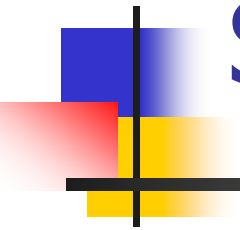
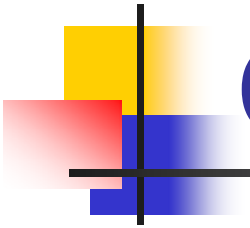


Scientific Applications

Jing Jiang, Jayanth Gummaraju,
Rohit Gupta



Outline

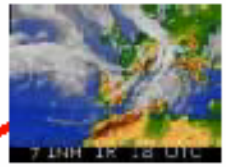
- Application Study
  - Vortex
- Architectural Issues
- Benchmarks

Applications

Government-Classified Work

Government - Research

OAK RIDGE NATIONAL LABORATORY



Instituto Nacional de Meteorología España

(Severe) Weather Prediction & Climate Modeling

Automotive Design & Safety

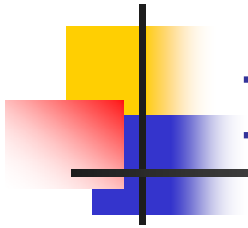
Drug Discovery & Genomic Research

Aircraft/Spacecraft Design & Fuel-Efficiency

Oil Exploration & Energy Research

Basic Scientific Research

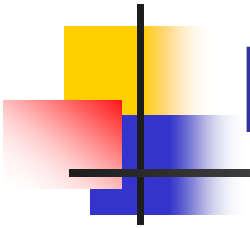
taken from Cray Research website: www.cray.com



Interesting Example: Weather Forecasting

- The atmosphere of a region is divided into 3-D cells
  - the weather conditions are computed in each cell
  - repeated for each time step
- Divide the area in grid cells of size 1x1x1 mile
  - total of 5×10^8 grid cells
  - 200 floating-point operations per cell and time step
  - 10^{11} operations for each time step
  - 10 day forecast, time step of 10 minutes = 1440 time steps
  - total of $1.4 * 10^{14}$ fp-operations
- With a computer doing 100 Mflops = 10^8 fp-ops/s this takes about $1.4 * 10^6$ seconds = 16 days

This slide has been taken from a course on Parallel Programming in Finland. It is very interesting because it shows that for predicting weather for the next day, using a 100MFLOPS machine would take about 16 days! Using the fastest supercomputer today, NEC's Earth Simulator, which has a peak performance of 40 TFLOPS, it would take about 10 hours!



Example Application: Vortex

- N-body Simulation

  The application models the evolution of vortices in in a 2-dimensional fluid.
- O (N^2) Interactions
  - All vortices have an effect on all other vortices.
  - At each stage, it calculates O (N^2) interactions among N vortices.
- Each Processor has N/P bodies
  - The load of calculation is evenly distributed among P processors. At each stage of simulation, results are kept in two copies, one of the which goes to the other processors.
  - They are actually shifted along a ring formed by all the processors.
  - Whenever a new vortex enters the processor, the interactions of it with other stationary vortices at that processor are calculated
- Binary Tree Reduction
  - After all of the stage simulations are done, the processors communicate in a binary tree with the root reporting the progress of the algorithm to the host computer.
- In this example:
  - 4096 bodies
  - 100 stages

Code Size

APPLICATION	LANG	SCL	MTYPE	PROCS
CLIMATE	Fortran	80K	Delta	256
SEMI	Fortran	50K	Delta	512
MOLECULE	Fortran	1K	nCUBE 2	512
RENDER	C	2K	Delta	32
EXFLOW	C & Fortran	12K	Delta	512
QCD	C	2K	nCUBE 1	256
VORTEX	Fortran	1K	nCUBE 2	64
REACT	C & Fortran	42K	Delta	512

Illustrated in the table are a wide range scientific applications running on 3 different machines. In the following slides, we will explore their behavior in terms of floating point operations, memory, I/0 and communication. They are:

