10-1-1990

Progressive Horizon Planning

Ron Rymon
University of Pennsylvania

Bonnie L. Webber
University of Pennsylvania, bonnie@inf.ed.ac.uk

John R. Clarke
Medical College of Pennsylvania

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-76.

Progressive Horizon Planning

Abstract

In an earlier paper [Rymon et a1 89], we showed how domain localities and regularities can be used to reduce the complexity of finding a trauma management plan that satisfies *a set* of diagnostic and therapeutic goals. Here, we present another planning idea - *Progressive Horizon* - useful for optimizing such plans in domains where planning can be regarded as an incremental process, continuously interleaved with situation - goals analysis and plan execution. In such domains, planned action cannot be delayed until all essential information is available: A plan must include actions intended to gather information as well as ones intended to change the state of the world.

Interleaving planning with reasoning and execution, a progressive horizon planner constructs a plan that answers all *currently known* needs but has only its first few actions optimized (those within its planning horizon). As the executor cames out actions and reports back to the system, the current goals and the plan are updated based on actual performance and newly discovered goals and information. The new plan is then optimized within a newly set horizon.

In this paper, we describe those features of a domain that are salient for the use of a progressive horizon planning paradigm. Since we believe that the paradigm may be useful in other domains, we abstract from the exact techniques used by our program to discuss the merits of the general approach.

Comments

University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-90-76.

Progressive Horizon Planning

MS-CIS-90-76
LINC LAB 185

Ron Rymon
Bonnie L. Webber
(University of Pennsylvania)

John R. Clarke
(Medical College of Pennsylvania)

Department of Computer and Information Science
School of Engineering and Applied Science
University of Pennsylvania
Philadelphia, PA 19104-6389

October 1990

Progressive Horizon Planning

Ron Rymon, Bonnie L. Webber
Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA *

John R. Clarke
Medical College of Pennsylvania, Philadelphia, PA

rymon@linc.cis.upenn.edu, bonnie@central.cis.upenn.edu, jclarke@grad1.cis.upenn.edu

October 16, 1990

Abstract

In an earlier paper [Rymon et al 89], we showed how domain localities and regularities can be used to reduce the complexity of finding a trauma management plan that satisfies a *set* of diagnostic and therapeutic goals. Here, we present another planning idea – *Progressive Horizon* – useful for optimizing such plans in domains where planning can be regarded as an incremental process, continuously interleaved with situation-goals analysis and plan execution. In such domains, planned action cannot be delayed until all essential information is available: A plan must include actions intended to gather information as well as ones intended to change the state of the world.

Interleaving planning with reasoning and execution, a progressive horizon planner constructs a plan that answers *all currently known* needs but has only its first few actions optimized (those within its planning horizon). As the executor carries out actions and reports back to the system, the current goals and the plan are updated based on actual performance and newly discovered goals and information. The new plan is then optimized within a newly set horizon.

In this paper, we describe those features of a domain that are salient for the use of a progressive horizon planning paradigm. Since we believe that the paradigm may be useful in other domains, we abstract from the exact techniques used by our program to discuss the merits of the general approach.

*This research was partially supported by the Office of Naval Research under Subcontract Grant N00129-89-C-006 and under ARO Grant DAAL03-89-C0031PRI

1 Introduction

Classical planners [Fikes and Nilsson 71] descend from general problem solvers (e.g. GPS [Ernst and Newell 69]) that themselves originated from theorem provers and search programs. As a result, they view actions much like operators – well-defined transformations on a space of states. Planning is viewed as an independent process that takes a goal and an initial world state as its input, and produces a plan as its output.

However, actual planning differs in many ways from proving a mathematical formula. Almost by definition, acting involves a dynamic environment. So it is possible that by the time planning is completed, its very assumptions are no longer valid. This is particularly true of planning systems that employ computationaly costly algorithms. But even if planning is instantaneous, unforeseen changes in the world, unpredictable effects of actions and new information acquired *while the plan is executed* may in fact invalidate it[1]. Indeed, more and more people argue for the elimination of the artificial separation between planning and execution, and there is a growing attention to issues of uncertainty and unpredictability in planning[2].

Much of the research addressing those problems of classical planning has focused on reactive (or reaction) planning paradigms [Agre and Chapman 87, Georgeff and Lansky 87, Schoppers 89]. While very efficient in runtime, reaction plans may require exponentially large space (and thus exponentially large preparation time, see [Ginsberg 89]), and it is still unclear whether situations can be indexed and accessed effectively.

