Colorado State University

CS 575 Parallel Processing
Lecture 2: Platforms

Sanjay Rajopadhye

Parallel Machine Architecture

- Microprocessors:
  - Build parallel machines out of COTS
  - Single core → ILP → VLIW → …
  - Multi core → many core → accelerators
- Memory System
- Control vs Communication
- Shared memory vs distributed memory

Until 2003: //ism for free

- Parallel machines (aka "supercomputers")
  - Users with deep pockets – governments
  - But still use COTS (lessons of early 90s)
- COTS riding Moore's law
  - Pipelining and superscalar execution
  - VLIW (Very large instruction width) arch
  - Speculation/branch prediction
  - Reservation stations
  - SIMD (subword) parallelism: SSE, MMX, etc.

Memory System Performance

Two main factors

- Latency (almost always can be "hidden")
  - Prefetch (anticipate a request)
  - Virtualization (do other things)
- Bandwidth (harder problem)
  - Improve the balance of the program/system

See HW0A-D

Caches on Multicores

- Uniform Memory Access (UMA) vs NUMA
- Coherence Problem
  - Multiple copies of the same data (i.e., memory address) may be present
  - Every write operation needs to ensure that other copies, if any, are marked as "invalid"
  - Many protocols – active research topic in architecture
- False sharing (cache transactions are at the granularity of a line)

Cache Coherence

- Classic tradeoff:
  - Communication to propagate the signals of the protocol and respond to them
  - Performance, i.e., speed of servicing the request (avoid stalling)
  - Scalability of the solution with the number of cores
- Snoopy caches (simple implementation)
- Directory based systems (better scalability)

Control Structure

- Flynn's taxonomy of parallel processors
  - SISD (sequential)
  - SIMD
    - SPMD
    - STMD
  - MIMD
  - MISD (??)
- Current state: SIMD-MIMD combination
  - Heterogeneous collection of homogeneous "cores"
  - Architectural question: what is the balance

Distributed Memory (communication)

Distributed memory multiprocessors

- Collection of nodes connected by an in interconnection network
- Each node is a COTS microprocessor
- Distinct address space on each processor

- Key questions: network topology
  - Number of nodes
  - Number of edges
  - Communication algorithms/API

Interconnection Networks

- Direct vs Indirect ("static" vs "dynamic")
  - Single stage vs multi-stage
- Network topology (parameterized)
  - Bus based
  - Crossbar based
  - Based on families of graphs

MINs: indirect networks

- Perfect shuffle & shuffle exchange networks (e.g., Omega network)
  - Routing algorithms (digit-based)
    - Asynchronous
    - Perfectly distributed (decentralized)
  - Blocking vs non-blocking
  - Contention

Direct Networks

- Linear arrays, meshes, rings, tori, and *k-d* meshes
- Hypercubes
  - special case of *k-d* meshes (*k=2*)
  - Many special properties
- Trees
  - Static vs dynamic
  - Fat trees

Network Performance Metrics

- Diameter
  Longest distance between any two nodes
- Connectivity
  minimum # of edges to remove to make graph disjoint
- Bisection width
  min # of edges to remove to get disjoint but (nearly) equal (sub) graphs
- Bisection bandwidth
  min communication volume between equal halves = bisection width/link-bandwidth

Communication Costs

- Startup cost
  - Pack/unpack message
  - Interface to router
  - Initiate/negotiate routes
- Per-hop time
  - Node latency
  - Buffering at router
- Per-word transfer time
  - Channel bandwidth (reciprocal of)
  - Often lumps network & buffering overheads

Store & forward vs cut-through

- Store & Forward
  - Each hop is like a separate message
- Packet routing
  - Pipeline the transfers of each hop
  - Pay startup only once
  - Acquire and buffers all along the route at once
- Cut-through routing plus
  - Optimization
  - Buffer allocation to guarantee no deadlock
  - Fixed routes and in-sequence delivery

Analytic models

- Store & Forward

$$T_{\text{comm}} = t_s + mlt_w$$

- Packet routing & cut through

$$T_{\text{comm}} = t_s + lt_h + t_w m$$
$$\approx t_s + t_w m$$

Task-to-Processor Mapping

- Theory of graph embedding
  - Some edges may be dilated
  - There may be congestion
- Active research in 90s
  - Optimality
  - Impact on design of interconnection networks
- Dimension ordered routing

Conclusions

- Brief summary of the architecture/hw parameters of parallel machines
- Book somewhat dated
- Many ideas reappear today (under new constraints)
  - Multi- and many-core architectures

  e.g., the memory traffic on NVIDIA GPUs is managed over a fat tree like interconnection network.