Thesis Proposal:
Reasoning about when to plan and what to plan

Richard Goodwin
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, Pennsylvania 15213-3890

May 2, 1994

Abstract

Agents plan in order to improve their performance. However, planning takes time and consumes resources that may in fact degrade an agents performance. Ideally, an agent should only plan when the expected improvement outweighs the expected cost and no resources should be expended on making this decision. To do this, an agent would have to be omniscient. The problem of how to approximate this ideal, without consuming too many resources in the process, is the meta-level control problem for a resource bounded rational agent.

In this research, I propose to develop techniques for meta-level control that can be used to create resource bounded rational agents. The base level problem of creating and selecting a plan for the agent to execute will be modeled as a standard decision problem. The planning process that generates approximate solutions to the decision problem can be viewed as consisting of three anytime algorithms: plan generation, plan refinement, variable estimation. The meta-level controller is responsible for allocating computation to each of these algorithms and for focusing them on specific aspects of the problem. The meta-level controller will be provided with performance profiles for each algorithm and a representation of the agents utility function. An approximate sensitivity analysis will then used as the basis for implementing meta-level control. Information derived from the sensitivity analysis can be used to estimate the value of further plan generation and refinement and to identify which variable estimates need to be refined. A sensitivity analysis also identifies plans that are dominated or not potentially optimal. The meta-level controller can then exclude such plans from further consideration.

The objective of this research is to develop techniques for efficient and effective decision-theoretic meta-level planning control that can be applied to a wide variety of agents and tasks.

Introduction

Planning is the process of creating a policy for acting that an agent can use to increase the likelihood of achieving its objectives. Classical artificial intelligence planning methods formulated an agent's objectives as a set of first order predicate logic clauses. A plan, in this framework, is a sequence of operators, possibly with branches and loops, that takes an agent from an initial state to a goal state. AI planning research has been concerned with how to represent plans, actions and states and how to efficiently create plans given these representations. An AI planning agent is rational if it takes an action whenever it believes the action would increase the likelihood that the agent would achieve its goals [Newell, 1982].

Decision theory, developed in the context of economics and psychology, recognizes that the agent's objectives may not be captured by a single set of goal states. Instead, a utility function is defined over the set of possible outcomes and an agent is rational if it acts to maximize the expected utility. This allows an agent to take into account the relative preference for outcomes as well as their likelihood. Decision theory does not address the problem of how alternatives should be generated in the way that artificial intelligence planning does [Simon, 1988].

The problem with both classical artificial intelligence planning and decision theory is that they ignore the cost of computation. An agent is not performing rationally if by the time it has calculated the optimal action, that action is not longer applicable. Taking the cost of computation into account leads to what Simon calls procedural rationality [Simon, 1976] and what Good refers to as type II rationality [Good, 1971]. An agent exhibits such bounded rationality if it maximizes its expected utility given its computational and other resource limits. The problem is how to efficiently allocate resources, including computational resources, to produce a bounded rational agent. This is especially difficult because computation may be needed to determine how to best allocate the computational resources.

The Problem

In decision theoretic planning, the object level problem of selecting actions for execution is modeled as a decision problem. Solving a decision problem consists of three steps: formulating alternatives, estimating resource use, probabilities and related variables and selecting the alternative with the highest expected utility. The problem for the meta-level controller it to judiciously allocate computational resources to approximate the complete solution.

The first step in solving a decision problem is to enumerate the possible alternatives. Enumerating all the possible alternatives will take too long for all but most trivial of problems. A plan generator should ideally only generate the most optimal alternative. This is not generally possible though. What can be done is to create a plan generator that tends to produce plans with a higher likelihood of being near optimal first. The plan generator can then be treated as an anytime algorithm that monotonically improves the quality of its current best plan as a function of the amount of time it is given to compute [Dean and Boddy, 1988]. Greater efficiency is possible if the plan generator creates abstract plans. Each abstract plan corresponds to a set of complete plans. If an abstract plan can be eliminated from consideration, then the whole set of corresponding complete plans is eliminated without having to generate the detailed plans. The problem for the meta-level controller is to decide when further abstract plans should be generated and which abstract plans should be elaborated and which should be discarded.

In order to evaluate the expected utility of a plan, estimates of variables such as the durations of actions, resource use and probabilities are needed. Action models used to generate plans can be augmented to include methods for estimating their resource requirements and duration. Probabilities can be estimated from prior knowledge about the environment or through simulation techniques like temporal projection [Hanks, 1990]. As with plan generation, greater efficiency is possible if the planner initially generates approximate estimates. Only those estimates deemed critical then need to be refined. The problem for the meta-level controller is to decide which variables need to be estimated and which estimates need to be refined.

The final step in a solving a decision problem is to select the plan with the highest expected utility. In a complete solution, this step is straight forward. The parameter estimates are used to calculate the expected utility for each alternative and the alternative with the highest expected utility is selected for execution. When only some alternatives have been generated and not all parameter estimates have been refined the problem is more difficult. The meta-level controller must decide whether the current best alternative should be selected or whether more computation should be done first.

Motivating Example

Why is meta level control important and useful? The argument can be made that current robots move relatively slowly compared to the rate at which micro-processors can compute. Isn't it is always better to do more planning before beginning to act?

Consider our experience with controlling a Hero 2000 robot that was operating in the Learning Robots Lab at CMU. The robot was charged with performing a number of office related tasks [Simmons *et al.*, 1990]. These included delivering printer output, taking objects from one workstation to another and finding cups on the floor and putting them in a bin. New tasks for the robot become active asynchronously when people make requests or the perception system notices a cup on the floor. Combining new and existing tasks intelligently can produce much more efficient plans. For instance, suppose the robot was heading towards a workstation to pick up a package and a request was received to deliver printer output to the same workstation. If the printer was on the route to the workstation, then stopping off to pickup the output on the way would save a lot of time.

We implemented a system that would generate and evaluate plans for incorporating new tasks into the current plan [Goodwin and Simmons, 1992]. A set of experiments was conducted using our simulator to determine whether it was worthwhile considering all possible orderings of the tasks or if it was better to maintain the original ordering of the existing tasks. Considering all possible orderings requires more computation. Maintaining the original ordering of the existing tasks requires less computation but may miss opportunities for improving the plan. The experiment was a test of the applicability of intention based planning. An intention based planner makes a commitment to its existing plan and only considers options that are consistent with this commitment [Bratman *et al.*, 1988].

The results of the experiment were initially confusing. Maintaining the order of the original tasks produced the best results. But the amount of time spent computing, even for considering all possible task orderings was insignificant. Why didn't considering all options do at least as well as just considering a subset?

The answer had to do with the precision of the estimates used to predict how long it would take to move from one location to another. The time taken to move from one place to another was modeled as a linear function of the distance between the two locations. (Equation 1). The expected sensing overhead and average speed were estimated from a large number of trial runs. The resulting approximation accurately predicted the expected or mean time to do a move given the distance between the start and end locations. The problem was with the precision of the estimates. For longer distances, the estimates were relatively good and had relatively small variances. For shorter distances, the relative variance was high because distance between end points does not provide sufficient information to distinguish between qualitatively different situations. One problem case is illustrated in figure 1. The robot is attempting to drop something in one of the two bins. The bin on the right is closer, so the estimated time to move to this bin would be less. However, there might be a cup or other obstacle on the floor that the robot will have to go around. Depending on the placement of obstacles, it may actually be significantly quicker to go to the bin that is farther away. The estimated time provides some indication about which bin would be quicker to get to but because of the large variance, the estimate does not justify selecting one bin and rejecting the other.

