Towards and Hot-topics in Parallel and Scientific Computing

Carlos Jaime Barrios Hernandez, PhD.
Bucaramanga, Colombia
Why Parallel and Scalable Computing (Distributed Computing)?

- Parallel and Scalable Hardware are not all.
  - Big and expensive is not always better.

- Parallel Software solutions are not all.
  - Depending of the complexity of the problem
  - Depending of the Accuracy of the Algorithms.
  - However Portability of many applications… often parallel computing software is not portable.

- Parallel and Scalable Computing change computer science
  - Society Influence
  - Paradigms Rupture
Plan

- What about Programming Tendencies?
  - Architecture and Programming Models
- What about Infrastructure Costs?
  - Green Computing
- What about Great Scale Platforms?
  - Large Scale Platforms
  - Cloud
- Final Comments
Programming Tendencies

- How exploit better infrastructure?
  - Low cost
  - Easy to programming
- How exploit better minds?
  - Easy to design
  - Easy to “traduce” models to programs
Parallel Programming and Computation Thinking

- PP Goals (Reminder)
  - Solve a problem in less time.
  - Solve bigger problems in a determined time.
  - Achieve better solutions for problems in a determined time.
- Good Candidates for Parallel Computing (Reminder):
  - Large Problem Sizes (Large Scale Problems)
  - High Complexity
  - High Modeling

Programmers “build code” and “organize data” to solve subproblems concurrently.

Computational Thinking thought process of formulating domain problems in terms of computation steeps and algorithms.
Architectural Considerations to PP

- Remember “concurrency”: it exploits better the resources (shared) within a computer.
- Exploit SIMD and MIMD Architectures
Architectural Considerations to PP

Shared Memory

- SMP
- NUMA

Distributed (Shared) Systems

MPP / Clusters
**PP Models**

- Shared Memory
  - OpenMP

- Distributed Memory (Shared)
  - MPI

- Hybrid Combination
  - OpenMP + MPI
  - CUDA, OpenCL, JAVA
Evolution of Configurable Architecture

- Dual Cores
  (Symmetric Multithreading)
- MultiCore Arrays
- Scalar + Many Cores
  (Highly threaded workloads)
- Manycore arrays

Large Scale Cores
(High Single Thread Performance)
GPUs vs CPUs Performance

GPUs vs CPUs
Introducing GPUs Computing

- GPUs are massively parallel numeric computing processors programmed in C with extensions

- GPUs allow:
  - Massive Multithreading
  - Small Cache Memory manage (Relative)
  - Bandwidth-Centric Memory interface Design

- GPUs were exploitable by APIs (80s-90s):
  - DirectX™ (Direct3D™)
  - OpenGL
Nvidia™ GeForce™ 8080 Pipeline

AMD™ ‘s RADEON™ HD2900XT Pipeline

From www.amd.com
TESLA™ Graphics and Computing Architecture

TESLA™ Graphics and Computing Architecture Features

- TESLA™ shader processors are fully programmable
  - Large instructions memory
  - Cache Instructions
  - Logic Sequence Instructions
- TESLA™ to non-graphics programs:
  - Hierarchical Parallel Threads
  - Barrier Synchronization
  - Atomic Operators (Manage Highly Parallel Computing Work)
CUDA™ Program Structure

- Host (CPU)
  - ANSI C Code
- Devices (MPP)
  - ANSI C + Data Parallel Functions (Kernels)

CUDA™ Execution

• Steps
  1. Host Execution
  2. Kernel Function Invocation or Launch
  3. Execution Moved to Devices (GPU)

• Threads Generated during by a kernel during an invocation are called collectively such a grid
CUDA™ SDK and MCUDA


From [http://upcrc.illinois.edu](http://upcrc.illinois.edu)
CUDA™ Processing Flow

CUDA™ Algorithm Trip

Comparative Example:
Vector Multiplication

In C++

```c
void compute(float A[N][N], float B[N][N], float C[N][N]){
    for (int i=0;i<N;i++)
        for (int j=0;j<N;j++)
            C[i][j] = A[i][j] + B[i][j];
}
int main(){
    compute(A,B,C);
}
```

In (Nvidia) CUDA

```c
__global__ void compute(float A[N][N], float B[N][N], float C[N][N]){
    int i = threadIdx.x;
    int j = threadIdx.y;
    C[i][j] = A[i][j] + B[i][j];
}
int main(){
    //call the GPU-kernel
    compute<<<N_blocks,N_threads>>>(A, B, C);
}
```
Plan

