Parity Automata for Quantitative Linear Time Logics

Corina Cîrstea[1], Shunsuke Shimizu[2], and Ichiro Hasuo[2]

1 **University of Southampton, UK**
`cc2@ecs.soton.ac.uk`

2 **National Institute of Informatics, Tokyo, Japan**
`shunsuke,hasuo@nii.ac.jp`

Abstract

We initiate a study of automata-based model checking for previously proposed quantitative linear time logics interpreted over coalgebras. Our results include: (i) an automata-theoretic characterisation of the semantics of these logics, based on a notion of extent of a quantitative parity automaton, (ii) a study of the expressive power of Büchi variants of such automata, with implications on the expressiveness of fragments of the logics considered, and (iii) a naïve algorithm for computing extents, under additional assumptions on the domain of truth values.

1998 ACM Subject Classification F.1.1 Models of Computation, F.4.1 Modal Logic

Keywords and phrases coalgebra, quantitative logic, linear time logic, parity automaton

Digital Object Identifier 10.4230/LIPIcs.CALCO.2017.7

1 Introduction

Linear time logics such as LTL or the linear time μ-calculus (see e.g. [9]), originally interpreted over non-deterministic transition systems, have been adapted and used successfully in model checking various types of non-deterministic and probabilistic systems [3]. These logics share the same notion of linear time behaviour, but depending on the branching present in the models, have either a qualitative or a quantitative interpretation (with $\{0, 1\}$, resp. the unit interval as domains of truth values). Despite commonalities between the logics and their automata-based verification techniques, a uniform account of the connection between (quantitative) linear-time logics and (quantitative) automata over infinite structures is still missing. This would allow existing verification techniques to be transferred to new models, including weighted ones (for which linear time logics have already been studied [15]).

This paper initiates a general study of the connection between quantitative linear time logics and quantitative automata on infinite structures, grounded in coalgebraic modelling, by building on recent work on *maximal traces* [6], quantitative linear time logics for systems with branching [4, 5], and coalgebraic trace semantics for Büchi and parity automata [17]. (Here, a maximal trace is either a finite, completed trace or an infinite trace.) The work in [6, 4, 5] models systems with branching as coalgebras of type $\mathsf{T} \circ F$, with $F : \mathsf{Set} \to \mathsf{Set}$ an endofunctor used to specify linear behaviour (the structure of individual transitions) and $\mathsf{T} : \mathsf{Set} \to \mathsf{Set}$ a monad used to specify branching structure (which typically associates quantities to individual transitions). Examples include non-deterministic/probabilistic/weighted labelled transition systems, with or without explicit termination, but also models with a more general "linear" behaviour, including tree-like behaviour; for example, taking $F : \mathsf{Set} \to \mathsf{Set}$ to be $A \times \mathsf{Id} \times \mathsf{Id}$ with $\mathsf{Id} : \mathsf{Set} \to \mathsf{Set}$ the identity functor captures linear behaviours given by infinite labelled binary trees. In this setting, a canonical definition of linear time behaviour of states in a coalgebra with branching associates, to each state and each possible maximal trace (element of the final F-coalgebra), a quantity measuring the extent of the given state exhibiting that trace, accumulated across all branches [6]. This recovers known concepts of infinite trace semantics, including the possibility (for the finite powerset monad), likelihood (for the sub-probability distribution monad) or minimal cost (for a weighted monad with weights modelling costs) of exhibiting a particular trace.

Coalgebraic linear time logics, interpreted over the same types of models, were studied in [4, 5], with formulas specifying properties of linear behaviour, and with their semantics measuring the extent with which such properties hold in states of coalgebras with branching. The monad T uniformly determines both the domain of truth values for the logics and the choice of propositional operators (e.g. finite disjunctions for non-deterministic systems, sub-convex combinations for probabilistic systems and linear combinations for weighted systems). The modal operators employed arise from the functor F, but their interpretation is based both on a choice of associated predicate liftings for F and on a canonical predicate lifting for T, with the latter being used to accumulate the quantities associated to different branches. As with the notion of linear time behaviour, exactly how this accumulation works depends on the type of branching. For non-deterministic systems, one recovers the *aconjunctive* linear-time μ-calculus. For probabilistic systems, the resulting logics differ from probabilistic variants of LTL or the linear time μ-calculus in their absence of boolean operators. This difference is beneficial, as it results in an expensive automata determinisation step required for model checking such probabilistic variants (see [3, Section 10.3]) being avoided with our logics, without a real loss in expressiveness (see Remark 6). For weighted systems, the logics are similar to previously proposed ones [15] in their absence of conjunction operators. Our choice of propositional operators is further supported by results in [5] on the equivalence of the original, step-wise semantics of the logics with an alternative path-based semantics akin to that of LTL, and by the close connection to quantitative automata described in this paper.

We now turn to the contributions of this paper. A definition of the extent with which *two* states in coalgebras with branching exhibit *similar* linear behaviour was given in [6]. Here we take this further by providing alternative characterisations of notions of *maximal* and *finite trace similarity* between two branching coalgebras. These instantiate to the existence/likelihood/minimal joint cost of a *common* trace, in the case of non-deterministic/probabilistic/weighted branching. When one of the coalgebras models the system of interest and the other captures (un)desirable behaviours, these notions provide the right concepts for automata-theoretic model checking. Our alternative characterisations involve a product construction, and a novel notion of *extent* of a coalgebra with branching.

We then extend these ideas to provide an automata-theoretic characterisation of the semantics of the logics in [4, 5]. For this, we use a generalisation of standard parity automata over words/trees, inspired by recent work on coalgebraic trace semantics for Büchi and parity automata [17]. Our automata are parameterised by (i) a *partial semiring* monad, specifying a type of branching, and (ii) a polynomial endofunctor F, specifying a type of linear behaviour. Key to our approach is the notion of *extent* of a parity automaton, which provides a *quantitative* notion of acceptance, not of a single maximal trace, but across *all* maximal traces that conform to the automaton. This instantiates to the existence/likelihood/minimal cost of an accepting trace (maximal trace satisfying the parity condition), for non-deterministic/probabilistic/weighted branching (with weights modelling costs), respectively (see Example 17). A key advantage of the *extent-based* characterisation of the semantics is a *localised view* of the satisfaction relation of the logics – fixpoints are computed locally on the reachable part of the product between model and formula automata, rather than globally as predicates on the model state space, as stipulated in the original semantics. Our *generic* translation from formulas to automata resembles known translations for non-deterministic systems (see e.g. [19]), while additionally exploiting the choices made in the semantics of our logics to simplify the construction. A translation from automata to formulas is also sketched.

For non-deterministic branching, Büchi automata over words are as expressive as parity ones [11], while over trees this fails to hold [16]. Here we generalise this result to a quantitative setting, under the additional assumption that the branching semiring is *total*. Given our translations from logics to automata and back, this also yields an important result about our logics, namely that for *word-like* linear behaviour (i.e. F isomorphic to a coproduct of constant and identity functors), the alternation degree 1 fragment of the logics is fully expressive. We stress that our translation from parity to Büchi automata preserves the *quantitative* language. To our knowledge, the only related result in a quantitative setting is a translation from probabilistic Rabin to probabilistic Büchi automata [2], which only preserves the *qualitative* language. Unlike the standard translation from parity to Büchi via Rabin automata [11], which uses the idempotence of disjunction, our translation does *not* require idempotence of the semiring addition. For *partial* semirings which arise as sub-semirings of total semirings satisfying our assumptions (as is the case for probabilistic branching), a similar translation is obtained by moving to the larger semiring; this time, the resulting Büchi automaton has branching as specified by the larger semiring, but can nonetheless be used in the same way, given the preservation of the quantitative language (see Remark 33).

Our final contribution is a naïve algorithm for computing extents of parity automata. This requires an additional assumption on the underlying semiring, satisfied by both non-deterministic branching and a bounded variant of weighted branching, to ensure that the computation of individual fixpoints terminates. To summarise, our contributions are:

1. automata-theoretic characterisations of *finite* and *maximal trace similarity* between states of coalgebras with branching (Theorem 14),
2. a notion of *extent* of a parity automaton, parameterised by a choice of branching monad and linear time behaviour (Definition 16),
3. translations from linear time formulas to automata and back, providing an automata-theoretic characterisation of the quantitative semantics of the logics in [4, 5] (Theorem 24),
4. a translation from quantitative parity to quantitative Büchi automata over words (Theorem 32), rendering the alternation degree 1 fragment of the logics fully expressive,
5. an algorithm for computing extents, similar in complexity to known algorithms for emptiness checking in the non-deterministic case [14].

We assume familiarity with the coalgebraic approach to modelling systems. Section 2 recalls the results of [6, 4, 5], while each subsequent section contains one of the above contributions.

2 Preliminaries

2.1 From Partial Commutative Semirings to Commutative Monads

▶ **Definition 1.** A *partial commutative semiring* is a tuple $S := (S, +, 0, \bullet, 1)$ with $(S, +, 0)$ a partial commutative monoid and $(S, \bullet, 1)$ a commutative monoid, with $\bullet$ distributing over $+$; that is, for all $s, t, u \in S$, $s \bullet 0 = 0$, and whenever $t + u$ is defined, then so is $s \bullet t + s \bullet u$ and moreover $s \bullet (t + u) = s \bullet t + s \bullet u$.

The addition operation of any partial commutative semiring induces a pre-order relation $\sqsubseteq$ on S, given by $x \sqsubseteq y$ iff there exists $z \in S$ with $x + z = y$, and having 0 as its least element.

