Appears in the Proceedings of the AIPS 2002 Workshop on Planning and Scheduling with Multiple Criteria

An Optimization Framework for Interdependent Planning Goals

Tara A. Estlin, Daniel M. Gaines

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109-8099
{firstname.lastname}@jpl.nasa.gov

Abstract

This paper describes an approach for optimizing over interdependent planning goals. Most planning systems allow only simple, static dependencies to be defined among goals where these dependencies remain constant between different problems. However, in many domains, goals are related through detailed utility models that may significantly change from problem to problem. For instance in one problem, a particular goal's utility may increase if other related goals can be achieved. In another problem, this utility increase may differ or actually decrease if the same combination of goals is achieved. To address these types of problem situations, we have implemented a methodology for representing and utilizing information about interdependent goals and their related utilities using the ASPEN planning and scheduling system. We show through experimental results that this approach significantly increases overall plan quality versus a standard approach that treats goal utilities independently.

Introduction

As the sophistication of planning techniques grows, these systems are being applied to an increasing number of real-world problems. For instance, planning and scheduling techniques are currently being applied with great success to handle problems in manufacturing, logistics, and space exploration. In a typical application, a planner is given a set of goals, and it then constructs a detailed plan to achieve the goals where the plan must respect any applicable domain rules and constraints. A limitation of most planning systems, however, is that they define relationships between input goals in a simple, static manner, which cannot be easily adjusted for different problem situations. In many domains, goals can be related in complex and varying ways that are best represented through utility metrics. These metrics are hard to include as part of a standard domain definition, since they are often dependent on current data and can vary widely from problem to problem.

When planning for NASA spacecraft or rover missions, planning goals are often dictated by science data that has just been collected. For instance, an initial visual sweep of an area by onboard rover cameras may indicate more specific terrain targets that should be investigated. Relations or dependencies between science goals often only become apparent when goals are being instantiated based on current data. One example of a dependency between goals is when the achievement of one goal affects the utility of another goal. For instance, when collecting geology samples of a new planetary terrain, it is often helpful to take not only measurements of a target rock, but also science readings of its surrounding terrain and other similar rocks that may be encountered. The original sample would still be useful without these additional measurements, but it is much more valuable when related measurements are also taken. Conversely, in others situation these interdependency relations may change. For instance, it may be deemed more important to collect samples of several distinct rock types located in different areas as opposed to collecting several samples from only one particular rock.

These types of goal interdependencies can be seen in other domains as well. For instance, consider a travel-planning domain where we have a set of goals for planning a business trip for several different people to the same location. Thus, all travelers need to arrive at the same destination and in the same general timeframe. In most cases, they would all like to arrive on the same day and time, however, plans that have some travelers arriving one day earlier are still valid and would still be considered. Furthermore, preferences for when people arrive could change from trip to trip. On one trip it may be important that a certain set of people arrive on the same day to attend a particular meeting. On other trips this criteria may be less important or apply to a different set of people. Representing such information in current planning systems would be difficult since most goal dependencies are typically pre-defined using domain models and cannot easily change between problem instances based on new preference information.

Approaches to goal handling and representation vary widely among planning and scheduling systems. In some approaches, all goals must be achieved (i.e., included in the final plan) for the planner to even reach a solution. In other approaches, goals can be given different priorities or

utilities, and the planner will try to create a plan that achieves the highest utility score. Some goals may be deleted, or not added to the plan, so that the final activity sequence will correctly obey any relevant state and resource constraints and so that other more important goals can be achieved. Other approaches enable a planner to accept both goals and other quality objectives, such as minimizing makespan, avoiding missed deadline costs, or minimizing the usage of a particular resource (Williamson and Hanks, 1994; Joslin and Clements, 1999; Rabideau, et al, 2000). However, even in approaches that allow the usage of more flexible optimization metrics, goal relationships are pre-defined in a domain model and typically remain relatively constant between problem instances. Furthermore, it is difficult to define utility metrics that involve specific goal instances as opposed to a general quality concept that applies to a certain class of goals (e.g., increasing the number of orders filled).

Most planning systems do allow you to define some types of *static dependencies* between goals. For instance, two goal or action types could be defined as related in a domain model, perhaps through a decomposition of a parent activity. In a travel domain, you might often want to tie a "board-plane" action with a "deboard-plane" action, since both will commonly occur in the same plan. Some static dependencies may also be defined automatically through other parts of the model definition. For instance, pre- and post-conditions links can relate certain goals. A domain model does typically allow goals to be linked in different ways (e.g., a goal that could decompose to several different sets of actions or goals), however, these options are usually limited to several commonly-seen combinations. Encoding a large number of dependency options in a domain model would be intractable both for modeling ease and search complexity. No current planning systems enable *dynamic dependencies* between goals to be easily defined and utilized where these dependencies can significantly vary from problem to problem and can be defined as part of the problem specification instead of the domain model.

This paper presents a method for handling interdependent planning goals while performing plan construction and optimization. In this approach, interdependencies between goals can be formulated dynamically and provided to the planning system as part of the goal input. The planning system can then reason about these dependencies and incorporate them into the overall objective function it uses to rate plan quality and direct its search process. This optimization approach has been implemented on top of the ASPEN planning and scheduling system (Chien, et al., 2000). ASPEN already has a base optimization framework that we have extended to handle this class of problems (Rabideau, et al., 2000). This new approach has been tested on a series of problems based on a team of rovers performing geological experiments in a new terrain. Experimental results show that by using information about related goals, our approach is able to significantly improve plan quality.

