Innovative Applications and Technology Pivots – A Perfect Storm in Computing

Wen-mei Hwu
Professor and Sanders-AMD Chair, ECE, NCSA
University of Illinois at Urbana-Champaign
with
Izzat El Hajj, Liwen Chang, Simon Garcia, Abdul Dakkak and Carl Pearson

ECE ILLINOIS
Agenda

• Revolutionary paradigm shift in applications
• Post-Dennard technology pivot
• Engineering high-efficiency scalable algorithm libraries for heterogeneous parallel computing
A major paradigm shift

- In the 20th Century, we were able to understand, design, and manufacture what we can measure
  - Physical instruments and computing systems allowed us to see farther, capture more, communicate better, understand natural processes, control artificial processes...
A major paradigm shift

- In the 20th Century, we were able to understand, design, and manufacture what we can measure
  - Physical instruments and computing systems allowed us to see farther, capture more, communicate better, understand natural processes, control artificial processes...

- In the 21st Century, we are able to understand, design, and create what we can compute
  - Computational models are allowing us to see even farther, going back and forth in time, learn better, test hypothesis that cannot be verified any other way, create safe artificial processes...
### Examples of Paradigm Shift

#### 20th Century
- Small mask patterns
- Electronic microscope and Crystallography with computational image processing
- Anatomic imaging with computational image processing
- Teleconference
- GPS

#### 21st Century
- Optical proximity correction
- Computational microscope with initial conditions from Crystallography
- Metabolic imaging sees disease before visible anatomic change
- Tele-emersion
- Self-driving cars
What is powering the paradigm shift?

• Large clusters (scale out) allow solving realistic problems
  • 1.5 Peta bytes of DRAM in Blue Waters
  • E.g., 0.5 Å (0.05 nm) grid spacing is needed for accurate molecular dynamics
    • interesting biological systems have dimensions of mm or larger
    • Thousands of nodes are required to hold and update all the grid points.

• Fast nodes (scale up) allow solution at realistic time scales
  • Simulation time steps at femtosecond (10^{-15} second) level needed for accuracy
    • Biological processes take milliseconds or longer
    • Current molecular dynamics simulations progress at about one day for each 100 microseconds of the simulated process.
    • Interesting computational experiments take weeks (used to be months)
Blue Waters Computing System
Operational at Illinois since 3/2013

49,504 CPUs -- 4,224 GPUs

12.5 PF
1.6 PB DRAM
$250M

Sonexion:
26 PBs
>1 TB/sec

Spectra Logic:
300 PBs
120+ Gb/sec

WAN

10/40/100 Gb Ethernet Switch
100 GB/sec

IB Switch
>1 TB/sec

12.5 PF
1.6 PB DRAM
$250M
What types of applications are demanding computing power today?

• First-principle-based models
  • Problems that we know how to solve accurately but chose not to because it would be “too expensive”
  • High-valued applications with approximations that cause inaccuracies and lost opportunities
  • Medicate imaging, earthquake modeling, weather modeling, astrophysics modeling, precision digital manufacturing, combustion modeling, ....

• Applications that we have failed to program
  • Problems that we just don’t know how to solve
  • High-valued applications with no effective computational methods
  • Computer vision, speech processing, stock trading, decision making
Some different modalities of Real-world Data

Images/video

- Image
  - Vision features
  - Detection

Audio

- Audio
  - Audio features
  - Speaker ID

Text

- Text
  - Text features
  - Text classification, machine translation, information retrieval, ....

Slide courtesy of Andrew Ng, Stanford University
We know what we want but don’t know how to get there.

Slide courtesy of Steve Oberlin, NVIDIA
Some different modalities of Real-world Data

This seems to be a combinational logic design problem.
## Combinations Logic Specification – Truth Table

<table>
<thead>
<tr>
<th>Input</th>
<th>output</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
What if we did not know the truth table?

• Look at enough observation data to construct the rules
  
  • 000 → 0
  • 011 → 0
  • 100 → 1
  • 110 → 0

• If we have enough observational data to cover all input patterns, we can construct the truth table and derive the logic!
LeNet-5, a convolutional neural network for hand-written digit recognition.

This is a $1024 \times 8$ bit input, which will have a truth table of $2^{8196}$ entries.
Forward Propagation Path of a Convolution Layer
Back-Propagation of $\partial E/\partial W$

$\partial E/\partial X = W^T \cdot \partial E/\partial Y$ and $\partial E/\partial W = \partial E/\partial Y \cdot X^T$
Behind the Scenes

• In 2010 Prof. Andreas Moshovos adopted the Illinois ECE498AL Programming Massively Parallel Programming Class

• Several of Prof. Geoffrey Hinton’s graduate students took the course

• These students developed the GPU implementation of the DNN that was trained with 1.2M images to win the ImageNet competition
IBM Watson Q&A Pipeline - 2012 Jeopardy! running on a 2,880 node cluster
A long way to go towards cognitive computing

Social Sciences
Use the cartoon to answer the next TWO questions.

24 What economic condition motivated Mr. Smith to request a raise?
A. Inflation
B. Specialization
C. Unemployment
D. Embargo

Taken from: http://www.ole.state

25 Without incentives, which would likely influence Mr. Smith in the marketplace?
A. He will increase his interest for higher-priced items.
B. He will increase his demand for higher-priced items.
C. He will decrease his demand for lower-priced substitutes.
D. He will decrease his demand for lower-priced substitutes.

