ABSTRACT

This paper applies and extends to (discrete propositional linear) interval logic results obtained in previous papers on the correspondence between extended discrete propositional linear temporal logic and sequential machines with infinite input/output. The machines obtained enable us to derive decision procedures for certain kinds of interval logics. One system of interval logic is considered, and its associated system of machines is constructed. With this construction we also obtain a kind of translation between this interval logic and extended discrete propositional linear temporal logic.

1. INTRODUCTION

Temporal logic has emerged over the last decade as a very useful tool in the general area of program specification and verification. Temporal logic was found especially useful when applied to the specification, verification and synthesis of concurrent processes and communication protocols (see e.g. [1], [2], [3], [4], [5]). Also, it is gaining interest in the community of information system researchers for the specification and verification of dynamic database constraints or as a modelling tool for temporal knowledge (see e.g. [6], [7], [8]).

Among the different systems of temporal logic which may be or may have been proposed, we are in fact particularly interested in systems of interval logic. The reason for this interest lies in the fact that the concept of intervals and of properties being true for an interval seems to pervade most temporal reasoning aspects (when not most temporal logic specifications), and seems to provide us with some kind of "higher level" temporal reasoning. For similar and related reasons, interval logic and the interval concept seems to have early attracted the attention of philosophers and researchers in artificial intelligence (for recent proposals see e.g. [9], [10], [11], [12]). Of course, the logical systems proposed in the literature may differ vastly from one another. In this paper we are not interested in the relative merits of these different systems. We will focus our attention on systems of what could be called discrete propositional linear interval logic, i.e. systems of interval logic with a discrete linear model.

This paper is not concerned with the properties of the logical systems proposed, or, for that matter, with axiomatic basis for these systems. We will instead investigate a promising correspondence between temporal constructs and operators, and sequential machines with infinite input/output (transducers with infinite behavior). This correspondence enables us to derive decision procedures for certain kinds of interval logics, based on algorithms on machines.

The idea of relating logic and automata of some sort is not new and seems to date back at least to the early 1960's (see e.g. [13]), and has recently been advocated as a basis for automatic program verification in [14]. Two papers, [15] and [16], have taken this idea further and provided a more algebraic basis for the construction of sequential transducers with infinite behavior associated with temporal formulas. In particular, algorithms are provided which enable us to automatically derive machines associated with any regular temporal operator. In some sense, methods in these papers also generalize the tableau method used in modal logic.
In this paper, we capitalize on this work and show that this correspondence is also valid for a class of systems of (discrete propositional linear) interval logic. We then proceed to construct the system of transducers associated with an interval logic which has been proposed as a tool for the specification of communication protocols in [17] and [18]. This interval logic does not seem to belong to the class of logical systems to which we can directly apply the previous results. The difficulties encountered with it are exposed and the construction of the associated machines is explained.

The paper is organized as follows: in the second section we review the results in [15] and [16] on the correspondence between temporal logic and sequential transducers with infinite behavior, and show how we can extend them to interval logic. In the third section we review the interval logic proposed in [17] and [18], and construct its associated system of transducers. We conclude on the possible extensions of this work.

2. TEMPORAL LOGIC AND TRANSDUCERS WITH INFINITE BEHAVIOR

2.1. Transducers for Extended LTL

We begin here by reviewing the results obtained in [15] and [16] concerning the correspondence between extended discrete propositional linear temporal logic (Extended LTL or ETL for short) and sequential transducers with infinite behavior (sequential transducers for short). The idea is to consider temporal formulae as machines with infinite behavior. These machines take as input infinite words on a set of atomic propositions (infinite sequences of "states"), and issue as output the truth value of the given formula on each state. Also, machines for formulae are constructed in an incremental fashion. Each temporal operator is associated with a (fixed) machine. A machine associated with a formula is constructed by composing machines associated with the different operators in the formula, following the syntactic structure of the formula.

The machines obtained are very complete in the sense that, by following the different paths on the machines, we have all the models of the given formulae. Also, we have a decision procedure since for a given formula there is a model which satisfies it at the beginning iff there exists an input word and a successful path of so the associated machine which issues the formula at the first step. These results are in no way restricted to the standard LTL: it is straightforward to build for example a machine which recognizes an extended temporal operator à la Wolper [19] (it essentially suffices to build the automaton associated with the right linear grammar which defines the temporal operator).

More formally, ETL is a modal logic whose models are (totally ordered, denumerable) infinite sequences of states with valuations associating each state with a subset of a finite set \( P \) of atomic propositions \( \{P_1, \ldots, P_n\} \). We note \( S \) such a model and \( i \) the \( i \)th state of the sequence \( S \). For instance, the semantics of standard LTL, which is a strict sub-system of ETL, is given below. We note \( p \) a propositional variable ranging over \( P \), \( f \) and \( g \) LTL formulae, \( U \) the "weak until" operator, \( o \) the "next" operator.

\[
\begin{align*}
- S,i \models p & \iff p \text{ is in } V(i).
- S,i \models \neg p & \iff \neg S,i \models p.
- S,i \models f \& g & \iff S,i \models f \text{ and } S,i \models g.
- S,i \models o f & \iff S,(i+1) \models f.
- S,i \models f U g & \iff (\text{for all } j \geq i, \ S,j \models f) \text{ or (there is a } k \text{ such that for all } j \leq i < k, \ S,j \models f \text{ and } S,k \models g)).
\end{align*}
\]

Now, if we let \( V_n \) be the set of valuations on atomic propositions \( P_1, \ldots, P_n \), a model \( S \) for ETL can be seen as an infinite word on the alphabet \( V_n \). An ETL formula \( f \) associates \( S \) and a state \( i \) with a value on \( 2 = \{0, 1\} \), according to the truth value of \( f \) in state \( i \). Equivalently, an ETL formula \( f \) associates \( S \) with an infinite word on \( 2 \), according to the truth value of \( f \) in the different states of \( S \): \( f \) is a function from \( V_n^\omega \) to \( 2^\omega \) (if \( A \) is an alphabet, we note \( A^\omega \) the set of infinite words on \( A \)). Similarly, an \( n \)-ary temporal operator \( O_p \) associates the infinite sequences of its entries with an infinite sequence on \( 2 \) of its values: it is a function from \( (2^\omega)^n \) to \( 2^\omega \). In language theory, the objects which can compute these functions are sequential transducers. We will then associate a transducer with each formula of ETL: to construct the machine associated with a given formula, it will suffice to compose machines for boolean and modal operators appearing in the formula, according to its syntactic structure.