
Expressive Power of Monadic Logics on Words, Trees, Pictures, and Graphs

Oliver Matz¹
Nicole Schweikardt²

¹ Institut für Informatik, Universität Kiel, Germany

² Institut für Informatik, Humboldt-Universität zu Berlin, Germany

matz@ti.informatik.uni-kiel.de schweika@informatik.hu-berlin.de

Abstract

We give a survey of the expressive power of various monadic logics on specific classes of finite labeled graphs, including words, trees, and pictures. Among the logics we consider, there are monadic second-order logic and its existential fragment, the modal μ -calculus, and monadic least fixed-point logic. We focus on nesting-depth and quantifier alternation as a complexity measure of these logics.

1 Introduction

There is a close relationship between (generalized) automata theory and the expressive power of certain monadic logics. Already in 1960, Büchi and Elgot proved that a word-language is recognizable by a finite automaton if, and only if, it can be characterized by a monadic second-order formula. Since then, various analogous results, e.g., for labeled trees rather than words, and also for more general classes of labeled graphs, have been obtained. Alluding to the notion of “descriptive complexity theory”, in his survey article [37] for the *Handbook of Formal Languages*, Wolfgang Thomas called the branch of research that investigates the relationship between generalized finite automata and monadic logics a “descriptive theory of recognizability”.

The present paper’s aim is to give a survey of the expressive power of various monadic logics (including monadic second-order logic and its existential fragment, the modal μ -calculus, and monadic least fixed-point logic), on specific classes of finite labeled graphs. In particular, we give details on the following topics:

It is known that on finite words and labeled trees, all the above mentioned monadic logics have the same expressive power and can characterize exactly the languages that are recognizable by a suitable notion of finite automata. Moreover, already one single existential set quantifier suffices to obtain the expressive power of existential monadic second-order logic on

words, trees, and pictures (i.e., two-dimensional words or, equivalently, labeled grid-graphs). This goes back to a paper by Wolfgang Thomas [36], in which he showed that a single existential set quantifier suffices for words. From the proof, one can also infer an elegant proof which shows that finite automata can be simulated by monadic least fixed-point logic. Wolfgang Thomas' Ph.D. students Potthoff [30] and Matz [23] obtained according results for trees and pictures, respectively. On the other hand, when going slightly beyond the class of pictures, it is known from work by Otto [29] that within existential monadic second-order logic, *more* set quantifiers lead to strictly more expressive power.

While on words and labeled trees, existential monadic second-order logic has the same expressive power as full monadic second-order logic, the situation is different for the class of pictures. From work by Giammarresi, Restivo, Seibert, and Thomas [15] it is known that existential monadic second-order logic can define exactly the *recognizable* picture languages, which are characterized by a suitably adapted automaton model, the *tiling-systems*. But full monadic second-order logic on pictures has considerably more expressive power and, in fact, precisely corresponds to the linear time hierarchy (i.e., the linear time analogue of Stockmeyer's polynomial time hierarchy). Similarly, building on results by Schweikardt [34], one obtains that the picture languages definable in monadic least fixed point logic can encode at least all problems that belong to Grandjean's deterministic linear time complexity class DLIN [16]. Furthermore, unless $P = NP$, the expressiveness of monadic least fixed point logic on pictures is strictly weaker than that of monadic second-order logic.

Also some aspects concerning the fine structure of monadic second-order logic over pictures and graphs are understood quite well by now: Matz, Schweikardt, and Thomas [26, 33, 25] showed that the monadic second-order quantifier alternation hierarchy is strict, i.e., that formulas in prenex normal form having a prefix of $k+1$ alternations of set quantifiers can describe strictly more picture languages (or, in general, graph properties) than formulas with only k quantifier alternations. Note, however, that this result does not have implications concerning the strictness of the linear time hierarchy (or the polynomial time hierarchy) as the levels of the monadic second-order quantifier alternation hierarchy do not correspond to the levels of the linear time hierarchy.

When considering the modal μ -calculus instead of monadic second-order logic on finite labeled graphs, an according hierarchy based on the alternation of least and greatest fixed point operators was proved independently by Bradfield [3] and Lenzi [20], see also Arnold [2] for an elegant proof. (The hierarchies proved in [3, 20, 2] are about general structures that are not necessarily finite; via the μ -calculus' *finite model property*

(cf., e.g., [4]), however, they can be directly transferred to the class of finite labeled graphs.)

Up to date, it still is an open question whether an analogous hierarchy can be proved for monadic least fixed point logic.

The rest of this paper is structured as follows: In Section 2 we fix the necessary notation concerning the logics and the structures that are considered in this paper. Section 3 concentrates on the relations between (finite-state) recognizability of word languages, tree languages, and picture languages and their definability in various monadic logics. In Section 4, we go beyond recognizability and study nesting-depth and quantifier alternation as a complexity measure of logics.

2 Logics and Structures Considered in this Paper

This section fixes some basic notations and conventions used throughout the remainder of the paper.

2.1 Structures

All structures considered in this paper are finite and can be viewed as particular kinds of labeled graphs. Namely, we consider labeled trees, words, and pictures (i.e., two-dimensional words).

Let us fix a finite alphabet Σ , whose elements serve as letters at positions in a word or a picture or as labels for nodes in a graph or a tree. For this exposition it is convenient (and no essential loss of generality) to assume that Σ is of the form $\{0, 1\}^t$ for some $t \geq 0$ (for $t = 0$, the alphabet Σ is a singleton).

A *word* (over Σ) is a finite sequence of elements in Σ . A *word language* is a set of words. In order to use logic formulas to define word languages we consider the signature $\{Succ, B_1, \dots, B_t\}$, where *Succ* is a binary relation symbol and B_1, \dots, B_t are unary relation symbols. We identify a word $w = w_1 \cdots w_n$ over Σ with the structure of signature $\{Succ, B_1, \dots, B_t\}$ whose universe is the set $[n] := \{1, \dots, n\}$ of positions in the word, and where *Succ* is interpreted by the natural successor relation on $[n]$ and, for every $i \in \{1, \dots, t\}$, the relation symbol B_i is interpreted by the set of all positions at which the word carries a letter $(\sigma_1, \dots, \sigma_t) \in \Sigma = \{0, 1\}^t$ with $\sigma_i = 1$.