$$E(Time(Move(a, b))) = \frac{distance(a, b)}{aveSpeed} + aveOverhead \tag{1}$$

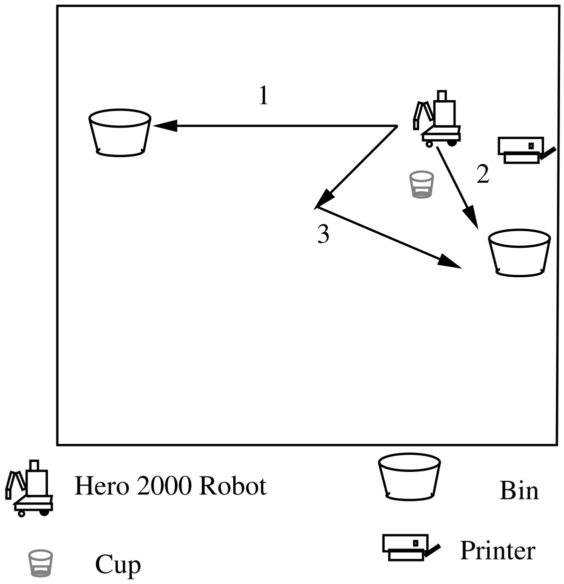

Figure 1: **Hero Robot example : Which bin to go to?**

A more accurate estimate of the travel time can be obtained by doing a detailed path plan. This involves plotting a path between the two end point that avoids obstacles and simulating execution of the path. In order to take account of objects like cups, the path planner needs sensor information. The robot can either used information stored from previous sensing or do new sensing. If old information exists, it is cheap to obtain but might not be accurate for the current situation. Sensing will provide current information but will be more costly to obtain. Also, since only one path will be taken, only a fraction of the sensor information collected will be used immediately. In the example shown in figure 1 plotting and simulating the two possible routes and selecting the shortest path would probably be the most efficient thing to do. However, if the bin on the left were two or three times further, then plotting both paths would probably be a waste of time. Always computing full paths has the potential to slow the robot down significantly. As the number of outstanding requests grows, the number of paths that could be considered grows exponentially. Also, in a dynamic environment, a plotted path loses value as time passes and the environment is more likely to have changed.

When can the inexpensive linear approximation be used? When is it worthwhile doing the more expensive path planning? To determine this, an estimate of the precision of the simple approximation is needed. When considering two alternatives, if the linear approximation shows that one clearly dominates another, given the precision of the approximation, then there is no need plan both paths. On the other hand, if the approximate estimates do not clearly distinguish the two alternatives, then further computation is likely warranted. One method of disambiguating these two cases is to use a sensitivity analysis. A sensitivity analysis would identify which parameters the choice between two alternatives is sensitive to. Focusing computation on refining those parameters will help to disambiguate the choice efficiently.

The Approach

A conceptual organization of a bounded rational agent is illustrated in figure 2. At the lowest level is the interface between the agent and its environment which provides the agent with its means of sensing and affecting the environment. The actions of the agent are controlled by an execution controller that is in

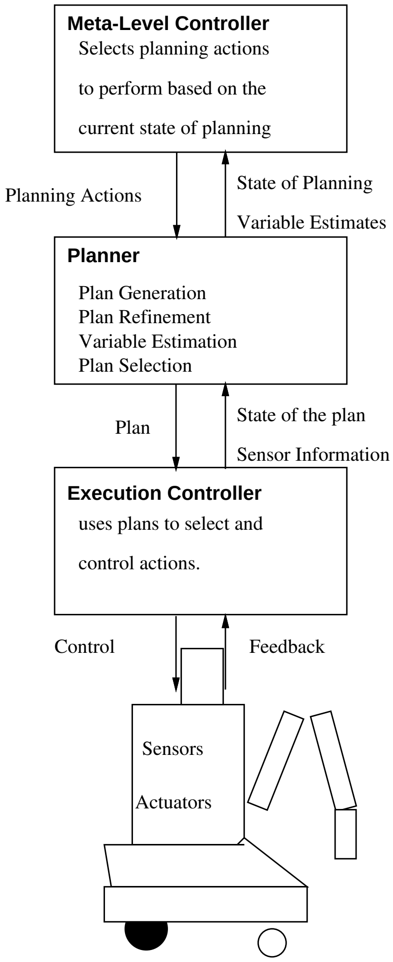

Figure 2: **Conceptual Organization.**

turn guided by plans provided by the planner. The planner gets information about the environment and the current state of plan execution from the controller and uses this information to generate and refine its plans. The meta-level controller gets information from the planner about the current state of the planning process and uses this information to direct the actions of the planner. The proposed work focuses on the meta-level controller that controls the planning process.

The basic approach to meta-level control is to use sensitivity analysis to focus computational effort to efficiently approximate a complete solution to the object level decision problem. Each of the computations required for solving the decision problem can, in general, be implemented using some form of iterative refinement. Each approximation improves as a function of the amount of computation done. To control the approximation process, we need to know its characteristics: the quality of the current approximation, the cost of improving it and the expected improvement. Given this information, sensitivity analysis can be used to select computations that will have the most significant affect on improving the overall approximation to the complete algorithm.

Sensitivity Analysis

Sensitivity analysis is concerned with determining how sensitive the results of a decision analysis are to the precision of estimated variable values. A decision is sensitive if relatively small changes in estimated values change the preferred alternative. If a system is found to be sensitive, the results of the sensitivity analysis indicates the relative sensitivity to changes in each variable or group of variables, suggesting which estimates should be refined.

A sensitivity analysis consists of a sequence of steps that first identifies the set of non-dominated and potentially optimal alternatives and then determines the minimum change in variable estimates that is needed to change the preferred alternative. The inputs to a sensitivity analysis are the set of estimated variable values, the bounds on these estimates and an evaluation function for evaluating each alternative. The output of a sensitivity analysis are the currently preferred alternative, the set of potentially competing alternatives, the relative sensitivity of the system and a sensitivity direction vector. The direction vector indicates the direction of change in variable space that the system is most sensitive to [Insua, 1990].

To understand the basis of sensitivity analysis, picture an n-dimensional space where each axis corresponds to the value of one of the plan variables. The bounds on variable estimates define a region in this variable space that is feasible. The evaluation function for each alternative gives the expected utility of the alternative in terms of variable values. It can be used to evaluate an alternative for points in the feasible space. Dominance is a relation between pairs of alternatives. One alternative dominates another if its expected utility is always at least as great as the other for variable values in the feasible space and for some feasible variable values is strictly greater. An alternative that is dominated by another can be eliminated from the set of potentially competing alternatives. Potential optimality is a relationship between the whole set of alternatives. An alternative is potentially optimal if there is some subset of the feasible variable space for which its expected utility is greater than or equal to the expected utility of all the other alternatives. If an alternative is not potentially optimal, it can also be eliminated from the set of competing alternatives.

The set of non-dominated and potentially optimal alternatives can be used to divide the feasible space into regions where a particular alternative or sets of alternatives are optimal. The current best estimate of variable values corresponds to a point in the region where the currently preferred alternative is optimal. The shortest distance from this point to one of the adjacent regions gives an indication of the absolute sensitivity of the system. Comparing this distance to the total distance spanned by the feasible space, gives a relative measure of sensitivity. The vector from the point representing the current variable estimate to a closest point in an adjacent region gives the direction of greatest sensitivity. Dividing the space up into regions where different alternatives are optimal also allows the potential opportunity cost to be calculated. The potential opportunity cost is the difference between the currently preferred alternative and the optimal alternative in a region. This cost represents the potential loss of opportunity to increase utility given that a completely refined variable estimate gave a value in this region.

The distance measure used to define closest in the sensitivity analysis does not have to correspond to a Euclidean distance. It is more useful if the distance measure reflects the relative cost in refining variable estimates. The sensitivity measure then indicates the minimum expected amount of computation that must be done before the preferred alternative changes. For instance, if there are separate computations to refine each variable, then a weighted Manhattan distance would be an appropriate distance measure.

Applying Sensitivity Analysis

To understand the role that sensitivity analysis can play in meta-level control, it is best to work backwards from the output of the planner. The ultimate purpose of the planning process is to create, refine and select plans to be passed to the execution controller. The selection of a particular plan must be justified by a belief that other known plans are not likely to be significantly better than the selected plan and a belief that significantly better and currently unknown plans either don't exist or are likely to be too computationally expensive to find.

A sensitivity analysis can be used to determine which of the known alternatives are potential competitors to the currently preferred plan. In addition, a sensitivity analysis gives the potential opportunity cost of selecting the current alternative and an estimate of the minimum amount of computation needed to change the preferred alternative. If the potential opportunity cost is low or the amount of computation required to switch alternatives is hight, then the agent is justified in selecting the current plan over the other known alternatives. If this selection is not justified, then the direction of highest sensitivity indicates where to focus planning in order to make a selection.

