

Relating levels of the mu-calculus hierarchy and levels of the monadic hierarchy

David Janin Giacomo Lenzi

LaBRI

Université de Bordeaux I - ENSERB

351 cours de la Libération,

F-33 405 Talence cedex

{janin|lenzi}@labri.u-bordeaux.fr

Abstract

As already known [14], the mu-calculus [17] is as expressive as the bisimulation invariant fragment of monadic second order Logic (MSO). In this paper, we relate the expressiveness of levels of the fixpoint alternation depth hierarchy of the mu-calculus (the mu-calculus hierarchy) with the expressiveness of the bisimulation invariant fragment of levels of the monadic quantifiers alternation-depth hierarchy (the monadic hierarchy).

From van Benthem's result [3], we know already that the fixpoint free fragment of the mu-calculus (i.e. polymodal Logic) is as expressive as the bisimulation invariant fragment of monadic Σ_0 (i.e. first order logic). We show here that the ν -level (resp. the $\nu\mu$ -level) of the mu-calculus hierarchy is as expressive as the bisimulation invariant fragment of monadic Σ_1 (resp. monadic Σ_2) and we show that no other level Σ_k for $k > 2$ of the monadic hierarchy can be related similarly with any other level of the mu-calculus hierarchy.

The possible inclusion of all the mu-calculus in some level Σ_k of the monadic hierarchy, for some $k > 2$, is also discussed.

1 Introduction

The propositional modal fixpoint calculus (or mu-calculus for short) introduced by Kozen [17] is considered in this paper. The mu-calculus was initially introduced as a specification formalism for processes modeled as states in transition systems.

However, using the mu-calculus as a logic of processes has a major drawback : the model-checking problem, which is to decide if a (finite) model (given as input) satisfies a formula (also given as input), remains somehow difficult. More precisely, the best model checking algorithms known

so far - see [16] for the latest development - have (time) complexity $O((mn)^{\lceil d/2 \rceil + 1})$ where m is the size of the input graph, n is the size of the formula and d is the fixpoint alternation-depth of the formula which depends on the input formula. Moreover the restriction to mu-calculus formulas with a bounded fixpoint alternation-depth is (theoretically) not an issue because it also strictly reduces the expressive power of the logic. Indeed, Bradfield [4] and, in some weaker sense, Lenzi [18], prove that the hierarchy induced by the fixpoint alternation-depth (the mu-calculus hierarchy) is strict.

In practice, temporal logics [6], which all belong to low levels of the alternation depth hierarchy, are often preferred to the full mu-calculus since in that case the model checking problem has a low degree polynomial (even linear) time complexity.

It is also known that the model-checking problem belongs to $NP \cap co-NP$ [15]. From Fagin's famous correspondence between the class NP and the existential fragment of second order logic [7], this upper bound tells us that all mu-calculus formulas belongs to the level $\Sigma_1 \cap \Pi_1$ of the second order quantifier alternation hierarchy.

Since all mu-calculus formulas can be translated into monadic second order logic (MSO) one may ask whether similar *descriptive complexity results* are available for the monadic quantifier alternation hierarchy (the monadic hierarchy) which is known to be strict (even over finite models as shown by Matz and Thomas [20]). More precisely, since the mu-calculus is as expressive as (or equivalent to) the bisimulation invariant fragment of MSO [14], one may ask whether the full mu-calculus or any level of the mu-calculus hierarchy is equivalent to the bisimulation invariant fragment of some level of the monadic hierarchy.

Van Benthem [3] already shows that the fixpoint free fragment of the mu-calculus (i.e. Polymodal Logic also called Hennessy-Milner logic among computer scientists) is equivalent to the bisimulation invariant fragment of

<i>Levels of the mu-calculus</i>	<i>Levels of the monadic hierarchy</i>	<i>Reference</i>
Mu-calculus	Monadic Second Order Logic	Janin-Walukiewicz 1996
Polymodal Logic	FOL	Van Benthem 1976
ν -level of the mu-calculus	monadic Σ_1	shown here
$\nu\mu$ -level of the mu-calculus	monadic Σ_2	shown here
Properties (all ¹) of arbitrary levels	monadic Σ_3	shown here

Figure 1. Correspondance between levels of the mu-calculus hierarchy and levels of the bisimulation invariant fragment of the monadic hierarchy

monadic Σ_0 (i.e. FOL).

Here, we complete the picture showing that :

Theorem 1.1 *The ν -level (resp. the μ -level) of the mu-calculus hierarchy is equivalent to the bisimulation invariant fragment of the level Σ_1 (resp. Π_1) of the monadic hierarchy.*

and

Theorem 1.2 *The $\nu\mu$ -level (resp. the $\mu\nu$ -level) of the mu-calculus hierarchy is equivalent to the bisimulation invariant fragment of the level Σ_2 (resp. Π_2) of the monadic hierarchy.*

From Arnold's proof of the strictness of the mu-calculus hierarchy [2], we also show that :

Theorem 1.3 *For each integer $k > 2$ there exists a bisimulation invariant formula of monadic Σ_3 that does not belong to the k th level of the mu-calculus hierarchy.*

In other words, no other equivalence similarly relates levels of the mu-calculus hierarchy with levels of the monadic hierarchy.

The question whether the mu-calculus is equivalent to the bisimulation invariant fragment of monadic Σ_k , for some integer $k > 2$, remains, strictly speaking, open. However, the following theorem, which is a consequence of the work of Courcelle [5], shows that, on a quite general class of graphs (or the class of all graphs¹), this is already true with monadic Σ_3 .

Theorem 1.4 *Over the class of graphs of bounded degree (or bounded tree-width) all mu-calculus formulas can be translated into monadic Σ_3 formulas.*

Figure 1 above summarizes all these results. One must be aware that, for these results, we are considering arbitrary finite and infinite models. Rosen [28] shows that van Benthem's result still holds over finite models only. All other statements mentioned in Figure 1 are open problems over finite models.

¹provided, as in MS₂ in [5], quantification over edges is available !

Although these new results essentially have a theoretical flavor they can also be seen as a general toolkit to analyse, from syntax, the model-checking complexity of logics of programs. Indeed, most logics of programs are (implicitly defined as) particular fragments of the bisimulation invariant fragment of MSO. The result above says that, as soon as these logics can be translated into monadic Δ_1 (resp. monadic Δ_2) then the model checking complexity is linear (resp. quadratic) in the size of the input program.