Pictures are two-dimensional analogues of words, i.e., a *picture* (over Σ) is a two-dimensional (rectangular) array over Σ . A *picture language* is a set of pictures. Like for words, it is straightforward to associate, with every picture, a model over a specific signature, this time with two binary relations $Succ_h$ and $Succ_v$ for the horizontal and the vertical successor relation, respectively.

For convenience, all *trees* considered in this paper will be ordered and binary, i.e., every node is either a leaf or has two children. Each node of a *labeled tree* (over Σ) is labeled by an element in Σ . A *tree language*

is a set of labeled trees. Similarly as words and pictures, also trees can be identified in a straightforward way by structures over the signature $\{Succ_1, Succ_2, B_1, \dots, B_t\}$, where the binary relations $Succ_1$ and $Succ_2$ are used for the edges from a node to its first child and to its second child, respectively.

2.2 Logics

We assume that the reader is familiar with first-order logic (FO), monadic second-order logic (MSO), least fixed-point logic (LFP), and the modal mu-calculus. We write MLFP for *monadic* least fixed-point logic, i.e., the fragment of LFP where only *monadic* second-order variables are allowed. It is straightforward to see that monadic least fixed-points can be defined in MSO, and thus the expressive power of MLFP lies between the expressive power of FO and the expressive power of MSO. Some focus of the present paper will also be on *existential monadic second-order logic* (EMSO), which consists of all MSO-formulas of the form

$$\exists X_1 \dots \exists X_\ell \varphi,$$

where φ is first-order, $\ell \geq 0$, and X_1, \dots, X_ℓ are set variables (i.e., monadic second-order variables). Further, we will write 1-EMSO for the fragment of EMSO where just a single set variable is available.

If φ is a sentence (over a suitable signature and a certain logic), the (word, picture, or tree) language *defined* by φ is the set of all words (or pictures or trees) whose associated word (or picture or tree) models make φ true.

3 Monadic Logics and Recognizability

This section concentrates on the relations between recognizability of word languages, tree languages, and picture languages and their definability in various monadic logics. Here, “recognizability” refers to non-deterministic finite automata or suitable adaptations thereof.

We will first quickly review the well-known results on words and trees which, basically, state that all the monadic logics mentioned in Section 2 have the same expressive power, namely of defining exactly the *regular* word languages and tree languages.

Afterwards, we will move over to the case of pictures, where things turn out to be much more subtle, since the various monadic logics differ with respect to their power of defining picture languages.

3.1 Monadic Logics and Recognizability of Words and Trees

The class of regular (or, recognizable) word languages plays a central role in the theory of formal languages. One reason for this is the large variety of its

conceptually different characterizations, for example by means of monoids, grammars, automata, closure properties, and logics. Concerning the subject of this paper, let us focus on the following two: non-deterministic finite automata (NFA) and monadic second-order logic.

Theorem 3.1 (Büchi [5], Elgot [11]). A word language is regular if, and only if, it can be defined by an MSO-sentence.

Since we will come back to this later (in the context of pictures instead of words), let us briefly point out the essential steps in the well-known proof of the above theorem.

Proof (sketch). One direction is simple to prove: Given a non-deterministic finite automaton \mathfrak{A} , we have to construct a monadic second-order sentence that asserts for a given word (model) that there exists an accepting run. The existence of such a run can be expressed by a formula of the form

$$\exists X_1 \cdots \exists X_\ell \varphi(X_1, \dots, X_\ell),$$

where an assignment to the set variables encodes an assignment of \mathfrak{A} 's states to positions in the word, and φ asserts that for any two consecutive positions, this assignment is compatible with the automaton's transition relation, the initial state and the final states. We observe that the resulting formula is in the existential fragment EMSO of monadic second-order logic.

The other direction is more intricate. Typically, it is done as follows: Given an MSO-sentence φ , we may pass to a similar sentence φ' in prenex normal form, where all first-order quantifiers are eliminated and special, new predicates *singleton*(X) are used instead, which assert for a set X that it has just one element. An NFA can be constructed by induction on the construction of such formulas. In this induction, one exploits that the class of regular word languages is closed under union, complementation, and projection, to handle disjunction, negation, and existential MSO quantification, respectively. Q.E.D.

The above proof, in particular, leads to:

Corollary 3.2. Over the class of words, every MSO-sentence is equivalent to an EMSO-sentence.

Even more, it is known that already a single existentially quantified set variable suffices:

Theorem 3.3 (Thomas [36]). Over the class of words, every MSO-sentence is equivalent to a 1-EMSO-sentence.

Proof (sketch). The proof relies on the following simple and elegant idea: Given a deterministic finite automaton \mathfrak{A} with r states and, w.l.o.g., state space $\{1, \dots, r\}$, each state i can be represented by the bit-string 01^i0^{r-i} of length $r' := r + 1$. If w is an input word, we can subdivide w into sub-words such that each of these sub-words has length r' , except for the last one, whose length is between r' and $2r' - 1$. Each of these sub-words can be decorated by the bit-string that represents \mathfrak{A} 's state when entering the first position of the sub-word. Such bit-strings, in turn, can of course be represented by an assignment to a single set variable, e.g., by assuming that the set consists of exactly those positions where the bit-string carries the letter 1.

Now, it is easy to construct a 1-EMSO-sentence of the form $\exists X \varphi(X)$, where φ is first-order and expresses that the bit-string represented by X encodes the list of states assumed by \mathfrak{A} at the beginnings of the sub-words. For constructing φ , note that (1) each sub-word has constant length $< 2r'$, (2) the leftmost positions of the sub-words can be identified from the fact that they do not belong to X but their successors do, and (3) the steps that \mathfrak{A} performs while reading the sub-word can be simulated by a first-order formula. This way, φ can check that the list of states represented by X is consistent with \mathfrak{A} 's transition relation and represents an accepting run of \mathfrak{A} . Q.E.D.

A closer look at this proof sketch shows that a similar set X can also be defined as a monadic least fixed-point of a suitable first-order formula: This time, sub-words of length $r' := 1 + 2r$ are considered, and each state $i \in \{1, \dots, r\}$ is represented by the bit-string $10^{i-1}10^{2r-i}$. Note that r' is chosen in such a way that the distance between two consecutive positions carrying the letter 1 tells us, which of the two positions marks the beginning of a sub-block and which of the two positions marks a state of the automaton. Using this, one obtains that every regular word language can be described by an MLFP-sentence which uses just a single monadic least fixed point operator (see Potthoff [30] for details). In a similar way, one can also prove that the modal mu-calculus can describe exactly the regular word languages. In summary, we thus have the following situation:

Theorem 3.4. On the class of words, MSO, EMSO, 1-EMSO, MLFP, and the modal mu-calculus have the same expressive power and can describe exactly the regular word languages.

