A Modal Temporal Logic for Reasoning about Change

Eric Mays

Department of Computer and Information Science
Moore School of Electrical Engineering/D2
University of Pennsylvania
Philadelphia, PA 19104

ABSTRACT

We examine several behaviors for query systems that become possible with the ability to represent and reason about change in data bases: queries about possible futures, queries about alternative histories, and offers of monitors as responses to queries. A modal temporal logic is developed for this purpose. A completion axiom for history is given and modelling strategies are given by example.

I INTRODUCTION

In this paper we present a modal temporal logic that has been developed for reasoning about change in data bases. The basic motivation is as follows. A data base contains information about the world: as the world changes, so does the data base -- probably maintaining some description of what the world was like before the change took place. Moreover, if the world is constrained in the ways it can change, so is the data base. We are motivated by the benefits to be gained by being able to represent those constraints and use them to reason about the possible states of a data base.

It is generally accepted that a natural language query system often needs to provide more than just the literal answer to a question. For example, [Kaplan 82] presents methods for correcting a questioner's misconceptions (as reflected in a query) about the contents of a data base, as well as providing additional information in support of the literal answer to a query. By enriching the data base model, Kaplan's work on correcting misconceptions was extended in [Mays 80] to distinquish between misconceptions about data base structure and data base contents. In either case, however, the model was a static one. By incorporating a model of the data base in which a dynamic view is allowed, answers to questions can include an offer to monitor for some condition which might possibly occur in the future. The following is an example:

U: "Is the Kitty Hawk in Norfolk?"
S: "No, shall I let you know when she is?"

[1]This work is partially supported by a grant from the National Science Foundation, NSF-MCS 81-07290.

But just <u>having</u> a dynamic view is not adequate, it is necessary that the dynamic view correspond to the possible evolution of the world that is modelled. Otherwise, behaviors such as the following might arise:

U: "Is New York less than 50 miles from Philadelphia?"
S: "No, shall I let you know when it is?"

An offer of a monitor is said to be <u>competent</u> only if the condition to be monitored can possibly occur. Thus, in the latter example the offer is not competent, while in the former it is. This paper is concerned with developing a logic for reasoning about change in data bases, and assessing the impact of that capability on the behavior of question answering systems. The general area of extended interaction in data base systems is discussed in [WJMM 83].

As just pointed out, the ability to represent and reason about change in data bases affects the range and quality of responses that may be produced by a query system. Reasoning about prior possiblity admits a class of queries dealing with the future possibility of some event or state of affairs at some time in the past. These queries have the general form:

"Could it have been the case that p?"

This class of queries will be termed counterhistoricals in an attempt to draw some parallel with counterfactuals. The future correlate of counterhistoricals, which one might call futurities, are of the form:

"Can it be the case that p?"
i.e. in the sense of:
"Might it ever be the case that p?"

The most interesting aspect of this form of question is that it admits the ability for a query system to offer a monitor as a response to a question for relevant information the system may become aware of at some future time. A query system can only competently offer such monitors when it has this ability, since otherwise it cannot determine if the monitor may ever be satisfied.

II REPRESENTATION

We have chosen to use a modal temporal logic. There are two basic requirements which lead us toward logic and away from methods such as Petri nets. First, it may be desirable to assert that some proposition <u>is</u> the case without necessarily specifying exactly when. Secondly, our knowledge may be disjunctive. That is, our knowledge of temporal situations may be incomplete and indefinite, and as others have argued [Moore 82] (as a recent example), methods based on formal logic (though usually first-order) are the only ones that have so far been capable of dealing with problems of this nature.

In contrast to first-order representations, modal temporal logic makes a fundamental distinction between variability over time (as expressed by modal temporal operators) and variability in a state (as expressed using propositional or first-order languages). Modal temporal logic also reflects the temporally indefinite structure of language in a way that is more natural than the commaon method of using state variables and constants in a first-order logic. On the side of first-order logic, however, is expressive power that is not necessarily present in modal temporal logic. (But, see [Kamp 68] and [GPSS 80] for comparisons of the expressive power of modal temporal logics with first-order theories.)

