1992-10

Deepak

WORKING NOTES

AAAI
SPRING SYMPOSIUM SERIES

Symposium:
Propositional Knowledge Representation

Program Committee:
Stuart C. Shapiro, SUNY at Buffalo, Chair
John Barnden, New Mexico State University
Deepak Kumar, SUNY at Buffalo
Joao P. Martins, Instituto Superior Tecnico, Portugal
John F. Sowa, IBM

MARCH 25, 26, 27, 1992

STANFORD UNIVERSITY

Deductive Efficiency + Belief Revision: How they affect an ontology of actions and acting

Deepak Kumar and Stuart C. Shapiro
Department of Computer Science
226 Bell Hall
State University of New York at Buffalo
Buffalo, NY 14260
kumard, shapiro@cs.buffalo.edu

Abstract

The SNePS inference engine is optimized for deductive efficiency, i.e., all beliefs acquired via inference are added to the agent's beliefs so that future queries may be answered by a retrieval rather than rederivation. An assumption-based truth maintenance system keeps track of the derivation histories of derived beliefs. We show how such an architecture simplifies the ontology of propositional representations of plans; acts; preconditions, and effects of actions. In addition, the deductive efficiency of the basic system automatically extends itself to efficient search of plans, and hierarchical plan decompositions.

1 Introduction

SNIP, the SNePS inference package [Hull 1986, Pinto-Ferreira *et al.* 1989] is optimized for deductive efficiency, i.e., all beliefs acquired via inference are added to the agent's beliefs so that future queries may be answered by a retrieval rather than rederivation. Of course, it is imperative, then, to have a built-in truth maintenance system (TMS) so as to detect inconsistencies that may arise because of new information and to guarantee a consistent belief space of the modeled agent. Traditionally truth maintenance (or belief revision) systems conjure up images of detection of contradicting beliefs and their subsequent revision; or, in some more adventurous cases, reasoning about the future, as in a planning situation; or, in a planning domain, detecting inconsistencies in an already formulated plan; or, more typically, reasoning about the beliefs of other agents. SNeBR, the SNePS system for belief revision [Martins and Shapiro 1988] has been used for some of these tasks. It forms an integral part of SNePS 2.1 [Shapiro and Group 1989, Shapiro and Martins 1989], i.e. anyone working with SNePS has at their disposal at least the facility of an ATMS (i.e. contradiction detection and subsequent belief revision). It turns out that the guarantee of the presence of such an integrated ATMS can be exploited to simplify certain propositional representations for planning and acting. Additionally, together with the deductive efficiency of the inference engine, one gets viable alternatives to overcome the STRIPS assumption while modeling agents that act. This paper presents evidence of some of these not so obvious results that we claim denote a partial "integration heaven" (term from Paul Rosenbloom). First we present an example of the SNePS inference engine and the TMS operations involved. Then we discuss our decisions about deductive efficiency. Then we present propositional representations for planning and acting affected by the presence of deductive efficiency and belief revision.

2 The SNePS Inference Engine

To illustrate some of the features of the SNePS inference engine, assume that the agent has the following beliefs:

A is a block.
All blocks are supports.

which are represented as SNePS propositions `M22` and `M1` (see Figure 1). In this paper we are using a linear predicate-logic notation to illustrate the examples. In reality, the propositions are represented as semantic network nodes. The notation used in the paper may appear as a higher-order logic. However, remember that all entities (individuals, propositions, rules, acts) are represented using SNePS nodes and all nodes form terms in SWM, the underlying logic of SNePS. Each formula in the example denotes a belief of the agent and is numbered. The numbers used for formulas are names of the corresponding SNePS nodes. The exclamation mark (!) after a node name indicates that the agent currently believes the proposition represented by the node. `M1` and `M22` are called *supported wffs* (swffs) which form the basic objects of SWM, the logic underlying the inference and belief revision system. Associated with each swff is a *support* containing an *origin tag*— which is hyp for hypotheses, and der for derived swffs; an *origin set*— which contains those (and only those) hypotheses used in the derivation of the swff; and a *restriction set*—which records inconsistency information. All beliefs of the agent reside in a *belief space* which is a set of all the hypotheses and all the swffs derived from them. Thus, the propositions shown in Figure 1 are hypotheses and form the agent's current belief space. When the agent is asked the query

