Compressed Large-scale Activity Scheduling and Planning (CLASP) Applied to DESDynI

Russell Knight and Steven Hu

Jet Propulsion Laboratory, California Institute of Technology
<first name>.<last name>@jpl.nasa.gov

Abstract

We demonstrate the use of compression techniques to accommodate scheduling of large swath coverage campaigns in the context of the Deformation, Ecosystem Structure, and Dynamics of Ice (DESDynI) mission. The underlying scheduler is a simple squeaky-wheel optimizer, which requires a greedy solver as a component. Normally, this approach poses no problems for conventional computers, but when attempting to schedule several years' worth of observations in detail, the schedule alone fails to fit in memory, thus our use of compression techniques.

Introduction

The DESDynI mission is an Earth orbiting mission flying a combination of SAR and LIDAR. The goal of the mission to gather data for science disciplines. These disciplines are Deformation, Ecosystem Structure, and Dynamics of Ice.

Our goal is to determine the timing of operating mode changes for the instruments onboard the spacecraft as well as pointing of the spacecraft such that the science goals of each discipline are met.

Consider the problem of scheduling the orientation and on/off times of a spacecraft or collection of spacecraft such that we adequately cover as many target points as possible. But, we must not oversubscribe memory or energy.

Many aspects of the problem need to be considered.

- Cost in terms of battery and energy when the instrument is on or off.
- Time to transition from one orientation to the next.
- Rate, time, and duration of downlinks and sun exposures.

For small versions of this problem, we can employ standard scheduling techniques. In our work, we are using a squeaky wheel optimizer.

But as the problem size grows, standard propagation techniques are too costly, and even estimating the final quality of a partial schedule becomes practically intractable. If the problem size increases such that it must be swapped out to disk, then even greedy solutions become intractable.

Copyright © 2009, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

The Scheduler

We note that many of these problems are cyclic in nature. That is, for spacecraft, we often have "repeat passes" where we overfly the same points at some predetermined interval. If we can take advantage of this cyclic nature in our representations, then we can compress the problem into a single-cycle problem. Unfortunately, memory and energy profiles are not guaranteed to match any cyclic representation, and either must be represented explicitly or compressed as well. Also, some observation campaigns retain cyclic observation availability but not cyclic behavior—that is, the same observations are not taken at each repeat pass. Again, these must be compressed.

Problem Representation

In general, we assume a scheduling problem characterization as such:

- a set of points P,
- a set of observations opportunities O each $o \in O$ consists of a start ($o.start$), and a duration ($o.duration$),
- a function $obs(p \in P)$ that returns the subset of O whose elements positively affect coverage of p,
- a function $coverage(p, sol \subseteq obs(p))$ that determines the level of coverage of p given sol,
- a bound on coverage $minQuality$ that must be met for a point to be considered covered,
- a bound on memory $memory$,
- a bound on energy $energy$,
- a rate at which memory is used while the instrument is on $memRate$,
- a rate at which energy is used while the instrument is on $useWatts$,
- a rate at which memory is recovered during a downlink $downRate$,

- a rate at which energy is collected while in sunlight *sunWatts*,
- a function that returns the minimum duration to transition from one observation orientation to another $trans(o_1 \in O, o_2 \in O)$,
- a set of sun exposures S where each $s \in S$ consists of a start ($s.start$) and a duration ($s.duration$),
- a set of downlinks D where each $d \in D$ consists of a start ($d.start$) and a duration ($d.duration$).

Our goal is to select $p.sol$ subset of $obs(p)$ for each $p \in P$ such that the number of $p \in P$ that $coverage(p, p.sol) \geq minQuality$ is maximized, yet at no time do we oversubscribe memory or energy.

Since *coverage* is a black-box function, optimizing this in general is problematic. But if we make the assumption that coverage is monotone (adding another o to $p.sol$ never makes it worse), then we can perform local optimization based on approximating that adding the o that gives us the best improvement now might give us good improvement overall.

The squeaky-wheel optimization algorithm initially labels each target with a scheduling priority. This priority will increase as the scheduler fails to schedule the target.

