,

Plan Steering and Mixed-Initiative Planning

Paul Cohen
Tim Oates
Robert St. Amant
Experimental Knowledge Systems Laboratory
Computer Science Department, LGRC
University of Massachusetts
Amherst MA 01003-4610
{cohen,oates,stamant}@cs.umass.edu

Abstract

We introduced the term plan steering to describe the process of predicting and avoiding plan failures, much as one steers a car around potholes. We will describe a plan steering system for managing transportation networks, and a suite of experiments which show that mixed-initiative plan steering works better than humans or programs working alone. More recently we built a mixed-initiative planner for data analysis. In a large dataset, the space of relationships between variables is vast, and many look attractively informative. It proved challenging to keep the human/computer analyst team working toward the same goals and informing each other of their progress.

Introduction

Plan steering addresses the problem of predicting and avoiding, in real time, pathological states that make it difficult or impossible to achieve goals. We describe a plan steering agent that assists humans in monitoring plans as they unfold, detecting and predicting pathological situations and suggesting beneficial plan modifications in real time. The agent helps the human to steer the plan out of or around trouble. To explore issues related to plan steering we have constructed a system for the related task of schedule maintenance.

Mixed-initiative planning is more challenging than plan steering for several reasons. First, the interaction between agent and user is focused in plan steering by the emergence of a problem; the control loop is simple: a pathology is predicted, then avoided. Control in mixed-initiative planning is much more complex, especially in domains that require rapid changes in focus of attention, instead of a stately progression of problems and solutions. Second, a mixed-initiative assistant has to "read the mind" of the user to a much greater extent than a plan steering assistant. To explore these issues we have built a mixed-initiative planning assistant for data analysis, a complex task with an enormous search space.

An Agent for Schedule Maintenance

We have constructed an agent to perform schedule maintenance in the transportation planning domain. Its goal is to move cargo quickly through a network starting with an initial schedule. This task is more difficult than simple load balancing due to the structure imposed by the initial schedule; we want to maintain as much of that structure as possible.

Our agent manages a transportation simulation called TransSim. A TransSim scenario consists of a collection of ports, ships, cargo, and routes. The routes, called "simple movement requirements" or SMRs, specify when each piece of cargo is to begin its journey and through which ports it is to travel. Ports and ships are limited resources, constraints govern which kinds of cargo can go on different kinds of ships, and the simulation is nondeterministic in that ship travel times may vary.

SMRs are fully specified at the start of each scenario. The specific ship on which a piece of cargo will travel is determined at run time, based on availability. There may be a free ship locally or one may be requested from a nearby port. Therefore, the behavior observed during a simulation is largely determined by the scenario's SMRs. If many SMRs travel through one port, that port is likely to become clogged. If SMR destinations are evenly distributed in time over all ports then problems are unlikely to arise. The key point is that there is really no plan in this domain. Cargo must travel specific routes starting at certain times, but how that happens is determined dynamically at run time. This is why we characterize the task as schedule maintenance, and why our agent is but a first stab at the more general plan steering problem.

A generic plan steering system comprises a human user, a plan steering agent, a pathology demon, and the plan execution exvironment. The architecture for schedule maintenance is identical, only the task changes. A pathology demon monitors the environment as a plan unfolds, detecting and predicting pathological situations. The plan steering agent monitors the demon's output and formulates plan modifications that address the problems found by the demon. A human user evaluates the agent's advice and effects plan changes when appropriate.

A pathology demon has been implemented to monitor TransSim as cargo is shipped about. It predicts when and where in the network a ***bottleneck*** is likely to arise. Bottlenecks result when too many ships attempt to travel through a port, causing reduced throughput. The demon uses simple domain information to make its predictions. It looks at the current state of each port and ships that are traveling in channels toward the port and assigns a probability to each possible state of the port on each day out to a horizon. That is, the demon only uses information local to a given port. Resource-based schedule revision with local information was used successfully in (Smith ***et al.*** 1990).

