\})$. This leads to existence of a $(2, \dot{=} \cap =_{\mathsf{rem}_{\{\tau\}}})$-simulation. It can be given by $\{(t_1, t_2),\ (\underline{3} + \underline{0}, \underline{3} + \underline{0}),\ (\underline{3} + \underline{1}, \underline{3} + \underline{1}),\ (\checkmark, \checkmark)\}$.

Cor. 1 is a consequence of a soundness property of preorder-constrained simulations. Namely, $(M, \mathbf{Q})$-simulations are sound with respect to a novel generalisation (Def. 4 below) of trace inclusion that is also parameterised by the observation preorder $\mathbf{Q}$.

Definition 4 (Q-trace inclusion). *We write* $x \sqsubseteq_{\mathbf{Q}} y$ *and say* $\mathbf{Q}$-trace inclusion *holds between* x *and* y, *if the following holds:* $\forall w \in L^*_{A_1}(x).\ \exists w' \in L^*_{A_2}(y).\ w \mathbf{Q} w'$.

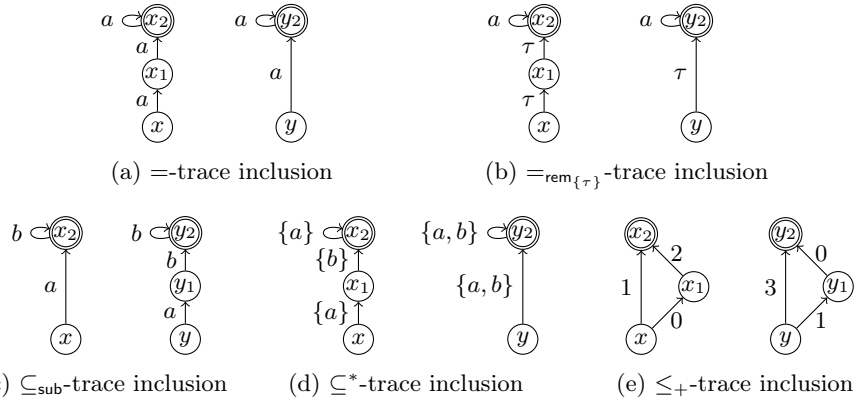
Theorem 1 ((†) soundness). *Let* $M \in \mathbb{N}_+$. *For any* $(x, y) \in X_1 \times X_2$, *it holds that* $x \lesssim_{M,\mathbf{Q}} y \implies x \sqsubseteq_{\mathbf{Q}} y$.

The three refinement relations for specific NAs $\mathcal{A}_{\Omega_{\mathsf{err}}}$, $\mathcal{A}_{\Omega_{\mathsf{nd}}}$ and $\mathcal{A}_{\Omega_{\mathsf{io}}}$, defined in Sec. 2.2, are instances of the generalised trace inclusion. Cor. 1 follows from Thm. 1 above and Prop. 2 below.

Proposition 2 (refinement as trace inclusion).

1. *For* $\mathcal{A}_{\Omega_{\mathsf{err}}}$, *for any* $t, u \in \mathbf{T}_{\Omega_{\mathsf{err}}}$, $t \preceq^Q_{\mathsf{err}} u \iff t \sqsubseteq_{\dot{Q}} u$.
2. *For* $\mathcal{A}_{\Omega_{\mathsf{nd}}}$, *for any* $t, u \in \mathbf{T}_{\Omega_{\mathsf{nd}}}$, $t \preceq^Q_{\mathsf{nd}} u \iff t \sqsubseteq_{\dot{Q} \cap =_{\mathsf{rem}_{\{\tau\} \cup \overline{\Omega_{\mathsf{nd}}}}}} u$.
3. *For* $\mathcal{A}_{\Omega_{\mathsf{io}}}$, *for any* $t, u \in \mathbf{T}_{\Omega_{\mathsf{io}}}$, $t \preceq^Q_{\mathsf{io}} u \iff t \sqsubseteq_{\dot{Q} \cap =_{\mathsf{rem}_{\{\tau\}}}} u$. □

The most basic instance of the $\mathbf{Q}$-trace inclusion is when $\mathbf{Q}$ is the equality; it coincides with the standard notion of (finite) trace inclusion. Another basic instance is when $\mathbf{Q}$ is the filtered equality $=_{\mathsf{rem}_{\{\tau\}}}$; it corresponds to the well-known notion of weak trace inclusion.



(a) =-trace inclusion (b) $=_{\mathsf{rem}_{\{\tau\}}}$-trace inclusion

(c) $\subseteq_{\mathsf{sub}}$-trace inclusion (d) $\subseteq^*$-trace inclusion (e) $\leq_+$-trace inclusion

Fig. 7: Pairs of NAs that exhibit $\mathbf{Q}$-trace inclusion

Example 7 ($\mathbf{Q}$*-trace inclusions for various preorders).* Fig. 7 shows examples of $\mathbf{Q}$-trace inclusions for some of the preorders from Ex. 5.

1. In Fig. 7a, we have $L^*(x) = aaa^* = \{a^{n+2} | n \in \mathbb{N}\}$ and $L^*(y) = aa^* = \{a^{n+1} | n \in \mathbb{N}\}$, and therefore, $L^*(x) \subseteq L^*(y)$. We have $x \sqsubseteq_{=} y$ for the equality $=$.
2. In Fig. 7b, we have $L^*(x) = \tau\tau a^*$ and $L^*(y) = \tau a^*$. These two sets coincide when we ignore τ. We therefore have $x \sqsubseteq_{=_{\mathsf{rem}_{\{\tau\}}}} y$ for the filtered equality $=_{\mathsf{rem}_{\{\tau\}}}$.
3. In Fig. 7c, we have $L^*(x) = ab^*$ and $L^*(y) = abb^*$. We hence have $x \sqsubseteq_{\subseteq_{\mathsf{sub}}} y$ where $\subseteq_{\mathsf{sub}}$ is the substring preorder from Ex. 5(4).
4. In Fig. 7d, we have $L^*(x) = \{a\}\{b\}\{a\}^*$ and $L^*(y) = \{a, b\}\{a, b\}^*$. We hence have $x \sqsubseteq_{\subseteq^*} y$ where $\subseteq^*$ is from Ex. 5(5).
5. In Fig. 7e, we have $L^*(x) = \{1, 02\}$ and $L^*(y) = \{3, 10\}$. It holds that $1 \leq 3$ and $0 + 2 \leq 3$. We hence have $x \sqsubseteq_{\leq_+} y$ where $\leq_+$ is from Ex. 5(6).

4.3 Basic Properties

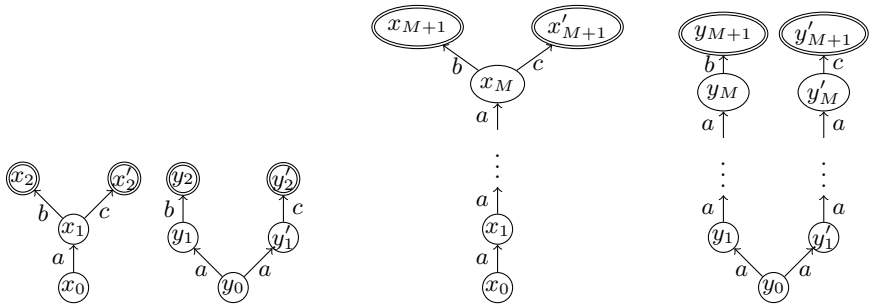
Here we investigate the look-ahead bound M of preorder-constrained simulation. As observed in Ex. 6 (3), the look-ahead bound determines the degree of awareness of branching; as M increases, $(M, \mathbf{Q})$-simulation can inspect and identify further branches. Technically, preorder-constrained simulation has a monotonicity property with respect to M.

Lemma 1 ((†) monotonicity of *Final* and *Step*). *Let $M, N \in \mathbb{N}_+$ such that $M \leq N$.*

1. $\mathit{Final}^N \implies \mathit{Final}^M$
2. $\mathit{Step}^M \implies \mathit{Step}^N$
3. $\mathit{Step}^M \wedge \mathit{Final}^M \implies \mathit{Final}^N$

□

Corollary 2 (monotonicity). *Let $M, N \in \mathbb{N}_+$ such that $M \leq N$. Each $(M, \mathbf{Q})$-simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$ is also an $(N, \mathbf{Q})$-simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$.*



(a) For $M = 1$ (b) For arbitrary M

Fig. 8: Pairs of NAs that distinguish $(M, =)$-similarity and $(M+1, =)$-similarity

When the observation preorder $\mathbf{Q}$ is the equality $=$, this monotonicity is strict; in other words, strict inclusion $\lesssim_{M,=} \subsetneq \lesssim_{M+1,=}$ holds. There exists a pair of NAs that does not have an $(M, =)$-simulation but has an $(M+1, =)$-simulation. Fig. 8 shows such pairs; one for $M = 1$, and the other for an arbitrary M. For Fig. 8a, we have a $(2, =)$-simulation $\{(x_0, y_0), (x_1, y_1), (x_1, y_1'), (x_2, y_2), (x_2', y_2')\}$. Similarly, for Fig. 8a, we have a $(M+1, =)$-simulation $\{(x_0, y_0), (x_M, y_M), (x_M, y_M'), (x_{M+1}, y_{M+1}), (x_{M+1}', y_{M+1}')\}$.

The monotonicity is in contrast to that of *nested simulations* [12] which are also parameterised by a positive number. As M increases, $(M, \mathbf{Q})$-simulations become larger (as a relation), while M-nested simulations become smaller. When $M = 1$ (and $\mathbf{Q}$ is the equality), $(1, =)$-simulation coincides with the standard simulation, and hence with 1-nested simulation.