Related works

The study of various bisimulation invariant fragments of logical formalisms leads to some other results.

Following Hafer and Thomas [10] logical characterization of CTL* over the binary tree, Moller and Rabinovich [21] obtain a similar characterization of CTL* over arbitrary trees : CTL* is as expressive as the bisimulation invariant fragment of MSO over trees with path quantifiers instead of general set quantifiers.

With a more expressive language than the mu-calculus, Grädel, Hirsch and Otto show the expressive completeness of the guarded fixpoint calculus w.r.t. the bisimulation invariant fragment of guarded second order logic [9].

Over finite models, Otto gives a fixpoint characterization of bisimulation invariant PTIME [25].

In his PhD thesis [11], Hollenberg also characterizes the bisimulation invariant fragment of MSO via *bisimulation-quantifiers* [8]. It is an open question whether his approach extends to the bisimulation invariant fragment of monadic Σ_1 or monadic Σ_2 .

Investigating bisimulation invariance inside MSO also leads to apply works on MSO over trees. The pioneering works of Rabin [26][27] on the monadic second order theory of the binary tree (S2S) are obviously relevant here. Also the many automata characterization of various mu-calculi over trees which starts in the early 80's with the results of Niwinski [24] or Street and Emerson [32] among others are fundamental. In this paper, we use one of the last and most achieved extension of these techniques and results obtained by Walukiewicz [33].

Note however, Theorems 1.1 and 1.2 are not immediate consequences of these results.

For the analysis of bisimulation invariance inside monadic Σ_1 , the restriction to trees is even misleading since, with properties definable in monadic Σ_1 , bisimulation invariance over trees is less restrictive than bisimulation invariance over arbitrary graphs. Indeed, the monadic Σ_1 formula $\exists xp(x)$, although bisimulation invariant over trees, would mean, as a bisimulation invariant property over graphs, that there is a directed path from a distinguished vertex (the root of the graph) to some vertex x where p holds. This property is at least as difficult to express as directed reachability which, as shown by Ajtai and Fagin [1], is not expressible in monadic Σ_1 .

For the analysis of bisimulation invariance inside monadic Σ_2 , it is true that bisimulation invariance over trees or graphs coincides. But then, there is no real characterizations of FOL or monadic Σ_1 logic of trees so no simple inductive proof is available. To prove Theorem 1.2, we shall extend to all trees a new similar result of Lenzi [19], reproved by Skurczyński [31] in a more automata theoretical way, which says that, on the binary tree, languages definable in monadic Σ_2 are exactly the languages recognizable by tree automata with Büchi conditions.

Overview

The paper is organized as follows. First we recall the definition of bisimulation equivalence. Then, in relation with it, we present the notions of κ -expansions which provide, in some sense, canonical representatives of bisimulation equivalences classes of graphs.

In the third part, we recall the definitions of Monadic Second Order Logic and the modal and counting mu-calculus. We also recall most of the known results relating these languages.

In the fourth part, we give a definition of tree automata which, with various acceptance criteria, will constitute the main technical tools to prove our results.

In the fifth and sixth parts, bisimulation invariance in monadic Σ_1 and in monadic Σ_2 are analyzed. Sketch of proofs for Theorem 1.1 and Theorem 1.2 are given.

In the last part, the case of levels Σ_k for $k > 2$ is considered and Theorem 1.3 and Theorem 1.4 are proved.

Acknowledgement

Thanks to André Arnold and Igor Walukiewicz for many stimulating and helpful discussions on this topic. Thanks to Mike Robson for his help writing this final version.

2 Graphs, Bisimulation and Expansion

We recall here the notions of transition systems, bisimulation equivalence and expansion of transition systems. Since a transition system is simply a directed graph with a distinguished vertex called its source or root, we use in the following the vocabulary of (directed) graphs.

Also, in order to simplify statements and proofs, we only consider here unlabeled directed graphs (built over a single binary relation symbol). One can check that all the results presented here can easily be generalized to (finitely) labeled directed graphs, i.e. graphs built over a finite set of binary relation symbols.

Let *Prop* be a set of unary predicate symbols and let R be a binary relation symbol. A *graph with a root*, simply called *graph* in the sequel, is a tuple:

$$M = \langle S^M, r^M, R^M, \{p^M\}_{p \in Prop} \rangle$$

with a set S^M of *vertices*, a *root* $r^M \in S^M$, a binary *successor relation* $R^M \subseteq S^M \times S^M$ and for each $p \in Prop$, a subset $p^M \subseteq S^M$.

Graphs M and N are called *bisimilar* when there exists a relation $R \subseteq S^M \times S^N$, called a *bisimulation relation*, such that $(r^M, r^N) \in R$ and for every $(s, t) \in R$ and $p \in Prop$, $s \in p^M$ iff $t \in p^N$, and whenever $(s, s') \in R^M$ for some s' , then there exists t' such that $(t, t') \in R^N$ and $(s', t') \in R$, and whenever $(t, t') \in R^N$ for some t' , then there exists s' such that $(s, s') \in R^M$ and $(s', t') \in R$.

Given any set κ (disjoint from S^M), a κ -*indexed path* in M is a non empty finite or infinite word $w \in S^M \cdot (\kappa \cdot S^M)^\infty$ such that whenever $w = u.s.k.s'.v$ with $u \in (S^M \cdot \kappa)^*$, $s \in S^M$, $k \in \kappa$, $s' \in S^M$ and $v \in (\kappa \cdot S^M)^\infty$ one has $(s, s') \in R^M$. The length $|w|$ of κ -index path w is defined as the number of occurrences of elements of S^M in w , e.g. when $w = s_0.k_1.s_1 \dots k_n.s_n$ we put $|w| = n + 1$. In this case, we say s_0 is the source of w , s_n is the target of w and w is a (κ -indexed) path from s_0 to s_n .

Remark that (up to isomorphism) the notion of κ -indexed path only depends on the cardinality of κ . In particular, when κ is a singleton, κ -indexed paths are nothing but the usual (directed) paths in a graph.

The κ -*expansion* $T^\kappa(M)$ of system M is defined as follows : set $S^{T^\kappa(M)}$ is the set of all finite κ -indexed paths of M with root r^M , the root $r^{T^\kappa(M)}$ equals r^M , relation $R^{T^\kappa(M)}$ is the set of all pairs of the form $(u.s, u.s.k'.s') \in S^{T^\kappa(M)} \times S^{T^\kappa(M)}$ with $u \in (S^M \cdot \kappa)^*$, s and $s' \in S^M$ and $k' \in \kappa$ such that $(s, s') \in R^M$, and, for any $p \in Prop$, $p^{T^\kappa(M)}$ is the set of all κ -indexed path of the form $u.s \in S^{T^\kappa(M)}$ with $u \in (S^M \cdot \kappa)^*$ and $s \in p^M$.