This paper is organized as follows. First, we introduce the rover-application domain and autonomous science system that originally inspired this optimization work. We also describe the base planning system and optimization framework that we have used as a base platform. Next we describe our approach for plan optimization using interdependent goals. This section discusses how inter-dependent goals and their utilities are represented, how our objective function works, and how the overall optimization framework is designed. We then present experimental results using the previously introduced rover domain that demonstrate the effectiveness of our optimization approach. Finally, we overview related work, discuss future research ideas, and present our conclusions.

Multi-Rover Science Application

In recent years, NASA has begun to focus on missions that utilize rovers to perform exploration and understanding of planetary terrains. Future missions will likely send teams of rovers to explore planetary surfaces. These rovers will need to behave in a coordinated fashion where each rover operates autonomously, requiring little communication with Earth, but is still able to appropriately coordinate its activities with other rovers.

MISUS System

The Multi-rover Integrated Science Understanding System (MISUS) (Estlin, et al., 1999) was designed to control and coordinate multi-rover science operations. Using MISUS, a team of rovers can autonomously generate and achieve planetary science goals. MISUS integrates techniques from planning and scheduling with machine learning to enable multi-rover behavior for 1) analyzing science data, 2) evaluating potential new science observations to perform, and 3) deciding what exact steps should be taken to perform them. One of the primary motivations for the work presented in this paper was to support science-goal specification and achievement, where goal utilities can dramatically shift over time as more and more data is collected and analyzed. We also believe that the work presented in this paper is applicable to a large number of domains where goal dependencies and utilities can vary from problem to problem. We next give a short background on the MISUS system and then further explain the planning-optimization approach utilized by this system.

MISUS is comprised of the following major components:

- **Planning:** A distributed planning system that produces rover-operation plans to achieve input science goals (Estlin, et al., 2000). This system distributes science goals among rovers and plans for operations on each individual rover. A domain model is used to ensure plans to do not violate any resource or other rover-operation constraints.

- **Data Analysis:** A distributed machine-learning system that performs unsupervised clustering to model the distribution of rock types observed by the rovers. This system is designed to direct rover sensing with the goal of continually improving the scientific model of planetary scene.
- **Rover Environment Simulator:** A multi-rover simulator that models different geological environments and rover science activities within them.

MISUS operates in a closed-loop fashion, as shown in Figure 1, where the data-analysis system can be seen as taking the role of the scientist driving the exploration process. Science data is received by the data-analysis system, which integrates all gathered data into an updated global model. A prioritization algorithm uses the clustering output to generate a new set of observation goals that will further improve the accuracy of the global model. The specific value of a new science measurement is often dependent on what data has already been collected as well as what other related measurements can be taken. Both individual goals and specified goal combinations are assigned utilities reflecting their overall value to the science system. These goals are passed to the planning system, which assigns individual rovers to goals and produces a set of actions for each rover to achieve as many of its assigned goals as possible. These action sequences are then executed in the environment simulator. Any gathered data is sent back to the data-analysis system. This cycle continues until enough data is gathered to determine the validity of a particular scientific hypothesis or accomplish a high-level goal.

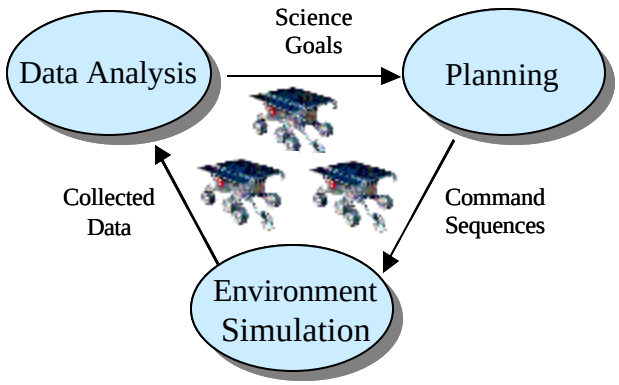

Figure 1: MISUS closed-loop data-flow

ASPEN Planning System

To produce individual rover plans for a team of rovers, we have adapted a version of the Automated Scheduling and Planning Environment (ASPEN) (Chien, et al., 2000). ASPEN is an object-oriented system that provides a reusable set of software components, which implement features commonly found in complex planning/scheduling systems. These features include a constraint management system for representing and maintaining temporal and resource constraints and a language for representing plan preferences and optimizing over these preferences.

ASPEN automatically generates the necessary activity sequence to achieve a set of input goals. One of the main algorithms used to produce this sequence is a local, early-commitment version of iterative repair (Minton and Johnston, 1988; Zweben et al., 1994), which classifies plan conflicts and attacks them individually. Conflicts occur when a plan constraint has been violated, where this constraint could be temporal or involve a resource, state or activity parameter. Conflicts are resolved by performing one or more schedule modifications such as moving, adding or deleting activities.

