Mixed-Initiative Schedule Maintenance: A First Step Toward Plan Steering

Tim Oates and Paul R. Cohen

Computer Science Technical Report 94-31

Experimental Knowledge Systems Laboratory
Department of Computer Science, Box 34610
Lederle Graduate Research Center
University of Massachusetts
Amherst, MA 01003-4610

Abstract

When a plan involves hundreds or thousands of events over time it can be difficult or impossible to tell whether those events are unfolding "according to plan" and to assess the impact of dynamic plan modifications. Pathological states may arise in which goals cannot be attained or are attained too slowly. Plan steering is an agent-based approach to this problem. The agent monitors an unfolding plan, detects and predicts pathological situations, and develops dynamic plan modifications that will steer the plan around the problem. We present results for a system that performs the related task of schedule maintenance in the transportation planning domain. We evaluate system performance at pathology prediction and pathology avoidance and show that the agent, using limited domain knowledge and simple heuristics, is able to improve throughput significantly. We describe experiments in which humans perform the same schedule maintenance task both with and without the aid of the agent, and show that the human and the agent working together achieve better results than either working alone.

This work was supported by ARPA/Rome Laboratory under contract #F30602-91-C-0076.

1 Introduction

Plans formulated to run in the real world will often fail due to the complexity and unpredictability of the environment. Existing methods to deal with this problem include real time recovery from plan failures [1] [3] [12] and post-hoc plan repair based on failures observed while executing the plan [9]. Failure recovery mechanisms, such as replanning, can be expensive, and it may not be feasible to repair a plan by letting it repeatedly fail. An alternative strategy is to monitor the execution of the plan, attempting to predict pathological states that make it difficult or impossible to achieve goals [8]. Doing so admits the possibility of effecting plan modifications in real time to avoid pathological states.

Plan steering is a mixed-initiative approach to real time prediction and avoidance of plan failures. A plan steering system comprises a pathology demon that monitors the execution environment to detect and predict pathological states, a plan steering agent that evaluates the demon's predictions and formulates plan modifications to avoid predicted pathologies, and a human user who monitors the environment, the demon, and the agent. The human and the agent work together to steer the plan away from potential problems by intervening before they develop. The benefits of keeping computers in the loop are clear. For large, complex plans, involving hundreds or thousands of events over time, determining whether events are unfolding according to plan and assessing the impact of dynamic plan modifications are impossible for humans.

As a first step toward plan steering, we built an agent for the related task of schedule maintenance in the transportation planning domain. We experimentally assessed the performance of the agent at its two primary tasks: predicting schedule pathologies and formulating schedule modifications to avoid those pathologies. In those experiments the agent was completely responsible for managing the schedule; no human intervention was allowed. Next, we evaluated the performance of humans at that same task both with and without the aid of the agent. We show that the agent can help human planners - indeed, working in concert, humans and an agent perform better than either does alone.

2 The Schedule Maintenance System

The architecture of a generic schedule maintenance system is shown in Figure 1. A pathology demon monitors the environment as a schedule unfolds, detecting and predicting pathological situations. The schedule maintenance agent monitors the demon's output and formulates schedule modifications that address the problems found by the demon. A human user evaluates the agent's advice and effects schedule changes when appropriate. The architecture for plan steering is identical, only the task changes.

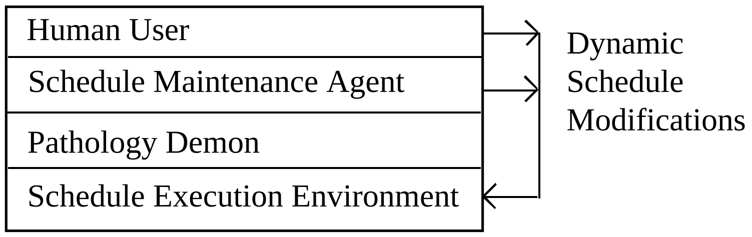

Human User
Schedule Maintenance Agent
Pathology Demon
Schedule Execution Environment
Dynamic Schedule Modifications