What is a support?

using backward chaining through the rule represented by `M1` the agent is able to deduce

A is a support.

which is represented by the proposition `M53` (see Figure 2). Note that the agent has now added the newly derived belief to its current belief space along with the origin tag of der and an origin set containing the hypotheses `M1` and `M22` indicating that these were used in its derivation.

3 Deductive Efficiency and Belief Revision

In the example above, the agent will continue to believe M53 as long as the beliefs M1 and M22 are held. If the earlier query is repeated the answer is retrieved (i.e. A is a support) by simply looking at the belief status of M53. This is what we mean by deductive efficiency, i.e. the inference engine does not repeat the inference process used in deriving M53 since it had already derived it earlier and its origin set still holds.

A derived proposition is automatically removed from the agent's belief space if any one of the hypotheses contained in its origin set is removed. As beliefs of the agent change because of actions this provides a built-in mechanism for revising a belief space after an action is performed. All that needs to be done by the agent after executing an action is to perform the acts of believing the action's consequences which

No.: Formula	Support
M22! : Isa(A, BLOCK)	< hyp, {M22!}, {} >
M1! : ∀x[Isa(x, BLOCK) → Isa(x, SUPPORT)]	< hyp, {M1!}, {} >

Figure 1: M22: A is a block. M1: All blocks are supports

No.: Formula	Support
M22! : Isa(A, BLOCK)	< hyp, {M22!}, {} >
M1! : ∀x[Isa(x, BLOCK) → Isa(x, SUPPORT)]	< hyp, {M1!}, {} >
M53! : Isa(A, SUPPORT)	< der, {M1!, M22!}, {} >

Figure 2: M22: A is a block. M1: All blocks are supports. M53: A is a support.

involves adding or deleting of hypotheses directly related to the act performed. SNePS provides two operations— add-to-context and remove-from-context to add or remove hypotheses from the agent's belief space.

Another important efficiency criterion incorporated in the design of the SNePS inference engine is automatic re-inclusion of derived beliefs into the current belief space if all of the hypotheses in their origin set come to be included. For example, if for some reason M1 (or M22) was removed from the agent's belief space (using remove-from-context), M53 would also be removed. At a later time if M1 (or M22) is again added to the agent's belief space, M53 is automatically replaced. Thus maintaining deductive efficiency.

The combination of above three features (deductive efficiency, automatic revision of the belief space, and automatic re-inclusion of derived beliefs) in an ATMS-based inference engine has a significant influence on the design of propositional representations for planning and acting and the mechanism of acting itself. We will discuss these next.

4 Propositional Representations for actions

We will present an overview of our representations of actions through an example. See [Kumar and Shapiro 1991a, Kumar and Shapiro 1991b] for more details. Consider the blocksworld action of picking up a block. We inform the agent about the action by first saying

All blocks are supports.
Picking up is a primitive action.

which results in the propositions represented by M1 and M2 (see Figure 3). Preconditions of acts are also represented as propositions. Thus the input

Before picking up a block the block must be clear.

is interpreted as a generic rule specifying a precondition for picking up a block. The rule is represented by node M3 in Figure 3. It could be paraphrased as, "For all x, if x is a block then the act of picking up x has the precondition that x is clear." Effects are similarly represented. Thus the following

After picking up a block the block is not clear and the block is held.

is represented by two rules— one specifying the effect that the block is no longer clear (M6); and the other specifying that the block is held (M8). Figure 4 shows the agent's belief space after the effects of the act of picking up a block are also added.

