Real Scale Experimentations of SLURM Resource and Job Management System

Yiannis Georgiou
PhD Candidate, BULL R&D Software Engineer

October 4, 2010

Plan

Introduction

- Technological evolutions have introduced extra levels of hiererchies that need to be treated by the resource management system
- The scientific needs and the increasing demands for computing power by applications made users more demanding in terms of robustness and certain quality of services.
- Continuous growth of cluster's sizes and network diameters lead to issues concerning scalability, scheduling efficiency for optimal communication speeds, fault tolerance and energy efficient management.

How can we make sure that a **Resource and Job Management System** will be able to **deal with those challenges**?

Experimental Methodologies for HPC

- The study of HPC systems depends on large number of parameters and conditions.
- Experimentation in HPC makes use of **simulators or emulators** which present advantages (control of experiments, ease of reproduction) but they **fail** to capture all the dynamic, variety and complexity of real life conditions.
- **Real-scale** experimentation is needed in HPC for study and evaluation of all internal functions as **one complete system**.

Workload Modelling

- Performance evaluation by executing a sequence of jobs. This sequence is the actual workload that will be injected to the system.
- Two common ways to use a workload for system evaluation.
  1. Either a *workload log* (trace): record of resource usage data about a stream of parallel and sequential jobs that were submitted and run on a real parallel system,
  2. or a *workload model* (synthetic workload): based on some probability distributions to generate workload data, with the goal of clarifying their underlying principles (ESP benchmark).

Workload Logs

Standard Workload Format

Definition of SWF format to describe the execution of a sequence of jobs.

```
; Computer: Linux cluster (Atlas)
; Installation: High Performance Computing - Lawrence Livermore National Laboratory
; MaxJobs: 60332
; MaxRecords: 60332
; Preemption: No
; UnixStartTime: 1163199901
; TimeZone: 3600
; TimeZoneString: US/Pacific
; StartTime: Fri Nov 10 15:05:01 PST 2006
; EndTime:   Fri Jun 29 14:02:41 PDT 2007
; MaxNodes: 1152
; MaxProcs: 9216
; Note: Scheduler is Slurm (https://computing.llnl.gov/linux/slurm/)
; MaxPartitions: 4
; Partition:  1 pbatch
; Partition:  2 pdebug
; Partition:  3 pbroke
; Partition:  4 moody20
;
;     j|       s|      w|      r|  p|    c|      m|  p|        u| m|s|  u| g|  e|q| p| p| t
;     o|       u|      a|      u|  r|    p|      e|  r|        s| e|t|  i| i|  x| | a| r| h
;     b|       b|      i|      n|  o|    u|      m|  o|        e| m|a|  d| d|  e|n| r| e| i
;      |       m|      t|      t|  c|     |       |  c|        r|  |t|   |  |   |u| t| v| n
;      |       i|       |      i|   |    u|      u|   |         | r|u|   |  |  n|m| i|  | k
;      |       t|       |      m|  a|    s|      s|  r|        e| e|s|   |  |  u| | t| j|
;      |        |       |      e|  l|    e|      e|  e|        s| q| |   |  |  m| | i| o| t
;      |        |       |       |  l|    d|      d|  q|        t|  | |   |  |   | | o| b| i
;      |        |       |       |  o|     |       |   |         |  | |   |  |   | | n|  | m
;      |        |       |       |  c|     |       |   |         |  | |   |  |   | |  |  | e
 2488  4444343      0   8714 1024     -1    -1 1024     -1    -1  0  17  -1 307 -1  1 -1 -1
 2489  4444343      0 103897 1024     -1    -1 1024     -1    -1  1  17  -1 309 -1  1 -1 -1
 2490  4447935      0    634 2336     -1    -1 2336  10800    -1  1   3  -1   5 -1  1 -1 -1
 2491  4448583      0    792 2336     -1    -1 2336  10800    -1  1   3  -1   5 -1  1 -1 -1
 2492  4449388      0    284 2336     -1    -1 2336  10800    -1  0   3  -1   5 -1  1 -1 -1
```

Experimental Methodology - General Principles

- Real-Scale experimentation upon **dedicated platforms**: Control and Reproduction of experiments.
- **Injection** of characterised **workloads** to observe the behaviour of the RJMS under particular conditions.
- Extraction of the produced workload trace and **post-treatment analysis** of the results.

Experimental Methodology - Platforms

Real-Scale experimentation upon dedicated controlled platform:

- **Grid5000 Experimental grid**[1], large-scale distributed platform that can be easily controlled, reconfigured and monitored.
- Specially designed for research in computer science, it provides the ability of **environment deployment** to facilitate the experimentation upon all different layers of the software stack.

[1] https://www.grid5000.fr/

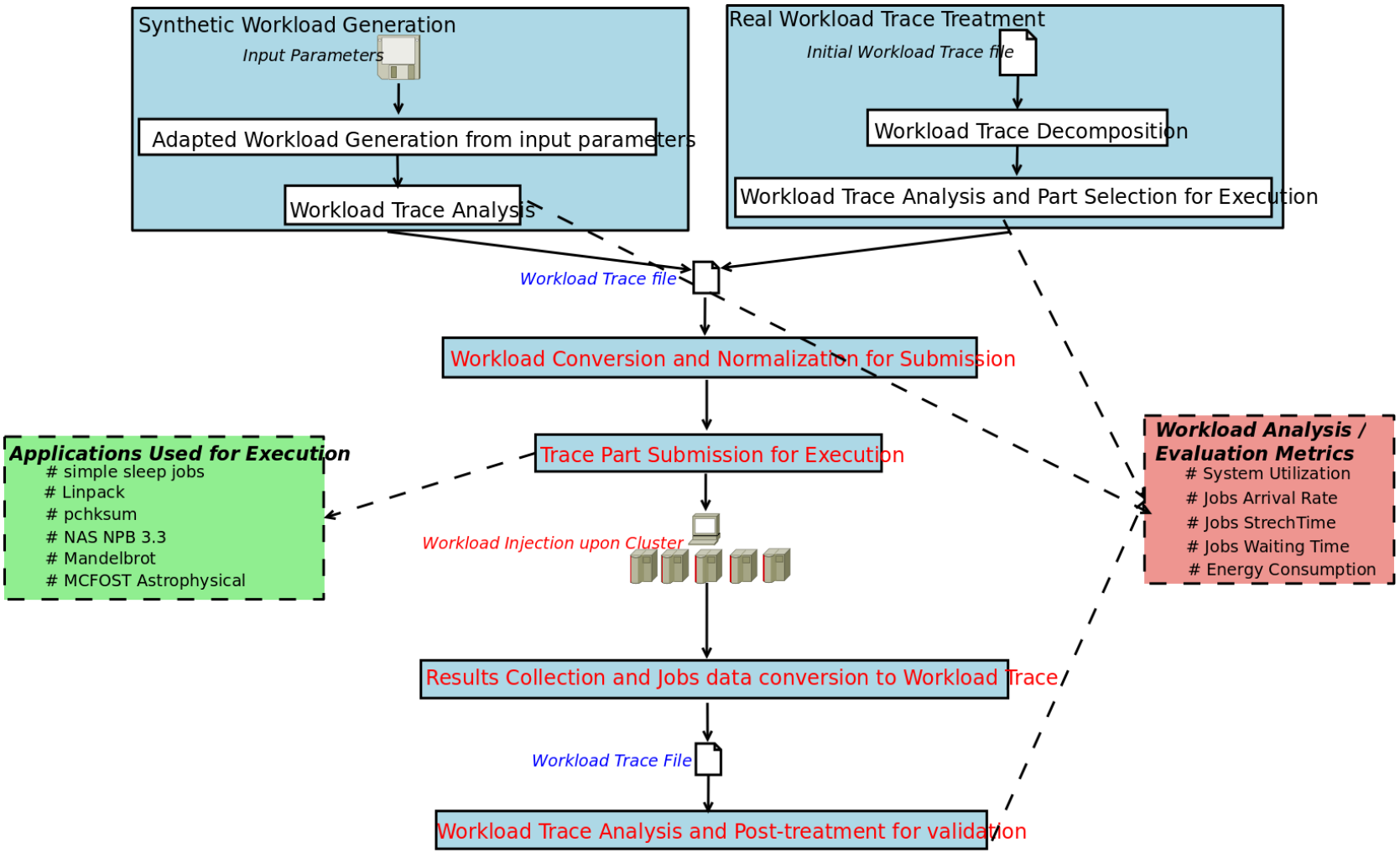
Experimental Methodology - Workloads

A particular case of synthetic workload is the ESP benchmark. It has been designed to:

