High Performance Fortran (HPF): The new Standard for Data Parallel Programming

Dr. Thomas Brandes

May 1995

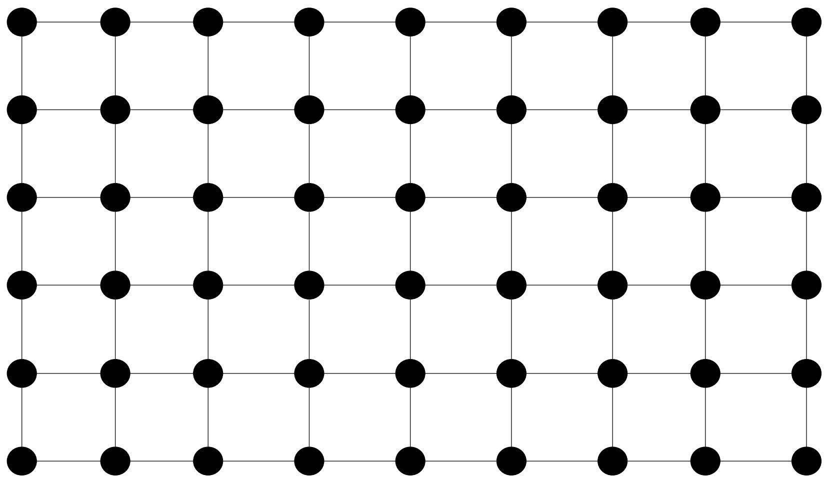
HPF : Data Parallel Programming

1. Data Parallel Programming
2. History of High Performance Fortran
3. Language Overview
4. Examples
5. HPF Compilers and Tools
6. Future of HPF

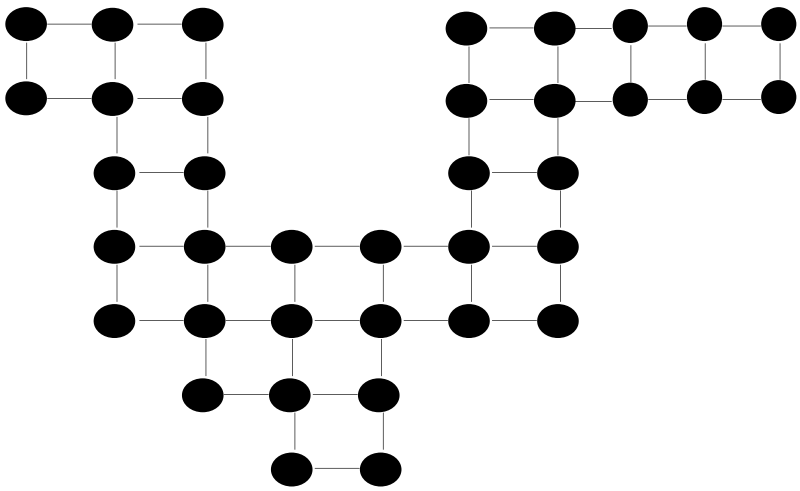
Data Parallelism

- single program, global name space
- using the inherent parallelism given by the operations on large arrays
- data decomposition by directives
- use of data locality
- implicit communication
- scalability
- suitable for many scientific applications
- suitable for all kind of architectures
- easy to use, easy to debug

