Working Together: Centralized Command Sequence Generation for Cooperating Rovers

Gregg Rabideau, Tara Estlin, Steve Chien, Anthony Barrett
Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, CA 91109
818-393-5364
{firstname.lastname}@jpl.nasa.gov

Abstract—In this paper, we describe how rover command generation can be automated to increase robustness and help relieve some of the burden on human operators. We describe the issues inherent in the operations planning for a distributed set of rovers, and how they can be addressed using a centralized framework that is a generalization of single rover paradigm. Finally, we describe a prototype system for automatically generating low-level commands to achieve high-level goals for a set of rovers simulated on a terrain of geological interest.

TABLE OF CONTENTS

1. INTRODUCTION

Landmark events have recently taken place in the areas of space exploration and planetary rovers. The Mars Pathfinder and Sojourner missions were major successes, not only demonstrating the feasibility of sending rovers to other planets, but displaying the significance of such missions to the scientific community. Upcoming missions are currently being planned to send robotic vehicles to Mars (Mars01, Mars03, Mars05) as well as the outer planets and their moons. In order to increase science return and enable advanced concepts, new missions are being proposed to employ larger sets of robotic workers. Whether it is an unmanned spacecraft with remote rovers, or a mix of humans and robotic assistants, the command and control task for these machines will be complex. While manual request and sequence generation was possible for Sojourner, new missions will need to automate much of this process.

Mission designers would need to determine the optimal number of rovers for a given mission. However, sending multiple rovers has at least three advantages: force multiplication, simultaneous presence and system redundancy.

- *Force multiplication*. Multiple rovers can collect more data than a single rover can. Designing a single rover to withstand harsh environments for long periods of time has proven difficult and costly. Several rovers can cover a larger area in a shorter time, with designated points of interest allocated over the team of rovers. We call these *cooperative* tasks.
- *Simultaneous presence*. Multiple rovers can perform tasks that are impossible for a single rover. We call these *coordinated* tasks. Certain types of instruments, such as interferometers, require simultaneous presence at different locations. Rovers landed at different locations can cover areas with impassable boundaries. Using communication relays, a line of rovers can reach longer distances without loss of contact. More complicated coordinated tasks can also be accomplished, such as those involved in hardware construction or repair.
- *System redundancy*. Multiple rovers can be used to enhance mission success through increased system redundancy. Several rovers with the same capability may have higher acceptable risk levels, allowing one rover, for example, to venture farther despite the possibility of not returning.

In all cases, the rovers can behave in a cooperative or even coordinated fashion, accepting goals for the team, performing group tasks and sharing acquired information.

In our approach, we use Artificial Intelligence (AI) planning and scheduling techniques to automatically generate appropriate low-level rover command sequences to achieve science goals. These planning techniques have been integrated with a science analysis tool, and a multi-rover and terrain simulator. We focus on a geological survey scenario that involves cooperative science tasks.

First, the science analysis tool using a clustering algorithm generates a set of high-level goals for examining and analyzing terrain objects. Then, the goals are communicated to the automated scheduler. From the goals and initial rover state information, the automated scheduler generates a sequence of low-level commands that satisfies the goals while obeying each of the rovers' resource constraints and flight rules. While the goals do not indicate which rover should be used for that request, the scheduler automatically assigns rovers to tasks in a fashion that best serves the entire set of requests. After generating the commands and simulating their execution, the rover telemetry set (i.e., gathered data) is sent back to the science analyzer. Using this new data, a new set of goals is generated that is estimated to provide the most useful information for classifying the observed objects. This process is repeated until the science goals are all accomplished or the rovers are no longer operational.

As the focus of this paper, we describe a centralized approach to automated planning. The central planner can be run on the ground, on an orbiter, on a lander, or on one of the rovers (see Figure 1). The relevant sub-plans (i.e., command sequences) are transmitted to each rover for execution. The intelligent capabilities (including automated planning), however, could be distributed across the agents. Other approaches to multi-agent planning have various degrees of distribution [1,2,12,13,16,17].

To produce the rover command sequences, we use the ASPEN (Automated Scheduling and Planning Environment) system. ASPEN uses an "iterative repair" algorithm, which classifies conflicts and attacks them each individually. Conflicts occur when a plan constraint has been violated; this constraint could be temporal or involve a resource, state or activity parameter. Conflicts are resolved by performing one or more schedule modifications such as moving, adding, or deleting activities. A goal with an unassigned rover is one type of conflict. Other conflicts may include a science request for examining a target location that does not have a planned visit. Resolving this conflict involves adding a traverse command to send one of the rovers to the designated site. The iterative repair algorithm continues until no conflicts remain in the schedule, or a timeout has expired. In order to enhance the quality of produced schedules, we have implemented heuristics for assigning rovers to goals and for deciding on the order in which to visit each of the specified locations. The heuristics borrow from algorithms for finding solutions to the Multiple Traveling Salesmen Problem (MTSP). With multiple rovers covering the same area, we want to choose paths that minimize the total traversing time of all the rovers. Both algorithms and heuristics are generic, and can be used for other applications similar to the multi-rover problem.