Figure 1: Schedule Maintenance Architecture

The task for our system is management of schedules in a simulated shipping network called TransSim. TransSim is capable of representing most of the objects and constraints in the IFD2 shipping domain provided by ISX, although we abstract some of those domain features for the sake of simulation speed. A TransSim scenario consists of ships, ports, cargo, and simple movement requirements (SMRs) for each piece of cargo. An SMR specifies the route that a piece of cargo is to take through the network and when it is to begin its journey. The SMRs of a scenario constitute its schedule and largely determine the behavior of the simulation. They may be thought of as the schedule generated by a separate scheduling program that must be adhered to as closely as possible and are based on the TPFDD's of IFD2. The scenarios used in experiments throughout this paper enforce constraints on the types of cargo that both ships and ports can accommodate. A typical scenario consists of seven ports, twenty eight ships, and forty SMRs.

If many SMRs reference any one port then it is likely that a *bottleneck* will develop at that port. Ports are limited resources and ships must queue for service when a port is being used to load or unload another ship's cargo. A bottleneck exists at a port when the docking queue at that port is "large" and results in reduced throughput. The goal of the schedule maintenance system is to maximize throughput. It does so by predicting the occurrence of bottlenecks at each port in a scenario and making changes to SMRs where appropriate. The system attempts to minimize the number of changes to preserve as much of the structure imposed by the initial SMRs as possible. These two goals are often at odds with one another so an appropriate balance must be found.

A pathology demon has been implemented to monitor TransSim as cargo is shipped about. It uses simple domain information to predict when and where in the network a bottleneck is likely to arise. The demon 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 [10]. The schedule maintenance agent combines the demon's predictions for multiple days in the future to determine which ports are likely to become substantial problems. 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 [7], that respond to unexpected events at the time 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.

3 Performance Assessment of System Components

We now focus on assessing the ability of our system to manage the TransSim domain. First, we will investigate performance of the pathology demon as influenced by environmental factors. Then, we will determine whether the agent's advice is helpful, harmful, or neither. We will also explore how much of the benefit is due to the intelligence built into the agent and how much is due to other factors. Finally, we want to ascertain to what extent environmental factors, such as pathology severity and problem complexity, affect the agent's performance. The general procedure is to run several simulations in each of a variety of experimental conditions, taking several dependent cost measures which are then compared to determine the effect of the condition [5]. Experiments in which humans manage the TransSim domain both with and without the aid of the agent are presented in Section 4.

3.1 Pathology Demon

The pathology prediction demon models each ship as a probability distribution of arrival times. Combining these distributions with the current state of each port, the demon arrives at a predicted docking queue length for each port. The model is similar to that used in [4] 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 ten 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.

3.2 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. These were used 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 arrived at its port of embarkation 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. This measure is the sum over all days of the number of pieces of idle cargo.

Queue Length Each port has a limited number of docks. Ships waiting to dock are placed in the docking queue. This measure is 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 Ships may travel empty when called from one port to another to service cargo. This measure is the sum over all simulated days of the number of ships traveling empty on a given day.

When the agent is not offering advice, we expect a positive correlation between Bottleneck Predictions and many of the other measures such as Queue Length. If the demon is predicting many bottlenecks, and the agent is not doing anything to avoid them, then if the demon is accurate our other cost measures will rise accordingly. The agent was designed to predict and avoid large docking queues so we expect its actions to reduce Queue Length. One extreme way to reduce Queue Length is to let there be only one ship traveling at any time. Both Cargo Transit and Simulated Days provide another view into the agent's performance on a global scale so we may ensure that it is not adopting a similarly pathological strategy.

3.3 Agent Advice vs No Advice

Several experiments were run to evaluate the agent's performance.[1] They typically consisted of two conditions. In the first condition the agent monitors a running simulation but offers no advice. In the second condition the agent monitors a running simulation and implements any advice that it may have. The goal in each of these experiments is to determine how the agent's performance compares to doing nothing. 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 is used. If the result is significant then the actions of the agent affect the cost measure and inspection of cell means will indicate if it is a beneficial effect. A total of ten trials (simulations) were run in each condition. Also, we allowed any type cargo to be carried by any type of ship. Experiments with more constraints are described in Section 3.5.

