TRAINS-95: Towards a Mixed-Initiative Planning Assistant

George Ferguson and **James Allen** and **Brad Miller**
Department of Computer Science
University of Rochester
Rochester, NY, USA, 14627-0226
{ferguson,james,miller}@cs.rochester.edu
http://www.cs.rochester.edu/research/trains/

Abstract

We have been examining mixed-initiative planning systems in the context of command and control or logistical overview situations. In such environments, the human and the computer must work together in a very tightly coupled way to solve problems that neither alone could manage. In this paper, we describe our implementation of a prototype version of such a system, TRAINS-95, which helps a manager solve routing problems in a simple transportation domain. Interestingly perhaps, traditional planning technology does not play a major role in the system, and in fact it is difficult to see how such components might fit into a mixed-initiative system. We describe some of these issues, and present our agenda for future research into mixed-initiative plan reasoning. At this writing, the TRAINS-95 system has been used by more than 100 people to solve simple problems at various conferences and workshops, and in our experiments.

Introduction

In command and control situations and logistics planning, a human planner faces several difficult problems. First, there is a surplus of data, only a small amount of which is actually information relevant to the current task. In fact, what data is relevant cannot be determined in advance and only becomes clear as the situation and the plan develop. Second, the plans being considered are large and complex, and it is beyond human capabilities to manage all the details effectively. Automated planning systems are better able in principle to handle the scale, but are hard to apply because of the under-specified initial situations, and the fact that many planning decisions are made on an intuitive basis that cannot be effectively quantified. As a result, neither the human or the computer can effectively solve such planning problems in isolation. This problem motivates an interest in mixed-initiative planning systems, where the computer acts as a collaborating assistant to the human, anticipating needs, performing the tasks it is well suited for, and leaving the remaining tasks to the human. By cooperating, the human-computer "team" may be able to effectively deal with problems that neither could handle alone.

To explore these aspects of mixed-initiative planning, we have built a prototype system, TRAINS-95, that helps a manager solve routing problems in a simple transportation domain. The manager is presented with a map displaying cities and rail connections between them, as shown in Figure 1. The system generates random problems that require planning routes for a set of engines to a set of destinations. Various environmental factors can arise during the interaction, which the manager and system must then plan to accommodate. The system can interact using both spoken and typed English, as well as graphical displays and controls.

Our goal was a robust, modular, multi-modal, mixed-initiative planning assistant. To be more specific, by "robustness" we mean that no matter what occurs during the interaction, the system not only doesn't crash, but does something to indicate its understanding of the manager's intentions and its own attempts to further the plan. By "modular" we are taking seriously the idea that there are, or will be shortly, a variety of knowledge sources, reasoning agents, and display engines available as resources that the system can employ. Examples include weather information, newsfeeds, map servers, and so on, as well as "off-the-shelf" technology such as speech recognizers and generators. As we will describe, many of the components of TRAINS-95 are themselves autonomous modules connected by a general-purpose message-passing process manager.

By "multi-modal" we mean that there are a multitude of ways of communicating between humans and computers, include speech input and output, written text, and graphical displays such as maps, charts, forms, and the like, for both input and output. From the outset, we intended that TRAINS-95 should accommodate all of these, allowing the human to interact in whatever way seems most natural to them. In some situations existing procedures may impose constraints on what form communication can take (*e.g.*, an operator may be trained on a particular type of display). In other cases determining the appropriate modality is an ongoing process, and the subject of current research.

