Online Reconfiguration of HPC Systems for Improving Energy Performance

Ghislain Landry Tsafack, Laurent Lefevre, Jean-Marc Pierson, Patricia Stolf, Georges Da Costa
ghislain.landry.tsafack.chetsa@ens-lyon.fr

Greendays @ Luxembourg – January, 2013

[1]This work is supported through Hemera

High Performance Computing (HPC) systems design

- place great emphasis of a few components: processor architecture, memory subsystems, storage subsystems and communication subsystems, management framework, platform architecture
  - performance of cpu-intensive workloads depends on processor architecture
- guarantee good performance on average over a wide verity of workloads.
  - can result in power dissipation for some workloads or execution phases of a specific workload
- most of them allow system reconfiguration (at least the processor)

On-line system reconfiguration

complete runtime methodology to efficiently reconfigure HPC systems for energy saving purpose

- phase detection
- phase characterization
- phase identification (program phase identification) / partial phase recognition
  - enable reuse of configuration information for recurring phases
- system reconfiguration

Phase detection and characterization

Execution Vectors (EV) based approach

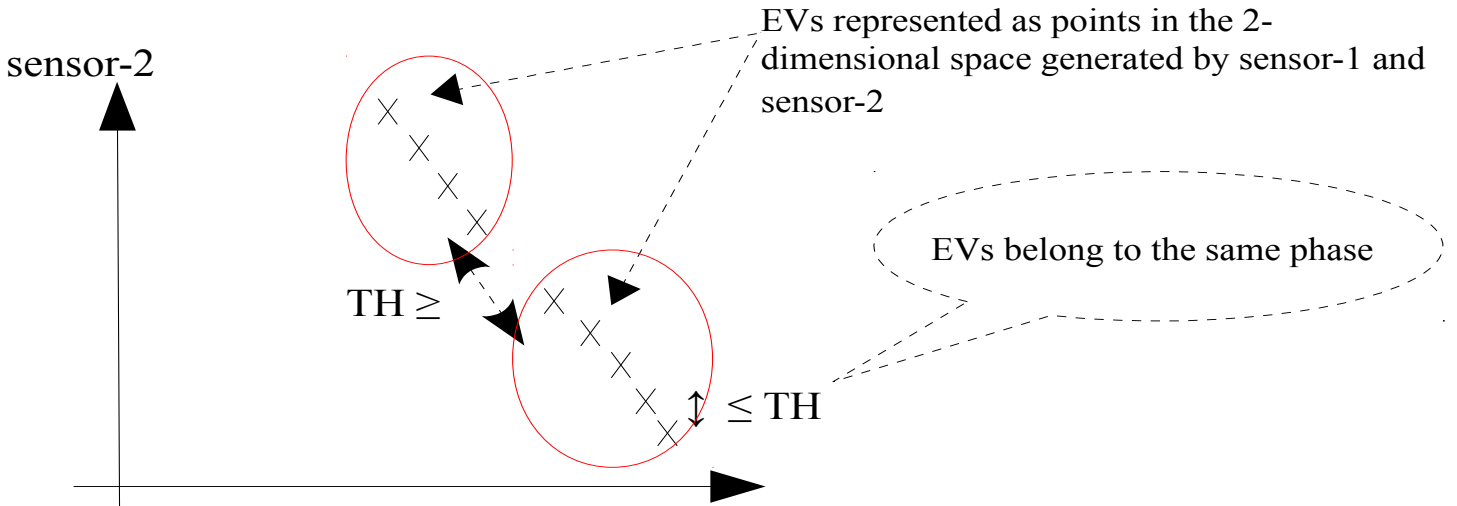
- column vector whose entries are sensors – including hardware performance counters, network bytes sent/received and disk read/write counts
- example

$$\begin{pmatrix} cache\_ref \\ branch\_ins \\ \vdots \\ byteSent \end{pmatrix}$$

Phase tracking and characterization (cont.)

Similarity/resemblance between EVs is used for phase detection

- the manhattan distance between consecutive EVs is the resemblance metric
  - phase changes occur when the distance between consecutive EVs exceeds a threshold (varies throughout the system's life-cycle)



Phase representation and characterization

Represented by reference vector

- closest EV to the centroid of the group of EVs belonging to the phase

Characterization via Principle Component Analysis

- PCA is applied to the data set made up of EVs belonging to the phase
  - select a 5 sensors providing information about the predominant behaviour of the system
    - those contributing less to the first principal axis of PCA are empirically the most appropriate

On-the-fly system adaptation

key idea: reuse of configuration information for recurrent phases/workloads

- rely on partial phase recognition technique (phase identification)
  - identifies an ongoing phase with an existing, before its completion
- use sensors selected from PCA to provide adequate system adaptation (green leverage)
  - processor, network, and disk
  - memory (ongoing)

On-the-fly system adaptation (cont.)

