Automata, tableaus and a reduction theorem for fixpoint calculi in arbitrary complete lattices

David Janin
LaBRI
Université de Bordeaux I - ENSERB
351 cours de la Libération,
F-33 405 Talence cedex
janin@labri.u-bordeaux.fr

Abstract

Fixpoint expressions built from functional signatures interpreted over arbitrary complete lattices are considered. A generic notion of automaton is defined and shown, by means of a tableau technique, to capture the expressive power of fixpoint expressions. For interpretation over continuous and complete lattices, when, moreover, the meet symbol $\wedge$ commutes in a rough sense with all other functional symbols, it is shown that any closed fixpoint expression is equivalent to a fixpoint expression built without the meet symbol $\wedge$. This result generalizes Muller and Schupp's simulation theorem for alternating automata on the binary tree.

Introduction

The induction principle (least fixpoint construction) is generally sufficient for the specification or analysis of classical input-output programs. However, for many systems such as reactive systems or networks, the co-induction principle (greatest fixpoint construction) is needed as well to express, for instance, correctness properties of their behaviors [4, 13].

This fact has led to various definitions of fixpoint calculus, depending on which model of behaviors one may consider. For instance, Park defined in his landmark paper [13] a fixpoint calculus over finite or infinite sequences of actions. Kozen's propositional μ-calculus [7] was introduced to handle models of behaviors as classes of bisimilar labeled transition systems.

From a mathematical point of view, this study of fixpoint calculi led to some striking results.

It has been known for a long time, say from Kleene's works, that fixpoint calculi have strong connection with automata theory and logic; regular expressions capture both definability in monadic second order logic (MSOL) and recognizability by finite automata.

Today similar connections have been established in many other contexts, e.g. for infinite sequences [13], binary trees [12] and, in a recent paper, for arbitrary tree-like structures [17].

All these results advocate that fixpoint calculi play a fundamental role between logic and automata theory. From logic they inherit a strong mathematical foundation, e.g. inductively defined semantics, and from automata they inherit nice algorithmical properties.

However, despite this series of theoretical successes, almost no general relationship between fixpoint calculi and automata theory has been established so far. One of the main mathematical tools available today to investigate the expressive power of an arbitrary fixpoint calculus is the notion of transfinite approximations from Knaster-Tarski!

In this paper we give automata semantics to fixpoint calculi in a quite general setting : arbitrary complete lattices with monotonic increasing functions.

More precisely, we introduce a notion of an automaton which runs over elements of these lattices. Such a notion of automaton is generic in the sense that it can be instantiated into such or such a classical framework to be seen as the corresponding usual notion of automaton (with accepting states for finite objects and parity or chain conditions for infinite objects [9]). For instance, in the boolean algebra of languages of infinite words, we recover the usual ω-words automata with parity conditions. In arbitrary complete lattices, we show that generic automata are as expressive as fixpoint expressions.

Then, to illustrate the relevance of this approach, we prove a reduction theorem for fixpoint calculi. In particular, together with the previous automata characterization, this theorem gives simple but powerful necessary conditions to

check closure properties of a particular notion of automata and, as a consequence, to relate its expressive power to logical definability.

In fact, this reduction theorem generalizes, to fixpoint calculi over continuous lattices, Muller and Schupp's simulation theorem [11] which appeared in an alternative proof [10] of Rabin's complementation lemma [14].

The paper is organized as follows. In the first part, we recall the usual definitions and properties of fixpoint expressions interpreted in complete lattices with monotonic functions as presented in [2].

In the second part we define generic automata in this general setting. We examine to what extent this notion is related to more classical notions of automata.

In the third part, we state the reduction theorem and give several applications such as the determinization theorem [1] for ω-automata and Muller and Schupp's simulation theorem.

In the fourth part, we extend the notion of tableau defined for the modal mu-calculus [15, 5] to more general fixpoint expressions. This enable us to prove that generic automata capture the expressive power of fixpoint calculi and, in the end, to prove the reduction theorem.

All through the paper, we illustrate most definitions and theorems with a slightly modified version of Park's fixpoint calculus over languages of infinite words [13].

Acknowledgement

This work started during a pleasant meeting of the French-Polish "μ-calculus group" at LaBRI (Bordeaux) in June 1996. I greatly thank all participants for their remarks, criticism and support.

1. Preliminaries

In this paper, we call *functional signature*, or *signature* for short, a set Σ of function symbols equipped with an arity function $\rho : \Sigma \to I\!N$.

Definition 1.1 *Over a signature Σ a* fixpoint algebra *is a complete lattice $\langle M, \vee_M, \wedge_M \rangle$ with bottom and top elements denoted by $\bot_M$ and $\top_M$ together with, for any symbol $f \in \Sigma$, a monotonic increasing function $f_M : M^{\rho(f)} \to M$ called the* interpretation of f in M.

As usual, for any set $E \subseteq M$ we will denote by $\bigvee_M E$ (resp. $\bigwedge_M E$) the least upper bound (resp. the greatest lower bound) of the set E.

[1] the proof of the reduction theorem relies on determinization so this paper can hardly be considered as a new proof of the determinization theorem.

In the sequel, to keep consistent with symbol names, we always assume that for any fixpoint algebra M, any symbol $\bot$, $\top$, $\wedge$ or $\vee$ which appears in Σ is respectively interpreted in M as $\bot_M$, $\top_M$, $\wedge_M$ or $\vee_M$.

We say a fixpoint algebra M is *continuous* when the meet operator $\wedge_M$ on M is continuous, i.e. for any directed sets E and $F \subseteq M$

$$\bigvee E \wedge \bigvee F = \bigvee \{e \wedge f : e \in E \text{ and } f \in F\}$$

Definition 1.2 *Over a signature Σ and a set $\mathcal{X}$ of variable symbols disjoint from Σ, we inductively define the set $\Sigma_\mu(\mathcal{X})$ of* fixpoint formulas*, simply called formulas in the sequel, by the following rules :*

1. *X is a formula for any variable $X \in \mathcal{X}$,*
2. *$f(\alpha_1, \cdots, \alpha_{\rho(f)})$ is a formula for any $f \in \Sigma$ and any formula $\alpha_1, \ldots, \alpha_{\rho(f)}$,*
3. *$\nu X.\alpha$ and $\mu X.\alpha$ are formulas for any $X \in \mathcal{X}$ and any formula α.*

We say a formula $\alpha \in \Sigma_\mu(\mathcal{X})$ is a closed formula *when any variable X occurring in α always occurs in a subformula of the form $\sigma X.t$ with $\sigma = \mu$ or ν. The set of all closed formulas of $\Sigma_\mu(\mathcal{X})$ is denoted by Σ_μ.*

In the sequel, we also denote by $\Sigma(\mathcal{X})$ the set of all formulas built without fixpoint construction.

Definition 1.3 (Formula semantics) *Given a fixpoint algebra M, given a valuation of variables $V : \mathcal{X} \to M$, any formula α is interpreted as an element $[\![\alpha]\!]^M_V$ of M inductively defined by :*

1. $[\![X]\!]^M_V = V(X)$,
2. $[\![f(\alpha_1, \cdots, \alpha_{\rho(f)})]\!]^M_V = f_M([\![\alpha_1]\!]^M_V, \cdots, [\![\alpha_{\rho(f)}]\!]^M_V)$,
3. $[\![\nu X.\alpha]\!]^M_V = \bigvee \left\{ e \in M : e \leq [\![\alpha]\!]^M_{V[e/X]} \right\}$,
4. $[\![\mu X.\alpha]\!]^M_V = \bigwedge \left\{ e \in M : e \geq [\![\alpha]\!]^M_{V[e/X]} \right\}$,

where $V[e/X]$ denotes the valuation defined for any variable $Y \in \mathcal{X}$ by :

$$\begin{cases} V[e/X](Y) = V(Y) \text{ when } Y \neq X \\ V[e/X](X) = e\,. \end{cases}$$

By the Knaster-Tarski theorem, $[\![\mu X.\alpha]\!]^M_V$ and $[\![\nu X.\alpha]\!]^M_V$ are respectively the least and greatest fixpoints of the mapping from M to M defined by $e \mapsto [\![\alpha]\!]^M_{V[e/X]}$.