▶ **Example 2.** Here we consider the *boolean semiring* $(\{0,1\},\vee,0,\wedge,1)$, the (partial) *probabilistic semiring* $([0,1],+,0,*,1)$, the *tropical semiring* $\mathbb{N}=(\mathbb{N}^\infty,\min,\infty,+,0)$ and its bounded variants $S_B=([0,B]^\infty,\min,\infty,+_B,0)$ with $B\in\mathbb{N}$, where $m{+_B}n=\begin{cases}m+n, & \text{if } m+n\le B\\ \infty, & \text{otherwise}\end{cases}$.
The associated pre-orders are $\le$ on $\{0,1\}$ and $[0,1]$, and $\ge$ on $\mathbb{N}^\infty$ and $[0,B]^\infty$.

We further assume that $\sqsubseteq$ has a top element and is an ω-chain complete as well as ω^{op}-chain complete partial order. This holds in all our examples.

A partial commutative semiring S induces a *semiring monad* (T_S,η,μ) with

$$\mathsf{T}_S(X)=\{\,\varphi:X\to S\mid \mathsf{supp}(\varphi)\text{ is finite},\ \sum_{x\in\mathsf{supp}(\varphi)}\varphi(x)\text{ is defined}\,\}$$

$$\eta_X(x)(y)=\begin{cases}1 & \text{if } y=x\\ 0 & \text{otherwise}\end{cases},\qquad \mu_X(\Phi)(x)=\sum_{\varphi\in\mathsf{supp}(\Phi)}\Phi(\varphi)\bullet\varphi(x)$$

where $\mathsf{supp}(\varphi)=\{x\in X\mid\varphi(x)\neq 0\}$ is the *support* of φ. Moreover, the monad T_S above is *strong* and *commutative*, with *strength* $\mathsf{st}_{X,Y}:X\times\mathsf{T}_SY\to\mathsf{T}_S(X\times Y)$ and *double strength* $\mathsf{dst}_{X,Y}:\mathsf{T}_SX\times\mathsf{T}_SY\to\mathsf{T}_S(X\times Y)$ given by

$$\mathsf{st}_{X,Y}(x,\psi)(z,y)=\begin{cases}\psi(y) & \text{if } z=x\\ 0 & \text{otherwise}\end{cases},\qquad \mathsf{dst}_{X,Y}(\varphi,\psi)(z,y)=\varphi(z)\bullet\psi(y)$$

We note that $\mathsf{T}_S1=S$, with 1 a final object in Set, and therefore S carries a T_S-algebra structure given by $\mu_1:\mathsf{T}_S^2 1\to\mathsf{T}_S1$. The relationship between monads and partial semirings was studied in [8, 6]. We use semiring monads to model branching, with the semirings in Example 2 modelling non-deterministic, probabilistic and weighted branching. In the latter case, we think of the weights as costs associated to individual system steps.

2.2 Finite and Maximal Traces

A coalgebraic approach to defining maximal traces of coalgebras of type $\mathsf{T}_S\circ F$, with F a *polynomial*[1] endofunctor and $\mathsf{T}_S:\mathsf{Set}\to\mathsf{Set}$ a semiring monad, is described in [6]. The monad T_S is used to specify branching structure, whereas the functor F is used to specify linear behaviour, with the elements of the final F-coalgebra defining individual traces. Since any polynomial endofunctor on Set can be written as a coproduct of finite products of identity functors, we readily assume this shape: $FX=\coprod_{\lambda\in\Lambda}X^{\mathsf{ar}(\lambda)}$ with Λ a set of operators with finite arities. The elements of the final F-coalgebra (initial F-algebra) are potentially infinite (resp. finite) trees with nodes labelled by some λ and having as many outgoing edges as $\mathsf{ar}(\lambda)$. The definition of maximal traces resembles the alternative partition-refinement definition of bisimilarity, but differs from it in two key ways: (i) what is defined is a *trace relation* between states of a $\mathsf{T}_S\circ F$-coalgebra and elements of the final F-coalgebra, and (ii) trace relations are *S-valued relations* that measure, for each state in a coalgebra with branching and each linear behaviour, the *extent* (e.g. ability, likelihood or minimal cost) of that state exhibiting the given linear behaviour. Concretely, for a $\mathsf{T}_S\circ F$-coalgebra $\gamma:C\to\mathsf{T}_SFC$, its maximal traces are given by the greatest fixpoint of the following operator on S-valued

[1] An endofunctor $F:\mathsf{Set}\to\mathsf{Set}$ is *polynomial* if it is constructed from constant and identity functors using finite products and arbitrary coproducts.

relations between C and the carrier of the final F-coalgebra (Z, ζ):

$$\mathsf{Rel}_{C,Z} \xrightarrow{\mathsf{Rel}(F)} \mathsf{Rel}_{FC,FZ} \xrightarrow{L_S} \mathsf{Rel}_{\mathsf{T}_S FC,FZ} \xrightarrow{(\gamma\times\zeta)^*} \mathsf{Rel}_{C,Z} \tag{1}$$

Here, $\mathsf{Rel}_{X,Y}$ is the category of S-valued relations on $X \times Y$, and $\mathsf{Rel}(F) : \mathsf{Rel}_{X,Y} \to \mathsf{Rel}_{FX,FY}$ "lifts" S-valued relations on $X \times Y$ to S-valued relations on $FX \times FY$, with the help of the semiring multiplication: for $R : X \times Y \to S$, $\mathsf{Rel}(F)(R)$ maps $(\iota_\lambda(x_1, \ldots, x_{\mathsf{ar}(\lambda)}), \iota_{\lambda'}(y_1, \ldots, y_{\mathsf{ar}(\lambda')}))$ to 0 if $\lambda \neq \lambda'$, and $(\iota_\lambda(x_1, \ldots, x_{\mathsf{ar}(\lambda)}), \iota_\lambda(y_1, \ldots, y_{\mathsf{ar}(\lambda)}))$ to $R(x_1, y_1) \bullet \ldots \bullet R(x_{\mathsf{ar}(\lambda)}, y_{\mathsf{ar}(\lambda)})$. Also, $L_S : \mathsf{Rel}_{X,Y} \to \mathsf{Rel}_{\mathsf{T}_S X,Y}$, called *extension lifting*, takes a relation $R : X \times Y \to S$ to the relation $\mu_1 \circ \mathsf{T}_S R \circ \mathsf{st}'_{X,Y} : \mathsf{T}_S X \times Y \to S$, with $\mathsf{st}'_{X,Y} : \mathsf{T}_S X \times Y \to \mathsf{T}_S(X \times Y)$ the *swapped strength map* of T_S. This choice for L_S is canonical, in the sense that $L_S(R)(\_, y)$ is the unique extension of $R(\_, y)$ to a T_S-algebra homomorphism, for $y \in Y$ (see [6] for details). Concretely, $L_S(R)(\sum_i c_i x_i, y) = \mu_1(\sum_i c_i R(x_i, y))$ for $x_i \in X$ and $y \in Y$. The effect of using L_S above is that the quantity ultimately associated to each pair $(c, z) \in C \times Z$ is accumulated across all branches from c, in a step-wise fashion.

A similar treatment of *finite* traces is obtained by replacing the *final* F-coalgebra (Z, ζ) with the *initial* F-algebra (I, ι), with $\iota : FI \xrightarrow{\simeq} I$, and taking the *least* fixpoint of the following operator on S-valued relations:

$$\mathsf{Rel}_{C,I} \xrightarrow{\mathsf{Rel}(F)} \mathsf{Rel}_{FC,FI} \xrightarrow{L_S} \mathsf{Rel}_{\mathsf{T}_S FC,FI} \xrightarrow{(\gamma\times\iota^{-1})^*} \mathsf{Rel}_{C,I}$$

▶ **Example 3.** When $S = (\{0,1\}, \vee, 0, \wedge, 1)$ (and thus T_S is isomorphic to the finite powerset monad), the (greatest, resp. least) fixpoints of the previous operators relate a state in a non-deterministic coalgebra with a (maximal, resp. finite) trace iff that state can exhibit the given trace. When $S = ([0,1], +, 0, *, 1)$ or $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$, the (greatest, resp. least) fixpoints give, for each state and each (maximal, resp. finite) trace, the likelihood, resp. minimal cost of that state exhibiting the given trace. The precise shape of a trace is determined by the choice of F; taking $F = 1 + A \times \mathsf{Id}$ captures words over A, whereas $F = 1 + A \times \mathsf{Id} \times \mathsf{Id}$ captures binary trees with non-leaf nodes labelled by A.

2.3 Quantitative Linear Time Logics for Coalgebras

We now recall (a variant of) the logics studied in [4, 5]. They are interpreted over $\mathsf{T}_S \circ F$-coalgebras, with T_S and F as before, and have syntax given by

$$\mu\mathcal{L}^{\mathcal{V}}_{\Lambda} \ni \varphi ::= x \mid [\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)}) \mid \textstyle\sum_{i\in I} c_i \bullet \varphi_i \mid \mu x.\varphi \mid \nu x.\varphi$$

with $x \in \mathcal{V}$, $\lambda \in \Lambda$ and $c_i \in S$ such that $\sum_{i\in I} c_i$ is defined. Here, $\mathcal{V}$ is a set of variables and I is a finite set. Writing $\iota_\lambda : X^{\mathsf{ar}(\lambda)} \to FX$ with $\lambda \in \Lambda$ for the coproduct injections, we define, for each modal operator $\lambda \in \Lambda$, an *S-valued predicate lifting* $[\![\lambda]\!] : S^{-} \times \ldots \times S^{-} \Rightarrow S^{F-}$ by:

$$[\![\lambda]\!](p_1, \ldots, p_{\mathsf{ar}(\lambda)})(\iota_{\lambda'}(x_1, \ldots, x_{\mathsf{ar}(\lambda')})) = \begin{cases} p_1(x_1) \bullet \ldots \bullet p_{\mathsf{ar}(\lambda)}(x_{\mathsf{ar}(\lambda)}), & \text{if } \lambda = \lambda' \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

▶ **Definition 4.** For a $\mathsf{T}_S \circ F$-coalgebra (C, γ) and a valuation $V : \mathcal{V} \to S^C$, the denotation $[\![\varphi]\!]^V_\gamma \in S^C$ of a formula $\varphi \in \mu\mathcal{L}^{\mathcal{V}}_{\Lambda}$ is defined inductively on the structure of φ by

- $[\![x]\!]^V_\gamma = V(x)$,
- $[\![\sum_{i\in I} c_i \bullet \varphi_i]\!]^V_\gamma = \mu_1(\sum_{i\in I} c_i\, [\![\varphi_i]\!]^V_\gamma)$,
- $[\![[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})]\!]^V_\gamma = \gamma^*(\mathsf{ext}_{FC}([\![\lambda]\!]_C([\![\varphi_1]\!]^V_\gamma, \ldots, [\![\varphi_{\mathsf{ar}(\lambda)}]\!]^V_\gamma)))$, where the *extension predicate lifting* $\mathsf{ext} : S^{-} \Rightarrow S^{\mathsf{T}_S -}$ takes an S-valued predicate $p : C \to S$ to the S-valued predicate $\mu_1 \circ \mathsf{T}_S p : \mathsf{T}_S C \to S$, while $\gamma^* : S^{\mathsf{T}_S FC} \to S^C$ is pre-composition with γ.

