The DATA-CHASER and Citizen Explorer Benchmark Problem Sets

Barbara Engelhardt [1], Steve Chien [1], Anthony Barrett [1], Jason Willis [2], Colette Wilklow [2]

[1]Jet Propulsion Laboratory, California Institute of Technology
{firstname.lastname}@jpl.nasa.gov

[2]Previously at Space Grant Consortium, University of Colorado, currently at (1)

Abstract. This paper introduces two benchmark problem sets based on actual space mission operations. Each benchmark problem set includes problem generators, declarative specification of the problem(s), and one or more simulations. The first mission is the DATA-CHASER shuttle payload that flew onboard space shuttle Discovery flight STS-85 in 1997, and demonstrated the ability of automated mission planning to both reduce commanding effort and improve science return. The second mission is the Citizen Explorer Mission (CX-1), which is a small, earth orbiting satellite currently being prepared for launch. We include three problem classes of increasing complexity (and realism) for each mission scenario: planning and scheduling with states and resources (PSSR), PSSR with functional dependencies, and PSSR with functional dependencies and plan quality. The actual implementations are available for download from web sites at the University of Colorado, which designed and operated these spacecraft and missions.

1 Introduction

Historically, the research and applications communities in the area of automated planning and scheduling have not had significant amounts of interaction. As a consequence, there has not been significant transfer of information between the communities in either direction. Specifically, the cross-fertilization of communities is limited to a small number of research systems deployed in an ongoing operational context, and similarly only a few real-world planning and scheduling problems have breached the research community.

There are many reasons for this situation. It takes an incredible investment of time and energy for a researcher to learn the intricacies of an application domain. The research community and research institutions generally have not rewarded this investment of effort. Likewise, the community solving actual planning and scheduling problems did not have adequate incentive to work with the research community. With many difficult problems to solve in building functional systems, many of the central research areas are of lower priority. And in the commercial arena, there is significant negative incentive to distribute lessons painfully learned, which represent an important competitive advantage after all.

Fortunately, this situation appears to be changing. Within the research community, there is an increasing understanding of the importance of being relevant to the real world. With the appearance of startup companies and venture capital, the financial incentive to develop mature algorithms has grown considerably. Furthermore, with the

maturation of the field (and technology), the incentive for applied organizations to engage the research community has become more urgent.

This paper represents an effort to leverage the research community in developing techniques for integrated planning and scheduling problems that occur commonly for space mission operations. The remainder of this paper is organized as follows. First, we describe the basic elements that we provide for each testbed domain: a domain description, a declarative model, problem generators, and a simulation. For each such domain, we provide three versions of increasing complexity. A basic version of each domain includes planning and scheduling with resources. A more complex version adds functional dependencies. And the most complex version includes both functional dependencies and plan quality. Next, we describe each of the domains described in this paper: DATA-CHASER shuttle payload operations and Citizen Explorer (CX-1) satellite mission operations. For each domain we describe the background for the mission and mission goals. We then provide more details about the planning and scheduling problems. Next we describe the provided problem generators and simulators. Finally, we compare the problem domains presented with previously published domains in both the planning and scheduling and the operations research communities.

2 The Elements of a Testbed Domain

The first element of each domain is a textual description. This description gives the context of the model, problem generators, and simulation. It explains the mission being modeled and the overall problem context. It also references previous work in automated planning and scheduling solutions to the problem.

The second element of each domain is a model. This model is provided in the ASPEN Modeling Language (AML) [Sherwood et al. 1998]. AML is a mature representation language that has been used to represent planetary rover operations constraints [Sherwood et al. 2000] as well as space mission operations constraints for actual deployments [Smith et al. 2001, Wilkins&desJardins 2001]. While ideally these domain models could be provided in a more generic language, this format was chosen for two reasons. First, using AML facilitates timely release of the domain models by minimizing release effort. Given that it is desirable to release as many domain models in a timely fashion, it is hoped that others in the community will translate these models into a generic format. Second, certain aspects of the domain model would be difficult to represent in current generic domain description languages. The hope is that release of these models will spur extension of domain description languages. A description of the modeling language used for the original models is available for download from http://aspen.jpl.nasa.gov.

The third element of each problem domain is a problem generator. This is an executable (in these cases, Perl scripts) that can be used to generate a large number of initial states and goals for a planner to solve. In most cases the problem generator is parameterized to enable generating problems of varying size and difficulty.