- provide a **quantitative evaluation** of launching and scheduling functions of a resource and job management system via a single metric, which is the smallest elapsed execution time of a representative workload
- Complete **independence** from the hardware performance (execution of a simple MPI application (pchcksum) with fixed target run-time)
- Ability for **efficiency** evaluation of different scheduling policies (injection of the same workload)
- Ability for **scalability** evaluation of the RJMS (dynamically adjusted proportional job mix to reflect the system scale)

▸ ESP benchmark

Experimental Methodology - Analysis



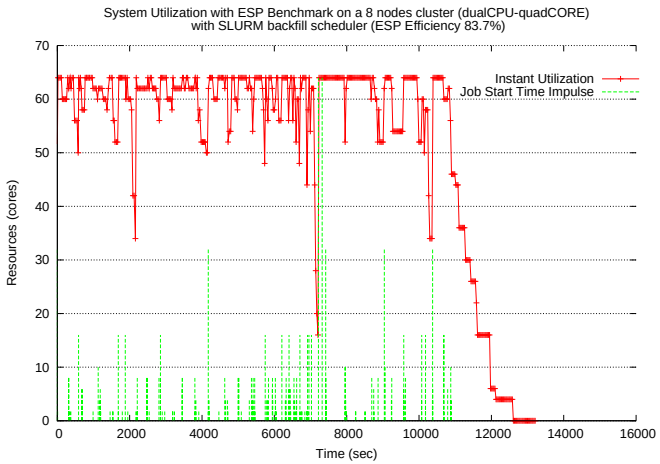
Scheduling Policies Comparisons - Experimentation

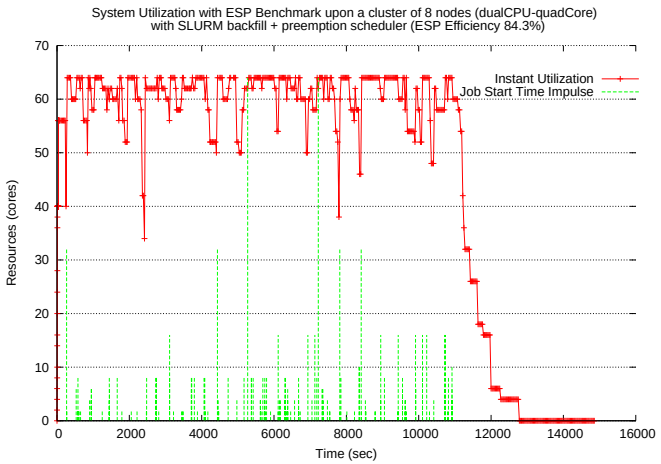
- Different runs of ESP benchmark: launching the same workload (default ESP) and executing the same application (pchcksum)
- upon the same **dedicated cluster**: 1 server, 8 nodes (Intel Xeon 2.5GHz 2 CPU-4 CORE, RAM 8 GB, Infiniband 20G)
- with the same conditions and **parameters**:
  - SLURM v2.2.0.0-pre3 with accounting (mysql-slurmdbd),
  - granularity (cons_res),
  - task confinement (task_affinity/cpuset),
  - NO topology plugin,
  - NO backup slurmctld
- but only one difference **the scheduling policy** : backfill, preemption, gang-scheduling

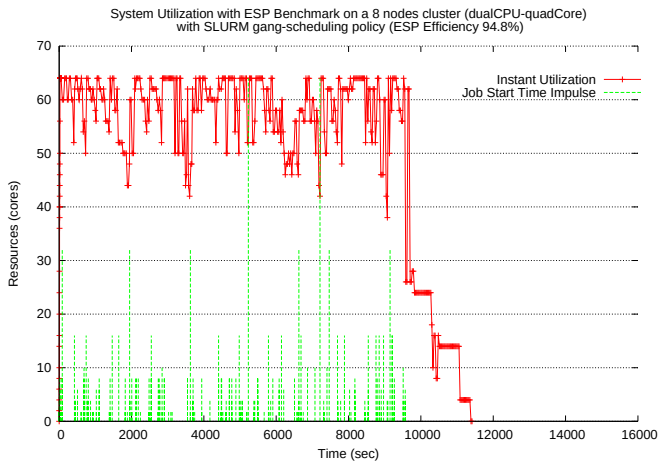
Scheduling Policies Comparisons - Results

Scheduling Policy	backfill	backfill+preemption	gang-scheduling
Total Workload Execution time	12877sec	12781sec	11384sec
ESP Efficieny	83.7%	84.3%	94.8%

Table: ESP benchmark results for SLURM scheduling policies upon a 8 nodes (64 cores) cluster







▶ Details

Scheduling Policies Comparisons - Analysis

- gang scheduling the best performance, allowing the efficient filling up of all the 'holes' in the scheduling space
- however this is due to the simplicity of the particular application suspend/resume happens on memory , no swapping needed
- **preemption** better than backfill: due to the 2 higher priority "all resources jobs".

Scalability Experimentations

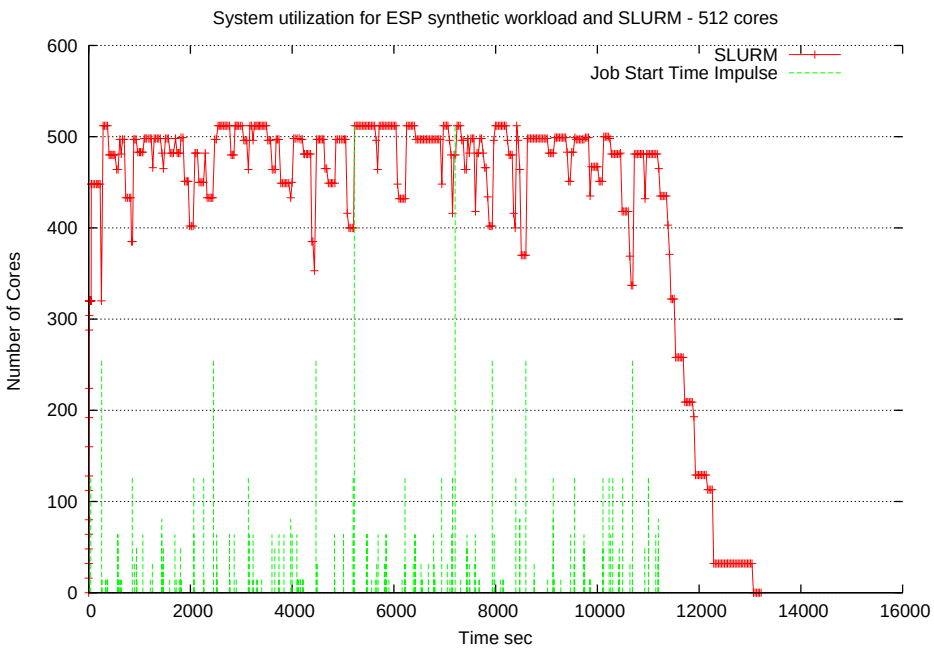
- Dedicated cluster 1 central controller, 320 computing nodes (quadCPU-octoCORE Intel Xeon 7500).
- with the same conditions and **parameters**:
  - SLURM v2.2.0.0-pre7 with accounting (mysql-slurmdbd),
  - granularity (cons_res),
  - task confinement (task_affinity/cpuset),
  - NO topology plugin,
  - NO backup slurmctld

