Logical Methods in Computer Science
Volume 21, Issue 1, 2025, pp. 13:1–13:59
https://lmcs.episciences.org/

Submitted May 14, 2020
Published Feb. 06, 2025

BISIMILARITY IN FRESH-REGISTER AUTOMATA

ANDRZEJ S. MURAWSKI [a], STEVEN J. RAMSAY [b], AND NIKOS TZEVELEKOS [c]

[a] University of Oxford, UK

[b] University of Bristol, UK

[c] Queen Mary University of London, UK

Abstract. Register automata are a basic model of computation over infinite alphabets. Fresh-register automata extend register automata with the capability to generate fresh symbols in order to model computational scenarios involving name creation. This paper investigates the complexity of the bisimilarity problem for classes of register and fresh-register automata. We examine all main disciplines that have appeared in the literature: general register assignments; assignments where duplicate register values are disallowed; and assignments without duplicates in which registers cannot be empty. In the general case, we show that the problem is EXPTIME-complete.

However, the absence of duplicate values in registers enables us to identify inherent symmetries inside the associated bisimulation relations, which can be used to establish a polynomial bound on the depth of Attacker-winning strategies. Furthermore, they enable a highly succinct representation of the corresponding bisimulations. By exploiting results from group theory and computational group theory, we can then show membership in PSPACE and NP respectively for the latter two register disciplines. In each case, we find that freshness does not affect the complexity class of the problem.

The results allow us to close a complexity gap for language equivalence of deterministic register automata. We show that deterministic language inequivalence for the no-duplicates fragment is NP-complete, which disproves an old conjecture of Sakamoto.

Finally, we discover that, unlike in the finite-alphabet case, the addition of pushdown store makes bisimilarity undecidable, even in the case of visibly pushdown storage.

1. Introduction

Register automata are one of the simplest models of computation over infinite alphabets. They consist of finite-state control and finitely many registers for storing elements from the infinite alphabet. Since their introduction by Kaminski and Francez [KF94] as a candidate

Key words and phrases: Register automata, bisimilarity, computational group theory, automata over infinite alphabets.

This is a revised and extended version of a paper that appeared in LICS'15 [MRT15].

This research was funded in whole or in part by the UK Engineering and Physical Sciences Research Council (EP/J019577/1, EP/L022478/1) and the Royal Academy of Engineering (RF 10216/111). For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

DOI:10.46298/LMCS-21(1:13)2025

formalism for capturing regularity in the infinite-alphabet setting, they have been actively researched especially in the database and verification communities: selected applications include the study of markup languages [NSV04] and run-time verification [GDPT13]. While register automata can detect symbols that are currently not stored in registers (local freshness), the bounded number of registers means that they are not in general capable of recognising inputs that are genuinely fresh in the sense that they occur in the computation for the first time (global freshness). Because such a feature is desirable in many contexts, notably dynamic resource allocation, the formalism has been extended in [Tze11] to fresh-register automata, which do account for global freshness. This paper is concerned with the problem of *bisimilarity testing* for register and fresh-register automata.

Bisimulation is a fundamental notion of equivalence in computer science. Its central role is, in part, derived from the fact that it is intensional and yet very robust. Consequently, the algorithmics of bisimilarity have attracted a lot of attention from researchers interested in the theory and practice of equivalence checking. When the set of observable actions available to a system is finite, a lot is already known about the complexity of the problem for specific classes of systems, although tight bounds are often difficult to obtain in the infinite-state cases [Srb08]. In this paper we prove a number of bounds on the complexity of bisimulation equivalence checking. We note that in this setting language equivalence is known to be undecidable [NSV04].

Our results are expressed using a unified framework that comprises all variations that have appeared in the literature. They differ in the allowed register assignment discipline, which turns out to affect complexity. Assignments are allowed to be

(S)**:** *single*, if the contents of all registers are required to be distinct; or
(M)**:** *multiple*, if we allow for duplicate values.

Furthermore, registers are required to

(F)**:** always be filled; or
$(\#_0)$**:** initially allowed to be empty; or
$(\#)$**:** allowed to be erased and filled during a run[1].

The complexity of bisimilarity checking for each combination is summarised in the table below, where we use the suffix "-c" to denote completeness for this class and "-s" to denote solvability only. The results hold regardless of whether one considers register or fresh-register automata.

$(M\#)$	$(M\#_0)$	(MF)	$(S\#)$	$(S\#_0)$	(SF)
EXP-c	EXP-c	EXP-c	EXP-c	PSPACE-c	NP-s

Our work thus provides a practical motivation for modelling systems with single assignment whenever possible — if the system does not need to erase the contents of registers mid-run, the corresponding equivalence problems are lower in the complexity hierarchy.

We start by giving coarse, exponential-time upper bounds for all the classes of system considered by showing how any such bisimilarity problem can be reduced to one for finite-state automata at exponential cost. For all the multiple assignment machines this bound is tight and, for single assignment, tightness depends upon whether or not erasing is allowed. The implied significance of being able to erase the contents of registers is explained by our proof that the bisimulation games associated with such systems can simulate the computations of

[1]Empty content is "#". A full definition of each of the automaton variants is given in Section 2.

alternating Turing machines running in polynomial space. Here we set up an encoding of the tape, determined by the presence or absence of content in certain registers, and erasing of registers corresponds to writing of tape cells.

Once erasure is forbidden under single assignments, we obtain better bounds by investigating the structure of the associated bisimulation relations. Such relations are generally infinite, but only the relationship between the register assignments in two configurations is relevant to bisimilarity, and so we work with a finite, though exponentially large, class of symbolic relations built over partial permutations (to link register indices). Due to the inherent symmetry and transitivity of bisimilarity, each such relation forms an inverse semigroup under function composition. Also, crucially, the relations are upward closed in the information order. Although, taken separately, neither of the preceding facts leads to an exponential leap in succinctness of representation, taken together they reveal an interconnected system of (total) permutation groups underlying each relation. What is more, in any play of the associated bisimulation game, the number of registers that are empty must monotonically decrease. This, together with an application of Babai's result on the length of subgroup chains in symmetric groups [Bab86], allows us to show that any violation of bisimilarity can be detected after polynomially many rounds of the bisimulation game. Consequently, in this case, we are able to decide bisimilarity in polynomial space.

From a conceptual point of view, the use of group theory helps us capture symmetries in bisimulation relations, express them in a succinct and structured way and manipulate them effectively. We regard the use of group-theoretic techniques in this context to be the technical highlight of the paper, and hope that it will inspire further fruitful interplay between automata over infinite alphabets and computational group theory.

The polynomial bound mentioned above enables us to close a complexity gap (between NP and PSPACE) in the study of deterministic language equivalence. Namely, we show that the language inequivalence problem for *deterministic* $\mathrm{RA}(S\#_0)$ is in NP, and thus NP-complete, refuting a conjecture by Sakamoto [Sak98].

Further, if registers are additionally required to be filled (SF), we can exhibit very compact representations of the relevant bisimulation relations. The fact that permutation groups have small generating sets [MN87] allows us then to design a representation for symbolic bisimulations that is at most polynomial in size. Furthermore, by exploiting polynomial-time membership testing for permutation groups given in terms of their generators [FHL80], we show that such a representation can be guessed and verified by a nondeterministic Turing machine in polynomial time.

Finally, we consider bisimilarity for visibly pushdown register automata (VPDRA) under the SF register discipline, and we show that the problem here is already undecidable. Since VPDRA(SF) are a particularly weak variant, this result implies undecidability for all PDRA considered in [MRT14]. In contrast, for finite alphabets, (strong) bisimilarity of pushdown automata is known to be decidable [Sén05] but non-elementary [BGKM13], with ACKERMANN being the best upper bound [JS19]. In the visibly pushdown case, the problem is EXPTIME-complete [Srb06].

Related Work. The complexity of bisimilarity problems has been studied extensively in the finite-alphabet setting and the current state of the art for infinite-state systems is summarised nicely in [Srb08]. Recent papers concerning the complexity of decision problems for register automata have, until now, not considered bisimulation equivalence. However, there are several related complexity results in the concurrency literature.

In his PhD thesis, Pistore [Pis99], gives an exponential-time algorithm for bisimilarity of HD-automata [MP97]. Since Pistore shows that bisimulation relations for HD-automata have many of the algebraic properties[2] as the relations we study here, it seems likely that our algorithm could be adapted to show that the bisimilarity problem for HD-automata is in NP. Indeed, a compact representation of symmetries using generators for such a purpose was envisaged by [CM10].

Jonsson and Parrow [JP93] and Boreale and Trevisan [BT00] consider bisimilarity over a class of data-independent processes. These processes are terms built over an infinite alphabet, but the behaviour of such a process does not depend upon the data from which it is built. In the latter work, the authors also consider a class of value-passing processes, whose behaviour may depend upon the result of comparing data for equality. They show that if such processes can be defined recursively then the problem is EXPTIME-complete. Since value passing can be seen as a purely functional proxy for multiple register assignments, this result neatly reflects our findings for RA($M\#$). Finally, decidability of bisimilarity for FRA($S\#_0$) was proven in [Tze11], albeit without a proper study of its complexity (the procedure given in *loc. cit.* can be shown to run in nondeterministic exponential time).

Finally, in a recent follow-up paper [MRT18], we showed that the language equivalence problem for deterministic RA(SF) is in P, in contrast to NP-completeness for RA($S\#_0$), established in the present paper. For RA(SF), this still leaves a complexity gap between NL and P.

It would be interesting to see to what extent our decidability and complexity results can be generalised, e.g. in settings with ordered infinite alphabets or nominal automata [BKL14].

Structure. In Section 2 we introduce the preliminaries and prove all of the EXPTIME bounds in Section 3. Then we start the presentation of other results with register automata, as the addition of global freshness requires non-trivial modifications. In Section 4 we show bounds for the ($S\#_0$) problems and apply the techniques to deterministic language equivalence in Section 5. Section 6 covers further improvements for the (SF) case. In Section 7 we generalise our techniques to fresh-register automata and, finally, consider the pushdown case in Section 8.

2. Preliminaries

We introduce some basic notation. Given a relation $R \subseteq X \times Y$, we define $\mathsf{dom}(R) = \{x \in X \mid \exists y.(x,y) \in R\}$ and $\mathsf{rng}(R) = \{y \in Y \mid \exists x.(x,y) \in R\}$. For natural numbers $i \leq j$, we write $[i,j]$ for the set $\{i, i+1, \ldots, j\}$. $\mathcal{P}(X)$ stands for the powerset of X.

2.1. Bisimilarity.

We define bisimulations generally with respect to a labelled transition system. As we shall see, the particular systems that we will be concerned with in this paper are the configuration graphs of various classes of (fresh-) register automata.

Definition 2.1. A ***labelled transition system*** (LTS) is a tuple $\mathcal{S} = (\mathbb{C}, \mathcal{A}ct, \rightarrow)$, where $\mathbb{C}$ is a set of *configurations*, $\mathcal{A}ct$ is a set of *action labels*, and $\rightarrow \subseteq \mathbb{C} \times \mathcal{A}ct \times \mathbb{C}$ is a *transition relation*. For $\ell \in \mathcal{A}ct$, we use $\xrightarrow{\ell}$ to refer to $\rightarrow \cap (\mathbb{C} \times \{\ell\} \times \mathbb{C})$.

[2] E.g. the *active names* of [Pis99] are comparable to our *characteristic sets*.

A binary relation $R \subseteq \mathbb{C} \times \mathbb{C}$ is a ***bisimulation*** if for each $(\kappa_1, \kappa_2) \in R$ and each $\ell \in \mathcal{A}ct$, we have:

(1) if $\kappa_1 \xrightarrow{\ell} \kappa_1'$, then there is some $\kappa_2 \xrightarrow{\ell} \kappa_2'$ with $(\kappa_1', \kappa_2') \in R$;
(2) if $\kappa_2 \xrightarrow{\ell} \kappa_2'$, then there is some $\kappa_1 \xrightarrow{\ell} \kappa_1'$ with $(\kappa_1', \kappa_2') \in R$.

We say that κ_1 and κ_2 are ***bisimilar***, written $\kappa_1 \sim \kappa_2$, just if there is some bisimulation R with $(\kappa_1, \kappa_2) \in R$.

Let us recall that bisimilarity has a very natural game-theoretic account. Given two configurations, one can consider a *bisimulation game* involving two players, traditionally called *Attacker* and *Defender* respectively. They play rounds in which Attacker fires a transition from one of the configurations and Defender has to follow with an identically labelled transition from the other configuration. In the first round, the chosen transitions must lead from the configurations to be tested for bisimilarity, while, in each subsequent round, they must start at the configurations reached after the preceding round. Defender loses if he cannot find a matching transition. In this framework, bisimilarity corresponds to the existence of a winning strategy for Defender. The process of playing a bisimulation game naturally favours Attacker as the decision maker but, thanks to the forcing technique of [JS08], it is possible to construct transition systems in which Defender effectively ends up making choices.

2.2. **Fresh-register automata.** We will be interested in testing bisimilarity of configurations generated by machines with registers and pushdown stack in the infinite-alphabet setting, i.e. as $\mathcal{A}ct$ we shall use the set $\Sigma \times \mathcal{D}$ for a finite alphabet Σ (with its elements sometimes called *tags*) and an infinite alphabet $\mathcal{D}$ (with its elements sometimes called *names*), *cf.* data words [NSV04].

Definition 2.2. An r-***fresh-register automaton*** (r-FRA) is a tuple $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$, where:

- Q is a finite set of states;
- Σ is a finite set of tags;
- $\delta \subseteq Q \times \Sigma \times (\mathcal{P}([1,r]) \cup \{\circledast\}) \times [0,r] \times \mathcal{P}([1,r]) \times Q$ is the transition relation, with elements written as $q \xrightarrow{t,X,i,Z} q'$. We assume that in any such transition $i \notin Z$.

Finally an r-***register automaton*** (r-RA) is a special case of an r-FRA such that all its transitions $q \xrightarrow{t,X,i,Z} q'$ satisfy $X \neq \circledast$.

An *r-register assignment* is a mapping of register indices to letters from the infinite alphabet $\mathcal{D}$ and the special symbol $\#$, i.e. a function:

$$\rho : [1,r] \to \mathcal{D} \uplus \{\#\}.$$

The $\#$ symbol is used to represent the fact that a register is empty, i.e. contains no letter from $\mathcal{D}$. Consequently, by slight abuse of notation, for any r-register assignment ρ we will be writing $\mathsf{rng}(\rho)$ for the set $\rho([1,r]) \cap \mathcal{D}$, and $\mathsf{dom}(\rho)$ for $\rho^{-1}(\mathsf{rng}(\rho))$, where $\rho^{-1} = \{(d,i) \mid d \in \mathcal{D} \wedge (i,d) \in \rho\}$. Finally, we shall use two kinds of assignment update. For any $d \in \mathcal{D}, i \in [0,r], Z \subseteq [1,r]$ and assignment ρ we set:

$$\rho[i \mapsto d] = \begin{cases} \{(i,d)\} \cup \{(j, \rho(j)) \mid j \in [1,n] \setminus \{i\}\} & \text{if } i \neq 0 \\ \rho & \text{otherwise,} \end{cases}$$

$$\rho[Z \mapsto \#] = \{(j,\#) \mid j \in Z\} \cup \{(j,\rho(j)) \mid j \in [1,n] \setminus Z\}.$$

Note that, in the former case, no update takes place when $i = 0$ but we keep the update notation for notational convenience.

The meaning of a transition $q \xrightarrow{t,X,i,Z} q'$ is described as follows. The components t and X are a precondition: for the transition to be applicable, it must be that the next letter of the input has shape (t, a) for some $a \in \mathcal{D}$ and, moreover:

- if $X \subseteq [1, r]$ then a is already stored in exactly those registers named by X;
- if $X = \circledast$ then a is (globally) *fresh*: it has so far not appeared in the computation of $\mathcal{A}$.

If the transition applies then taking it results in changes being made to the current register assignment, namely: a is written into register i (unless $i = 0$, in which case it is not written at all) and all registers named by Z have their contents erased.

Definition 2.3. A ***configuration*** κ of an r-FRA $\mathcal{A}$ is a triple (q, ρ, H) consisting of a state $q \in Q$, an r-register assignment ρ and a finite set $H \subseteq \mathcal{D}$, called the *history*, such that $\mathsf{rng}(\rho) \subseteq H$. If $q_1 \xrightarrow{t,X,i,Z} q_2$ is a transition of $\mathcal{A}$, then a configuration (q_1, ρ_1, H_1) can make a transition to a configuration (q_2, ρ_2, H_2) accepting input (t, d), written $(q_1, \rho_1, H_1) \xrightarrow{(t,d)} (q_2, \rho_2, H_2)$, just if:

- $X = \{j \mid \rho_1(j) = d\}$, or $X = \circledast$ and $d \notin H$;
- $\rho_2 = \rho_1[i \mapsto d][Z \mapsto \#]$;
- $H_2 = H_1 \cup \{d\}$.

We will sometimes write the set of configurations of $\mathcal{A}$ by $\mathbb{C}_{\mathcal{A}}$ and the induced transition relation by $\rightarrow_{\mathcal{A}}$. We let $\mathcal{S}(\mathcal{A})$ be the LTS $\langle \mathbb{C}_{\mathcal{A}}, \Sigma \times \mathcal{D}, \rightarrow_{\mathcal{A}} \rangle$.

On the other hand, a configuration κ of an r-RA $\mathcal{A}$ is a pair (q, ρ) of a state $q \in Q$ and an r-register assignment ρ. The LTS $\langle \mathbb{C}_{\mathcal{A}}, \Sigma \times \mathcal{D}, \rightarrow_{\mathcal{A}} \rangle$ is defined precisely as above, albeit excluding histories and fresh transitions. More precisely, if $q_1 \xrightarrow{t,X,i,Z} q_2$ is a transition of $\mathcal{A}$, then $(q_1, \rho_1) \xrightarrow{(t,d)} (q_2, \rho_2)$ just if $X = \{j \mid \rho_1(j) = d\}$ and $\rho_2 = \rho_1[i \mapsto d][Z \mapsto \#]$.

We define several specific classes of fresh-register automata that we will study in this work by considering configurations and transitions restricted according to the register assignment discipline followed.

Duplication in assignment. We consider two register storage policies, namely single assignment (S) or multiple assignment (M). In single assignment, we restrict register assignments to be injective on non-empty registers, i.e. for all $i, j \in [1, r]$, $\rho(i) = \rho(j)$ just if $i = j$ or $\rho(i) = \# = \rho(j)$. In multiple assignment there is no such restriction. To ensure that all configurations respect the register assignment discipline, in the (S) case every transition $q_1 \xrightarrow{t,X,i,Z} q_2$ is required to satisfy the following condition: if $X \subseteq [1, r]$ then $|X| \leq 1$ and if $X \neq \emptyset$ then $i = 0$. This simply corresponds to the fact that $d \in \mathcal{D}$ matches the content of at most one register and, if d is already stored in a register, it will not be written back to any (other) register.

Emptiness of registers. We consider the automaton's ability to process empty registers. We say that either all registers must always be filled (F), that registers may be initially empty ($\#_0$) or that the contents of registers may be erased ($\#$) during a run. Under condition (F), r-register assignments are restricted so that $\# \notin \rho([1, r])$. Under conditions (F) and ($\#_0$), every transition $q_1 \xrightarrow{t,X,i,Z} q_2$ must have $Z = \emptyset$. Condition ($\#$) imposes no specific restrictions.

We describe particular classes by the acronym FRA(XY) in which

$$X \in \{M,\, S\} \text{ and } Y \in \{F,\, \#_0,\, \#\}.$$

The class FRA(XY) refers to specialisations of Definitions 2.2, 2.3 to transitions and register assignments satisfying the constraints imposed by X and Y. For instance, FRA($S\#_0$)-configurations are functions from $[1, r]$ to $\mathcal{D} \cup \{\#\}$ that are injective on non-empty registers, and every transition of such a machine is of the form $q_1 \xrightarrow{t,X,i,Z} q_2$ with $X \in \{\circledast, \emptyset\} \cup \{\{j\} \mid j \in [1, r]\}$ and $Z = \emptyset$ such that $X = \{j\}$ implies $i = 0$. In a similar manner, we define the classes RA(XY).

Remark 2.4. The class RA(MF) follows the register assignment discipline of the register automata defined by Segoufin [Seg06]. The class RA($M\#_0$) follow the register assignment discipline of the M-Automata defined by Kaminski and Francez [KF94] and the class of RA($S\#_0$) follows the assignment discipline of the finite memory automata considered in the same paper. The class RA(SF) contain automata that follow the register assignment discipline of the machines considered by Nevin, Schwentick and Vianu [NSV04]. The class FRA($S\#_0$) follow the register assignment discipline of the automata defined in [Tze11]. We note that the automata from [KF94, NSV04, Tze11] mentioned above are a little more restrictive in that every name encountered by the automaton must be stored in some register, i.e. $i \neq 0$.[3]

In this paper we are concerned with the following family of decision problems.

Definition 2.5. Let $X \in \{M,\, S\}$ and $Y \in \{F,\, \#_0,\, \#\}$.

- The problem $\sim$-FRA(XY) is: given an FRA(XY) $\mathcal{A}$ and configurations $\kappa_1 = (q_1, \rho_1, H)$ and $\kappa_2 = (q_2, \rho_2, H)$, does $\kappa_1 \sim \kappa_2$ hold in $\mathcal{S}(\mathcal{A})$?
- The problem $\sim$-RA(XY) is: given an RA(XY) $\mathcal{A}$ and configurations κ_1 and κ_2 , does $\kappa_1 \sim \kappa_2$ hold in $\mathcal{S}(\mathcal{A})$?

We shall relate the various classes of bisimilarity problems that we study by their complexity. We write $P_1 \leq P_2$ to denote that there is a polynomial-time many-one reduction from problem P_1 to problem P_2.

Lemma 2.6. *The considered bisimilarity problems can be related as in Figure 1.*

Proof. First note that, for all XY, any RA(XY) $\mathcal{A}$ can be trivially seen as an FRA(XY) $\mathcal{A}'$ (i.e. $\mathcal{A}'$ has the same components as $\mathcal{A}$). We claim that, for any pair $(q_1, \rho_1), (q_2, \rho_2)$ of RA-configurations of $\mathcal{A}$,

$$(q_1, \rho_1) \sim (q_2, \rho_2) \iff (q_1, \rho_1, H) \sim (q_2, \rho_2, H) \qquad (*)$$

where $H = \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$ and $(q_1, \rho_1, H), (q_2, \rho_2, H)$ are configurations of $\mathcal{A}'$. Indeed, we can show that the relation between $\mathcal{A}$- and $\mathcal{A}'$-configurations given by:

$$R = \{\ ((q, \rho), (q, \rho, H)) \mid \mathsf{rng}(\rho) \subseteq H\ \}$$

is a bisimulation, from which we obtain $(*)$.

[3] In the conference version of the paper, we added this restriction to the definitions of F and $\#_0$. Also, the definition of S was slightly different therein: we stipulated that $X \subseteq \{i\}$, i.e. we allowed an input letter already present in a register to be unnecessarily overwritten with itself rather than simply preserved (as in the current version). These differences between the conference version and the current one were triggered by reviewers' suggestions and do not affect any of the results.

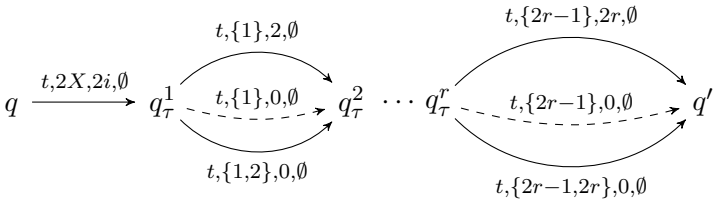
We next show the FRA-bisimilarity reductions; the RA-bisimilarity reductions are shown in a similar (simpler) way.

Observe that, for any $X \in \{S, M\}$, $\sim$-FRA$(XF) \leq \sim$-FRA$(X\#_0) \leq \sim$-FRA$(X\#)$. This is because any FRA(XF) can be viewed trivially as an FRA$(X\#_0)$ in which all registers begin filled and, similarly, any FRA$(X\#_0)$ can be viewed trivially as an FRA$(X\#)$ in which no registers are ever erased.

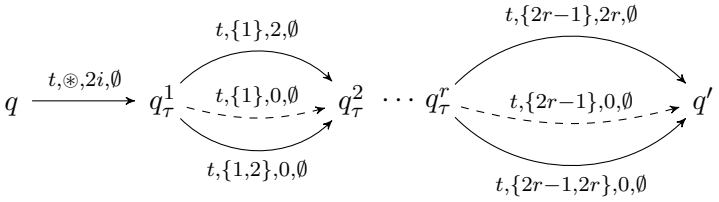
Now, given an r-FRA$(S\#)$ $\mathcal{A}$ and two configurations κ_1 and κ_2 we construct a $2r$-FRA(MF) $\mathcal{A}'$ and configurations $\widehat{\kappa_1}$ and $\widehat{\kappa_2}$ in which every register k of $\mathcal{A}$ is simulated by two registers $2k-1$ and $2k$ of $\mathcal{A}'$. The representation scheme is as follows: if registers $2k-1$ and $2k$ of $\mathcal{A}'$ contain the same letter then register k of $\mathcal{A}$ is empty, otherwise the register k in $\mathcal{A}$ contains exactly the contents of register $2k$ in $\mathcal{A}'$. Additionally, the content of odd-numbered registers in $\widehat{\kappa_1}$, $\widehat{\kappa_2}$ will be the same, which will make it easy to simulate erasures: to simulate the erasure of register k in $\mathcal{A}$ it will suffice to copy the content of register $2k-1$ into $2k$ in $\mathcal{A}'$.

The states of $\mathcal{A}'$ are the states of $\mathcal{A}$ augmented by an additional state q^i_τ for every $q \in Q$, $i \in [1, r]$ and every $\tau \in \delta$. The extra states subscripted with τ will be used to simulate potential erasures caused by τ.

Each transition $\tau = q \xrightarrow{t,X,i,Z} q'$ of $\mathcal{A}$, in which $X \subseteq [1, r]$ and $|X| \leq 1$, is simulated by a sequence of transitions of $\mathcal{A}'$ with the following shape:



where $2X$ is a shorthand for $\{2x \mid x \in X\}$. For each $j \in [1, r]$ the solid (upper) arrow labelled $(t, \{2k-1\}, 2k, \emptyset)$ exists just if $k \in Z$: this transition models erasure of a non-empty register. The dashed arrow labelled $(t, \{2k-1\}, 0, \emptyset)$ exists just if $k \notin Z$: it models lack of erasure for non-empty register k, but we add these transitions so that $\mathcal{A}'$ can behave uniformly regardless of whether erasures are needed or not. The solid (lower) arrow labelled $(t, \{2k-1, 2k\}, 0, \emptyset)$ applies in case register k is empty (we do nothing, regardless of whether $k \in Z$ or not). On the other hand, each transition $\tau = q \xrightarrow{t,\circledast,i,Z} q'$ of $\mathcal{A}$ is simulated by the following sequence of transitions of $\mathcal{A}'$:



where solid and dashed arrows are as above.

We say that a pair of configurations $(q_1, \widehat{\rho_1})$, $(q_2, \widehat{\rho_2})$ of $\mathcal{A}'$ *represents* a pair of configurations (q_1, ρ_1), (q_2, ρ_2) of $\mathcal{A}$ just if $\widehat{\rho_1}$ is a representation of ρ_1 and $\widehat{\rho_2}$ is a representation of ρ_2 as discussed above and, furthermore:

- for all $k \in [1, r]$, $i \in [1, 2r]$, $j \in \{1, 2\}$: if $\widehat{\rho_j}(2k-1) = \widehat{\rho_j}(i)$ then $i \in \{2k-1, 2k\}$

$$\begin{array}{ccccccccccc}
\sim\text{-FRA}(SF) & \leq & \sim\text{-FRA}(S\#_0) & \leq & \sim\text{-FRA}(S\#) & \leq & \sim\text{-FRA}(MF) & \leq & \sim\text{-FRA}(M\#_0) & \leq & \sim\text{-FRA}(M\#) \\
\vee| & & \vee| & & \vee| & & \vee| & & \vee| & & \vee| \\
\sim\text{-RA}(SF) & \leq & \sim\text{-RA}(S\#_0) & \leq & \sim\text{-RA}(S\#) & \leq & \sim\text{-RA}(MF) & \leq & \sim\text{-RA}(M\#_0) & \leq & \sim\text{-RA}(M\#)
\end{array}$$

Figure 1: Relationship between the main bisimilarity problems considered in this work.

- for all $k \in [1, r]$: $\widehat{\rho_1}(2k-1) = \widehat{\rho_2}(2k-1)$

These latter two properties can easily be seen to be an invariant of configurations reachable from any pair that initially satisfy it, since transitions of $\mathcal{A}'$ only write to even numbered registers $2k$ and only with a fresh letter or the contents of the adjacent register $2k-1$.

By construction, the automaton $\mathcal{A}'$ faithfully simulates the original in the following sense, given configurations (q_1, ρ_1), (q_2, ρ_2) of $\mathcal{A}$ and $\mathcal{A}'$ representations $\widehat{\rho_1}$ of ρ_1 and $\widehat{\rho_2}$ of ρ_2: $(q_1, \rho_1) \sim (q_2, \rho_2)$ in $\mathcal{S}(\mathcal{A})$ iff $(q_1, \widehat{\rho_1}) \sim (q_2, \widehat{\rho_2})$ in $\mathcal{S}(\mathcal{A}')$. □

2.3. **Groups and permutations.** Next we introduce notation related to groups and semigroups. Their use will be intstrumental to improving upon our initial EXPTIME bounds. Group-theoretic arguments and computational procedures based on them will be employed in Sections 4, 5, 6 to study register automata, and in Section 7 in the fresh-register case.

For any $S \subseteq [1, n]$, we shall write $\mathcal{S}_S$ for the group of permutations on S, and $\mathcal{IS}_S$ for the inverse semigroup of partial permutations on S. For economy, we write $\mathcal{S}_n$ for $\mathcal{S}_{[1,n]}$; and $\mathcal{IS}_n$ for $\mathcal{IS}_{[1,n]}$. For partial permutations σ and τ, we write $\sigma;\tau$ for their relational composition:

$$\sigma;\tau = \{\, (i,j) \mid \exists k.\sigma(i) = k \wedge \tau(k) = j \,\}.$$

Given $i, j \in [1, n]$, we write $(i\ j)$ for the permutation swapping i and j, that is, $(i\ j) = \{(i,j),(j,i)\} \cup \{(k,k) \in [1,n]^2 \mid k \neq i, j\}$.