Disadvantage: not everything is data parallel



(a) regular domain



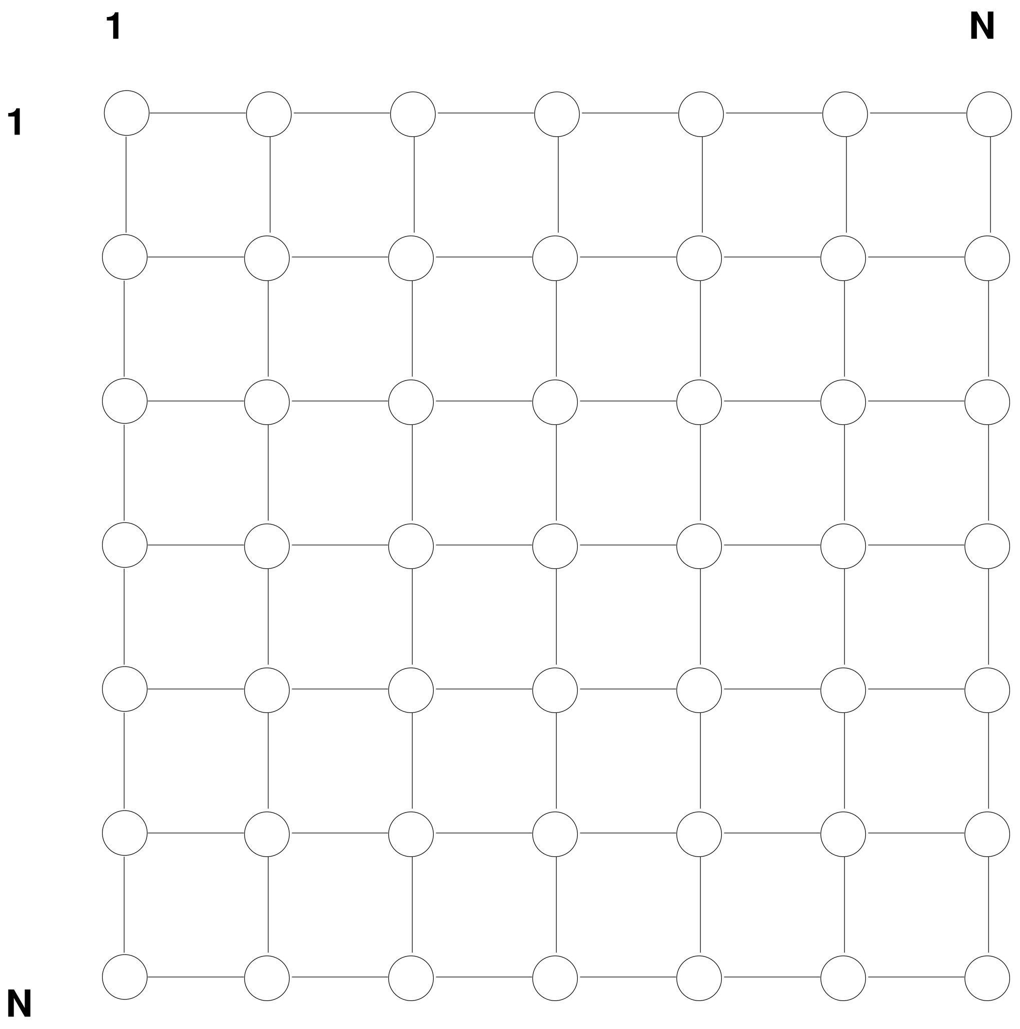
(b) irregular domain

Data Domain / Topologies / Templates

```
PARAMETER (N=100)
REAL u_old (N,N), v_old (N,N), h_old (N,N)
REAL u_new (N,N), v_new (N,N), h_new (N,N)
REAL u(N,N), v(N,N), h(N,N)
REAL f(N,N)
REAL dt, g, H

....

u_new (:,:) = u_old (:,:) + 2 * dt * f(:,:) * v(:,:) &
                        + 2 * dt * g * (EOSHIFT (h(:,:), 1, 1) - EOSHIFT (h(:,:), 1, -1)
v_new (:,:) = v_old (:,:) + 2 * dt * f(:,:) * u(:,:) &
                        + 2 * dt * g * (EOSHIFT (h(:,:), 2, 1) - EOSHIFT (h(:,:), 2, -1)
h_new (:,:) = h_old (:,:) + 2 * dt * H * (EOSHIFT (u(:,:),1,1) - EOSHIFT (u(:,:),1,-1)) &
                        + 2 * dt * H * (EOSHIFT (v(:,:), 2, 1) - EOSHIFT (v(:,:), 2, -1))
...
```

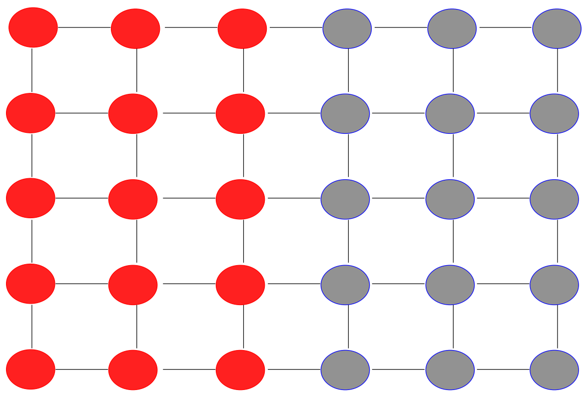
- serial in time
- parallel in space

Data Parallel Code




```
REAL u (N,N)
REAL v(N,N)          parallel data
REAL h (N,N)
REAL f(N,N)

REAL dt, g           serial data
```

