

Process Analysis Tradeoff Tool (PATT). Typical inputs to PATT models include industry-average values of product size (expressed as number of lines of code), productivity (number of lines of code per hour), and number of defects per source line of code. The user provides the number of resources, the overall percent of effort that should be allocated to each process step, and the number of desired staff members for each step. The output of PATT includes the size of the product, a measure of effort, a measure of rework effort, the duration of the entire process,

and the numbers of injected, detected, and corrected defects as well as a number of other interesting features.

In the development of the present model, steps were added to the IEEE 12207 waterfall process, and this model and its implementing software were made to run repeatedly through the sequence of steps, each repetition representing an iteration in a spiral process. Because the IEEE 12207 model is founded on a waterfall paradigm, it enables direct comparison of spiral and waterfall processes. The model can be

used throughout a software-development project to analyze the project as more information becomes available. For instance, data from early iterations can be used as inputs to the model, and the model can be used to estimate the time and cost of carrying the project to completion.

*This work was done by Carolyn Mizell of Kennedy Space Center, Charles Curley of ASRC Aerospace Corp., and Umanath Nayak of Portland State University. Further information is contained in a TSP (see page 1). KSC-13094*

---

## ➤ Algorithm That Synthesizes Other Algorithms for Hashing

**A synthesized algorithm is guaranteed to be executable in constant time.**

*NASA's Jet Propulsion Laboratory, Pasadena, California*

An algorithm that includes a collection of several subalgorithms has been devised as a means of synthesizing still other algorithms (which could include computer code) that utilize hashing to determine whether an element (typically, a number or other datum) is a member of a set (typically, a list of numbers). Each subalgorithm synthesizes an algorithm (e.g., a block of code) that maps a static set of key hashes to a somewhat linear monotonically increasing sequence of integers. The goal in formulating this mapping is to cause the length of the sequence thus generated to be as close as practicable to the original length of the set and thus to minimize gaps between the elements.

The advantage of the approach embodied in this algorithm is that it completely avoids the traditional approach

of hash-key look-ups that involve either secondary hash generation and look-up or further searching of a hash table for a desired key in the event of collisions.

This algorithm guarantees that it will never be necessary to perform a search or to generate a secondary key in order to determine whether an element is a member of a set. This algorithm further guarantees that any algorithm that it synthesizes can be executed in constant time. To enforce these guarantees, the subalgorithms are formulated to employ a set of techniques, each of which works very effectively covering a certain class of hash-key values. These subalgorithms are of two types, summarized as follows:

- Given a list of numbers, try to find one or more solutions in which, if each number is shifted to the right by a con-

stant number of bits and then masked with a rotating mask that isolates a set of bits, a unique number is thereby generated. In a variant of the foregoing procedure, omit the masking. Try various combinations of shifting, masking, and/or offsets until the solutions are found. From the set of solutions, select the one that provides the greatest compression for the representation and is executable in the minimum amount of time.

- Given a list of numbers, try to find one or more solutions in which, if each number is compressed by use of the modulo function by some value, then a unique value is generated.

*This work was done by Mark James for Caltech for NASA's Jet Propulsion Laboratory. Further information is contained in a TSP (see page 1). NPO-45175*

---

## ➤ Algorithms for High-Speed Noninvasive Eye-Tracking System

**One of the algorithms enables tracking at a frame rate of several kilohertz.**

*NASA's Jet Propulsion Laboratory, Pasadena, California*

Two image-data-processing algorithms are essential to the successful operation of a system of electronic hardware and software that noninvasively tracks the direction of a person's gaze in real time. The system was described in "High-Speed Noninvasive Eye-Tracking System" (NPO-30700) *NASA Tech Briefs*, Vol. 31, No. 8 (August 2007), page 51.

To recapitulate from the cited article: Like prior commercial noninvasive eye-

tracking systems, this system is based on (1) illumination of an eye by a low-power infrared light-emitting diode (LED); (2) acquisition of video images of the pupil, iris, and cornea in the reflected infrared light; (3) digitization of the images; and (4) processing the digital image data to determine the direction of gaze from the centroids of the pupil and cornea in the images. Most of the prior commercial noninvasive eye-

tracking systems rely on standard video cameras, which operate at frame rates of about 30 Hz. Such systems are limited to slow, full-frame operation.

The video camera in the present system includes a charge-coupled-device (CCD) image detector plus electronic circuitry capable of implementing an advanced control scheme that effects readout from a small region of interest (ROI), or subwindow, of the full image. Inas-