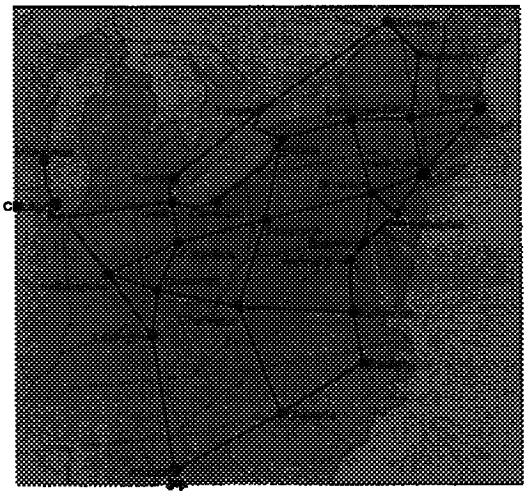

Figure 1: TRAINS-95 System Map Display

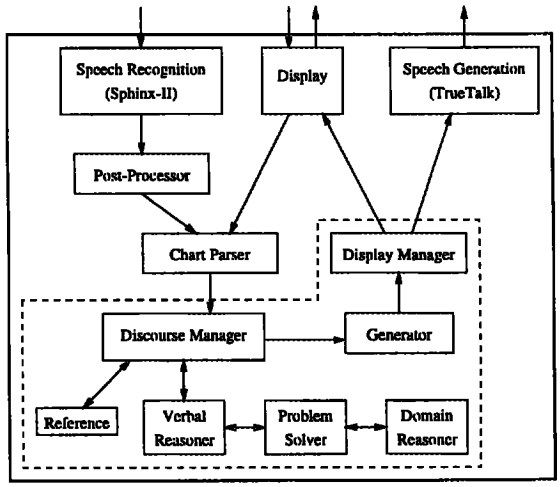

Figure 2: TRAINS-95 System Architecture

By treating all modalities as *linguistic*, that is, as a form of language, we obtain a powerful unifying model of the interaction as a form of *dialogue*. Of course, multi-modal communication raises a new set of issues for all aspects of the system, as we will discuss later.

Finally, by "mixed-initiative" we mean that both the system and the human are on roughly equal ground as participants in the dialogue. This is not to say that they are equals, since clearly they are good at different things. But in a truly mixed-initiative system, both participants can do what they do best. The human typically has knowledge of the high-level goals and means of achieving them, while of course, the computer is good at managing the multitude of low-level details that go into such plans. As well, in realistic situations, the human may be guided by principles that are difficult or impossible to quantify precisely, making it impossible to fully automate the task and replace them. Even if this were possible in principle, the resulting master-slave style of communication would hardly be natural and probably wouldn't be very efficient.

The next section describes our current implementation, TRAINS-95, including design goals, our approach to the aspects outlined above, and lessons learned. The key points are: (1) that dialogue provides the context required for successful interaction independent of input modality; and (2) that the plan reasoning requirements of mixed-initiative systems differ markedly from the specifications of traditional planners. We elaborate on these points in the following section. In the final section we present some of the many research issues raised by our work on TRAINS-95, and describe the directions we are pursuing in addressing them.

The TRAINS-95 System

The TRAINS-95 system is implemented as a set of independent modules, as shown in Figure 2. The modules shown in the dashed box are implemented within a single Lisp program, otherwise each module is a separate Unix process, and modules communicate by exchanging messages. We have developed a general-purpose Unix process manager that manages the various processes and their communication. It performs the mapping between logical module names and physical processes, and provides services such as broadcast messages and complete logging and replay of message exchanges. It is itself controlled by messages, for example to create a new process, restart or kill a module, register interest in another module, and so on.

In building the current system, we have concentrated on the interface and infrastructure components, in order to get a system that naive users could interact with as soon as possible. In this section we will describe the various modules and their interactions, concentrating on those aspects central to the goal of a robust, multi-modal application. The next section provides an example of the system in action and discusses the implications for mixed-initiative planning systems.

Speech and Display Modules

For speech input in TRAINS-95, we use the Sphinx-II recognizer from CMU (Huang *et al.* 1992), trained on the ATIS airline reservation corpus, with a vocabulary of approximately 3000 words. We deliberately did not build a special recognizer for our task or domain, since we wanted to investigate using off-the-shelf components directly as "black boxes" in our system. Of course, this adversely impacts the rest of the system, since many of the words in the ATIS corpus are meaningless in the simple transportation domain we are dealing with in TRAINS-95 (*e.g.*, "DC3," "Riyadh," and "movie"). We considered this an appropriate test of robustness, and something which any system that intends to use such "plug-and-play" components must

ultimately address. In this particular case, the output of the module is simple enough, namely a sequence of words (and control messages for when a hypothesis needs to be revised), that the semantics of its messages is obvious.

To compensate for the under-constrained speech recognizer, its output is post-processed by a domain- and/or speaker-dependent module (Ringger 1995). This module uses techniques from machine translation to "correct" the output of the recognizer, based on a statistical model derived from a corpus of previous runs of the system.