2.4. **Update notation.** We shall be applying updates to partial permutations $\sigma \in \mathcal{IS}_n$, by adding new mappings $[i \mapsto j]$ or pre- or post-composing them with swappings $(i\ j)$. For notational convenience it is useful to have $i, j \in [0, n]$, but extra care is needed when $i = 0$ or $j = 0$. Given $\sigma \in \mathcal{IS}_n$ and $i, j \in [0, n]$, we let:

$$\sigma[i \mapsto j] = \begin{cases} \{(i,j)\} \cup \{(i',j') \in \sigma \mid i' \neq i \wedge j' \neq j\} & \text{if } i, j \in [1, n] \\ \{(i',j') \in \sigma \mid j' \neq j\} & \text{if } i = 0 \text{ and } j \neq 0 \\ \{(i',j') \in \sigma \mid i' \neq i\} & \text{if } i \neq 0 \text{ and } j = 0 \\ \sigma & \text{if } i = j = 0 \end{cases}$$

$$\sigma[i \leftrightarrow j] = \begin{cases} (i\ j);\sigma & \text{if } i, j \in [1, n] \\ \sigma & \text{if } i = 0 \text{ or } j = 0 \end{cases}$$

$$[i \leftrightarrow j]\sigma = \begin{cases} \sigma;(i\ j) & \text{if } i, j \in [1, n] \\ \sigma & \text{if } i = 0 \text{ or } j = 0 \end{cases}$$

Similarly, given $S \subseteq [1,n]$ and $i,j \in [0,n]$, we let:

$$S[i \leftrightarrow j] = \begin{cases} \{(i\ j)(k) \mid k \in S\} & \text{if } i,j \in [1,n] \\ S & \text{otherwise} \end{cases} \quad \Big| \quad S[j] = \begin{cases} S \cup \{j\} & \text{if } j \in [1,n] \\ S & \text{otherwise} \end{cases}$$

Lemma 2.7. *Given $\sigma, \tau \in \mathcal{IS}_n$ and $i, j, i_x, i'_x \in [0,n]$ (for $x = 1,2,3$):*

- $\sigma[i \mapsto j]^{-1} = \sigma^{-1}[j \mapsto i]$ *and* $(\sigma[i \leftrightarrow j])^{-1} = [i \leftrightarrow j]\sigma^{-1}$
- $\mathsf{dom}(\sigma[i \leftrightarrow j]) = \mathsf{dom}(\sigma)[i \leftrightarrow j]$ *and* $\mathsf{rng}([i \leftrightarrow j]\sigma) = \mathsf{rng}(\sigma)[i \leftrightarrow j]$
- $([i_2 \leftrightarrow i'_2]\sigma)[i_1 \leftrightarrow i'_1] = [i_2 \leftrightarrow i'_2](\sigma[i_1 \leftrightarrow i'_1])$
- $([i_2 \leftrightarrow i'_2]\sigma[i_1 \leftrightarrow i'_1]); ([i_3 \leftrightarrow i'_3]\tau[i_2 \leftrightarrow i'_2]) = [i_3 \leftrightarrow i'_3](\sigma;\tau)[i_1 \leftrightarrow i'_1]$
- $(\sigma[i_1 \mapsto i_2]); (\tau[i_2 \mapsto i_3]) \subseteq (\sigma;\tau)[i_1 \mapsto i_3]$.

Proof. We only look at the last claim and leave the remaining ones as exercises. Given a partial permutation π on an arbitrary finite set X, and $x, y \in X$, let us write:

$$\pi\langle x \mapsto y\rangle = \{(x,y)\} \cup \{(x',y') \mid x \neq x' \wedge x \neq x'\}.$$

Given π, π' and $x, y, z \in X$, we can show that

$$(\pi\langle x \mapsto y\rangle); (\pi'\langle y \mapsto z\rangle) \subseteq (\pi;\pi')\langle x \mapsto z\rangle. \tag{2.1}$$

Back to the claim, for any $\sigma \in \mathcal{IS}_n$ and $i,j \in [0,n]$, setting $\hat{\sigma} = \sigma \cup \{(0,0)\}$ and viewing it as a partial permutation on $[0,n]$, we have that $\sigma[i \mapsto j] = (\hat{\sigma}\langle i \mapsto j\rangle) \cap [1,n]^2$. Hence:

$$\begin{aligned} (\sigma[i_1 \mapsto i_2]); (\tau[i_2 \mapsto i_3]) &= (\hat{\sigma}\langle i_1 \mapsto i_2\rangle \cap [1,n]^2); (\hat{\tau}\langle i_2 \mapsto i_3\rangle \cap [1,n]^2) \\ &\subseteq (\hat{\sigma}\langle i_1 \mapsto i_2\rangle; \hat{\tau}\langle i_2 \mapsto i_3\rangle) \cap [1,n]^2 \\ &\subseteq (\hat{\sigma};\hat{\tau})\langle i_1 \mapsto i_3\rangle \cap [1,n]^2 \quad \text{by (2.1)} \end{aligned}$$

and the latter is $(\sigma;\tau)[i_1 \mapsto i_3]$, as required. □

3. Bisimilarity problems complete for EXPTIME

In this section we show that the upper four classes in our two hierachies of automata all have bisimilarity problems that are complete for exponential time.

Theorem 3.1. *All of the problems $\sim$-RA(S#), $\sim$-RA(MF), $\sim$-RA(M#$_0$), $\sim$-RA(M#), $\sim$-FRA(S#), $\sim$-FRA(MF), $\sim$-FRA(M#$_0$) and $\sim$-FRA(M#) are EXPTIME-complete.*

Proof. The result follows immediately from Propositions 3.4 and 3.9 and Lemma 2.6. □

Our argument proceeds by showing that $\sim$-FRA($M\#$) is in EXPTIME (Proposition 3.4) and $\sim$-RA($S\#$) is already EXPTIME-hard (Proposition 3.9). In the latter case, we shall rely on alternating linear bounded automata, whose acceptance problem is known to be EXPTIME-complete [CKS81].

Definition 3.2. An ***alternating linear bounded automaton*** (ALBA) is a tuple

$$\mathcal{A} = \langle \Gamma, Q_\forall, Q_\exists, q_0, q_{\mathrm{acc}}, q_{\mathrm{rej}}, \delta\rangle.$$

We let $Q = Q_\forall \uplus Q_\exists \uplus \{q_{\mathrm{acc}}\} \uplus \{q_{\mathrm{rej}}\}$ and call it the set of states, assuming the four constituent subsets are pairwise disjoint. The components are:

- a finite tape alphabet Γ containing end-of-tape markers $\triangleleft$ and $\triangleright$;
- disjoint finite sets of universal states $Q_\forall$ and existential states $Q_\exists$;

- distinguished initial state $q_0 \in Q$;
- distinct accepting and rejecting states $q_{\text{acc}} \neq q_{\text{rej}}$;
- a transition function $\delta : (Q \setminus \{q_{\text{acc}}, q_{\text{rej}}\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{-1, +1\})$, satisfying the following properties:
  - (i) if $(q', a, z) \in \delta(q, \triangleright)$ then $a = \triangleright$ and $z = +1$;
  - (ii) if $(q', a, z) \in \delta(q, \triangleleft)$ then $a = \triangleleft$ and $z = -1$;
  - (iii) if $(q', a, z) \in \delta(q, b)$ then $b \in \Gamma \setminus \{\triangleleft, \triangleright\}$ implies $a \in \Gamma \setminus \{\triangleleft, \triangleright\}$.

A ***configuration*** of such a machine is a triple $c = (q, k, t)$ with q a state, t the current tape contents and $k \geq 0$ the index of the cell currently under the head of the machine. We assume that the tape contents are of the form

$$\triangleright\, a_1 \cdots a_n \,\triangleleft$$

for some letters $a_i \in \Gamma \setminus \{\triangleleft, \triangleright\}$. We write $t(k)$ for the content of cell k of tape t. We say that a configuration (q, k, t) is ***accepting*** (respectively ***rejecting***, ***universal***, ***existential***) just if $q = q_{\text{acc}}$ (respectively $q = q_{\text{rej}}$, $q \in Q_\forall$, $q \in Q_\exists$).

A configuration (q_1, k_1, t_1) can make a transition to a ***successor*** (q_2, k_2, t_2) just if there is $a \in \Gamma$ and $z \in \{-1, +1\}$ such that $(q_2, a, z) \in \delta(q_1, t_1(k_1))$ and $k_2 = k_1 + z$ and $t_2 = t_1[k_1 \mapsto a]$.

Given an input $w \in \Gamma \setminus \{\triangleleft, \triangleright\}$, a ***computation tree on*** w for such a machine is an unordered tree labelled by configurations which additionally satisfies the following conditions:

- The tree is rooted at $(q_0, 0, \triangleright w \triangleleft)$.
- If a universal configuration c labels some node of the tree then this node has one child for each possible successor to c.
- If an existential configuration c labels some node of the tree then this node has exactly one child which can be any successor to c.

A computation tree is *accepting* if it is finite and all of its leaves are accepting. We say that an input w is ***accepted*** just if there is an accepting computation tree on w.

Definition 3.3. The problem ALBA-Mem is, given an ALBA $\mathcal{M}$ and an input w, to determine whether w is accepted by $\mathcal{M}$.

As mentioned above, ALBA-Mem is EXPTIME-complete [CKS81].

3.1. **EXPTIME algorithm.** Given an instance of the r-register FRA($M\#$) bisimilarity problem, the main idea is to consider a bounded version of the associated bisimulation game that uses a finite subset $N \subseteq \mathcal{D}$ of size $2r + 2$ as the alphabet. One can then determine the winner using an alternating algorithm running in polynomial space. This finite set of names is sufficient in order to faithfully capture the full bisimulation game, though a careful discipline is required when making moves with names that are not in the current sets of registers. Such names need to be sourced from the set N, in effect re-using names that have appeared before in the game. The crux of the argument is showing that such re-use does not affect the outcome of the (full) game.

Proposition 3.4. *$\sim$-FRA($M\#$) is in EXPTIME.*

Given an instance $\langle \mathcal{A}, (q_{01}, \rho_{01}, H_0), (q_{02}, \rho_{02}, H_0) \rangle$ of the bisimilarity problem for FRA($M\#$), where $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ has r registers, we first consider a *restricted* bisimilarity problem concerning configurations that contain names from a bounded subset of $\mathcal{D}$. Let us

pick a set $N \subseteq \mathcal{D}$ of cardinality $2r+2$, with a fixed enumeration $N = \{d_1, d_2, \ldots, d_{2r+2}\}$, such that:

(1) $H_0 \subseteq N$, if $|H_0| < 2r+2$;
(2) $\mathsf{rng}(\rho_{01}) \cup \mathsf{rng}(\rho_{02}) \subseteq N \subseteq H_0$, otherwise.

In the former case, N is a superset of H_0, while in the latter it is a subset. In either case, N includes all names in ρ_{01}, ρ_{02}. We also let the set of ***N-configurations***:

$$\mathbb{C}_{\mathcal{A},N} = \{(q, \rho, H) \in \mathbb{C}_{\mathcal{A}} \mid H \subsetneq N\}$$

contain all configurations involving names from N and whose histories are strictly included in N. Given ρ_1, ρ_2, H with $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq H \subseteq N$ (and hence $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subsetneq N$), we will sometimes refer to the following trimmed version of H:

$$\lceil H \rceil^N_{\rho_1,\rho_2} = \begin{cases} H & \text{if } H \subsetneq N \\ H \setminus \{\min(N \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))\} & \text{otherwise (i.e. if } H = N) \end{cases}$$

In the second case, $\lceil H \rceil^N_{\rho_1,\rho_2}$ is obtained from H by deleting the first name (according to the enumeration of N) that is not present in ρ_1 or ρ_2. Intuitively, the removed name will be recycled and available to simulate global freshness later.

We can now define a notion of bisimilarity adapted to N-configurations.

Definition 3.5. Given $\mathcal{A}$ and N as above, a binary relation $R \subseteq \mathbb{C}_{\mathcal{A},N} \times \mathbb{C}_{\mathcal{A},N}$ is an ***N-bisimulation*** if for each $((q_1, \rho_1, H_1), (q_2, \rho_2, H_2)) \in R$ we have $H_1 = H_2 (= H)$ and for all (t, d) with $d \in N$:

(1) if $(q_1, \rho_1, H) \xrightarrow{(t,d)} (q_1', \rho_1', H')$ and one of the following conditions holds:
   (a) $d \in \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$,
   (b) $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subsetneq H$ and $d = \min(H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)))$,
   (c) $d = \min(N \setminus H)$,

   then $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', \rho_2', H')$ and $((q_1', \rho_1', \lceil H' \rceil^N_{\rho_1',\rho_2'}), (q_2', \rho_2', \lceil H' \rceil^N_{\rho_1',\rho_2'})) \in R$;
(2) dual conditions hold for $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', \rho_2', H')$.

We say that κ_1 and κ_2 are ***N-bisimilar***, written $\kappa_1 \sim_N \kappa_2$, just if there is some N-bisimulation R with $(\kappa_1, \kappa_2) \in R$.

Remark 3.6. The idea behind N-bisimulations is that $2r+2$ names suffice in order to decide the bisimilarity problem. Given a pair of configurations $((q_1, \rho_1, H), (q_2, \rho_2, H))$, the specific names in ρ_1, ρ_2, H are immaterial; instead, of importance are:

- the sets of the registers in ρ_1 and ρ_2 containing the same names;
- whether the register assignments contain all names that are included in H.

$2r+1$ names are sufficient for encoding the above information. By allowing $2r+2$ names in total, we are then able to represent the full bisimulation game using only configurations from $\mathbb{C}_{A,N}$.

To see this, suppose we are at a pair $((q_1, \rho_1, H), (q_2, \rho_2, H)) \in \mathbb{C}_{A,N}$ in the (full) bisimulation game and WLOG Attacker chooses to play on the Left, say some $(q_1, \rho_1, H) \xrightarrow{(t,d)} (q_1', \rho_1', H')$. While there may be infinitely many possible choices for d, we can narrow them down to finitely many. We can partition $\mathcal{D}$ as:

$$\mathcal{D} = (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)) \uplus (H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))) \uplus (\mathcal{D} \setminus H)$$

and, for each block, only consider a finite number of representatives:

INPUT: FRA($M\#$) $\mathcal{A}, N, q_{01}, \rho_{01}, q_{02}, \rho_{02}, \hat{H}_0$

$q_1, \rho_1, q_2, \rho_2, H := q_{01}, \rho_{01}, q_{02}, \rho_{02}, \hat{H}_0$

repeat

- existentially choose $i \in \{1, 2\}$ and valid $(q_i, \rho_i, H) \xrightarrow{(t,d)} (q_i', \rho_i', H')$, or REJECT in the absence of any such choice;
- universally choose valid $(q_{3-i}, \rho_{3-i}, H) \xrightarrow{(t,d)} (q_{3-i}', \rho_{3-i}', H')$, or ACCEPT in the absence of any such choice;
- $q_1, \rho_1, q_2, \rho_2, H := q_1', \rho_1', q_2', \rho_2', \lceil H' \rceil^N_{\rho_1', \rho_2'}$

Figure 2: Alternating algorithm determining whether Attacker wins the N-bisimulation game.

(a) For $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$ we consider all elements.
(b) For $H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$ we can restrict our attention to the least d in $H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$ and ignore all others, as the specific choice of d from this set has no bearing on the outcome of the bisimulation game.
(c) For $\mathcal{D} \setminus H$, similarly to the previous case, the specific choice of d is not important, so we may as well pick d to be the least element in $N \setminus H$ (which is not empty as $H \subsetneq N$).

These three cases precisely correspond to cases (a-c) in Definition 3.5. Our analysis above would allow us to capture bisimilarity using N-configurations, if target configurations like (q_1', ρ_1', H') were still in $\mathbb{C}_{A,N}$. This does not always hold, as case (c) can lead us to $H' = N$. In this case, we use $\lceil H' \rceil^N_{\rho_1', \rho_2'}$ instead of H' so as to remain in $\mathbb{C}_{A,N}$. Since N has at least 2 more names than ρ_1' and ρ_2' combined, we can always pick a name from $N \setminus (\mathsf{rng}(\rho_1') \cup \mathsf{rng}(\rho_2'))$ to remove from $H' = N$ so that $\lceil H' \rceil^N_{\rho_1', \rho_2'} \setminus (\mathsf{rng}(\rho_1') \cup \mathsf{rng}(\rho_2'))$ remains non-empty. Such a choice will not affect the outcome of the bisimulation game.

Lemma 3.7. *Given $\mathcal{A}, (q_{01}, \rho_{01}, H_0), (q_{02}, \rho_{02}, H_0)$ and N as above, let $\hat{H}_0 = \lceil H_0 \cap N \rceil^N_{\rho_{01}, \rho_{02}}$. Then, $(q_{01}, \rho_{01}, H_0) \sim (q_{02}, \rho_{02}, H_0)$ iff $(q_{01}, \rho_{01}, \hat{H}_0) \sim_N (q_{02}, \rho_{02}, \hat{H}_0)$.*

It suffices to demonstrate that N-bisimilarity can be decided in alternating polynomial space, using the fact that APSPACE = EXPTIME.

Lemma 3.8. *Given $\mathcal{A}, N$ and $(q_{01}, \rho_{01}, \hat{H}_0), (q_{02}, \rho_{02}, \hat{H}_0)$ as above, we can decide*

$$(q_{01}, \rho_{01}, \hat{H}_0) \not\sim_N (q_{02}, \rho_{02}, \hat{H}_0)$$

with an alternating algorithm using space $O(r \log r + \log(|Q|))$.

Proof. We use the algorithm in Figure 2, which simply plays the N-bisimulation game, exploring existentially a strategy for Attacker. It accepts as soon as Defender cannot defend himself. Consequently, the algorithm accepts iff $(q_{01}, \rho_{01}, \hat{H}_0) \not\sim_N (q_{02}, \rho_{02}, \hat{H}_0)$. Moreover, the space it uses consists of q_1, q_2, the assignments ρ_1, ρ_2 (each bounded in space by $r \log(2r + 2)$), and the history H (bounded in space by $(2r + 2)$). Thus, the overall space used is $O(r \log r + \log(|Q|))$. □

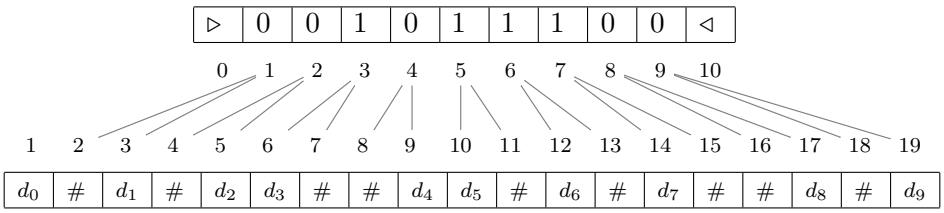


Figure 3: Encoding of a bounded tape of length 9 (top) using 18 registers (bottom, registers 2-19). The first register stores an auxiliary name which is used in the reduction of ALBA-Mem to $\sim$-RA($S\#$).

3.2. **EXPTIME hardness.** Further down the hierachy, we show that $\sim$-RA($S\#$) is EXPTIME-hard by reduction from ALBA-Mem. The idea is to use the registers of this class of automata to represent the tape content of ALBA's.

For the purposes of the argument, we will assume without loss of generality that we examine ALBA's such that $\Gamma \setminus \{\triangleleft, \triangleright\} = \{0, 1\}$ and, for all (q, a), $|\delta(q, a)| \leq 2$. Thus all choices presented by the alternation are binary. Starting from an instance of the ALBA-Mem problem $\langle \mathcal{M}, w \rangle$, we construct a bisimulation problem for RA($S\#$) in which two configurations are bisimilar iff $\mathcal{M}$ accepts w. From the ALBA $\mathcal{M}$ we construct an RA($S\#$) $\mathcal{A}$ that simulates it, with the binary tape content of $\mathcal{M}$ encoded by the register assignment of $\mathcal{A}$. We assume that cells numbered 0 and $|w| + 1$ contain end-markers and, to each tape cell $k \in [1, |w|]$, assign a corresponding pair of registers ($2k$ and $2k + 1$, to be exact) with exactly one of them being full and the other one being empty (i.e. containing #). Then, cell k will have 0 written on it iff register $2k$ is empty, and it has 1 written on it iff register $2k+1$ is empty. This is depicted in Figure 3. At every step of the bisimulation game, we arrange for Defender to choose transitions from existential states (using Defender forcing [JS08]) and for Attacker to make choices from universal states. Without loss of generality, for technical convenience, we will assume that the given ALBA does not diverge, i.e. it generates only finite computation paths (Theorem 2.6(b) [CKS81]).

Proposition 3.9. *$\sim$-RA($S\#$) is EXPTIME-hard.*

Given an instance $\langle \mathcal{M}, w \rangle$ of the ALBA-Mem problem, we construct a $2|w| + 1$ register RA($S\#$) $\mathcal{A}^w_{\mathcal{M}}$ whose induced bisimulation game simulates the computations of $\mathcal{M}$. A configuration of a computation of $\mathcal{M}$ will be represented, in duplicate, by a pair of configurations of $\mathcal{A}^w_{\mathcal{M}}$, which together make up a single configuration of the bisimulation game. These configurations will track the current state of $\mathcal{M}$ and the current position of the head of $\mathcal{M}$ in their state and the current tape contents of $\mathcal{M}$ will be represented by their current register assignment $\mathcal{A}^w_{\mathcal{M}}$. We will not require the use of any tags (*cf.* data words) in our construction, so we assume that Σ is a unary alphabet and omit this component in transitions.

Tape encoding. The first register is used to help implement a simulation of alternation and will never be empty. The last $2|w|$ registers of $\mathcal{A}^w_{\mathcal{M}}$ will be used to encode the (non-endmarker) tape content of $\mathcal{M}$ according to the following scheme: the tape cell $k \in [1, |w|]$

- contains 0 iff register $2k$ is empty iff register $2k + 1$ contains a name;
- and it contains 1 iff register $2k$ contains a name iff register $2k + 1$ is empty.

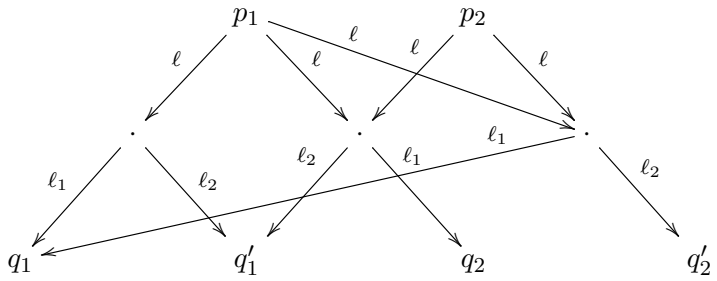


Figure 4: Defender forcing gadget $\mathsf{DF}(p_1, p_2, \ell, \ell_1, \ell_2, q_1, q_2, q_1', q_2')$. Labels ℓ_1 and ℓ_2 must be semantically distinct.

States. The set of states of $\mathcal{A}^w_{\mathcal{M}}$ is built from the states of $\mathcal{M}$, tape cell indices, tape letters, and special tags L, R:

$$Q' = (Q \times [0, |w| + 1] \times \{L, R\}) \uplus Q_{\text{aux}},$$

where Q_{aux} is a polynomially-sized set of auxiliary states whose role will be explained later on. Thus, each state $p \in Q' \setminus Q_{\text{aux}}$ is a tuple (q, k, C) where $q \in Q$ and:

- k is an index representing the position of the head of the tape of $\mathcal{M}$,
- and $C \in \{L, R\}$ is a tag allowing us to have two copies of each state.

Given $p \in Q' \setminus Q_{\text{aux}}$ and $x \in \{L, R\}$, we write $p[x]$ for the tuple p with its final component replaced by x. Taking an encoding ρ_I of w, our construction of $\mathcal{A}^w_{\mathcal{M}}$ shall ensure that configurations $((q_0, 0, L), \rho_I)$ and $((q_0, 0, R), \rho_I)$ are bisimilar iff $\mathcal{M}$ accepts w.

We motivate the construction by looking at the bisimulation game that it induces. A configuration in that game is a pair of configurations $((p_1, \rho_1), (p_2, \rho_2))$ of $\mathcal{A}^w_{\mathcal{M}}$. Our construction shall impose the following invariant at each round of the induced bisimulation game. If the game is at configuration $((p_1, \rho_1), (p_2, \rho_2))$ then:

$$\rho_1 = \rho_2 \wedge (p_1 = p_2 \vee \exists p. p_1 = p[L] \wedge p_2 = p[R]).$$

The idea is that when the two configurations are of the form $((q, k, C), \rho)$, with $C \in \{L, R\}$, the play is simulating a configuration of $\mathcal{M}$ which is in state q, with the head over tape cell k and the tape contents itself encoded by the last $2|w|$ registers of ρ.

Defender forcing. In order to describe the transition relation of the automaton we will make use of a gadget to implement defender forcing. Since the configurations of the induced bisimulation game are guaranteed, by the invariant, to have the same register contents, we are able to instantiate the general construction of [JS08], in which Attacker is punished for making choices inconsistent with Defender's wishes by allowing Defender to move his configuration into a configuration identical with that of Attacker.

The gadget is shown in Figure 4. The states denoted by dots are the ones constituting the set Q_{aux}. The gadget $\mathsf{DF}(p[L], p[R], \ell, \ell_1, \ell_2, p'[L], p'[R], p''[L], p''[R])$ ensures that, when the game configuration consists of two automata configurations of shape $(p[L], \rho)$ and $(p[R], \rho)$, then Defender can force the play so that the game enters a configuration consisting of either two automata configurations of shape $(p'[L], \rho')$ and $(p'[R], \rho')$, or two automata configurations of shape $(p''[L], \rho'')$ and $(p''[R], \rho'')$, where ρ' (respectively ρ'') is determined by transition labels ℓ and ℓ_1 (respectively ℓ and ℓ_2). It is by this defender forcing gadget

that we will be able to ensure that the two players correctly simulate existential choices made by $\mathcal{M}$, essentially by allowing Defender to make the choice.

Transitions. We describe the transitions of $\mathcal{A}^w_{\mathcal{M}}$ as part of a general description of how the induced bisimulation game simulates $\mathcal{M}$. Recall that a configuration of the form $((q,k,C),\rho)$ is used to simulate $\mathcal{M}$ in operating in state q with the head over cell k of tape encoded by ρ. Simulating a transition of $\mathcal{M}$ from this configuration requires reading and updating the tape, but also universally/existentially choosing the successor state.

Given a state (q,k,C) of $\mathcal{A}^w_{\mathcal{M}}$ and $a\in\Gamma$, we do a case analysis on $|\delta(q,a)|$. If $|\delta(q,a)|=0$ then there are no transitions to add. Otherwise, we proceed as follows. Let us fix transition labels $A=(\{1\},0,\emptyset)$ and $B=(\emptyset,1,\emptyset)$; these only involve the auxiliary register 1 and are semantically disjoint (from any given configuration, they cannot accept the same d). Below, where we use states denoted by dots, these are sourced from Q_{aux}.

I. $\delta(q,a)=\{(q',b,z)\}$**.** In this case, it suffices to decode the tape content, update it, and move to the next state. For reasons of uniformity, we will always employ two transitions at this step. The decoding and updating of the tape is split into two cases, depending on whether the head of the machine being simulated is over an endmarker or not. If $k\in\{0,|w|+1\}$ then the head is over an endmarker, and the content of cell k is completely determined by k, and not updated. Hence, in such cases we use transitions of the shape

$$(q,k,C)\xrightarrow{A}\cdot\xrightarrow{A}(q',k+z,C).$$

It is also useful to define labels $\ell_{\triangleright}=\ell'_{\triangleright}=\ell_{\triangleleft}=\ell'_{\triangleleft}=A$.

Otherwise, $k\in[1,|w|]$ and the head is over a cell which is encoded in the way described above. To decode and update it, we use transitions:

$$(q,k,C)\xrightarrow{\ell_a}\cdot\xrightarrow{\ell'_b}(q',k+z,C)$$

where ℓ_a allows us to decode a from the simulating registers (and reset them), and ℓ_b to update them with b. According to our encoding scheme:

$$\ell_0=(\{2k+1\},0,\{2k+1\}) \qquad \ell'_0=(\emptyset,2k+1,\emptyset)$$
$$\ell_1=(\{2k\},0,\{2k\}) \qquad \ell'_1=(\emptyset,2k,\emptyset)$$

Thus, for instance, if $ab=00$ then we use transitions

$$(q,k,C)\xrightarrow{\{2k+1\},0,\{2k+1\}}\cdot\xrightarrow{\emptyset,2k+1,\emptyset}(q',k+z,C)$$

so the first transition will read a name from register $2k+1$ (representing 0 in position k of the tape) and set that register to #. The next transition will update register $2k+1$ storing a new name d' (representing 0 again).

II. $\delta(q,a)=\{(q_1,b_1,z_1),(q_2,b_2,z_2)\}$**.** We consider whether q is a universal or existential move. In the former case, we add transitions:

$$(q_1,k+z_1,C)\xleftarrow{\ell'_{b_1}}\cdot\xleftarrow{A}\cdot\xleftarrow{\ell_a}(q,k,C)\xrightarrow{\ell_a}\cdot\xrightarrow{B}\cdot\xrightarrow{\ell'_{b_2}}(q_2,k+z_2,C)$$

If, on the other hand, q is existential, we use an instance of the Defender forcing gadget:

$$\mathsf{DF}((q,k,L),(q,k,R),\ell_a,\vec{\ell}_{b_1},\vec{\ell}_{b_2},(q_1,k+z_1,L),(q_1,k+z_1,R),(q_2,k+z_2,L),(q_2,k+z_2,R))$$

where, by abuse of notation, $\vec{\ell}_{b_1},\vec{\ell}_{b_2}$ are sequences of labels defined below.

$$\vec{\ell}_{b_1}=A;\ell'_{b_1} \qquad \vec{\ell}_{b_2}=B;\ell'_{b_2}$$

We note that the use of A and B ensures disjointness so that the gadget can be applied. This ensures that Defender can steer the simulation into her choice whilst maintaining the invariant about the shape of configurations.

Accepting and rejecting states. If the simulation reaches an accepting state then Defender should win. We organise for this to happen by forbidding any transition out of any state of shape (q_{acc}, k, C). In this way, any two configurations that are both in states of this form are trivially bisimilar since neither can perform an action. Conversely, Attacker should win if the simulation reaches a rejecting state. We organise for this to happen by transitions of the following shape:

$$(q_{\text{rej}}, k, L) \xrightarrow{\{1\},0,\emptyset} (q_{\text{rej}}, k, L)$$

Notice that such transitions only occur in those states that are tagged L. By construction, when the simulation arrives at a rejecting state, one configuration will in such a state tagged with L and the other with R and it follows that the two configurations will not be bisimilar.

Lemma 3.10. *Given an ALBA $\mathcal{M}$ and input w, $\mathcal{M}$ accepts w iff $((q_0, 0, L), \rho_I) \sim ((q_0, 0, R), \rho_I)$ in $\mathcal{S}(\mathcal{A}^w_{\mathcal{M}})$, where ρ_I is a register assignment encoding w in the way described above.*

Proof. By construction and our assumption that all ALBA computations terminate, there are only two ways Defender can win a play of the associated bisimulation game.

(i) By Attacker choosing a move in the Defender forcing gadget that results in a punishment response from Defender so that every game configuration that follows in the play is of shape $((p, \rho), (p, \rho))$, i.e. the components are trivially bisimilar.
(ii) By the play reaching a game configuration in which the two component configurations are of the shape $((q, k, L), \rho)$ and $((q, k, R), \rho)$ for $q = q_{\text{acc}}$, which are bisimilar by construction.

In the forward direction, assume that $\mathcal{M}$ accepts w. Then there is a computation tree T for w in which every leaf is accepting. Hence Defender can win every play of the corresponding bisimulation game by using T as a representation of a winning strategy. In particular, for any given play there are two possibilities. If Attacker plays badly inside a Defender forcing gadget and is punished then the result is (i) above. Otherwise, as long as Defender makes choices consistent with T then every play will eventually reach a configuration which simulates $\mathcal{M}$ in accepting state q_{acc}. By construction, the corresponding game configuration must have component configurations of shape $((q_{\text{acc}}, k, L), \rho)$ and $((q_{\text{acc}}, k, R), \rho)$ and Defender wins as described in (ii).

In the backward direction, assume that Defender has a winning strategy W for the bisimulation game. Then, since this strategy must specify which transition to choose when simulating a computation from an existential state and because we assume that the given ALBA terminates, the strategy can be used to build a finite computation tree T for $\mathcal{M}$ on w. Since, by construction, Attacker can always avoid being punished whilst playing in a defender forcing gadget, it follows that W must allow Defender to win any such play by the criterion (ii). Hence, every simulation which follows W ends in an accepting state and it follows that every leaf of T is accepting. □

4. PSPACE-completeness for RAs with single assignment without erasure (RA($S\#_0$))

We next prove that the EXPTIME bound can be improved if duplicate values and erasures are forbidden. We handle register automata first to expose the flavour of our technique. The main result is given below, it follows from Propositions 4.19 and 4.20.