The final element of each problem domain is a simulator. Once a planner has specified a plan, an execution simulation can be used to stochastically evaluate the effectiveness and the robustness of the plan for simulated missions operations. Effectiveness determines how well the planner satisfied the spacecraft goals, and can be measured by assessing the operational results, such as the science return, resource consumption, or state changes. Robustness, on the other hand, measures the ability of the planner to enable a successful mission in spite of significant run-time variations and anomalies, such as an action finishing early, consuming excessive resources, or

simply not executing correctly. Robustness can be measured by determining the number of inappropriate actions that are sent to the simulator, which in turn violate the constraints of the domain or put the spacecraft in an unsafe state.

The simulator itself has three parts: The database which stores the current state at time *t*, a set of specifications and constraints, and an executive, which receives action commands from the planner, attempts to execute them using the specification and constraint set, and updates the current state. The simulator is scalable so that there can be large or small simulations. The modeling of the simulator depends on how the planning language is defined, which determines whether activities are time-stamped, connected by constraints, or have conditional activities, or whether the planner can update its plan based on simulation feedback during the simulation.

For our domains, batch planners and continuous planners both use the same simulator. The planner is required to submit activities some number of seconds in advance of their scheduled execution, which is described as the *commit window.* The commit window must be greater than or equal to one second, and the planner must register the commit window length with the simulator when execution starts. The commit window size can change during execution, but the planner cannot modify activities once they have entered the commit window. The planner can receive updates from the simulator regarding activity parameter changes (such as start time or duration), state and resource updates, and current time. The simulator can warp so that the plans can run much faster than real time, relative to the commit window described by the particular planner. Batch planners can control the simulator either by setting the commit window to the duration of the plan (in which case the simulator can quickly warp through the entire simulation), or by passing parameterized activities at the appropriate times, but not replanning during the simulation. An important point to note here is that the domain information does not pose any restrictions on the use of planning technology to solve the problems. The planners could use constraint-based methods, committed search methods, or any other methods. Indeed, there is no restriction that a planner must be used, a smart executive or even arbitrary C code could be used to command the simulator. This opens the competition to truly test if planning technology is useful.

The stochastic model, or run-time variations, are stored as part of the specifications, which specify distributions instead of single values for certain variable features. Three different aspects of the mission can be impacted at run time: activity failure, resource consumption, and time and duration of state changes. Each domain has a stochastic and a non-stochastic simulator included in the release.

3 The DATA CHASER Mission

The DATA-CHASER was a Hitchhiker payload that flew onboard the Space Shuttle Discovery flight STS-85 in August 1997. (Figure 1) It had 3 co-aligned instruments that take data in the far and extreme ultraviolet wavelengths: far and extreme ultra-violet spectrometer (FARUS), soft x-ray and extreme ultraviolet experiment (SXEE), and a Lyman-Alpha solar imaging telescope (LASIT). In the actual DATA-CHASER mission, mission operations were automated using the DCAPS (DATA-CHASER Automated Planner and Scheduler) planning system [Chien et al. 1999]. The DATA-CHASER

Fig. 1. DATA-CHASER payload integrated into the STS-85 Shuttle Bay, STS-85 launching, and Payload Operator Jason Willis using DCAPS to command DATA-CHASER.

domain as modeled for the actual mission uses 67 resources and 58 activity types. Examples of resources include onboard power, a 4 MB memory buffer, and a 2 GB digital tape drive. Most of the systems have at least one state variable, which represents whether or not they are activated. Shuttle orientation is also modeled as a state variable. There are many concurrency resource constraints, for instance a downlink or uplink can only occur during contact with a TDRSS satellite. The activities include taking a picture with LASIT, changers for each state variable (such as opening/closing instrument doors), and descriptions of exogenous events like the shuttle passing to/from the Earth's shadow. Unfortunately, software integration difficulties before launch disabled part of the hardware during the mission. Our problems are based on the mission as originally designed.

DATA-CHASER problems involve trying to take observations within specified time windows given a number of exogenous events that change at different times. For instance, one consequence of flying on the shuttle system is that shuttle resources are shared and, hence, limited, with availability subject to change every 12 hours (the frequency at which NASA changes shuttle flight plans). These resources include access to uplink and downlink channels, and time that the payload is allowed to operate. Moreover, scientists would like to perform dynamic rescheduling during the mission. For instance, a solar flare can occur at random and drive a scientist's desire to rapidly alter the DATA-CHASER's activity schedule to reflect new requirements and goals, such as altered instrument priorities or longer integration times.

