Situated Reasoning Within Tight Deadlines and Realistic Space and Computation Bounds*

Madhura Nirkhe[1], Sarit Kraus[2], Donald Perlis[1]

[1]Department of Computer Science

University of Maryland, College Park, MD 20742

[2]Department of Mathematics and Computer Science

Bar Ilan University Ramat Gan, 52900 Israel

Abstract

Challenging among numerous resource limited realistic scenarios in commonsense reasoning are situations involving tight deadlines. An agent under severe time-pressure may spend substantial amount of the available time in reasoning toward and about a plan of action. In a realistic setting, the same agent must also measure up to two other crucial resource limitations as well, namely space and computation bounds. We describe here these concerns and offer some solutions as part of our ongoing work in fully deadline-coupled planning.

1 Introduction

Formal commonsense reasoning deals with numerous difficult tasks. One broadening area is that of realism, i.e., trying to get formalisms to face up to more and more realistic scenarios and data. Realism can involve large quantities of data, rapid presentation of data, noisy data, and missing data, to name a few possibilities which have been given at least some study in the literature already. Another realistic scenario is that of severe time-pressure in the form of tight deadlines, where any effort of the agent is judged above all on the basis of its ability to meet the deadline.

In this paper we describe our recent and ongoing work in deadline-coupled reasoning. Such reasoning is of necessity real-time, in the sense that the agent must take account of the actual passage of time in the external world as it reasons. The paradigmatic and illustrative example on which we have focused is the *Nell & Dudley* problem, in which Dudley must find and enact a plan to save Nell who is tied to the railroad tracks as a train approaches; thus Dudley is an actor-planner. Clearly Dudley must take account of the fact that his very reasoning is costing precious time; thus this topic is one of resource-limited reasoning.

The underlying mechanism we have chosen for this work is that of Elgot-Drapkin's step-logics, which are briefly described in section 2. We then describe some modifications that are needed for a more fully resource-limited treatment of our problem, including concerns of space-limitations (memory limitations) that must constrain the agent's reasoning abilities. This is the topic of section 3.

In section 4 we sketch our design for a logic-based time-situated planning mechanism in terms of step-logics, which we have implemented in several phases, with emphasis on the limited resource reasoning aspects.

2 Step-Logics: A mechanism for reasoning situated in time

Step-logics [?, ?, ?] were introduced to model a commonsense agent's ongoing process of reasoning in a changing world. A *step* is defined as a fundamental unit of inference time. Beliefs are parameterized by the time taken for their inference, and these time parameters can themselves play a role in the specification of the inference rules and axioms. The most obvious way time parameters can enter is via the expression $Now(i)$, indicat-

*This material is based upon work supported in part by the National Science Foundation under Grant No. IRI-9123460. Nirkhe and Kraus are also affiliated with the Institute for Advanced Computer Studies, University of Maryland, College Park. e-mail:{madhura,perlis}@cs.umd.edu, sarit@umiacs.umd.edu

ing the time is now i. Observations are inputs from the external world, and may arise at any step i. When an observation appears, it is considered a belief in the same time-step. Each step of reasoning advances i by 1. At each new step i, the only information available to the agent upon which to base his further reasoning is a snap-shot of his deduction process completed up to and including step $i-1$. The agent's world knowledge is in the form of a database of beliefs. These contain domain specific axioms. A number of inference rules constitute the inference engine. Among them are rules such as Modus Ponens and rules to incorporate new observations into the knowledge base as well as rules specific to deadline-coupled planning such as checking the feasibility of a partial plan or refining a partial plan. To illustrate reasoning with step-logics, here is an instance of the application of Modus Ponens in step 10 following Dudley's observation that Nell has blue eyes. In each step we have underlined new beliefs, i.e., ones that were not present in the previous step.

The MP rule:

$$\frac{i : \ldots, \alpha, \alpha \rightarrow \beta}{i+1 : \ldots, \beta}$$

10: $\underline{Now(10)}; \underline{color\_of\_eyes(nell, blue)};$
$color\_of\_eyes(X, blue) \rightarrow loves(dudley, X);$
$loves(dudley, X) \rightarrow wants\_to\_marry(dudley, X);$
$\ldots$

11: $\underline{Now(11)}; color\_of\_eyes(nell, blue);$
$color\_of\_eyes(X, blue) \rightarrow loves(dudley, X);$
$loves(dudley, X) \rightarrow wants\_to\_marry(dudley, X);$
$\underline{loves(dudley, nell)}; \ldots$

12: $\underline{Now(12)}; color\_of\_eyes(nell, blue);$
$color\_of\_eyes(X, blue) \rightarrow loves(dudley, X);$
$loves(dudley, X) \rightarrow wants\_to\_marry(dudley, X);$
$loves(dudley, nell); \underline{wants\_to\_marry(dudley, nell)};$
$\ldots$

By step 11, Dudley can only realize that he loves Nell, he must wait until step 12 to conclude that he wants to marry her.

