Workstations/PC

RAMS Computer History

- 1975-1978: NCAR CDC 7600 - punch cards
- 1978-1982: NCAR Cray 1 - punch cards - 8 Mb memory
- 1982-1985: NCAR Cray X-MP - interactive - 32 Mb memory
- 1985-1988: NCAR Cray Y-MP - 64 Mb memory
- 1986: Development moved to CSU MicroVAX II
- 1988: CDC CYBER 205
- 1988-1991: Ardent/Stardent Titan
- 1988-1995: IBM RS6000
- 1991: Parallel development begins
- 1995+: All other UNIX workstations
- 1996: First operational parallel version installed
- 1997+: Many other parallel machines
- 1998: Pentium clusters

How did we do it?

- 1980-1982
  - Printouts, printouts and more printouts
  - Boxes of cards
  - Keypunch machines - high school revisited
  - Punch cards, submit job, wait for results, stare at printouts, punch cards again
  - Drop cards, swear a lot!
  - At best, 4-6 jobs per day

How did we do it?

- 1983-1985
    - Interactive access to front-end machine
    - No more cards!
    - Better productivity, but… still batch submission
    - At best, 6-8 jobs per day

Local processing

- 1986
    - First local machine - MicroVAX II
    - Slow, but dramatic increase in efficiency
    - Development, but not runs
- 1988
    - First RISC workstations
    - At last could do smaller runs locally

Local processing

- 1990’s
  - Workstation use expands
  - Students who had to use supercomputers complain
  - Most runs done locally
  - Parallel development begins and matures

RAMS Parallel Design Considerations

- Concentrating on larger problems (only 3-dimensional)
- No compromises in physics and numerics
- Target distributed memory architectures
    - workstation cluster first platform
- Single code version for single processor and parallel platforms
- No performance degradation on single processor platforms
- Dynamic load balancing needed

RAMS Parallel Components and Structure

- Main structure
- Memory structure
- Variable tables
- Domain decomposition
- Communication procedures
- Concurrent computation/communication
- Bookkeeping arrays
- File output
- Nested grid considerations
- Dynamic load balancing

RAMS Parallel Structure

For parallel execution, serial (single-processor) code divided into two types:

- Master process
  - Initialization
  - All input/output functions
- Node (“compute”) processes
  - All computation