The same result holds true for the class of labeled trees (cf. [35, 9, 30, 18]).

If we leave the classes of words and labeled trees and pass over to pictures, this is not the case any more. We will give details on this in the next subsection.

3.2 EMSO-Definability and Recognizability of Pictures

In [14], Giammarresi and Restivo suggested a natural adaptation of NFA to picture languages: the so-called *tiling-systems*.

Definition 3.5. A tiling-system is a quadruple $(\Sigma, \Gamma, \Delta, \pi)$, where Σ and Γ are finite alphabets, $\pi : \Gamma \rightarrow \Sigma$ is an *alphabet projection*, and Δ is a set of 2×2 -pictures over alphabet $\Gamma \cup \{\#\}$, where $\#$ is a fresh boundary symbol. The mapping π is lifted to pictures in the obvious way.

A picture p over Σ is *accepted* by such a tiling-system iff there is a picture r over Γ such that $\pi(r) = p$ and Δ contains all 2×2 -sub-blocks of the picture that results by surrounding r with the boundary symbol $\#$. The picture language *recognized* by some tiling-system T is the set of pictures accepted by T .

Example 3.6. Consider the tiling-system $T = (\{a\}, \{0, 1\}, \Delta, \pi)$, where $\pi(0) = \pi(1) = a$, and where Δ is the set of 2×2 -subblocks of

$$\begin{array}{cccccccc} \# & \# & \# & \# & \# & \# & \# & \# & \# \\ \# & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \# & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ \# & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ \# & \# & \# & \# & \# & \# & \# & \# & \# \end{array}$$

Then T recognizes the set of all pictures p over $\{a\}$ for which there exists $m \geq 1$ such that p has size $m \times 2^m$. Intuitively, T establishes a mechanism of binary counting the columns.

More examples of recognizable picture languages can be found in Giammarresi and Restivo's article in the present book.

Unlike the regular word languages, which are pretty simple to understand, the recognizable picture languages can be very complex, both from an intuitive and from a computational point of view. For example, in [31, 32], Reinhard has found examples of picture languages whose proofs of recognizability are very difficult and which disproved previous conjectures by Matz, e.g. [22]. Still, examples near the borderline between recognizable and non-recognizable picture languages are subject of current research, see [6].

It is known that the class of recognizable picture languages is closed under union, intersection, row- and column-concatenation, and row- and column-Kleene-star [14], but we have:

Theorem 3.7 (Giammarresi, Restivo, Seibert, Thomas [15]). If the alphabet has at least two symbols, the class of recognizable picture languages is not closed under complement.

A witness for the above theorem is given by:

Example 3.8 ([15]). Let L be the set of pictures over $\{0, 1\}$ that result from the concatenation of two identical pictures of quadratic shape. Then L is not recognizable, but its complement is.

The statement of Theorem 3.7 is true also for singleton alphabets, see Theorem 4.12 below.

MSO logic, of course, is closed under negation, so Theorem 3.7 immediately implies that the statement of Theorem 3.1 is not true when replacing the terms “word language” and “regular” with “picture language” and “recognizable”. However, it is known that the existential fragment of monadic second-order logic, EMSO, has exactly the right power for expressing recognizable picture languages:

Theorem 3.9 (Giammarresi, Restivo, Seibert, Thomas [15]). A picture language is recognizable if, and only if, it can be defined by an EMSO-sentence.

The “easy” direction in the proof is to show that recognizability by a tiling-system can be described by an EMSO-formula. This case can be handled in a similar way as in the proof of Theorem 3.1.

The other direction, however, cannot be handled in a similar way as in that proof because the initial replacement of first-order quantifiers by set quantifiers would force us to deal with the negation during the induction, but the class of recognizable picture languages is not closed under complement. Thus, one essential step in the proof given in [15] is a specific treatment of the first-order quantifiers with Ehrenfeucht-Fraïssé games. This yields, as a side-product, also the characterization of the *first-order* definable picture languages as the *locally threshold testable* ones, analogously to the one-dimensional case.

The characterization of the EMSO-definable picture languages given in Theorem 3.9 opened the door to several combinatorial arguments that allow to show that certain picture languages are not EMSO-definable, see for example [13, 22]. This was the basis for the original proof of the strictness of the monadic second-order quantifier alternation hierarchy [26], see Section 4.3 below.

Similarly as for words it is known that for defining recognizable picture languages, already a single existentially quantified set variable suffices:

Theorem 3.10 (Matz [23]). Over the class of pictures, every EMSO-sentence is equivalent to a 1-EMSO-sentence.

The proof is by an adaptation of the proof of Theorem 3.3 to the two-dimensional case: A tiling-system plays the role of the finite automaton \mathfrak{A} , with the minor technical inconvenience that tiling-systems are inherently

non-deterministic. However, the determinism of the automaton \mathfrak{A} in the proof of Theorem 3.3 is not essential.

Let us mention that by results of Otto [29] it is known that when going slightly beyond the class of pictures, EMSO does not collapse to 1-EMSO but, quite to the contrary, there is a strict hierarchy within EMSO with respect to the number of existentially quantified set variables. To precisely state Otto's result, let us write k -EMSO for the fragment of EMSO where k set variables are available. Instead of pictures, Otto considers particular structures over a signature which consists of two binary relation symbols R and C . These *Otto-grids* are structures whose universe forms a rectangular array and where R and C are interpreted by the relations stating that two vertices belong to the same row, respectively, the same column of the array.

Theorem 3.11 (Otto [29]). For every $k \geq 0$ there is a $(k+1)$ -EMSO-sentence that is not equivalent (over the class of Otto-grids) to any k -EMSO-sentence.

The proof is by showing that the set L_k of all Otto-grids with the property that the number of columns is $\leq 2^{(k+1) \cdot \text{number of rows}}$ is definable in $(k+1)$ -EMSO but not in k -EMSO (for the latter, an Ehrenfeucht-Fraïssé game argument is used).

To close the subsection on recognizable picture languages, let us have a quick look at the *computational complexity* of standard decision problems concerning recognizable picture languages.