Alternative plans that are not potential competitors are either dominated or not potentially optimal and can be eliminated from consideration. The planning is more efficient if these plans can be eliminated earlier in the planning process. This saves work that would be wasted on refining these plans. Also, since plans at an early stage in the process are more abstract, eliminating a plan from consideration as early as possible cuts down the search space faster. This suggests that the computations needed for the sensitivity analysis should be done as early as possible in the planning process.

A sensitivity analysis can also assist in justifying a belief that no significantly better alternatives exists or that such alternatives would be too computationally expensive to find. One method of doing this is to by constructing a plan that gives a limit of the upper bound on utility, but that might not be operational. For example, a plan that ignored resource constraints or subgoal interactions. Including this plan in the sensitivity analysis would allow the planner to get a bound on the potential opportunity cost of forgoing additional plan generation. If this bound on opportunity cost were low, then the planner would be justified in believing that no significantly better plans exist. A performance curve giving the expected improvement versus planning time for the plan generator could be used to justify a belief that finding significantly better plans would require too much computation.

Examples

To illustrate this approach and its utility, I will present a set of related examples using two mobile robot domains: the Ratler and Odysseus, a Hero 2000. The set of examples follows roughly the set of steps needed to complete a sensitivity analysis. Each example is designed to highlight one aspect of the sensitivity analysis that is useful for meta-level control.

The APEX Planetary Rover Domain

The aim of the Apex project is to develop and demonstrate technologies that can be used to operate an unmanned lunar rover. The tasks for the APEX rover, called the Ratler, include visiting sites of interest, taking samples and performing scientific experiments. Some site visits and experiments will be pre-scheduled based on existing maps of the surface and information from previous manned expeditions. In addition, new tasks will be added on-the-fly as images and data from the rover are processed and analyzed. The rover must be able to efficiently incorporate these requests into its currently executing plan.

The operators for a simplified Apex rover domain include moving from one location to another along a given trajectory, performing sensing operations to determine the characteristics of the local terrain and taking samples. Figure 3 gives estimates of the expected speed of the rover and times for sensing operations.

In benign terrain, the rover can move forty centimeters per second. In rough terrain, the speed is reduced because the rover's power supply is limited and can't provide the power needed to maintain a higher speed. In the roughest terrain the rover can negotiate, the speed is reduced to ten centimeters per second. Equation 2 gives the range of expected travel times for moving between two points. The lower bound corresponds to benign terrain while the upper bound corresponds to rough terrain. The expected travel time is based on the a priori probability of rough terrain and existing sensor information.

Parameter Value		
Parameter	Value	Description
takeSample	120 s	Obtain a rock sample.
takePicture	1 s	Take and transmit a picture.
minSpeed	0.1 m/s	Speed over roughest ground.
maxSpeed	0.4 m/s	Speed over flat ground.
aveSpeed	0.25 m/s	Expected Speed.

Figure 3: **Expected Action Times**, (seconds or meters/second).

$$Time(Move(a,b)) = \left[\frac{distance(a,b)}{maxSpeed} \cdots \frac{distance(a,b)}{minSpeed}\right] \quad (2)$$

$$E(Time(Move(a,b))) = \frac{distance(a,b)}{aveSpeed} \quad (3)$$

Controlling Alternative Generation

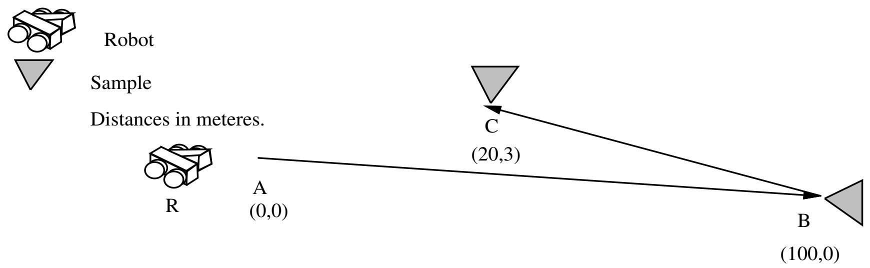

Figure 4: **Plan1 : makes use of the current intentions.**

In the course of executing a plan, new goals or tasks can become active. A request may be received or some new opportunity may be perceived. Whenever this happens, the rover must interrupt and reformulate its plan of action on-the-fly. The first step in this reformulation is to generate abstract plans that incorporate actions for accomplishing the new goal and the pre-existing goals. Even if we use a linearity assumption and limit the plan generator to decomposing the problem and solving each goal independently, there are still a combinatorial number of possible goal orderings. Removing this restriction further increase the number of potential plans that could be generated. How much time should be spent generating alternatives?

To illustrate my approach, consider a simple situation where the rover is proceeding towards a location to take a sample when a request to take another sample is received. One potential abstract plan is to service the new request by adding it to the end of the current plan as shown in figure 4. Such a plan is consistent with the rovers current intention of going to location B first.

From the geometry of the example, it is easy for us to see that this is probably a very sub-optimal solution and that more alternatives should be considered. How is the meta-level planing to determine this? One method would be to try to estimate the upper bound on the utility of the optimal solution. Figure 6 shows a plan where each goal is achieved in parallel from the current state. It assumes that the robot can do two things at once. The upper bound of utility is calculated by taking the maximum of the cost of each sub plan and the sum of the benefits. This in effect assumes that both tasks can be achieved at the cost of the most expensive one.

In this example, let's assume that there is some prior information about the roughness of the local terrain that allows the expected speed of the rover to be limited to the range 0.175 . . . 0.30 m/s. The parameters for the example are shown in figure 5. The calculations to determine the expected utility of the initial plan and the limiting plan are shown in equations 4 - 10. The tri-valued result gives the lower bound, the expected value and the upper bound.

$$\begin{aligned}
E(Time(SampleB|Plan1)) &= E(Time(Move(A,B))) + E(Time(takeSample)) && (4)\\
E(Time(SampleC|Plan1)) &= E(Time(SampleB|Plan1)) + \\
&\quad E(Time(Move(B,C))) + E(Time(takeSample)) && (5)\\
E(Time(SampleB|Plan0)) &= E(Time(Move(A,C)) + E(Time(takeSample)) && (6)\\
E(Time(SampleC|Plan0)) &= E(Time(Move(A,B)) + E(Time(takeSample)) && (7)
\end{aligned}$$

$$U(plan) = \alpha * numSamples(plan) - \beta * Time(plan) \qquad (8)$$

Parameter	Value
α	50
β	0.10
minSpeed	0.175 m/s
maxSpeed	0.3 m/s

Figure 5: **Parameters for Example.**

$$\begin{aligned}
E(U(Plan0)) &= (\alpha * numSamples(Plan0) - \beta * Time(Plan0)) && (9)\\
&= 2 * \alpha - \beta(E(Time(takeSample)) + E(Time(Move(A,B)))) \\
&= [30.9 \ldots 48.0 \ldots 54.7] \\
E(U(Plan1)) &= (\alpha * numSamples(Plan1) - \beta * Time(Plan1)) && (10)\\
&= 2 * \alpha - \beta(2 * E(Time(takeSample)) \\
&\quad + E(Time(Move(A,B))) + E(Time(Move(B,C)))) \\
&= [-26.9 \ldots 4.0 \ldots 16.0]
\end{aligned}$$

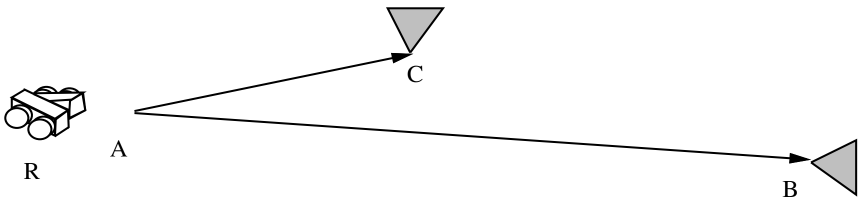

Figure 6: **Plan0, Upper Limit : do all actions in parallel.**

The calculation shows that there is a significant difference between the range of expected values for first alternative generated and the limiting plan. This indicates that there is a potentially significant opportunity cost of forgoing more alternative generation. Also, since only one alternative has been generated, it is likely that the rate of improvement for generating more alternatives will be high. This information can be obtained from the performance profile of the plan generator. In this situation, further alternatives should be generated for consideration.