Theorem 4.1. *$\sim$-RA($S\#_0$) is PSPACE-complete.*

Simplified notation. Recall that, in any transition $q_1 \xrightarrow{t,X,i,Z} q_2$ of an r-RA($S\#_0$), we have that $X \subseteq [1,r]$, $|X| \leq 1$, $Z = \emptyset$, and $X \neq \emptyset$ implies $i = 0$. These restrictions allow for a simpler notation for transitions, with $\delta \subseteq Q \times \Sigma \times ([1,r] \cup \{i^\bullet \mid i \in [0,r]\}) \times Q$:

(a) we write each transition $q_1 \xrightarrow{t,\{i\},0,\emptyset} q_2$ as $q_1 \xrightarrow{t,i} q_2$, where $i \in [1,r]$;
(b) and each transition $q_1 \xrightarrow{t,\emptyset,i,\emptyset} q_2$ as $q_1 \xrightarrow{t,i^\bullet} q_2$, where $i \in [0,r]$.

Thus, transitions of type (a) correspond to the automaton reading an input (t,a) where a is the name in the i-th register; while in (b) transitions the automaton reads (t,a) if a is *locally fresh*, that is, it does not appear in the registers, and in this case a will be stored in register i (for $i \in [1,r]$) or not stored in any register ($i = 0$).

Composition of assignments. Recall that register assignments in the S case are injective on non-empty registers, we will refer to them as *assignments of type S*. In what follows we will be composing r-register assignments ρ_1, ρ_2 of type S to obtain partial permutations capturing the positions of their common names.

Definition 4.2. Given an r-register assignment ρ of type S, let us define its inverse by

$$\rho^{-1} = \{(d,i) \in \mathcal{D} \times [1,r] \mid \rho(i) = d\},$$

i.e. as the inverse of $\rho \cap ([1,r] \times \mathcal{D})$.

We can observe that, if ρ_1, ρ_2 are r-register assignments of type S then $\rho_1; \rho_2^{-1}$ is a partial permutation. One can show that updates of assignments and permutations are related as follows.

Lemma 4.3. *Given r-register assignments ρ_1, ρ_2 of type S, $d \in \mathcal{D}$ and $i, j \in [0,r]$ such that*

$$(d \in \mathsf{rng}(\rho_1) \implies d = \rho_1(i)) \wedge (d \in \mathsf{rng}(\rho_2) \implies d = \rho_2(j)),$$

we have $(\rho_1; \rho_2^{-1})[i \mapsto j] = \rho_1[i \mapsto d]; \rho_2[j \mapsto d]^{-1}$.

4.1. Symbolic bisimulations.

We attack the bisimulation problem *symbolically*, i.e. by abstracting actual names in the bisimulation game to the indices of the registers where these names reside. This will lead us to consider groups of finite permutations and inverse semigroups of partial finite permutations. In symbolic bisimulations we shall consider pairs (q,S) of a state q and a set of register indices $S \subseteq [1,r]$, as representing configurations of the form (q,ρ) where $\mathsf{dom}(\rho) = S$. In this way, the locations of the empty registers $[1,r] \setminus S$ are made explicit. Configurations in a symbolic bisimulation relation will consist of triples of the form $(q_1, S_1, \sigma, q_2, S_2)$ where (q_i, S_i) will be as above, while $\sigma \in \mathcal{IS}_r$ shall be a partial permutation matching register indices in S_1 to indices in S_2. Such tuples will represent

concrete configuration pairs of the form $((q_1, \rho_1), (q_2, \rho_2))$ where the $\sigma = \rho_1; \rho_2^{-1}$: in words, σ contains all pairs of registers that contain the same name in ρ_1 and ρ_2 respectively.

Definition 4.4. Let $\mathcal{A} = \langle Q, \Sigma, \delta \rangle$ be an r-RA($S\#_0$). We first set:

$$\begin{aligned} \mathcal{U}_0 &= Q \times \mathcal{P}([1,r]) \times \mathcal{IS}_r \times Q \times \mathcal{P}([1,r]) \\ \mathcal{U} &= \{\, (q_1, S_1, \sigma, q_2, S_2) \in \mathcal{U}_0 \mid \sigma \subseteq S_1 \times S_2 \,\} \end{aligned}$$

A *symbolic simulation* on $\mathcal{A}$ is a relation $R \subseteq \mathcal{U}$, with membership $(q_1, S_1, \sigma, q_2, S_2) \in R$ often written infix $(q_1, S_1)\, R_\sigma\, (q_2, S_2)$, such that all $(q_1, S_1, \sigma, q_2, S_2) \in R$ satisfy the following *symbolic simulation conditions* (SyS):[4]

- for all $q_1 \xrightarrow{t,i} q_1'$,
  - if $i \in \mathsf{dom}(\sigma)$ then there is some $q_2 \xrightarrow{t,\sigma(i)} q_2'$ with $(q_1', S_1)\, R_\sigma\, (q_2', S_2)$,
  - if $i \in S_1 \setminus \mathsf{dom}(\sigma)$ then there is some $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1)\, R_{\sigma[i \mapsto j]}\, (q_2', S_2[j])$;
- for all $q_1 \xrightarrow{t,i^\bullet} q_1'$,
  - there is some $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1[i])\, R_{\sigma[i \mapsto j]}\, (q_2', S_2[j])$,
  - for all $j \in S_2 \setminus \mathsf{rng}(\sigma)$, there is some $q_2 \xrightarrow{t,j} q_2'$ with $(q_1', S_1[i])\, R_{\sigma[i \mapsto j]}\, (q_2', S_2)$.

We let the inverse of R be

$$R^{-1} = \{\, (q_2, S_2, \sigma^{-1}, q_1, S_1) \mid (q_1, S_1, \sigma, q_2, S_2) \in R \,\}$$

and call R a ***symbolic bisimulation*** if both R and R^{-1} are symbolic simulations. We let *symbolic bisimilarity*, denoted $\overset{\mathsf{s}}{\sim}$, be the union of all symbolic bisimulations. We say that (q_1, ρ_1) and (q_2, ρ_2) are *symbolic bisimilar* if $(q_1, \mathsf{dom}(\rho_1), \rho_1; \rho_2^{-1}, q_2, \mathsf{dom}(\rho_2)) \in \overset{\mathsf{s}}{\sim}$, i.e. $(q_1, \mathsf{dom}(\rho_1)) \overset{\mathsf{s}}{\sim}_{\rho_1;\rho_2^{-1}} (q_2, \mathsf{dom}(\rho_2))$. We will then also write $(q_1, \rho_1) \overset{\mathsf{s}}{\sim} (q_2, \rho_2)$.

Symbolic bisimulation provides a means to finitely represent an otherwise infinite bisimulation relation. The following result proves that this representation is precise. Its proof is based on a case analysis showing that symbolic bisimulation rules capture concrete ones, and vice versa.

Lemma 4.5. *Given configurations (q_1, ρ_1), (q_2, ρ_2) of an r-RA($S\#_0$), $(q_1, \rho_1) \sim (q_2, \rho_2) \iff (q_1, \rho_1) \overset{\mathsf{s}}{\sim} (q_2, \rho_2)$.*

It will be useful to approximate symbolic bisimilarity by a sequence of ***indexed bisimilarity*** relations $\overset{i}{\sim} \subseteq \mathcal{U}$ defined inductively as follows. First, we let $\overset{0}{\sim}$ be the whole of $\mathcal{U}$. Then, for all $i \in \omega$, $(q_1, S_1, \tau, q_2, S_2) \in \overset{i+1}{\sim}$ just if $(q_1, S_1, \tau, q_2, S_2)$ and $(q_2, S_2, \tau^{-1}, q_1, S_1)$ both satisfy the (SyS) conditions in $\overset{i}{\sim}$. We can show the following.

Lemma 4.6. *For all $i \in \omega$, $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$ and $(\bigcap_{i \in \omega} \overset{i}{\sim}) = \overset{\mathsf{s}}{\sim}$.*

Remark 4.7. Given Lemmata 4.5 and 4.6, to obtain a polynomial-space algorithm for bisimilarity, it suffices to obtain a polynomial-space algorithm for symbolic bisimilarity. For the latter, it is enough to establish that symbolic bisimulation games can be decided in polynomially many rounds. In other words, it suffices to show that there is polynomial bound B (dependent on the examined $\mathcal{A}$) such that $\overset{B}{\sim} = \overset{\mathsf{s}}{\sim}$.

[4] We say that $(q_1, S_1, \sigma, q_2, S_2)$ *satisfies the* (SyS) *conditions in R.*

Our next aim is to show that $\overset{\mathsf{s}}{\sim}$ and each $\overset{i}{\sim}$ are closed under composition and extension of partial permutations. Such a closure for $\overset{i}{\sim}$ will allow us to polynomially bound the convergence of indexed bisimilarities by finding within them strict chains of subgroups (cf. Lemma 4.16). The closure of $\overset{\mathsf{s}}{\sim}$, on the other hand, will help us represent $\overset{\mathsf{s}}{\sim}$ succinctly by appropriate choices of representatives (cf. Section 6).

Given $S_1, S_2 \subseteq [1,r]$ and $\sigma, \sigma' \in \mathcal{IS}_r$ we write $\sigma \leq_{S_1,S_2} \sigma'$ just if $\sigma \subseteq \sigma' \subseteq S_1 \times S_2$. Moroever, given $X \subseteq S \subseteq [1,r]$, we write id_X for the partial map from S to S that acts as identity on X (and is undefined otherwise). For any $R \subseteq \mathcal{U}$, we define its ***closure*** $Cl(R)$ to be the smallest relation R' containing R and closed under the following rules.

$$\frac{}{(q,S,\mathsf{id}_S,q,S) \in R'}\ (\textsc{Id}) \qquad \frac{(q_1,S_1,\sigma_1,q_2,S_2) \in R' \qquad (q_2,S_2,\sigma_2,q_3,S_3) \in R'}{(q_1,S_1,\sigma_1;\sigma_2,q_3,S_3) \in R'}\ (\textsc{Tr})$$

$$\frac{(q_1,S_1,\sigma,q_2,S_2) \in R'}{(q_2,S_2,\sigma^{-1},q_1,S_1) \in R'}\ (\textsc{Sym}) \qquad \frac{(q_1,S_1,\sigma,q_2,S_2) \in R' \qquad \sigma \leq_{S_1,S_2} \sigma'}{(q_1,S_1,\sigma',q_2,S_2) \in R'}\ (\textsc{Ext})$$

We say that R is *closed* in case $Cl(R) = R$.

Much of the following development relies upon the fact that bisimilarity and indexed bisimilarity are closed. Intuitively, this amounts to showing that the (SyS) conditions are compatible with the rules above, i.e. if their premises satisfy the conditions then so do the conclusions. The interesting cases are (Tr) and (Ext). For the former, the argument is a symbolic version of showing that (bi)simulation is transitive. The case of (Ext) is subtler, as we need to argue that it is sound to relate previously unrelated registers.

Lemma 4.8. *Let $P, R \subseteq \mathcal{U}$. If all $g \in R \cup R^{-1}$ satisfy the* (SyS) *conditions in P then all $g \in Cl(R)$ satisfy the* (SyS) *conditions in $Cl(P)$.*

Corollary 4.9. *(Closures) Bisimilarity and indexed bisimilarity for $RA(S\#_0)$ are both closed:*

(1) $\overset{\mathsf{s}}{\sim} \;=\; Cl(\overset{\mathsf{s}}{\sim})$ *;*

(2) *for all $i \in \omega$: $\overset{i}{\sim} \;=\; Cl(\overset{i}{\sim})$.*

Proof. For 1 note that $\overset{\mathsf{s}}{\sim} = (\overset{\mathsf{s}}{\sim})^{-1}$ and all its elements satisfy the (SyS) conditions in $\overset{\mathsf{s}}{\sim}$. Hence, by Lemma 4.8 we have that $Cl(\overset{\mathsf{s}}{\sim})$ is a symbolic bisimulation, i.e. $Cl(\overset{\mathsf{s}}{\sim}) \subseteq \overset{\mathsf{s}}{\sim}$. The result then follows. For 2 we proceed by induction on i. When $i = 0$ then the result follows from the fact that $\overset{0}{\sim}$ is the universal relation. For the inductive case, note first that $\overset{i+1}{\sim}$ is symmetric by construction and all $g \in \overset{i+1}{\sim}$ satisfy the (SyS) conditions in $\overset{i}{\sim}$. Hence, by Lemma 4.8, all elements of $Cl(\overset{i+1}{\sim})$ satisfy the (SyS) conditions in $Cl(\overset{i}{\sim})$. By IH, $Cl(\overset{i}{\sim}) = \overset{i}{\sim}$ so $Cl(\overset{i+1}{\sim}) \subseteq \overset{i+1}{\sim}$, as required. □

4.2. **Bounding indexed bisimilarity convergence using permutation groups.** To bound the rate of convergence of indexed bisimilarities we study the strict sub-chains:

$$\{\overset{i}{\sim} \mid (\overset{i+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2})\} \tag{4.1}$$

that we obtain for a given pair of sets $S_1, S_2 \subseteq [1,r]$, where:

$$\mathcal{U}_{S_1,S_2} = \{(q_1,S_1',\sigma,q_2,S_2') \in \mathcal{U} \mid S_1 = S_1', S_2 = S_2'\}.$$

Our aim is to find a bound for i in (4.1), independent of S_1, S_2. To this end, below we introduce two auxiliary notions that will help us identify some structure within the $\overset{i}{\sim}$ relations. In particular, we shall study self-symmetries, which lead to group-theoretic considerations and enable us to relate the evolution of $\overset{i}{\sim}$ to descending subgroup chains.

Definition 4.10. Let $p \in Q, S \subseteq [1,r]$ and $R \subseteq \mathcal{U}$ be closed. We define:

- the ***characteristic set*** *of* (p,S) *in* R as: $X^p_S(R) = \bigcap\{X \subseteq S \mid (p,S)\ R_{\mathsf{id}_X}\ (p,S)\}$,
- the ***characteristic group*** *of* (p,S) *in* R as: $\mathcal{G}^p_S(R) = \{\sigma \subseteq X^p_S(R) \times X^p_S(R) \mid (p,S)\ R_\sigma\ (p,S)\}$.

Note that $R_1 \subseteq R_2$ implies $X^p_S(R_1) \supseteq X^p_S(R_2)$. We are going to show (in Lemma 4.16) that changes in $\overset{j}{\sim} \cap\, \mathcal{U}_{S_1,S_2}$ (as j increases) can be traced back to either expansion of a characteristic set $X^p_S(\overset{j}{\sim})$ ($S \in \{S_1, S_2\}$), or shrinkage of some $\mathcal{G}^p_S(\overset{j}{\sim})$ ($S \in \{S_1, S_2\}$) or disappearance of all tuples $(q_1, S_1, \sigma, q_2, S_2)$ for some $q_1, q_2 \in Q$. The number of changes of each kind can be bounded by a polynomial. In the second case, we shall rely on the fact that each $\mathcal{G}^p_S(\overset{j}{\sim})$ is indeed a group (Lemma 4.15) and on the following result which concerns subgroup chains in a group G:

$$G = G_0 > G_1 > \cdots > G_m = I$$

in which I is the trivial identity group and, for all $i \in [0, m-1]$, G_{i+1} is a strict subgroup of G_i.

Theorem 4.11 [Bab86]**.** *For $n \geq 2$, the length of every subgroup chain in $\mathcal{S}_{[1,n]}$ is at most $2n-3$.*

Remark 4.12. The above result provides a linear bound, which we will be using in subsequent calculations. Note, though, that the existence of a quadratic bound follows easily from Lagrange's theorem. In particular, it implies $|G_i| \geq 2|G_{i+1}|$ $(0 \leq i < m)$ and, thus, $|G| \geq 2^m$. Consequently, $m \leq \log_2(|G|) \leq \log_2(n!) \leq n \log_2(n) \leq n^2$.

Before tackling Lemma 4.15, we prove an auxiliary lemma.

Lemma 4.13. *Let p, S, R be as above. Suppose $(p,S)\ R_\sigma\ (q,S)$, then:*

- $\mathsf{dom}(\sigma) \supseteq X^p_S(R)$ *and* $\mathsf{rng}(\sigma) \supseteq X^q_S(R)$.
- *Setting $\sigma' = \sigma \cap (X^p_S(R) \times X^q_S(R))$, we have $\mathsf{dom}(\sigma') = X^p_S(R)$, $\mathsf{rng}(\sigma') = X^q_S(R)$ and $(p,S)\ R_{\sigma'}\ (q,S)$. In particular, $(p,S)\ R_{\mathsf{id}_{X^p_S(R)}}\ (p,S)$.*

Remark 4.14. The above Lemma shows that $R \cap \mathcal{U}_{S,S}$ can be generated from elements of the form $(p,S)\ R_\sigma\ (q,S)$, where σ is a bijection between $X^p_S(R)$ and $X^q_S(R)$, using up-closure under $\leq_{S,S}$. That is, $(p,S)\ R_{\sigma'}\ (q,S)$ iff there exists a bijection $\sigma : X^p_S(R) \to X^q_S(R)$ such that $\sigma \leq_{S,S} \sigma'$ and $(p,S)\ R_\sigma\ (q,S)$.

Lemma 4.15. *$\mathcal{G}^p_S(R)$ is a group (under composition). In particular, it is a subgroup of $\mathcal{S}_{X^p_S(R)}$.*

Proof. By the last part of Lemma 4.13, we have $\mathsf{id}_{X^p_S(R)} \in \mathcal{G}^p_S$. Now let $\sigma \in \mathcal{G}^p_S(R)$, i.e. $(p,S)\ R_\sigma\ (p,S)$ with $\sigma \subseteq X^p_S(R) \times X^p_S(R)$. By first part of Lemma 4.13, we have $\sigma \in \mathcal{S}_{X^p_S(R)}$. Moreover, $(p,S)\ R_{\sigma^{-1}}\ (p,S)$, by closure of R, hence $\sigma^{-1} \in \mathcal{G}^p_S(R)$. Finally, if $\sigma' \in \mathcal{G}^p_S(R)$, again using closure of R, we get $\sigma;\sigma' \in \mathcal{G}^p_S(R)$. □

We can now use the above structure in indexed bisimilarities to bound their rate of convergence.

Lemma 4.16. *Given $S_1, S_2 \subseteq [1,r]$, the sub-chain $\{\overset{i}{\sim} \mid (\overset{i+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2})\}$ has size $O(|Q|^2 + r^2|Q|)$.*

Proof. Fix $S_1, S_2 \subseteq [1,r]$. We argue that $\{\overset{i}{\sim} \mid (\overset{i+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2})\}$ has length at most $|Q|^2 + 4r^2|Q| - 2r|Q|$.

Let us say that two configurations (q_1, S_1) and (q_2, S_2) are *separated* in $\overset{i}{\sim}$ just if there is no σ such that $(q_1, S_1) \overset{i}{\sim}_\sigma (q_2, S_2)$; we say they are *unseparated* otherwise. We claim that if $(\overset{i+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2})$ then:

(i) there is some $q \in Q$ and $S \in \{S_1, S_2\}$ such that $X^q_S(\overset{i+1}{\sim}) \supsetneq X^q_S(\overset{i}{\sim})$,

(ii) or there is some $q \in Q$ and $S \in \{S_1, S_2\}$ such that $\mathcal{G}^q_S(\overset{i+1}{\sim})$ is a strict subgroup of $\mathcal{G}^q_S(\overset{i}{\sim})$,

(iii) or there are configurations $(q_1, S_1), (q_2, S_2)$ that are unseparated in $\overset{i}{\sim}$ and become separated in $\overset{i+1}{\sim}$.

We argue as follows. If $(\overset{i+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2})$ then there are some $p, q \in Q$ and σ such that $(q_1, S_1) \overset{i}{\sim}_\sigma (q_2, S_2)$ but $(q_1, S_1) \overset{i+1}{\not\sim}_\sigma (q_2, S_2)$. Note that, in such a case it follows that also $(q_1, S_1) \overset{i}{\sim}_{\sigma'} (q_2, S_2)$ and $(q_1, S_1) \overset{i+1}{\not\sim}_{\sigma'} (q_2, S_2)$, where $\sigma' = \sigma \cap (X^{q_1}_{S_1}(\overset{i}{\sim}) \times X^{q_2}_{S_2}(\overset{i}{\sim}))$, by closure of $\overset{i}{\sim}$ (Tr for $\mathsf{id}_{X^{q_1}_{S_1}(\overset{i}{\sim})}$ and $\mathsf{id}_{X^{q_2}_{S_2}(\overset{i}{\sim})}$) and $\overset{i+1}{\sim}$ (contraposition with (Ext)). Hence, we assume wlog that $\mathsf{dom}(\sigma) = X^{q_1}_{S_1}(\overset{i}{\sim})$ and $\mathsf{rng}(\sigma) = X^{q_2}_{S_2}(\overset{i}{\sim})$. Now, suppose that, for all $q \in Q$, $S \in \{S_1, S_2\}$, $X^q_S(\overset{i+1}{\sim}) = X^q_S(\overset{i}{\sim})$ and no previously unseparated pair of configurations become separated in $\overset{i+1}{\sim}$. It follows from $(q_1, S_1) \overset{i+1}{\not\sim}_\sigma (q_2, S_2)$ that there is some τ such that $(q_1, S_1) \overset{i+1}{\sim}_\tau (q_2, S_2)$ and thus $\sigma; \tau^{-1} \in \mathcal{G}^{q_1}_{S_1}(\overset{i}{\sim})$ but $\sigma; \tau^{-1} \notin \mathcal{G}^{q_1}_{S_1}(\overset{i+1}{\sim})$. Hence $\mathcal{G}^{q_1}_{S_1}(\overset{i}{\sim}) > \mathcal{G}^{q_1}_{S_1}(\overset{i+1}{\sim})$.

To bound the length of the chain $\{\overset{i}{\sim} \mid (\overset{i+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2})\}$, observe that we always have $X^q_S(\overset{i+1}{\sim}) \supseteq X^q_S(\overset{i}{\sim})$ because of $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$. Thus, (i) may happen at most $2r|Q|$ times inside the chain. If (i) does not hold then $X^q_S(\overset{i+1}{\sim}) = X^q_S(\overset{i}{\sim})$ for all q and $S \in \{S_1, S_2\}$. For fixed $X^q_S(\overset{i}{\sim})$, by Theorem 4.11, (ii) may happen at most $2r - 2$ times (we include the case $r = 1$), which gives an upper bound of $2r|Q|(2r-2)$ for the number of such changes inside the whole chain (under the assumption that the changes are not of type (i), which have already been counted). Finally, the remaining changes must be of type (iii) and may happen at most $|Q|^2$ times across the whole chain. Overall, we obtain $2r|Q| + 2r|Q|(2r-2) + |Q|^2 = |Q|^2 + 4r^2|Q| - 2r|Q|$ as a bound on the length of the given chain. □

Note that it does not quite follow from the above result that the sequence $(\overset{i}{\sim})$ converges in polynomially many steps, because there are exponentially many pairs (S_1, S_2). Next we shall establish such a bound by studying more closely the overlap in evolutions of different (S_1, S_2).

Lemma 4.17. *Let ℓ be the bound from Lemma 4.16 and $B = (2r+1)\ell$. Then, for any S_1, S_2, $\overset{B}{\sim} \cap\, \mathcal{U}_{S_1,S_2} = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}_{S_1,S_2}$.*

Proposition 4.18. *For any RA($S\#_0$) bisimulation problem, if there is a winning strategy for Attacker then there is one of depth $O(r|Q|^2 + r^3|Q|)$.*

Proof. We first observe that bisimulation strategies and their corresponding symbolic bisimulation strategies have the same depth. Thus, it suffices to bound symbolic strategies for Attacker. The $O(r|Q|^2 + r^3|Q|)$ bound follows from the preceding Lemma. □

Proposition 4.19. *$\sim$-RA($S\#_0$) is in PSPACE.*

Proof. Thanks to the bound obtained in Lemma 4.17, to decide symbolic bisimilarity it suffices to play the corresponding symbolic bisimulation game for polynomially many steps. The existence of a winning strategy can then be established by an alternating Turing machine running in polynomial time, analogously to Figure 2. The PSPACE bound follows from APTIME = PSPACE. □

4.3. **PSPACE hardness.** For PSPACE-hardness, we reduce from the well-known PSPACE-complete problem of checking validity of totally quantified boolean formulas in prenex conjunctive normal form. One possibility is to decompose this reduction via the acceptance problem for ALBA that are not allowed to overwrite non-blank tape cells – *write-once ALBA*. Given an instance of QBF, one can construct a write-once ALBA with enough space on its tape to store the formula and a truth assignment, which it guesses by alternating moves according to the quantifiers, and then verifies deterministically. Then our reduction of Section 3.2 applies to obtain an instance of the bisimilarity problem for RA($S\#$) but, because the ALBA is write-once, so the corresponding RA obeys $S\#_0$. However, there is a more straightforward, direct reduction, which we present below.

In our construction, universal quantification and selection of conjuncts is performed by Attacker. For existential quantification and disjunctions, we rely on Defender Forcing. The choices of truth values by both players are recorded in registers by using, for each variable x_i, registers $2i, 2i+1$, both initialised to #. If a player chooses *true* for x_i, we fill register $2i$ leaving $2i+1$ empty; we do the opposite otherwise. This makes it possible to arrange for bisimilarity/non-bisimilarity (as appropriate) in the final stage of the game, depending on whether the resulting literal is negated.

Proposition 4.20. *$\sim$-RA($S\#_0$) is PSPACE-hard.*

Proof. We reduce from TQBF, i.e. the problem of deciding whether a formula Φ of the shape $\Box_1 x_1 \cdots \Box_h x_h.\phi(x_0, \cdots, x_h)$ (with ϕ in conjunctive normal form and each $\Box$ a quantifier) is true.

We shall construct a $(2h+1)$-register RA($S\#_0$) and configurations κ_L, κ_R such that $\kappa_L \sim \kappa_R$ if and only if Φ is true. We will not require the use of any tags in our construction, so we assume that Σ is a unary alphabet and omit this component in transitions. We pick some name d_0. For $C \in \{L, R\}$, we shall have

$$\kappa_C = ((q_1, C), \rho_0)$$

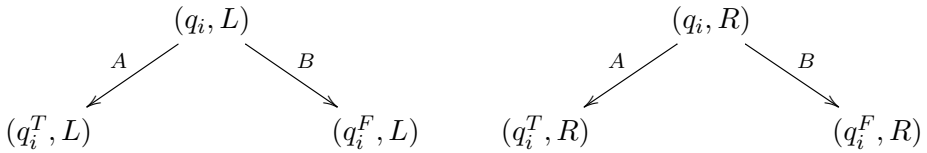
with $\rho_0(1) = d_0$ and $\rho_0(i) = \#$ for all other i.

The first register is used to let Attacker/Defender make choices. Registers $2, \cdots, 2h+1$ will represent truth-value assignments. Registers $2i, 2i+1$ will be used to represent the value of x_i $(i = 1, \cdots, h)$ subject to the following conditions:

- register $2i$ is filled if and only if the value of x_i is true,

• register $2i+1$ is filled if and only if the value of x_i is false.

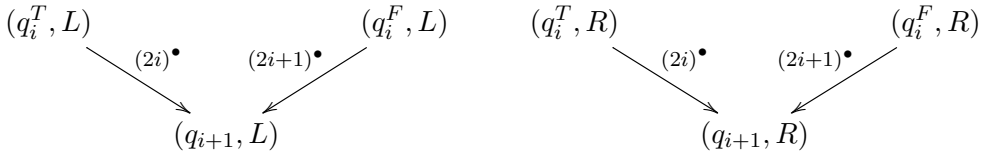
The values will be selected by Attacker (when $\Box_i = \forall$) or Defender (when $\Box_i = \exists$). Formally, if $\Box_i = \forall$ then we add the following transitions, where $A = 1$ and $B = 1^\bullet$,



which allows Attacker to force the play from $((q_i, L), (q_i, R))$ into either $((q_i^T, L), (q_i^T, R))$ or $((q_i^F, L), (q_i^F, R))$. On the other hand, if $\Box_i = \exists$ then we add (cf. Figure 4):

$$\mathsf{DF}((q_i, L), (q_i, R), A, A, B, (q_i^T, L), (q_i^T, R), (q_i^F, L), (q_i^F, R))$$

We follow up the above transitions with register-setting ones:

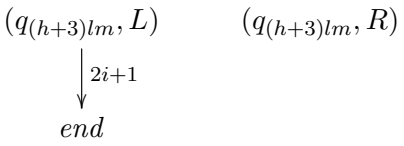


The above handles quantification. To represent the formula $\phi = \phi_1 \wedge \cdots \wedge \phi_k$, we allow Attacker to force the play from $((q_{h+1}, L), (q_{h+1}, R))$ into any of $((q_{(h+1)l}, L), (q_{(h+1)l}, R))$ for $l = 1, \cdots, k$ using e.g. transition sequences with labels from $\{A, B\}^{k-1}$.

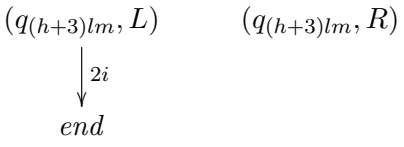
Now assume $\phi_l = \phi_{l1} \vee \cdots \vee \phi_{ln_l}$, where $\phi_{lm} = X_i$ or $\phi_{lm} = \neg X_i$ $(m = 1, \cdots, n_l)$. To represent ϕ_l, we iterate the DF circuit $n_l - 1$ times so that Defender can force the play from $((q_{(h+2)l}, L), (q_{(h+2)l}, R))$ into any of $((q_{(h+3)lm}, L), (q_{(h+3)lm}, R))$ for $m = 1, \cdots, n_l$.

Finally, we need to handle the formulas ϕ_{lm}.

• If $\phi_{lm} = X_i$ we add



• If $\phi_{lm} = \neg X_i$ we add



Note that the outgoing transitions are added only for states tagged with L. They give Attacker a chance to win if ϕ_{lm} does *not* hold after Defender's choices.

Overall the construction yields a winning strategy for Defender if and only if the given formula is true. □

5. Language equivalence for $RA(S\#_0)$

The results of the previous section can be used to close an existing complexity gap for deterministic language equivalence of register automata. Recall that, in the non-deterministic case, language equivalence (even universality) is undecidable [NSV04]. In the deterministic case, however, the problem can be solved in polynomial space. Sakamoto [Sak98] conjectured that the language inequivalence problem is not in NP. Below we refute the conjecture, showing that, for RA($S\#_0$), the complexity of deterministic language inequivalence actually matches that of nonemptiness [SI00]. Because we discuss language equivalence, in this section we assume that RA($S\#_0$) are given as $\langle Q, \Sigma, q_0, \rho_0, \delta, F\rangle$, where $q_0 \in Q$ is the initial state, ρ_0 is an initial register assignment conforming to the $S\#_0$ policy, and $F \subseteq Q$ is a set of accepting states.

We call an r-RA($S\#_0$) $\mathcal{A}$ *deterministic* if, for all states q of $\mathcal{A}$:

(i) for all $(t, i) \in \Sigma \times [1, r]$ there is at most one transition of the form $q \xrightarrow{t,i} q'$, and
(ii) for all $t \in \Sigma$ there is at most one transition of the form $q \xrightarrow{t,i^\bullet} q'$ for $i \in [0, r]$.

On the other hand, an LTS is deterministic if, for all $\kappa \in \mathbb{C}$ and $\ell \in \mathcal{A}ct$, there is at most one transition $\kappa \xrightarrow{\ell} \kappa'$. Note that if $\mathcal{A}$ is deterministic then so is its transition system $\mathcal{S}(\mathcal{A})$.[5] Then, from Proposition 4.18, one obtains the following.