This rest of this paper is organized in the following manner. We begin by characterizing the multiple cooperating rovers application domain and describe some of the interesting challenges. Next, we introduce the ASPEN planning and scheduling system and explain how automated planning and scheduling techniques can be applied to this problem. We discuss several heuristics for solving the MTSP problem and present some results on how they improve both system and final plan efficiency. We then discuss the overall framework that is used to achieve a set of geology related science goals. Next, we discuss both how to extend this system to provide the long-term goal of rover and spacecraft autonomy and how this extension compares with related efforts. Finally, we present our conclusions and discuss several of the issues being addressed in future work.

2. MULTIPLE COOPERATING ROVERS

With all the success of recent space missions, we still have relatively little knowledge of our solar system. In addition, the quest for cheaper missions makes space information gathering an even more challenging problem. One approach to this problem is to deploy many smaller probes instead of fewer, more complicated spacecraft. While one well-equipped spacecraft can do a detailed analysis of one

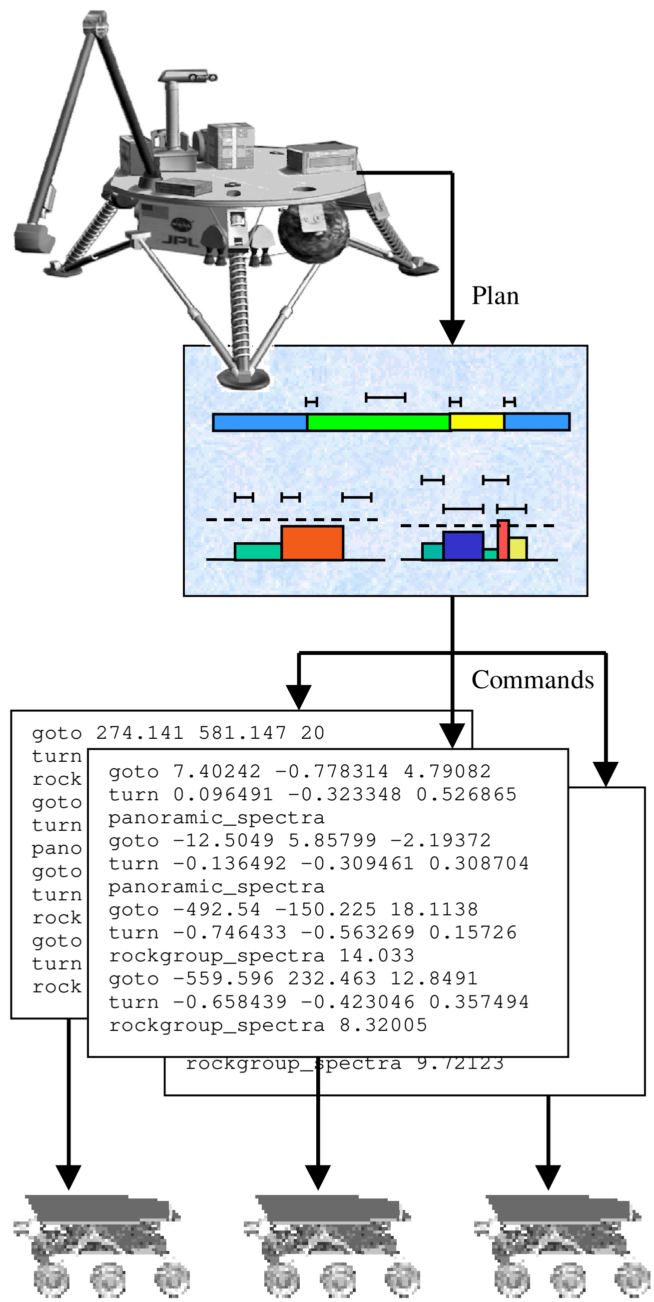


Figure 1 Rover command sequences generated by a centralized planner.

site, several small spacecraft can gather information over a larger area. We have already seen this approach on recent missions such as the atmospheric probe on Galileo and the Sojourner rover on Pathfinder. In each case, separating a small piece of the spacecraft allowed science instruments to reach out to new ground. Recent advances in robotics are enabling more flexible and mobile systems, while retaining robustness. Future missions will likely have multiple components, all of which need to be coordinated to serve the mission goals.

Whether they are spacecraft, probes or rovers, coordinating multiple distributed agents introduces some interesting new challenges for automated planning and other supporting technology [1,2]. Issues arise concerning:

- Interfaces between agents
- Communication bandwidth
- Group command and control
- Onboard capabilities

For example, mission designers will need to decide on interfaces for each of the rovers. This includes not only interfaces to the ground operations team, but also to other rovers and the lander and/or orbiter. A certain level of communication capabilities will need to be assigned to each, limiting the amount of information that can be shared between the rovers (and ground). The mission design will need to include a "chain of command" for the team of spacecraft/rovers, indicating which rovers are controlled directly from the ground, and which are controlled by other rovers or orbiting/landed spacecraft. Finally, the onboard capabilities will need to be considered, including computing power and onboard data storage capacity. This will limit the level of autonomy each of the rovers can have.