In this paper we argue that in unpredictable domains in which planning has to be done with only partial information available, a new planning paradigm – Progressive Horizon (*PH*) planning – can often serve as both a plausible and a computationally feasible compromise between the two extremes. We are currently using a *PH* planning technique in TraumAID for planning a management recommendation for trauma patients. Besides describing the paradigm we also characterize those features of the domain that justify its use.

2 Motivating Partial Plans

In this section we describe a few of the motivations leading us to prefer partial plans over complete ones. Plans can be partial in many ways: plans that rely on unproved (default) assumptions are partial, reactive plans that specify only the next action are obviously partial and plans that are verified only in a high, non-operational level of abstraction are also partial.

The type of partial plans of interest to us are *Partial Global Plans* – plans that are global in addressing *all* known goals, but yet partial in some other sense. These constitute a particularly interesting class of partial plans because they establish and keep to a plan's

[1] Strips' designers have later noticed the need to adapt plans to actual events and added *Triangle Tables* as a form of keeping parts of plan for future reuse [Fikes et al 72].

[2] "Planning in Uncertain, Unpredictable and Changing Environment" was the theme of [AAAI Spring Symposium 90].

overall course. Progressive Horizon Planning is one paradigm for constructing partial global plans. The plans it constructs are partial in that (a) not all goals are explicitly known at the time of planning, and (b) only their first few actions are globally optimized. The Progressive Horizon paradigm itself is described in section 3.

2.1 Problem Introduction

In trauma, it is common for patients to suffer multiple injuries. A choice of protocols for addressing single injuries, presented alone, are readily available, but in cases of multiple injuries the physician has to make choices as to what protocols to use and in what order. A good management plan is often a function of the *combination* of injuries.

Throughout patient management, a physician often faces two types of goals: diagnostic (acquiring evidence for or against the existence of a problem), and therapeutic (treatment of a diagnosed problem). In general, each goal can be satisfied with a one of a number of alternative procedures. These alternative protocols may be ranked to reflect their relative preference with respect to a given problem (based on their respective risk, cost, inconvenience etc). It is important to note that a procedure may satisfy several goals, all at once, and that so, two procedures may be ranked differently with respect to one another for different goals.

We would like to see a planner pick the best *set* of procedures for addressing the specified *set* of goals, and order them into a plan. Unfortunately, even if the need to order the chosen procedures is neglected, we have proven the procedure selection sub-task is NP-Hard [Rymon 90]. For reasons that will soon become clearer, even if this were feasible, *any* plan may soon become obsolete due to inherent unpredictability and uncertainty.

2.2 Lack of knowledge (Uncertainty)

In trauma, as in most areas of medicine, there is always uncertainty as to what injuries, external and internal, a patient has sustained from his/her wounds. Questions and tests are standardly used to get rid of uncertainties, but in multiple trauma one cannot settle all uncertainties before taking a therapeutic action. For example, if a patient is in shock, it must be eliminated before further diagnosis proceeds. As a result, one is forced to come to the following conclusions:

1. Often a plan has to be constructed with sub-optimal knowledge of a patient's condition and consequently with incomplete knowledge of all the therapeutic goals that will be required.

2. Since missing information may be crucially important to management success, any plan must include actions that are exploratory in nature and that do not necessarily have any corrective effect.

3. Emerging goals and new information acquired while the plan is *executed* may jeopardize the validity of the rest of a plan or require substantial changes.

2.3 Unpredictability

In most real systems, not everything is within the full control of the planner or plan executor. This is particularly true of systems such as TraumAID that does not even have its own perception or execution capabilities. As opposed to uncertainty (lack of knowledge), the effects of unpredictable events often become apparent only after-the-fact. There are many different sources of unpredictability that affect an agent:

1. Actions that were not executed correctly.

2. Actions that were executed correctly, but had unexpected results. This unexpected result may in some cases be failure, but may also be the detection of a good condition that was unaccounted for.

3. Actions taken by other agents – in TraumAID, there are at least two agents other than the system. One is the patient, whose condition may change independently of the system's expectations. The second autonomous agent is the physician. TraumAID has no control over the physician's actions. The system has no perception, and can only recommend action. As a result it must be able to make use of new information reported by the physician whenever this information becomes available and accommodate whatever the physician chooses to do regardless of its correlation with the planned activity. The system may well need to change its plan accordingly.

