Deep Space Network Scheduling Using Evolutionary Computational Methods

Alexandre Guillaume[1], Seugnwon Lee, Yeou-Fang Wang, Richard J. Terrile
Jet Propulsion Laboratory, 4800 Oak Grove drive, 91109-8099 Pasadena CA
[1]alexandre.guillaume@jpl.nasa.gov , tel: 818-393-6899

Abstract- The Deep Space Network (DSN) is an international network of antennas that supports all of NASA's deep space missions. The allocation of the DSN resources should be optimally scheduled to satisfy the requirements of as many missions as possible. The nature of the DSN scheduling problem is that it involves over 16 antennas and about 30 missions, the resources are oversubscribed, there are many mission-specific constraints, some schedule items have complex relationships, and there are over 500 schedule items per week for 26 weeks of schedules with hundreds of changes. Currently, the DSN schedules are manually and iteratively generated through several meetings to resolve conflicts. In an attempt to ease the burden of the DSN scheduling task, we have applied evolutionary computational techniques to the DSN scheduling problem. These methods provide a decision support system by automatically generating a population of optimized schedules under varying conflict conditions. These schedules are used to decide the simplest path to resolve conflicts as new scheduled items are added or changed along the scheduled 26 weeks. An example would be to optimize scheduled items such that overlap is not only minimized but is also restricted to a minimum number of impacted missions. This would have the cost advantage of reducing the number of meeting required to formulate a conflict free schedule.

The paper presents the specific approach taken to formulate the problem in terms of gene encoding, fitness function, and genetic operations. The genome is encoded such that a subset of the scheduling constraints is automatically satisfied. Several fitness functions are formulated to emphasize different aspects of the scheduling problem. The optimal solutions of the different fitness functions demonstrate the trade-off of the scheduling problem and provide insight into a conflict resolution process.

TABLE OF CONTENTS

1. INTRODUCTION

The Deep Space Network (DSN) is an international network of antennas that supports all of NASA's deep space missions. This network consists of three communication facilities spaced by 120 degrees (in longitude) from each other: the Goldstone Deep Space Communications Complex in California, the Madrid Deep Space Communications Complex in Spain and the Camberra Deep Space Communications Complex in Australia. This strategic placement permits constant observation of spacecraft as the Earth rotates.
Each facility houses five or six antennas.

2. CURRENT DSN SCHEDULING PROCEDURE

Hence, once all the requirements have been centralized, the establishment of the schedule is equivalent to an optimization problem. There are several efficient numerical techniques to solve this kind of problems: genetic algorithms, simulated annealing, hill climbing, … Several reasons motivated our choice of technique. First, over the years, genetic algorithms have been used with great success to solve scheduling problems. Secondly, this technique is well suited to explore complex search spaces such as this one. Finally, we have both the expertise in genetic algorithms and the knowledge of the DSN operation. In this work, we present an evolutionary approach to the scheduling problem of the Deep Space Network.

Any user can have a remote access to the current forecast week and to a routine that looks for conflicts between a newly proposed schedule and the forecast week. This centralized service is currently located at the Jet Propulsion Laboratory in Pasadena.

3. OPTIMIZATION WITH A GENETIC ALGORITHM

We first build the genome with the essential characteristics of the forecast week. Each task is characterized by an identification number, a duration, a set up and tear down period and schedulable intervals (view periods). The algorithm looks for candidate solutions (schedules) among the different view periods. In this case, the gene is simply an integer number (in minute) chosen randomly between zero and the sum of all view periods. A tracking number is assigned to each period so that when a random number is chosen within the interval, it can be traced back to a given schedulable interval. In other words, this tracking number implements the mapping between the phenotype and the genotype. There are typically about 500 tasks. As a first step towards establishing an efficient scheduling tool for the Deep Space Network, we only consider single events, e.g. disregard events involving several, either, antennas (antenna arraying) or spacecrafts (Multiple Spacecraft Per Antenna, MSPA). There are usually 140 of those normal tasks that also possess schedulable intervals. Each chromosome has therefore around 140 genes.

Before discussing further the specifics of the genetic algorithm, we describe the implementation in details as the optimization of the algorithm was greatly influenced by it. In order to use the existing infrastructure, we use a dedicated web service located at JPL. The public interface to this service is achieved through a WSDL, or Web Services Description Languages. The SOAP protocol is used to exchange XML files between the users and the service. The genetic algorithm is coded in C/C++. We used the gSOAP toolkit to bind the XML data to a C++ structure. So the genetic algorithm run on a client machine that makes conflict calls to the server at each iteration. Using the DSN framework presents the advantage that our code could readily be inserted into the current DSN schedule tools. Having, however, the computation and the conflict check performed at two different points introduce a serious overhead as approximately two thirds of each iteration is spent on communication.

Each solution should fulfill numerous scheduling objectives: maximize the number of tasks without conflicts, maximize the mission coverage, minimize the number of conflicts, minimize schedule items overlaps. We initially built different fitness functions for each objective. We then implemented a multi-objective approach by simply building a fitness function as a linear combination of different objectives. This allows us to fine tune the fitness function by adjusting the relative weights of each component. The number of tasks without conflict is obtained by subtracting the number of conflicts from the number of tasks in the corresponding schedule. To avoid redundancy, we count the number of conflicts in a different fashion to evaluate the third objective function. The conflicts check returns a detail list of antennas associated with each conflict. We count the number of those conflicts. The total mission coverage is obtained by adding the duration of each task without conflict. Overlaps are determined by the so-called facility conflicts.