Suppose the alternative shown in figure 7 is generated next. The expected utility of this plan is much closer to the upper limit on plan utility (Eq. 11). The remaining potential opportunity cost of forgoing further plan generation is greatly reduced. In this simple example, the possible goal orderings have been exhausted. In general, it would be appropriate for the meta-level controller to forgo generating additional alternatives and expend effort on detailed path planning, when the potential opportunity cost is sufficiently low. This is also true if the performance profile of the plan generator indicates that the rate of plan improvement is low.

$$\begin{aligned} E(U(Plan2)) &= (\alpha * numSamples(Plan2) - \beta * Time(Plan2)) \\ &= 2 * \alpha + \beta(2 * E(Time(takeSample)) \\ &\quad +E(Time(Move(A, C))) + E(Time(Move(C, B)))) \\ &= [18.7 \ldots 35.9 \ldots 42.6] \end{aligned} \tag{11}$$

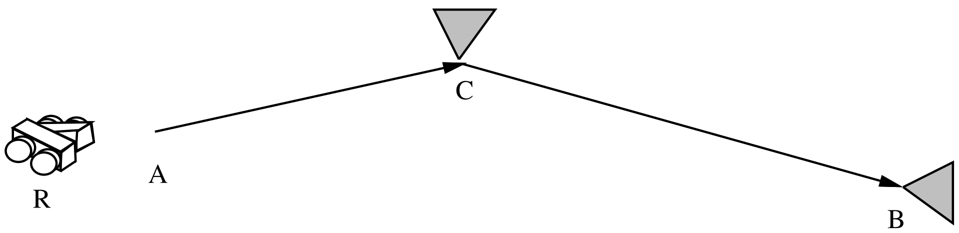

Figure 7: **Plan2 : second plan generated dominates the first.**

Eliminating Dominated Alternatives

The plan generator has now created two alternatives for consideration (Plan1 and Plan2). Since the range of expected utility of the second plan dominates that of the first, the first plan can be discarded. No refining of the estimated expected utility will ever make the first plan the preferred plan.

It is not always the case that the range of expected values can be used to determine when one plan dominates another. Consider an extended version of the first example where the rover is completing a series of samples when a new request is received. Figure 8 shows both alternatives in this context. In this example, the ranges of expected utility do not show dominance even though the difference in the means between the two plans is that greater than in the first example. The problem is that the uncertainty in the tail of both plans swamps the difference in utility of the head of each plan. Simply looking at the range of expected utilities does not take into account the fact that the tails of both plans are the same (d->e->f). Instead, when trying to establish dominance, the difference between two plans must be considered. Equation 14 gives the difference between the two plans calculated and simplified symbolically. Evaluating the symbolic difference shows that the difference is always negative, indicating that the second plan is dominant and that the first plan can be discarded.

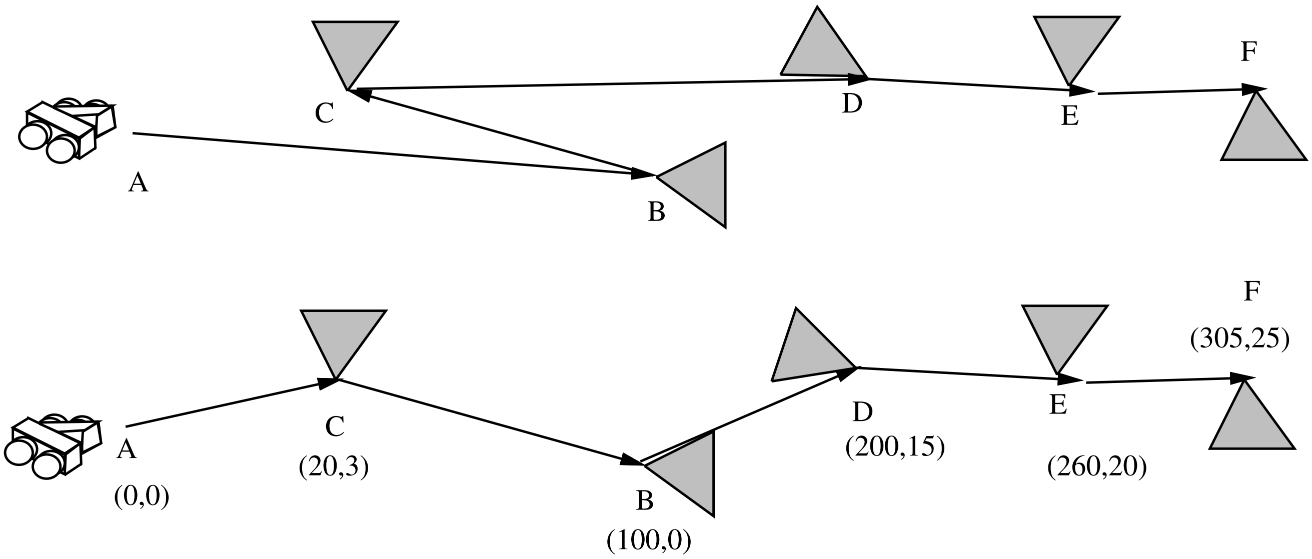

Figure 8: **Second plan still dominates the first.**

$$
\begin{aligned}
E(U(Plan1)) &= [-76.3 \ldots 3.6 \ldots 34.7] && (12)\\
E(U(Plan2)) &= [14.6 \ldots 67.2 \ldots 87.7] && (13)\\
E(U(Plan1)) - E(U(Plan2)) &= \alpha * numSamples(Plan1) - \beta * Time(Plan1) && (14)\\
&\quad -(\alpha * numSamples(Plan2) - \beta * Time(Plan2) \\
&= \beta(E(Time(Move(A,B))) + E(Time(Move(B,C))) \\
&\quad +E(Time(Move(C,D))) - E(Time(Move(A,C))) \\
&\quad -E(Time(Move(C,B))) - E(Time(Move(B,D)))) \\
&= [-119.8 \ldots -63.6 \ldots -24.1]
\end{aligned}
$$

Taking the difference between the expected utility of the two plans provides additional information for the meta-level controller. In the simplified expression for the difference, the terms common to both plans have dropped out. If the range does not indicate which plan is dominant, then the remaining terms are the ones that could be further refined to determine the best plan. The range of the difference also gives a bound on the opportunity cost of selecting one of the plans without further refinement. If this opportunity cost is relatively small, then it would be better to just select the highest estimated expected utility.

Taking the difference between two plans and simplifying the expression may not always show when one plan dominates another. The reason that it works for the previous example is that the expected utility is linearly dependent on the uncertain variables. This does not hold for more complex utility functions.

Equation 15 is a modified form of the original utility function that takes the time distribution of utilities into account. Each future cost or benefit is discounted by a fix discount rate (d). Discounting future values creates a preference for benefits that accrue sooner and costs that occur further in the future. A utility function that discounts future values is not linear in the parameters of the plan. Equation 16 gives the difference between the expected utilities for the two plans in the first example (Figures 4 and 7). Even making the simplifying assumption that Time(move(B,C)) = Time(move(C,B)) and doing symbolic simplification does not remove Time(move(B,C)) from the difference expression.