CLIMATE: global climate simulation

SEMI: 3-D semiconductor device simulation

MOLECULE: molecular dynamics simulation

RENDER: 3-D perspective rendering

EXFLOW: 3-D flow using adaptive grids

Vortex: 2-D fluid dynamics

QCD: quantum chemical reaction dynamics

Memory Requirements

APPL	DATA	DATA/PROC	CODE/PROC	OS/PROC	% USED
CLIMATE	1750MB	7168KB	4096KB	4096KB	94
SEMI	1000MB	2048KB	NA	4096KB	NA
MOLECULE	1000MB	2048KB	200KB	200KB	60
RENDER	280MB	8960KB	260KB	4096KB	81
EXFLOW	732MB	1464KB	720KB	4096KB	38
QCD	17MB	70KB	98KB	100KB	52
VORTEX	0MB	3KB	492KB	200KB	17
REACT	536MB	1072KB	432KB	4096KB	34
AVE	665MB	2854KB	900KB	4096KB	54
MAX	1750MB	8960KB	4096KB	4096KB	94

Total memory required for data varies from 193kb to 1.75gb and averages 665MB. Memory per processor required ranges from 900KB to 4MB. For programs like VORTEX, which fetches data at the beginning of the program and work on the same dataset during most of the execution, much smaller memory space is needed than some other programs, which are constantly fed with new data.

Processing Requirements

APPLICATION	FLOAT OPS.	FLOAT OPS/PROC	PRECISION
CLIMATE	2970G	12200M	32
SEMI	10000G	20000M	64
MOLECULE	1000G	2000M	32
RENDER	24G	768M	32
EXFLOW	3994G	7987M	32
QCD	119G	474M	32
VORTEX	42G	677M	32
REACT	27648G	55296M	64

The single most notable common characteristic of the applications studied is that they are all floating point intensive. Investment in improving floating point operations would seem justifiable. However, everything else might scale as well and we will cover that issue in a later slide.

I/O Requirements

APPLICATION	INPUT	OUTPUT	VOLUME	VOL/MFLOP	DISK	TAPE
CLIMATE	1MB	1500MB	1500MB	517B	10MB	1500MB
SEMI	10MB	100MB	1000MB	100B	1000MB	0MB
MOLECULE	0MB	0MB	0MB	0B	0MB	0MB
RENDER	180MB	28MB	208MB	8858B	208MB	0MB
EXFLOW	0MB	1MB	1MB	0B	1MB	0MB
QCD	0MB	6MB	6MB	52B	6MB	0MB
VORTEX	0MB	0MB	0MB	3B	0MB	0MB
REACT	0MB	160MB	3400MB	126B	1600MB	0MB
AVE	24MB	224MB	764MB	1207B	353MB	187MB
MAX	180MB	1500MB	3400MB	8858B	1600MB	1500MB

Again the numbers vary here.

2 categories:

CLIMATE, SEMI, RENDER and REACT

---- perform I/O regularly throughout the run

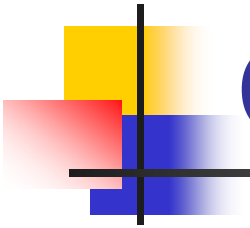
MOLECULE, EXFLOW, QCD and VORTEX

---- perform I/O only at the beginning and end of the run

Communication Requirements

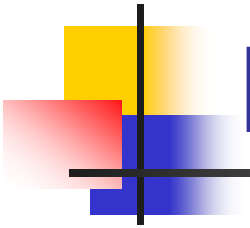
APPLICATION	VOLUME	VOL/MFLOP	COUNT	COUNT/MFLOP	AVE SIZE
CLIMATE	965GB	325KB	1956M	660.0	505B
SEMI	120GB	12KB	15M	1.5	8192B
MOLECULE	1956GB	1956KB	44M	44.0	45568B
RENDER	2GB	98KB	0M	0.2	512000B
EXFLOW	562GB	144KB	256M	65.6	2248B
QCD	7GB	57KB	94M	810.0	72B
VORTEX	1GB	35KB	1M	29.4	1245B
REACT	132GB	5KB	12M	0.4	11264B
AVE	468GB	329KB	297M	201.4	72637B
MAX	1956GB	1956KB	1956M	810.0	512000B

Most applications here were developed with the communication capabilities of the current generation of parallel machines in mind. As a result, it seems likely that future applications will have even larger communication loads than those presented here. In additions, communication demands tend to come in bursts, further increasing the need for high bandwidth in the communication network.



