Towards Efficient Computing

Per Stenström

Chalmers University of Technology
Sweden
Congratulations Yale!

...“Ö”...
(...Island ahead...)
Threats and Opportunities

• Parallelism is ubiquitous but hard to deal with
• Power is heavily constraining performance growth
• Moore’s Law is running out of steam

A radical new way of thinking of compute efficiency is needed
Yale’s Transformation Hierarchy

- Problem
- Algorithm
- Program
- ISA
- Microarchitecture
- Circuits
- Electrons
First Revision

"Electrons: Do my task efficiently"

“Roger that!”
Second Revision: My Hierarchy

Problem

Algorithm

Program

ISA

Resource Management

Microarchitecture

Circuits

Electrons
(My) Vision for Efficient Computing

• **P1: Parallelism:**
  – **Programmers:** Unlock parallelism and give hints
  – **Resource manager:** Translate it into higher performance “under the hood”

• **P2: Power:**
  – **Programmers:** Express quality of service attributes
  – **Resource manager:** Translate it into more efficient use of hardware resources “under the hood”

• **P3: Predictability:**
  – **Programmers:** Express deadlines (absolute or “soft”)
  – **Resource manager:** Manage parallelism predictably “under the hood”
Approach – Interaction Across Layers

ISA

Resource Management

Microarchitecture
Parallelism Management
Task-based Dataflow Prog. Models

void TaskA (float a[M][M]);

void TaskB (float a[M][M]);

void TaskC (float a[M][M]);

• Programmer annotations for task dependences
• Annotations used by run-time for scheduling
• Dataflow task graph constructed dynamically

**Important**: Conveys semantic information to run-time for efficient scheduling
Possible Optimizations

Dependency annotations allow for optimizations with high accuracy (like in message passing)

- Bulk data transfer
- Forwarding
- Prefetching
- Migratory sharing optimization
Run-time Guided Cache Coherence

- Self-invalidation provides significant gains
- SP+D+I provides added gains
Other Opportunities

- Give run-time system the responsibility to manage cache hierarchy resources just like virtual memory manager or hypervisor manages memory resources.
- Use data-flow graph notion (explicit or inferred dynamically) to exploit speculative parallelism with high success rate.
- Migrating computation rather than data, by exploiting semantic information about data usage.

*MECCA is investigating these opportunities*
Power Management

What if

• Users expressed how long time a computation must take?
• Resource manager could track progress against deadlines?
• Resource manager could predict the remaining time as a function of resources?

Opportunities:

• Controlled throttling of resources
• Controlled scheduling of computations on heterogeneous substrates

In general: Considerable room for trading performance for reduced power consumption

MECCA is investigating these opportunities
Predictability Management

**Context:** Real-time applications

**Sequential processing:** Establishing tight bounds on execution time (WCET) is fairly well understood

**Parallel processing:** Unexplored terrain

Deterministic scheduling => New playground for trading performance for power under strict timing guarantees

**MECCA is investigating these opportunities**
Questions?

...“Ö”...
(...Island ahead...)