- $[\![\mu x.\varphi]\!]_\gamma^{V\setminus\{x\}}$ ($[\![\nu x.\varphi]\!]_\gamma^{V\setminus\{x\}}$) is the least (resp. greatest) fixpoint of the operator on S^C taking $p : C \to S$ to $[\![\varphi]\!]_\gamma^{V[p/x]}$, where the valuation $V[p/x] : \mathcal{V} \to S^C$ takes x to p and $y \in \mathcal{V} \setminus \{x\}$ to $V(y)$.

(In the second clause, both the formal sum notation and the action of μ_1 have been extended pointwisely to functions on C.) We write $\mu\mathcal{L}_\Lambda$ for the set of *closed* formulas ($\mathcal{V} = \emptyset$).

For the operator in the last clause of Definition 4 to be order-preserving, monotonicity of both ext and $[\![\lambda]\!]$, with $\lambda \in \Lambda$, is required, and proved in [5]. The existence of lfp*s*, respectively gfp*s* then follows by [10, Theorem 8.22], which assumes an order-preserving operator on a cpo. The use of extension lifting in the third clause of Definition 4 results in $[\![[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})]\!]_\gamma^V(c)$ accumulating the values $[\![\lambda]\!]_C([\![\varphi_1]\!]_\gamma^V, \ldots, [\![\varphi_{\mathsf{ar}(\lambda)}]\!]_\gamma^V)(f_i)$, with $\gamma(c) = \sum_i c_i f_i$, by taking into account the weights c_i:

$$[\![[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})]\!]_\gamma^V(c) = \mu_1(\sum_i c_i [\![\lambda]\!]_C([\![\varphi_1]\!]_\gamma^V, \ldots, [\![\varphi_{\mathsf{ar}(\lambda)}]\!]_\gamma^V)(f_i))$$

The presence of weighted sums in the logics is supported by results in [5] showing that the inclusion of such sums preserves the equivalence of the above step-wise semantics with an alternative, *path-based* semantics, akin to that of LTL. Finite conjunctions are missing from the logics, and our step-wise semantics prevents their inclusion. It is worth noting, however, that fixpoint logics for weighted systems are similar in their absence of conjunctions [15]. For *total* semirings S, finite disjunctions are present as finite weighted sums with the weights equal to 1. Their interpretation is as expected: the formula $\sum_i 1 \bullet \varphi_i$ (written more simply $\sum_i \varphi_i$) measures the extent of conforming to one of the φ_is.

▶ **Example 5.** Taking $F = 1 + A \times \mathsf{Id} \simeq 1 + \coprod_{a \in A} \mathsf{Id}$ yields a logic with a nullary modality $*$ (for termination), and unary modalities $[a]$ with $a \in A$, with the usual interpretation. Taking $F = A \times \mathsf{Id} \times \mathsf{Id} \simeq \coprod_{a \in A}(\mathsf{Id} \times \mathsf{Id})$ yields a logic with binary modalities $[a]$ with $a \in A$, with associated predicate liftings given by $[\![a]\!]_X(p_1, p_2)(\iota_{a'}(x, y)) = \begin{cases} p_1(x) \bullet p_2(y), & \text{if } a' = a \\ 0, & \text{otherwise} \end{cases}$, for $p_1, p_2 \in S^X$ and $x, y \in X$. Irrespective of the choice of F, when $S = (\{0, 1\}, \vee, 0, \wedge, 1)$, a formula φ holds in a state of a $\mathsf{T}_S \circ F$-coalgebra iff that state admits a maximal trace satisfying φ. For $S = ([0, 1], +, 0, *, 1)$ or $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$, $[\![\varphi]\!]_\gamma : C \to S$ measures the likelihood, resp. minimal cost of states of $\mathsf{T}_S \circ F$-coalgebras conforming to φ (suitably scaled according to the weighted sums present in φ).

▶ Remark 6. Taking $S = ([0, 1], +, 0, *, 1)$ and $F = 1 + A \times \mathsf{Id}$ yields a linear time logic for probabilistic transition systems. The absence of pure disjunctions makes this logic different from probabilistic LTL. A variant of our logics which incorporates disjunctions that can be resolved in one step (e.g. $[a]\varphi \vee [b]\psi$ with $a \neq b$) as new modalities turns out to be more expressive than probabilistic LTL (see [5, Example 4.3]). In this case, deterministic parity automata (known to be more expressive than LTL) can be directly encoded in the logic.

▶ Remark 7. By casting our logics into a dual adjunction framework, as done in [5], it follows immediately that $\mathsf{T}_S \circ F$-behavioural equivalence implies logical equivalence. This, however, is not very interesting, given the linear time nature of our logics. A detailed study of a weaker, trace-based notion of equivalence for which our logics are both sound and expressive is left as future work.

2.4 Equational Systems

The use of nested fixpoints in the semantics of fixpoint logics can elegantly be rephrased in terms of solutions of equational systems [1].

▶ **Definition 8.** An *equational system* over posets $L_1, \ldots, L_n$ is a sequence of equations $u_1 =_{\eta_1} f_1(u_1, \ldots, u_n)$, ..., $u_n =_{\eta_n} f_n(u_1, \ldots, u_n)$, where $u_1, \ldots, u_n$ are *variables*, $\eta_i \in \{\mu, \nu\}$ and $f_i : L_1 \times \ldots \times L_n \to L_i$ is a monotone function. A variable u_j is a *μ-variable* (*ν-variable*) if $\eta_j = \mu$ (resp. $\eta_j = \nu$).

A precise definition of the solution of an equational system can be found in [1, Section 1.4.4]. Intuitively, assuming that the L_is have enough suprema and infima, the *solution* of an equational system is defined as follows:

1. the first equation is solved (by taking either the least or the greatest solution, depending on η_1), to obtain an interim solution $u_1 = l_1^{(1)}(u_2, \ldots, u_n)$;
2. this is substituted for u_1 in the second equation, yielding a new equation $u_2 =_{\eta_2} f_2^{\ddagger}(u_2, \ldots, u_n)$;
3. the second equation is solved to obtain an interim solution $u_2 = l_2^{(2)}(u_3, \ldots, u_n)$;
4. continuing this way from left to right eventually eliminates all the variables and leads to a closed solution $u_n = l_n^{(n)} \in L_n$; and
5. closed solutions are propagated back from right to left to yield closed solutions for all of $u_1, \ldots, u_n$.

Instead of μ- and ν-annotations, some of the equational systems appearing later in the paper use natural numbers, with odd (even) values indicating μ- (resp. ν-) variables, and with the order of equations being determined by the natural order on $\mathbb{N}$. We also use a *generalised form* for equational systems, which allows several equations indexed by the same value, all of which are to be solved simultaneously.

3 An Automata-Based Approach to Trace Similarity

As already sketched in [6], notions of *maximal* and resp. *finite trace similarity* between states of $\mathsf{T}_S \circ F$-coalgebras can be defined by using a *double extension lifting* in place of the extension lifting L_S of (1). This section paves the way towards an automata-based characterisation of the semantics of the logic $\mu\mathcal{L}_\Lambda$, by providing a similar (and simpler) characterisation of maximal and resp. finite trace similarity.

The *double extension lifting* $L'_S : \mathsf{Rel}_{X,Y} \to \mathsf{Rel}_{\mathsf{T}_S X, \mathsf{T}_S Y}$ takes an S-valued relation $R : X \times Y \to S$ to the S-valued relation $\mu_1 \circ \mathsf{T}_S R \circ \mathsf{dst}_{X,Y} : \mathsf{T}_S X \times \mathsf{T}_S Y \to S$. Compared to L_S, L'_S uses the double strength map of T_S in place of the swapped strength map to yield a relation on $\mathsf{T}_S X \times \mathsf{T}_S Y$. As with L_S, this choice for L'_S is canonical (see [6]), and satisfies $L'_S(R)(\sum_i c_i x_i, \sum_j d_j y_j) = \mu_1(\sum_i \sum_j (c_i \bullet d_j) R(x_i, y_j))$.

▶ **Definition 9.** The *maximal* (resp. *finite*) *trace similarity relation* between two $\mathsf{T}_S \circ F$-coalgebras (C, γ) and (D, δ) is the greatest (resp. least) fixpoint of the following operator on S-valued relations between C and D:

$$\mathsf{Rel}_{C,D} \xrightarrow{\mathsf{Rel}(F)} \mathsf{Rel}_{FC,FD} \xrightarrow{L'_S} \mathsf{Rel}_{\mathsf{T}_S FC, \mathsf{T}_S FD} \xrightarrow{(\gamma \times \zeta)^*} \mathsf{Rel}_{C,D}$$

We write $\simeq^{\nu}_{\gamma,\delta} : C \times D \to S$ and $\simeq^{\mu}_{\gamma,\delta} : C \times D \to S$ for these relations.