RAMS Memory Structure

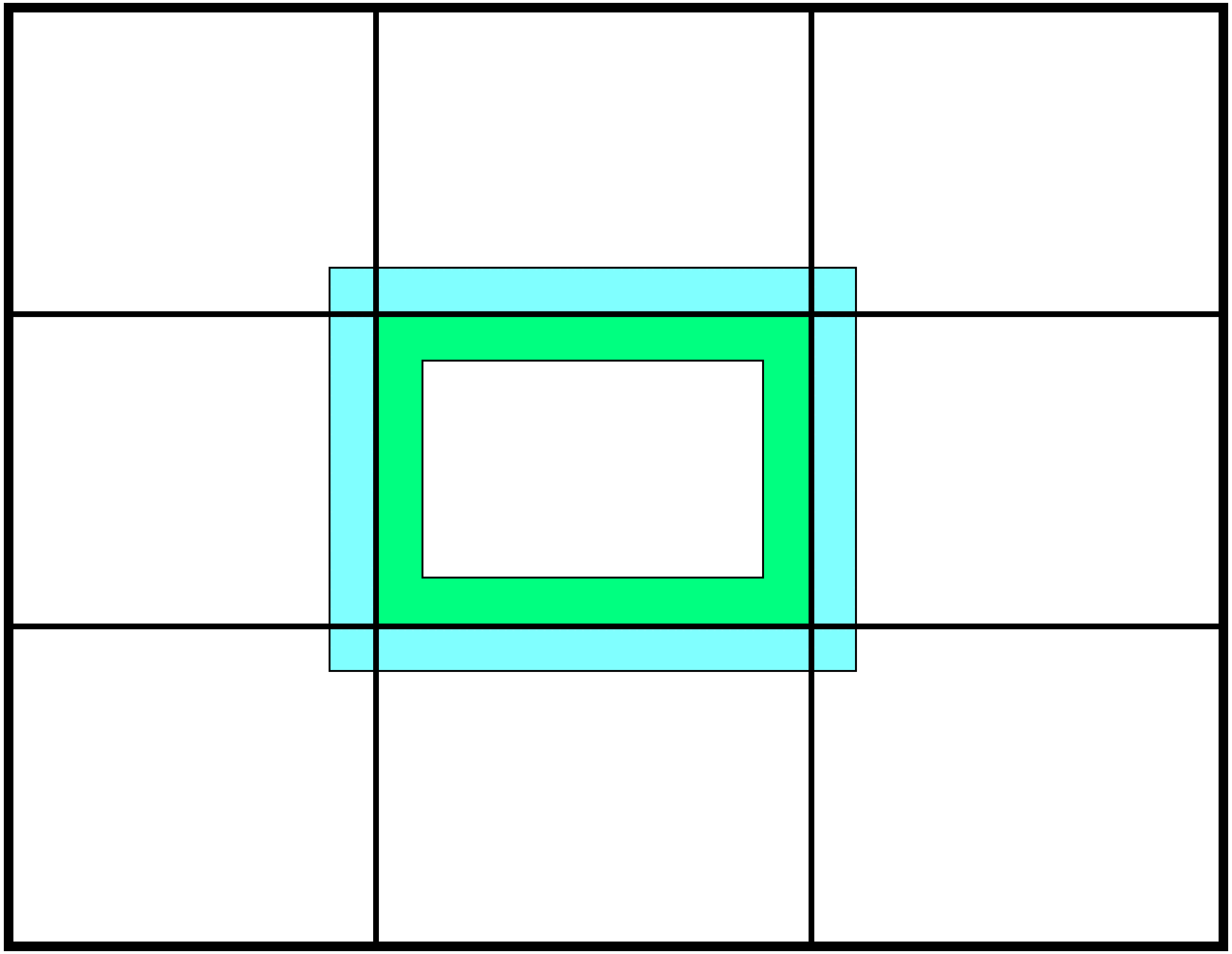
- Many options / array space / nests with variable numbers of grid points
- Dynamically-allocate “A” array
    - “1-D” array with all memory
    - C routine to *malloc*
    - “Pointers” to beginning of each field (integer indices in COMMON)
    - Computational routines dynamically-dimension arrays
    - Each node/master process allocates its own memory

RAMS Variable Tables

Data file defines characteristics of each model array

- Existence (whether on node or master processes)
- Output
- Communication

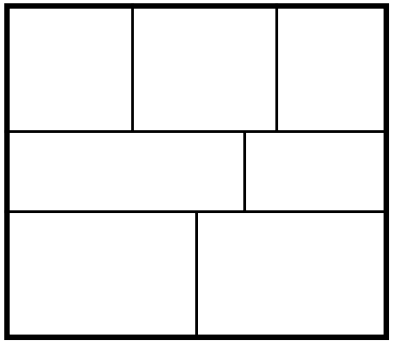
Domain Decomposition



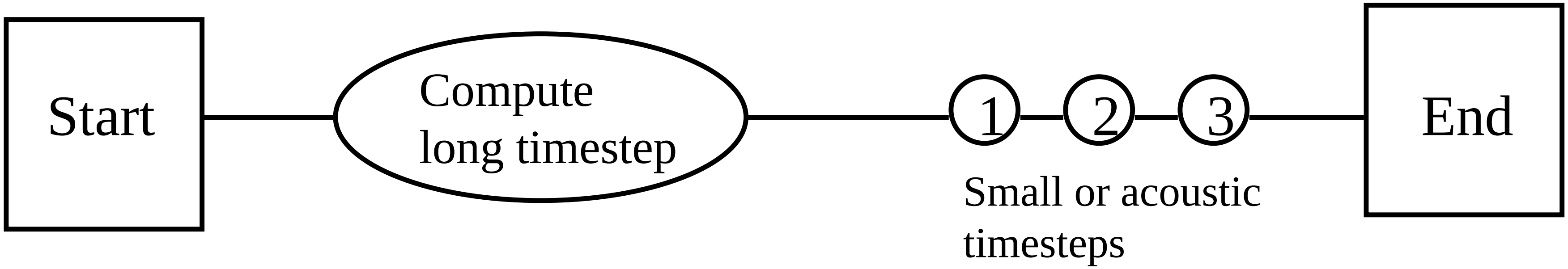
Nodes exchange "overlap" rows (halo regions) with adjacent nodes