Any κ -expansion is a tree. Moreover, when κ is a singleton, the κ -expansion of M , from now on denoted by $T(M)$, is nothing but what is usually called the unwinding or un-

raveling of graph M from its root r^M . Vertices of $T(M)$ are all finite paths from the root.

When M is a tree, i.e. when M and $T(M)$ are isomorphic, we shall use the notation \leq^M for the *order relation* induced by the tree-structure of M , i.e. relation \leq^M is the reflexive and transitive closure of relation R^M .

The notion of κ -expansion gives in some sense canonical representatives of equivalence classes under bisimulation as illustrated by the following fact.

Fact 2.1 *For any infinite set κ and for any graphs M and N of cardinality at most $|\kappa|$, M and N are bisimilar iff $T^\kappa(M)$ and $T^\kappa(N)$ are isomorphic.*

3 First order and monadic second order logic and the propositional μ -calculus

In this section we define first order logic (FO) and monadic second order logic (MSO) and two variants of the propositional μ -calculus [17]. All logics are interpreted over transition systems. Note that a transition system M , as defined above, is a FO-structure with domain $dom(M) = S^M$ on the vocabulary $\{r, R\} \cup Prop$ with r a constant symbol standing for the root, R a binary relation symbol and $Prop$ a set of unary relation symbols.

3.1 FO and MSO

Let $var = \{x, y, \dots\}$ and $Var = \{X, Y, \dots\}$ be respectively some disjoint sets of first order and monadic second order variable symbols.

First order logic over the vocabulary $\{r, R\} \cup Prop$ can be defined as follows. The set of FO formulas is the smallest set containing formulas $p(t)$, $t = t'$, $R(t, t')$, $X(t)$ for $p \in Prop$, $X \in Var$ and $t \in var \cup \{r\}$ and closed under negation \neg , disjunction \vee , conjunction \wedge and existential \exists and universal \forall quantifications over FO variables.

Monadic second order logic over the vocabulary $\{r, R\} \cup Prop$ can be defined as follows. The set of MSO formulas is the smallest set containing all FO formulas and closed under negation \neg , disjunction \vee , conjunction \wedge and existential \exists and universal \forall quantifications over set variables.

For any MSO formula, we use the notation $\varphi(x_1, \dots, x_m, X_1, \dots, X_n)$ for the formula φ with free first order variables among $\{x_1, \dots, x_m\}$ and free set variables among $\{X_1, \dots, X_n\}$. For any graph M , any elements $s_1, \dots, s_m \in S^M$, any sets $S_1, \dots, S_n \subseteq S^M$, we use the notation

$$M \models \varphi(s_1, \dots, s_m, S_1, \dots, S_n)$$

to say that formula φ is true in M , or M satisfies φ , under the interpretation of each FO variable x_i by the vertex s_i

and each set variable X_j by the set S_j . We do not repeat here the definition of this satisfaction relation.

A class \mathcal{C} of graph is said *MSO definable* when there exists a sentence $\varphi \in MSO$, i.e. a formula with no free variable, such that $M \in \mathcal{C}$ iff $M \models \varphi$. A class \mathcal{C} of transition systems is *bisimulation closed* (resp. *closed under unwinding*) if whenever $M \in \mathcal{C}$ and M' is bisimilar to M then $M' \in \mathcal{C}$ (resp. if for any M , $M \in \mathcal{C}$ iff $T(M) \in \mathcal{C}$). A sentence φ is *bisimulation invariant* (resp. *unwinding invariant*) if the class of transition systems it defines is bisimulation closed (resp. closed under unwinding). Remark that bisimulation invariance implies unwinding invariance since any graph M is bisimilar to its unwinding $T(M)$.

The notion of bisimulation invariance (or unwinding invariance) extend to arbitrary formula $\varphi(X_1, \dots, X_n)$ with no free FO variable considering graphs over the set of predicate symbols $Prop' = Prop \cup \{X_1, \dots, X_n\}$. Since fixpoint formulas, which we will consider later, may have free set variables, we shall implicitly consider this extension of graph to $Prop'$ whenever there is no ambiguity.

Finally, the monadic quantifier alternation-depth hierarchy is defined as follows. The first level $\Sigma_0 = \Pi_0$ is defined as the set of all formulas of first order logic. Then, for each integer k , level Σ_{k+1} (resp. level Π_{k+1}) is defined as the set of all formulas of the form $\exists X_1 \dots \exists X_n \varphi$ with $\varphi \in \Pi_k$ (resp. $\forall X_1 \dots \forall X_n \varphi$ with $\varphi \in \Sigma_k$). The bisimulation invariant (resp. unwinding invariant) fragment of the level Σ_k of MSO formulas is defined as the set of all bisimulation invariant (resp. unwinding invariant) formulas of Σ_k with no free first order variables.

3.2 Modal and counting μ -calculus

The set of the modal μ -calculus formulas is the smallest set containing $Prop \cup Var$ which is closed under negation, disjunction and the following formation rules:

- if α is a formula then $\diamond\alpha$ and $\square\alpha$ are formulas,
- if $\alpha(X)$ is a formula and X occurs only positively (i.e. under even number of negations) in $\alpha(X)$ then $\mu X.\alpha(X)$ and $\nu X.\alpha(X)$ are formulas.

The set of counting μ -calculus formulas is defined as above replacing standard modalities \diamond and \square by counting modalities \diamond_k and \square_k for any integer k .

We use the same convention as for MSO with free set variables, i.e. we denote by $\alpha(X_1, \dots, X_n)$ a formula with free variables among $\{X_1, \dots, X_n\}$. For convenience, we may also omit these free set variables in formula α considering then implicitly that graphs have been built over the set of unary predicate symbols $Prop' = Prop \cup \{X_1, \dots, X_n\}$. In the sequel, we call *fixpoint formula* any formula of the modal or counting μ -calculus.

