Explanations and Recommendations for Temporal Inconsistencies

John L. Bresina and **Paul H. Morris**
NASA Ames Research Center
M/S 269-2, Moffett Field, CA 94043
John.L.Bresina@nasa.gov and Paul.H.Morris@nasa.gov

Abstract

This paper presents an approach to automatically providing explanations and recommendations for temporal inconsistencies within the context of constraint-based, mixed-initiative planning. In this style of planning, a user operation can cause the planner's temporal constraint network to become inconsistent. For example, if a partial plan has been built and the user is trying to add another activity into the plan, an inconsistency will arise if the activity's associated constraints conflict with the current plan's constraints. In such cases, it is helpful if the system can explain to the user why the user's operation led to the inconsistency, and it is even better if the system can give the user recommendations on how to fix the inconsistency. Our approach to explanation and recommendation is generally applicable to constraint-based reasoning systems; however, the motivation and validation of our approach derive from a deployed planning system for planetary rover missions.

Introduction

The context of this paper is constraint-based, mixed-initiative planning. Inherent in constraint-based planning is the issue of handling inconsistencies that arise in the constraint network underlying the plan. Inconsistencies can be described by a (minimal) *nogood*, which identifies a (minimal) subset of the constraint network that is contributing to the problem. Inherent in mixed-initiative systems is collaborative problem solving involving humans and machines. In this paper, we address the general issue of collaboratively resolving temporal inconsistencies that arise during the planning process.

We present a general approach to summarizing the essential aspects of nogoods to improve human understanding of temporal inconsistencies. In addition, we present an approach to generating recommendations on how the user can resolve the inconsistency. A successful recommendation restores consistency to the temporal network while achieving what the user was trying to do when the inconsistency arose.

Our motivation derives from the problem of activity planning for the Mars Exploration Rover (MER) mission and the system used to accomplish this task: MAPGEN, Mixed-initiative Activity Plan GENerator (Bresina, et al., 2005a). Within the activity planning process, the role of the TAP is to direct construction of the plan and fine-tune it by bringing to bear expertise that is outside MAPGEN's domain model and beyond its scope of reasoning, such as optimal use of the battery and scientist preferences. The system makes automated planning capabilities available to the user and performs potentially tedious tasks. The planning process is an incremental one in which the user interleaves automatic plan generation and plan editing.

The MER mission has been in operation for over a Martian year (more than two Earth years) and MAPGEN is still being employed for activity plan generation for the Spirit and Opportunity rovers. One of the key lessons learned from this experience (Bresina, et al., 2005b) is the need for the automated reasoning component to provide better explanations of its behavior. Especially important is explanation of failure, such as why an activity cannot fit in the plan at a particular time. Our explanation approach addresses this need; in addition, our recommendation approach advises the user on how to work around the failure in order to achieve the user's original intent.

As a starting point for our research, we have used a standalone version of the MAPGEN system. Furthermore, we have modified the MER planning domain model to drastically reduce the number of abstraction levels, where we determined that certain expansions were not important for the planning process. (They were originally intended to facilitate post-planning sequencing.) In fact, many of the activities now are not expanded at all. Nevertheless, the examples in this paper are based on real problem instances from the MER mission that have been translated into our modified domain.

In the rest of this paper, we first present further background information on the context of this work, primarily regarding the mixed-initiative, constraint-based planning approach employed in MAPGEN for MER mission operations. We then give more details about, and examples of, the inconsistencies that can arise and present a unifying characterization of these inconsistencies. This characterization forms the basis of the explanation approach, which is described next. This is followed by a description of how recommendations are generated and presented to the user. We then briefly describe some related research and conclude with some remarks on our contributions, current status, and planned next steps.

Background

In the MER mission, the daily commanding cycle proceeds as follows. The engineering and science data from the previous Martian day (sol) are analyzed to determine the status of the rover and its surroundings. Based on this, and on a strategic longer-term plan, the scientists determine a

