Reprinted from Philosophy and AI: Essays at the Interface, chap.4, Cummins and Pollock, eds., MIT Press (1991)79--103

Memory, Reason, and Time: the Step-logic Approach

Jennifer J. Elgot-Drapkin*
Department of Computer Science
College of Engineering and Applied Sciences
Arizona State University
Tempe, AZ 85287-5406

Michael Miller†
Department of Computer Science
University of Maryland
College Park, MD 20742

Donald Perlis‡
Department of Computer Science
and
Institute for Advanced Computer Studies
University of Maryland
College Park, MD 20742

Abstract

Concern with the real-time nature of effective reasoning led us to develop a memory-based model of reasoning. Later efforts convinced us that a formal treatment of this approach would be fruitful. The current paper surveys the evolution of this work and discusses potential future research endeavors.

1 Motivation

Traditional theoretical treatments of reasoning do not, in our view, address reasoning *per se* at all. Rather they seek to characterize the *end results* of reasoning. We have four complaints with this. First, it is not clear that reasoning has clearly identifiable end results, but rather goes on and on as part of the active history of an individual reasoner. Second, reasoning is a process or activity, and this is simply ignored in traditional studies. Third, as a consequence, crucial issues of temporal and spatial resources are not taken into account. Finally, if we are going to understand realizable intelligent systems, we must look at what they actually do, i.e., we must accept at least a certain amount of cognitive plausibility (whether human or computer).

The paradigm for such a reasoning agent would seem to be that suggested by Nilsson [17], namely, a computer individual with a lifetime of its own. What is of interest is not its ‘‘ultimate’’ set of conclusions, but rather its changing set of conclusions over time. Indeed, there will be, in general, no ultimate or limiting set of conclusions.

The notion of time enters the reasoning process in two ways. Not only does reasoning take time, but it often deals with time as an object of reasoning. The latter has of course been extensively studied, in so-called temporal and tense logics. But there, once again, it is the *end results* and not the *process* of drawing conclusions that is studied. We seek

* Supported in part by the IBM Corporation.
† Supported in part by the U.S. Army Research Office (DAAL03-88-K0087).
‡ Supported in part by the U.S. Army Research Office (DAAL03-88-K0087).

to do both, in one model of reasoning. This pointedly includes the case of reasoning going on in time (of course, as it must) while the system is focused on (reasoning about) that very passage of time as it reasons. This, we will argue, is no mere curiosity, but rather a central feature of intelligent thought.

Similarly, space enters in two ways. On the one hand there is spatial reasoning, again well-studied. On the other hand there is the fact that reasoning takes up space, in the form of memory. Again we are interested in modeling both. Moreover, we want both forms of temporal reasoning and both forms of spatial reasoning to go on together in the same system.

To this end we initiated a project some years ago, toward the development of an automatic reasoner. This work is summarized in Section 2. Later we came to realize that in it there lurked certain features that suggested formalization, and so we embarked on the development of a logic of sorts. The logic retains much of the traditional approach and yet also breaks loose from that in several key respects; in particular, the new formalization -- which we call *step-logic* -- does not focus on end results. This will be explained in Section 3. So far, we have concentrated on time rather than space (memory) in this formalization. Section 4 suggests future directions including the incorporation of space concerns.

To illustrate what we called the end-result character of traditional approaches to formalizing commonsense reasoning, note that the traditional approaches suffer from the problem of logical omniscience: if an agent has α_1, ..., α_n in its belief set, and if γ follows from $\alpha_1, \ldots, \alpha_n$ according to the agent's rules of inference, then the agent also believes γ (i.e., γ is also in the belief set). As a specific example, if a typical omniscient agent believes α, and also believes $\alpha \rightarrow \beta$, then the agent believes β. As an illustration, refer to Figure 1. The reasoner begins with a set of axioms, and the deductive mechanism generates theorems along the way, e.g., α, later $\alpha \rightarrow \beta$, still later β. Such mechanisms have usually been studied in terms of the set of all theorems deducible therein, what we call the ''final tray of conclusions'' into which individually proven theorems are represented as dropping, thereby ignoring their time and means of deduction. One asks, for instance, whether a wff α is a theorem (i.e., is in the final tray), *not* whether α is a theorem proven in i steps.

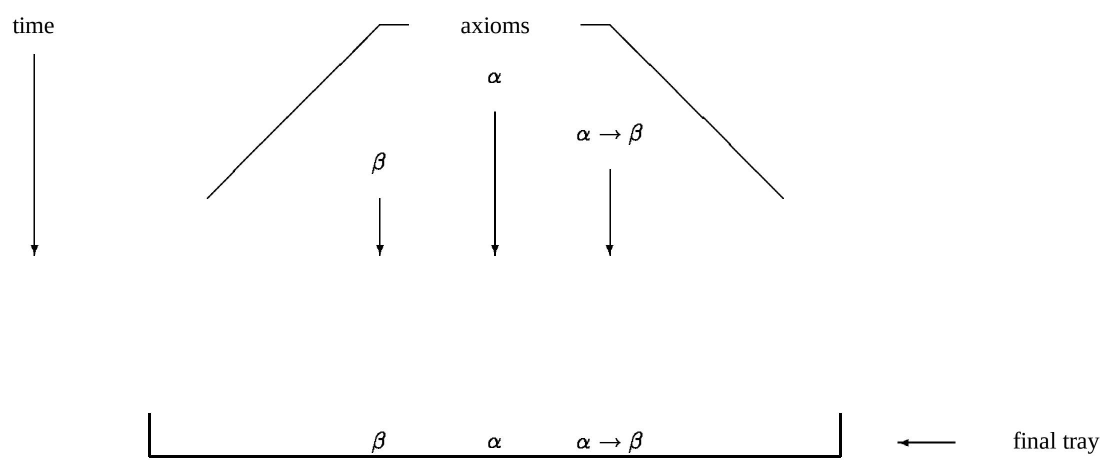

Figure 1: Final-tray logical studies

A particularly vexing aspect of this type of reasoning is what we call the ''swamping problem''---namely that from a contradiction all wffs are concluded. For this reason most formal studies of reasoning deliberately avoid contradictions; those that do not (e.g., Doyle [2]), provide a separate device for noting contradictions and revising beliefs while the ''main'' reasoning engine sits quiescent. In general, however, this will not do, since the knowledge needed to resolve conflicts will depend on the same wealth of world knowledge used in any other reasoning. Thus reasoning about birds involves inference rules applied to beliefs about birds, whether used to resolve a conflict or simply to produce non-conflicting conclusions. We contend, then, that one and the same on-going process of reasoning should be responsible both for keeping itself apprised of contradictions and their resolution, and for other forms of reasoning.