In particular, $[\![\mu X.X]\!]^M_V = \bot_M$ and $[\![\nu X.X]\!]^M_V = \top_M$. In the sequel, *we will always assume, without increase of expressive power, that both constant symbols $\top$ and $\bot$ belong to Σ.*

Definition 1.4 *Given $\mathcal{C}$ a class of structures, we say formulas α_1 and α_2 are* semantically equivalent w.r.t. *$\mathcal{C}$, which is written $\alpha_1 \simeq_{\mathcal{C}} \alpha_2$ when, for any fixpoint algebras $M \in \mathcal{C}$, any valuation of variables $V : \mathcal{X} \to M$, $[\![\alpha_1]\!]^M_V = [\![\alpha_2]\!]^M_V$.*

When this equivalence holds for arbitrary fixpoint algebras and arbitrary valuations the subscript will be omitted.

In this framework, a *fixpoint calculus* can be seen as a pair $\langle \Sigma, \mathcal{C} \rangle$ for Σ a functional signature and $\mathcal{C}$ a class of Σ-fixpoint algebras.

Example 1.5 Given an alphabet $A = \{a_1, \ldots, a_n\}$, given a signature $\Sigma_1 = \{\bot, \top, \wedge, \vee, S_1\}$ with $\rho(S_1) = n$, we define the (continuous) fixpoint algebra of languages of infinite words (ω-languages for short) on the alphabet A as $M = \langle \mathcal{P}(A^\omega), \cup, \cap \rangle$ with, for any $L_1, \ldots, L_n \in \mathcal{P}(A^\omega)$,

$$S_{1\,M}(L_1, \cdots, L_n) = \bigcup_{i \in [1,n]} a_i.L_i$$

where $a.L = \{a.w : w \in L\}$.

In this fixpoint algebra, one can check that formula

$$\alpha = \nu X(\mu Y(b.X \vee a.Y))$$

denotes the set of all infinite words on the alphabet $\{a, b\}$ with infinitely many b. An equivalent regular expression for this language is $(a^*b)^\omega$.

The following result, from Knaster and Tarski, is a fundamental tool to investigate fixpoint calculus.

Proposition 1.6 (Transfinite approximation) *For any fixpoint algebra M, there exists an ordinal τ_M such that for any formula $\alpha(X)$:*

1. $[\![\nu X.\alpha(X)]\!]^M_V = [\![\nu^{\tau_M} X.\alpha(X)]\!]^M_V$,
2. $[\![\mu X.\alpha(X)]\!]^M_V = [\![\mu^{\tau_M} X.\alpha(X)]\!]^M_V$,

with semantics of $\nu^\tau.\alpha(X)$ and $\mu^\tau.\alpha(X)$ inductively defined by :

1. $[\![\nu^0 X.t]\!]^M_V = \top$
   (resp. $[\![\mu^0 X.t]\!]^M_V = \bot$),
2. $[\![\nu^{\tau+1} X.t]\!]^M_V = [\![t[\nu^\tau X.T/X]]\!]^M_V$
   (resp. $[\![\mu^{\tau+1} X.t]\!]^M_V = [\![t[\mu^\tau X.T/X]]\!]^M_V$),
3. *and, for any limit ordinal τ,*

$$[\![\nu^\tau X.t]\!]^M_V = \bigwedge_{\tau' < \tau} [\![\nu^{\tau'} X.t]\!]^M_V$$

and

$$[\![\mu^\tau X.t]\!]^M_V = \bigvee_{\tau' < \tau} [\![\mu^{\tau'} X.t]\!]^M_V$$

The rest of the section illustrates one simple use of transfinite approximation.

Definition 1.7 *We say a variable X in formula $\alpha(X)$ is* guarded *with respect to function symbol f when every occurrence of X in $\alpha(X)$ is in the scope of function symbols g distinct from f, i.e. it always occurs in subformulas of the form $g(\alpha_1, \cdots, \alpha_{\rho(g)})$ with $g \neq f$.*

A formula α is said guarded w.r.t. f when, for any subformula of α of the form $\sigma X.\alpha_1(X)$, variable X in $\alpha_1(X)$ is guarded w.r.t. f.

Lemma 1.8 (Guardedness) *For any class of fixpoint algebra $\mathcal{C}$, any formula of $\Sigma_\mu(\mathcal{X})$ is equivalent w.r.t. $\mathcal{C}$ to a formula guarded w.r.t. $\vee$.*

Proof: One can easily prove, using transfinite approximations, that, for any formula $\alpha(X)$, any variable $Y \neq X$, the following equivalences hold :

$$\nu X.(X \vee \alpha(X)) \simeq \top \quad (1)$$
$$\mu X.(X \vee \alpha(X)) \simeq \mu X.\alpha(X) \quad (2)$$

and provided $X \neq Y$ with σ denoting μ or ν

$$\sigma X.(Y \vee \alpha(X)) \simeq Y \vee \sigma X.\alpha(X \vee Y) \quad (3)$$

With these equivalences, for any formula α, one can easily build by induction on the structure of α an equivalent formula α' guarded w.r.t. the join operator $\vee$. □

Remark: Dual arguments hold for $\wedge$ henceforth one may always assume that all formulas one consider are guarded w.r.t. both $\wedge$ and $\vee$. Technically, we do not need such a restriction. It may however help intuition as shown below.

2. Generic automata

Given a functional signature Σ, let $\mathcal{F}_\Sigma$ be the set of (syntactic) functions one can build from signature Σ and composition. More precisely, using lambda notation, we define $\mathcal{F}_\Sigma$ as the set of all classes of functions (equivalent under consistent renaming of bound variables) of the form $G = \lambda X_1 . \cdots . \lambda X_n . \alpha(X_1, \cdots, X_n)$ for any formula $\alpha(X_1, \cdots, X_n) \in \Sigma(\mathcal{X})$ built without fixpoint construction with all free variables of α taken among $X_1, X_2, \ldots, X_n$.

Notions of arity and interpretation are extended to $\mathcal{F}_\Sigma$ in a straightforward way. In the sequel, in order to avoid heavy notation, set $\mathcal{F}_\Sigma$ is considered as a functional signature and the previous notation applies. In particular, for any function $G \in \mathcal{F}_\Sigma$ (seen as a functional symbol), any fixpoint algebra M over signature Σ, we denote by G_M the interpretation of G in M. We also denote by ϵ the particular symbol of $\mathcal{F}_\Sigma$ which is always interpreted as the identity.

Definition 2.1 *A* generic automaton $\mathcal{A}$ *over signature* Σ *is a tuple* $\mathcal{A} = \langle Q, q_0, \delta, \tau, \Omega \rangle$ *with :*

1. *a finite set of states* Q,
2. *an initial state* q_0,
3. *a transition function* $\delta : Q \to Q^*$,
4. *a type function* $\tau : Q \to \mathcal{F}_\Sigma$,
5. *an index function* $\Omega : Q \to \mathbb{N}$,

such that, for any state $q \in Q$, *the length of* $\delta(q)$ *equals the arity of the functional type* $\tau(q) \in \mathcal{F}_\Sigma$ *of state* q.