Note that unpredictable events can make new information available that may in fact *improve* TraumAID's ability to plan. In some cases, an emerging goal may be satisfied with procedures already scheduled for the satisfaction of other problems, or may require a simple subsumption of one procedure – originally selected for a pre-existing goal – for another that satisfies both goals. In any case, a robust system has to be able to reason about such events as they occur, and be able to change its plans accordingly.

3 Progressive Horizon Planning

3.1 Planning as Search in the Situation-Action Space

There is more than one way to view planning as a search problem: Classical planners viewed it as searching a space of plans (e.g. Strips [Fikes and Nilsson 71], Noah [Sacerdoti 77], etc). Alternatively, one can think of planning as searching through a connected space of world states. States in this space are world descriptors, connected to each other with directed edges. Each of these edges corresponds to an operator that transforms one world state into another. Actions constitute a particularly interesting class of operators, but operators such as *No-Action*, or *Information-Is-Received* in which the agent is not engaged in activity may also represent possible change to the world or to the agent's view of the world.

Ideally, each state in this *Situation-Action* (SA) space would consist of a *complete* description of the corresponding world state, or at least all relevant information about it. A planner would then be given an *initial world state*, a sufficient description of what should hold in a *goal state* and a set of *operators*, with a complete description of their preconditions and effects. Its task would be to construct a sequence of actions (a path in the SA-space) that when followed from the initial state would reach one of possibly several goal states. In the context of this ideal SA-space, planning can be viewed as simply searching a (possibly infinite) *directed* tree, rooted at the initial situation. We call this tree the *Situation-Action tree* (SA-tree).

Obviously, planning by simply searching some representation of the SA-space is not practical. Nevertheless, because the SA-space so closely resembles a simulation of reality, plans that were originally constructed using other representations may easily be transformed into the SA-space. Thus, the SA-space can serve as a common ground for evaluating artificial (and also human) planning techniques. We therefore chose the SA-tree as a platform for demonstrating the principles of *PH* planning. As will be noted in section 4, the *PH* ideas are not limited to the SA representation and can be extended to other representations as well.

The presence of *unpredictability* may complicate things: earlier, common assumptions were that (a) actions' effects are known and predictable, and (b) the acting agent is the sole source of change in the domain. It follows that $do(a, s)$ – performing action a in a situation s – is well defined as a unique next state in the SA-space. With unpredictability, these assumptions are no longer valid and a given action performed at a given situation may result in one of a *number* of possible states, each corresponding to a different outcome. This makes the use of brute force search even less desirable for planning. More importantly, *any* plan may fail as a consequence of unexpected change.

Uncertainty may also complicate things. In many cases, trauma management included, one may be forced to act before one has all necessary information. As a result a plan must include actions aimed at gathering more information having little to do with the physical achievement of particular goals. Note that without uncertainty, or for that matter in the presence of an uncertainty-clearing oracle, these actions could be eliminated without affecting the correctness of the rest of the plan. Uncertainty thus contributes to the *length* of the plan. In the context of an SA-tree search, the length of the plan corresponds to the *depth* of the search process.

3.2 An Overview

PH planning requires an architecture in which a basic cycle of reasoning, planning and execution is employed. In this cycle, a reasoner is responsible for continuous processing of available information, producing a list of current goals for the planner. The planner takes these goals together with goals-means information and passes on to the executor a complete plan for addressing all currently known goals. The executor reports back all actions performed, changes observed, and information acquired. All these are directly added to the reasoner's input to begin a new cycle.

PH planning proceeds in two steps: in the first stage an approximate plan is constructed which is then partially optimized in the second stage. Optimization is only applied to the first few actions – those that are within the planner's horizon. Then, as actions are carried out and their results are reported, this horizon progresses to include new actions. It is important to note that further actions are not completely ignored, but are considered when optimizing the first actions.

In TraumAID a rule-based reasoner is responsible for analyzing the current patient condition, and for generating diagnostic and therapeutic goals. The planner addresses these goals and presents the physician with the system's recommended plan. The physician is not expected to carry the plan to its end, but rather to report back to the system the results of any activity, whether suggested by the system or not, as well as any new information that becomes available. A new cycle of reasoning, planning and execution is then initiated[3]. Figure 1 depicts the system's cycle of operation.