The speech generation component of TRAINS-95 is built on the TrueTalk generator from Entropics, Inc. This program provides high-quality speech generation and a wide range of intonational and inflectional controls.

Although speech is the most exciting and challenging modality, other modalities are provided by an object-oriented display module. For input, it allows typed input, menu selections, and direct mousing on display elements, all of which are broadcast to other modules. Output to the display is performed using a fairly rich object-oriented language, allowing a variety of graphical elements to be defined and displayed. The current version of the display supports object types such as "engine," "track," and "region;" in the immediate future this will be generalized even further. Several types of dialog box can be popped up (under the control of the language components), allowing yet another input modality.

Language and Dialogue Processing

The core of the TRAINS-95 system is the set of modules that perform language understanding and dialogue processing tasks. The first of these is a robust parsing system based on a bottom-up chart parser (Allen 1994). The parser uses a fairly comprehensive lexicon of 1700 words and grammar of 275 rules, both built from the TRAINS dialogue corpus (Heeman & Allen 1995). The parser accepts input from all the input modules—we treat spoken, typed, and moused input uniformly as linguistic communication. The parser's output is a set of *speech acts* expressing the content of the utterance. Since the majority of utterances in spoken dialogue are not complete syntactic sentences, the traditional chart parser is augmented with a set of *monitors*. If a complete parse is not found, these monitors scan the chart and attempt to piece together a partial representation of what was said. This approach to natural language understanding is crucial to meeting the goal of a robust system. It also means that the system can often interact effectively even at very low speech recognition rates.

After the parser has determined a set of possible speech acts for the utterance, the reference module deals with grounding any terms to the objects in the domain (*e.g.*, "to Chicago, " "send it"), or turning indefinite references into a query to be resolved later (*e.g.*, "an engine").

The verbal reasoner is then responsible for interpreting the speech acts in context, querying various knowledge sources in order to understand or address the speech act, formulating responses, and maintaining the system's idea of the state of the discourse. The verbal reasoner architecture is based on a set of prioritized rules that match patterns in the input speech acts. These rules again allow robust processing the face of partial or ill-formed input. The presence of an element such as "from Avon" may allow it to determine that the user is requesting a plan incorporation. These rules rely heavily on the *discourse context*, that collection of knowledge and beliefs that describe the state of the interaction. In fact, it is precisely this context that makes dialogue-based interaction so much more flexible and powerful than one-shot transaction-based communication.

Speech acts that appear to be requests to incorporate a constraint on a prior system action or to extend the current plan are passed on to the problem solver. This module is also based on a prioritized set of rules, this time for dealing with user responses to system questions or with plan corrections and extensions. These rules turn the information into queries and updates to a domain reasoner, *e.g.*, for a path that fits the constraints "from Chicago to New York via Pittsburgh." If fragmentary information is supplied from the verbal reasoner (which will generally try to fill it out with information from the discourse context), then the problem solver attempts to incorporate the fragment into what it knows about the current state of the plan. For instance, if the prior action had been to propose a route, and the verbal reasoner did not indicate in its request to the problem solver that the current request is part of a rejection, then it will attempt to select another goal the user may have informed it of, and incorporate the fragment with respect to that goal (presuming the fragment does not contain information that indicates a goal shift).

The output of the verbal reasoner is a set of speech acts that the system wants to communicate to the user. The generation module decides how to actually achieve the communication using either the display or the speech generator, or both. This module also uses prioritized rules to match requested speech acts to appropriate means of expressing them. For instance, a request to the generator to warn the user of track repair along the route between Chicago and Toledo will result in telling the display to flash the track segment red, and simultaneously speak the phrase "the track between Chicago and Toledo is currently undergoing repair." The generation module (along with reference) maintains a representation of what the system believes the user already knows, to prevent overly verbose or confusing output.

The verbal reasoning components of TRAINS-95 at-

tempt to provide a robust agent that keeps the dialogue going. The approach is generally to instantiate the current plan as soon as possible, communicate the results, and allow the user to correct or elaborate it. This "look then leap" strategy will be further described in the next section.