We seek potentially inconsistent, but nevertheless useful, logics where a real-time self-referential feature allows a direct contradiction to be spotted and corrective action taken, as part of the same system of reasoning. We will suggest some specific inference mechanisms for real-time default reasoning, notably a form of introspection relevant to default reasoning. This facilitates the study of *fallible* agents reasoning over time. A fallible agent may derive or encounter an inconsistency, identify it as such, and then proceed to remedy it. Contradictions then need not be bad; indeed, they can be good, in that they allow sources of error to be isolated (see [18]).

The agent should be able to reason *about* its own ongoing reasoning efforts, and in particular, reason whether it has or has not yet reached a given conclusion. One of our main focuses here is the problem of an agent's determining that in fact it does *not* (currently) know something. This *negative introspection* will be a key feature of the deduction, and subsequent resolution, of contradictions in our later examples of default reasoning in Section 3.2.4. It turns out that negative introspection presents certain temporal constraints that will strongly influence the formal development.

The literature contains a number of approaches to limited (non-omniscient) reasoning, apparently with similar motivation to our own. However, with very little exception, the idealization of a ''final'' state of reasoning is maintained, and the limitation amounts to a reduced set of consequences rather than an ever-changing set of tentative conclusions. Thus Konolige [12] studies agents with fairly arbitrary rules of inference, but assumes logical closure for the agents with respect to those rules, ignoring the effort involved in performing the deductions. Similarly, Levesque [13] and Fagin and Halpern [8] provide formal treatments of limited reasoning, so that, for instance, a contradiction may go unnoticed; but the conclusions that *are* drawn are done so instantaneously, i.e., the steps of reasoning involved are not explicit. Fagin and Halpern in particular postulate a notion of awareness, so that if α and $\alpha \rightarrow \beta$ are known, still β will not be concluded unless the agent is aware of β; just how it is that β fails to be in the awareness set in unclear. Goodwin [11] comes a little closer to meeting our desiderata but still maintains a largely final-tray-like perspective.

This next section describes the automatic reasoner.

2 A Pragmatic Approach

2.1 Background

Most of the A.I. systems built today are designed to solve one problem only. That is, the system is turned on, labors away for some period of time, then spits out the (hopefully, correct) answer. It is then turned off, or works on another problem with no knowledge of its past. In contrast, consider a system with a ''life-time of its own''; one whose behavior significantly depends on its continued dealings with a variety of issues.

In the current section, we imagine a robot (i.e., some kind of reasoning agent) situated on a desert island (i.e., some kind of robust world) left to its own devices. It has been endowed with a data base of information that it can use in order to get along in this world.

It is clear that an autonomous robot will have to deal with a world about which it will have only partial knowledge. Conclusions will frequently be drawn without full justification. As a consequence, some facts will have to be retracted in the face of further information. An underlying premise is that a real-world reasoner is limited, at least in terms of the scope and accuracy of the information to which it has access. We must back off a bit from the more traditional topic of idealized reasoning agents that are infallible and omniscient. Additionally, if we limit the computing resources of the robot (much as people are limited), then some of the difficulties of formal representation of commonsense reasoning become more tractable. That is, greater limitations serve to constrain solutions to the point that answers may be more easily seen.

A centerpiece, and bugbear, of formal research in commonsense reasoning has been that of (global and derivational) consistency tests. Even when, as in the case of circumscription (see [14]), direct testing of consistency is avoided by clever syntactic manipulations, there is still implicit reference to global properties of the reasoning system (i.e., its set of axioms). Time is then taken to assess logical consequences of these properties before a commonsense conclusion is drawn. Thus a strong flavor of idealized reasoning has persisted.

Suppose we would like to have a reasoning system use a rule such as $\alpha \rightarrow \beta$ to conclude β, given α, and later, in the face of new evidence, be able to retract its belief in β. A somewhat standard (idealized) way of dealing with this is to use a rule such as, ''If α, and it is consistent to believe β, then conclude β''. This is called a default rule, and is the source of the aforementioned consistency tests. The point of the ''it is consistent to conclude β'' is to see whether there *already* is evidence to retract β, i.e., to prevent the conclusion β in the first place. Instead of holding up the system's conclusion until such a test can be made, our approach allows the system to jump directly to the conclusion β, and *then* decide whether it was rash.

In the robot's world, it seems, nearly all rules are actually defaults, since we can rarely be sure of anything. It is then tempting to use a ''brute force'' method of encoding these defaults: simply encode the rule as ''If α, then conclude β'', with no ''unless it is not consistent to do so'' condition. One would then proceed as normal in the inference process until a direct contradiction is somehow brought to the reasoner's attention (a process which will be explained shortly). At this point something would have to be done to resolve the inconsistency. It is important to note that a contradiction should not be something that will incapacitate our system. Rather, it is something that is to be expected, recognized (but only when relevant to the current focus of attention), and resolved by the system as part of its normal operation. In particular this allows the use of a rule such as $\alpha \rightarrow \beta$, even though it may be recognized as not strictly true. (Though this is not a trivial point, we will leave further discussion for a future paper. See [10] for a related idea.)

We can now think of our robot as a real-world, resource-limited reasoner acting, albeit somewhat slowly, over time. It uses defaults to allow itself a short-cut to quick deductions, correcting fallacy when it is recognized.

2.2 Details of the Model

Our robot is essentially a *memory model* that is controlled by an *inference cycle* mechanism. The model's components hold data in various forms so that as time evolves the inference mechanism is able to simulate reasoning. The model has been implemented, and the examples later in this section are from actual computer trials.

2.2.1 Architecture

The memory model contains five key elements: STM, LTM, ITM, QTM, and RTM. The first three of these are standard parts of cognitively-based models of memory. See Figure 2

STM is meant to represent the reasoner's current focus of attention. Its chief purpose is to allow access to a very large database (LTM), yet not suffer an exponential explosion of inferences. STM is a small set of beliefs that are currently ''active''. These beliefs are represented as logical formulae and are used to help establish STM's next state, a process which will be described in the next section. STM is structured as a FIFO queue. Since STM's size is limited[1], as new facts are brought into STM, old facts must be discarded. That is, the older, discarded facts are no longer in focus.

[1] In our implementation, STM's size is fixed, yet easily changed for experimental purposes. An interesting sidelight is that an STM size of eight is the smallest that has led to effective task-oriented behavior over several domains, and that larger sizes have offered no advantage. This is in surprising accord with psychological data which measure human short-term memory to hold seven plus-or-minus two ''chunks'' of data at one time [15].