The schedule maintenance agent combines the demon's predictions for multiple days in the future to determine which ports are likely to become clogged. The agent then uses simple heuristics to generate advice that, when implemented, will either alleviate or avoid the predicted bottleneck. This may be contrasted with reactive approaches, such as (Ow, Smith, & Thiriez 1988) and (Prosser 1989), that respond to unexpected events at the time that they occur. Currently, the only advice the agent offers is based on a simple rerouting heuristic. If a piece of cargo is being loaded onto a ship bound for a potential bottleneck, the agent changes the cargo's SMR so that it travels to the port closest to the original destination that is not problematic. This seems to be a reasonable approach in that rerouted cargo continues to make progress toward its destination. We assume that ports that are close geographically are "equivalent" in some sense.

Performance Assessment

The pathology prediction demon models each ship as a probability distribution of arrival times. Combining this distribution with the current state of each port, the demon arrives at a predicted docking queue length. The model is similar to that used in (Muscetolla & Smith 1987) for exploring the effects of resource allocation decisions. Long docking queues indicate the presence of a bottleneck. Several factors affect the accuracy of the demon's predictions. They are (1) the distance into the future for which predictions are made, (2) the certainty of the demon's knowledge about ship arrivals, and (3) a prediction threshold that controls how aggressive the demon is at making predictions (low values are aggressive, high values are conservative). We expect predictions to be less accurate when made far into the future or when there is a lot of variance in ship arrivals. There should be an optimal prediction threshold, at least for a given level of the other two factors.

The accuracy of the demon was explored by running a fully factorial experiment with 10 simulations for each of three levels of the factors (a total of 270 trials). The demon's predictions for each port as well as the actual state of each port were recorded and average squared prediction error was calculated for each trial. For predictions 2, 4, and 6 days into the future, average squared error was 0.137, 0.244, and 0.214; increasing but not monotonically. Though the effect of variance in the demon's knowledge about ship arrivals was not a significant main effect, it did interact significantly with prediction distance. High variance had an increasingly adverse effect on error as prediction distance was increased. Finally, plotting error at several prediction threshold values resulted in a bowl shaped curve with 0.2 being the optimal threshold.

Measures of Cost

Having established some influences on demon accuracy, we turned next to the schedule maintenance agent and its performance. We defined several measures of cost associated with a single run of TransSim, and used them to evaluate the effect of moving from one experimental condition to another: ***Bottleneck Predictions*** – the sum over all days of the number of ports marked as potential trouble spots by the demon for a given day; ***Cargo Transit*** – the sum over all pieces of cargo of the number of days from when the item first came on line until it reached its final destination; ***Idle Cargo*** – cargo is considered idle if it is ready to be loaded onto a ship but none is available or if it is sitting in a ship queued for docking, and this measure is the sum over all days of the number of pieces of idle cargo; ***Queue Length*** – the sum over all days and all ports of the number of ships queued for docking; ***Simulated days*** – the number of simulated days required to ship all cargo to its final destination; ***Ship utility*** – the sum over all simulated days of the number of ships traveling empty on a given day.

Agent Advice vs. No Advice

Several experiments were run to evaluate the performance of the agent working alone, without input from the user. In each, the agent monitors a running simulation and either implements its own advice or does nothing. By varying the number of SMRs for a simulation we have some crude control over the frequency and duration of bottlenecks. Few SMRs results in low traffic intensity and few bottlenecks. Many SMRs has the opposite effect. Therefore, we ran an advice vs. no advice experiment at each of three levels of the number of SMRs (25, 30 and 35) in the scenario. A total of 10 trials (simulations) were run in each condition. We assess the impact of the agent's actions by performing an analysis of variance (ANOVA) of each cost measure on whether or not agent advice was implemented.