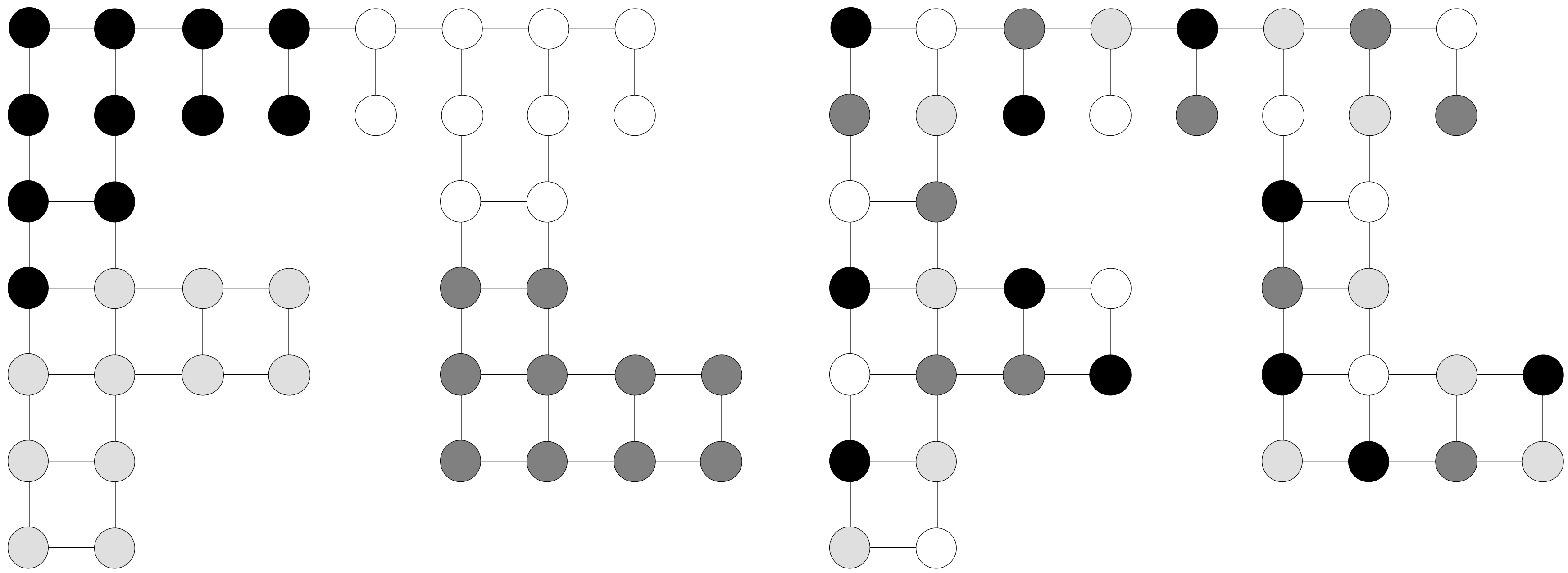
Data Domain (regular)



- for every update some other data is needed
- if this data is on the same processor, the update is very fast
- data layout should be done in such a way that communication overhead is minimal

Note: nearly every architecture has non-uniform memory access costs

Machine	Node Perf.	Start-Up Comm.
Intel Paragon	100 MFlops	20 us
Meiko CS-2	200 MFlops	10 us
CM-5	128 MFlops	10 us
SP 2	256 MFlops	10 us



Good Data Layout **Bad Data Layout**

Processor 1

Processor 2

Processor 3

Processor 4

<u>Related data should be mapped to the same processor</u>

Example: Data Locality

SIMD Machines	SIMD Languages	Year
Illiac IV	Glypnir, CFD	~ 1970
ICL DAP, AMT DAP	DAP Fortran	~ 1980
Connection Machine CM-1, CM-2, CM-5	*LISP, C*, CM Fortran	~ 1987 ~ 1988
MasPar	MasPar Fortran, MPL	~ 1990
MIMD Machines	High Performance Fortran (HPF)	~ 1992

History of Data Parallel Languages

High Performance Fortran : Main Goals

- Support of the data parallel programming model

    * global name space

    * single threaded

    * loosely synchronous

- Good / Top performance on MIMD and SIMD architectures with non-uniform memory access costs

- Code tuning for various architectures

2. History of HPF

HPF : Other Goals

- minimal deviation from other standards
- minimal direct conflicts with Fortran 77 / 90
- maximal simplicity
- open for other languages and programming styles
- input for future standards (especially C)
- produce validation criteria
- final proposal in November 1992
- final document in January 1993 (was May)
- development of an HPF test suite