Many of these design issues are related, and all of them have an impact on automated planning and scheduling technologies used for the mission. The interfaces determine what activities can be planned for each rover. The amount of communication available will determine how much each rover can share its plan. In addition, the execution of a rover may need to be monitored for replanning purposes. If bandwidth is low and reaction time is critical, a rover may need to monitor and replan its own activities in response to unexpected or fortuitous events. The control scheme will also determine which rovers execute activities in the plans. If one rover controls another, the "master" rover will send activities from its plan to the "slave" rover for execution. Decisions on the onboard capabilities of each rover, however, will limit the independence of each rover. With little computing power, one rover may only be able to execute commands. More power may allow it to plan and execute. Still more power may allow it to plan, execute, monitor and replan activities. The planning, execution and monitoring functions can be for the rover itself, as a service to other rovers or in cooperation with other rovers.

In recent work at the Jet Propulsion Laboratory (JPL), we have demonstrated a prototype of automated planning and command sequence generation for multiple cooperating rovers. For the initial work, we have chosen a configuration with one "master" planner with several "slave" rovers. The centralized, master planner (which can be on the lander or on one of the rovers) generates and transmits commands to the simulated rovers for execution. The rovers have identical interfaces for traversing, communicating and performing science experiments. Issues with faults, monitoring and replanning are being addressed in future work.

```
Activity image {
  int x, y, z; // location of image
  Decompositions =
    rover1_image (x, y, z) or
    rover2_image (x, y, z) or
    rover3_image (x, y, z);
}

Activity rover1_image {
  int x, y, z; // location of image
  Reservations =
    rover1_battery use 10,
    rover1_memory use 1000,
    rover1_location must_be <x,y,z>;
}

Activity rover1_traverse {
  int x, y, z; // location to go to
  Reservations =
    rover1_battery use 100,
    rover1_location change_to <x,y,z>;
}
```

Figure 2 Rover Activity Definitions

3. Automated Scheduling and Planning

Artificial Intelligence (AI) planning and scheduling technologies can be used to automatically generate command sequences for spacecraft and rovers. Given a set of high-level science goals, the planner/scheduler automatically generates low-level command sequences that accomplish the goals and that obey flight rules and resource constraints. To provide this technology, we extend the ASPEN application framework.

ASPEN [3] is a reusable application framework that can be adapted for many planning and scheduling applications. First, models of the spacecraft and rovers are defined in the ASPEN modeling language. This language allows the user to define the set of activities, resources, and state variables as well as the interactions and constraints between them. An activity has a start time and an end time. It may also have a set of temporal constraints to other activities. Each constraint indicates a required order of execution for a pair of activities. An activity may also have a set of reservations. A reservation is a scheduled usage or requirement on a system resource or state variable. An activity can also decompose into a set of activities with temporal constraints between them. Finally, an activity can have any number of parameters with arbitrary functions to compute their values.