Domain decomposition

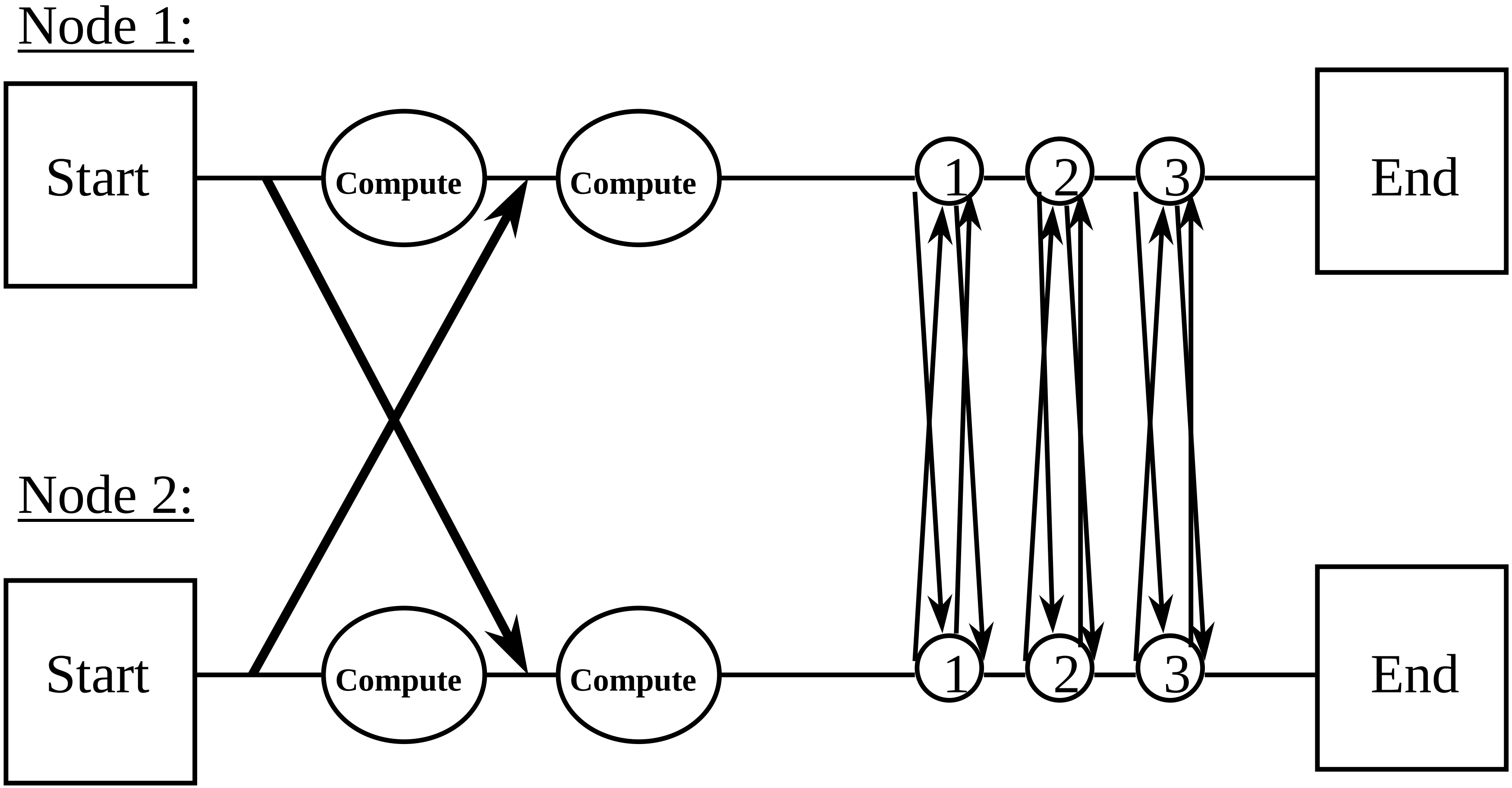
- Each column assigned a “work” factor
- Factor based on boundary, physics, machine performance, etc.
- Domain horizontally decomposed across processors
- “2-D” decomposition (3 or more nodes)