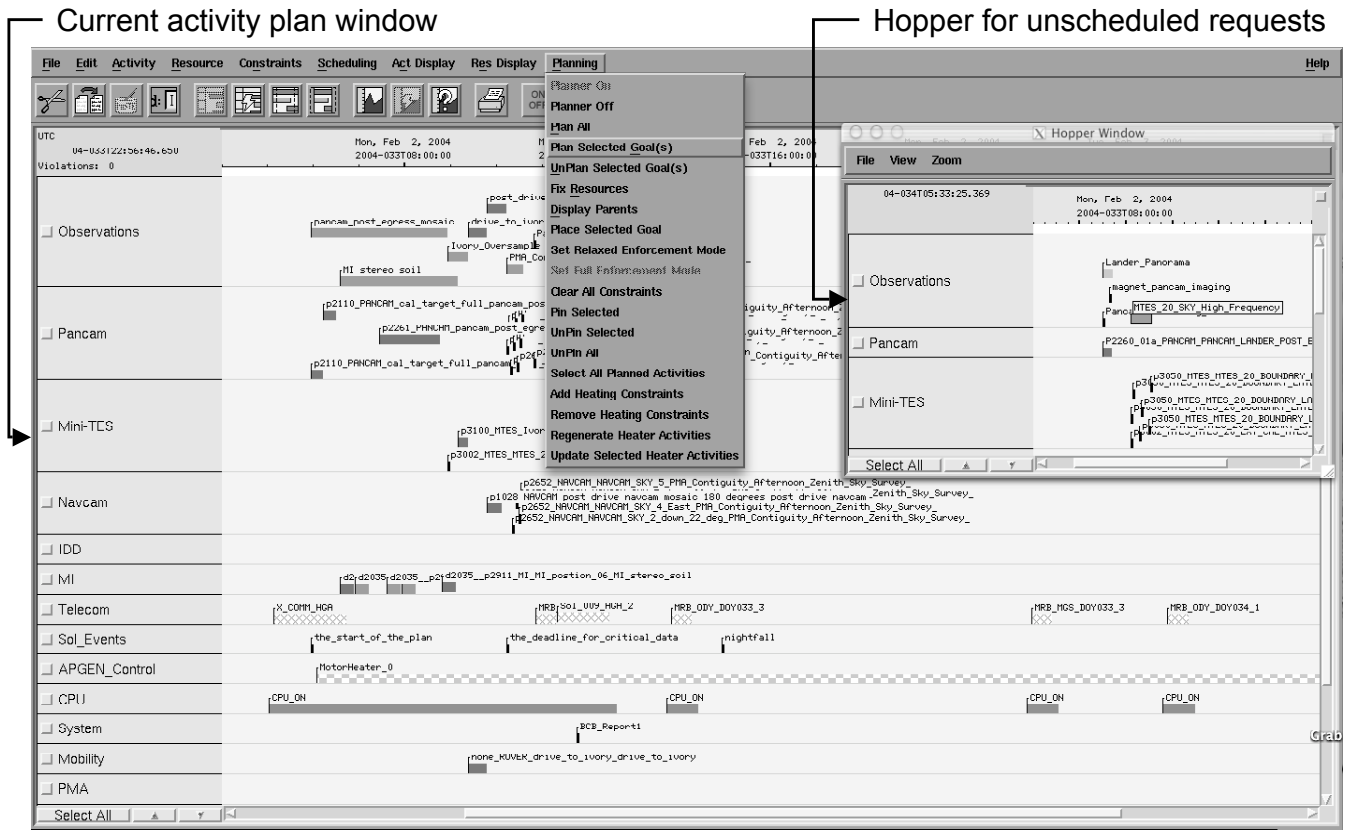

Figure 1: MAPGEN interface with planning menu and hopper.

set of scientific objectives for the next sol. At this stage only rough resource guidance is available. Hence, the scientists are encouraged to oversubscribe to ensure that the rover's resources will be fully utilized in the final plan.

In the next step in the commanding process, the science observation requests are merged with the engineering requirements (e.g., testing the thermal profile of an actuator heater) and a detailed plan and schedule of activities is constructed for the upcoming sol. The plan must obey all applicable *flight rules* that specify how to safely operate the rover and its instrument suite and remain within specified resource limitations. It is in this step that the Tactical Activity Planner (TAP) employs MAPGEN.

Once approved, the activity plan is used as the basis to create sequences of low-level commands, which drive onboard execution. This sequence structure is then validated, packaged, and communicated to the rover. This completes the commanding cycle.

The MAPGEN system (Figure 1) is the result of integrating two legacy systems: APGEN, Activity Plan GENerator (Maldague, et al., 1998), and EUROPA, Extendable Uniform Remote Operations Planning Architecture (Frank & Jónsson, 2003). APGEN is an interactive activity plan editor developed by Jet Propulsion Laboratories and used as MAPGEN's graphical user interface. EUROPA is a constraint-based planning framework, developed at NASA Ames Research Center and used to provide MAPGEN's plan representation and reasoning capabilities. A joint JPL-Ames team developed and deployed MAPGEN.

In constraint-based planning, domain rules are specified in terms of activity/state patterns and constraint schemas. A given constraint schema is applied to any instance matching the associated pattern. The EUROPA framework performs sound constraint reasoning and provides a mechanism for firing applicable domain rules. Search methods and other techniques for manipulating partial plans then build on this framework. Partial plans have an underlying simple temporal constraint network, or STN, (Dechter, Meiri, & Pearl, 1991). The consistency of STNs can be determined by checking for arc consistency. Furthermore, each value in an arc-consistent temporal variable domain appears in at least one legal solution of the STN. The set of such values defines a temporal interval that can be represented by its bounds.

It is not necessary to immediately ground the variables. Plans with temporal variables left ungrounded are called *flexible plans*; these are the type of plans that MAPGEN constructs. However, the plan that is displayed to the user is a *grounded plan*; i.e., a specific instantiation of the underlying flexible plan. This is chosen to be close to a *reference schedule*, according to the following solution grounding algorithm.

For each timepoint x with reference position t do:
(*i*) If t is within the STN bounds for x,
then add a grounding constraint that sets x to t.
Else if t is less than the lower bound (lb) for x,
then add a grounding constraint that sets x to lb.
Else if t is greater than the upper bound (ub) for x,
then add a grounding constraint that sets x to ub.
(*ii*) Propagate the effect of the new constraint.

The reference schedule is initially based on the science constraints, which are set by the scientists, and the nominal start times of the activities (which can be somewhat arbitrary), and thereafter set to the previous grounded plan (Bresina et al., 2005a). This initial reference is computed by first solving a relaxed version of the planning problem composed of only the science constraints (i.e., it does not include flight rules); the solution produced is a flexible plan. The reference schedule is then determined by grounding this flexible plan, by the above algorithm, to be close to the nominal activity start times.

Explicit temporal constraints fall into three main categories: *model* constraints, *problem-specific* constraints, and *expedient* constraints. The model constraints encompass definitional constraints and mutual-exclusion flight rules. For example, "you cannot move the arm while the rover is moving" or "you cannot have more than one activity trying to simultaneously point the rover's mast".

