Distributed Network Scheduling

Bradley J. Clement, Steven R. Schaffer
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, M/S 126-347
Pasadena, CA 91109 USA
+1-818-393-4729, +1-818-354-8637

Bradley.J.Clement@jpl.nasa.gov, Steven.R.Schaffer@jpl.nasa.gov

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*coherence and coordination, multiagent systems*

General Terms

Performance, Design, Experimentation

Keywords

scheduling, network, space, Mars, rover

1. INTRODUCTION

We investigate missions where communications resources are limited, requiring autonomous planning and execution. Unlike typical networks, spacecraft networks are also suited to automated planning and scheduling because many communications can be planned in advance. Because the network of spacecraft can represent multiple missions, missions will be reluctant to give up control of the spacecraft. Because communication among spacecraft is often intermittent (due to orbital and resource constraints), a spacecraft that can make scheduling decisions autonomously will be more responsive to unexpected events. Thus, a centralized planning system will not be sufficient to enable reactive communications, so we propose a distributed network scheduling system.

The software automatically negotiates the rescheduling of these communications with other spacecraft while respecting constraints with communication resources (such as memory and transceiver availability). Each node (spacecraft) tracks only its own communication activities and makes its own scheduling decisions but can propose communications with others. It provides an interface for a user or automated process to request communication service and to receive a reservation with updates on the expected or resulting quality of service (QoS). The communication needed to coordinate planning ("meta-communications") are not scheduled by the system because the overhead is insignificant compared to science image transfers. However, simulations of the system limit this communication to available view periods.

While prior work on distributed scheduling has been applied to remote spacecraft networks [1], the software reported here focuses on modeling communication activities in greater detail and including quality of service constraints.

AAMAS'05, JULY 25-29, 2005, UTRECHT, NETHERLANDS.

2. COMMUNICATION REQUESTS, RESERVATIONS, AND STATUS

An application or user requests future communication from the network by providing values for the following variables:

- `int` **`id`** – index for tracking
- `string` **`source`** – who is sending data
- `string` **`destination`** – who is receiving the data
- `int` **`size`** – estimate of size of data to be sent in Kbits
- `real` **`bandwidth_min`** – minimum required bandwidth Kbits/s
- `real` **`bandwidth_max`** – maximum usable bandwidth in Kbits/s
- `real` **`priority`** – importance of fulfilling request (larger numbers indicate greater importance)
- `int` **`start_time_min`** – minimum requested start time of communication
- `int` **`start_time_max`** – maximum requested start time of communication
- `int` **`duration_min`** – minimum needed time duration of initial data transmission
- `int` **`duration_max`** – maximum requested time duration of initial data transmission
- `int` **`delivery_time_min`** – minimum required delivery time
- `int` **`delivery_time_max`** – maximum requested delivery time
- `bool` **`progressive`** – whether data is recreated as it is received (= true) or transmission is only valuable when completed, i.e. all or nothing (= false)
- `real` **`loss_overall`** – maximum percentage loss tolerance of overall data
- `real` **`loss_per_block`** – maximum percentage loss tolerance for any block
- `real` **`loss_block_size`** – size of block for which the loss tolerance is specified

- `string` **`protocol`** – what protocol(s) should be used for transmission and with what options (e.g. CFDP -noack); this string has no generic structure and is to be generated and interpreted by the network protocol stack.

Upon receiving a request, the network will schedule ("reserve") the communication and reply with the expected QoS for the same variables above and a real-valued `percent_delivered` variable, indicating the percentage of the data delivered or expected to be delivered. Status during and upon completion of execution is also reported through the same construct.

3. LOCAL SCHEDULING

We use the ASPEN planning system [2] to schedule communications according to constraints on memory, transceiver availability, and available windows of communication between scheduling nodes (spacecraft). The main activities scheduled are `send`, `receive`, and `relay`, for transmitting, receiving, and relaying data files. Segmentation and reassembly of files is supported for when files are too large to be sent in available communication windows. In addition, scheduling supports cut-through switching, receiving and relaying a file simultaneously when multiple transceivers are available. The timing and duration of activities takes into account constraints on communication delay and bandwidth. While quality of service estimates/status is propagated through the network, the scheduler currently does not handle failures, such as over-tolerance data loss.

Scheduling avoids conflicts with the following resource and state variables:

memory – Decisions about when to store and delete data are based on memory availability.

data – It may be important to keep track of whether particular data files are stored or deleted in case one needs retransmission due to an unexpected failure.

antenna(s) – Spacecraft can only communicate with one (or maybe two) others at a time.

communication windows – Spacecraft can only communicate

4. DISTRIBUTED SCHEDULING

Scheduling is distributed by propagating information through the network to nodes that are affected and by giving each node some level of decision-making authority with respect to local scheduling. We use Shared Activity Coordination (SHAC) [1] to implement this.

SHAC is an interface between planning/scheduling systems, a general algorithm for coordinating distributed planning, and a framework for designing and implementing more specific distributed planning algorithms. Within SHAC, a shared activity is an activity that some set of planners must collectively schedule. It can be a coordinated measurement, a team plan in which planners have different roles, a use of shared resources, or simply an information sharing mechanism. Planners are coordinated when they reach consensus on the shared activity. Consensus is achieved when they agree on values for members of the shared activity structure:

Parameters: Shared variables (e.g. start time, duration, bandwidth)

Constraints: Each planner's constraints on parameter values

Roles: Subset of planning agents assigned to roles

Permissions: Variables that determine how each planner is allowed to add, remove, and modify a shared activity

Protocols (distributed planning algorithms) specify how constraints, roles, and permissions of the shared activities change over time and are used to resolve conflicts among the planners.

For distributed network scheduling, we specified shared activities between pairs of spacecraft mapping transmit activities of one spacecraft to relay activities in another. Shared parameters include those of the request/reservation. The roles specify which local activity (`transmit` or `relay`) corresponds to each agent (spacecraft) potentially participating. The transmitter is assigned a delegation protocol for choosing a spacecraft to relay the data.

5. EVALUATION

A prototype network scheduling system was implemented for communication models of MER-A, MER-B, MGS (Mars Global Surveyor), Odyssey, and Mars Express. Theoretical and experimental results include identifying a maximum opportunity of improving rover traverse exploration rate by a factor of three; a simulation showing reduction in one-way delivery times from a rover to Earth from as much as 5 to 1.5 hours; simulated response to unexpected events averaging under an hour onboard; and ground schedule generation ranging from seconds to 50 minutes for 15 to 100 communication goals.

6. ACKNOWLEDGMENTS

Work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

7. REFERENCES

[1] B. Clement, A. Barrett. Continual Coordination through Shared Activities. 2nd International Conference on Autonomous and Multi-Agent Systems (AAMAS 2003). Melbourne, Australia. July 2003.

[2] S. Chien, G. Rabideau, R. Knight, R. Sherwood, B. Engelhardt, D. Mutz, T. Estlin, B. Smith, F. Fisher, T. Barrett, G. Stebbins, D. Tran , ASPEN - Automating Space Mission Operations using Automated Planning and Scheduling, Proc. SpaceOps 2000, Toulouse, France, June 2000.