Then, iteratively, for each target in scheduler priority, the scheduler attempt to schedule the target. Any target not scheduled gets its scheduling priority bumped up. After each scheduling session, the quality of the current schedule is checked against the quality of the best schedule found thus far. If the current schedule quality exceeds that of the previous best schedule, then we replace the best schedule with the current schedule.

If we run out of time or if the little improvement seems likely (stagnation), we stop optimization and return the best schedule as the solution.

This process is illustrated in Figure 1.

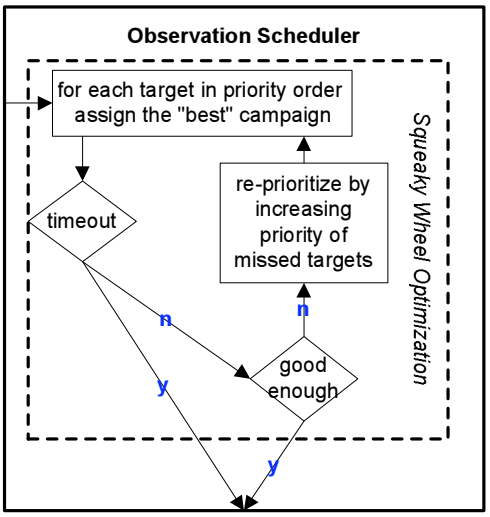

Figure 1 Squeaky Wheel Optimizer

Detecting Stagnation

As we iterate over various schedules, we see a pattern of increasing and decreasing quality over time of the current schedule. Figure 2 illustrates such a cycle. To estimate when stagnation has occurred, we set a limit of the number of quality cycles without improvement by the system, and when this limit is exceeded, we presume (possibly falsely) that the search is stagnant and not likely to lead to improvement. Of course, this is a heuristic and is provably incorrect, thus care should be taken before selecting quality cycle limits.

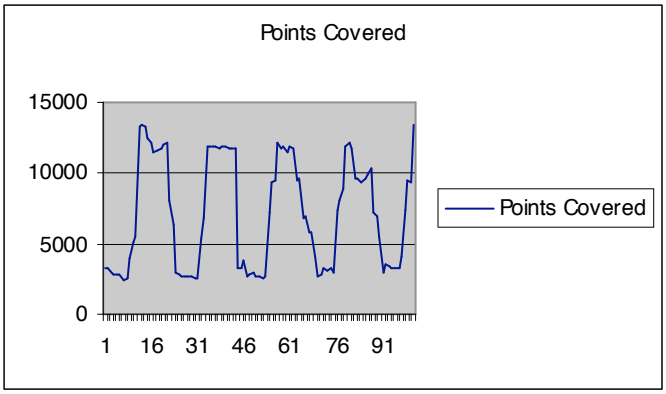

Figure 2 Schedule Quality over Time (Vertical is number of Points, Horizontal is Iteration)

Schedule Compression

In general, our technique for compression is to save a succinct uncompressed schedule and summarizations of profile data that are implied by the succinct schedule, up to available memory.

The succinct schedule data structure consists of a map keyed by time of instrument modes.

Energy, battery level, and duty cycle profiles are summarized. In general, the schedule is iteratively divided starting from the beginning and each spanned area is summarized with the appropriate characteristics. The summaries are updated when scheduling occurs without being recomputed until we detect a potential false negative scheduling result. Then the actual profile is computed from the nearest waypoint to determine legal scheduling options. This "uncompression" of the schedule can be costly, and in the worst case can induce a walk of half of the succinct schedule, which in turn can take up to hundreds of megabytes. Thus, we leverage the fact that we know that the squeaky-wheel optimizer will be looking in detail at earlier candidate times for observation schedules.

What results is a characteristic dynamic trading-off of abstract schedule information for detailed information in the current context of observation scheduling.

DESDynI Domain

As mentioned previously, the DESDynI domain consists of scheduling observations in support of 3 science disciplines:

Deformation: Surface deformation is linked directly to earthquakes, volcanic eruptions, and landslides. Observations of surface deformation are used to forecast

the likelihood of earthquakes occurring as a function of location, as well as predicting both the place and time that volcanic eruptions and landslides are likely. Advances in earthquake science leading to improved time-dependent probabilities would be significantly facilitated by global observations of surface deformation, and could result in significant increases in the health and safety of the public due to decreased exposure to tectonic hazards. Monitoring surface deformation is also important for improving the safety and efficiency of extraction of hydrocarbons, for managing our ground water resources, and, in the future, providing information for managing CO2 sequestration. Example targets are illustrated in Figure 3.