5 Acting and Belief Revision

Beliefs of the agent change frequently during acting. Every time an action is performed the world changes. Accordingly, the agent's beliefs about the world should also change. Typically, effects of an action (represented as STRIPS-style operators [Fikes and Nilsson 1971]) are specified as add-delete lists so that after the action is performed the belief space is updated by using the operator's add-delete list. A STRIPS assumption underlies such implementations. Traditional schemes for using an ATMS for an acting system recommend that effects of performing actions should only be added and a consistency maintenance function applied to detect inconsistencies in the belief space, and select an appropriate set of beliefs to be removed so as to make the belief space consistent [Drummond 1987]. While belief revision systems are built to detect inconsistencies, automated selection of beliefs to be removed is not a viable option. Typically an ATMS, as does SNeBR, enters a dialog with the user so that the user can select the beliefs to be removed upon detection of some inconsistency.

In the SNePS acting system we define two *mental* actions— BELIEVE and DISBELIEVE, that are used to update the beliefs of the agent after an action is performed. The effectory components of the two actions are the operations add-to-context and remove-from-context respectively. The TMS facilitates automatic revision of the belief space after a hypothesis is removed as a result of some DISBELIEVE action (all derived beliefs having the disbelieved hypothesis in their origin set are also removed). This implements the *extended* STRIPS assumption [Georgeff 1987]. For example, if the agent's belief space is that of Figure 4 and it is told

A is a block. A is clear.

which get represented by nodes M22 and M23 respectively in Figure 5, and asked to perform the act

Pick up A.

the acting system infers the propositions

A precondition of picking up A is that A is clear.
An effect of picking up A is that A is no longer clear.
An effect of picking up A is that A is held.

No.: Formula	Support
M1! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{Isa}(x, \mathrm{SUPPORT})]$	$<$ hyp, {M1!}, {} $>$
M2! : Isa(PICKUP, PRIMITIVE)	$<$ hyp, {M2!}, {} $>$
M3! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActPrecondition}(\mathrm{PICKUP}(x), \mathrm{Clear}(x))]$	$<$ hyp, {M3!}, {} $>$

Figure 3: M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear.

No.: Formula	Support
M1! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{Isa}(x, \mathrm{SUPPORT})]$	$<$ hyp, {M1!}, {} $>$
M2! : Isa(PICKUP, PRIMITIVE)	$<$ hyp, {M2!}, {} $>$
M3! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActPrecondition}(\mathrm{PICKUP}(x), \mathrm{Clear}(x))]$	$<$ hyp, {M3!}, {} $>$
M6! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActEffect}(\mathrm{PICKUP}(x), \neg\mathrm{Clear}(x))]$	$<$ hyp, {M6!}, {} $>$
M8! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActEffect}(\mathrm{PICKUP}(x), \mathrm{Held}(x))]$	$<$ hyp, {M8!}, {} $>$

Figure 4: The SNePS representation of the act of picking up a block. (M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear. M6: After picking up a block the block is not clear. M8: After picking up a block the block the block is held.)

No.: Formula	Support
M1! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{Isa}(x, \mathrm{SUPPORT})]$	$<$ hyp, {M1!}, {} $>$
M2! : Isa(PICKUP, PRIMITIVE)	$<$ hyp, {M2!}, {} $>$
M3! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActPrecondition}(\mathrm{PICKUP}(x), \mathrm{Clear}(x))]$	$<$ hyp, {M3!}, {} $>$
M6! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActEffect}(\mathrm{PICKUP}(x), \neg\mathrm{Clear}(x))]$	$<$ hyp, {M6!}, {} $>$
M8! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActEffect}(\mathrm{PICKUP}(x), \mathrm{Held}(x))]$	$<$ hyp, {M8!}, {} $>$
M22! : Isa(A, BLOCK)	$<$ hyp, {M22!}, {} $>$
M23! : Clear(A)	$<$ hyp, {M23!}, {} $>$
M26! : ActPrecondition(PICKUP(A), Clear(A))	$<$ der, {M22!, M3!}, {} $>$
M29! : ActEffect(PICKUP(A), Held(A))	$<$ der, {M22!, M8!}, {} $>$
M30! : ActEffect(PICKUP(A), $\neg$Clear(A))	$<$ der, {M22!, M6!}, {} $>$