Domain Reasoning

The domain reasoner in TRAINS-95 maintains a knowledge base describing the state of the world and provides planning and plan recognition services to the language modules. For the simple route-planning domain, of course, it would be easy to build a perfect reasoner that solved the problems as soon as the manager had stated their goals. However, not only is it unlikely that we will ever be able to build such a reasoner for a realistic domain, in the next section we claim that such a system is not necessarily appropriate for mixed-initiative planning. We therefore deliberately weakened the TRAINS-95 domain reasoner in order to force the manager to interact in order to overcome its shortcomings. The route planner can therefore only plan route segments less than four hops long, and for those it chooses a random path. The knowledge base maintains an accurate view of the map, and allows various "natural" events such as bad weather or track maintenance to arise during the interaction. These also force interaction in order to revise plans to take account of them.

An important aspect of the type of domain reasoning that is performed in TRAINS-95 is that it is highly incremental and incorporates aspects of both planning and plan recognition in a tightly-coupled way. For example, the domain reasoner may be asked to incorporate a new constraint on an existing plan, *e.g.*, that it go via a particular city. The domain reasoner must first recognize how that constraint fits into the plan (a task shared with the language modules, for example in determining which route is being modified). It then adds the constraint to the plan, possibly removing other softer constraints (such as to try an avoid cities known to be congested). It must then plan a new route satisfying the current constraints, preferably one that makes the smallest possible change to the plan already under consideration.

This model, it seems to us, is quite unlike the traditional view of planning, where a goal is specified (completely) and the planner goes off and produces a plan to achieve it. This raises a question as to the role of traditional planning in mixed-initiative systems. The next section elaborates on this point.

Where's the Planning in Mixed-Initiative Planning?

Figure 3 presents a example of an actual dialogue between a user and the TRAINS-95 system, starting from the situation shown in Figure 1. The task is admit-

```
1.  M: I need to get to Detroit, Syracuse,
       and Montreal.
2.  S: Ok. [highlights the three cities]
3.  M: Let's send the trains from Chicago to
       Detroit.
4.  S: Ok. [draws route from Chicago to Detroit
       via Toledo] But the terminal at Toledo is
       delaying traffic due to bad weather.
       Trains will take an additional five hours
       to move through there.
       [highlight Toledo red]
5.  M: That's ok. Now let's go from Atlanta
       up to Syracuse.
6.  S: Please tell me a route to use to get
       from Atlanta to Syracuse. [highlight
       engine at Atlanta and city Syracuse]
7.  M: Go via Charleston and Pittsburgh.
8.  S: Ok. [draws route via Cincinnati]
       But traffic through Cincinnati is
       being delayed due to construction.
9.  M: Ok, let's go via Charleston and Buffalo
       instead.
10. S: No problem. [draws route]
11. M: That's great.
12. S: I hope it was helpful.
13. M: Finally, I want to go to Montreal
       from Philadelphia.
14. S: Ok. [draws route]
15. M: That's great. I'm done.
```

Figure 3: Actual TRAINS-95 Dialogue

tedly very simple and, as we noted previously, the system's planning abilities were deliberately limited in order to encourage interaction. The point of the dialogue is to emphasize that this is indeed a mixed-initiative *planning* system. The user and the system are constructing, evaluating, and correcting plans, albeit simple ones. The environment is dynamic, with problems such as weather and traffic occurring independent of the planner. The human manager's goals and preferences need not be explicit, as when they choose to ignore the problem at Toledo. The system has clear limitations, as it would in any realistic domain, and it understands how to deal with these by interacting with the manager.

Although it's doing planning, however, a mixed-initiative planning system isn't doing what we might recognize as "traditional" planning, that is, constructing a sequence of operators from a fully-specified initial situation to a stated goal. In fact, in an informal analysis of one hour of human-human problem-solving dialogues (part of a larger eight hour study (Heeman & Allen 1995)), we found that a relatively small percentage of the utterances, 23%, dealt with explicitly adding or refining actions in the plan. Figure 4 summarizes this analysis. Note the importance of being able to explicitly evaluate and compare options, even between humans of roughly equal ability. In human-

Evaluation/comparison of options	25%
Suggesting courses of action	23%
Establishing world state	13%
Clarifying/confirming communication	13%
Discussing problem solving strategy	10%
Summarizing courses of action	10%
Identifying problems/alternatives	7%

Figure 4: Analysis of one hour of human-human problem-solving dialogues

computer interactions, we would expect this number to increase, as the computer can perform more and larger analyses. Similarly, even in the very simple TRAINS world, a significant portion of the communication was concerned with establishing the state of the world, and we would expect this to increase in a more realistic environment. Similar conclusions about the nature of interactive planning are presented in (Ferguson 1995; Pollack 1992).