We have applied the step-logic mechanism to tackle the fully deadline-coupled reasoning problem in [?, ?][1]. The following features of this framework relate and contrast it to conventional commonsense reasoning systems:[2]

Thinking takes time: Reasoning actions occur concurrently with other physical actions of the agent and with the ticking of a clock. The agent can not only keep track of the approaching deadline as he enacts his plan, but can treat other facets of planning (including plan formulation and its simultaneous or subsequent execution and feasibility analysis) as deadline-coupled. Related to this feature of step-logics is the fact that there is no longer a one final theorem set. Rather, theorems (beliefs) are proven (believed) at certain times and sometimes no longer believed at later times. Provability is time-relative and best thought of in terms of the agent's ongoing lifetime of changing views of the world. This leads to the issue of contradictions below.

Handling contradictions: An agent reasoning with step-logics is not omniscient, i.e. , his conclusions are not the logical closure of his knowledge at any instant, but rather only those consequences that he has been actually able to draw. Also, since commonsense agents have a multitude of defeasible beliefs, they often encounter contradictions as more knowledge is obtained and default assumptions have to be withdrawn. While a contradiction completely throws an omniscient agent off track (the swamping problem), the step-logic reasoner is not so affected. The agent only has a finite set of conclusions from his past computation, hence contradictions may be detected and resolved in the course of further reasoning.

Nonmonotonicity: Step-logics are inherently nonmonotonic, in that further reasoning always leads to retraction of some prior beliefs. The most obvious one is $Now(i)$, which is believed at step i but not at $i+1$. The nonmonotonic behavior enables the frame-default reasoning that the commonsense agent must be capable of [?].

[1] This version has been implemented in prolog. Step-logic was also used for multi-agent coordination without communication using focal points [?].

[2] This description is necessarily very brief; for details see the various papers by Elgot-Drapkin et al.

2.1 Shortcomings

The space problem: As time advances, more knowledge is gathered as a result of observations from the agent's environment and as a result of the deduction processes within. The knowledge base which is continuously expanding could potentially become so formidable that it would be completely unrealistic to assume that the agent could possibly apply all the inferences to this complete knowledge base. Usually, most of this information is not directly relevant either to the development of the agent's current thread of reasoning. Step-logics and our treatment of deadline-coupled planning in the past have disregarded the space problem in preference to dealing adequately with time-related issues. The space issue deserves serious attention where the original number of beliefs of the agent is large, and where very many new beliefs are added to the agent's knowledge base over time.

Unrealistic parallelism: A step is defined as the time required by the agent to perform one inference or one primitive physical action in the world. Actions can be carried out in parallel if the sensors and effectors permit. For example, an agent can walk and eat simultaneously. Step-logics planners treat 'think' actions within the agent in the same spirit as physical actions and recognize that they sap precious time resources. The original step-logic inference system assumes that during a given step i the agent can apply all available inference rules in parallel, to the beliefs at step $i-1$. There are two problems with this. One is the unrealistic amount of parallelism that potentially allows the agent to draw so many inferences in one time step that the meaning of what constitutes a step begins to blur. Secondly, it is unreasonable to expect that all inference rules would have the same time granularity. For example, it is unlikely that a simple application of Modus Ponens will take just as long to fire as an inference rule to refine a plan or check for plan feasibility, especially as plans become very large. While the representation is uniformly declarative, some rules have more procedural flavor than others, and can be imagined to take more time steps. Just as there is a limit on the physical capabilities of the agent as to how many physical actions can be done in parallel in the same time step, there must be a limit to the parallel capacity of the inference engine as well.

A claim towards fully deadline-coupled reasoning would be a tall one if the model depicts an agent with an infinite attention span and infinite think capacity. In this paper we propose an extension of the original step-logic formalism to take into consideration space and computation constraints. We revisit the fully deadline-coupled planning problem in the light of this new framework.