Timestep flow - single grid - serial



Timestep flow - single grid - parallel



Concurrent communication/computation

- Overlap rows sent first thing in timestep flow.
- Model processes that can be computed without overlap rows are done first.
- Sub-domain interior is computed for some terms
- Messages are received
- Sub-domain boundary terms are computed

Communication "types"

Initialization / re-decomposition

- Master to node; Full sub-domains; Most variables

Long timestep overlap region

- Node to node; 1 row; Prognostic variables

🚫 Turbulence overlap region

- Node to node; 1 row; Eddy diffusivities, strain rates

Small timestep overlap region (2 events)

- Node to node; 1 row; Selected variables

File output

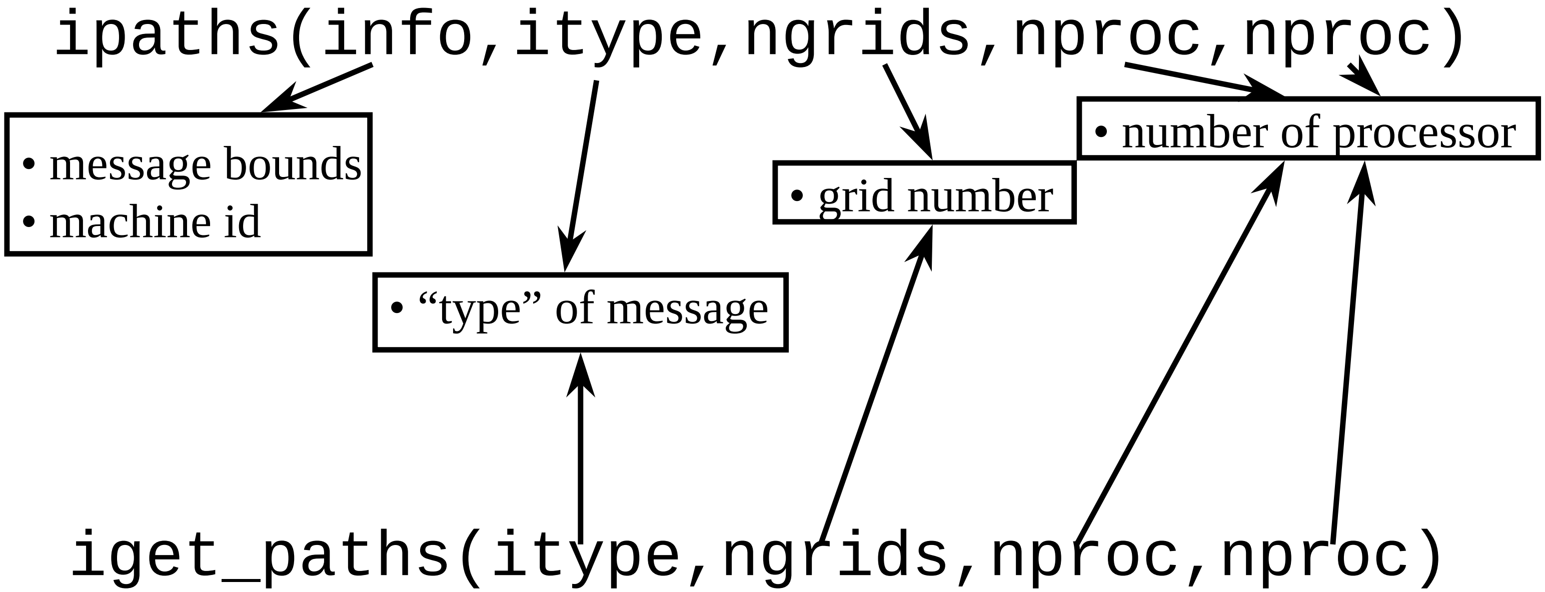
- Node to master; Full sub-domains; Prognostic variables

All node to node types package variables in a single message before sending.