Why is it that mixed-initiative planning seems to involve so little traditional planning (finding courses of action from initial situation to goal)? In the first place, it is because the initial situation is usually incompletely specified, whether because of changing conditions as in our simple example or in a more realistic setting because it is simply too huge to represent. Rather, the determination of what aspects of the world are relevant to the plan is a dynamic process that is intimately connected with the exploration and elaboration of the plan. It would be impossible in practice to extract such criteria from the human manager offline.

Similarly, the goals of the plan in the mixed-initiative setting are also poorly specified. Not only do they change over time as the manager's preferences and concerns change, but they typically cannot be extracted and that knowledge codified into the system's operation. Of course, in our simple example the only goal the manager can have is to get trains to their destinations, so the system can make simplifying assumptions. But even so, treating human-imposed constraints on the plan as part of the "goals," the reasons for rejecting a particular plan may not be made explicit to the system.

The upshot of this is that even if we had implemented a "perfect" route planner for the simple TRAINS-95 domain, we would still need all the other components of the system. If the goal is a natural, interactive planning assistant, the solution will not be found in the language of traditional planning. The question becomes how to incorporate traditional systems that are good at what they do but relatively inflexible and brittle within the context of a robust, mixed-initiative system.

We have developed a model of the mixed-initiative planning process from analysis of the TRAINS dialogues and implemented a simple version of it in TRAINS-95. This model consists of four steps:

A. Focus: Identify the goal/subgoal under consideration.

B. Gather Constraints: Collect constraints on the solution, selecting resources, gathering background information, and adding preferences.

C. Instantiate Solution: As soon as a solution can be generated efficiently, one is generated.

D. Criticize, Correct or Accept: If the instantiation is criticized and modifications are suggested, the process continues at step (B). If the solution appears acceptable (given the current focus), then we continue at step (A) by selecting a new focus.

At first glance, this model seems quite similar to the expand-criticize cycle found in hierarchical planners since Sacerdoti (Sacerdoti 1977). The significant difference is in step (C). Rather than pursuing a least commitment strategy and incremental top-down refinement, we "leap" to a solution as soon as possible. You might call this a "look then leap" strategy rather than the traditional "wait and see" strategy used in least-commitment planning. As such, our strategy shares similarities with Sussman's original work with an emphasis on plan debugging and repair (Sussman 1974). When the plan must be a result of a collaboration, it seems that it is more efficient to throw out a solution and criticize it, rather than to work abstractly as long as possible. Notice that this is exactly why committees often start with a "straw man" when trying to develop new policies. This helps the committee focus its discussion, just as the the plans under discussion in mixed-initiative planning provide part of the context within which the manager is exploring the problem.

To see how the "look then leap" model fits well with the discourse strategies used in our dialogues, consider the simple TRAINS-95 dialogue in Figure 3 again. In (1), the manager clearly sets out a set of goals, limiting but not fully specifying the focus (step A). Statement (3) implicitly focuses on the goal of getting to Detroit, and starts adding constraints (step B). With statement (4), the route planner generates a "straw plan" (step C), and then the system offers its own criticism (step D). The display at this point is shown in Figure 5. This self-criticism might seem strange at first glance, but consider that the system has not been informed of any evaluation criteria for the goals, so does not know whether the time for the trip is a critical factor. In (5), the manager indicates that the delay is not a problem by accepting the straw plan, and then moves on to a new focus (implicitly) by adding constraints relevant to the subgoal of getting to Syracuse (steps A and B). The system does not have enough constraints to create a straw plan solution, so (6) prompts for additional constraints. Statement (7) specifies these additional constraints (step B), and statement (8) proposes a straw