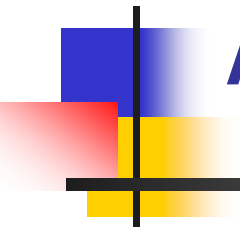
General Characteristics

- Number Crunching Applications
  - Typically have large number of arithmetic (floating point) operations.
- Typically large data and working sets
  - However, this depends on the application.
  - For example, Vortex has a small data set.
- Typically low temporal locality
  - Data once used, is not typically not reused in the near future.
  - For example, in vortex the position of the vortices are loaded, the forces between them are computed and the positions change.
  - This process is repeated for every stage. There is hardly any temporal locality as the position and forces are loaded atmost once every stage.
- Depending on regularity of application – can have high spatial locality



Parallelism

- Lots of DLP, TLP, & ILP
- DLP
  - Same operation performed on all bodies
- TLP
  - Convert DLP to TLP
  - More flexibility compared to DLP
- ILP
  - Parallelism within “threads”
- Example: Vortex
  - Mostly DLP
  - DLP converted to TLP

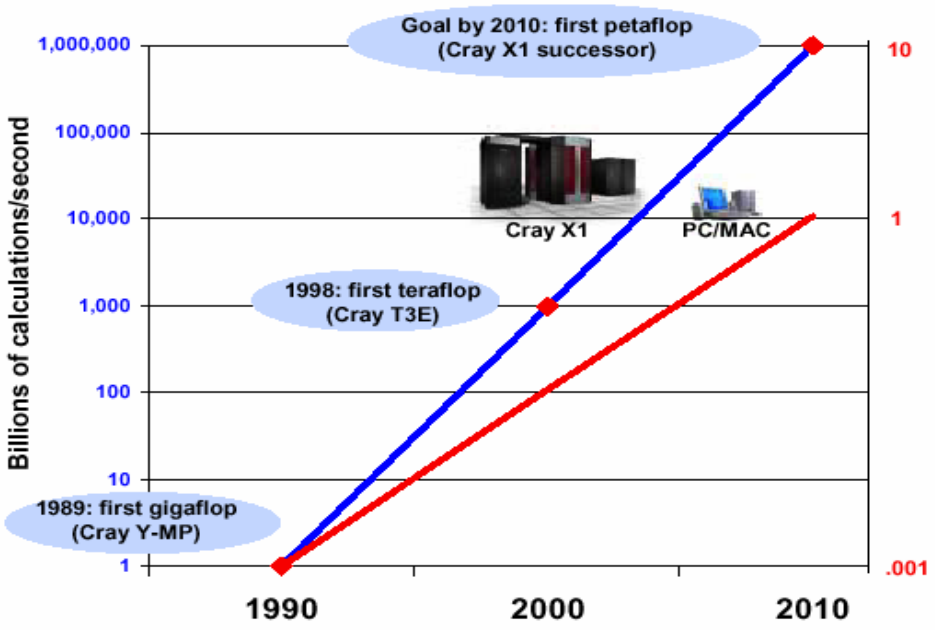


Architectural Issues

Performance Trends

Scaling faster than Moore's law!

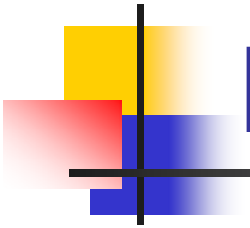
The graph below shows how supercomputers have scaled, and are projected to scale over period of 20 years. Starting from the Cray Y-MP machine at 1990, we can see that the expectation of supercomputer performance has grown faster than conventional desktops which scale with Moore's law.
This is because not only have the processors scaled themselves, but so has the number of processors in any given system, and the number of processing units in each chip.



