Topic 14: Parallelism

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

Final Exam!

- Thursday May 3 in class
- Closed book, closed notes

Moore’s Law

Source: Intel/Wikipedia
Decoupled Software Pipelining

Decoupled Software Pipelining (DSWP)

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

[MICRO 2005]
Implementing DSWP

Optimization: Node Splitting
To Eliminate Cross Thread Control

Optimization: Node Splitting To Reduce Communication
Constraint: Strongly Connected Components

Consider:

A \rightarrow r1
B \rightarrow r2
C \rightarrow r3
D \rightarrow r1
E \rightarrow r4

SPAWN(A
x) = A
x + M[1]

A:
B:
C:
D:
E:

\text{PRODUCE}[1] = r3
\text{CONSUME}[1] = r1
\text{PRODUCE}[2] = r2
\text{CONSUME}[2] = r1

\text{register control memory}

\text{intra-iteration loop-carried}

Solution: \text{DAG}_{\text{SCC}}

A \rightarrow F \rightarrow G
B
C \rightarrow D \rightarrow E

Eliminates pipelined/decoupled property

Era of DIY:

- Multicore
- Reconfigurable
- GPUs
- Clusters

10 Cores!
10-Core Intel Xeon
"Unparalleled Performance"

P6 Superscalar Architecture (Circa 1994)

Automatic Speculation
Automatic Pipelining
Commit
Parallel Resources
Automatic Allocation/Scheduling

SPECLINT Performance (If-scalar)

CPU92
CPU95
CPU2000
CPU2006

Year

656 years behind
MULTICORE ARCHITECTURE (Circa 2010)

- Automatic Pipelining
- Parallel Resources
- Automatic Speculation
- Automatic Allocation/Scheduling
- Commit

Realizable parallelism

Parallel Library Calls

Threads

Credit: Jack Dongarra
Compiler Advances Double Computing Power Every 18 Years! – Proebsting’s Law

P6 SUPERSCALAR ARCHITECTURE

Spec-PS-DSWP

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

Program Dependence Graph

A
B
C
D

A1
B1
C1
D1
A2
B2
C2
D2

Control Dependence
Data Dependence
Comparison: Spec-DOACROSS and Spec-DSWP

Spec-DOACROSS vs. Spec-DSWP

[MICRO 2010]
Geomean of 11 benchmarks on the same cluster