Parallel programming issues, achievements and trends in high-performance and adaptive computing

Marco Danelutto, Marco Vanneschi
Dept. Computer Science
Univ. of Pisa


X-HPC : Experimenting with XtreemOS to provide HPC services
Pisa, 16 Jun 2010, Sala Gerace, Dept. Comp. Science
(R)evolution in parallel computing

- unprecedented change in parallel computing perspective
  - hw factors
  - sw factors
The “hw”

- from single core
  - pipeline, superscalar, SIMD extended
- to multi/many core
  - SMP, network on chip, ...
The “sw”

• from focus on algorithms
  • computing the result
• to focus on non functional aspects
  • affecting the way the result is computed
Hardware

- Currently
  - dual/quad core in mainstream
  - 48-64 core in niche/research areas
  - heterogeneous (Cell, WireSpeed)
  - co-processor (gp(?)GPUs)
- Expected: moore law (#cores)
Hardware ➔ Software

- parallelizing compilers do not target O(100) cores
- Parallel code needed from the very beginning
  - design, compile, RTS, optimize
  - care of "shared memory" implementations
Software

• “ab initio” parallel sw
• structured
  • algorithmic skeletons (since ‘90s)
  • design patterns (early ’00)
  • Berkeley vision + Intel

➤ structure simplifies problems
Software
(last but not least)

- non functional concerns

- security, fault tolerance, power consumption, performance, ...

- cross cutting concerns usually in charge of the application programmer

→ must be moved (solved) to system sw (separation of concerns)
Group experience

P3L (1991)
SkIE (1996)
ASSIST (2000)
Ocaml P3L (1996)
SkeLib (2000)
Lithium (2001)
Muskel (2002)
GCM (2004)
Autonomic ASSIST (2005)
ASSISTANT (2008)
BS (2007)
Libero (2010)
Group experience

P3L (1991)
SkIE (1996)
ASSIST (2000)
Ocaml P3L (1996)
SkeLib (2000)
Lithium (2001)
Muskel (2002)
GCM (2004)
Autonomic ASSIST (2005)
BS (2007)
ASSISTANT (2008)
Libero (2010)
Group experience

P3L (1991)
SkIE (1996)
ASSIST (2000)
Ocaml P3L (1996)
SkeLib (2000)
Lithium (2001)
GCM (2004)
Muskel (2002)
Autonomic ASSIST (2005)
BS (2007)
ASSISTANT (2008)
Libero (2010)
Group experience

P3L (1991)


First NFC manager

First adaptive component

Autonomic ASSIST (2005)

Muskel (2002)

GCM (2004)

BS (2007)

ASSISTANT (2008)

Libero (2010)
Group experience

P3L (1991)

SkIE (1996)

ASSIST (2000)

Ocaml P3L (1996)

ASSISTANT (2008)

SkeLib (2000)

Lithium (2001)

Muskel (2002)

GCM (2004)

BS (2007)

Libero (2010)

First NFC manager

First adaptive component

Hierarchical NFC management
Group experience

P3L (1991)


First NFC manager

First adaptive component

Multiple layer management

Hierarchical NFC management


Group experience

P3L (1991)


First NFC manager

Libero (2010)

First adaptive component


Multiple layer management

Hierarchical NFC management


MultiConcern NFC management


Hierarchical NFC management
Focus (this talk)

- adaptive, structured, parallel programming environments

1. ASSIST ➔ ASSISTANT
2. CoreGRID/GridCOMP GCM BS
In both cases

- Control theory
- drives adaptation loop
- reactive
  - monitor $\rightarrow$ analyse $\rightarrow$ operate
- proactive
- learning + reactive
Insyeme programming model: ASSISTANT

Computational view of information flows: parallel and distributed composition of application modules

Managers:
- adaptivity, context-awareness
- dynamic reconfiguration and deployment
- QoS control

Context interfaces: devices, nodes and networks monitoring
ASSISTANT

• **Events** related to:
  • Network & monitoring (availability, connectivity, bandwidth, performance, load, ...)
  • Energy consumption
  • QoS (response/completion time, precision, client load balancing, priority, ...)

• **Adaptivity** actions:
  • Increase QoS of current config
  • Switch to different config (algorithm, parallel pattern, ...)