Cor. 2 validates the spectrum of preorder-constrained similarities in Fig. 1, which is governed by the look-ahead bound. Its "limit" can be given by the notion of $\mathbf{Q}$-trace inclusion, thanks to the soundness property (Thm. 1).

The spectrum in Fig. 1 is *generative* because, by instantiating the observation preorder $\mathbf{Q}$, we can obtain various concrete spectra, as shown in Tab. 1. For instance, when $\mathbf{Q}$ is the equality $=$ or the filtered equality $=_{\mathsf{rem}_{\{\tau\}}}$, the spectrum refines a part of (the asymmetric version of) van Glabbeek's spectrum [13,14]. The other spectra of Tab. 1 are more of our interest here, which yield simulation notions that characterise program refinement.

Finally, while $(M, \mathbf{Q})$-simulations are closed under union, they are not closed under the standard composition of relations.

Lemma 2 ((†) basic properties). *Let $M \in \mathbb{N}_+$, and I be an arbitrary set.*

1. *Given a family $\{R_i\}_{i \in I}$ of $(M, \mathbf{Q})$-simulations, $\bigcup_{i \in I} R_i$ is an $(M, \mathbf{Q})$-simulation.*
2. *The $(M, \mathbf{Q})$-similarity $\lesssim_{M,\mathbf{Q}}$ is the largest $(M, \mathbf{Q})$-simulation.*

Example 8 (no closedness under composition).

1. Let $\top = \{\tau\}^* \times \{\tau\}^*$. For the three NAs in Fig. 9, $\{(x, y), (x21, y11), (x22, y12), (x31, y21), (x32, y22), (x33, y23), (x34, y24)\}$ and $\{(y, z), (y21, z11), (y22, z12),$

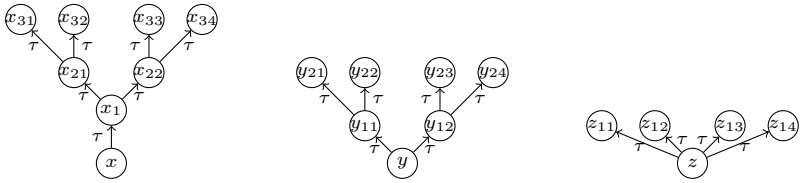


Fig. 9: A counterexample of closedness under composition

(a) $(2,=)$-simulations (b) $(2,\mathbf{Q}_e)$-simulations

Fig. 10: More counterexamples of closedness under composition

$(y23, z13), (y24, z14)\}$ are $(2,\top)$-simulations, but their composition $\{(x,z),$ $(x31, z11), (x32, z12), (x33, z13), (x34,\ z14)\}$ is not a $(2,\top)$-simulation. The composition is a $(3,\top)$-simulation instead.

2. For the three NAs in Fig. 10a, $\{(x,y),(x_1,y_1)\}$ and $\{(y,z)\}$ are $(2,=)$-simulations, but their composition $\{(x,z)\}$ is not a $(2,=)$-simulation.
3. Let $\mathbf{Q}_e$ be the reflexive and c-closed closure of $\{(a,bc),(b,d)\}$. For the three NAs in Fig. 10b, $\{(x,y),(x_1,y_2)\}$ and $\{(y,z),(y_1,z_1)\}$ are $(2,\mathbf{Q}_e)$-simulations, but their composition $\{(x,z)\}$ is not a $(2,\mathbf{Q}_e)$-simulation.

5 Game-Theoretic Characterisation

Preorder-constrained simulations can be characterised by two-player reachability games. The game is parameterised by the observation preorder $\mathbf{Q}$, the look-ahead bound M, and additionally a *catch-up bound* N. Both numerical bounds M, N are now taken from $\mathbb{N}_+ \cup \{\infty\}$.

Definition 5 ($\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$). *Let $M, N \in \mathbb{N}_+$. A two-player game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ between* Challenger *and* Simulator *is defined by Fig. 11. Simulator wins if they reach the state* sim-win*, Challenger has no possible move, or the play continues forever.*

In a game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, Challenger is in charge of $\mathcal{A}_1$, and Simulator is in charge of $\mathcal{A}_2$. Most of the positions are of the form (w,x,y), where $w \in \Sigma^*$ represents a queue of labels that Challenger has inputted into $\mathcal{A}_1$. The two numerical parameters M, N both constrain Simulator's ability to make a move. The look-ahead bound M limits the length of the queue w, and equivalently the number of turns that Simulator can skip consecutively. The catch-up bound N limits the number of transitions Simulator can make in $\mathcal{A}_2$ to dequeue w.

Position	Player	Move	Guard	
(w, x, y) $\in \Sigma^* \times X_1 \times X_2$	Challenger	(wa, x', y)	$x \overset{a}{\rightsquigarrow}_1 x'$	①
		$(\checkmark, w, x, y)$	$x \in F_1$	②
(w, x', y) $\in \Sigma^* \times X_1 \times X_2$	Simulator	(w, x', y)	$\lvert w\rvert < M$	③
		(ε, x', y')	$\exists w' \in \Sigma^*.$ $\lvert w'\rvert < N \wedge y \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y' \wedge w\mathbf{Q}w'$	④
$(\checkmark, w, x, y)$ $\in \{\checkmark\} \times \Sigma^* \times X_1 \times X_2$	Simulator	sim-win	$\exists w' \in \Sigma^*.\exists y' \in F_2.$ $\lvert w'\rvert < N \wedge y \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y' \wedge w\mathbf{Q}w'$	⑤

① Challenger chooses $x \overset{a}{\rightsquigarrow}_1 x'$ from the current state x and enqueues the label a.
② Challenger is at an accepting state $x \in F_1$. Challenger forces Simulator to check whether an accepting state is reachable from $y \in X_2$.
③ Simulator skips the turn. This move is always possible when $M = \infty$.
④ Simulator simulates Challenger's moves in the queue w.
⑤ Simulator simulates Challenger's moves in the queue w and reaches an accepting state.

Fig. 11: Two-player game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ characterising (M-bounded) $\mathbf{Q}$-constrained simulation.

The games $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ satisfy two expected properties: monotonicity with respect to both M and N (Lem. 3 below), and correspondence with $(M, \mathbf{Q})$-similarity (Prop. 3 below). These results validate a generative (two-dimensional) spectrum of games and preorder-constrained simulations shown in Fig. 12.

Lemma 3 (monotonicity). *Let $M, M' \in \mathbb{N}_+$ such that $M \leq M'$, and let $N, N' \in \mathbb{N}_+$ such that $N \leq N'$. If Simulator is winning from a state (w, x, y) in $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, Simulator is also winning from the state in $\mathcal{G}^{M',N',\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$.* □

Proposition 3 (correctness). *Let $M \in \mathbb{N}_+$. Simulator is winning from a state (ε, x, y) in $\mathcal{G}^{M,\infty,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, if and only if $x \lesssim_{M,\mathbf{Q}} y$.* □

We conclude this section with complexity analysis of determining winning positions in $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, and hence of determining $(M, \mathbf{Q})$-similarity $\lesssim_{M,\mathbf{Q}}$.

Proposition 4 (complexity). *Assume that $w\mathbf{Q}w'$ can be checked in linear time to the lengths of w and w'. For $M, N \in \mathbb{N}_+$, whether Simulator is winning from a state (ε, x, y) in $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ can be checked in $\mathcal{O}(|\Sigma|^{M+N} \times |X_1| \times |X_2|^2)$ time.*

Proof. For a set X and for $k, l \in \mathbb{N}$ such that $k \leq l$, let $X^{[k,l]}$ denote the set $\{x_1 x_2 \dots x_n \mid k \leq n \leq l; x_1, \dots, x_n \in X\}$.

We first approximate the complexity to construct the game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, assuming that the membership to F_1, F_2, $\rightsquigarrow_1$ and $\rightsquigarrow_2$ can be checked in constant time. The number of positions of the game is $\mathcal{O}(|\Sigma|^M \times |X_1| \times |X_2|)$, because Challenger's

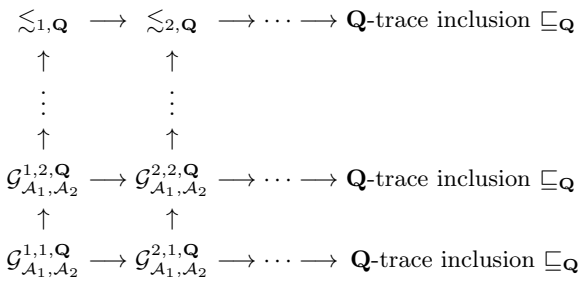


Fig. 12: A generative spectrum of games $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ and similarities $\lesssim_{M,\mathbf{Q}}$

positions are elements of $\Sigma^{[0,M-1]} \times X_1 \times X_2$, and Simulator's positions are elements of $\Sigma^{[0,M]} \times X_1 \times X_2$ or $\{\checkmark\} \times \Sigma^{[0,M]} \times X_1 \times X_2$.

For each pair (p, q) of positions, we approximate the complexity to determine if a move is possible from p to q, by case analysis on the possible moves (see Fig. 11).

- Case ①. It suffices to check $x \overset{a}{\rightsquigarrow}_1 x'$, which can be done in constant time.
- Case ②. It suffices to check $x \in F_1$, which can be done in constant time.
- Case ③. It suffices to check $|w| < M$, which can be done in constant time, because M is a constant here.
- Case ④. By the condition $|w'| < N$, w' can be chosen from $\Sigma^{[0,N-1]}$. Checking $y \overset{w'}{\rightsquigarrow}_2 y'$ is $\mathcal{O}(N-1)$ time and hence constant time, and checking $w\mathbf{Q}w'$ is also constant time, because both M and N are constants here. Overall, this move can be determined in $\mathcal{O}(|\Sigma|^N)$ time.
- Case ⑤. This case is similar to the case ④, with extra choice of y' from $F_2 \subseteq X_2$. This move can be determined in $\mathcal{O}(|\Sigma|^N \times |X_2|)$ time.