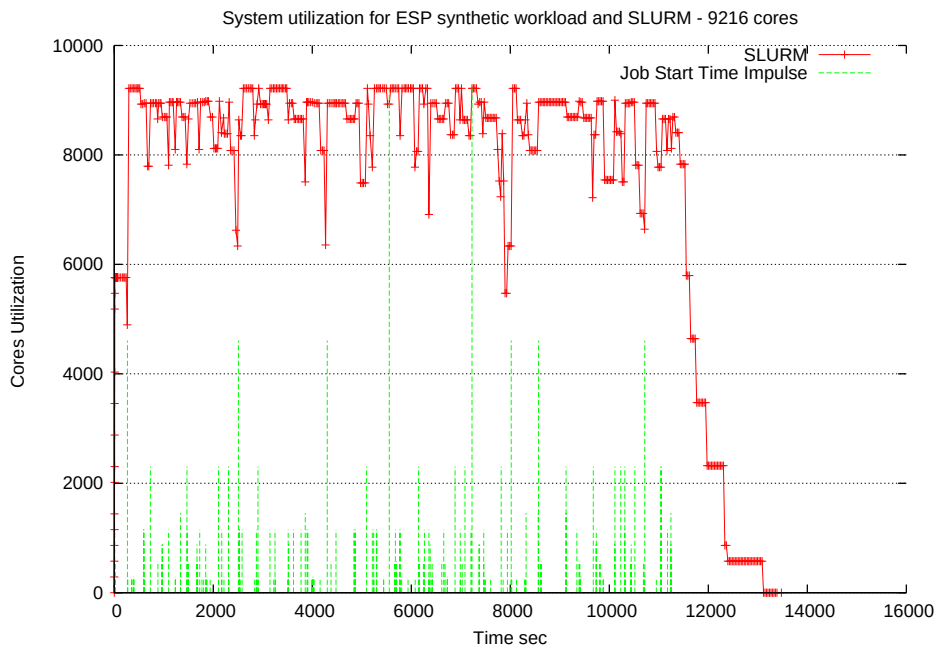
1. Scaling in the **size of the cluster** and execute ESP benchmark for launching and scheduling scalability evaluation.
2. Scaling in the **number of simultaneously submitted jobs** and evaluate the throughput of the scheduler.

Scaling the size of the cluster

Cluster size (Number of cores)	512	9216
Average Jobs Waiting time (sec)	2766	2919
Total Workload Execution time (sec)	12992	13099
ESP Efficieny for backfill+preemption policy	82.9%	82.3%

Table: ESP benchmark results for SLURM backfill+preemption scheduling and different cluster sizes