The problem-specific constraints comprise relations between specific activities in a planning problem instance. In MER, the problem-specific constraints were either science constraints or engineering constraints. The role of the *science constraints* is to ensure that science objectives are satisfied and that the data collected is scientifically useful. The scientists specify two types of constraints: temporal bounds and temporal ordering relations. Temporal bounds typically constrain when an activity can start due to, for example, lighting conditions or temperature. Typical ordering relations are constraints between the end of one activity and the start of another; for example, activity B must immediately follow activity A, or there must be a gap of at least 30 minutes between activities A and B. Temporal bounds are represented in the planner's constraint network as a temporal relation between a start (or end) point of an activity and a distinguished time point called the *origin*.

The engineering constraints are used by TAPs to pin down certain activities in the plan so they could not be moved. Typically, the pinned activities were engineering activities, such as communication activities, but TAPs also pin science activities as part of their plan fine-tuning process. We refer to this type of problem-specific constraints as *pin constraints*.

Expedient constraints are added during search in automated planning. These constraints result from decisions made to guarantee compliance with higher-level constraints that cannot be directly expressed in an STN. For example, a model constraint might specify that two activities, A and B, are mutually exclusive. This is really the disjunctive constraint that either A must precede B or B must precede A. The choice between the two is made based on heuristics, and is subject to backtracking during the original search.

Planning with MAPGEN

The Tactical Activity Planner employs MAPGEN to collaboratively plan the activities of each rover, with the objective of achieving as much science as possible while ensuring rover safety and keeping within the limitations of the rover's resources. The intended collaboration is that the system handles constraint enforcement constantly in the background, in response to user invocation of a variety of plan editing and automated construction tools. The planning process is an incremental one in which the TAP interleaves automatic plan generation and plan editing phases. Each planning operation is done in the context of the current partial plan and its constraints; thus, previous planner decisions affect future planning operations.

One of the key design characteristics of MAPGEN is *user-adjustable autonomy*: MAPGEN provides a spectrum of automated planning services with different degrees of automation and human guidance. The science requests that are not currently in the plan are kept in a separate display window, called the *hopper*. Due to the incremental planning approach, the TAPs often apply the *plan-selected* operation. With this operation, the user selects a set of observation requests not in the plan, and requests that these be inserted anywhere into the current partial plan, such that all constraints are satisfied. The user can exercise even more control over the planning process via the *place-selected* operation, which is applicable only to individual activities. This operation allows the user to select an activity in the hopper and then choose an approximate temporal placement for it in the plan. The planning algorithm treats the user-chosen time as heuristic guidance and searches for a plan where the selected activity is as close to the desired time as possible.

The system also supports an activity movement operation, called *constrained-move*, which takes advantage of the flexibility in MAPGEN's plans. As long as an activity is moved only within the flexibility range defined by the domain in the underlying arc-consistent flexible plan, the result is necessarily another consistent instantiation. During a constrained move, the system actively restricts the movements of the selected activity to stay within the permitted range. Then, once the user places the activity, any dependent activity is automatically updated as necessary to yield a new valid plan instance. Note that the enforcement takes into account *all* the constraints, including expedient constraints that arbitrarily order mutually exclusive activities.

When extending the plan, the planner tries to minimally perturb the previous plan, i.e., move activities as little as possible from their previous positions. Each application of a planning operation is done in the context of the current plan and its constraints; thus, previous planning decisions

affect what future operations are possible and what additional activities can be fit into the plan. This incremental commitment approach helps achieve a fast response time; however, it can lead to planning failures due to inconsistencies resulting from prior commitments.

Mixed-initiative planning systems must respond and return control quickly to the user. For an automated planning operation, which involves a cascading decision process, MAPGEN relaxes completeness in favor of responsiveness. This has to be done carefully to maximize chances of finding near-optimal solutions within limited time. We developed a backtracking algorithm that noted the difficulty of planning activities, and when the effort to plan an activity exceeds an allowance determined by its priority, the activity is rejected from the plan.

Temporal Inconsistencies

In the MER application, all the constraints, and, hence, all the inconsistencies, are temporal. Temporal inconsistencies in an STN are much easier to handle than the more general case of inconsistencies that arise in Constraint Satisfaction Problem (CSP) solvers. Extracting a nogood from an STN is of polynomial complexity, whereas in the general case, it is an NP-Hard problem. Furthermore, the nogoods extracted from an STN have a simple structure that can be exploited in the explanation and recommendation process.

There are several contexts in which inconsistencies can arise during planning. First, when an activity is considered for insertion, it may be inconsistent with the current plan even before any location is examined. When MAPGEN attempts to insert an additional activity into the plan, it first starts enforcing the constraints associated with that activity. Since the existing plan was formulated without those constraints, they may be inconsistent with previous ordering decisions made to prevent forbidden overlaps. Second, the activity may be inconsistent with the *specific* location chosen in a place-selected operation. Third, it may be inconsistent with *each one* of the possible locations identified during a plan-selected operation.

The first context generates an immediate nogood in the underlying STN as soon as the constraints associated with the new activity are added. For example, the added constraints might require two activities in the existing plan to be in a different order. In the second context, a nogood may be extracted from the modified STN obtained by temporarily placing the activity in the infeasible location.