3 Scarce resources: Limiting time, space and computation

3.1 A limited span of attention

We propose a solution to the space problem partially based on [?] as follows. The agent's current focus of attention is limited to a small fixed number of beliefs forming the STM (short term memory), while the complete belief set is archived away in a bigger associative store, namely, the LTM (long term memory). In addition, we use a QTM which is a technical device to hold the conclusions that result in each step since further inferencing with these must be stalled until the next time step. The size of the STM is a fixed number K[3].

In the most simplistic model, the STM could be represented as a queue, in which case the inference/retrieval algorithm reduces to a simple depth first or breadth first strategy depending upon whether new observations and deductions are added to the head or tail of the queue respectively. It seems that choosing the STM elements without focus consideration may lead the reasoning astray quite easily, and also lead to often incomplete threads of reasoning due to thrashing. We propose to maintain a predicate called **Focus**(...) which keeps track of the current line of reasoning. This is dynamically changed by the agent's inference mechanism and is responsible for steering the reasoning back to a particular thread even when a large number of seemingly

[3]What is a realistic K for a commonsense reasoner? There is psychological basis that suggests that human short-term memory holds seven-plus-or-minus-two 'chunks' of data at one time [?].

irrelevant inferences are drawn. Among the agent's inference rules is a set of *focus changing (FC)* rules, which when fired alter the focus. Those K beliefs from the associative LTM which are most[4] relevant to the current focus are highlighted to form the STM.

In short, the framework can be described as follows. The $QTM_{i/i+1}$ is an intermediate store of formulae that are theorems derived through the application of inference rules to the formulae in STM_i (the STM at step i). They are candidates for the STM at step $i+1$, although only K among them will be selected. Thus the results of the inference rules, can be imagined to fall into $QTM_{i/i+1}$ and are available for selection to form the STM at the next step[5]. The $focus$ and Now which are crucial to time-situated reasoning are always accesible to the agent.

FRAMEWORK:

$$\frac{i : STM_i\{...\}, Now(i), Focus(i,...), LTM_i\{...\}}{i+1 : STM_{i+1}\{...\}, Now(i+1), Focus(i+1,...), LTM_{i+1}\{...\}}$$

$QTM_{i/i+1}$ holds β if β is an i-theorem. It includes relevant formulae which are retrieved from the LTM using the retrieval rule. Step i concludes by selecting K formulae from $QTM_{i/i+1}$ which are relevant to $Focus_i$ to form STM_{i+1}. LTM_{i+1} is LTM_i appended with $QTM_{i/i+1}$.

The main problem in limiting the space of reasoning is to decide what should be in the focus. In our planning framework, we have developed a mechanism that is at work to limit the focus to a single feasible plan at a given time step. A list of actions, conditions and results from the plan that need further processing (we call it the active list), form a list of keywords in the focus. We describe some details of this mechanism in section 4. Heuristic rules are proposed to maximize the probability of finding a solution within the deadline. This would correspond to a sort of best first strategy or a beam search of width K in the general framework. Although these heuristic rules are independent of the instance of the problem in question, they are likely to be different depending upon the category of the problem being solved. A deadline-coupled actor-planner is likely to maintain a much narrower focus than a long-range 'armchair' planner. Later in the paper, we outline some of the specific heuristic strategies employed for the tightly time-constrained planner.

[4]There is then a ranking among the relevant formulae and the K at the top of the list are picked. In our implementation, we select the K formulae at random from the candidate formulae.

[5]This has the feature that all thinking does not pass through the STM unless it is relevant to the focus.

3.2 A limited think capacity

Next, we address the bounded computation resource problem. An intelligent agent can be expected to have a sizable reservoir of inference rules acquired during its lifetime. Firing of an inference rule corresponds to a 'think' action. Without a bound on its inferencing power, the agent could fire all the inference rules applicable (termed in conventional production systems as the conflict set) simultaneously during a time step. We limit the inference capacity of the engine to I. Each inference rule j is assigned a drain factor d_j. This is a measure of the drain incurred by the inference engine while firing an instance of this rule. For instance, Modus Ponens and the more elaborate inference rule for plan refinement, would be given different drain factors to reflect this difference in granularity [6].

Our limited-capacity inference engine fires only a subset of the applicable rules in each time step. Among the various alternatives, it is possible to pick the inference rules either completely nondeterministically up to the engine capacity I, or one could again apply some heuristics to improve the agent's chances. Several parameters, such as agent attitudes, the uncertainty of the environment, or the urgency to act could dictate this choice.

