FEATURE ARTICLES

12 Guest Editor's Introduction: Domesticating Parallelism
   David Gelernter
   Parallel computers are becoming increasingly widespread. The question now is what to do with them. Can programming-language researchers make the power of parallelism accessible to programmers?

20 Parallel Processing in Ada
   David A. Mundle and David A. Fisher
   Ada's tasking mechanism represents a bold attempt to allow large, complex parallel processing applications to be written in a portable high-level language.

26 Linda and Friends
   Sudhir Ahuja, Nicholas Carriero, and David Gelernter
   Linda consists of a few simple primitives that support an "uncoupled" style of parallel programming. Implementations exist on a broad spectrum of parallel machines.

35 Parallel Symbolic Computing
   Robert H. Halstead, Jr.
   Futures find parallelism in symbolic programs by allowing the manipulation of partially computed data.

44 Concurrent Prolog: A Progress Report
   Ehud Shapiro
   A process-oriented language, Concurrent Prolog embodies dataflow synchronization and guarded-command indeterminacy as its basic control mechanisms.

60 Para-Functional Programming
   Paul Hudak
   This methodology treats a multiprocessor as a single autonomous computer onto which a program is mapped, rather than as a group of independent processors.

72 A Survey of Advanced Microprocessor and HLL Computer Architectures
   A. Silbey, V. Milutinovic, and V. Mendoza-Grado
   This survey classifies high-level language computer architectures and gives case studies of representative examples of each class of architecture.
Update: Zemanek receives Computer Pioneer medal at NCC; Teradata wins AFIPS Product of the Year award; Fiber optic chip runs at 400M bps

New Products
Microsystem Announcements
IC Announcements
New Literature
Call for Papers
Calendar
Career Opportunities
Advertiser/Product Index
Reader Service Card
Change-of-Address Form

On the cover
Domesticating parallelism. Illustration by Jay Simpson.

In the next issue
How super are supercomputers? See p. 96.