DATA-CHASER requires data and power management while gathering science. An automated scheduler searches for an optimal "data taking" schedule, while adhering to the constraints and resource restrictions. In its basic formulation, DATA-CHASER is a straightforward resource and state constrained scheduling problem that serves as a good introduction to the types of operations constraints common in spacecraft operations. A more complicated formulation requires representation of a number of functional dependencies including thermal and power constraints. In the full-blown formulation, DATA-CHASER represents a complex scheduling problem involving deadlines, observation windows, science preferences, linked observations, and engineering optimization criteria such as minimizing tape starts and stops as well as instrument door operations. There is no substantive planning (e.g. subgoaling) in the DATA-CHASER domain.

3.1 The DATA-CHASER Models and Problem Generators

The simplest DATA-CHASER model has over 46 activity types that are defined in terms of their effects on 19 resources and 9 states, which collectively represent the DATA-CHASER's external environment and subsystems. Such resources and states include the memory buffer, available power, communications availability periods, subsystem modes, and shuttle orientation. Most activities are possible commands to payload subsystems like performing an observation, moving data to a DAT recorder, or downlinking data. A smaller set of activities is for representing uncontrollable exogenous events like a shuttle orientation shift or entering a communications availability period. The five types of exogenous events to schedule around include:

- Shuttle orientation: The shuttle can point its cargo bay in one of four directions: Earth, Sun, Moon, and Deep Space. Given that the DATA-CHASER is a low priority Hitchhiker payload, it has no control over the orientation.
- Shuttle contamination: Occasionally the shuttle needs to fire its maneuvering rockets for orbit maintenance. In addition to accelerating the shuttle, this activity

contaminates local space for a short time. The DATA-CHASER has to close its main canister door during this time to keep its optics clean.

- Low data rate communications windows: During most of the mission the shuttle can provide a 1200 byte/sec downlink through the TDRSS satellite network, but a one to ten minute window exists in each orbit when no TDRSS satellite is in view.
- Medium data rate communications windows: Occasionally the mission will have a 25000 byte/sec downlink to a ground station, but availability depends on ground station visibility and the needs of other more important missions.
- Eclipse events: Once every orbit the shuttle travels through the Earth's shadow, and no solar observations are possible.

DCAPS-RES is our simplest DATA-CHASER planning model and illustrates planning with resources. The objective is to perform observations when the shuttle is not in the Earth's shadow, the cargo bay is facing the sun, and the shuttle has not recently contaminated the space around it. FARUS, SXEE and LASIT respectively take 72, 181, and 52 seconds and generate 5120 bytes, 48 bytes, and 2 megabytes per observation, and whole system generates a kilobyte of engineering telemetry per hour. Given that the memory buffer only has 4 MB, it is the most constrained resource. Since data can be rapidly transferred to the 2GB DAT recorder, there are naïve approaches to scheduling the observations by simply transferring the data as soon as it collected, but data on the DAT cannot be downlinked for rapid analysis during the mission. Rapid analysis is desired to let scientists alter the priorities of different observations to improve data quality. Thus some goals have explicit downlink requirements, making the scheduling problem slightly more difficult.

While our first model had fairly simple actions that took constant amounts of time and had static effects, our second model (called *DCAPS-PARM*) is slightly more complex in that it uses parameter dependency functions capture the context dependent thermal management problem. Since DATA-CHASER was mounted on a poorly conducting trellis in the vacuum of space, the only way to dissipate heat was through radiation. This means that the payload warmed when the sun beat down on it, and cooled during eclipses and when it pointed at deep space. We model this in terms of the temperature of the payload changing at rates determined by the shuttle's orientation, whether or not the canister door is open, and the power requirements of current activities. Given our model of heat, a schedule has a conflict due to calibration loss whenever the temperature falls outside of an 18° to 22° Celsius range.

Given *DCAPS-PARM*, we define *DCAPS-OPT* as an even harder third model to optimize science collection during a 12 hour period where different observations have context dependent payoffs depending on varying solar activity – an exogenous event.