Once the game is constructed, its winning region (i.e. the positions from which Simulator is winning) can be determined in $\mathcal{O}(|\Sigma|^M \times |X_1| \times |X_2|)$ time multiplied by $\mathcal{O}(|\Sigma|^N \times |X_2|)$ time, because the game is a reachability game. Consequently, whether Simulator is winning from a state (ε, x, y) in $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ can be checked in $\mathcal{O}(|\Sigma|^{M+N} \times |X_1| \times |X_2|^2)$ time. □

6 Preorder-Constrained Simulation Up-To

We here integrate the up-to technique [32] into preorder-constrained simulation. The technique is widely used for enhancing (bi-)simulation notions. It allows a smaller relation, which is not necessarily a simulation itself, to witness trace inclusion. The up-to version of preorder-constrained simulation is additionally parameterised by a pair of binary relations (R_1, R_2) on state spaces. The following definition is obtained by simply replacing R in Def. 3 with $R_1; R; R_2$, where ; is the composition of binary relations.

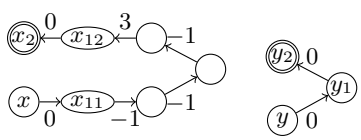


Fig. 13: A $(1, \geq_+)$-simulation up-to

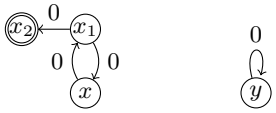


Fig. 14: An unsound $(1, \geq_+)$-simulation up-to

Definition 6 ($(M, \mathbf{Q})$-simulations up to (R_1, R_2)). *Let $R_1 \subseteq X_1 \times X_1$ and $R_2 \subseteq X_2 \times X_2$. For each $M \in \mathbb{N}_+$, a binary relation $R \subseteq X_1 \times X_2$ is an $(M, \mathbf{Q})$-simulation* up to (R_1, R_2) *from $\mathcal{A}_1$ to $\mathcal{A}_2$ if, for any $(x, y) \in R$, $\textbf{\textit{Final}}^M$ (see Def. 3) and $\textbf{\textit{U-Step}}^M$ below hold.*

$\textbf{\textit{U-Step}}^M$ *For each $w = a_1 \dots a_M \in \Sigma^M$ and $x_1 \dots x_M \in X_1^M$ such that $x \overset{a_1}{\rightsquigarrow}_1 x_1 \overset{a_2}{\rightsquigarrow}_1 \cdots \overset{a_M}{\rightsquigarrow}_1 x_M$, there exist $k \in \{1, \dots, M\}$, $w' \in \Sigma^*$ and $y' \in X_2$ such that $a_1 \cdots a_k \mathbf{Q} w'$, $y \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y'$ and $x_k (R_1; R; R_2) y'$.*

The question now is when a preorder-constrained simulation up to (R_1, R_2) is sound with respect to $\mathbf{Q}$-trace inclusion. The relations R_1 and R_2 cannot be arbitrary, and they should be *consistent* to the $\mathbf{Q}$-trace inclusion $\sqsubseteq_{\mathbf{Q}}$. Formally, they should satisfy $R_1 \subseteq \sqsubseteq_{\mathbf{Q}_1}$ and $R_2 \subseteq \sqsubseteq_{\mathbf{Q}_2}$ for some preorders $\mathbf{Q}_1, \mathbf{Q}_2 \subseteq \Sigma^* \times \Sigma^*$ such that $(\mathbf{Q}_1; \mathbf{Q}; \mathbf{Q}_2) \subseteq \mathbf{Q}$.

Example 9 ($(M, \geq_+)$-constrained simulation up to $(\sqsubseteq_{\geq_+}, \sqsubseteq_{\geq_+})$). Recall the preorder $\geq_+$ from Ex. 5(6). It is feasible to use a $\geq_+$-constrained simulation up to $(\sqsubseteq_{\geq_+}, \sqsubseteq_{\geq_+})$. The first reason is that the consistence property is satisfied, thanks to transitivity of the preorder $\geq_+$. The second reason is that the $\geq_+$-trace inclusion $\sqsubseteq_{\geq_+}$ between two states of the same NA can be checked efficiently with a graph algorithm. Specifically, for two states x, x' of the same NA, $x \sqsubseteq_{\geq_+} x'$ holds if the summation of weights on each path from x to x' is non-negative.

Example 10 ($(1, \geq_+)$-simulation up-to). For a pair of NAs shown in Fig. 13, $x \sqsubseteq_{\geq_+} y$ holds, but there exists no $(1, \geq_+)$-simulation that relates x with y. In contrast, a relation $R = \{(x, y), (x_{12}, y_1), (x_2, y_2)\}$ is a $(1, \geq_+)$-simulation up to $(\sqsubseteq_{\geq_+}, \sqsubseteq_{\geq_+})$. The pair (x, y) satisfies the condition $\textbf{\textit{U-Step}}^1$ of Def. 6, because we have $x_{11} \sqsubseteq_{\geq_+} x_{12}\ R\ y_1 \sqsubseteq_{\geq_+} y_1$. The up-to allows us to *move* $\textbf{\textit{U-Step}}^1$ *towards the accepting state* x_2, that is, it allows us to deal with $\textbf{\textit{U-Step}}^1$ of (x_{12}, y_1) instead of (x_{11}, y_1).

However, the consistence property $(\mathbf{Q}_1; \mathbf{Q}; \mathbf{Q}_2) \subseteq \mathbf{Q}$ is not enough to achieve soundness. A naive combination of the weak simulation notion, which coincides with our similarity $\lesssim_{1, =_{\mathsf{rem}_{\{\tau\}}}}$, and an up-to technique is known to be unsound, and requires special care [30,31]. The following counterexample is inspired by the one for weak simulation from the literature [30,31].

Example 11 (unsound $(1, \geq_+)$-simulation up-to). For a pair of NAs shown in Fig. 14, $x \sqsubseteq_{\geq_+} y$ does not hold, because no accepting state is reachable from y.

However, a relation $\{(x, y)\}$ is a $(1, \geq_+)$-simulation up to $(\sqsubseteq_{\geq_+}, \sqsubseteq_{\geq_+})$. The pair (x, y) indeed satisfies $\textbf{\textit{U-Step}}^1$, because $x_1 \sqsubseteq_{\geq_+} x \; R \; y \sqsubseteq_{\geq_+} y$. Here, the up-to allows us to *move* $\textbf{\textit{U-Step}}^1$ *away from the accepting state* x_2: namely, it allows us to deal with $\textbf{\textit{U-Step}}^1$ of (x, y) instead of (x_1, y).

From the two examples above, we can extract an additional condition on the preorders $\mathbf{Q}, \mathbf{Q}_1, \mathbf{Q}_2$, namely on $\mathbf{Q}_1$. We can observe that, while it is safe to move $\textbf{\textit{U-Step}}^M$ *toward* an accepting state (Ex. 10), it is unsafe to move it *away from* an accepting state (Ex. 11), for the sake of soundness. To prohibit the unsafe move, the additional condition is required, namely $w\mathbf{Q}_1 w' \implies |w| \geq |w'|$. Examples of such $\mathbf{Q}_1$ are the equality $=$, the preorder $\dot{\geq}$ induced by $\geq \subseteq \mathbb{N} \times \mathbb{N}$, the inverse $(\subseteq_{\mathsf{sub}})^{-1}$, and $\subseteq^*$ (see Ex. 5).

The condition is inspired by a soundness criterion for counting simulation up-to [27, Def. 4.3.13]. A similar idea can be found in Pous' work [30,31] on weak simulation up-to.

Theorem 2 ((†) soundness). *Let $M \in \mathbb{N}_+$. Let $R \subseteq X_1 \times X_2$, $R_1 \subseteq \sqsubseteq_{\mathbf{Q}_1}$ and $R_2 \subseteq \sqsubseteq_{\mathbf{Q}_2}$ be binary relations, for preorders $\mathbf{Q}_1, \mathbf{Q}_2 \subseteq \Sigma^* \times \Sigma^*$ such that (i) $(\mathbf{Q}_1; \mathbf{Q}; \mathbf{Q}_2) \subseteq \mathbf{Q}$ and (ii) $w\mathbf{Q}_1 w' \implies |w| \geq |w'|$. If R is an $(M, \mathbf{Q})$-simulation up to (R_1, R_2), it holds that $xRy \implies x \sqsubseteq_{\mathbf{Q}} y$ for any $(x, y) \in X_1 \times X_2$.*

Example 12 (weak simulation up to "expansion/contraction"). Let $\mathbf{Q}_e$ be a preorder $=_{\mathsf{rem}_{\{\tau\}}} \cap \dot{\geq}$. The $(1, \mathbf{Q}_e)$-similarity $\lesssim_{1,\mathbf{Q}_e}$ is akin to *expansion* [34] and *contraction* [33] that have been used with an up-to technique. It is indeed valid to have the weak simulation up to the similarity $\lesssim_{1,\mathbf{Q}_e}$. More precisely, we can have a $(1, =_{\mathsf{rem}_{\{\tau\}}})$-simulation, which is the weak simulation, up to $(\lesssim_{1,\mathbf{Q}_e}, =)$. The three preorders $=_{\mathsf{rem}_{\{\tau\}}}$, $\mathbf{Q}_e$ and $=$ satisfy the two conditions in Thm. 2; we have $(\mathbf{Q}_e; =_{\mathsf{rem}_{\{\tau\}}}; =) \subseteq =_{\mathsf{rem}_{\{\tau\}}}$, and the preorder $\mathbf{Q}_e$ satisfies $\mathbf{Q}_e \subseteq \dot{\geq}$. We also have $\lesssim_{1,\mathbf{Q}_e} \subseteq \sqsubseteq_{\mathbf{Q}_e}$ by soundness (Thm. 1), and $= \subseteq \sqsubseteq_=$.