Proposition 3.12 (Giammarresi, Restivo [14]). The *emptiness* problem for tiling-systems is undecidable.

Proof (sketch). We sketch a reduction of the emptiness problem for Turing machines. Let \mathfrak{A} be a Turing machine. It is straightforward to encode a configuration of \mathfrak{A} by a finite word over a fixed alphabet Σ . Each step in a computation of \mathfrak{A} corresponds to a local modification of that code. Every finite—and hence every accepting—run R of \mathfrak{A} can be encoded by a picture p over Σ , where p contains, in each row i , the code of the i -th configuration of R (possibly padded with blank symbols).

Now it is easy to effectively construct a tiling-system T that accepts all pictures that encode an accepting run. Then the language recognized by T is non-empty iff \mathfrak{A} has an accepting run. Q.E.D.

Furthermore, the *membership* problem for tiling-systems is NP-complete:

Proposition 3.13 (Schweikardt [33]). (a) The following problem belongs to NP: Given a tiling-system T and a picture p , does T accept p ?

- (b) There exists a tiling-system T such that the following problem is NP-complete: Given a picture p , does T accept p ?

Proof (sketch). The proof of (a) is straightforward. (b) is obtained by coding the (NP-complete) problem of satisfiability of propositional formulas in conjunctive normal form into an EMSO-definable picture language. To this end, each propositional formula α is represented by a picture which has a row for each variable and a column for each clause of α , such that the entry in row i and column j of the picture is labeled by the letter P (resp. N , resp. \ominus) if the i -th propositional variable occurs unnegated (resp. negated, resp. not at all) in the j -th clause of α . A *truth assignment* to the variables of α is represented by a set X of positions in the picture which, for each row, contains either none or all positions of that row. I.e., if the i -th propositional variable is assigned the value *true* (resp., *false*), then X contains *all* (resp. *none*) of the positions in the i -th row. It is not difficult to find an EMSO-formula ψ which expresses that there exists such a set X which encodes a satisfying assignment for α . Altogether, this gives us a reduction from the NP-complete satisfiability problem to the problem of deciding whether an input picture belongs to the picture language defined by ψ . Q.E.D.

It is current research interest to determine computationally feasible subclasses of recognizable picture languages, see e.g. the article of Giammarresi and Restivo in the present book.

3.3 Picture Languages Definable in MSO and MLFP

From the previous subsection we know that the EMSO-definable picture languages coincide with the 1-EMSO-definable and the recognizable picture languages. Furthermore, recall that Example 3.8 exposes a picture language that is not definable in EMSO. It is not difficult to see that this language is definable in MSO as well as in MLFP. The present subsection aims at a deeper understanding of the MSO-definable and the MLFP-definable picture languages.

Let us first concentrate on the MSO-definable picture languages. It is easy to see that the *membership* problem for each MSO-definable picture language belongs to LINH, i.e., the *linear time hierarchy* (cf., e.g. [10]), which is the linear time analogue to Stockmeyer's polynomial time hierarchy. On the other hand, it is not difficult to see that, in fact, the MSO-definable picture languages precisely correspond to the linear time hierarchy, since every decision problem that belongs to LINH can be encoded by an MSO-definable picture language. This can be obtained as follows: From [27] we know that LINH is the class of all word languages that can be defined in MSO(*Bit*), i.e., in monadic second-order logic on words where in addition

to the successor relation, also the *Bit predicate* on the set of positions in the word is available (the *Bit* predicate is the set of all tuples (i, j) such that the i -th bit in the binary representation of the natural number j is 1). The basic idea now is to represent a word of length n by a picture as follows: Let ℓ be the largest integer such that $n \geq \ell \cdot 2^\ell$, cut the word into sub-words of length 2^ℓ , and arrange the consecutive sub-words into consecutive rows of the resulting picture (if necessary, pad the last row with dummy entries to obtain a rectangular picture). Of course, the successor relation *Succ* of the original word can easily be simulated by an MSO-formula over the corresponding picture. Furthermore, it is a not too difficult exercise to also construct an MSO-formula over the picture which simulates the *Bit* predicate of the original word (*hint*: use an existentially quantified unary relation to encode a “column-numbering” which writes the binary representations of the numbers $0, 1, 2, \dots, 2^\ell - 1$ into the consecutive columns of the picture). It then is not difficult to see that every MSO(*Bit*)-definable set of strings is represented by an MSO-definable set of pictures. In this sense, the MSO-definable picture languages can encode all problems that belong to the linear time hierarchy.

Let us now concentrate on the MLFP-definable picture languages. Of course, for each picture language defined by a fixed MLFP-sentence, the membership problem belongs to P. Together with Proposition 3.13 and the fact that the expressive power of MLFP lies between FO and MSO, this implies the following:

Fact 3.14. Unless $P = NP$, MLFP is strictly less expressive on the class of pictures than MSO.

On the other hand, MLFP is still quite expressive as it can define picture languages corresponding to every problem in the deterministic linear time complexity class DLIN introduced by Grandjean in [16]. The class DLIN is based on linear time random access machines. In a series of papers, Grandjean made a convincing point that DLIN might be viewed as “the” adequate mathematical formalization of linear time complexity. For example, DLIN contains all problems in $\text{DTIME}(n)$, i.e., all problems solvable by deterministic linear time multi-tape Turing machines; but DLIN also contains problems such as the *sorting* problem, which are conjectured not to belong to $\text{DTIME}(n)$.

In a similar way as described above for MSO and LINH, one obtains that every problem in DLIN can be encoded by an MLFP-definable picture language—instead of using the characterization of LINH as the MSO(*Bit*)-definable word languages, one now just has to use a result from [34] stating that every word language which belongs to DLIN can be defined by an MLFP(*Bit*)-sentence.

4 Alternation Hierarchies

In descriptive complexity theory it is a general task to classify properties by the complexity a formula must have to describe this property. But what is the suitable measure for the complexity of a formula? A typical approach is to measure the complexity by the nesting depth of the “most powerful ingredient” of the logic under consideration.

For example, a measurement for the complexity of a first-order formula is the nesting depth of first-order quantifiers, neglecting the complexity introduced by boolean combinations. Another example is the modal mu-calculus, where it is the nesting depth of fixpoint iterations that is the natural means to measure the complexity of a formula. MSO is a third example, where the nesting depth of the most powerful quantifications (in this case, the monadic ones) establishes a measure of formula complexity.