HPF : Other Goals

HPF Forum was convened by G. Fox (Syracuse) and K.Kennedy (Rice)

Represented Organizations:

Alliant Computer Systems Cor.	MasPar Computer Corporation
Amoco Production Company	Meiko, Inc.
Applied Parallel Research	nCUBE, Inc.
Archipel	Ohio State University
CONVEX Computer Corporation	Oregon Graduate Inst. of Science/T.
Cornell Theory Center	Portland Group, Inc.
Cray Research, Inc.	Research Inst. for Advanced C.S.
DEC MPS Group	Rice University
Fujitsu America	Schlumberger
Fujitsu Laboratories	Shell
GMD, I1.T	State Univ. of N.Y. at Buffalo
Hewlett Packard	SunPro, Sun Microsystems
IBM	Syracuse University
ICASE	TNO-TU Delft
Intel Supercomper S. Div.	Thinking Machines Corporation
Lahey Computer	Unitied Technologies Corporation
Lawrence Livermore Nat. Lab.	University of Stuttgart
Los Alamos National Lab.	University of Vienna
Louisiana State University	Yale University

High Performance Fortran: Language Specification

Nov, 6th, 1992	Version 0.4	-> Supercomputer '92
Jan, 25th, 1993	Version 1.0 Draft	
May 23th, 1993	Version 1.0 Final	

High Performance Fortran: Journal of Development

- > becomes own document

How to get it:

by anonymous ftp:

titan.cs.rice.edu : public/HPFF
ftp.gmd.de : hpf-europe

by netlib

mail softlib@cs.rice.edu
send hpf-v10.ps

mail netlib@ornl.gov
send hpf-v10.ps from hpff

published in:

Scientific Programming 2 (1-2), pp. 1 - 170

HPF Handbook:

Ch. Koelbel, D. Loveman, R. Schreiber, G. Steele, M. Zosel:
The High Performance Fortran Handbook
The MIT Press, Cambridge (MA)

HPF in World Wide Web

http://www.erc.msstate.edu/hpff/home.html

HPF : Language Definition

New Features in High Performance Fortran

- Directives for Data Mapping

  (ALIGN, DISTRIBUTE)

- Expressing Data Parallelism

  (FORALL, INDEPENDENT)

- Intrinsics and Library Procedures

  (PROCESSORS, ...)

. Extrinsic Procedures

  (Interface for other programming models / languages)