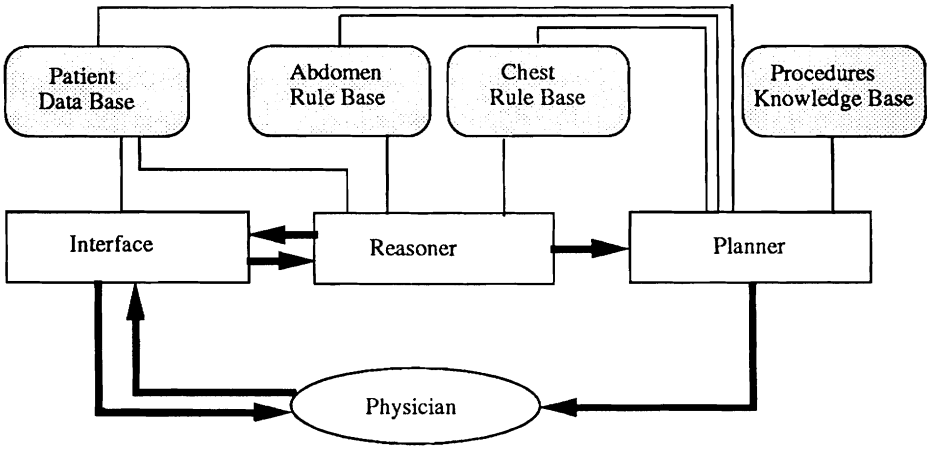

Figure 1: TraumAID's Cycle of Operation

3.3 Approximate Planning in TraumAID

In the approximate planning [4] stage of *PH* planning, we try to find a minimal set of procedures that will address all *known* goals, and that can be consistently ordered. As noted, the mapping between goals and procedures is many:many, in that a goal may be satisfied by more than one procedure and a procedure may be used to satisfy more than one goal. (For example, the diagnostic goal of ruling out *Tension Pneumothorax* can be satisfied by either a *Needle Decompression* or a *Survey Chest X-Ray*. On the other hand, a *Survey Chest X-Ray* can be used for many other diagnostic goals such as ruling out a *Hemothorax*, *Fractured Ribs*, etc.)

[3]For a more complete account, see [Webber et al 90].

[4]There is more than one interpretation for the term *approximate* in the planning community. Here, it describes a plan of a sketchy nature, which correctness has not been established.

3.3.1 Domain Regularities

Our algorithm takes advantage of the localities and regularities in the domain as it interleaves procedure selection and ordering in an iterated fashion. While these domain features do reduce the computational cost of plan construction, a complete consideration of all remaining combinations is still intractable.

1. Logistic regularities. Tracing the patient path within the hospital, we found that typically a patient is brought to the *Emergency Room* (ER). Then if radiographic tests are required that cannot be performed in the ER, the patient is transferred to the *X-Ray room* (XR). In either case, the patient may then be brought to the *Operating Room* (OR) for major surgery. After surgery is completed, or if it was not necessary, the patient is monitored in the *Trauma unit* (TU). Note that the patient never goes back in that chain. This suggests that actions performable at one site should always precede those that require facilities of a later site.

2. Standardized priorities. Standard practices of trauma care (ABC's[5]) call for attention to management goals based on their classification in the following order: Airway (and problems that interfere with patients breathing), Circulation (of blood), Neurological, Contamination, Stabilization (orthopedic) and only then all other problems.

3. Urgency-motivated alterations. We have identified that instability is not just another diagnosis or condition. The urgency to stabilize a patient has a major impact on the planning physician and a shock-causing problem will be addressed before other problems that under normal conditions would precede it.

3.3.2 The Algorithm

The input to the first phase consists of: (1) the set of goals Γ proposed by the *reasoner*, (2) all patient-specific data acquired in response to questions and previous tests, and (3) a knowledge base of *goals* and *procedures*, describing what procedures are capable of addressing what goals.

Given this input, the proposed selection and ordering algorithm does the following:

1. It sorts the set of goals Γ based on urgency and goal priority. Urgency is related to shock, or cause of shock, and indicates the need to address this goal promptly. Goal priority reflects standard practices in the management of multiple trauma (ABC's): first, goals involving patient airways, then ones involving circulation, then neurological problems, then contamination, then orthopedic stability, etc.

2. It then constructs a plan Π through the following iterated steps, stopping when Γ is exhausted:

[5] ABC stands originally for Airway, Breathing, Circulation etc. In our implementation, we have combined the first two.