Thus, in effect, during each step, K beliefs are highlighted from the knowledge base (LTM) to constitute the STM. From among the rules applicable to these K beliefs, a subset of rules is chosen such that sum of the drain factors does not exceed the engine's inference ca-

[6]How to calibrate the inference rules for the assignment of these drain factors is a separate and interesting issue, but we will not address it presently. Also, how thinking actions compare with physical actions is a technical issue that could be resolved by trying to calibrate the system to check on the relative speed of its inference cycle with that of its sensors and motors. We skip this implementation sensitive issue for the present.

pacity I. The results of the inferencing are put in the QTM. Finally, the contents of the QTM are copied to the LTM.

4 Dudley, Nell and the rushing train

A fully deadline-coupled planning mechanism that uses step-logics was developed in [?, ?]. We report here on a more realistic time, space and computation constrained Dudley. Some of Dudley's beliefs are directly relevant to the synthesis and execution of his plan to save Nell, others are inconsequential, such as the color of Nell's eyes. We also show how Dudley chooses between two possible alternatives: One is to run to Nell and untie the ropes, the other is to telephone the driver of the train and request him to stop the train. The telephone is at the neighbor's house and Dudley must run there to make the call. For the purpose of this sketchy illustration, we regard most actions to be primitive.

Our research draws ideas for the skeleton of its real-time planning rules from much existing research in real-time reasoning and planning [?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?]. In our formalism, plans are beliefs consisting of triplets, each in turn consisting of an action, its corresponding conditions and results. The actions and other beliefs have time interval arguments, some of which are bound, others constrained by deadlines. A context is maintained for each plan representing the expected state of the world if the plan is carried out to completion. It consists of the set of relevant observations along with the actions, effects and extended effects of the plan. For example, in the context of the partial plan to untie Nell, she will be not tied, and in the context of the plan to call the driver of the train the train will have stopped midway on the tracks. A projection over time is made in the context of each plan to aid in planning. What is not already true in the projection is planned for by searching for axiom(s) to achieve the desired subgoal. The system has the flexibility to adapt its reasoning to changes in its knowledge base (as a result of new observations or ongoing deductions) in a nonmonotonic manner. For more details on the planning aspects and on the temporal reasoning for context maintenance and projection, we refer to [?, ?, ?]. Here we describe an outline of a limited resource reasoning aspects of a mechanism that we have developed for the deadline-coupled planning problem. We recognize that the general problem of effectively keeping reasoning directed by a focus is extremely difficult for an automated reasoner. We attempt to develop a formalism for planning that restricts the focus of the agent to a single plan in a given time step so that reasoning is restricted to computing the context, projection in this context, the working estimate of time, and feasibility analysis of the plan currently in focus.

4.1 Some simplified sample axioms

We present a small number of sample axioms here, necessarily simplified and made more specific for the example.

Relevant to moving:

- $Run(T_1 : T_2, Y, L_1 : L_2)$
  $\rightarrow At(T_2, Y, L_2)$,$T_2 = T_1 + (L_2 - L_1)/v_Y$[7]
- $condition(Run(T_1 : T_2, Y, L_1 : L_2), At(T_1, Y, L_1))$
- $result(Run(T_1 : T_2, Y, L_1 : L_2), At(T_2, Y, L_2))$

Relevant to untying and releasing:

- $Pull(T : T + 1, Y, X, L) \rightarrow Out\_of\_danger(T + 1, X, L)$
- $condition(Pull(T : T+1, Y, X, L), \neg Tied(T, X, L))$
- $result(Pull(T : T + 1, Y, X, L),$
  $Out\_of\_danger(T + 1, X, L))$

Relevant to telephones and warning:

- $Stop\_train(\text{T:T} + 2, driver) \rightarrow$
  $Out\_of\_danger(T + 2, nell, railroadtrack)$
- $condition(Stop\_train(\text{T:T}+ 2, driver),$
  $Knows\_about(T, nell, driver))$
- $Warn(S : T, X, Y, Z) \rightarrow Knows\_about(T, Z, Y)$
- $condition(Warn(\text{S:T}, X, Y, Z),$
  $In\_contact(\text{S:T}, X, Y))$
- $result(Dial(\text{S:T}, X, Y), In\_contact(T, X, Y))$
- $condition((Dial(\text{S:T}, dudley, driver),$
  $at(\text{S:T}, dudley, neighbor\_house))$

[7] v_Y is Y's speed while running.

4.2 Heuristic strategies for deadline-coupled planning

4.2.1 Focus and keywords

As a general approach to limiting space, we proposed that beliefs be organized in the LTM by association with some topics or keywords. When one or more of these topics are in the focus, the related beliefs become candidates for retrieval into the STM, as a result of a retrieval rule. Formulae in the STM are not automatically inherited from one step to the next. Only when they are still relevant to the current focus do they become candidates and must compete with other relevant formulae to fit into the limited size STM.