The third context gives rise to separate nogoods in every one of the STN extensions arising from the different location choices. This can happen even though there is no immediate nogood; for example, the new activity may simply not fit in the amount of space available in each of the separate locations, although the region containing all the locations is large enough. However, it may be possible to combine the separate nogoods to form a compound nogood that includes the disjunctive choice.

This issue arises because we are using an STN, not a Disjunctive Temporal Network (DTN) (Tsamardinos & Pollack, 2003), to test consistency. There is no immediate STN nogood. We can simulate the process of getting a DTN nogood by trying out the different placements. Each of the choices may give rise to an STN nogood, and their combination is, in effect, a DTN nogood. We also note that the second and third contexts have even more complex variations where the placements are ruled out after extensive backtracking involving subgoals (and possibly other sibling goals) that fail as a consequence of those placements. These cases stretch beyond even the DTN framework. In this paper we focus on the first context, which seemed to be a common source of TAP puzzlement. Moreover, within this simpler context, we address a core set of issues also inherent in the other two.

When an inconsistency occurs, MAPGEN does extract a minimal nogood; however, the nogood typically involves complex chains of activities and constraints that could not easily be grasped by the user. For example, during MER, nogoods encountered during planning could involve hundreds of constraints. It is obviously impractical to expect a time-pressured TAP to read, let alone grasp the significance of, a lengthy nogood. Thus, our explanation technique attempts to summarize the essential aspects of nogoods for presentation to the user.

A *temporal nogood* can be defined as a cycle of time points such that the sum of upper bounds on the links between adjacent time points is negative (Dechter, Meiri, & Pearl, 1991). Since an upper bound of x between A and B is equivalent to a lower bound of $-x$ between B and A, an equivalent, complementary definition is as a cycle such that the sum of *lower* bounds is *positive*. This latter view turns out to be more useful for our application because most of the constraints that are active in inconsistencies are more naturally viewed as lower bounds. Additionally, this view yields an intuitive "no room" explanation for why activities cannot fit in the plan.

A time point represents a start or an end of an activity, the origin, or a *sol event*. A sol event is a distinguished absolute time point; for example, the start or end time of the plan. The first step in the summarization process is to categorize each active constraint in a nogood based on its source; e.g., whether it is an expedient constraint imposed by the planner, a science constraint imposed by the scientists, an expansion constraint arising from the model, or a pin constraint imposed by the TAP.

Part of the reason the nogoods from MER are so lengthy was because the durations of high-level activities are derived from numerous low-level constraints that expand them into more primitive sub-activities with fixed durations. If a high-level duration is active in an inconsistency, then the nogood involves all these low-level constraints. Similarly, the mutual exclusion rules are implemented by means of a hierarchy of planner subgoals that are ordered on timelines, and the low-level constraints between such subgoals are also included in the nogoods. This is illustrated in Figure 2, which shows the low-level

Origin [181196592]→ Plan_Start	(Pin)
Plan_Start [0]→ Start APXS_1	(Science)
Start APXS_1 [0]→ Start ARM_MOVE_1	(Expand)
Start ARM_MOVE_1 [100]→ End ARM_MOVE_1	(Dur)
End ARM_MOVE_1 [0]→ Start APXS_ON_1	(Expand)
Start APXS_ON_1 [480]→ End APXS_ON_1	(Dur)
End APXS_ON_1 [0]→ Start APXS_ACQ_1	(Expand)
Start APXS_ACQ_1 [28800]→ End APXS_ACQ_1	(Dur)
End APXS_ACQ_1 [0]→ Start APXS_OFF_1	(Expand)
Start APXS_OFF_1 [60]→ End APXS_OFF_1	(Dur)
End APXS_OFF_1 [1719]→ End APXS_1	(Expand)
End APXS_1 [0]→ Start APXS_2	(Science)
Start APXS_2 [0]→ Start ARM_MOVE_2	(Expand)
Start ARM_MOVE_2 [100]→ End ARM_MOVE_2	(Dur)
End ARM_MOVE_2 [0]→ Start APXS_ON_2	(Expand)
Start APXS_ON_2 [480]→ End APXS_ON_2	(Dur)
End APXS_ON_2 [0]→ Start APXS_ACQ_2	(Expand)
Start APXS_ACQ_2 [28800]→ End APXS_ACQ_2	(Dur)
End APXS_ACQ_2 [0]→ Start APXS_OFF_2	(Expand)
Start APXS_OFF_2 [60]→ End APXS_OFF_2	(Dur)
End APXS_OFF_2 [0]→ End APXS_2	(Expand)
End APXS_2 [0]→ Start MB	(Science)
Start MB [0]→ Start ARM_MOVE_3	(Expand)
Start ARM_MOVE_3 [100]→ End ARM_MOVE_3	(Dur)
End ARM_MOVE_3 [0]→ Start MB_ON	(Expand)
Start MB_ON [120]→ End MB_ON	(Dur)
End MB_ON [420]→ End UHF_MUTEX	(Planner)
End UHF_MUTEX [0]→ Start UHF_MUTEX	(Planner)
Start UHF_MUTEX [0]→ Start UHF	(Planner)
Origin [-181241466]→ Start UHF	(Pin)

Figure 2: Example nogood after step 1.

edges (i.e., active constraints). Here, [x]→ indicates an edge with a lower bound of x, and the constraint edges are labeled as follows: expedient (Planner), problem-specific (Science), activity expansion (Expand), duration (Dur), and pinned activity (Pin). APXS and MB refer to different spectrometers on the rover's arm, and UHF refers to communication with an orbiter.