The problem generators create random start states with subsequent 12-hour exogenous event scenarios and either a requested collection of observations or an observation payoff metric for the *DCAPS-OPT* model. We describe the exogenous events either in terms of cycles that start at a random point or markov models where time to take a transition is uniformly distributed between an upper and lower bound. For instance, the shuttle will be at some random point in its orbit and the day/night transition is a cycle starting at that point. Shuttle orientation provides an example of a markov model where scheduling to satisfy other payload needs results in changing the shuttle's orientation in random ways. In order to inject some realism into our markov-model-based exogenous events, we built our markov models from the 12-hour shuttle event sequences used during actual DATA -CHASER operations.

3.2 The DATA-CHASER Simulator

We evaluate solution plans for a problem by simulating them. The simulator takes exogenous events and grounded activities and determines what happens to the payload. For instance, having the contamination event with the CHASER door open may result in the instruments failing due to dirty optics. To evaluate solutions in each of the three models, the simulator has a flag to control the temperature component. For the simplest model, the simulator holds the temperature constant, and for the other models the simulator lets the temperature vary.

To make the problem more realistic, the simulator has a second flag to control a stochastic element. While actions in planning domains have explicit durations and effects, actions in reality have results that vary stochastically. For instance, a model might pessimistically state that it takes two minutes to transfer a LASIT image to the DAT recorder, but the actual time might vary from 100 to 120 seconds. In addition to time four other effects can vary around nominal operations:

- Datatakes: Datatakes will fail randomly 9% of the time. Actual power usage will differ from predicted power usage based on a normal distribution with a small variance. Failure rates and variances increase as the instruments' temperature approach the 18° and 22° Celsius bounds due to calibration problems.
- DAT Transfers: Transferring data to the DAT fails 2% of the time with data loss.
- Communications: Communications windows can drift slightly due to small variances between the Shuttle's orbit and the TDRSS satellite network.
- Thermal: The payload's rate of temperature change can vary by up to 5%.

Once given a problem description, the simulator takes grounded activities some set time in advance of executing them and giving sensory feedback in the form of failure notifications and the actual changes to the payload, which can stochastically different from the modeled expectations. This approach facilitates being able to test both batch planning approaches as well as incremental approaches. Upon completing the plan the simulator returns the plan's resulting score based on multiple criteria:

- The number of violated operational conflicts
- The number of successfully downlinked observations by observation type
- The number of observations stored on the DAT by observation type
- The number of times that the CHASER door opens and closes
- The number of activity failures by activity
- The total amount of power used while performing the observations
- An observation time based utility function for *DCAPS-OPT* problems.

4 The Citizen Explorer 1 (CX-1) Mission

The CX-1 spacecraft is a student designed and built spacecraft (Figure 2), developed by the Colorado Space Grant Consortium [Willis et.al. 1999]. The CX-1 satellite has a gravity gradient boom to keep it pointed to the Earth and uses two instruments to make atmospheric observations: a photometer to measure visible light intensity near the 365nm wavelength and a spectrophotometer to measure light intensities at different wavelengths between 280 and 350nm. These instruments will be used to perform atmospheric and climatological science coordinated with ground-based ozone and aerosol measurement. CX-1 was originally scheduled to launch on November 17, 2000. However, due to software and hardware difficulties with the communications subsystem, this launch has been delayed. Future launch possibilities are currently being negotiated. When launched, CX-1 will be in a sun-synchronous orbit around the

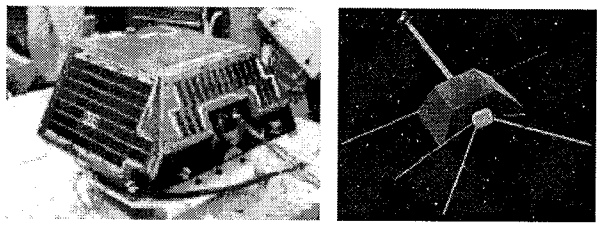

Fig. 2. The CX-1 Spacecraft, undergoing integration and test, and an artist's depiction of the CX-1 Spacecraft deployed and in flight

Earth at an altitude of 705 kilometers. The main ground tracking station will be in Colorado, and will downlink stored satellite data, provide real time health and status data, and uplink commands and control directives. CX-1 will also broadcast science and engineering data at UHF frequencies to schools at locations in the US.

CX-1 mission planning scenarios focus on the problem of acquiring the appropriate atmospheric measurements, downlinking the data to the correct schools during the upcoming pass, uplinking real-time activity requests from the University of Colorado, and monitoring spacecraft health and orbit patterns. Multiple constraints make this problem difficult. First, small data buffer sizes make CX-1 planning a highly resource constrained scheduling problem. Second, the time windows when CX-1 can downlink to a ground station are extremely limited by the limited onboard power and the small size of K-12 school ground stations (due to cost constraints),. Third, a large amount of data will be requested from the spacecraft so that it is important to optimize use of available downlink. Fourth, measurements made by CX-1 are categorized and driven by sun levels, hence the operations will vary based on near real-time feedback. While CX-1 operations does involve a small amount of planning (subgoaling), it is primarily a scheduling problem.