There are several possible structures that one could reasonably imagine over states in time. The one we have in mind is discrete, backwards linear, and infinite in both directions. We allow branching into the future to capture the idea that it is open, but the past is determined. Due to the nature of the intended application, we also have assumed that time is discrete. It should be stressed that this decision is not motivated by the belief that time itself is discrete, but rather by the data base application. Furthermore, in cases where it is necessary for the temporal structure to be dense or continuous, there is no immediate argument against modal temporal logic in general. (That is, one could develop a modal temporal logic that models a continuous structure of time [RU 71].)

A modal temporal structure is composed of a set of states. Each state is a set of propositions which are true of that state. States are related by an immediate predecessor-successor relation. A branch of time is defined by taking some possible sequence of states accessible over this relation from a given state. The future fragment of the logic is based on the unified branching temporal logic of [BMP 81], which introduces branches and quantifies over them to make it possible to describe properties on some or all futures. This is extended with an "until" operator (as in [Kamp 68], [GPSS 80]) and a past fragment. Since the structures are backwards linear the existential and universal operators are merged to form a linear past fragment.

A. Syntax

Formulas are composed from the symbols,

- A set $\mathfrak{P}$ of atomic propositions.
- Boolean connectives: v, ~.
- Temporal operators: AX (every next), EX (some next), AG (every always), EG (some always), AF (every eventually), EF (some eventually), AU (every until), EU (some until), L (immediately past), P (sometime past), H (always past), S (since). AU, EU, and S are binary; the others are unary. For the operators composed of two symbols, the first symbol ("A" or "E") can be thought of as quantifying universally or existentially over branches in time; the second symbol as quantifying over states within the branch. Since branching is not allowed into the past, past operators have only one symbol.

using the rules,

- If $p \in \mathfrak{P}$, then p is a formula.
- If p and q are formulas, then (~p), (p v q) are formulas.
- If m is a unary temporal operator and p is a formula, then (m p) is a formula.
- If m is a binary temporal operator and p and q are formulas, then (p m q) is a formula.

Parentheses will occasionally be omitted, and &, $\rightarrow$, $\leftrightarrow$ used as abbreviations. (In the next section: "Ax" should be read as the universal quantifier over the variable x, "Ex" as the existential quantifier over x.)

B. Semantics

A temporal structure T is a triple (S, Π, R) where,

- S is a set of states.
- $\Pi : (S \rightarrow 2^{\mathfrak{P}})$ is an assignment of atomic propositions to states.
- $R \subseteq (S \times S)$ is an accessibility relation on S. Each state is required to have at least one successor and exactly one predecessor -- i.e., As (Et (sRt) & E!t (tRs)).

Define b to be an s-branch
$b = (\ldots, s_{-1}, s=s_0, s_1, \ldots)$ such that $s_i R s_{i+1}$.

The relation ">" is the transitive closure of R.

The satisfaction of a formula p at a state s in a structure T, $\langle T,s \rangle \models p$, is defined as follows:

$\langle T,s \rangle \models p$ iff $p \in \Pi(s)$, for $p \in \mathfrak{P}$

$\langle T,s \rangle \models \sim p$ iff not $\langle T,s \rangle \models p$

$\langle T,s \rangle \models p \vee q$ iff $\langle T,s \rangle \models p$ or $\langle T,s \rangle \models q$

$\langle T,s\rangle \models AGp$ iff $AbAt((t \in b \;\&\; t \geq s) \rightarrow \langle T,t\rangle \models p)$
(p is true at every time of every future)

$\langle T,s\rangle \models AFp$ iff $AbEt(t \in b \;\&\; t \geq s \;\&\; \langle T,t\rangle \models p)$
(p is true at some time of every future)

$\langle T,s\rangle \models pAUq$ iff
$AbEt(t \in b \;\&\; t \geq s \;\&\; \langle T,t\rangle \models q \;\&\; At'((t' \in b \;\&\; s \leq t' < t) \rightarrow \langle T,t'\rangle \models p)))$
(q is true at some time of every future and until q is true p is true)