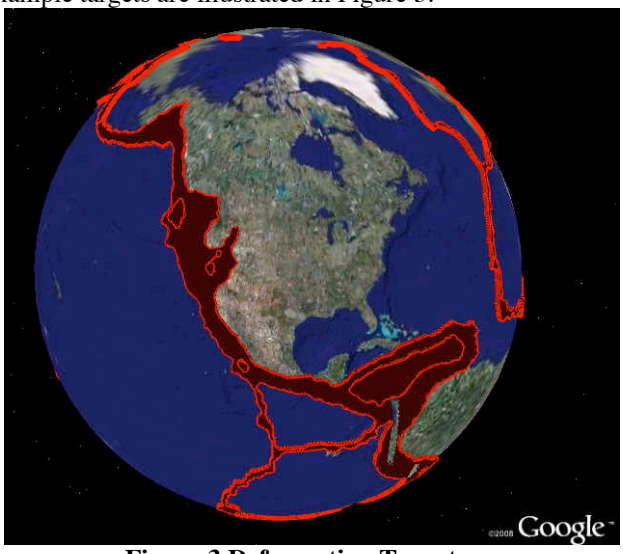

Figure 3 Deformation Targets

Ecosystem Structure: Radar and lidar measurements will help us understand responses of terrestrial biomass, which stores a large pool of carbon, to changing climate and land management. Benefits would include the potential for development of more effective land-use management, especially as climate-driven effects become more pronounced. Example targets are illustrated in Figure 4.

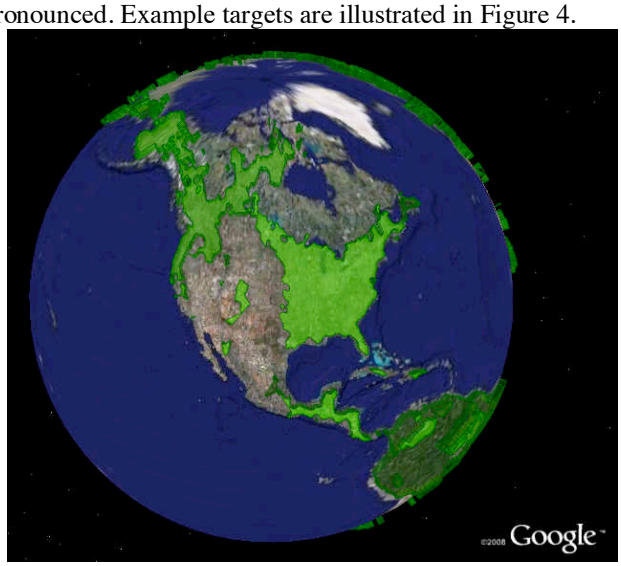

Figure 4 Ecosystem Structure Targets

Dynamics of Ice: The poorly-understood dynamic response of the ice sheets to climate change is one of the major sources of uncertainty in forecasts of global sea level rise. DESDynI's InSAR measurements of the variations in ice flow patterns and velocities provide important constraints on their dynamic response to climate change. This knowledge will help to determine how fast society must adapt to sea level changes - knowledge crucial in planning how to allocate scarce resources. Example targets are illustrated in xxx.

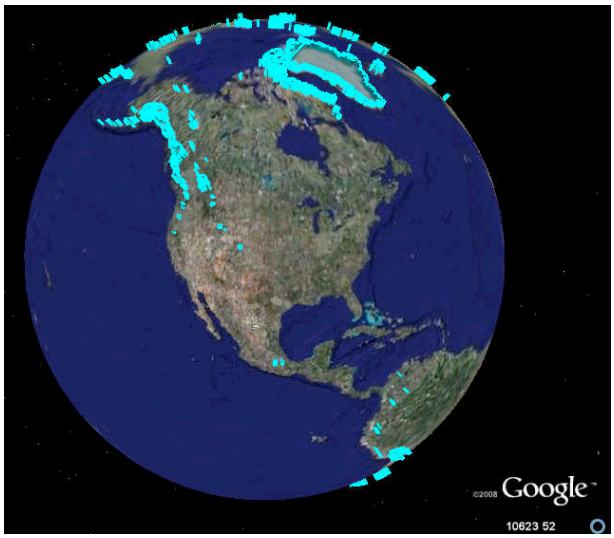

Figure 5 Dynamics of Ice Targets

As can be seen from our figures, we use Google Earth KML files to communicate targets and scheduling solutions to our systems and to our scientists.