Definition 2.2 *Given a generic automaton* $\mathcal{A}$, *given a fixpoint algebra* M *for signature* Σ, *given a point* $e_0 \in M$, *a* run *of automata* $\mathcal{A}$ *on* e_0 *is a (possibly infinite) tree* T *labeled by pairs* $(e, q) \in M \times Q$ *such that :*

1. *the root* n_0 *of tree* T *is labeled by* (e_0, q_0),
2. *from any node* n *of tree* T *labeled by some pair* (e, q) *with* $e \neq \bot_M$ *one of the following rules applies :*
   (a) $\bigvee$**-move** *: there exists a subset* $\{e_i : i \in \mathcal{I}\}$ *of* M *such that* $e = \bigvee_{i \in \mathcal{I}} e_i$ *and node* n *has exactly one son* n_i *labeled by pair* (e_i, q) *for each index* $i \in \mathcal{I}$,
   (b) G**-move** *: when* $\tau(q) = G \in \mathcal{F}_\Sigma$, *given* $k = \rho(G)$ *the arity of* G, *noting* $q_1 . \cdots . q_k = \delta(q)$ *the sequence of successors of* q *:*
   (1) if $k > 0$ *then there exist* $e_1, \ldots, e_k \in M$ *such that* $e = G_M(e_1, \cdots, e_k)$ *and node* n *has exactly* k *sons* $n_1, \ldots, n_k$ *respectively labeled by pairs* (e_i, q_i) *for* $i \in [1, \cdots, k]$,
   (2) if $k = 0$ *then* $e \leq_M G_M$,
3. *on any infinite path of tree* T, *infinitely many* G*-moves occur.*

We say that run T *is an* accepting run *when, for any infinite path* $n_0, n_1, \ldots$ *of nodes of* T *labeled by pairs* (e_0, q_0), (e_1, q_1), ... , *the smallest index* $\Omega(q_i)$ *which appears infinitely often on this path is even (such a condition is called a parity or a chain condition [9]). In this case, we say that* e *is accepted by* $\mathcal{A}$ *and we denote by* $L_M(\mathcal{A})$ *the set of elements of* M *accepted by* $\mathcal{A}$.

Remark: When $G = \epsilon$ (i.e. the identity symbol) a G-move can be seen as firing an ϵ-transition in classical automata theory. Indeed, when such a move occurs from a node n labeled by (e, q) then node n has exactly one son labeled by $(e, \delta(q))$, i.e. no further reading of the input is made during the move. The next proposition shows, as in the usual case, that these ϵ-states are useless in terms of expressive power.

Proposition 2.3 *For any automaton* $\mathcal{A}$ *there exists an automaton* $\mathcal{A}_1$ *with no states of type* ϵ *equivalent to* $\mathcal{A}$ *in the following sense : for any fixpoint algebra* M, $L_M(\mathcal{A}) = L_M(\mathcal{A}_1)$.

Proof: Given an automaton $\mathcal{A} = \langle Q, q_0, \delta, \tau, \Omega \rangle$, for any state $q \in Q$ of type ϵ, let $s_q = q_0 . q_1 . \cdots$ be the longest (possibly infinite but unique) sequence of states such that, $q = q_0$ and for any relevant i, $\delta(q_i) = q_{i+1}$ with $\tau(q_i) = \epsilon$, i.e. in the case $s_q = q_0 . \cdots . q_n$ then $\delta(q_i) \neq \epsilon$ only when $i = n$. Automaton $\mathcal{A}_1$ is built from automaton $\mathcal{A}$ replacing any state $q \in Q$ of type ϵ by the sequence s_q, extending parity, type and successor functions to these sequences as follows :

1. for any finite sequence $s_q = q_0 . \ldots . q_n$,
   $$\Omega_1(s_q) = min\{\Omega(q_i) : i \in [0, \cdots, n]\}$$
   with $\tau_1(s_q) = \tau(q_n)$ and $\delta_1(s_q) = \delta(q_n)$,
2. for any infinite sequence $s_q = q_0 . q_1 . \cdots$,
   $$\Omega_1(s_q) = min\{\Omega(q_i) : i \in \mathbb{N}\}$$
   with $\tau_1(s_q) = \top$ when $\Omega_1(s_q)$ is even, $\tau_1(s_q) = \bot$ otherwise and $\delta(q)$ equals to the empty word.

From the construction of $\mathcal{A}_1$ one can easily check that, for any fixpoint algebra M, $L_M(\mathcal{A}) = L_M(\mathcal{A}_1)$, accepting runs on automaton $\mathcal{A}$ immediately inducing accepting runs on automaton $\mathcal{A}_1$ and vice versa. □

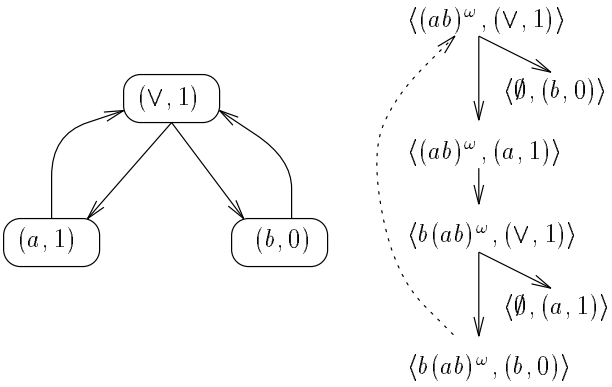

Figure 1. An automaton and a run.

Example 2.4 In Figure 1 above, an automaton, which accepts any subset of $(a^*b)^\omega$ on the algebra of ω-languages, illustrates the previous definitions. In this figure, any state q is labeled by its type and its parity index, i.e. pair $(\tau(q), \Omega(q))$ where, as in Example 1.5, function a and resp. function b stand for the mapping $L \mapsto a.L$ and resp. the mapping $L \mapsto b.L$. The initial state is labeled by $(\vee, 0)$. In addition to the automaton on the left, an accepting run on language $(ab)^\omega$ is given on the right.

Remark: The previous example illustrates two major aspects of generic automata.
(1) In the classical definition of non deterministic automata firing a transition is implicitly preceded by the choice of the transition to fire; union of languages is implicitly modelized by non determinism. In the present approach these two successive steps become explicit; non determinism is explicitly modeled by states of type $\vee$. One reason for this trick is that the joint operator $\vee$ can then be treated like any other operator. In particular, it helps to realize that determinization of ω-automaton is a particular instance of Muller and Schupp simulation (see Example 3.3 of next section).
(2) In this example no $\bigvee$-move occurs; none is needed. This is not the case in general. For instance, in any accepting run of the previous automaton over language $(a^*b)^\omega$ a $\bigvee$-move must occur (otherwise, by Koenig's Lemma, there would exist an accepting run over a^ω).

Proposition 2.5 *For any automaton $\mathcal{A}$, any fixpoint algebra M:*

1. *$L_M(\mathcal{A})$ is closed under $\vee$,*
2. *$\bigvee L_M(\mathcal{A}) \in L_M(\mathcal{A})$.*

Proof: Obvious, applying a $\bigvee$-move at the beginning of the accepting run one intends to build. □

The following definition and theorem give a straightforward condition for $L_M(\mathcal{A})$ to be downward closed.

Definition 2.6 *We say the* decomposability property *holds on a fixpoint algebra M over a signature Σ when, for any $f \in \Sigma$ with $\rho(f) \neq 0$, for any $d \in M$, any e_1, ..., $e_{\rho(f)} \in M$ if $d \leq f_M(e_1, \cdots, e_{\rho(f)})$ then there exists d_1, ..., $d_{\rho(f)} \in M$ such that, for any i, $d_i \leq e_i$ and $d = f_M(d_1, \cdots, d_{\rho(f)})$ (equivalently the inverse image $f_M^{-1}(I)$ of any ideal I of M is an ideal of $M^{\rho}(f)$.).*

Remark: In particular, when decomposability holds on M for any $f \in \Sigma$ with $\rho(f) \neq 0$ the function f_M is strict. Note that in the modal μ-calculus the universal modality $X \mapsto [a]X$ is not strict. In [5], to translate formulas into automata, an equivalent signature of strict functions was introduced to remedy this.

