Topic 14: Parallelism

COS 320
Compiling Techniques
Princeton University
Spring 2018
Prof. David August

Moore's Law

Source: Intel/Wikipedia

Single-Threaded Performance Not Improving

Source: Intel/Wikipedia

Final Exam!

- Thursday May 3 in class
- Closed book, closed notes
Decoupled Software Pipelining (DSWP)

A: while (node)  
B: ncost = doit(node);  
C: cost += ncost;  
D: node = node->next;

Inter-thread communication latency is a one-time cost.

Implementing DSWP

DFG

L1:

Aux:

Optimization: Node Splitting To Eliminate Cross Thread Control

L1:

A: \ r1 = M[r1] 
B: \ r2 = r1 + 4 
C: \ r3 = M[r2] 
D: \ r4 = r3 + 1 
E: \ M[r2] = r4

L2:

A: \ r1 = M[r1] 
B: \ r2 = r1 + 4 
C: \ r3 = M[r2] 
D: \ r4 = r3 + 1 
E: \ M[r2] = r4

CONSUME r1 = [1] 
CONSUME p1 = [2] 
G: br p1, L2

Register control memory

- intra-iteration - loop-carried

[MICRO 2005]
Optimization: Node Splitting To Reduce Communication

Constraint: Strongly Connected Components

Era of DIY:
- Multicore
- Reconfigurable
- GPUs
- Clusters

P6 SUPERSCALAR ARCHITECTURE (CIRCA 1994)
MULTICORE ARCHITECTURE (Circa 2010)

Automatic Pipelining
Automatic Speculation
Automatic Allocation/Scheduling
Commit
Parallel Resources

Realizable parallelism

Compiler Advances Double Computing Power Every 18 Years!
– Proebsting’s Law

Credit: Jack Dongarra
**Example**

A: while (node) {
B:   node = node->next;
C:   res = work(node);
D:   write(res);
}

Program Dependence Graph

---

**Spec-DOALL**

Example

A: while (node) {
B:   node = node->next;
C:   res = work(node);
D:   write(res);
}

Program Dependence Graph
Example

A: while (node) {
B: node = node->next;
C: res = work(node);
D: write(res);
}

Program Dependence Graph

Spec-DOALL

Spec-DOACROSS

Spec-DSWP

Comparison: Spec-DOACROSS and Spec-DSWP

Throughput: 1 iter/cycle

Comparison: Spec-DOACROSS vs. Spec-DSWP

Geomean of 11 benchmarks on the same cluster

Throughput: 1 iter/cycle

Performance Speedup (X)

(Number of Total Cores, Number of Nodes)