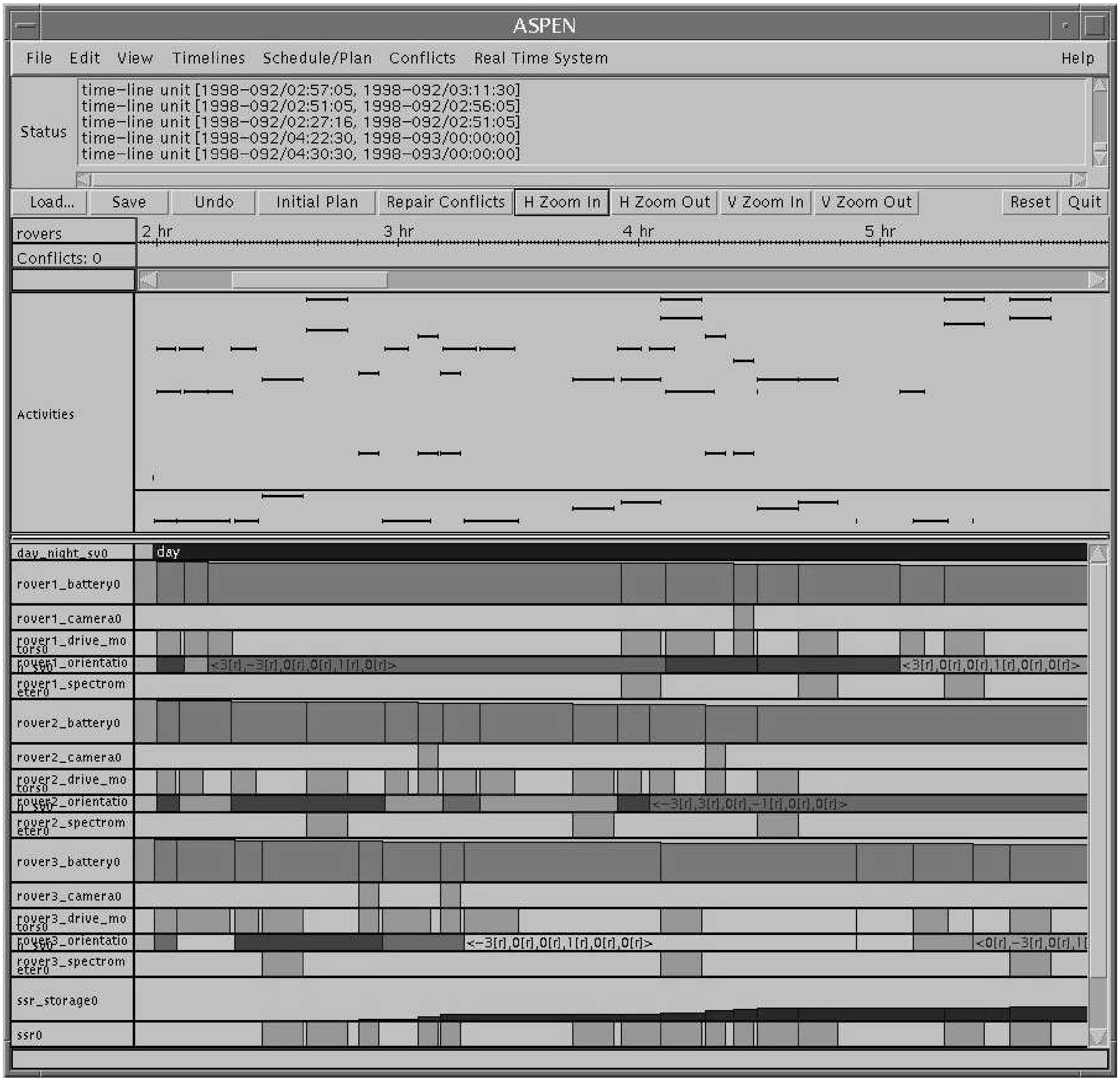


Figure 3 A Plan/Schedule for Multiple Rovers Viewed with the ASPEN GUI

The application model essentially defines the types of activities and resources that can occur in a given schedule. Figure 2 shows some examples of activity types for the multiple rover domain. A plan/schedule is a particular configuration of instances of the activity and resource types. An example plan/schedule for the multi-rover application is shown in Figure 3, with activities represented as horizontal bars in the top section, and resource profiles in the bottom section. Some activities are uncontrollable but may have effects that are required by other activities. For example, sunrise and sunset determine when solar panels are operational. These activities are simply loaded at the start of planning. Next, the high-level science goals can be inserted into the schedule. Typically, these are unexpanded activities that have unspecified parameter values, including the start time. In addition, goals will usually have unsatisfied requirements that can only be resolved with other activities. From this, the planner/scheduler must generate a plan that has all of these problems resolved.

In ASPEN, unexpanded activities, unspecified parameter values, unsatisfied requirements and violated constraints are all considered conflicts in the schedule. Therefore, the problem becomes one of finding a conflict-free schedule. ASPEN has a library of algorithms designed to search for a conflict-free schedule. One of the more widely used algorithms is based on a technique called “iterative repair” [4]. In this algorithm, conflicts are classified and addressed in a local, iterative fashion. First, a conflict from the set of conflicts is chosen for repair. Then, a repair method is chosen as an operation for resolving the conflict. Repair methods include moving activities, adding new activities, deleting activities, setting parameter values, and decomposing activities into subactivities. For each method, other decisions may be required. For example, when moving, an activity and location must be selected. When setting a parameter, a new value must be chosen. After making the appropriate decisions, the scheduling operation is performed in hopes of resolving the conflict. Finally, the new set of conflicts is collected, and the algorithm repeats until no conflicts are found, or a user-defined time limit has been exceeded.

In the multi-rover application, activities and resources are modeled for the lander and each of the rovers. The lander provides the communication link as well as temporary data storage. Each rover has activities such as traversing, turning, taking images, taking spectrometer readings, and digging. Each rover has its own resources such as battery power, solar array power, and science instruments, and state variables representing location and orientation.

Science goals are defined as abstract activities that do not specify which rover to carry out the science. One example is the “image” activity in Figure 2. These types of activities can be decomposed into a science activity on one of the rovers. For example, the “image” activity can be decomposed into one of several more specialized image activities involving one of the possible rovers. When a science goal is first created, it is typically not decomposed, and therefore considered a conflict. Resolving this conflict involves selecting a decomposition (i.e., one of the specific rover activities) and creating the new subactivity. In this way, work can be delegated to each of the rovers while considering the total available resources. For example, the current traverse times can be considered when selecting a rover to achieve a science goal at a new location. A science goal, such as taking a spectrometer reading, simply requires a rover to be at a particular location. New traverse and turn activities (e.g., “rover1_traverse” in Figure 2) must be added to move the selected rover to the desired location and orientation. The planner/scheduler must also select specific start times for each of the activities (obviously, the order of the traverses will matter). Some activities may even need to be deleted. For example, if all activities do not fit in the available time, some low-priority goals may be rejected.

4. TRAVELING SALESMEN HEURISTICS

One of the dominating characteristics of the multi-rover application is the rover traversals to designated waypoints. Decisions must be made not only to satisfy the requested goals, but also to provide more optimal (i.e., efficient) schedules. When not considering efficiency, one possible schedule that achieves all science goals is to send one rover to every target location. However, usually this would not be the desired behavior, and therefore some schedule optimization must be done. We have chosen to do this optimization during the repair process. As certain types of conflicts are resolved, heuristics are used to guide the search into making decisions that will produce more optimal schedules. In other words, when several options are available for repairing a conflict, these options are ordered based on predictions of how favorable the resulting schedule will be.