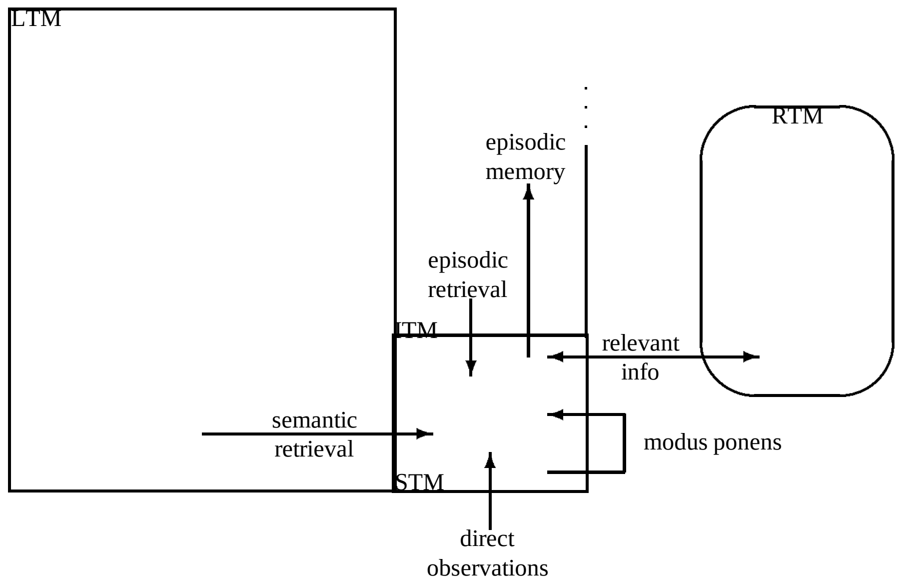

Figure 2: architecture of the memory model

It is convenient to conceive of STM as a theorem prover to which LTM supplies axioms. LTM may then be thought of as a data base of information to which the robot has access. LTM is implemented as a series of tuples of the form:

$$< T_1, \ldots, T_n, B >$$

where the T_i and B represent logical formulae. The idea behind LTM is that beliefs are held as a series of associations. Thinking about certain triggers (i.e., the T_i's) causes possibly many past associations (i.e., the B's) to be brought into focus.

ITM is of unbounded size and holds all information which is discarded from STM in chronological order, i.e., in order of entry. Thus, ITM is implemented similarly to a stack, in that the most recently entered facts are the most easily accessed, although they are never removed.

QTM is a technical device that controls the flow of information into STM.

RTM is the repository of default resolution and relevance. RTM is implemented as a list of facts that have most recently been in STM. Facts are coded with a time decay variable so that they can decay out of RTM as they are no longer relevant, i.e., not found in STM for a specified number of inference cycles.

2.2.2 Inference

An *inference cycle* can be thought of as the process of updating the system's current focus of attention. Given some state of STM, four different mechanisms work simultaneously to produce a new state. These four mechanisms are direct observation, modus ponens (MP), semantic retrieval from LTM, and episodic retrieval from ITM.

To model this simultaneity, our implementation uses a temporary waiting queue (QTM) which holds the next cycle's STM facts until all four mechanisms have finished working on the old STM facts. Once they have finished, elements of QTM are placed into STM one at a time, disallowing repetition of facts in STM. Throughout this process, older items in STM are moved into ITM as needed to maintain STM's size. Note that if QTM is very large, other means will be required to reasonably select elements to go into STM[2].

Currently, direct observation is modeled by allowing outsiders to simply assert a fact to the system. This allows us the pretense of an autonomous system noting events in a dynamic environment.

MP is applied in the following form: from Ac and ($Ax \rightarrow Bx$), Bc is inferred. That is, Bc is brought into QTM if Ac and ($Ax \rightarrow Bx$) are already contained in STM. Consequently, at the end of such a cycle, Bc is a candidate for STM.

Facts from LTM are brought into STM by association. When facts in STM unify with the first n elements of some $< T_1, \ldots, T_n, \beta >$, in LTM, then β will be brought into QTM (and subsequently into STM), with its variables properly bound.

Information retrieved from ITM into STM can take several forms. For example, since ITM is a chronological listing of all past STM facts, its structure allows for the retrieval of goal statements that are not yet satisfied, but that have already been pushed out of STM. This allows the system to work through a goal-subgoal process.

2.2.3 General Features

Before illustrating the memory model with an example, it is worthwhile making brief mention of several of its features.

First, in most of our work we have limited the size of STM to eight elements. That is, eight formulae can be held in STM at any one time. We have had a fair amount of success solving problems with an STM of that size. However, adjustments can easily be made to STM's size and, indeed, our examples will use a size of four.

Second, LTM can hold inconsistent data without the usual disastrous consequences of customary inference systems. That is, as long as a direct contradiction does not occur in STM, no inconsistency is even detected.

Third, the system is capable of meta-inference or ''introspection'' very simply by searching its list of STM elements. For example, it can determine whether a given formula and its negation are both currently in STM. This activity occurs via inference steps no different in principle from any of its other inferences. In effect, the system may look at snapshots of itself as it runs, rather than extrapolating to some final state.

Fourth, the utility of RTM is to allow for such things as prohibiting faulty default conclusions. That is, since STM is so small, it is likely that information that would typically block a default conclusion from being drawn has recently left STM, but it still remains in RTM. Being in RTM is sufficient to prohibit a faulty default, as we consider RTM's entries as relevant enough to have a bearing on reasoning, yet not central enough to be the catalyst of further inference.

Finally, information stored in ITM and in RTM is at times accessible to STM. Thus, information from the past can be brought back into focus when appropriate. This allows the system to use such information in working through goal-subgoal behavior as well as using past information as a default when the frame problem arises.

2.2.4 An Example

As an example of this mechanism in action, consider the following state of affairs. STM contains the fact that Tweety is a bird. LTM contains two pieces of information: the presence of bird(x) in STM is to trigger the fact that birds fly; the presence of flies(x) in STM is to trigger the fact that flying things have wings. ITM is initially empty. To simplify

[2] In part this is handled by RTM.

the example, the size of STM is fixed at a maximum size of four. A star, $\star$, indicates a newly placed item in STM.

$$\begin{array}{ll} ITM: & \emptyset \\ STM: & bird(Tweety) \\ LTM: & < bird(x), bird(x) \rightarrow flies(x) > \\ & < flies(x), flies(x) \rightarrow winged(x) > \end{array}$$

The fact that Tweety is a bird will trigger the rule that birds fly, resulting in:

$$\begin{array}{lll} STM: & & bird(Tweety) \\ & \star & bird(x) \rightarrow flies(x) \end{array}$$

An application of MP would then leave:

$$\begin{array}{lll} STM: & & bird(Tweety) \\ & & bird(x) \rightarrow flies(x) \\ & \star & flies(Tweety) \end{array}$$

Again, a new association will be triggered from LTM, resulting in:

$$\begin{array}{lll} STM: & & bird(Tweety) \\ & & bird(x) \rightarrow flies(x) \\ & & flies(Tweety) \\ & \star & flies(x) \rightarrow winged(x) \end{array}$$

This new fact would then trigger MP again (and have the side-effect of pushing $bird(Tweety)$ into ITM).

$$\begin{array}{lll} ITM: & & bird(Tweety) \\ STM: & & bird(x) \rightarrow flies(x) \\ & & flies(Tweety) \\ & & flies(x) \rightarrow winged(x) \\ & \star & winged(Tweety) \end{array}$$

