HPF: Language, Compilers and Environments

Ken Kennedy
Center for Research on Parallel Computation
Rice University

1. HPF Status Report
2. Problems with HPF 1
   performance, missing features, tools
3. Responses
   HPF enhancement activities
   Language, compiler, and tool research
4. Future Directions
Status of Scalable Parallelism

• Dream
  - virtually limitless computing power at low cost
  - performance scalable from one to thousands of processors
  - easy portable programming

• Reality
  - successful at only moderate levels of scalability
  - modest progress in programmability and scalability
  - limited penetration in industry
    independent software vendors (ISVs) still reluctant
    limited protection of programming investment

• Remedy: Architecture-Independent Programming
  - a programming language and its compilers support
    architecture-independent parallel programming if
    compiled code ≅ hand code for same algorithm
    for each target architecture
HPF Goals

• Support for Scalable Parallel Systems
  - scaling from one to thousands of processors

• Focus on Data Parallelism
  - parallelism through subdivision of data domain

• Machine Independent Programming Support
  - object program achieves performance comparable to hand-coded MPI on each target machine on the same algorithm

• High Level of Abstraction
  - more accessible programming model
    - single thread of control
    - shared memory
    - implicit generation of communication
HPF Strategy

Fortran 90
Sequential Machine

Data Distribution Directives
Free

CM-2
IBM SP-2

HPF
HP/Convex SPP2000
HPF Language

- Fortran 90 + Data Distribution
  - !HPF$ TEMPLATE D(256,256)
  - !HPF$ ALIGN A(I,J) WITH D(I+1,J)
  - !HPF$ DISTRIBUTE D(BLOCK,CYCLIC)
  - virtual processor array
  - align data elements
  - map to processors

- Explicit and Implicit Parallelism
  - Fortran 90 array statements, DO loops
  - "owner computes"

- Extended looping and aggregate assignment
  - FORALL (single-statement and block)
  - DO INDEPENDENT

- HPF Library
  - scatter, gather, reduction, segmented scan

- Extrinsic Interface
  - permits calls to message passing, other languages
(Block, Cyclic) Distribution

Processors

Template Elements
REAL A(1023,1023), B(1023,1023), APRIME(511,511)

!HPF$ TEMPLATE T(1024,1024)
!HPF$ ALIGN A(I,J) WITH T(I,J)
!HPF$ ALIGN B(I,J) WITH T(I,J)
!HPF$ ALIGN APRIME(I,J) WITH T(2*I-1,2*J-1)
!HPF$ DISTRIBUTE T(BLOCK,BLOCK)

!HPF$ INDEPENDENT, NEW(I)
DO J = 2, 1022  ! Multigrid Smoothing Pass (red-black relaxation)
!HPF$   INDEPENDENT
DO I = MOD(J,2), 1022, 2
   A(I,J) = 0.25*(A(I+1,J) + A(I+1,J) + A(I,J-1) + A(I,J+1)) + B(I,J)
END DO
END DO

!HPF$ INDEPENDENT, NEW(I)
DO J = 2, 510  ! Multigrid Restriction
!HPF$   INDEPENDENT
DO I = 2, 510
   APRIME(I,J) = 0.05*(A(2*I-2,2*J-2) + 4*A(2*I-2,2*J-1) + &
   & A(2*I-2,2*J) + 4*A(2*I-1,2*J-2) + 4*A(2*I-1,2*J) + &
END DO
END DO

! Multigrid convergence test
ERR = MAXVAL( ABS(A(:, :) - B(:, :) ) )
• Bad News
  - Acceptance of language has been slow in coming
  MPI has achieved penetration much faster
  compilers complex and not yet mature
  performance was initially disappointing
  additional features needed

• Good News
  - Compilers now available for every HPC platform
  - Applications beginning to emerge
  - Promising benchmark results
HPF Commercial Interest

• Announced HPF Products
  - Applied Parallel Research
  - CDAC
  - Cray Research
  - Digital Equipment
  - Fujitsu
  - Hitachi
  - HP
  - IBM
  - Intel
  - Meiko
  - Motorola
  - NA Software
  - NEC
  - Pacific Sierra Research
  - Portland Group
  - Sun
  - Transtech