7 Related Work

Our two-player game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ is similar to *buffered simulation game* [17,16]. While ours is for NAs with finite languages, the latter is for Büchi automata. We contribute to coinductively defining the simulations and investigating the generative spectrum.

Quantitative simulation notions are known for *weighted automata*: many are for probabilistic systems [24,19,15]; a general simulation notion for automata weighted with semirings (e.g. $\mathbb{R}$ with $+$ and $\times$, and $\mathbb{R}$ with max and $+$ a.k.a. tropical semiring) was introduced as a matrix over real numbers [37].

Preorder-constrained simulations are defined for NAs, but can be quantitative, in the sense that they can compare lengths of accepted runs (e.g. using the preorder $\dot{\geq}$). Known (bi-)simulation notions such as *expansion* [34] and *contraction* [33] are also capable of such quantitative comparison. As mentioned in Ex. 12, the $(1, =_{\mathsf{rem}_{\{\tau\}}} \cap \dot{\geq})$-similarity is akin to these (bi-)simulation notions seen as simulation notions.

Prop. 4 suggests that we can reduce the problem of checking refinement between programs to determining a winning region of the reachability game $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$. *Algorithmic game semantics* [2,20] is another approach to reduce program refinement to solving games. A notable difference is that algorithmic game semantics restricts *types* of programs, while our approach would require a program t to induce a *finite* automaton $\mathcal{A}_\Omega(t)$.

Our work is not the first to characterise or refine the LT–BT spectrum using a simulation notion or a game. It would be interesting to compare existing work, e.g. [10,8], to ours in details. Ordered words are not a new research topic either, but the literature seems to focus on decidability of their theory, e.g. [22,23].

8 Conclusion and Future Work

We proposed a notion of preorder-constrained simulation. Being parameterised by the observation preorder on traces, it can uniformly characterise quantitative notions of observational refinement for different algebraic effects: exception, nondeterminism and I/O. We demonstrated this using reduction semantics for the LSC.

Being additionally parameterised by the look-ahead bound, preorder-constrained simulations form a generative spectrum. Its "limit" is given by a novel generalisation of trace inclusion. The spectrum is generative in the sense that it can be instantiated variously according to the observation preorder.

We additionally presented a characterisation of preorder-constrained simulation as a two-player reachability game, and showed that preorder-constrained similarity can be determined in polynomial time in the finite setting. Finally, we studied enhancement of preorder-constrained simulation, showing how to integrate an up-to technique, in terms of observation preorders.

One direction of future work is to extend the characterisation of program refinement to probabilistic choice. This would require preorder-constrained simulation to work on weighted automata instead of NAs. One can try to accommodate probabilistic choice into the current work, using the preorder $\leq_+$, but a naive approach does not work. It results in a false refinement such as $\mathtt{or}_{0.5}(\underline{1},\underline{1}) \sqsubseteq_{\leq_+} \mathtt{or}_{0.5}(\underline{0},\underline{1})$, where $\mathtt{or}_{0.5}$ is an operation that chooses either argument with probability 0.5. It would also be interesting to connect preorder-constrained simulation to a generic metatheory of algebraic effects [18,35].

Another future work is to develop a methodology to constructing a preorder-constrained simulation. For the automaton $\mathcal{A}_\Omega$ induced by a signature Ω, it would be particularly important to construct a preorder-constrained simulation that is closed under term construction. Such a simulation would characterise *contextual refinement*, which asserts refinement between two terms in an arbitrary context. Counting simulation was originally used in this way [27].

Acknowledgments. We are grateful to Ichiro Hasuo and Shigeru Chiba for insightful comments. The first and second authors are supported by JST, ACT-X Grant No. JPMJAX190U, Japan, and JSPS, KAKENHI Project No. 22K17850, Japan. The third

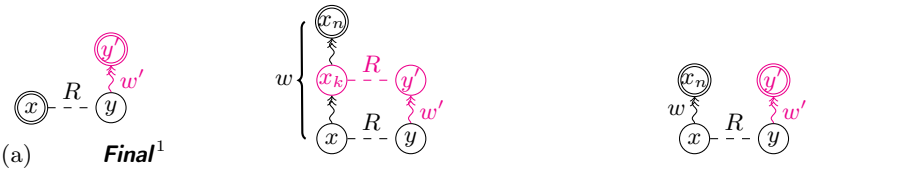
author is supported by JST ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603).

References

1. Abramsky, S.: The lazy lambda-calculus, p. 65–117. Addison Wesley (1990)
2. Abramsky, S.: Algorithmic Game Semantics, NATO Science Series, vol. 62, pp. 21–47. Springer Netherlands, Dordrecht (2002). https://doi.org/10.1007/978-94-010-0413-8_2, https://_doi.org/10.1007/978-94-010-0413-8_2
3. Accattoli, B., Barenbaum, P., Mazza, D.: Distilling abstract machines. In: ICFP 2014. pp. 363–376. ACM (2014). https://doi.org/10.1145/2628136.2628154
4. Accattoli, B., Dal Lago, U., Vanoni, G.: The machinery of interaction. In: PPDP '20: 22nd International Symposium on Principles and Practice of Declarative Programming, Bologna, Italy, 9-10 September, 2020. pp. 4:1–4:15. ACM (2020), https://doi.org/10.1145/3414080.3414108
5. Accattoli, B., Kesner, D.: The structural lambda-calculus. In: CSL 2010. vol. 6247, pp. 381–395. Springer (2010). https://doi.org/10.1007/978-3-642-15205-4_30
6. Baier, C., Katoen, J.: Principles of model checking. MIT Press (2008)
7. Banerjee, A., Pal, B., Das, S., Kumar, A., Dasgupta, P.: Test generation games from formal specifications. In: Sentovich, E. (ed.) Proceedings of the 43rd Design Automation Conference, DAC 2006, San Francisco, CA, USA, July 24-28, 2006. pp. 827–832. ACM (2006), https://doi.org/10.1145/1146909.1147120
8. Bisping, B., Jansen, D.N., Nestmann, U.: Deciding all behavioral equivalences at once: A game for linear-time-branching-time spectroscopy. Log. Methods Comput. Sci. **18**(3) (2022), https://doi.org/10.46298/lmcs-18(3:19)2022
9. Cormie-Bowins, E., van Breugel, F.: Measuring progress of probabilistic LTL model checking. In: Wiklicky, H., Massink, M. (eds.) Proceedings 10th Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL 2012, Tallinn, Estonia, 31 March and 1 April 2012. EPTCS, vol. 85, pp. 33–47 (2012), https://doi.org/10.4204/EPTCS.85.3
10. de Frutos-Escrig, D., Gregorio-Rodríguez, C., Palomino, M., Romero-Hernández, D.: Unifying the linear time-branching time spectrum of process semantics. Log. Methods Comput. Sci. **9**(2) (2013), https://doi.org/10.2168/LMCS-9(2:11)2013
11. De Giacomo, G., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: Rossi, F. (ed.) IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, Beijing, China, August 3-9, 2013. pp. 854–860. IJCAI/AAAI (2013), http://www.aaai.org/ocs/index.php/IJCAI/IJCAI13/paper/view/6997
12. de Frutos-Escrig, D., Gregorio-Rodríguez, C.: Constrained simulations, nested simulation semantics and counting bisimulations. In: Pimentel, E. (ed.) Proceedings of the Seventh Spanish Conference on Programming and Computer Languages, PROLE 2007, Zaragoza, Spain, September 12-14, 2007. Electronic Notes in Theoretical Computer Science, vol. 206, pp. 41–58. Elsevier (2007), https://doi.org/10.1016/j.entcs.2008.03.074
13. van Glabbeek, R.J.: The linear time-branching time spectrum (extended abstract). In: Baeten, J.C.M., Klop, J.W. (eds.) CONCUR '90, Theories of Concurrency: Unification and Extension, Amsterdam, The Netherlands, August 27-30, 1990, Proceedings. Lecture Notes in Computer Science, vol. 458, pp. 278–297. Springer (1990), https://doi.org/10.1007/BFb0039066