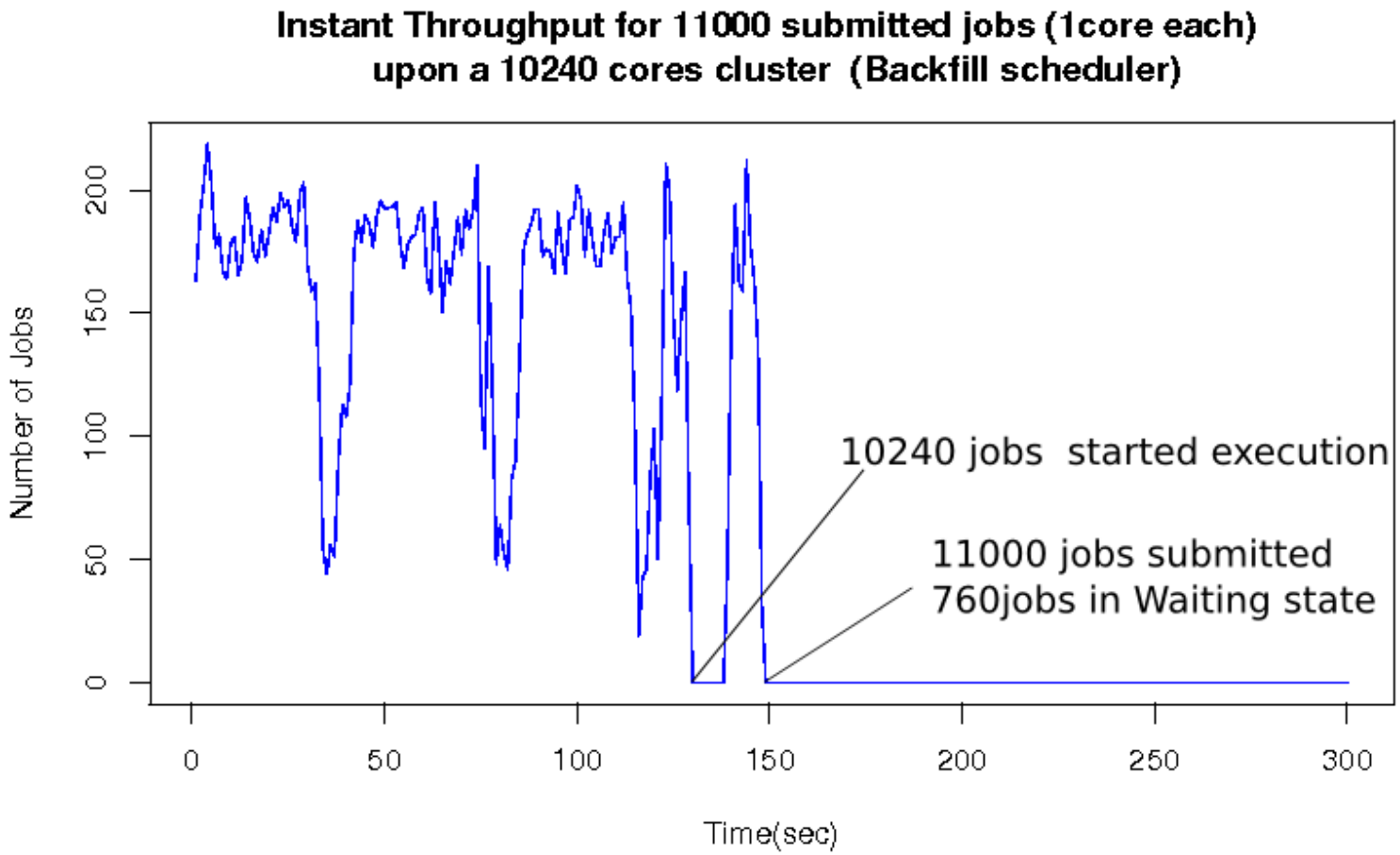




CDF Graphs

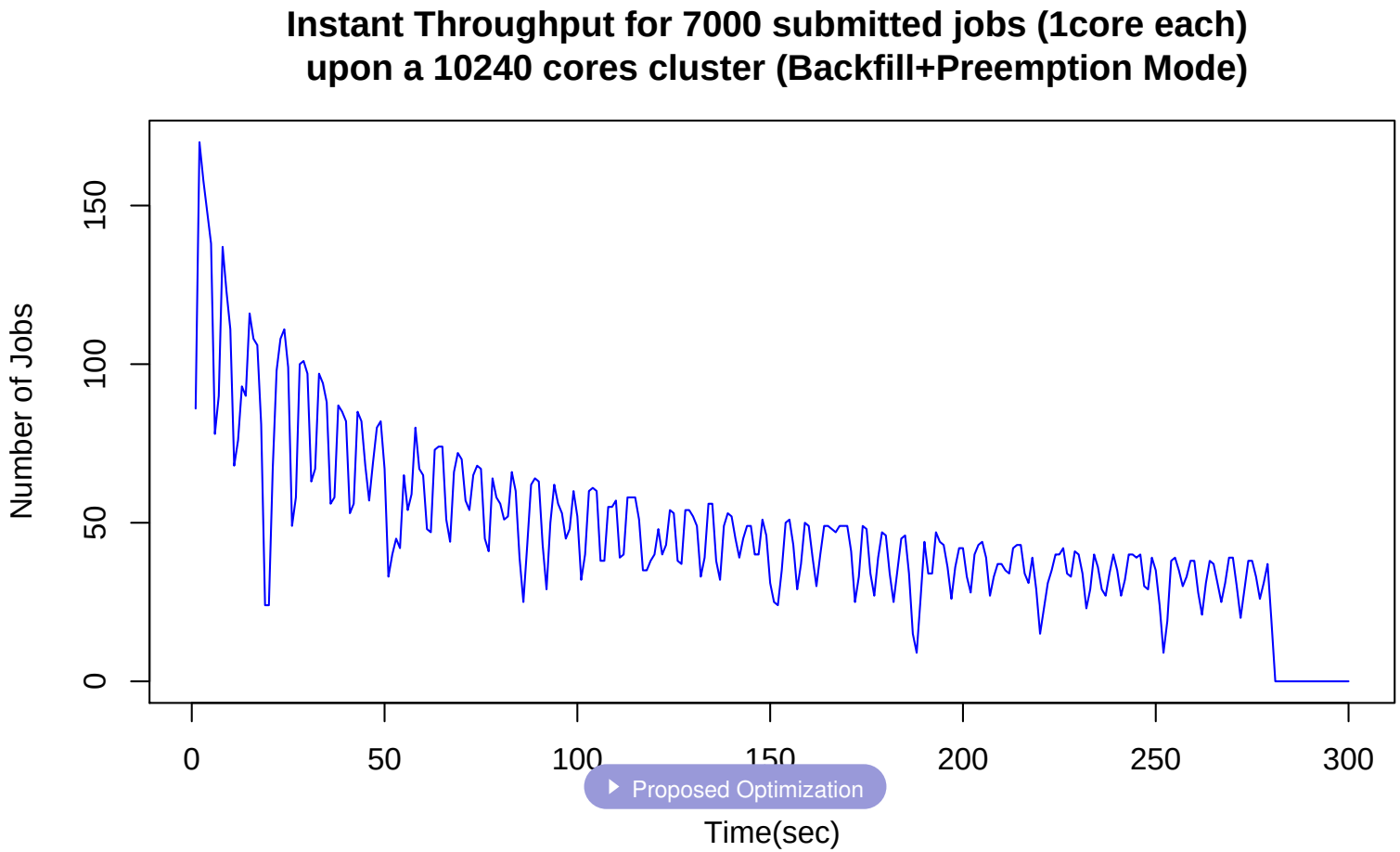
Scaling the number of submitted jobs: Backfill policy

- Submission burst of small granularity jobs (srun -n1 sleep 1000)
- Good throughput performance for backfill scheduler



Scaling the number of submitted jobs: Backfill with Preemption policy

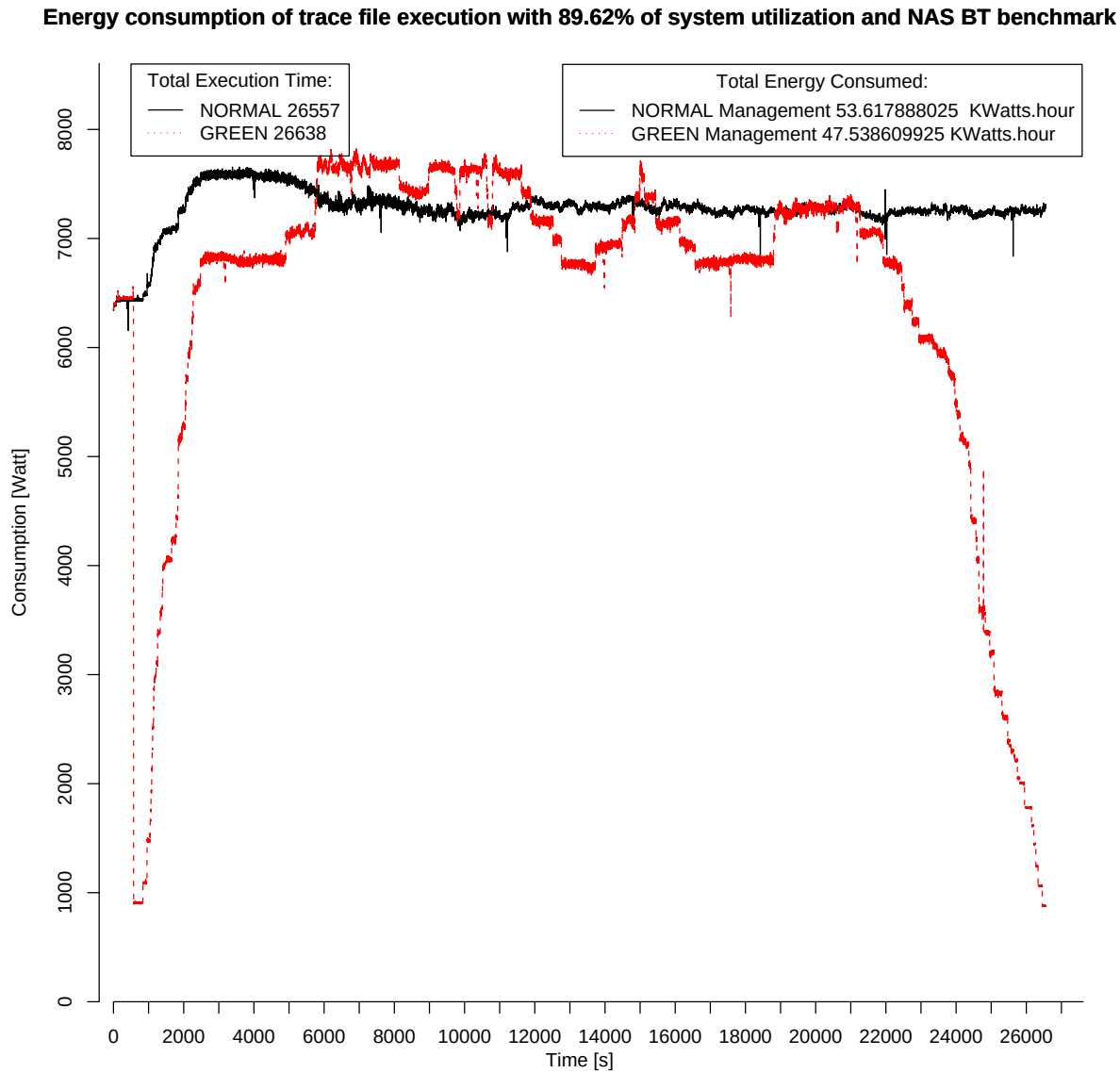
- Submission burst of small granularity jobs (srun -n1 sleep 1000)
- **No different priorities** between jobs, hence no possibility of preemption
- Degredation problems with backfill+preemption



Energy Efficient Resource Management

- Dedicated cluster 1 SLURM central controller, 32 computing nodes (DualCPU, 2GB, Gigabit Ethernet).
- with the same conditions and parameters SLURM-2.1
- launch a workload based on a trace file with 89.7% system utilization, execute NAS BT class C applications and measure the energy consumption of the whole cluster
- only difference: enable or not the **power saving mode**.
- Energy Consumption collected by Watt-meters (per node measures).