4.1 The CX-1 Models and Problem Generators

The model for CX-1 has approximately thirty-two activities, including data-takes, data downlinks, data uplinks, and engineering activities. These activities are defined in terms of their effects on thirteen resources and eight states, which collectively represent the CX-1's subsystems and external environment. These resources and states include the flash memory buffer, communications link, available battery power, solar array power, a sun sensor, ground station in-view periods, climate modes, and transmitter modes. Activities are either commands to satellite subsystems or exogenous events affecting the satellite's states or resources, such as entering/exiting the view of a ground station or entering/exiting direct sunlight.

The model also has mission and operations constraints, which can impact activities (temporal constraints), resources, or states. Temporal constraints include requirements that a series of critical housekeeping operations must be performed at the transitions between light and dark. Resource constraints include requirements that the flash memory buffer cannot exceed its capacities and that the battery power cannot go below half of its maximum capacity. State constraints include requirements that uplinks and downlinks only occur when the Colorado station in in-view and that the transmitter is in transmit mode. Also, school downlinks (which do not move any memory off of the flash memory buffer) can only occur when the schools are in-view and the transmitter is in broadcast mode, engineering activities must take place when the sun is not directly visible, and data takes must occur while the sun is in-view.

CX-1 problems involve obtaining and downlinking the maximum amount of atmospheric data while staying inside power and memory resource guidelines. The in-view duration for the Colorado ground station varies depending on the orbit of the satellite and outages in the ground station. Removing data from the flash memory buffer before it fills requires downlinking as much as possible during each visibility window, and this requires dynamic rescheduling as viewing periods change. Also,

changes in power consumption by certain activities affects the power profile and can drain excessive power from the satellite. Here dynamic rescheduling facilitates discarding activities to stay within power guidelines.

There are three different CX-1 models with increasing complexity. In *CX1-RES*, activities have constant durations and have constant resource and state needs. This model represents a straightforward space mission operation domain for planning and scheduling with constrained states and resources. The *CX1-PARM* model adds a number of parameter dependencies. For some activities, power usage is a function on activity duration and lighting state, and takes as much power as possible from the solar power source before relying on battery power. Battery usage is a function of current levels of battery charge and the duration of the activity. Downlinking bandwidth is a function of the modes of the satellite. Finally, the *CX1-OPT* model further increases the complexity by including optimality criteria. These criteria are based on the total amount of data downlinked to the ground stations (preferring more data), the largest amount of data contained in the memory buffer at any one time (with a preference for less), the amount of data acquired while in non-tropical climates (preferring more), the number of mode switches in the transmitter, and the smallest charge on the available power (preferring a higher minimum charge).

With these models the CX-1 problem generators create six different files containing activity instantiations for a user specified number of orbits. Some of these activities are changers for exogenous events and they cannot be deleted, removed, or modified in the schedule. Others are requests for downlinking, engineering events, and datatakes, and can be modified, added to, or removed in the final schedule.

- Initialization Activities: In the start state, there is a random amount of power in the battery and a random amount of data currently stored in the flash memory.
- Engineering Data Requests: Engineering scans are requested approximately every 280 seconds while the spacecraft is in both in-view of the sun and in darkness.
- Datatake Requests: These requests occur approximately every 170 seconds when the spacecraft is in-view of the sun.
- Sun In-View Periods: These periods are generated randomly based on an estimate of a CX-1 proposed orbit and a somewhat random start position. The spacecraft always begins the simulator in darkness. The climate (tropical or non-tropical) transitions are generated relative to the sun in-view periods.
- Power Activities: Solar power activities add a random amount of power to the available power resource, and occur approximately every 300 seconds while the spacecraft is in-view of the sun.
- Downlink Station In-View Periods: Downlink windows to both the Colorado ground station and participating schools are generated randomly based on viewing windows for an estimated orbit and sun in-view periods. Each window's duration is based on both satellite position and the strength of the receiving station.