Compare the complexity of equation 16 with the expression for the linear case, equation 14. In the linear case, if we make the simplifying assumption that Time(move(B,C)) = Time(move(C,B)), then both can be removed from the expression. This is not the case with the non-linear utility function. Time(move(B,C)) remains in the expression although determining its exact values is not critical. The value of Time(move(B,C)) can affect the magnitude of the difference, but not the sign. The sign is determined by the relative magnitude of Time(move(A,B)) and Time(move(A,C)). How can the meta-level controller identify the relevant variables? Qualitative reasoning offers a partial solution. Each of the terms in the non-linear difference expression is either independent of Time(move(A,B)) or monotonically decreasing. Raising the lower bound on Time(move(A,B)) will necessarily lower the upper bound on the difference. Likewise, lowering the upper bound on Time(move(A,B)) will necessarily raise the lower bound on the difference. The converse holds for Time(move(A,C)). However, the affect of refining Time(move(B,C)) is not clear. Some terms in the difference equation monotonically increase with Time(move(B,C)) while others are monotonically decreasing. The affect of refining the parameter value will increase some terms and decrease others. The overall affect can't be easily determined. The qualitative relationship between the difference and the plan variables suggests that Time(move(A,C)) and Time(move(A,B)) should be refined before Time(move(B,C)).

$$\begin{aligned} U(plan) &= \sum_{i=1}^{n} \frac{\alpha}{(1+d)^{Time(sample(i|plan))}} - \int_{t=0}^{Time(plan)} \frac{\beta}{(1+d)^{T}} dt \\ &= \sum_{i=1}^{n} \frac{\alpha}{(1+d)^{Time(sample(i|plan))}} \\ &\quad - \frac{\beta((1+d)^{-Time(plan)} + Time(plan)(1+d)^{-Time(plan)} \ln(1+d) - 1)}{\ln(1+d)^2} \end{aligned} \tag{15}$$

$$\begin{aligned} E(U(Plan1)) - E(U(Plan2)) &= \frac{\alpha}{(1+d)^{Time(Move(A,B))+Time(takeSample)}} \\ &\quad + \frac{\alpha}{(1+d)^{Time(Move(A,B))+Move(B,C)+2*Time(takeSample)}} \\ &\quad - \frac{\alpha}{(1+d)^{Time(Move(A,C)+Time(takeSample)}} \\ &\quad - \frac{\alpha}{(1+d)^{Time(Move(A,C))+Move(C,B)+2*Time(takeSample)}} \\ &\quad - \frac{\beta\left((1+d)^{-Time(plan1)} + (1+d)^{Time(plan1)} \ln(1+d) Time(plan1) - 1\right)}{\ln(1+d)^2} \end{aligned} \tag{16}$$

$$+\frac{\beta\left((1+d)^{-Time(plan2)}+(1+d)^{Time(plan2)}\ln(1+d)(Time(plan2))-1\right)}{\ln(1+d)^2} \quad (17)$$

$$\begin{aligned} Time(plan1) &= Time(Move(A,B)) + Move(B,C) + 2*Time(takeSample) \\ Time(plan2) &= Time(Move(A,C)) + Move(C,B) + 2*Time(takeSample) \end{aligned}$$

Eliminating Never Optimal Alternatives

The preceding examples demonstrate how some plans can be eliminated from consideration because they are dominated by another plan. Plans can also be eliminated from consideration if they can be shown to be never optimal.

Consider an example were the rover can perform one of two destructive tests on a particular rock sample. One test is useful if the rock is igneous, while the other is more useful if the rock is not igneous. The rover can either just select one of the tests or it can visually inspect the sample to try to determine if it is igneous or not. Suppose that there are two visual inspection routines that can be used. Since these routines would be designed for the rover to use, it is reasonable for the rover to know their expected cost and accuracy. Knowing the likelihood of whether a randomly selected rock sample is igneous or not is not as reasonable. The rover may have geologic models that can be used to predict the likelihood of igneous samples or it may be able to consult a geologist. Using either of these methods may be expensive. For meta-level control, it is useful to be able to determine if some of the sensing routines can be eliminated without an estimate of the a priori probability of the sample being igneous. It is also useful if the meta-level controller knows the critical values for the a priori probability.

To show how a sensitivity analysis can answer these questions, consider the diagram in figure 9. The information that will be derived from this diagram can also be obtained from the sensitivity analysis. In the diagram, the four plans, two that use sensing and two that don't, are plotted as a "+"s. The x co-ordinate of each "+" corresponds to the expected utility, given that the a priori probability that the sample is igneous is 1.0. The y co-ordinate corresponds to the expected utility, given that the a priori probability is 0.0. The expected value of a plan for a particular a priori probability can evaluated geometrically. Simply draw a line from the origin with a slope is equal to $\frac{1-P(igneous)}{P(igneous)}$ and project the point representing a plan onto this line. The distance from the origin to the projection of the point on the line is the expected value of the plan.

The diagram shows thin dashed lines corresponding to P(igneous) = 0.42 and 0.61. The heavy dashed lines show the projection of the preferred plans onto these lines. For the two cases plotted, the plan that uses the first sensing operation has the same expected value as one of the plans without sensing. For a priori probabilities below 0.42, the preferred plan is to always perform test A. Above 0.61, the preferred plan is to always perform test B. Between these limits, the preferred plan is to use the first sensing action and then select the test based on the results. The two a priori probability values, 0.42 and 0.61, divide the probability space into three regions. The meta-level controller needs only determine which range the a priori probability lies in to determine the preferred plan. The exact value is not important. This information can be used decide which technique to use to estimate the a priori probability and to decide how accurate the estimate needs to be.

Note also that the point corresponding to the second sensing plan falls below and to the left of both heavy dashed lines. This plan is not dominated by any of the other plans. When the probability is 1.0, the second sensing operation is better than the first. When the probability is 0.0, the second sensing operation is better than always performing test B. There is however no a priori probability for which the second sensing plan is

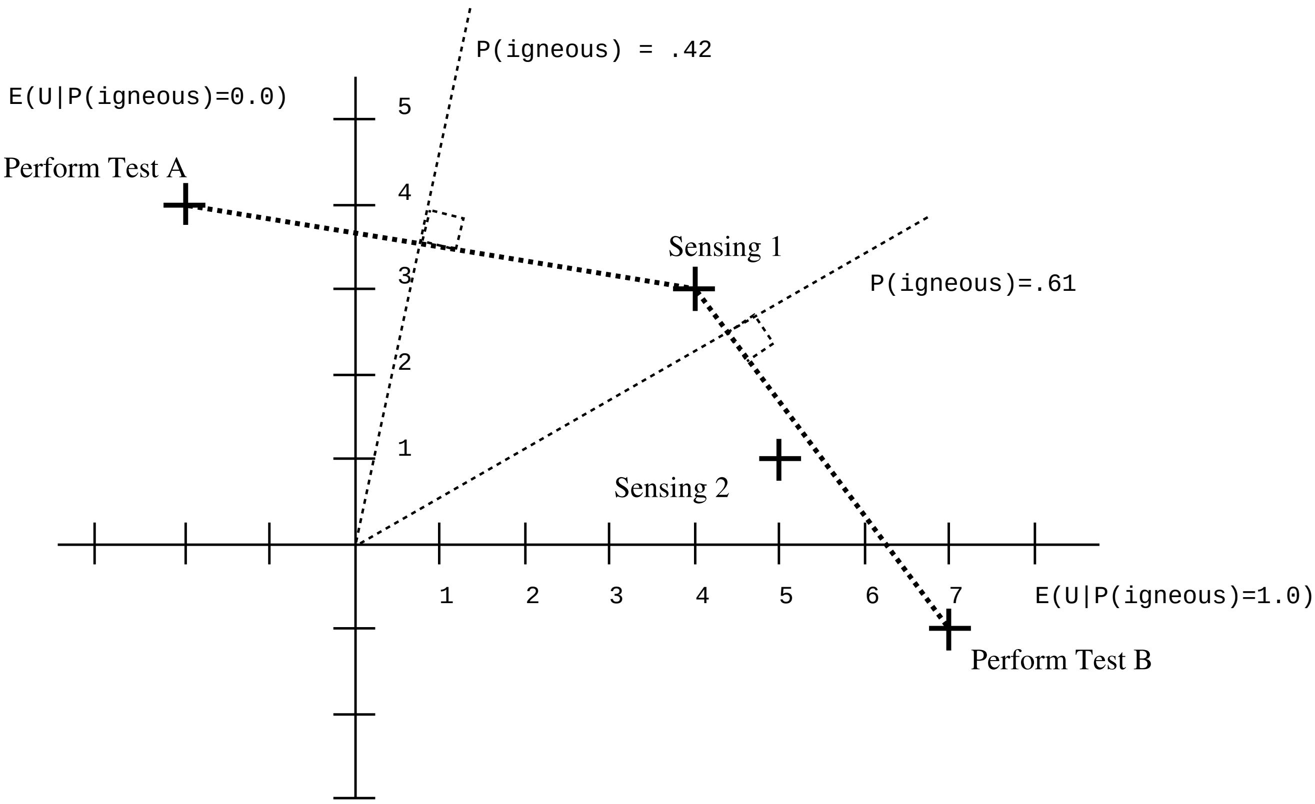

Figure 9: **Value of Sensing Plans.**

at least as good as all the other plans. Either the first sensing plan or the plan to always perform test B will be preferred. In the diagram, the heavy dashed line corresponds to the upper right edge of the convex hull of the points that represent plans. Only plans that correspond to points on this edge are potentially optimal. Any plan that falls below this edge is not potentially optimal and can removed from further consideration.