2.3 Real-time Non-monotonicity

It does not take an especially large effort to produce a rudimentary form of non-monotonic reasoning using our architecture. As an illustration we present the following example. This time we start the system in the following state, where the second entry in LTM is different from before:

$$\begin{array}{ll} ITM: & \emptyset \\ STM: & bird(Tweety) \\ LTM: & < bird(x), bird(x) \rightarrow flies(x) > \\ & < ostrich(x), ostrich(x) \rightarrow \neg flies(x) > \end{array}$$

As before, the fact that Tweety is a bird will trigger the rule that birds fly, resulting in:

$$\begin{array}{lll} STM: & & bird(Tweety) \\ & \star & bird(x) \rightarrow flies(x) \end{array}$$

An application of MP would then result in:

$$\begin{array}{lll} STM: & & bird(Tweety) \\ & & bird(x) \rightarrow flies(x) \\ & \star & flies(Tweety) \end{array}$$

Now suppose the system discovers (through direct observation, or some other means) that Tweety is an ostrich. We would then have:

$$\begin{array}{lll} STM: & & bird(Tweety) \\ & & bird(x) \rightarrow flies(x) \\ & & flies(Tweety) \\ & \star & ostrich(Tweety) \end{array}$$

This new fact would then trigger the rule from LTM that ostriches do not fly (and have the side-effect of pushing $bird(Tweety)$ into ITM).

$$\begin{array}{lll} ITM: & & bird(Tweety) \\ STM: & & bird(x) \rightarrow flies(x) \\ & & flies(Tweety) \\ & & ostrich(Tweety) \\ & \star & ostrich(x) \rightarrow \neg flies(x) \end{array}$$

Again MP is applied, resulting in:

$$\begin{array}{lll} ITM: & & bird(Tweety) \\ & & bird(x) \rightarrow flies(x) \\ STM: & & flies(Tweety) \\ & & ostrich(Tweety) \\ & & ostrich(x) \rightarrow \neg flies(x) \\ & \star & \neg flies(Tweety) \end{array}$$

At this point STM contains both the belief that Tweety does not fly, as well as the belief that Tweety does fly. Is this a problem? We think not. We would like to be able to say that the fact that Tweety flies was concluded *by default*; through the use of a rule of typicality. Now given the additional information that, in fact, Tweety is an ostrich, we would like the system to be able to retract the belief that Tweety flies, and instead conclude that Tweety, in fact, does not fly.

Our approach then is first to let an inconsistency arise, and then once both α and $\neg\alpha$ are together in STM, we want to be able to decide which (if either) of the two should be kept as a belief. Since STM is small, we will always be able to determine quickly and easily whether such a direct contradiction exists.

Several methods of conflict resolution are available to us, each requiring little more than providing an extra term to facts in STM which indicates the justification for bringing that fact into focus. For example, something that is brought into STM as a result of direct observation can be tagged with the term ‘‘OBS’’, while a fact deduced through modus ponens can be tagged with ‘‘MP’’, etc. These tags then allow the system to favor, say, an observed fact over a deduced fact, and a more recent observation over an earlier one.

Notice also that all information about Tweety may soon leave STM, but will remain in RTM for some number of inference cycles (and thus still remain relevant). If at a later time (not too late, as decay out of RTM may eventually occur), Tweety is in focus again, RTM's record of Tweety's inability to fly will block the statement $flies(Tweety)$ from reappearing in STM. Thus the default rule is no longer applicable. That is, once a contradiction arises in STM and we have resolved the contradiction in favor of one of the contradictory facts, we can simply remove the other fact from STM.[3] Furthermore, we can just as easily remove this fact from RTM so that it no longer bears any relevance to the reasoning from that point on.

Any number of conflict resolution heuristics of this sort can be implemented rather easily. This is not to say that resolution of such contradictions is trivial; on the contrary, it is in general very hard, but at the very least we have a model in which to test different approaches.

With the above model of commonsense reasoning in place it is natural to try to formalize the behavior and properties exhibited by such a reasoner. We discuss this theory in the following section.

3 A Theoretical Approach

3.1 Background

We have argued that we do not want to characterize the *end results* of reasoning; rather we seek to understand reasoning as an *on-going process*. This requires that the formalism be capable of dealing with time as an object of reasoning. This can be done in ordinary logic, if the representation is in the meta-theory, through the use of a time argument to a predicate representing the agent's proof process. However, in order for the agent to reason about the passage of time that occurs *as* it reasons, time arguments must be put into the agent's own language. But now since time goes on as the agent reasons, and since this phenomenon is part of what is to be reasoned about, the agent will need to take note of facts that come and go, e.g., ''It is now 3pm and I am just starting this task . . . Now it is no longer 3pm, but rather it is 3:15pm, and I still have not finished the task I began at 3pm.'' This immediately puts us in a non-traditional setting, for we lose monotonicity: as the history evolves, conclusions may be lost. Because of this, the formalism cannot in general retain or inherit all conclusions from one step to the next. The formalism must be augmented with a notion of ''now'', which appropriately changes as deductions are performed. It turns out that this is not an easy task.

We propose step-logic then as a model of reasoning that focuses on the on-going process of deduction. As a simple example, refer to Figure 3. The reasoner starts out with an empty set of beliefs at time 0. Certain ''conclusions'' or ''observations'' may arise at discrete time steps. At some time, i, it may have belief α, concluded based on earlier beliefs, or as an observation arising at step i. At some later time, j, it comes up with $\alpha \rightarrow \beta$. Later still, the agent might deduce β. Of course, much the same might be said of any deductive logic. However in step-logic these time parameters can figure in the on-going reasoning itself. Note that we are focusing on the *on-going* reasoning process and *not* on the end results of reasoning.

The memory model (Section 2) of course proceeds in a step-like fashion, and indeed it was what motivated the effort toward the theoretical approach in this section. The memory model even records the times at which its conclusions are drawn, in further step-like fashion. However, that model does not serve well as a theoretically concise set of principles for understanding broad issues of reasoning behavior at an abstract level. For that purpose we need something more like a step-logic.

[3] In our implementation we actually retain the fact that has been determined to be incorrect, but we tag this fact in such a way so that its incorrect nature is evident. This is done so that ITM can maintain a complete chronological listing of STM facts. We feel that this will be important in the future as we may attempt to implement a learning device that scans ITM, attempting to identify *patterns of reasoning*.

