Reasoning with limited resources: An LDS-based approach

Mikael Asker and Jacek Malec
Department of Computer Science
Lund University
Box 118
221 00 Lund, Sweden
mikael.asker@fagotten.org, jacek@cs.lth.se

Abstract. Reasoning with limited computational resources (such as time or memory) is an important problem, in particular in knowledge-intensive embedded systems. Classical logic is usually considered inapropriate for this purpose as no guarantees regarding deadlines can be made. One of the more interesting approaches to address this problem is built around the concept of *active logics*. Although a step in the right direction, active logics still do not offer the ultimate solution.

Our work is based on the assumption that *Labelled Deductive Systems* offer appropriate metamathematical methodology to study the problem. As a first step, we have reformulated a pair of systems, namely the *step logic* and the *memory model* proposed by Elgot-Drapkin, as Labelled Deductive Systems. This paper presents our motivation behind this project and presents in some detail the first system.

1 Introduction

Reasoning with limited computational resources (such as time or memory) is an important problem, in particular in knowledge-intensive embedded systems. Usually a decision, based on a principled reasoning process, needs to be taken within limited time and given constraints on the processing power and resources of the reasoning system. Therefore symbolic logic is often considered as an inadequate tool for implementing reasoning in such systems: logic does not guarantee that all relevant inferences will be made within prescribed time nor does it allow to limit the required memory resources satisfactorily. The paradigm shift that occured in Artificial Intelligence in the middle of 1980s can be attributed to increasing awareness of those limitations of the predominant way of representing and using knowledge.

Since then there have been some attempts to constrain the inference process performed in a logical system in a principled way. One possibility is to limit the expressive power of the first-order logical calculus (as, e.g., in description logics) in order to guarantee polynomial-time computability. Another is to use polynomial approximations of the reasoning process. Yet another is to constrain the inference process in order to retain control over it. More details about each of these lines of research can be found in Section 6.

One of the more interesting lines of research in this area during 1990s has focused on logic as a model of an on-going reasoning process rather than as a static characterization of contents of a knowledge base. It begun with step-logic [6] and evolved into a family of *active logics*. The most recent focus of this research is on modeling dialog and discourse. However, other interesting applications like planning or multi-agent systems have also been investigated, while some other possibilities wait for analysis. In particular, the possibility of applying this framework to resource-bounded reasoning in embedded systems is in the focus of our interest.

Finally, one should name the relations to the large area of *belief revision* that also investigates the process of knowledge update rather than the static aspects of logical theories. However, there has been recently little attention paid to possibilities of using this approach in resource-bounded reasoning - the work has rather focused on the non-monotonicity aspect of knowledge revision process.

The rest of the paper is divided as follows. Section 2 presents the background of the idea leading to our investigation. Section 4 presents an LDS-based formalization of a step logic. Next section presents a formalization of a more complex system, involving a psychologically plausible memory model. In Section 6 we briefly discuss related work. Finally the conclusions and some suggestion of further work are presented.

2 Background

The very first idea for this investigation has been born from the naive hypothesis that in order to be able to use symbolic logical reasoning in a real-time system context it would be sufficient to limit the depth of reasoning to a given, predefined level. This way one would be able to guarantee predictability of a system using this particular approach to reasoning. Unfortunately, such a modification performed on a classical logical system yields a formalism with a heavily modified and, in principle, unknown semantics [20]. It would be necessary to relate it to the classical one in a thorough manner. This task seems very hard and it is unclear for us what techniques should be used to proceed along this line. But the very basic idea of "modified provability": *A formula is a theorem iff it is provable within n steps of reasoning*, is still appealing and will reappear in various disguises in our investigations.

The next observation made in the beginning of this work was that predictability (in the hard real-time sense) requires very tight control over the reasoning process. In the classical approach one specifies a number of axioms and a set of inference rules, and the entailed consequences are expected to "automagically" appear as results of an appropriate consequence relation. However, this relation is usually very hard to compute and normally requires exponential algorithms, while a useful (from practical point of view) consequence relation must necessarily be practically computable. One possibility is to modify the consequence relation in such way that it becomes computable. However, the exact way of achieving it far from obvious. We have investigated previous approaches (listed in Section 6) and concluded that a reasonable technique for doing this would be to introduce a mechanism that would allow one to control the inference process. One such mechanism is available in Labelled Deductive Systems [9].