The focus holds the keywords of current interest. It has similarities with the RTM proposed in [?]. We imagine that in a more general framework the focus would contain keywords arranged in a partial order according to priorities.[8] Beliefs related to high priority topics are given preference for being brought into the STM. As mentioned before, for our actor-planner Dudley we restrict the focus to equal priority keywords related to a single plan at a given time step. Non-primitive actions that appear in the triplets of a given plan, that still need to be refined are appropriate keywords for goal-directed retrieval. Also, the results that appear in these triplets serve as keywords to deduce the effects of the plan. These are kept in the focus as the formula $plan\_in\_focus(p, PKWL)$ where p is the name of the partial plan and $PKWL$ is the list of keywords for p.

Observations are put into the current focus at least for a few time steps, since it is possible that they may be important, and may trigger some new threads of reasoning[9]. Current observations are kept in the focus as the formula $obs\_in\_focus(OBL)$ where OBL is the list of observations that serve as keywords. Together, the focus is a predicate $\mathbf{Focus}(i, plan\_in\_focus(p, PKWL), obs\_in\_focus(OBL))$ at step i.

When there are multiple options in the STM for achieving a goal, more than one partial plan is spawned. All plans for achieving a certain goal may be given equal priority at first, thus continuing to develop them in a time shared manner and bringing them into focus sequentially. However, in a deadline situation, it may be advisable to commit to a plan (to put it in focus and the others in a background queue for backtracking) and continue with it unless it seems infeasible.

4.2.2 How long will it take?

A WET (working estimate of time) is calculated by firing an estimation rule. This is an estimate of the total time remaining to achieve the goal. It can be seen to have three components: 1. execution time 2. planning time 3. decision time. Execution time estimate is an estimate (for each plan) of how long the actions contained in the partial plan will take to execute. The planning is hierarchical, and as it proceeds, as actions at higher abstraction levels are broken into more primitive actions, better estimates of execution time become available. This component increases initially and then may fluctuate depending upon how planning interleaves with execution. Planning time estimate (for each plan) is the time required to refine the plan to the level of primitive actions (the depth of the reasoning). Decision time estimate is the component that accounts for all the other deliberations made by the reasoner. This include checking for feasibility of plans, choosing between alternatives, committing to an alternative when deemed essential, making the difficult decision of whether to act *Now* or deliberate further etc. An extended paper will show how Dudley faces some of these challenges.

[8] The main question is how to choose the "keywords" that are in the focus at a given time, and how to assign priorities to them. Our ideas presented here are aimed at a commonsense agent engaged in deadline-coupled planning.

[9] How to in fact select some crucial observations from all the stray input to the sensors remain unaddressed, but it is not among the problems we will solve at present. A tutor or a human hint to the automated agent that some observations are worthy of more consideration. In our example, Dudley may first start to think about running to Nell to rescue her, when he suddenly sees a telephone. This brings 'calling', and subsequently the related axiom of calling the driver to stop the train into focus. This spawns the generation of a second plan.

4.2.3 Some inference rules for resource limited reasoning

At each step, the agent reflects on its long term memory reservoir to pick out formulae that are relevant to its current focus of reasoning using a retrieval rule. The LTM is an associative store and hence this retrieval is fast.[10]

Focus directed retrieval rule (FDRR):

$$\frac{i : ..., LTM\{..., \beta, ...\}, \quad \textbf{Focus}(i, plan\_in\_focus(p, PKWL), obs\_in\_focus(OBL)), ..}{QTM_{i/i+1}\{..., \beta, ...\}}$$

if β is relevant to either p or a keyword in $PKWL$ or OBL.

In our work on planning, the **Focus** includes keywords related to a *feasible* plan. A (partial) plan is *feasible* if the sum of Now and the plan's working estimate of time is still within the deadline. A list of feasible partial plans is maintained. From among these a subset of plans is selected to work on and is called the interleaving list (IL). Dudley works on each plan in the interleaving list for a *period* number of steps, then goes on to the next plan in the IL in round robin fashion. The interleaving rule (ILR) serves this purpose by periodically selecting the next plan in the IL to put into the focus. This is one of the focus changing (FC) rules in Dudley's inference engine[11]. This rule time-shares between plans and always fires. A separate rule controls the contents of IL.
Interleaving Rule (ILR):