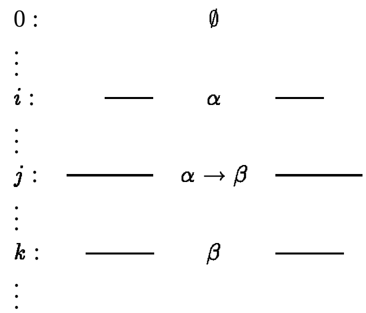

Figure 3: Step-like logical studies

3.2 Details of Step-logic

3.2.1 The Basics

A step-logic is characterized by a language, observations, and inference rules. At each step i *all* immediate consequences of the rules of inference applied to the previous step are drawn (and therefore are among the wffs at step i). However, for real-time effectiveness and cognitive plausibility, at each step we want only a finite number of conclusions to be drawn.[4]

In order for a reasoner to be able to deal appropriately with the commonsense world, three three major mechanisms are necessary: introspection, awareness of time, and retraction. Since the agent must be able to reason about its own processes, a belief, or knowledge, predicate is needed. We employ a predicate symbol, K, for this purpose: $K(i, `\alpha\text{'})$ is intended to mean that the agent knows wff $\alpha$ at time $i$.[5] We drop the quotes in $K(i, `\alpha\text{'})$ in the remainder of the paper.

The agent needs information as to how the i in $K(i, \alpha)$ relates to the on-going time as deductions are performed. This requires the agent to have information as to what time it is *now*. We achieve this through the use of the predicate expression $Now(i)$, which is intended to mean the time currently is i. Therefore, this is a belief which must change as deductions are performed.

Default reasoning is an integral part of a commonsense reasoner's thought processes. By default reasoning we mean the process of believing a particular fact when there is no evidence to suspect the contrary; however, later, in the face of new evidence, the former belief may be retracted. We allow default reasoning through the use of a retraction mechanism. Retraction is facilitated by focusing on the dual: inheritance. We do *not* assume that all deductions at time i are inherited (retained) at time $i + 1$. Thus by carefully restricting inheritance we achieve a rudimentary kind of retraction. The most obvious case is that of $Now(i)$. If at a given step the agent knows the time to be i, by having the belief $Now(i)$, then that belief is not inherited to the next time-step.

We have developed two distinct types of formalisms, that occur in pairs: the meta-theory SL^n *about* an agent, and the agent-theory SL_n itself. n is an index serving to distinguish different versions of step-logics. It is the latter, SL_n, that is to be step-like; the former, SL^n, is simply our assurance that we have been honest in describing what we mean by a particular agent's reasoning. The two theories together form a step-logic *pair*. In [5] we proposed eight such step-logic pairs, arranged in increasing sophistication, with respect to the three mechanisms above (introspection, awareness of time, and retraction), $< SL_0, SL^0 >, \ldots, < SL_7, SL^7 >$. SL_0 has none of the three mechanisms, and SL_7 has all.

[4] Indeed it should be not just finite, but small. Our current idealization does not go this far; we intend, however, to eventually make broad use of a ''retraction'' mechanism to keep things to a reasonable size. Specifically, we anticipate the introduction of a notion of relevance, along the lines we pursued in our memory model.

[5] We are not distinguishing here between belief and knowledge. See [9] for a discussion of belief vs. knowledge.

The meta-theories all are consistent, first-order theories, and therefore complete with respect to standard first-order semantics. However, their associated agent-theories are another matter. These we do not even *want* in general to be consistent, for they are (largely) intended as formal counterparts of the reasoning of fallible agents.

3.2.2 Definitions

In this section we present several definitions, most of which are analogous to standard definitions from first-order logic. Consequently certain results follow trivially from their first-order counterparts.

Intuitively, we view an agent as an inference mechanism that may be given external inputs or observations. Inferred wffs are called beliefs; these may include certain observations.

Let $\mathcal{L}$ be a first-order language, and let $\mathcal{W}$ be the set of wffs of $\mathcal{L}$.

Definition .1 *An* observation-function *is a function $OBS : \mathbf{N} \rightarrow \mathcal{P}(\mathcal{W})$, where $\mathcal{P}(\mathcal{W})$ is the powerset of $\mathcal{W}$, and where for each $i \in \mathbf{N}$, the set $OBS(i)$ is finite. If $\alpha \in OBS(i)$, then α is called an i-*observation.

Definition .2 *A* history *is a finite tuple of pairs of finite subsets of $\mathcal{W}$. $\mathcal{H}$ is the set of histories.*

Definition .3 *An* inference-function *is a function $INF : \mathcal{H} \rightarrow \mathcal{P}(W)$, where for each $h \in \mathcal{H}$, $INF(h)$ is finite.*

Intuitively, a history is a conceivable temporal sequence of belief-set/observation-set pairs. The history is a *finite* tuple; it represents the temporal sequence up to a certain point in time. The inference-function extends the temporal sequence of belief sets by one more step beyond the history. In the example in Figure 4 we see how these ideas are used to generate an actual history based on an inference-function and an observation-function. Definitions .4 and .5 formalize this in terms of a step-logic SL_n.

Let

- $OBS(i) = \begin{cases} \{bird(x) \rightarrow flies(x)\} & \text{if } i = 1 \\ \{bird(tweety)\} & \text{if } i = 3 \\ \emptyset & \text{otherwise} \end{cases}$
- $Thm_i \subseteq \mathcal{W}$, $0 \leq i < n$; $Thm_0 = \emptyset$;
- $INF(<< Thm_0, OBS(1) >, \ldots, < Thm_{n-1}, OBS(n) >>) =$
  $Thm_{n-1} \cup OBS(n) \cup \{\alpha(t) \mid (\exists\beta)(\beta(t), \beta(x) \rightarrow \alpha(x) \in (Thm_{n-1} \cup OBS(n)))\}$.

The history h of the first five steps then would be:

$$\begin{array}{rlccl} h = & << & \emptyset & ,\{bird(x) \rightarrow flies(x)\} & >, \\ & < & \{bird(x) \rightarrow flies(x)\} & , \quad \emptyset & >, \\ & < & \{bird(x) \rightarrow flies(x)\} & , \quad \{bird(tweety)\} & >, \\ & < & \{bird(x) \rightarrow flies(x), bird(tweety), flies(tweety)\}, & \emptyset & >, \\ & < & \{bird(x) \rightarrow flies(x), bird(tweety), flies(tweety)\}, & \emptyset & >> \end{array}$$

Figure 4: Example of OBS and INF

Definition .4 *An SL_n-*theory over a language *$\mathcal{L}$ is a triple, $< \mathcal{L}, OBS, INF >$, where $\mathcal{L}$ is a first-order language, OBS is an observation-function, and INF is an inference-function. We use the notation, $SL_n(OBS, INF)$, for such a theory (the language $\mathcal{L}$ is implicit in the definitions of OBS and INF). If we wish to consider a fixed INF but varied OBS, we write $SL_n(\cdot, INF)$.*

Let $SL_n(OBS, INF)$ be an SL_n-theory over $\mathcal{L}$.