Figure 5: The agent's belief space after the preconditions and effects of PICKUP(A) have beed deduced (M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear. M6: After picking up a block the block is not clear. M8: After picking up a block the block the block is held. M22: A is a block. M23: A is clear. M26: A precondition of picking up A is that A is clear. M29: An effect of picking up A is that A is held. M30: An effect of picking up A is that A is no longer clear.)

No.: Formula	Support
M1! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{Isa}(x, \mathrm{SUPPORT})]$	$<$ hyp, {M1!}, {} $>$
M2! : Isa(PICKUP, PRIMITIVE)	$<$ hyp, {M2!}, {} $>$
M3! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActPrecondition}(\mathrm{PICKUP}(x), \mathrm{Clear}(x))]$	$<$ hyp, {M3!}, {} $>$
M6! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActEffect}(\mathrm{PICKUP}(x), \neg\mathrm{Clear}(x))]$	$<$ hyp, {M6!}, {} $>$
M8! : $\forall x[\mathrm{Isa}(x, \mathrm{BLOCK}) \rightarrow \mathrm{ActEffect}(\mathrm{PICKUP}(x), \mathrm{Held}(x))]$	$<$ hyp, {M8!}, {} $>$
M22! : Isa(A, BLOCK)	$<$ hyp, {M22!}, {} $>$
M26! : ActPrecondition(PICKUP(A), Clear(A))	$<$ der, {M22!, M3!}, {} $>$
M29! : ActEffect(PICKUP(A), Held(A))	$<$ der, {M22!, M8!}, {} $>$
M30! : ActEffect(PICKUP(A), $\neg$Clear(A))	$<$ der, {M22!, M6!}, {} $>$
M28! : $\neg$Clear(A)	$<$ hyp, {M28!}, {} $>$
M27! : Held(A)	$<$ hyp, {M27!}, {} $>$

Figure 6: Belief space of the agent after the act PICKUP(A) is performed (M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear. M6: After picking up a block the block is not clear. M8: After picking up a block the block the block is held. M22: A is a block. M26: A precondition of picking up A is that A is clear. M29: An effect of picking up A is that A is held. M30: An effect of picking up A is that A is no longer clear. M28: A is not clear. M27: A is held.)

which are represented by nodes M26, M30 and M29 respectively (Figure 5). Since the precondition is satisfied (i.e. the agent believes M23) the action will be executed and thereafter the agent will perform the acts

BELIEVE(Held(A))
DISBELIEVE(Clear(A))

of believing the effects (indicated by beliefs M30, M29). The effector component of the mental act of believing (BELIEVE(p)) is implemented using remove-from-context on $\neg$p and add-to-context on p and DISBELIEVE(p) is remove-from-context on p and add-to-context on $\neg$p. Thus after the act is performed and its effects believed, we will have the revised belief space shown in Figure 6.

This example illustrates one of the advantages of deductive efficiency employed by the inference engine— once the agent has derived the preconditions and effects of performing an action on a specific object they become derived beliefs in the agent's belief space, hence future retrieval of preconditions/effects of the same act will not require rederivation. (I.e. as long as the assumptions underlying the propositions M26, M29 and M30 hold, they will be believed. The assumptions being M3 and M22 for M26, M8, M22 for M29, and M6, M22 for M30. As soon as the agent comes to disbelieve any one of the underlying assumptions, the corresponding derived beliefs will be removed from the agent's belief space. Thus, if an action has any context-sensitive effects, we can include the condition qualifying the context in the antecedent part of the rule specifying the effects. This is presented next.