14. van Glabbeek, R.J.: The linear time - branching time spectrum II. In: Best, E. (ed.) CONCUR '93, 4th International Conference on Concurrency Theory, Hildesheim, Germany, August 23-26, 1993, Proceedings. Lecture Notes in Computer Science, vol. 715, pp. 66–81. Springer (1993), https://doi.org/10.1007/3-540-57208-2_6
15. Hughes, J., Jacobs, B.: Simulations in coalgebra. Theor. Comput. Sci. **327**(1-2), 71–108 (2004), https://doi.org/10.1016/j.tcs.2004.07.022
16. Hutagalung, M.: Buffered Simulation for Büchi Automata. Ph.D. thesis, University of Kassel, Germany (2019), https://kobra.uni-kassel.de/bitstream/handle/123456789/11329/DissertationMilkaHutagalung.pdf?sequence=7&isAllowed=y
17. Hutagalung, M., Lange, M., Lozes, É.: Buffered simulation games for büchi automata. In: Ésik, Z., Fülöp, Z. (eds.) Proceedings 14th International Conference on Automata and Formal Languages, AFL 2014, Szeged, Hungary, May 27-29, 2014. EPTCS, vol. 151, pp. 286–300 (2014), https://doi.org/10.4204/EPTCS.151.20
18. Johann, P., Simpson, A., Voigtländer, J.: A generic operational metatheory for algebraic effects. In: Proceedings of the 25th Annual IEEE Symposium on Logic in Computer Science, LICS 2010, 11-14 July 2010, Edinburgh, United Kingdom. pp. 209–218. IEEE Computer Society (2010), https://doi.org/10.1109/LICS.2010.29
19. Jonsson, B., Larsen, K.G.: Specification and refinement of probabilistic processes. In: Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS '91), Amsterdam, The Netherlands, July 15-18, 1991. pp. 266–277. IEEE Computer Society (1991), https://doi.org/10.1109/LICS.1991.151651
20. Kiefer, S., Murawski, A.S., Ouaknine, J., Wachter, B., Worrell, J.: Algorithmic probabilistic game semantics - playing games with automata. Formal Methods Syst. Des. **43**(2), 285–312 (2013), https://doi.org/10.1007/s10703-012-0173-1
21. Koutavas, V., Levy, P.B., Sumii, E.: From applicative to environmental bisimulation. In: Mislove, M.W., Ouaknine, J. (eds.) Twenty-seventh Conference on the Mathematical Foundations of Programming Semantics, MFPS 2011, Pittsburgh, PA, USA, May 25-28, 2011. Electronic Notes in Theoretical Computer Science, vol. 276, pp. 215–235. Elsevier (2011), https://doi.org/10.1016/j.entcs.2011.09.023
22. Kuske, D.: Theories of orders on the set of words. RAIRO Theor. Informatics Appl. **40**(1), 53–74 (2006), https://doi.org/10.1051/ita:2005039
23. Kuske, D., Zetzsche, G.: Languages ordered by the subword order. In: Bojanczyk, M., Simpson, A. (eds.) Foundations of Software Science and Computation Structures - 22nd International Conference, FOSSACS 2019, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2019, Prague, Czech Republic, April 6-11, 2019, Proceedings. Lecture Notes in Computer Science, vol. 11425, pp. 348–364. Springer (2019), https://doi.org/10.1007/978-3-030-17127-8_20
24. Larsen, K.G., Skou, A.: Bisimulation through probabilistic testing. Inf. Comput. **94**(1), 1–28 (1991), https://doi.org/10.1016/0890-5401(91)90030-6
25. Morgenstern, A., Schneider, K.: A LTL fragment for GR(1)-synthesis. In: Reich, J., Finkbeiner, B. (eds.) Proceedings International Workshop on Interactions, Games and Protocols, iWIGP 2011, Saarbrücken, Germany, 27th March 2011. EPTCS, vol. 50, pp. 33–45 (2011), https://doi.org/10.4204/EPTCS.50.3
26. Morris Jr, J.H.: Lambda-calculus models of programming languages. Ph.D. thesis, Massachusetts Institute of Technology (1969), https://dspace.mit.edu/handle/1721.1/64850
27. Muroya, K.: Hypernet semantics of programming languages. Ph.D. thesis, University of Birmingham, UK (2020), http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.817915

28. Muscholl, A., Walukiewicz, I.: An NP-complete fragment of LTL. In: Calude, C., Calude, E., Dinneen, M.J. (eds.) Developments in Language Theory, 8th International Conference, DLT 2004, Auckland, New Zealand, December 13-17, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3340, pp. 334–344. Springer (2004), https://doi.org/10.1007/978-3-540-30550-7_28
29. Plotkin, G.D., Power, J.: Adequacy for algebraic effects. In: Honsell, F., Miculan, M. (eds.) Foundations of Software Science and Computation Structures, 4th International Conference, FOSSACS 2001 Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2001 Genova, Italy, April 2-6, 2001, Proceedings. Lecture Notes in Computer Science, vol. 2030, pp. 1–24. Springer (2001), https://doi.org/10.1007/3-540-45315-6_1
30. Pous, D.: Up-to techniques for weak bisimulation. In: Caires, L., Italiano, G.F., Monteiro, L., Palamidessi, C., Yung, M. (eds.) Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3580, pp. 730–741. Springer (2005), https://doi.org/10.1007/11523468_59
31. Pous, D.: New up-to techniques for weak bisimulation. Theor. Comput. Sci. **380**(1-2), 164–180 (2007), https://doi.org/10.1016/j.tcs.2007.02.060
32. Sangiorgi, D.: On the bisimulation proof method. Mathematical Structures in Computer Science **8**(5), 447–479 (1998). https://doi.org/10.1017/S0960129598002527
33. Sangiorgi, D.: Equations, contractions, and unique solutions. ACM Trans. Comput. Log. **18**(1), 4:1–4:30 (2017), https://doi.org/10.1145/2971339
34. Sangiorgi, D., Milner, R.: The problem of "weak bisimulation up to". In: Cleaveland, R. (ed.) CONCUR '92, Third International Conference on Concurrency Theory, Stony Brook, NY, USA, August 24-27, 1992, Proceedings. Lecture Notes in Computer Science, vol. 630, pp. 32–46. Springer (1992), https://doi.org/10.1007/BFb0084781
35. Simpson, A., Voorneveld, N.F.W.: Behavioural equivalence via modalities for algebraic effects. In: Ahmed, A. (ed.) Programming Languages and Systems - 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10801, pp. 300–326. Springer (2018), https://doi.org/10.1007/978-3-319-89884-1_11
36. Tiuryn, J., Wand, M.: Untyped lambda-calculus with input-output. In: Kirchner, H. (ed.) Trees in Algebra and Programming - CAAP'96, 21st International Colloquium, Linköping, Sweden, April, 22-24, 1996, Proceedings. Lecture Notes in Computer Science, vol. 1059, pp. 317–329. Springer (1996), https://doi.org/10.1007/3-540-61064-2_46
37. Urabe, N., Hasuo, I.: Generic forward and backward simulations III: quantitative simulations by matrices. In: Baldan, P., Gorla, D. (eds.) CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings. Lecture Notes in Computer Science, vol. 8704, pp. 451–466. Springer (2014), https://doi.org/10.1007/978-3-662-44584-6_31
38. Yachi, T., Sumii, E.: A sound and complete bisimulation for contextual equivalence in λ-calculus with call/cc. In: Igarashi, A. (ed.) Programming Languages and Systems - 14th Asian Symposium, APLAS 2016, Hanoi, Vietnam, November 21-23, 2016, Proceedings. Lecture Notes in Computer Science, vol. 10017, pp. 171–186 (2016), https://doi.org/10.1007/978-3-319-47958-3_10



(a) ***Final***^{1} where $\varepsilon \mathbf{Q} w'$ (b) ***Step***$^\infty$ (i) where $a_1 \cdots a_k \mathbf{Q} w'$ (c) ***Step***$^\infty$ (ii) where $w \mathbf{Q} w'$

Fig. 15: Conditions of Def. 7. Black parts are universally quantified, and magenta parts are existentially quantified.

A Completeness

It is possible to adjust the definition of preorder-constrained simulation (Def. 3) and obtain a complete simulation notion with respect to **Q**-trace inclusion.

Definition 7. *A binary relation $R \subseteq X_1 \times X_2$ is a* **Q**-constrained simulation *(***Q**-simulation *in short) from $\mathcal{A}_1$ to $\mathcal{A}_2$ if, for any $(x, y) \in R$,* ***Final***^{1} *and the following* ***Step***$^\infty$ *hold.*

Step$^\infty$ *For each $a_1 \dots a_n \in \Sigma^+$ and $x_1 \dots x_n \in X_1^+$ such that $x \overset{a_1}{\rightsquigarrow}_1 x_1 \overset{a_2}{\rightsquigarrow}_1 \cdots \overset{a_n}{\rightsquigarrow}_1 x_n$ and $x_n \in F_1$, there exist $k \in \{1, \dots, n\}$, $w' \in \Sigma^*$ and $y' \in X_2$ such that (1) $a_1 \cdots a_k \mathbf{Q} w'$; (2) $y \overset{w'}{\rightsquigarrow}_2 y'$; and (3-i) $x_k R y'$, or (3-ii) $k = n$ and $y' \in F_2$.*

Fig. 15 illustrates the conditions of Def. 7. The condition ***Final***^{1} is an instance of ***Final***^{M} (see also Fig. 6). The condition ***Step***$^\infty$ has two possibilities (i) and (ii), due to the clauses (3-i) and (3-ii). Note that ***Step***$^\infty$ is not quite a limit of ***Step***^{M}. While the length of the sequence $x \overset{w}{\rightsquigarrow} x_M$ is restricted to M in ***Step***^{M}, the length n of the sequence $x \overset{w}{\rightsquigarrow} x_n$ in ***Step***$^\infty$ can be an arbitrary positive number. Additionally, the last state x_n is required to be accepting, i.e. $x_n \in F_1$.

Theorem 3 ((†) soundness and completeness). *For any $(x, y) \in X_1 \times X_2$, $x \sqsubseteq_{\mathbf{Q}} y \iff x \lesssim_{\mathbf{Q}} y$.* □

We note that it is the extra clause of the definition of **Q**-simulations, namely (3-ii) of ***Step***$^\infty$, that makes the completeness possible.

Corollary 3 (correctness of Q-simulations wrt. refinement).

1. *For any $t, u \in \mathbf{T}_{\Omega_{\mathsf{err}}}$, $t \lesssim_{\dot{Q}} u \iff t \preceq^Q_{\mathsf{err}} u$.*
2. *For any $t, u \in \mathbf{T}_{\Omega_{\mathsf{nd}}}$, $t \lesssim_{\dot{Q} \cap =_{\mathsf{rem}\{\tau\} \cup \overline{\Omega_{\mathsf{nd}}}}} u \iff t \preceq^Q_{\mathsf{nd}} u$.*
3. *For any $t, u \in \mathbf{T}_{\Omega_{\mathsf{io}}}$, $t \lesssim_{\dot{Q} \cap =_{\mathsf{rem}\{\tau\}}} u \iff t \preceq^Q_{\mathsf{io}} u$.* □

Whereas Q-counting simulation (for $Q \subseteq \mathbb{N} \times \mathbb{N}$) is defined for NAs (see Def. 2), the definition is tailored to branching-free NAs all whose accepting states are stuck states. The NA $\mathcal{A}_{\Omega_{\mathsf{err}}}$ is an example. For these automata, preorder-constrained simulation indeed generalises counting simulation. However, the opposite does not hold.