$$\frac{i : Now(i), IL([p_{j_1}, ..., p_{j_n}]), ...}{i+1 : \textbf{Focus}(i+1, plan\_in\_focus(p_{j_1}, ...), ...), IL([p_{j_2}, ..., p_{j_n}])}$$

if i mod period = 0

When there are two or more plans in the IL, and when it is time to choose between them, a rule fires to narrow the focus to only one plan. We stipulate that the difficult problem of 'when to decide to choose' depends on mental states and attitudes of agents [?]. A more 'cautious' type of agent will skeptically continue to process two alternatives, perhaps risking overshooting the deadline, but a more 'dashing' type of agent will take the risk to pursue just one plan. We have developed a heuristic rule under the following commonsense observation: An agent can continue to work on several plans provided there is *ample* time ahead to try and pursue them one after another in the interest of fault tolerance. For example, even after calling the driver to stop the train, Dudley may want to run to the railroad track and attempt the rescue Nell nevertheless, if there is enough residual time. An agent may do so as a guard against possible failure of his own or other agents' plans, or perhaps as an extra precaution when the plans are not recognized to be mutually exclusive. We look then at the sum of the WET's of all the plans in the IL as a measure of the overhead planning time. When the sum of the WET's and Now exceeds the deadline, he drops a plan from the IL. We currently have the simple heuristic of dropping the plan with the highest WET, but recognize that this may very well be the most refined plan as well[12]. Additional bookkeeping is necessary to ensure that two rules do not alter the IL or the focus simultaneously. We skip these implementation details in this description.

[10] The retrieval rule is a weak parallel of the inheritance rule in Elgot-Drapkin's step logics, in the sense that formulae in the STM at the previous step reappear in the STM at the current step provided they are *still* relevant.

[11] Probably, other scheduling procedures that were developed by operating Systems researchers can be used here,but it is beyond the scope of our paper. We only demonstrate how such procedures can be used *in* time.

[12] If one can find a way to include good estimation of the planning time (and probably decision time) into the WET it seems that more refined plans will have less planning time than other plans. Maybe, the three parts of the WET should not be combined and the decision whether to knock out a plan from the IL should be made using some sort of multi attribute decision rule (i.e., based on executing time, planning time and decision time). Again, we can't get into it in this paper.

Reduce IL rule (RILR):

$$\frac{i : Now(i), IL(L), wet\_ordering([p_{j_k}, ...]), ...}{i+1 : IL(L - p_{j_k})}$$

if $\sum_{l \in L} WET_{p_{j_l}} + Now > Deadline$.

An agent may be forced into a decision if two or more plans are ripe for action and the actions are mutually exclusive. The agent must evaluate the relative merits of the plans before making the decision, if acting on one will commit the agent to one plan. Although we do allow planning and acting to be interleaved, we allow the agent to act on a plan if it is the only one in IL. This is to avoid the complex interactions between plans as the result of the changed state of the world following the execution of one plan. We continue to examine this issue in ongoing work.

4.2.4 Capacity of the inference engine

As mentioned earlier, we suggested a limited capacity inference engine that would fire a cumulative set of inference rules to not exceed its inference capacity in each time step. In the simplistic examples that we present, there is a very limited number of rules firing at each step. Furthermore, if the plan length is within a reasonable bound, drain factors of the rules are also quite small and as a first approximation we postulate them to each take roughly the same time and fire in parallel in a single step whenever applicable. It should be noted that the meta rules for resource limited reasoning which were described above fire alongside the other object level inferencing at each step as part of a uniform framework. If we limit the capacity of the engine, the meta rules that are fired will limit the number of planning rules that are fired in each step.

4.2.5 Some illustrations from two plans

Dudley begins to formulate a plan *save* to get Nell *Out_of_danger*. Initially, the focus consists of $\mathbf{Focus}(j, plan\_in\_focus(save, [Out\_of\_danger(...)]), ...)$, and the interleaving list is $IL([save])$. Here, *save* is the name of the partial plan and is used to retrieve formulae related to the plan such as its WET, its context set, projection etc. The list of keywords for this plan contains *Out_of_danger*. It is used to retrieve axioms from the LTM whose right hand side matches the keyword. Thus, the plan *save* bifurcates into $save_1$ and $save_2$ based on the following axioms which are retrieved from the LTM:

$Pull(T : T+1, Y, X, L) \rightarrow Out\_of\_danger(T+1, X, L)$

$Stop\_train(\text{T:T} + 2, driver) \rightarrow$
$Out\_of\_danger(T+2, nell, r)$