Supercomputer speed *
Moore's Law rate (2x every 18 mos.)

Year	Supercomputer speed: microprocessor speed
2010	100,000:1
2000	10,000:1
1990	1,000:1

taken from Cray Research website: www.cray.com



Processing Requirements

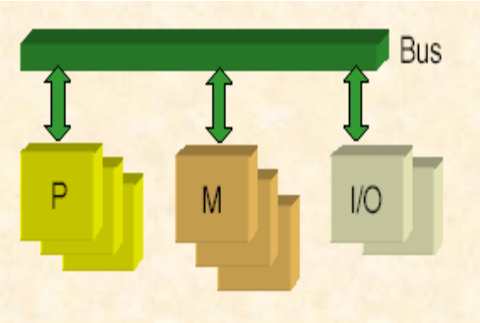
Petaflops?

- One of the standard measures of performance is the peak, or theoretical upper bound, of the number of floating point operations that can be executed per second. Several applications are now expected to need in the order of 10^{15} FLOPS or Petaflops over the next few years. Currently, the fastest system is the NEC Earth simulator clocking approximately 40 TFLOPS.
- It is important to note that peak performance is not an accurate measure; It is the sustained performance, which can be drastically lower due to other bottlenecks in the system, which gives a true measure of performance. Some systems currently run at as low as 10-15% efficiency.

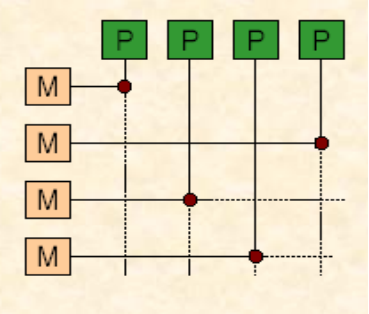
- Two approaches to achieve computational capacity:
  - Cluster Systems: typically 100s-1000s of processors.
  - Stream/Vector Systems: fewer custom designed highly powerful processors E.g. Stanford Streaming Supercomputer

Interconnection Networks

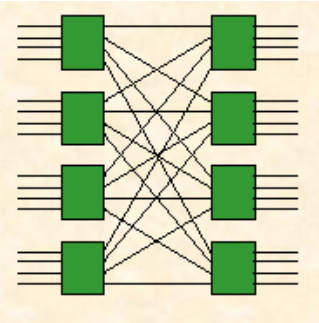
Both BW and latency important – In many systems, there is a potential to tradeoff BW for latency, or vice versa. It is not the case here since the data sets involved are often very large and so is the number of processing elements – the latency from memory to processor could result in several millions of cycles in fast processors being lost waiting for data.



Bus:
- Simplest network – single medium shared by all elements.
- Need arbitration protocol
- Only one device at a time.
- This network doesn't provide the BW required to support multi-processor networks