The second step in our inconsistency summarization process compresses the nogood by combining low-level sub-chains in the cycle that have the same source. For example, the Start APXS_1 to End APXS_1 sub-sequence above, which arises from the expansion of the APXS_1 activity, contains nine edges and is reduced to a single edge of length 0 + 100 + 0 + 480 + 0 + 28800 + 0 + 60 + 1719 = 31159.

For raw MER nogoods, the compression step typically achieves close to an order of magnitude reduction in size. Compression does not achieve as large a reduction for our modified model because the expansions have been significantly flattened, but it is, nevertheless, a useful step. For the example we consider to illustrate our approach, the original MER nogood contains forty-nine edges. In our modified model, it contains thirty edges, shown above. The nogood compresses down to nine edges, shown in Figure 3.

Origin [181196592]→ Plan_Start	(Pin)
Plan_Start [0]→ Start APXS_1	(Science)
Start APXS_1 [31159]→ End APXS_1	(Expand)
End APXS_1 [0]→ Start APXS_2	(Science)
Start APXS_2 [29440]→ End APXS_2	(Expand)
End APXS_2 [0]→ Start MB	(Science)
Start MB [220]→ End MB_ON	(Expand)
End MB_ON [420]→ Start UHF	(Planner)
Start UHF [-181241466]→ Origin	(Pin)

Figure 3: Example compressed nogood.

Note that the final edge of this compressed nogood has a negative lower bound; this is equivalent to a positive upper bound in the opposite direction, i.e., from the origin to the start of UHF. Observe also that the nogood tends to alternate between Start/End activity-expansion constraints and End/Start activity-separation constraints, as is typical.

This nogood arose when the TAP tried to bring APXS_1 into the plan. Its science constraints conflicted with constraints associated with the other activities in the plan. We note that a previous planner decision ordered MB_ON (which is a sub-activity of the high-level MB) before the UHF communication, because they cannot overlap. This has the effect that there is not enough room for the APXS_1, which must come before the sequence of APXS_2 followed by MB.

Explanations

Given the types of temporal constraints, there are many types of nogoods that could theoretically arise; Figure 4 shows six abstract examples, which are illustrative of the diversity of cases. In these examples, the TAP is trying to insert activity B into the plan, which causes the nogood; we refer to B as the goal activity. Activity B is constrained to be placed "between" A and C; more precisely, start(B) is constrained by end(A) and end(B) is constrained by start(C). Both A and C are activities, except in case 4 where A is a sol event. Thick vertical lines indicate the temporal boundaries of the nogood. The arrows represent edges in the nogood. The dotted arrows represent an edge-pair through the origin. There is at most one such pair since an STN nogood corresponds to a minimal negative cycle.

The edges adjacent to B may have non-zero lower bounds; we refer to these as *buffers*. In case 4, the line on the arrow from end(B) to start(C) indicates a positive buffer; this means that C cannot start earlier than m time units *after* the end of B, where the buffer is +m. In case 6, there is a negative buffer on the edge from end(A) to start(B); this means that B cannot start earlier than m time units *before* the end of A, where the buffer is –m.

In case 5, the middle unlabelled block indicates a *backwards* chain of activities; in general, there could be several of these in a nogood, causing it to zigzag in direction. To understand this, observe that if we "cut" a nogood by removing the goal activity and its adjacent

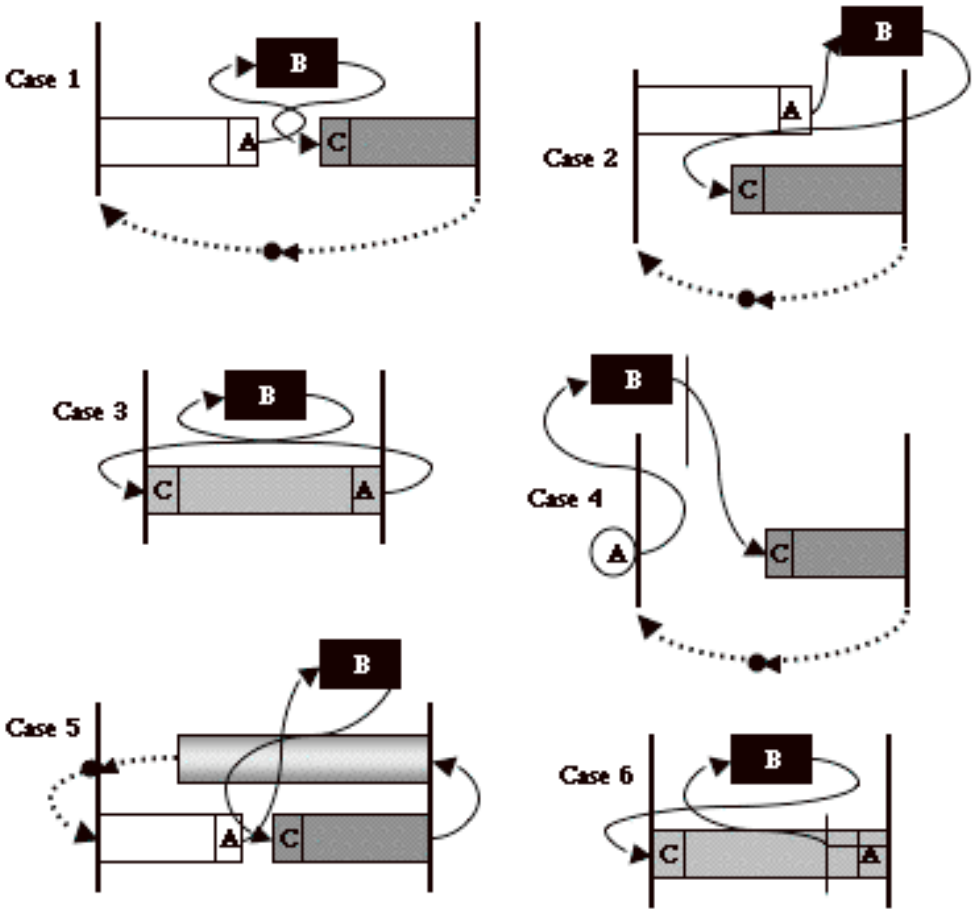

Figure 4: Abstract nogood examples.