Theorem 2.7 *For any automaton $\mathcal{A}$, any continuous algebra M where decomposability holds for any element of M, $L_M(\mathcal{A})$ is downward closed, i.e. for any e and $f \in M$, if $e \leq_M f$ with $f \in L_M(\mathcal{A})$ then $e \in L_M(\mathcal{A})$.*

Proof: Given e and $f \in M$ with $e \leq f$ and an accepting run T_f of automaton $\mathcal{A}$ on f one can build from T_f, by induction on the depth of nodes, an accepting run T_e of automaton $\mathcal{A}$ on e. Indeed, decomposability ensures easy construction steps for ϵ-moves and G-moves. When a $\bigvee$-move is applied in T_f from a node n labeled by (f_n, q_n) with successors labeled by (f_i, q_i) for any $i \in \mathcal{I}$, then a $\bigvee$-move occurs also in T_e from node n labeled by (e_n, q_n) with successors labeled by $(f_i \wedge e_n, q_i)$ for $i \in \mathcal{I}$. Here, continuity is required since the definition of runs implies $e_n = \bigvee_{i\in\mathcal{I}} e_n \wedge f_i$ with $e_n \leq_M \bigvee_{i\in\mathcal{I}} f_i$. The construction ends in any node n such that $e_n = \bot$. □

Remark: In particular, when M is an atomic boolean algebra, for any automata $\mathcal{A}$ where decomposability holds, the language $L_M(\mathcal{A})$ is characterized by its projection on the atoms, i.e. the set of all atoms accepted by $\mathcal{A}$. We almost recover here the usual settings of automata theory where acceptance is only defined on atoms, e.g. words for usual automata.

Lemma 4.10 and Lemma 4.12 below show the equivalence, in terms of expressive power, of generic automata and fixpoint expression interpreted in complete lattices.

Remark: Strictly speaking, we need one more restriction in our definition of automata to recover, for instance over words, usual definitions. Namely, there should be no loops from states of type $\vee$, i.e. states modeling non determinism, since such loops cannot occur with usual definitions of automata where non determinism is modeled implicitly in the definition of the transition function. In terms of fixpoint expression such a restriction is captured by the notion of guardedness w.r.t. the join operator $\vee$. Lemma 1.8 and Lemma 4.10 show that, indeed, such a restriction has no effect in terms of expressive power.

3. The reduction theorem

In this section we present the reduction theorem and give several applications. We shall use bold-math letters to denote tuples (such as $\boldsymbol{x}$ denoting $(x_1, x_2, \ldots, x_n)$).

Definition 3.1 *Given a class of fixpoint algebras $\mathcal{C}$, we say that the meet operator $\wedge$* commutes *with Σ on $\mathcal{C}$ when, for any finite multiset $\{f_i\}_{i\in[1,n]}$ of functional symbols of $\Sigma - \{\wedge\}$ there exists a function $G \in \mathcal{F}_\Sigma$ built without the symbol $\wedge$ such that an equation of the form:*

$$\bigwedge_{i\in[1,n]} \{f_i(\boldsymbol{x}_i)\} = G(\bigwedge \boldsymbol{y}_1, \cdots, \bigwedge \boldsymbol{y}_m)$$

holds on $\mathcal{C}$, where :

1. *$\boldsymbol{x}_i$s are vectors of distinct variables of the appropriate length,*
2. *$\boldsymbol{y}_j$s are vectors of distinct variables taken among those appearing in $\boldsymbol{x}_i$s,*
3. *$\bigwedge \boldsymbol{y}_j$s denote the g.l.b. applied to the set of all variables occurring in $\boldsymbol{y}_j$.*

In the sequel, such an equation, oriented from left to right, is called the [2] commutation rule *associated with* $\{f_i\}_{i\in[1,n]}$. *In particular, when* $n = 1$, *we always consider the degenerated commutation rule* $f(\boldsymbol{x}) = f(\boldsymbol{x})$.

Theorem 3.2 (Reduction) *When the meet operator* $\wedge$ *commutes with* Σ *on* $\mathcal{C}$ *any closed fixpoint formula* α *is equivalent over* $\mathcal{C}$ *to a formula* $\widehat{\alpha}$ *built without the symbol* $\wedge$.

Proof: As described in next part, one can build a regular tableau for α then, applying Lemma 4.10, translate it into an equivalent generic automaton on the signature $\Sigma - \{\wedge\}$ which itself, applying Lemma 4.12, can be translated into an equivalent fixpoint formula on the same signature. □

As an illustration, we give, in the rest of this section, some applications of this reduction theorem.

Example 3.3 Continuing example 1.5 of ω-languages, we have :

Observation 3.3.1 (Unary simulation) *Any fixpoint formula* $\alpha \in \Sigma_{1\mu}$ *is equivalent over* ω*-languages to a fixpoint formula* $\widehat{\alpha}$ *without conjunction.*

Proof: It is almost immediate that for ω-languages, $\wedge$ fulfills the hypothesis of theorem 3.2, e.g.

$$S_1(\boldsymbol{x}) \wedge S_1(\boldsymbol{y}) = S_1(\boldsymbol{x} \wedge \boldsymbol{y})$$

where $\boldsymbol{x} \wedge \boldsymbol{y}$ denotes $(x_1 \wedge y_1, \cdots, x_n \wedge y_n)$ when $\boldsymbol{x} = (x_1, \cdots, x_n)$ and $\boldsymbol{y} = (y_1, \cdots, y_n)$. □

Observation 3.3.2 (Determinization) *Any fixpoint formula* $\alpha \in \Sigma_{1\mu}$ *built without symbol* $\wedge$ *is equivalent over* ω*-languages to a fixpoint formula* $\widehat{\alpha}$ *built without either* $\wedge$ *or* $\vee$.

Proof: Here again, over signature $\Sigma_1 - \{\wedge\}$, function $\vee$ fulfills the hypothesis of the dual version of Theorem 3.2, e.g.

$$S_1(\boldsymbol{x}) \vee S_1(\boldsymbol{y}) = S_1(\boldsymbol{x} \vee \boldsymbol{y})$$

with a similar notation. Hence, starting with all formulas of $\Sigma_{1\mu}$ built without $\wedge$, the reduction theorem applies in its dual version giving us the result. □

In other words, applying the theorem twice, we have proved that any formula $\alpha \in \Sigma_{1\mu}$ is equivalent over ω-languages to a formula $\widehat{\alpha}$ built without either the symbol $\wedge$ or the symbol $\vee$.

As a consequence, since the class of languages definable by fixpoint expressions over the signature Σ_1 is obviously closed under complement, union and intersection we obtain the usual closure properties of deterministic languages.

[2] we assume that one such rule is selected for each multiset of symbols of $\Sigma - \{\wedge\}$

Remark: We do not obtain here a new proof of the determinization theorem since we use this result in the proof of the reduction theorem.