Energy Efficient Resource Management



▶ CDF Graphs

Network Topology Aware Placement Evaluations

- Dedicated cluster 1 SLURM central controller, 128 computing nodes (quadCPU-octoCORE Intel Xeon 7500).
- network topological constraints, 2 different islands , 64 nodes per island : higher bandwidth in the same island
- with the same conditions and parameters SLURM v2.2.0.0-pre7
- launch a workload based on ESP workload but without fixed run time for applications so as to observe the real execution time of the application
- in place of pchksum execute NAS MPI CG class D applications (sensitive in communications) and observe the total execution time of the whole workload
- only difference: enable or not the **topology plugin**.

Network Topology Aware Placement Evaluations

SLURM NB cores-TOPO Cons / Topo-ESP-NAS-Results	Theoretic- Ideal values	4096 NO-Topology Aware	4096 Topology Aware
Total Execution time(sec)	12227	17518	16985
Average Wait time(sec)	-	4575	4617
Average Execution time(sec)	-	502	483
Efficieny for Topo-ESP-NAS	100%	69.8%	72.0%
Jobs on 1 island	228	165	183
Jobs on 2 islands	2	65	47

Table: TOPO-ESP-NAS benchmark results for 4096 resources cluster

Conclusions SLURM

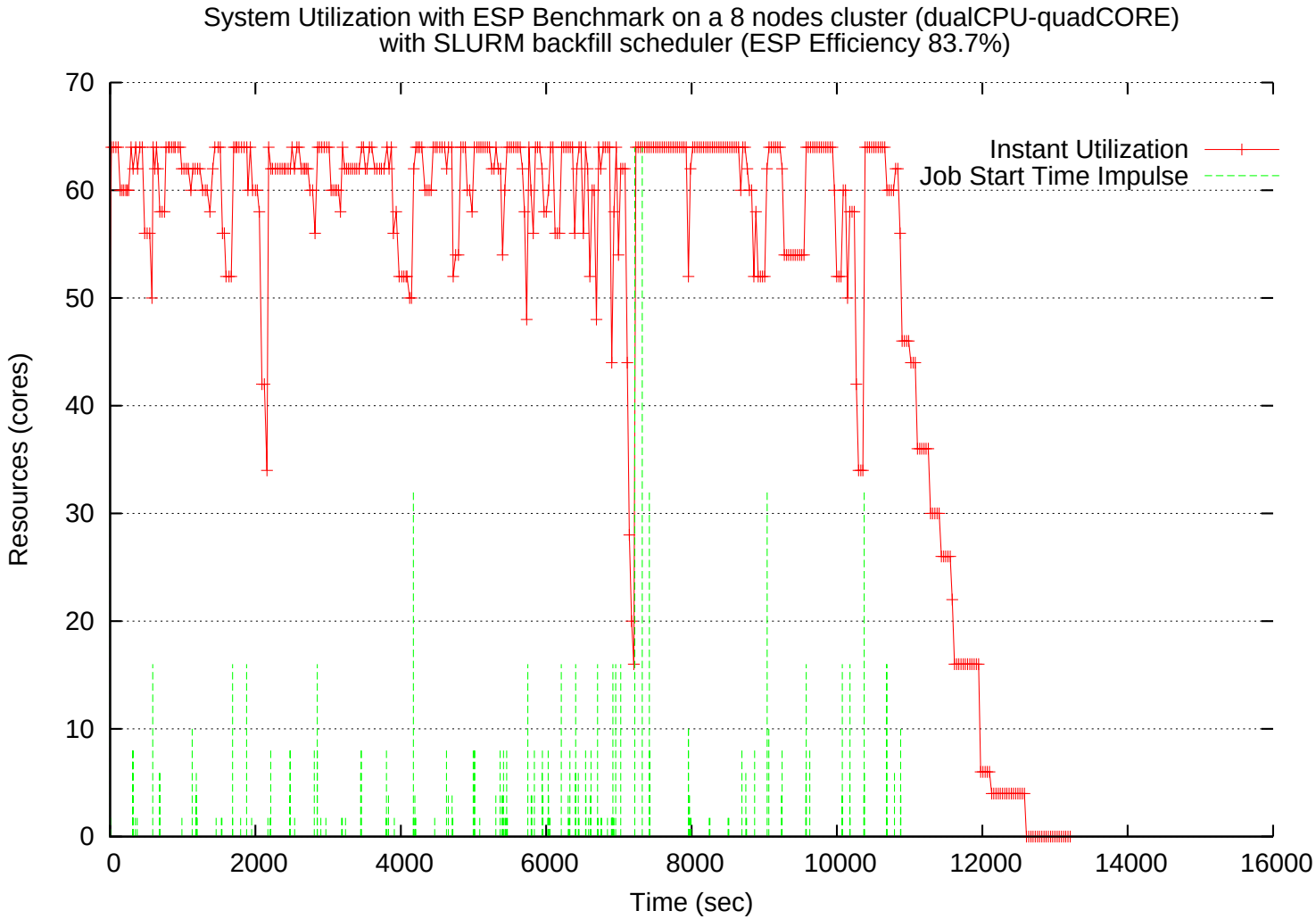
SLURM Resource and Job Management System has been choosen for BULL High Performance Computing Petaflopic Offer:

- Scalable, Efficient and Robust
- Passing the **Petaflop barrier** allowed us to experiment with **scalability** levels that are attained for the first time,
- **Real scale - controlled experimentation** guarantees that SLURM performance and efficiency will stay as expected.
- Research for ways of simulating an even larger platform in order to prepare the Resource and Job Management Systems for the next level of scalability.
- Deeper Workload Traces Modelization and replays of trace models for possible optimization (application performance, energy consumption) by considering the side-effects, tradeoffs