Definition .5 *Let the set of 0-theorems, denoted Thm_0, be empty. For $i > 0$, let the set of i-theorems, denoted Thm_i, be $INF(<< Thm_0, OBS(1) >, < Thm_1, OBS(2) >, \ldots, < Thm_{i-1}, OBS(i) >>)$. We write $SL_n(OBS, INF) \vdash_i \alpha$ to mean α is an i-theorem of $SL_n(OBS, INF)$.*[6]

Definition .6 *Given a theory $SL_n(OBS, INF)$, a corresponding SL^n-*theory*, written $SL^n(OBS, INF)$, is a first-order theory having binary predicate symbol K,*[7] *numerals, and names for the wffs in $\mathcal{L}$, such that*

$$SL^n(OBS, INF) \vdash K(i, \alpha) \;\; \textit{iff} \;\; SL_n(OBS, INF) \vdash_i \alpha.$$

Thus in $SL^n(OBS, INF)$, $K(i, \alpha)$ is intended to express that α is an i-theorem of $SL_n(OBS, INF)$.[8]

3.2.3 SL_7

SL_7 is the most ambitious step-logic: it has all three mechanisms which we claim are necessary for a commonsense reasoner. We use the notation SL_7 for any of a family of step-logics whose OBS and INF involve the predicates Now and K and contain a retraction mechanism. Choosing OBS and INF therefore fixes the theory within the family.

SL_7 is *not* intended in general to be consistent. If SL_7 is supplied *only* with logically valid wffs that do not syntactically contain the predicate Now, then indeed SL_7 will remain consistent over time: there will be no step i at which the conclusion set is inconsistent, for its rules of inference are sound (see [6]). However, virtually all the interesting applications of SL_7 involve providing the agent with some non-logical and potentially false axioms, thus opening the way to derivation of contradictions. This behavior is what we are interested in studying, in a way that avoids the swamping problem. The controlled growth of deductions in step-logic provides a convenient tool for this, as we will see.

The language of SL_7 is first-order, having unary predicate symbol, Now, binary predicate symbol, K, and ternary predicate symbol, $Contra$, for time, introspection, and contradiction, respectively. We write $Now(i)$ to mean the time is now i; $K(i, \alpha)$ means that α is known [9] at step i; and $Contra(i, \alpha, \beta)$ means that α and β are in direct contradiction (one is the negation of the other) and both are i-theorems.

The formulae that the agent has at step i (the i-theorems) are precisely all those that can be deduced from step $i - 1$ using the applicable rules of inference. As previously stated, the agent is to have only a finite number of theorems (conclusions, beliefs, or simply wffs) at any given step. We write:

$$\begin{array}{rl} i: & \ldots, \alpha \\ i+1: & \ldots, \beta \end{array}$$

to mean that α is an i-theorem, and β is an $i+1$-theorem. There is no implicit assumption that α (or any other wff other than β) is present (or not present) at step $i+1$. The ellipsis indicates that there might be other wffs present. Wffs are not assumed to be inherited or retained in passing from one step to the next, unless explicitly stated in an inference rule. In Figure 5 below, we illustrate one possible inference function, denoted INF_B, involving a rule for special types of inheritance; see Rule 7.

[6] Note the non-standard use of the turnstile here.

[7] We see that the predicate letter K has two roles: in SL^n and in SL_n. The context will make the role clear.

[8] In [5, 4] we used $K(i, \alpha)$ for $K(i, \alpha)$.

[9] known, believed, or concluded. As already stated, we are not distinguishing between these terms. See [9, 18, 19] for a discussion of these.

For *time*, we envision a clock which is ticking as the agent is reasoning. At each step in its reasoning, the agent looks at this clock to obtain the time.[10] The wff $Now(i)$ is an i-theorem. $Now(i)$ corresponds intuitively to the statement ''The time is now i.''

Introspection involves the predicate K, and (in INF_B) a new rule of inference; see Rule 5 in Figure 5 below. This rule allows the agent to negatively introspect, i.e., to reason at step $i+1$ that it did not know β at step i.[11] To keep things cognitively plausible, and also to keep the number of conclusions at any given step finite, we allow the agent to negatively introspect only on those wffs of which it is aware. We say the agent is aware of a wff α at step i if α appears as a closed sub-formula at step i.[12] Therefore, $\neg K(i, \alpha)$ is to be deduced at step $i+1$ if α is not an i-theorem, but does appear as a closed sub-formula at step i. See [8] for another treatment of awareness.

Retractions are used to facilitate removal of certain conflicting data. Currently we handle contradictions by simply not inheriting the formulae directly involved.[13] In $SL_7(\cdot, INF_B)$, a conclusion in a given step, i, is inherited to step $i+1$ if it is not contradicted at step i and it is not the predicate $Now(j)$, for some j; see Rule 7 in Figure 5 below.

$SL_7(\cdot, INF_B)$ was formulated with applications such as the *Brother Problem* (see Section 3.2.4) in mind. This led us to the rules of inference listed in Figure 5. Rule 3 states, for instance, that if α and $\alpha \rightarrow \beta$ are i-theorems, then β will be an $i+1$-theorem. Rule 3 makes no claim about whether or not α and/or $\alpha \rightarrow \beta$ are $i+1$-theorems.

3.2.4 An Example: The Brother Problem

In this section we show how $SL_7(\cdot, INF_B)$ can be formulated to provide a real-time solution to Moore's Brother Problem (see [16]). One reasons, ''Since I don't know I have a brother, I must not.'' This problem can be broken down into two: the first requires that the reasoner be able to decide he doesn't know he has a brother; the second that, on that basis, he, in fact, does not have a brother (from *modus ponens* and the assumption that ''If I had a brother, I'd know it.'') The first of these seems to lend itself readily to step-logic, in that the negative reflection problem (determining when something is not known) reduces to a simple look-up.

We present synopses of computer-generated results for three different scenarios where the agent determines whether or not a brother exists. Let B be a 0-argument predicate letter representing the proposition that a brother exists. Let P be a 0-argument predicate letter (other than B) that represents a proposition that implies that a brother exists.[14] In each case, at some step i the agent has the axiom $P \rightarrow B$, and also the following autoepistemic axiom which represents the belief that not knowing B ''now'' implies $\neg B$.

Axiom 1 $(\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B]$

The following behaviors are illustrated:

- If B is among the wffs of which the agent is aware at step i, but not one that is believed at step i, then the agent will come to know this fact ($\neg K(i, B)$, that it was not believed at step i) at step $i+1$. As a consequence of this, other information may be deduced. In this case, the agent concludes $\neg B$ from the autoepistemic axiom (Axiom 1). Clearly the Now predicate plays a critical role.
- The agent must refrain from such negative introspection when in fact B is already known.

[10] Richard Weyhrauch analyzed this idea in a rather different way in his talk at the Sardinia Workshop on Meta-Architectures and Reflection, 1986; see [20].

[11] For a discussion of why the agent can't reason at step i about its beliefs at that same step, see [7].

[12] A sub-formula of a wff is any consecutive portion of the wff that itself is a wff. Note that there are only finitely many such sub-formulae at any given step.

[13] In future work we hope to have a mechanism for tracing the antecedents and consequents of a formula α when α is suspect, a la Doyle and deKleer (see [3, 1]), though in the context of a real-time reasoner.