Planning for MISUS can be performed in either a centralized fashion, where one planner controls multiple rovers, or in a distributed fashion, where each rover has a separate onboard planner controlling its operations (Estlin, et al., 2000). In either case, goals are distributed among rovers in a way that should minimize overall distance traversed and power utilized. For this paper, we have tested our optimization approach utilizing the centralized planning option. In future work, the techniques presented in this paper will be migrated to operate in a distributed planning system.

Plan Optimization

ASPEN provides an optimization framework that allows the representation of continuous *soft constraints* (i.e., preferences) (Rabideau, et al., 2000). These constraints differ from the traditional *hard constraints* that must be satisfied for a plan to be considered correct and feasible. Soft constraints allow the quality of a plan to be represented and allow evaluation of a plan at a finer granularity than simply consistent or violated. Similar to the iterative-repair approach, a preference can be improved by making repair-like modifications to the plan.

In ASPEN, a preference is defined as a mapping from a plan variable to a quality metric (i.e., score). Plan variables include items such as local activity parameters, activity/goal count, resource/state variables, etc. Specifically, a preference indicates whether the score is monotonically increasing or decreasing with respect to the plan variable within certain bounds. Example preferences include "prefer linearly more observation occurrences", which encourages more observation activities to be performed, or "prefer exponentially more battery min value," which favors keeping the minimum battery charge at a high level. Each preference also includes a weight for specifying the relative importance of the preference to overall plan quality. The score of a plan is computed as the weighted average of scores for plan variables with preferences.

For each defined preference, an improvement expert automatically generates modifications that could potentially improve the preference score. Improvement experts are based on the class of preference and variable for which

they are constructed. An instance of an expert uses the preference specification to calculate plan modifications that will improve the score for the given preference and current plan. The set of modifications produced by the expert are not guaranteed to improve the overall plan score but instead define the search space of possible improvements, which the planner can consider.

The overall *iterative optimization* algorithm works by first selecting a preference to work on, and then deciding what type of plan modification should be performed to try and improve the selected preference. These modification options are often similar to the options considered by the iterative repair algorithm, and include adding, deleting, and moving activities as well as changing parameter values. Iterative optimization can also utilize iterative repair to find a feasible plan in situations where trying to improve upon a particular preference has created new plan conflicts.

While ASPEN's optimization framework enables the planner to generate high quality plans, in its current form it does not consider the value of interdependent goal combinations. In the following sections we will formalize what we mean by goal interdependencies and describe how we extended ASPEN to improve plan quality when goal interdependencies occur in problem specifications.

Interdependent Goals and Utilities

Historically in planning and scheduling systems, goal selection has been a linear process in which goals are independently selected and prioritized based on their expected reward. However, in some applications, this model is insufficient to correctly characterize the utility of a plan. For instance, in the case of performing science experiments in a new planetary terrain, goal priorities should be determined by the expected reduction in uncertainty associated with the current scientific model. There are many situations in this type of domain where the value of a science goal will be increased if other related science goals can also be achieved. For instance, collecting images of a particular rock from different angles and distances often increases the value of all images taken of that rock, since a better overall analysis of the rock can be done. Conversely, there are situations in which it is very important to achieve one of a set of goals, but having accomplished one in the set, the others become less important. For instance, we may have the case where we only want a rover to collect one or two more samples of a particular rock type but there are a large number of possible targets from where to collect such a sample. In this situation, we would like to direct the planner to collect a couple samples and then move on to other science experiments. If samples were collected at all target sites, this data would be overly redundant and somewhat lower the utility of the overall set since time had been wasted collecting unneeded data.

To represent a goal's value, we have extended a typical goal-utility representation (where goals can have individual rewards representing their importance) so that complex interdependencies and their relevant utilities can be represented and utilized by a planning system. Furthermore these interdependencies and utilities can change between problem specifications without requiring any changes to the planner domain model. In our representation, a list of goals and goal combinations are provided to the planner. A utility value is also assigned to each goal and to each specified goal combination. As an example, consider the spectral measurement and image goals shown in Table 1, which are from the previously introduced rover domain. Let's assume these goals are interdependent in several ways. First, Goals 1-3 are for spectrometer readings for the same type of rock and it has been deemed necessary to obtain only one such reading and any more would add little value to the current set of collected data. Second, Goals 4-7 are for the same rock or rock area and it has been determined desirable to obtain all of those observations. However, if only a few can be obtained that data would still be beneficial but not provide as much scientific value as the entire set.

These types of goal combinations are difficult to represent in standard planning-optimization approaches. As mentioned previously, a number of systems represent goal rewards in the form of utility functions or preferences, however, these approaches typically try to maximize a certain goal type or minimize usage of a certain resource. For instance, a utility function may try to minimize the amount of fuel used in transporting objects, or may try to maximize the number of factory orders that can be filled. This type of representation is limited in that it prefers to decrease or increase the number of goals or activities of a general type, where each goal or activity is viewed as relatively equal (or interchangeable). The goal inter-

Goal Num	Target Description	Location (x,y,z)	Reward
1	Spectrometer read for rock type x	(3.4, -34.6, 2.0)	10
2	Spectrometer read for rock type x	(162.3, 43.9, 1.1)	10
3	Spectrometer read for rock type x	(-4.1, 145.8, 0.4)	10
4	Spectrometer read for rock type y in area A	(104.3, -12.1, 1.5)	12
5	Soil sample from area A	(103.5, -13.4, 0.2)	15
6	Rock image for rock type y in area A	(104.3, -12.1, 1.5)	10
7	Dust collection experiment from area A	(105.1, -13.7, 1.5)	12

Table 1: Example sets of science goals given to planning system

dependencies required for deducing many scientific hypotheses are often much more complex since each individual goal may play a different role in the overall success of an experiment.