Bookkeeping arrays

Keep track of communication “paths”

source node / destination node



```
ipaths(info,itype,ngrids,nproc,nproc)
```

- message bounds
- machine id

- “type” of message

- grid number

- number of processor

```
iget_paths(itype,ngrids,nproc,nproc)
```

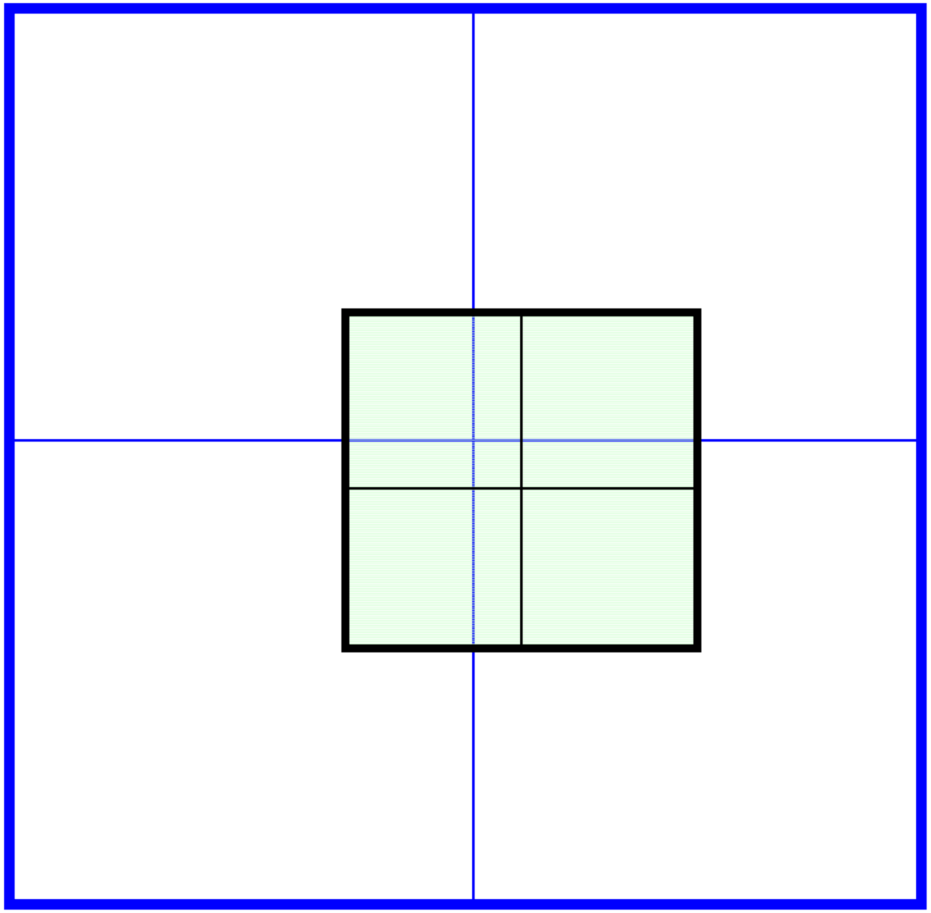
File output

- Nodes transfer necessary data to master process
- Transfer done with message passing
- No local disk space or NFS needed for file output
- Nodes can continue computing
- Data available on master for other activities
- Parallel I/O not needed