Lemma 5.1. *Let $\mathcal{A}_i = \langle Q_i, \Sigma, q_{0i}, \rho_{0i}, \delta_i, F_i\rangle$ be a deterministic r_i-RA($S\#_0$) ($i = 1, 2$), $r = \max(r_1, r_2)$ and $N = |Q_1| + |Q_2|$. If $\mathcal{L}(\mathcal{A}_1) \neq \mathcal{L}(\mathcal{A}_2)$ then there is some $w \in (\mathcal{L}(\mathcal{A}_1) \cup \mathcal{L}(\mathcal{A}_2)) \setminus (\mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\mathcal{A}_2))$ with $|w| \in O(rN^2 + r^3N)$.*

Proof. We view $\mathcal{A}_1, \mathcal{A}_2$ as r-RA($S\#_0$)s with some unused registers and consider the r-RA($S\#_0$)

$$\mathcal{A} = \langle Q_1 \uplus Q_2 \uplus \{q_0\} \uplus \{q_s\}, q_0, \Sigma, \{(i, \#) \mid i \in [1, r]\}, \delta_1 \cup \delta_2 \cup \delta_s \cup \delta'_s \cup \delta_F, \emptyset\rangle,$$

where q_0 is a "blind" initial state, q_s is a sink state, $\delta_s = \{q \xrightarrow{t,i} q_s \mid \delta(q) \upharpoonright (t, i) = \emptyset\} \cup \{q \xrightarrow{t,1^\bullet} q_s \mid \delta(q) \upharpoonright (t, i^\bullet) = \emptyset\}$ adds any missing outgoing transitions to $\delta = \delta_1 \cup \delta_2$, $\delta'_s = \{q_s \xrightarrow{t,i} q_s \mid t \in \Sigma \wedge i \in [1, r]\} \cup \{q_s \xrightarrow{t,1^\bullet} q_s \mid t \in \Sigma\}$ is a set of sink transitions, and $\delta_F = \{q \xrightarrow{t_F,i} q_0 \mid i \in [1, r] \wedge q \in F_1 \cup F_1\} \cup \{q \xrightarrow{t_F,1^\bullet} q_0 \mid q \in F_1 \cup F_2\}$ is a set of "final" transitions for some newly introduced constant t_F.

Assume WLOG that $\mathcal{L}(\mathcal{A}_1) \not\subseteq \mathcal{L}(\mathcal{A}_2)$. Then, there is some transition path for $\mathcal{A}_1$ from (q_{01}, ρ_{01}) to some $q_1 \in F_1$ that, when simulated by $\mathcal{A}_2$ from (q_{02}, ρ_{02}), does not lead in F_2. For $\mathcal{A}$, this means that (q_{01}, ρ_{01}) and (q_{02}, ρ_{02}) are not bisimilar: Attacker can lead the game to a configuration pair $((q_1, \rho_1), (q_2, \rho_2))$, with $q_2 \in (Q_2 \setminus F_2) \cup \{q_s\}$, where he wins by playing some (t_F, a) from (q_1, ρ_1). By Proposition 4.18, Attacker has some strategy $\mathcal{T}$ of depth $O(rN^2 + r^3N)$ for winning the same game. We observe that, because $\mathcal{A}$ is saturated with sink transitions, the latter can only be achieved by Attacker being able to play a final transition with label (t_F, a) in one part of the game. Suppose the happens in the part starting from (q_{01}, ρ_{01}) and let $w\,(t_F, a)$ be the string accepted by the corresponding transition path, so $w \in \mathcal{L}(\mathcal{A}_1)$. By determinacy of $\mathcal{A}_2$, $w \notin \mathcal{L}(\mathcal{A}_2)$. □

Theorem 5.2. *Language inequivalence for deterministic RA($S\#_0$) is NP-complete.*

[5] The converse may fail due to transitions of $\mathcal{A}$ not being fireable in $\mathcal{S}(\mathcal{A})$.

Proof. Membership in NP is achieved via Lemma 5.1. NP-hardness follows from NP-completeness of language non-emptiness for deterministic RA($S\#_0$) [SI00]. □

6. NP bound for single assignment with filled registers (RA(SF))

In Section 4 we showed, in the setting with single assignment and no erasures (denoted by RA($S\#_0$)) the bisimilarity problem was solvable in polynomial space. Here we show that a further improvement is possible in the RA(SF) case, i.e. if the registers are required to be filled from the very start. We shall show an NP upper bound.

We start off with a series of results aiming to identify succinct (polynomial-size) sets of generators for $\overset{\mathsf{s}}{\sim}$, which we shall call *generating systems*. In Section 4 we already found that parts of $\overset{\mathsf{s}}{\sim}$ exhibit group-theoretic structure. Namely, Lemma 4.15 shows that, for any $p \in Q$ and $S \subseteq [1,r]$, $\mathcal{G}^p_S(\overset{\mathsf{s}}{\sim}) = \{\sigma \cap (X^p_S \times X^p_S) \,|\, (p,S) \overset{\mathsf{s}}{\sim}_\sigma (p,S)\}$ is a group, where $X^p_S(\overset{\mathsf{s}}{\sim}) \subseteq S$ is the characteristic set of (p,S).

Note that, for RA(SF), we only have the case $S = [1,r]$. Furthermore, $\overset{\mathsf{s}}{\sim}$ will be the only closed relation that we shall consider. For these reasons, we write simply X^p for characteristic set $X^p_{[1,r]}(\overset{\mathsf{s}}{\sim})$ and $\mathcal{G}^p$ for group $\mathcal{G}^p_{[1,r]}(\overset{\mathsf{s}}{\sim})$.

The group-theoretic structure implies that $\mathcal{G}^p$ can be generated by linearly many generators with respect to r.

Lemma 6.1 [MN87]**.** *Every subgroup of $\mathcal{S}_n$ has a generating set with at most* $\max(2, \lfloor \frac{n}{2} \rfloor)$ *elements.*

To handle the more general case $(p,S) \overset{\mathsf{s}}{\sim}_\sigma (q,S)$ of different states, consider

$$\mathcal{K}^{p,q} = \{\sigma \cap (X^p \times X^q) \mid (p,[1,r]) \overset{\mathsf{s}}{\sim}_\sigma (q,[1,r])\}.$$

Observe that, for $\sigma_1, \sigma_2 \in \mathcal{K}^{p,q}$, we have $\sigma_2 = (\sigma_2; \sigma_1^{-1}); \sigma_1$, because $\sigma_1^{-1}; \sigma_1 = \mathsf{id}_{X^q}$. Moreover, $\sigma_2; \sigma_1^{-1} \in \mathcal{G}^p$, so σ_2 has been obtained from σ_1 and an element of $\mathcal{G}^p$. Consequently, in presence of generators of $\mathcal{G}^p$, one member of $\mathcal{K}^{p,q}$ suffices to generate the whole of $\mathcal{K}^{p,q}$ by composition. This observation motivates the following definition of a generating system.

Definition 6.2. A ***generating system*** $\mathcal{G}$ consists of:

- a partitioning of Q into $P_1, \cdots, P_k$;
- for each partition P_i, a single representative $p_i \in P_i$ and:
  - a characteristic set $X^{p_i} \subseteq [1,r]$;
  - a set G^{p_i}, of up to $\max(2, \lfloor \frac{r}{2} \rfloor)$ permutations $\sigma \in \mathcal{S}_{X^{p_i}}$;
  - for each $q \in P_i \setminus \{p_i\}$, a partial permutation $\mathsf{ray}^{p_i}_q \in \mathcal{IS}_{[1,r]}$ such that $\mathsf{dom}(\mathsf{ray}^{p_i}_q) = X^{p_i}$; for technical convenience, we also add $\mathsf{ray}^{p_i}_{p_i} = \mathsf{id}_{X^{p_i}}$.

We write $rep(\mathcal{G})$ for the set $\{p_1, \cdots, p_k\}$ of representatives.

A generating system is used to generate a relation $gen(\mathcal{G}) \subseteq (Q \times \{[1,r]\} \times \mathcal{IS}_r \times Q \times \{[1,r]\})$ as follows. First, set

$$\begin{aligned}\mathrm{BASE}_{\mathcal{G}} = &\{(p_i,[1,r],\sigma,p_i,[1,r]) \,|\, p_i \in rep(\mathcal{G}), \sigma \in G^{p_i}\} \\ &\cup \{(p_i,[1,r],\mathsf{ray}^{p_i}_q,q,[1,r]) \,|\, p_i \in rep(\mathcal{G}), q \in P_i\}\end{aligned}$$

and then take $gen(\mathcal{G}) = Cl(\mathrm{BASE}_{\mathcal{G}})$.

Lemma 6.3. *There exists a generating system $\mathcal{G}$ such that $gen(\mathcal{G}) = \overset{\mathsf{s}}{\sim}$.*

Proof. We partition Q into equivalence classes defined by: $p \sim q$ if and only if there exists σ such that $(p, [1,r], \sigma, q, [1,r]) \in \overset{\mathsf{s}}{\sim}$. For each equivalence class P_i, we pick a single member p_i arbitrarily and let G^{p_i} consist of the generators of $\mathcal{G}^{p_i}$ provided by Lemma 6.1. Consider $q \in P_i \setminus \{p_i\}$. Because $q \in P_i$, there exists σ such that $(p_i, [1,r], \sigma, q, [1,r]) \in \overset{\mathsf{s}}{\sim}$. Then we can take $\mathsf{ray}_q^{p_i} = \sigma \cap (X^{p_i} \times [1,r])$. By the previous discussion, this delivers the sought generating system. □

Lemma 6.4. *For any generating system $\mathcal{G}$, membership in $gen(\mathcal{G})$ can be determined in polynomial time.*

Proof. To determine whether $(q_1, [1,r], \sigma, q_2, [1,r]) \in gen(\mathcal{G})$, we proceed as follows. If q_1, q_2 belong to different partitions we return NO. Suppose $q_1, q_2 \in P_i$. Recall that $\mathrm{BASE}_{\mathcal{G}}$ contains $(p_i, [1,r], \mathsf{ray}_{q_j}^{p_i}, q_j, [1,r])$ with $\mathsf{dom}(\mathsf{ray}_{q_j}^{p_i}) = X^{p_i}$. Then $(q_1, [1,r], \sigma, q_2, [1,r]) \in gen(\mathcal{G})$ is equivalent to $(p_i, [1,r], \sigma', p_i, [1,r]) \in gen(\mathcal{G})$, where $\sigma' = \mathsf{ray}_{q_1}^{p_i}; \sigma; (\mathsf{ray}_{q_2}^{p_i})^{-1}$. This is in turn equivalent to $\sigma' \cap (X^{p_i} \times X^{p_i})$ being generated from permutations in G^{p_i}. That the latter problem is solvable in polynomial time is a well-known result in computational group theory [FHL80]. □

Theorem 6.5. *$\sim$-RA(SF) is in NP.*

Proof. First we guess a generating system $\mathcal{G}$ and verify whether $gen(\mathcal{G})$ is a bisimulation. By Lemma 6.3, there exists at least one generating system with this property. Because generating systems involve polynomially many components of polynomial size, they can be guessed in polynomial time. Next, in order to check whether the guessed generating system generates a bisimulation, we need to verify the (SyS) conditions (for $S_1 = S_2 = [1,r]$) for each of the polynomially many elements of $\mathrm{BASE}_{\mathcal{G}}$. Note that this will involve polynomially many membership tests for $gen(\mathcal{G})$, each of which can be performed in polynomial time by Lemma 6.4. If the guess leads to a non-bisimulation, we return NO. Otherwise, we use another membership test for $gen(\mathcal{G})$ to check whether the given instance of the bisimilarity problem belongs to $gen(\mathcal{G})$. We return the outcome of that test as the final result. □

Remark 6.6. Note that symbolic bisimulations are based on *partial finite permutations*, which form inverse semigroups. Consequently, inverse semigroup-theoretic structure could seem the most natural kind of structure with which to approach our problems. Unfortunately, inverse semigroups do not admit analogous results.

- There exist inverse subsemigroups of $\mathcal{IS}_n$ that require $\binom{n}{\frac{n}{2}} \approx 2^n \sqrt{\frac{2}{\pi n}}$ generators, e.g. $\{\mathsf{id}_X | X \subseteq [1,n], |X| = \frac{n}{2}\}$.
- It is possible to show that the membership problem for inverse subsemigroups of $\mathcal{IS}_n$ is PSPACE-complete, sharpening a result of Kozen [Koz77]. We present the argument in Appendix D.

Consequently, we were forced to look a bit deeper, and base generating systems on groups.

Remark 6.7. Note that we do not have a matching lower bound for RA(SF), which raises the intriguing prospect that there may still be scope for improvement in this case. A closely related problem to $\sim$-RA(SF) is *graph automorphism (GA)*, i.e. given a graph G decide whether it has a non-trivial automorphism. While it is easy to see that GA is in NP, it is not known whether it is in P or, for that matter, in coNP. We can reduce graph-automorphism to the following problem in our setting: given a DRA(SF) $\mathcal{A}$ (without locally fresh transitions) and a configuration (q, ρ), is there a non-identity permutation π such that $(q, \rho) \sim (q, \rho \circ \pi)$?

This observation introduces a possible barrier to methods we can pursue to efficiently solve $\sim$-RA(SF), such as partition refinement, which aim to construct a representation of the whole bisimilarity relation.

7. Fresh-register automata with single assignment without erasure (FRA($S\#_0$))

In this section we examine the problems tackled in Sections 4-6 albeit in the general case of FRAs. We would like to apply the same techniques, aiming to produce the same upper bounds, yet the FRA setting raises significant additional challenges. Our approach for RAs relied on symbolic bisimulations and the group-theoretic structure that emanated from them. While we can express bisimilarity in FRAs symbolically following [Tze11], we shall see that such symbolic bisimulations do not support the group-theoretic representations. The reason is the treatment of the history of the computation, which affects bisimilarity in subtle ways, especially in the initial stages of the bisimulation game. In those stages, global and local freshness can inter-simulate another, under certain conditions, which leads us to extending our symbolic representations beyond the r names that each system can have in its registers.

Simplified notation. We extend the simplified notation for RA($S\#_0$) by including transition labels for global freshness. Recall that, in any transition $q_1 \xrightarrow{t,X,i,Z} q_2$ of an r-FRA($S\#_0$), we have that $X \in \{\circledast, \emptyset\} \cup \{\{j\} \mid j \in [1,r]\}$, $Z = \emptyset$ and $X = \{j\}$ implies $i = 0$. We thus follow a simpler notation for transitions, with $\delta \subseteq Q \times \Sigma \times ([1,r] \cup \{i^\bullet, i^\circledast \mid i \in [0,r]\}) \times Q$:

(a) we write each transition $q_1 \xrightarrow{t,\{i\},0,\emptyset} q_2$ as $q_1 \xrightarrow{t,i} q_2$;
(b) and each $q_1 \xrightarrow{t,\emptyset,i,\emptyset} q_2$ as $q_1 \xrightarrow{t,i^\bullet} q_2$;
(c) and each $q_1 \xrightarrow{t,\circledast,i,\emptyset} q_2$ as $q_1 \xrightarrow{t,i^\circledast} q_2$.

(a),(b) are as in RA($S\#_0$). In (c), the automaton reads (t,a) if a is *globally fresh*, i.e. it has not appeared in the history so far, and stores it in register i. Formally, $q \xrightarrow{t,i^\circledast} q'$ can induce a transition $(q,\rho,H) \xrightarrow{t,a} (q', \rho[i \mapsto a], H \cup \{a\})$ just if $a \notin H$.[6]

Assignment pre-updates. Recall the operations we introduced in Section 2.4 on partial bijections and in particular the pre-composing of generalised swaps (i.e. $[i \leftrightarrow j]$ with $i,j \in [0,r]$). We extend this operation to register assignments by setting:

$$\rho[i \leftrightarrow j] = \begin{cases} (i\,j);\rho & \text{if } i,j \in [1,r] \\ \rho & \text{otherwise} \end{cases}$$

We can then show the following.

Lemma 7.1. *Given r-register assignments ρ_1, ρ_2 (of S-type) and $i, i', j, j' \in [0,r]$:*

(1) $[j \leftrightarrow j'](\rho_1; \rho_2^{-1})[i \leftrightarrow i'] = \rho_1[i \leftrightarrow i']; \rho_2[j \leftrightarrow j']^{-1}$;
(2) *for any $a \in \mathcal{D}$ such that $a \in \mathsf{rng}(\rho_1) \implies a = \rho_1(i')$ and $a \in \mathsf{rng}(\rho_2) \implies a = \rho_2(j')$ we have $[j \leftrightarrow j']((\rho_1; \rho_2^{-1})[i' \mapsto j'])[i \leftrightarrow i'] = \rho_1[i' \mapsto a][i \leftrightarrow i']; \rho_2[j' \mapsto a][j \leftrightarrow j']^{-1}$.*

[6]The latter condition above is slightly different but equivalent to that used in [Tze11]. In *loc. cit.*, the names of ρ are not necessarily included in H and hence in this rule one stipulates that $a \notin \mathsf{rng}(\rho) \cup H$.

7.1. **Symbolic bisimulation.** Recall that, in the case of RAs, we were able to capture bisimilarity symbolically by using tuples of the form $(q_1, S_1, \sigma, q_2, S_2)$, whereby S_k represented $\mathsf{dom}(\rho_k)$ of the actual configuration (q_k, ρ_k) being represented (for $k = 1, 2$), and partial bijection $\sigma : S_1 \to S_2$ captured the matching names of ρ_1 and ρ_2. Moving to FRAs, the first obstacle we face is that actual configurations contain the full history of names and have therefore unbounded size. For bisimulation purposes, though, keeping track of the whole history, or its size, is not necessary. In fact, history only plays a role in globally fresh transitions and one can easily see that the rule

> "Every globally fresh transition from q_1 must be matched by a globally or a locally fresh transition from q_2."

is sound for simulation of globally fresh transitions.

However, global freshness leads to complications in the simulation of locally fresh transitions. For example, consider configurations $(q_1, \rho_1, H), (q_2, \rho_2, H)$ with $H = \{d_1, d_2\}$ and a transition $q_1 \xrightarrow{t,1^\bullet} q_1'$. We look at three scenarios:

(1) If $\mathsf{rng}(\rho_1) = \{d_1, d_2\}$, then the transition from q_1 can be matched by some $q_2 \xrightarrow{t,1^\circledast} q_2'$, as the local names of q_1 coincide with all the names in H.
(2) If $\mathsf{rng}(\rho_1) = \{d_1\}$ and $\rho_2 = \{(1, d_2)\}$, then the transition from q_1 cannot be matched by some $q_2 \xrightarrow{t,1^\circledast} q_2'$ alone, unless there is also a transition $q_2 \xrightarrow{t,1} q_2''$ (to capture the fact that $q_1 \xrightarrow{t,1^\bullet} q_1'$ can produce d_2).
(3) On the other hand, if $\mathsf{rng}(\rho_1) = \mathsf{rng}(\rho_2) = \{d_1\}$ then q_2 must use a locally fresh transition in order to match the transition from q_1 (as the latter can produce d_2).

More generally, if $|H| > 2r$ then there will be some $d \in H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$, which makes impossible for locally fresh transitions in one system to be matched by globally fresh transitions in the other one.

Thus, under certain circumstances which include the fact that $|H| \leq 2r$, local freshness can be captured by global freshness and some known-name transitions. To accommodate this feature, we will design symbolic bisimulations with an additional component $h \in [0, 2r] \cup \{\infty\}$ that will abstract the size of $|H|$. The value $h = \infty$ will signify that $|H| > 2r$ and therefore local-fresh cannot be matched by global-fresh. On the other hand, $h \leq 2r$ will mean that $|H| = h \leq 2r$ and therefore extra cases need to be considered for fresh transitions. For $h \leq 2r$, we will consider symbolic configurations (q_i, S_i) $(i = 1, 2)$ where $S_i \subseteq [1, 3r]$ and $h = |S_i|$, related by bijections $\sigma : S_1 \to S_2$.

- The component $S_i \cap [1, r]$ of S_i will still represent the domain of ρ_i.
- The complementary part $S_i \setminus [1, r]$ will represent the remaining names, those that have passed but no longer reside in ρ_i (i.e. $H \setminus \mathsf{rng}(\rho_i)$), in some canonical fashion.

Effectively, the above will allow us to symbolically represent the history of each FRA, up to the size $2r$, in an ordered way. It will also offer us a way to decide the simulation game for locally fresh transitions. Let us suppose that one system performs a transition $q_1 \xrightarrow{t,i^\bullet} q_1'$:

1. Such a transition can capture any name d that is represented in some $i' \in S_1 \setminus [1, r]$. If $\sigma(i') \in [1, r]$ then the other system has the name in its registers and can (only) capture it by some $q_2 \xrightarrow{t,\sigma(i')} q_2'$.

2. If $\sigma(i') \in S_2 \setminus [1,r]$ then the name is historical and the other system does not currently have it in its registers. It is therefore obliged to simulate by some locally fresh transition $q_2 \xrightarrow{t,j^\bullet} q_2'$.
3. The transition can also capture any name d that is not in H and, in this case, the other system can capture it by any $q_2 \xrightarrow{t,j^\bullet/j^\circledast} q_2'$. Moreover, such a simulation step would increase the size of h by one.

We therefore formulate symbolic bisimulation as follows.

Definition 7.2. Let $\mathcal{A} = \langle Q, \Sigma, \delta\rangle$ be an r-FRA($S\#_0$). We first set:

$$\mathcal{U}_0 = Q \times \mathcal{P}([1,3r]) \times \mathcal{IS}_{3r} \times Q \times \mathcal{P}([1,3r]) \times ([0,2r] \cup \{\infty\})$$

$$\mathcal{U} = \{(q_1, S_1, \sigma, q_2, S_2, h) \in \mathcal{U}_0 \mid \sigma \subseteq S_1 \times S_2 \wedge (h \leq 2r \implies |\sigma| = |S_1| = |S_2| = h) \\ \wedge (h = \infty \implies \sigma \in \mathcal{IS}_r \wedge S_1, S_2 \subseteq [1,r])\}$$

A *symbolic simulation* on $\mathcal{A}$ is a relation $R \subseteq \mathcal{U}$, with membership $(q_1, S_1, \sigma, q_2, S_2, h) \in R$ often written $(q_1, S_1)\, R^h_\sigma\, (q_2, S_2)$, such that all $(q_1, S_1, \sigma, q_2, S_2, h) \in R$ satisfy the following *fresh symbolic simulation conditions* (FSyS):[7,8]

(a) for all $q_1 \xrightarrow{t,i} q_1'$,
   1. if $\sigma(i) \in [1,r]$ then there is $q_2 \xrightarrow{t,\sigma(i)} q_2'$ with $(q_1', S_1)\, R^h_\sigma\, (q_2', S_2)$,
   2. if $\sigma(i) = j' \in [r{+}1, 3r]$ then there is $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1)\, R^h_{[j \leftrightarrow j']\sigma}\, (q_2', S_2[j \leftrightarrow j'])$,
   3. if $i \in S_1 \setminus \mathsf{dom}(\sigma)$ then there is $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1)\, R^h_{\sigma[i \mapsto j]}\, (q_2', S_2[j])$;

(b) for all $q_1 \xrightarrow{t,i^\bullet} q_1'$, $i' \in S_1 \setminus [1,r]$ and $j \in S_2 \setminus \mathsf{rng}(\sigma)$,
   1. if $\sigma(i') \in [1,r]$ then there is $q_2 \xrightarrow{t,\sigma(i')} q_2'$ with $(q_1', S_1[i \leftrightarrow i'])\, R^h_{\sigma[i \leftrightarrow i']}\, (q_2', S_2)$,
   2. if $\sigma(i') = j' \in [r{+}1, 3r]$ then there is $q_2 \xrightarrow{t,j^\bullet} q_2'$ with
   $$(q_1', S_1[i \leftrightarrow i'])\, R^h_{[j \leftrightarrow j']\sigma[i \leftrightarrow i']}\, (q_2', S_2[j \leftrightarrow j']),$$
   3. there exists $q_2 \xrightarrow{t,j} q_2'$ with $(q_1', S_1[i])\, R^h_{\sigma[i \mapsto j]}\, (q_2', S_2)$;

(c) for all $q_1 \xrightarrow{t,\ell_1} q_1'$ with $\ell_1 \in \{i^\bullet, i^\circledast\}$ there is some $q_2 \xrightarrow{t,\ell_2} q_2'$ with $\ell_2 \in \{j^\bullet, j^\circledast\}$ and,
   1. if $h < 2r$ then, taking $i' = \min([r{+}1, 3r] \setminus S_1)$ and $j' = \min([r{+}1, 3r] \setminus S_2)$, we have $(q_1', S_1[i'][i \leftrightarrow i'])\, R^{h+1}_{[i \leftrightarrow i'](\sigma[i' \mapsto j'])[j \leftrightarrow j']}\, (q_2', S_2[j'][j \leftrightarrow j'])$;
   2. if $h = 2r$ then $(q_1', S_1[i] \cap [1,r])\, R^\infty_{\sigma[i \mapsto j] \cap [1,r]^2}\, (q_2', S_2[j] \cap [1,r])$;
   3. if $h = \infty$ then $(q_1', S_1[i])\, R^\infty_{\sigma[i \mapsto j]}\, (q_2', S_2[j])$ and if $\ell_1 = i^\bullet$ then $\ell_2 = j^\bullet$.

Define the inverse of R by:

$$R^{-1} = \{\, (q_2, S_2, \sigma^{-1}, q_1, S_1, h) \mid (q_1, S_1, \sigma, q_2, S_2, h) \in R \,\}$$

and call R a ***symbolic bisimulation*** if both R and R^{-1} are symbolic simulations. We let *s-bisimilarity*, denoted $\overset{\mathsf{s}}{\sim}$, be the union of all symbolic bisimulations.

We define a sequence of ***indexed bisimilarity*** relations $\overset{i}{\sim} \subseteq \mathcal{U}$ inductively as follows. We

[7] We say that $(q_1, S_1, \sigma, q_2, S_2, h)$ *satisfies the* (FSyS) *conditions in* R.

[8] Note how the (FSyS) conditions are divided with respect to the value of h: conditions (a2), (b1), (b2), (c1) and (c2) all require $h \leq 2r$; while conditions (a3), (b3) and (c3) are for $h = \infty$. On the other hand, (a1) applies to all h.

let $\overset{0}{\sim}$ be the whole of $\mathcal{U}$. Then, for all $i \in \omega$ and $h \in [0, 2r] \cup \{\infty\}$, $(q_1, S_1)\,(\overset{i+1}{\sim})^h_\tau\,(q_2, S_2)$ just if both $(q_1, S_1, \tau, q_2, S_2, h)$ and $(q_2, S_2, \tau^{-1}, q_1, S_1, h)$ satisfy the (FSyS) conditions in $\overset{i}{\sim}$.

Let $\kappa_i = (q_i, \rho_i, H)$ $(i = 1, 2)$ be configurations with common history H and let $n = |H|$. Their symbolic representation will depend on n. We take $\mathsf{symb}(\kappa_1, \kappa_2) \subseteq \mathcal{U}$ to be:

$$\mathsf{symb}(\kappa_1, \kappa_2) = \begin{cases} \{(q_1, \mathsf{dom}(\hat\rho_1), \hat\rho_1; \hat\rho_2^{-1}, q_2, \mathsf{dom}(\hat\rho_2), n) \in \mathcal{U} \mid \theta(\hat\rho_1, \hat\rho_2)\} & n \le 2r \\ \{(q_1, \mathsf{dom}(\rho_1), \rho_1; \rho_2^{-1}, q_2, \mathsf{dom}(\rho_2), \infty)\} & n > 2r \end{cases}$$

where $\theta(\hat\rho_1, \hat\rho_2)$ is the condition stipulating that $\hat\rho_i$ range over all $3r$-register assignments of type $S\#_0$ such that $\mathsf{rng}(\hat\rho_i) = H$ and $\hat\rho_i \restriction [1, r] = \rho_i$, for $i = 1, 2$. In particular, $\mathsf{symb}(\kappa_1, \kappa_2)$ is singleton in case $n > 2r$ but not necessarily so if $n \le 2r$. The following lemma ensures that, with respect to bisimilarity, the specific choice of element from $\mathsf{symb}(\kappa_1, \kappa_2)$ is not important.

Lemma 7.3. *For all κ_1, κ_2 as above, if $|H| < 2r$ then either $\mathsf{symb}(\kappa_1, \kappa_2) \subseteq \overset{\mathsf{s}}{\sim}$ or $\mathsf{symb}(\kappa_1, \kappa_2) \cap \overset{\mathsf{s}}{\sim} = \emptyset$.*

Definition 7.4. We say that κ_1 and κ_2 are *s-bisimilar*, written $\kappa_1 \overset{\mathsf{s}}{\sim} \kappa_2$, if $\mathsf{symb}(\kappa_1, \kappa_2) \subseteq \overset{\mathsf{s}}{\sim}$.

Remark 7.5. The definition of symbolic bisimulation we give here is substantially more fine-grained than the one in [Tze11]. Although in *loc. cit.* the symbolic bisimulation is also given parametrically to the size of the history h (up to the given bound[9]), for $h \le 2r$ that formulation is simplistic in that it only keeps track of names that reside in registers of the automata,[10] which in turn prohibits us to derive $(q_1, S_1)\,R^h_{\sigma_1;\sigma_2}\,(q_3, S_3)$ from $(q_1, S_1)\,R^h_{\sigma_1}\,(q_2, S_2)$ and $(q_2, S_2)\,R^h_{\sigma_2}\,(q_3, S_3)$ and apply the group-theoretic approach.

Using the intuition described above about the bounded representation of histories, we can show the following correspondence. Similarly to Lemma 4.5, the proof of the next lemma is based on matching concrete and symbolic bisimulations and doing a careful, if somewhat tedious, case analysis of possible transitions in each case.

Lemma 7.6. *Let κ_1 and κ_2 be configurations of an r-FRA($S\#_0$). Then $\kappa_1 \sim \kappa_2 \iff \kappa_1 \overset{\mathsf{s}}{\sim} \kappa_2$.*

Lemma 7.7. *For all $i \in \omega$, $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$ and $(\bigcap_{i\in\omega} \overset{i}{\sim}) = \overset{\mathsf{s}}{\sim}$.*

Similarly to symbolic bisimulations for $\mathrm{RA}(S\#_0)$, we have the following closure properties. Given $R \subseteq \mathcal{U}$ we split R into *components*:

$$R = \sum\nolimits_{h\in[0,2r]\cup\{\infty\}} R^h$$

where $R^h = \{(q_1, S_1, \sigma, q_2, S_2) \mid (q_1, S_1, \sigma, q_2, S_2, h) \in R\}$. We now write $Cl(R)$ for the componentwise closure of R with respect to identity, symmetry, transitivity and extension of partial permutations, i.e. $Cl(R) = \sum_{h\in[0,2r]\cup\{\infty\}} Cl(R^h)$.

The following lemma will play a key role in the forthcoming technical development. It is proved similarly to Lemma 4.8, i.e. by showing that the (FSyS) rules are compatible with the closure rules. While the proof is longer, there is no essential novelty: the approach is similar, only the case analysis required is more extensive.

[9] In fact, the bound used in [Tze11] is smaller $(2r-1)$, due to the fact that it examines bisimulation between configurations with common initial names.

[10] that is, in $(q_1, S_1)\,R^h_\sigma\,(q_2, S_2)$ we always have $S_1, S_2 \subseteq [1, r]$.

Lemma 7.8. *Let $R, P \subseteq \mathcal{U}$. If all $g \in R \cup R^{-1}$ satisfy the* (FSyS) *conditions in P then all $g \in Cl(R)$ satisfy the* (FSyS) *conditions in $Cl(P)$.*

Proposition 7.9. *Symbolic bisimilarity and indexed symbolic bisimilarity for FRA($S\#_0$) are closed.*

(1) $Cl(\overset{\mathsf{s}}{\sim}) = \overset{\mathsf{s}}{\sim}$;

(2) *for all $i \in \omega$:* $\overset{i}{\sim} = Cl(\overset{i}{\sim})$.

Proof. For $Cl(\overset{\mathsf{s}}{\sim}) = \overset{\mathsf{s}}{\sim}$, since $\overset{\mathsf{s}}{\sim}$ is symmetric and satisfies the (FSyS) conditions in itself, from the previous lemma we have that $Cl(\overset{\mathsf{s}}{\sim})$ satisfies the (FSyS) conditions in itself and is therefore a symbolic bisimulation. Thus, $Cl(\overset{\mathsf{s}}{\sim}) \subseteq \overset{\mathsf{s}}{\sim}$.