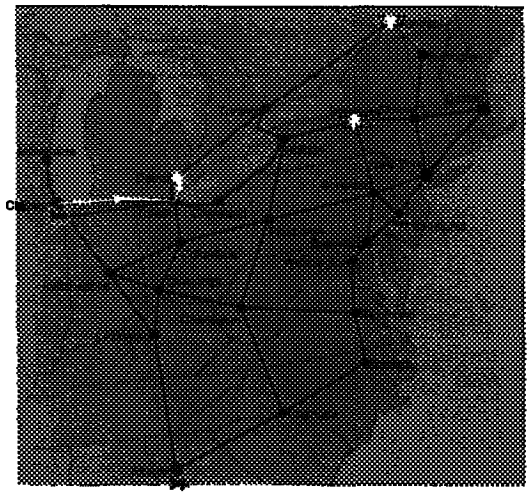

Figure 5: TRAINS-95 System Map Display after (4)

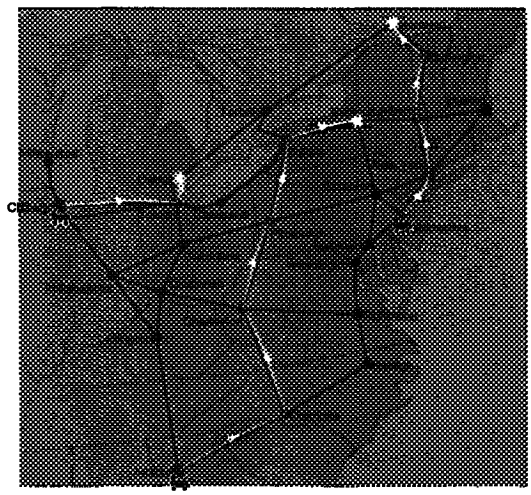

Figure 6: Final TRAINS-95 System Map Display

plan route (step C) and potential criticisms (step D). In this case, the manager adds additional constraints in (9) that address the problems the system mentioned and a new solution is generated as part of interaction (10). The rest of the dialogue continues in much the same way and doesn't raise any substantially new issues. The final display is shown in Figure 6.

To support this model of mixed-initiative planning, we are developing a four layer architecture that generalizes the TRAINS-95 system architecture. This is shown in Figure 7. The *discourse level* maintains information important for reference identification and for speech act interpretation and generation. It maintains what objects have been recently mentioned as well as the immediate discourse situation including what speech act was performed last and what discourse obligations each agent has. While critical to the overall system, the other three levels are more centrally concerned with the planning process.

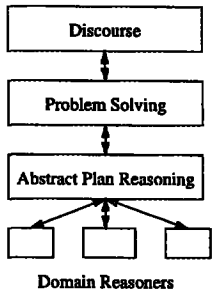

Figure 7: Mixed-initiative planning architecture

The *problem solving level* maintains meta-level information about the problem solving tasks, similar to the problem solving level actions described in (Litman & Allen 1987) and (Lambert & Carberry 1991). Actions at this level are problem solving actions such as "solve goal directly," "decompose goal into subproblems," "resolve resource conflict," and so on. Note that we need explicit representations of such actions because they can be discussed in the dialogue. Either agent might request the other to do one of these actions, or discuss how one might do the action, or assign general responsibility to an agent to perform this type of action throughout the interaction. This level supports processes such as identifying the task desired (*e.g.*, distinguishing between an elaboration of the plan for the current goal and shifting to another goal). It does this by maintaining an abstract tree of goals and information on which part of the problem is currently in focus. It also supports the process of determining the scope of problem solving actions such as cancelations and modifications. By maintaining a history of previous problem solving states, it supports intelligent "undo" commands and the explicit discussion of the history of the problem solving interaction. Finally, it supports discussion of the problem solving strategy to be used, and can maintain ordering constraints on when certain problem solving tasks should be performed.

The problem solving level reasons at the meta-level about the plans under consideration. These plans are themselves represented in terms of an *abstract plan representation level.* This representation captures aspects of plans independent of the details of reasoning about them. It can perform operations such as plan recognition in cases where deep domain reasoning is unnecessary, *e.g.*, identifying which plan achieves a particular goal. It provides a uniform representation of plans suitable for use by other components of the system, such as the problem solving, discourse, and generation components.

When more sophisticated domain-level reasoning is required, we rely on the services of a set of *domain reasoners*. The abstract plan representation level manages the interface between the mixed-initiative system and these various domain specialists, matching open issues with reasoners and coordinating the responses. These reasoners might be, for example, a scheduler or an Internet agent that can retrieve information. They are likely to include traditional planning technologies that are invoked in a context that is sufficiently well-specified for them to function effectively. We call these components "specialists" to emphasize that they need to be quick and effective in a well-defined domain, although they definitely need not always produce "perfect" answers.