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 in the scenario. The results for the 35 SMR case are shown below in Table 1.

By allowing the agent to follow its own advice, we obtain significant reductions in all cost measures except Ship Utility. In fact, the percent reduction in all cost measures was largest in the most pathological 35 SMR case (compared to the other cases which are not shown). 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

[1]All experiments and analysis utilized CLASP/CLIP [2].

Cost	p Value	No Advice Mean	Advice Mean	% Reduction
BP	0.0	232	169.9	26.8
CT	0.0	2659.9	2261.8	15.0
IC	0.0	1542.9	1208.2	21.7
QL	0.0	911.1	598.5	34.3
SD	0.0325	168.2	149	11.4
SU	0.018	181.1	216.1	-19.3

Table 1: Effects of Agent Advice - 35 SMRs

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. Ten trials per condition were run. Again, we obtain significant reductions in all cost measures except Ship Utility. Comparing percentage reductions with results from the previous experiments, we found that increasing the complexity of the task increases the extent to which the agent is able to help.

Cost	p Value	No Advice Mean	Advice Mean	% Reduction
BP	0.0	283.5	204.6	27.8
CT	0.0	4284.6	3523.7	17.7
IC	0.0	2222.8	1577.6	29.0
QL	0.0	1406.7	817.4	41.9
SD	0.0	189.2	157.5	16.8
SU	0.5178	320.35	330.6	-3.2

Table 2: Effects of Agent Advice - Large Scenario

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.

3.4 Control Condition - Random Advice

An experiment was run to determine the effects of demon accuracy on agent performance. The factors that affect demon accuracy (see Section 3.1) were varied with the surprising result that there was no significant impact 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. 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. As expected, cost measures decrease with increasing frequency of random rerouting. Random rerouting can be used to improve throughput if one is willing to pay for incremental improvements with additional disruption to the initial schedule.

The most important result of this experiment is that the agent performs better than random rerouting when both throughput and the number of reroutes are considered. Regardless of pathology intensity, there existed a level of random rerouting that matched the performance of the agent when measured in terms of throughput. However, at that level of performance the agent always used significantly fewer reroutes. For a given level of throughput, the agent uses a few well directed rerouting decisions rather than large numbers of random ones, thereby doing less violence to the initial schedule.

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. Increasing the amount of random advice seems to help monotonically. The amount of rerouting performed by the agent was statistically equivalent only to the lowest level of random rerouting. In that condition, 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 highest level of random rerouting 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 equaled 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.

3.5 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 (breakbulk), 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, such as sending CONT cargo to a port only equipped to handle GENL, it was penalized with a short delay for the offending piece of cargo. The results are presented below. When a Scheffe' test[2] determines that one of the means at a random level is significantly different from the mean at the agent advice level, an '*' is used to mark the mean at the random level.

Level	**Reroutes**	**CT**	**IC**	**QL**
Real Advice	9.6	1833.0	990.9	488.5
Random 5	4.6 *	2010.2 *	1135.6 *	617.4 *
Random 15	16.6 *	1770.4	961.3	534.5
Random 25	22.4 *	1697.0 *	877.9	443.0
Random 35	31.8 *	1651.0 *	860.7 *	452.8

Table 3: Real vs. Random Advice - 30 SMR Constrained Scenario

Again we see 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.