We can visually represent goal interdependencies between a set of two goals by using a graph structure where vertices represent individual goal rewards and edges represent inter-dependent goal rewards. For example, Figure 3 shows two goals that have individual rewards (rep-resented by G_1 and G_2) and a combined reward represented by R_{12}. There may also be dependencies between sets of three and four goals (i.e., you get reward R_{12} if goals G_1 and G_2 are achieved, you get reward R_{23} if goals G_2 and G_3 are achieved, and if all three goals are achieved, you also get reward R_{123}). In general, the graph may contain hyperedges linking several goals to their combined value. Table 2 shows interdependent goal rewards for the goals introduced in Table 1. Goal combinations for goals 1-3 are given slight negative rewards to show that achieving more than one goal in this set actually has less value than just achieving one. The goal combination for goals 4-7 shows that achieving all of the goals in that set has a large bonus reward.

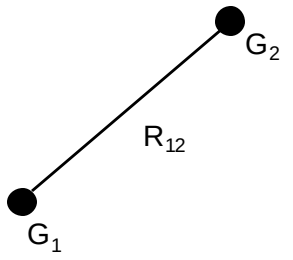

Figure 3: Two related goals

Goal Combination	Reward
<Goal 1, Goal 2>	-5
<Goal 1, Goal 3>	-5
<Goal 2, Goal 3>	-5
<Goal 4, Goal 5, Goal 6, Goal 7>	60

Table 2: Goal interdependencies and corresponding rewards

Plan Optimization for Interdependent Goals

We extended the ASPEN optimization system to support the inclusion of goal interdependencies with a planning problem description. The extension consists of two main components: an objective function to compute the value of the plan with respect to the goal interdependencies and an optimization framework for selecting goals to achieve and coordinating optimization with plan repair.

Objective Function

As is the case with most planners, the ASPEN problem specification includes a description of the goals that must be achieved to accomplish a particular problem. In addition, ASPEN can accept a set of optional goals that, while not required, will increase the quality of the plan as more of these goals are accomplished. This is useful when the planner is given more goals than are feasible to achieve given its resource constraints. In this case, ASPEN will use an objective function to try to find a subset of goals that result in a valid, high quality plan.

Our extended version of ASPEN also takes as input a set of goal interdependencies specified as a graph of goal nodes as described in the previous section. The graph consists of a set of vertices V where each vertex corresponds to a goal that can be added to the plan, including both mandatory and optional goals, and a set of edges E. Each edge consists of a tuple of vertices: $<v_1, v_2, \ldots v_n>$. For each vertex and each edge, there is an associated weight $w_{<v1, v2, \ldots vn>}$ indicating the value that will be added to the plan if the plan includes these goals. This representation allows us to express singleton goal values, that is a goal whose contribution to the plan does not change as other goals are added, and any n-ary goal relationship to indicate the value that combination of goals add to the plan.

We use a simple objective function to calculate the plan quality with respect to these optional goals. Let G be the set of goals that occur in the plan. The value of plan P is then given by Equation 1. This function sums up the values of all goals that occur in the plan along with the weight for each edge for which all of the edge's vertices occur in the plan.

$$O(P) = \sum_{v \in V} o(\prec v \succ) + \sum_{\prec v_1, v_2, \ldots, v_n \succ \in E} o(\prec v_1, v_2, \ldots, v_n \succ)$$

$$o(\prec v_1, v_2, \ldots, v_n \succ) = \begin{cases} w_{\prec v_1, v_2, \ldots, v_n \succ} \text{ if } \{v_1, v_2, \ldots, v_n\} \subseteq P \\ 0 \text{ otherwise} \end{cases}$$

Equation 1: Objective function for calculating plan utility when using interdependent goals

Please note, however, that although Equation 1 represents the quality of the plan with respect to interdependent goals, this is not the only type of preference a user might specify for a given problem. As described earlier, there is a large set of preferences that may be included, each with their own objective function. The overall score for a plan is computed by calculating the objective function for each preference and then performing a weighted sum of their scores.

Optimization Framework

The objective function above enables ASPEN to calculate the score of a plan. The next step is to provide an improvement expert that can suggest what changes ASPEN should make to the plan to increase this score. Clearly, the improvement expert for interdependent goals should suggest adding more optional goals to the plan. However, adding a goal will likely result in conflicts in the plan since it is usually necessary to add or modify other activities in the plan to achieve a goal. Therefore it is also necessary to

coordinate the process of improving the plan score with ASPEN's repair process to fix conflicts in plans.

Our current approach to performing optimization for interdependent goals is randomized hill-climbing with restart. We begin by first creating a plan that achieves all of the mandatory goals in the plan. We then perform a series of optimization steps where each optimization step consists of *i* iterations. At each iteration, if there are no conflicts in the plan, we use the improvement expert to suggest the next optional goal to add, and then add this goal to the plan. If there are conflicts, we perform an iteration of repair. Whenever we have a conflict free plan, if its score is the best we have seen, we record its point in the search space. At the end of the *i*th iteration, we return to point in the search space with the highest valued, conflict-free plan and begin the next optimization step. This approach protects against the possibility of adding a goal to the plan that cannot be solved. By always returning to a valid plan, we can restart the hill climbing process.