$\langle T,s\rangle \models AXp$ iff $At(sRt \rightarrow \langle T,t\rangle \models p)$
(p is true at every immediate future)

$\langle T,s\rangle \models EGp$ iff $EbAt((t \in b \;\&\; t \geq s) \rightarrow \langle T,t\rangle \models p)$
(p is true at every time of some future)

$\langle T,s\rangle \models EFp$ iff $EbEt(t \in b \;\&\; t \geq s \;\&\; \langle T,t\rangle \models p)$
(p is true at some time of some future)

$\langle T,s\rangle \models EXp$ iff $Et(sRt \;\&\; \langle T,t\rangle \models p)$
(p is true at some immediate future)

$\langle T,s\rangle \models pEUq$ iff
$EbEt(t \in b \;\&\; t \geq s \;\&\; \langle T,t\rangle \models q \;\&\; At'((t' \in b \;\&\; s \leq t' < t) \rightarrow \langle T,t'\rangle \models p)))$
(q is true at some time of some future and in that future until q is true p is true)

$\langle T,s\rangle \models Hp$ iff $AbAt((t \in b \;\&\; t \leq s) \rightarrow \langle T,t\rangle \models p)$
(p is true at every time of the past)

$\langle T,s\rangle \models Pp$ iff $AbEt(t \in b \;\&\; t \leq s \;\&\; \langle T,t\rangle \models p)$
(p is true at some time of the past)

$\langle T,s\rangle \models Lp$ iff $At(tRs \rightarrow \langle T,t\rangle \models p)$
(p is true at the immediate past)

$\langle T,s\rangle \models pSq$ iff
$AbEt(t \in b \;\&\; t \leq s \;\&\; \langle T,t\rangle \models q \;\&\; At'((t' \in b \;\&\; s \geq t' > t) \rightarrow \langle T,t'\rangle \models p)))$
(q is true at some time of the past and since q is true p is true)

A formula p is valid iff for every structure T and every state s in T, $\langle T,s\rangle \models p$.

III MODELLING CHANGE IN KNOWLEDGE BASES

As noted earlier, this logic was developed to reason about change in data bases. Although ultimately the application requires extension to a first-order language to better express variability within a state, for now we are restricted to the propositional case. Such an extension is not without problems, but should be manageable.

The set of propositional variables for modelling change in data bases is divided into two classes. A state proposition asserts the truth of some atomic condition. An event proposition associates the occurence of an event with the state in which it occurs. The idea is to impose constraints on the occurence of events and then derive the appropriate state description. To be specfic, let Qs1...Qsn be state propositions and Qe1...Qem be event propositions. If PHI is a boolean formula of state propositions, then formulas of the form:
(PHI → EX Qei) are event constraints. To derive state descriptions from events frame axioms are required:
(Qei → ((L PHI1) → PHI2)),
where PHI1 and PHI2 are boolean formulas of state propositions. In the blocks world, and event constraint would be that if block A was clear and block B was clear then move A onto B is a next possible event:
((cleartop(A) & cleartop(B)) → EX move(A,B)).
Two frame axioms are:
(move(A,B) → on(A,B)) and
(move(A,B) → ((L on(C,D)) → on(C,D))).

If the modelling strategy was left as just outlined, nothing very significant would have been accomplished. Indeed, a simpler strategy would be hard to imagine, other than requiring that the state formulas be a complete description. This can be improved in two non-trivial ways. The first is that the conditions on the transitions may reference states earlier than the last one. Secondly, we may require that certain conditions might or must eventually happen, but not necessarily next. As mentioned earlier, these capabilities are important considerations for us. By placing biconditionals on the event constraints, it can be determined that some condition may never arise, or from knowledge of some event a reconstruction of the previous state may be obtained.