Example 3.4 For the binary case things are similar. Let $\Sigma_2 = \{\top, \bot, \vee, \wedge, S_2\}$ be a functional signature with the usual arity for usual symbols and arity $2n$ for symbol S_2. Given $A = \{a_1, \cdots, a_n\}$ an alphabet, let M be the Σ_2-fixpoint algebra defined as $M = \langle \mathcal{P}(\{l, r\}^* \to A), \cup, \cap \rangle$ for $\{l, r\}^* \to A$ the set of all total functions which map any finite word over the alphabet $\{l, r\}$ to a letter in A, interpreting S_2 by :

$$S_{2M}(L_1, L_2, \cdots, L_{2n-1}, L_{2n}) = \bigcup_{i\in[1,n]} a_i(L_{2i-1}, L_{2i})$$

noting for any L_1 and $L_2 \in \mathcal{P}(\{l, r\}^* \to A)$ and any $a \in A$,

$$\begin{aligned} a(L_1, L_2) \quad = \quad & \{f : \exists f_1 \in L_1, f_2 \in L_2, f(\epsilon) = a \\ & \wedge \forall w \in \{l, r\}^*, f(l.w) = f_1(w) \\ & \wedge f(r.w) = f_2(w)\} \end{aligned}$$

Observation 3.4.1 (Binary simulation) *Any formula* $\alpha \in \Sigma_{2\mu}$ *is equivalent to a formula* $\widehat{\alpha}$ *built without conjunction symbols.*

Proof: As before theorem 3.2 applies. □

This result is a reformulation, in terms of fixpoint, of Muller and Schupp's simulation theorem [10]. Translated back into automata, it immediately shows that non alternating parity automata on the binary tree are closed under complementation: an essential step in Rabin's proof of the decidability of S2S [14].

Remark: Another application is Walukiewicz's work on tree-shaped structures [17]. The functional signature Σ is given by all possible *basic formulas* as defined in this approach, the interpretation of these function symbols being the underlying model theoretical interpretation. Checking that the symbol $\wedge$ commutes in the sense above with the signature Σ is quite obvious and thus, the reduction theorem applies. In the case of tree-shaped structures this solves the difficult part of the proofs. In particular, in the case of trees (of arbitrary degrees) the reduction theorem enables us to obtain the (more restrictive) automata characterization used in [6] to characterize the expressive power of the modal mu-calculus.

4. Tableaus for fixpoint formulas

In this section we extend the notion of tableau used for instance in [5] in order to prove the automata characterization stated in Lemmas 4.12 and 4.10 and the reduction theorem stated in Theorem 3.2. We assume $\mathcal{C}$ is a given class

of continuous fixpoint algebras over the signature Σ where $\wedge$ commutes with Σ (see Definition 3.1).

Remark: To translate formulas into automata and automata into formulas, the hypothesis can be weakened. More precisely, all results below still hold dropping the continuity hypothesis and renaming $\wedge$ in Σ into some fresh symbol name in order to avoid all the machinery needed to prove the reduction theorem. In this case, $\mathcal{C}$ can be any class of fixpoint algebra on signature Σ.

Definition 4.1 *We call a formula* well named *iff every variable is bound at most once in the formula and free variables are distinct from bound variables. For a variable X bound in a well named formula α there exists a unique sub-formula of α of the form $\sigma X.\beta(X)$, from now on called the* binding definition of X in α *and denoted by $\mathcal{D}_\alpha(X)$.*

*We call X a μ-*variable *when $\sigma = \mu$, otherwise we call X a ν-*variable*.*

Remark: Every formula is equivalent to a well-named one which can be obtained by some consistent renaming of bound variables.

Definition 4.2 *Given a formula α we define the* dependency order *over the bound variables of α, denoted $\leq_\alpha$, as the least partial order relation such that if X occurs free in $\mathcal{D}_\alpha(Y)$ then $X \leq_\alpha Y$.*

Remark: For instance, if $\alpha = \mu X.(\nu Y.f(Y)) \vee g(X)$ then variables X and Y are incomparable in $\leq_\alpha$ ordering. This partial order leads to the alternation depth defined by Niwinski [12], a notion which, in the context of the modal mu-calculus, has been shown to lead to an infinite hierarchy [3].

Definition 4.3 (Tableaus) *Given a closed well-name fixpoint formula α a tableau for a formula α is a tuple*

$$\mathcal{G} = \langle Q, q_0, \delta, L \rangle$$

with $\langle Q, q_0, \delta \rangle$ the graph of a generic automaton and $L : N \to \mathcal{P}(\Sigma_\mu(\mathcal{X}))$ a labeling function, such that the initial state q_0 is labeled by $L(q_0) = \{\alpha\}$ and, for any state $q \in Q$ with $\delta(q) = r_1.r_2.\cdots.r_n$,

$$\frac{\{L(r_1)\} \quad \cdots \quad \{L(r_n)\}}{L(q)}$$

is an instance of one of the following transition rules :

- **$\wedge$-rule :**

$$(\wedge\text{-rule}) \ \frac{\{\alpha_1, \alpha_2\} \cup \Gamma}{\{\alpha_1 \wedge \alpha_2\} \cup \Gamma}$$

- **Fixpoint rule :**

$$(\sigma\text{-rule}) \ \frac{\{\alpha_1(X)\} \cup \Gamma}{\{\sigma X.\alpha_1(X)\} \cup \Gamma}$$

with $\sigma = \mu$ or ν

- **Regeneration rule :**

$$(\text{reg-rule}) \ \frac{\{\alpha_1(X)\} \cup \Gamma}{\{X\} \cup \Gamma}$$

provided $\mathcal{D}_\alpha(X) = \sigma X.\alpha_1(X)$,

- **Functional rules :**

$$(G\text{-rule}) \ \frac{\{\boldsymbol{\beta}_j\}_{j\in[1,m]}}{\{f_1(\boldsymbol{\alpha}_1), f_2(\boldsymbol{\alpha}_2), \cdots, f_n(\boldsymbol{\alpha}_n)\}}$$

provided $f_1(\boldsymbol{x}_1) \wedge \cdots \wedge f_n(\boldsymbol{x}_n) = G(\bigwedge \boldsymbol{y}_1, \cdots, \bigwedge \boldsymbol{y}_m)$ is the commutation rule associated with multiset $\{f_i\}_{i\in[1,n]}$ (see definition 3.1) and, considering the mapping that maps any variable of a vector $\boldsymbol{x}_i$ to the corresponding formula in the corresponding vector $\boldsymbol{\alpha}_i$, vectors of formulas $\boldsymbol{\beta}_j$s are built from vectors of variable $\boldsymbol{y}_j$s applying this mapping.

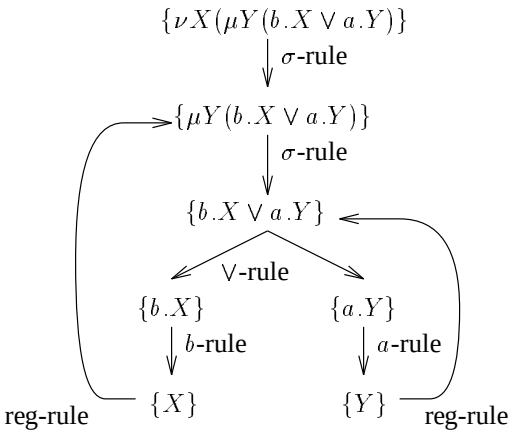

Figure 2. A tableau for α without $\wedge$.

Example 4.4 As an illustration, a tableau for formula $\alpha = \nu X.(\mu Y.(b.X \vee a.Y))$ (already presented in Example 1.5) is given in Figure 2 above. Since no symbol $\wedge$ occurs in α, the states of the tableau are labeled by singletons.

Remark: For technical reasons, it is more convenient to allow infinite graphs as well. Notice however that for any closed formula α, any tableau $\mathcal{G} = \langle Q, q_0, \delta, L \rangle$, the set $L(Q)$ is finite, i.e. only finitely many labels can be used. Henceforth, for any formula α, there exists at least one regular tableau for formula α and, henceforth, there exists also one finite tableau for α.

Definition 4.5 *Given a tableau $\mathcal{G} = \langle Q, q_0, \delta, L\rangle$ for formula α, an* infinite path *on $\mathcal{G}$ is an infinite sequence*

$$\mathcal{P} = q_0, q_1, \cdots$$

of states of Q such that, for any index i, q_{i+1} occurs in $\delta(q_i)$.