For $Cl(\overset{i}{\sim}) = \overset{i}{\sim}$ we do induction on i. When $i = 0$ then the result follows from the fact that $\overset{0}{\sim}$ is the universal relation. For the inductive case, note first that $\overset{i+1}{\sim}$ is symmetric by construction and all $g \in \overset{i+1}{\sim}$ satisfy the (FSyS) conditions in $\overset{i}{\sim}$. Hence, by Lemma 7.8, all elements of $Cl(\overset{i+1}{\sim})$ satisfy the (FSyS) conditions in $Cl(\overset{i}{\sim})$. By IH, $Cl(\overset{i}{\sim}) = \overset{i}{\sim}$ so $Cl(\overset{i+1}{\sim}) \subseteq \overset{i+1}{\sim}$, as required. □

More explicitly, the last part of Proposition 7.9 means that, given $(q_1, S_1)\ (\overset{i}{\sim})^h_\tau\ (q_2, S_2)$:

(1) Then, $(q_2, S_2)\ (\overset{i}{\sim})^h_{\tau^{-1}}\ (q_1, S_1)$.

(2) For all τ', if $\tau \leq_{S_1,S_2} \tau'$ then $(q_1, S_1)\ (\overset{i}{\sim})^h_{\tau'}\ (q_2, S_2)$.

(3) For all $(q_2, S_2)\ (\overset{i}{\sim})^h_{\tau'}\ (q_3, S_3)$, $(q_1, S_1)\ (\overset{i}{\sim})^h_{\tau;\tau'}\ (q_3, S_3)$.

We therefore observe that the extension of symbolic representations to the size $3r$, and the ensuing history representation up to size $2r$ along with the extended symbolic bisimulation conditions, have paid off in yielding the desired closure properties. The group-theoretic behaviour of a closed relation R differs between different components:

- R^∞ has the same structure as the closed relations R examined in Section 4.2.
- For $h \in [0, 2r]$, the tuples $(q_1, S_1, \sigma, q_2, S_2) \in R^h$ respect the condition $|S_1| = |S_2| = |\sigma| = h$. In particular, σ is a bijection from S_1 to S_2 and, hence, in this case closure under extension is trivial, and so are characteristic sets ($X^p_S(R^h) = S$). Moreover, $\sigma \in \mathcal{IS}_{3r}$ and $S_1, S_2 \subseteq [1, 3r]$.

We can hence see that the same groups arise as in the case of RA($S\#_0$), and actually simpler in the case $h \in [0, 2r]$, albeit parameterised over h. This allows for a similar group-theoretic treatment.

7.2. **PSPACE bound for bisimulation game.** Before we come to the proof of the main result, recall Theorem 4.11 which says that, for $n \geq 2$, the length of every subgroup chain in $\mathcal{S}_{[1,n]}$ is at most $2n - 3$.

Lemma 7.10. *Let $h \in [0, 2r] \cup \{\infty\}$, $S_1, S_2 \subseteq [1, 3r]$ and $\mathcal{U}^h_{S_1,S_2} = Q \times \{S_1\} \times \mathcal{IS}_r \times Q \times \{S_2\} \times \{h\}$. Then the sub-chain $\{\overset{i}{\sim} \mid (\overset{i+1}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2})\}$ has size $O(|Q|^2 + r^2|Q|)$.*

Proof. We argue that $\{\overset{i}{\sim} \mid (\overset{i+1}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2})\}$ has at most $|Q|^2 + r^2|Q| - 2r|Q|$ elements. We shall say that (q_1, S_1, h, q_2, S_2) is *separated* in $\overset{i}{\sim}$ if there is no σ such that

(q_1, S_1) $(\overset{i}{\sim})^h_\sigma$ (q_2, S_2); we say it is *unseparated* otherwise. We claim that if $(\overset{i+1}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2})$ then there is some $q \in Q$ and $S \in \{S_1, S_2\}$ such that

(i) either $X^q_S((\overset{i}{\sim})^h) \subsetneq X^q_S((\overset{i+1}{\sim})^h)$
(ii) or $\mathcal{G}^q_S(\overset{i+1h}{\sim})$ is a strict subgroup of $\mathcal{G}^q_S(\overset{i}{\sim}{}^h)$
(iii) or there is a tuple (q_1, S_1, h, q_2, S_2) that is unseparated in $\overset{i}{\sim}$ and becomes separated in $\overset{i+1}{\sim}$.

We reason as follows. If $(\overset{i+1}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}) \subsetneq (\overset{i}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2})$ then there are $q_1, q_2 \in Q$ and σ such that (q_1, S_1) $(\overset{i}{\sim})^h_\sigma$ (q_2, S_2) but not (q_1, S_1) $(\overset{i+1}{\sim})^h_\sigma$ (q_2, S_2). From closure properties for $\overset{i}{\sim}, \overset{i+1}{\sim}$ it follows that (q_1, S_1) $(\overset{i}{\sim})^h_{\sigma'}$ (q_2, S_2) and not (q_1, S_1) $(\overset{i+1}{\sim})^h_{\sigma'}$ (q_2, S_2), where $\sigma' = \sigma \cap (X^{q_1}_{S_1}((\overset{i}{\sim})^h) \times X^{q_2}_{S_2}((\overset{i}{\sim})^h))$. Consequently, we can assume wlog that $\mathsf{dom}(\sigma) = X^{q_1}_{S_1}((\overset{i}{\sim})^h)$ and $\mathsf{rng}(\sigma) = X^{q_2}_{S_2}((\overset{i}{\sim})^h)$. Now, suppose that, for all $q \in Q$, $S \in \{S_1, S_2\}$, we have $X^q_S((\overset{i+1}{\sim})^h) = X^q_S((\overset{i}{\sim})^h)$ (i.e. not (i)) and that no previously unseparated tuple becomes separated in $\overset{i+1}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}$ (i.e. not (iii)). From the latter, It follows that there is some τ such that (q_1, S_1) $(\overset{i+1}{\sim})^h_\tau$ (q_2, S_2). Hence, $\sigma; \tau^{-1} \in \mathcal{G}^{q_1}_{S_1}(h, i)$ but $\sigma; \tau^{-1} \notin \mathcal{G}^{q_1}_{S_1}(h, i+1)$ so that $\mathcal{G}^{q_1}_{S_1}((\overset{i}{\sim})^h) > \mathcal{G}^{q_1}_{S_1}((\overset{i+1}{\sim})^h)$.

Because $X^q_S((\overset{i}{\sim})^h) \subseteq X^q_S((\overset{i+1}{\sim})^h)$, (i) may happen at most $2r|Q|$ times in the whole chain. For fixed $X^q_S((\overset{i}{\sim})^h)$, by Theorem 4.11, (ii) may happen at most $2r-2$ times (we include the case $r = 1$), which gives an upper bound of $2r|Q|(2r-2)$ for the number of such changes inside the whole chain (under the assumption that the changes are not of type (i), which have already been counted). Finally, the remaining changes must be of type (iii) and may happen at most $|Q|^2$ times across the whole chain. Overall, we obtain $2r|Q| + 2r|Q|(2r-2) + |Q|^2 = |Q|^2 + 4r^2|Q| - 2r|Q|$ as a bound on the length of the given chain. □

Given $S_1, S_2 \subseteq [1, 3r]$ and $h \in [0, 2r] \cup \{\infty\}$, let us call the triple (S_1, S_2, h) ***proper*** just if: either $|S_1| = |S_2| = h$, or $h = \infty$ and $S_1, S_2 \subseteq [1, r]$. For such (S_1, S_2, h), let us define:

$$\hat{\gamma}(S_1, S_2, h) = \begin{cases} \gamma(S_1 \cap [1, r], S_2 \cap [1, r]) + h & \text{if } h \in [0, 2r] \\ \gamma(S_1, S_2) + 2r + 1 & \text{if } h = \infty \end{cases}$$

The measure $\hat{\gamma}$ enables us to show the following bound for stabilising indexed bisimulation, proven similarly to Lemma 4.17.

Lemma 7.11. *Let ℓ be the bound from Lemma 7.10 and $B = (4r+2)\ell$. For any proper (S_1, S_2, h), we have $\overset{B}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2} = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}$.*

Proof. Observe that $0 \leq \hat{\gamma}(S_1, S_2, h) \leq 4r + 1$. For each $m \in [0, 4r+1]$, let

$$k_m = \min\{i \mid \overset{i}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2} = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2} \text{ for any } S_1, S_2, h \text{ with } \hat{\gamma}(S_1, S_2, h) \geq m\}.$$

Consider S_1, S_2, h with $\hat{\gamma}(S_1, S_2, h) \geq m$, where $m < 4r + 1$.

Observe that, for $k \geq k_{m+1}$, if $\overset{k}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2} = \overset{k+1}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}$, then we must have $\overset{k}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2} = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}$, because the (FSyS) conditions for (S_1, S_2, h) refer to either (S_1, S_2, h)

or (S_1', S_2', h') with $\hat{\gamma}(S_1', S_2', h') > \hat{\gamma}(S_1, S_2, h)$. Consequently, if $\overset{k}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2} \neq \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}$, the sequence $(\overset{k}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2})$ $(k = k_{m+1}, k_{m+1}+1, \cdots)$ must change in every step before stabilisation. By Lemma 7.10, at most ℓ extra steps from $\overset{k_{m+1}}{\sim}$ will be required to arrive at $\overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}^h_{S_1,S_2}$, which implies $k_m \leq k_{m+1} + \ell$. By a similar argument, we can conclude that $k_{4r+1} \leq \ell$. Consequently, $k_0 \leq (4r+2)\ell$, as required. □

We can therefore establish solvability in polynomial space.

Proposition 7.12. *For any FRA($S\#_0$) bisimulation problem, if there is a winning strategy for Attacker then there is one of depth $O(r|Q|^2 + r^3|Q|)$.*

Proposition 7.13. *$\sim$-FRA($S\#_0$) is in PSPACE.*

7.3. **Generating systems and NP routines.** We proceed to generating systems for FRA(SF), which are h-parameterised versions of the ones for RA(SF), except that now they are built over $[1, 3r]$ rather than $[1, r]$. Since we again consider only characteristic sets and groups with relation parameter $R = \overset{\mathsf{s}}{\sim}$, we will typically leave this argument implicit in what follows. We call a pair (S, h) proper just if (S, S, h) is proper.

Definition 7.14. A ***generating system*** $\mathcal{G}_{S,h}$ for proper (S, h) (in which case $|S| \leq 2r$), consists of:

- a partitioning of Q into $P_1, \cdots, P_k$;
- for each partition P_i, a single representative $p_i \in P_i$ and:
  - a characteristic set $X^{p_i}_{S,h} \subseteq S$;
  - a set $G^{p_i}_{S,h}$, of up to $\max(2,r)$ permutations $\sigma \in \mathcal{S}_{X^{p_i}_{S,h}}$;
  - for each $q \in P_i \setminus \{p_i\}$, a partial permutation $\mathsf{ray}^{p_i}_q \in \mathcal{IS}_S$ such that $\mathsf{dom}(\mathsf{ray}^{p_i}_q) = X^{p_i}_{S,h}$; for technical convenience, we also add $\mathsf{ray}^{p_i}_{p_i} = \mathsf{id}_{X^{p_i}_{S,h}}$.

We write $rep(\mathcal{G}_{S,h})$ for the set $\{p_1, \cdots, p_k\}$ of representatives.
From $\mathcal{G}_{S,h}$ we generate $gen(\mathcal{G}_{S,h}) \subseteq (Q \times \{S\} \times \mathcal{IS}_{3r} \times Q \times \{S\})$ by setting

$$\begin{aligned}\mathrm{BASE}_{\mathcal{G}_{S,h}} = {} & \{(p_i, S, \sigma, p_i, S) \,|\, p_i \in rep(\mathcal{G}_{S,h}) \wedge \sigma \in G^{p_i}_{S,h}\} \\ & \cup\, \{(p_i, S, \mathsf{ray}^{p_i}_q, q, S) \,|\, p_i \in rep(\mathcal{G}_{S,h}) \wedge q \in P_i\}\end{aligned}$$

and taking $gen(\mathcal{G}_{S,h}) = Cl(\mathrm{BASE}_{\mathcal{G}_{S,h}})$.

The following lemma, proved in the same way as Lemmata 6.3 and 6.4, enables us to prove an NP upper bound for bisimilarity in FRA(SF).

Lemma 7.15. (1) *For any proper (S, h) there exists a generating system $\mathcal{G}_{S,h}$ such that $gen(\mathcal{G}_{S,h}) = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}^h_{S,S}$.*
(2) *For any generating system $\mathcal{G}_{S,h}$, membership in $gen(\mathcal{G}_{S,h})$ can be determined in polynomial time.*

Theorem 7.16. *$\sim$-FRA(SF) is in NP.*

Proof. Given an input tuple $(q_1, S_1, \sigma, q_2, S_2, h^0)$, note first that $[1, r] \subseteq S_1, S_2$ (by F) and $|S_1| = |S_2|$. We can therefore convert it to an equivalent $(q_1, S_1', \sigma', q_2, S_2, h^0)$, with $S_1' = S_2$, by applying a permutation on the indices in $S_1 \setminus [1, r]$. Hence, we can assume wlog that our input is some $(q_1, S^0, \sigma, q_2, S^0, h^0)$. Moreover, because the expansion of S in the symbolic

bisimulation game (when $h \in [0, 2r]$) always occurs in its first free register ($\min([r{+}1, 3r] \setminus S)$), we can compute the sequence $(S^0, h^0, S^0), (S^1, h^0{+}1, S^1), \cdots$ of distinct triples considered in the game (in the $h \in [0, 2r]$ phase), which must thence be bounded in length by $2r$. Including the final bisimulation phase ($h = \infty$), this gives us $2r + 1$ phases. We first generate for each of them a generating system, say $\mathcal{G}_{S^i,h^i}$, and then verify whether each $gen(\mathcal{G}_{S^i,h^i})$ is a symbolic bisimulation, similarly to Theorem 6.5. Note that each such check can be achieved in polynomial time. If the guess leads to some $gen(\mathcal{G}_{S^i,h^i})$ being a non-symbolic-bisimulation, we return NO. Otherwise, we use another membership test for $gen(\mathcal{G}_{S^0,h^0})$ to check whether the given instance of the bisimilarity problem belongs to $gen(\mathcal{G}_{S^0,h^0})$. We return the outcome of that test as the final result. □

8. Visibly pushdown automata with single assignment and filled registers (VPDRA(SF))

Finally, we consider a variant of register automata with visible pushdown storage [AM04]. We only consider the most restrictive register discipline (SF), as undecidability will be shown to apply already in this case.

Definition 8.1. A ***visibly pushdown r-register automaton*** (r-VPDRA(SF)) $\mathcal{A}$ is a tuple

$$\langle Q, \Sigma_C, \Sigma_N, \Sigma_R, \Gamma, \delta \rangle,$$

where:

- Q is a finite set of states;
- Σ_C, Σ_N, Σ_R are disjoint finite sets of *push-*, *no-op-* and *pop-tags* respectively;
- Γ is a finite set of *stack tags*;
- $\delta = \delta_C \cup \delta_N \cup \delta_R$, the transitions, have $Lab = \{1, \ldots, r\} \cup \{1^\bullet, \ldots, r^\bullet\}$ and:
  - $\delta_C \subseteq Q \times \Sigma_C \times Lab \times \Gamma \times \{1, \cdots, r\} \times Q$
  - $\delta_N \subseteq Q \times \Sigma_N \times Lab \times Q$
  - $\delta_R \subseteq Q \times \Sigma_R \times Lab \times \Gamma \times \{1, \cdots, r, \bullet\} \times Q$

Configurations of r-VPDRA(SF) are triples (q, ρ, s), where $q \in Q$, ρ is a register assignment and $s \in (\Gamma \times \mathcal{D})^*$ is the stack. An LTS arises by having a labelled edge $(q_1, \rho_1, s_1) \xrightarrow{(t,d)} (q_2, \rho_2, s_2)$ just if there exist $i \in [1, r]$ and $l \in \{i, i^\bullet\}$ such that:

(1) $\rho_1(x) = \rho_2(x)$ for all $x \neq i$;
(2) if $l = i$ then $\rho_1(i) = \rho_2(i)$, otherwise $\rho_2(i) \notin \mathsf{rng}(\rho_1)$;

and (iii) one of the following conditions holds:

- $(q_1, t, l, t', j, q_2) \in \delta_C$ and $s_2 = (t', \rho_2(j))s_1$,
- $(q_1, t, l, q_2) \in \delta_N$ and $s_2 = s_1$,
- $(q_1, t, l, t', j, q_2) \in \delta_R$, $s_1 = (t', d')s_2$,

where if $j \in [1, r]$ then $d' = \rho_2(j)$, otherwise $d' \notin \mathsf{rng}(\rho_2)$.

We show that even the visibly pushdown with SF register discipline is undecidable. To do so, we reduce from the undecidable emptiness problem for (one-way) *universal* register automata with two registers (URA_2) [DL09].

Definition 8.2 [DL09]**.** A one-way universal n-register automaton (URA_n) is a tuple $\langle \Sigma, Q, q_I, n, \delta \rangle$ such that Σ is a finite alphabet, Q is a finite set of states, $q_I \in Q$ is the initial state and $\delta : Q \to \Delta(\Sigma, Q, n)$ is the transition function, where

$$\begin{array}{rcl} \Delta(\Sigma, Q, n) & = & \{\, \bot,\ \top,\ q \wedge q',\ q \triangleleft \beta \triangleright q',\ Xq,\ \overline{X}q,\ \downarrow_r q \\ & \mid & \quad q, q' \in Q,\ r \in \{1, \cdots, n\},\ \beta \in B(\Sigma, n)\ \} \\ B(\Sigma, n) & = & \{a, \mathsf{end}\} \cup \{\uparrow_r \mid r \in \{1, \cdots, n\}\} \end{array}$$

The emptiness problem for URA_2 is undecidable [DL09]. We shall reduce it to bisimilarity testing. We first sketch the argument and then later give all the details.

Given a URA_2 U, we shall devise a 2-VPDRA $\mathcal{A}_U$ with two configurations κ_1, κ_2 such that U accepts a word iff $\kappa_1 \not\sim \kappa_2$. $\mathcal{A}_U$ is constructed to induce a bisimulation game in which Attacker gets a chance to choose a word to be accepted by U and simulate an accepting run (if one exists). It consists of two nearly identical components, which are linked by the Defender Forcing circuit in places. Other differences between them stem from the need to arrange for non-bisimilarity, in cases when the bisimulation game reaches a stage indicating acceptance or Attacker tried to cheat while simulating a run. We sketch the design of the components.

Input stage. Initially, we want Attacker to start choosing input letters and pushing them on the stack. This is to continue until Attacker decides to finish the input phase. Defender will simply copy the moves in other component. Technically, both kinds of choices can be implemented by deterministic push transitions that cover the range of input in both components. Observe that, in order to win (uncover non-bisimilarity), Attacker will eventually need to abandon the input stage to avoid infinite copying.

Transitions. Once the input phase is over, the automaton enters the simulation stage. Recall that the input word chosen by Attacker will be available on the stack in both components. The top of the stack will play the role of the head of U and we can use the two registers of $\mathcal{A}_U$ to emulate the two registers of U. To make transitions, we need to be able to access the tag at the top of the stack as well as compare the corresponding data value with the content of registers. The only way of inspecting the top of the stack is by popping, but then we could lose the data value if it does not already occur in a register (the value might be needed later, e.g. the automaton might want to move it into a register). To avoid such a loss, we will let Attacker guess the outcome of the comparisons. However, Defender will be allowed to verify the correctness of such guesses (via Defender Forcing). During the verification the top of the stack will indeed be popped, but we shall be no longer concerned about losing it, because it will survive in a different branch of the game, which will carry on simulating the run. In order to implement the detection of incorrect guesses, we will need to break symmetry between the components and arrange for non-bisimilarity if Attacker's guess is correct.

Universal states. To simulate these, we can delegate the choice to Defender through Forcing. This will allow Defender to direct the game towards a failing branch, if one exists.

Head movements. To advance the tape, we simply use one of the pop-instructions.

Register reassignment. To move the currently scanned data value into a register, let us assume that the symbol is not in a register yet. Then we can refresh the content of the relevant register (to guess the data value at the top of the stack) and then perform a pop.

Note that a wrong guess by Attacker will lead to a deadlock (no ability to pop), which gives Attacker the necessary incentive to guess correctly.

Accepting/rejecting states. If the simulation reaches a rejecting state, we arrange for bisimilarity (to attract Defender there). In accepting states, we arrange non-bisimilarity.

Theorem 8.3. *VPDRA(SF) bisimiliarity is undecidable.*

Proof. Given a URA_2 $U = \langle \Sigma, Q, q_I, 2, \delta \rangle$, we shall construct a 2-VRPDA $\mathcal{A}_U$ such that $\kappa_1 \sim \kappa_2$ if and only if U does not accept any input, where $\kappa_j = (\mathit{init}^j, \tau_I, \epsilon)$ $(j = 1, 2)$ and $\mathit{init}^1, \mathit{init}^2$ are states. $\mathcal{A}_U$ will be constructed so as to induce a bisimulation game in which Attacker gets a chance to choose a word to be accepted and simulate an accepting run (if one exists). Without loss of generality, we shall assume injectivity of register assignments and that, whenever $\downarrow_r$ is used, the $\mathcal{D}$-value on the tape is not present in registers (these conditions can be enforced by modifying the transition function with the help of the finite control and appropriate book-keeping). Moreover, to avoid complications with borderline cases, we shall assume that U does not accept the empty word.

$\mathcal{A}_U$ will consist of two mostly identical components involving superscripted states from U as well as a number of auxiliary states implicit in the definitions below. The only connections between the two components will be due to the use of the Defender Forcing circuit. The only differences between the components will stem from the need to arrange for non-bisimilarity, in cases when the bisimulation game reaches a stage indicating acceptance or when Attacker makes a simulation mistake.

Below we explain the design of $\mathcal{A}_U$ at various stages of simulating U. We use arrows to define transitions according to the following conventions.

- $q_1 \xrightarrow{(t,l)/(t',j)} q_2$ stands for $(q_1, t, l, t', j, q_2) \in \delta_C$
- $q_1 \xrightarrow{(t,l)} q_2$ stands for $(q_1, t, l, q_2) \in \delta_N$
- $q_1 \xrightarrow{(t,l),(t',j)} q_2$ stands for $(q_1, t, l, t', j, q_2) \in \delta_R$

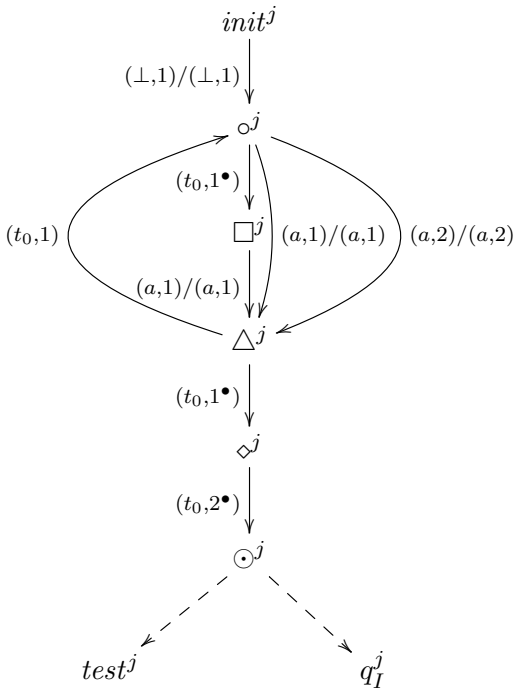
Given $q \in Q$, we write q^j $(j = 1, 2)$ for its superscripted variants to be included in $\mathcal{A}_U$. We shall rely on the following sets of tags.

$$\begin{array}{rcl} \Sigma_C & = & \{\bot\} + \Sigma \\ \Sigma_N & = & \{t_0, t_1, t_2\} \\ \Sigma_R & = & \{t_R\} \end{array}$$

For the stack alphabet, we shall have $\Gamma = \Sigma_C$.

We start off by introducing new states $\mathit{init}^1, \mathit{init}^2$ that will be used to start the initial phase in which Attacker can choose an input word and push it on the stack.

Input Phase. When drawing a diagram featuring states superscripted with j, we mean to say that *two* copies of the design should be included into $\mathcal{A}_\mathcal{U}$, one for $j = 1$ and another for $j = 2$. We use $\circ, \Box, \triangle, \diamond, \odot$ to indicate auxiliary states to be included in each component. We shall reuse them in different cases on the understanding that they refer to *different* states in each case.



a ranges over Σ above. Consequently, if the bisimulation game starts from (κ_1, κ_2) then the above design gives Attacker a chance to pick a data word and push it on the stack. The three outgoing transitions from state $\circ^j$ correspond to (from left to right) Attacker picking for the next data value: a fresh data value not currently in either register, the data value currently stored in register 1 or the data value currently stored in register 2. The stack content in both copies will be the same. Attacker also decides when to end the input selection phase and proceed to $(\diamond^1, \diamond^2)$. The transition sequence $(t_0, 1^\bullet)(t_0, 2^\bullet)$ is intended to give Attacker a chance to pick the right initial register assignment to support the simulation. For a match with URA, we need the initial values to be different from any data values present in the selected input word. Once Attacker generates the values and $(\odot^1, \odot^2)$ is reached, Defender will have an option to challenge the choice or to proceed with the simulation to (q_I^1, q_I^2). This will be achieved through Defender Forcing, represented by dashed lines. We shall return to the exact design of $test^j$, after we apply Defender Forcing in simpler cases.

The subsequent part of the construction corresponds to checking that the selected word is accepted (we want Attacker to win iff this is the case). We analyze each kind of transition in turn.

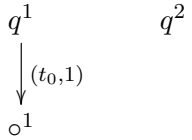
Transitions.

$\delta(q) = \bot$ *(rejection).*

$$q^j$$

We do not add any transitions from q^1 or q^2. This ensures bisimilarity, should the game enter configurations with states q^1, q^2 respectively.

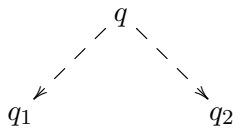
$\delta(q) = \top$ *(acceptance).*



Note that we do not add any transitions from q^2 in order to generate non-bisimilar configurations.

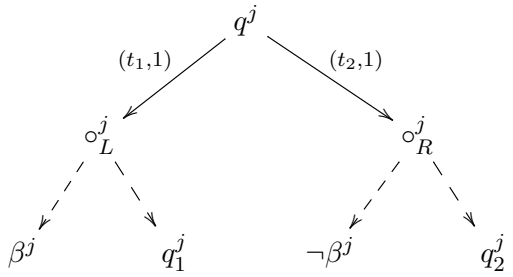
$\delta(q) = q_1 \wedge q_2$ *(universal choice).* We will let Defender choose the state (q_1 or q_2) that should be pursued. Note that this is consistent with the goal of relating emptiness with bisimilarity. To that end, we use the Defender Forcing circuit from Section 2.1 (Figure 4). Recall that in order for the technique to work with VPDRA, we need to be sure that the stacks and registers are used in the same way by each of the components. This is an easily verifiable property of our constructions. In order to implement DF we need two different labels, e.g. $(t_1, 1)$ and $(t_2, 1)$.

For brevity, in what follows, we shall write



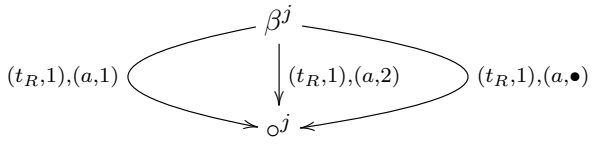
to refer to the use of $DF(q^1, q^2, (t_1, 1), (t_1, 1), (t_2, 1), q_1^1, q_1^2, q_2^1, q_2^2)$.

$\delta(q) = q_1 \triangleleft \beta \triangleright q_2$. Here we shall let Attacker choose between q_1 and q_2 but the Defender will later be able to challenge the decision (and check whether it is consistent with β). For this purpose we use

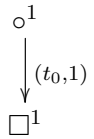


where $\beta^1, \beta^2, \neg\beta^1, \neg\beta^2$ will be constructed so that the first two induce bisimilarity iff β fails and the last two induce bisimilarity iff β holds. We do case analysis on β.

$\beta = a$ *(stack tag comparison).* To handle β^j, we introduce

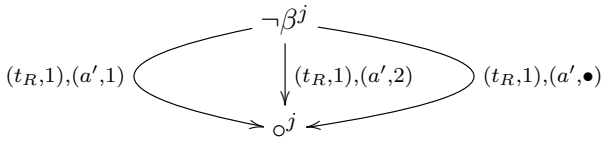


and

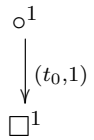


We explain the idea behind this first gadget, the rest are similar. If Defender was correct to challenge Attacker because Attacker cheated, i.e. the letter under the head (top of stack) is not tagged by a (despite Attacker's claim), then Attacker will not be able to play any transition from β_j and hence Defender will win. If Defender challenged Attacker incorrectly, then Attacker will be able to play exactly one of the transitions, according to the current register assignment, and Defender will copy the move. However, in the following move Attacker will win, since Attacker will play the only transition out of $\circ^1$ and Defender cannot match this in $\circ^2$, since it has no available transitions.

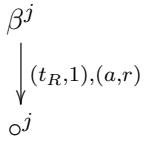
For $\neg\beta^j$ we can take



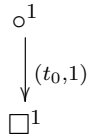
where a' ranges over $\Sigma \setminus \{a\}$, and:



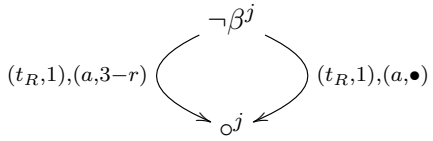
$\beta =\uparrow_r$ *(stack $\mathcal{D}$-value comparison).* To handle β^j, we introduce



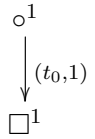
with a ranging over Σ, and:



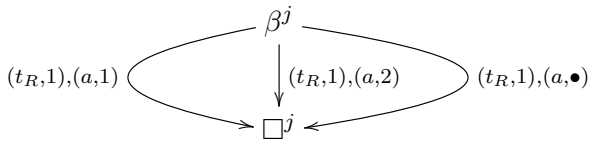
For $\neg\beta^j$ we can take



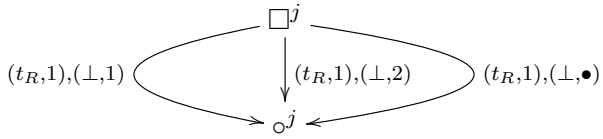
with a ranging over Σ, and



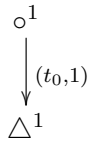
$\beta = \mathsf{end}$ *(last tape-symbol).* To handle β^j, we introduce



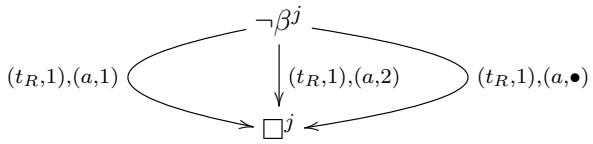
with a ranging over Σ,



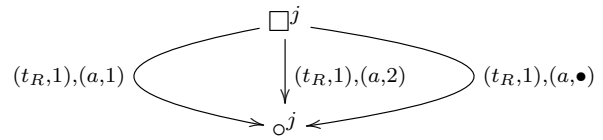
and:



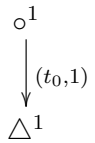
For $\neg\beta^j$ we can take



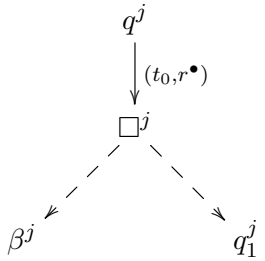
with a ranging over Σ,



with a ranging over Σ again, and:

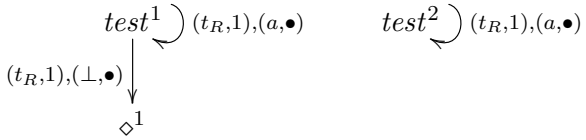


$\delta(q) = \downarrow_r q_1$. We add

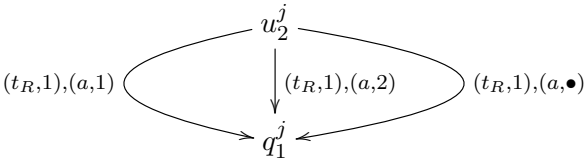


and also add outgoing transitions for β^j, as for the case $\beta = \uparrow_r$. Note that the arrangement forces Attacker to guess the $\mathcal{D}$-value stored on top of the stack (and place it in register r).