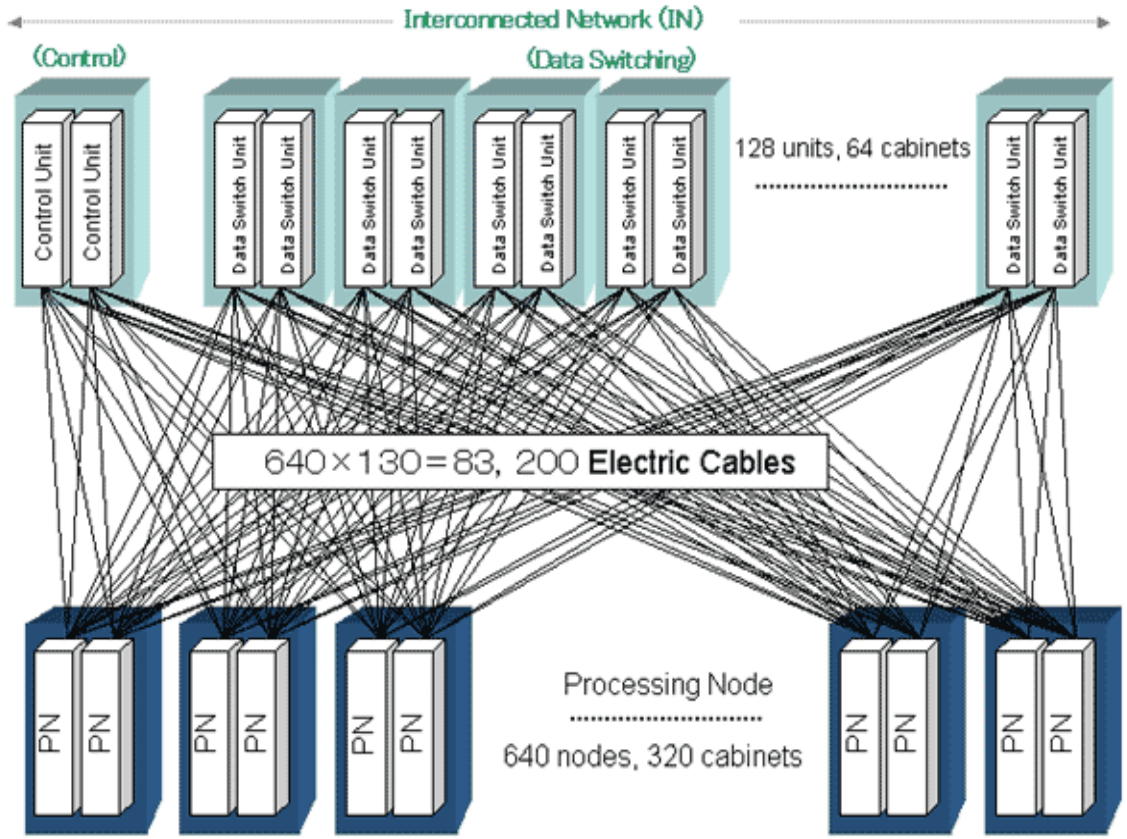


Crossbar Switch:
- The other extreme – connected all elements to each other.
- All processors & memories connected
- $O(N^2)$ – doesn't scale well can end up with several Ks of wiring.
- Used in smaller networks.



Multistage Switch:
- A hybrid approach that is scalable.
- N X N switches built from smaller switches E.g. 16X16 built from 2 stages of 4X4
- Has many issues with contention resolution and overhead of setting up connection every time.

Example: NEC's Earth Simulator*



* taken from NEC Earth simulator website: www.nec.co.jp

Architectural Issues in Vortex

Here, we examine the scalability of the Vortex benchmark, and the requirements it imposes on the underlying architecture, as we scale both the number of processors –P, and the size of the data set (number of vortices) - N

Data Memory	O (N)	The memory needed is proportional to the number of vortices.
FLOPS	O (N^2 + P)	The number of floating point operations executed per cycle depends quadratically on the number of nodes (since every node reacts with every other node) and linearly with the number of processors, since the final step in each iteration combines the data accumulated at each processor.
I/O Volume	O (N)	The total data passed between the disk and processors is just the data per vortex at the beginning and end of each iteration.
Communication Volume	O (NP + P^2)	The total amount of data exchanged has two components: one dependent on data (O(N/P) per processor) and one overhead (O(1) per processor). By the argument presented in the next point, the effective volume is O((N/P +P) *P^2)
Communication Count	O (P^2)	Since each processor eventually has to communicate with each other processor, the total number of messages sent scales quadratically with the number of processors.

Since the number of processors generally scales along with the data set, and taking into account that processor speeds scale faster than interconnect speeds, the bottleneck in this system is the communication, or interconnection, network.

Vortex Running Time and Scalability

APPLICATION	TIME	COMPUTATION	I/O	COMMUNICATION
CLIMATE	292 min	25	3	72
SEMI	108 min	58	38	4
MOLECULE	59 min	35	0	65
RENDER	3 min	65	15	20
EXFLOW	216 min	76	4	20
QCD	133 min	77	9	14
VORTEX	24 min	90	0	10
REACT	132 min	83	10	7

The table shows the running time of different applications and also the percentage of the running time spent in computation, I/O, and communication. The time spent on each of the components varies considerably depending on the application – for example, vortex spends negligible time on I/O where as SEMI spends about 40% of the time doing I/O.