In Section 3 we have already considered the nesting depth of set quantifiers as a complexity measure of MSO-formulas and have seen (Theorem 3.3) that the corresponding hierarchy collapses for the classes of words and of trees whereas it is infinite for Otto-grids (Theorem 3.11).

However, for many logics and classes of structures, the complexity measurement obtained by simply counting syntactic nesting of single quantifiers is (1) not sufficiently robust, and (2) does not result in the natural parameters for the computational complexity, e.g. of the model checking or the satisfiability problem of formulas.

To illustrate the first reason, consider two 1-EMSO-sentences on the class of finite structures. Their conjunction is in 2-EMSO but, unlike their disjunction, in general not in 1-EMSO, so that the class of 1-EMSO-definable properties is not necessarily closed under intersection.

To illustrate the second reason, let us consider MSO over words. A good approach for solving the model checking problem relies on the well-known construction of an NFA for a given MSO-formula (see Theorem 3.1 and its proof sketch). The constructions for conjunction, disjunction, and existential quantification can be done directly on NFA and result in no essential increase of the number of states. However, the construction for the negation of a formula requires a deterministic automaton and therefore the famous powerset construction, which results in an exponential state blow-up. Thus it is the *alternation* of existential quantifications and negation (or, equivalently: the alternation of existential and universal quantifications) that is significant for the increase of the state set size and therefore for the computational complexity of the model checking problem.

4.1 First-Order Alternation

As motivated above, one passes to a coarser view of “nesting” by considering a block of only existential (or only universal) quantifiers as one single,

“vectorial” quantifier. This vectorial approach is the basis for the *first-order quantifier alternation hierarchy*. For example, a property of finite labeled graphs is in the third level of that hierarchy iff it can be defined by a first-order formula that has a prenex normal form with a quantifier prefix of type

$$\exists^* \forall^* \exists^*,$$

i.e., a quantifier prefix with three blocks of first-order quantifications, starting with an existential one, and the following kernel formula is quantifier-free. Level k of the first-order quantifier alternation hierarchy is usually denoted Σ_k^0 , its “complement” Π_k^0 (i.e., Π_k^0 is the set of all graph properties that can be defined by a first-order formula in prenex normal form that has a quantifier prefix with k blocks of first-order quantifications, starting with a universal one).

Theorem 4.1 (Chandra, Harel [7]; Thomas [36]). The first-order quantifier alternation hierarchy is strict over the class of finite labeled graphs, i.e., for every $k \geq 0$, $\Sigma_k^0 \subsetneq \Sigma_{k+1}^0$. Furthermore, for every $k \geq 1$, $\Sigma_k^0 \neq \Pi_k^0$.

Chandra and Harel’s proof in [7] explicitly provides, for each $k \geq 0$, a property of finite labeled directed graphs that belongs to Σ_{k+1}^0 but not to Σ_k^0 . They consider graphs that are equipped with a distinguished “start node” and a subset of nodes called “winning positions”. With each such graph, they associate a 2-player game in which a token is moved along the edges of the graph. At the beginning, the token is placed on the “start node”. The players take turns, starting with player 1, and in each move one of the players moves the token along an edge of the graph. After $k+1$ such moves, player 1 has *won* the game, if the token lies on a “winning position”. It is now easy to find a Σ_{k+1}^0 -sentence which expresses that player 1 has a winning strategy for $k+1$ moves; and by an Ehrenfeucht-Fraïssé game argument it can be shown that this cannot be expressed by any Σ_k^0 -sentence.

A different proof of the strictness of the first-order quantifier alternation hierarchy is given in [36], where Wolfgang Thomas considers first-order formulas over word models with a different signature than in the present paper, namely with the ordering $<$ instead of the successor relation on the word positions. He shows that the first-order quantifier alternation hierarchy over that signature corresponds to the *dot-depth alternation hierarchy*, which is shown to be strict in [8].

However, for words, trees, and pictures (over the signatures introduced in Section 2.1, i.e., without ordering but with successor relation(s)), the first-order quantifier alternation hierarchy collapses to boolean combinations of its first level. This is a consequence of the characterization of first-order definable properties of words, trees, and pictures by *local threshold*

testability, cf., e.g., the survey [37] and the article [15].

4.2 Fixpoint Alternation in the mu-Calculus

Niwiński [28] introduced vectorial fixpoints to result in a sufficiently coarse and robust definition for the modal mu-calculus fixpoint alternation hierarchy which relies on the number of alternations of least and greatest fixed point quantifiers—see [3] for a detailed discussion of that subject.

Theorem 4.2 (Bradfield [3]). The modal mu-calculus alternation hierarchy is strict over the class of finite labeled graphs, i.e., for every $k \geq 0$, there is a property of finite labeled graphs that is definable in level $k+1$ of the Niwiński alternation hierarchy of the modal mu-calculus, but not in level k .

In [20], Lenzi proved a corresponding but slightly weaker result referring to a different variant of fixpoint alternation, the Emerson-Lei hierarchy. An elegant proof of Bradfield’s and Lenzi’s hierarchy was given by Arnold in [2]. Let us mention that the hierarchies proved in [3, 20, 2] are about general structures that are not necessarily finite; via the mu-calculus’ *finite model property* (cf., e.g., [4]), however, they can be directly transferred to the class of finite labeled graphs.

On the other hand, when considering the class of finite words (instead of the class of finite labeled graphs), the modal mu-calculus alternation hierarchy is known to collapse (this can be proved in a similar way as discussed in the paragraph before Theorem 3.4). More details on the collapse of the modal mu-calculus hierarchy on particular classes of structures can be found in [21, 38, 19].

It is a challenging future task to settle the following question:

Question 4.3. Does a similar result as Theorem 4.2 hold for monadic least fixed point logic MLFP instead of the modal mu-calculus? I.e., is there a strict hierarchy within MLFP that is based on the number of alternations of least and greatest fixed point quantifiers?

4.3 Monadic Second-Order Logic

Let us now consider monadic second-order logic MSO. In that logic, the most powerful ingredient is the set quantification. The quantifier structure of an MSO-formula in prenex normal form can be represented by a word over the four-element alphabet $\{\exists, \forall, \exists, \forall\}$, where \exists, \forall represent set quantifiers, and \exists, \forall represent first-order quantifiers. In the following, we use regular expressions over that alphabet to describe quantifier prefixes of formulas in prenex normal form.

