Nominal Modal Logics for Fresh-Register Automata (work in progress)

M. H. Bandukara[*]

N. Tzevelekos

Queen Mary University of London
London, UK

1 Introduction

Fresh-Register Automata are an extension of the Register Automata model by Kaminski and Francez [7]. Each automaton is equipped with a finite set of registers where it can store data values. Register automata can verify whether or not a given input data value (referred to as *names*) is stored in one of its registers, and store it in a register overwriting its current value. This design allows register automata to capture languages over infinite alphabets. Fresh-register automata expand on this by having the option of allowing the automaton to accept a name just if it is *globally-fresh*, that is, it has not appeared so far in the input. Fresh-register automata can be used to capture computational models that use names and name-generation, for example (finitary) π-calculus processes [16, 1], a paradigmatic process language for concurrent interactions involving name passing [11, 15]. To date, there are results for bisimulation equivalence of fresh-register automata [16, 13, 1]. In this work we are interested in furthering formal verification for fresh-register automata by designing a Hennessy-Milner logic (HML) for them.

Hennessy-Milner logic (HML) is a modal behavioural logic that captures branching trace-based properties of labelled transition systems. It was initially introduced in 1980 by Hennessy and Milner [5], for the purpose of being an alternative exposition of observational equivalence [2]. Observational equivalence has a focus on testing whether two systems can simulate each other in a step-by-step manner, whereas HML logic focuses on the expressiveness of a single system. Under certain assumptions, two processes are equivalent just if they satisfy the same HML formulas [6]. The extension of HML with recursion, called *modal μ-calculus*, was introduced by Kozen in [10].

In this presentation of work in progress, we examine a nominal extension thereof, where modalities are allowed to include names and which we use to capture properties of Fresh-Register Automata and π-calculus processes. HML for the π-calculus was first examined in [12]. For properties involving paths of unbounded length, it is natural to examine recursive extensions thereof such as the modal μ-calculus and variants thereof [2]. The logics we are examining are based on the π-μ-calculus of Dam [3] and the recent work on nominal modal calculi of Klin and collaborators [8, 9, 4]. Our points of focus are expressiveness of the logics with respect to fresh-register automata and previous nominal calculi, but also decidability, complexity and implementations of model checking.

2 Nominal modal μ-calculus

In this section, we define details of our nominal modal μ calculus according to the specification of register automata. To do this, we must first define a set of names and tags as inputs for the register automata transitions. Let us fix a countably infinite set $\mathbb{A}$ of *names* (or *atoms*), ranged over by a and variants, and a finite set Σ of *tags* ranged over by t and variants.

[*]supported by EPSRC DTP EP/R513106/1.

Submitted to:
FICS 2024

Definition 1 (Nominal modal μ-calculus)**.** *Given countably infinite set of variables Var (x,y etc.) and recursion variables VAR (X,Y etc.), we define:*

$$\begin{aligned} \mathit{Formulae} \ni \ \phi &::= \ u = u \mid \phi \vee \phi \mid \neg\phi \mid \bigvee_{x\in\mathbb{A}} \phi \mid \langle\ell\rangle\phi \mid (\mu X(\vec{x}).\phi)(\vec{u}) \mid X(\vec{u}) \\ \mathit{Values} \ni \ u &::= \ x \mid a \\ \mathit{Labels} \ni \ \ell &::= \ \tau \mid (t,u) \end{aligned}$$

The design follows previous works [2, 3], incorporating recursion variables and additionally being equipped with infinite choice (i.e., $\bigvee_{x\in\mathbb{A}}$) to express the ability for register automata to accept a data value not currently stored in a register. Recursion variables have given *arities*, given by a map $ar : VAR \to \mathbb{N}$, which is respected by constructs $X(\vec{u})$ and $(\mu X(\vec{x}).\phi)(\vec{u})$ (i.e. $|\vec{u}| = |\vec{x}| = ar(X)$). Variables x and recursion variables X can be free and bound, with their binders being the constructs $\bigvee_{x\in\mathbb{A}}\phi$ and $\mu X(\vec{x}).\phi$, and $\mu X(\vec{x}).\phi$ respectively. We shall usually write $\bigvee_x \phi$ as abbreviation for $\bigvee_{x\in\mathbb{A}}\phi$. As usually, we impose that each recursion variable X appears within an even number of negation operations from its binder to ensure monotonicity. The semantics of nominal modal μ-calculus formulas is given within nominal LTSs.