(a) Pick the next goal γ on Γ. If γ is not already addressed by a procedure included in Π (recall that procedures can satisfy more than one goal), identify the most preferred procedure π for addressing γ that does not violate patient-specific contra-indications or require equipment that is not available – i.e., the procedure that would have been chosen for that patient, were γ the only problem to be addressed.

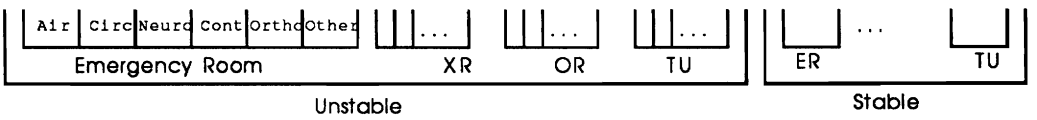

Figure 2: Sorting Procedures

(b) Add π to Π at a position that conforms to the order depicted in figure 2. If the patient is unstable (shock), instability-related procedures take precedence over other procedures. Otherwise, procedures are ordered logistically, with those performable in the Emergency Room ordered first, then those only performable in X-Ray, then those only performable in the Operating Room, and then those only performable in the Trauma Unit. Within a given location, procedures are ordered according to the standard practices of trauma care. Finally, location and priority being equal, therapeutic procedures for one condition take precedence over diagnostic procedures for another.

(c) Check that π does not violate any procedure already in the plan. If there is no way to locate π in the given plan, choose the next best procedure that addresses γ and repeat.

(d) If there is no valid way of addressing γ in the current plan, leave it unaddressed and inform the physician. Note that having ordered goals in Step 1 by urgency and priority, any goals left unaddressed will be less urgent and less important (vis-a-vis standard practices of trauma care) than any goal already addressed by the plan.

3.4 Optimizing to a Progressive Horizon

The algorithm described in section 3.3 is greedy. As a result, the plan constructed is not optimal in many ways: it possibly contains redundant procedures, a number of procedures may potentially be subsumed by a single procedure resulting in a more efficient plan, etc. Furthermore, scheduled procedures may then block the execution of a procedure that would ideally be recommended at the next cycle of reasoning and planning. The problem is that a complete optimization is very costly in terms of computation, and even if all necessary resources were allocated, unpredictability would avert any validation of that plan.

The main idea captured in the *PH* planning paradigm is that (a) an agent should limit the computational resources put into planning if it is to be practical, and (b) if so, and if the

domain has the characteristics previously described, more time should be spent on planning actions to be carried out in the near future than subsequently. This idea by itself is not new, Korf [Korf 90] has suggested the exploration of actions within a specific horizon. The *PH* approach differs in that a *complete* plan is constructed, but only its first part is optimized.

As mentioned before, in terms of the SA-tree, planning amounts to search and complete search is too costly, particularly as unpredictability may invalidate any plan even while it is constructed. A classical planner's search for a plan can be seen as a plan-space version of a complete search, taking exponentially long time. At the other extreme we find reactive planners trying to remedy this deficiency of classical planners by minimizing runtime consideration of alternative plans. However, while these planners differ in the way in which they index and identify the current situation[6] and in the form of their plans[7], all of them must represent, one way or another, *all* anticipated situations. In terms of an SA-tree, this can theoretically amount to representing all its nodes.

The *Real-Time A** algorithm [Korf 90] uses a representation that is very similar to the SA-space[8], and cuts search at a predetermined level, replacing further search with an evaluation function. This has obvious computational advantages over the classical and reactive approaches but it suffers from two problems: first, because it considers only the first few actions, it is bound to have an extremely short-term view of things. Second, in systems such as ours, it is not acceptable to provide the interacting physician with only the next few actions. For a plan being acceptable it must somehow address all currently known needs in a coherent fashion.

Instead of searching the entire SA-tree for an ideal plan P, a process that may fail anyway, a *PH* planner first comes up with an approximate plan P' and then works down the tree to a constant depth (denoted by C_{opt}) optimizing actions within that horizon (see figure 3). Whenever, a variant on the initial segment of the plan is tested, its effect on the *entire* plan is evaluated (i.e. including parts of the plan beyond the planning horizon). Doing that, we expect to *reduce*, but not eliminate the differences between P' and P. Like *RTA**, *PH* planning enjoys obvious computational advantages (see section 3.5), but it is also capable of generating global plans that address all known goals.

