CHIST-ERA Projects Seminar
Topic Heterogeneous Distributed Computing

Projects HPDCJ, DIVIDEND and DIONASYS

Paris, April 12th, 2018
Heterogeneity everywhere

Distributed deployments
- IoT/Cloud, edge/core
- Mobile vs fixed
- Specific vs mainstream

Hardware
- Memory types
- CPU vs GPU
- WAN vs LAN
- Dedicated vs shared
- Storage classes

Programming environments and languages
- Dynamic vs close to metal
- Specific vs mainstream
- Global properties vs local actions
- Mobile vs fixed
Heterogeneity in the data center

Code doesn’t fit on one machine

All different

Even different interconnect
Heterogeneity in high-performance computing

- **Different hardware**
  - Multicores vs GPU vs FPGAs

- **Complexities for programming**
  - Many programming models

- **Discrepancies**
  - Between engineers’ common skills and available toolsets
  - Between HPC tools and Big Data tools
Heterogeneity in large-scale distributed systems

- Failure models and availability guarantees
  - Mobile vs fixed
  - Shared vs dedicated
  - Single-purpose vs generic

- Different APIs, different programming models & runtimes

- Discrepancies in non-functional properties
  - QoS, resilience, ...
## Call’s expected outcome

<table>
<thead>
<tr>
<th></th>
<th>HPDCJ</th>
<th>DIVIDEND</th>
<th>DIONASYS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Programming Model</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Dependability</strong></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Data Management</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Optimisation Techniques</strong></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td><strong>Versatility</strong></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Distributed Techniques</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Vertical integration
Programming model
Energy accounting
Auto tuning
More heterogeneity
Fast networks

Prototypes Open Sourced
Already saving 22% energy
High Performance Computing

- Parallel distributed computing in Java
  - PCJ library for parallel computing in Java
- Scalability up to 200,000 cores
- CPU and GPGPU
- Fault Tolerance

- Easy for non expert programmers
  - New approach to teach students
Adaptive Overlay Networks
- Systems of systems
- Multi-site Clouds
- Mobile heterogeneous systems

Self-organization
- Robustness

Declarative construction
- Programming models

Co-adaptation of application and network layers
- Intent-driven networking and emergent overlays
## A few realization highlights

<table>
<thead>
<tr>
<th></th>
<th>HPDCJ</th>
<th>DIVIDEND</th>
<th>DIONASYS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Programming Model</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Dependability</strong></td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Data Management</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Optimisation Techniques</strong></td>
<td></td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td><strong>Versatility</strong></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td><strong>Distributed Techniques</strong></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
Realization 1: PCJ library for parallel programming in Java (HPDCJ)

- PCJ scales up to 200,000 cores
- Graph500 implementation in Java with PCJ
- Parallelization of the Genetic Algorithm
  - Scalability up to 1,500 cores
- Parallelization of the sequence alignment (PCJ-blast)
  - Scalability up to 6,144 cores
- Parallelization of spare matrix multiplications
  - Scalability up to 100 cores
- Integration with JCuda (GPU use)
- Streaming library developed
Realization 2: Cross-Stack Energy Savings (DIVIDEND)

- Vertical integration
- Programming model
- Energy accounting
- Auto tuning
- More heterogeneity
- Fast networks

DHSA Prototypes Open Sourced
Already saving 22% energy
Realization 3: Declarative complex overlays with PLEIADES (DIONASYS)

Config file

Many nodes

Initial State

2 rounds

6 rounds: converged
Reflecting on the expected impact

<table>
<thead>
<tr>
<th>Expected impact</th>
<th>Highlights of how the three projects achieved this</th>
</tr>
</thead>
</table>
| **Build a community**                   | Exploitation of results for teaching and training purposes  
                                         | Submission of H2020 projects including industry  
                                         | Standardization efforts                       |
| **Integrate HW and SW**                 | Solutions for co-design of distributed apps., APIs and network interfaces  
                                         | Adaptive network overlays for multi-site clouds |
| **performance, optimisation, (…)**      | Greatly improved scalability for Java HPC programming  
                                         | Easier-to-use solutions for code acceleration in heterogeneous data centers |
| **reliability, fault tolerance (…)**    | Self-organizing solutions for declarative overlay networks  
                                         | Addition of fault-tolerance capabilities to the PCJ library |
| **energy efficiency**                   | Energy monitoring mechanisms across entire data center stacks  
                                         | Emergent overlays for heterogeneous mobile networks |
| **new technologies and services**       | Applications of full-stack optimization to machine learning frameworks  
                                         | Applications of overlay adaptation techniques to microservices-based apps. |
Upcoming challenges and needs

- **HDC call very much on-topic**
  - Heterogeneity issues highlighted in the call getting increasing attention in science and technology communities

- **Distributed systems are (and will continue to get) more and more complex**
  - Need programming models for the ordinary engineer
  - Software engineering research to the rescue?
    - Link with DevOps, microservices, etc.
    - How are startups doing?

- **HPC & Big Data convergence**
  - HPC models for multiprocessors not amenable to heterogeneous computing platforms and accelerators
    - Clear need in Machine Learning!
Possible roadmap

- **In 5 Years we need**
  - Programming models for major domains
  - DSLs to specialize to all devices (CPU, GPGPU, FPGA)
  - Application placement, network topology and performance need to be transparent to applications

- **In 10 Years we need**
  - Distributed languages for the masses
  - Toolchains to co-design platforms and fabricate logic/network/memory blocks for services
Application-level revolutions start with systems’ progress
✓ Machine learning revolution due to GPUs, accelerators, large-scale data centers...
✓ Cloud service model thanks to virtualization
✓ Pervasive computing and IoT due to progress in mobile computing, embedded systems...

2018 call Topic 2. Intelligent Computation for Dynamic Networked Environments
✓ Programming models for edge/fog/ambient computing will lead to novel computations
Questions