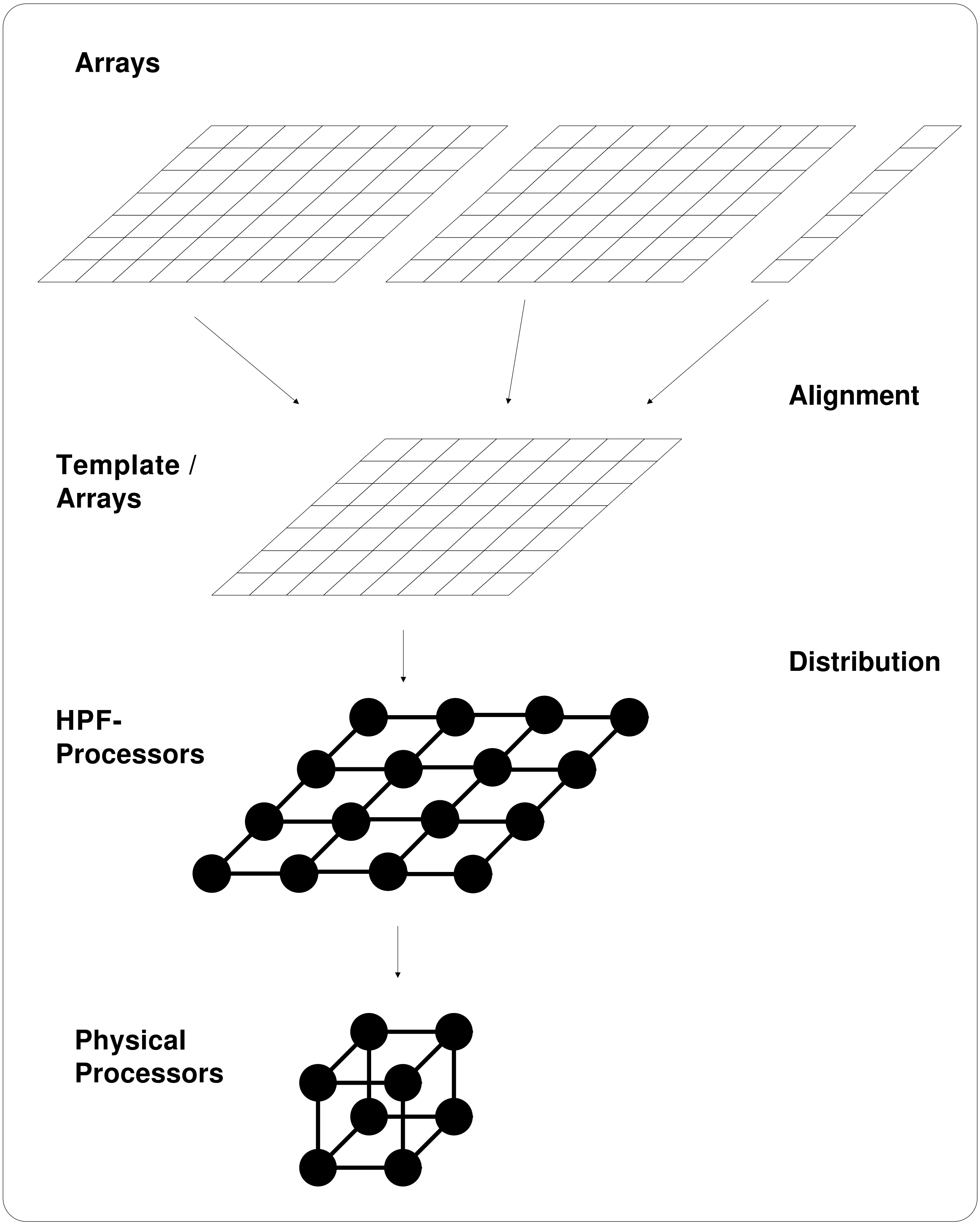
- Storage / Sequence Association

  (EQUIVALENCE, Reshaping)

- Subset High Performance Fortran

Considered: Parallel I/O

(no standardization)



Data Mapping : Overview

How to specify data parallelism ?

a) Array Operations

```
A (2:N-1) = ( A(1:N-2) + A(3:N) ) * 0.5

A = ( CSHIFT (A, DIM=1, POS=1) + CSHIFT (A, 1, -1) ) * 0.5

B(1:N,1:N) = SPREAD (B(J, 1:N) , DIM=1, NCOPIES = N)
```

b) FORALL statement

- new in HPF
- included in Fortran 95

```
FORALL (I=2 : N-1)
  A (I) = ( A(I-1) + A(I+1) ) * 0.5
END FORALL

FORALL (I=1:N, K=1:N)
  B (I, K) = B (J, K)
END FORALL
```

