Commodity High Performance Computing

David May

Trends

Simulation and modelling pervading engineering and science

Increasing need for access to immense volumes of on-line data everywhere

Data-mining and real-time analysis becoming essential for competitiveness

Trends

End of desktop PCs and workstations

Thin clients - terminals, laptops, mobiles,

High performance servers - in the basement or on the internet

Exabytes!

The Internet in 2010:

- Exabytes everywhere!
- Petabyte servers, petaflop supercomputers, a billion hosts, a zetabyte online
- Services to ubiquitous and nomadic clients including storage and compute (note Amazon EC2, S3, SQS)
- Many different forms of content - audio, video, games, 3D ...

HPC technology

HPC chip volumes will overtake PC volumes, with potentially disruptive changes in architecture - HPC components will be the new commodity

A key driver is likely to be technology optimised for internet servers

Specialised versions will do modelling, simulation, data mining

History

Architecture and implementation of general purpose processors driven by commodity desktop PC

Features driven by games requirements!

Power consumption limited only by wall socket

Performance impossible to understand - processor clock frequency used as performance measure!

Technology limitations forcing move to multiple cores

Multicore PC chips

A lot of effort is going into big multicore chips

Predictably these focus on symmetric multiprocessors with complex memory hierarchies

Its not clear what these are for

- we won't need - or want - big processors in PCs - we'll be using thin clients and portables
- we will need big servers - but these can be built from larger numbers of slower, more power-efficient processors

Expectations vs. Reality

Symmetric multiprocessors are easy to program

... provided you're not bothered about performance!

A single general purpose processor places a heavy demand on its memory system

... you can't expect the memory to support many processors

Multiprocessors need *concurrent programs* and/or *parallel algorithms* - but then the architecture can be much simpler!

Three programs (1)

```
{ int i;
  int n;
  int sum;
  sum = 0;
  n = 0;
  for (i=0; i<maxdata; i++)
  { sum = sum + a[n];
    n = n + 1;
    if (n > maxdata)
      n = n - maxdata;
  }
}
```

Three programs (2)

```
{ int i;
  int n;
  int sum;
  sum = 0;
  n = 0;
  for (i=0; i<maxdata; i++)
  { sum = sum + a[n];
    n = n + 1000;
    if (n > maxdata)
      n = n - maxdata;
  }
}
```

Three programs (3)

```
{ int i;
  int n;
  int sum;
  sum = 0;
  n = 0;
  for (i=0; i<maxdata; i++)
  { sum = sum + a[n];
    n = a[n] + 1000;
    if (n > maxdata)
      n = n - maxdata;
  }
}
```

Three programs - performance

In 1980, these programs would have had almost the same performance

Today, the difference on a state-of-the-art computer is at least 50

How can efficient software be designed for unpredictable machines?

How much software is running 50 times slower than it should?

How many 50-CPU clusters have been bought where a single computer would do?

Technology evolution

We will see changes in technology aiming to maximise energy efficiency

These will include

- more, simpler, lower-speed processors
- programmable accelerators for specialised tasks such as transcoding, cryptography, searching ...

The *accelerator-in-server* market will be opened up by the trend towards Ethernet as the standard interconnect

Power-efficiency - Google

“Servers are commodity-class x86 PCs running customised versions of Linux. Indeed, the goal is to purchase CPU generations that offer the best performance per unit of power, not absolute performance. Estimates of the power required for over 450,000 servers range upwards of 20 megawatts, which could cost on the order of US$2 million per month in electricity charges”

Wikipedia

Software

May's Law: *Software efficiency halves every 18 months, compensating Moore's Law*

Throw away software if no-one understands it

... there's no disposal cost and no need to re-cycle it!

You may not need a supercomputer - just a better program or algorithm!

Use simple concurrent programming tools - not Java and MPI!

Design algorithms to exploit concurrency!

Software and Algorithms

This is a big opportunity - efficient software can add a lot of value by increasing performance and power-efficiency

And big data sets bring big opportunities for better software and algorithms:

Reducing the number of operations from $N \times N$ to $N \times log(N)$ has a dramatic effect when N is large

... for $N = 30\,billion$ this change is as good as 50 years of technology improvements!

HPC technology

A lot of processors will fit on a state-of-the-art chip!

Multicore arrays

... could have 1000 interconnected processors within 3 years

... could deliver up to one trillion operations/second chip

... can be designed to support simple concurrent programming

They are potentially an important generic platform technology

Multicore arrays

Concurrency, concurrency and concurrency

- inside processors
- in multicore chips
- in concurrent computers

Expose it and exploit it - don’t hide it

Multicore arrays

May's second law: *Compiler technology doubles efficiency no faster than once a decade*

Design the hardware to support the software; don't expect sudden advances in tools and programming

Slow the clock and simplify the processor

Build simple general purpose processor arrays with non-blocking interconnect

Personal HPC

Power efficient processor arrays will become cheap standard components

An increasing variety of applications will exploit them - including games!

... will HPC move back to the desktop?

Embedded HPC

Low cost embedded HPC will

- replace applications specific hardware
- open up new application areas and accelerate existing ones

Examples are

- basestations
- transcoding, compression and crytography
- monitoring, control and robotics

Generic platforms used to build “applications-specific” devices by programming.

Robotics

Scalable real-time control - computers in the loop:

“... there are about 650 named muscles in the human body and thousands of unnamed ones ...”

or perhaps:

“... there are around 650 named actuators in a humanoid robot and thousands of unnamed ones ...”

and every one will need a processor to control it!

Summary

HPC will become commodity; concurrency will play a central role

There is plenty of scope for new ventures:

- processors and processor arrays optimised for HPC
- programming tools
- implementing and re-implementing applications software
- applications specific systems designed by programming EHPC platforms
- fast communications and switches
- on-line HPC services