Every MSO-formula is equivalent (over the class of all structures) to an MSO-formula whose quantifier prefix is of type

$$\{\exists, \forall\}^* \{\exists, \forall\}^*.$$

A transformation of a given MSO-formula ψ into the above form can be done in three steps: Firstly, replace every sub-formula of the form $\exists x \varphi(x)$ with an equivalent formula of the form $\exists X (\text{singleton}(X) \wedge \varphi'(X))$, where $\text{singleton}(X)$ is an auxiliary first-order formula asserting that X is a singleton, and where φ' results from φ by replacing every atomic formula $\alpha(x_1, \dots, x_n)$ with a suitable auxiliary first-order formula $\alpha'(X_1, \dots, X_n)$. Note that the resulting formula ψ' contains first-order quantifiers only within the new auxiliary formulas $\text{singleton}(X)$ and $\alpha'(X_1, \dots, X_n)$. Secondly, transform ψ' into prenex normal form, treating the auxiliary formulas like atoms. Now, viewing the auxiliary formulas again as first-order formulas, the resulting MSO-formula ψ'' obviously consists of a quantifier prefix of set quantifiers that is followed by a first-order formula. By transforming the first-order part of this formula into prenex normal form, one then obtains an MSO-formula in prenex normal form whose quantifier prefix is of type $\{\exists, \forall\}^* \{\exists, \forall\}^*$.

4.3.1 The MSO Quantifier Alternation Hierarchy

The definition of the *monadic second-order quantifier alternation hierarchy* (or “MSO alternation hierarchy” for short) is based on the above representation. For each $k \geq 0$, level k of this hierarchy consists of those properties (of, say, finite labeled graphs) that can be defined by an MSO-formula in prenex normal form where the set quantifiers are grouped into k blocks, existential and universal in alternation, starting with an existential one. While most parts of Section 3 are devoted to EMSO, the first level of this hierarchy, we consider the higher levels now. For example, a property is in level three of that hierarchy iff it can be defined by a formula in prenex normal form of type

$$\exists^* \forall^* \exists^* \{\exists, \forall\}^*.$$

i.e., one that starts with three blocks of set quantifiers, the first one being existential, and continues with a first-order kernel formula.

Let us denote level k of the MSO quantifier alternation hierarchy by $\text{mon-}\Sigma_k^1$, its “complement” by $\text{mon-}\Pi_k^1$ (i.e., $\text{mon-}\Pi_k^1$ consists of all graph properties whose complement belongs to $\text{mon-}\Sigma_k^1$), and their intersection by $\text{mon-}\Delta_k^1$. Furthermore, we write $BC(\text{mon-}\Sigma_k^1)$ to denote the class of all properties that can be defined by a boolean combination of sentences suitable for $\text{mon-}\Sigma_k^1$. (Thus $BC(\text{mon-}\Sigma_k^1)$ is the smallest superclass of $\text{mon-}\Sigma_k^1$ that is closed under union and complement.)

By slightly abusing notation, we will sometimes also speak of $\text{mon-}\Sigma_k^1$ *formulas* to address the particular kind of formulas suitable for defining properties that belong to $\text{mon-}\Sigma_k^1$.

Fagin has shown that *connectivity* of finite graphs is (analogously to Example 3.8) definable by a sentence in prenex normal form of type $\forall^* \{\exists, \forall\}^*$, but not by one of type $\exists^* \{\exists, \forall\}^*$. This leads to the following result:

Theorem 4.4 (Fagin [12]). $mon-\Sigma_1^1 \neq mon-\Pi_1^1$ and thus, in particular, $mon-\Sigma_1^1 \subsetneq mon-\Sigma_2^1$.

Fagin raised the question whether the MSO quantifier alternation hierarchy collapses on some higher level. The question has been answered negatively in [26]. Refining that proof, [33, 25] shows that a witness for the separation of level $k+1$ from level k is the set of all pictures of size $m \times f(m)$ for a specific $(k+1)$ -fold exponential function: this picture language is definable by a sentence with $k+1$ alternations of set quantifiers, but not by one with just k alternations of set quantifiers. The same witness even separates $mon-\Delta_{k+1}^1$ from $BC(mon-\Sigma_k^1)$. Using standard techniques, the results can be transported to the class of graphs. We thus obtain

Theorem 4.5 (Matz, Schweikardt, Thomas [25]). For every $k \geq 0$, $mon-\Sigma_k^1 \subsetneq mon-\Sigma_{k+1}^1$. Moreover, there even exists a picture language over a singleton alphabet that belongs to $mon-\Delta_{k+1}^1$ but not to $BC(mon-\Sigma_k^1)$.

However, the proof of this theorem has also exhibited the following: it is *not* the alternation of set quantifiers that gives the expressive power needed to leave a fixed level of that hierarchy—it is the nesting of *first-order* quantifiers, followed by one single block of set quantifiers. For example, there is an MSO-sentence with quantifier prefix of type

$$\forall^* \exists^* \forall^* \{\exists, \forall\}^*,$$

that is not equivalent to any sentence with quantifier prefix of type

$$\exists^* \forall^* \exists^* \{\exists, \forall\}^*$$

(and likewise for values larger than three).

How is this possible? The definition of the MSO quantifier alternation hierarchy allows to neglect first-order quantifications inside the kernel formula, but it does not allow to neglect first-order quantifications completely. This is so because first-order quantifications do not factor through *monadic* second-order quantifications, unlike for the full second-order logic, in which quantification is available over relations of arbitrary arity. We will take a closer look at this phenomenon in the following paragraph.

4.3.2 The Closed MSO Hierarchy

As motivated above, the value of the strictness of the MSO quantifier alternation hierarchy would be much higher if first-order quantification was, by definition, neglectable. This point was made by Ajtai, Fagin, and Stockmeyer in [1]. In that paper, the authors suggest the *closed MSO alternation hierarchy*, which is coarser and more robust than the ordinary MSO alternation hierarchy because it allows to intersperse first-order quantifiers “for

free” between set quantifiers. For example, a property is in level three of that hierarchy iff can be defined by an MSO-formula which has a prenex normal form of type

$$\{\exists, \exists, \forall\}^* \{\forall, \exists, \forall\}^* \{\exists, \exists, \forall\}^* \{\exists, \forall\}^* .$$

As noted in [1], the strictness of the *closed MSO alternation hierarchy* would be implied by the conjectured strictness of the polynomial time hierarchy, because each level of the latter is closed under first-order quantification and each level of the MSO alternation hierarchy contains a complete problem for the polynomial time hierarchy. The following is a challenging future task:

Task 4.6. Show, without relying on complexity theoretic assumptions, that the closed MSO alternation hierarchy is strict.