• Announced HPF Efforts
  - ACE
  - Lahey
  - NAG
  - nCUBE

• Interested
  - EPC
  - SGI
  - Tera
HPF Usage

- Installations
  - PGI reports over 125 site licenses
  - NCSA: PGI on SGI is migration platform for CM-5

- Benchmarks
  - NAS Benchmarks: PGI compiler within factor of 2 (or better) of MPI

- Real Applications
  - NOAA, Princeton: Modular Ocean Model (100K lines)
  - Amoco: Falcon Reservoir Model (20K lines)
  - MATRA BAe Dynamics: AEROLOG (10K lines)
  - Quetzal: EPIC Crash Model (125K lines)
  - NCSA, UIUC: RIEMANN TVD and ENO Flow Code (50K lines)
  - U New Mexico: DFT Density Function Theory Code (6K lines)
  - MIT Earth Atmospheric and Planetary Science: Eulerian Ocean Model (3K lines)
  - Delft Univ TNO: Wish3D CFD code (125K lines, in progress)
  - U Houston: N-body Simulation (4K lines)
  - UIUC, Civil Eng: Nonlinear Multigrid Code (1K lines)
  - CRS4: GeoComp Seismic Migration Code
Problems for HPF

• Compilers slow to mature
  - Fortran 90 features supported inconsistently
  - compilation for highest efficiency complex
  - initially, efficiency of object programs unsatisfactory
  - early users may become discouraged

• Library support lacking
  - no CMSSL equivalent

• Needed features are missing
  - support for irregular problems
  - task parallelism
  - high performance input/output

• Complex relationship between program and performance
  - explanatory and diagnostic tools are needed
HPF Enhancement Activities

• HPF 2 Language Standard
  - Core Language: emphasis on implementable performance
    redistribution and realignment moved
  - Approved Extensions: future features
    support for irregular distributions
    computation distribution via ON clause
    task parallelism
    new extrinsic modules and interoperability components

• Library Development
  - effort to build interfaces to scientific libraries
    linear algebra (Dongarra) and FFT (J ohnsson)
    definition effort underway, implementation funding sought

• HPF User Group
  - First meeting: February 24-26, 1997 in Albuquerque
Specific Language Changes

• Changes in HPF 2 from HPF 1.1
  - Explicit interfaces **required** if remapping.
  - Intrinsic, commutative reductions.
  - Pointers cannot be mapped or point to mapped objects.
  - Mapping of aggregate covers **eliminated** from the language.
  - DYNAMIC and related features (REALIGN, REDISTRIBUTE) moved to Approved Extensions.

• Approved Extensions
  - DYNAMIC and related features (REALIGN, REDISTRIBUTE).
  - Extrinsic: HPF_SERIAL, HPF_LOCAL.
  - Pointer mapping.
  - All other new features passed in second reading.

irregular support, computation partitioning with ON clause, task parallelism.
D System Research Responses

• Compiler Optimization Research
  - dHPF research compiler
    new optimizations, new machine targets (e.g., DSM)
  - collaboration with Portland Group

• New Features
  - support for irregular problems
    collaboration with Joel Saltz
  - scalable I/O
    part of SIO initiative

• D System Tools
  - support for construction of efficient HPF programs
    intelligent editor displaying compiler feedback
    display of run-time information from Pablo (Dan Reed)
Future Research Areas

• Support for Heterogeneous Parallelism
  - language extensions for specification of task/object parallelism
  - automatic load balancing

• Java Compilation
  - compilation for high-performance servers
    freedom from portable Java restrictions
  - interprocedural analysis framework
  - autoparallelization

• Compiler Support for Problem-Solving Environments
  - PSE Toolkit: front-end plus interprocedural analysis

• Petaflops Computing
  - emphasis on 10 M-way parallelism and latency management
Summary

• HPF compilers emerging
  - language available on nearly every parallel system

• HPF 1 has some problems
  - performance
  - missing features
    - irregular support, task parallelism, I/O
  - programming tools and libraries needed

• HPF 2 addresses language issues
  - focus on implementable language for performance
  - advanced features as Approved Extensions

• CRPC research is developing new HPF technologies
  - advanced compiler optimization
  - support for new language features
  - libraries and programming tools
  - technology transfer to compiler vendors and end users