c) INDEPENDENT directive

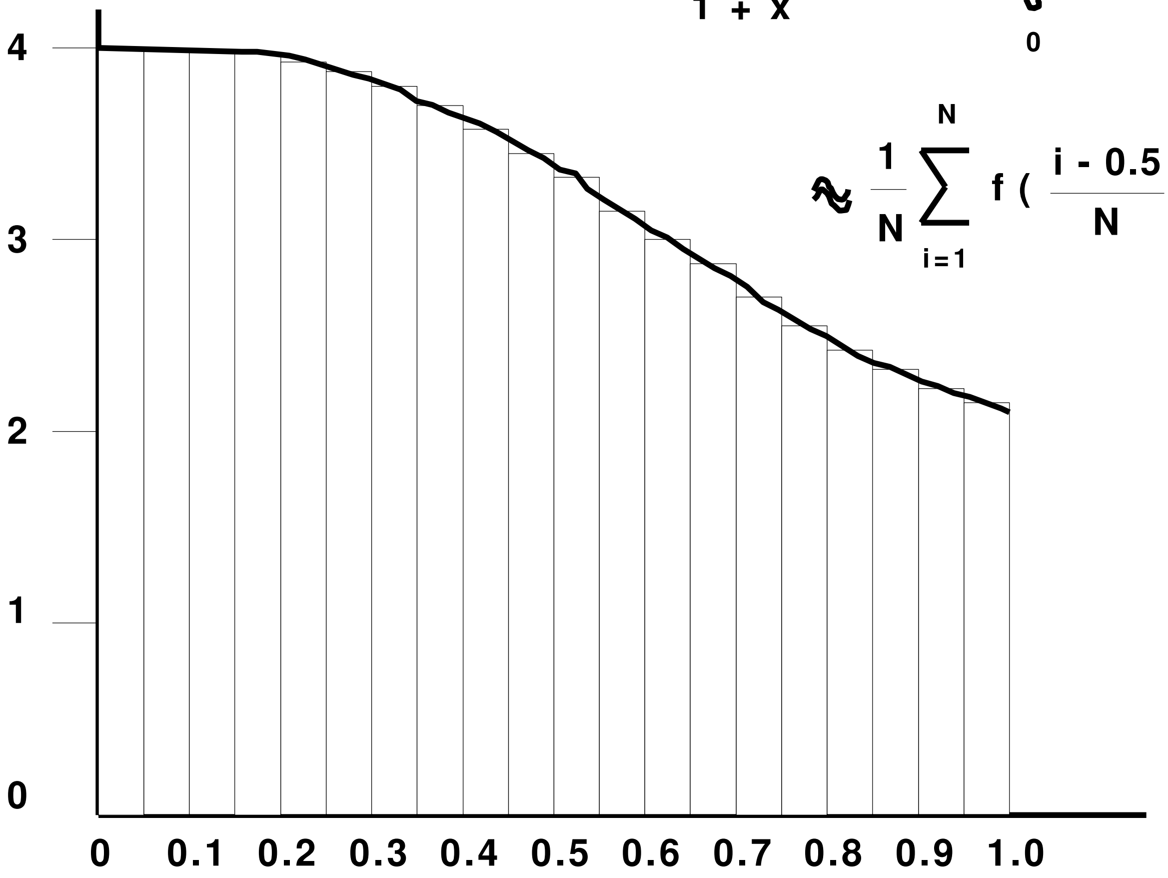
```
!HPF$ INDEPENDENT
    DO i = 1, N
       Y( K(i) ) = X (i) + C ( K(i) ) * Y ( K(i) )
    END DO
```

Data Parallelism : Overview

Computation of PI

$$f(x) = \frac{4}{1 + x^2} \qquad Pi = \int_0^1 f(x)\, d(x)$$

$$\approx \frac{1}{N} \sum_{i=1}^{N} f\left(\frac{i - 0.5}{N}\right)$$



Pseudo-Code:

Input N for the number of stripes
Compute the width of the stripes
For every stripe
   determine the hight of the stripe by function value
   sum this value up
multiply the sum with the width
Output of PI

Example: Computation of PI

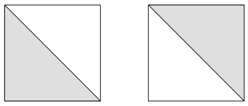
Fortran 90 - Code

```
program  picalc
integer n, i
double precision w, gsum
double precision, allocatable :: v(:)
print *, 'Input number of stripes : '
read *, n
w = 1.0 / n
allocate (v(1:n))
v = ([1..n] - 0.5) * w
v = 4.0 / (1.0 + v*v)
gsum = sum(v)
print *, 'Pi is approximately : ', gsum*w
deallocate (v)
end
```