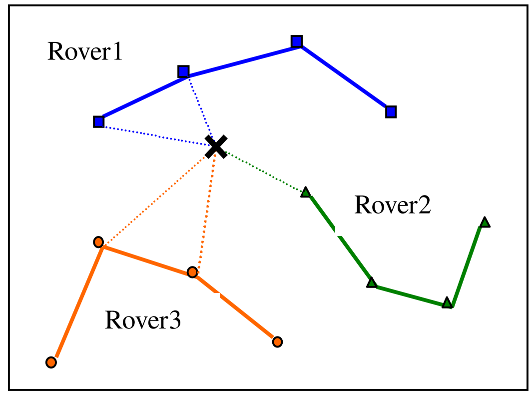


Figure 4 Traveling Rovers

The heuristics we have implemented are based on techniques from the Multi-Traveling Salesmen Problem (MTSP). The Traveling Salesman Problem (TSP) [5] is one of finding a minimal tour for a salesman that needs to visit a number of cities (and typically return home). For MTSP, at least one member of a sales team must visit each city such that total traveling time is minimized. Salesmen are allowed to travel in parallel with each other.

Many algorithms exist for solving both TSP and MTSP problems. For a small number of locations ($N<10$) optimal solutions can be found in a reasonable amount of time. However, for larger sets of locations, finding optimal solutions is too expensive (NP-hard) and approximate algorithms can be used [14]. Greedy techniques can be used to find near optimal solutions in polynomial time ($O(N^2)$), where the resulting tour lengths have been proven to be at most $\lceil \lg N \rceil + 1$ times the optimal length. Two possible types of greedy TSP algorithms are insertion and appension algorithms. In an appension algorithm, unvisited locations are selected and appended to the end of the existing planned tour. The next location to append must be chosen carefully in order to minimize the final tour. The greedy strategy would simply choose the unvisited location that, when appended, results in the shortest intermediate tour. For insertion algorithms, the order in which unvisited locations are chosen is less critical, because subsequent locations can be inserted between any two locations in the current tour. In this case, the greedy strategy chooses the insertion point that results in the shortest intermediate tour. Both algorithms can be easily extended to multiple travelers. Unvisited locations are either appended or inserted into *any* of the tours when looking for the shortest tour.

The multi-rover scenario fits naturally into the MTSP class of problems, with only a few differences. First, the rovers are typically not required to return to their original locations (however, for sample return missions, this would be necessary). This is a minor difference and does not change the general problem[1]. Figure 4 shows three possible insertions (one from each path) for a new location. Second, while planning activities for multiple rovers, one is also be concerned with the earliest finish time (i.e., makespan) of the schedule. The schedule with the minimum total path length (sum of rover path lengths) may not necessarily be the schedule where all activities finish the earliest. Reducing the total traverse time will reduce wear on the rovers, while reducing the makespan will increase the available science time. Finally, generating command sequences requires reasoning about more than just the paths of the rovers. Each rover has a set of flight rules and a limited amount of resources. All commands, including traverses, must be scheduled in a way that does not violate any of the flight rules or resource constraints. Some of these constraints may inherently require sub-optimal travel paths.

When generating command sequences for multiple rovers, ASPEN uses two heuristics that implement a greedy insertion MTSP algorithm. One is used to select a decomposition of a generic science goal into a specific science activity for one of the rovers. The other is used to select a temporal location for the science activities when they are moved. Both use the same evaluation criteria: make the selection that results in the shortest path. For the decomposition heuristic, this means choosing the rover that has the shortest path after including a visit to the new location. For the move heuristic, the new science activity is moved to a time between two existing science activities which creates a new path shorter than any other possible new path.

Table 1 MTSP vs. Random Heuristics

Avg Time For	MSTP	Random
Traverse	2 hrs 21 min	3 hrs 4 min
Makespan	4 hrs 1 min	8 hrs 24 min
Planning	60 sec	68 sec

Using the new heuristics had a significant impact on the generated command sequences. The results of using these heuristics were compared with results from using random heuristics (i.e., heuristics that make selections randomly) and are shown in Table 1. Twenty trials were run on the same set of goals but with different random seeds. The MSTP heuristics reduced both makespan and total traverse time of the final sequences. Traverse time was decreased by 24% and makespan by 52%, approximately doubling the available time for rover activities. In addition, the time required to generate the sequences was reduced. Approximately 13% less time was needed to repair all of the conflicts created by the same set of science goals.

5. ANALYZE-PLAN-EXECUTE

ASPEN was demonstrated on a science scenario involving three planetary rovers on a simulated rock field. The planner was integrated with a science simulator and a science analysis module. Together, they implemented an analyze-plan-execute cycle shown in Figure 5. In each cycle, ASPEN receives a set of goals from the science analyzer and initial rover positions from the simulator. Both are represented as activity instances in the planner. A message is sent to ASPEN instructing it to begin receiving the activity instances, followed by a text representation of the activities. Next, a message is sent to ASPEN telling it to start planning with the current set of activities. Finally, ASPEN is instructed to send the resulting command sequence to the simulator when planning is complete and no conflicts remain in the schedule.