Our improvement expert takes a greedy approach to selecting the next goal to add. For each optional goal that is not currently in the plan, it computes the score that the plan *would* have if it were in the plan. It suggests adding the goal that improves the plan the most. However, the improvement expert must avoid the trap of continually suggesting the addition of a goal that cannot be achieved, even if achieving it would provide the best improvement to the score. Therefore, an element of randomness is included. The highest scoring goal is returned with probability 1 - ε, otherwise a goal is selected randomly with a probability of picking each goal relative to the amount that that goal improves the objective function.

Some care is needed in selecting the parameters for this algorithm. If it is usually possible to achieve any optional goal, then ε should be kept small, otherwise the performance of the optimizer will not do much better than one would do without the use of the goal interdependencies.

It is also important to adjust the number of iterations per optimization step. Because the planner restarts from a valid plan at the end of each optimization step, the number of iterations per step, *i*, should be large enough to provide enough iterations to enable the planner to perform the necessary repairs if the current set of goals can be achieved. However, if a valid plan cannot be produced, or the iterative repair algorithm gets the planner into a poor area of the search space, then a smaller *i* will mean that the planner will restart more quickly and save time. Like the selection of ε, this parameter has much to do with the difficulty of the domain. If goals can usually be solved with a small number of iterations, then a small *i* will be best. For our domain, we ran a number of randomly generated problems with different values of *i* to find a value that would allow the planner to effectively add goals to the plan and make repairs within each optimization step.

Evaluating ASPEN's Performance with Interdependent Goals

Our main concern in evaluating our system was to see whether or not explicitly taking into account goal interdependencies during optimization would significantly improve the quality of the plan. We expected to see some improvement over a system that did not use goal interdependencies in its objective function, but we were not sure if the improvement in quality would be worth a potential increase in time to produce the plans. We were also curious to see how much of an improvement would be provided by the relatively simple objective function described in the previous section.

Methodology

To evaluate the impact of reasoning about interdependent goals, we compared our extended version of ASPEN, which we will refer to as ASPEN+IDGS (for ASPEN with InterDependent Goal Support) to two other versions of ASPEN: ASPEN+Random and ASPEN+SimpleReward. All three versions used the randomized hill-climbing algorithm described in the previous section. The only difference is in how each of the three selects the next optional goal to add to the plan. ASPEN+IDGS uses the objective function from Equation 1 to pick the next goal. ASPEN+Random simply selects a goal at random without considering rewards. Finally, ASPEN+SimpleReward uses an objective function that looks at rewards for individual goals without considering goal interdependencies.

We ran each system on a set of randomly generated problems from a Mars exploration domain. In this domain, a team of three rovers must collect different types of science data, including images and spectrometer reads, at various locations on the planet's surface. The planner must decide which goals to assign to each rover, determine a sequence for each rover to use in visiting the different locations and plan for activities such as raising and lowering the rover masts and communicating with the ground to transmit data. Generated plans must also respect resource and temporal constraints, such as not exceeding onboard memory limitations when collecting data and performing activities during times when enough sunlight will be available.

The randomly generated problems varied in the number and location of the science goals. The science targets in the problem are contained within a 30m by 30m region of the planet's surface divided into 36 areas of 5m by 5m each. For this experiment, each problem consisted of 6 randomly selected areas from this set. For each selected area we generated up to 4 rocks of interest with random positions within the area. Table 3 shows the types of goals that are given to the planner along with the possible values for each individual goal. Note that some goals have a range of values in which case a specific value is drawn randomly from a uniform from this range. The goal types can be described as follows. A mandatory panoramic image must

Goal	Reward
A: Panoramic Image of an Area (Mandatory)	20
B: Long-Range Image of a Rock	12-25
C: Close-Up Image of a Rock	7-20
D: Close-Up Spectrometer Read of a Rock	2-15

Table 3: Individual goals and rewards

Goal Combination	Reward
<Goal B, Goal C>	(Value(B) + Value(C)) * 1.75
<Goal B, Goal D>	(Value(B) + Value(C)) * 2.25, 90% (Value(B) + Value(C)) * 10.0, 10%
<Goal C, Goal D>	(Value(C) + Value(D)) * 1.25

Table 4: Goal interdependencies and rewards

be taken from the center of each selected area. For each rock of interest, the rovers must take an image at a distance of 1m from the rock, to place the rock in context with its surroundings, a close up image and a close up spectrometer reading. Thus, there will always be 6 mandatory panoramic images in each problem. The number of optional goals will depend on the number of rocks that were generated. There will be three optional goals for each rock. With up to 4 rocks per area, this results in problems that can range in size from 6 goals (when 0 rocks are picked per area) to 78 goals (when 4 rocks are picked per area).

The rovers are given 1 Martian day to complete these goals. Depending on the relative locations of the targets, each rover can typically handle about 10 goals in this time. With three rovers this means that most of the problems will be too large to complete and the planner will have to take into account the different goal values to determine which goals should be achieved.