These files are processed and result in a combined set of targets, as seen in Figure 6.

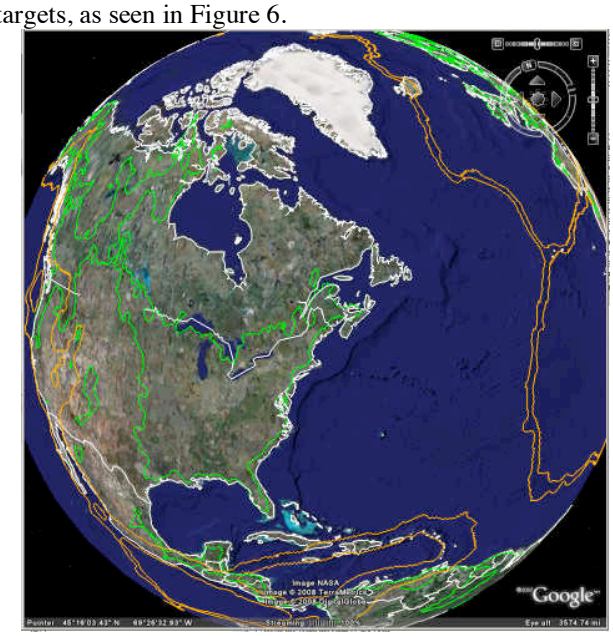

Figure 6 Combined Targets

Results

Our system then decomposes the entire set of targets into a set of target points, and attempts to gather each point

using the squeaky wheel optimization technique. We have already seen in Figure 2 the evolution of quality over time. Not surprisingly, the smaller the schedule, the faster reasoning takes place, thus it is faster to solve schedules that are of low quality than it is to solve schedules of higher quality. This is illustrated in Figure 7.

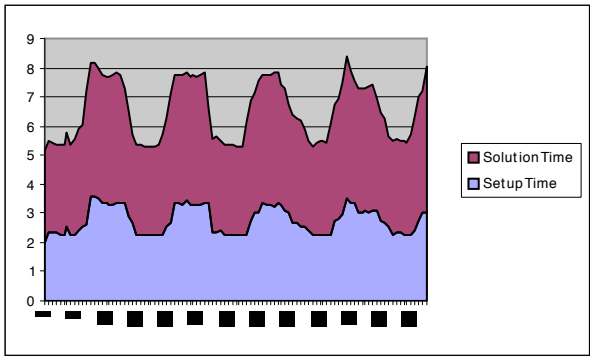

Figure 7 Solution and Setup Times (Vertical axis is minutes and horizontal axis is iteration)

Figure 8 illustrates a solution, with the gathered points being represented by pins.

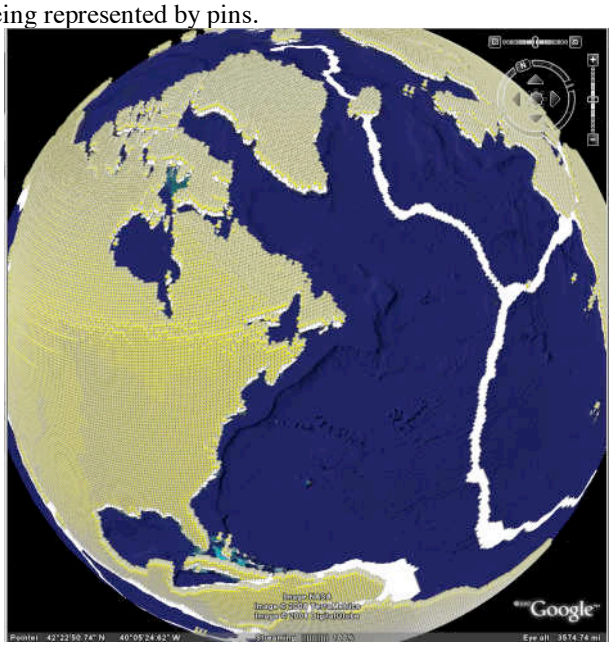

Figure 8 Solution

When solving the larger problem of the entire schedule over 330 days (30 cycles), we see that all possible points that can be covered are, and this solution only requires 15 minutes to setup and solve. This is a greater than 20 fold increase in speed over using non-compressed representations and an even greater speed-up over using other systems that require about a week to produce a single schedule.