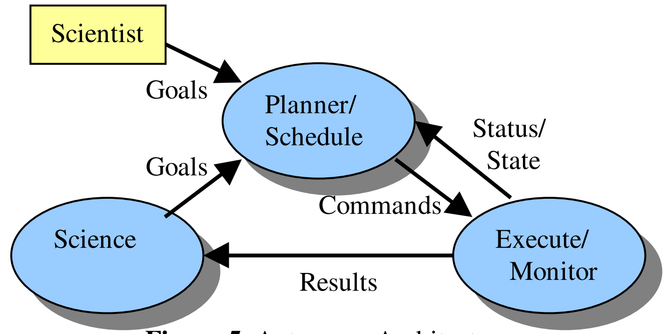


Figure 5 Autonomy Architecture

The science simulator generates the rover environment as well as simulating execution results. By changing parameters in the simulator, various rock fields can be generated consisting of several different types of rocks. The simulator also represents the location and orientation of each rover. From this, the field of view of each rover can be estimated. The simulator accepts commands for simulating science experiments as well as moving and turning the rovers. Given a command for a science experiment, such as a spectrometer read of a rock, the simulator would generate results based on the instrument type and rover location.

The science analysis module consists of a clustering algorithm [15] that automatically generates goals for science instruments based on results of previous experiments. The clustering algorithm attempts to classify observed rocks. The new requested observations are those selected by the algorithm that are predicted to provide the most useful information for refining the classification. Goals are also prioritized and those with priorities below a user-specified threshold are considered optional. If the planner is having difficulty finding a solution, optional goals can be rejected to free up time for higher priority goals.

6. TOWARDS AUTONOMY

One of the goals of automated planning and scheduling is to provide important capabilities for autonomous systems. An autonomous system must be able to achieve objectives in the face of uncertainties with little outside guidance. An onboard planning system, together with systems for execution, monitoring, and science analysis, can provide the capabilities for an autonomous space vehicle [11].

[1] We use the term "path" as opposed to "tour" to distinguish from traversals that return to the original location. Here, a path is a traversal between science waypoints. We do not address path planning for the purpose of obstacle avoidance.

With these systems onboard, a rover can operate unaided for long periods of time given only a set of high-level objectives. The science analysis would use the objectives, and the data collected so far, to generate new sets of science goals for the rover. These goals are then sent to the planner for expansion and sequencing. Once the valid command sequence is generated, commands would be executed and monitored by the corresponding onboard systems. As information is acquired regarding command status and actual resource utilization, the planner can update future projections. From these updates, new conflicts and opportunities may arise, requiring the planner to replan in order to accommodate the unexpected events. In addition, new goals may be presented to the rover from the ground or from the onboard science analysis. Finally, the data resulting from execution can then be fed back to the science analysis system to use in generating new science goals. All of this can happen onboard the rover with little or no human interaction.

Although this may seem like a very optimistic goal, steps are being made in this direction. The New Millennium Deep Space One (DS1) spacecraft will demonstrate the onboard Remote Agent experiment [6,7,11] that has all but a science analysis module. Prototype demonstrations have shown the ASPEN planning system successfully integrated with an execution architecture for preliminary models of the Deep Space Four (DS4) comet lander mission. In this case, the Continuous Planner [8] (an extension of ASPEN) constantly monitors the simulated execution of the spacecraft and replans when conflicts are detected or new goals inserted. Finally, ASPEN has been integrated with two science tools. The first automatically generates science goals, which was described in the previous section. The second, called WITS (Web Interface for TeleScience) [9] is used for manual selection of science goals on actual images of the rover surroundings.

Rather than fully committing to an autonomous rover, autonomy can be inserted incrementally. At first, manual science tools, such as WITS, can be used to generate goals for a planner running on the ground. With crude models, the results of the planner can simply be used for analyzing the feasibility of the science requests. It can also be used to allow the scientists to provide rover engineers with a more complete sequence rather than just a set of requests. This sequence will have a better chance of being accepted by the engineers, and fewer negotiations will be required. Also, engineers will be required to do less work in order to complete and validate the sequence. As the planner models become more accurate, the resulting sequences will require fewer modifications. With a refined model, sequences will require no further validation, and the planner can be moved onboard the rover. With the planner onboard, the science analysis module can be inserted to generate low priority goals in addition to those given by human scientists. This would keep the rovers busy, even when ground contact is lost or when high priority goals cannot be achieved.

7. RELATED WORK

While we are unaware of any other papers on multiple cooperating rovers for exploring an extraterrestrial terrain, the multi-agent community has developed prototype systems for a similar problem involving servicing multiple transportation requests with a population of agents. These agents can be trucks for a shipping company [16], forklifts on an automated loading dock [17], or robots servicing a set of office cubicles [18]. Like other multi-agent systems, the main objective motivating these examples is to study various mechanisms for high-level task allocation and low-level activity coordination. All of these examples give their agents the autonomy to plan their activities while using the mechanisms to manage interactions.

