The Future of Microprocessor-based Computation (is Highly Parallel)

Kurt Keutzer
and the PALLAS Team
Michael Anderson, Bryan Catanzaro, Jike Chong, Katya Gonina, Dorothea Kolossa, Chao-Yue Lai, Mark Murphy, Bor-Yiing Su, Narayanan Sundaram
UC Berkeley

Moore’s Law: Transistor Count

© Kurt Keutzer
Pipeline Evolution

<table>
<thead>
<tr>
<th>Instruction Fetch</th>
<th>Instruction Decode</th>
<th>Execute</th>
<th>Memory</th>
<th>Writeback</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFU1</td>
<td>IFU2</td>
<td>IFU3</td>
<td>DEC1</td>
<td>DEC2</td>
</tr>
<tr>
<td>RAT</td>
<td>ROB</td>
<td>D/S</td>
<td>EX</td>
<td>RET1</td>
</tr>
<tr>
<td>RET2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Netburst</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

What’s wrong with that?

- 5 stage pizza pipeline – 1 pizza/minute
- 10 stage pizza pipeline – 1 pizza 30/seconds
- 20 stage pizza pipeline – 1 pizza/20 seconds
- 31 stage pizza pipeline – too much heat in the kitchen

http://jansdough.janktheproofer.com/Make-Pizza-tutorial.htm
Results: Pipelined Sequential Performance

Alternative: Parallelism

10 stage pizza pipeline – 1 pizza 30/seconds

two 10 stage pizza pipelines – 1 pizza 15/seconds

four 10 stage pizza pipelines – 1 pizza 7.5/second???
Parallelism in HW

- Intel: Larrabee
- Nvidia: Fermi

We can build them … but …

- No question we can build these processors
- Big question: How do we program 32 – 512 processors to solve your favorite application?
  - Perform content-based image retrieval
  - Large-vocabulary speech recognition
  - MRI image reconstruction
  - Value-at-risk analysis in quantitative finance
  - Etc.
- Parallel processors are more efficient, if we can program them.
- So … the future of microprocessors is really about software.
- That’s why the rest of the talk is all about programming applications on parallel microprocessors
How NOT to parallel program

Initial Code

Profiler

Re-code with more threads

Performance profile

Not fast enough

Fast enough

Lots of failures

Software running on N processor cores is slower than on 1

Ship it

What is this person thinking of?

Re-code with more threads

Lots of failures

Threads, locks, semaphores, data races

Mood: anxious, depressed

Edward Lee, “The Problem with Threads”
Our PALLAS Methodology

- Application Specification
  - Architect SW to Identify Parallelism
  - Thought Experiment: Map SW Arch. to HW Arch.

Thought Experiment:
Map SW Arch. to HW Arch.

Performance profile
- Not fast enough
- Fast enough

Write Code
- Ship it

What is this person thinking of?

- Today’s code is serial but the world is parallel
  - we just need to find the parallelism in our applications and reflect it in our software

Re-architect with patterns

- Computational patterns
- Structural patterns

Software architecture
Mood: well-adjusted, optimistic

© Kurt Keutzer
Our formula

- State-of-the-art algorithms
- Tall-skinny parallel programmers
- Domain experts
- Compelling applications
- Architect software with patterns
- Parallel hardware

Game changing speed-ups (with scalability)

Applications

- Our formula in a bit of detail: MRI reconstruction
- Other applications where our formula has proven itself
- Applications in progress
Compelling Application: Fast, Robust Pediatric MRI

- Pediatric MRI is difficult:
  - Children cannot sit still, breathhold
  - Low tolerance for long exams
  - Anesthesia is costly and risky
- Like to accelerate MRI acquisition
  - Advanced MRI techniques exist, but require data- and compute-intensive algorithms for image reconstruction
- Reconstruction must be fast, or time saved in accelerated acquisition is lost in computing reconstruction
  - Non-starter for clinical use

Domain Experts and State-of-the-Art Algorithms

- Collaboration with MRI Researchers:
  - Miki Lustig, Ph.D., Berkeley EECS
  - Marc Alley, Ph.D., Stanford EE
  - Shreyas Vasanawala, M.D./Ph.D., Stanford Radiology
- Advanced MRI: Parallel Imaging and Compressed Sensing to dramatically reduce MRI image acquisition time
- Computational IOU: Must solve constrained L1 minimization

\[
\begin{align*}
\text{minimize} & \quad ||Wx||_1 \\
\text{s.t} & \quad F_\Omega x = y, \\
& \quad ||Gx - x||_2 < \epsilon
\end{align*}
\]
**SW architecture of image reconstruction**

- **Pipe and Filter**
  - Data Parallelism / Fourier Transforms

- **Fork-Join**
  - Linear Alg.