Atomic formulas :	$\varphi_p = p(r), \varphi_X = X(r),$
Boolean connectives :	$\varphi_{\alpha \wedge \beta} = \varphi_\alpha \wedge \beta, \varphi_{\alpha \vee \beta} = \varphi_\alpha \vee \beta$ and $\varphi_{\neg \alpha} = \neg \varphi_\alpha$
Modalities :	$\varphi_{\diamond \alpha} = \exists z R(r, z) \wedge \varphi_\alpha[z/r], \varphi_{\square \alpha} = \forall z R(r, z) \Rightarrow \varphi_\alpha[z/r]$
Counting modalities :	$\varphi_{\diamond_k \alpha} = \exists z_1, \dots, z_k \text{diff}(z_1, \dots, z_k) \wedge \bigwedge_{i \in [1, k]} R(r, z_i) \wedge \varphi_\alpha[z_i/r]$ and $\varphi_{\square_k \alpha} = \forall z_1, \dots, z_k (\text{diff}(z_1, \dots, z_k) \wedge \bigwedge_{i \in [1, k]} R(r, z_i)) \Rightarrow \bigvee_{i \in [1, k]} \varphi_\alpha[z_i/r]$
Fixpoints :	$\varphi_{\mu X. \alpha(X)} = \forall X (\forall z \varphi_{\alpha(X)}[z/r] \Rightarrow X(z)) \Rightarrow X(r)$ and $\varphi_{\nu X. \alpha(X)} = \exists X (\forall z X(r) \Rightarrow \varphi_{\alpha(X)}[z/r]) \wedge X(r)$

Figure 2. Semantics of fixpoint formulas

The meaning of a fixpoint formula α in a transition system M can be defined as an MSO formula φ_α with no free first order variables and with the same free set variables. The inductive definition of φ_α is described in Figure 2 below. In this figure, $\text{diff}(z_1, \dots, z_k)$ is the quantifier free FO formula stating that $z_i \neq z_j$ for all $i \neq j$, α and β are arbitrary formulas, k is any integer, X any second order variable, and z, z_1, \dots, z_k any FO variables. Formula $\varphi_\alpha[z/r]$ is the formula obtained from φ_α by replacing any occurrence of r by z , provided FO variable z has been chosen in such a way it is never captured by a FO quantification during this replacement.

Remark that one can choose FO variables in such a way that, for any modal mu-calculus formulas α , formula φ_α is defined using at most two FO variables and, for any counting mu-calculus formulas α , φ_α is defined using at most $k + 1$ variables where k is the greatest integer such that modality \diamond_k or \square_k occurs in α .

For any fixpoint formula α , we shall write $M \models \alpha$ when $M \models \varphi_\alpha$. We say that an MSO formula φ is equivalent to a fixpoint formula α when $\models \varphi_\alpha \Leftrightarrow \varphi$.

The following fact follows from the above definitions :

Fact 3.1 *For any fixpoint formula, if α is a modal (resp. counting) mu-calculus formula then φ_α is bisimulation invariant (resp. unwinding invariant).*

The following theorems show that the above invariance properties characterize in some sense the expressive power of these fixpoint calculi.

Theorem 3.2 (from Walukiewicz [33]) *A MSO sentence is invariant under unwinding iff it is equivalent to some counting mu-calculus formula.*

and

Theorem 3.3 (Janin-Walukiewicz [14]) *A MSO sentence is invariant under bisimulation iff it is equivalent to some modal mu-calculus formula.*

Finally, the (modal or counting²) fixpoint alternation-depth hierarchy defined as follows. The first level $N_0 = M_0$ is defined as the set of all (modal or counting) fixpoint free formula with negation only applied to propositional constants of *Prop*. Then, for each integer k , level N_{k+1} (resp. level M_{k+1}) is defined as the closure of $N_k \cup M_k$ under disjunction, conjunction, substitution - provided no free variable becomes bounded during the substitution process - and greatest fixpoint construction (resp. least fixpoint construction). In the sequel, we shall also call ν -level (resp. μ -level) or $\nu\mu$ -level (resp. $\mu\nu$ -level) of the fixpoint hierarchies, the level N_1 (resp. M_1) or N_2 (resp. M_2).

Theorem 3.4 (Bradfield [4]) *For each integer k there is a modal mu-calculus formula $\alpha \in N_k$ which is not equivalent to any modal mu-calculus formula in $N_{k'}$ with $k' < k$.*

Arnold [2] shows that the above result still holds restricted to the binary tree. From this stronger result we also have :

Theorem 3.5 (From Arnold [2]) *For each integer k there is a counting mu-calculus formula $\alpha \in N_k$ which is equivalent to no counting mu-calculus formula in $N_{k'}$ with $k' < k$.*

Proof. Observe first that the binary tree is definable in the counting mu-calculus with a formula of N_1 . Moreover, over the binary tree (with distinct left and right successors) the counting and the modal mu-calculus are - level by level - equally expressive. So Arnold's result extends to the counting fixpoint hierarchy. \square

4 Infinite tree automata

We define here tree automata that characterize the expressive power of the two mu-calculi defined above. Although the main ideas and proof techniques go back to, at least, the work of Streett and Emerson on the mu-calculus [32], it took some times for these techniques to

²depending on the modalities one allows

be really understood and generalized to wider settings than the non emptiness or the model checking problem for the modal mu-calculus alone. In this section, we more or less follow Walukiewicz's general approach [33].

In the sequel, the *alphabet* Σ is defined as the powerset $\mathcal{P}(\text{Prop})$ of Prop . The intuition behind this is that a vertex x in a tree M is labeled by the “letter” $\lambda(x) \in \Sigma$ defined as the set $\lambda(x) = \{p \in \text{Prop} : x \in p^M\}$.

An *alternating counting tree-automaton* is a tuple

$$\mathcal{A} = \langle Q, \Sigma, q_0, \Omega, \delta \rangle$$

for a finite set of *states* Q , the finite *alphabet* Σ , an *initial state* $q_0 \in Q$, a *parity index function* $\Omega : Q \rightarrow \mathbb{N}$ and the *transition function* $\delta : Q \times \Sigma \rightarrow L(Q)$ where $L(Q)$ is the set of positive FO sentences, called *transition specifications*, built on the vocabulary Q where each state $q \in Q$ is seen as a unary predicate, i.e. the least set of FO formulas containing formulas $q(x)$, $x = y$, $x \neq y$, and closed under conjunction, disjunction, existential and universal FO quantifications.

Remark that here counting means that the automaton is capable, via equality and inequality inside transition specifications, to count up to some bound the number of successors of vertices.