For any infinite path $\mathcal{P}$ on $\mathcal{G}$, a infinite trace *on $\mathcal{P}$ is a function $\mathbf{tr} : \mathbb{N} \to \Sigma_\mu(\mathcal{X})$ such that, for any $i \in \mathbb{N}$, $\mathbf{tr}(i) \in L(q_i)$ and $\mathbf{tr}(i+1)$ is related to $\mathbf{tr}(i)$ according to one of the following rules.*

1. *When the fixpoint rule or regeneration rule is applied from state q_i to state q_{i+1} then $\mathbf{tr}(i+1)$ equals $\mathbf{tr}(i)$ possibly modified by the application of the rule,*
2. *When the $\wedge$-rule is applied from state q_i to q_{i+1} then :*
   (a) *if this application does not modify $\mathbf{tr}(i)$ then $\mathbf{tr}(i) = \mathbf{tr}(i+1)$,*
   (b) *otherwise $\mathbf{tr}(i) = \alpha_1 \wedge \alpha_2$ with α_1 and $\alpha_2 \in L(q_{i+1})$ then $\mathbf{tr}(i+1)$ equals α_1 or α_2,*
3. *When the G-rule is applied from state q_i to state q_{i+1} with commutation rule*

$$f_1(\boldsymbol{x}_1) \wedge \cdots \wedge f_n(\boldsymbol{x}_n) = G(\bigwedge \boldsymbol{y}_1, \cdots, \bigwedge \boldsymbol{y}_m)$$

with $\mathbf{tr}(i) = f_i(\boldsymbol{\alpha}_i)$ for some $i \in [1,m]$ and $L(q_{i+1}) = \{\boldsymbol{\beta}_j\}$ for some $j \in [1,n]$ then $\mathbf{tr}(i+1)$ equals to the kth component of $\boldsymbol{\beta}_j$ for some k such that the kth variable of $\boldsymbol{y}_j$ occurs in $\boldsymbol{x}_i$.

Proposition 4.6 *On any infinite path $\mathcal{P}$ of $\mathcal{G}$, for any infinite trace $\mathbf{tr} \in \mathcal{T}r(\mathcal{P})$ there exists a variable X which is the smallest variable regenerated infinitely often on $\mathbf{tr}$.*

*We say that trace $\mathbf{tr}$ is a μ-*trace *when this smallest variable is a μ-variable (see Definition 4.1), otherwise, we say that trace $\mathbf{tr}$ is a ν-*trace*.*

Definition 4.7 *Given a tableau $\mathcal{G} = \langle Q, q_0, \delta, L\rangle$ for a formula α we define a* marking *$\mathcal{M}$ of the tableau $\mathcal{G}$ (with respect to an algebra M) by $e \in M$ as a run in the sense of Definition 2.2 on the underlying graph of automaton $\langle Q, q_0, \delta, \tau\rangle$ taking, for any $q \in Q$, $\tau(q) = G$ when the G-rule applies on state q in the construction of $\mathcal{G}$, $\tau(q) = \epsilon$ otherwise.*

Remark: Notice here that any finite path of a marking either ends in a node labeled $(\bot_M, q)$ for arbitrary state q or ends in a node labeled by (e, q) with $e \leq_M f_M$ and some constant functional symbol $f \in \Sigma$ such that $L(q) = \{f\}$.

Definition 4.8 *A marking $\mathcal{M} = \langle \delta, L\rangle$ is* consistent *when for any infinite path of $\mathcal{M}$ labeled by*

$$(e, q_0), (e_1, q_1), (e_2, q_2) \cdots$$

all infinite traces on the tableau path $q_0, q_1, \ldots$ are ν-traces.

Theorem 4.9 *For any closed formula α, any continuous fixed point algebra $M \in \mathcal{C}$, for any tableau $\mathcal{G}$ for α, $[\![\alpha]\!]^M$ is the greatest element of M which admits a consistent marking of tableau $\mathcal{G}$.*

Proof: We first build by induction a consistent marking of $\mathcal{G}$. More precisely, for any node n of the marking we build a closed formula α_n such that n is labeled by pair $([\![\alpha_n]\!]^M, q_n)$ for some tableau state q with formula α_n obtained from formula $\bigwedge L(q)$ inductively replacing, in an appropriate order to obtain a closed formula :

- any free ν-variable X by its binding definition $\nu X.\mathcal{D}_\alpha(X)$,
- any free μ-variable Y by some transfinite approximation $\mu^\tau X.\mathcal{D}_\alpha(X)$ of the fixpoint subformula of α binding X.

The basic idea of the induction step is to apply to formula α_n the tableau rule applied in state q_n in order to build formulas α_m one for each successor m of node n.

1. When the G-rule is applied in state q_n then a G-move occurs. Given $r_1, \ldots, r_k$ the k successors of state q_n in $\mathcal{G}$ with $k = \rho(G)$, by the induction hypothesis and applying the appropriate commutation rule, formula α_n is equivalent to a formula of $G(\beta_1, \cdots, \beta_k)$ with, for any $i \in [1,k]$ formula β_i obtained from the formula $\bigwedge L(r_i)$ as above and we label m_i the ith successor of node n by pair $([\![\beta_i]\!]^M, r_i)$.
2. When the $\wedge$-rule is applied in state q_n then an ϵ-move occurs on the marking we build with m the unique successor of n and we put $\alpha_m = \alpha_n$.
3. When the fixpoint rule is applied in state q_n on a formula $\sigma X.\mathcal{D}_\alpha(X) \in L(q_n)$ with $\sigma = \mu$ or ν then α_n is of the form $(\sigma X.\beta_1(X)) \wedge \beta_2$ and node n has exactly one son m labeled by:
   (a) $\alpha_m = \beta_1[\nu X.\beta_1(X)/X] \wedge \beta_2$ when $\sigma = \nu$,
   (b) $\alpha_m = \beta_1[\mu^\tau X.\beta_1(X)/X] \wedge \beta_2/X]$ when $\sigma = \mu$, where τ is the smallest ordinal such that $[\![\alpha_n]\!]^M = [\![\alpha_m]\!]^M$.
4. When the regeneration rule applies in state q_n on $X \in L(q_n)$ then, since rule 3 above has necessarily been applied before reaching such a state, one of the following conditions is satisfied :
   (a) α_n is of the form $(\nu X.\beta_1(X)) \wedge \beta_2$ then n has exactly one son m labeled by $\alpha_m = \beta_1[\nu X.\beta_1(x)/X] \wedge \beta_2$,

(b) α_n is of the form $(\mu^\tau X.\beta_1(X)) \wedge \beta_2$ for some ordinal τ: in the case τ is a successor ordinal $\tau' + 1$ node n has exactly one son m labeled by $\alpha_m = \beta_1[\mu^{\tau'} X.\beta_1(X)/X] \wedge \beta_2$, otherwise τ is a limit ordinal hence a $\bigvee$-move occurs and, for each ordinal $\tau' < \tau$, node n as one son $m_{\tau'}$ labeled by $\beta_1[\mu^{\tau'} X.\beta_1/X] \wedge \beta_2$; in this last case, continuity ensures such a move is valid.

On the marking obtained there cannot be an infinite path with a μ-trace. Otherwise on this trace, after some point, the smallest variable ever regenerated would be a μ-variable. Given n_1, n_2, ..., the sequence of nodes, after this particular point, where a regeneration of X occurs, one can check that for any $i \in I\!N$, α_{n_i} is of the form $\mu^{\tau_i} X.\beta_{1,i}(X) \wedge \beta_{2,i}$ with $t_i > \tau_{i+1}$ which is absurd since ordinals are well-founded.