Perfect Sensing

When considering plans that gather information by sensing, it is useful to consider the value of perfect information. For this example, the value of a plan that uses a sensing operation that provided perfect, zero cost information is shown with an "X" in figure 10. This point represent a bound on how much the plan could be improved by doing further sensing. If the difference between the expected utility of the plan with perfect sensing and the current best plan is small, then it is not useful to spend more computation on trying to improve the current plan by modifying the sensing.

Randomized Strategies

It is also useful to consider plans that don't use sensing, but instead randomly select one of a set of potentially optimal plans. Points on the heavy dashed line in figure 10 represent the set of strategies that randomly selects test A or test B with some fixed probability. Any plan that falls below this line will be dominated by some of the random strategies. This allows us to place a bound on the acceptable cost of sensing. In situations where the cost of sensing is independent of the outcome and the a priori probability, then the line

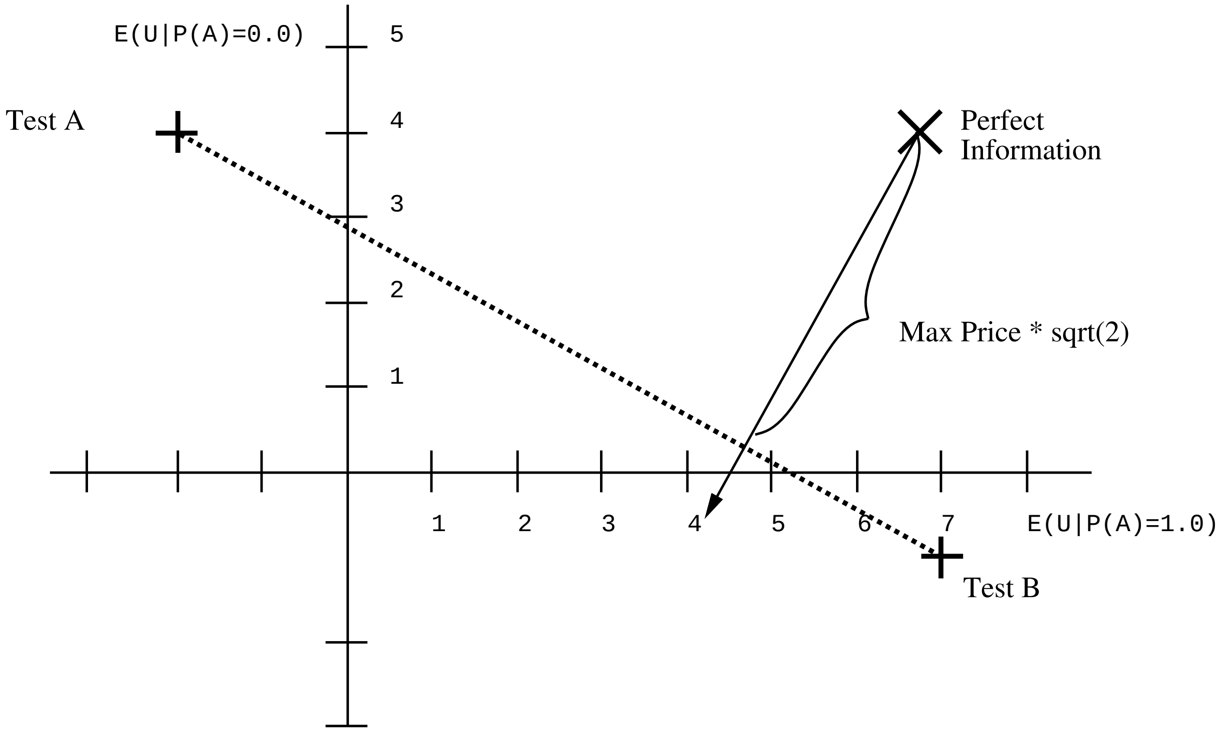

Figure 10: **Value of Sensing Plans.**

from the “X” in figure 10 shows how the expected value of a plan with perfect sensing changes as the cost of sensing increases. The expected value of the plan moves one unit down and one unit to the left for every unit of sensing cost. The point at which the expected value of the plan is on the heavy dashed line gives the maximum price that should be paid for perfect information.

Deciding Which Estimates to Refine

Sometimes the current parameter estimates are not sufficiently precise to form the basis for selecting one alternative over another. If there is a high potential opportunity cost associated with picking the current best alternative over other alternatives, it is worth while refining the parameters before making a decision. The direction of maximum sensitivity can be used to identify which parameters should be refined and indicates the relative importance of refining each one.

Below are the results of a sensitivity analysis of an example from the Hero domain. The example is taken from [Goodwin and Simmons, 1993]. In this example, the robot is putting a cup in the bin when it receives a request to deliver some printer output. The decision is whether to complete the cup collection or deliver the printer output first. (See figure 11)

The utility functions for the Hero domain are given in figure 12. The utility of delivering printer output depends on the delay between when the request is made and when the output is delivered. The utility of cup collection is not time dependent. The overall utility is discounted to give present values in order to create a preference for obtaining benefits sooner and delaying expenditures of resource.

There are three alternatives to consider:

a_1 Suspend cup collection and retrieve printer output.

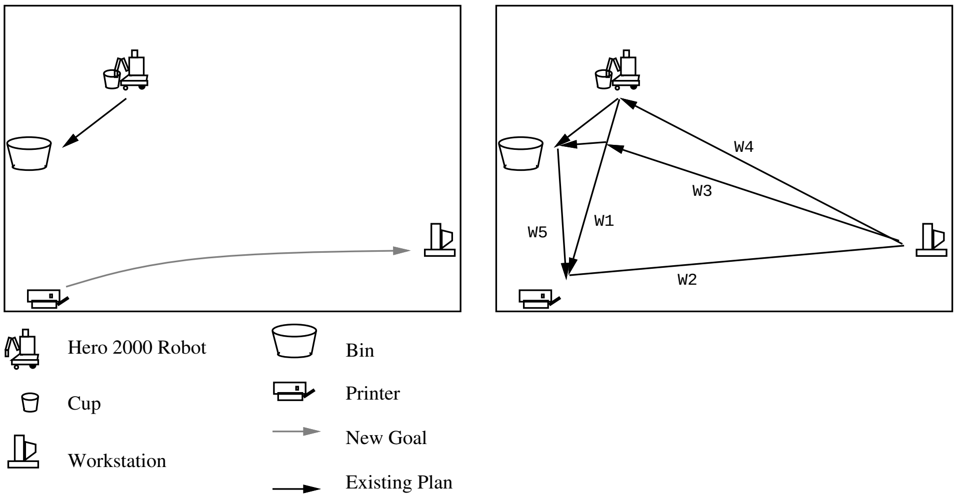

Figure 11: **Cup Collection Example.**

$$Utility(printer, delay) = \frac{100}{1 + e^{\frac{delay-200}{10}}} \tag{18}$$

$$Utility(delivery, delay) = \frac{200}{1 + e^{\frac{delay-200}{10}}} \tag{19}$$

$$Utility(collectCup, delay) = 10 \dots 15 \tag{20}$$

$$Utility(recharge, delay) = -e^{delay-200} \tag{21}$$

Figure 12: **Hero utility functions**

$E(Tprint1) = PutCupDown + W1 + RtoGrab + Grab + W2 + RtoRelease + Release$

$E(Tcup1) = E(Tprint1) + W4 + LineUp + GrabCup + Move((180, 50), (200, 68.0)) + DropCup$

$u1 = PV(Uprint(E(Tprint1)), E(Tprint1)) + PV(Ucup(0), E(Tcup1))$

a_2 Suspend cup collection on the way to the printer. Head towards the printer and put the cup down at the point closest to the bin.

