Acta Informatica 20, 207-226 (1983)

The Temporal Logic of Branching Time* **

Mordechai Ben-Ari[1], Amir Pnueli[2], and Zohar Manna[2,3]

[1] Department of Computer Science, School of Mathematical Sciences, Tel Aviv University, Ramat Aviv 69978, Tel Aviv, Israel
[2,3] Department of Applied Mathematics, The Weizmann Institute of Science, Rehovot, Israel 76100
[3] Department of Computer Science, Stanford University, Stanford, CA 94305, USA

Summary. A temporal logic is defined which contains both linear and branching operators. The underlying model is the tree of all possible computations. The following metatheoretical results are proven: 1) an exponential decision procedure for satisfiability; 2) a finite model property; 3) the completeness of an axiomatization.

1. Introduction

From the first introduction of the Temporal Logic formalism as a tool for reasoning about programs, there arose a basic question which later almost developed into a controversy. The question involves the nature of the underlying structure of time on which the formalism is based. The dichotomy is between the linear time approach which considers time to be a linear sequence, and the branching time approach, which adopts a tree structured time, allowing some instants to have more than a single successor.

The difference in approaches has very little to do with the philosophical question of the structure of physical time which leads to the metaphysical problems of determinancy versus free will.

Instead it is pragmatically based on the choice of the type of programs and properties one wishes to formalize and study.

Linear time is the correct model to use in order to characterize the set of all execution sequences which a program generates and to study properties which uniformly hold for all the execution sequences of a program. Even a

---

* Portions of this work are based on the first author's Ph.D. thesis done at the Tel Aviv University under the supervision of the second author. A Preliminary version of this paper was presented at the Eighth ACM Symposium on Principles of Programming Languages, Williamsburg, VA, 1981

** This research was supported in part by: NSF under grant MCS-80-6930, Office of Naval Research under grant N000-14-76-C-0687, the United States Air Force Office of Scientific Research under Grant AFOSR-81-0014, and the Israeli Academy of Sciences and Humanities, the Basic Research Foundation

nondeterministic program generates for each set of possible choices a linear execution sequence in which each execution state has a unique successor. The class of properties related to the set of execution sequences are the *universal* properties such as universal total correctness and universal responsiveness. Both these properties require that *every* execution sequence of the program will eventually achieve some goal such as termination with a correct result or correct response to some request. The interpretation of temporal formulas over execution sequences of a given program was found to be very useful for reasoning about both sequential deterministic programs and concurrent programs. In the case of concurrent programs, where the nondeterminism is caused by different scheduling scripts, we generally wish to prove that the program terminates or responds correctly regardless of how the individual processes are scheduled. This approach is pursued in [9, 11, 12].

The branching time approach, on the other hand, considers for a given program the set of all *execution trees* generated by the program. With a nondeterministic program P and a given input x we can associate the tree of all possible computations of P on x. Since the program is nondeterministic, some of the execution states will have more than one successor corresponding to a nondeterministic choice. Over execution trees we can study *existential* properties such as correct termination for at least one possible computation (for every input). More generally, we may study the property that there is always one possible computation which realizes some goal. This certainly does not imply that all computations will realize the same goal. Consequently, this approach is useful for nondeterministic programs which are executed by systematically exploring all possible choices by methods such as breadth first search, etc. This interpretation of nondeterminism is recommended for example in [4] as a design tool and is the one classically used in automata and complexity theory. The branching time approach is implied in the underlying structure of Dynamic Logic ([1, 6]), but was not previously studied in a temporal framework.

In the end, the choice between linear and branching models cannot be made on philosophical grounds but instead should be dictated by the type of programs, execution policies and properties which one wishes to study. For a fuller discussion of this issue see [9].

A natural step at this point would be to formalize and investigate a branching time temporal logic which incorporates both universal operators similar to those used in linear temporal logic, and existential operators similar to those used in simpler branching systems. It turns out that such a unified system which combines both approaches is not significantly more complex than the two separate systems.

We will define a logic $\mathcal{UB}$: the *unified* system of *branching* time. The underlying model will be the branching tree of all possible computations of a program. We define, however, additional temporal operators that allow reference either to all possible execution sequences or only to a single sequence. The metatheoretical results in $\mathcal{UB}$ include:

1) An exponential decision procedure for satisfiability.

2) A finite model property: If a formula in $\mathcal{UB}$ is satisfiable then it has a finite (exponential) model.