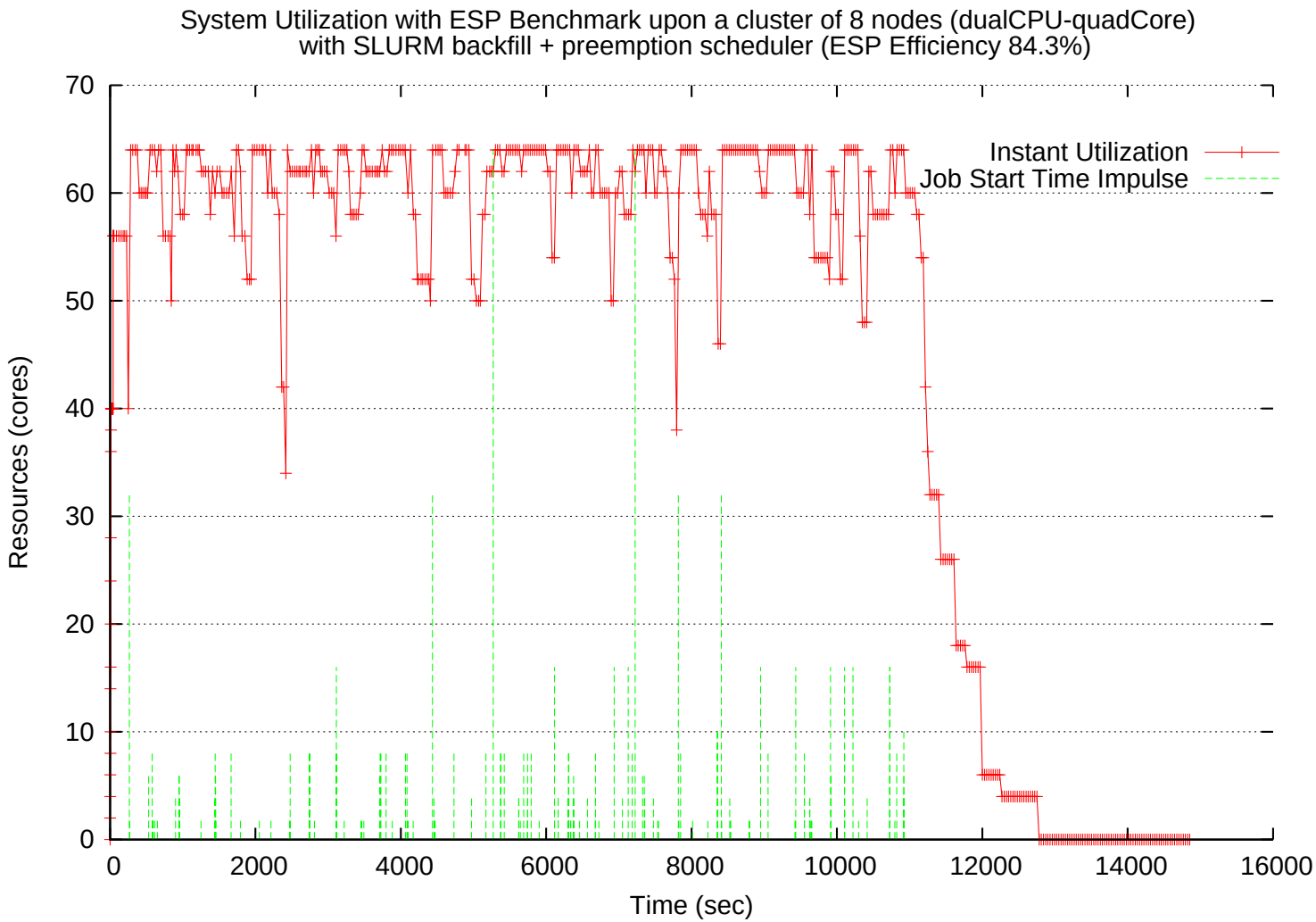
Ongoing Works and Collaborations

- **BULL's Team** for SLURM Developments and Support: Martin Perry (cgroups), Nancy Kritkausky, David Egolf(Licenses Management), Rod Schultz(topology extraction), Eric Lynn, Dan Rusak(sview), Doug Parisek(Power Management), Bill Brophy (strigger,scheduling optimizations), Yiannis Georgiou
- Collaborations with **CEA**: Matthieu Heutreux, Francis Belot
- **Research continuous** in modelization and experimentation Gael Gorgo (BULL, INRIA), Joseph Emeras (INRIA, CEA), Olivier Richard (INRIA)

Scheduling Evaluation: Backfill



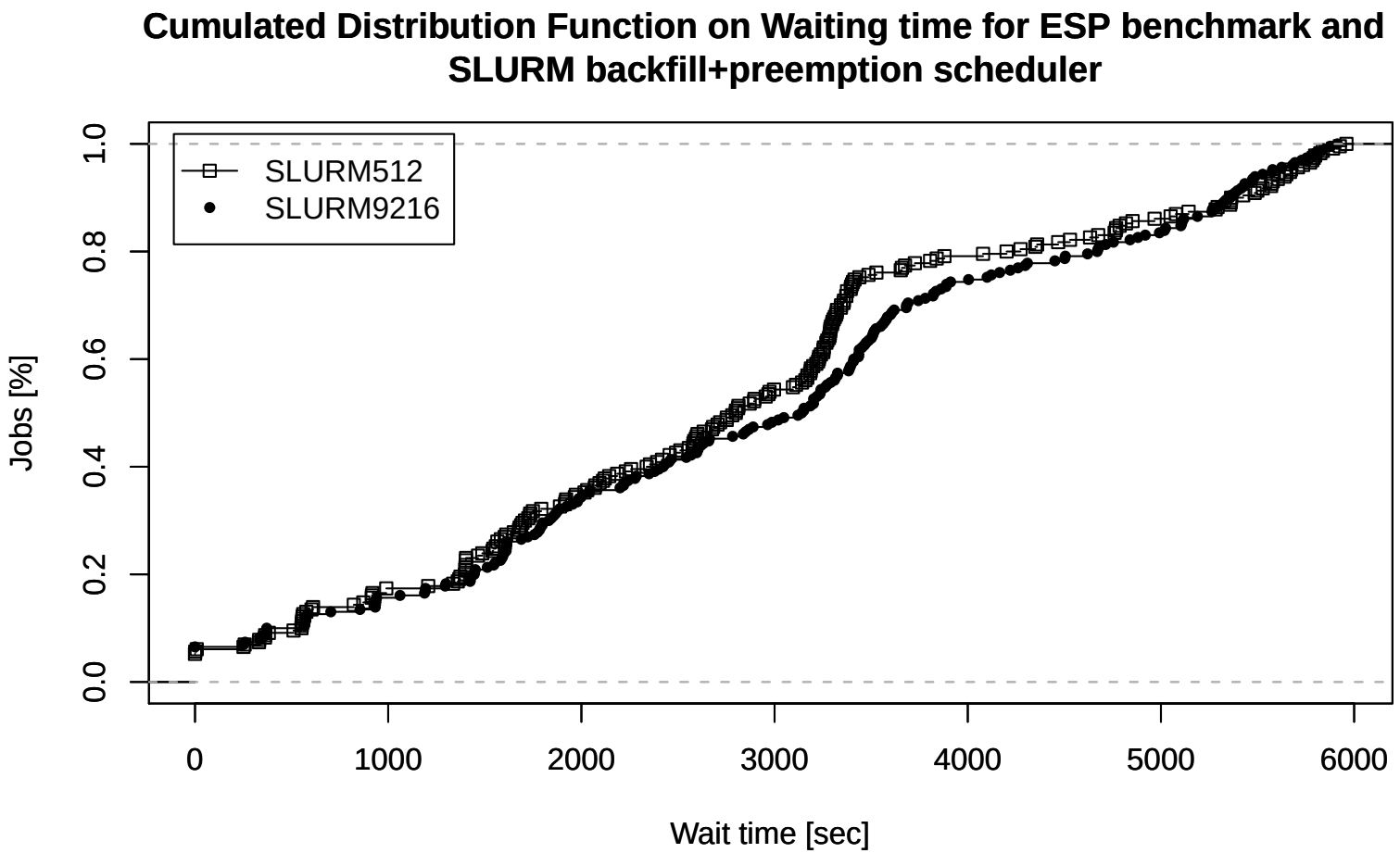
Scheduling Evaluation: Backfill + Preemption



Scheduling Evaluation: Gang Scheduling

Back.