In Supremë Dignitatis 1343
Experiment: flood forecast

RED: Information flows
- Generator Module
  - Functional Logic
  - Control Logic
- Flood Forecasting Module
  - Functional Logic
  - Control Logic
- Client Module
  - Functional Logic
  - Control Logic

BLUE: Manager
- Context interface
- Network monitoring
- Performance profiling

GREEN: Context
- Context interface
Central Server

Communication network a

Interconnection node

Communication network b

PDAs

Integrated Systems for Emergency - FIRB Project
**Integrated Systems for Emergency - FIRB Project**

Communication network a

Central Server

Interconnection node

Communication network b

Data streaming control and network interfaces

Forecasting model: version 0

Client - visualization

PDAs
Integrated Systems for Emergency - FIRB Project

Communication network a

Communication network b

Interconnection node

Forecasting model: version 1

data streaming control and network interfaces

Central Server

PDAs

Context events

Forecasting model: version 0

Client - visualization
Integrating Systems for Emergency - FIRB Project

Communication network a

Context events

Forecasting model: version 1

Interconnection node

Central Server

Communication network b

data streaming control and network interfaces

Client - visualization

Forecasting model: version 0

Context events

Forecasting model: version 2

PDAs
Communication network a

Context events

Forecasting model: version 1

Interconnection node

Data streaming control and network interfaces

Dynamic transitions between alternative versions: RECONFIGURATION and DEPLOYMENT

Context events

Forecasting model: version 0

Central Server

PDAs

Forecasting model: version 2

Client - visualization

Integrated Systems for Emergency - FIRB Project
Results

The graph shows the service time (sec) in relation to the parallelism degree for different configurations:

- **Cluster 8M**
- **Cell 8M**
- **Cluster 32M**
- **Cell 32M**

Key points:

- At a parallelism degree of 0, the service time for Cluster 8M is approximately 24.4 sec.
- At a parallelism degree of 8, the service time for Cell 8M is approximately 3.1 sec.
- At a parallelism degree of 10, the service time for Cell 32M is approximately 0.98 sec.
Behavioural skeletons

- co-design of
- structured parallelism exploitation pattern
- rule based autonomic manager of NFC
BS: the concept

Provides monitoring & actuator mechanisms

Control loop: {pre-condition ➔ action}

Efficiently implements well known parallelism exploitation pattern

Autonomic Manager

Autonomic Controller

Parallel component
BS: the concept

Contract interface

Black box component with functional & non functional ports/interfaces
BS: structure exploit

- Hierarchical management
  - after hierarchical composition
- manager modes
  - active (ensure local contract)
  - passive (report violations to upper level manager)
BS: results

- medical image processing
- adapts parallelism degree to match user SLA (throughput)
- medical image processing
- adapts parallelism degree to match user SLA (throughput)
BS: ongoing results

- Co-management of NFC (security + performance)
BS: ongoing results

- Co-management of NFC (security + performance)
Heterogeneous BS

- Adaptivity so far
  - performance, security, fault tolerance, power, ...

- Already assessed
  - skeletons offloaded to M2C & GPUs

- Now: adaptivity to tune GPU exploitation
Skeleton(s) on GPU

- Prototype skeleton framework
- Parameter to use CPU or GPU
- Dual Core + GTX285
  - Simple op: >32K items GPU speedup 4.5
  - Complex op: Taylor polynome >4K items GPU speedup 6.5
- More complex: 400x400 Mandelbrot GPU speedup 18.8
Lessons: the OS impact

- portable
  - big clusters to single PDA
- lightweight
  - predictable impact on performance
- unmediated monitoring & actuator mechanisms
Open problems and perspectives

- Theory of control
  - much more result applicability to be investigated
- Reactive/proactive adaptation
  - learning
- Multi concern management
  - efficient co-management policies
Related projects & cooperations

- GRID.it
- SFIDA
- InSYeME
- CoreGRID
- GridCOMP

- Roma Sapienza
- Milano Bicocca
- ASI
- Firenze Ingegneria
- Pisa Ingegneria
- Queen’s Univ. Belfast
- INRIA
- Univ. of Muenster
- Univ. Politecnica de Cataluna
The vision

- System knowledge
- Optimization techniques
- Control theory
- Learning
- NFC management (adaptive)
- High-level programming abstraction

Domain specific expertise
- "Abstract" program

Tools
- IDE, compiler, etc.
- Efficient / portable / executable
• thank you for your attention

• any questions?
  • marco.danelutto@di.unipi.it