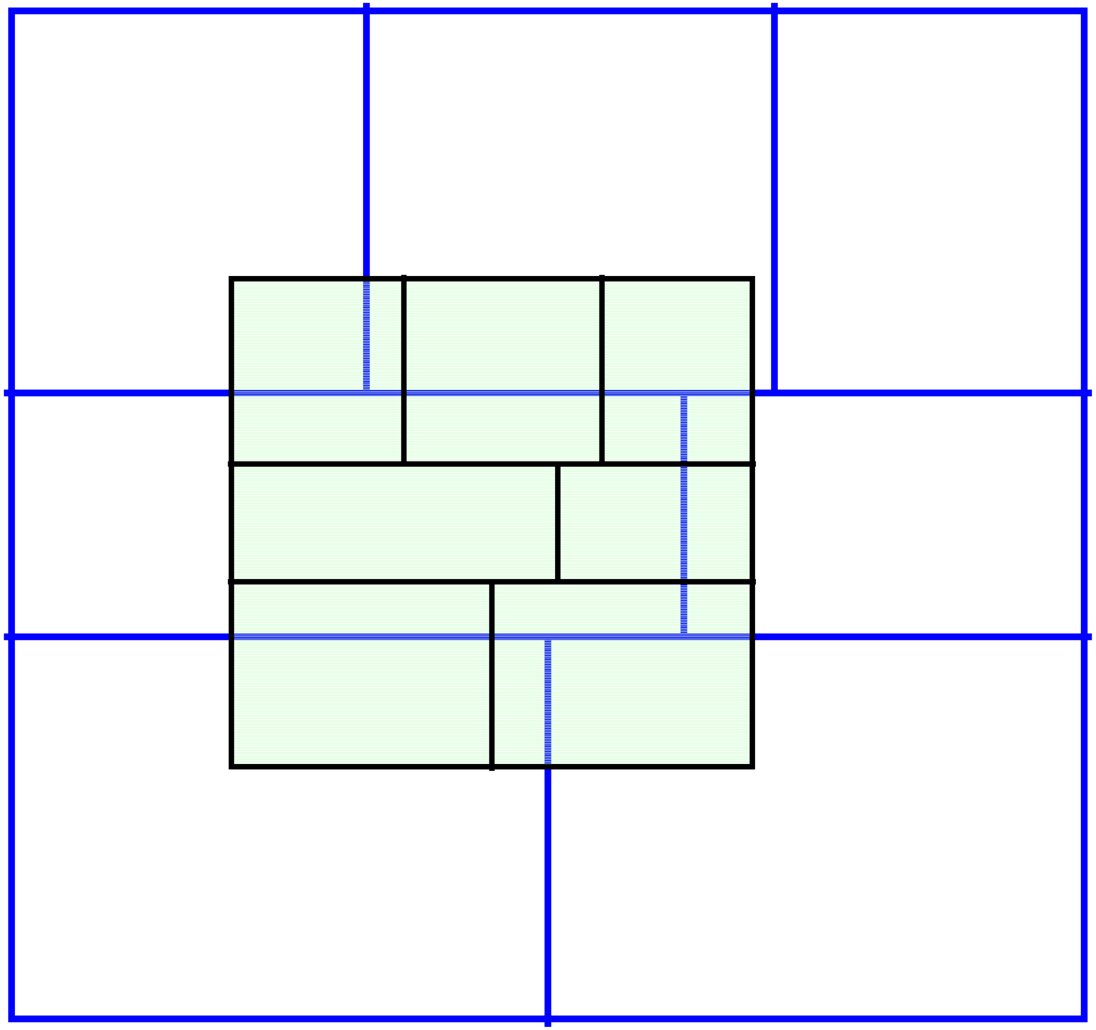
Nested grid considerations

- all grids decomposed independently
- 2-way interactive communication
- interpolation of nested grid boundaries performed on fine grid nodes, coarse grid information sent to fine grid
- feedback (averaging) of fine grid information performed on fine grid node, averaged values sent to coarse grid
- minimizes data needed to be communicated

Nested Grid Decomposition - 4 processors



Nested Grid Decomposition



Dynamic load balancing

- Regions of domain can require additional physics as model progresses (microphysics, radiation)

- Nodes send timing information back to master at end of timestep

- At opportune times (coinciding with file output times), master process will:
  - re-compute “work” factor
  - re-decomposes the domain
  - re-initializes the nodes