$E(Tprint2) = W1 + PutCupDown + RtoGrab + Grab + W2 + RtoRelease + Release$

$E(Tcup2) = E(Tprint2) + W3 + LineUp + GrabCup + Move((184, 71.16), (200, 68.0)) + DropCup$

$u2 := PV(Uprint(E(Tprint2)), E(Tprint2)) + PV(Ucup(0), E(Tcup2));$

a_3 Complete cup collection then deliver the printer output.

$E(Tcup3) = Move((180, 50), (200, 68.0)) + DropCup$

$E(Tprint3) = E(Tcup3) + W5 + RtoGrab + Grab + W2 + RtoRelease + Release$

$u3 = PV(Uprint(E(Tprint3)), E(Tprint3)) + PV(Ucup(0), E(Tcup3))$

There are five partially estimated variables in the sensitivity analysis that correspond to five distinct move operators. Let $w_{1...5}$ correspond to the five move variables.

$$
\begin{aligned}
w &= [112, 201, 183, 185, 92] \textit{ expected variable values} \\
a_* &= a_3 \textit{ is the current preferred alternative.} \\
S &= \{[w_1, w_2, w_3, w_4, w_5] | \begin{pmatrix} 102 \leq w_1 \leq 250 \\ 191 \leq w_2 \leq 250 \\ 173 \leq w_3 \leq 250 \\ 175 \leq w_4 \leq 250 \\ 82 \leq w_5 \leq 250 \\ |w_3 - w_4| \leq 21.5 \\ w_1 \leq 26.9 + w_5 \end{pmatrix} \} \textit{ the set of feasible parameter values.}
\end{aligned}
$$

How sensitive is the choice of a plan to refinements of parameter values? This question can be answered by finding the minimum amount by which the parameter estimates have to change to cause a change in the preferred strategy. Since, in general, parameters can be expressed in different units, some method is needed to relate the change in each parameter to a common scale. A distance measure can to be defined to give the relative distance between two points in parameter space. The minimum change is then defined in terms of the distance measure. For this example, each parameter is already in the same units, so no conversion is needed.

Performing a sensitivity analysis shows that the system is relatively sensitive. A combined change of the equivalent of 45 centimeters in distance estimates causes the preferred alternative to change. Taking into account the allowable change in parameter values, the relative insensitivity of the system is 0.19. In this relative scale, 1.0 indicates an insensitive system and 0.0 indicates a completely sensitive system. The direction vector from the current best estimate of parameter values to the closest point where the preferred alternative changes is:[-2.3, 2.6, -0.13, 0.0, 4.9]. The components of this vector indicate the relative importance of refining each of these parameter values. In this case, refining the distance estimate for w5 would most quickly resolve the preferred alternative.

Methodology

The preceding examples highlight specific aspects of the utility of this approach to meta-level control. The list below gives a more complete and concise summary of points related to the methodology. Some of the points are tersely stated and this section can be skipped without loss of continuity.

- Model the object level action selection problem as a decision problem.
- Provide the agent with a representation of utility function so it can reason about it. More specifically, the agent needs to be able to evaluate the utility function and its derivatives at points and for ranges of parameter values. Qualitative evaluation may be sufficient in some cases.
- Model plan generation as an anytime algorithm and characterize the algorithms performance as a function of time. This will allow the meta-level controller to estimate the expected computational cost for improving the current best plan.
- Estimate the upper bound on expected utility by generating a plan where all tasks are accomplished in parallel and resource conflicts are ignored. The upper bound of utility is calculated by taking the

maximum of the cost of each sub plan and the sum of the benefits. Use this limit to estimate the potential opportunity cost of not generating anymore alternatives.

- Generate plans where the strategy is to randomly select one of a pair of non-dominated, potentially optimal alternatives. The expected value of such a plan can be used to determine the value of further computation for selecting among plans.
- Maintain information about the specificity of parameter value estimates and represent these as ranges.
- Use a sensitivity analysis to identify non-dominated potentially optimal plans. Dominated and never optimal plans can be discarded.
- Identify the potential opportunity cost associated with not selecting a potentially optimal alternative. This information can be used to decide if refining estimates or plans is warranted when trying to decide between alternatives.
- When calculating sensitivity, use a distance measure that is indicative of the expected cost of refining parameter values. Using such a measure to locate the "closest" point where another plan becomes optimal gives an indication of the amount of computation needed before the preferred option could change.
- If the potential opportunity cost warrants, focus planning effort in a way to reduce the sensitivity of the system. The direction vector to the closest point where another alternative becomes optimal can be used to select the relevant computations.

Approximate Sensitivity Analysis

Many of the situations where meta-level control would be useful tend to be complicated and non-linear. A complete sensitivity analysis for large, non-linear problems is computationally expensive. Fortunately, the information needed for meta-level control does not have to be accurate to ten decimal places. For meta-level control, a computationally less expensive approximation to a full sensitivity analysis is more appropriate.

One method of approximating a solution to the non-linear problem is to replace each non-linear constraint with its linear approximation and to solve the resulting linearized problem. This method can provides reasonable approximations, especially when the non-linear constraints are almost linear. For the Hero sensitivity analysis presented earlier, the linear approximation gives a result that is off by 4 %. If more accuracy is required, iteratively linearizing and solving the linearized problem can be used to refine a solution.

Qualitative reasoning provides a second approach to approximating a sensitivity analysis. Comparative analysis techniques give the qualitative response of a system to qualitative changes in system parameters values[Weld, 1987]. This can be combined with order of magnitude reasoning to determine which effects are significant [Raiman, 1991].

Related Work

The basic idea of bounded rationality arises in the work of Simon with his definition of procedural rationality [Simon, 1976]. Simon's work has addressed the implications of bounded rationality in the areas of psychology, economics and artificial intelligence [Simon, 1982]. In the area of agent design, he has considered how the nature of the environment can determine how simple an agent's control algorithm can be and still produce rational behaviour [Simon, 1956].

Etzioni's work on tractable decision-theoretic control produces efficient meta-level control by using an approximate greedy control algorithm [Etzioni, 1989]. Each time through the control cycle, the computational action with the highest marginal utility is selected for execution. The objective of the meta-level controller is to allocate the computational time before a given deadline to create a plan to achieve a set of goals. The planner learns the expected marginal return of each computational action by recording its performance on previous runs.

Decision theory has also been applied to controlling search. Russell and Wefald's DTA* algorithm uses estimates of the expected cost and expected gain in utility for possible computations to decide which computation to perform or whether to act [Russell and Wefald, 1991]. The algorithm is myopic and considers only the implications for the next action to be executed. The method for deciding which node in the search tree to expand can be cast in terms of a sensitivity analysis. The sensitivity analysis though, only considers the affect of changing one variable at a time. In some circumstances, this is an appropriate simplification.

Another method of directing search effort is McAllester's use of conspiracy numbers [McAllester, 1988]. This method for directing mini-max search can also be related to sensitivity analysis. Rather than measuring sensitivity as the change needed in one variable to affect the chosen action, a conspiracy number measures the number of variables that need to change value. A conspiracy number is a surrogate for sensitivity in that the sensitivity of a system tends to decrease as the conspiracy number increases. The search controller directs searching effort to increase the conspiracy number and thereby reduce the sensitivity of the system.

The idea of modeling planners as anytime algorithms that improve their output as a function of computing time was developed by Dean [Dean and Boddy, 1988]. Anytime models of the planning process can then be used to construct optimal deliberation schedules for the agent [Boddy and Dean, 1989]. In such systems, each of the possible computations contributes to the expected utility of the agent by refining different parts of the plan. Recent work has modeled the interactions of the agent and the environment as Markov chains[Dean *et al.*, 1991]. The planner alternates between doing policy iteration and modifying the subset of the states that the policy iteration is performed on. The currently, heuristic methods are used to decide how to modify the set of states considered for policy iteration. A sensitivity analysis approach could be used as the basis for making this decision.

Wellman's Sudo planner used qualitative reasoning to infer dominance relations between plans [Wellman, 1988]. The planner used qualitative probabilistic networks to deduce how changing one of the variables under control by the agent changed the variables that contributed to plan quality. The use of only qualitative models lead to many circumstances where the planner was unable to determine dominance relationships between plans. It was unable to switch to more precise representations that might have allowed additional dominance relations to be found.