[14] P might be something like ''My parents have two sons,'' together with appropriate axioms.

The inference rules given here correspond to an inference-function, INF_B. For any given history, INF_B returns the set of all immediate consequences of Rules 1--7 applied to the last step in that history. Note that Rule 5 is the only default rule.

Rule 1 :	$\dfrac{i : \ldots}{i+1 : \ldots, Now(i+1)}$	Corresponds to looking at clock
Rule 2 :	$\dfrac{i : \ldots}{i+1 : \ldots, \alpha}$	If $\alpha \in OBS(i+1)$---Obs. become beliefs
Rule 3 :	$\dfrac{i : \ldots, \alpha, \alpha \rightarrow \beta}{i+1 : \ldots, \beta}$	Modus ponens
Rule 4 :	$\dfrac{i : \ldots, P_1 a, \ldots, P_n a, (\forall x)[(P_1 x \wedge \ldots \wedge P_n x) \rightarrow Qx]}{i+1 : \ldots, Qa}$	Another version of modus ponens
Rule 5 :	$\dfrac{i : \ldots}{i+1 : \ldots, \neg K(i, \beta)}$	Negative introspection[a]
Rule 6 :	$\dfrac{i : \ldots, \alpha, \neg\alpha}{i+1 : \ldots, Contra(i, \alpha, \neg\alpha)}$	Presence of (direct) contradiction
Rule 7 :	$\dfrac{i : \ldots, \alpha}{i+1 : \ldots, \alpha}$	Inheritance[b]

[a] where β is not a theorem at step i, but is a closed sub-formula at step i.

[b] where nothing of the form $Contra(i-1, \alpha, \beta)$ nor $Contra(i-1, \beta, \alpha)$ is an i-theorem, and where α is not of the form $Now(\beta)$. That is, contradictions and time are not inherited. The intuitive reason time is not inherited is that time changes at each step. The intuitive reason contradicting wffs α and β are not inherited is that not both can be true, and so the agent should, for that reason, be unwilling to simply assume either to be the case without further justification. This does not mean, however, that neither will appear at the next step, for either or both may appear for other reasons, as will be seen. Note also that the wff $Contra(i, \alpha, \neg\alpha)$ *will* be inherited, since it is not itself either time or a contradiction, and (intuitively) it expresses a fact (that there was a contradiction at step i) that remains true.

Figure 5: Rules of inference corresponding to INF_B

- A conflict may occur if something is coming to be known while negative introspection is simultaneously leading to its negation. The third illustration shows this being resolved in an intuitive manner (though not one that will generalize as much as we would like; this is an area we are currently exploring).

Simple negative introspection succeeds In this example the agent is not able to deduce the proposition B, that he has a brother, and hence is able to deduce $\neg B$, that he does *not* have a brother. See Figure 6. For ease of reading we underline in each step those wffs which are new (i.e., which appear through other than inheritance). For the purposes of illustration, let i be arbitrary and let

$$OBS_{B_1}(j) = \begin{cases} \{P \rightarrow B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B]\} & \text{if } j = i \\ \emptyset & \text{otherwise} \end{cases}$$

Since B is not an i-observation (and thus is not an i-theorem), the agent uses Rule 5, the negative introspection rule, to conclude $\neg K(i, B)$ at step $i+1$. At step $i+2$ the agent concludes $\neg B$ from the autoepistemic knowledge stated above (Axiom 1) and the use of the alternate version of modus ponens, Rule 4.

$$\begin{aligned} i: \quad & \underline{Now(i)}, \underline{P \rightarrow B}, \underline{(\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B]} \\ i+1: \quad & \underline{Now(i+1)}, P \rightarrow B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B], \underline{\neg K(i, B)}, \underline{\neg K(i, \neg B)}, \\ & \underline{\neg K(i, P)} \\ i+2: \quad & \underline{Now(i+2)}, P \rightarrow B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B], \neg K(i, B), \neg K(i, \neg B), \\ & \neg K(i, P), \underline{\neg B}, \underline{\neg K(i+1, B)}, \underline{\neg K(i+1, \neg B)}, \underline{\neg K(i+1, P)} \end{aligned}$$

Figure 6: Negative introspection succeeds

Simple negative introspection fails (appropriately) In this example, let

$$OBS_{B_2}(j) = \begin{cases} \{P \rightarrow B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B], B\} & \text{if } j = i \\ \emptyset & \text{otherwise} \end{cases}$$

Thus the agent has B at step i, and is blocked (appropriately for this example) from deducing at step $i+1$ the wffs $\neg K(i, B)$ and $\neg B$. See Figure 7.

$$\begin{aligned} i: \quad & \underline{Now(i)}, \underline{P \rightarrow B}, \underline{(\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B]}, \underline{B} \\ i+1: \quad & \underline{Now(i+1)}, P \rightarrow B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \rightarrow \neg B], B, \underline{\neg K(i, \neg B)}, \underline{\neg K(i, P)} \end{aligned}$$

Figure 7: Negative introspection fails appropriately

Note that a traditional final-tray-like approach could produce quite similar behavior to that seen in Figures 6 and 7 if it is endowed with a suitable introspection device, although it would not have the real-time step-like character we are trying to achieve.

Introspection contradicts other deduction In this example, let

$$OBS_{B_3}(j) = \begin{cases} \{P \to B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \to \neg B], P\} & \text{if } j = i \\ \emptyset & \text{otherwise} \end{cases}$$

In Figure 8 we see then that the agent does not have B at step i, but is able to *deduce* B at step $i+1$ from $P \to B$ and P at step i. Since the agent is *aware* (in our sense) of B at step i, and yet does not have B as a *conclusion* at i, it will deduce $\neg K(i, B)$ at step $i+1$. Thus both B and $\neg K(i, B)$ are concluded at step $i+1$. At step $i+2$ Axiom 1 (the autoepistemic axiom), together with $Now(i+1)$ and $\neg K(i, B)$ and Rule 4, will produce $\neg B$. A conflict results, which is noted at step $i+3$. This then inhibits inheritance of both B and $\neg B$ at step $i+4$. Although neither B nor $\neg B$ is *inherited* to step $i+4$, B is *re-deduced* at step $i+4$ via modus ponens from step $i+3$. Thus B ''wins out'' over $\neg B$ due to its existing justification in other wffs, while $\neg B$'s justification is ''too old'': $\neg K(i+2, B)$, rather than $\neg K(i, B)$, would be needed. We see then that the conflict resolves due to the special nature of the time-bound ''now'' feature of introspection.

