Relativizations for the Logic-Automata Connection*

NILS KLARLUND†
AT&T Labs, Research, Florham Park, New Jersey

Dedicated to the memory of Bob Paige and his contributions to automata algorithms

Abstract. BDDs and their algorithms implement a decision procedure for Quantified Propositional Logic. BDDs are a kind of acyclic automata. But unrestricted automata (recognizing unbounded strings of bit vectors) can be used to decide monadic second-order logics, which are more expressive. Prime examples are WS1S, a number-theoretic logic, or the string-based logical notation of introductory texts. One problem is that it is not clear which one is to be preferred in practice. For example, it is not known whether these two logics are computationally equivalent to within a linear factor, that is, whether a formula φ of one logic can be transformed to a formula φ′ of the other such that φ′ is true if and only if φ is and such that φ′ is decided in time linear in that of the time for φ.

Another problem is that first-order variables in either version are given automata-theoretic semantics according to relativizations, which are syntactic means of restricting the domain of quantification of a variable. Such relativizations lead to technical arbitrations that may involve normalizing each subformula in an asymmetric manner or may introduce spurious state space explosions.

In this paper, we investigate these problems through studies of congruences on strings. This algebraic framework is adapted to language-theoretic relativizations, where regular languages are intersected with restrictions. The restrictions are also regular languages. We introduce ternary and sexpartite characterizations of relativized regular languages. From properties of the resulting congruences, we are able to carry out detailed state space analyses that allow us to address the two problems.

We report briefly on practical experiments that support our results. We conclude that WS1S with first-order variables can be robustly implemented in a way that efficiently subsumes string-based notations.

Keywords:

1. Motivation

The relationship between automata and logic has been very successfully exploited through Binary Decision Diagrams [4]. This technique allows formulas of propositional logic to be decided through the use of automata representations for sets of strings of bounded length. But, a more general logic-automata connection exists: Büchi [5], Elgot [6], and Trakhtenbrot [17] argued forty years ago that a logical notation, now called the Weak Second-order theory of 1 Successor or WS1S, would be a more natural alternative to what already was known as regular expressions. WS1S has an extremely simple syntax and semantics: it is a variation of predicate logic with first-order variables that denote natural

* Some of the material in this paper appeared in Computer Aided Verification, CAV ’99, LNCS 1633, 1999, under the title “A theory of restrictions for logics and automata.” This work was carried out while the author was at AT&T Labs–Research; it was also supported in part by grant CCR-0341658 from the National Science Foundation.
† Present address: Bell Labs, Lucent Technologies, Murray Hill, New Jersey
numbers and second-order variables that denote finite sets of natural numbers; it has a single function symbol, which denotes the successor function, and has usual binary operators such as $\leq$, $=$, $\in$ and $\supseteq$.

Büchi, Elgot, and Trakhtenbrot showed that a decision procedure exists for this logic. The idea is to view interpretations as finite strings over bit vectors and then to show by explicit constructions of automata that the set of satisfying interpretations for any subformula is a regular language. In this way, an automaton becomes an object that represents the logical semantics of a subformula, and it makes sense to talk about automata-theoretic semantics, which characterizes the computational approach to the logic. As with Binary Decision Diagrams, the idea behind the decision procedure is to construct inductively a deterministic automaton for each subformula. This method, which we shall review in detail, handles each connective in the logic through an automata-theoretic operation, such as product or subset construction. To decide a sentence is then under this view the process of building the automaton inductively.

A main motivation of this article is to make the decision procedure feasible in practice. Since 1994, we have explored the practicality of the logic-automata connection in the Mona project, first described in [7]. The Mona tool has been used for a variety of tasks, for example in linguistics [18], pointer verification [13], protocol verification [14], and hardware verification [2]. Among other implementation challenges [12], we discovered spurious state space explosions in intermediate automata. Sometimes, we would be prevented from solving even trivial problems due to phenomena that we will uncover and overcome in the present article.

1.1. The two approaches to a logic of one successor

The problems we encountered are in part linked to the existence of two formulations of monadic second-order logic of one successor. Using the same syntax, the two approaches depend on different structures and different semantics.

The logic WS1S, the first approach, is a very natural notation. For example, a sentence in WS1S is either true or false and a formula with $k$ free first-order variables defines a relation that is simply a subset of $\mathbb{N}^k$, where $\mathbb{N}$ is the set of non-negative integers. Thus, we call this approach to the logic-automata connection number-theoretic. The automaton for a sentence is a simple one: it has one state (if minimized)! If it is accepting, the formula is true; if rejecting, the formula is false.

The second approach, the one emphasized in presentations of the logic-automata connection (such as in [15, 16]) is more complicated to explain, at least when it comes to the semantics, which is tied to a parameterized, finite, unbounded domain represented by a number $n \geq 0$. This number defines a set $\{0, \ldots, n-1\}$ of positions. All second-order terms are interpreted as subsets of $\{0, \ldots, n-1\}$ and all first-order terms are interpreted as numbers in $\{0, \ldots, n-1\}$. Under this view, the truth status of a sentence depends on $n$. The successor function denoted by the term $p + 1$ is interpreted as ordinary addition except if $p$ is $n - 1$; in that case, we may choose—arbitrarily—to define the meaning of $p + 1$ to be $p$.

For example, a sentence may be defined that is true if and only if $n$ is even; intuitively, the sentence expresses:

- there a set $P$ that is maximal (not properly contained in any other set);