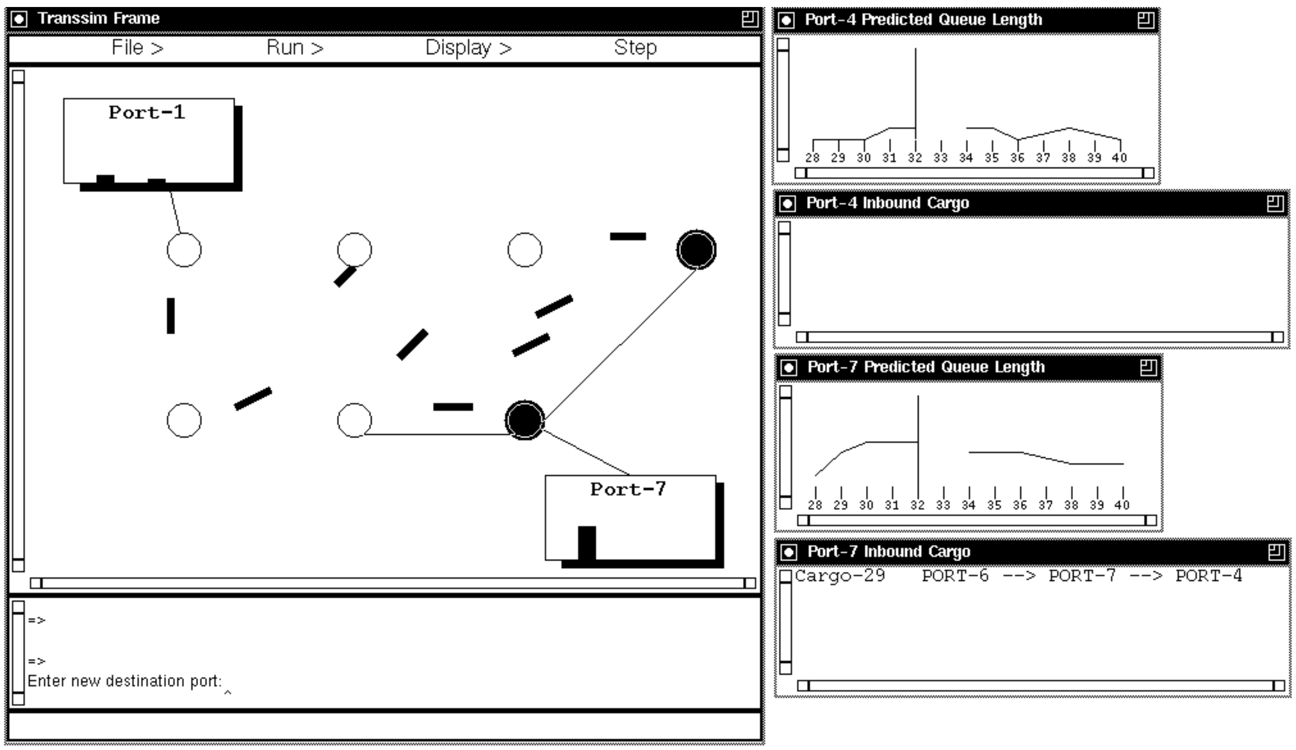
[2] A Scheffe' test is a t-test for multiple pairwise comparisons that preserves a specified experimentwise error.

4 Bringing Humans into the Loop

Part of the motivation for plan steering is the belief that humans find it extremely difficult to perform tasks such as the one for which our agent was designed. Tracking hundreds of events over time and understanding primary and secondary effects of schedule modifications is not something that people do well. Therefore, we ran a series of experiments in which humans were asked to perform the same task at which the agent was previously evaluated. We provided a set of graphical displays that gave the human user essentially the same information and rerouting capabilities available to the agent. In one half of the trials the human worked alone, and in the other half the human and the agent worked together. This experiment design and experimental results are presented below [6].

4.1 Experiment Design

The schedule maintenance agent was designed to increase throughput in TransSim simulations while minimizing schedule disruptions. The goal of this set of experiments is to determine how both an unassisted human and a human working in concert with the agent perform at that task. In each case the human has quick access to roughly the same information and schedule modification capabilities available to the agent. The transportation network is displayed as a connected graph with ship icons moving along the arcs as they traverse simulated channels. The pathology demon's predictions are displayed as a moving graph of queue length vs. simulated day. A sliding window shows both current history of actual queue lengths and predicted queue lengths for several days into the future. One predicted queue length window is on screen for each port during the simulation. Taken together with the network display window, the human user has a simple but informative visualization of the information used by the demon and the agent. A snapshot of the TransSim user interface is shown below in Figure 2.



```
=>

=>
Enter new destination port:
```

```
Cargo-29    PORT-6 --> PORT-7 --> PORT-4
```

Figure 2: TransSim User Interface

Just as the agent makes schedule changes by rerouting cargo, so does the human. An inbound cargo window for each port lists the pieces of cargo that are bound for the port but have not yet been loaded into ships and placed in a channel. Cargo routes may be highlighted and modified by simply clicking on a new destination port.