Fig. 16: Branching-free NAs with a trivial $\doteq$-simulation

Proposition 5 (Q-counting simulation being $\dot{Q}$-simulation). *If $\mathcal{A}_1$ is branching-free and all its accepting states are stuck states, then each Q-counting simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$ is also a $\dot{Q}$-simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$.* □

Example 13 ($\dot{Q}$-simulation not being Q-counting simulation). For the pair of branching-free NAs shown in Fig. 16, $\{(x, y)\}$ is a $\doteq$-simulation, because these NAs have no accepting states. However, there exists no $=$-counting simulation that includes (x, y). If such a simulation R exists, $(x_1, y_1) \in R$ must hold by ***C-Step***, but (x_1, y_1) does not satisfy ***C-Step***.

The monotonicity property satisfied by $(M, \mathbf{Q})$-simulations can be extended to $\mathbf{Q}$-simulations.

Lemma 4 ((†) monotonicity of *Step*). *Let $M \in \mathbb{N}_+$.*

1. $\mathit{Step}^M \wedge \mathit{Final}^M \implies \mathit{Step}^\infty$

Corollary 4 (monotonicity). *Let $M \in \mathbb{N}_+$. Each $(M, \mathbf{Q})$-simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$ is also a $\mathbf{Q}$-simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$.* □

Notably, we do not need to modify the definition of two-player games $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$; we can simply set $M = N = \infty$ to achieve games that correspond with $\mathbf{Q}$-simulations. We can extend the spectrum in Fig. 12 and obtain a generative spectrum shown in Fig. 17.

Lemma 5 (monotonicity). *Let $M, N \in \mathbb{N}_+$. If Simulator is winning from a state (w, x, y) in $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, Simulator is also winning from the state in $\mathcal{G}^{\infty,\infty,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$.* □

Proposition 6 (correctness). *Simulator is winning from a state (ε, x, y) in $\mathcal{G}^{\infty,\infty,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$, if and only if $x \lesssim_{\mathbf{Q}} y$.* □

B Connection to Logical Implication

We briefly investigate the relationship between $\mathbf{Q}$-trace inclusion and logics, namely fragments of the *linear temporal logic (LTL)*, aiming at a potential application to model checking.

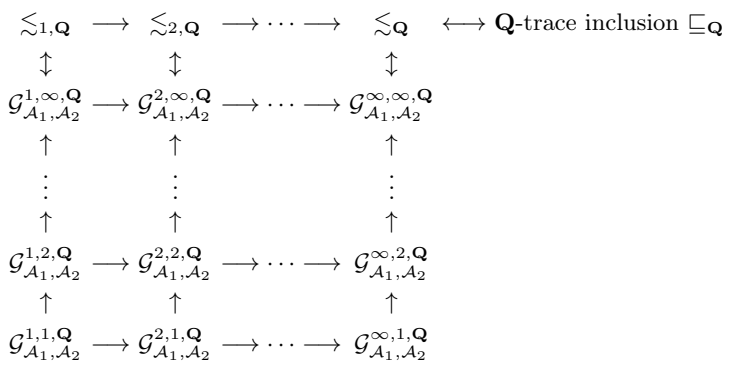


Fig. 17: A generative spectrum of games $\mathcal{G}^{M,N,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ and similarities $\lesssim_{M,\mathbf{Q}}, \lesssim_{\mathbf{Q}}$

Definition 8 (LTL, see e.g. [6]). *A* linear temporal logic formula *(LTL formula) over a set* AP *is given by the following BNF notation where* $p \in \mathsf{AP}$*.*

$$\varphi ::= \mathsf{true} \mid \mathsf{false} \mid p \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\,\varphi \mid \varphi\ \mathsf{U}\ \varphi \mid \varphi\ \mathsf{R}\ \varphi \tag{1}$$

The semantics of an LTL formula φ is usually defined as a subset $[\![\varphi]\!] \subseteq (2^{\mathsf{AP}})^\omega$ of infinite words over 2^{AP}. We use its finitary variant here, as we are focusing on finite traces.

Definition 9 (LTL semantics on finite traces, cf. [11]). *For an LTL formula* φ*, we inductively define* $[\![\varphi]\!]_{\mathsf{fin}} \subseteq (2^{\mathsf{AP}})^*$ *as follows:*

- $[\![\mathsf{true}]\!]_{\mathsf{fin}} := (2^{\mathsf{AP}})^*$
- $[\![\mathsf{false}]\!]_{\mathsf{fin}} := \emptyset$
- $[\![p]\!]_{\mathsf{fin}} := \{a_1 \dots a_n \mid n > 0, p \in a_1\}$
- $[\![\neg p]\!]_{\mathsf{fin}} := \{a_1 \dots a_n \mid n > 0, p \notin a_1\}$
- $[\![\varphi_1 \vee \varphi_2]\!]_{\mathsf{fin}} := [\![\varphi_1]\!]_{\mathsf{fin}} \cup [\![\varphi_2]\!]_{\mathsf{fin}}$
- $[\![\varphi_1 \wedge \varphi_2]\!]_{\mathsf{fin}} := [\![\varphi_1]\!]_{\mathsf{fin}} \cap [\![\varphi_2]\!]_{\mathsf{fin}}$
- $[\![\mathsf{X}\,\varphi]\!]_{\mathsf{fin}} := \{a_1 \dots a_n \mid n > 0, a_2 \dots a_n \in [\![\varphi]\!]_{\mathsf{fin}}\}$
- $[\![\varphi_1\ \mathsf{U}\ \varphi_2]\!]_{\mathsf{fin}} := \left\{ a_1 \dots a_n \,\middle|\, \exists i \in \{1,\dots,n\}.\ \begin{pmatrix} (\forall j < i.\ a_j \dots a_n \in [\![\varphi_1]\!]_{\mathsf{fin}}) \\ \wedge\ \ a_i \dots a_n \in [\![\varphi_2]\!]_{\mathsf{fin}} \end{pmatrix} \right\}$
- $[\![\varphi_1\ \mathsf{R}\ \varphi_2]\!]_{\mathsf{fin}} := \left\{ a_1 \dots a_n \,\middle|\, \forall i \in \{1,\dots,n\}.\ \begin{pmatrix} (\exists j < i.\ a_j \dots a_n \in [\![\varphi_1]\!]_{\mathsf{fin}}) \\ \vee\ \ a_i \dots a_n \in [\![\varphi_2]\!]_{\mathsf{fin}} \end{pmatrix} \right\}$

Let $A = (X, \Sigma, \rightsquigarrow, F)$ *be a* (2^{AP})*-labelled NA. For* $x \in X$*, we write* $x \models_{\mathsf{fin}} \varphi$ *when* $L^*(x) \subseteq [\![\varphi]\!]_{\mathsf{fin}}$*. We also use the following syntactic sugar:* $\mathsf{F}\,\varphi := \mathsf{true}\ \mathsf{U}\ \varphi$ *and* $\mathsf{G}\,\varphi := \mathsf{false}\ \mathsf{R}\ \varphi$*.*

For a state x of an NA $\mathcal{A}$, we write $x \models_{\mathsf{fin}} \varphi$ when $L^*_{\mathcal{A}}(x) \subseteq [\![\varphi]\!]_{\mathsf{fin}}$. Then finite trace inclusion $L^*_{\mathcal{A}_1}(x) \subseteq L^*_{\mathcal{A}_2}(y)$ implies $y \models_{\mathsf{fin}} \varphi \implies x \models_{\mathsf{fin}} \varphi$ for each LTL formula φ. If we replace $L^*_{\mathcal{A}_1}(x) \subseteq L^*_{\mathcal{A}_2}(y)$ with $x \sqsubseteq_{\mathbf{Q}} y$, what happens to φ? We attempt to answer this question, using preorders from Ex. 5 and fragments of LTL found in the literature [28,25,7].

The $\subseteq^*$-trace inclusion implies implication of a fragment of LTL "without" negation.

Proposition 7. *If* $x \sqsubseteq_{\subseteq^*} y$ *then* $y \models_{\mathsf{fin}} \varphi \implies x \models_{\mathsf{fin}} \varphi$ *for each* φ *given by:*

$$\varphi ::= \mathsf{true} \mid \mathsf{false} \mid \neg p \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{X}\,\varphi \mid \varphi\ \mathsf{U}\ \varphi \mid \varphi\ \mathsf{R}\ \varphi\,. \tag{2}$$

We can consider the dual. If we replace $\subseteq^*$ with its inverse $\supseteq^*$, then $\neg p$ in (2) becomes p. This fragment is known as a *positive fragment* (LTL_+) [9].

Next, the $\subseteq_{\mathsf{sub}}$-trace inclusion implies implication of *safety properties* guarded by G.

Proposition 8. *If* $x \sqsubseteq_{\subseteq_{\mathsf{sub}}} y$ *then* $y \models_{\mathsf{fin}} \varphi \Rightarrow x \models_{\mathsf{fin}} \varphi$ *for each* φ *given by:*

$$\varphi ::= \mathsf{true} \mid \mathsf{false} \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \mathsf{G}\,\psi\,, \qquad \psi ::= p \mid \neg p \mid \psi \vee \psi \mid \psi \wedge \psi \mid \varphi\,. \tag{3}$$

We can again consider the dual. Let $\supseteq_{\mathsf{sup}}$ be the *superstring relation* defined by $w \supseteq_{\mathsf{sup}} w' \overset{\text{def}}{\Leftrightarrow} w' \subseteq_{\mathsf{sub}} w$. Then $\supseteq_{\mathsf{sup}}$-trace inclusion implies implication of *liveness properties* guarded by F: namely, $\mathsf{G}\,\psi$ in Prop. 8 is replaced by $\mathsf{F}\,\psi$.