In TraumAID, we take the approximate plan Π, constructed by the algorithm of section 3.3, and perform a 1-depth optimization (i.e. looking for possible modifications of the first procedure). We currently employ two optimizers. Both are concerned with coverage properties, independent of the specific domain, and are generally applicable. Any possible alternation of the first procedure is evaluated in the context of the plan as a whole. Since in TraumAID, procedures are originally ordered based on urgency and importance, it follows that under a progressive horizon method, actions with higher urgency-importance rating at a given time are those that are optimized at that time.

[6]Universal plans [Schoppers 89] uses a decision tree structure, PRS [Georgeff and Lansky 87] uses rules, Pengi [Agre and Chapman 87] uses deictic representation.

[7]PRS [Georgeff and Lansky 87] uses KAs, which are a type of a program, Schoppers [Schoppers 89] constructs sequences of actions, TraumAID's first version flushes a preordered set of prescriptions, and other programs would even provide the executor a single action to be carried out next.

[8]It differs in assuming a single predictable result for each action.

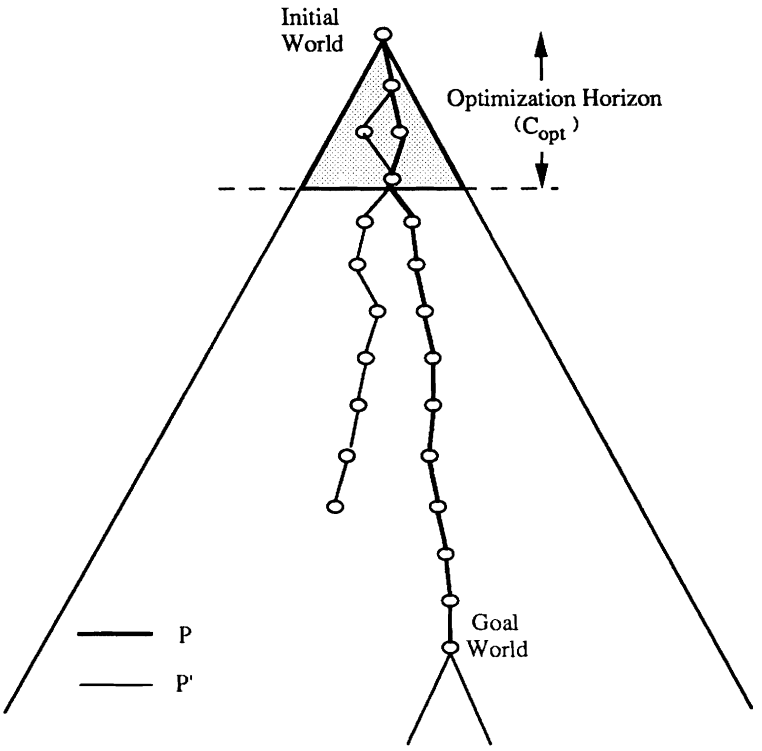

Figure 3: The SA-Tree and the Progressive Horizon Paradigm

3.5 Computational Analysis

Here we use the SA-tree platform to compare *PH* planning to other planning paradigm.

Let:

- a represent the size of the agent's repertoire of actions,
- n represent the size of the plan (the number of action instances included in the plan)
- b represent the branching factor of a system's *model* of the SA-tree ($b \neq a$, because an action may result in a number of possible states).

1. Complete run-time search – In a general SA-space, a complete Breadth-First search may require $O(b^n)$ time. However, if one assumes that actions' outcome is fully predictable, BF-search may "only" take $O(a^n)$. In any case, the time consumed is exponentially large in the length of the plan.

2. Reactive planning techniques may, in theory, need to represent all anticipated situations with corresponding recipes of action. For similar reasons this may require space (and pre-processing run time) of $O(b^n)$.

3. Under *RTA**, one searches to a given horizon, applies an evaluation function to the leaves, and propagates the results back up the tree. Let C_{opt} denote the search horizon and $f(a,n)$ denote the cost of evaluating a leaf at level n. Them the total cost of *RTA** is $O(a^{C_{opt}} + a^{C_{opt}} \cdot f(a,n))$ (replace b for a, for the more general SA-space).

4. Under the progressive horizon paradigm, one follows the steps:

   (a) Generate an approximate plan. For this purpose, one can use domain specific heuristics or other methods (e.g. [Elkan 89]). We denote the runtime requirement for this stage by $x(b,n)$.

   (b) Reason about each combination of the first C_{opt} actions and its appropriateness together with other parts of the plan. Let $y(n)$ be the time required to process one action, then this takes $O(y(n) \cdot a^{C_{opt}})$ time.