Each problem description also included a randomly generated set of goal interdependencies. Although the interdependencies were randomly generated, they were based on preferences derived from our conversations with planetary geologists and represent the type of utility values considered by human experts. Table 4 shows the goal combinations used for the experiment and the associated values, which are based on the values of the individual goals making up the combination. To increase the variance among goal combinations, we used two different factors for computing the value of the goal pair B and D. Most of the time the value of this combination is 2.25 times the sum of goals B and D's individual values. However, sometimes a much higher value (10 times the sum of their individual values) is given. Finally, for a given rock, each of the three goal combinations is removed with probability 0.5.

In selecting parameters for the randomized hill-climbing algorithm used in each planner, we decided to use 50 iterations per optimization step as it seemed to provide the best balance between allowing the planner enough time to repair goals but not so long that it would waste a lot of time if it got stuck and needed to back up to a previous plan. For ε, we selected a small value of 0.02.

Results

We generated a set of 30 problems and because there is an element of randomness both to the ASPEN iterative repair algorithm and to our optimization approach, we ran the three versions of ASPEN on each problem 5 times. The systems were run on a Sun Blade 1000 with 1 Gigabyte of RAM.

At the end of each optimization step we recorded the current plan score based on the objective function from Equation 1, the current number of goals in the plan, and the number of seconds spent during that step. Note that even though the ASPEN+Random and ASPEN+SimpleReward versions of the planner did not make use of the objective function to select goals to add, we still used that objective function to score their plans for the purpose of the experiment.

Figures 4-6 present the results from these runs. Objective function scores are compared in Figure 4, while Figures 5 and 6 compare the total number of goals achieved and the planning time used by each method. The data points in each graph are averaged over the 150 runs from each system. In each graph, the data point at optimization step 0 represents the planner performing repair on a plan containing all mandatory goals. We performed two-tailed t tests between each pair of the three systems with a Bonferroni correction. The only graph that showed significant differences among the systems was the graph of plan scores in Figure 4. ASPEN+IDGS was found to be significantly better than both ASPEN+Random and ASPEN+SimpleReward at the 0.01 confidence level. ASPEN+Random outperformed ASPEN+SimpleReward but only the data points between optimization steps 6 and 14 showed significant difference at confidence level 0.01.

Discussion

Figure 4 shows that ASPEN+IDGS outscores both ASPEN+Random and ASPEN+SimpleReward. In fact, ASPEN+IDGS showed a significant improvement over both versions at each data point. The plot of the number of goals included in each plan (Figure 5) shows that all three systems were achieving about the same number of goals. This means that ASPEN+IDGS was selecting higher quality goals. This factor is particularly important because none of the planners were able to achieve all of the goals thus it is better to achieve the higher quality subset.

It is also important to note that ASPEN+IDGS's biggest improvements in performance occur in the early optimization steps. Thus, even if the planner is capable of solving all the goals it is given but it is under tight time constraints, then using ASPEN+IDGS will allow the planner to find a much higher quality set of goals. This feature is especially important in real-world problems where planning time can be tightly bound.

The shapes of the curves reveal some interesting characteristics about each algorithm. The curve for ASPEN+IDGS rises sharply in the early optimization steps

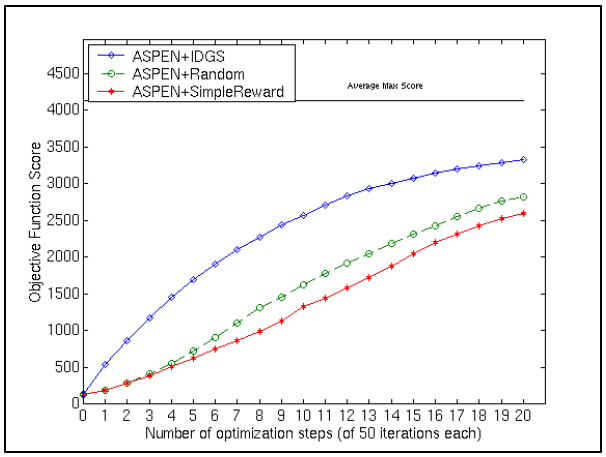

Figure 4: Objective function score

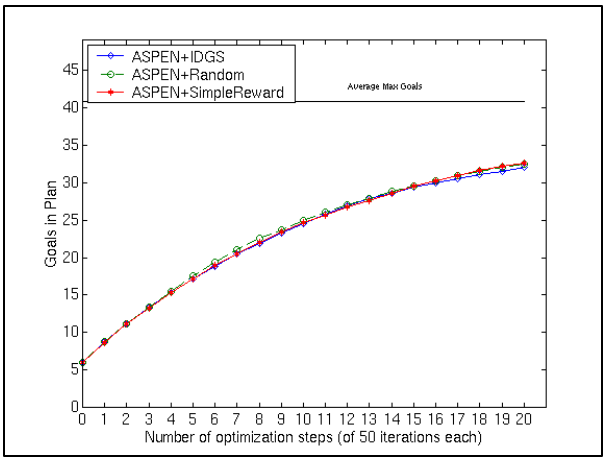

Figure 5: Number of goals achieved

and then tapers off, while ASPEN+Random starts rising more slowly, increases in its rate of growth, and then begins to taper off at the end. Given that both planners were adding about the same number of goals to the plan at each time step, the differences in the curve shapes is a result of the way each algorithm selected goals. The sharp rise in the ASPEN+IDGS curve can be explained by the fact that ASPEN+IDGS is explicitly looking to add goals that will improve the objective function. However, as more goals are added to the plan, and therefore the rovers' resources are beginning to be stretched to their limit, making repairs to the plan becomes more difficult and the planner spends more iterations fixing problems with the plan and fewer iterations adding goals. As a result, the curve begins to level off. As can be seen in Figure 5, the number of goals added to the plan at each optimization step begins to decrease at about the same time that ASPEN+IDGS's score begins to taper off in Figure 4.