5.1 Context-sensitive effects

In a world where blocks are considered supports, the following additional effects need to be specified for picking up a block:

If a block is on a support then after picking up the block the block is not on the support and the support is clear.

The belief space after the above two propositions and

B is a block.
A is on B.

are added to the belief space of Figure 4 is shown in Figure 7. Next, if the agent is now requested to

Pickup A.

the agent will deduce two additional preconditions

An effect of picking up A is that A is no longer on B.
An effect of picking up A is that B is clear.

which are represented by derived propositions M35 and M36 (Figure 8 shows the new belief space).

However, these propositions hold only in the case where A is on B. Notice that the origin sets of M35 and M36 contain the hypotheses M1, M21 (which were used to derive M34), M22, and M24. After the act is performed the mental actions

BELIEVE(Held(A))
DISBELIEVE(Clear(A))
BELIEVE(Clear(B))
DISBELIEVE(On(A,B))

will be performed. The last mental action removes M24 from the agent's belief space. Since M35 and M36 contain M24 in their respective origin sets they are also removed. The revised belief space after this is shown in Figure 9.

Two things are illustrated by this example— that the representation of context-sensitive effects is trivial; and automatic removal of derived propositions can be achieved by the belief revision system. The former, in traditional STRIPS style representations of the blocksworld requires the use of an extra operator, unstack [Nilsson 1980], which is unnatural. Thus, not only do we eliminate the need for operators as separate representations for actions, we also end up with fewer, simpler, and at the same time more versatile representations for actions. The latter (automatic belief revision by removal of derived propositions) implements the *extended* STRIPS assumption.

If, in a future situation, A is on B, the context-dependent effect removed earlier will automatically be re-included. This way of specifying context-dependent effects seems to be better than that used by SIPE [Wilkins 1988] for several reasons. For one, we have eliminated the need for a separate specification of actions as *operators*. In SIPE, context-dependent effects are represented by *domain rules*. While domain rules of SIPE help to make the operators more applicable the applicability of the domain rules themselves (as a representation of a causal theory) is not uniform in SIPE. In our representations the so called traditional operator is constructed dynamically at the time of acting, i.e. each time an act is performed, its preconditions and effects are deduced. Coupling deductive efficiency with belief-revision provides a more natural, yet efficient, way of representing actions, and at the same time, a more uniform notion of a causal theory.

6 Conditional Plans

Like acts, we treat plans as mental entities that the agent can have beliefs about. So, plans, once derived will be believed by the agent as long as their underlying assumptions are believed. The agent can also be provided generic pre-packaged abstract plans that form the agent's plan library. Before indulging in a plan generation phase, the agent, when asked to do something, can retrieve specific plans from the plans it already has beliefs about. We have propositional representations for plans that represent decompositions of complex actions as well as those that specify ways of achieving goals. The representations of plans are defined in terms of primitive control actions which, in our repertoire, include sequencing, conditional, and iterative acts (among others).

Retrieval of plans, similarly, benefits from the deductive efficiency of the inference engine. Additionally, conditional plans like the simple one below

If a block is on a support then a plan to achieve that the support is clear is to pick up the block and then put the block on the table.

can be represented, as in the case of context-sensitive effects, by specifying the qualifying propositions as antecedents of the rule specifying the plan. i.e.

$$\forall x, y[\mathrm{Isa}(x, \mathrm{BLOCK}) \wedge \mathrm{Isa}(y, \mathrm{SUPPORT}) \wedge \mathrm{On}(x, y)$$

$$\rightarrow \mathrm{GoalPlan}(\mathrm{Clear}(y), \mathrm{SEQUENCE}(\mathrm{PICKUP}(x), \mathrm{PUT}(x, \mathrm{TABLE})))]$$

Once again, in a situation where A is on B, to clear B the agent will use the above rule to derive

A plan to clear B is to first pickup A and then put A on the table.

which is a derived proposition having the qualifying situation On(A, B) as one of its assumptions. Once the plan is executed it will no longer be believed. It will, as usual, be