   An algorithm that searches to a progressive horizon would therefore have a total runtime of $O(x(b,n) + y(n) \cdot b^{C_{opt}})$. Assuming that $x(b,n)$, and $y(n)$ are both polynomial in b and n, then because C_{opt} is constantly fixed (or at least bounded by a constant) the total runtime is also polynomial of degree $Max\{C_{opt} + deg(y(n)), deg(x(b,n))\}$ in b and n.

4 Progressive Horizon from a Plan-Space Perspective

In a plan-space, nodes represent partial plans, edges stand for operators that map the space of partial plans onto itself, often expanding one action into several others. Figure 4 depicts one such plan: One usually begins with a very rough overall plan (drawn as the root of the tree in the figure) and adds and refines it until a plan with the desired features is found (finally drawn at the lowest level of the tree).

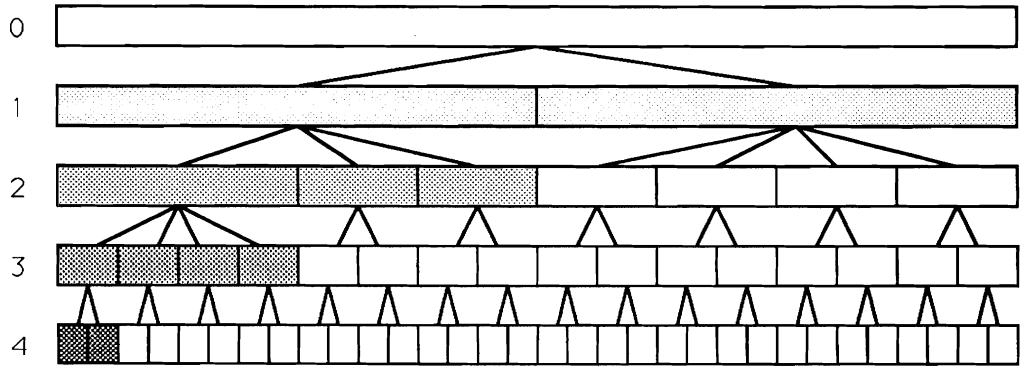

Figure 4: Plans-Space and the Progressive Horizon Paradigm

Early planners used means-ends analysis, or other information of a heuristic nature for locating actions that would contribute to their plans. Hierarchical planners [Sacerdoti 74,

Tenenberg 88], exploit a structured representation in which plans are discovered and maintained in various degrees of abstraction. If one takes the plan in figure 4: its higher levels would contain fewer, very abstract, action operators and going down the hierarchy, operators expand and their level of detail increases.

But most hierarchical planners ([Elkan 89] is an exception) continue their planning process until a plan is exposed in its entirety. One problem, noticed and addressed by other researchers, is that much effort is spent on resolving unimportant details. A hierarchical planner (e.g. [Tenenberg 88]) begins search of a detailed space only after a solution has been established in a more abstract level.

Another problem is that late actions receive as much attention as forthcoming actions. As explained before, in an uncertain, unpredictable environment this is not the most sensible thing to do as latter parts of the plan are often vulnerable to unexpected circumstances.

The progressive horizon idea can be used here as well: expand only nodes that would be scheduled in the beginning of the plan. Figure 4 shadows nodes visited by a progressive horizon planner: it would visit the whole plan in the lowest level of detail. Then, recursively, going down the hierarchy, it would visit only descendants of the leftmost procedure[9], leaving late actions in higher level of abstraction.

5 Summary

The full planning problem, however defined, is probably too hard to be solved in its entirety [Chapman 85]. It might therefore be of advantage to break it down into smaller *classes* of planning problems, based on common domain and problem characteristics.

In TraumAID's planner, we exploit domain and problem characteristics in two different ways: first [Rymon et al 89], we use domain localities and regularities to divide our search space into what [Lansky & Missiaen 90] calls "(almost) disjoint regions", facilitating the formation of an approximate plan. We then use a progressive horizon paradigm to limit the cost of an optimization pass.

The progressive horizon paradigm is useful for planning domains where uncertainty and unpredictability play a major role. We take unpredictability to stand for failed actions, actions that have unexpected results and other events that do not fall within the system's expectations. By uncertainty we refer to lack of knowledge, reflected in the fact that, most notably in early stages, one has to plan (and act) before diagnosis is completed. We show that *PH* planning is (a) computationally feasible, and (b) provides a plausible solution for such domains.