▶ **Example 10.** For $S = (\{0,1\}, \vee, 0, \wedge, 1)$, maximal (finite) trace similarity relates precisely those states which admit a common maximal (resp. finite) trace. For $S = ([0,1], +, 0, *, 1)$ or $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$, maximal (finite) trace similarity measures the likelihood, resp. minimal joint cost of two states exhibiting a common maximal (resp. finite) trace.

We now introduce notions of ν- and *μ-extent* of a $\mathsf{T}_S \circ F$-coalgebra, and rephrase the definitions of $\simeq^{\nu}_{\gamma,\delta}$ and $\simeq^{\mu}_{\gamma,\delta}$ using these notions. The idea is to measure the weight with which a state in a coalgebra with branching can exhibit *any* maximal (resp. finite) trace. This weight is cumulative across all the branches.

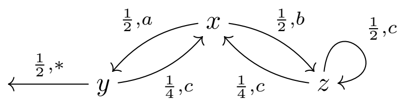
▶ **Definition 11.** The *ν-extent* (*μ-extent*) of a $\mathsf{T}_S \circ F$-coalgebra (A, α) is the gfp (resp. lfp) of the operator on S^A taking $p : A \to S$ to the composition

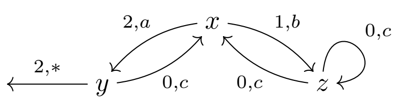
$$A \xrightarrow{\alpha} \mathsf{T}_S FA \xrightarrow{\mathsf{T}_S Fp} \mathsf{T}_S FS \xrightarrow{\mathsf{T}_S(\bullet_F)} \mathsf{T}_S S = \mathsf{T}_S^2 1 \xrightarrow{\mu_1} \mathsf{T}_S 1 = S$$

where $\bullet_F : FS \to S$ is given by $\bullet_F(\iota_\lambda(s_1, \ldots, s_{\mathsf{ar}(\lambda)})) = s_1 \bullet \ldots \bullet s_{\mathsf{ar}(\lambda)}$ for $\lambda \in \lambda$.

The above operator uses a one-step unfolding of the coalgebra structure to compute a finer approximation of the extent on a state based on the extent on its immediate successors. As the generality of F allows for immediate successors which are tuples of states, the semiring multiplication may also need to be used (in $\bullet_F$). The monad multiplication is used to accumulate the values from different branches. The composition in Definition 11 takes $a \in A$ with $\alpha(a) = \sum_i c_i(a_i^1, \ldots, a_i^{j_i})$ to $\mu_1(\sum_i c_i(p(a_i^1) \bullet \ldots \bullet p(a_i^{j_i})))$. That is, the extent associated to a particular state accumulates the extents associated to its immediate successors, scaled by the weights of the corresponding branches.

▶ **Example 12.** For the $\mathsf{T}_S \circ F$-coalgebras below, with $F = 1 + A \times \mathsf{Id}$ and $S = ([0,1], +, 0, *, 1)$ (resp. $S = (\mathbb{N}^\infty, \min, \infty, +, 0)$), the ν-extent maps x to 0.4, y to 0.6 and z to 0.2 (resp. x and y to 1 and z to 0), whereas the μ-extent again maps x to 0.4, y to 0.6 and z to 0.2 (resp. x and z to 4 and y to 2). Intuitively, the reason for the μ- and ν-extents being the same in the probabilistic case is that the likelihood of never reaching y from either x or z is 0.





The notions of ν- and μ-extent turn out to be particularly useful when applied to the *product* of two $\mathsf{T}_S \circ F$-coalgebras; this collects their common F-behaviour, suitably quantified with the help of the monad structure of T_S.

▶ **Definition 13.** The *product* of $\mathsf{T}_S \circ F$-coalgebras (C, γ) and (D, δ) is the $\mathsf{T}_S \circ F$-coalgebra with carrier $C \times D$ and transition function $\gamma \otimes \delta$ given by

$$C \times D \xrightarrow{\gamma \times \delta} \mathsf{T}_S FC \times \mathsf{T}_S FD \xrightarrow{\mathsf{dst}_{FC,FD}} \mathsf{T}_S(FC \times FD) \xrightarrow{\langle F\pi_1, F\pi_2 \rangle^*} \mathsf{T}_S F(C \times D)$$

where $\langle F\pi_1, F\pi_2 \rangle^*$ is pre-composition with $\langle F\pi_1, F\pi_2 \rangle : F(C \times D) \to FC \times FD$.

The effect of pre-composing with $\langle F\pi_1, F\pi_2 \rangle$ is that pairs of non-matching one-step behaviours are discarded from the resulting coalgebra.

Our first result relates the ν- and μ-extents of the product of two coalgebras with the maximal and respectively finite trace similarity relation between them.

▶ **Theorem 14.** *Let (C, γ) and (D, δ) be two $\mathsf{T}_S \circ F$-coalgebra. The ν-extent (μ-extent) of the product coalgebra $(C \times D, \gamma \otimes \delta)$ coincides with the maximal trace similarity relation $\simeq^{\nu}_{\gamma,\delta}$ (resp. the finite trace similarity relation $\simeq^{\mu}_{\gamma,\delta}$).*

Proof (sketch). The proof involves showing that the operators used in the definition of $\simeq^{\nu}_{\gamma,\delta}$ and $\simeq^{\mu}_{\gamma,\delta}$ on the one hand, and of the ν-/μ-extent of the product automaton on the other, coincide. ◀

4 Parity (S, F)-Automata and their Extent

We now consider *parity* (S, F)*-automata*, as extensions of $\mathsf{T}_S \circ F$-coalgebras with a parity map, and define their *extent*. Parity maps are a natural way to formulate acceptance conditions for automata on infinite words/trees: the states of the automaton are assigned natural number *parities*, and a run of the automaton is *accepting* iff the largest parity occurring infinitely often along the run is even (see e.g. [11]). We use parity maps in a similar way, but this time in a quantitative setting where the emphasis moves away from individual runs.

▶ **Definition 15.** A *parity* (S, F)*-automaton* is given by a $T_S \circ F$-coalgebra (A, α) together with a function $\Omega : A \to \{1, 2, \dots\}$ with finite range, called *parity map*.

A similar notion called (T, F)*-system* was considered in [17], albeit under different assumptions on the monad T, which rule out semiring monads as considered here. A trace semantics for a (T, F)-system was defined in loc. cit. as a Kleisli map obtained by taking least and greatest fixpoints, and this was proved to instantiate to the standard notions of acceptance by a non-deterministic, resp. probabilistic parity automaton. As in [17], our assignment of parities to *all* states of the automaton allows for a smooth relationship to equational systems.

The *extent* of a parity automaton generalises the ν- and μ-extents of a coalgebra by taking into account the different parities associated to the automaton states. The only difference is that now the extent involves a collection of *nested* fixpoints, one for each parity.

▶ **Definition 16.** Let (A, α, Ω) be a parity (S, F)-automaton with $\mathsf{ran}(\Omega) \subseteq \{1, \dots, n\}$, let $A_k = \{a \in A \mid \Omega(a) = k\}$, and let $\alpha_k = \alpha \circ \iota_k : A_k \to \mathsf{T}_S FA$ denote the restriction of α to A_k. The *extent* $e = [e_1, \dots, e_n] : A \to S$ *of* (A, α, Ω) is the solution of the equational system

$$\begin{bmatrix} u_1 & =_\mu & \mu_1 \circ \mathsf{T}_S(\bullet_F) \circ T_S F[u_1, \dots, u_n] \circ \alpha_1 \\ & \vdots & \\ u_n & =_\eta & \mu_1 \circ \mathsf{T}_S(\bullet_F) \circ T_S F[u_1, \dots, u_n] \circ \alpha_n \end{bmatrix} \tag{3}$$

with $\eta = \mu$ ($\eta = \nu$) if n is odd (resp. even), with variables u_k ranging over the poset $(S^{A_k}, \sqsubseteq)$ (and therefore $[u_1, \dots, u_n] : A \to S$), and with the rhss of the equations pictured below:

$$A_k \xrightarrow{\alpha_k} \mathsf{T}_S FA \xrightarrow{\mathsf{T}_S F[u_1,\dots,u_n]} \mathsf{T}_S FS \xrightarrow{\mathsf{T}_S(\bullet_F)} \mathsf{T}_S S = \mathsf{T}_S^2 1 \xrightarrow{\mu_1} \mathsf{T}_S 1 = S$$

▶ **Example 17.** For non-deterministic/probabilistic/weighted systems, the extent captures the existence/likelihood/minimal cost of an accepting run. For the coalgebras in Example 12, assigning parities 1 to the states x and y and 2 to the state z yields automata with extents mapping x to 0.4, y to 0.6 and z to 0.2, and resp. x to 1, y to 1 and z to 0. The reason for the extent being similar to the μ- and ν-extents of the underlying coalgebra in the probabilistic case is that, although runs which visit z infinitely often contribute to the extent, under the current assignment of probabilities to transitions, the contributed quantity is 0. The situation would be different if the transition from z to x was removed and the one from z to itself had probability 1, in which case the extent would map x to $\frac{12}{14}$, y to $\frac{5}{7}$ and z to 1.