Finally, we can combine Ex. 5(4) with Ex. 5(5), and define $\mathbf{Q}$ so that $a_1 \dots a_k \mathbf{Q} a'_1 \dots a'_{k'}$ means $k \leq k'$ and existence of $j_1 < \dots < j_k$ such that $a_i \subseteq a'_{j_i}$ for each $i \in \{1, \dots, k\}$. Then $\mathbf{Q}$-trace inclusion implies implication of φ's which are given by removing p from (3).

C Omitted Proofs

C.1 Proof of Prop. 1

Proof. This is a consequence of Cor. 3 and Prop. 5.

C.2 Proof of Prop. 5

Proof. Let $R \subseteq X_1 \times X_2$ be a Q-counting simulation, and take an arbitrary pair $(x, y) \in R$. When $x \in F_1$, because the pair satisfies ***C-Final***, $y \in F_1$ follows. Therefore $(x, y) \in R$ also satisfies ***Final***[1].

When $(x, y) \in R$ satisfies ***C-Step***, let $a_1 \cdots a_n \in \Sigma^+$ and $x_1 \cdots x_n \in X_1^+$ satisfy $x \overset{a_1}{\rightsquigarrow}_1 x_1 \overset{a_2}{\rightsquigarrow}_1 \cdots \overset{a_n}{\rightsquigarrow}_1 x_n$ and $x_n \in F_1$. There are two possibilities.

- When the condition (1) holds, there exist $x'' \in X_1$ and $w \in \Sigma^*$ such that $x_1 \overset{w}{\rightsquigarrow\!\!\!\!\rightarrow}_1 x'' \not\rightarrow_1$ and $x'' \notin F_1$. Because $\mathcal{A}_1$ is branching-free and its accepting states are stuck states, $x_n \not\rightarrow_1$ and $x_n = x''$ must hold. We have $x_n \in F_1$ and $x'' \notin F_1$, which is contradiction.
- When the condition (2) holds, there exist $x'' \in X_1$, $y' \in X_2$ and $w, w' \in \Sigma^*$, such that $x_1 \overset{w}{\rightsquigarrow\!\!\!\!\rightarrow}_1 x''$, $y \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y'$, $|a_1 w| Q |w'|$, and $x'' R y'$. Because $\mathcal{A}_1$ is branching-free and its accepting states are stuck states, $x_n \not\rightarrow_1$ holds and there exists $k \in \{1, \dots, n\}$ such that $x'' = x_k$ and $w = a_2 \cdots a_k$. Therefore, the conditions (1), (2) and (3-i) of ***Step***$^\infty$ holds.

C.3 Proof of Prop. 2

Proof. The proof is straightforward by definitions (i.e. Def. 1 and Def. 4).

C.4 Proof of Prop. 7

Proof. It suffices to prove the following:

$$\forall \varphi : \text{generated by the BNF in Prop. 7.}\ \forall w \in \Sigma^*.\ \forall w' \in \Sigma^*.\\ w \subseteq^* w' \wedge w' \in [\![\varphi]\!]_{\mathsf{fin}} \implies w \in [\![\varphi]\!]_{\mathsf{fin}}\,. \quad (4)$$

We prove it by the induction on the structure of φ. Without loss of generality, we can assume $|w| = |w'|$. Let $w = a_1, \ldots, a_n$ and $w' = a'_1 \ldots a'_n$.

(When $\varphi = \mathsf{tt}$) We always have $a_1 \ldots a_n \in [\![\varphi]\!]_{\mathsf{fin}}$.
(When $\varphi = \mathsf{ff}$) It contradicts to $a'_1 \ldots a'_n \in [\![\varphi]\!]_{\mathsf{fin}}$.
(When $\varphi = \neg p$) As $p \notin a'_1$, we have $p \notin a_1$.
(When $\varphi = \varphi_1 \vee \varphi_2$) By $a'_1 \ldots a'_n \in [\![\varphi_1]\!]_{\mathsf{fin}} \cup [\![\varphi_2]\!]_{\mathsf{fin}}$ and the induction hypothesis, we have $a_1 \ldots a_n \in [\![\varphi_1]\!]_{\mathsf{fin}} \cup [\![\varphi_2]\!]_{\mathsf{fin}}$.
(When $\varphi = \varphi_1 \wedge \varphi_2$) Similar to the above.
(When $\varphi = \mathsf{X}\varphi'$) By $a'_2 \ldots a'_n \in [\![\varphi']\!]$ and the induction hypothesis, we have $a_2 \ldots a_n \in [\![\varphi']\!]$.
(When $\varphi = \varphi_1\ \mathsf{U}\ \varphi_2$) By $\exists i \in \{1, \ldots, n\}.\ \forall j < i.\ a'_j \ldots a'_n \in [\![\varphi_1]\!]_{\mathsf{fin}}$ and $a'_i \ldots a'_n \in [\![\varphi_2]\!]_{\mathsf{fin}}$ and the induction hypothesis, we have $\exists i \in \{1, \ldots, n\}.\ \forall j < i.\ a_j \ldots a_n \in [\![\varphi_1]\!]_{\mathsf{fin}}$ and $a_i \ldots a_n \in [\![\varphi_2]\!]_{\mathsf{fin}}$.
(When $\varphi = \varphi_1\ \mathsf{R}\ \varphi_2$) Similar to the above.

C.5 Proof of Prop. 8

Proof. It suffices to prove the following two.

$$\forall \varphi : \text{generated by the BNF in Prop. 8.}\\ \forall w = a_1 \ldots a_n \in \Sigma^*.\ \forall w' = a'_1 \ldots a'_{n'} \in \Sigma^*.\\ w \subseteq_{\mathsf{sub}} w' \wedge w' \in [\![\varphi]\!]_{\mathsf{fin}} \implies w \in [\![\varphi]\!]_{\mathsf{fin}} \quad (5)$$

$$\forall \psi : \text{generated by the BNF in Prop. 8.}\\ \forall w = a_1 \ldots a_n \in \Sigma^+.\ \forall w' = a'_1 \ldots a'_{n'} \in \Sigma^+.\\ w \subseteq_{\mathsf{sub}} w' \wedge w' \in [\![\varphi]\!]_{\mathsf{fin}} \wedge a_1 = a'_1 \implies w \in [\![\varphi]\!]_{\mathsf{fin}} \quad (6)$$

We prove it by the mutual induction on the structures of φ and ψ.

(When $\varphi = \mathsf{tt}$, ff, $\varphi_1 \vee \varphi_2$ or $\varphi_1 \wedge \varphi_2$) Similar to the proof of Prop. 7.
(When $\varphi = \mathsf{G}\psi$) In this case, we have $a'_i \ldots a'_{n'} \in [\![\psi]\!]_{\mathsf{fin}}$ for each $i \in \{1, \ldots, n'\}$. By $a_1 \ldots a_n \subseteq_{\mathsf{sub}} a'_1 \ldots a'_{n'}$ and the induction hypothesis, we have $a_j \ldots a_n \in [\![\psi]\!]_{\mathsf{fin}}$ for each $j \in \{1, \ldots, n\}$.
(When $\psi = p$ or $\neg p$) Immediate by $a_1 = a'_1$.
(When $\psi = \psi_1 \vee \psi_2$ or $\psi_1 \wedge \psi_2$) Similar to the proof of Prop. 7.
(When $\psi = \varphi$) Immediate.

C.6 Proof of Lem. 3

Proof. The parameters M and N constrain Simulator's moves only. The bigger M or N is, the more moves Simulator can make, and hence the more positions Simulator can be winning from.

C.7 Proof of Prop. 3

Proof. We first prove the $\Longleftarrow$ direction, i.e. the "if" part, assuming the existence of an $(M, \mathbf{Q})$-simulation R such that xRy. We fix Simulator's strategy so that Simulator chooses ④ and moves to (ε, x', y') if that is possible and $x'Ry'$, and chooses ③ or ⑤ otherwise. For an arbitrary strategy of Challenger, our goal is to show that Simulator is winning from (ε, x, y). Specifically, we prove the following $(\star)$: from each position (ε, x_0, y_0) such that $x_0 R y_0$, either another position $(\varepsilon, x_0', y_0')$ such that $x_0' R y_0'$ is reachable or Simulator wins.

We let the players play the game $\mathcal{G}^{M,\infty,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ from (ε, x_0, y_0), and pause the play when Challenger makes a move ② or Simulator makes a move ④.