Freshness testing ($test^j$). We design $test^1$ and $test^2$ in such a way that they will lead to non-bisimilarity iff Attacker guessed an initial register assignment that does not contain any data values encountered during the input phase. a ranges over Σ.



$\delta(q) = Xq_1$ *(move head right/reject).* To take advantage of previous cases, we represent the transition as $u_1 \triangleleft \mathsf{end} \triangleright u_2$ with $\delta(u_1) = \bot$ and $\delta(u_2) = Xq_1$. This makes sure that X is only invoked when we are not at the end of the word. Consequently, we can reuse the previous constructions for $u_1 \triangleleft \beta \triangleright u_2$ and $\bot$ cases. To handle u_2, we can now add



with a ranging over Σ.

$\delta(q) = \overline{X}q_1$ *(move head right/accept).* This is nearly the same as the previous case: now we decompose the transition into $u_1 \triangleleft \mathsf{end} \triangleright u_2$ with $\delta(u_1) = \top$ and $\delta(u_2) = Xq_1$.

Altogether, we obtain $\kappa_1 \sim \kappa_2$ if and only if U does not accept any words. This implies Theorem 8.3. □

The argument above also reduces URA_1 emptiness to 1-VPDRA, which implies a non-primitive-recursive lower bound for 1-VPDRA.

9. Conclusion

We have demonstrated bounds on the bisimilarity problem for broad classes of (fresh-)register automata, which include those studied in the literature. The ability to start with empty registers, erase their contents (or equivalently, store duplicate values) and use of a stack all affect the inherent problem complexity. Global freshness, however, does not seem to affect complexity. Except for the SF discipline, all bounds are tight.

Although our problem formulation is with respect to two configurations of a single automaton, extending our results to problems concerning two automata is unproblematic. If the automata have different numbers of registers, the game can be played on an automaton with a number equal to the larger of the two, with additional registers initialised (and left) empty. Even in F register disciplines our arguments show that, since these extra registers are never assigned to, the system can be treated as a $\#_0$ system without change in complexity.

Acknowledgments

We would like to thank M. Jerrum, R. Gray, J. Mitchell and M. Beaudry for useful discussions regarding computational group theory. We are also grateful to the anonymous referees for many helpful suggestions. The research was supported by the Engineering and Physical Sciences Research Council (EP/J019577/1, EP/L022478/1) and the Royal Academy of Engineering (RF 10216/111).

References

[AM04] R. Alur and P. Madhusudan. Visibly pushdown languages. In *Proceedings of STOC'04*, pages 202–211. ACM, 2004.

[Bab86] L. Babai. On the length of subgroup chains in the symmetric group. *Commun. Algebra*, 14(9):1729–1736, 1986.

[BGKM13] M. Benedikt, S. Göller, S. Kiefer, and A. S. Murawski. Bisimilarity of pushdown automata is nonelementary. In *Proceedings of LICS*, pages 488–498. IEEE Computer Society, 2013.

[BKL14] M. Bojańczyk, B. Klin, and S. Lasota. Automata theory in nominal sets. *LMCS*, 10(3), 2014.

[BT00] M. Boreale and L. Trevisan. A complexity analysis of bisimilarity for value-passing processes. *Theor. Comput. Sci.*, 238(1-2):313–345, 2000.

[CKS81] A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981.

[CM10] V. Ciancia and U. Montanari. Symmetries, local names and dynamic (de)-allocation of names. *Inf. Comput.*, 208(12):1349 – 1367, 2010.

[DL09] S. Demri and R. Lazić. LTL with the freeze quantifier and register automata. *ACM Trans. Comput. Log.*, 10(3), 2009.

[FHL80] M. L. Furst, J. E. Hopcroft, and E. M. Luks. Polynomial-time algorithms for permutation groups. In *Proceedings of FOCS*, pages 36–41. IEEE Computer Society, 1980.

[GDPT13] R. Grigore, D. Distefano, R. L. Petersen, and N. Tzevelekos. Runtime verification based on register automata. In *Proceedings of TACAS*, volume 7795 of *LNCS*, pages 260–276. Springer, 2013.

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

[JS08] P. Jančar and J. Srba. Undecidability of bisimilarity by defender's forcing. *J. ACM*, 55(1), 2008.

[JS19] P. Jančar and S. Schmitz. Bisimulation equivalence of first-order grammars is ACKERMANN-complete. In *Proceedings of LICS*, pages 1–12. IEEE, 2019.

[KF94] M. Kaminski and N. Francez. Finite-memory automata. *Theor. Comput. Sci.*, 134(2):329–363, 1994.

[Koz77] D. Kozen. Lower bounds for natural proof systems. In *Proceedings of FOCS*, pages 254–266. IEEE Computer Society, 1977.

[LP82] H. R. Lewis and C. H. Papadimitriou. Symmetric space-bounded computation. *Theor. Comput. Sci.*, 19:161–187, 1982.
[MN87] A. McIver and P. M. Neumann. Enumerating finite groups. *Quart. J. Math. Oxford Ser.*, 38(4):473–488, 1987.
[MP97] U. Montanari and M. Pistore. An introduction to history dependent automata. *Electr. Notes Theor. Comput. Sci.*, 10, 1997.
[MRT14] A. S. Murawski, S. J. Ramsay, and N. Tzevelekos. Reachability in pushdown register automata. In *Proceedings of MFCS*, volume 8634 of *LNCS*, pages 464–473. Springer, 2014.
[MRT15] A. S. Murawski, S. J. Ramsay, and N. Tzevelekos. Bisimilarity in fresh-register automata. In *Proceedings of LICS*, pages 156–167. IEEE Computer Society, 2015.
[MRT18] A. S. Murawski, S. J. Ramsay, and N. Tzevelekos. Polynomial-time equivalence testing for deterministic fresh-register automata. In *Proceedings of MFCS*, volume 117 of *LIPIcs*, pages 72:1–72:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018.
[NSV04] F. Neven, T. Schwentick, and V. Vianu. Finite state machines for strings over infinite alphabets. *ACM Trans. Comput. Log.*, 5(3):403–435, 2004.
[Pis99] M. Pistore. *History Dependent Automata*. PhD thesis, University of Pisa, 1999.
[Pit13] A. M. Pitts. *Nominal Sets*. CUP, 2013.
[Sak98] H. Sakamoto. *Studies on the Learnability of Formal Languages via Queries*. PhD thesis, Kyushu University, 1998.
[Seg06] L. Segoufin. Automata and logics for words and trees over an infinite alphabet. In *Proceedings of CSL*, volume 4207 of *Lecture Notes in Computer Science*. Springer, 2006.
[Sén05] G. Sénizergues. The bisimulation problem for equational graphs of finite out-degree. *SIAM J. Comput.*, 34(5):1025–1106, 2005.
[SI00] H. Sakamoto and D. Ikeda. Intractability of decision problems for finite-memory automata. *Theor. Comput. Sci.*, 231(2):297–308, 2000.
[Srb06] J. Srba. Visibly pushdown automata: From language equivalence to simulation and bisimulation. In *Proceedings of CSL*, volume 4207 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 2006.
[Srb08] J. Srba. Roadmap of infinite results. `http://www.brics.dk/~srba/roadmap/`, 2008.
[Tze11] N. Tzevelekos. Fresh-register automata. In *Proceedings of POPL*, pages 295–306. ACM Press, 2011.

Appendix A. Proofs from Section 3

Given $d, d' \in \mathcal{D}$, let us write $(d\ d')$ for the bijection on $\mathcal{D}$ defined as:

$$(d\ d')(x) = \begin{cases} d' & \text{if } x = d \\ d & \text{if } x = d' \\ x & \text{otherwise} \end{cases}$$

In what follows, we will consider various finite sets that involve elements of $\mathcal{D}$, e.g. finite subsets of $\mathcal{D}$, register assignments and tuples thereof. Given such a finite set X, we write $(d\ d') \cdot X$ for the result of applying $(d\ d')$ recursively to the elements of X. Put otherwise, $(d\ d') \cdot X$ will be X, where d and d' have been swapped.[11] In particular,

- if X does not involve names, then $(d\ d') \cdot X = X$;
- if $X \subseteq \mathcal{D}$, then $(d\ d') \cdot X = \{(d\ d')(x) \mid x \in X\}$;
- if X is some register assignment, then $(d\ d') \cdot X = \{(i, (d\ d')(X(i))) \mid X(i) \in \mathcal{D}\} \cup \{(i, \#) \mid X(i) = \#\}$;
- if X is some tuple $(X_1, \ldots, X_n)$ then $(d\ d') \cdot X = ((d\ d') \cdot X_1, \ldots, (d\ d') \cdot X_n)$.

[11] Formally, this can be defined as an action of the group of permutations on a nominal set; see [Pit13] for a detailed exposition.

Moreover, we shall consider finite name-permutations, i.e. ones taken from the set:

$$\mathsf{Perm}_{\mathcal{D}} = \{\, \pi : \mathcal{D} \xrightarrow{\cong} \mathcal{D} \mid \exists X \subseteq \mathcal{D}.\ X \text{ finite} \wedge \forall d \in \mathcal{D} \setminus X.\, \pi(d) = d \,\}$$

and use π to range over them. Each $\pi \in \mathsf{Perm}_{\mathcal{D}}$ can be decomposed as $\pi = (d_1\ d_1') \circ \cdots \circ (d_n\ d_n')$, for some n and $d_1, d_1', \ldots, d_n, d_n' \in \mathcal{D}$. We then define $\pi \cdot X = (d_1\ d_1') \cdot \ldots \cdot (d_n\ d_n') \cdot X$.

Finally, given ρ_1, ρ_2, H with $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq H \subseteq N$ or $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq N \subseteq H$, we extend the trim operation for H as:

$$\lceil H \rceil^N_{\rho_1,\rho_2} = \begin{cases} H & \text{if } H \subsetneq N \\ N \setminus \{\min(N \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))\} & \text{if } N \subseteq H \end{cases}$$

In either case, $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq \lceil H \rceil^N_{\rho_1,\rho_2} \subseteq N \cap H$ and $\lceil H \rceil^N_{\rho_1,\rho_2} \subsetneq N$. Moreover, given $\rho_1, \rho_2, H, \hat{H}$, we say that H *can restrict to* $(\rho_1, \rho_2, \hat{H})$, written $H \rhd_N (\rho_1, \rho_2, \hat{H})$, if:

- $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq H \subsetneq N$ and $\hat{H} = H$, or
- $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq N \subseteq H$ and $\hat{H} = N \setminus \{d\}$ for some $d \in N \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$.

Note that, in either case, $\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \subseteq \hat{H} \subseteq N \cap H$ and $\hat{H} \subsetneq N$. In particular, when $\lceil H \rceil^N_{\rho_1,\rho_2}$ is well defined, we have $H \rhd_N (\rho_1, \rho_2, \lceil H \rceil^N_{\rho_1,\rho_2})$.

Proof of Lemma 3.7. We show a correspondence between bisimulations and N-bisimulations from which the result follows.

bisim $\to$ N-bisim. Let R be a bisimulation on $\mathcal{A}$ that is closed in the following manner: for all permutations π, if $(q_1, \rho_1, H)\, R\, (q_2, \rho_2, H)$ then $(\pi \cdot (q_1, \rho_1, H))\, R\, (\pi \cdot (q_2, \rho_2, H))$. We claim that the relation $\hat{R} \subseteq \mathbb{C}_{\mathcal{A},N} \times \mathbb{C}_{\mathcal{A},N}$, defined by

$$\hat{R} = \{\, ((q_1, \rho_1, \hat{H}), (q_2, \rho_2, \hat{H})) \mid \exists H.\ (q_1, \rho_1, H)\, R\, (q_2, \rho_2, H) \wedge H \rhd_N (\rho_1, \rho_2, \hat{H}) \,\},$$

is an N-bisimulation.

Let $(q_1, \rho_1, \hat{H})\, \hat{R}\, (q_2, \rho_2, \hat{H})$, due to some $(q_1, \rho_1, H)\, R\, (q_2, \rho_2, H)$, and suppose $(q_1, \rho_1, \hat{H}) \xrightarrow{(t,d)} (q_1', \rho_1', \hat{H}')$ for some $t, d, q_1', \rho_1', \hat{H}'$. Next we reason by case analysis.

(a) Suppose $d \in \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$. Then, $\hat{H}' = \hat{H}$ and, since R is a bisimulation, we have $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', \rho_2', H)$ for some q_2', ρ_2' such that $(q_1', \rho_1', H)\, R\, (q_2', \rho_2', H)$. Consequently, $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d)} (q_2', \rho_2', \hat{H})$. It suffices to show that $(q_1', \rho_1', \hat{H})\, \hat{R}\, (q_2', \rho_2', \hat{H})$, i.e. $H \rhd_N (\rho_1', \rho_2', \hat{H})$. But this follows from $H \rhd_N (\rho_1, \rho_2, \hat{H})$ and $\mathsf{rng}(\rho_1') \cup \mathsf{rng}(\rho_2') \subseteq \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$.

(b) Suppose $d = \min(\hat{H} \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)))$. Then, again $\hat{H}' = \hat{H}$ and, reasoning as in the previous case, $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d)} (q_2', \rho_2', \hat{H})$ for some q_2', ρ_2' such that $(q_1', \rho_1', H)\, R\, (q_2', \rho_2', H)$. Since $\mathsf{rng}(\rho_1') \cup \mathsf{rng}(\rho_2') \subseteq \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2) \cup \{d\}$, it follows that $H \rhd_N (\rho_1', \rho_2', \hat{H})$ and, thus, $(q_1', \rho_1', \hat{H})\, \hat{R}\, (q_2', \rho_2', \hat{H})$, as required.

(c1) Suppose $d = \min(N \setminus \hat{H})$ and $\hat{H} = H \subsetneq N$. Then, $\hat{H}' = \hat{H} \uplus \{d\}$ and, since R is a bisimulation, we have $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d)} (q_2', \rho_2', \hat{H}')$ for some q_2', ρ_2' with $(q_1', \rho_1', \hat{H}')\, R\, (q_2', \rho_2', \hat{H}')$. Moreover, $\hat{H}' \rhd_N (\rho_1', \rho_2', \lceil \hat{H}' \rceil^N_{\rho_1',\rho_2'})$ and, thus, $(q_1', \rho_1', \lceil \hat{H}' \rceil^N_{\rho_1',\rho_2'})\, \hat{R}\, (q_2', \rho_2', \lceil \hat{H}' \rceil^N_{\rho_1',\rho_2'})$.

(c2) Suppose $d = \min(N \setminus \hat{H})$ and $\hat{H} = N \setminus \{\hat{d}\}$ for some $\hat{d} \in (N \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$, and $N \subseteq H$. Clearly, $d = \hat{d}$ and $\hat{H}' = N$. Since d is a fresh name for $\hat{H}$, the transition from $(q_1, \rho_1, \hat{H})$ must be a globally fresh one, i.e. $\rho_1' = \rho_1[i \mapsto d]$. This

implies that $(q_1, \rho_1, H) \xrightarrow{(t,d')} (q_1', \rho_1[i \mapsto d'], H \uplus \{d'\})$ for some fresh d' and, therefore, $(q_2, \rho_2, H) \xrightarrow{(t,d')} (q_2', \rho_2[j \mapsto d'], H \uplus \{d'\})$ with $(q_1', \rho_1[i \mapsto d'], H \uplus \{d'\})\, R\, (q_2', \rho_2[j \mapsto d'], H \uplus \{d'\})$, for some q_2', j. Moreover, $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d)} (q_2', \rho_2[j \mapsto d], N)$. Let us set $\rho_2' = \rho_2[j \mapsto d]$. By closure of R under permutations of $\mathcal{D}$, we also have that $(q_1', \rho_1', H \uplus \{d'\})\, R\, (q_2', \rho_2', H \uplus \{d'\})$, so it suffices to show that $(H \uplus \{d'\}) \triangleright_N (\rho_1', \rho_2', \lceil N \rceil^N_{\rho_1', \rho_2'})$, which holds by definition.

N-bisim $\to$ bisim. Let R be an N-bisimulation on $\mathcal{A}$. We claim that the relation $\hat{R} \subseteq \mathbb{C}_{\mathcal{A}} \times \mathbb{C}_{\mathcal{A}}$, defined by

$$\hat{R} = \{\, \pi{\cdot}((q_1, \rho_1, H), (q_2, \rho_2, H)) \mid \pi \in \mathsf{Perm}_{\mathcal{D}} \wedge \exists \hat{H}. (q_1, \rho_1, \hat{H})\, R\, (q_2, \rho_2, \hat{H}) \wedge H \triangleright_N (\rho_1, \rho_2, \hat{H}) \,\}$$

is a bisimulation.
Let $(q_1, \rho_1, H)\, \hat{R}\, (q_2, \rho_2, \hat{H})$, due to some $(q_1, \rho_1, \hat{H})\, R\, (q_2, \rho_2, \hat{H})$, so WLOG assume that π is the identity, and suppose $(q_1, \rho_1, H) \xrightarrow{(t,d)} (q_1', \rho_1', H')$ for some t, d, q_1', ρ_1', H'. Next we reason by case analysis.

(a) Suppose $d \in \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$. Then, $H' = H$ and, since R is an N-bisimulation, we have $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d)} (q_2', \rho_2', \hat{H})$ for some q_2', ρ_2' such that $(q_1', \rho_1', \hat{H})\, R\, (q_2', \rho_2', \hat{H})$. Hence, $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', \rho_2', H)$. We need to show that $(q_1', \rho_1', H)\, \hat{R}\, (q_2', \rho_2', H)$. For this, it suffices that $H \triangleright_N (\rho_1', \rho_2', \hat{H})$, which follows from $H \triangleright_N (\rho_1, \rho_2, \hat{H})$ and $\mathsf{rng}(\rho_1') \cup \mathsf{rng}(\rho_2') \subseteq \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$.

(b) Suppose $d \in H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$ and let $d' = \min(\hat{H} \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)))$. Note that $d' \in H$. Then, we also have $(q_1, \rho_1, H) \xrightarrow{(t,d')} (q_1', (d\ d') \cdot \rho_1', H)$ and, hence, $(q_1, \rho_1, \hat{H}) \xrightarrow{(t,d')} (q_1', (d\ d') \cdot \rho_1', \hat{H})$. By N-bisimulation, we get $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d')} (q_2', \rho_2', \hat{H})$ and $(q_1, (d\ d') \cdot \rho_1', \hat{H})\, R\, (q_2, \rho_2', \hat{H})$. But then $(q_2, \rho_2, H) \xrightarrow{(t,d')} (q_2', \rho_2', H)$ and therefore $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', (d\ d') \cdot \rho_2', H)$, so it suffices to show that $(q_1', \rho_1', H)\, \hat{R}\, (q_2', (d\ d') \cdot \rho_2', H)$. Note that $H \triangleright_N (\rho_1, \rho_2, \hat{H})$ implies that $H \triangleright_N ((d\ d') \cdot \rho_1', \rho_2', \hat{H})$, thus $(q_1', (d\ d') \cdot \rho_1', H)\, \hat{R}\, (q_2', \rho_2', H)$, and hence $(q_1', \rho_1', H) = ((d\ d') \cdot (q_1', (d\ d') \cdot \rho_1', H))\ \hat{R}\ ((d\ d') \cdot (q_2', \rho_2', H)) = (q_2', (d\ d') \cdot \rho_2', H)$.

(c1) Suppose $d \notin H$, $\hat{H} = H \subsetneq N$, and pick $d' = \min(N \setminus \hat{H})$. Then, $H' = H \uplus \{d\}$ and we also have $(q_1, \rho_1, H) \xrightarrow{(t,d')} (q_1', (d\ d') \cdot \rho_1', (d\ d') \cdot H')$ and hence, since R is an N-bisimulation, we get $(q_2, \rho_2, H) \xrightarrow{(t,d')} (q_2', \rho_2', (d\ d') \cdot H')$ for some q_2', ρ_2' with $(q_1', (d\ d') \cdot \rho_1', \hat{H'})\, R\, (q_2', \rho_2', \hat{H'})$ and $\hat{H'} = \lceil (d\ d') \cdot H' \rceil^N_{(d\ d') \cdot \rho_1', \rho_2'}$. The latter implies that $(q_1', (d\ d') \cdot \rho_1', (d\ d') \cdot H')\, \hat{R}\, (q_2', \rho_2', (d\ d') \cdot H')$ and, by closure of $\hat{R}$, $(q_1', \rho_1', H')\, \hat{R}\, (q_2', (d\ d') \cdot \rho_2', H')$. We conclude by noting that also $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', (d\ d') \cdot \rho_2', H')$.

(c2) Suppose $d \notin H$, $N \subseteq H$ and $\hat{H} = N \setminus \{d'\}$ for some $d' \in N \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$, so $d' \in H$. Then, $H' = H \uplus \{d\}$ and we also have $(q_1, \rho_1, \hat{H}) \xrightarrow{(t,d')} (q_1', (d\ d') \cdot \rho_1', N)$ and hence, since R is an N-bisimulation, $(q_2, \rho_2, \hat{H}) \xrightarrow{(t,d')} (q_2', \rho_2', N)$ for some q_2', ρ_2' with $(q_1', (d\ d') \cdot \rho_1', \hat{H'})\, R\, (q_2', \rho_2', \hat{H'})$ and $\hat{H'} = \lceil N \rceil^N_{(d\ d') \cdot \rho_1', \rho_2'}$. But then $(q_2, \rho_2, H) \xrightarrow{(t,d)} (q_2', (d\ d') \cdot \rho_2', H')$, so it suffices to show that $(q_1', \rho_1', H')\, \hat{R}\, (q_2', (d\ d') \cdot \rho_2', H')$. Noting

that $H' \triangleright_N ((d\ d') \cdot \rho_1', \rho_2', \hat{H}')$, we get $(q_1', (d\ d') \cdot \rho_1', H')\ \hat{R}\ (q_2', \rho_2', H')$, from which the claim follows by closure of $\hat{R}$.

The lemma follows from the two reductions above, using the fact that bisimilarity satisfies the permutation-closure assumption used in the first reduction. □

Appendix B. Proofs from Section 4

Proof of Lemma 4.3. We do a case analysis on i, j being 0 or not. Assume first that $i, j \neq 0$. Then:

$$\begin{aligned}(\rho_1; \rho_2^{-1})[i \mapsto j] &= \{(i,j)\} \cup \{(i',j') \in [1,r]^2 \mid i' \neq i \wedge j' \neq j \wedge \exists a'.\, \rho_1(i') = \rho_2(j') = a'\} \\ &= \{(i,j)\} \cup \{(i',j') \in \rho_1[i \mapsto a]; \rho_2[j \mapsto a]^{-1} \mid i' \neq i \wedge j' \neq j\} \\ &= \rho_1[i \mapsto a]; \rho_2[j \mapsto a]^{-1}\end{aligned}$$

On the other hand, if $i = j = 0$ then the claim is trivial. Suppose now $i = 0, j \neq 0$. Then:

$$\begin{aligned}(\rho_1; \rho_2^{-1})[i \mapsto j] &= \{(i',j') \in [1,r]^2 \mid j' \neq j \wedge \exists a'.\, \rho_1(i') = \rho_2(j') = a'\} \\ &= \{(i',j') \in [1,r]^2 \mid j' \neq j \wedge \exists a' \neq a.\, \rho_1(i') = \rho_2(j') = a'\} \quad (\text{as } a \notin \mathsf{rng}(\rho_1)) \\ &= \rho_1; \rho_2[j \mapsto a]^{-1} = \rho_1[i \mapsto a]; \rho_2[j \mapsto a]^{-1}\end{aligned}$$

Finally, if $i \neq 0, j = 0$ then we can show that $(\rho_1; \rho_2^{-1})[i \mapsto j]^{-1} = (\rho_1[i \mapsto a]; \rho_2[j \mapsto a]^{-1})^{-1}$ using the previous case above. □

Proof of Lemma 4.5. We show a correspondence between bisimulations and symbolic bisimulations from which the result follows.

bisim → s-bisim. Let R be a bisimulation on $\mathcal{A}$. We claim that the relation $R' \subseteq \mathcal{U}$,

$$R' = \{\, (q_1, S_1, \sigma, q_2, S_2) \mid \exists \rho_1, \rho_2.\ (q_1, \rho_1) R (q_2, \rho_2) \wedge \sigma = \rho_1; \rho_2^{-1} \wedge \mathsf{dom}(\rho_i) = S_i \,\}$$

is a symbolic bisimulation. For the latter (by symmetry in the definition) it suffices to show that R' is a symbolic simulation. So suppose that $(q_1, S_1, \sigma, q_2, S_2) \in R'$ due to some $(q_1, \rho_1) R (q_2, \rho_2)$. Let $q_1 \xrightarrow{t,i} q_1'$ for some $i \in S_1$. Then, $(q_1, \rho_1) \xrightarrow{t,a} (q_1', \rho_1)$ with $a = \rho_1(i)$ and, hence, $(q_2, \rho_2) \xrightarrow{t,a} (q_2', \rho_2')$ with $(q_1', \rho_1) R (q_2', \rho_2')$.

- If $i \in \mathsf{dom}(\sigma)$ then $a = \rho_2(\sigma(i))$ and therefore the above transition is due to some $q_2 \xrightarrow{t,\sigma(i)} q_2'$, and $\rho_2' = \rho_2$. Hence, $(q_1', S_1) R'_\sigma (q_2', S_2)$.
- If $i \notin \mathsf{dom}(\sigma)$ then the transition is due to some $q_2 \xrightarrow{t,j^\bullet} q_2'$, and $\rho_2' = \rho_2[j \mapsto a]$. Hence, since $\sigma[i \mapsto j] = \rho_1; (\rho_2[j \mapsto a])^{-1}$ and $\mathsf{dom}(\rho_2') = S_2[j]$, we have $(q_1', S_1) R'_{\sigma[i \mapsto j]} (q_2', S_2[j])$.

Now let $q_1 \xrightarrow{t,i^\bullet} q_1'$. For each $a \notin \mathsf{rng}(\rho_1)$, $(q_1, \rho_1) \xrightarrow{t,a} (q_1', \rho_1')$ with $\rho_1' = \rho_1[i \mapsto a]$ and, hence, there is some $(q_2, \rho_2) \xrightarrow{t,a} (q_2', \rho_2')$ with $(q_1', \rho_1') R (q_2', \rho_2')$.

- Select some $a \notin \mathsf{rng}(\rho_2)$. Then, the transition above is due to some $q_2 \xrightarrow{t,j^\bullet} q_2'$, and $\rho_2' = \rho_2[j \mapsto a]$. Moreover, since $\sigma[i \mapsto j] = \rho_1[i \mapsto a]; (\rho_2[j \mapsto a])^{-1}$, $\mathsf{dom}(\rho_1') = S_1[i]$ and $\mathsf{dom}(\rho_2') = S_2[j]$, we have $(q_1', S_1[i]) R'_{\sigma[i \mapsto j]} (q_2', S_2[j])$.
- Let $j \in S_2 \setminus \mathsf{rng}(\sigma)$. Then, we can take a to be $\rho_2(j)$, so the transition is due to some $q_2 \xrightarrow{t,j} q_2'$, and $\rho_2' = \rho_2$. We moreover have $(q_1', S_1[i]) R'_{\sigma[i \mapsto j]} (q_2', S_2)$.

s-bisim $\to$ bisim. Let R be a symbolic bisimulation on $\mathcal{A}$. We claim that the relation

$$R' = \{\, ((q_1,\rho_1),(q_2,\rho_2)) \mid (q_1,S_1)R_\sigma(q_2,S_2) \\ \wedge\, \sigma = \rho_1;\rho_2^{-1} \wedge S_i = \mathsf{dom}(\rho_i) \,\}$$

is a bisimulation, for which it suffices to show that R' is a simulation. So suppose that

$$((q_1,\rho_1),(q_2,\rho_2)) \in R'$$

due to some $(q_1,S_1)R_\sigma(q_2,S_2)$, and let $(q_1,\rho_1) \xrightarrow{t,a} (q_1',\rho_1')$ for some $(t,a) \in \Sigma \times \mathcal{D}$. If $a \in \mathsf{rng}(\rho_1)$, say $a = \rho_1(i)$, then $q_1 \xrightarrow{t,i} q_1'$ and $\rho_1' = \rho_1$. We distinguish two cases:

- If $a \in \mathsf{rng}(\rho_2)$ then $i \in \mathsf{dom}(\sigma)$, so $q_2 \xrightarrow{t,\sigma(i)} q_2'$ and $(q_1',S_1)R_\sigma(q_2',S_2)$. Hence, $(q_2,\rho_2) \xrightarrow{t,a} (q_2',\rho_2)$ and $(q_1',\rho_1)R'(q_2',\rho_2)$.
- If $a \notin \mathsf{rng}(\rho_2)$ then $i \in S_1 \setminus \mathsf{dom}(\sigma)$, so $q_2 \xrightarrow{t,j^\bullet} q_2'$ and $(q_1',S_1)R_{\sigma[i\mapsto j]}(q_2',S_2[j])$. Hence, $(q_2,\rho_2) \xrightarrow{t,a} (q_2',\rho_2[j\mapsto a])$ and $(q_1',\rho_1)R'(q_2',\rho_2[j\mapsto a])$.

If $a \notin \mathsf{rng}(\rho_1)$ then there is $q_1 \xrightarrow{t,i^\bullet} q_1'$ such that $\rho_1' = \rho_1[i\mapsto a]$.

- If $a \notin \mathsf{rng}(\rho_2)$ then, since $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1',S_1[i])R_{\sigma[i\mapsto j]}(q_2',S_2[j])$, we obtain $(q_2,\rho_2) \xrightarrow{t,a} (q_2',\rho_2[j\mapsto a])$ and $(q_1',\rho_1')R'(q_2',\rho_2[j\mapsto a])$.
- If $a \in \mathsf{rng}(\rho_2)$, say $a = \rho_2(j)$, then $j \in S_2 \setminus \mathsf{rng}(\sigma)$. Hence, $q_2 \xrightarrow{t,j} q_2'$ with
$$(q_1',S_1[i])R_{\sigma[i\mapsto j]}(q_2',S_2),$$
from which we get $(q_2,\rho_2) \xrightarrow{t,a} (q_2',\rho_2)$ and $(q_1',\rho_1')R'(q_2',\rho_2)$. □

Proof of Lemma 4.6. By induction on i we prove that, for all $i \in \omega$, $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$. When $i = 0$, the result is trivial as $\overset{i}{\sim}$ is the universe. Let us assume $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$ (IH) and $(q_1,S_1) \overset{i+2}{\sim}_\tau (q_2,S_2)$. It follows by definition that (q_1,S_1,τ,q_2,S_2) and $(q_2,S_2,\tau^{-1},q_1,S_1)$ satisfy the (SyS) conditions in $\overset{i+1}{\sim}$. Because $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$, the tuples also satisfy the (SyS) conditions in $\overset{i}{\sim}$, whence $(q_1,S_1) \overset{i+1}{\sim}_\tau (q_2,S_2)$, as needed.

We next show that $\bigcap_{i\in\omega} \overset{i}{\sim} = \overset{\mathsf{s}}{\sim}$. We start with the $\supseteq$ direction and argue that, for all $i \in \omega$, $\overset{i}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$. The proof is by induction on i. When $i = 0$ the result is trivial. Let us assume $\overset{i}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$ (IH) and $(q_1,S_1) \overset{\mathsf{s}}{\sim}_\tau (q_2,S_2)$. We wish to show that (q_1,S_1,τ,q_2,S_2) and its inverse satisfy the (SyS) conditions in $\overset{i}{\sim}$. By definition, they satisfy the (SyS) conditions in $\overset{\mathsf{s}}{\sim}$. Because $\overset{i}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$, the tuples satisfy the (SyS) conditions in $\overset{i}{\sim}$. Hence, $\overset{i+1}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$.