edges, the remaining edges fall into a sequence that inherits the ordering from the cyclic nogood. However, this ordering is not necessarily chronological with respect to the start times of the activities, because the lower bounds on the edges between consecutive timepoints can be negative. For example, P [-2]→ Q says that the point Q is at or after a time 2 units before P, rather than being at or after P itself, so Q could actually precede P. Negative lower bounds in the nogood typically arise from constraints that are more naturally viewed as upper bounds, for example, the upper bound part of an activity duration constraint. In the nogood, this produces a path that passes through the activity in the reverse direction, as seen in case 5. Similarly, inconsistencies that involve temporal bound constraints may give rise to backward paths through the origin in the nogood. (These are shown as dotted arrow paths in the examples.) Because of backward paths, A may precede C in the plan, as seen in several of the examples. In principle, the complete path through a nogood can "change direction," i.e., switch from a forwards subsequence to a backwards subsequence, or vice versa, several times. However, in practice, except for paths through the origin, backwards paths were rare in the MER application. There are good reasons for this. Most of the constraints involve non-negative lower bounds. In particular, the expedient constraints are all simple precedences (lower bound zero).

Although the examples look quite different, it is remarkable that they can all be explained in a unified way. The intuition behind the explanation is that there is not enough "room between" A and C to fit B. Note that both the room needed and the room available can be either positive or negative depending on the buffers and on backward blocks. In order to make room, either C has to be moved to the right or A has to be moved to the left, or both. The amount of temporal distance that A and C have to be moved "apart" is determined by the duration of B as well as by the buffers (if any) in the edges from A to B and from B to C. This movement of A and/or C will cause one or more activities within the nogood's boundaries to be moved outside of the boundaries.

We can thus divide the explanation into two parts. The first part describes why A needs to move relative to C, which involves only the constraints associated with the edges adjacent to B. The second part summarizes why A cannot be moved relative to C in the way desired because of the remaining portion of the nogood, which comes from constraints in the existing plan. Thus, we have broken the

temporal cycle into two more easily grasped chains of "new" and "old" constraints that contradict each other. The following is the explanation generated for the example presented earlier.

> For APXS_1 to fit in the plan, the planner would need to slide Start of APXS_2 to no earlier than 31159 after Plan_Start (because of science constraints requiring Start of APXS_1 to be no earlier than Plan_Start and Start of APXS_2 to be no earlier than End of APXS_1).
>
> Currently, Start of APXS_2 is barred from going later than 14794 after Plan_Start because of planner orderings involving End of MB_ON before Start of UHF together with science constraints or pins.

This style of summarization obeys certain principles that we believe facilitate understanding. The first part of the explanation (why C needs to be moved) is given in detail. This follows the principles of *locality*, i.e., adjacent to the goal, and *recency*, i.e., newly activated constraints. The second part (why C cannot be so moved) spells out in detail only the planner constraints. This follows a principle of *visibility*. We can expect the TAP to have an internal mental model of the constraints in the enfolding plan; however, that picture usually does not include the expedient constraints, which are installed "behind the scenes" by the planner and are not graphically shown in MAPGEN's interface.

Recommendations

Following these explanation principles and presenting the user with a concise summary of the inconsistency is very useful, and it may be all the user needs to work around the problem. However, the system can offer further assistance to the user by recommending how to resolve the inconsistency and fit the goal activity into the plan.

When the science constraints are presented to MAPGEN, they are checked for consistency, and if inconsistent, the user must correct them before using the planner. The inputting of science constraints is actually done in a separate tool, the Constraint Editor (CE). In this tool, the notification of inconsistency to the user is relatively simple because (1) there are ONLY science constraints, which are all STN representable (no expedient constraints, no mutex rules), and (2) there are only toplevel activities, thus no decompositions, so no compression is relevant. In the CE, the "raw" nogood is simple enough that it can be presented directly to the user.

Hence, the recommendation approach focuses on the expedient constraints. Recall that an expedient constraint imposes an arbitrarily chosen order between activities in order to ensure that they do not overlap; thus, it may be possible to impose the reverse order instead. The idea is that by reversing one or more of the expedient orderings in the nogood, we might be able to facilitate the necessary activity movements, as specified in the explanation. This is the idea behind the recommendation; however, it must be expressed in operations available to the user in MAPGEN. There is no user operation that directly deletes an expedient constraint; instead, such constraints are deleted as side effects of other operations.

One way the user can cause an expedient constraint to be deleted is by unplanning one of the (mutually exclusive) activities involved. However, as a side effect of the unplan operation, *re-linking* will occur, i.e., new expedient constraints will be imposed to maintain ordering decisions previously made for activities still in the plan. These new constraints may again frustrate the achievement of our intent, which is to get the goal activity into the plan. Hence, the only sure way to forestall such blockage is to eliminate all current and possible expedient constraints between planned activities in the nogood. This can be achieved by unplanning a subset of the activities from the nogood such that no mutual exclusion rules apply to the remaining planned activities in the nogood. Ideally, we want to unplan the minimal number of activities necessary.