The key to the integration of such components is the ability to specify and reason about their capabilities. That is, the abstract plan reasoner needs to be able to reason about which specialists might be suitable for what tasks, and interpret their results. This is complicated by the desire to use existing components "off the shelf" as much as possible. We are currently looking at using KQML (Finin *et al.* 1994) for part of this purpose, as well as other work on specifying agent capabilities (Tate 1995).

Current and Future Directions

Our experiences with TRAINS-95 have opened up many research directions that we are currently pursuing, and this paper has already discussed several of them. In this section we provide a brief survey of some additional issues for current and future work.

First, although we have argued strongly for an interactive, language-based approach to plan reasoning, this is not in itself enough. In fact, our strategy of attempting to instantiate the plan and then discuss and correct it will need to be generalized as the plans become larger and more detailed. This problem of plan presentation is pervasive, and requires work on many aspects of multi-modal interaction, since clearly drawing a huge graph with Lisp-like labels is not going to be sufficient. We believe that the dialogue-based approach provides the context in which presentation decisions can be made effectively. The TRAINS-95 system provides the infrastructure for examining the multi-modal aspects of such communication.

Second, and something of a corollary to this, is the problem of the semantics of external displays or data sources. That is, many systems are currently being proposed to make use of existing resources, such as weather servers or radar displays. However, while it's one thing to fetch the latest weather map via the Internet and display it, it's quite another to be able to discuss what is being displayed intelligently. We have emphasized how context is the big benefit of dialogue-base systems—the other side of the coin is that to make use of something in a dialogue requires that it fit into the context. Part of the solution to this lies in the type of capability descriptions that we discussed in the previous section. We are also moving towards use of a uniform display language, via a translator if necessary, in order to be able to understand exactly what is being displayed where. The impact of this on the ability to "plug and play" components has yet to be seen.

Third, concentrating on the type of planning that goes on in mixed-initiative systems, it is clear that much of it can be seen as replanning, in the sense of debugging a plan ala Sussman (Sussman 1974). We are looking at the various approaches to replanning (*e.g.*., (Kambhampati 1989; Kambhampati & Hendler 1992)), but to some extent these seem to be burdened by the same constraints that make traditional planning systems unsuitable for direct use in the mixed-initiative situation.

Fourth, we have been concerned from the outset with the evaluation of our work, that is, how to know if we are making progress. This has traditionally been a problem for interactive systems, and to some extent for planners. Our current system has built into it a simple set of task-related metrics that are recorded for each interaction. As we refine those metrics, we can explore whether particular strategies are better than others at getting the task done, or whether the presence of certain components helps or hinders performance.

In recent work (Allen *et al.* 1996), we have evaluated the effects of input mode (speech vs. keyboard) on task performance and explored user input-mode preferences. Task performance was evaluated in terms of the amount of time taken to arrive at a solution and the quality of the solution was determined by the amount of time needed to travel the planned routes. Sixteen subjects for the experiment were recruited from undergraduate computer science courses. None of them had previous experience with the system, and in fact only five reported that they had previously used a speech recognition system. A preliminary analysis of the experimental results shows that the plans generated when speech input was used were of similar quality to those generated when keyboard input was used. However, the time needed to develop the plan was significantly lower when speech input was used. Averaging over five scenarios, problems were solved using speech in 68% of the time using keyboard, despite speech recognition rates averaging only about 75%! Although still rough, we take these results to support our approach to building robust interactive systems by treating the interaction as a dialogue.

The point as far as mixed-initiative planning is concerned is that with only a two-minute training video, and despite low speech recognition rates and the prototypical status of the system, 73 of the 80 sessions resulted in successful completion of the route planning task. We suspect that this is probably a record for naive users of AI planning systems. Of course, the task was very simple, and it remains to be seen whether we can maintain this level of effectiveness as we move into

a more complex planning domain.