▶ Remark 18. A trace semantics for parity (S, F)-automata similar to that of [17] can also be defined, using an equational system whose variables u_k range over $S^{A_k \times Z}$. In this case, the rhs of the kth equation is given by:

$$\begin{tikzcd}
A_k \times Z \arrow[r, "\alpha \times (\eta_{FZ} \circ \zeta)"] & \mathsf{T}_S FA \times \mathsf{T}_S FZ \arrow[r, "\mathsf{dst}"] & \mathsf{T}_S(FA \times FZ) \arrow[r, "\langle F\pi_1, F\pi_2 \rangle^*"] & \mathsf{T}_S F(A \times Z) \arrow[d, "\mathsf{T}_S F[u_1,\dots,u_n]"] \\
 & S & \mathsf{T}_S S \arrow[l, "\mu_1"'] & \mathsf{T}_S FS \arrow[l, "\mathsf{T}_S(\bullet_F)"']
\end{tikzcd}$$

5 From Linear Time Logics to Parity (S, F)-Automata

We now show how to assign, to each *clean* and *guarded* formula of $\mu\mathcal{L}_\Lambda$ (Definition 19), a parity (S, F)-automaton. We then give an automata-theoretic characterisation of the semantics of a $\mu\mathcal{L}_\Lambda$-formula, using the extent of a product automaton (between a model and a formula automaton). The next definition is standard for fixpoint logics (see e.g. [18]).

▶ **Definition 19.** For a set $\mathcal{V}$ of variables, a formula $\varphi \in \mu\mathcal{L}_\Lambda^\mathcal{V}$ is *clean* if no variable appears both free and bound, or is bound more than once, in φ, and *guarded* if each occurrence of a bound variable inside its defining fixpoint formula lies within the scope of a modal operator.

For a clean formula $\varphi \in \mu\mathcal{L}_\Lambda^\mathcal{V}$, we write $\mathsf{BVar}(\varphi)$ for its set of bound variables. Also, for $x, y \in \mathsf{BVar}(\varphi)$, we write $\varphi_x = \eta x.\psi_x$ for the unique sub-formula of φ which binds x, and $y \leq x$ iff φ_y is a sub-formula of φ_x. Our technical development will require first transforming a guarded formula to a *strictly guarded* one, as defined below.

▶ **Definition 20.** A formula $\varphi \in \mu\mathcal{L}_\Lambda^\mathcal{V}$ is (i) *strongly guarded* if every occurrence of a fixpoint variable x inside the defining formula ψ_y of a variable y (with $y \leq x$) appears within the scope of a modal operator, and (ii) *strictly guarded* if every occurrence of a fixpoint variable x inside ψ_x is immediately preceded by a modal operator.

Guardedness requires that, in the formula syntax tree, one cannot pass from a fixpoint quantification $\eta y.\psi_y$ to an occurrence of y without encountering a modal operator. Strong guardedness additionally requires that this is the case when passing from $\eta y.\psi_y$ to *any* fixpoint variable x occurring inside ψ_y. It is easy to see that every guarded formula is equivalent to a strongly guarded one. To see this, assume for simplicity that $S = (\{0,1\}, \vee, 0, \wedge, 1)$ and therefore the logic only contains non-weighted sums (i.e. disjunctions) which we denote by $+$. Then, a non-strongly guarded occurrence of variable x, necessarily of the form $\eta x.\varphi[\eta' y.(x+\psi)]$ with φ a formula with a guarded hole, is equivalent to $\eta x.\varphi[x + \eta' y'.\psi[(x + y')/y]]$, where x is now guarded in $\psi[(x + y')/y]$ as y was initially guarded in ψ. (This argument generalises to weighted sums, where now $x+\psi$ is replaced by $\sum_i c_i\psi_i$ with $\psi_i = x$ for some i. However, when S is a *partial* semiring, an additional unfolding of the formula $\eta' y'.\psi[(x + y')/y]$ is required, as the counterpart of the sum $x + \eta' y'.\psi[(x + y')/y]$ may not be defined.) Strict guardedness additionally requires that occurrences of x inside ψ_y are *immediately* preceded by modal operators. Given the distributivity of modal operators over weighted sums (an immediate consequence of the definition of $[\![\lambda]\!]$), one can translate a strongly guarded formula into an equivalent, strictly guarded one by pushing weighted sums outside the modal operators.

The next definition can be traced back to the notion of *Fischer-Ladner closure* [12].

▶ **Definition 21.** A set $C \subseteq \mu\mathcal{L}_\Lambda^\mathcal{V}$ of formulas is *closed* if
- $\psi[\eta x.\psi/x] \in C$ whenever $\eta x.\psi \in C$, for $\eta \in \{\mu, \nu\}$,
- $\varphi_i \in C$ for $i \in \{1, \ldots, n\}$, whenever $\sum\limits_{i\in\{1,\ldots,n\}} c_i \bullet \varphi_i \in C$,
- $\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)} \in C$ whenever $[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)}) \in C$.

The *closure* $\mathsf{Cl}(\varphi)$ of a $\mu\mathcal{L}_\Lambda^\mathcal{V}$-formula φ is the smallest closed set containing φ.

We note that $\mathsf{Cl}(\varphi)$ is always finite: the second and third clauses above can only be applied finitely many times before the first clause applies; and applications of the first clause only lead to the inclusion of a new formula in $\mathsf{Cl}(\varphi)$ once for each fixpoint sub-formula of φ.

We now exploit the close relationship between the logic $\mu\mathcal{L}_\Lambda$ on the one hand and the semiring S and endofunctor F on the other to associate, to each clean and strictly guarded formula $\varphi \in \mu\mathcal{L}_\Lambda$, a parity (S, F)-automaton with carrier $\mathsf{Cl}(\varphi)$. To this end, we assign natural numbers n_x to variables $x \in \mathsf{BVar}(\varphi)$ in a way which is consistent with the order $\mathsf{BVar}(\varphi)$, that is, $n_x \leq n_y$ whenever $x \leq y$, and which differentiates between least and greatest fixpoints, that is, n_x is odd (even) if $\varphi_x = \mu x.\psi$ (resp. $\varphi_x = \nu x.\psi$). The number of parities can be optimised to equal the *alternation depth* of φ, given by the number of alternations between least and greatest fixpoints [1]. Hereafter we assume that φ is a fixpoint formula (otherwise the formula $\nu x.\varphi$ with x a fresh variable can be considered instead).

▶ **Definition 22.** The *parity* (S, F)*-automaton* $(\mathsf{Cl}(\varphi), \beta, \Omega)$ associated to a clean and strictly guarded formula $\varphi \in \mu\mathcal{L}_\Lambda$ with $\varphi = \varphi_z = \eta z.\psi_z$ has $\beta : \mathsf{Cl}(\varphi) \to \mathsf{T}_S F\mathsf{Cl}(\varphi)$ and $\Omega : \mathsf{Cl}(\varphi) \to \{1, 2, \ldots\}$ defined by induction on the structure of $\mathsf{Cl}(\varphi)$:

- $\Omega(\eta x.\psi_x) = n_x$ (and hence $\Omega(\varphi) = n_z$),
- $\beta(\eta x.\psi_x) = \beta(\psi[\eta x.\psi_x/x])$; $\Omega(\psi[\eta x.\psi/x]) = n_x$, unless $\psi[\eta x.\psi/x]$ is a fixpoint formula itself, in which case the previous clause applies,
- $\beta(\sum_{i\in\{1,\ldots,n\}} c_i \bullet \varphi_i) = \mu_{F\mathsf{Cl}(\varphi)}(\sum_{i\in\{1,\ldots,n\}} c_i\, \beta(\varphi_i))$; $\Omega(\varphi_i) = \Omega(\sum_{i\in\{1,\ldots,n\}} c_i \bullet \varphi_i)$ for $i \in \{1, \ldots, n\}$, unless φ_i is a fixpoint formula, in which case the first clause applies,
- $\beta([\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})) = \eta_{F\mathsf{Cl}(\varphi)}(\iota_\lambda(\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)}))$; $\Omega(\varphi_i) = \Omega([\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)}))$ for $i \in \{1, \ldots, \mathsf{ar}(\lambda)\}$, unless φ_i is a fixpoint formula, in which case the first clause applies,

with $\eta : \mathsf{Id} \Rightarrow \mathsf{T}_S$ and $\mu : \mathsf{T}_S^2 \Rightarrow \mathsf{T}_S$ the unit and resp. multiplication of T_S. (Note the difference in denotations between the first and the second occurrences of the $\sum$ symbol in the third clause: the first is part of a propositional symbol of $\mu\mathcal{L}_\Lambda$, whereas the second describes an element of $\mathsf{T}_S\mathsf{T}_S F\mathsf{Cl}(\varphi)$ as a formal sum.)

Thus, formulas in $\mathsf{Cl}(\varphi)$ are assigned parities starting from the outermost formula, and with a formula receiving the same parity as the smallest formula which contains it – unless the given formula is a fixpoint one $\eta_x.\psi_x$, in which case its parity is n_x. This is standard (see e.g. [7]), and gives $\Omega(\psi') \leq \Omega(\psi)$ whenever ψ' is a sub-formula of ψ different from a fixpoint formula. However, if a formula $\psi \in \mathsf{Cl}(\varphi)$ occurs several times within φ, $\Omega(\psi)$ is defined more than once according to the above definition. In this case, a copy of ψ should be made within $\mathsf{Cl}(\varphi)$ for each occurrence of ψ, with different copies assigned possibly different parities. To avoid complicating the definition of the parity automaton associated to φ, and our subsequent exposition, we assume (for the above definition) that φ does not contain several occurrences of any sub-formula. Our technical development does not, however, depend on this assumption.