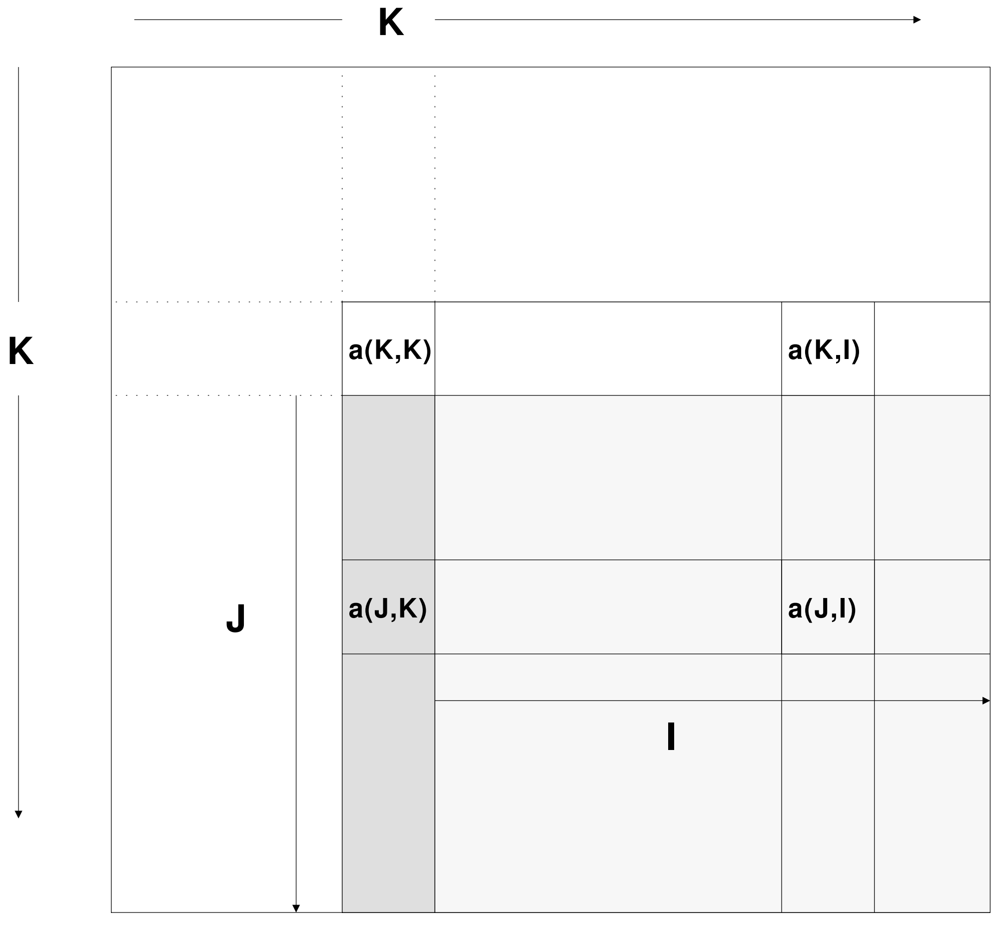
Computation of PI : HPF Code

LU Decomposition

$A = L * U$



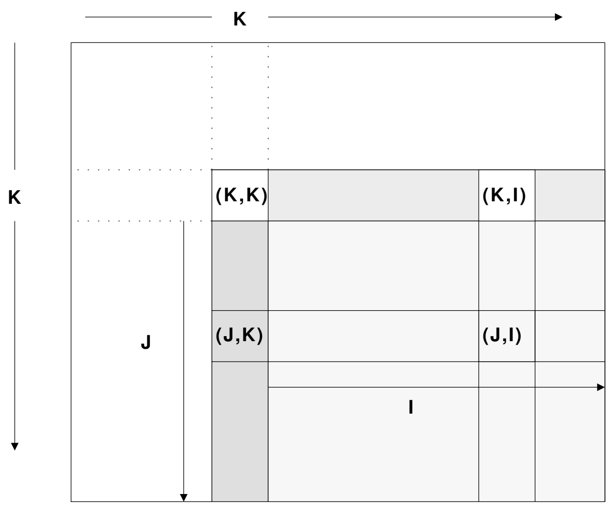
```
real A(N,N)

do K = 1, N-1
   do J = K+1, N
      A(J,K) = A(J,K) / A(K,K)
      do I=K+1, N
         A(J,I) = A(J,I) - A(J,K) * A(K,I)
      end do
   end do
end do
```

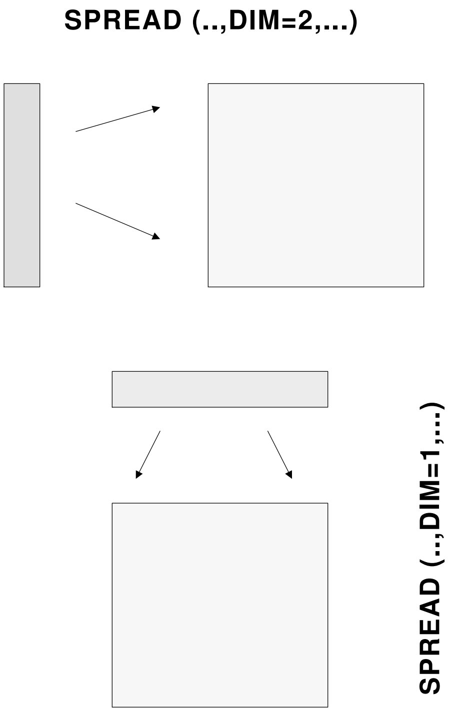


Example 2 : LU Decomposition

```
do K = 1, N-1
   forall (J= K+1 : N)  A(J,K) = A(J,K) / A(K,K)
   forall (J = K+1 : N, I = K+1 : N)
        A(J,I) = A(J,I) - A(J,K) * A(K,I)
   end forall
end do
```

```
do K = 1, N-1
   A(K+1:N,K) = A(K+1:N,K) / A(K,K)
   A(K+1:N, K+1:N) = A(K+1:N,K+1:N) -
              SPREAD ( A(K+1:N,K), DIM=2, NCOPIES = N-K ) *
              SPREAD ( A(K,K+1:N), DIM=1, NCOPIES = N-K )
end do
```