- **Data Parallelism / Fourier Transforms**

- **Fork-Join**

- **Data Parallelism / Fourier Transforms**

**Iterative POC5 Algorithm:**
1. Apply SPIRIT Operator:
   \[ x_n \leftarrow \sum g_{ij} \cdot x_j \]
2. Wavelet Soft-Thresholding:
   \[ x \leftarrow W S_{\lambda} \{ W^* x \} \]
3. Fourier-space projection:
   \[ x \leftarrow F(G^T y + F_t D_y F^T x) \]

**Game-Changing Speedup**

- 100X faster reconstruction
- Higher-quality, faster MRI
- This image: 8 month-old patient with cancerous mass in liver
  - 256 x 84 x 154 x 8 data size
  - Serial Recon: 1 hour
  - Parallel Recon: 1 minute
- Fast enough for clinical use
  - Software currently deployed at Lucile Packard Children's Hospital for clinical study of the reconstruction technique

This image: 8 month-old patient with cancerous mass in liver.
Applications

- Our formula in a bit of detail: MRI reconstruction
- Other applications where our formula has proven itself
- Applications in progress

Support-Vector Machine Mini-Framework

- Algorithmic changes and parallel implementation lead to performance speedup: Core 2 Duo versus G80

<table>
<thead>
<tr>
<th>Computation</th>
<th>LIBSVM</th>
<th>Our algorithm 2-core Parallel CPU</th>
<th>Our algorithm, 16-core Parallel GPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM Training (geo-mean)</td>
<td>771.8 s</td>
<td>---</td>
<td>38.5 s</td>
</tr>
<tr>
<td>SVM Classification</td>
<td>41.42 s</td>
<td>4.21 s</td>
<td>0.38 s</td>
</tr>
</tbody>
</table>

100X speed-up
793 downloads since release in 10/2008

Fast support vector machine training and classification, Catanzaro, Sundaram, Keutzer, International Conference on Machine Learning 2008
**Image Contour Detection**

- **Mini-Framework**
- **Original Serial Code (Nehalem)**: C + Pthreads, our algorithms 8 threads, 2 sockets (Barcelona)
- **Our algorithms, GTX280**
  - 222 seconds
  - 29.79 seconds
  - 1.8 seconds

  **130X speed-up**
  **490 downloads since release in 6/2009**

“Efficient, High-Quality Image Contour Detection” Catanzaro, Su, Sundaram, Lee, Murphy, Keutzer International Conference on Computer Vision, 2009

---

**Speech Recognition Framework**

- **Input**: Speech audio waveform
- **Output**: Recognized word sequences

  - Achieved 11x speedup over sequential version
  - Allows 3.5x faster than real time recognition
  - Our technique is being deployed in a hotline call-center data analytics company
  - Used to search content, track service quality and provide early detection of service issues

Scalable HMM based Inference Engine in Large Vocabulary Continuous Speech Recognition, Kisu You, Jike Chong, Youngmin Yi, Ekaterina Gonina, Christopher Hughes, Wonyong Sung and Kurt Keutzer, IEEE Signal Processing Magazine, March 2010
- **Value-at-Risk Computation with Monte Carlo Method**
  - Summarizes a portfolio’s vulnerabilities to market movements
  - Important to algorithmic trading, derivative usage and highly leveraged hedge funds
  - Improved implementation to run 60x faster on a parallel microprocessor

Four Steps of Monte Carlo Method in Finance

- Uniform Random Number Generation
- Market Parameter Transformation
- Instrument Pricing
- Data Assimilation


- Our formula in a bit of detail: MRI reconstruction
- Other applications where our formula has proven itself
- Applications in progress
Poselet Human Detection

- Can locate humans in images
- **20x speedup** through algorithmic improvements and parallel implementation
- Work can be extended to **pose estimation** for controller-free video game interfaces using ordinary web cameras

Option Pricing Application

- Price an option - a tradable financial security whose value depends on the value of an underlying asset and market parameters.
- Black-Scholes equation:

\[
\frac{\partial V}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS \frac{\partial V}{\partial S} - rV = 0.
\]

**Speedup:**

- **6x** pricing 1 option
- **25x** pricing 128 options on Larrabee
Optical Flow

- Optical Flow involves computing the motion vectors ("flow field") between the consecutive frames of a video.
- Involves solving a non-linear optimization problem:
  \[ J(w) = \int \nabla I(x + w) \cdot \nabla I(x) \, dx + \gamma \int |\nabla I(x + w) - \nabla I(x)|^2 \, dx + \alpha \int |\nabla v|^2 + |\nabla f|^2 \, dx \]

Speedup

- 32x linear solver
- 7x overall

![Parallel vs. Serial Pie Chart](chart.png)
Parallelism Delivers Moore’s Law (and More) in the Future

Transistors
Serial Performance
Parallel Performance

Tip of the Iceberg

- Today we’re only exploiting the smallest tip of the iceberg of computation that will be available in the future
Limits of our formula

- State-of-the-art algorithms
- Tall-skinny parallel programmers
- Domain experts
- Architecting software with patterns
- Compelling applications
- Parallel hardware

Game changing speed-ups (with scalability)

Our research formula

- Tall skinny programmers
- Application frameworks
- Domain experts use frameworks
- Compelling applications
- Parallel hardware

Game-changing speed-ups
How do we deliver all those new applications to the user?

- I’ve talked about how to translate Moore’s Law into useful applications via microprocessors.
- Our next speakers will talk about how to deliver those applications to you.

- The Future of Mobile
  - Eric Brewer

- The Future of the Cloud
  - Michael Franklin
SW architecture of reconstruction

Pipe and Filter
- Data Parallelism / Fourier Transforms
  - Fork-Join
    - Linear Alg.
  - Data Parallelism / Fourier Transforms
    - Fork-Join
      - Data Parallelism / Fourier Transforms

Iterative POCS Algorithm:
1. Apply SPIRiT Operator:
   \[ z_0 \leftarrow \sum_{i} g_{ij} * x_j \]
2. Wavelet Soft-Thresholding
   \[ z \leftarrow W S_A \{ W^* z \} \]
3. Fourier-space projection
   \[ z \leftarrow F (F^T y + P^c F^T P^c F^T z) \]