In its most simple, somewhat trivialized, setting a labelled deductive system (LDS) attaches a *label* to every well-formed formula and allows the inference rules to analyze and modify labels, or even trigger on specific conditions defined on the labels. E.g. instead of the classical Modus Ponens rule

$$\frac{A, A \to B}{B}$$

a labelled deduction system would use

$$\frac{\alpha : A, \quad \beta : A \to B}{\gamma : B}$$

where α, β, γ belong to a well-defined language (or, even better, algebra defined over this language) of labels, and where γ would be an appropriate function of α and β. If we were to introduce our original idea of limited-depth inference, then γ could be, e.g., $\max(\alpha, \beta) + 1$ provided that α and β are smaller than some constant N.

A similar idea, although restricted to manipulation of labels which denote time points, has been introduced in *step-logic* [6] which later evolved into a family of *active logics* [8]. Such a restriction is actually a reasonable first step towards developing a formal system with provable computational properties. Active logics have been used so far to describe a variety of domains, like planning [19], epistemic reasoning [7], reasoning in the context of resource limitations [16] or modeling discourse. We are definitely interested in pursuing this line of investigations, however in a manner that is more amenable to metamathematical investigations. LDS seems to be a perfect technical choice for that purpose. In particular, various possibilities offered by the freedom of choice of the labeling algebras used to define the inference rules can be studied. Properties of the consequence relations defined this way are definitely worth analyzing in order to gather understanding of what can be achieved in the resource-limited setting, and what (semantical) price is paid for this.

3 Active Logics

Active logics originated from an attempt to formalize a memory model, inspired by cognitive psychology research, which was studied at the University of Maryland during the 1980s [4]. It has been first formalized by *step logic* (cf. section 3.2). However, this formalisation has left many of the interesting properties of the model outside its scope. In order to justify our interest in step logic we begin with a short presentation of the original model, followed by the description of the original formalisation due to Elgot-Drapkin and Perlis.

3.1 The Memory Model

The memory model (MM later on) consists of five parts:

- LTM, the *long term memory*, which contains rules consisting of pairs of formulae: (trigger, contents). Semantic retrieval is associative, based on trigger formulae.
- STM, the *short term memory*, which acts as the current focus of attention. All new inferences must include a formula from the STM.
- QTM, the *quick term memory*, which is a technical device for buffering the next cycle's STM content.
- RTM, the *relevant term memory*, which is the repository for default reasoning and relevance. It contains formulae which have been pushed out of the STM but still may be important for default resolution.

- ITM, the *intermediate term memory*, which contains all facts which have been pushed out of the STM. The contents of the ITM provides the history of the agents reasoning process. ITM provides support for goal-directed behavior.

Three of the parts, LTM, STM and ITM, originate from cognitive pychology research. The other two, QTM and RTM, have been invented by Drapkin, Miller and Perlis, as auxiliary technical devices. Figure 1 shows how the parts are connected to each other.

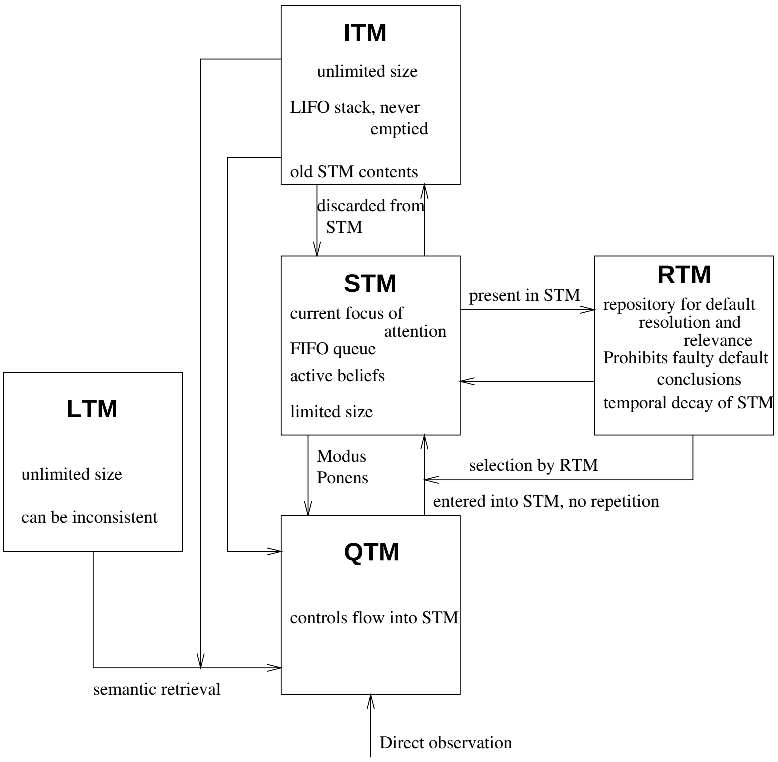

Figure 1: The memory model from [4].