While others start from a multi-agent paradigm, this work starts with that of a single agent, and its approach to planning and execution is very similar to that of the DS1 spacecraft's remote agent experiment [6,7]. Instead of treating the rovers as distinct agents, we treat them as connected appendages of the lander. Given that the lander plans and schedules all rovers from global information, it can generate more optimal plans than the multi-agent approach's rover based planners – which only plan from local information on each rover.

Although optimality does not sufficiently motivate using a multi-agent approach, scalability and robustness will. The rovers-as-appendages approach uses an assumption that the lander can appropriately monitor all rovers and respond to unexpected events. This requires a highly reliable communications channel with enough bandwidth to overcome the unexpected event frequency. This implies that all rovers are in constant radio contact and the system has either a very high-bandwidth communications channel or a moderate number of rovers sparsely distributed over a static landscape. Upon dropping the assumption, the rovers will have to locally manage uncertainties, and this will motivate using a multi-agent approach in the future. Given the current mission scenario, a single agent approach is sufficient, and the incremental commitment to autonomy requirement makes it preferable.

8. CONCLUSIONS

Using multiple rovers can significantly increase the capabilities and science return of a mission. Rovers can work in parallel, cooperating to achieve a set of global objectives. Such multi-agent systems, however, introduce new challenges to control and autonomy. Issues of interfaces, communication, control and resource distribution will need to be addressed. An automated planner must be considered when making decisions on these issues. The use of an automated planner can significantly reduce the required rover operations time. Ultimately, the planner can be moved onboard providing a key component in making the rovers more autonomous.

In our work, we have demonstrated the ASPEN planner generating command sequences for a team of rovers. As part of the analyze-plan-execute cycle, this work was the first step towards an architecture for autonomous rovers. In our first approach, we used a design with a single centralized master planner generating commands for several distributed slave rovers. We assumed that no problems occurred while executing the plans, and that communication bandwidth is high. A centralized approach, however, may be necessary for missions with centralized computing power (i.e., an insufficient CPU on each rover).

Considerable work remains to be done for the multiple cooperative rover effort at JPL. First, autonomous rovers will also need replanning capabilities in order to cope in harsh, unpredictable environments. Unexpected events may occur during execution of a previously constructed plan. These events may invalidate the current plan, requiring a new plan to be generated. The planner can repair the existing plan (or generate a new plan) while considering the new information. The idea of continuously monitoring and repairing plans has been called continuous or dynamic planning [8]. With a continuous planner onboard a lander or rover, new valid plans can be made available much faster than if the rover required the plan to be transmitted from earth. To provide integrated rover planning and execution, we intend to adapt the continuous planner to the multi-rover application.

To support more advanced missions with multiple autonomous rovers, we need to consider distributed planning. This would include rovers planning for themselves, and for other rovers. If there is a slow communication link between rovers, or between rover and lander, a continuous planner may be useful on each rover. This would eliminate the need to constantly transmit monitoring information across the slow communication link. By balancing the workload, distributed planning can also be helpful when individual computing resources are limited. Distributed planning is especially difficult, however, when rovers do cooperative or coordinated activities with shared resources. This would include, for example, several rovers communicating with a single lander. The concept of distributed planning is a relatively young area in planning research [10,12,13]. While evaluating the various approaches, we must consider the needs specific to multiple rovers.

ACKNOWLEDGMENTS

This work was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract to the National Aeronautics and Space Administration. Work described in this paper was supported by the Autonomy Technology Program, managed by Richard Doyle, NASA Code SM and by the Advanced Concepts Office, managed by Neville Marzwell, NASA Code S.

Eric Mjolsness, Alex Gray, Tobias Mann and Becky Castano provided the simulator and clustering algorithm mentioned in this work. Thanks to Russell Knight for his expertise on Traveling Salesman Problems. The help of other members of the Artificial Intelligence Group is also appreciated.

REFERENCES

[1] Mataric, M., "Issues and Approaches in the Design of Collective Autonomous Agents," *Robotics and Autonomous Systems*, 16 (2-4), Dec. 1995, pp. 321-331.

[2] Parker, L., "ALLIANCE: An Architecture for Fault Tolerant Multi-Robot Cooperation," *IEEE Transactions on Robotics and Automation*, 14 (2), 1998.

[3] Fukunaga, A., Rabideau, G., Chien, S., and Yan, D., "Towards an Application Framework for Automated Planning and Scheduling," *Proceedings of the IEEE Aerospace Conference*, Snowmass, CO, 1997, pp. 375-386.

[4] Zweben, M., Daun, B., Davis, E., and Deale, M., "Scheduling and Rescheduling with Iterative Repair," *Intelligent Scheduling*, edited by M. Zweben and M. Fox, Morgan Kaufmann, 1994, pp. 241-255.

[5] Johnson, D., McGeoch, L., "The Traveling Salesman Problem: A Case Study in Local Optimization," *Local Search in Combinatorial Optimization*, edited by E. H. L. Aarts and J. K. Lenstra, John Wiley and Sons, London, 1997, pp. 215-310.

[6] Pell, B., Bernard, D., Chien, S., Gat, E., Muscettola, N., Nayak, P., Wagner, M., Williams, B., "An Autonomous Spacecraft Agent Prototype," in *Proceedings of the First Annual Workshop on Intelligent Agents*, Marina Del Rey, CA, 1997.