4.2 CX-1 Simulation

We can describe the mission operations of the CX-1 model, as described above, in terms of the stochastic element of the planning simulation in order to illustrate how the simulation can differ from predicted (or nominal) operations. For instance, many activities have actual power usages that are normally distributed with a small variance around their predicted power usages.

- Engineering Scans: Engineering scans fail randomly 7% of the time.
- Datatakes: Datatakes will fail randomly 9% of the time.

- Solar Power Functions: The actual amount of solar power is normally distributed around the model's predicted solar power with a small variance.
- Light/Dark Cycle: The sunlight/darkness cycle may be shifted based on a cyclic model of satellite drift. The shift also impacts entering and exiting tropical zones.
- Downlinking: Satellite drift impacts all downlink opportunity windows. The start time may be moved (i.e., delayed lock up) and the duration may be shortened (i.e., signal cut off) based on a cyclic model of satellite drift.
- Bandwidth: The bandwidth to downlink both spacecraft data and science data is a uniformly distributed number, owing to possible downlink problems.

These stochastic parameters can remain ungrounded until run-time where they impact the planner's effectiveness. Many runs can be performed on the simulator to estimate the expected performance of the planner. For testing purposes, we also include a "happy simulator" which runs nominal operations with zero variance.

The CX-1 simulator receives parameterized activities from the planner and simulates operations for the entire duration of the plan. Throughout the the plan's execution, the simulator calculates the score based on the following criteria:

- Number of violated operational conflicts (e.g., dual usage of atomic resources, overflowing the memory buffer, or downlinks to nonexistent ground stations)
- Total amount of data downlinked to the Colorado ground station
- Flash memory usage (preferring a lower mean usage and smaller variance)
- Available power (preferring a higher mean and smaller variance)
- Total amount of data downlinked to the school ground stations
- Total amount of data acquired while in a non-tropical zone
- Total number of transmitter mode switches

5 Accessing the Problem Set Information

The problem set information is downloadable from the University of Colorado Space Grant Web Site (www-sgc.colorado.edu). While the release sets are preliminary and are still undergoing slight revisions (and testing), they are very close to the final sets. This site will also be the focal point for future updates and releases. While the domains are being made available to the public, specific terms are listed on the web site – including acknowledgement of the source in the event of any usage of the material, prohibitions on commercial use, etc.

6 Discusion: Future Work, Related Work, Conclusions

We hope that these domains will be the first in a series of space mission operations domains released for use by the automated planning and scheduling community. As we have less mature collaboration efforts with three other University Nanosatellite projects, we hope that eventually we will be able to release similar problem sets relating to these. An important aspect of this work involves determining standards for releasing domains – i.e. a formalization of the domain, problem generator, and simulator specifications. Developing a sufficiently expressive domain representation standard would facilitate use of the problem sets by other research groups.

While there has been a historical disconnect between the research and applications oriented planning communities, a number of planning-oriented domains and testbeds have been made available. These testbeds have originated in the research community by experimenters as they either improve an existing planner's performance or define

algorithms that plan with ever more expressive action representations. For instance, the classical PRODIGY and UCPOP planners had multiple test domains included in their releases. The sensory GRAPHplan package [Graphplan] has accompanying domains as well. Our testbeds differ from this work due to our underlying focus on real world problems instead of planner test cases.

On the planner comparison side, many planner optimization papers report planner performance on a number of benchmark problems, and the most realistic of these is a logistics problem to move packages around an artificial map. Additionally, the AIPS 98 [McDermott 2000] and 2000 [Bacchus] planning competitions had problems defined in a standard modeling language called PDDL, and test domains with problem generators and plan simulators were used. Our domains are different than these previously released testbeds in that they are derived from actual space mission operations problems and address integrated planning and scheduling with metric time, resources, functional dependencies, and optimization (although a number of AIPS 2000 domains had some of these elements).

The part-machining domain [Gil 1991] and the elevator control domain [Koehler & Schuster 2000] were motivated by real problems. The part machining was an attempt to extract domain knowledge about how to turn a mass of metal into a machined part and encode it into a PRODIGY domain. Similarly, the elevator control domain involved taking a set of services and constraints and encoding them in PDDL to be solved by planners such as those participating in the AIPS competitions. One of the results from these efforts involved determining where the established modeling language cannot represent a desired feature of the real world problem. For instance, the elevator control problem has a capacity constraint that PDDL could not represent. Our work differs from this research in two places. We do not avoid time and other metric constraints, and we altered our simulators to facilitate experimenting with interleaved planning and execution.