Increasing the number of SMRs did in fact increase the number or severity of bottlenecks. The mean queue length (an objective measure of bottleneck severity) in the "no advice" condition is positively correlated with the number of SMRs in the scenario. The agent was beneficial regardless of pathology intensity. In fact, the percent reduction in all cost measures was largest in the most pathological 35 SMR case. The agent achieved its design goal of reducing queue length. In doing so, it reduced the amount of time cargo spends sitting idle and actually increased the speed with which cargo travels to its destination locally (decreased Cargo Transit) and globally (decreased Simulated Days). To investigate the effects of increasing the complexity of the task on the agent's ability to perform, a similar experiment was run with a larger scenario. The number of ports and ships were doubled (to 10 and 40 respectively) and the number of SMRs was set at 60. Again, 10 trials per condition were run. We obtained significant reductions in all cost measures except Ship Utility. Comparing percentage reductions with all three of the previous experiments we found that increasing the complexity of the task increases the extent to which the agent is able to help.

The obvious conclusion is that in a wide variety of conditions, the agent is able to reduce the costs associated with a simulation. Neither pathology intensity nor problem complexity seem to nullify its ability to perform. In fact, the agent shines most brightly in precisely those situations where it is needed most, highly pathological and large/complex scenarios.

Control Condition: Random Advice

The previous experiment lacked a control condition. Perhaps the agent's advice isn't so good; perhaps ***any*** actions would help alleviate bottlenecks. In fact, we ran an experiment to determine the effects of demon accuracy on agent performance, with the surprising result that there was no significant impact of demon accuracy on the efficacy of the agent. If the agent performs equally well with good and bad predictions of bottlenecks, then perhaps its ability to reduce costs is due to shuffling of routes, not to its ability to predict. Random rerouting, periodically picking a piece of cargo and sending it on a newly chosen random route, may be as good as the agent's "informed" advice. Random rerouting has the advantage of tending to evenly distribute cargo over the network, minimizing contention for any one port. The disadvantage is that it destroys the structure inherent in the initial schedule.

To investigate the utility of random advice, we ran an experiment in which the agent, with varying frequency, rerouted a randomly selected piece of cargo. This was done with no regard for bottleneck predictions. We varied the probability of performing a reroute on each day over four values: 5%, 15%, 25%, 35%. As with the advice vs. no advice experiments, the number of SMRs in the simulation was varied to get a feel for how these effects changed with pathology intensity. ANOVA was used to identify significant effects.

As expected, cost measures decrease with increasing frequency of random rerouting. For the 25 SMR case the decrease tends to flatten out with the two highest levels of randomness being nearly equivalent. For the 30 SMR case the decrease continued monotonically whereas for the 35 SMR case the cost measures actually spiked back up at the highest level of randomness. It appears that for highly pathological scenarios, there is a limit beyond which random shuffling hurts more than it helps.

In the 25 SMR case any amount of randomness is indistinguishable from the agent's advice when performance is measured in terms of Idle Cargo and Queue Length, and virtually identical when measured by Cargo Transit. The situation is somewhat better in the 30 SMR case. With an average of 12 reroutes, the agent is able to equal the performance of randomly rerouting 20 times in the Idle Cargo and Queue Length columns. (Note that a Scheffé test indicates that the amount of real advice given is significantly lower than the amount given in the random 25 case.) This is an important point. Remember that there is no "plan" in this domain, only the structure imposed by the initial SMRs. Improving performance with the minimal number of rerouting decisions is key to maintaining that structure. The results in the 35 SMR case are somewhat inconclusive. The agent performed better than the random 5 level but only matched performance of the other random levels. The two highest levels of randomness (which had significantly higher mean numbers of reroutes) were statistically equivalent to the agent in terms of simulation costs. Again we see that the demon is able to achieve good results with a few well directed rerouting decisions rather than with large numbers of random ones.

The same large scenario described previously was used to investigate the effects of problem size and complexity on the efficacy of random advice. The results here are striking. The amount of rerouting performed by the agent was statistically equivalent to the random 25 condition only. In that condition, Scheffé tests show that the agent's performance is significantly better than random advice; its domain knowledge is paying great rewards. Looking at the data another way, the agent performed equally as well as the random 35 condition with 34% fewer reroutes.

Our initial proposition that random rerouting would help to lower the various cost measures was borne out. In fact, it seems that more random rerouting is better than less, except perhaps in highly bottlenecked scenarios. There existed some level of randomness that equalled the performance of the agent for each of the previous experiments. However, the agent typically rerouted many fewer pieces of cargo than the equivalent level of randomness, thereby preserving more of the structure of the simulation. Finally, it appears that for large/complex scenarios the difference between randomness and the agent is more pronounced.