Definition 2 (Nominal set and nominal LTS [14])**.** *A nominal set is a set X along with an action (denoted $\cdot$) of the group of finite permutations of $\mathbb{A}$, such that all elements of X are finitely supported. A set of names $S \subseteq \mathbb{A}$* supports *an element $x \in X$ if for all $\pi \in Perm(\mathbb{A})$:*

$$(\forall a \in S.\, \pi\cdot a = a) \implies \pi\cdot x = x$$

We say that x is finitely supported when there is a finite $S \subseteq \mathbb{S}$ supporting x. We write $\mathrm{supp}(x)$ *for the least set S supporting x. A relation R over a nominal set X is called* equivariant *when, for all $x \in X$ and permutations π, $x \in R$ iff $\pi\cdot x \in R$.*
A nominal Labelled-Transition System (nominal LTS) *is a tuple $\mathscr{L} = \langle \mathscr{S}, L, \to\rangle$, where $\mathscr{S}$ is a nominal set of states, L is a nominal set of actions and $\to \subseteq \mathscr{S}\times L\times\mathscr{S}$ is an equivariant transition relation.*

We next fix a nominal LTS $\mathscr{L}$ over a set of states $\mathscr{S}$ and let $\mathscr{U}$ be the powerset of $\mathscr{S}$. We let a *$\mathscr{U}$-variable assignment* be a finite map $\xi : VAR \rightharpoonup \bigcup_n(\mathbb{A}^n \to \mathscr{U})$ such that, for each $X \in \mathrm{dom}(\xi)$, $\xi(X) \in (\mathbb{A}^{|ar(X)|} \to \mathscr{U})$. We next define the semantics of formulas.

Definition 3. *Let a and b denote distinct names. Given a variable assignment ξ, the semantics of a formula ϕ with respect to ξ, written $[\![\phi]\!]_\xi$ is given inductively by:*

$$\begin{aligned} [\![a = b]\!]_\xi &= \emptyset \\ [\![a = a]\!]_\xi &= \mathscr{S} \\ [\![\phi_1 \vee \phi_2]\!]_\xi &= [\![\phi_1]\!]_\xi \cup [\![\phi_2]\!]_\xi \\ [\![\neg\phi]\!]_\xi &= \mathscr{S} \setminus [\![\phi]\!]_\xi \\ [\![\textstyle\bigvee_{x\in\mathbb{A}}\phi]\!]_\xi &= \textstyle\bigcup_{a\in\mathbb{A}} [\![\phi\{a/x\}]\!]_\xi \\ [\![\langle\ell\rangle\phi]\!]_\xi &= \{s \in \mathscr{S} \mid \exists s \xrightarrow{\ell} s'.\, s' \in [\![\phi]\!]_\xi\} \\ [\![(\mu X(\vec{x}).\phi)(\vec{a})]\!]_\xi &= (\mathrm{lfp}(\lambda f.\lambda\vec{b}.[\![\phi\{\vec{b}/\vec{x}\}]\!]_{\xi[X\mapsto f]}))(\vec{a}) \\ [\![X(\vec{a})]\!]_\xi &= \xi(X)(\vec{a}) \end{aligned}$$

Note that the semantics is only defined on closed formulas.

The imposition of each recursion variable X having a negative number of negations from its binder allows us to show that the fixed points in the semantics are well defined. In order to prove that model checking is decidable, we show that a finite representation of the semantics function is possible. Given a finite set $S \subseteq \mathbb{A}$, we define the restricted notion of translation with support within S, written $[\![\phi]\!]^S_\xi$, using the same rules as given prior, apart from the cases of negation, infinite choice and recursion. It is necessary to select an appropriately large set S to accommodate all name choices inside a given formula (e.g. because of an infinite choice construct) we are able to show that $[\![\phi]\!]^S_\xi$ is a representation of $[\![\phi]\!]_\xi$. Below, we say that a nominal set X is *orbit-finite* if there is a finite subset $\{x_1,\dots,x_n\} \subseteq X$ such that:

$$X = \bigcup_i \{\pi \cdot x_i \mid \text{permutation } \pi\}$$

We say that an LTS is orbit-finite if its S and L are orbit-finite.