In contrast, the ASPEN+Random curve in Figure 4 begins slowly because it is randomly adding goals to the plan and, early on, it is unlikely that the interdependent goal combinations will be satisfied in the plan. However, as more goals are added, the probability of satisfying goal combinations when a new goal is added increases, and the score begins to rise more rapidly. But, just like ASPEN+IDGS, the planner begins to spend more time performing repairs and fewer goals are added to the plan causing the curve to taper off.

The fact that ASPEN+SimpleReward was the worst performer is particularly interesting. Recall that this version of the system is selecting new goals based on the each goals individual contribution to the plan. In other words, it is using the rewards from Table 3. Therefore, the planner will favor the addition of long-range images and avoid adding close-up spectrometer reads. The problem with this approach is that the goal interdependencies do not necessarily preserve the relative reward values of the individual goals. For example, although the close-up spectrometer read is the lowest rank score individually, when it is combined with a long-range image, it becomes much more valuable. However, since ASPEN+Simple-Reward typically avoids adding this goal to the plan, it does not satisfy these high-quality goal combinations. As a result, its score grows slowly and, like the other curve, tapers off in later optimization steps.

Figures 4 and 5 show that ASPEN+IDGS provides considerable benefit when the planner cannot achieve all the goals in a plan. In this case, ASPEN+IDGS selects a higher quality subset of goals than either of the two competing systems in this study. This is already advantageous, but we were also interested in whether or not ASPEN+IDGS could increase plan quality without a significant increase in planning time. The plot of each system's processing time per optimization step in Figure 6 shows ASPEN+IDGS did not significantly increase planning time.

These results show that ASPEN+IDGS provides a significant improvement in plan score over versions of the planner that do not consider goal interdependencies without a significant increase in planning time. This benefit is most important when a planner is given more goals than it can achieve as well as when the planner is under time constraints and may not have enough time to plan for all of its goals.

Related Work

The ASPEN methodology for optimizing over generic preferences (Rabideau, et al., 2000) was described in a previous section of this paper. The new optimization approach presented in this paper is an extension to this methodology that enables goal interdependencies and relevant utilities to be represented and utilized in a problem specification. The original ASPEN optimization approach does allow for more generic preferences than other methods that focus on specific preference types, such as trying to achieve goals early in time or conserving resources. However it still assumes that only static goal relationships

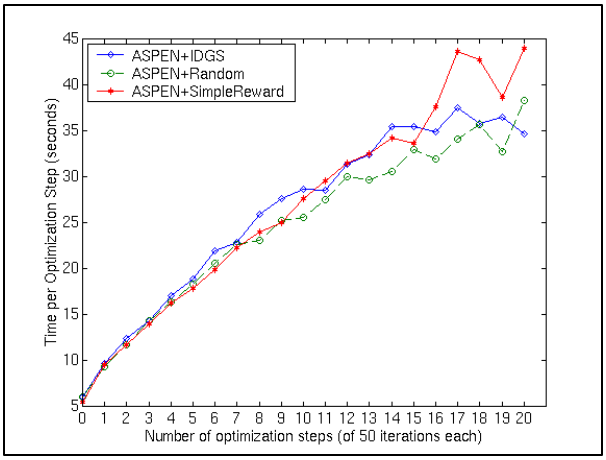

Figure 6: Plan generation time

can be defined (as part of a domain model) and these relationships are typically not tied to any utility measure. The standard ASPEN preference language does not allow for goal dependencies and related utilities to change between different problems. Instead it concentrates on defining preferences over general classes of goals as opposed to individual goals or goal subsets.

Other work in planning optimization has used utility models to improve on particular types of quality measures. PYRRHUS (Williamson and Hanks, 1994) extends the UCPOP partial-order planner to handle metric time, resources, and a utility model. However, PYRRHUS has typically been applied to domains where goals are related through simple, static model definitions (e.g., two transport goals are related since they both require fuel). Conversely, we are examining domains and problems where goal utility can be affected by achieving certain goal combinations and this utility is defined by the problem instance. Furthermore this utility can change from problem to problem.

Over the last several years many planning researchers have explored the use of decision theory in planning. In particular, Markov Decision Processes (MDPs) provide a rich formalism for expressing many types of planning domains (Boutilier, et al, 1999). It should be possible to encode the types of goal interdependencies discussed in this paper in an MDP reward function. Standard techniques for solving MDPs such as Policy or Value Iteration could be used to find a plan that maximizes this reward. However, MDPs have yet to be demonstrated on real problems of significant size in domains with time and resource constraints and it is likely that the large computational cost would be prohibitive. Instead, ASPEN uses a randomized iterative repair algorithm that does not guarantee optimality, but, as we have demonstrated in the paper, is capable of finding high quality plans.