No.: Formula	Support
M1! : ∀x[Isa(x, BLOCK) → Isa(x, SUPPORT)]	< hyp, {M1!}, {} >
M2! : Isa(PICKUP, PRIMITIVE)	< hyp, {M2!}, {} >
M3! : ∀x[Isa(x, BLOCK) → ActPrecondition(PICKUP(x), Clear(x))]	< hyp, {M3!}, {} >
M6! : ∀x[Isa(x, BLOCK) → ActEffect(PICKUP(x), ¬Clear(x))]	< hyp, {M6!}, {} >
M8! : ∀x[Isa(x, BLOCK) → ActEffect(PICKUP(x), Held(x))]	< hyp, {M8!}, {} >
M22! : Isa(A, BLOCK)	< hyp, {M22!}, {} >
M23! : Clear(A)	< hyp, {M23!}, {} >
M9! : ∀x, y[Isa(x, BLOCK) ∧ Isa(y, SUPPORT) ∧ On(x, y) → ActEffect(PICKUP(A), ¬On(x, y))]	< hyp, {M9!}, {} >
M10! : ∀x, y[Isa(x, BLOCK) ∧ Isa(y, SUPPORT) ∧ On(x, y) → ActEffect(PICKUP(A), Clear(y))]	< hyp, {M10!}, {} >
M21! : Isa(B, BLOCK)	< hyp, {M21!}, {} >
M24! : On(A, B)	< hyp, {M24!}, {} >

Figure 7: The SNePS representation of the act of picking up a block including context-sensitive effects. (M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear. M6: After picking up a block the block is not clear. M8: After picking up a block the block the block is held. M22: A is a block. M23: A is clear. M9: If a block is on a support then an effect of picking up the block is that the block is not on the support. If a block is on a support then an effect of picking up the block is that the support is clear. M21: B is a block. M24: A is on B.)

No.: Formula	Support
M1! : ∀x[Isa(x, BLOCK) → Isa(x, SUPPORT)]	< hyp, {M1!}, {} >
M2! : Isa(PICKUP, PRIMITIVE)	< hyp, {M2!}, {} >
M3! : ∀x[Isa(x, BLOCK) → ActPrecondition(PICKUP(x), Clear(x))]	< hyp, {M3!}, {} >
M6! : ∀x[Isa(x, BLOCK) → ActEffect(PICKUP(x), ¬Clear(x))]	< hyp, {M6!}, {} >
M8! : ∀x[Isa(x, BLOCK) → ActEffect(PICKUP(x), Held(x))]	< hyp, {M8!}, {} >
M22! : Isa(A, BLOCK)	< hyp, {M22!}, {} >
M23! : Clear(A)	< hyp, {M23!}, {} >
M26! : ActPrecondition(PICKUP(A), Clear(A))	< der, {M22!, M3!}, {} >
M29! : ActEffect(PICKUP(A), Held(A))	< der, {M22!, M8!}, {} >
M30! : ActEffect(PICKUP(A), ¬Clear(A))	< der, {M22!, M6!}, {} >
M9! : ∀x, y[Isa(x, BLOCK) ∧ Isa(y, SUPPORT) ∧ On(x, y) → ActEffect(PICKUP(~~A~~), ¬On(x, y))]	< hyp, {M9!}, {} >
M10! : ∀x, y[Isa(x, BLOCK) ∧ Isa(y, SUPPORT) ∧ On(x, y) → ActEffect(PICKUP(~~A~~), Clear(y))]	< hyp, {M10!}, {} >
M21! : Isa(B, BLOCK)	< hyp, {M21!}, {} >
M24! : On(A, B)	< hyp, {M24!}, {} >
M34! : Isa(B, SUPPORT)	< der, {M21!, M1!}, {} >
M35! : ActEffect(PICKUP(A), ¬On(A, B))	< der, {M1!, M21!, M22!, M24!}, {} >
M36! : ActEffect(PICKUP(A), Clear(B))	< der, {M1!, M21!, M22!, M24!}, {} >