The recommendation technique determines this minimal subset of activities to unplan as follows. The first step is to identify all the pairs of mutually exclusive activities in the nogood. Note that the goal activity is not planned at this point, so it is not considered in this step. The second step is to replace any lower-level activity in a pair with its top-level activity. The reason for this step is that, in MAPGEN, it is not possible to unplan only part of an activity expansion structure; i.e., the top-level activity and its descendants are either all in the plan or all in the hopper. The third step is to determine a minimal set cover of these pairs; i.e., the smallest set that contains a member from each pair. This third step is equivalent to the vertex-covering problem and we use a standard greedy approach (Cormen, et al., 1990), which guarantees the result to be no worse than twice the optimal. The recommendation presented to the user is the following:

1. Unplan the activities in the computed set cover; this places them back in the hopper.
2. Apply plan-selected to the goal activity.
3. Replan the previously unplanned activities, using the plan-selected operation on the set.

However, this recommendation is not guaranteed to succeed; it may fail due to a new conflict outside the scope of the original nogood. In this case, a new explanation and recommendation will be generated. A recommendation may also fail due to aspects of the planning process, e.g., the order in which a set of activities is (re)planned, since the search is by design incomplete. Nevertheless, even a failed recommendation may lead to greater understanding on the part of the TAP. If the recommendation is indeed unsuccessful, the user can try the following (more drastic) alternative recommendation:

1. *Set relaxed enforcement mode.*
2. Move A left and/or move C right until they are at the recommended relative positions.

3. *Set full enforcement mode*.
4. Apply the plan-selected operation to the goal activity.

The first step, in MAPGEN, deletes all the expedient constraints, allowing the user to move activities much more freely than the constrained-move operation; however, the planner still enforces the science constraints in this mode. The third step re-establishes whatever new expedient constraints are necessary to enforce the flight rules. The A and C mentioned in step 2 are defined as in Figure 4, i.e., the goal is intended to fit "between" A and C, and the recommended distance is presented in the explanation. Our recommendation process currently presents the user with both of these alternatives in case the first one does not succeed. The first alternative is preferred, if it works, because it involves less potential change to the existing plan. If the alternate recommendation also fails, then the user will most likely have to sacrifice some of the previously planned activities in order to get the goal activity into the plan. This tradeoff involves human-level scientific judgment.

We now illustrate this recommendation process with our previous example nogood. The first step finds two mutual exclusive pairs of planned activities: (APXS_2, MB) and (MB_ON, UHF). The second step converts the pairs to (APXS_2, MB) and (MB, UHF). The third step computes a minimal set cover of {MB}. The full recommendation generated is as follows:

1. Unplan the following activities: {MB}.
2. Plan the goal APXS-1.
3. Replan the unplanned activities.

If that doesn't succeed, try the following alternative:

1. Go into relaxed enforcement mode.
2. Move Plan_Start and/or APXS_2 so that Start of APXS_2 is no earlier than 31159 after Plan_Start.
3. Go into full enforcement mode.
4. Plan the goal activity.

In this case, the first part of the recommendation does work; that is, as a result of following the prescribed steps, the goal activity APXS_1 fits into the plan without having to sacrifice any activity that had been in the plan. The order between MB and UHF is reversed, i.e., MB is moved to the right of UHF, thus creating enough room on the left of the UHF for both of the APXS activities.

Related Research

There has been work related to explanations in a wide variety of research areas, including expert systems (e.g., Wick & Slagle, 1989), case-based reasoning (e.g., Sørmo, Cassen, & Aamodt, 2005), web-based systems (e.g., McGuiness & da Silva, 2004), and constraint-based reasoning. Within this last research area, many earlier systems tended to indicate or "explain" a failure, exhibited as an inconsistency in a constraint network, by outputting a complex, low-level nogood. However, as mixed-initiative systems are becoming more common, more attention is being paid to helping the user understand and resolve such failures. Work on improved explanation facilities has been carried out within different constraint-based research areas, including constraint programming (e.g. Ouis, Jussien, & Boizumault, 2002), multi-agent planning (e.g. Burstein, Ferguson, & Allen, 2000), and scheduling (e.g. Smith, et al., 2004).

This last referenced effort by Smith, et al. is probably the most similar to the work reported in this paper. Their approach employs a *filtering* step to reduce the constraints in the nogood. This step bears some similarity to our compression step, in that both eliminate the details of the expansion (or decomposition) structure in the domain model. Whereas our recommendation approach focuses exclusively on expedient constraints, their approach considers both *search constraints* (which would include expedient constraints) as well as constraints resulting from user decisions. The greatest difference between the two approaches is in the types of resolution methods recommended. Their resolution options all seem to involve user-alteration of specific domain constraints; e.g., changing task duration, due-date, or release-date, which relax the problem specification. In contrast, our generated recommendations are in terms of high-level planning operations that the user can perform; such recommendations change the solution rather than the problem specification.