By monitoring the demon's predictions for each port, it is possible to judge the effects of sending additional cargo. If the predicted queue length at a port is low or is trending downward, then it might be a good idea to let cargo continue to be dispatched to that port. If the predicted queue length at a port is high or is trending upward, then it might be a good idea to reroute cargo around the port to a less congested area. The benefit of rerouting is that the cargo in question will not waste time sitting in a long docking queue and the queue at the bottlenecked port will be given time to clear itself out. It is important to remember that the demon's predictions include some error. A predicted bottleneck may never materialize and the rerouting action may have been wasted.

In one half of the trials the human works alone. In the other half the human has the aid of the schedule steering agent. We call these the *unassisted* and *assisted* conditions respectively. In the assisted condition the agent evaluates the state of the network and generates advice for the user. Advice identifies both a port that is thought to be a potential bottleneck and a piece of cargo bound for that port, and suggests an alternative route. The human evaluates the agent's advice via the various displays described earlier and may decide to accept or reject the advice. In either case the human may implement a rerouting decision of their own construction.

Each of the four participants in the experiment ran ten simulations. The first two simulations were training trials to get the user familiar with the task and the available tools. One training trial was assisted and the other unassisted with the order chosen randomly. Next, two groups of 4 trials were presented where all trials in a group were either assisted or unassisted. Again, that ordering was randomized to counterbalance for possible order effects. A total of six different scenarios were used: the first two were always used for training and the remaining four were presented in both the assisted and unassisted conditions. The order of presentation of the scenarios within a grouping was also randomized.

4.2 Overall Performance

Several measures of cost were recorded for each simulation (see Section 3.2). One-way ANOVA showed a significant main effect of scenario on all of the cost measures. Variance in the structure of the schedules for each of the four scenarios resulted in differing pathology intensities and was ultimately reflected in simulation costs. Therefore, to determine if the presence of agent advice impacted simulation score, we performed two way ANOVA of each cost measure on trial type (assisted or unassisted) and the scenario number. This controlled for variance due to the scenario. There was a significant main effect of trial type on four of the costs: docking queue length, the amount of time that cargo spends sitting idle, the total amount of time that cargo spends in transit, and the number of schedule modifications. All other cost measures were lower in the assisted condition than in the unassisted condition, though they were not significant.

To determine whether agent assistance helped or hurt performance, we used D-tests to compare cost measure means in those conditions.[3] In addition, we let the agent run unhindered on each of the four scenarios, taking its own advice, to see how well it performed. Both assisted and unassisted scores were then compared to means obtained by the agent. The results are presented in the tables below. It is clear from Table 4 that humans working with the help of the agent are able to obtain better throughput than humans working alone. All cost means are lower in the assisted condition although Cargo Transit is not significantly so. Not only does agent assistance result in reduced docking queues and reduced idle time for cargo, but it reduces the amount of time that it takes cargo to travel to its final destination (lower Cargo Transit). This improved performance comes at the expense of disrupting the scenario to a greater extent: on average, about 6 reroutes without assistance, compared to about 12 reroutes with assistance. Since performance is better in the assisted condition, it is not the case that the agent's advice makes things worse and therefore more intervention is required. Apparently, the agent is bringing pathological states to the attention of the human user that they would otherwise have missed and that the human believes require attention. The agent is serving its intended purpose of helping the human track large numbers of events as they occur in a complex environment.

[3] A D-test is a randomization two sample t-test that is robust against deviations from parametric assumptions.