Highly Constrained Scenarios

To increase the realism of the agent's task, several constraints were added to the scenarios. There are three types of cargo: CONT (containerized), GENL (general), and RORO (roll-on,roll-off). Rather than using a single cargo type, we used multiple types and limited the cargo handling capabilities of both ships and docks. Now for cargo to flow through the network, it must match in type with any ship that is to carry it and any dock where it will be handled. We ran agent advice vs. random advice experiments under these conditions after modifying the agent to consider the additional constraints. Whenever random advice generated an incompatible routing assignment, it was penalized with a short delay for the offending piece of cargo.

Again we found that the agent is able to match the performance of random rerouting with many fewer changes to the schedule. The fact that Queue Length for the agent is different only from the random 5 condition and that Cargo Transit for the agent is different only from the highest and lowest levels of random advice points to a result of constraining the scenario: the performance of random advice in any one condition is highly variable. The variance associated with Cargo Transit for random advice was on average 3.5 times higher than for agent advice. Likewise, the variance associated with Idle Cargo was 4.1 times higher and the variance associated with Queue Length was 3.2 times higher. The agent is able to achieve good average case performance with much higher consistency when compared to random rerouting by making a few appropriate rerouting decisions.

Humans and Agents Working Together

In another study, we demonstrated that humans and the plan steering agent perform better together than either does alone (Oates & Cohen 1994). Space precludes a detailed account of the study. The major conclusions, however, are these: Humans are not good at steering plans in real time; they make fewer adjustments than are necessary. The plan steering agent, in contrast, makes too many adjustments. Although plans require less time to finish when they are steered by the agent, they are also modified a lot more often. If plan modification is costly, which it often is, then the agent's advice should be moderated. The ideal configuration turns out to be humans and the plan steering agent working together. Humans can't generate plan steering actions quickly enough, but they can review the agent's suggestions and moderate them. In this condition, the throughputs and run times of plans were almost as good as in the agent-alone condition, but much less violence was done to the structure of the plans.

A Mixed-initiative Planner for Exploratory Data Analysis

Analysts in the military, business, and government spend a great deal of time looking at data with outmoded, cumbersome tools. We have developed a mixed-initiative assistant for exploratory data analysis (EDA). Although EDA is regarded as the province of statisticians, it is, in fact, an extremely general activity. Basically, EDA involves noticing and then tracking down explanations of patterns in data. Both activities are difficult: Interesting patterns are often obscured by noise or uninteresting patterns, and it can be tricky, painstaking work to explain a pattern in terms of variables or subsets of variables in one's data.

Viewed as a search process, EDA is intractable. The number of EDA operations is very large, and the number of subsets of variables and observations to which these operations are applied grows exponentially. Statistical packages provide comfortable interfaces for these operations and variables, but they don't solve the more basic control problem: If one views EDA as the application of statistical operations to data, then its search space is too large.

We have designed and implemented an Assistant for Intelligent Data Exploration, a knowledge-based, mixed-initiative planning system that helps users carry out EDA. In AIDE, data-directed mechanisms extract simple observations and suggestive indications from the data. EDA operations then act in a goal-directed fashion to generate more extensive descriptions of the data. The system autonomously pursues its own goals while still allowing the user to guide or override its decisions.

Notably, the user interacts with **AIDE** not at the level of statistical operations, but in terms of data analysis ***strategies***. Thus we reformulate the search space of mixed-initiative EDA to make it smaller.

The mixed-initiative approach imposes some special requirements on our design of AIDE. In the domain of EDA, the quality of the final result strongly depends on the decisions made at intermediate points, and these decisions can often benefit from human judgment. We must thus be concerned with more than whether the planner can construct a solution to a problem. We must also ensure that the system's planning and the user's understanding proceed along the same path, so that the system and the user work cooperatively rather than at odds. This can be difficult: the system should anticipate the user's actions whenever possible, incorporating knowledge of standard statistical practice; the system should not constrain the user from taking the analysis in any desired direction; the system must decide at which points the user can come into the planning process, and must be able to exchange relevant, context-dependent information with the user when necessary.

