geometrical variations include significant out-of-plane shifts, as well as in-plane shifts. Further, the GEO satellite is almost continuously in view of a lunar halo orbiter. High-fidelity simulations demonstrate that LiAISON technology improves the navigation of GEO orbiters by an order of magnitude, relative to standard ground tracking. If a GEO satellite is navigated using LiAISON-only tracking measurements, its position is typically known to better than 10 meters. If LiAISON measurements are combined with simple radiometric ground observations, then the satellite's position is typically known to better than 3 meters, which is substantially better than the current state of GEO navigation.

There are two features of LiAISON that are novel and advantageous compared with conventional satellite navigation. First, ordinary satellite-to-satellite tracking data only provides relative navigation of each satellite. The novelty is the placement of one navigation satellite in an orbit that is significantly perturbed by both the Earth and the Moon. A navigation satellite can track other satellites elsewhere in the Earth-Moon system and acquire knowledge about both satellites' absolute positions and velocities, as well as relative positions and velocities in space.

The second novelty is that ordinarily one requires many satellites in order to achieve full navigation of any given customer's position and velocity over time. With LiAISON navigation, only a single navigation satellite is needed, provided that the satellite is significantly affected by the gravity of the Earth and the Moon. That single satellite can track another satellite elsewhere in the Earth-Moon system and obtain absolute knowledge of both satellites' states.

This work was done by Jeffrey S. Parker and Rodney L. Anderson of Caltech; and George H. Born, Jason M. Leonard, Ryan M. McGranaghan, and Kohei Fujimoto of the University of Colorado at Boulder for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1).

The software used in this innovation is available for commercial licensing. Please contact Dan Broderick at Daniel.F.Broderick@jpl.nasa.gov. Refer to NPO-48736.

Risk-Constrained Dynamic Programming for Optimal Mars Entry, Descent, and Landing

NASA's Jet Propulsion Laboratory, Pasadena, California

A chance-constrained dynamic programming algorithm was developed that is capable of making optimal sequential decisions within a user-specified risk bound. This work handles stochastic uncertainties over multiple stages in the CEMAT (Combined EDL-Mobility Analyses Tool) framework. It was demonstrated by a simulation of Mars entry, descent, and landing (EDL) using real landscape data obtained from the Mars Reconnaissance Orbiter.

Although standard dynamic programming (DP) provides a general framework for optimal sequential decision-making under uncertainty, it typically achieves risk aversion by imposing an arbitrary penalty on failure states. Such a penalty-based approach cannot explicitly bound the probability of mission failure. A key idea behind the new approach is called risk allocation, which decomposes a joint chance constraint into a set of individual chance constraints and distributes risk over them. The joint chance constraint was reformulated into a constraint on an expectation over a sum of an indicator function, which can be incorporated into the cost function by dualizing the optimization problem. As a result, the chance-constraint optimization problem can be turned into an unconstrained optimization over a Lagrangian, which can be solved efficiently using a standard DP approach.

This work was done by Masahiro Ono and Yoshiaki Kuwata of Caltech for NASA's Jet Propulsion Laboratory. For more information, contact iaoffice@jpl.nasa.gov. NPO-48606

Scheduling Operations for Massive Heterogeneous Clusters

Goddard Space Flight Center, Greenbelt, Maryland

High-performance computing (HPC) programming has become increasingly difficult with the advent of hybrid supercomputers consisting of multicore CPUs and accelerator boards such as the GPU. Manual tuning of software to achieve high performance on this type of machine has been performed by programmers. This is needlessly difficult and prone to being invalidated by new hardware, new software, or changes in the underlying code.

A system was developed for task-based representation of programs, which when coupled with a scheduler and runtime system, allows for many benefits, including higher performance and utilization of computational resources, easier programming and porting, and adaptations of code during runtime.

The system consists of a method of representing computer algorithms as a series of data-dependent tasks. The series forms a graph, which can be scheduled for execution on many nodes of a supercomputer efficiently by a computer algorithm. The schedule is executed by a dispatch component, which is tailored to understand all of the hardware types that may be available within the system. The scheduler is informed by a cluster mapping tool, which generates a topology of available resources and their strengths and communication costs.