The memory model has been employed for simulations of reasoning for some simple examples. An interesting property observed by its authors was that the size of STM sufficient for non-trivial reasoning tasks was eight — surprisingly close to the size of the human STM.

3.2 Step logic

The chronologically first active logic was created in 1988 [6], then named *step logic*. Actually, step logic is not a single system but a family of different logics. Each of the step logics in the family contains two distinct types of formalisms: the *metatheory* SL^n *about* the reasoning agent and the *agent theory* SL_n itself, which is step-like in the sense that the formalism refers explicitly to the reasoning steps. The two theories together form a *step-logic pair* $< SL_n, SL^n >$.

The subscript n serves to distinguish different versions of the step logics. The versions differ in the mechanisms that the agent has at its disposal: self knowledge, time and retraction. Self knowledge gives the agent the capability to introspect and determine what it does and does not know. The time mechanism allows the on-going process of deduction to be part of the agent's own reasoning. Retractions can be used to implement non-monotonic reasoning.

Elgot-Drapkin proposed eight step-logic pairs, arranged in increasing sophistication with respect to the three mechanisms above. The agent theories and the mechanisms that they included were as follows (S = self knowledge, T = time, R = retraction):

SL_0: none	SL_2: T	SL_4: S, R	SL_6: R, T
SL_1: S	SL_3: R	SL_5: S, T	SL_7: S, T, R

Beliefs are parameterized by the time taken for their inference.

In the following definitions, $\mathbb{N}$ (the set of natural numbers) is used as a model of discrete time. S_{wff} is the set of all well-formed formulae of the language of predicate logic. $\mathcal{P}(S)$ denotes the power set of the set S.

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

Definition 2 (History). *A* history *is a finite tuple of belief set/observation set pairs; the sets are finite subsets of S_{wff}:*

$$<< Thm_0, OBS(1) >, < Thm_1, OBS(2) >, ..., < Thm_{i-1}, OBS(i) >>$$

$\mathcal{H}$ denotes the set of all histories.

Intuitively, a history is a conceivable temporal sequence of belief set/observation set pairs.

The *inference function* extends the temporal sequence of belief sets by one more step beyond the history:

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

We can now define the first type of formalism, the agent theory:

Definition 4 (SL_n-theory). *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).*

Members of the set S_{wff} of well-formed formulae over the language $\mathcal{L}$ are called *agent wffs*. SL_n-theories will often be called *agent theories*.

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

Intuitively, α is an i-theorem if it can be inferred in i steps from the observations.

Definition 6 (Meta-theory corresponding to SL_n). *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, numerals and names for the wffs in S_{wff}, such that*

$$SL^n(OBS, INF) \vdash K(i, \ulcorner\alpha\urcorner) \text{ iff } SL_n(OBS, INF) \vdash_i \alpha$$

In $SL^n(OBS,INF)$, $K(i,\ulcorner\alpha\urcorner)$ is intended to express that α is an i-theorem of $SL_n(OBS,INF)$.

The predicate letter K has two roles: in SL^n it is used to reason *about* the knowledge of the agent while in SL_n it may be used as an autoepistemic knowledge operator. In the agent theory it refers to constant symbols which are *names* of agent wffs.

A notion of completeness for a meta-theory can be defined as follows:

Definition 7 (Analytical completeness). *A meta-theory SL^n is* analytically complete*, if for every positive integer i, and every constant α naming an agent wff of the corresponding agent theory, either $SL^n \vdash K(i,\ulcorner\alpha\urcorner)$ or $SL^n \vdash \neg K(i,\ulcorner\alpha\urcorner)$.*

SL^0 is in fact analytically complete [6].

In order to talk about the semantics of SL_n we need to define its models.