An Example

To see how an automated assistant might contribute to the exploration process, consider the dialog below. We present the exchange as a natural conversation; in reality, the dialog is conducted with mouse clicks, menu selections, typed input, and graphical displays. The information exchanged on the part of both conversants is accurately rendered.

USER: Load dataset "planner-evaluation.data".

...

USER: Show me all highly correlated bivariate relationships.

AIDE: (Displays graph of variables, relevant relationships highlighted.)

USER: Select relationship (Resources-Consumed, Task-Duration).

AIDE: (Resources, Duration) has these indications: high correlation ($r = 0.93$); outliers in both Resources and Duration; clustering. Suggested courses of action are (1) generating a linear fit and (2) exploring clusters.

USER: Why the linear fit?

AIDE: A linear fit can be appropriate for relationships with high correlation.

USER: Why the given ordering of actions?

AIDE: When there is a high correlation and no evidence of extreme outliers, we fit a line before trying other possibilities.

USER: Okay.

AIDE: (Generates linear fit.) A three-group resistant line is appropriate; its intercept is -0.96, its slope 0.0017. (Presents graphical display of linear fit.) There are outliers in the residuals, as well as evidence of clustering.

USER: Show recent decisions.

AIDE: (1) With the goal of exploring (Resources, Duration), we decided to fit a line. The active alternative is to explore clusters. (2) With the goal of fitting a line, I decided without consultation on a resistant fit. Alternative is a regression fit.

USER: Make a note: I am confident that the relationship (Resources, Duration) is linear, despite indications of outliers and clustering.

AIDE: (Accepts note.)

USER: Okay.

AIDE: (Considers linear fit phase to be complete.) The possibilities remaining to explore (Resources, Duration) are (1) the linear fit, which is in progress, and (2) cluster exploration, which has not started.

USER: Okay.

AIDE: (Begins cluster exploration.) There are several different ways (Resources, Duration) can be broken into clusters...

AIDE relies on specific abilities in participating in this kind of exploration. First, AIDE makes heuristic judgments about the data. This lets it decide on its own which procedures are appropriate, rather than relying entirely on the user. In general, AIDE takes advantage of indications in the data as more detail is gradually uncovered, to focus its exploration on promising areas. Second, AIDE does more than simply react to user commands. While the familiar "submit command; view result" cycle is supported, AIDE can sometimes anticipate the user's actions through its representation of common statistical practice. Third, AIDE does not go to the other extreme of being completely autonomous; its decision process is accessible to the user. At any time the user can review the assistant's decisions concerning statistical procedures, parameter settings, variable selection, and so forth. Further, the user is given a means of navigating through the decisions, to reevaluate and modify them when necessary.

A mixed-initiative planning environment

AIDE is built around a partial hierarchical planner (St. Amant & Cohen 1996). Systems that use this approach include PRS (Georgeff & Lansky 1987), the PHOENIX planner (Cohen *et al.* 1989) and the RESUN system (Carver & Lesser 1993). Rather than constructing plans from primitive actions, starting from scratch for each new problem, the AIDE planner relies on a library of abstract plans and primitive actions for its processing. These plans and actions implement statistical procedures and strategies at different levels of abstraction. AIDE explores a dataset by searching through its library, instantiating appropriate plans, and fitting them into a growing network of plans already executing.

The existence of pre-defined plans greatly reduces the search space. AIDE nevertheless still faces the meta-level problem of deciding which plans are appropriate. Several plans in the library may potentially satisfy a single goal: in the dialog above, the planner might satisfy a **`describe relationship`** goal with a **`linear-fit`** plan or a **`decompose-clusters`** plan. Similarly, plan variables may be bound in a variety of ways: there are often many plausible criteria by which a relationship can be clustered, and it is not always obvious which is the best way. We refer to choices between plans and plan variable bindings as focus points. Planning, at the meta-level, is a matter of deciding which focus point to concentrate attention on and which of its choices should be pursued. AIDE makes these decisions based on indications in the data, results that have been generated up to the current point, and the context in which the focus points are generated.

