1-nc-1

GPU Microarchitecture Note Set 1c—Program Needs and System Cap

Performance Measures

Latency v. Throughput

1-nc-1

Program Needs and System Capabilities

1-nc-2

Our Goal:

Determine whether we should be satisfied...
... with the performance of our parallel program...
... on our system.

We are not satisfied when the program runs more slowly then we expect.

Knowing when to be satisfied is a key skill.

1-nc-3

The Idea:

- Estimate the *computation needs* of our problem, algorithm, or program.

  For example, 10^{20} floating-point operations.

- Determine the *computation capabilities* of our system.

  For example, 10^{17} FLOPS.

- Use these to estimate execution time.

  For example, $10^{20}/10^{17} = 1000\,\mathrm{s}$.

If the estimate does not match the measured value, find the cause.

1-nc-4

Computation Needs

These refer to a program, algorithm, or problem.

Determined by analyzing a problem, algorithm, or program.

Common Computation Needs:

- Floating-Point Operations
- Data Transfer (bytes read from and written to memory).
- Instruction Count (Number of executed instructions)

1-nc-5

Computation Capabilities

These refer to the capabilities of a system (CPU, GPU, etc.)

Common Computation Capabilities

Floating Point Execution Rate - FLOPS

Number of floating-point operations divided by amount of time.

Instruction Execution Rate - IPC

Number of executed instructions divided by amount of time.

Data Transfer Bandwidth - B/s

Number of bytes crossing some line, divided by amount of time.

1-nc-6

Computation Need: Number of Floating Point Operations

Commonly used for scientific programs...
... because such operations are considered essential (can't be avoided) and...
... because of the historic high cost of FP hardware.

Example: fragment of some scientific code.

```
double *x, *a, *b, *c;
for ( i=0; i<1e9; i++ ) x[i] = a[i] + b[i] * c[i];
```

Uses $10^9 \times 2 = 2 \times 10^9$ FP operations.

Consider a system capable of 100 GFLOPS (10^{11} FLOPS).

Execution time bound is 20 ms.

If measured execution time was 100 ms we would not be satisfied.

1-nc-7

Computation Need: Data Transfer

Used for all kinds of programs.

Easy to measure when each data item read exactly once.

Otherwise, need to account for cache and scratchpad performance.

1-nc-8

Data Transfer Example:

```
double *x, *a, *b;
for ( i=1; i<1e9-1; i++ ) x[i] = a[i-1] * l + a[i] + a[i+1] * r;
```

The size of a double is 8 bytes.

In an iteration, three elements of `a` are accessed...
... two were accessed in a prior iteration...
... and so shouldn't need to be read from memory...
... leaving one new element of `a` accessed per iteration.

Total data transfer needs: $10^9 \times 8 \times 2 = 16\,\text{GB}$.

Suppose system can transfer $30\,\text{GB/s}$.

Performance bound: $16/30 = 533\,\text{ms}$.

1-nc-9

Data Transfer Example, Continued.

Suppose measured execution time were just 1.07 s. We are not satisfied.

Maybe we were wrong about each element of `a` being read once.

Maybe we should look at number of FP operations.

Maybe we should look at number of instructions.

1-nc-10

Computation Needs, Matrix Multiplication

Code for multiplying two $d \times d$ matrices.

```
for ( int r=0; r<d; r++ )
  for ( int c=0; c<d; c++ )
    for ( int k=0; k<d; k++ )
      c[r,c] += a[r,k] * b[k,c];   // This line executed d^3 times.
```

Data Transfer

Size of each matrix: d^2 elements.

Naïve data bound: $3d^3$.

Minimum data bound: $3d^2$.

Computation

Straightforward: d^3 multiply/adds.

But, $O(d^{\log_2 7}) \approx O(d^{2.807})$ possible.

1-nc-11

Instruction Count and Execution Rate

Considered a less fundamental bound than FLOPS and data transfer.

Number of instructions determined by many factors:

How program is written.

Quality of compiler.

Instruction set of processor.

1-nc-12

Performance Estimation

With above needs and capabilities can compute three performance bounds.

Actual performance will be no greater than the smallest of these.

Latency and Throughput

Commonly used types of performance measure...
... for describing components or whole systems.

Latency:
The time needed to do something from start to finish.

In discussion of GPUs, latency often refers...
... to the execution time of a single instruction or of a single thread.

A processor is called *latency oriented* ...
... if it has low latency (runs a single thread quickly).

1-nc-14

Throughput:
The number of things completed per unit time.

Throughput can refer to many possible things, including:

Instructions per cycle (or second).

FLOPS.

Vertices per second. (For graphics.)

Transactions per second. (Database operations.)

A processor is called *throughput oriented* if it has high throughput.

Latency v. Throughput

1-nc-15

Can't Have Both