Lemma 1. *Model checking nominal modal μ-calculus is decidable over any orbit-finite LTS.*

2.1 History-dependent extension (global freshness)

Currently, our logic accounts for names not currently stored in registers, however, it is vital to extend it to account for global freshness. Recall that global-freshness is when a data value can be accepted if it has not yet appeared as input for the current run of the automaton. The formulae of *History-Dependent nominal Modal μ-calculus (HD modal μ-calculus)* are given by extending the grammar of our previous definition, with the following addition:

$$\text{Formulae} \ni \ \phi ::= \ \cdots \mid \#u$$

The semantics of $\#a$ will be that a is fresh in the current state, that is, it is a new name that has been written to the register upon entering that state. For that, we need to extend our notion of nominal LTS to account for histories.

Definition 4. *A* History-Dependent nominal Labelled-Transition System (HD-LTS) *is a tuple $\mathscr{L} = \langle \mathscr{S}, L, \rightarrow\rangle$, where $\mathscr{S}$ is a nominal set of states, L is a nominal set of actions and $\rightarrow \subseteq \mathscr{S} \times \mathscr{P}_{\mathrm{fin}}(\mathbb{A}) \times L \times \mathscr{S} \times \mathscr{P}_{\mathrm{fin}}(\mathbb{A})$ is an equivariant transition relation such that for all transitions $(s,H) \xrightarrow{\ell} (s',H')$ (i.e. whenever $(s,H,\ell,s',H') \in \rightarrow$):*

- $\mathrm{supp}(s) \subseteq H$*,* $\mathrm{supp}(s') \subseteq \mathrm{supp}(s) \cup \mathrm{supp}(\ell)$ *and* $H' = H \cup \mathrm{supp}(\ell)$*.*

HD modal μ-calculus is an adaptation to our aforementioned modal μ-calculus geared towards using a HD-LTS and including semantics for global-freshness. We now define the semantics of HD modal μ-calculus.

Definition 5. *For each HD-LTS $\mathscr{L} = \langle \mathscr{S}, L, \rightarrow\rangle$ with $L = \{\tau\} \cup (\Sigma \times \mathbb{A})$, let us set*

$$\mathscr{U} = \{U \subseteq \mathscr{S} \times \mathscr{P}_{\mathrm{fin}}(\mathbb{A}) \mid \forall (s,H) \in U.\, \mathrm{supp}(s) \subseteq H\}.$$

Let a and b denote distinct names. Given a $\mathscr{U}$-variable assignment ξ, the semantics of a formula ϕ with *respect to ξ, written $[\![\phi]\!]_\xi$ is given inductively by:*

$$\begin{aligned}
[\![\#a]\!]_\xi &= \{(s,H) \mid a \notin H\} \\
[\![a = b]\!]_\xi &= \emptyset \\
[\![a = a]\!]_\xi &= \mathscr{U} \\
[\![\phi_1 \vee \phi_2]\!]_\xi &= [\![\phi_1]\!]_\xi \cup [\![\phi_2]\!]_\xi \\
[\![\neg\phi]\!]_\xi &= \mathscr{U} \setminus [\![\phi]\!]_\xi \\
[\![\bigvee_{x\in\mathbb{A}} \phi]\!]_\xi &= \bigcup_{a\in\mathbb{A}} [\![\phi\{a/x\}]\!]_\xi \\
[\![\langle\ell\rangle\phi]\!]_\xi &= \{(s,H) \mid \exists (s,H) \xrightarrow{\ell} (s',H').\,(s',H') \in [\![\phi]\!]_\xi\} \\
[\![(\mu X(\vec{x}).\phi)(\vec{a})]\!]_\xi &= (\mathrm{lfp}(\lambda f.\lambda\vec{b}.[\![\phi\{\vec{b}/\vec{x}\}]\!]_{\xi[X\mapsto f]}))(\vec{a}) \\
[\![X(\vec{a})]\!]_\xi &= \xi(X)(\vec{a})
\end{aligned}$$

Note that as before, the semantics is only defined on closed formulas.

