Temporal Reasoning

Henry Kautz
kautz@research.att.com

Temporal reasoning problems arise in many areas of AI, including planning, reasoning about physical systems, discourse analysis, and analysis of time-dependent data. Work in temporal reasoning can be classified in three general categories: algebraic systems; temporal logics; and logics of action. Although useful for many practical tasks, there is little evidence that any of these approaches accurately model human cognition about time. Less formal but more psychologically grounded approaches are discussed in some of the work in AI on plan recognition (Schmidt et al. 1978), work in linguistics on SEMANTICS and TENSE (Jackendoff 1983), and the vast psychological literature on MEMORY.

Algebraic systems concentrate on the relationships between time points and/or time intervals, which are represented by named variables. A set of either quantitative or qualitative equations constrain the values that could be assigned to the temporal variables. These equations could take the form a constraint satisfaction problem (CSP), a set of linear equations, or even a set of assertions in a restricted subset of first-order logic. The goal of the reasoning problem may be to determine consistency, to find a minimal labeling of the CSP, or to find consistent bindings for all the variables over some set of mathematical objects. In all of the algebraic systems described below, time itself is modeled as a continuous linear structure, although there has also been some investigation of discrete linear-time models (Dechter et al. 1991) and branching-time models (Ladkin et al. 1990).

The qualitative temporal algebra, originally devised by Allen (1983) and formalized as an algebra by Ladkin and Maddux (1994), takes time intervals to be primitive. There are 13 primitive possible relationships between a pair of intervals; for example, before ($\neg$), meets (m) (the end of the first corresponds to the beginning of the second), overlaps (o), etc. These primitive relationships can be combined to form $2^{13}$ complex relationships. For example, the constraint $I_1 \preceq m \preceq I_2$ means that $I_1$ is either before, meets, or is after $I_2$. Allen showed how a set of such constraints could be represented by a CSP, and how path-consistency could be used as an incomplete algorithm for computing a minimal set of constraints. The general problem of determining consistency is NP-complete (Vilain et al. 1989).

Quantitative algebras allow one to reason about durations of intervals and other metric information. The simple temporal constraint problems (STCSP) of Dechter et al. (1991) are a restricted form of linear equations. A time interval $I$ is identified with the pair of its starting point $I_s$ and ending point $I_e$. Difference equations allow one to place constraints on the relative ordering of points. For example, $I_e - J_s \in [-\infty, 0]$ means that $I$ is before $J$, and $I_s - I_e \in [3, 5]$ means that the duration of $I$ is between 3 and 5 units. Because they are just linear
programs, STCSP's can be solved in polynomial time. General TCSP's allow
the right-hand side of an equation to be a union of intervals, rather than a
single interval, and solving them is NP-complete. However, TCSP's still
cannot express certain complex constraints, such as that two intervals are
disjoint but unordered \((I \not\subseteq J)\), which would involve four points \((I_a < J_a \lor J_b < I_b)\).

Many researchers have explored tractable subsets of these algebras. A subset
is specified by the form of the constraints allowed in the statement of the problem
instance. Vilain et al. (1989) noted that the subset of Allen’s algebra that can
be exactly translated into equalities and inequalities over start and end points
is polynomial. Nebel and Branchert (1995) generalized this to relations that
can be translated into "ord-Horn constraints", the largest tractable subclass
that includes all the primitive relations. Koubarakis (1996) and Jonsson and
Blickström (1996) further showed that tractability still holds if such constraints
contain linear combinations of variables. However, none of the these classes can
express interval disjointed \((\not\subseteq)\), and in fact any tractable class that includes
disjointedness cannot contain all of the primitive relations. Tractable classes
including interval disjointedness include some of the "chordal graph" classes of
Columbic and Shamir (1993) and two of the algebras described in Drakengren
and Jonsson (1996).

Temporal algebras say nothing about how time intervals or points are associ-
ated with events or propositions. In practice, some external mechanism (such as
a planning system) generates interval and point tokens that are used to time-
tamp or index statements in a knowledge base. This external mechanism then
computes some of the constraints between the temporal tokens that must hold
according to the semantics of the knowledge base (for example, that a token
associated with a proposition is disjoint from one associated with the negation
of that proposition), and then asks the algebraic reasoning engine to compute
consequences of these assertions.

By contrast, temporal logics (van Benthem 1983, Thayse 1989) directly re-
present the temporal relationships between propositions, and do away with any
explicit tokens to represent time points or intervals. These are modal logics,
which extend propositional or first-order logic with temporal operators. For
eexample, propositional linear time temporal logic models time as a discrete se-
quence of states, and adds the modal operators next \(\Box\), always \(\Box\), eventually
\(\Diamond\), and until \(U\). For example, the formula \(\Box p \supset \Diamond q\) means that whenever \(p\)
holds, then \(q\) must hold in the next state.

The most successful applications of temporal logics have been in the area
of program verification. One approach to this task exploits the fact that any
formula in linear temporal logic can be converted into a a kind of finite automata
called a Büchi automata. The input language to the automata is sequences
of states. The automata accepts exactly those sequences that are models of
the corresponding temporal logic formula. In this application, the program
to be verified is written as a finite automata, and properties one wishes to
verify are written as formulas in temporal logic. The negation of the temporal
logic formula is then converted to a Böchi automata, which is then intersected with the program automata. It is then easy to check whether the combined automata accepts any inputs; if it does not, then the program satisfies the desired properties. The worst-case complexity of this procedure is high, since the automata may be exponentially larger than the formula.

Although temporal logics are frequently used in the verification and temporal database communities, they are just beginning to find widespread use in AI, particularly in planning. Applications include the specification and verification of real-time, reactive planners (Rosenschein and Kaelbling 1995, Williams and Nayak 1996), and specification of temporal-extended goals and search control rules (Bacchus and Kabanza 1996).

Finally, temporal reasoning is implicitly performed by all systems used to represent and reason about action and change, such as the situation calculus (McCarthy and Hayes 1969) or dynamic logic (Harel 1979). The situation calculus is simply a style of using first-order logic, in which the final argument to a predicate represents the state in which it holds. Actions are functions from state to state. Thus, the semantics for the situation calculus is based on a discrete, forward-branching model of time. The general approach can also be used to model continuous branching time (Reiter 1996). Successful planning systems have also been built (Blum and Furst 1995, Kautz and Selman 1996) that use a discrete linear model of time, where states are simply natural numbers used to index time-varying predicates.

References