• What about Programming Tendencies?
  • Architecture and Programming Models
• What about Infrastructure Costs?
  • Green Computing
• What about Great Scale Platforms?
  • Large Scale Platforms
  • Cloud
• Final Comments
Green IT

• Using computing resources efficiently, accounting for triple bottom line to measure success
  • Economical
  • Social
  • Environmental

• Why?
  • Environment
  • $$
  • Computing lifestyles
  • Political Correctly
Green IT Challenges
(From little programmer point of view)

- Reduction of MFlop per Watt
- Scheduling Algorithms
  - Reduce Consumption
  - Reduce Energy Impact
  - Permanent Efficiency
Plan

- What about Programming Tendencies?
  - Architecture and Programming Models

- What about Infrastructure Costs?
  - Green Computing

- What about Great Scale Platforms?
  - Large Scale Platforms
  - Cloud

- Final Comments
Grid Computing Architecture and the Middleware

Parallel Applications

Sequential Applications

Parallel Applications

Parallel Programming Environment

Middleware

(Single System Image)

PC/Workstation/Cluster/Devices/Sensors

Communications Software

Network Interface Hardware

PC/Workstation/Cluster/Devices/Sensors

Communications Software

Network Interface Hardware

Interconnection Network

[*]From: http://gridcafe.web.cern.ch
Volunteer Computing

- Volunteer computing is a type of distributed computing in which computer owners donate their computing resources (such as processing power and storage) to one or more "projects".
  - BOINC (Seti@home)
  - Xgrid
  - GridMP
- Associated with P2P
- Can be associated with High Throughput Computing (HTC) or High Performance Computing (HCP)
Cloud Computing

- Internet-based computing, whereby shared resources, software, and information are provided to computers and other devices on demand.
- Cloud computing describes a new supplement, consumption, and delivery model for IT services based on the Internet, and it typically involves over-the-Internet-provision of dynamically scalable and often virtualized resources.
Cloud Computing Features and Issues

- **Agility**: rapidly and inexpensively re-provision technological infrastructure resources.
- **Cost**
- **Device and location independence**
- **Multi-tenancy**
  - Centralization of infrastructure in locations with lower costs
  - Peak-load capacity increases
  - Utilization and efficiency improvements for systems.
- **Reliability**: improved if multiple redundant sites are used
- **Scalability via dynamic ("on-demand")**:  
  - Resources on a fine-grained, self-service basis near real-time.
  - Performance is monitored, and consistent and loosely coupled architectures are constructed using web services as the system interface.
- **Security and Safety**
- **Maintenance**: Applications are easier to maintain
- **Metering** cloud computing resources usage should be measurable and should be metered.
- **Privacy**
- **Compliance (Data Protection Directives)**
- **Legal (Trademarks, Licenses)**
- **Open Source**
- **Security**
- **Safety**
- **Sustainability (cloud computing is often assumed to be a form of "green computing")**
Cloud Computing Architecture

- Cloud computing, typically involves multiple cloud components communicating with each other over application programming interfaces, usually web services.
  - Platforms as a service
  - Infrastructure as a service
Cloud Computing Deployment Types

- Private Cloud
- Public Cloud
  - Resources are dynamically provisioned on a fine-grained, self-service basis over the Internet, via web applications/web services
- Community Cloud
  - Established where several organizations have similar requirements and seek to share infrastructure
- Hybrid Cloud
- InterCloud
  - Cloud of Clouds
Final Comments

- Guidelines to Parallel and Distributed Programming (should) require identifying patterns
  - Finding Concurrency
  - Algorithm Structure
  - Supporting Structures
  - Implementation Mechanisms
    - Supported Platforms and Infrastructures

- *Problems need minds, minds need applications, applications need platforms, platforms need infrastructure.*
And what about Extreme Computing?
Extreme Computing

- Really HARD-ware and correct SOFT-ware
- Optimal Orchestration of Services
- User needs and Potentialities
  - Flexibility
  - Portability
  - Efficiency
  - Simplicity
- Interact with different computing paradigms
“System Engineering is more for Artists than Formalists”
“Knowledge does not belong to any one” (H.A.)

“Always a Good Scientific is questioning their own beliefs and theories”
“The question is more important than the answer” (M.C.)

“The Problem with Computer Science Research is that the research work Can not occurs only for curiosity, research must be an utility… and it must be sold.” (M.R.)