A tree-automaton \mathcal{A} is called an *alternating modal tree-automaton* when, for each $q \in Q$, each $a \in \Sigma$, the FO formula $\delta(q, a)$ is built without the atomic formulas $x = y$ and $x \neq y$.

A tree-automaton \mathcal{A} is called a *non deterministic counting tree-automaton* when, for each $q \in Q$, $a \in \Sigma$, $\delta(q, a)$ is a disjunction of formulas of the form

$$\begin{aligned} \exists x_1, \dots, x_k \text{diff}(x_1, \dots, x_k) \wedge q_{i_1}(x_1) \wedge \dots \wedge q_{i_k}(x_k) \wedge \\ \forall z, \text{diff}(z, x_1, \dots, x_n) \Rightarrow \bigvee_{q' \in Q'} q'(z) \end{aligned}$$

with any states q_{i_1}, \dots, q_{i_k} not necessarily distinct and any $Q' \subseteq Q$ where, again, *diff* predicates only says that each variable is distinct from any other.

Note that non deterministic modal automata can also be defined (see [13]) but, apart for the non emptiness problem, they don't have all the interesting properties of usual notions of non deterministic automata such as, for instance, closure under projection. This comes from the fact the modal mu-calculus (or even polymodal logic) is not closed under set quantifiers as shown by the “formula” $\exists X (\diamond X \wedge \diamond \neg X)$.

Given a graph M , a *run* of \mathcal{A} over M is a graph ρ which set of vertices V^ρ is some subset of the set of pairs $(s, q) \in S^M \times Q$ with $(r^M, q_0) \in V^\rho$ and which set of edges $E^\rho \subseteq V^\rho \times V^\rho$ is such that : for any pair $(s, q) \in V^\rho$, given the local structure $L_{s,q}^\rho$ over the vocabulary Q defined by $\text{dom}(L_{s,q}^\rho) = \{s' \in S^M : (s, s') \in R^M\}$ and, for each

$p \in Q$, $p^{L_{s,q}^\rho} = \{s' : ((s, q), (s', p)) \in E^\rho\}$, one has

$$L_{s,q}^\rho \models \delta(q, \lambda(s))$$

A run ρ is called *functional* when, for any $s \in S^M$ there is at most one $q \in Q$ such that $(s, q) \in V^\rho$.

A run ρ of \mathcal{A} over M is an *accepting run* when, for each infinite path π in ρ of the form $\pi = (r^M, q_0) \cdot (s_1, q_1) \cdot \dots$ the minimum $\min\{\Omega(q_i) : |\{j \in \mathbb{N} : q_i = q_j\}| = \infty\}$ is even.

The next lemma shows that, although runs are defined over arbitrary graphs, these automata implicitly “read” trees as input.

Lemma 4.1 *For each graph M there is an accepting run of \mathcal{A} over M iff there is an accepting run of \mathcal{A} over $T(M)$.*

Proof. From left to right just notice that the unwinding of an accepting run of \mathcal{A} over M is an accepting run of \mathcal{A} over $T(M)$. The converse, less immediate, can be proven within parity game theory, the existence of an accepting run of \mathcal{A} over M being equivalent to the existence of a memoryless winning strategy in some parity game built from \mathcal{A} and M . \square

For the next lemmas and theorems, we shall concentrate on trees.

Given an automaton \mathcal{A} , we denote by $L(\mathcal{A})$ the class of all trees M such that there exists an accepting run of \mathcal{A} over M . The class $L(\mathcal{A})$ is called the language of trees recognized by \mathcal{A} .

The following theorem can be obtained from the results presented in [33]. It also follows from [12].

Theorem 4.2 *For each class of tree L , the following statements are equivalent :*

1. L is definable with an MSO sentence,
2. L is definable with a counting mu-calculus formula,
3. $L = L(\mathcal{A})$ for some alternating counting tree automaton \mathcal{A} ,
4. $L = L(\mathcal{A})$ for some non deterministic³ counting tree automaton \mathcal{A} .

and the next one follows from [32] and [14]

Theorem 4.3 *For each class of tree L , the following statements are equivalent :*

1. L is definable with a bisimulation invariant MSO sentence,
2. L is definable with a modal μ -calculus formula,
3. $L = L(\mathcal{A})$ for some modal tree automaton \mathcal{A} .

Some particular subclasses of tree-automaton that will be useful in the sequel. Automaton $\mathcal{A} = \langle Q, \Sigma, q_0, \Omega, \delta \rangle$

³possibly with more parity indices

is called a ν -automaton (resp. $\nu\mu$ -automaton or Büchi automaton) when $\Omega(Q) = \{0\}$ (resp. when $\Omega(Q) = \{0, 1\}$).

These automata characterize the ν -levels and $\nu\mu$ -levels of the counting and modal mu-calculi in the following sense.

Lemma 4.4 (Expressiveness) *A class of tree L is recognized by a (counting or modal) ν -automaton (resp. $\nu\mu$ -automaton) iff L is definable by a (modal or counting) mu-calculus formula of the ν -level (resp. of the $\nu\mu$ -level).*

Proof. This lemma is a particular case of the well-known correspondence between level of the mu-calculus hierarchy and the number of parity indices needed in alternating tree-automata. This correspondance was first achieved, in the case of the binary tree, by Niwiński [24]. See [33] for a proof in the counting mu-calculus case. \square

This implies in particular that the classes of languages recognized by ν -automata or $\nu\mu$ -automata are closed under union and intersection.

For counting automata more properties are available :

Lemma 4.5 *The class of languages recognizable by counting ν -automata (resp. by counting $\nu\mu$ -automata) is closed under projection.*

Proof. This lemma follows from the next two. \square

Lemma 4.6 (Simulation) *A language recognized by a counting ν -automaton (resp. a counting $\nu\mu$ -automaton) is also recognized by a non deterministic counting ν -automaton (resp. a non deterministic counting $\nu\mu$ -automaton).*

Proof. Extension to arbitrary trees of (a part of) Muller and Schupp's simulation theorem [23] for alternating tree automata over the binary tree. \square

and

Lemma 4.7 (Projection) *The projection of a language recognized by a non deterministic counting automaton is also recognized by a non deterministic automaton with the same set of states and parity function.*

Proof. When \mathcal{A} is non deterministic counting one can restrict runs (over trees) to be functional without changing the language recognized by \mathcal{A} . Closure under projection immediately follows from this restriction. \square

To conclude this section on automata, we recall here the heart of the bisimulation invariance result presented in [14] as the following lemma which will be used in the sequel :

Lemma 4.8 *For each non deterministic counting tree automaton \mathcal{A} there exists a modal automaton \mathcal{B} , with the same set of states and parity function, such that, for each tree M , any infinite set κ , $T^\kappa(M) \in L(\mathcal{A})$ iff $M \in L(\mathcal{B})$.*

Proof. See [14] for a complete proof. The main idea is to define \mathcal{B} as the automaton obtained from \mathcal{A} by replacing all equalities or inequalities in the FO formula of δ by some true formula. \square

5 Bisimulation invariance in monadic Σ_1

In this section, we prove theorem 1.1. For this, we first prove the analogue for unwinding invariance, from which, applying Lemma 4.4 and Lemma 4.8, we obtain the desired result.