Definition 8 (Step interpretation). *Let $\mathcal{L}'$ be a language having the symbols of $\mathcal{L}$ and the (possibly additional) predicate symbols K and Now. A* step-interpretation *for $\mathcal{L}'$ is a sequence $M =< M_0, M_1, ..., M_i, ... >$, such that*

1. *Each M_i is an ordinary first-order interpretation of $\mathcal{L}'$.*
2. *$M_i \models Now(i)$.*

Definition 9 (Step model). *A* step model *for $SL_n(OBS, INF)$ is a step-interpretation M satisfying for all $i \in \mathbb{N}$*

1. *$M_i \models K(i,\ulcorner\alpha\urcorner)$ iff $SL_n(OBS, INF) \vdash_i \alpha$.*
2. *$M_i \models \alpha$ whenever $SL_n(OBS, INF) \vdash_i \alpha$.*

Condition 1 ensures that a historical record of the i-theorems exists. Condition 2 ensures that the i-theorems are in fact true.

Definition 10 (i-truth in a step model). *A wff α is* i-true in a step model M *($M \models_i \alpha$) if $M_i \models \alpha$.*

Definition 11 (Stepwise consistent theory). *$SL_n(OBS, INF)$ is* stepwise consistent *if for each $i \in \mathbb{N}$, the set of i-theorems is consistent.*

Even if $SL_n(OBS,INF)$ is stepwise consistent, it can have conflicting wffs at different steps. For example, $SL_n(OBS, INF) \vdash_{10} Now(10)$ and $SL_n(OBS, INF) \vdash_{11} \neg Now(10)$.

Definition 12 (Eventually consistent theory). *$SL_n(OBS, INF)$ is* eventually consistent *if $\exists i \forall j > i$, the set of j-theorems is consistent.*

Any stepwise consistent theory is eventually consistent.

Definition 13 (Finite observation function). *An observation function OBS is* finite *if $\exists i \forall j > i, OBS(j) = \emptyset$.*

Definition 14 (Self-stabilizing theory). *$SL_n(\cdot, INF)$ is* self-stabilizing *if for every finite OBS, $SL_n(OBS, INF)$ is eventually consistent.*

Intuitively, a self-stabilizing theory $SL_n(\cdot, INF)$ corresponds to a fixed agent that can regain and retain consistency after being given arbitrarily (but finitely) many contradictory initial beliefs.

Theorem 1 (Soundness, [6]). *Every step-logic $SL_n(OBS, INF)$ is sound with respect to step-models. That is, every i-theorem α of $SL_n(OBS, INF)$ is i-true in every step-model M of $SL_n(OBS, INF)$, i.e., if $SL_n(OBS, INF) \vdash_i \alpha$ then $M \models_i \alpha$.*

The major problem with step logics is that the set of beliefs may grow in an uncontrolled fashion. When the memory model was formalized in step logic, some of its good properties were lost. In particular, in the memory model the short term memory (STM) simulates a focus of attention. The size of the STM limits the number of inferences per step and thus reduces combinatorial explosion which is important in implementations. In step logics this limitation is lost so that the number of formulae in each step increases rapidly.

4 Step logic as an LDS

As the first step in the process sketched above, we have chosen a simple active logic, namely the step logic SL_7 defined in [6]. Below we present a possible reformulation of SL_7 using LDS and then, in the next section, we mention how this formalization can be extended to the original MM. It needs to be stressed that we formalize only the agent theory SL_7, leaving out the metatheory SL^7 outside the scope of this presentation.

4.1 LDS

Traditionally a logic was perceived as a consequence relation on a *set* of formulae. Problems arising in some application areas have emphasized the need for consequence relations between *structures* of formulae, such as multisets, sequences or even richer structures. This finer-tuned approach to the notion of a logical system introduces new problems which call for an improved general framework in which many of the new logics arising from computer science applications can be presented and investigated. LDS, *labelled deductive systems*, was presented in [9] as such a unifying framework.

The first step in understanding LDS is to understand the intuitive message, which is very simple: Traditional logics manipulate formulae, while an LDS manipulates *declarative units*, pairs $formula : label$ of formulae and labels. The labels should be viewed as more information about the formulae, which is not encoded inside the formulae. E.g., they can contain reliability (in an expert system), where and how a formula was deduced, or time stamps.