4.3.3 The First-Order Closure

As pointed out above, it is desirable to understand more about the role of first-order quantifications in the context of monadic second-order quantifier alternation. Let us mention two approaches that have been made to achieve progress in this area. Both deal with the *first-order closure* of some subclass \mathcal{L} of MSO, meaning the smallest superset of \mathcal{L} that is closed under first-order quantification and boolean combinations.

In [17], the authors develop a technique to infer new separation results dealing with the first-order closure. Specifically, they show the following:

Theorem 4.7 (Janin, Marcinkowski [17]). Let $V, W \subseteq \{\exists, \forall, \exists, \forall\}^*$. Let S be a graph property definable by a prenex normal form of type V but not by one of type W , then there is another property definable by a prenex normal form of type $\exists \forall V$ but not by one of type $\{\exists, \forall\}^* W$.

This technique works for the class of graphs, but it does not work for the classes of words, trees, or pictures. The authors of [17] apply it to show the following corollary (previously shown directly in [1]).

Corollary 4.8. There exists a graph property definable by a prenex normal form of type $\exists^* \{\exists, \forall\}^* \exists^* \{\exists, \forall\}^*$ but not with one of type $\{\exists, \forall\}^* \exists^* \{\exists, \forall\}^*$.

Apart from this, not many separation results are known by now. In fact, to our best knowledge, even the following remains open:

Question 4.9. Is every MSO-formula equivalent to one of the form

$$\exists^* \{\exists, \forall\}^* \exists^* \{\exists, \forall\}^* \quad ?$$

For the class of pictures, [24] contains another preliminary step towards understanding the expressive power of the first-order closure of logics. In that paper, the *MSO alternation hierarchy with first-order closure* is considered. A property belongs to level k of that hierarchy iff it is definable in the first-order closure of the set of $mon\text{-}\Sigma_k^1$ formulas.

Theorem 4.10 (Matz [24]). The MSO alternation hierarchy with first-order closure is strict.

The proof shows, for example, that there is a prenex normal form of type $\forall^* \exists^* \forall^* \exists^* \forall^* \{ \exists, \forall \}^*$ that is not equivalent to a prenex normal form of type $\{ \exists, \forall \}^* \exists^* \forall^* \exists^* \{ \exists, \forall \}^*$. That means, to exceed some level of the MSO alternation hierarchy with first-order closure, only *two* blocks of set quantifiers are needed.

4.4 Labels and Complement

Let us review the mentioned results and see what they imply concerning the question whether the levels of the MSO quantifier alternation hierarchy are closed under complement. Theorem 4.5 considers the class of picture languages over a singleton alphabet and shows that, for every k , there is a picture language that belongs to level $k+1$, but not to level k of the MSO alternation hierarchy. This implies

Corollary 4.11. For every $k \geq 1$ there exists a $t \geq 0$ such that there is a picture language over alphabet $\Sigma := \{0, 1\}^t$ which belongs to $mon\text{-}\Sigma_k^1$ but not to $mon\text{-}\Pi_k^1$.

By standard encoding techniques it can be deduced that $t = 1$ suffices. In other words, if the alphabet Σ is fixed and of size ≥ 2 , then all separation results of Figure 4.4 hold. Even more, the above is true also for a singleton alphabet, so Theorem 3.7 can be generalized to:

Theorem 4.12 (Matz [24]). For every $k \geq 1$ there is a picture language over a singleton alphabet which belongs to $mon\text{-}\Sigma_k^1$ but not to $mon\text{-}\Pi_k^1$.

A picture language which witnesses the difference between $mon\text{-}\Sigma_k^1$ and $mon\text{-}\Pi_k^1$ is the set of all pictures of size $m \times n$ for which n is not a multiple of $f(m)$, where f is a specific $(k+1)$ -fold exponential function.

Again, the witness sentence actually makes little use of set quantifiers. For example, if $k = 5$, it is of the form

$$\exists^* \forall^* \exists^* \forall^* \exists^* \{ \exists, \forall \}^*.$$

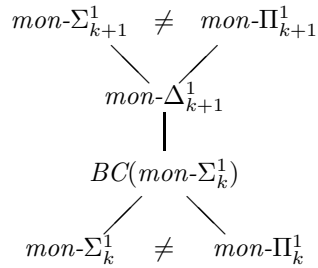


FIGURE 1. The MSO quantifier alternation hierarchy

5 Acknowledgments

Both authors wish Wolfgang Thomas all the best for this jubilee. Matz would like to express his gratitude for Wolfgang Thomas' careful supervision during the preparation of Matz' Ph.D. thesis. Furthermore, we would like to thank the anonymous referee for the detailed remarks.

References

- [1] M. Ajtai, R. Fagin, and L. Stockmeyer. The closure of monadic NP. *Journal of Computer and System Sciences*, 60(3):660–716, 2000. Journal version of STOC'98 paper.
- [2] A. Arnold. The μ -calculus alternation-depth hierarchy is strict on binary trees. *RAIRO – Theoretical Informatics and Applications (ITA)*, 33:329–339, 1999.
- [3] J. Bradfield. The modal mu-calculus alternation hierarchy is strict. *Theoretical Computer Science*, 195(2):133–153, 1998. Journal version of CONCUR'96 paper.
- [4] J. Bradfield and C. Stirling. Modal logics and mu-calculi: An introduction. In J. Bergstra, A. Ponse, and S. Smolka, editors, *Handbook of Process Algebra*, pages 293–330. Elsevier Science, 2001.
- [5] J. Büchi. Weak second-order arithmetic and finite automata. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik*, 6:66–92, 1960.
- [6] G. Castiglione and R. Vaglica. Recognizable picture languages and polyominoes. In *Conference of Algebraic Informatics*, Thessaloniki, Greece, 2007. (To appear).