So our goal is to prove the following theorem :

Theorem 5.1 *The unwinding invariant fragment of the level Σ_1 (resp. Π_1) in the monadic hierarchy equals the ν -level (resp. the μ -level) of the counting mu-calculus hierarchy.*

Proof. By duality, it is sufficient to prove the result for monadic Σ_1 . Moreover, it is a classical result, from Lemma 4.4 stated above, that properties definable in the ν -level of the counting mu-calculus are definable in monadic Σ_1 . So it remains to prove that :

Lemma 5.2 *Any unwinding invariant formula of monadic Σ_1 is equivalent to a formula of the ν -level of the counting mu-calculus.*

In order to do so, one must understand that, as stated in the introduction, it is not sufficient to restrict our analysis to trees - although an unwinding invariant property is fully determined by its models among trees - because over trees, monadic Σ_1 is strictly more expressive than the ν -level of the counting mu-calculus as the (even FO) formula $\exists xp(x)$ shows.

First, remark that an unwinding invariant property only speaks about the vertices reachable from the root because any graph M has the same unwinding as the subgraphs induced by these vertices. This leads to the following definitions. Let $c(r_M)$ be the set of all vertices which are reachable from the root r_M via a (directed) path (called in the sequel the *directed connected component* induced by r_M). For each MSO sentence φ , let us define φ^c as the formula φ relativized to the directed connected component $c(r)$ of r , i.e. φ^c is obtain from φ replacing any first order or set quantification by quantifications over vertices or subsets of $c(r)$. With this definition and the previous remark it appears that if φ is invariant under unwinding then φ is equivalent to φ^c ; in particular, if φ is in monadic Σ_1 then φ^c is also (definable) in monadic Σ_1 .

So let φ be an unwinding invariant monadic Σ_1 formula. By the Gaifman normal form theorem for first order logic, there is some integer k such that φ is of the form

$$\varphi = \exists \vec{Z}. \varphi_1$$

with \vec{Z} a finite vector of sets variables and φ_1 is a finite boolean combination of FO formulas $G(\vec{Z})$ of the form

$$G(\vec{Z}) = \exists u_1, \dots, u_l. \theta(u_1, \dots, u_l, \vec{Z})$$

where $\theta(u_1, \dots, u_l, \vec{Z}, \vec{Y})$ is a formula stating that for all distinct indices s and t among $[1, l]$, $\text{dist}(u_s, u_t) > 2k$ and $\text{Ball}(u_s, k) \models \psi_s(\vec{Z})$ for some FO formulas $\psi_s(\vec{Z})$, with $\text{dist}(x, y)$ defined as the length of the shortest undirected path from x to y and $\text{Ball}(x, k)$ is defined as the substructure of M induced by the set of all vertices y such that $\text{dist}(x, y) \leq k$.

For notational simplicity we assume that φ is of the form

$$\varphi = \exists \vec{Z}. G(\vec{Z}) \wedge \neg G'(\vec{Z})$$

with $G(\vec{Z})$ of the form $\exists u \theta(u, \vec{Z})$ and $G'(\vec{Z})$ of the form $\exists u' \theta'(u', \vec{Z})$. One can check that this proof easily extends to the general case.

The relativization φ^c of φ to the strongly connected components of r is then given by :

$$\varphi^c = \exists \vec{Z}. G^c(\vec{Z}) \wedge \neg G'^c(\vec{Z})$$

with $G^c(\vec{Z})$ given by $\exists u \in c(r). \theta^c(u, \vec{Z})$ and $G'^c(\vec{Z})$ given by $\exists u' \in c(r). \theta'^c(u', \vec{Z})$.

Now, we know that the formula φ^c cannot have for arbitrarily large integers n a model M_n , where the points of $c(r)$ satisfying θ^c have (directed) distance more than n from r . Otherwise, the ultraproduct of the M_n s modulo any non principal ultrafilter, would not satisfy φ^c , contrary to the Σ_1 definability of φ^c and Łos ultraproduct theorem (see for instance [29]) which says that the class of models of any Σ_1 formula is closed under ultraproduct.

So given integer \bar{n} such that no model M_n for $n > \bar{n}$ satisfies φ^c , it turns out that formula φ^c is equivalent to formula $\exists \vec{Z}. \neg G'^c(\vec{Z}) \wedge G^{\bar{n}}(\vec{Z})$ with

$$G^{\bar{n}}(\vec{Z}) = \exists u \in c_{\bar{n}}(r). \theta^c(u, \vec{Z})$$

and $c_{\bar{n}}(r)$ the set of all points directly accessible from r in at most \bar{n} steps.

Now it is not difficult to see that $\neg G'^c(\vec{Z})$ is a fixpoint formula of the ν -level over trees (i.e. unwindings) and $G^{\bar{n}}(\vec{Z})$ is even a fixpoint free formula on unwindings as well. By unwinding invariance, this says that φ is equivalent to some formula of the form $\exists \vec{Z} \varphi_{\alpha'}(\vec{Z})$ with $\alpha' \in N_1$.