The recursive definition of β, with base case $[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})$, deviates from the more standard approach to translating fixpoint formulas to automata (see e.g. [18]), which applies more generally to unguarded formulas and involves an intermediate automaton with silent steps. Here, silent steps are automatically absorbed into the next non-silent transition, with no unwanted consequences. This is possible due to our strict guardedness assumption:

1. Guardedness ensures well-definedness of β, as only a finite number of applications of the second and third clauses of Definition 19 are possible before the last clause applies.
2. Strict guardedness ensures that the implicit elimination of silent steps is such that every path from φ_y to x in the formula syntax tree corresponds to a "path" from $\varphi_y[\varphi_x/x]$ to φ_x in the resulting automaton. Thus, states (such as φ_x) with parities greater than the parity associated to φ_y are not sidestepped during the implicit elimination of silent steps. The first formula of Example 23 illustrates why strict guardedness is needed.

▶ **Example 23.** Assume $F = 1 + A \times \mathsf{Id}$. Thus, the associated logic contains a single nullary modality $*$ and unary modalities $[a]$ with $a \in A$.

1. The strongly guarded formula $\xi_x := \nu x.\xi_y$ with $\xi_y := \mu y.[a](x + y)$, with associated automaton shown below on the left, is not strictly guarded. This automaton does not

faithfully represent ξ_x, as it does not accept a^ω (since $\Omega(\xi_x + \xi_y[\xi_x/x]) = \Omega(\xi_y) = 1$). On the other hand, the automaton associated to the equivalent, strictly guarded formula $\xi'_x := \nu x.\xi'_y$ with $\xi'_y := \mu y.([a]x + [a]y)$, shown below on the right, correctly accepts a^ω (as $\Omega(\xi'_x) = 2$). The problem with ξ_x is that the path from ξ_y to x in the formula syntax tree does not correspond to a path from $\xi_x + \xi_y[\xi_x/x]$ to ξ_x in the resulting automaton (and similarly for the path from ξ_x to x).

$$\begin{tikzcd}
\xi_x \arrow[r, "a"] & \xi_x + \xi_y[\xi_x/x] \arrow[loop above, "a"] & \xi_y[\xi_x/x] \arrow[l, "a"']
\end{tikzcd}
\qquad
\begin{tikzcd}
\xi'_x \arrow[loop above, "a"] \arrow[r, bend left, "a"] & \xi'_y[\xi'_x/x] \arrow[loop above, "a"] \arrow[l, bend left, "a"]
\end{tikzcd}$$

2. The automata associated to the strictly guarded formulas $\varphi_x := \nu x.(\varphi_y + [b]x)$ with $\varphi_y := \mu y.(* + [a]y)$, and $\psi_x := \nu x.\psi_y$ with $\psi_y := \mu y.[c]([a]x + [b]y)$, are:

$$\begin{tikzcd}
{} & \varphi_x \arrow[l, "*"'] \arrow[loop above, "b"] \arrow[r, "a"] & \varphi_y \arrow[loop above, "a"] \arrow[r, "*"] & {}
\end{tikzcd}
\qquad
\begin{tikzcd}
\psi_x \arrow[r, bend right, "c"'] & {[a]\psi_x + [b]\psi'_y} \arrow[l, bend right, "a"'] \arrow[r, bend left, "b"] & \psi'_y \arrow[l, bend left, "c"]
\end{tikzcd}$$

with $\psi'_y := \psi_y[\psi_x/x]$, $\Omega(\varphi_x) = \Omega(\psi_x) = 2$ and $\Omega(\varphi_y) = \Omega(\psi'_y) = \Omega([a]\psi_x + [b]\psi'_y) = 1$.

Given a $\mathsf{T}_S \circ F$-coalgebra (C, γ) and $\varphi \in \mu\mathcal{L}_\Lambda$ with associated automaton $(\mathsf{Cl}(\varphi), \beta, \Omega)$, one can endow the product of (C, γ) and $(\mathsf{Cl}(\varphi), \beta)$ with a parity map by setting $\Omega(c, \psi) = \Omega(\psi)$ for $(c, \psi) \in C \times \mathsf{Cl}(\varphi)$. The next result provides a characterisation of $[\![\varphi]\!]_\gamma$ using the extent of the resulting parity automaton.

▶ **Theorem 24.** *If $(\mathsf{Cl}(\varphi), \beta, \Omega)$ with $\mathsf{ran}(\Omega) \subseteq \{1, \ldots, n\}$ is the parity automaton for a clean and strictly guarded formula $\varphi \in \mu\mathcal{L}_\Lambda$, (C, γ) is a $\mathsf{T}_S \circ F$-coalgebra and $e = [(e_n)_{n \in \mathsf{ran}(\Omega)}] : A \to S$ is the extent of the product parity automaton (A, α, Ω) of (C, γ) and $(\mathsf{Cl}(\varphi), \beta, \Omega)$, then $[\![\varphi]\!]_\gamma(c) = e(c, \varphi)$ for $c \in C$.*

The proof of Theorem 24 involves defining an equational system E_φ in generalised form (see Section 2.4), whose solution is known to provide an alternative characterisation of $[\![\varphi]\!]_\gamma$, and proving that this solution can alternatively be characterised using the extent of the product automaton (A, α, Ω). This intermediary result makes use of solution-preserving substitutions of rhs*s* of equations in E_φ for the respective variables, to transform E_φ into an equational system equivalent to the one used to define the extent of the product automaton.

The equational system E_φ employs a variable u_ψ ranging over S^C for each formula $\psi \in \mathsf{Cl}(\varphi)$. In order to specify the rhs*s* of E_φ, we define, for each $\psi \in \mathsf{Cl}(\varphi)$, a term $d_\psi : (S^C)^{\mathsf{Cl}(\varphi)} \to S^C$ over these variables:

$$\begin{aligned}
d_{[\lambda](\varphi_1, \ldots, \varphi_{\mathsf{ar}(\lambda)})} &= \gamma^*(\mathsf{ext}_{FC}([\![\lambda]\!](u_{\varphi_1}, \ldots, u_{\varphi_{\mathsf{ar}(\lambda)}}))), \\
d_{\sum_{i \in \{1,\ldots,n\}} c_i \bullet \varphi_i} &= \mu_1\Big(\sum_{i \in \{1,\ldots,n\}} c_i u_{\varphi_i}\Big), \\
d_{\eta x.\psi} &= u_{\eta x.\psi}.
\end{aligned} \tag{4}$$

▶ **Definition 25.** For $\varphi \in \mu\mathcal{L}_\Lambda$ clean and strictly guarded, the *system E_φ* collects (i) all equations $u_{\eta x.\psi} =_{\Omega(\eta x.\psi)} d_{\psi[\eta x.\psi/x]}$ with $\eta x.\psi \in \mathsf{Cl}(\varphi)$, and (ii) all equations $u_\xi =_{\Omega(\xi)} d_\xi$ with $\xi \in \mathsf{Cl}(\varphi)$, $\xi \neq \eta x.\psi$, where the u_ψs range over S^C for $\psi \in \mathsf{Cl}(\varphi)$.

The following result is folklore (see e.g. [7] for a similar result).

▶ **Proposition 26.** *For $\varphi \in \mu\mathcal{L}_\Lambda$ clean and guarded, and (C, γ) a $\mathsf{T}_S \circ F$-coalgebra, let $(v_\psi)_{\psi \in \mathsf{Cl}(\varphi)}$ denote the solution of the equational system E_φ. Then, for $\psi \in \mathsf{Cl}(\varphi)$, $[\![\psi]\!]_\gamma : C \to S$ coincides with $d_\psi[(v_\xi/u_\xi)_{\xi \in \mathsf{Cl}(\varphi)}]$.*

▶ **Example 27.** The systems of equations associated to ξ_x and resp. φ_x of Example 23 are

$$\begin{bmatrix} u_{\xi_x} & =_2 & u_{\xi_y[\xi_x/x]} \\ u_{\xi_y[\xi_x/x]} & =_1 & \gamma^*(\mathsf{ext}_{FC}(\llbracket a \rrbracket u_{\xi_x+\xi_y[\xi_x/x]})) \\ u_{\xi_x+\xi_y[\xi_x/x]} & =_1 & \mu_1(u_{\xi_x} + u_{\xi_y[\xi_x/x]}) \end{bmatrix}, \quad \begin{bmatrix} u_{\varphi_x} & =_2 & \mu_1(u_{\varphi_y} + u_{[b]\varphi_x}) \\ u_{[b]\varphi_x} & =_2 & \gamma^*(\mathsf{ext}_{FC}(\llbracket b \rrbracket u_{\varphi_x})) \\ u_{\varphi_y} & =_1 & \mu_1(u_* + u_{[a]\varphi_y}) \\ u_* & =_1 & \gamma^*(\mathsf{ext}_{FC}(\llbracket * \rrbracket)) \\ u_{[a]\varphi_y} & =_1 & \gamma^*(\mathsf{ext}_{FC}(\llbracket a \rrbracket u_{\varphi_y})) \end{bmatrix}$$

The definitions of both E_φ and β are driven by the structure of $\mathsf{Cl}(\varphi)$, and use the same strategy to associate parities to formulas. However, the definition of β sidesteps certain formulas during the implicit elimination of silent steps from the resulting automaton. The proof of Theorem 24, not included here for space reasons, shows that, under the assumption that φ is strictly guarded, carrying out a suitable choice of substitutions (those implicitly performed in the definition of β) on E_φ results in an equational system equivalent to both E_φ and the system of equations defining the extent of the product automaton.

For the modal μ-calculus, a converse translation, from parity automata to fixpoint formulas, also exists. This is defined by induction on the number of parities and uses *vectorial syntax* as an intermediary step. Vectorial syntax [1] generalises standard fixpoint calculus syntax by replacing fixpoint variables with arrays of such variables, all with the same parity, with the corresponding fixpoints being computed simultaneously. A similar translation from parity (S, F)-automata to $\mu\mathcal{L}_\Lambda$-formulas can be defined here. A translation from a parity (S, F)-automaton to vectorial syntax is straightforward: each automaton state yields a new variable, with parity given by the automaton, and whose defining formula is taken from the coalgebra map of the automaton. A translation from vectorial to standard syntax is then carried out by appealing to the Bekič principle – this allows reducing a simultaneous fixpoint to a sequence of individual fixpoints (see e.g. [1, Lemma 1.4.2]). The formula associated to the original automaton can now be read directly from the resulting equational system.