Finally, we are continually improving the TRAINS system. The next version will include many of the ideas presented in these last sections of the paper, as well as a richer transportation domain involving cargoes and different vehicle types. This will increase the complexity of the planning task, as we noted above was necessary for further evaluation. Several parts of the TRAINS-96 system have already been re-implemented, including new infrastructure components, KQML message-passing, and the separation of many of the functions of the "discourse manager" into separate modules. Work is currently proceeding on separating problem solver and domain reasoner functionality, and increasing the complexity of the domain. Our goal is to always have the most recent version of the system running and available for demos, and we hope to have a version of it running over the World-Wide Web.

Conclusion

The TRAINS-95 system is a concrete first step towards a mixed-initiative planning assistant. We have demonstrated the feasibility of the dialogue-based approach to interactive planning, and have developed a substantial infrastructure for future research.

The TRAINS-95 system is being developed as part of the TRAINS Project (Allen *et al.* 1995), a long-term research effort to develop intelligent assistants that interact with their human managers using natural language. More information is available via the World-Wide Web at URL: `http://www.cs.rochester.edu/research/trains/`.

Acknowledgements

This material is based upon work supported by ARPA – Rome Laboratory under research contract no. F30602-95-1-0025, the U.S. Air Force – Rome Laboratory under research contract no. F30602-91-C-0010, and by the Office of Naval Research under research grant no. N00014-95-1-1088. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

References

Allen, J. F.; Schubert, L. K.; Ferguson, G.; Heeman, P.; Hwang, C. H.; Kato, T.; Light, M.; Martin, N. G.; Miller, B. W.; Poesio, M.; and Traum, D. R. 1995. The TRAINS project: A case study in building a conversational planning agent. *Journal of Experimental and Theoretical AI* 7:7–48. Also available as University of Rochester, Dept. of Computer Science TRAINS Technical Note 94-3.

Allen, J. F.; Miller, B.; Ringger, E. K.; and Sikorski, T. 1996. Robust understanding in a dialogue system. Submitted for publication.

Allen, J. F. 1994. *Natural Language Understanding.* Benjamin Cummings, 2nd edition edition.

Ferguson, G. 1995. *Knowledge Representation and Reasoning for Mixed-Initiative Planning.* Ph.D. Dissertation, Department of Computer Science, University of Rochester, Rochester, NY. Available as Technical Report 562.

Finin, T.; Fritzson, R.; McKay, D.; and McEntire, R. 1994. KQML as an agent communication language. In *Proc. Third International Conference on Information and Knowledge Management (CIKM-94).* ACM Press.

Heeman, P. A., and Allen, J. F. 1995. The TRAINS-93 dialogues. TRAINS Technical Note 94-2, Dept. of Computer Science, University of Rochester, Rochester, NY.

Huang, X.; Alleva, F.; Hon, H.-W.; Hwang, M.-Y.; and Rosenfeld, R. 1992. The SPHINX-II speech recognition system: An overview. Technical Report CS-92-112, Carnegie Mellon University, School of Computer Science, Pittsburgh, PA.

Kambhampati, S., and Hendler, J. A. 1992. A validation-structure-based theory of plan modification and reuse. *Artificial Intelligence* 55:193–258.

Kambhampati, S. 1989. *Flexible reuse and modification in hierarchical planning: A validation structure based approach.* Ph.D. Dissertation, Department of Computer Science, University of Maryland, College Park, MD. Available as CS Technical Report 2334.

Lambert, L., and Carberry, S. 1991. A tripartite plan-based model of dialogue. In *Proc. Twenty-Ninth Annual Meeting of the Association for Computational Linguistics*, 47–54. Berkeley, CA: University of California.

Litman, D., and Allen, J. F. 1987. A plan-recognition model for subdialogues in conversations. *Cognitive Science* 11(2).

Pollack, M. E. 1992. The uses of plans. *Artificial Intelligence* 57.

Ringger, E. K. 1995. A robust loose coupling for speech recognition and natural language understanding. Technical Report 592, Dept. of Computer Science, University of Rochester, Rochester, NY.

Sacerdoti, E. 1977. *A Structure for Plans and Behaviour.* New York, NY: Elsevier, North-Holland.

Sussman, G. J. 1974. The virtuous nature of bugs. In *Proc. First Conference of the Society for the Study of AI and the Simulation of Behaviour.* Brighton, UK: Sussex University.

Tate, A. 1995. The O-Plan knowledge source framework. ARPA/RL O-Plan TR 21, Artifical Intelligence Applications Institute, University of Edinburgh, Edinburgh, UK.