Then, over trees, Lemma 4.5, ensures $\exists \vec{Z} \varphi_{\alpha'}(\vec{Z})$ is equivalent to some φ_α for some α in the ν -level as well hence, again by invariance under unwinding, φ is equivalent over arbitrary models to φ_α . \square

6 Bisimulation invariance in monadic Σ_2

In this section, we prove theorem 1.2. For this, again, we first prove the analogue for unwinding invariance, from

which, applying Lemma 4.4 and Lemma 4.8 we obtain the desired result. So our goal is to prove the following theorem :

Theorem 6.1 *The unwinding invariant fragment of the level Σ_2 (resp. Π_2) in the monadic hierarchy equals the $\nu\mu$ -level (resp. the $\mu\nu$ -level) of the counting mu-calculus hierarchy.*

Proof. By duality, it is again sufficient to prove the result for monadic Σ_2 . Moreover, it is again a classical result, from Lemma 4.4, that properties definable in the $\nu\mu$ -level of the counting mu-calculus are definable in monadic Σ_2 . So it remains to prove that :

Lemma 6.2 *Any unwinding invariant formula of monadic Σ_2 is equivalent to a formula of the $\nu\mu$ -level of the counting mu-calculus.*

Proof. Somehow, the proof in the case of Σ_2 is simpler than Σ_1 for it is true that, over trees, any monadic Σ_2 formula is equivalent to a $\nu\mu$ -formula which remains to be shown.

For this, we use definability in weak monadic second order logic as an intermediate step. Remember that weak monadic second order logic is monadic second order logic with set quantification restricted to finite sets.

A priori, using weak MSOL doesn't make sense. Indeed, over arbitrary trees, weak MSOL is incomparable with MSOL. However, Theorem 4.2 and the definition of tree automata show that analyzing MSOL over trees can be made over finitely branching trees only. In fact any MS formula satisfiable over the class of trees has a model which is finitely branching, i.e. with finitely many successors from each vertex.

For this reason, we can restrict our study to finitely branching trees and then weak MSOL is a fragment of MSOL since, in this case, finite sets are definable in MSOL.

The sketch of the proof is then the following. First we prove

Lemma 6.3 *Any language of (finitely branching) trees definable in monadic Σ_1 is definable in weak MSOL.*

Then, by closure of weak MSOL under negation, this shows that monadic Π_1 is also included into weak MSOL. Hence monadic Σ_2 is included into the existential projection of weak MSOL. Now, because the class of languages definable by $\nu\mu$ -automaton is closed under projection (see Lemma 4.5) we prove

Lemma 6.4 *Any languages of (finitely branching) trees definable in weak MSOL is recognizable by a $\nu\mu$ -automaton.*

which conclude the proof of Lemma 6.2. \square

In order to prove Lemma 6.3 we can adapt the work of Lenzi [19], to the case of finitely branching trees. Another

approach, following the idea of Skurczyński [31], is to use weak $\nu\mu$ -automaton as an intermediate step.

We recall here that a tree automaton \mathcal{A} is a *weak automaton* when, for any $q \in Q$, any $a \in \Sigma$, for each states q' occurring in formula $\delta(q, a)$, $\Omega(q) \leq \Omega(q')$.

Then, adapting the proof presented in [30] for the k -ary case, one has :

Lemma 6.5 *Any FO definable tree languages is recognizable by a weak strongly non deterministic $\nu\mu$ -automaton.*

But then, since languages recognizable by strongly non deterministic weak $\nu\mu$ -automaton are closed under projection, it is sufficient to show that

Lemma 6.6 *Any languages of (finitely branching trees) recognizable by a weak automaton is definable by a weak MSOL formula.*

And this last lemma is an adaptation of similar result, by Mostowski [22], over the binary tree. \square

For Lemma 6.4, it shall be clear that it can be proved extending, in a quite straightforward way an analogous proof due to Rabin [27] in the binary case.

This concludes the proof of Theorem 6.1 for, applying Lemma 4.4, languages recognizable by $\nu\mu$ -automata equal languages definable by (counting) fixpoint formulas of the $\nu\mu$ -level. \square

7 Above the level Σ_2

In this section, we prove Theorem 1.3 and Theorem 1.4. For this, we assume that the reader has a general knowledge of the theory of parity games⁴. If not, Jurdziński's [16] gives an appropriate, and up to date, overview of the topic.

From [4] we know that, given an integer k , expressing the fact that a position in an arbitrary parity game with sets of parity indices $[0, k]$ cannot be done with any mu-calculus formula of the level N_k . From [2] we know that this is still the case restricted to games of degree two.

Remark that in monadic second order logic, this may also be difficult to express because in some sense it requires some, at least implicit, construction of a (memoryless) strategy for player 0 which is winning for any plays starting in the distinguished position. And winning strategies are peculiar sets of edges which are, in general, not even definable in MSOL.

Still we prove Theorem 1.3 redefining binary games on graphs (over a more complex signature) on which guessing a winning strategy will become possible with a single existential set quantification. The main difficulty is only to

⁴with the winning criteria defined as an even minimal index met infinitely often...!

ensure that such a definition leads to bisimulation invariant class of parity games.

More precisely, given some integer $k > 2$, given *Prop* defined by $Prop = \{p_l, p_r, p_0, \dots, p_k\}$, any graph M such that both $\{p_l^M, p_r^M\}$ and $\{p_0^M, \dots, p_k^M\}$ are partitions of the set of vertices S^M reachable from the source r - which is a bisimulation invariant property - is from now on interpreted as a parity game as follows :

1. any position (reachable from the root) is a position of player 0,
2. a move from such a position is made as follows : player 0 chooses one predicate $p_x \in \{p_l, p_r\}$ and then player 1 chooses the new position $y \in S^M$ such that $y \in p^M(y)$ and $(x, y) \in R^M$,
3. disjoint predicates p_0, \dots, p_k encode the parity indices of each of these positions.

Theorem 1.3 is then a consequence of the following lemma :

Lemma 7.1 *For each integer $k > 2$, the class W_0^k of (encoded) games over the set of indices $[0, k]$ where the root is a winning position for player 0 is bisimulation closed, definable with a monadic Σ_3 formula and not definable in the level N_k of the mu-calculus.*

Proof. First observe that any bisimulation relation relates winning positions for player 0 to winning position for player 0 so the class W_0^k is indeed bisimulation closed.

Then, it is clear that any binary game can be encoded in such a way. Moreover, computing with a mu-calculus formula the fact that the root r is a winning positions for player 0 in this encoding is as difficult - in terms of number of alternations of least and greatest fixpoints - as computing the fact that the root r is a winning position for the same player in binary games so, following the result of Arnold [2], it requires at least $k + 1$ alternations of least and greatest fixpoints.