For the $\subseteq$ direction, we argue that the left-hand side is a symbolic bisimulation. To see this, assume $(q_1,S_1,\tau,q_2,S_2) \in \bigcap_{i\in\omega} \overset{i}{\sim}$ so that (q_1,S_1,τ,q_2,S_2) and its inverse satisfy the (SyS) conditions in $\overset{i}{\sim}$, for all $i \in \omega$. The satisfaction of the (SyS) conditions in $\overset{i}{\sim}$ by (q_1,S_1,τ,q_2,S_2) (and, analogously, by its inverse) is witnessed by a subset $C_i \subseteq \overset{i}{\sim} \subseteq \mathcal{U}$ for each i. Because $\mathcal{U}$ is finite, there exists C such that $C = C_i$ for infinitely many i. Consequently, in view of $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$, C witnesses satisfaction of the (SyS) conditions in $(\bigcap_{i\in\omega} \overset{i}{\sim})$. □

Proof of Lemma 4.8. We first observe that $Cl(R) = Cl^-(R \cup R^{-1})$ where, for any relation X, we let $Cl^-(X)$ be the smallest relation that contains X and is closed under the rules

(Id), (Tr) and (Ext) above. Let $R' = Cl^{-}(R \cup R^{-1})$ and $P' = Cl(P)$. We show that all elements in R' satisfy the (SyS) conditions in P', by rule induction on $Cl^{-}(R \cup R^{-1})$. For the base cases, either the element is in $R \cup R^{-1}$ or is an identity. In both cases the result is clear. For the inductive step, consider the rule:

$$\frac{(q_1, S_1, \sigma_1, q_2, S_2) \in R' \qquad (q_2, S_2, \sigma_2, q_3, S_3) \in R'}{(q_1, S_1, \sigma_1; \sigma_2, q_3, S_3) \in R'} \text{ (Tr)}$$

and assume that the premises satisfy the (SyS) conditions in P'. Let us write σ for $\sigma_1; \sigma_2$. Suppose $q_1 \xrightarrow{t,i} q_1'$ with $i \in S_1$.

- If $i \in \mathsf{dom}(\sigma_1)$ and $j = \sigma_1(i) \in \mathsf{dom}(\sigma_2)$ then $q_2 \xrightarrow{t,j} q_2'$ with $(q_1', S_1)\, P'_{\sigma_1}\, (q_2', S_2)$, and $q_3 \xrightarrow{t,k} q_3'$ with $(q_2', S_2)\, P'_{\sigma_2}\, (q_3', S_3)$ and $k = \sigma_2(j) = \sigma(i)$. By (Tr) we obtain $(q_1', S_1)\, P'_{\sigma}\, (q_3', S_3)$.
- If $i \in \mathsf{dom}(\sigma_1)$ and $j = \sigma_1(i) \notin \mathsf{dom}(\sigma_2)$ then $q_2 \xrightarrow{t,j} q_2'$ with $(q_1', S_1)\, P'_{\sigma_1}\, (q_2', S_2)$, and $q_3 \xrightarrow{t,k^\bullet} q_3'$ with $(q_2', S_2)\, P'_{\sigma_2[j \mapsto k]}\, (q_3', S_3[k])$ for some k. By (Tr) we obtain $(q_1', S_1)\, P'_{\sigma[i \mapsto k]}\, (q_3', S_3[k])$.
- If $i \notin \mathsf{dom}(\sigma_1)$ then $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1)\, P'_{\sigma_1[i \mapsto j]}\, (q_2', S_2[j])$, for some j, so $q_3 \xrightarrow{t,k^\bullet} q_3'$ with $(q_2', S_2[j])\, P'_{\sigma_2[j \mapsto k]}\, (q_3', S_3[k])$ for some k. By (Tr,Ext), using $\sigma_1[i \mapsto j]; \sigma_2[j \mapsto k] \leq_{S_1, S_3[k]} \sigma[i \mapsto k]$, we get $(q_1', S_1)\, P'_{\sigma[i \mapsto k]}\, (q_3', S_3[k])$.

Now suppose $q_1 \xrightarrow{t,i^\bullet} q_1'$.

- Then, $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1[i])\, P'_{\sigma_1[i \mapsto j]}\, (q_2', S_2[j])$, for some j, so $q_3 \xrightarrow{t,k^\bullet} q_3'$ with
$$(q_2', S_2[j])\, P'_{\sigma_2[j \mapsto k]}\, (q_3', S_3[k])$$
for some k. By (Tr,Ext), $(q_1', S_1[i])\, P'_{\sigma[i \mapsto k]}\, (q_3', S_3[k])$.
- If $k \in \mathsf{rng}(\sigma_2)$ and $j = \sigma_2^{-1}(k) \notin \mathsf{rng}(\sigma_1)$ then $q_2 \xrightarrow{t,j} q_2'$ with $(q_1', S_1[i])\, P'_{\sigma_1[i \mapsto j]}\, (q_2', S_2)$, and $q_3 \xrightarrow{t,k} q_3'$ with $(q_2', S_2)\, P'_{\sigma_2}\, (q_3', S_3)$. By (Tr) obtain $(q_1', S_1[i])\, P'_{\sigma[i \mapsto k]}\, (q_3', S_3)$.
- If $k \in S_3 \setminus \mathsf{rng}(\sigma_2)$ then $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1[i])\, P'_{\sigma_1[i \mapsto j]}\, (q_2', S_2[j])$, for some j, and so $q_3 \xrightarrow{t,k} q_3'$ with $(q_2', S_2[j])\, P'_{\sigma_2[j \mapsto k]}\, (q_3', S_3)$. By (Tr,Ext) we obtain $(q_1', S_1[i])\, P'_{\sigma[i \mapsto k]}\, (q_3', S_3)$.

Consider now the rule:

$$\frac{(q_1, S_1, \sigma, q_2, S_2) \in R' \qquad \sigma \leq_{S_1, S_2} \sigma'}{(q_1, S_1, \sigma', q_2, S_2) \in R'} \text{ (Ext)}$$

and assume $(q_1, S_1, \sigma, q_2, S_2)$ satisfies the (SyS) conditions in P'. Suppose $q_1 \xrightarrow{t,i} q_1'$ with $i \in S_1$.

- If $i \in \mathsf{dom}(\sigma)$ then $q_2 \xrightarrow{t,\sigma(i)} q_2'$ and $(q_1', S_1)\, P'_{\sigma}\, (q_2', S_2)$. Since $\sigma \subseteq \sigma'$, we have $\sigma(i) = \sigma'(i)$ and $(q_1', S_1)\, P'_{\sigma'}\, (q_2', S_2)$.
- If $i \notin \mathsf{dom}(\sigma')$ then also $i \notin \mathsf{dom}(\sigma)$ and therefore $q_2 \xrightarrow{t,j^\bullet} q_2'$, for some j, and
$$(q_1', S_1)\, P'_{\sigma[i \mapsto j]}\, (q_2', S_2[j]).$$
From $\sigma \leq_{S_1, S_2} \sigma'$ we obtain $\sigma[i \mapsto j] \leq_{S_1, S_2[j]} \sigma'[i \mapsto j]$, so $(q_1', S_1)\, P'_{\sigma'[i \mapsto j]}\, (q_2', S_2[j])$.
- If $i \in \mathsf{dom}(\sigma') \setminus \mathsf{dom}(\sigma)$ then we reason as follows. Let $\sigma'(i) = j \in S_2$.

I. Since $i \notin \mathsf{dom}(\sigma)$, there is some $q_2 \xrightarrow{t,j'^{\bullet}} q_2''$ with $(q_1', S_1)\, P'_{\sigma[i\mapsto j']}\, (q_2'', S_2[j'])$;

II. hence, there is some $q_1 \xrightarrow{t,i'^{\bullet}} q_1''$ with $(q_1'', S_1[i'])\, P'_{\sigma[i'\mapsto j']}\, (q_2'', S_2[j'])$;

III. then, there is some $q_2 \xrightarrow{t,j} q_2'$ with $(q_1'', S_1[i'])\, P'_{\sigma[i'\mapsto j]}\, (q_2', S_2)$.

Taking stock (and using symmetry of P'),

$$(q_1', S_1)\, P'_{\sigma[i\mapsto j']}\, (q_2'', S_2[j'])\, P'_{\sigma^{-1}[j'\mapsto i']}\, (q_1'', S_1[i'])\, P'_{\sigma[i'\mapsto j]}\, (q_2', S_2)$$

and thus, since $\sigma[i\mapsto j'];\sigma^{-1}[j'\mapsto i'];\sigma[i'\mapsto j] \leq_{S_1,S_2} \sigma[i\mapsto j]$, we have

$$(q_1', S_1)\, P'_{\sigma[i\mapsto j]}\, (q_2', S_2).$$

Suppose now $q_1 \xrightarrow{t,i^{\bullet}} q_1'$.

- Then, $q_2 \xrightarrow{t,j^{\bullet}} q_2'$ and $(q_1', S_1[i])\, P'_{\sigma[i\mapsto j]}\, (q_2', S_2[j])$. Since $\sigma[i\mapsto j] \leq_{S_1[i],S_2[j]} \sigma'[i\mapsto j]$, we have $(q_1', S_1[i])\, P'_{\sigma'[i\mapsto j]}\, (q_2', S_2[j])$.
- If $j \in S_2 \setminus \mathsf{rng}(\sigma')$ then $j \notin \mathsf{rng}(\sigma)$, hence $q_2 \xrightarrow{t,j} q_2'$ and $(q_1', S_1[i])\, P'_{\sigma[i\mapsto j]}\, (q_2', S_2)$. Again, we obtain $(q_1', S_1[i])\, P'_{\sigma'[i\mapsto j]}\, (q_2', S_2)$.

Hence, all elements of R' satisfy the (SyS) conditions in P'. □

Proof of Lemma 4.13. (First part). Since R is closed, $(p,S)\, R_{\sigma;\sigma^{-1}}\, (p,S)$. Because $\sigma;\sigma^{-1} = \mathsf{id}_X$ for some $X \subseteq S$, we have $X \supseteq X^p_S(R)$. Moreover, $\mathsf{dom}(\sigma) \supseteq \mathsf{dom}(\sigma;\sigma^{-1}) = X$, hence $\mathsf{dom}(\sigma) \supseteq X^p_S(R)$. A symmetric argument establishes that $\mathsf{dom}(\sigma^{-1}) \supseteq X^q_S(R)$.

(Second part). By definition, we have that $\mathsf{dom}(\sigma') \subseteq X^p_S(R)$ and $\mathsf{rng}(\sigma') \subseteq X^q_S(R)$. Observing that $\sigma' = \mathsf{id}_{X^p_S(R)};\sigma;\mathsf{id}_{X^q_S(R)}$, by closure of R we get $(p,S)\, R_{\sigma'}\, (q,S)$. By the first part, $\mathsf{dom}(\sigma') \supseteq X^p_S(R)$ and $\mathsf{rng}(\sigma') \supseteq X^q_S(R)$, hence $\mathsf{dom}(\sigma') = X^p_S(R)$ and $\mathsf{rng}(\sigma') = X^q_S(R)$. The final claim follows from the fact that $(p,S)\, R_{\mathsf{id}_S}\, (p,S)$. □

Proof of Lemma 4.17. Let us write $\gamma(S_1,S_2)$ for $|S_1| + |S_2|$, i.e. $0 \leq \gamma(S_1,S_2) \leq 2r$. For each $m \in [0,2r]$, let

$$k_m = \min\{i \mid \overset{i}{\sim} \cap\, \mathcal{U}_{S_1,S_2} = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}_{S_1,S_2} \text{ for any } S_1,S_2 \text{ with } \gamma(S_1,S_2) \geq m\}.$$

Consider S_1,S_2 with $\gamma(S_1,S_2) \geq m$, where $m < 2r$.

Observe that, for $k \geq k_{m+1}$, if $\overset{k}{\sim} \cap\, \mathcal{U}_{S_1,S_2} = \overset{k+1}{\sim} \cap\, \mathcal{U}_{S_1,S_2}$, then we must have $\overset{k}{\sim} \cap\, \mathcal{U}_{S_1,S_2} = \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}_{S_1,S_2}$, because the (SyS) conditions for (S_1,S_2) refer to either (S_1,S_2) or (S_1',S_2') with $\gamma(S_1',S_2') > \gamma(S_1,S_2)$. Consequently, if $\overset{k}{\sim} \cap\, \mathcal{U}_{S_1,S_2} \neq \overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}_{S_1,S_2}$, the sequence $(\overset{k}{\sim} \cap\, \mathcal{U}_{S_1,S_2})$ $(k = k_{m+1}, k_{m+1}+1, \cdots)$ will have to change in every step before stabilisation. By Lemma 4.16, at most ℓ extra steps from $(\overset{k_{m+1}}{\sim})$ will be required to arrive at $\overset{\mathsf{s}}{\sim} \cap\, \mathcal{U}_{S_1,S_2}$, which implies $k_m \leq k_{m+1} + \ell$. By a similar argument, we can conclude that $k_{2r} \leq \ell$. Consequently, $k_0 \leq (2r+1)\ell$, as required. □

Appendix C. Proofs from Section 7

Proof of Lemma 7.1. For the first claim, note that it suffices to consider the case where the product $ii'jj'$ is not 0 as e.g. if $ii' = 0$ then $[i \leftrightarrow i'] = [1 \leftrightarrow 1]$. In this case, the claim follows by composition of partial permutations, noting that $\rho_2^{-1}; (j\, j') = ((j\, j'); \rho_2)^{-1}$.
Claim 2 then follows as Lemma 4.3 implies that $(\rho_1; \rho_2^{-1})[i' \mapsto j'] = \rho_1[i' \mapsto a]; \rho_2[j' \mapsto a]^{-1}$. □

Proof of Lemma 7.3. Let $(q_1, S_1, \sigma, q_2, S_2, h)$, $(q_1, S_1', \sigma', q_2, S_2', h) \in \mathsf{symb}(\kappa_1, \kappa_2)$ be distinct and produced from $\hat\rho_i$ and $\hat\rho_i'$ respectively (for $i = 1, 2$). Let us assume that $(q_1, S_1', \sigma', q_2, S_2', h) \in \overset{\mathsf{s}}{\sim}$. Take $\sigma_i = \hat\rho_i; \hat\rho_i'^{-1}$. By definition, $\sigma_i \restriction [1, r] = \mathsf{id}_{S_i \cap [1,r]}$, and we can verify that $(q_i, S_i)(\overset{\mathsf{s}}{\sim})^h_{\sigma_i}(q_i, S_i')$. Hence, $(q_1, S_1)\,(\overset{\mathsf{s}}{\sim})^h_{\sigma_1}\,(q_1, S_1')\,(\overset{\mathsf{s}}{\sim})^h_{\sigma'}\,(q_2, S_2')\,(\overset{\mathsf{s}}{\sim})^h_{\sigma_2^{-1}}\,(q_2, S_2)$ and, using Proposition 7.9 (which does not depend on this lemma), we get $(q_1, S_1, \sigma_1; \sigma'; \sigma_2^{-1}$, $q_2, S_2, h) = (q_1, S_1, \sigma, q_2, S_2, h) \in \overset{\mathsf{s}}{\sim}$. □

Proof of Lemma 7.6. Let $\mathcal{A}$ be an r-FRA($S\#_0$). We show a correspondence between bisimulations and symbolic bisimulations for $\mathcal{A}$ from which the result follows.

bisim → s-bisim. Let R be a bisimulation on $\mathcal{A}$. We claim that the relation $P \subseteq \mathcal{U}$,

$$P = \bigcup \{\, \mathsf{symb}(\kappa_1, \kappa_2) \mid (\kappa_1, \kappa_2) \in R \wedge \kappa_i = (q_i, \rho_i, H_i) \wedge H_1 = H_2 \}$$

is a symbolic bisimulation. For the latter (by symmetry) it suffices to show that P is a symbolic simulation, which reduces to showing the (FSyS) conditions true. So suppose that $(q_1, S_1, \sigma, q_2, S_2) \in P^h$ due to some $(q_1, \rho_1, H)R(q_2, \rho_2, H)$. If $h \leq 2r$ then let $\hat\rho_i$ be some $3r$-register assignment of type $S\#_0$ used by symb (for $i = 1, 2$), so $\hat\rho_i \restriction [1, r] = \rho_i$, $S_i = \mathsf{dom}(\hat\rho_i)$, $\mathsf{rng}(\hat\rho_i) = H$ and $\sigma = \hat\rho_1; \hat\rho_2^{-1}$.

Let $q_1 \xrightarrow{t,i} q_1'$ for some $i \in S_1 \cap [1, r]$. Then, $(q_1, \rho_1, H) \xrightarrow{t,a} (q_1', \rho_1, H)$ with $a = \rho_1(i) \in H$ and, hence, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2', H)$ with $(q_1', \rho_1, H)R(q_2', \rho_2', H)$.

- If $\sigma(i) \in [1, r]$ then $a = \rho_2(\sigma(i))$ and therefore the above transition is due to some $q_2 \xrightarrow{t,\sigma(i)} q_2'$, and $\rho_2' = \rho_2$. Hence, $(q_1', S_1)P^h_\sigma(q_2', S_2)$.
- If $\sigma(i) = j' \in [r{+}1, 3r]$ then $a = \hat\rho_2(j') \notin \mathsf{rng}(\rho_2)$ and the above transition is due to some $q_2 \xrightarrow{t,j^\bullet} q_2'$, and $\rho_2' = \rho_2[j \mapsto a]$. Now, taking $\hat\rho_2' = \hat\rho_2[j \leftrightarrow j']$, we have $(q_1', S_1, \hat\rho_1; \hat\rho_2'^{-1}, q_2', \mathsf{dom}(\hat\rho_2')) \in P^h$. Since $\hat\rho_1; \hat\rho_2'^{-1} = [j \leftrightarrow j']\sigma$ and $\mathsf{dom}(\hat\rho_2') = S_2'[j \leftrightarrow j']$, we obtain
$$(q_1', S_1)P^h_{[j \leftrightarrow j']\sigma}(q_2', S_2[j \leftrightarrow j']).$$
- If $i \notin \mathsf{dom}(\sigma)$ then $h = \infty$ and the transition is due to some $q_2 \xrightarrow{t,j^\bullet} q_2'$, and $\rho_2' = \rho_2[j \mapsto a]$. Hence, since $\sigma[i \mapsto j] = \rho_1; (\rho_2[j \mapsto a])^{-1}$ and $\mathsf{dom}(\rho_2') = S_2[j]$, we have
$$(q_1', S_1)P^h_{\sigma[i \mapsto j]}(q_2', S_2[j]).$$

Let $q_1 \xrightarrow{t,i^\bullet} q_1'$. For each $a \in H \setminus \mathsf{rng}(\rho_1)$, $(q_1, \rho_1, H) \xrightarrow{t,a} (q_1', \rho_1', H)$ with $\rho_1' = \rho_1[i \mapsto a]$ and, hence, there is some $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2', H)$ with $(q_1', \rho_1', H)R(q_2', \rho_2', H)$. Now, let $a = \hat\rho_1(i')$ for $i' \in S_1 \setminus [1, r]$ (if $h \leq 2r$), and $a = \hat\rho_2(j)$ for $j \in S_2 \setminus \mathsf{rng}(\sigma)$ (if $h = \infty$); in the former case, set $\hat\rho_1' = \hat\rho_1[i \leftrightarrow i']$.

- If $\sigma(i') \in [1,r]$ then $a = \rho_2(\sigma(i'))$ so the transition above is due to some $q_2 \xrightarrow{t,\sigma(i')} q_2'$ and $\rho_2' = \rho_2$. Thus, $(q_1', \mathsf{dom}(\hat\rho_1'))P^h_{\hat\rho_1';\hat\rho_2^{-1}}(q_2', S_2)$ i.e. $(q_1', S_1[i \leftrightarrow i'])P^h_{\sigma[i\leftrightarrow i']}(q_2', S_2)$.
- If $\sigma(i') = j' \in [r{+}1, 3r]$ then $a = \hat\rho_2(j') \notin \mathsf{rng}(\rho_2)$ so the transition above is due to some $q_2 \xrightarrow{t,j^\bullet} q_2'$ and $\rho_2' = \rho_2[j \mapsto a]$. Thus, setting $\hat\rho_2' = \hat\rho_2[j \leftrightarrow j]$, we obtain
$$(q_1', \mathsf{dom}(\hat\rho_1'))P^h_{\hat\rho_1';\hat\rho_2'^{-1}}(q_2', \mathsf{dom}(\hat\rho_2')),$$
i.e. $(q_1', S_1[i \leftrightarrow i'])P^h_{[j\leftrightarrow j']\sigma[i\leftrightarrow i']}(q_2', S_2[j \leftrightarrow j'])$.
- For $a = \hat\rho_2(j)$ with $j \in S_2 \setminus \mathsf{rng}(\sigma)$, the transition is due to some $q_2 \xrightarrow{t,j} q_2'$, and $\rho_2' = \rho_2$. We moreover have $(q_1', S_1[i])P^h_{\sigma[i\mapsto j]}(q_2', S_2)$.

Finally, let $q_1 \xrightarrow{t,\ell_i} q_1'$ with $\ell_i \in \{i^\bullet, i^{\circledast}\}$. For each $a \notin H$, we have $(q_1, \rho_1, H) \xrightarrow{t,a} (q_1', \rho_1', H')$ with $\rho_1' = \rho_1[i \mapsto a]$ and $H' = H \cup \{a\}$ and, hence, there is some $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2', H')$ with $(q_1', \rho_1', H')R(q_2', \rho_2', H')$. The latter must be due to $q_2 \xrightarrow{t,\ell_j} q_2'$, for some $\ell_j \in \{j^\bullet, j^{\circledast}\}$, in which case $\rho_2' = \rho_2[j \mapsto a]$.

- If $h < 2r$ then let $\hat\rho_1' = \hat\rho_1[i' \mapsto a][i \leftrightarrow i']$ and $\hat\rho_2' = \hat\rho_2[j' \mapsto a][j \leftrightarrow j']$, where $i' = \min([r{+}1, 3r] \setminus \mathsf{dom}(\hat\rho_1))$ and $j' = \min([r{+}1, 3r] \setminus \mathsf{dom}(\hat\rho_2))$. We have $\rho_1' = \hat\rho_1' \upharpoonright [1,r]$, similarly for ρ_2', and $\hat\rho_1'; \hat\rho_2'^{-1} = [j \leftrightarrow j'](\sigma[i' \mapsto j'])[i \leftrightarrow i']$, so
$$(q_1', S_1[i \leftrightarrow i'])P^{h+1}_{[j\leftrightarrow j'](\sigma[i'\mapsto j'])[i\leftrightarrow i']}(q_2', S_2[j \leftrightarrow j']).$$
- If $h = 2r$ then $(q_1', \mathsf{dom}(\rho_1'))P^\infty_{\rho_1';\rho_2'^{-1}}(q_2', \mathsf{dom}(\rho_2'))$. Now observe that $\hat\rho_1[i \mapsto a] \upharpoonright [1,r] = \rho_1'$, similarly for ρ_2', and hence $\sigma[i \mapsto j] \cap [1,r]^2 = \rho_1'; \rho_2'^{-1}$.
- If $h = \infty$ then, since $\sigma[i \mapsto j] = \rho_1[i \mapsto a]; (\rho_2[j \mapsto a])^{-1}$, $\mathsf{dom}(\rho_1') = S_1[i]$ and $\mathsf{dom}(\rho_2') = S_2[j]$, we have $(q_1', S_1[i])P^h_{\sigma[i\mapsto j]}(q_2', S_2[j])$. Moreover, if $\ell_i = i^\bullet$ then, since $|H| > |\mathsf{rng}(\rho_1)| + |\mathsf{rng}(\rho_2)|$, there is some $a' \in H \setminus (\mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2))$. We can therefore pick $a = a'$ and the latter would impose $\ell_j = j^\bullet$.

Hence, P is a symbolic bisimulation.

s-bisim $\to$ bisim. Let R be a symbolic bisimulation on $\mathcal{A}$ such that, for all pairs of configurations κ_1, κ_2, either $\mathsf{symb}(\kappa_1, \kappa_2) \subseteq R$ or $\mathsf{symb}(\kappa_1, \kappa_2) \cap R = \emptyset$. We claim that the relation
$$R' = \{\, (\kappa_1, \kappa_2) \mid \kappa_i = (q_i, \rho_i, H_i) \wedge H_1 = H_2 \wedge \mathsf{symb}(\kappa_1, \kappa_2) \subseteq R \,\}$$
is a bisimulation, for which it suffices to show that R' is a simulation. So suppose that
$$((q_1, \rho_1, H), (q_2, \rho_2, H)) \in R'$$
and let $(q_1, S_1, \sigma, q_2, S_2, h) \in \mathsf{symb}((q_1, \rho_1, H), (q_2, \rho_2, H)) \subseteq R$, and if $h \le 2r$ let $\hat\rho_i$ be some $3r$-extension of ρ_i used by symb. Let $(q_1, \rho_1, H) \xrightarrow{t,a} (q_1', \rho_1', H')$ for some $(t, a) \in \Sigma \times \mathcal{D}$.

If $a \in \mathsf{rng}(\rho_1)$, say $a = \rho_1(i)$, then $q_1 \xrightarrow{t,i} q_1'$ and $\rho_1' = \rho_1$. We distinguish three cases:

- If $a \in \mathsf{rng}(\rho_2)$ then $\sigma(i) \in [1,r]$, so $q_2 \xrightarrow{t,\sigma(i)} q_2'$ and $(q_1', S_1)R^h_\sigma(q_2', S_2)$. Hence, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2, H)$ and $(q_1', \rho_1, H)R'(q_2', \rho_2, H)$.
- If $a \notin \mathsf{rng}(\rho_2)$ and $h \le 2r$ then $\sigma(i) = j' \in [r{+}1, 3r]$, so $q_2 \xrightarrow{t,j^\bullet} q_2'$ and
$$(q_1', S_1)R^h_{[j\leftrightarrow j']\sigma}(q_2', S_2[j \leftrightarrow j']),$$

for some j. Hence, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2[j \mapsto a], H)$ and, taking $\hat{\rho}_2' = \hat{\rho}_2[j \leftrightarrow j']$ (so $\hat{\rho}_2' \restriction [1,r] = \rho_2[j \mapsto a]$), we have $(q_1', \mathsf{dom}(\hat{\rho}_1)) R^h_{\hat{\rho}_1;\hat{\rho}_2'^{-1}} (q_2', \mathsf{dom}(\hat{\rho}_2'))$ hence

$$(q_1', \rho_1, H) R' (q_2', \rho_2[j \mapsto a], H).$$

- If $a \notin \mathsf{rng}(\rho_2)$ and $h = \infty$ then $i \in S_1 \setminus \mathsf{dom}(\sigma)$, so $q_2 \xrightarrow{t,j^\bullet} q_2'$ and $(q_1', S_1) R^h_{\sigma[i \mapsto j]} (q_2', S_2[j])$. Hence, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2[j \mapsto a], H)$ and this $(q_1', \rho_1, H) R' (q_2', \rho_2[j \mapsto a], H)$.

If $a \in H \setminus \mathsf{rng}(\rho_1)$, and either $h \leq 2r$ (so $a = \hat{\rho}_1(i')$ for some $i' > r$) or $h = \infty$ and $a \in \mathsf{rng}(\rho_2)$, then $H' = H$ and there is some $q_1 \xrightarrow{t,i^\bullet} q_1'$ and $\rho_1' = \rho_1[i \mapsto a]$.

- If $h \leq 2r$ and $\sigma(i') \in [1,r]$ then $q_2 \xrightarrow{t,\sigma(i')} q_2'$ and $(q_1', S_1[i \leftrightarrow i']) R^h_{\sigma[i \leftrightarrow i']} (q_2', S_2)$. Thus, since $\rho_2(\sigma(i')) = a$, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2, H)$ and, setting $\hat{\rho}_1' = \hat{\rho}_1[i \leftrightarrow i']$, we obtain $(q_1', \rho_1', H) R' (q_2', \rho_2, H)$.
- If $h \leq 2r$ and $\sigma(i') = j' \in [r{+}1, r]$ then $q_2 \xrightarrow{t,j^\bullet} q_2'$ with $(q_1', S_1[i \leftrightarrow i']) R^h_{[j \leftrightarrow j']\sigma[i \leftrightarrow i']}$ $(q_2', S_2[j \leftrightarrow j'])$, for some j. Thus, since $a \notin \mathsf{rng}(\rho_2)$, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2[j \mapsto a], H)$ and, setting $\hat{\rho}_1' = \hat{\rho}_1[i \leftrightarrow i']$ and $\hat{\rho}_2' = \hat{\rho}_2[j \leftrightarrow j']$, we obtain $(q_1', \rho_1', H) R' (q_2', \rho_2[j \mapsto a], H)$.
- If $h = \infty$ and $a \in \mathsf{rng}(\rho_2)$, say $a = \rho_2(j)$, then $j \in S_2 \setminus \mathsf{rng}(\sigma)$. Hence, $q_2 \xrightarrow{t,j} q_2'$ with $(q_1', S_1[i]) R_{\sigma[i \mapsto j]} (q_2', S_2)$, from which we get $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2, H)$ and

$$(q_1', \rho_1', H) R' (q_2', \rho_2, H).$$

If either $h \leq 2r$ and $a \notin H$, or $h = \infty$ and $a \notin \mathsf{rng}(\rho_1) \cup \mathsf{rng}(\rho_2)$ then $q_1 \xrightarrow{t,\ell_i} q_1'$, for some $\ell_i \in \{i^\bullet, i^\circledast\}$, and $H' = H \cup \{a\}$ and $\rho_1' = \rho_i[i \mapsto a]$. Thus, $q_2 \xrightarrow{t,\ell_j} q_2'$ for some $\ell_j \in \{j^\bullet, j^\circledast\}$. Let $\rho_2' = \rho_2[j \mapsto a]$.

- If $h < 2r$ then, taking $i' = \max([r{+}1, 3r] \setminus S_1)$ and $j' = \max([r{+}1, 3r] \setminus S_2)$, we have $(q_1', S_1[i \leftrightarrow i']) R^{h+1}_{[j \leftrightarrow j']\sigma[i' \mapsto j'][i \leftrightarrow i']} (q_2', S_2[j \leftrightarrow j'])$. Setting $\hat{\rho}_1' = \hat{\rho}_1[i' \mapsto a][i \leftrightarrow i']$ and $\hat{\rho}_2' = \hat{\rho}_2[j' \mapsto a][j \leftrightarrow j']$, we obtain $(q_1', \rho_1', H') R' (q_2', \rho_2', H')$.
- If $h = 2r$ then $(q_1', S_1[i] \cap [1,r]) R^\infty_{\sigma[i \mapsto j] \cap [1,r]^2} (q_2', S_2[j] \cap [1,r])$, from which we obtain

$$(q_1', \rho_1', H') R' (q_2', \rho_2', H').$$

- If $h = \infty$ then $(q_1', S_1[i]) R^h_{\sigma[i \mapsto j]} (q_2', S_2[j])$. In particular, if $a \in H$ then $\ell_i = i^\bullet$ and therefore $\ell_j = j^\bullet$. Thus, in each case, $(q_2, \rho_2, H) \xrightarrow{t,a} (q_2', \rho_2[j \mapsto a], H')$ and $(q_1', \rho_1', H') R' (q_2', \rho_2', H')$.

Hence, R' is a bisimulation.