A number of additional pure scheduling benchmarks exist [Fox & Ringer] as well as makespan benchmarks. These are designed to be more manufacturing and enhanced job-shop scheduling problems. In contrast, our work emphasizes the integrated planning and scheduling inherent in space mission operations.

Other research on interleaved planning and execution has resulted in shared testbeds like tileworld, truckworld, and the phoenix testbeds [Hanks et al. 1993]. These worlds were generated as benchmark cases for agent design within a multi-agent context. While tileworld and truckworld were relatively simplistic testbeds for testing agent systems, the phoenix testbed focused on a forest firefighting domain. Each of these testbeds offered defining problems with varying complexity, but only the phoenix testbed had an underlying operations scenario like our testbeds. The Robocup rescue project [Kitano et al 2001] also focuses on providing testbeds with an underlying operations scenario. It targets distribution of a complex multi-agent simulation environment. This environment has the potential to provide an extremely rich planning and scheduling testbed.

This paper introduced two benchmark problem sets based on actual space mission operations. Each benchmark problem set includes problem generators, declarative specification of the problem(s), and one or more simulations. The first mission is the DATA-CHASER, which demonstrated the ability of automated mission planning to both reduce commanding effort and improve science return [Chien et al. 1999]. The second mission is the Citizen Explorer Mission (CX-1), which is currently being rescheduled for launch. We include three problem classes of increasing complexity (and realism): planning and scheduling with states and resources (PSSR), PSSR with functional dependencies, and PSSR with functional dependencies and plan quality.

The domain descriptions, problem generators, and simulators are available for download from a web site at the University of Colorado, which designed and built these spacecraft and missions (and operated DATA-CHASER). It is our hope that release of this information will help to focus the planning and scheduling research community on key issues in planning and scheduling including: domain model expressiveness, representing functional dependencies, and plan optimization.

7 References

1. Sherwood, R., et al. "Using ASPEN to Automate EO-1 Activity Planning," Proceedings 1998 IEEE Aerospace Conference, Aspen, Colorado, March, 1998.
2. Sherwood, R., Estlin, T., Chien, S., Rabideau, F., Engelhardt, B., Mishkin, A., and Cooper, B., "An Automated Rover Command Generation Prototype for the Mars 2001 Marie Curie Rover," SpaceOps 2000, Toulouse, France, June 2000.
3. Smith, B.D., Engelhardt, B.E., Mutz, D., "Automated Mission Planning for the Modified Antarctic Mapping Mission," Proceedings 2001 IEEE Aerospace Conference, Big Sky, Colorado, March, 2001.
4. Wilkins, D. and desJardins, M., "A Call for Knowledge-based Planning," AI Magazine, 11(1): 99-115, 2001.
5. Chien, S., Rabideau, G., Willis, J., and Mann, T., "Automating Planning and Scheduling of Shuttle Payload Operations," Artificial Intelligence Journal 114 (1999) 239-255.
6. Willis, J., Rabideau, G., Wilklow, C., "The Citizen Explorer Scheduling System," Proceedings of the IEEE Aerospace Conference, Aspen, CO, March 1999.
7. Sensory GraphPlan home page, http://www.cs.washington.edu/ai/sgp .html
8. McDermott, D. The 1998 AI Planning Systems Comp. AI Mag 21(2), 2000.
9. Bacchus, F. The AIPS-00 Planning Competition. http://www.cs.toronto.edu/aips2000
10. Gil, Y., "*A Specification of Manufacturing Processes for Planning,*" CMU-CS-91-179.
11. Koehler, J. and Schuster, K., "Elevator Control as a Planning Problem," Proc 5th Intl Conf on Art Intelligence Planning Systems, Breckenridge, CO. April, 2000.
12. B. Fox and M. Ringer, Resource Constrained Scheduling Problem Home Page, http://www.neosoft.com/~benchmrx/
13. S. Hanks, M. E. Pollack, and P.Cohen, "Benchmarks, Testbeds, Controlled Experimentation, and the Design of Agent Architectures, AI Magazine, 14(4):17-42, 1993. (also see http://www.cs.pitt.edu/~pollack/distrib/tileworld.html)
14. H. Kitano and S. Tadakoro, "RoboCup Rescue: A Grand Challenge for Multiagent and Intelligent Systems," AI Magazine, 11(1): 39-52, 2001. (also see http://www.r.cs.kobe-u.ac.jp/robocup-rescue/)