LU Decomposition: Fortran 90 Code

Announced Products

- Applied Parallel Research (APR)
- Digital Equipment Corporation (DEC)
- Portland Group, Intel (PGI)
- NA Software
- Pacific Sierra Research
- Kuck and Associates

- SofTech
- Meiko (PGI)
- Hitachi
- Motorola
- Intel (PGI)
- IBM (APR, PGI)
- NEC

Formally Announced Efforts

- ACE
- Convex
- Cray Computer
- Fujitsu
- Lahey
- NAG
- nCUBE
- MasPar
- Thinking Machines Corporation

Interested:

- ACSET
- HP
- Silicon Graphics
- SUN

HPF Implementations (State: April 95)

Applied Parallel Research

http://www.informall.org.apri

* FORGE Explorer

* Explorer / DMP

* "Magic" (automatic parallelization)

* xHPF

  - implements HPF subset

  - distribution along one dimension

Available for:

- a lot of different machines

The Portland Group, Inc. (PGI)

http://www.pgroup.com

* High Performance Fortran Compiler
* C, F77, C++ Compiler
* Software Development Tools

HPF compiler available on:

- Intel Paragon
- Meiko CS-2
- SGI Power Challenge
- Solaris Sparc
- Cray T3D
- IBM SP 2
- and others

Supported communication protocols:

- PVM
- MPI
- PARMACS
- Intel NX
- Meiko ELAN
- IRIX Shared Memory
- Solaris Shared Memory

PGI HPF compiler implements HPF subset !!

NA Software Fortran 90 Toolset

Contact: Prof. Mike Delves (Liverpool)

* Fortran 90 Plus Compiler and Debugger
    - available for PCs and SPARC

* HPF Mapper and Debugger
    - generates Fortran 90 with message passing

* LOFT90 Fortran 77 - Fortran 90 Translator

* Fortran Encyclopadia

HPF Codes / Benchmarks

* Parkbench Suite

  http://www.epm.ornl.gov/~walker/parkbench

* NCPA (Northeast Parallel Architectures Center at Syracuse University)

  http://www.npac.syr.edu/hpfa

* NAS Benchmarks

  http://www.nas.nasa.gov/NAS/NPB

* APR Benchmarks

  ftp.informall.org/tenants/apri/Bench

* Portlands Example Codes

  ftp.pgroup.com/pub/HPF/examples

* GMD

  ftp.gmnd.de/gmd/adaptor/hpf_examples.tar.Z

Research Tools /Compilers

Fortran 90 D compiler (Syracuse, New York)

Fortran 77 D compiler (Rice, Texas)

Vienna Fortran compilation system (Uni Vienna)

CAPTools (University of Greenwich)

HPFC, PIPS (Ecole de Mines, France)

PTRAN II (IBM)

Systems Software for Parallel Programming (ICASE)

PARADIGM (Center for Supercomputer Research & Development)

Environment for Portable Distributed Memory PP (CSCS)

xxx (University of Maryland)

Experimental Research (PD software):

ADAPTOR (GMD, Germany)

<u>1994</u>

- encourage 'industrial strength' implementations of HPF by vendors

- clarifications, corretions and interpretations of HPF 1.0 (CCI).

- identify and flesh out requirements for HPF 2.0

  - parallel I/O
  - irregular grids
  - additional tasking

- produce HPF 1.x document

Meetings: April 6-8, June 1-3, Aug 17-19, Oct 12-14

mail hpff-request@ds.rice.edu
add hpf

<u>1995</u>

- general meeting in early 1995

- development of HPF 2.0

Future Plans of the HPF Forum

Enhanced Mapping

- irregular mapping of arrays
- mapping of linked data structures
- mapping of derived type components

Computation Control and Task Parallelism

- computation mapping
- reductions and atomic operations
- DOACROSS loops
- Mulitiple HPF Process Model
- Parallel Sections

Input-Output

- Transparent parallel accesses to files
- Striping / distributing files over disks

Communication Optimizations

- reuse of communication patterns
- locality assertions

Language Processor Environment

- language interoperability
- tool support

HPF-2: New Capabilities