The form of the frame axioms may be inverted using the until operator to obtain a form that is perhaps more intuitive. As specified above the form of the frame axioms will yield identical previous and next state propositions for those events that have no effect on them. The standard example from the blocks world is that moving a block does not alter the color of the block. If there are a lot of events like move that don't change block color, there will be a lot of frame axioms around stating that the events don't change the block color. But if there is only one event, say paint, that changes the color of the block, the "every until" (AU) operator can be used to state that the color of the block stays the same until it is painted. This strategy works best if we maintain a single event condition for each state; i.e. no more than a single event can occur in each state. For each application, a decision must be made as to how to best represent the frame axioms. Of course, if the world is very complicated, there will be a lot of complicated frame axioms. I see no easy way around this problem in this logic.

A. Completion of History

As previously mentioned, we assume that the past is determined (i.e. backwards linear). However this does not imply that our knowledge of the past is complete. Since in some cases we may wish to claim complete knowledge with respect to one or more predicates in the past, a completion axiom is developed for an intuitively natural conception of history. Examples of predicates for which our knowledge might be complete are presidential inaugurations, employees of a company, and courses taken by someone in college.

In a first order theory, T, the completion axiom with respect to the predicate Q where (Q c1)...(Q cn) are the only occurences of Q in T is:
Ax((Q x) <-> x=c1 v...v x=cn). From right to left on the biconditional this just says what the orginal theory T did, that Q is true of c1...cn. The completion occurs from left to right, asserting that c1...cn are the only constants for which Q holds. Thus for some c′ which is not equal to any of c1...cn, it is provable in the completed theory that ~(Q c′), which was not provable in the original theory T. This axiom captures our intuitive notions about Q.[2] The completion axiom for temporal logic is developed by introducing time propositions. The idea is that a conjunct of a time proposition, T, and some other proposition, Q, denotes that Q is true at time T. If time propositions are linearly ordered, and Q occurs only in the form
P(Q & T1) &...& P(Q & Tn) in some theory M, then the history completion axiom for M with respect to Q is
H(Q <-> T1 v...v Tn). Analogous to the first-order completion axiom, the direction from left to right is the completion of Q. An equivalent first-order theory to M in which each temporal proposition Ti is a first-order constant ti and Q is a monadic predicate,
(Q t1) &...& (Q tn), has the first-order completion axiom (with Q restricted to time constants of the past, where t0 is now):
Ax≤t0 ((Q x) <-> x=t1 v...v x=tn).

B. Example

The propositional variables T-reg, T-add, T-drop, T-enroll, and T-break are time points intended to denote periods in the academic semster on which certain activities regarding enrollment for courses is dependent. The event proposition are Qe-reg, Qe-pass, Qe-fail, and Qe-drop; for registering for a course, passing a course, failing a course, and dropping a couirse, respectively. The only state is Qs-reg, which means that a student is registered for a course.

[2] [Clark 78] contains a general discussion of predicate completion. [Reiter 82] discusses the completion axiom with respect to circumscription.

T-reg <-> (AX T-add)

T-add <-> (AX T-drop) - drop follows add

T-drop <-> (AX T-enroll) - enroll follows drop

T-enroll <-> (AX T-break) - break follows enroll

((T-reg v T-add) & ~Qs-reg & ~(P Qe-pass)) <-> (EX Qe-reg) - if the period is reg or add and a student is not registered and has not passed the course then the student may next register for the course

((T-add v T-drop) & Qs-reg) <-> (EX Qe-drop) - if the period is add or drop and a student is registered for a course then the student may next drop the course

(T-enroll & Qs-reg) <-> (EX Qe-pass)) - if the period is enroll and a student is registered for a course then the student may next pass the course

(T-enroll & Qs-reg) <-> (EX Qe-fail)) - if the period is enroll and a student is registered for a course then the student may next fail the course

Qe-reg -> (Qs-reg AU (Qe-pass v Qe-fail v Qe-drop)) - if a student registers for a course then eventually the student will pass or fail or drop the course and until then the student will be registered for the course

((L ~Qs-reg) & ~Qe-reg) -> ~Qs-reg) - not registering maintains not being registered