Jobs Waiting Time



Back.

Throughput Preemption - Proposed Optimization

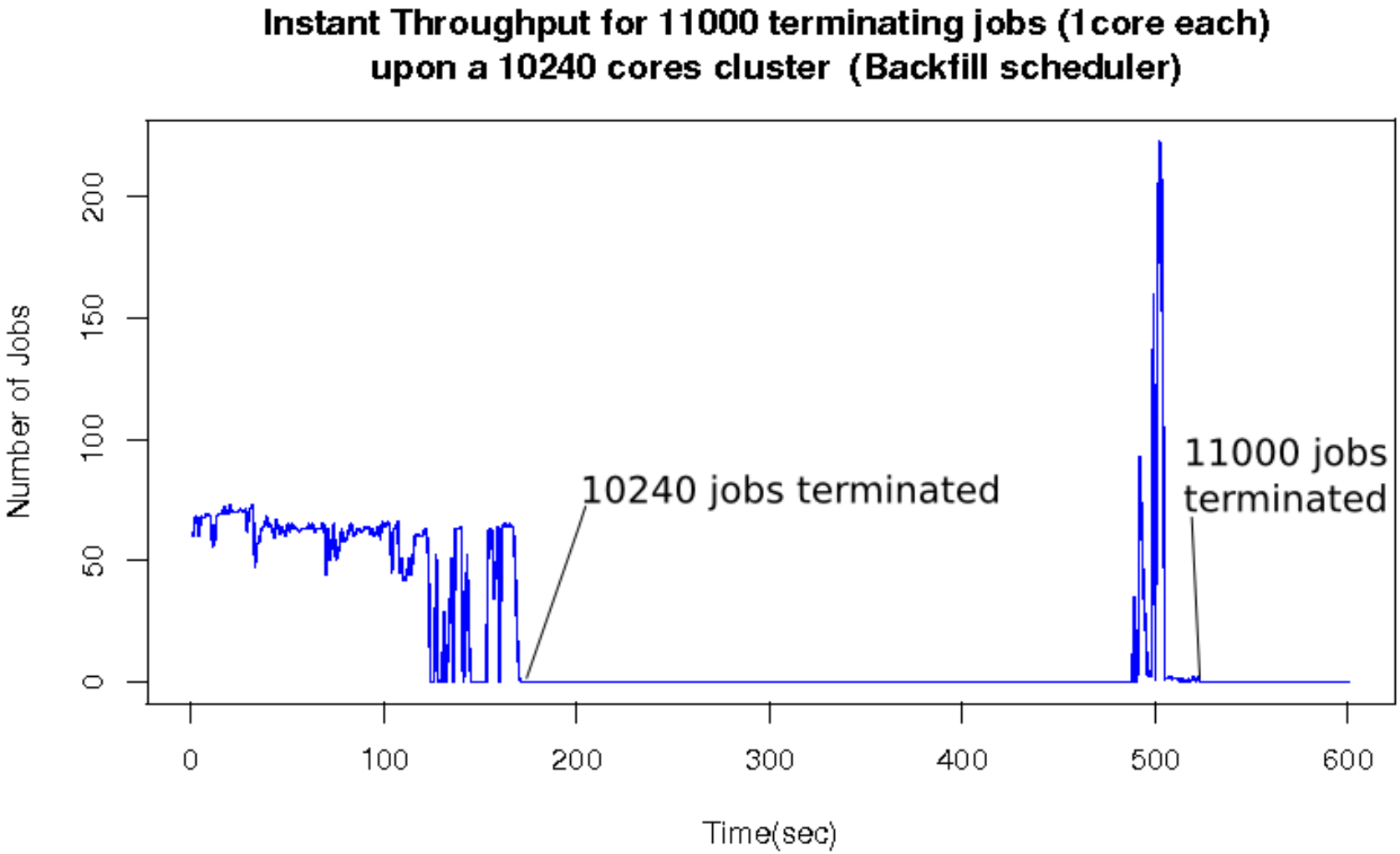
Degredation is due to unnecessary checks for each job for a preemption space even if all jobs have the same priority.

- In gang logic, skip shadow processing if the priority of all configured partitions are the same
- Skip unnecesary linear search of job list if submitted jobs have the same priority

2 patches recently developped (Bill Brophy), not yet verified.

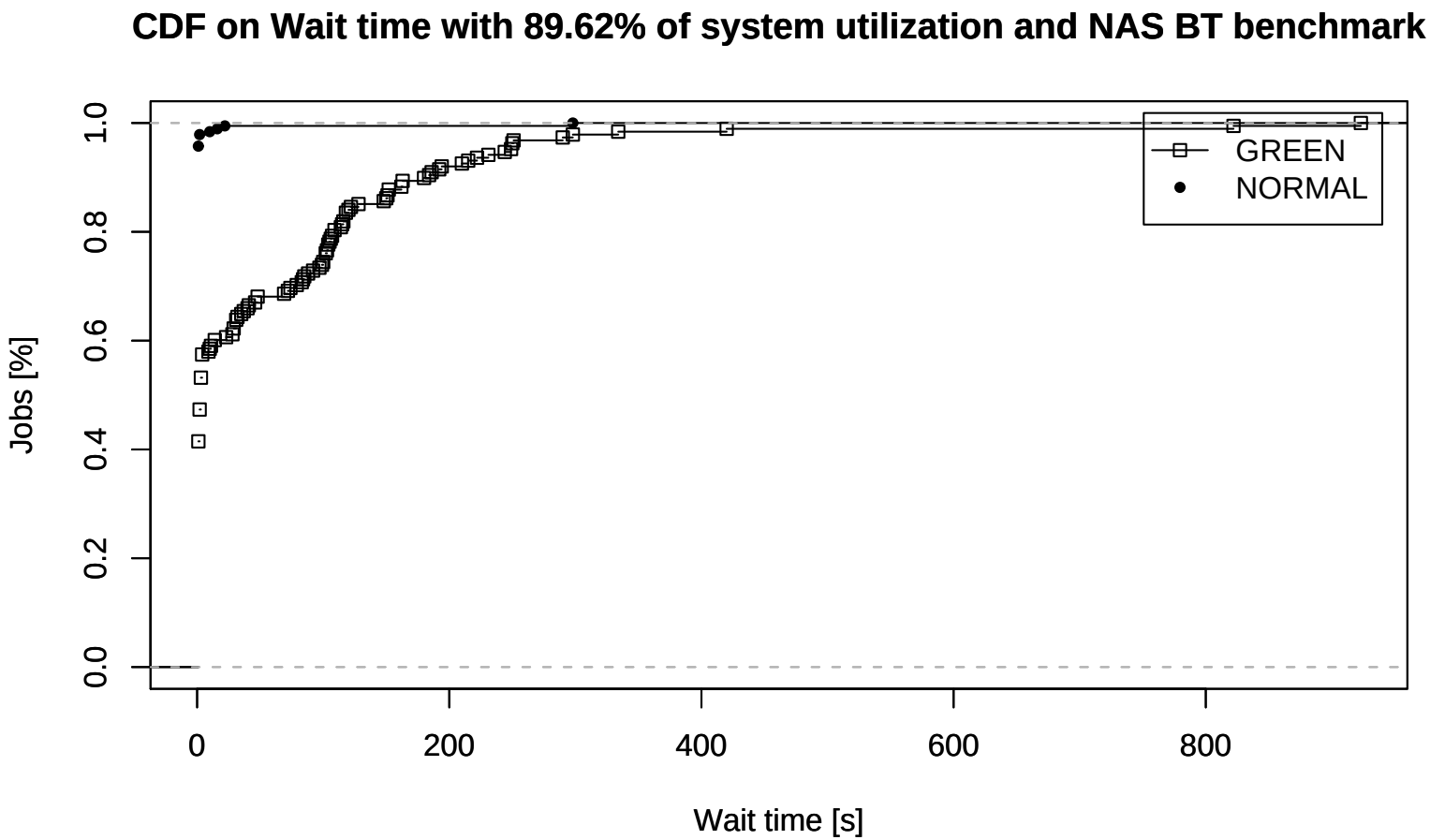
Back.