Cost	Assisted	Unassisted	p Value
Queue Length	742.38	828.19	0.0560
Idle Cargo	1366.56	1497.75	0.0240
Cargo Transit	2750.63	2849.44	0.1420
Reroutes	12.0	6.25	0.0010

Table 4: Comparison of Costs in Assisted vs. Unassisted Trials

How does the human's performance in either condition compare to the agent's? We see in Table 5 that the unassisted human performs significantly worse than the agent in all categories. However, the agent implements almost three times as many changes to the scenario. Neither seems to be striking a good balance between maximizing throughput and minimizing schedule disruption. The results in Table 6 tell a different story. The performance of the assisted human is indistinguishable from the agent's performance; none of the cost measures are significantly different. This result alone is interesting since the agent performs quite well. The difference is that the assisted human is able to achieve this feat with significantly fewer changes to the scenario: 12 reroutes for the assisted human compared to more than 18 for the agent. Apparently our mixed-initiative approach to schedule maintenance is working. As noted before, the agent is probably flagging potential pathologies that the human would have otherwise missed and suggesting schedule modifications. However, the human is selectively filtering the suggestions to implement only those that seem most crucial and that are not wasteful.

Cost	Unassisted	Agent	p Value
Queue Length	828.19	682.88	0.0020
Idle Cargo	1497.75	1272.88	0.0010
Cargo Transit	2849.44	2649.56	0.0150
Reroutes	6.25	18.63	0.0000

Table 5: Comparison of Costs in Unassisted Trials vs. Agent

Cost	Assisted	Agent	p Value
Queue Length	742.38	682.88	0.2220
Idle Cargo	1366.56	1272.88	0.1650
Cargo Transit	2750.63	2649.56	0.2050
Reroutes	12.0	18.63	0.0100

Table 6: Comparison of Costs in Assisted Trials vs. Agent

4.3 Evaluating User Decision Points

During an assisted trial, the user is constantly evaluating the state of the network and deciding whether or not to act. We focus on three specific action decisions to determine why the assisted human's performance is so good. They are: the agent offers advice and it is accepted, the agent offers advice and it is rejected, the human makes a rerouting decision independent of the agent. The problem is one of credit assignment. Is good performance due to the intelligence of the agent? Is it due to the human's ability to differentiate between good and bad advice? Or is it due to the human's ability to formulate schedule modifications independently?

The metric we have chosen for this credit assignment task is daily queue length summed over all ports. Every time during the course of a single simulation that the human makes one of the three decisions, we look at total queue length over a window of fixed size in the future to determine if the decision was good or bad.

This is complicated by that fact that there is a heavy trend in queue length. As more and more cargo enters the network, queue lengths grow slowly but steadily to somewhere near the midpoint of the scenario. As cargo leaves the network for final destinations, queue lengths fall off until the scenario ends. The impact of a single user action is easily swamped by the effect of trend. To combat that effect we generated a baseline queue length curve for each of the four scenarios to serve as a standard for expected queue length. That baseline was created by averaging the queue lengths measured for each day over all four of the participants' assisted trials in a scenario and then performing a 3-mean smooth [11]. To score a decision point on a given day in an individual trial, we simply look at future queue lengths in that trial and compare them to future queue lengths in the same time range in the appropriate baseline curve. Subtracting baseline scores from actual scores eliminates trend and gives some idea of performance relative to expected values.

Action	**Mean Difference**	**p Value**
Accept Advice	-0.32	0.1790
Reject Advice	0.583	0.0190
User Modification	-1.02	0.0070

Table 7: Decisions Points in Scenario 3

The results for scenario 3 are shown in Table 7. For each action type we computed actual queue length minus expected queue length and compared the mean of those numbers to a mean of zero. In that way we can determine how the actions affect performance over the course of a single simulation when compared to expectation. Accepting the agent's advice results in smaller than expected queue lengths, but the result is not significant. Rejecting the agent's advice led to significantly larger than expected queues. It appears that in this scenario, the agent's advice tends to stave off potential pathologies and ignoring its advice is detrimental. In terms of making beneficial schedule modifications, the human fares quite well. When compared to expectations, the results of the human's rerouting decisions are significantly better. With the tools that we provided, the human was able to evaluate the state of the transportation network, identify potential trouble spots, and formulate a preventative plan. Therefore, poor human performance in the unassisted trials was not due to an inability to understand and manipulate the domain.