With the HD modal μ-calculus, it is possible to examine and express properties of fresh-register automata, and by extension, finitary π-calculus processes. The use of nominal sets allows concrete representation of fresh names as is necessary by the π-calculus, and this logic allows considering equivalence of capabilities of π-calculus processes and fresh-register automata.

References

[1] M.H. Bandukara & N. Tzevelekos (2023): *On-the-fly bisimulation equivalence checking for fresh-register automata*. Journal of Systems Architecture 145, p. 103010, doi:https://doi.org/10.1016/j.sysarc.2023.103010. Available at `https://www.sciencedirect.com/science/article/pii/S1383762123001893`.

[2] Julian C. Bradfield & Colin Stirling (2007): *Modal mu-calculi*. In Patrick Blackburn, J. F. A. K. van Benthem & Frank Wolter, editors: Handbook of Modal Logic, Studies in logic and practical reasoning 3, North-Holland, pp. 721–756.

[3] Mads Dam (2003): *Proof Systems for π-Calculus Logics*. In Ruy J. G. B. de Queiroz, editor: Logic for Concurrency and Synchronisation, Trends in Logic 18, Kluwer, pp. 145–212.

[4] Clovis Eberhart & Bartek Klin (2019): *History-Dependent Nominal μ-Calculus*. In: 34th Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2019, Vancouver, BC, Canada, June 24-27, 2019, IEEE, pp. 1–13.

[5] Matthew Hennessy & Robin Milner (1980): *On observing nondeterminism and concurrency*. In Jaco de Bakker & Jan van Leeuwen, editors: Automata, Languages and Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 299–309.

[6] Matthew Hennessy & Robin Milner (1985): *Algebraic Laws for Nondeterminism and Concurrency*. J. ACM 32(1), p. 137–161, doi:10.1145/2455.2460. Available at `https://doi.org/10.1145/2455.2460`.

[7] Michael Kaminski & Nissim Francez (1994): *Finite-memory automata*. Theoretical Computer Science 134(2), pp. 329–363.

[8] Bartek Klin & Mateusz Lelyk (2017): *Modal mu-Calculus with Atoms*. In Valentin Goranko & Mads Dam, editors: 26th EACSL Annual Conference on Computer Science Logic, CSL 2017, August 20-24, 2017, Stockholm, Sweden, LIPIcs 82, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 30:1–30:21.

[9] Bartek Klin & Mateusz Lelyk (2019): *Scalar and Vectorial mu-calculus with Atoms*. Log. Methods Comput. Sci. 15(4).

[10] Dexter Kozen (1983): *Results on the propositional μ-calculus*. Theoretical Computer Science 27(3), pp. 333–354, doi:https://doi.org/10.1016/0304-3975(82)90125-6. Available at `https://www.sciencedirect.`

`com/science/article/pii/0304397582901256`. Special Issue Ninth International Colloquium on Automata, Languages and Programming (ICALP) Aarhus, Summer 1982.

[11] Robin Milner, Joachim Parrow & David Walker (1992): *A Calculus of Mobile Processes, I and II*. Inf. Comput. 100(1).

[12] Robin Milner, Joachim Parrow & David Walker (1993): *Modal Logics for Mobile Processes*. Theor. Comput. Sci. 114(1), pp. 149–171.

[13] Andrzej S. Murawski, Steven J. Ramsay & Nikos Tzevelekos (2015): *Bisimilarity in Fresh-Register Automata*. In: LICS Proceedings, pp. 156–167.

[14] Andrew M. Pitts (2013): *Nominal Sets: Names and Symmetry in Computer Science*. Cambridge Tracts in Theoretical Computer Science, Cambridge University Press, doi:10.1017/CBO9781139084673.

[15] Davide Sangiorgi & David Walker (2001): *The Pi-Calculus - a theory of mobile processes*. Cambridge University Press.

[16] Nikos Tzevelekos (2011): *Fresh-Register Automata*. In: POPL Proceedings, ACM.