The user interacts with AIDE at the meta-level. Managing the interaction at this level gives the AIDE environment much of its power. Relieved of the necessity of calling individual actions directly, the user can take a strategic view of the process. AIDE explores a dataset by incrementally expanding a set of active focus points representing relevant decisions: which variables or relationships should be examined, which plans should be active to explore them, how plans should be parameterized. The user participates in the exploration by viewing the sequential presentation of these focus points and providing acknowledgement or guidance when appropriate.

Mixed-initiative systems like AIDE face a two-sided problem. AIDE may take actions on its own, without consulting the user; AIDE mustn't confuse the user, however, by presenting results seemingly unrelated or irrelevant to the current state of exploration. Similarly, the user may take actions for which AIDE can find no justification; AIDE must nevertheless follow the path set by the user, to contribute to the exploration once it again reaches a familiar point. This issue involves maintaining shared knowledge about a problem to be cooperatively solved (Allen 1994).

AIDE's solution relies partially on the notion of common statistical practice and its relationship to planning. In many situations data analysis follows well-understood paths, involving procedures familiar to most users. For example, if one generates a regression fit for a relationship, one also generates residuals. The residuals may contain patterns not captured by the fit; by examining the residuals one ensures that the fit is an adequate description. Thus if the user tells AIDE that a linear fit is appropriate, AIDE will present both the fit and the residuals. AIDE's plans implement common statistical procedures, so that when AIDE executes such plans without consultation, the user sees nothing unusual in the system's initiative.

For several reasons we need to provide direct communication about the exploration process, in addition to relying on implicit knowledge about common practice. The user may get temporarily distracted and need to review the last few decisions. AIDE may make a mistake; the user needs a way to tell AIDE to return to an earlier decision point to fix it. AIDE may make a sequence of decisions, without consultation, that can only be explicated by a review of each one's justification and context. Exploration is finally an opportunistic process. Intermediate results can cause different points in the search space to become more promising, points that may not appear directly connected to the current state. For example, the user may reach a conclusion beyond AIDE's scope, say, that a variable is relevant to a relationship examined earlier, based on exogenous information about the domain that AIDE cannot know. When this conclusion is reached, the user may want to tell AIDE to return to the earlier relationship, so that it might be examined further. For these reasons AIDE needs to give the user access to its decision-making process.

Recent research has drawn a useful analogy between mixed-initiative planning and a dialog between problem-solving agents (Allen 1994). James Allen identifies three distinguishing characteristics of mixed-initiative planning: flexible, opportunistic control of initiative; the ability to change focus of attention; mechanisms for maintaining shared, implicit knowledge. AIDE meets these requirements as follows.

AIDE's control of initiative changes with context. That is, whether a decision is presented to the user or is settled internally depends on situation-dependent factors. For example, if AIDE determines that only one plan is able to satisfy a given goal (i.e., all others rendered inactive by evaluation rules), then this decision point will not be presented to the user. An exception is made in the case where a plan is being selected for the initial exploration of a variable or relationship. Choosing an appropriate initial plan for exploring a relationship can often depend heavily on external knowledge; thus the decision about how to proceed from the new point is presented to the user even if only one course of action seems appropriate. At any other point in the exploration, the user is free to take the initiative. Because AIDE supports the interaction style of a conventional statistical package, the user can simply select operations from the menu bar, entirely disregarding the suggestions and results AIDE generates.

When the user changes focus of attention, AIDE follows suit. For example, the user may decide to abandon exploration of relationship (x, y) for exploration of (z, w), simply by choosing **`Select variable/relationship`** from the menu bar and constructing the new relationship. AIDE interprets this as a shift in focus of attention, and runs the appropriate meta-level actions to match the shift. AIDE also makes limited decisions on its own to change focus. For example, after fitting a line to a relationship and generating residuals, AIDE presents a set of residual examination and other plans to the user. An **`Okay`** gesture, which usually indicates that the top-rated plan for the current decision should be activated, rather in this context causes AIDE to refocus on other plans for exploring the relationship.