AX(Qe-reg & Qe-pass & Qe-fail & Qe-drop & Qe-null) - one of these events must next happen

~(Qe-i & Qe-j), for ~i=j (e.g. ~(Qe-reg & Qe-pass)) - but only one

IV COUNTERHISTORICALS

A counterhistorical may be thought of as a special case of a counterfactual, where rather than asking the counterfactual, "If kangaroos did not have tails would they topple over?", one asks instead "Could I have taken CSE110 last semester?". That is, counterfactuals suppose that the present state of affairs is slightly different and then question the consequences. Counterhistoricals, on the other hand, question how a course of events might have proceeded otherwise. If we picture the underlying temporal structure, we see that although there are no branches into the past, there are branches from the past into the future. These are alternative histories to the one we are actually in. Counterhistoricals explore these alternate histories.

Intuitively, a counterhistorical may be evaluated by "rolling back" to some previous state and then reasoning forward, disregarding any events that actually took place after that state, to determine whether the specified condition might arise. For the question, "Could I have registered for CSE110 last semester?", we access the state specified by last semester, and from that state description, reason forward regarding the possibility of registering for CSE110.

However, a counterhistorical is really only interesting if there is some way in which the course of events is constrained. These constraints may be legal, physical, moral, bureaucratic, or a whole host of others. The set of axioms in the previous section is one example. The formalism does not provide any facility to distinquish between various sorts of constraints. Thus the mortal inevitability that everyone eventually dies is given the same importance as a university rule that you can't take the same course twice.

In the logic, the general counterhistorical has the form: P(EFp). That is, is there some time in the past at which there is a future time when p might possibly be true. Constraints may be placed on the prior time:
P(q & EFp), e.g. "When I was a sophomore, could I have taken Phil 6?". One might wish to require that some other condition still be accessible:
P(EF(p & EFq)), e.g. "Could I have taken CSE220 and then CSE110?"; or that the counterhistorical be immediate from the most recent state:
L(EXp). (The latter is interesting in what it has to say about possible alternatives to -- or the inevitability of -- what is the case now. [WM 83] shows its use in recognizing and correcting event-related misconceptions.) For example, in the registration domain if we know that someone has passed a course then we can derive from the axioms above the counterhistorical that they could have not passed:
((P Qe-pass) → P(EF~Qe-pass).

V FUTURITIES

A query regarding future possibility has the general logical form: EFp. That is, is there some future time in which p is true. The basic variations are: AFp, must p eventually be true; EGp, can p remain true; AGp, must p remain true. These can be nested to produce infinite variation. However, answering direct questions about future possibility is not the only use to be made of futurities. In addition, futurities permit the query system to competently offer monitors as responses to questions. (A monitor watches for some specified condition to arise and then performs some action, usually notification that the condition has occurred.) A monitor can only be offered competently if it can be shown that the condition might possibly arise, given the present state of the data base. Note that if any of the stronger forms of future possibility can be derived it would be desirable to provide information to that effect.

For example, if a student is not registered for a course and has not passed the course and the time wasprior to enrollment, a monitor for the student registering would be competently made given some question about registration, since
((~Qs-reg & ~(P Qe-pass) & AX(AF Te)) →
(EF Qe-reg)). However, if the student had previously passed the course, the monitor offer would not be competent, since
((~Qs-reg & (P Qe-pass) & AX(AF Te)) →
~(EF Qe-reg)).

Note that if a monitor was explicity requested, "Let me know when p happens," a futurity may be used to determine whether p might ever happen. In addition to the processing efficiency gained by discarding monitors that can never be satisfied, one is also in a position to correct a user's mistaken belief that p might ever happen, since in order to make such a request s/he must believe p could happen. Corrections of this sort arise from intensional failures of presumptions in the sense of [Mays 80] and [WM 83]. If at some future time from the monitor request, due to some intervening events p can no longer happen, but was originally possible, an extensional failure of the presumption (in the sense of [Kaplan 82]) might be said to have occurred.