A *logic* is here a pair $(\vdash, S_\vdash)$ where $\vdash$ is a structured, possibly non-monotonic consequence relation on a language L and $S_\vdash$ is an LDS which is used to present the consequence relation. $\vdash$ is essentially required to satisfy no more than identity (i.e. $\{A\} \vdash A$) and a version of cut.

A simple form of LDS is the *algebraic LDS*. There are more advanced variants, *metabases*, in which the labels can be databases.

An *LDS proof system* is a triple $(\mathcal{A}, \mathbf{L}, \mathbb{R})$ where $\mathcal{A}$ is an algebra of labels (with some operations), $\mathbf{L}$ is a logical language (connectives and wffs) and $\mathbb{R}$ is a a family of deduction rules. $\mathbb{R}$ implicitly contains a discipline of labeling formulae of the logic (with labels from the algebra A), together with a notion of a *database* and with agreed ways of propagating the labels via application of the deduction rules.

4.2 LDS for SL_7

The *observation function*, $OBS : \mathbb{N} \to \mathcal{P}(S_{wff})$, where S_{wff} is the set of well-formed formulae of SL_7 and $\mathcal{P}(X)$ denotes the power set of a set X, is used to generate axioms. For every $i \in \mathbb{N}$, $OBS(i)$ is assumed to be finite.

As labels we use the natural numbers that represent the time at which a formula has been asserted (observed or deduced); simply, $S_{labels} \stackrel{df}{=} \mathbb{N}$. The *declarative units* of the system are pairs $label : formula$:

$$S_{du} \stackrel{df}{=} S_{labels} \times S_{wff} \tag{1}$$

Sometimes, to improve readability, we use the notation $\boxed{label : formula}$, instead of just $label : formula$, where the box serves purely as a delimiter and has no additional meaning by itself.

The axioms below (actually, axiom schemata) express either the time flow or results of observations. At each time point i the formula $Now(i)$ is true. If OBS is the observation function then at every time point i we assert observations associated with this time point.

(A1)	$i : Now(i)$	for all $i \in \mathbb{Z}_+$	CLOCK
(A2)	$i : \alpha$	for all $\alpha \in OBS(i),\ i \in \mathbb{Z}_+$	OBSERVATIONS

Except for the clock axioms, the Now predicate has no special status in the system. Just like in step logic it is part of the time mechanism and it should be used in applications to control the reasoning process.

The inference rules used in the system, $\mathbb{R}_{SL_7}$, come from INF_B defined in [6], although in somewhat modified form in order to take account of labels. First come two versions of Modus Ponens:

(I1) $$\frac{i : \alpha, i : \alpha \to \beta}{i + 1 : \beta}$$ MODUS PONENS

(I2) $$\frac{i : P_1 a, \ldots, i : P_n a, i : (\forall x)[(P_1 x \wedge \ldots \wedge P_n x) \to Qx]}{i + 1 : Qa}$$ EXTENDED MODUS PONENS

I.e., all the prerequisites of the Modus Ponens rules must be present at a time point i in order to assert the conclusion at the time point $i + 1$.

The next rule, Negative Introspection, allows one to infer lack of knowledge of a particular formula at time i. In order to express that we need to define the set $S_{th}(i)$ of conclusions that can be drawn at time i. We begin with the set of all axioms, S_{axioms}, obtained from (A1) and (A2). Then we look for all formulae that can be derived from it using the rules of inference. Finally, we choose out those that have the label i in front and call them *i-theorems*:

$$S_{th}(i) \stackrel{df}{=} \left\{ \boxed{i : \alpha} \in S_{du} \mid S_{axioms} \vdash \boxed{i : \alpha} \right\} \tag{2}$$

$S_{th}(i)$ can be computed by purely syntactical operations and it can be defined recursively using the inference rules. It is well-defined for every $i \in \mathbb{N}$ because the consequence relation is "directed" by the natural ordering of the set $\mathbb{N}$. Every inference rule necessarily increments the label. Therefore all the elements in $S_{th}(i)$ will be inferred from a finite number of instances of axiom (A1), namely those for which labels vary between 0 and $i-1$, and from the finite amount of observations performed until the time i. As every inference rule increments the label, only a finite number of applications of every rule is possible before the label reaches i.