The use of sensitivity analysis can also be considered in the more general context of model selection for reasoning. Any model of the environment is necessarily an abstraction. More abstract the models typically take less computation to estimate values, but the resulting estimates are less exact. A sensitivity analysis can help to determine whether estimates are sufficient or if models should be refined. In order to perform a sensitivity analysis, some estimate of the precision of the model predictions is needed. Chrisman's thesis work deals with relating the predicative power of more abstract model to more detailed models. The work is progressing towards a theoretically sound method of determining which conclusions are justified when interactions or distinct states are abstracted away [Chrisman, 1993].

Research Plan

Proposed Work

The core of the proposed thesis work consists of developing, implementing and evaluating the meta-level control methodology described in this proposal. The domains used to demonstrate the methodology will include the Hero robot domain and the Apex robot domain.

The work needed to develop and implement the proposed methodology can be divided in to two aspects: work on the planning components and work on the meta-level controller. The planning components consist of software to generate plans, to refine plans and to refine parameter estimates. Each of these components will be implemented in a manor suitable for the chosen domain. The plan generator will be augmented to produce randomized plans and to produce plans that can be used to determine upper limits on expected utility. Work on creating the meta-level controller will consist of developing and implementing algorithms to perform sensitivity analysis and algorithms to select computation for the planning components to perform.

Evaluation of the meta-level controller will be carried out through a series of experiments in each domain. The objective of these experiments is to determine if the meta-level controller can efficiently focus planning effort to create plans with high expected value. The performance of the sensitivity analysis based controller will be compared to strawman controllers that either always act or always plan. The purpose of the comparison is to asses the relative strengths of the sensitivity analysis based controller. The methodology will be considered successful if it can produce rational behaviour given the agent's resource limits.

Schedule

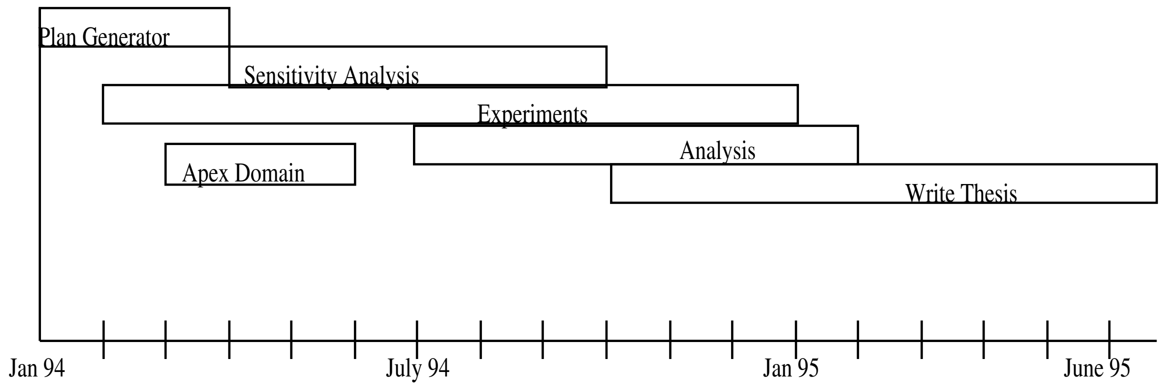

Figure 13: **Time Line.**

Future Work

Deciding when to begin execution is an important aspect of meta-level control that is not addressed by the proposed work. It is intended that the work proposed here on controlling the planning process can be used as a basis for extending the methods used to decide when to begin execution. Some initial work suggests that this may be feasible [Goodwin, 1994]. Such extensions are however beyond the scope of this proposed thesis and are left as future work.

Expected Contributions

Techniques for implementing efficient meta-level control for use in creating bounded rational agents. Theses include

- Techniques for applying sensitivity analysis to direct planning effort.
- Techniques for deciding when to generate more plans.
- Algorithm for generating potentially useful randomized strategies.
- Techniques for deciding when to refine parameters estimates and when to elaborate plans in order to select among them.

A further contribution will be an example meta-level controller that uses sensitivity analysis for focusing planning effort, with empirical results about its usefulness.

Bibliography

References

[Boddy and Dean, 1989] Mark Boddy and Thomas Dean. Solving time-dependent planning problems. In *IJCAI-89 Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*. AAAI, August 1989.

[Bratman *et al.*, 1988] M. E. Bratman, D. J. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4(4), 1988.

[Chrisman, 1993] Lonnie Chrisman. *Representing and Reasoning About Linitations in World Models (Thesis Proposal)*. PhD thesis, Carnegie Mellon University, April 1993.

[Dean and Boddy, 1988] Thomas Dean and Mark Boddy. An analysis of time dependent planning. In *Proceedings AAAI-88*, pages 49–54. AAAI, 1988.

[Dean *et al.*, 1991] Thomas Dean, Leslie Pack Kaelbling, Jack Kirman, and Ann Nicholson. Planning with deadlines in stochastic domains. In *Proceedings, Ninth National Conference on Artificial Intelligence*. AAAI, July 1991.

[Etzioni, 1989] Oren Etzioni. Tractable decision-analytic control. In *Proceedings, First International Conference on Principles of Knowledge Representation and Reasoning*, May 1989.

[Good, 1971] I. J. Good. Twenty seven principles of rationality. In *FOUNDATIONS of STATISTICAL INFERENCE; proceedings of the Symposium on the FOUNDATIONS of STATISTICAL INFERENCE*. Rene Descartes Foundation, March 1971.

[Goodwin and Simmons, 1992] Richard Goodwin and Reid Simmons. Rational handling of multiple goals for mobile robots. In J. Hendler, editor, *Proceedings of the First International Conference (AIPS92)*. Artificial Intelligence Planning Systems, June 1992.

[Goodwin and Simmons, 1993] Richard T. Goodwin and Reid Simmons. Rational handling of multiple goals for mobile robots. Technical Report CMU-CS-93-204, Carnegie Mellon University, Pittsburgh PA. USA, October 1993.

[Goodwin, 1994] Richard Goodwin. When to plan and when to act? In *AAAI 1994 Spring Symposium on Decision-Theoretic Planning*. AAAI, March 1994.

[Hanks, 1990] Steve Hanks. Practical temporal projection. In *Proceedings, Eighth National Conference on Artificial Intelligence*. AAAI, July 1990.

[Insua, 1990] David Rios Insua. *Sensitivity Analysis in Multi-objective Decision Making*. Lecture Notes in Economics and Mathematical Systems. Springer-Verlag, 1990.

[McAllester, 1988] D. A. McAllester. Conspiracy numbers for min-max search. *Artificial Intelligence*, 35(3):287–310, July 1988.

[Newell, 1982] Allan Newell. The knowledge level. *Artificial Intelligence*, 18(1):87–127, 1982.

[Raiman, 1991] Olivier Raiman. Order of magnitude reasoning. *Artificial Intelligence*, 51:11–38, 1991.

[Russell and Wefald, 1991] Stuart Russell and Eric Wefald. *Do the Right Thing*. MIT Press, 1991.

[Simmons *et al.*, 1990] Reid Simmons, Long Ji Lin, and Chris Fedor. Autonomous task control for mobile robots. In *Proc. IEEE Symposium on Intelligent Control*, Philadelphia, PA, September 1990.

[Simon, 1956] Herbert A. Simon. Rational choice and the structure of the environment. *Psychological Review*, 63(2):129–138, 1956.

[Simon, 1976] Herbert A. Simon. *From Substantive to Procedural Rationality*, pages 129–148. Cambridge University Press, 1976.

[Simon, 1982] Herbert A. Simon. *Theories of Decision-Making in Economics and Behavioral Science*, pages 287–317. The MIT Press, 1982.

[Simon, 1988] Herbert Simon. Problem formulation and alternative generation in the decision making process. Technical Report AIP-43, CMU psychology, 1988.

[Weld, 1987] Daniel S. Weld. Comparative analysis. Technical Report AI Memo 951, MIT, November 1987.

[Wellman, 1988] Michael P. Wellman. Formulation of tradeoffs in planning under uncertainty. Technical Report MIT/LCS/TR-427, MIT, August 1988.