This work was done by Terrance L. Huntsberger, Andrew B. Howard, Hrand Aghazarian, and Arturo L. Rankin of Caltech for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1).

In accordance with Public Law 96-517, the contractor has elected to retain title to this invention. Inquiries concerning rights for its commercial use should be addressed to:

Innovative Technology Assets Management
JPL
Mail Stop 202-233
4800 Oak Grove Drive
Pasadena, CA 91109-8099
E-mail: iaoffice@jpl.nasa.gov

Refer to NPO-47533, volume and number of this NASA Tech Briefs *issue, and the page number.*

LEGION: Lightweight Expandable Group of Independently Operating Nodes

LEGION is a lightweight C-language software library that enables distributed asynchronous data processing with a loosely coupled set of compute nodes. Loosely coupled means that a node can offer itself in service to a larger task at any time and can withdraw itself from service at any time, provided it is not actively engaged in an assignment. The main program, i.e., the one attempting to solve the larger task, does not need to know up front which nodes will be available, how many nodes will be available, or at what times the nodes will be available, which is normally the case in a “volunteer computing” framework. The LEGION software accomplishes its goals by providing message-based, inter-process communication similar to MPI (message passing interface), but without the tight coupling requirements. The software is lightweight and easy to install as it is written in standard C with no exotic library dependencies.

LEGION has been demonstrated in a challenging planetary science application in which a machine learning system is used in closed-loop fashion to efficiently explore the input parameter space of a complex numerical simulation. The machine learning system decides which jobs to run through the simulator; then, through LEGION calls, the system farms those jobs out to a collection of compute nodes, retrieves the job results as they become available, and updates a predictive model of how the simulator maps inputs to outputs. The machine learning system decides which new set of jobs would be most informative to run given the results so far; this basic loop is repeated until sufficient insight into the physical system modeled by the simulator is obtained.

This work was done by Michael C. Burl of Caltech for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1).

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47910.

Real-Time Projection to Verify Plan Success During Execution

The Mission Data System provides a framework for modeling complex systems in terms of system behaviors and goals that express intent. Complex activity plans can be represented as goal networks that express the coordination of goals on different state variables of the system. Real-time projection extends the ability of this system to verify plan achievability (all goals can be satisfied over the entire plan) into the execution domain so that the system is able to continuously re-verify a plan as it is executed, and as the states of the system change in response to goals and the environment.

Previous versions were able to detect and respond to goal violations when they actually occur during execution. This new capability enables the prediction of future goal failures; specifically, goals that were previously found to be achievable but are no longer achievable due to unanticipated faults or environmental conditions. Early detection of such situations enables operators or an autonomous fault response capability to deal with the problem at a point that maximizes the available options.

For example, this system has been applied to the problem of managing battery energy on a lunar rover as it is used to explore the Moon. Astronauts drive the rover to waypoints and conduct science observations according to a plan that is scheduled and verified to be achievable with the energy resources available. As the astronauts execute this plan, the system uses this new capability to continuously re-verify the plan as energy is consumed to ensure that the battery will never be depleted below safe levels across the entire plan.

In particular, this enables an execution system to predict problems such as resource exhaustion before they occur. The models are expressed and executed in a way that can be optimized for real-time use in an embedded system.

This work was done by David A. Wagner, Daniel L. Dvorak, Robert D. Rasmussen, Russell L. Knight, John R. Morris, Matthew B. Bennett, and Michel D. Ingham of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov.

This software is available for commercial licensing. Please contact Daniel Broderick of the California Institute of Technology at danielb@caltech.edu. Refer to NPO-47734.

Automated Performance Characterization of DSN System Frequency Stability Using Spacecraft Tracking Data

This software provides an automated capability to measure and qualify the frequency stability performance of the Deep Space Network (DSN) ground system, using daily spacecraft tracking data. The results help to verify if the DSN performance is meeting its specification, therefore ensuring commitments to flight missions; in particular, the radio science investigations. The rich set of data also helps the DSN Operations and Maintenance team to identify the trends and patterns, allowing them to identify the antennas of lower performance and implement corrective action in a timely manner.

Unlike the traditional approach where the performance can only be obtained from special calibration sessions that are both time-consuming and require manual setup, the new method taps into the daily spacecraft tracking data. This new approach significantly increases the amount of data available for analysis, roughly by two orders of magnitude, making it possible to conduct trend analysis with good confidence.

The software is built with automation in mind for end-to-end processing. From the inputs gathering to computation analysis and later data visualization of the results, all steps are done automatically, making the data production at near zero cost. This allows the limited engineering resource to focus on high-level assessment and to follow up with the exceptions/deviations.

To make it possible to process the continual stream of daily incoming data without much effort, and to understand the results quickly, the processing needs to be automated and the data summarized at a high level. Special attention needs to be given to data gathering, input validation, handling anomalous conditions, computation, and present-