▶ **Theorem 28.** *For a parity (S, F)-automaton (A, α, Ω) with $A_{\max(\mathsf{ran}(\Omega))} = \{a\}$, there exists $\varphi_a \in \mu\mathcal{L}_\Lambda$ such that, for every $\mathsf{T}_S \circ F$-coalgebra (C, γ), if $e : [(e_n)_{n\in\mathsf{ran}(\Omega)}]$ is the extent of the product parity automaton between (C, γ) and (A, α, Ω) then $\llbracket \varphi_a \rrbracket_\gamma(c) = e(c, a)$ for $c \in C$.*

Proof (sketch). The proof closely follows that of [19, Theorem 34]. ◀

▶ Remark 29. The results of this section can easily be extended to the variant of $\mu\mathcal{L}_\Lambda$ described in Remark 6: while formula and model automata now have slightly different types, their product, itself a parity (S, F)-automaton, can be constructed in a similar way.

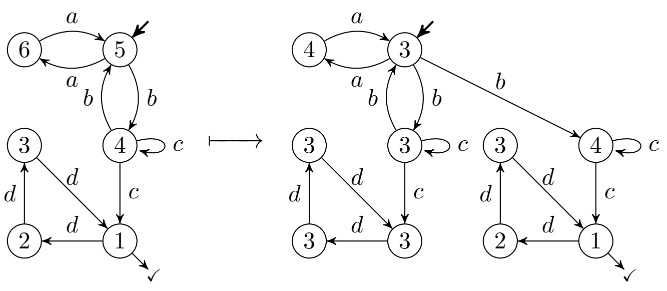
6 From Parity to Büchi Automata

We now present a direct reduction from *parity word automata* to *Büchi ones*. Parity word automata are parity $(S, F_{\Sigma,\Delta})$-automata, with $F_{\Sigma,\Delta} = \Sigma \times \mathsf{Id} + \Delta$ for alphabets Σ and Δ. Büchi automata have $\mathsf{ran}(\Omega) = \{1, 2\}$. Here we additionally assume the semiring S to be total. Our reduction involves manipulating "*linear*" equational systems; their rhs*s* use operations that resemble matrix-vector multiplication and vector addition, as defined below.

▶ **Definition 30.** Let X and Y be sets, and let $M : X \times Y \to S$ and $v : Y \to S$ be a relation and a predicate respectively. Assume M and v to have finite support. $M \bullet v : X \to S$ is defined by $(M \bullet v)(x) = \sum_{y\in Y} M(x, y) \bullet v(y)$. For predicates $v_1, v_2 : Y \to S$, $v_1 + v_2$ is defined by pointwisely extending $+$ on S.

By the finite support property, $M \bullet v$ is well-defined even for X or Y infinite. The next lemma concerning linear equational systems is key to our reduction.

Figure 1 A step of the reduction from a parity automaton to a Büchi automaton. The number on each node denotes its parity. This step does not change the accepted language $((b\,c{*}\,b){*}\,(aa){*})\omega \mid ((b\,c{*}\,b){*}\,(aa){*}){*}\,b\,c\omega \mid ((b\,c{*}\,b){*}\,(aa){*}){*}\,b\,c{*}\,c(ddd){*}$.



▶ **Lemma 31.** *Let a set $X = X_1+\ldots+X_n$, a number $k \in [1,n]$ and a relation $M : X \times X \to S$ be fixed. Let $X_{\le k} = X_1+\ldots+X_k$ and $X_{>k} = X_{k+1}+\ldots+X_n$. For a predicate $v : X_{\le k} \to S$, the equational systems $E^{\eta}_k(v)$ and $E^{\mu}_k(v)$, whose solutions belong to $S^{X_1} \times \ldots \times S^{X_k} \cong S^{X_{\le k}}$, are defined as follows. Polarities $\eta_1, \ldots, \eta_k \in \{\mu,\nu\}$ of $E^{\eta}_k(v)$ can be chosen arbitrarily.*

$$E^{\eta}_k(v) = \begin{bmatrix} u_1 & =_{\eta_1} & M_1 \bullet [u_1,\ldots,u_k] + v_1 \\ & \vdots & \\ u_k & =_{\eta_k} & M_k \bullet [u_1,\ldots,u_k] + v_k \end{bmatrix}, \; E^{\mu}_k(v) = \begin{bmatrix} u_1 & =_{\mu} & M_1 \bullet [u_1,\ldots,u_k] + v_1 \\ & \vdots & \\ u_k & =_{\mu} & M_k \bullet [u_1,\ldots,u_k] + v_k \end{bmatrix}$$

Here each sub-relation $M_i : X_i \times X_{\le k} \to S$ and each sub-predicate $v_i : X_i \to S$ are given by domain restriction. Then for any $v^{\eta}, v^{\mu} : X_{\le k} \to S$, the solution of $E^{\eta}_k(v^{\eta} + v^{\mu})$ is obtained by summing the solutions of $E^{\eta}_k(v^{\eta})$ and $E^{\mu}_k(v^{\mu})$.

The reduction proceeds in a step-wise manner, decrementing the largest parity, assumed without loss of generality to be even, by 2 at each step. An example is given in Figure 1. One reduction step is as follows:

1. Create a copy of states with parity lower than $n-1$. Incoming (only!) edges to those states are also copied. (This is only possible if the semiring is *total*.)
2. For the old copy of states with parity lower than $n-1$, let the new parity be $n-3$ and drop explicit terminations (i.e. transitions with a nullary letter).
3. Decrement the priorities of the other states by 2.

The intuition is that, from old copies of states, a state with parity higher than $n-2$ must be visited again for acceptance; whereas from new copies, such a state must *not* be visited. The correctness of this translation is proved using Lemma 31. This also explains why the shape of F must be restricted to $F_{\Sigma,\Delta}$ – this makes extents be given by *linear* equational systems.

By repeatedly applying the reduction step, any parity $(S, F_{\Sigma,\Delta})$-automaton $\mathcal{A}$ can be reduced to a Büchi $(S, F_{\Sigma,\Delta})$-automaton $\mathcal{A}^{\mathrm{B}}$. Since the reduction is such that an accepting path of $\mathcal{A}$ corresponds to exactly *one* accepting path of $\mathcal{A}^{\mathrm{B}}$, idempotence of $+$ is not required.

▶ **Theorem 32.** *Let $\mathcal{A} = (A, \alpha, \Omega)$ be a parity $(S, F_{\Sigma,\Delta})$-automaton with $\mathsf{ran}(\Omega) = [1,n]$. Assume without loss of generality that n is even. The Büchi $(S, F_{\Sigma,\Delta})$-automaton $\mathcal{A}^{\mathrm{B}} = (A^{\mathrm{B}}, \alpha^{\mathrm{B}}, \Omega^{\mathrm{B}})$ is given by*

$$\begin{aligned} A^{\mathrm{B}} &= \textstyle\coprod_{k\in[1,n]} A_k \times \{i \mid i \in [k,n], i \text{ is even}\} \\ \alpha^{\mathrm{B}}(a,i)(\iota_\sigma(a',j)) &= \begin{cases} \alpha(a)(\iota_\sigma(a')) & \text{if } i = j, \text{ or } \lceil\Omega(a)\rceil = i \text{ and } i > j \\ 0 & \text{otherwise} \end{cases} \\ \alpha^{\mathrm{B}}(a,i)(\iota_\delta) &= \begin{cases} \alpha(a)(\iota_\delta) & \text{if } \lceil\Omega(a)\rceil = i \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

$$\Omega^{\mathrm{B}}(a,i) \;=\; \begin{cases} 2 & \textit{if } \lceil \Omega(a) \rceil = i \textit{ and } i \textit{ is even} \\ 1 & \textit{otherwise} \end{cases}$$

Here $\lceil \Omega(a) \rceil = \min\{k \mid k \geq \Omega(a)$ *and* k *is even*$\}$*. The automaton* $\mathcal{A}^{\mathrm{B}}$ *satisfies*

$$\mathsf{ext}(\mathcal{A} \times \mathcal{M})(a,m) \;=\; \textstyle\sum_{(a,i)\in A^{\mathrm{B}}} \mathsf{ext}(\mathcal{A}^{\mathrm{B}} \times \mathcal{M})((a,i),m) \tag{5}$$

for any $\mathsf{T}_S \circ F_{\Sigma,\Delta}$*-coalgebra* $\mathcal{M}$ *and each* $(a,m) \in A \times M$*.*

▶ Remark 33. The definition of $\mathcal{A}^{\mathrm{B}}$ does not generalise to partial semirings S, due to the duplication of states involved. However, if S can be extended to a total semiring S' satisfying our assumptions, then carrying out the reduction for $\mathcal{A}$ as an $(S', F_{\Sigma,\Delta})$-automaton yields the desired result, namely a Büchi $(S', F_{\Sigma,\Delta})$-automaton $\mathcal{A}^{\mathrm{B}}$ that satisfies condition (5). In particular, this means that one can treat the probabilistic case ($S = ([0,1], +, 0, *, 1)$) by moving to the total semiring $S' = (\mathbb{R}^{\infty}, +, 0, *, 1)$. Specifically, one can model check formulas in our probabilistic linear time μ-calculus by (i) taking the product of the given model and formula automata, (ii) reducing the resulting automaton, viewed as an $(S', F_{\Sigma,\Delta})$-automaton, to a Büchi one, and (iii) computing the extent of the $(S', F_{\Sigma,\Delta})$-automaton thus obtained.