Figure 8: Belief space after the preconditions and effects of PICKUP(A) are deduced (M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear. M6: After picking up a block the block is not clear. M8: After picking up a block the block the block is held. M22: A is a block. M23: A is clear. M26: A precondition of picking up A is that A is clear. M29: An effect of picking up A is that A is held. M30: An effect of picking up A is that A is no longer clear. M28: A is not clear. M9: If a block is a support then an effect of picking up a block is that the block is no longer on the support. M10: If a block is on a support then an effect of picking up a block is that the support is clear. M21: B is a block. M24: A is on B. M34: B is a support. M35: An effect of picking up A is that A is not on B. M36: An effect of picking up A is that B is clear.)

No.: Formula	Support
M1! : $\forall x[\text{Isa}(x, \text{BLOCK}) \rightarrow \text{Isa}(x, \text{SUPPORT})]$	< hyp, {M1!}, {} >
M2! : Isa(PICKUP, PRIMITIVE)	< hyp, {M2!}, {} >
M3! : $\forall x[\text{Isa}(x, \text{BLOCK}) \rightarrow \text{ActPrecondition}(\text{PICKUP}(x), \text{Clear}(x))]$	< hyp, {M3!}, {} >
M6! : $\forall x[\text{Isa}(x, \text{BLOCK}) \rightarrow \text{ActEffect}(\text{PICKUP}(x), \neg\text{Clear}(x))]$	< hyp, {M6!}, {} >
M8! : $\forall x[\text{Isa}(x, \text{BLOCK}) \rightarrow \text{ActEffect}(\text{PICKUP}(x), \text{Held}(x))]$	< hyp, {M8!}, {} >
M22! : Isa(A, BLOCK)	< hyp, {M22!}, {} >
M26! : ActPrecondition(PICKUP(A), Clear(A))	< der, {M22!, M3!}, {} >
M29! : ActEffect(PICKUP(A), Held(A))	< der, {M22!, M8!}, {} >
M30! : ActEffect(PICKUP(A), $\neg$Clear(A))	< der, {M22!, M6!}, {} >
M9! : $\forall x, y[\text{Isa}(x, \text{BLOCK}) \wedge \text{Isa}(y, \text{SUPPORT}) \wedge \text{On}(x, y) \rightarrow \text{ActEffect}(\text{PICKUP}(A), \neg\text{On}(x, y))]$	< hyp, {M9!}, {} >
M10! : $\forall x, y[\text{Isa}(x, \text{BLOCK}) \wedge \text{Isa}(y, \text{SUPPORT}) \wedge \text{On}(x, y) \rightarrow \text{ActEffect}(\text{PICKUP}(A), \text{Clear}(y))]$	< hyp, {M10!}, {} >
M21! : Isa(B, BLOCK)	< hyp, {M21!}, {} >
M34! : Isa(B, SUPPORT)	< der, {M21!, M1!}, {} >
M37! : $\neg$On(A, B)	< hyp, {M37!}, {} >
M38! : Clear(B)	< hyp, {M38!}, {} >
M28! : $\neg$Clear(A)	< hyp, {M28!}, {} >
M27! : Held(A)	< hyp, {M27!}, {} >

Figure 9: Belief space after the act PICKUP(A) is performed (M1: All blocks are supports. M2: Picking up is a primitive action. M3: Before picking up a block the block must be clear. M6: After picking up a block the block is not clear. M8: After picking up a block the block the block is held. M22: A is a block. M26: A precondition of picking up A is that A is clear. M29: An effect of picking up A is that A is held. M30: An effect of picking up A is that A is no longer clear. M9: If a block is a support then an effect of picking up a block is that the block is no longer on the support. M10: If a block is on a support then an effect of picking up a block is that the support is clear. M21: B is a block. M34: B is a support. M37: A is not on B. M38: B is clear. M28: A is not clear. M27: A is held.)

re-included in the belief space if a similar situation (i.e. A is on B) is attained. While preconditions of plans can be dealt by specifying them in the antecedents of rules, plans that include conditional actions still use conditional control actions (See [Kumar and Shapiro 1991b]) as part of the plan.