Concluding Remarks

The current implementation of this work is integrated with our stand-alone version of MAPGEN such that our explanations and recommendations pop up in an APGEN dialog window when the user's planning operation causes an inconsistency in EUROPA's constraint network. We are still in the process of mining the examples of nogoods generated by MAPGEN during MER operations in order to evaluate the generality of the implementation and the quality of the recommendations in terms of how often do they solve the current problem. We would also like to evaluate the quality of the explanations, which would require a careful study of MAPGEN users. The overall objective of this work is to make the TAPs more effective users of MAPGEN. Hence, carrying out a user study that would determine whether we have achieved this objective would be ideal, but involves practical difficulties since the potential user community is very small and quite busy with running missions. We are currently involved with the design of a new ground operations system that is baselined for the Mars Science Laboratory (MSL) mission. The possibility exists of importing the explanation and recommendation techniques into this new system and of collecting feedback from mission operations engineers during readiness exercises and during mission operations.

In addition to pursuing evaluations of the techniques, there are also some future research directions that we'd like to explore. One of the extensions we are investigating is to combine related inconsistencies into one explanation and recommendation. Consider the example of planning three goal activities that are constrained to be in sequence, and there is no room in the current plan for any of them to fit. Currently, the implementation would explain to the user how to make room for each goal activity; whereas, it would be better to generate a single recommendation that made room for all three goal activities.

One of the issues we are currently debating is validation of recommendations. Currently, the system does not test out the recommendations before presenting them to the user; rather it leaves it up to the user to choose whether to follow the recommendation and, if so, to try it out. There are different degrees of validation that could be carried out by the system. Making progress on resolving one inconsistency may cause another inconsistency; thus, full validation would involve chasing down all the inconsistencies until the plan was completely repaired. Since we want the user to have the initiative in the resolution decision, it does not make sense for the system to spend much effort on plan repair just to generate a recommendation. Furthermore, the user gains understanding of the problem by dealing with the multiple inconsistencies one at a time.

In this paper, we have presented a general approach for a mixed-initiative planning system to explain temporal constraint inconsistencies and recommend to the user how to resolve them. The foundation of our approach is a unified characterization of the varied types of nogood structures. Although we were motivated by the MER mission experience and we validated our approach by incorporating it into a version of MAPGEN, it should be easy to adapt it for many other mixed-initiative systems that employ temporal reasoning. Further, we expect the principles underlying our approach to be widely applicable.

Acknowledgements: This work was funded by NASA, initially under the SISM program and then the ETDP. The authors would like to acknowledge the rest of the MAPGEN team: Mitch Ai-Chang, Len Charest, Brian Chafin, Adam Chase, Kim Farrell, Jennifer Hsu, Ari Jónsson, Bob Kanefsky, Adans Ko, Pierre Maldague, Kanna Rajan, Richard Springer, and Jeffrey Yglesias. Secondly, we would like to acknowledge our collaborators, on this research project, for helpful discussions: Lina Khatib, Conor McGann, Karen Myers, and Michael Wolverton. We want to again thank Ari Jónsson, who was an early collaborator on this project and produced the standalone version of MAPGEN that we use.

References

J. Bresina, A. Jonsson, P. Morris, & K. Rajan (2005a). Activity Planning for the Mars Exploration Rovers. In *Proc. of Fourteenth International Conference on Automated Planning and Scheduling*.

J. Bresina, A. Jonsson, P. Morris, & K. Rajan (2005b). Mixed-Initiative Planning in MAPGEN: Capabilities and Shortcomings. In *Workshop on Mixed-Initiative Planning and Scheduling, ICAPS05*.

M. Burstein, G. Ferguson, & J. Allen (2000). Integrating Agent-Based Mixed-Initiative Control with an Existing Multi-Agent Planning System. In *Fourth International Conference on Multi-Agent Systems* (ICMAS'00).

T. Cormen, C. Lieserson, & R. Rivest (1990). *Introduction to Algorithms*. MIT Press/McGraw-Hill 1990.

R. Dechter, I. Meiri, & J. Pearl (1991). Temporal constraint networks. *Artificial Intelligence*, 49:61– 95.

J. Frank & A. Jonsson (2003) Constraint-Based Interval and Attribute Planning, *Journal of Constraints* Special Issue on Constraints and Planning.

Deborah L. McGuinness & Paulo Pinheiro da Silva (2004). Explaining Answers from the Semantic Web: The Inference Web Approach. *Journal of Web Semantics*. Vol.1 No.4. pages 397-413.

P. Maldague, A. Ko. D. Page, & T. Starbird (1998). APGEN: A multi-mission semi-automated planning tool. *First International NASA Workshop on Planning and Scheduling*, Oxnard, CA.

S. Ouis, N. Jussien, & P. Boizumault (2002). COINS: a constraint-based interactive solving system. In Alexandre Tessier (ed), Proc. of the *Twelfth International Workshop on Logic Programming Environments*.

S.F. Smith, G. Cortellessa, D.W. Hildum, & C.M. Ohler (2004). Using a Scheduling Domain Ontology to Compute User-oriented Explanations. In Proc. of the Workshop *Planning and Scheduling: Bridging Theory to Practice*, ECAI-04, Valencia, Spain, pp 85-94.

Frode Sørmo, Jörg Cassens, & Agnar Aamodt (2005). Explanation in Case-Based Reasoning – Perspectives and Goals. *Artificial Intelligence Review*; Volume 24, number 2, pp. 109-143.

I. Tsamardinos & M. Pollack (2003), Efficient solution techniques for disjunctive temporal reasoning problems, Artificial Intelligence, Vol. 151, no. 1-2, pp. 43-89.

M.R. Wick & J.R. Slagle (1989). An explanation facility for today's expert systems. *IEEE Expert*, Volume 4, Issue 1, Spring 1989 Page(s):26 – 36.