Table: Translation of phase characteristics into system adaptation (IO related sensors include network and disk activities).

Sensors selected from PCA for phase characterization	**Decisions**
cache_references & cache_misses & IO related sensors	CPU frequency set to its maximum spin down the disk network speed scaled down
no IO related sensors	CPU frequency set to its lowest network speed scaled up
instructions & last level cache misses (llc)	CPU frequency set to its minimum network speed scaled up
instructions or llc & IO related sensors	CPU frequency set to its average value network speed scaled down spin down the disk
IO related sensors	CPU frequency set to its maximum spin down the disk network speed scaled down

Experimental platform description

25 node cluster of Intel Xeon X3440 set up on Grid5000

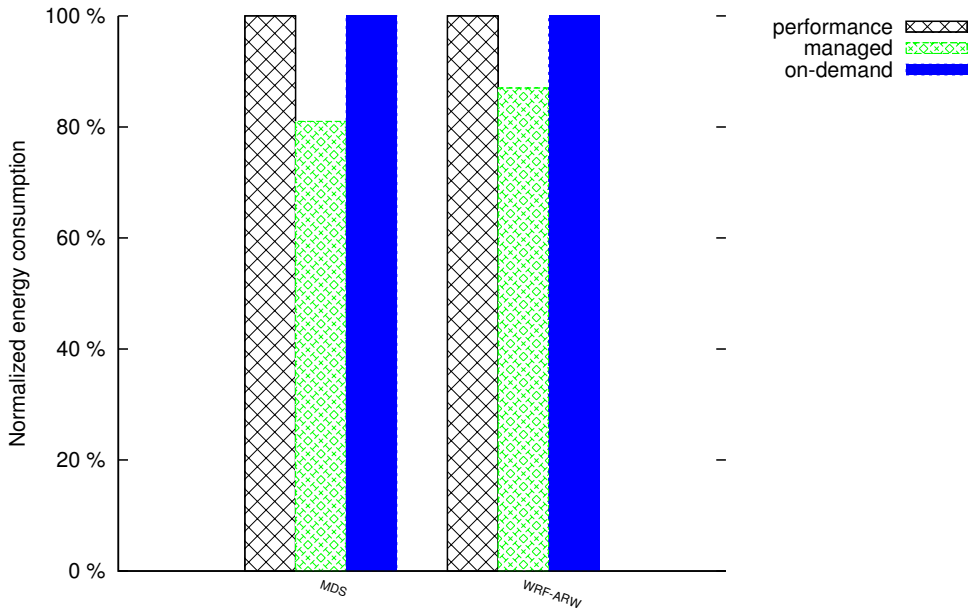
- Linux kernel 2.6.35 runs on each node, where sensors are collected on a per second basis
- high computation level corresponds to 2.53Ghz in CPU frequency, medium and low to 2 GHz and 1.2GHz respectively
- network interconnect speed scaled between 1GB and 10MB
- active and sleep states for the disk

consider two real-life applications (100 processes)

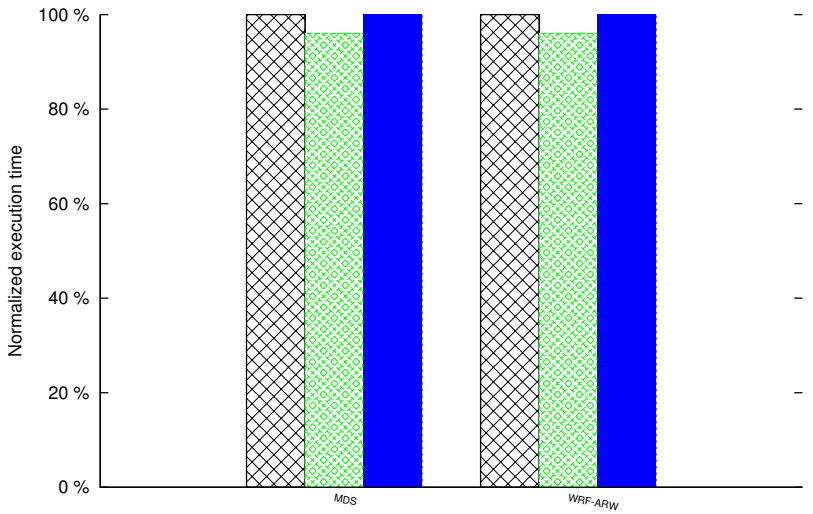
- Advance Research Weather Research Forecasting (WRF-AWR)
- Molecular Dynamics Simulation (MDS)