$$\begin{array}{rl}
i: & \underline{Now(i)}, \underline{P \to B}, \underline{(\forall x)[(Now(x) \wedge \neg K(x-1, B)) \to \neg B]}, \underline{P} \\[1em]
i+1: & \underline{Now(i+1)}, P \to B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \to \neg B], P, \underline{B}, \underline{\neg K(i, B)}, \underline{\neg K(i, \neg B)} \\[1em]
i+2: & \underline{Now(i+2)}, P \to B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \to \neg B], P, B, \neg K(i, B), \neg K(i, \neg B), \\
 & \underline{\neg B}, \underline{\neg K(i+1, \neg B)} \\[1em]
i+3: & \underline{Now(i+3)}, P \to B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \to \neg B], P, B, \neg K(i, B), \neg K(i, \neg B), \\
 & \neg B, \neg K(i+1, \neg B), \underline{Contra(i+2, B, \neg B)} \\[1em]
i+4: & \underline{Now(i+4)}, P \to B, (\forall x)[(Now(x) \wedge \neg K(x-1, B)) \to \neg B], P, \neg K(i, B), \neg K(i, \neg B) \\
 & \neg K(i+1, \neg B), Contra(i+2, B, \neg B), \underline{B}, \underline{Contra(i+3, B, \neg B)}
\end{array}$$

Figure 8: Introspection conflicts with other deduction and resolves

A traditional final-tray-like approach would encounter difficulties with this third example, for at step $i+2$ there is a contradiction. This means that the final tray for a tray-like model of a reasoning agent would simply be filled with all wffs in the language---and no basis for a resolution is possible *within* such a logic.

4 Future Work

We have presented a brief overview of a model of memory that is motivated by a psychological approach to reasoning. Combining that model with the idea that reasoning occurs in time has led us to build a computational model of reasoning; one that isn't bound to a final state, conclusion, or goal. It turns out that the foundational theory that we have developed to study this model has the nice property of being grounded in first-order logic without suffering the unpleasantries typically associated with contradiction.

This, however, is only the beginning of our work. We have identified at least three major categories of research that is yet to be undertaken. First there is a need to extend and refine the memory model's implementation. Similarly, step-logic can be extended to further conjoin it to the memory model. Finally, we have identified problems in reasoning research that the memory model and step-logic seem well-suited to handle. Below is a short list of endeavors that fall into one or more of these categories. The list is certainly not exhaustive of related research, nor do we expect that the items on it are exclusive of one another. We do, however, feel that it illustrates the breadth of ideas that are relevant to this work.

Parallel retrieval. If LTM is really intended to represent long term memory, then it makes sense to retrieve associations concurrently. There is certainly no benefit to be had from serial retrieval, and it may, in fact, impose orderings on STM that are undesirable. We seek a model of concurrent retrieval that may (or may not) require a different implementation representation of LTM, but doesn't infringe on the general model.

Restricting STM. We need a robust mechanism that will restrict the size of STM. LTM is likely to be very, very large. Lots of associations can be expected at each inference cycle. Yet it seems that context ought to be able to prevent some of these associations from actually entering STM. After all, it isn't necessary, nor would it be desirable, to recall each time the concept of bird is encountered that male birds are generally more colorful than females . But, if the task is to identify a particular bird, this fact is indeed relevant. We hope to extend the power of RTM to handle this task.

Formalize LTM in step-logic. To better understand the memory model, it is necessary to make precise more of its components. Step-logic is the formalism we would like to use to do this. There is currently no notion of association or retrieval in step-logic, though there doesn't seem to be anything that would prohibit this addition to the theory.

Make LTM dynamic. The utility of LTM is severely limited unless it can be updated with both additions and deletions. This ability is part of any psychological model of intelligence and will add a necessary flexibility to the memory model. Our hope is that a dynamic LTM is one way to model learning in the memory model. (It's too early to make any claim about sufficiency.) An interesting addendum to this would be to formalize the dynamic LTM in step-logic.

Plan generation in step-logic. The Nell and Dudley problem is paradigmatic. In it, time is explicitly part of the problem solving process. Although time is explicit in step-logic, it has not yet been used in a real-time problem solving setting.

References

[1] J. deKleer. An assumption-based TMS. *Artificial Intelligence*, 28:127--162, 1986.

[2] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12(3):231--272, 1979.

[3] J. Doyle. Some theories of reasoned assumptions: An essay in rational psychology. Technical report, Department of Computer Science, Carnegie Mellon University, 1982.

[4] J. Drapkin and D. Perlis. A preliminary excursion into step-logics. In *Proceedings SIGART International Symposium on Methodologies for Intelligent Systems*, pages 262--269. ACM, 1986. Knoxville, Tennessee.

[5] J. Drapkin and D. Perlis. Step-logics: An alternative approach to limited reasoning. In *Proceedings of the European Conf. on Artificial Intelligence*, pages 160--163, 1986. Brighton, England.

[6] J. Elgot-Drapkin. *Step-logic: Reasoning Situated in Time*. PhD thesis, Department of Computer Science, University of Maryland, College Park, Maryland, 1988.

[7] J. Elgot-Drapkin and D. Perlis. Reasoning situated in time i: Basic concepts. Technical Report CS-TR-2016, Department of Computer Science, University of Maryland, College Park, Maryland, April 1988.

[8] R. Fagin and Y. Halpern, J. Belief, awareness, and limited reasoning. *Artificial Intelligence*, 34(1):39--76, 1988.

[9] E. Gettier. Is justified true belief knowledge? *Analysis*, 23:121--123, 1963.

[10] C. Glymour and R. Thomason. Default reasoning and the logic of theory perturbation. In *Proceedings of the Workshop on Non-monotonic Reasoning*. AAAI, October 1984. New Paltz, New York.

[11] J. Goodwin. *A Theory and System for Non-Monotonic Reasoning*. PhD thesis, Department of Computer and Information Science, Linkoping University, Linkoping, Sweden, 1987.

[12] K. Konolige. A deductive model of belief. In *Proceedings of the 8th Int'l Joint Conf. on Artificial Intelligence*, pages 377--381, 1983. Karlsruhe, West Germany.

[13] H. Levesque. A logic of implicit and explicit belief. In *Proceedings of the 3rd National Conf. on Artificial Intelligence*, pages 198--202, 1984. Austin, TX.

[14] J. McCarthy. Circumscription-a form of non-monotonic reasoning. *Artificial Intelligence*, 13(1,2):27--39, 1980.

[15] G. Miller. The magical number seven plus or minus two. *The Psychological Review*, 63:81--97, 1956.

[16] R. Moore. Semantical considerations on nonmonotonic logic. In *Proceedings of the 8th Int'l Joint Conf. on Artificial Intelligence*, 1983. Karlsruhe, West Germany.

[17] N. Nilsson. Artificial intelligence prepares for 2001. *AI Magazine*, 4(4):7--14, 1983.

[18] D. Perlis. On the consistency of commonsense reasoning. *Computational Intelligence*, 2:180--190, 1986.

[19] D. Perlis. Languages with self reference II: Knowledge, belief, and modality. *Artificial Intelligence*, 34:179--212, 1988.

[20] R. Weyhrauch. The building of mind. In *1986 Workshop on Meta-Architectures and Reflection*, 1986. Sardinia, Italy.