Given a finite set $S_{th}(i)$ of i-theorems, we can identify all closed subformulae occurring in them and not occuring as separate theorems. The process of finding all closed subformulae for a given finite set of formulae is computable. We begin with the "is a closed subformula" relation $\mathcal{R}_{csf} \subseteq S_{wff} \times S_{wff}$, which can be easily tested. Then we define the function f_{csf}, $f_{csf} : \mathcal{P}(S_{du}) \to \mathcal{P}(S_{wff})$, constructing (unlabelled) closed subformulae out of a given set of (labelled) formulae $S \in \mathcal{P}(S_{du})$:

$$f_{csf}(S) \stackrel{df}{=} \{\alpha \in S_{wff} | (\exists \boxed{i : \beta} \in S)(\alpha \mathcal{R}_{csf} \beta)\} \tag{3}$$

Finally, the (projection) function $f_{formulae}$, $f_{formulae} : \mathcal{P}(S_{du}) \to \mathcal{P}(S_{wff})$ extracts unlabelled formulae out of a set $S \in \mathcal{P}(S_{du})$ of labelled formulae:

$$f_{formulae}(S) \stackrel{df}{=} \{\alpha \in S_{wff} | (\exists i \in S_{labels})(\boxed{i : \alpha} \in S)\} \tag{4}$$

We can now formulate the Negative Introspection rule:

$$\text{(I3)} \quad \frac{\alpha \in f_{csf}(S_{th}(i)), \alpha \notin f_{formulae}(S_{th}(i))}{i+1 : \neg K(i, \ulcorner \alpha \urcorner)} \qquad \text{NEGATIVE INTROSPECTION}$$

The agent is supposed to be aware of all the closed subformulae in $S_{th}(i)$. They provide a relevant and finite subset of S_{wff} for the self-knowledge mechanism to operate on.

Apparently, rule (I3) is used to introduce in SL_7 a possibility of non-monotonic autoepistemic reasoning of the agent. Except for rule (I3), the K predicate has no special status in the system. Just like in step logic it is part of the self-knowledge mechanism and it should be used in applications appropriately.

Finally, we can define two rules that propagate consistent knowledge onward in time.

$$\text{(I4)} \quad \frac{i : \alpha, i : \neg\alpha}{i+1 : Contra(i, \ulcorner \alpha \urcorner, \ulcorner \neg\alpha \urcorner)} \qquad \text{CONTRADICTION DETECTION}$$

$$\text{(I5)} \quad \frac{\begin{array}{l} i : \alpha \\ Contra(i-1, \ulcorner \alpha \urcorner, \ulcorner \beta \urcorner) \notin f_{formulae}(S_{th}(i)) \\ Contra(i-1, \ulcorner \beta \urcorner, \ulcorner \alpha \urcorner) \notin f_{formulae}(S_{th}(i)) \\ \alpha \neq Now(i) \end{array}}{i+1 : \alpha} \qquad \text{INHERITANCE}$$

We can now define the LDS which is intended to express the SL_7 agent theory:

Definition 15. $\mathbb{L}_{SL_7} \stackrel{df}{=} (\mathbb{N}, \mathbf{L}, \mathbb{R}_{SL_7})$, *where* $\mathbb{N}$ *denotes the algebra consisting of the set of natural numbers, with the successor (+1) as the only operation,* $\mathbf{L}$ *is the first order language and the set of inference rules* $\mathbb{R}_{SL_7}$ *is (I1)-(I5).*

A database $\Delta_{ax}(OBS)$ containing the declarative units in S_{axioms} can be generated from the observation function OBS with another function Δ_{ax}.

It is now possible to prove that $\mathbb{L}_{SL_7}$ works as expected:

Theorem 2. *A formula α can be derived in the step logic SL_7(OBS,INF_B) at time point i if and only if it can be derived as $\boxed{i : \alpha}$ in the labelled deductive system $\mathbb{L}_{SL_7}$ defined above. For all observation functions OBS, points in time i, and well-formed formulae α:*

$SL_7(OBS, INF_B) \vdash_i \alpha \Leftrightarrow \Delta_{ax}(OBS) \vdash_{\mathbb{L}_{SL_7}} \boxed{i : \alpha}$

Proof:

The statement above is the same as $\alpha \in Thm_i \Leftrightarrow \boxed{i : \alpha} \in S_{th}(i)$ which is the same as $Thm_i = f_{formulae}(S_{th}(i))$. That can be proved by induction over time: Thm_0 and $S_{th}(0)$ are empty by definition so $Thm_i = f_{formulae}(S_{th}(i))$ is true for $i = 0$. Assume that $Thm_i = f_{formulae}(S_{th}(i))$ is true for i. By definition $Thm_{i+1} = INF_B(<< Thm_0, OBS(1) >, < Thm_1, OBS(2) >, ..., < Thm_i, OBS(i+1) >>)$. INF_B applies its inference rules to Thm_i and "adds" the result of this to the observations in $OBS(i+1)$. In the LDS $\mathbb{L}_{SL_7}$ we get from $S_{th}(i)$ to $S_{th}(i+1)$ by applying the $\mathbb{R}_{SL_7}$ rules to the declarative units in $S_{th}(i)$ and "adding" the result of this to the axioms with label $i + 1$. The resulting unlabelled formulae are the same in both systems, which can be found by comparing the inference rules one by one. □

5 A note on Elgot-Drapkin's memory model as an LDS

In our opinion the formalisation of MM in step logic is an oversimplification. In particular, the STM size limit is omitted so that the number of formulae in each step may increase rapidly. This problem has also been recognized in later research of the active logic group ([16], [15] and [12]).

Our next step is to extend the $\mathbb{L}_{SL_7}$ to include all aspects of MM. This time the labels get much more complicated:

$$S_{labels} \stackrel{df}{=} \{LTM, QTM, STM, ITM\} \times S_{wff} \times \{C, U\} \times \mathbb{N} \times \mathbb{N} \times \mathbb{N} \tag{5}$$

where the interpretation of a tuple in S_{labels} is the following. If

$(location, trigger, certainity, time, position, \textit{time-left-in-rtm}) \in S_{labels}$

is a label, then *location* encodes the memory bank location of the formula (one of LTM, QTM, STM or ITM), *trigger* is used for encoding the triggering formula for LTM items (in particular, ε is used to denote the empty triggering formula), *certainity* is used in case of defeasible reasoning to encode the status of the formula (certain or uncertain), *time* is the inference time, *position* denotes the formula's position in STM or ITM, and, finally, *time-left-in-rtm* denotes the time the labelled formula should remain in the RTM. $R \in \mathbb{N}$ is a constant used to limit the time a formula remains in RTM after it has left STM.

The inference process must be defined very thoroughly in order to capture all the intricacies of the model, but the results are satisfactory. We expect to proceed along this line of thought further in our project. The details of the current status may be found in [1].

6 Related work

The attempts to constrain in a principled way the inference process performed in a logical system have been done as long as one has used logic for knowledge representation and reasoning. One possibility is to limit the expressive power of the first-order logical calculus (as, e.g., in description logics) in order to guarantee polynomial-time computability. There is a number of theoretical results in this area (see e.g [5]) but we are rather interested in investigations aimed at practical computational complexity. One of the more popular approaches is to use a restricted language (like, e.g., description logics), see [11, 17, 18] for examples of this approach in practice.

Another possibility is to use polynomial approximations of the reasoning process. This approach is tightly coupled to the issue of theory compilation. The most important contributions in this area are [20, 2, 3, 13]. However, this approach, although it substantially reduces the computational complexity of the problem, still does not provide tight bounds on the reasoning process.

Yet another possibility is to constrain the inference process in order to retain control over it. An early attempt has been reported in [14]. The next step in this direction was the step-logic [6] that evolved into a family of *active logics* [8]. Such a restriction is actually a reasonable first step towards developing a formal system with provable computational properties. Active logics have been used so far to describe a variety of domains, like planning [19], epistemic reasoning [7], reasoning in the context of resource limitations [16] or modeling discourse and dialog.

There is a growing insight that logic, if it is to be considered as a useful tool for building autonomous intelligent agents, has to be used in a substantially different way than before. Active logics are one example of this insight, while other important contributions might be found, e.g., in [10] or [21].

7 Conclusions and future work

We have presented an LDS-based formalization for an early active logic, namely SL_7 and mentioned a similar result for its more complex predecessor, the memory model. This allows us to expect that even in the case of more complex, time-limited reasoning patterns, LDS will appear to be a useful and powerful tool. Actually, the problem with restricting the inference rule applications to a particular order in order to get hold of non-monotonic dependencies, can be solved satisfactorily by just extending the labeling algebra and then constraining the inference rule invocations by appropriately constructed predicates over these labels. This result, both for SL_7 and MM formalizations, will be reported soon [1].