The application of the constraints when attempting to determine the validity of an update to the data base is important to the determination of monitor competence. The approach we have adopted is to require that when some formula p is considered as a potential addition to the data base that it be provable that EXp. Alternatively one could just require that the update not be inconsistent, that is not provable that AX~p. The former approach is preferred since it does not make any requirement on decidability. Thus, in order to say that a monitor for some condition p is competent, it must be provable that EFp.

VI DISCUSSION

This work has been influenced most strongly by work within theory of computation on proving program correctness ([BMP 81] and [GPSS 80]) and within philosophy on temporal logic [RU 71]. The work within AI that is most relevant is that of [McDermott 82]. Two of McDermott's major points are regarding the openess of the future and the continuity of time. With the first of these we are in agreement, but on the second we differ. This difference is largely due to the intended application of the logic. Ours is applied to changes in data base states (which are discrete), whereas McDermott's is physical systems (which are continuous). But even within the domain of physical systems it may be worthwhile to consider discrete structures as a tool for abstraction, for which computational methods may prove to be more tractable. At least by considering modal temporal logics we may be able to gain some insight into the reasoning process whether over discrete or continuous structures.

We have not made at serious effort towards implementation thus far. A tableau based theorem prover has been implemented for the future fragment based on the procedure given in [BMP 81]. It is able to do problems about one-half the size of the example given here. Based on this limited experience we have a few ideas which might improve its abilities. Another procedure based on the tableau method which is based on ideas from [BMP 81] and [RU 71] has been developed but we are not sufficiently confident in its correctness to present it at this point.

ACKNOWLEDGEMENTS

I have substantially benefited from comments, suggestions, and discussions with Aravind Joshi, Sitaram Lanka, Kathy McCoy, Gopalan Nadathur, David Silverman, Bonnie Webber, and Scott Weinstein.

REFERENCES

[BMP 81] M. Ben-Ari, Z. Manna, A. Pneuli, "The Temporal Logic of Branching Time," Eighth ACM Symposium on Principles of Programming Languages, Williamsburg, Va., January 1981.

[Clark 78] K.L. Clark, "Negation as Failure," in Logic and Data Bases, H. Gallaire and J. Minker (eds.), Plenum, New York.

[GPSS 80] D. Gabbay, A. Pneuli, S. Shelah, J. Stavi, "On the Temporal Analysis of Fairness, Seventh ACM Symposium on Principles of Programming Languages, 1980.

[Kamp 68] J.A.W. Kamp, Tense Logic and the Theory of Linear Order, PhD Thesis, UCLA, 1968.

[Kaplan 82] S.J. Kaplan, "Cooperative Responses from a Portable Natural Language Query System," Artificial Intelligence (19, 2), October 1982.

[Mays 80] E. Mays, "Failures in Natural Language Systems: Applications to Data Base Query Systems," Proceedings of AAAI 80, Stanford, Ca., August 1980.

[Mays 82] E. Mays, "Monitors as Responses to Questions: Determining Competence," Proceedings of AAAI 82, Pittsburgh, Pa., August 1982.

[McDermott 82] D. McDermott, "A Temporal Logic for Reasoning About Processes and Plans," Cognitive Science (6), 1982.

[Moore 82] R.C. Moore, "The Role of Logic in Knowledge Representation and Commensense Reasoning," Proceedings of AAAI 82, Pittsburgh, Pa., August 1982.

[RU 71] N. Rescher and A. Urquhart, Temporal Logic, Springer-Verlag, New York, 1971.

[Reiter 82] R. Reiter, "Circumscription Implies Predicate Completion (Sometimes)," Proceedings of AAAI 82, Pittsburgh, Pa., August 1982.

[WJMM 83] B. Webber, A. Joshi, E. Mays, K. McKeown, "Extended Natural Language Data Base Interactions," International Journal of Computers and Mathematics, Spring 83.

[WM 83] B. Webber and E. Mays, "Varieties of User Misconception: Detection and Correction", Proceedings of IJCAI 83.