The program starts by requesting the current forecast week, which is the schedule for the coming week which has been optimized by the DSN scheduling team. Most of the time, the forecast week is therefore conflict free. The performance of our code can be simply evaluated by comparing our solution to this original solution. The initial population is chosen, as it is customary in genetic algorithms, by randomly picking the genes in the pool of possible solutions (schedulable intervals).

We also implemented an evolution strategy for the mutation step. The size of the mutation steps increased linearly with the number of generations.

4. RESULTS

We present the results obtained from the forecast for the second week of October. There were 412 tasks for this week among which 153 were normal with schedulable intervals. The fitness of the initial schedule was evaluated before the initialization of the genetic algorithm. There were 343 tasks without conflict, 10.5 week mission coverage, 706 conflicts and 0 facility conflicts. These values are indicated with a red horizontal line as a reference on Figure 1.

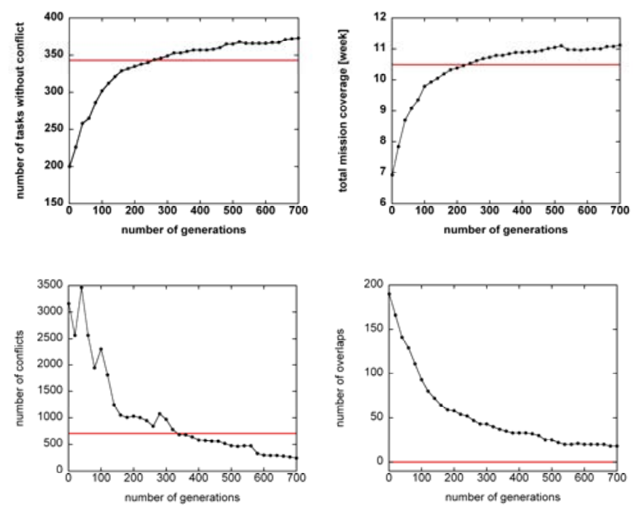

Figure 1: detail of the evolution of the four different objectives composing the fitness function. The red line indicates the fitness of the initial schedule.

The genetic algorithm clearly outperforms the initial schedule for three of the four objectives. However, even though the relative weight for the minimization of the number of overlaps is eight times higher than the other objectives, the number of overlaps is still higher (18) than the initial solution (0). It is also worth noticing that one overlap between a given task *i* and task *j* appears as two overlaps in our count (task *i* with task *j*, and task *j* with task *i*). So, there are really 9 facility conflicts in the solution at the 700th generation. Moreover, the program really operates according to the general rules we described. In reality, there are exceptions to these rules that permit some flexibility to resolve conflicts.

5. FUTURE PROSPECTS AND IMPROVEMENTS

There are many improvements possible. Since the time to perform a simulation was mostly spent in communication between the client and the server, a substantial amount of time (about two thirds) could be gained by moving the conflict check on the same machine that is running the program. This improvement, however desirable, would only speed up the adjustment of the program. For this work we used a mutation step that was calculated with respect to the upper bound of the gene. However, it possible to achieve the same upper bound of the genes, *e.g.* the same sum of all schedulable intervals, with two different series of schedulable intervals. The sum of a few intervals can be equal to the sum of more, but smaller intervals. A percentage doesn't capture this possibility.
Since this program is supposed to, at least, mimic the current scheduling process, and at most propose alternative

solutions, it would be logical to include most of the constraints and objectives currently used.

6. Conclusion

We proposed an evolutionary computational method to help the scheduling process of the Deep Space Network. Using a multi-objective approach, we were able to consistently obtain solutions that would improve the current best solution for three out of the four used objectives. Considering, the proximity of our solution to the ideal solution, for the last objective (minimizing the number of facility conflicts), our program already constitutes a very interesting tool for the DSN scheduling process. Moreover, considering the simplicity of the approach (limited number of objectives and constraints), we are optimistic that we will be able to propose a tool in the near future that will both improve the quality of the current schedule and probably generate alternative schedules meeting all the requirements.

References

[1] References should be in 10-point type.

[2] Michael Hammer and James Champy, Reengineering the Corporation, New York: Harper Business, 1993.

[3] G. Edward Bryan, "Not All Programmers Are Created Equal," 1994 IEEE Aerospace Applications Conference Proceedings, February 5–12, 1994.

[4] G. Edward Bryan, "CP-6: Quality and Productivity Measures in the 15-Year Life Cycle of an Operating System," Software Quality Journal 2, 129–144, June 1993.

[5] Aerospace Conference Web site http://www.aeroconf.org/

Biography

Yeou-Fang Wang is a member in the Planning and Execution Systems Section at the Jet Propulsion Laboratory. He has developed methodologies and tools for antenna load forecasting and scheduling systems for NASA's Deep Space Network. He has authored several papers in the fields of artificial Neural Networks and scheduling.. Dr. Wang's recent work includes automatic schedule generation algorithm development using computational intelligence, collaborative engineering, service oriented software architecture design, and multi-agent systems application. He has BS degree in Control Engineering from the National Chiao-Tung University of Taiwan and MS and PhD in Electrical and Computer Engineering from the University of California, Irvine.