Now, to conclude the proof it is sufficient to show that the class W_0^k is definable in monadic Σ_3 . But this can easily be achieved as follows : first, with some existential set quantifier, one can guess a winning strategy for player 0, e.g. guessing the set of positions X from which player 0 chooses predicate p_r . Then it is clear that a $\mu\nu$ -formula of the mu-calculus (henceforth a monadic Π_2 formula) is sufficient to check that this set X is indeed a winning strategy for player 0 in any plays that start at the root. Indeed, one has to check the minimal parity condition on any cycle reachable from the root when player 0 follows the strategy given by set X . In the intended $\mu\nu$ -formula, one least fixpoint enables us to reach any of these cycles and then, one nested greatest fixpoint enables us to check that the minimum parity index met on each of these cycles is even.

Guessing a winning strategy and checking that it is winning for player 0 can thus be expressed in monadic Σ_3 . \square

The proof of Theorem 1.4 is also almost done. Indeed, from the proof of previous lemmas it is clear that *with one existential quantification over sets of edges* the winning position for player 0 can be expressed as a monadic Σ_3 unary predicate. But it also follows from Lemma 4.4 that checking a fixpoint formula on a graph can be done via a monadic Σ_1 transduction which leads to computing winning positions with as many parity indices as the alternation depth of the formula. Moreover, if the input graph is of bounded degree (or bounded tree-width) then the resulting parity game is also of bounded degree (or bounded tree-width). Now Courcelle shows that over graphs with bounded degree (or tree-width) quantification over edges can be “simulated” by quantifications over vertices via, again, a monadic Σ_1 transduction. Altogether, this says that over graphs of bounded degree (or bounded tree-width) mu-calculus formulas can be translated into monadic Σ_3 formulas. This concludes the proof of Theorem 1.4. \square

References

- [1] M. Ajtai and R. Fagin. Reachability is harder for directed rather than undirected finite graphs. *Journal of Symbolic Logic*, 55:113–150, 1990.
- [2] A. Arnold. The mu-calculus alternation-depth hierarchy over the binary tree is strict. *Theoretical Informatics and Applications*, 33:329–339, 1999.
- [3] J. Benthem. *Modal Correspondance Theory*. PhD thesis, University of Amsterdam, 1976.
- [4] J. Bradfield. The modal mu-calculus alternation hierarchy is strict. *Theoretical Comp. Science*, 195:133–153, 1998.
- [5] B. Courcelle. The monadic second-order logic of graphs VI: On several representations of graphs by logical structures. *Discrete Applied Mathematics*, 54:117–149, 1994.
- [6] E. Emerson. Temporal and modal logic. In J. Van Leeuwen, editor, *Handbook of Theor. Comp. Science (vol. B)*, pages 995–1072. Elsevier, 1990.
- [7] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In *Complexity of Computation*, volume 7. SIAM-AMS, 1974.
- [8] S. Ghilardi and M. Zawadowski. A sheaf representation and duality for finitely presented Heyting algebras. *Journal of Symbolic Logic*, 60:911–939, 1995.
- [9] E. Grädel, C. Hirsch, and M. Otto. Back and Forth Between Guarded and Modal Logics. In *Proceedings of 15th IEEE Symposium on Logic in Computer Science LICS 2000*, pages 217–228, 2000.
- [10] T. Hafer and W. Thomas. Computationnal tree logic CTL* and path quantifiers in the monadic theory of the binary tree. In *ICALP’87*, pages 269–279. LNCS 267, 1987.
- [11] M. Hollenberg. *Logic and Bisimulation*. PhD thesis, Utrecht University, 1998.
- [12] D. Janin. *Propriétés logiques du non déterminisme et mu-calcul modal*. PhD thesis, Université de Bordeaux I, 1996.
- [13] D. Janin and I. Walukiewicz. Automata for the modal mu-calculus and related results. In *Math. Found of Comp. Science*. LNCS 969, 1995.
- [14] D. Janin and I. Walukiewicz. On the expressive completeness of the modal mu-calculus w.r.t. monadic second order logic. In *CONCUR’96*, pages 263–277. LNCS 1119, 1996.
- [15] M. Jurdziński. Deciding the winner in parity games is in $UP \cap co-UP$. *Information Processing Letters*, 68(3):119–124, 1998.
- [16] M. Jurdziński. Small progress measures for solving parity games. In *Symp. on Theor. Aspects of Computer Science*, pages 290–301. LNCS 1770, 2000.
- [17] D. Kozen. Results on the propositional μ -calculus. *Theoretical Comp. Science*, 27:333–354, 1983.
- [18] G. Lenzi. *The Mu-calculus and the Hierarchy Problem*. PhD thesis, Scuola Normale Superiore, Pisa, 1997.
- [19] G. Lenzi. A new logical characterization of Büchi automata. In *Symp. on Theor. Aspects of Computer Science*, 2001.
- [20] O. Matz and W. Thomas. The monadic quantifier alternation hierarchy over finite graphs is infinite. In *IEEE Symp. on Logic in Computer Science*, pages 236–244, 1997.
- [21] F. Moller and A. Rabinovich. On the expressive power of CTL*. In *IEEE Symp. on Logic in Computer Science*, pages 360–369, 1999.
- [22] A. Mostowski. Hierarchies of weak automata on weak monadic formulas. *Theoretical Comp. Science*, 83:323–335, 1991.
- [23] D. E. Muller and P. E. Schupp. Simulating 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.
- [24] D. Niwiński. On fixed point clones. In *13th ICALP*, pages 464–473, 1986.
- [25] M. Otto. Bisimulation-invariant Ptime and higher-dimensional mu-calculus. *Theoretical Comp. Science*, 224:237–265, 1999.
- [26] M. O. Rabin. Decidability of second order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, 1969.
- [27] M. O. Rabin. Weakly definable relations and special automata. In *Mathematical Logic and Foundation of Set Theory*, pages 1–23. North Holland, 1970.
- [28] E. Rosen. Modal logic over finite structures. *Journal of Logic, Language and Information*, 6:427–439, 1997.
- [29] J. R. Shoenfield. *Mathematical Logic*. Addison-Wesley, Reading, Mass., 1967.
- [30] J. Skurczyński. On three hierarchies of weak SkS formulas. Aachener Informatik-Berichte 90-3, RWTH Aachen, 1990.
- [31] J. Skurczyński. A characterization of Büchi tree automata. unpublished manuscript, Gdansk University, 2000.
- [32] R. S. Streett and E. A. Emerson. An automata theoretic decision procedure for the propositional mu-calculus. *Information and Computation*, 81:249–264, 1989.
- [33] I. Walukiewicz. Monadic second order logic on tree-like structures. In *Symp. on Theor. Aspects of Computer Science*, 1996. LNCS 1046.