Software is decoupled from its hardware, which aids in porting to future architectures. A computer algorithm schedules all operations, which for systems of high complexity (i.e., most NASA codes), cannot be performed optimally by a human. The system aids in reducing repetitive code, such as communication code, and aids in the reduction of redundant code across projects.

It adds new features to code automatically, such as recovering from a lost node or the ability to modify the code while running.

In this project, the innovators at the time of this reporting intend to develop two distinct technologies that build upon each other and both of which serve as building blocks for more efficient HPC usage. First is the scheduling and dynamic execution framework, and the second is scalable linear algebra libraries that are built directly on the former.

This work was done by John Humphrey and Kyle Spagnoli of EM Photonics, Inc. for Goddard Space Flight Center. Further information is contained in a TSP (see page 1). GSC-16472-1

Deepak Condenser Model (DeCoM)

Goddard Space Flight Center, Greenbelt, Maryland

Development of the DeCoM comes from the requirement of analyzing the performance of a condenser. A component of a loop heat pipe (LHP), the condenser, is interfaced with the radiator in order to reject heat. DeCoM simulates the condenser, with certain input parameters. Systems Improved Numerical Differencing Analyzer (SINDA), a thermal analysis software, calculates the adjoining component temperatures, based on the DeCoM parameters and interface temperatures to the radiator. Application of DeCoM is (at the time of this reporting) restricted to small-scale analysis, without the need for in-depth LHP component integrations. To efficiently develop a model to simulate the LHP condenser, DeCoM was developed to meet this purpose with least complexity. DeCoM is a single-condenser, single-pass simulator for analyzing its behavior. The analysis is done based on the interactions between condenser fluid, the wall, and the interface between the wall and the radiator.

DeCoM is based on conservation of energy, two-phase equations, and flow equations. For two-phase, the Lockhart-Martinelli correlation has been used in order to calculate the convection value between fluid and wall. Software such as SINDA (for thermal analysis analysis) and Thermal Desktop (for modeling) are required. DeCoM also includes the ability to implement a condenser into a thermal model with the capability of understanding the code process and being edited to user-specific needs. DeCoM requires no license, and is an open-source code. Advantages to DeCoM include time dependency, reliability, and the ability for the user to view the code process and edit to their needs.

This work was done by Deepak Patel of Goddard Space Flight Center. Further information is contained in a TSP (see page 1). GSC-16296-1

Flight Software Math Library

Goddard Space Flight Center, Greenbelt, Maryland

The flight software (FSW) math library is a collection of reusable math components that provides typical math utilities required by spacecraft flight software. These utilities are intended to increase flight software quality reusability and maintainability by providing a set of consistent, well-documented, and tested math utilities. This library only has dependencies on ANSI C, so it is easily ported.

Prior to this library, each mission typically created its own math utilities using ideas/code from previous missions. Part of the reason for this is that math libraries can be written with different strategies in areas like error handling, parameters orders, naming conventions, etc. Changing the utilities for each mission introduces risks and costs. The obvious risks and costs are that the utilities must be coded and revalidated. The hidden risks and costs arise in miscommunication between engineers. These utilities must be understood by both the flight software engineers and other subsystem engineers (primarily guidance navigation and control).

The FSW math library is part of a larger goal to produce a library of reusable Guidance Navigation and Control (GN&C) FSW components. A GN&C FSW library cannot be created unless a standardized math basis is created. This library solves the standardization problem by defining a common feature set and establishing policies for the library's design. This allows the libraries to be maintained with the same strategy used in its initial development, which supports a library of reusable GN&C FSW components.

The FSW math library is written for an embedded software environment in C. This places restrictions on the language features that can be used by the library. Another advantage of the FSW math library is that it can be used in the FSW as well as other environments like the GN&C analyst's simulators. This helps communication between the teams because they can use the same utilities with the same feature set and syntax.

This work was done by David McComas of Goddard Space Flight Center. Further information is contained in a TSP (see page 1). GSC-16102-1