Theorem 32 together with the existence of semantics preserving translations from $\mu\mathcal{L}_\Lambda$-formulas to parity (S,F)-automata and back (Theorems 24 and 28) now give us the following:

▶ **Corollary 34.** *For* $F = F_{\Sigma,\Delta}$*, the alternation degree* 1 *fragment of* $\mu\mathcal{L}_\Lambda$ *is fully expressive.*

7 An Algorithm for Computing Extents

We now describe an algorithm for computing the extent of a parity (S,F)-automaton, under the additional assumption that the length of any strictly ascending/descending chain in $(S, \sqsubseteq)$ is bounded. Both the boolean semiring and the bounded version of the tropical semiring (see Example 2) satisfy this assumption.

We fix a parity (S,F)-automaton (A, α, Ω). Thus, for $a \in A$, $\alpha(a)$ is a finite weighted sum of tuples $\iota_\lambda(a_1, \ldots, a_k)$ with $\lambda \in \Lambda$, $k = \mathsf{ar}(\lambda)$ and $a_i \in A$. The algorithm is described in Figure 2, and employs a recursive procedure $\mathit{Extent}(n \in \omega)$.

To see that the algorithm terminates, note that each call of $\mathit{Extent}(n)$ either increases or decreases all the values in A_n. The assumed bound on the length of strictly ascending/descending chains guarantees termination of each call: each iteration of the **repeat** . . . **until** changes at least one of the values $e(a)$ with $a \in A_n$. Several improvements to the algorithm are possible, e.g. only remembering certain extent values (those with parity n) in the variable *old*; and deferring the recursive call until the approximation on the current parity saturates.

Next, we discuss complexity. If $|\mathsf{ran}(\Omega)| = 1$, and assuming for simplicity that S is finite, the algorithm has a time complexity which is quadratic in the size of the automaton – the length of a strictly increasing/decreasing chain in B^{A_i} is at most $|A_i||B|$, each iteration (lines 8-10 in the algorithm) increases/decreases at least one of the values $e(a)$, and takes time linear in $|A|$. If $|\mathsf{ran}(\Omega)| = m > 1$, then since each recursive call of Extent only updates values $e(a)$ with $a \in A_i$, for some $i \in \mathsf{ran}(\Omega)$, the time complexity is $O(|A|^{|\mathsf{ran}(\Omega)|}|B|^{|\mathsf{ran}(\Omega)|}\prod_{i\in\mathsf{ran}(\Omega)}|A_i|)$.

We conclude by noting that, for *Büchi* (S,F)-automata, the algorithm can be generalised to semirings S where only strictly *ascending* chains in $(S, \sqsubseteq)$ are required to have bounded length. This extends applicability e.g. to the tropical semiring. Instead of the exact extent of a Büchi automaton, the generalised algorithm computes increasingly finer over-approximations of the extent. The boundedness assumption guarantees that *inner* calls to Extent terminate.

Figure 2 Algorithm for computing the extent $e : A \to S$ of (A, α, Ω)

```
Input: parity automaton $(A, \alpha, \Omega)$
Output: extent $e : A \to S$ of $(A, \alpha, \Omega)$
 1. let $e := [a \mapsto 0_S]$
 2. $\mathit{Extent}(\max(\mathsf{ran}(\Omega)))$
Procedure $\mathit{Extent}(n \in \mathbb{N})$
 1. if $n = 0$ then return endif
 2. for $a \in A_n$ do
 3.    $e(a) \leftarrow \begin{cases} 0_S, & \text{if } n \text{ is even} \\ 1_S, & \text{otherwise} \end{cases}$
 4. endfor
 5. repeat
 6.   let $\mathit{old} := e$
 7.   $e \leftarrow \mathit{Extent}(n-1)$
 8.   for $a \in A_n$ do
 9.     $e(a) \leftarrow \sum_{i \in I} v_i \bullet \mathit{old}(a_1) \bullet \ldots \bullet \mathit{old}(a_{j_i})$
          where $\alpha(a) = \sum_{i \in I} v_i \iota_\lambda(a_1, \ldots, a_{j_i})$
10.   endfor
11. until $e = \mathit{old}$
```

- Lines 8-10 of the *Extent* procedure compute a better approximation of the extent for automaton states with the current parity n, based on a one-step unfolding of the automaton transition structure.
- Line 7 computes the extent of states with immediately lower parity, relative to the current values for states with parity n, through a recursive call to *Extent* (which may involve further recursive calls).
- Recursive calls to *Extent* update the same copy of e, and only make an additional copy to remember values from the previous step.

8 Related Work and Concluding Remarks

A translation from fixpoint calculi over an arbitrary signature Σ to Σ-*automata* is defined at an abstract level in [1, Section 7]. However, the fixpoint calculi of loc. cit. are intrinsically boolean (their semantics is given in terms of parity games), and thus do not subsume the quantitative logics described here. Our logics share many features with concrete μ-calculi, and our translations from formulas to automata and back resemble existing ones (see e.g. [19, Section 5.3]). Yet, our logics differ in their quantitative semantics, which in particular means that a semantics based on parity games is not available anymore.

Our translation from formulas to automata exposes and exploits the coalgebraic structures present in both models and formulas. Theorems 24 and 28 are thus similar to results in [18], which also provide a coalgebraic perspective on the connection between fixpoint logics and automata. Differently from [18], our construction of the automaton for a formula avoids the use of silent transitions and, more importantly, applies also to *quantitative* logics.

Our results go beyond existing ones, both in the case of probabilistic systems (given our choice of logics) and in the case of weighted systems (where we are not aware of similar translations). In particular, for probabilistic systems, our logics are subtly different from existing ones (they contain sub-convex combinations of formulas but no pure disjunctions or conjunctions), yet are at least as expressive (see Remark 6).

Our parity to Büchi translation (Theorem 32) is novel in two ways: (i) unlike existing translations (e.g. from probabilistic Rabin to probabilistic Büchi automata [2]) which preserve only the *qualitative* language, our translation preserves the *quantitative* language, and (ii) our result comes with a proof which is not a generalisation of any proof we are aware of in the qualitative case. Thus, the jump from a qualitative to a quantitative setting is non-trivial.

Our admittedly trivial model checking algorithm has complexity similar to that of known algorithms for the qualitative case [14], while also being applicable in quantitative settings. Currently this only includes semirings with *finite* strictly ascending/descending chains. Such semirings can e.g. be used to model resource usage with bounded resources. The study of more generally applicable algorithms is left as future work. For this, our recent lattice-based generalisation of the notion of progress measure [13] is expected to prove useful.

Acknowledgments We thank the anonymous reviewers for their insightful comments. S.S. and I.H. are supported by ERATO HASUO Metamathematics for Systems Design Project (No. JPMJER1603), JST, and Grants-in-Aid No. 15KT0012 & 15K11984, JSPS. Part of this work was carried out during a visit of the second author to the University of Southampton, supported by an EPSRC Institutional Sponsorship Award EP/P511407/1.

References

1 A. Arnold and D. Niwiński. *Rudiments of μ-Calculus.* Studies in Logic and the Foundations of Mathematics. North-Holland, 2001.

2 C. Baier, M. Größer, and N. Bertrand. Probabilistic ω-automata. *J. ACM*, 59(1):1, 2012.

3 C. Baier and J.-P. Katoen. *Principles of model checking.* MIT Press, 2008.

4 C. Cîrstea. A coalgebraic approach to linear-time logics. In A. Muscholl, editor, *Foundations of Software Science and Computation Structures, 17th International Conference*, pages 426–440. Springer, 2014.

5 C. Cîrstea. A coalgebraic approach to quantitative linear time logics. *CoRR*, abs/1612.07844, 2016. URL: http://arxiv.org/abs/1612.07844.

6 C. Cîrstea. From branching to linear time, coalgebraically. *Fundamenta Informaticae*, 150:1–28, 2017.

7 R. Cleaveland, M. Klein, and B. Steffen. Faster model checking for the modal μ-calculus. In *Computer Aided Verification, 4th International Workshop*, pages 410–422. Springer, 1993.

8 D. Coumans and B. Jacobs. Scalars, monads, and categories. In *Quantum Physics and Linguistics. A Compositional, Diagrammatic Discourse*, pages 184–216. Oxford Univ. Press, 2013.

9 M. Dam. Fixed points of Büchi automata. In R. Shyamasundar, editor, *Foundations of Software Technology and Theoretical Computer Science, 12th Conference*, pages 39–50. Springer, 1992.

10 B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order (2. ed.).* Cambridge University Press, 2002.

11 B. Farwer. ω-Automata. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata, Logics, and Infinite Games: A Guide to Current Research*, pages 3–20. Springer, 2002.

12 M. J. Fischer and R. E. Ladner. Propositional dynamic logic of regular programs. *Journal of Computer and System Sciences*, 18(2):194 – 211, 1979.

13 I. Hasuo, S. Shimizu, and C. Cîrstea. Lattice-theoretic progress measures and coalgebraic model checking. In *43rd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 718–732, 2016.

14 D. Kirsten. Alternating tree automata and parity games. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata Logics, and Infinite Games: A Guide to Current Research*, pages 153–167. Springer, 2002.

15 I. Meinecke. A weighted μ-calculus on words. In *13th International Conference on Developments in Language Theory*, pages 384–395. Springer, 2009.

16 M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

17 N. Urabe, S. Shimizu, and I. Hasuo. Coalgebraic trace semantics for büchi and parity automata. In J. Desharnais and R. Jagadeesan, editors, *27th International Conference on Concurrency Theory*, volume 59 of *LIPICS*, pages 24:1–24:15, 2016.

18 Y. Venema. Lectures on the modal μ-calculus. Lecture notes, Institute for Logic, Language and Computation, University of Amsterdam, 2012.

19 I. Walukiewicz. Automata and logic, 2001. Notes available from http://www.labri.fr/perso/igw/Papers/igw-eefss01.ps.