Related Work

[Knight 2005] solves smaller problems of this sort optimally using a combination of branch and bound techniques combined with flow network approximations.

For a good example of a polyhedral solution to a combinatorial optimization problem having to do with satellite scheduling (formulated as a pick-up and delivery problem), see [Ruland 1986].

[Oddi 2003] solves a constrained-memory domain with fewer types of constraints called the Mars Express Memory Dumping Problem. The system uses a portfolio approach to solving the problem as formulated in a constraint-based framework. The portfolio consists of a tabu search strategy, a random sampling strategy, and a greedy strategy.

More general constraint-based frameworks for scheduling that have been applied to spacecraft operations include that of [Dungan 2002], [Ghallab 1994], and [Chien 2000]. In each of these, the problem is expressed as a set of constraints to be satisfied. In the case of [Dungan 2002], and [Ghallab 1994], the systems search the feasible space of domains in the constraint space. In the case of [Chien 2000] the system searches both the infeasible and feasible space of value assignments, using randomized local search.

Acknowledgements

This research was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology.

References

[Chien 2000] S. Chien, G. Rabideu, R. Knight, R. Sherwood, B Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, and D. Tran. “Automating space mission operations using automated planning and scheduling.” In *Proc. SpaceOps*, 2000.

[Corman *et al*] Thomas H. Corman, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to Algorithms*. McGraw-Hill, 1990.

[Dungan 2002] J. Dungan, J. Frank, A. Jonsson, R. Morris, and D. Smith. “Advances in Planning and Scheduling of Remote Sensing Instruments for Fleets of Earth Orbiting Satellites.” *Earth Science Technology Conference*, 2002. Pasadena, California.

[Frank 2000] J. Frank. "SOFIA's Choice: Automating the Scheduling of Airborne Observations" Proceedings of

the 2d NASA Workshop on Planning and Scheduling for Space, March 2000.

[Garey and Johnson 1979] M.R. Garey and D.S. Johnson. Computers and Intractability. W.H. Freeman and Company, San Francisco, 1979.

[Ghallab 1994] M. Ghallab and H. Laruelle, "Representation and control in IxTeT, a temporal planner", in *Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems* (AIPS-94), pp. 61-67, Chicago, IL, (1994). AAAI Press, Menlo Park.

[Knight 2005] R. Knight, "Solving the Constrained Coverage Problem using Flow Networks as Linear Program Approximations," Ph.D. Dissertation, University of California, Los Angeles, 2005.

[Karp 1972] R. M. Karp "Reducibility among combinatorial problems." In R. E. Miller and J. W. Thatcher (eds.) *Complexity of Computer Computations*, Plenum Press, New York, 85-103.

[Muraoka 1998] H. Muraoka, R. H. Cohen, T. Ohno, and N.Doi. "ASTER Observation Scheduling Algorithm." SpaceOps 98. 1998, 1-5 June, Tokyo, Japan.

[Oddi 2003] A. Oddi, N. Policella, A. Cesta and G. Cortellessa. "Generating High Quality Schedules for Spacecraft Memory Downlink Problems." *Ninth International Conference on Principles and Practice of Constraint Programming*, 29 September - 3 October, 2003, Kinsale, County Cork, Ireland.

[Papadimitriou and Steiglitz 1982] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, NJ, 1982.

[Ruland 1986] K. Ruland. Polyhedral solution to the pickup and delivery problem. Washington University, Sever Institute of Systems Science and Mathematics. http://rodin.wustl.edu/~kevin/dissert/dissert.html (Dissertation). St. Louis Missouri, 1995.

[Schrijver 1986] A. Schrijver. Theory of Linear and Integer Programming, Wiley, 1986.