It is important to note that progressive horizon planning does not address *constructively* specific *types* of unpredictability and uncertainty exhibited by the domain. We nevertheless hope that it can be taken at least as a general advice while specifics are addressed in the actual implementation of search within the horizon and in confronting alternative plan initiations with the plan's completion.

[9]Note that we assume that actions in the plan are ordered left to right based on their temporal order in the plan.

Another question of particular interest is how to determine the appropriate value for C_{opt}. A planner that optimizes less than is appropriate would end up with too rough a plan, while too far an optimization is obviously wasteful. In general, C_{opt} appears to be inversely related to the degree of predictability in the domain. More specifically, the rate of unpredictable effects, ratio of positive/negative diagnosis, ratio of successful diagnosis and treatment, specificity of diagnosis and treatment, and number of alternatives for each procedure as well as the availability of informational, computational and time resources may all influence C_{opt}.

References

[AAAI Spring Symposium 90] Planning in Uncertain, Unpredictable or Changing Environments, *Working Notes*. AAAI Spring Symposium Series, Stanford, CA, March 1990.

[Agre and Chapman 87] Agre, P. and Chapman, D., *Pengi: An Implementation of a Theory of Activity*. AAAI-87. Seattle WA, August 1987, pp. 268-272.

[Chapman 85] Chapman, D., *Planning for Conjunctive Goals*. Masters Thesis, MIT-AI-TR-802, MIT Laboratory for Artificial Intelligence, Cambridge, MA, 1985.

[Elkan 89] Elkan, C., *Incremental, Approximate Planning*. KRR-TR-89-12, Technical Report, University of Toronto, November 1989.

[Ernst and Newell 69] Ernst, G., Newell, A., *GPS: A Case Study in Generality and Problem Solving*. Academic Press, New York, 1969.

[Fikes and Nilsson 71] Fikes, R. E., Nilsson, N. J., *STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving*. Artificial Intelligence 2, 1971, pp. 189-208.

[Fikes et al 72] Fikes, R. E., Hart, P. E. and Nilsson, N. J., *Learning and Executing Generalized Robot Plans*. Artificial Intelligence 3, 1972, pp. 251-288.

[Georgeff and Lansky 87] Georgeff, M. P. and Lansky, A., *Reactive Reasoning and Planning*. AAAI-87, Seattle WA, August 1987, pp. 677-682.

[Ginsberg 89] Ginsberg, M. L., *Universal Planning: An (Almost) Universally Bad Idea*. AI Magazine, Vol. 10, Number 4, Winter 1989, pp. 40-44.

[Korf 90] Korf, R. E., *Real-time Search for Dynamic Planning*. Proc. of AAAI Spring Symposium on Planning in Uncertain, Unpredictable or Changing Environments, Stanford, CA, March 90.

[Lansky & Missiaen 90] Lansky, A. L. and Missiaen, L., *Localized Search in GEMPLAN*. NASA Report RIA-90-04-032.

[Rymon et al 89] Rymon, R., Webber, B. L. and Clarke, J. R., *Responsive Planning In TraumAID*. Proc. 6th Israeli Conference on Artificial Intelligence and Computer Vision, Tel Aviv, December 1989, pp. 257-271.

[Rymon 90] Rymon, R., *Ph. D. Proposal (in preparation)*. Computer and Information Science, University of Pennsylvania.

[Sacerdoti 74] Sacerdoti, E. D., *Planning in a Hierarchy of Abstraction Spaces*. Artificial Intelligence, 5, 1974, pp. 115-135.

[Sacerdoti 77] Sacerdoti, E. D., *A Structure of Plans and Behavior*. New York: American Elsevier, 1977.

[Schoppers 89] Schoppers, M. J., *Representation and Automatic Synthesis of Reaction Plans*. Ph. D. Thesis. Computer Science, University of Illinois, October 1989.

[Tenenberg 88] Tenenberg, J. D., *Abstraction in Planning*. Ph. D. Thesis. Computer Science, University of Rochester, 1988.

[Webber et al 90] Webber, B. L., Clarke, J. R., Niv M., Rymon, R. and Ibanez, M. M., *TraumAID: Reasoning and Planning in the Initial Definitive Management of Multiple Injuries*. TR-MS-CIS-90-50, Computer and Information Science, University of Pennsylvania. Submitted to *Computers and Biomedical Research.*