Thus, to prove Lemma 7.6, given such κ_1 and κ_2, if $\kappa_1 \overset{\mathsf{s}}{\sim} \kappa_2$ then we can construct a symbolic bisimulation P such that $\mathsf{symb}(\kappa_1, \kappa_2) \subseteq P$. Conversely, if $\kappa_1 \overset{\mathsf{s}}{\sim} \kappa_2$ then, using also Lemma 7.3, there is a bisimulation R' such that $\kappa_1 R' \kappa_2$. □

Proof of Lemma 7.7. For the first part, we argue by induction on i. For $i = 0$ we need to show $\overset{1}{\sim} \subseteq \overset{0}{\sim}$, which is true because $\overset{0}{\sim} = \mathcal{U}$. Next, assuming $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$, we argue that $\overset{i+2}{\sim} \subseteq \overset{i+1}{\sim}$. Suppose (q_1, S_1) $(\overset{i+2}{\sim})^h_\tau$ (q_2, S_2). It follows by definition that $(q_1, S_1, \tau, q_2, S_2, h)$ and $(q_2, S_2, \tau^{-1}, q_1, S_1, h)$ satisfy the (FSyS) conditions in $\overset{i+1}{\sim}$. Because $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$, the tuples also satisfy the (FSyS) conditions in $\overset{i}{\sim}$, which implies (q_1, S_1) $(\overset{i+1}{\sim})^h_\tau$ (q_2, S_2).

For the second part, we start with $\supseteq$ and argue that, for all $i \in \omega$, $\overset{i}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$. The proof is by induction on i. For $i = 0$ the result is trivial, because $\overset{0}{\sim} = \mathcal{U}$. Next, assuming $\overset{i}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$, we will show $\overset{i+1}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$. Suppose $(q_1, S_1)(\overset{\mathsf{s}}{\sim})^h_\tau(q_2, S_2)$. We wish to show that $(q_1, S_1, \tau, q_2, S_2, h)$ and its inverse satisfy the (FSyS) conditions in $\overset{i}{\sim}$. By definition, they satisfy the (FSyS) conditions in $\overset{\mathsf{s}}{\sim}$. Because of $\overset{i}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$, this implies that they satisfy the (FSyS) conditions in $\overset{i}{\sim}$. Hence, $\overset{i+1}{\sim} \supseteq \overset{\mathsf{s}}{\sim}$, as required.

For the $\subseteq$ direction, we argue that the left-hand side is a symbolic bisimulation. To see this, assume $(q_1, S_1, \tau, q_2, S_2, h) \in \bigcap_{i\in\omega} \overset{i}{\sim}$ so that $(q_1, S_1, \tau, q_2, S_2, h)$ and its inverse satisfy the (FSyS) conditions in $\overset{i}{\sim}$, for all $i \in \omega$. The satisfaction of the (FSyS) conditions in $\overset{i}{\sim}$ by $(q_1, S_1, \tau, q_2, S_2, h)$ (and, analogously, by its inverse) is witnessed by a subset $C_i \subseteq \overset{i}{\sim} \subseteq \mathcal{U}$ for each i. Because $\mathcal{U}$ is finite, there exists C such that $C = C_i$ for infinitely many i. Consequently, in view of $\overset{i+1}{\sim} \subseteq \overset{i}{\sim}$, C witnesses satisfaction of the (FSyS) conditions in $(\bigcap_{i\in\omega} \overset{i}{\sim} q)$. □

Proof of Lemma 7.8. We first observe that $Cl(R) = Cl^-(R \cup R^{-1})$ where, for any relation X, we let $Cl^-(X)$ be the smallest relation that contains X and is closed under the rules (Id), (Tr) and (Ext) above. Let $\hat{R} = Cl^-(R \cup R^{-1})$ and $\hat{P} = Cl(P)$. We show that all elements in $\hat{R}$ satisfy the (FSyS) conditions in $\hat{P}$, by rule induction on $Cl^-(R \cup R^{-1})$. For the base cases, either the element is in $R \cup R^{-1}$ or is an identity. In both cases the result is clear. For the inductive step, consider the rule:

$$\frac{(q_1, S_1, \sigma_1, q_2, S_2) \in \hat{R}^h \qquad (q_2, S_2, \sigma_2, q_3, S_3) \in \hat{R}^h}{(q_1, S_1, \sigma_1; \sigma_2, q_3, S_3) \in \hat{R}^h}\ (\textsc{Tr})$$

and assume that the premises satisfy the (FSyS) conditions in $\hat{P}$. Let us write σ for $\sigma_1; \sigma_2$. Suppose $q_1 \xrightarrow{t,i_1} q_1'$.

- If $\sigma_1(i_1) = i_2 \in [1, r]$ then, by the (FSyS) conditions on $(q_1, S_1, \sigma_1, h, q_2, S_2)$, we have $q_2 \xrightarrow{t,i_2} q_2'$ with $j_2 = \sigma_1(j_1)$ and $(q_1', S_1)\hat{P}^h_{\sigma_1}(q_2', S_2)$.
  - If $\sigma_2(i_2) = i_3 \in [1, r]$ then $q_3 \xrightarrow{t,i_3} q_3'$ with $(q_2', S_2)\hat{P}^h_{\sigma_2}(q_3', S_3)$. By (Tr), $(q_1', S_1)\hat{P}^h_{\sigma}(q_3', S_3)$.
  - If $\sigma_2(i_2) = i_3' \in [r{+}1, 3r]$ then $q_3 \xrightarrow{t,i_3^\bullet} q_3'$ with $(q_2', S_2)\hat{P}^h_{[i_3\leftrightarrow i_3']\sigma_2}(q_3', S_3[i_3 \leftrightarrow i_3'])$. By (Tr), and using also Lemma 2.7, we obtain $(q_1', S_1)\hat{P}^h_{[i_3\leftrightarrow i_3']\sigma}(q_3', S_3[i_3 \leftrightarrow i_3'])$, as required.
  - If $i_2 \in S_2 \setminus \mathsf{dom}(\sigma_2)$ then $q_3 \xrightarrow{t,i_3^\bullet} q_3'$ with $(q_2', S_2)\hat{P}^h_{\sigma_2[i_2\mapsto i_3]}(q_3', S_3[i_3])$. By (Tr), we obtain $(q_1', S_1)\hat{P}^h_{\sigma_1;\sigma_2[i_2\mapsto i_3]}(q_3', S_3[i_3])$, which is what is required since $\sigma[i_1 \mapsto i_3] = \sigma_1; \sigma_2[i_2 \mapsto i_3]$.
- If $\sigma_1(i_1) = i_2' \in [r{+}1, 3r]$ then $q_2 \xrightarrow{t,i_2^\bullet} q_2'$ with $(q_1', S_1)\hat{P}^h_{[i_2\leftrightarrow i_2']\sigma_1}(q_2', S_2[i_2 \leftrightarrow i_2'])$.
  - If $\sigma_2(i_2') = i_3 \in [1, r]$ then $q_3 \xrightarrow{t,i_3} q_3'$ with $(q_2', S_2[i_2 \leftrightarrow i_2'])\hat{P}^h_{\sigma_2[i_2\leftrightarrow i_2']}(q_3', S_3)$. By (Tr) we obtain $(q_1', S_1)\hat{P}^h_{\sigma}(q_3', S_3)$.
  - If $\sigma_2(i_2') = i_3' \in [r{+}1, 3r]$ then $q_3 \xrightarrow{t,i_3^\bullet} q_3'$ with $(q_2', S_2[i_2 \leftrightarrow i_2'])\hat{P}^h_{[i_3\leftrightarrow i_3']\sigma_2[i_2\leftrightarrow i_2']}$ $(q_3', S_3[i_3 \leftrightarrow i_3'])$. By (Tr) we have $(q_1', S_1)\hat{P}^h_{[i_3\leftrightarrow i_3']\sigma}(q_3', S_3[i_3 \leftrightarrow i_3'])$.

- If $i_1 \in S_1 \setminus \mathsf{dom}(\sigma_1)$ then we have $h = \infty$ and $q_2 \xrightarrow{t,i_2^\bullet} q_2'$ with $(q_1', S_1)\hat{P}^h_{\sigma_1[i_1 \mapsto i_2]}(q_2', S_2[i_2])$, for some i_2, so $q_3 \xrightarrow{t,i_3^\bullet} q_3'$ with $(q_2', S_2[i_2])\hat{P}^h_{\sigma_2[i_2 \mapsto i_3]}(q_3', S_3[i_3])$ for some i_3. By (Tr,Ext), using $\sigma_1[i_1 \mapsto i_2]; \sigma_2[i_2 \mapsto i_3] \leq_{S_1, S_3[i_3]} \sigma[i_1 \mapsto i_3]$, we get $(q_1, S_1)\hat{P}^h_{\sigma[i_1 \mapsto i_3]}(q_3, S_3[i_3])$.

Now suppose $q_1 \xrightarrow{t,i_1^\bullet} q_1'$ and let $i_1' \in S_1 \setminus [1, r]$ (so $h \leq 2r$).

- If $\sigma_1(i_1') = i_2 \in [1, r]$ then $q_2 \xrightarrow{t,i_2} q_2'$ with $(q_1', S_1[i_1 \leftrightarrow i_1'])\hat{P}^h_{\sigma_1[i_1 \leftrightarrow i_1']}(q_2', S_2)$.
  - If $\sigma_2(i_2) = i_3 \in [1, r]$ then $q_3 \xrightarrow{t,i_3} q_3'$ with $(q_2', S_2)\hat{P}^h_{\sigma_2}(q_3', S_3)$. By (Tr) we obtain
$$(q_1', S_1[i_1 \leftrightarrow i_1'])\hat{P}^h_{\sigma[i_1 \leftrightarrow i_1']}(q_3', S_3).$$
  - If $\sigma_2(i_2) = i_3' \in [r{+}1, 3r]$ then $q_3 \xrightarrow{t,i_3^\bullet} q_3'$ with $(q_2', S_2)\hat{P}^h_{[i_3 \leftrightarrow i_3']\sigma_2}(q_3', S_3[i_3 \leftrightarrow i_3'])$. By (Tr) we have $(q_1', S_1[i_1 \leftrightarrow i_1'])\hat{P}^h_{[i_3 \leftrightarrow i_3']\sigma[i_1 \leftrightarrow i_1']}(q_3', S_3[i_3 \leftrightarrow i_3'])$.
- If $\sigma_1(i_1') = i_2' \in [r{+}1, 3r]$ then $q_2 \xrightarrow{t,i_2^\bullet} q_2'$ with $(q_1', S_1[i_1 \leftrightarrow i_1'])\hat{P}^h_{[i_2 \leftrightarrow i_2']\sigma_1[i_1 \leftrightarrow i_1']}(q_2', S_2[i_2 \leftrightarrow i_2'])$.
  - If $\sigma_2(i_2') = i_3 \in [1, r]$ then $q_3 \xrightarrow{t,i_3} q_3'$ with $(q_2', S_2[i_2 \leftrightarrow i_2'])\hat{P}^h_{\sigma_2[i_2 \leftrightarrow i_2']}(q_3', S_3)$. By (Tr) we obtain $(q_1', S_1[i_1 \leftrightarrow i_1'])\hat{P}^h_{\sigma[i_1 \leftrightarrow i_1']}(q_3', S_3)$.
  - If $\sigma_2(i_2') = i_3' \in [r{+}1, 3r]$ then $q_3 \xrightarrow{t,i_3^\bullet} q_3'$ with $(q_2', S_2[i_2 \leftrightarrow i_2'])\hat{P}^h_{[i_3 \leftrightarrow i_3']\sigma_2[i_2 \leftrightarrow i_2']}(q_3', S_3[i_3 \leftrightarrow i_3'])$. By (Tr), $(q_1', S_1[i_1 \leftrightarrow i_1'])\hat{P}^h_{[i_3 \leftrightarrow i_3']\sigma[i_1 \leftrightarrow i_1']}(q_3', S_3[i_3 \leftrightarrow i_3'])$.

On the other hand, if $q_1 \xrightarrow{t,i_1^\bullet} q_1'$ and $i_3 \in S_3 \setminus \mathsf{rng}(\sigma)$ (so $h = \infty$).

- If $i_3 \in \mathsf{rng}(\sigma_2)$ and $i_2 = \sigma_2^{-1}(i_3) \notin \mathsf{rng}(\sigma_1)$ then $q_2 \xrightarrow{t,i_2} q_2'$ with $(q_1', S_1[i_1])\hat{P}^h_{\sigma_1[i_1 \mapsto i_2]}(q_2', S_2)$, and so $q_3 \xrightarrow{t,i_3} q_3'$ with $(q_2', S_2)\hat{P}^h_{\sigma_2}(q_3', S_3)$. By (Tr) obtain $(q_1', S_1[i])\hat{P}^h_{\sigma[i_1 \mapsto i_3]}(q_3', S_3)$.
- If $i_3 \in S_3 \setminus \mathsf{rng}(\sigma_2)$ then, since $q_2 \xrightarrow{t,i_2^\bullet} q_2'$ with $(q_1', S_1[i_1])\hat{P}^h_{\sigma_1[i_1 \mapsto i_2]}(q_2, S_2[i_2])$ for some i_2, we also have $q_3 \xrightarrow{t,i_3} q_3'$ with $(q_2', S_2[i_2])\hat{P}^h_{\sigma_2[i_2 \mapsto i_3]}(q_3', S_3)$. By (Tr,Ext) we obtain
$$(q_1', S_1[i_1])\hat{P}^h_{\sigma[i_1 \mapsto i_3]}(q_3', S_3).$$

Finally, let $q_1 \xrightarrow{t,i_1^\bullet/i_1^\circledast} q_1'$. Then, $q_2 \xrightarrow{t,i_2^\bullet/i_2^\circledast} q_2'$ and $q_3 \xrightarrow{t,i_3^\bullet/i_3^\circledast} q_3'$ with $(q_1', S_1')\hat{P}^{h'}_{\sigma_1'}(q_2', S_2')$ and $(q_2', S_2')\hat{P}^{h'}_{\sigma_2'}(q_3', S_3')$.

- If $h < 2r$ then $h' = h + 1$ and $i_k' = \min([r{+}1, 3r] \setminus S_k)$, $S_k' = S_k[i_k'][i_k \leftrightarrow i_k']$ and $\sigma_k' = [i_{k+1} \leftrightarrow i_{k+1}'](\sigma_k[i_k' \mapsto i_{k+1}'])[i_k \leftrightarrow i_k']$, for $k = 1, 2, 3$. By (Tr), we have $(q_1', S_1')\hat{P}^h_{\sigma_1'; \sigma_2'}(q_3', S_3')$, which is as required since $\sigma_1'; \sigma_2' = [i_3 \leftrightarrow i_3'](\sigma[i_1' \mapsto i_3'])[i_1 \leftrightarrow i_1']$.
- If $h = 2r$ then $h' = \infty$ and $S_k' = S_k[i_k] \cap [1, r]$ and $\sigma_k' = \sigma_k[i_k \mapsto i_{k+1}] \cap [1, r]^2$. By (Tr), we have $(q_1', S_1')\hat{P}^h_{\sigma_1'; \sigma_2'}(q_3', S_3')$ and, hence, by (Ext) we obtain the required result since $\sigma_1'; \sigma_2' \leq_{S_1', S_2'} \sigma[i_1 \mapsto i_3] \cap [1, r]^2$.
- If $h = \infty$ then $h' = \infty$ and $S_k' = S_k[i_k]$ and $\sigma_k' = \sigma_k[i_k \mapsto i_{k+1}]$. By (Tr,Ext),
$$(q_1', S_1[i_1])\hat{P}^h_{\sigma[i_1 \mapsto i_3]}(q_3', S_3[i_3]).$$

Moreover, if the transition from q_1 to q_1' is localy fresh then so is the one from q_2 to q_2', and from q_3 to q_3'.

We now consider the rule:

$$\frac{(q_1, S_1, \sigma, q_2, S_2) \in \hat{R}^h \qquad \sigma \leq_{S_1,S_2} \sigma'}{(q_1, S_1, \sigma', q_2, S_2) \in \hat{R}^h} \text{ (Ext)}$$

and assume $(q_1, S_1, \sigma, h, q_2, S_2)$ satisfies the (FSyS) conditions in $\hat{P}$. Note that if $h < \infty$ then $h = |\sigma| = |\sigma'|$, hence $\sigma = \sigma'$ and the required result is trivial. So let us assume $h = \infty$. Suppose $q_1 \xrightarrow{t,i} q_1'$.

- If $i \in \mathsf{dom}(\sigma)$ then $q_2 \xrightarrow{t,\sigma(i)} q_2'$ and $(q_1', S_1)\hat{P}^\infty_\sigma(q_2', S_2)$. Since $\sigma \subseteq \sigma'$, we have $\sigma(i) = \sigma'(i)$ and $(q_1', S_1)\hat{P}^\infty_{\sigma'}(q_2', S_2)$.
- If $i \notin \mathsf{dom}(\sigma')$ then also $i \notin \mathsf{dom}(\sigma)$ and therefore $q_2 \xrightarrow{t,j^\bullet} q_2'$, for some j, and
$$(q_1', S_1)\hat{P}^\infty_{\sigma[i\mapsto j]}(q_2', S_2[j]).$$
From $\sigma \leq_{S_1,S_2} \sigma'$ we obtain $\sigma[i \mapsto j] \leq_{S_1,S_2[j]} \sigma'[i \mapsto j]$, so $(q_1', S_1)\hat{P}^\infty_{\sigma'[i\mapsto j]}(q_2', S_2[j])$.
- If $i \in \mathsf{dom}(\sigma') \setminus \mathsf{dom}(\sigma)$ then we reason as follows. Let $\sigma'(i) = j \in S_2$.
  I. Since $i \notin \mathsf{dom}(\sigma)$, there is some $q_2 \xrightarrow{t,j'^\bullet} q_2''$ with $(q_1', S_1)\hat{P}^\infty_{\sigma[i\mapsto j']}(q_2'', S_2[j'])$;
  II. hence, there is some $q_1 \xrightarrow{t,i'^\bullet} q_1''$ with $(q_1'', S_1[i'])\hat{P}^\infty_{\sigma[i'\mapsto j']}(q_2'', S_2[j'])$;
  III. then, there is some $q_2 \xrightarrow{t,j} q_2'$ with $(q_1'', S_1[i'])\hat{P}^\infty_{\sigma[i'\mapsto j]}(q_2', S_2)$.

  Taking stock (and using symmetry of $\hat{P}$),
$$(q_1', S_1)\hat{P}^\infty_{\sigma[i\mapsto j']}(q_2'', S_2[j'])\hat{P}^\infty_{\sigma^{-1}[j'\mapsto i']}(q_1'', S_1[i'])\hat{P}^\infty_{\sigma[i'\mapsto j]}(q_2', S_2)$$
and thus, since $\sigma[i \mapsto j']; \sigma^{-1}[j' \mapsto i']; \sigma[i' \mapsto j] \leq_{S_1,S_2} \sigma[i \mapsto j] \leq_{S_1,S_2} \sigma'$, we have $(q_1', S_1)\hat{P}^\infty_{\sigma'}(q_2', S_2)$.

Suppose now $q_1 \xrightarrow{t,i^\bullet} q_1'$.

- Then, $q_2 \xrightarrow{t,j^\bullet} q_2'$ and $(q_1', S_1[i])\hat{P}^\infty_{\sigma[i\mapsto j]}(q_2', S_2[j])$. Since $\sigma[i \mapsto j] \leq_{S_1[i],S_2[j]} \sigma'[i \mapsto j]$, we have $(q_1', S_1[i])\hat{P}^\infty_{\sigma'[i\mapsto j]}(q_2', S_2[j])$.
- If $j \in S_2 \setminus \mathsf{rng}(\sigma')$ then $j \notin \mathsf{rng}(\sigma)$, hence $q_2 \xrightarrow{t,j} q_2'$ and $(q_1', S_1[i])\hat{P}^\infty_{\sigma[i\mapsto j]}(q_2', S_2)$. Again, we obtain $(q_1', S_1[i])\hat{P}^\infty_{\sigma'[i\mapsto j]}(q_2', S_2)$.

Finally, let $q_1 \xrightarrow{t,i^\circledast} q_1'$.

- Then, $q_2 \xrightarrow{t,j^\circledast} q_2'$ and $(q_1', S_1[i])\hat{P}^\infty_{\sigma[i\mapsto j]}(q_2', S_2[j])$. Since $\sigma[i \mapsto j] \leq_{S_1[i],S_2[j]} \sigma'[i \mapsto j]$, we have $(q_1', S_1[i])\hat{P}^\infty_{\sigma'[i\mapsto j]}(q_2', S_2[j])$.

Hence, $\hat{R}$ satisfies the (FSyS) conditions in $\hat{P}$. □

Appendix D. PSPACE completeness of inverse subsemigroup membership

Given an inverse semigroup $\mathcal{G}$, an *inverse subsemigroup* of $\mathcal{G}$ is some inverse semigroup $\mathcal{H} \subseteq \mathcal{G}$. The problem of inverse subsemigroup membership of $\mathcal{G}$:

For a set G of elements of $\mathcal{G}$ and a distinguished element g of $\mathcal{G}$, does $g \in \langle G \rangle$?

where $\langle G \rangle$ is the inverse semigroup generated by the members of G via composition and inversion. In this section we prove the following result.

Theorem D.1. *Checking membership in inverse subsemigroups of $\mathcal{IS}_n$ is PSPACE-complete.*

Note first that PSPACE membership follows from Kozen's corresponding PSPACE result for functions, as members of $\mathcal{IS}_n$ can be seen as functions on $[1, n] \cup \{\#\}$.

Theorem D.2 [Koz77]**.** *Checking whether a function $h : [1, n] \to [1, n]$ can be generated from given functions $f_1, \cdots, f_k : [1, n] \to [1, n]$ is PSPACE-complete.*

For hardness, we shall make use of a result of Lewis and Papadimitriou which shows that PSPACE computations correspond to computations performed in polynomial space by Turing machines with symmetric transitions.

Definition D.3 [LP82]**.** A *symmetric Turing Machine* is a tuple $\mathcal{M} = \langle Q, q_0, \delta, F \rangle$ where:

- Q is a set of states, $q_0 \in Q$ is initial and $F \subseteq Q$ are final,
- $\delta \subseteq (Q \times \{0, 1\} \times \{0\} \times \{0, 1\} \times Q) \cup (Q \times \{0, 1\}^2 \times \{-1, +1\} \times \{0, 1\}^2 \times Q)$ is the transition relation,

such that $\delta = \delta^{-1}$, where $\delta^{-1} = \{t^{-1} \mid t \in \delta\}$ and:

- $(q, a, 0, b, q')^{-1} = (q', b, 0, a, q)$,
- $(q, a, b, A, c, d, q')^{-1} = (q', c, d, -A, a, b, q)$.

Note that our machines have input and tape alphabet $\{0, 1\}$. Moreover, since we are only examining machines running in polynomial space, we assume a single tape (i.e. no separate input/work tapes), which is initially empty.[12] A symmetric TM $\mathcal{M}$ operates just as a TM, with the feature that $\mathcal{M}$ can look 2 symbols ahead:[13] e.g. a transition $(q, a, b, +1, c, d, q')$ means that, if the automaton is at state q, with the tape symbol at the head being a and the tape symbol to the right of the head being b, then the automaton will rewrite those symbols to c, d respectively, move the head to the right and go to state q'. In a transition $(q, a, b, -1, c, d, q')$ we have the dual behaviour: the automaton looks one symbol to the left ahead, and moves the head to the left. Transitions of the form $(q, a, 0, b, q')$ leave the head unmoved.

Given $f : \mathbb{N} \to \mathbb{N}$, we let SSPACE($f$) be the class of problems decided by a symmetric TM in space $O(f)$.

Theorem D.4 [LP82]**.** *For any $f : \mathbb{N} \to \mathbb{N}$,*

$$\mathrm{DSPACE}(f) \subseteq \mathrm{SSPACE}(f) \subseteq \mathrm{NSPACE}(f).$$

Hence, setting $\mathrm{SPSPACE} = \bigcup_{i \in \mathbb{N}} \mathrm{SSPACE}(n^i)$, using also Savitch's theorem we have $\mathrm{SPSPACE} = \mathrm{PSPACE}$.

[12] Lewis & Papadimitriou work with multi-tape automata, which they reduce to 2-tape automata with one tape for input and one work tape. The same procedure can be used to reduce to just one tape, retaining the same space complexity if the initial complexity is at least polynomial.

[13] This feature does not add expressiveness to a TM but allows one to define symmetric machines.

Proof of Theorem D.1. It suffices to show that the problem is PSPACE-hard. Suppose that $\mathcal{M}$ is a symmetric TM with set of states $Q = [1, K]$ and a tape of size N. By convention, we assume that the initial state is 1, the initial head position is 1 and the unique final state is K. We will simulate its computation using partial permutations from $\mathcal{IS}_n$, where $n = 2N + N + K + 1$.

The first $2N$ numbers in n are used for modelling the tape, the next N numbers for storing the position of the head on the tape, and the last $K + 1$ ones for storing the current state, where we include an extra dummy state $(K + 1)$ to be used at the beginning of the simulation. The way we model these data (tape, head, state) is by employing $N + 1 + 1$ "tokens" which we distribute among our n numbers as follows:

- One token is shared between $2i - 1$ and $2i$, for each $i \in [1, N]$. This token represents the value of bit i of the tape. E.g. if the tape is $10 \cdots 0$, then we can think of the tokens being on numbers $2, 3, 5, \cdots, 2N - 1$.
- One token is shared between the numbers $2N + 1, \cdots, 3N$. This token represents the position of the head. E.g. if the tape is on position 5, then this token will be on number $2N + 5$.
- One token is shared between the numbers $3N + 1, ..., 3N + K + 1$. This token represents the current state.

Initially, we will require all tokens to be on positions $2i-1$ ($i \in [1, N]$), $2N+1$ and $3N+K+1$. The latter means that the last token is initially placed on the dummy state $K + 1$.

We model transitions as partial permutations that pass on the $2N + 2$ tokens. E.g. consider the transition $t = (3, 0, 0, +1, 1, 0, 5)$.[14] Then, t is modelled by partial permutations:

$$\begin{aligned}\pi_t^i = &\{(2i-1, 2i)\} \cup \{(2(i+1)-1, 2(i+1)-1)\} \\ &\cup \{(j, j) \in [1, 2N] \times [1, 2N] \mid j \neq 2i-1, 2i, 2i+1, 2i+2\} \\ &\cup \ \{(2N+i, 2N+i+1)\} \\ &\cup \ \{(3N+3, 3N+5)\}\end{aligned}$$

for $i \in [1, N-1]$. The first line above says "at position i, read 0 and write 1" and "at position $i + 1$, read 0 and write 0"; the second line "leave the remaning cells unchanged"; third line "move right"; and the fourth one "from state 3 go to state 5". This can be generalised to all of δ:

- for all $t = (x, a, b, A, c, d, y)$ and $i \in [1, N]$ such that $i + A \in [1, N]$, set $\pi_t^i = \{(2i-2+A+a, 2i-2+A+c)\} \cup \{(2i+A+b, 2i+A+d)\} \cup \{(j, j) \in [1, 2N] \times [1, 2N] \mid j \notin [2i - 2 + A, 2i + 1 + A]\} \cup \{(2N + i, 2N + i + A)\} \cup \{(3N + x, 3N + y)\}$
- for all $t = (x, a, 0, b, y)$ and $i \in [1, N]$, set $\pi_t^i = \{(2i - 1 + a, 2i - 1 + b)\} \cup \{(j, j) \in [1, 2N] \times [1, 2N] \mid j \notin [2i - 1, 2i]\} \cup \{(2N + i, 2N + i)\} \cup \{(3N + x, 3N + y)\}$

Note that, in the latter case, $(\pi_t^i)^{-1} = \pi_{t^{-1}}^i$ and, in the former one, $(\pi_t^i)^{-1} = \pi_{t^{-1}}^{i+A}$.

Let us write X for the set of all such partial permutations. If $\mathcal{M}$ has d many transitions then the size of X is at most $d \cdot N$. Let us also select Y to be a minimal set of generators for the group of partial permutations of the form:

$$\pi' = \pi_1 \cup \pi_2 \cup \{(3N + K, 3N + K)\}$$

[14] i.e. from state 3, if the head of the tape and its right-successor read 00 then write 10 to them, move right and go to state 5.

where $\pi_1 : [1, 2N] \xrightarrow{\cong} [1, 2N]$ and $\pi_2 : [2N + 1, 3N] \xrightarrow{\cong} [2N + 1, 3N]$. Note that $|Y| \leq 3n/2$. Moreover, let us take

$$\pi_0 = \{(2i - 1, 2i - 1) \mid i \in [1, N]\} \cup \{(2N + 1, 2N + 1)\} \cup \{(3N + K + 1, 3N + 1)\}$$

to be a permutation setting up the initial positions of the tokens. We then have that:

$$\mathcal{M} \text{ terminates} \iff \pi_{\mathcal{M}} \in \langle X \cup Y \cup \{\pi_0\}\rangle \tag{D.1}$$

where $\pi_{\mathcal{M}}$ is the partial permutation:

$$\pi_{\mathcal{M}} = \{(2i - 1, 2i - 1) \mid i \in [1, N]\} \cup \{(2N + 1, 2N + 1)\} \cup \{(3N + K + 1, 3N + K)\}$$

To prove (D.1), note first that any accepting run of $\mathcal{M}$, say

$$(q_0, H_0, \alpha_0) \xrightarrow{t_1} (q_1, H_1, \alpha_1) \cdots \xrightarrow{t_k} (q_k, H_k, \alpha_k)$$

where $q_0 = 1$, $H_0 = 1$, $\alpha_0 = 0^N$ and $q_k = K$, yields a permutation $\pi = \pi_0; \pi_{t_1}^{H_0}; \cdots ; \pi_{t_k}^{H_k}$ with the property that $\mathsf{dom}(\pi) = \mathsf{dom}(\pi_0)$ and $\pi(3N + 1) = 3N + K$. We can now select some $\pi' \in \langle Y\rangle$ such that $\pi' \restriction \mathsf{dom}(\pi) = (\pi \restriction [1, 3N]) \cup \{(3N + K, 3N + K)\}$ and, hence, $\pi; \pi'^{-1} = \pi_{\mathcal{M}}$.

Conversely, suppose that $\pi_{\mathcal{M}} \in \langle X \cup Y \cup \{\pi_0\}\rangle$ and in particular let $\pi_{\mathcal{M}} = \pi_0; \pi_1; \cdots ; \pi_k$ be a production (so each π^i is in $X \cup Y \cup \{\pi_0\} \cup X^{-1} \cup Y^{-1} \cup \{\pi_0^{-1}\}$). Note that, because π_0 is the only generator with $3N + K + 1$ in its domain, it must be the leftmost one in the production. Let $k' \leq k$ be the least index such that $\pi_{k'} \notin Y \cup Y^{-1}$ and, for all $j > k'$, $\pi_j \in Y \cup Y^{-1}$, and assume the production is minimal with respect to the value (k', k) (in the lexicographic ordering). We first claim that there is no π_j with $j < k'$ such that $\pi_j \in Y \cup Y^{-1}$. Because if that were the case then $\pi' = \pi_0; \cdots ; \pi_{j-1}$ would satisfy $\mathsf{dom}(\pi') = \mathsf{dom}(\pi_{\mathcal{M}})$ and $\pi'(3N + K + 1) = 3N + K$ so there would be some $\pi'' \in \langle Y\rangle$ such that $\pi_{\mathcal{M}} = \pi_0; \cdots ; \pi_{j-1}; \pi''$, and the latter would lead to a production with size $(j - 1, \cdots)$ which would be smaller than (k', k). Moreover, if $\pi_i = \pi_0$ for some $i > 0$ then we must have $\pi_{i-1} = \pi_0^{-1}$. Because $\pi_0^{-1}; \pi_0 = \mathsf{id}_{\mathsf{rng}(\pi_0)}$ and $|\pi_0^{-1}; \pi_0| = |\pi_{\mathcal{M}}| = N + 2$, we have that $\pi_0^{-1}; \pi_0$ can be safely removed from the production of π, thus contradicting the minimality of the latter. For similar reasons, $\pi_i \neq \pi_0^{-1}$, for all $i \in [1, k]$. Hence, π_0 only occurs at the beginning of the production and π_0^{-1} does not occur at all. Summing up, $\pi = \pi_0; \pi_A; \pi_B$ with $\pi_A \in \langle X\rangle$ and $\pi_B \in \langle Y\rangle$. We can now see that π_A represents a computation of $\mathcal{M}$ from 1 to K. □