Plan 1: Pull her away from the tracks

$$\mathbf{Ppl}(save1, \left\{ \begin{bmatrix} \neg Tied(t_1, n, r) \\ Pull(t_1 : t_2, d, n, r) \\ Out\_of\_danger(t_2 \bullet\!\!\rightarrow Deadline, n, r) \end{bmatrix} \right\},$$
$\{t_2 \leq Deadline, t_1 = t_2 - 1\})$

Plan 2: Stop the train

$$\mathbf{Ppl}(save_2, \left\{ \begin{bmatrix} Knows\_about(\tau_1, n, dr) \\ Stop\_train(\tau_1 : \tau_2, dr) \\ Out\_of\_danger(\tau_2 \bullet\!\!\rightarrow Deadline, n, r) \end{bmatrix} \right\},$$
$\{\tau_2 \leq Deadline, \tau_1 = \tau_2 - 2\})$

The interleaving list is expanded to contain both $save_1$ and $save_2$ and Dudley continues to work on both feasible plans in a time-shared fashion. The focus thus contains $save_1$ for an interleaving period during which axioms for untying Nell and running to her are progressively retrieved from the LTM. Other facts of no relevance to the plan such as $color\_of\_eyes(...)$ or that are relevant to the other plan such as the axioms about dialing to get a connection are left alone in the LTM. After the period expires, $save_2$ is brought into focus and worked on in a similar fashion. It is not until much later that Dudley realizes that the sum of the WET's of both plans and *Now* is going to overshoot the deadline, and he must restrict the IL using the RILR rule. We show a snapshot of the two plans when this happens in Figure 1.

Dudley gives up the plan with the higher WET, which in this case happens to be the one to run to Nell, and executes the plan to go to the neighbor's house to call the driver to stop the train instead. The run to the railroad tracks is longer than the run to the neighbor's house. The sum of the WET's exceeding the Deadline, Dudley starts to run in the direction of the neighbor's house and removes $save_1$ from the IL, still retaining it in the list of feasible plans to be available in case of unanticipated

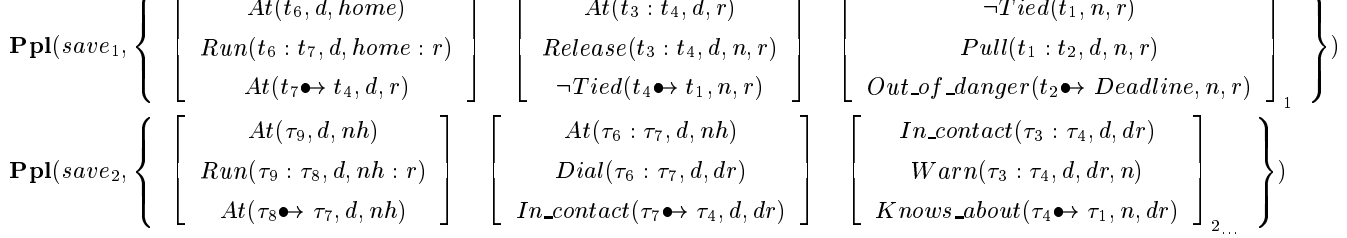

Figure 1: Pursuing two alternatives within space limitations. Dudley develops two alternative plans in a time-shared fashion until there comes a time when sum of their WET's is no longer within the deadline. The figure shows a snapshot of the two plans at such a time step. Dudley exercises a choice through the rule RILR which reduces the interleaving list to the plan to call the driver of the train. Abbreviations used are: n=nell, d=dudley, r=railroad track, nh=neighbor's house, and dr=driver of the train.

run-time failure.

5 Conclusions and future work

We have presented here a formalism for commonsense reasoning that attempts to address concerns of limited time, space, and computation. This is part of our continued work on fully deadline coupled planning. We have developed heuristics for the deadline-coupled planner that will allow it to function within tight resource limitations. We see two future directions for work. One that we are already pursuing is to improve these heuristics to scale up to more and more realistic deadline situations. The other may be to develop similar heuristics for other commonsense reasoning problems such as theorem proving or hypothetical reasoning.

References

[Allen and Koomen1983] Allen, J. and Koomen, J. 1983. Planning using a temporal world model. In *Proceedings of the 8th Int'l Joint Conference on Artificial Intelligence*, pages 741–747.

[Dean and Boddy1988] Dean, T. and Boddy, M. 1988. Reasoning about partially ordered events. *Artificial Intelligence*, 36(3):375–399.