Human Instructions

Image Recognition

Diagram Understanding

Knowledge Indexing

IR

Knowledge Inferencing

Natural Language Processing

Programming Framework

Speech Recognition

Text Extraction

Hardware Platform
How did we end up with GPU computing anyway?
Dennard Scaling of MOS Devices

- In this ideal scaling, as $L \rightarrow \alpha * L$
  - $V_{DD} \rightarrow \alpha * V_{DD}$, $C \rightarrow \alpha * C$, $i \rightarrow \alpha * i$
  - Delay = $CV_{DD}/i$ scales by $\alpha$, so $f \rightarrow 1/\alpha$
  - Power for each transistor is $CV^2 * f$ and scales by $\alpha^2$
    - keeping total power constant for same chip area

JSSC Oct 1974, page 256

$\alpha$ has been 1.44 every 18 months
Frequency Scaled Too Fast 1993-2003
Total Processor Power Increased
(super-scaling of frequency and chip size)
Post-Dennard Pivoting

- Multiple cores with more moderate clock frequencies
- Heavy use of vector execution
- Employ both latency-oriented and throughput-oriented cores
### Production Use Results CPU+GPU vs. CPU+CPU

<table>
<thead>
<tr>
<th>Application</th>
<th>Description</th>
<th>Application Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>NAMD</td>
<td>100 million atom benchmark with Langevin dynamics and PME once every 4 steps, from launch to finish, all I/O included</td>
<td>1.8</td>
</tr>
<tr>
<td>Chroma</td>
<td>Lattice QCD parameters: grid size of 483 x 512 running at the physical values of the quark masses</td>
<td>2.4</td>
</tr>
<tr>
<td>QMCPACK</td>
<td>Full run Graphite 4x4x1 (256 electrons), QMC followed by VMC</td>
<td>2.7</td>
</tr>
<tr>
<td>ChaNGa</td>
<td>Collisionless N-body stellar dynamics with multipole expansion and hydrodynamics</td>
<td>2.1</td>
</tr>
<tr>
<td>AWP</td>
<td>Earthquake anelastic wave propagation with staggered-grid finite-difference and realistic plastic yielding</td>
<td>3.7-5.0</td>
</tr>
</tbody>
</table>
More Heterogeneity Is Coming

- Beyond traditional CPUs and GPUs
  - FPGAs (e.g., Microsoft FPGA cloud)
  - ASICs (e.g., Google’s TPU)

- Beyond traditional DRAM
  - Stacked DRAM for more memory bandwidth
  - Non-volatile RAM for memory capacity
  - Near/in memory computing for reduced power used in data movement
Engineering high-efficiency software for heterogeneous computing
Performance-Portability: One Source for All

Challenges
- Granularity of Parallelism
- Levels of Hierarchy
- Memory Characteristics
- Resource Sizes
- Micro-architecture

Solutions
- Coarsening
- Codelet Composition
- Automatic Data Placement
- Autotuning
- Algorithmic Choice
Hierarchical Compute Organization of Devices

1. Grid
2. Block
3. Warp
4. Thread
5. Instruction-level Parallelism

---

tile = (len + blockDim.x - 1)/gridDim.x;
sub_tile = (tile + blockDim.x - 1)/blockDim.x;
accum = 0
#pragma unroll
for(unsigned i = 0; i < sub_tile; ++i) {
    accum += in[blockIdx.x*tile + i*blockDim.x + threadIdx.x];
}
tmp[threadIdx.x] = accum;
__syncthreads();
for(unsigned s=1; s<blockDim.x; s *= 2) {
    if(id >= s)
        tmp[threadIdx.x] +=
            tmp[threadIdx.x - s];
    __syncthreads();
}
partial[blockIdx.x] = tmp[blockDim.x-1];
return; // Launch new kernel to sum up partial
Tangram: Codelet-based Programming Model

(a) Atomic autonomous codelet

```cpp
__codelet
int sum(const Array<1,int> in) {
    unsigned len = in.size();
    int accum = 0;
    for(unsigned i=0; i < len; ++i) {
        accum += in[i];
    }
    return accum;
}
```

(b) Atomic cooperative codelet

```cpp
__codelet __coop __tag(kog)
int sum(const Array<1,int> in) {
    __shared int tmp[coopDim()];
    unsigned len = in.size();
    unsigned id = coopIdx();
    tmp[id] = (id < len)? in[id] : 0;
    for(unsigned s=1; s<coopDim(); s *= 2) {
        if(id >= s)
            tmp[id] += tmp[id - s];
    }
    return tmp[coopDim()-1];
}
```

(c) Compound codelet using adjacent tiling

```cpp
__codelet __tag(asso_tiled)
int sum(const Array<1,int> in) {
    __tunable unsigned p;
    unsigned len = in.size();
    unsigned tile = (len+p-1)/p;
    return sum( map( sum, partition(in,
                    p,sequence(0,tile,len),sequence(1),sequence(tile,tile,len+1))));
}
```

(d) Compound codelet using strided tiling

```cpp
__codelet __tag(stride_tiled)
int sum(const Array<1,int> in) {
    __tunable unsigned p;
    unsigned len = in.size();
    unsigned tile = (len+p-1)/p;
    return sum( map( sum, partition(in,
                    p,sequence(0,1,p),sequence(p),sequence((p-1)*tile,1,len+1))));
}
```
Tangram: Composition Example
What is the stake?

• Scalable and portable software can empower many hardware generations

Scalable algorithms and libraries could be the best legacy we can leave behind from this era
Thank you!

Any more questions?