7 Remarks

A basic premise of our approach stems from the empirical observation that, typically, good knowledge representation and reasoning systems are bad candidates for planning/acting modeling and vice versa. If one wishes to extend a good KR system for planning/acting modeling one can take the easy way out by simply integrating a mutually exclusive off-the-shelf planning/acting system. This only results in paradigm soups. The approach we have taken is to extend the KR system by extending its ontology and at the same time preserving its foundations. The resulting architecture is simple, more uniform, and offers viable solutions to some of the standard problems. In this paper we have tried to demonstrate that in a deductive approach to modeling rational agents, where a uniform representational formalism is used, certain unusual and appealing benefits can be gained by integrating deductive efficiency with an assumption based truth maintenance system. The resulting agent architecture is simple, uniform, has simpler yet versatile representations, providing deductive efficiency as well as alternate approaches to dealing with the STRIPS assumption.

References

[Drummond 1987] Mark E. Drummond. A representation of action and belief for automatic planning systems. In Michael P. Georgeff and Amy L. Lansky, editors, *Reasoning about Actions and Plans - Proceedings of the 1986 Workshop*, pages 189–212, Los Altos, CA, 1987. AAAI and CSLI, Morgan Kauffmann.

[Fikes and Nilsson 1971] Richard E. Fikes and Nils J. Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 5:189–208, 1971.

[Georgeff 1987] Michael P. Georgeff. Planning. In *Annual Reviews of Computer Science Volume 2*, pages 359–400. Annual Reviews Inc., Palo Alto, CA, 1987.

[Hull 1986] R. G. Hull. A new design for SNIP the SNePS inference package. SNeRG Technical Note 14, Department of Computer Science, SUNY at Buffalo, 1986.

[Kumar and Shapiro 1991a] Deepak Kumar and Stuart C. Shapiro. Architecture of an intelligent agent in SNePS. *SIGART Bulletin*, 2(4):89–92, August 1991.

[Kumar and Shapiro 1991b] Deepak Kumar and Stuart C. Shapiro. Modeling a rational cognitive agent in SNePS. In B. Barahona, L. Moniz Pereira, and A. Porto, editors, *EPIA 91: 5th Portugese Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence 541*, pages 120–134. Springer-Verlag, Heidelberg, 1991.

[Martins and Shapiro 1988] J. P. Martins and S. C. Shapiro. A model for belief revision. *Artificial Intelligence*, 35(1):25–79, 1988.

[Nilsson 1980] Nils J. Nilsson. *Principles Of Artificial Intelligence*. Tioga Publishing Company, Palo Alto, CA, 1980.

[Pinto-Ferreira *et al.* 1989] Carlos Pinto-Ferreira, Nuno J. Mamede, and Jo ao P. Martins. SNIP 2.1— The SNePS Inference Package. Technical Report GIA 89/5, Technical University of Lisbon, December 1989.

[Shapiro and Group 1989] S. C. Shapiro and The SNePS Implementation Group. *SNePS-2 User's Manual.* Department of Computer Science, SUNY at Buffalo, 1989.

[Shapiro and Martins 1989] S. C. Shapiro and J. P. Martins. Recent advances and developments – the SNePS 2.1 report. In D. Kumar, editor, *Current Trends in SNePS–Semantic Network Processing System: Proceedings of the First Annual SNePS Workshop*, pages 1–13, Buffalo, NY, 1989. Springer-Verlag.

[Wilkins 1988] David E. Wilkins. *Practical Planning–Extending the Classical AI Planning Paradigm.* Morgan Kaufmann, Palo Alto, CA, 1988.