[7] Muscettola, N., Smith, B., Fry, C., Chien, S., Rajan, K., Rabideau, G., Yan, D., "On-Board Planning for New Millennium Deep Space One Autonomy," in *Proceedings of the IEEE Aerospace Conference*, Snowmass, CO, 1997.

[8] Chien, S., Knight, R., Stechert, A., Sherwood, R., Rabideau, G., "Integrated Planning and Execution for Autonomous Spacecraft," in *Proceedings of the IEEE Aerospace Conference*, Aspen, CO, 1999.

[9] Backes, P., Tso, K., and Tharp, G., "Mars Pathfinder Mission Internet-Based Operations Using WITS," in *Proceedings IEEE International Conference on Robotics and Automation*, Leuven, Belgium, 1998.

[10] Mali, A., Kambhampati, S., "Distributed Planning," *Encyclopedia of Distributed Computing*, Kluwer Academic Publishers.

[11] Chien, S., Muscettola, N., Rajan, K., Smith, B., Rabideau, G., "Automated Planning and Scheduling for Goal-based Autonomous Spacecraft," *IEEE Intelligent Systems*, September/October 1998, pp. 50-55.

[12] Hagopian, J., Maxwell, T., Reed, T., "A Distributed Planning Concept for Space Station Payload Operations," *Third Symposium on Space Mission Operations and Ground Data Systems*, Greenbelt, MD, November 15-18, 1994.

[13] Cook, D., Gmystrasiewicz, P., Holder, L., "Decision-Theoretic Cooperative Sensor Planning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(18), 1996.

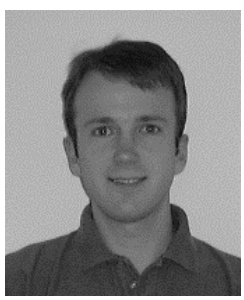
[14] D. Hochbaum, *Approximation Algorithms for NP-hard Problems*, PWS Publishing Company, 1997.

[15] Hathaway, R. J., "Another Interpretation of the EM Algorithm for Mixture Distributions," *Statistics and Probability Letters* 4, 1986, 53-56.

[16] Fischer, K., Müller, J., Pischel, M., Schier, D., "A Model For Cooperative Transportation Scheduling," in *Proceedings of the First International Conference on Multi-Agent Systems*. San Francisco, CA, 1995.

[17] Müller, J., *The Design of Intelligent Agents, A Layered Approach*, Lecture Notes in Artificial Intelligence, Springer-Verlag, 1996.

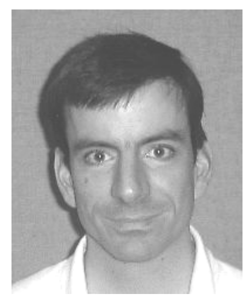
[18] Ohko, T., Hiraki, K., Anzai, Y., "Reducing Communication Load on Contract Net by Case-Based Reasoning – Extension with Directed Contract Forgetting," in *Proceedings of the Second International Conference on Multi-Agent Systems*. Kyoto, Japan, 1996.



Gregg Rabideau *is a member of the Artificial Intelligence Group at the Jet Propulsion Laboratory in Pasadena, California. Gregg earned his B.S. and M.S. degrees in Computer Science at the University of Illinois where he specialized in Artificial Intelligence. His main focus is in research and development of planning and scheduling systems for automated spacecraft and rover commanding. Gregg is currently working on planning and scheduling for future Mars rover missions. Past projects include the New Millennium Deep Space One mission and the DATA-CHASER shuttle payload.*

Tara Estlin *is a member of the Artificial Intelligence Group at the Jet Propulsion Laboratory in Pasadena, California where she performs research and development of planning and scheduling systems for rover automation and ground station scheduling. She received a B.S. in computer science from Tulane University in 1992, an M.S. in computer science from the University of Texas at Austin in 1994, and a Ph.D. in computer science from the University of Texas at Austin in 1997. Her current research interests are in the areas of planning, scheduling and machine learning.*

Steve Chien *is Technical Group Supervisor of the Artificial Intelligence Group of the Jet Propulsion Laboratory, California Institute of Technology where he leads efforts in research and development of automated planning and scheduling systems for science data analysis, ground station automation, and highly autonomous spacecraft. He is also on the Faculty of the Department of Computer Science at the University of Southern California. He holds a B.S., M.S., and Ph.D. in Computer Science from the University of Illinois. His research interests are in the areas of: planning and scheduling, operations research, and machine learning and he has published numerous articles in these areas. In 1995 he received the Lew Allen Award for Excellence and in 1997 he received a NASA Exceptional Engineering Achievement Medal both for his research and engineering work in automated planning and scheduling systems.*



Anthony Barrett *is a member of the Artificial Intelligence Group at the Jet Propulsion Laboratory in Pasadena, California where his research and development activities involve planning and scheduling applied to controlling constellations of spacecraft and to managing workflow in a flight project. He holds a B.S. in Computer Science and Applied Mathematics from James Madison University, an M.S. in Computer Science from the University of Washington, and a Ph.D. in Computer Science from the University of Washington. His research interests are in the areas of planning, scheduling, and multi-agent systems.*