Throughput Experiments



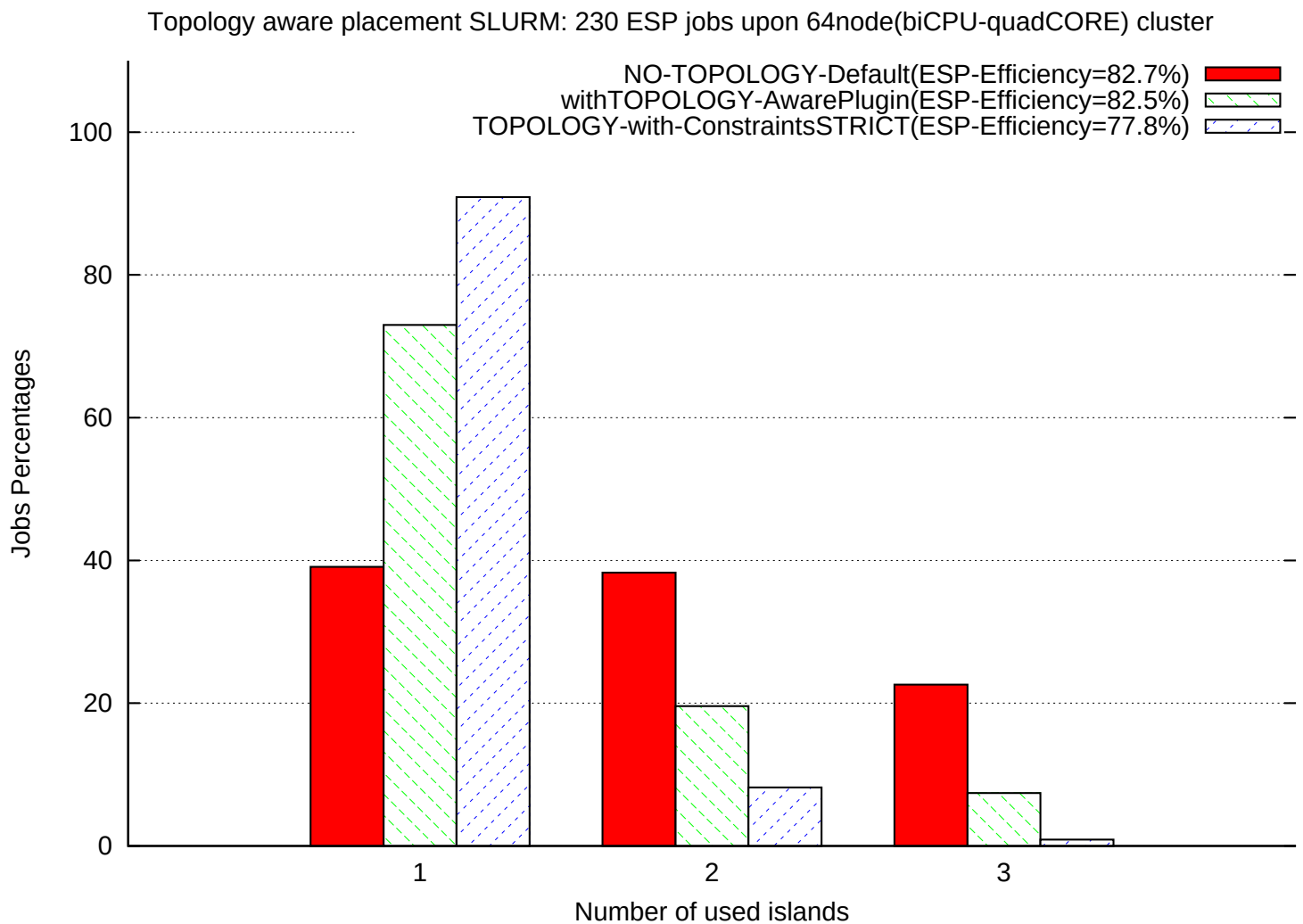
Back.

Energy Reductions Tradeoffs



Back.

Network Topology Aware Placement Evaluations



Archive of Real Parallel Workloads

Workload Traces	From	Until	Months	CPUS	Jobs	Users	Utilization %
LANL O2K	Nov 1999	Apr 2000	5	2,048	121,989	337	64.0
OSC Cluster	Jan 2000	Nov 2001	22	57	80,714	254	43.1
SDSC BLUE	Apr 2000	Jan 2003	32	1,152	250,440	468	76.2
HPC2N	Jul 2002	Jan 2006	42	240	527,371	258	72.0
DAS2 fs0	Jan 2003	Jan 2004	12	144	225,711	102	14.9
DAS2 fs1	Jan 2003	Dec 2003	12	64	40,315	36	12.1
DAS2 fs2	Jan 2003	Dec 2003	12	64	66,429	52	19.5
DAS2 fs3	Jan 2003	Dec 2003	12	64	66,737	64	10.7
DAS2 fs4	Feb 2003	Dec 2003	11	64	33,795	40	14.5
SDSC DataStar	Mar 2004	Apr 2005	13	1,664	96,089	460	62.8
LPC EGEE	Aug 2004	May 2005	9	140	244,821	57	20.8
LLNL uBGL	Nov 2006	Jun 2007	7	2,048	112,611	62	56.1
LLNL Atlas	Nov 2006	Jun 2007	8	9,216	60,332	132	64.1
LLNL Thunder	Jan 2007	Jun 2007	5	4,008	128,662	283	87.6

Table: Logs of Real Parallel Workloads from Production Systems `[Feitelson's Logs Archive http://www.cs.huji.ac.il/labs/parallel/workload/logs.html]`

Back.

ESP Benchmark

Job Type	Fraction of Job Size relative to total system size	Job size for a 512cores cluster (in cores)	Least number of needed islands	Count of the number of total jobs	Percentage of job instance	Target Run Time (Seconds)
A	0.03125	16	1	75	32.6%	267
B	0.06250	32	1	9	3.9%	322
C	0.50000	256	2	3	1.3%	534
D	0.25000	128	1	3	1.3%	616
E	0.50000	256	2	3	1.3%	315
F	0.06250	32	1	9	3.9%	1846
G	0.12500	64	1	6	2.6%	1334
H	0.15820	81	1	6	2.6%	1067
I	0.03125	16	1	24	10.4%	1432
J	0.06250	32	1	24	10.4%	725
K	0.09570	49	1	15	6.5%	487
L	0.12500	64	1	36	15.6%	366
M	0.25000	128	1	15	6.5%	187
Z	1.00000	512	3	2	0.9%	100
Total				230		

Table: ESP benchmark characteristics for 512 cores cluster [Kramer, William TC; 'PERCU: A Holistic Method for Evaluating High Performance Computing Systems]

Back.