Performance

- Performance varies, dependent on architecture, communication hardware, model configuration
- Speedups and efficiencies
- Results
  - CSU IBM RS/6000 cluster, Ethernet
  - KSC IBM RS/6000 Power PC cluster, Ethernet
  - Maui IBM SP2
  - SMP machines from HP, SGI

Measures of parallel performance

Speed up:

$$\boldsymbol{S} = \frac{\text{Time to run serial code}}{\text{Time to run parallel code}}$$

Efficiency:

$$\boldsymbol{E} = \text{S} \,/\, \text{\# processors}$$

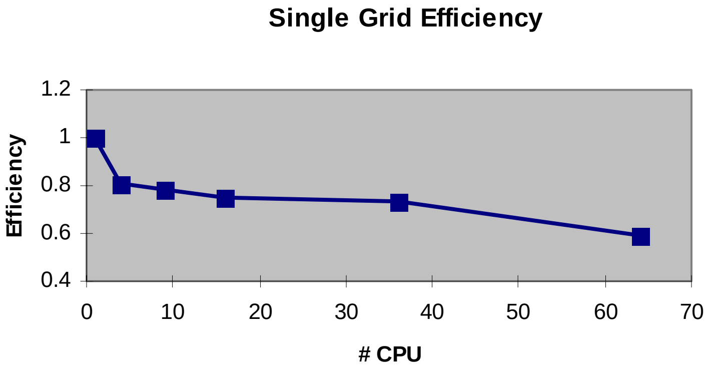
Results - IBM RS/6000 clusters

- CSU cluster: 1-6 IBM RS/6000 Model 550
  - Ethernet, PVM, old communication schemes
  - Single grid: 95-72% efficiency
  - Nested grid: 90-72% efficiency

- KSC cluster: 7 IBM RS/6000 Model 250 Power PC
  - Ethernet, MPICH, new schemes
  - Operational, 4 grids, small numbers of points (35x35)
  - 55-65% efficiency

Results - Maui IBM SP2

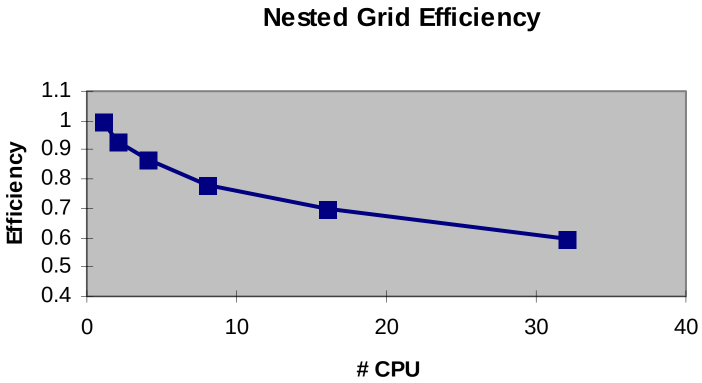
grid 1: 61x61x23 points



grid 1: 61x61x23 points

grid 2: 62x62x23

grid 3: 62x62x23



Results - SMP platforms

- Hewlett-Packard S-Class (Exemplar)
    - 8 processors 120 MHz PA-7200
    - Crossbar memory hardware
    - HP-modified MPI
    - Operational configuration, 3 grids: 105% efficiency

- Silicon Graphics Origin 200
    - 4 processors, 180 MHz R10000
    - Standard version MPICH
    - Operational configuration, 3 grids: 95% efficiency

RAMS and the PC

- 1990 - First port to 80386 20 MHz
- Because of “wonderful” experience, many years go by…
- 1998 - Elebra in Sao Paulo show good results

Today’s Supercomputer

- “Beowulf” cluster
- Stripped down compute nodes
- Larger memory master/graphics nodes
- Good network switch - no hubs!
- KVM switch
- 16 CPU cluster < $20,000

RAMS and the PC

- Our experiences have shown the viability for single processors and parallel clusters
- Efficiencies as good as some SMP machines
- Advantages in:
    - machine cost (purchase and maintenance)
    - software and peripheral cost
    - compiler and debug environments