- [7] A. K. Chandra and D. Harel. Structure and complexity of relational queries. *Journal of Computer and System Sciences*, 25(1):99–128, 1982. Journal version of FOCS’80 paper.
- [8] R. S. Cohen and J. A. Brzozowski. Dot-depth of star-free events. *Journal of Computer and System Sciences*, 5(1):1–16, 1971.
- [9] J. Doner. Tree acceptors and some of their applications. *Journal of Computer and System Sciences*, 4:405–451, 1970.
- [10] A. Durand and M. More. Nonerasing, counting, and majority over the linear time hierarchy. *Information and Computation*, 174(2):132–142, 2002.
- [11] C. Elgot. Decision problems of finite automata design and related arithmetics. *Transactions of the American Mathematical Society*, 98:21–52, 1961.
- [12] R. Fagin. Generalized first-order spectra and polynomial-time recognizable sets. In R. Karp, editor, *Complexity of Computation*, volume 7 of *SIAM-AMS-Proceedings*, pages 43–73, 1974.
- [13] D. Giammarresi. Two-dimensional languages and recognizable functions. In *Proceedings of Developments in Language Theory (DLT’93)*, pages 290–301, 1994.
- [14] D. Giammarresi and A. Restivo. Recognizable picture languages. *International Journal of Pattern Recognition and Artificial Intelligence*, 6(2-3):241–256, 1992.
- [15] D. Giammarresi, A. Restivo, S. Seibert, and W. Thomas. Monadic second-order logic and recognizability by tiling systems. *Information and Computation*, 125(1):32–45, 1996. Journal version of STACS’94 paper.
- [16] E. Grandjean. Sorting, linear time, and the satisfiability problem. *Annals of Mathematics and Artificial Intelligence*, 16:183–236, 1996.
- [17] D. Janin and J. Marcinkowski. A toolkit for first order extensions of monadic games. In *Proceedings of 18th Annual Symposium on Theoretical Aspects of Computer Science (STACS’01)*, volume 2010 of *Lecture Notes in Computer Science*, pages 353–364. Springer-Verlag, 2001.
- [18] D. Janin and I. Walukiewicz. On the expressive completeness of the propositional mu-calculus with respect to monadic second order logic. In U. Montanari and V. Sassone, editors, *CONCUR*, volume 1119 of *Lecture Notes in Computer Science*, pages 263–277. Springer, 1996.

- [19] O. Kupferman and M. Y. Vardi. Weak alternating automata and tree automata emptiness. In *Proceedings of the 30th ACM Symposium on Theory of Computing (STOC'98)*, pages 224–233, 1998.
- [20] G. Lenzi. A hierarchy theorem for the μ -calculus. In *Proceedings of 23rd International Colloquium on Automata, Languages and Programming (ICALP'96)*, volume 1099 of *Lecture Notes in Computer Science*, pages 87–97. Springer-Verlag, 1996.
- [21] R. Mateescu. Local model-checking of modal mu-calculus on acyclic labeled transition systems. In J.-P. Katoen and P. Stevens, editors, *TACAS*, volume 2280 of *Lecture Notes in Computer Science*, pages 281–295. Springer, 2002.
- [22] O. Matz. On piecewise testable, starfree, and recognizable picture languages. In *Proceedings of 1st International Conference on Foundations of Software Science and Computation Structures (FoSSaCS'98)*, volume 1378 of *Lecture Notes in Computer Science*, pages 203–210. Springer-Verlag, 1998.
- [23] O. Matz. One quantifier will do in existential monadic second-order logic over pictures. In *Proceedings of 23rd International Symposium on Mathematical Foundations of Computer Science (MFCS'98)*, volume 1450 of *Lecture Notes in Computer Science*, pages 751–759. Springer-Verlag, 1998.
- [24] O. Matz. Dot-depth, monadic quantifier alternation, and first-order closure over grids and pictures. *Theoretical Computer Science*, 270(1-2):1–70, 2002.
- [25] O. Matz, N. Schweikardt, and W. Thomas. The monadic quantifier alternation hierarchy over grids and graphs. *Information and Computation*, 179(2):356–383, 2002.
- [26] O. Matz and W. Thomas. The monadic quantifier alternation hierarchy over graphs is infinite. In *Proceedings of 12th Annual IEEE Symposium on Logic in Computer Science (LICS'97)*, pages 236–244. IEEE, 1997.
- [27] M. More and F. Olive. Rudimentary languages and second order logic. *Mathematical Logic Quarterly*, 43:419–426, 1997.
- [28] D. Niwiński. On fixed-point clones (extended abstract). In *Proceedings of 13th International Colloquium on Automata, Languages and Programming (ICALP'86)*, volume 226 of *Lecture Notes in Computer Science*, pages 464–473. Springer, 1986.

- [29] M. Otto. Note on the number of monadic quantifiers in monadic Σ_1^1 . *Information Processing Letters*, 53(6):337–339, 1995.
- [30] A. Potthoff. *Logische Klassifizierung regulärer Baumsprachen*. PhD thesis, Christian-Albrechts-Universität zu Kiel, Germany, 1994.
- [31] K. Reinhardt. On some recognizable picture-languages. In *Proceedings of 23rd International Symposium on Mathematical Foundations of Computer Science (MFCS'98)*, volume 1450 of *Lecture Notes in Computer Science*, pages 760–770. Springer-Verlag, 1998.
- [32] K. Reinhardt. The $\#a = \#b$ pictures are recognizable. In *Proceedings of 18th Annual Symposium on Theoretical Aspects of Computer Science (STACS'01)*, volume 2010 of *Lecture Notes in Computer Science*, pages 527–538. Springer-Verlag, 2001.
- [33] N. Schweikardt. The monadic quantifier alternation hierarchy over grids and pictures. In *Proceedings of 11th International Workshop on Computer Science Logic (CSL'97)*, volume 1414 of *Lecture Notes in Computer Science*, pages 441–460. Springer-Verlag, 1997.
- [34] N. Schweikardt. On the expressive power of monadic least fixed point logic. *Theoretical Computer Science*, 350(2-3):325–344, 2006. Journal version of ICALP'04 paper.
- [35] J. Thatcher and J. Wright. Generalized finite automata with an application to a decision problem of second-order logic. *Mathematical Systems Theory*, 2(1):57–81, 1968.
- [36] W. Thomas. Classifying regular events in symbolic logic. *Journal of Computer and System Sciences*, 25(3):360–376, 1982.
- [37] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3. Springer-Verlag, 1996.
- [38] I. Walukiewicz. Notes on the propositional μ -calculus: Completeness and related results. Technical Report NS-95-1, BRICS, Department of Computer Science, University of Aarhus, Denmark, 1995.