Conversely, we assume that some element $e_0 \in M$ admits a consistent marking of tableau $\mathcal{G}$ with $e_0 \not\leq [\![\alpha]\!]^M$ and show that this leads to a contradiction.

For this, we first show, by induction, that there exists an infinite path on $\mathcal{M}$ labeled by a sequence of pairs $\{(e_i, q_i)\}_{i \in I\!N}$ such that, denoting by $\mathcal{P}$ the underlying infinite tableau path, no μ-trace occurs on the path $\mathcal{P}$, and there exists a ν-trace **tr** on $\mathcal{P}$ such that, for any state q_i occurring on $\mathcal{P}$, $e_i \not\leq [\![\alpha_i]\!]^M$ for some closed formula α_i obtained from formula $\mathbf{tr}(i)$ with a construction dual from the one above.

The initial induction step for this construction comes from the root n_0 of the marking labeled by pairs (e, q_0) where we take n_0 as the initial node of the path we build and we put $\mathbf{tr}(0) = \alpha_0 = \alpha$.

Let us assume now we have build this path in $\mathcal{M}$ up to node n_i labeled by (e_i, q_i) with $e_i \not\leq [\![\alpha_i]\!]^M$. The construction proceeds then according to one of the following rules.

1. When a $\bigvee$-move occurs in node n_i: given $(f_j, r_j)_{j \in \mathcal{J}}$ the label of the successor nodes of node n_i one have

$$e = \bigvee_{j \in \mathcal{J}} f_j \not\leq [\![\alpha_i]\!]^M$$

Hence there exists at least one $j \in \mathcal{J}$ such that $e_j \not\leq [\![\alpha_i]\!]^M$ and we put $e_{i+1} = f_j$ with $q_{i+1} = q_i$ and $\alpha_{i+1} = \alpha_i$, taking for n_{i+1} the corresponding successor of n_i.

2. When an ϵ-move occurs in node n_i:

   (a) if the $\wedge$-rule is not applied from state q_i or if it is applied on a formula distinct from $\mathbf{tr}(i)$: the construction of n_{i+1}, q_{i+1}, $\mathbf{tr}(i+1)$ and α_{i+1} is obvious (perhaps applying a rule dual from Rule 3 or Rule 4 above in order to build α_{i+1}).

   (b) if the $\wedge$-rule is applied from state q_i on formula $\mathbf{tr}(i)$ of the form $\beta_1 \wedge \beta_2$: taking for q_{i+1} the unique successor of state q_i, formula α_i is, by induction hypothesis, of the form $\beta'_1 \wedge \beta'_2$ with b'_1 and b'_2 related to β_1 and β_2; in other words

$$e_i \not\leq [\![\alpha_i]\!]^M = [\![\beta'_1 \wedge \beta'_2]\!]^M$$

   hence there exists $j \in \{1, 2\}$ such that $e_i \not\leq [\![\beta'_j]\!]^M$ and we put $e_{i+1} = e_i$ with $\mathbf{tr}(i+1) = \beta_j$ and $\alpha_{i+1} = \beta'_j$.

3. Otherwise a G-rule is applied on n_i: given $L(n_i) = \{f_k(\boldsymbol{\beta}_k) : k \in [1, m]\}$ and applying the induction hypothesis we have $e_i \not\leq [\![\alpha_i]\!]^M$ with

$$\alpha_i = f_p(\boldsymbol{\beta}'_p)$$

   for some $p \in [1, m]$ and some vector of formulas $\boldsymbol{\beta}'_p$ built from $\boldsymbol{\beta}_p$ as above. From definition of G-moves we also have

$$e_i = G_M(d_1, \cdots, d_n)$$

   where d_ls occur on the labels of the sons of node n_i. Given then the commutation rule which applies in this node:

$$\bigwedge_k f_k(\boldsymbol{x}_k) = G(\bigwedge \boldsymbol{y}_1, \cdots, \bigwedge \boldsymbol{y}_n)$$

   we instantiate vector $\boldsymbol{x}_k$ above by $\boldsymbol{\beta}'_k$ when $k = p$ and by $(\top_M, \cdots, \top_M)$ otherwise, and we obtain an equivalence of the form

$$f_p(\boldsymbol{\beta}'_p) \wedge d =_M G(\bigwedge \boldsymbol{\beta}''_1, \cdots, \bigwedge \boldsymbol{\beta}''_n)$$

   From the fact that $e_i \not\leq [\![f_p(\boldsymbol{\beta}'_p)]\!]^M$ we immediately deduce

$$e_i = G_M(d_1, \cdots, d_n) \not\leq [\![G(\bigwedge \boldsymbol{\beta}''_1, \cdots, \bigwedge \boldsymbol{\beta}''_n)]\!]^M$$

   from which we conclude there exists one $l \in [1, n]$ and then one formula β'' occurring in $\boldsymbol{\beta}''_l$ (and by construction occurring in $\boldsymbol{\beta}'_p$) such that $d_l \not\leq [\![\beta'']\!]^M$. From this we take for n_{i+1} the lth successor of node n_i putting $e_{i+1} = d_l$ and $\alpha_{i+1} = \beta''$ and we take for $\mathbf{tr}(i+1)$ the corresponding formula in $\boldsymbol{\beta}_p$; notice here that $\boldsymbol{\beta}_l$ can indeed be chosen distinct from $(\top_M, \cdots, \top_M)$ otherwise we immediately conclude $e \not\leq \top_M$ which is absurd.

Such a construction is infinite. Otherwise we end in a node n_i of $\mathcal{M}$ which is a leaf and then, since $\mathcal{M}$ is a marking, either $e_i = \bot_M$ which contradict $e_i \not\leq [\![\alpha_i]\!]^M$ or

$\bigwedge L(n_i)$ is a closed formula with, by definitions of markings, $e_i \leq [\![\bigwedge L(n_i)]\!]^M \leq [\![\alpha_i]\!]^M$ which, again, contradicts the induction hypothesis.

Moreover, no μ-trace occurs on this infinite path for $\mathcal{M}$ is a consistent marking. Hence trace **tr** is a ν-trace.

The last argument that leads to a contradiction is similar to the one above. Considering the smallest ν-variable regenerated infinitely often on trace **tr** we show the sequence of ordinals associated to these regenerations is strictly decreasing after some point which contradicts the well-foundedness of ordinals. □

Let us now prove the two fundamental lemmas which conclude our proofs.

Lemma 4.10 *For any closed formula α on signature Σ, any regular tableau $\mathcal{G}$ for α, there exists a generic automaton $\mathcal{A}_\mathcal{G}$ on signature $\Sigma - \{\wedge\}$ such that, for any algebra $M \in \mathcal{C}$, any $e \in M$, there exists a consistent marking of $\mathcal{G}$ by e iff automaton $\mathcal{A}_\mathcal{G}$ accepts e.*

Proof: Let $\mathcal{G}_1$ be the infinite unwinding of tableau $\mathcal{G}$. From automata theory on ω-languages, applying complementation and determinization, one can show that there exists a deterministic and complete word automaton with parity condition which read any path of $\mathcal{G}$ and accepts only an infinite path with no μ-trace on it. Running this automaton from the initial state q_0 of $\mathcal{G}$ on all paths, we obtain a unique labeling of all states of $\mathcal{G}_1$ by states of the automaton. Then we define the parity index $\Omega(q)$ of any state q of $\mathcal{G}_1$ as the parity index of the corresponding state of the word automaton, and we define the type $\tau(q)$ of state q as G when the G-rule applies in state q, as ϵ otherwise. By construction, the resulting tree is the infinite unwinding of a finite generic automaton with the word automaton translating, in any infinite path of any marking or run, any consistency requirement into a parity requirement and vice versa. Moreover, since the G-rule always applies in tableau $\mathcal{G}$ with function G built without symbol $\wedge$, automaton $\mathcal{A}_\mathcal{G}$ is, indeed, built on signature $\Sigma - \{\wedge\}$. □

Before stating and proving the last lemma, we introduce the notion of tree-shaped automaton which, intuitively, characterizes generic automata which, roughly speaking, directly corresponds to fixpoint formulas. One can check that, in general, generic automata can be considered as sets of fixpoint equations instead of single formulas.