An interesting observation from this data is the scalability of the application. Applications that have significant amount of communication do not scale very well with increase in the number of processors for the same data set size. This is because the data is divided among more number of processors, causing the communication to increase further.

Current Design Challenges

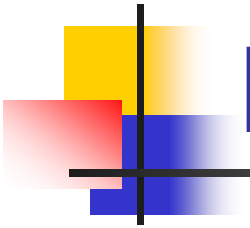
- **System Performance-to-Cost ratio**: millions of dollars to build
  - Custom vs. Cluster systems – The development cost associated with designing custom vector or streaming processors can often be offset by increased overall throughput generated over a long period of time. Often, generic micro-processors use a lot of chip area in caches, h/w to exploit ILP etc that is not critical to supercomputer applications.
  - It is the $/FLOP metric which is important to optimize.
- **Programming model not very intuitive**
  - Parallel programming is a non-obvious task, that is very important in machines such as this where it is critical to be able to distribute the data among hundreds of parallel processing units efficiently and in a scalable fashion.
- **I/O Scalability**
  - Increasingly, the bottleneck in high performance systems is becoming the IO (Memory-Processor) interconnects. A hot topic of research is to replace the wires with optical links; the cost for receiver and transmitters are still prohibitive for this option
- Load Balancing
  - Depending on the application, allocating "work" to the different processors such that the workload is balanced, is an important design challenge. There has been a big effort to achieve load balancing in software from Sandia labs – Sierra framework.
- **Power**
  - Becoming an increasingly important issue as the number of processors has increased, as has the number of execution units per chip. Liquid cooling system are not uncommon, and this is becoming an increasingly important criteria in system design.

Benchmarks

DLAB suite –measuring performance of distributed & resource sharing systems on scientific applications

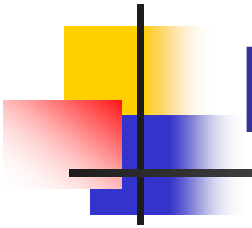
Performance Type	Measuring Benchmark
Floating Point	LFK, Linpack, Nbench-byte-2.1, EuroBen-V3.9, NPB2.3-serial
Integer Arithmetic	Nbench-byte-2.1, NPB2.3-serial (IS)
Memory Subsystem	Stream, Nbench-Byte-2.1, Stream_OpenMp
Communications	PMB-MPI1, Eff_bandwidth, NPB2.3-parallel
Sample Compact Applications	NPB2.3-Parallel (CFD), Parallel_Chem (Chemistry)
Full Applications	Angus (CFD), SBLI (CFD), DLPOLY-2.13 (Molecular Dynamics)

taken from CCLRC UK website: http://www.cse.clrc.ac.uk



Benchmarks

- Two important performance measures:
  - Peak Performance – dependent on maximum computation capacity. eg. Linpack
  - Sustained Performance – depends on overall system architecture (interconnects, memory BW)
- DLAB measures this and other characteristics



References

1. Cypher R, Ho A, Konstantinidou S, Messina P
   A Quantitative Study of Parallel Scientific Application with Explicit Communication
   JOURNAL OF SUPERCOMPUTING 10 (1): 5-24 1996
2. A. Agarwal and A. Gupta. Memory-reference characteristics of multiprocessor applications under MACH. In Proc. 1988 ACM SIGMETRICS Conf. On Measurement and Modeling of Computer Systems
3. Roberto Ansaloni, Paolo Malfetti and Tiziana Paccagnella

   Porting A Limited Area Numerical Weather Forecasting Model on a Scalable Shared Memory Parallel Computer. 5th European SGI/Cray MPP Workshop, 1999
4. Cray: www.cray.com
5. Sierra Framework: http://csmr.ca.sandia.gov/projects/ftalg/Edwards02.pdf