Beyond CPU Frequency Scaling for a Fine-grained Energy Control of HPC Systems

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

SBAC-PAD 2012 – October 25, 2012

1

[1]This work is supported through Hemera

1. Motivations
   - Today's High Performance Computing (HPC) systems
   - Varying workloads and energy performance
2. Our methodology
   - Phases tracking and characterization
   - On-the-fly system adaptation
3. Evaluation results and analysis
   - Experimental platform description
   - Results analysis: processor's only optimization
   - Results analysis: processor, disk and network optimization
4. Summary

Today's High Performance Computing (HPC) systems

- enable new levels of innovation and insights for organizations that seek out differentiation with excellence
  - raw performance is key to this!
- constantly available
  - increase powering and cooling costs
- have varying workload
  - results in resource over provisioning.
    - processor, memory, storage and network capabilities
  - leads to any kind of system optimization
    - energy performance optimization

Varying workloads and energy performance improvement

- HPC workloads can be roughly divided into compute/memory-intensive and I/O intensive (including network)
  - feature subsystems including the processor, memory, storage (disk) and network
- HPC subsystems are provided with energy saving technologies
  - e.g. DVFS, disk sleeping, etc.

Can we leverage available technologies to improve energy performance of HPC systems potentially shared by multiple workloads/applications without any knowledge of these,?

Overview

HPC applications keep growing in complexity and often share the same infrastructure

- optimizations made for saving energy considering some applications are likely to impact the performance of others

Our approach focuses on the infrastructure instead

- detect and characterize system's runtime behaviours/phases
- partial phase recognition for phase identification
- systems adaptation (storage, memory, interconnect, CPU) accordingly

Phase tracking and characterization

Execution Vectors (EV) based approach

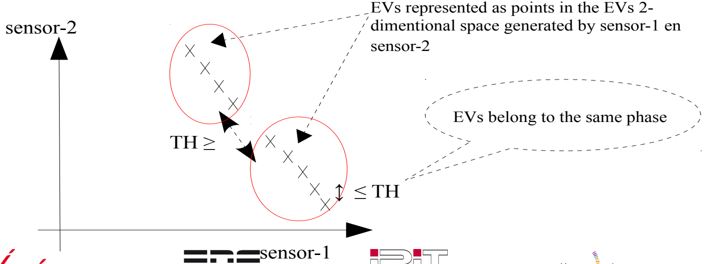
- column vector whose entries are sensors – including hardware performance counters, network bytes sent/received and disk read/write counts
- example

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

Phase tracking and characterization (cont.)

Similarity/resemblance between EVs is used for phase detection

- the manhattan distance between consecutive EVs is the resemblance criterion
  - two EVs belong to the same phase if their distance is below X% of the maximum existing distance between all consecutive EVs; X% is the detection threshold



Phase tracking and characterization: example

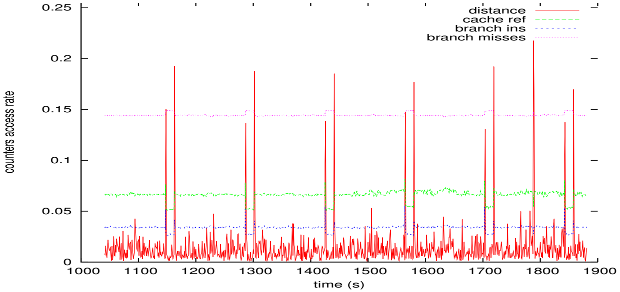


Figure: Phase identification using the similarity between consecutive execution vector as phase identification metric (zoomed-in view of the traces collected on one node when the system was running Molecular Dynamics Simulation) – similarity threshold: 50%, max 0.2.

Phase tracking and characterization (cont.)

Represented by reference vector

- closest EV to the centroid of the group of EV 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

- rely on partial phase recognition technique
  - identifies an ongoing phase with an existing, before its completion
- use sensors selected from PCA to provide adequate system adaptation (green leverage)
  - processor-related adaptation
    - high or cpu-bound; medium or memory-bound; low (non memory/cpu-bound workloads)
  - disk-related adaptation
  - network-related adaptation

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

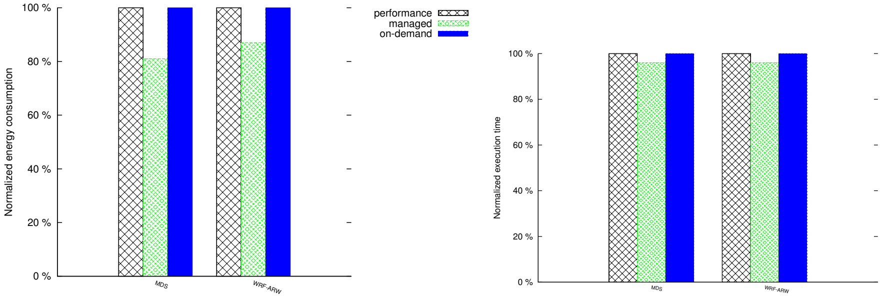
- Linux kernel 2.6.35 runs on each node, where sensors are collected on a per second basis
- high computation level corresponds to 2.53Ghz in CPU frequency, medium and low to 2 GHz and 1.2GHz respectively
- network interconnect speed scaled between 1GB and 10MB
- active and sleep states for the disk

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

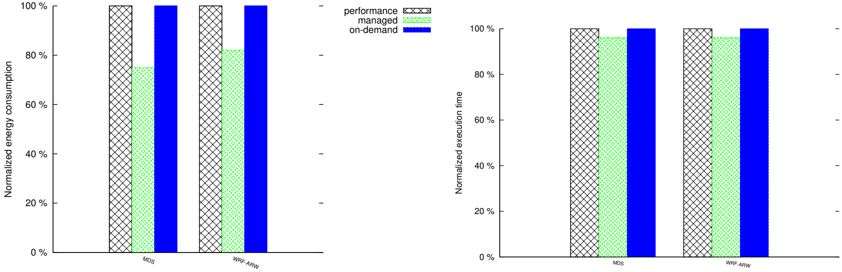


(a) Energy performance

(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