Definition 4.11 *A generic automaton $\mathcal{A}$ is* tree-shaped with back-edges *when:*

1. *for any state q of $\mathcal{A}$ there exits a unique acyclic path from the initial state q_0 to state q,*
2. *for any cycle C in the graph induced by $\mathcal{A}$, there exists a unique state q_C such that, for any state q in cycle C,*
   (a) q_C belongs to the unique acyclic path from the state q_0 to the state q,
   (b) and $\Omega(q_C) \leq \Omega(q)$.

Figure 3 below illustrates the notion of tree-shaped automata.

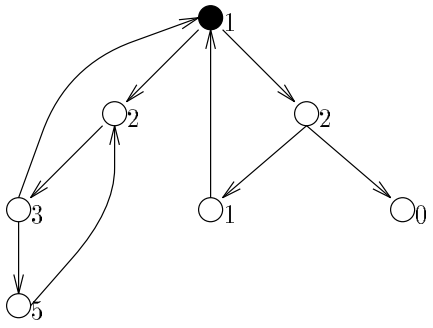

Figure 3. A tree-shaped automaton.

Remark: Obviously: for any generic automaton $\mathcal{A}$ there exists an automaton $\mathcal{A}_1$ tree-shaped with back edges such that, for any algebra M, $L_M(\mathcal{A}) = L_M(\mathcal{A}_1)$. One possible construction is first to unwind automaton $\mathcal{A}$ into an infinite tree and then replace any infinite path by a cycle, building an edge back towards ancestors as soon as possible provided condition 2b is not violated. One can check such a turn back can always occur at a depth smaller than twice the depth of automaton $\mathcal{A}_1$.

Lemma 4.12 *For any generic automaton $\mathcal{A}$ there exists a formula $\alpha_\mathcal{A}$ such that, for any algebra $M \in \mathcal{C}$,*

$$[\![\alpha]\!]^M = \bigvee L_M(\mathcal{A})$$

Proof: Starting from a tree-shaped automaton $\mathcal{A}_1$ equivalent to automaton $\mathcal{A}$ as above, for each state q of $\mathcal{A}_1$, let formula α_q be built by induction from the (pseudo) leaves of $\mathcal{A}_1$ to the root q_0 as follows:

1. if $\tau(q) = \epsilon$ with $\delta(q) = r$ then we put
$$\alpha_q = \sigma_q X_q.\beta_{q,r}$$
2. if $\tau(q) = G$ with $\delta(q) = r_1.\cdots.r_{\rho(G)}$ then we put
$$\alpha_q = \sigma_q X_q.G(\beta_{q,r_1}, \cdots, \beta_{q,r_n})$$

where, in both cases, $\{X_q\}_{q \in Q}$ is a set of distinct variables from $\mathcal{X}$, $\sigma_q = \nu$ when $\Omega(q)$ is even otherwise $\sigma_q = \mu$ and $\beta_{r,q}$ is given by one of the following rules:

- if state r occurs on the unique path from initial state q_0 to state q, i.e. the edge from state q to state r is a back-edge, then $\beta_{q,r} = X_r$,

- otherwise $\beta_{q,r} = \alpha_r$.

From this construction, taking $\alpha_{\mathcal{A}} = \alpha_{q_0}$, one can easily build a tableau $\mathcal{G}_\alpha$ for α equivalent to $\mathcal{A}_1$ (henceforth to $\mathcal{A}$) in the sense that, for any algebra $M \in \mathcal{C}$, any $e \in M$, there exists a consistent marking of tableau $\mathcal{G}_\alpha$ by e iff there is a consistent run of $\mathcal{A}_1$ on e. □

5. Conclusion

In this paper, we have shown that semantics of fixpoint calculi in arbitrary complete lattices can be captured by tableau or automaton techniques. With the continuity hypothesis, we have shown that a reduction theorem holds generalizing fundamental results in automata theory.

Our purpose here is not to give yet more proofs of well-known results but, indeed, providing an automaton and tableau semantics to (almost) arbitrary fixpoint calculi, to participate in the establishment of a general theory of fixpoint calculi and, to some extent, a better understanding of fixpoint logics.

Yet a general theory of fixpoint calculi can still be developed. Indeed, many other results or techniques have been established or investigated for some fixpoint calculi, in particular, the modal mu-calculus. Let us recall for instance the completeness result obtained by Walukiewicz [16], the infiniteness of the alternation depth hierarchy proved by Bradfield [3] and many other results on the model checking problem, e.g. the best algorithm known today [8] or a proof systems for process algebra [1]. Since these works often use tableau or automata techniques we can hope the material presented here will help in generalizing them.

References

[1] H. Andersen, C. Stirling, and G. Winskell. A compositional proof system for the modal mu-calculus. In *IEEE Symp. on Logic in Computer Science*, pages 144–153, 1994.

[2] A. Arnold and D. Niwiński. Fixed point characterization of weak monadic logic definable sets of trees. In *Tree, Aut. and Languages*, pages 159–188. Elsevier, 1992.

[3] J. Bradfield. The alternation hierarchy for Kozen's mu-calculus is strict. In *CONCUR'96*. LNCS 1119, 1996.

[4] E. A. Emerson and E. M. Clark. Characterizing correctness properties of parallel programs using fixpoints. In *Int. Call. on Aut. and Lang. and Programming*, 1980.

[5] D. Janin and I. Walukiewicz. Automata for the modal mu-calculus and related results. In *Math. Found of Comp. Science*. LNCS 969, 1995.

[6] D. Janin and I. Walukiewicz. On the expressive completeness of the modal mu-calculus w.r.t. monadic second order logic. In *CONCUR'96*. LNCS 1119, 1996.

[7] D. Kozen. Results on the propositional μ-calculus. *Theoretical Comp. Science*, 27:333–354, 1983.

[8] D. E. Long, A. Browne, E. C. Clarke, S. Jha, and W. R. Marrero. An improved algorithm for the evalutation of fixpoint expressions. In *Computer Aided Verification*, pages 338–350. LNCS 818, 1994.

[9] A. W. Mostowski. Regular expressions for infinite trees and a standard form of automata. In *Computation Theory*. LNCS 208, 1984.

[10] D. E. Muller and P. E. Schupp. Alternating automata on infinite trees. *Theoretical Comp. Science*, 54:267–276, 1987.

[11] D. E. Muller and P. E. Schupp. Simultating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems of Rabin, McNaughton and Safra. *Theoretical Comp. Science*, 141:67–107, 1995.

[12] D. Niwiński. Fixed point vs. infinite generation. In *IEEE Symp. on Logic in Computer Science*, 1988.

[13] D. Park. Concurrency and automata on infinite sequences. In *5th GI Conf. on Theoret. Comput. Sci.*, pages 167–183, Karlsruhe, 1981. LNCS 104.

[14] M. O. Rabin. Decidability of second order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141, 1969.

[15] C. Stirling and D. Walker. Local model cheking in the modal mu-calculus. *Theoretical Comp. Science*, 89:161–177, 1991.

[16] I. Walukiewicz. Completeness of Kozen's axiomatization of the propositional μ-calculus. In *IEEE Symp. on Logic in Computer Science*, 1995.

[17] I. Walukiewicz. Monadic second order logic on tree-like structures. In *Symp. on Theor. Aspects of Computer Science*, 1996. LNCS 1046.