The technique described in this paper raises a number of interesting questions that we intend to investigate. First, what is the actual status of the consequence relations $\vdash_{\mathbb{L}_{SL_7}}$ and $\vdash_{\mathbb{L}_{mm}}$ in the span of labelled consequence relations defined in [9]? Can this knowledge be used to better characterize the logic they capture? Is it possible to characterize the time-limited reasoning in such manner that the worst-case reasoning time (analogously to the worst-case execution time, known from the area of real-time systems) could be effectively computed? What would be then the semantical characterization of such a logic?

Speaking slightly more generally, we hope that LDS may serve as a tool for specifying logics that artificial intelligence is looking for: formalisms describing the knowledge in flux (to quote the famous title of Peter Gärdenfors) that serve intelligent agents to reason about the world they are embedded in and about other agents, without resorting to artificial, extra-logical mechanisms.

8 Acknowledgment

The second author would like to thank Michael Fisher for pointing out the LDS mechanism as a potential tool for implementing time-limited reasoning.

References

[1] M. Asker. Logical reasoning with temporal constraints. Master's thesis, Department of Computer Science, Lund University, August 2003.

[2] M. Cadoli and F. Donini. A survey on knowledge compilation. *AI Communications*, 2001.

[3] M. Cadoli and M. Schaerf. Approximate reasoning and non-omniscient agents. In *Proc. TARK 92*, pages 169–183, 1992.

[4] J. Drapkin, M. Miller, and D. Perlis. A memory model for real-time commonsense reasoning. Technical Report TR-86-21, Department of Computer Science, University of Maryland, 1986.

[5] H.-D. Ebbinghaus. Is there a logic for polynomial time? *L. J. of the IGPL*, 7(3):359–374, 1999.

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

[7] J. Elgot-Drapkin. Step-logic and the three-wise-men problem. In *Proc. AAAI*, pages 412–417, 1991.

[8] J. Elgot-Drapkin, S. Kraus, M. Miller, M. Nirkhe, and D. Perlis. Active logics: A unified formal approach to episodic reasoning. Technical report, Department of Computer Science, University of Maryland, 1996.

[9] D. Gabbay. *Labelled Deductive Systems, Vol. 1*. Oxford University Press, 1996.

[10] D. Gabbay and J. Woods. The new logic. *L. J. of the IGPL*, 9(2):141–174, 2001.

[11] G. De Giacomo, L. Iochhi, D. Nardi, and R. Rosati. A theory and implementation of cognitive mobile robots. *J. Logic Computation*, 9(5):759–785, 1999.

[12] A. Globerman. A modal active logic with focus of attention for reasoning in time. Master's thesis, Department of Mathematics and Computer Science, Bar-Illan University, 1997.

[13] G. Gogic, C. Papadimitriou, and M. Sideri. Incremental recompilation of knowledge. *JAIR*, 8:23–37, 1998.

[14] H. Levesque. A logic of implicit and explicit belief. In *Proc. AAAI 84*, pages 198–202, 1984.

[15] M. Nirkhe. *Time-Situated Reasoning Within Tight Deadlines and Realistic Space and Computation Bounds*. PhD thesis, Department of Computer Science, University of Maryland, 1994.

[16] M. Nirkhe, S. Kraus, and D. Perlis. Situated reasoning within tight deadlines and realistic space and computation bounds. In *Proc. Common Sense 93*, 1993.

[17] P. F. Patel-Schneider. A decidable first-order logic for knowledge representation. In *Proc. IJCAI 85*, pages 455–458, 1985.

[18] P. F. Patel-Schneider. A four-valued semantics for frame-based description languages. In *Proc. AAAI 86*, pages 344–348, 1986.

[19] K. Purang, D. Purushothaman, D. Traum, C. Andersen, D. Traum, and D. Perlis. Practical reasoning and plan execution with active logic. In *Proceedings of the IJCAI'99 Workshop on Practical Reasoning and Rationality*, 1999.

[20] B. Selman and H. Kautz. Knowledge compilation and theory approximation. *JACM*, 43(2):193–224, 1996.

[21] M. Wooldridge and A. Lomuscio. A computationally grounded logic of visibility, perception, and knowledge. *L. J. of the IGPL*, 9(2):257–272, 2001.