5 Conclusions and Future Work

We have seen that an intelligent agent may be constructed that takes advantage of minimal domain knowledge and that uses simple heuristics to manage a complex problem domain. Such an agent was constructed for schedule maintenance in a simulated transportation network and its performance was evaluated. The presence of the agent significantly reduced cost measures when compared to doing nothing. In addition, the benefit of the agent grew with problem size and complexity. A default rule for managing the domain with random rerouting was explored and shown to have substantial utility in reducing cost. Unfortunately, random rerouting tended to destroy the structure of the scenarios. At a given level of simulation cost, the agent used fewer rerouting actions than the default rule. As problem size and complexity grew this difference became more pronounced.

Simultaneously achieving the two goals of maximizing throughput and minimizing the number of changes to the initial schedule proved to be difficult for both the human and the agent. The human rerouted few pieces of cargo at the expense of high simulation costs. Experimental results indicate that the humans' individual decisions resulted in significantly better than expected performance. Therefore, poor overall performance by human subjects is not due to their inability to understand the domain. The agent's simulation costs were quite low but the number of cargo rerouting decisions was high. The optimal balance was struck by the agent and the human working together. The agent enhanced the human's ability to identify potential pathologies in a complicated environment, and the human evaluated and filtered away schedule modifications with dubious utility that were suggested by the agent.

The goal of this research is to arrive at a generalizable architecture for plan steering that scales up to the demands of large, complex planning problems. We want to be able to replace TransSim with the real world and have agents working with humans to avoid pathologies in plans and schedules. To that end, we will continue to push on this system by investigating pathologies other than bottlenecks, advice other than rerouting, and methods for increasing predictive accuracy. We want to develop explanatory theories for why our rerouting heuristic works so well in this domain and for why bringing humans into the loop has such a dramatic impact on performance. We then hope to study other problem domains to understand how they are different from transportation planning and how those differences impact the efficacy of our architecture.

Acknowledgments

This research is supported by ARPA/Rome Laboratory under contract #F30602-91-C-0076. The US Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

References

[1] Ambros-Ingerson, J. A. and Steel, S. Integrating planning, execution and monitoring. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 83-88, Minneapolis, Minnesota, 1988.

[2] Anderson, S.D., Carlson, A., Westbrook, D.L., Hart, D.M. and Cohen, P.R. CLASP/CLIP: Common Lisp Analytical Statistics Package/Common Lisp Instrumentation Package. Department of Computer Science Technical Report 93-55, University of Massachusetts, Amherst.

[3] Lopez-Mellado, E. and Alami, R. A failure recovery scheme for assembly workcells. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 1, pages 702-707, 1990.

[4] Muscetolla, N. and Smith, S.F. A probabilistic framework for resource-constrained multi-agent planning. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pp. 1063-1066, Morgan Kaufmann, 1987.

[5] Oates, T. and Cohen, P.R. Toward a plan steering agent: experiments with schedule maintenance. Department of Computer Science Technical Report 94-02, University of Massachusetts, Amherst. Submitted to AIPS-94.

[6] Oates, T. and Cohen, P.R. Humans plus agents maintain schedules better than either alone. Department of Computer Science Technical Report 94-03, University of Massachusetts, Amherst. Submitted to AAAI-94.

[7] Ow, P.S., Smith, S.F. and Thiriez A., Reactive plan revision. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pp. 77-82, Morgan Kaufmann, 1988.

[8] Sadeh, N. Micro-opportunistic scheduling: the micro-boss factory scheduler. To appear in *Intelligent Scheduling*, edited by M. Zweben and M. Fox, Morgan Kaufmann, 1993.

[9] Simmons, R.G. A theory of debugging plans and interpretations. In *Proceedings of the Seventh National Conference on Artificial Intelligence*, pages 94-99, Minneapolis, Minnesota, 1988.

[10] Smith, S.F., Ow, P.S., Muscetolla, N., Potvin, J., and Matthys, D.C. An integrated framework for generating and revising factory schedules. In *Journal of the Operational Research Society*, Vol. 41. No. 6, 1990.

[11] Tukey, J.W. *Exploratory Data Analysis.* Addison-Wesley Publishing Company, 1977.

[12] Wilkins, D.E. Recovering from execution errors in SIPE. Technical Report 346, Artificial Intelligence Center, Computer Science and Technology Center, SRI International, 1985.