- If the play never ceases, Challenger keeps choosing ① and Simulator only chooses ③.
  - If Challenger gets stuck before making M moves, Simulator wins the play.
  - Otherwise, Challenger can make M moves of ①. After the M moves of Challenger, the position is necessarily $(a_1 \cdots a_M, x_M, y_0)$ such that $x_0 \overset{a_1 \cdots a_M}{\rightsquigarrow\!\!\!\!\rightarrow}_1 x_M$, since Simulator has only chosen ③. Because R is an $(M, \mathbf{Q})$-simulation, by $\textbf{\textit{Step}}^M$, there exist $k \in \{1, \ldots, M\}$, $w' \in \Sigma^*$ and $y_0' \in X_2$ such that $a_1 \cdots a_k \mathbf{Q} w'$, $x_0 \overset{a_1 \cdots a_k}{\rightsquigarrow\!\!\!\!\rightarrow}_1 x_k \overset{a_{k+1} \cdots a_M}{\rightsquigarrow\!\!\!\!\rightarrow}_1 x_M$, and $x_k R y_0'$.
    - If $k < M$, Simulator should have chosen ④ after the k-th move of Challenger, according to the strategy. This leads to a contradiction.
    - Otherwise, $k = M$. This means that after the M-th move of Challenger, Simulator can (and has to, according to the strategy) choose ④. This also leads to a contradiction.
- If the play ceases with Challenger choosing ②, after the first choice of ②, we are at a position $(\checkmark, w, x_0', y_0)$ such that $x_0 \overset{w}{\rightsquigarrow\!\!\!\!\rightarrow}_1 x_0' \in F_1$ and $|w| < M$. We have $|w| < M$ because Simulator has only made moves ③, if any. Since R is an $(M, \mathbf{Q})$-simulation, by $\textbf{\textit{Final}}^M$, there exist $w' \in \Sigma^*$ and $y_0' \in X_2$ such that $w \mathbf{Q} w'$ and $y_0 \overset{w'}{\rightsquigarrow\!\!\!\!\rightarrow}_2 y_0' \in F_2$. Therefore Simulator can choose ⑤ and win.
- If the play ceases with Simulator choosing ④, the first choice of ④ changes a position (w, x_0', y_0) to $(\varepsilon, x_0', y_0')$ for some $y_0' \in X_2$ such that $x_0' R y_0'$.

As a result, $(\star)$ holds. Consequently, from (ε, x, y), Simulator can either win or infinitely continue a play (and win).

We next prove the $\Longrightarrow$ direction, i.e. the "only if" part, assuming that Simulator is winning from the position (ε, x, y). We define $R \subseteq X_1 \times X_2$ by $R := \{(x', y') \mid \text{Simulator is winning from } (\varepsilon, x', y')\}$. It holds that xRy. For an arbitrary pair $(x', y') \in R$, we prove that it satisfies $\textbf{\textit{Final}}^M$ and $\textbf{\textit{Step}}^M$.

Final^M We assume a sequence $x' \overset{w}{\rightsquigarrow}_1 x'' \in F_1$ such that $|w| < M$.

- If $|w| = 0$, we have $x' = x'' \in F_1$. From a position (ε, x', y'), we let Challenger choose ② and move to $(\checkmark, \varepsilon, x', y')$. Simulator is necessarily winning from this position, and hence Simulator can choose ⑤ at $(\checkmark, \varepsilon, x', y')$. This means that there exist $w' \in \Sigma^*$ and $y'' \in X_2$ such that $y' \overset{w'}{\rightsquigarrow}_2 y'' \in F_2$ and $\varepsilon \mathbf{Q} w'$. Therefore (x', y') satisfies ***Final***^{M}.
- Otherwise, i.e. if $|w| > 0$, from a position (ε, x', y'), we let Challenger choose ① for $|w|$ times along $x' \overset{w}{\rightsquigarrow}_1 x''$ and then choose ②. Because Simulator is winning from (ε, x', y'), Simulator has a winning strategy for the $|w| + 1$ moves of Challenger.
  - If the first $|w|$ moves of Simulator according to the strategy are ③, after Challenger's $|w|+1$ moves, we are at a position $(\checkmark, w, x'', y')$. Simulator is winning from (ε, x', y') and necessarily from $(\checkmark, w, x'', y')$. Therefore, the strategy must let Simulator choose ⑤ at the position $(\checkmark, w, x'', y')$.
  - Otherwise, i.e. if the first $|w|$ moves of Simulator according to the strategy are ③ interleaved with ④, after Challenger's $|w|+1$ moves, we are at a position $(\checkmark, w_2, x'', y'')$ for some $w_2 \in \Sigma^*$ and $y'' \in X_2$ such that: there exist $w_1, w'' \in \Sigma^*$ such that $x' \overset{w_1}{\rightsquigarrow}_1 \overset{w_2}{\rightsquigarrow}_1 x'' \in F_1$, $y' \overset{w''}{\rightsquigarrow}_2 y''$ and $w_1 \mathbf{Q} w''$. Simulator is necessarily winning from the position $(\checkmark, w_2, x'', y'')$. Therefore, the strategy must let Simulator choose ⑤ at this position.

  As a result, (x', y') satisfies ***Final***^{M}. Note that $\mathbf{Q}$ is closed under concatenation.

Step^M We assume a sequence $x' \overset{a_1 \cdots a_M}{\rightsquigarrow}_1 x_M$. We let Challenger choose ① for M times. Because Simulator is winning from (ε, x', y'), Simulator has a winning strategy for the M moves of Challenger.

- If Simulator's first move is ④, the move changes a position (a_1, x_1, y') to (ε, x_1, y'') for some $x_1 \in X_1$ and $y'' \in X_2$. Simulator must be winning from the position (ε, x_1, y''), which means $x_1 R y''$.
- Otherwise, i.e. if the first $0 < k < M$ moves of Simulator are ③ and Simulator's next move is ④ according to the strategy, the move ④ changes a position $(a_1 \cdots a_k, x_k, y')$ to (ε, x_k, y'') for some $x_k \in X_1$ and $y'' \in X_2$.

  There exists $w' \in \Sigma^*$ and we have $a_1 \cdots a_k \mathbf{Q} w'$ and $y' \overset{w'}{\rightsquigarrow}_2 y''$. Simulator must be winning from the position (ε, x_k, y''), which means $x_k R y''$.

As a result, (x', y') satisfies ***Step***^{M}.

Consequently, R is an $(M, \mathbf{Q})$-simulation.

C.8 Proof of Prop. 6

Proof. We first prove the $\Longleftarrow$ direction, i.e. the "if" part, assuming the existence of a $\mathbf{Q}$-simulation R such that xRy. We fix Simulator's strategy so that Simulator chooses ④ and moves to (ε, x', y') if that is possible and $x'Ry'$, and chooses ③ or ⑤ otherwise. For an arbitrary strategy of Challenger, our goal is to show that

Simulator is winning from (ε, x, y). Specifically, we prove the following $(\star)$: from each position (ε, x_0, y_0) such that $x_0 R y_0$, either another position $(\varepsilon, x_0', y_0')$ such that $x_0' R y_0'$ is reachable or Simulator wins.

We let the players play the game $\mathcal{G}^{\infty,\infty,\mathbf{Q}}_{\mathcal{A}_1,\mathcal{A}_2}$ from (ε, x_0, y_0), and pause the play when Challenger makes a move ② or Simulator makes a move ④.

- If the play never ceases, Challenger keeps choosing ① and Simulator only chooses ③.
  - If Challenger gets stuck, Simulator wins the play.
  - Otherwise, Simulator can win the infinite play by keep choosing ③.
- If the play ceases with Challenger choosing ②, after the first choice of ②, we are at a position $(\checkmark, w, x_0', y_0)$ such that $x_0 \overset{w}{\rightsquigarrow}_1 x_0' \in F_1$.
  - If $|w| = 0$, i.e. $w = \varepsilon$, we have $x_0 = x_0' \in F_1$. By $\textbf{\textit{Final}}^1$, there exist $w' \in \Sigma^*$ and $y_0' \in X_2$ such that $\varepsilon \mathbf{Q} w'$ and $y_0 \overset{w'}{\rightsquigarrow}_2 y_0' \in F_2$. This means that Simulator can choose ⑤ and win.
  - Otherwise, $|w| > 0$. Because Simulator has only chosen ③, by $\textbf{\textit{Step}}^\infty$, there exist $w' \in \Sigma^*$ and $y_0' \in X_2$ such that $w \mathbf{Q} w'$ and $y_0 \overset{w'}{\rightsquigarrow}_2 y_0' \in F_2$. This means that Simulator can choose ⑤ and win.
- If the play ceases with Simulator choosing ④, the first choice of ④ changes a position (w, x_0', y_0) to $(\varepsilon, x_0', y_0')$ for some $y_0' \in X_2$ such that $x_0' R y_0'$.

As a result, $(\star)$ holds. Consequently, from (ε, x, y), Simulator can either win or infinitely continue a play (and win).

We next prove the $\Longrightarrow$ direction, i.e. the "only if" part, assuming that Simulator is winning from the position (ε, x, y). We define $R \subseteq X_1 \times X_2$ by $R := \{(x', y') \mid \text{Simulator is winning from } (\varepsilon, x', y')\}$. It holds that xRy. For an arbitrary pair $(x', y') \in R$, we prove that it satisfies $\textbf{\textit{Final}}^1$ and $\textbf{\textit{Step}}^\infty$.

$\textbf{\textit{Final}}^1$ We assume that $x' \in F_1$. From a position (ε, x', y'), we let Challenger choose ② and move to $(\checkmark, \varepsilon, x', y')$. Simulator is necessarily winning from this position, and hence Simulator can choose ⑤ at $(\checkmark, \varepsilon, x', y')$. This means that there exist $w' \in \Sigma^*$ and $y'' \in X_2$ such that $y' \overset{w'}{\rightsquigarrow}_2 y'' \in F_2$ and $\varepsilon \mathbf{Q} w'$. Therefore (x', y') satisfies $\textbf{\textit{Final}}^1$.

$\textbf{\textit{Step}}^\infty$ We assume a sequence $x' \overset{a_1 \cdots a_n}{\rightsquigarrow}_1 x_n \in F_1$. We let Challenger choose ① for n times, and then choose ②, moving to a position $(\checkmark, a_1 \cdots a_n, x_n, y')$. Simulator must be winning from this position, which means that Simulator can choose ⑤. This means that there exist $y'' \in F_2$ and $w' \in \Sigma^*$ such that $a_1 \cdots a_n \mathbf{Q} w'$ and $y' \overset{w'}{\rightsquigarrow}_2 y''$. Therefore (x', y') satisfies $\textbf{\textit{Step}}^\infty$.

Consequently, R is a $\mathbf{Q}$-simulation.