Past work in planning systems has been directed at improving the overall quality of a plan by using machine-learning techniques to acquire relevant search-control knowledge. Plan quality in these systems is defined as either plan execution cost (Perez and Carbonell, 1994) or some other specific cost measure applied to the final plan (Iwamoto, 1994). However, such methods have only been applied in STRIPS-style planning domains that do not have the complex resource or temporal constraints common to real-world domains. Furthermore, goal interactions or dependencies all stem from the domain model and cannot be introduced as part of problem definitions.

Work in mixed-initiative planning has also looked at generating qualitatively different plans by developing biases that focus the planner towards solutions with certain characteristics (Myers and Lee, 1999). Our work however, has been more focused on automated planning where the planner operates with little human involvement. Using our methodology, however, a user could specify utility preferences in a problem specification by encouraging certain goal combinations.

Future Work

We have identified a number of areas for further investigation. First, we would like to test our approach on different types of goal interdependencies. Our experiments thus far have only considered pairs of related goals where each selected pair increases the overall utility of a plan. As mentioned previously, goal interdependencies can be defined in different manners. For instance, certain goal combinations may actually decrease overall plan utility. We would also like to experiment with problem sets where only so many goals of a certain set are added before utility gains level off or actually start to decrease. Furthermore, we would like to explore using this method in other domains where goal interdependencies can affect plan utility.

In addition, we plan to test whether our simple objective function and approach scales up to goal combinations using much larger sets of goals. Currently we have only looked at goal pairs, but goal relations can obviously span much larger sets of goals. In experiments discussed in this paper, problems could contain over 70 goals and there are many larger goal combinations that could be identified and used to further define problem utility. To better address the scalability issue, we also plan to enhance our current optimization algorithm to better recognize potential high-utility goal combinations. Currently we use a simple hill-climbing approach that only considers one goal at a time. Incorporating some degree of look-ahead may enable the algorithm to catch even more goal combinations that will further increase plan utility.

Finally, we intend to further test our approach using the entire MISUS system and evaluate its capability to collect and analyze valuable science data and to eventually determine the validity of different scientific hypotheses. Though currently this system is operated only in simulation, we intend to ultimately test its capabilities using real rovers examining actual terrain features.

Conclusions

In this paper we have presented a method for utilizing interdependent goal utilities, where goal relations can be dictated by current information and can vary from problem to problem. In typical planning systems, only simple, static goal relations can be defined that remain relatively constant between problem instances. However, in many application areas, goal dependencies and their related utility metrics can dramatically change based on current information or even user preferences. To address this problem, we have implemented a new method for representing and reasoning about interdependent goals. We have also presented experimental results that show how this approach can significantly improve overall plan quality in a multi-rover application.

Acknowledgments

This work was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

The authors acknowledge the contributions of Eric Mjolsness, Rebecca Castano, Ashley Davies, and Martha Gilmore for their help in defining relevant geology scenarios for this work. We also thank Gregg Rabideau for providing the ASPEN optimization framework and for his help in implementing our approach.

References

Boutilier, C., Dean, T. and Hanks, S. 1999. Decision-Theoretic Planning: Structural Assumptions and Computational Leverage. *Journal of Artificial Intelligence Research*, 11:1-94.

Chien, S., Rabideau, G., Knight, R., Sherwood, R., Engelhardt, B., Mutz, D., Estlin, T., Smith, B., Fisher, F., Barrett, T., Stebbins, G., and Tran, D. 2000. ASPEN - Automating Space Mission Operations using Automated Planning and Scheduling, In *Proceedings of the SpaceOps 2000 Conference*, Toulouse, France.

Estlin, T., Gray, A., Mann, T., Rabideau, G., Castano, R., Chien, S. and Mjolsness, E. 1999. An Integrated System for Multi-Rover Scientific Exploration. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 613-620. Orlando, FL.

Estlin, T., Rabideau, G., Mutz, D., and Chien, S. 2000. Using Continuous Planning Techniques to Coordinate Multiple Rovers. *Electronic Transactions on Artificial Intelligence* 4:45-57.

Iwamoto, M. 1994. A Planner with Quality Goal and Its Speed-up Learning for Optimization Problem. In *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems*, Chicago, IL.

Joslin, D., and Clements., D. 1999. "Squeaky Wheel" Optimization. *Journal of Artificial Intelligence Research* 10:353-373.

Minton, S., and Johnston, M. 1988. Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems." *Artificial Intelligence*, 58:161-205.

Myers, K., and Lee, T. 1999. Generating Qualitatively Different Plans Through Metatheoretic Biases. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, Orlando, FL.

Perez, A., and Carbonell, J. 1994. Control Knowledge to Improve Plan Quality. In *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems*, Chicago, IL.

Rabideau, G., Engelhardt, B., and Chien, S. 2000. Using Generic Preferences to Incrementally Improve Plan Quality. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*, Breckenridge, CO.

Williamson, M., and Hanks, S. 1994. Optimal Planning with a Goal-Directed Utility Model. In *Proceedings of the Second International Conference on Artificial Intelligence Planning Systems*, Chicago, IL.

Zweben, M., Daun, B., Davis, E., and Deale, M. 1994. Scheduling and Rescheduling with Iterative Repair, In *Intelligent Scheduling*, Morgan Kaufmann, San Francisco, CA. 241-256.