[Dean *et al.*1988] Dean, T., Firby, R. J., and Miller, D. 1988. Hierarchical planning involving deadlines, travel time and resources. *Computational Intelligence*, 4:381–389.

[Dean1984] Dean, T. 1984. Planning and temporal reasoning under uncertainty. In *IEEE Workshop on Principles of Knowledge Based Systems*, Denver, Colorado.

[Elgot-Drapkin and Perlis1990] Elgot-Drapkin, J. and Perlis, D. 1990. Reasoning situated in time I: Basic concepts. *Journal of Experimental and Theoretical Artificial Intelligence*, 2(1):75–98.

[Elgot-Drapkin *et al.*1987] Elgot-Drapkin, J., Miller, M., and Perlis, D. 1987. Life on a desert island: Ongoing work on real-time reasoning. In Brown, F. M., editor, *Proceedings of the 1987 Workshop on The Frame Problem*, pages 349–357. Morgan Kaufmann. Lawrence, Kansas.

[Elgot-Drapkin1988] Elgot-Drapkin, J. 1988. *Step-logic: Reasoning Situated in Time*. PhD thesis, Department of Computer Science, University of Maryland, College Park, Maryland.

[Elgot-Drapkin1991] Elgot-Drapkin, J. 1991. Step-logic and the three wise men problem. In *Proceeding, AAAI-91*, Anaheim, California.

[Haas1985] Haas, A. 1985. Possible events, actual events, and robots. *Computational Intelligence*, 1.

[Horvitz *et al.*1989] Horvitz, E., Cooper, G., and Heckerman, D. 1989. Reflection and action under scare resources: Theoretical principles and empirical study. In *Proceedings of IJCAI-89*, pages 1121–1127, Detroit, Michigan.

[Horvitz1988] Horvitz, E. J. 1988. Reasoning under varying and uncertain resource constraints. In *Proceeding, AAAI-88*, pages 111–116, St. Paul, Minnesota.

[Korf1990] Korf, R. E. 1990. Real-time heuristic search. *Artificial Intelligence*, 42:189–211.

[Kraus and Rosenschein1992] Kraus, S. and Rosenschein, J. 1992. The role of representation in interaction: Discovering focal points among alternative solutions. In *Decentralized Artificial Intelligence, Volume 3*, Germany. Elsevier Science Publishers.

[Kraus *et al.*1990] Kraus, S., Nirkhe, M., and Perlis, D. 1990. Deadline-coupled real-time planning. In *Proceedings of 1990 DARPA workshop on Innovative Approaches to Planning, Scheduling and Control*, pages 100–108, San Diego, CA.

[McCarthy and Hayes1969] McCarthy, J. and Hayes, P. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B. and Michie, D., editors, *Machine Intelligence*, volume 4, pages 463–502. Edinburgh University Press.

[McDermott1982] McDermott, D. 1982. A temporal logic for reasoning about processes and plans. *Cognitive Science*, 6.

[Miller1956] Miller, G. 1956. The magical number seven plus or minus two. *The Psychological Review*, 63:81–97.

[Morgenstern1986] Morgenstern, L. 1986. A first order theory of planning, knowledge, and action. In Halpern, J. Y., editor, *Proceedings of the First Conference on Theoretical Issues in Reasoning about Knowledge*, pages 99–114, Monterey, CA. Morgan-Kaufmann.

[Nirkhe *et al.*1991] Nirkhe, M., Kraus, S., and Perlis, D. 1991. Fully deadline-coupled planning: One step at a time. In *Proceedings of the sixth international symposium on methodologies for intelligent systems*, Charlotte, NC.

[Nirkhe *et al.*1993] Nirkhe, M., Perlis, D., and Kraus, S. 1993. Reasoning about change in a changing world. In *Proc. of the Florida AI research symposium 93*, Fort Lauderdale. to appear.

[Pollack and Ringuette1990] Pollack, M. E. and Ringuette, M. 1990. Introducing the tileworld: Experimentally evaluating agent architectures. In *Proceedings, AAAI-90*, pages 183–189.

[Rosenschein and Kaelbling1989] Rosenschein, S. and Kaelbling, L. 1989. Integrating planning and reactive control. In *Proceedings of NASA Telerobotics conference*. Pasadena, CA.

[Russell and Wefald1989] Russell, S. and Wefald, E. 1989. Principles of metareasoning. In *Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*. Morgan-Kaufman.

[Shoham1991] Shoham, Y. 1991. Agent0: A simple agent language and its interpreter. In *Proc. Ninth National Conference on Artificial Intelligence*, pages 704–709. American Association for Artificial Intelligence.