Results analysis: performance (energy and execution time)

Comparison to Linux on-demand and performance governors



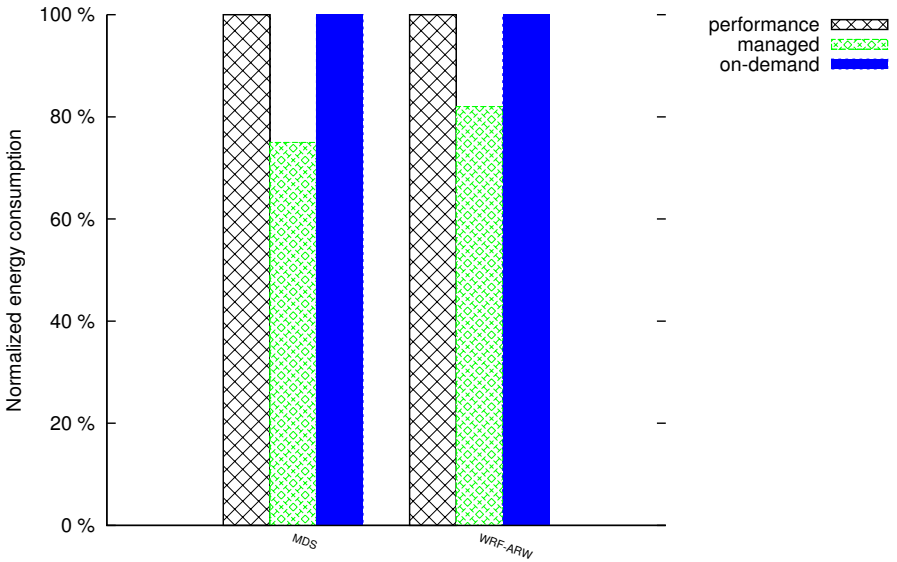
(a) Energy performance



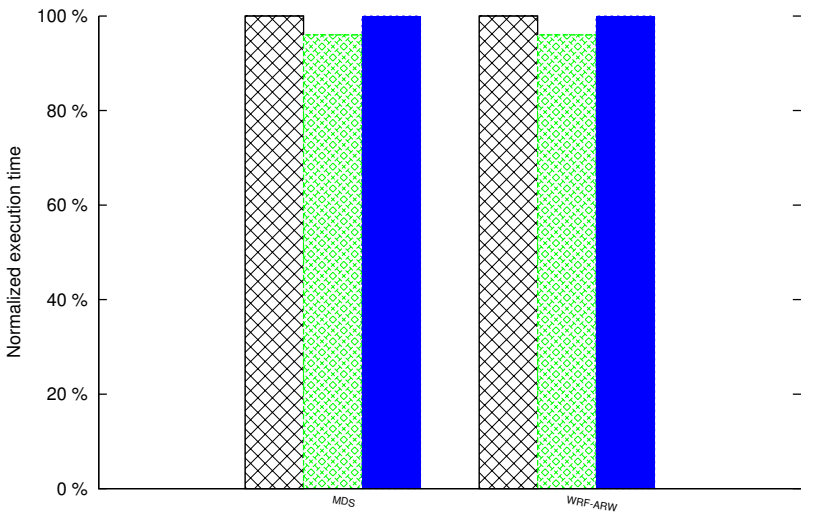
(b) Performance (execution time)

Figure: Phase tracking and partial recognition guided CPU optimization results.

Energy performance: processor, disk and network



(a) Energy performance



(b) Performance (execution time)

Figure: Phase tracking and partial recognition guided processor, disk and network interconnect optimization results: the chart shows average energy consumed by each application under different configurations.

Summary

- demonstrate that we can significantly improve energy performance without any knowledge of applications (up to 24%)
- introduce an on-line general purpose methodology for improving energy performance of HPC systems
  - processor, disk, and network interconnect
  - demonstrates that HPC systems can benefit from more than CPU frequency scaling
- the approach can easily be extended to a large number of energy-aware clusters
  - does not require any specific knowledge of the application
- future directions: more applications, evaluation with multiple applications

Summary

Thank you!

Questions are welcomed!