Part of the shared context between the user and AIDE is implicit in plans designed to implement common statistical procedures. Navigation mechanisms provide an explicit means of supporting this context. The user can view the data under consideration, results constructed, commands carried out, and planning structures leading to the current state. There is no complementary facility for AIDE to ask for clarification of user actions, however, which could clearly be helpful in some situations.

Besides making decisions at the local level of focus points, the user can also change the AIDE's focus of attention through a navigation facility. The navigation facility shows the user different views of the exploration process:

A ***command history*** view displays a list of the commands that the user has selected from the menus. Selecting a command shows the operation, the data to which it was applied, and the result generated. This facility gives a superficial history of exploratory actions taken—though not of the decisions that led to their selection.

An ***overview*** view presents a graph of the dataset, with variables represented as nodes and relationships as arcs, marked appropriately to show which have been explored and the types of descriptions that have been established for them. The user can review a variable or relationship along with its indications from this view, and can direct AIDE to return to the initial point of exploring that variable or relationship.

A *data* view presents a display of the dataset, similar to the overview, but organized in textual tree form. In addition to the variables and relationships of the dataset, this view also presents derived data (e.g., transformations, partitions) and results.

A *plan* view displays the plan structure as it has been elaborated to the current point. Each focus point is presented, with documentation about the decision it manages. By browsing through this graph, the user can review past decisions. The user can also cause AIDE to return to an earlier decision to resume at that point.

Through this interface the user can review past decisions, revisit them, and if necessary modify them.

Conclusion

We have described work in mixed-initiative planning. The first project involved an agent that steers transportation schedules, modifying them dynamically to avoid bottlenecks. We learned that humans and the plan steering agent, working together, work better than either, alone. The second project involves tight collaboration between human data analysts and an assistant for exploratory data analysis. The challenge was to give AIDE autonomy but not have it "run away" from the user, to keep the user and AIDE thinking along the same lines during data analysis.

Acknowledgments

This work is supported by ARPA/Rome Laboratory under contract #'s F30602-93-C-0100 and F30602-95-1-0021, and by ARO under an AASERT grant, #DAAH04-95-1-0466. The US Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements either expressed or implied, of the Advanced Research Projects Agency, Rome Laboratory or the US Government.

References

Allen, J. F. 1994. Mixed initiative planning: Position paper. Presented at the ARPA/Rome Labs Planning Initiative Workshop. URL http://www.cs.rochester.edu/research/trains/mips/.

Carver, N., and Lesser, V. 1993. A planner for the control of problem solving systems. *IEEE Transactions on Systems, Man, and Cybernetics, special issue on Planning, Scheduling, and Control* 23(6).

Cohen, P. R.; Greenberg, M. L.; Hart, D. M.; and Howe, A. E. 1989. Trial by fire: Understanding the design requirements for agents in complex environments. *AI Magazine* 10(3):32–48.

Georgeff, M. P., and Lansky, A. L. 1987. Reactive reasoning and planning. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, 677–682. American Association for Artificial Intelligence.

Muscetolla, N., and Smith, S. F. 1987. A probabilistic framework for resource-constrained multiagent planning. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, 1063–1066.

Oates, T., and Cohen, P. R. 1994. Toward a plan steering agent: Experiments with schedule maintenance. In *Proceedings of the Second International Conference on AI Planning Systems*, 134–139. AAAI Press.

Ow, P. S.; Smith, S. F.; and Thiriez, A. 1988. Reactive plan revision. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, 77–82.

Prosser, P. 1989. A reactive scheduling agent. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1004–1009.

Smith, S. F.; Ow, P. S.; Muscetolla, N.; Potvin, J.; and Matthys, D. C. 1990. An integrated framework for generating and revising factory schedules. *Journal of the Operational Research Society* 41(6):539–552.

St. Amant, R., and Cohen, P. R. 1996. A planner for exploratory data analysis. In *Proceedings of the Third International Conference on Artificial Intelligence Planning Systems*. To appear.