Energy Consumption Evaluation and Strategies for a Small Heterogeneous Cluster

Brian Hall
University of Hawai’i at Manoa
ICS 668 / FA14 / Casanova

Abstract

Using an inline power meter, I measured power draw across several states on three nodes of a small heterogeneous compute cluster. With this information and a history of job submissions, I estimated the cluster’s historical energy consumption. I then benchmarked the three nodes’ performance. Using this data and the power draw, several “energy-aware” scheduling algorithms were simulated and tested on the same job submission data, as well as denser datasets. Wait times, energy efficiency and savings are evaluated at the node level, and recommendations for increasing energy efficiency are made.

1. Introduction

The Moana computing cluster at the USDA Agricultural Research Service Daniel K Inouye Pacific Basin Agricultural Research Center is a heterogeneous cluster used to carry out scientific computing jobs including large simulations and genome assemblies. It enjoys the distinction of being one of the few high performance computing clusters in the state of Hawai’i [5]. Hawai’i, for its part, enjoys the distinction of having the highest electricity costs in the nation [3].Given the cluster’s unique position and the diversity of workloads that it sees, a customized strategy for maximizing energy efficiency may be called for.

During the period of study, the cluster consisted of three compute nodes, henceforth referred to as “SLOW_1”, “SLOW_2”, and “FAST”. Both of the SLOW nodes are Supermicro nodes featuring H8QG6 motherboards with 48 AMD Opteron 6168 Processors running at 1.9 GHz. SLOW_1 has 256 GB of RAM, while SLOW_2 has 192 GB. FAST has an X9DRW motherboard with 256 GB of RAM and 32 Intel Xeon E5-2687W CPUs running at 3.1 GHz.

Moana’s scheduling policy is unusual in that nodes are shared amongst jobs. Users request a queue and a number of processors, and their job is assigned to a node accordingly. The queues available are “fast.q”, which assigns all jobs to FAST, “slow.q”, which assigns jobs to SLOW_1 or SLOW_2 and “all.q”, which chooses the first available node from the three. It is thus possible that several jobs may be running on a single node at a given time.

As Moana is a fairly young facility, little study has been done on its utilization. It is hoped that this project will cast some insight on the cluster’s operations and be helpful in guiding usage policies and hardware acquisitions in the future.

2. Related Work

Goiri et al investigated the impact of energy-aware scheduling algorithms in data centers equipped with solar power panels [2]. They found that running jobs during periods when solar energy is available can significantly reduce energy consumption. Because they considered the usual case in which each job is run in isolation on one or more dedicated nodes, they were able to calculate energy consumption per job. In such a model, the energy consumption of a node is simply the sum of the per-job energy consumption for all jobs run on that node. This approach is not possible with the usage policy of the Moana cluster, though the conclusion that an energy-aware scheduling algorithm can in fact reduce consumption is encouraging.

Other researchers have investigated the effect of selectively powering down compute nodes and only turning them on as workload requires [7]. This approach seems to be effective, and implementing an option to remotely power nodes on and off is a project in the works on Moana. However, since no such feature is currently in place, I limited my experiment to simpler, immediately implementable policies.

3. Methods

Each of the three nodes to be investigated was plugged into a Raritan Dominion PX Power Distribution Unit (PDU). The “fast” node has two plugs; these were aggregated into a single group using the Raritan’s web interface.

Each node’s power consumption was then measured several times in each of four states. The first state, Powered Down, corresponds to a machine which has been shut off via the command line. The second, Idle, represents a booted-up machine performing no compute tasks. The third state, Working, represents a machine running a transcriptome assembly with full core usage enabled, which is a common function of the Moana cluster. The last state, Stressed, involves running the Linux program ‘stress’, which was configured to maximize cpu load [8]. From this data an average was taken to represent each machine’s estimated power consumption in each of the four states. (This experiment only considers the Idle and Working states.)

With this information, a simulation was carried out using historical job submission data on the cluster. Each node’s power consumption was tracked during the month of June 2014, using the Working power consumption for times when a job was being computed and using the Idle consumption for downtime. During the period of study, the price of electricity was $0.326671 per kilowatt-hour [4]. This was used to compute each node’s energy cost. The results of these calculations appear in Figure 1.

Node	Working Energy Consumption (W)	Idle Energy Consumption (W)	Compute Time (dd:hh:mm:ss)	Idle Time (dd:hh:mm:ss)	Kwh	Cost
FAST	254.095	147	10:21:8:25	16:7:18:39	123.8771314	$40.47
SLOW_1	375.8	360.5	2:7:51:16	24:20:35:48	236.0631986	$77.12
SLOW_2	303	254	7:5:32:31	19:22:54:33	174.2261375	$56.91
Total	932.895	761.5	20:10:32:12	61:02:49:00	534.1664674	$174.50

Figure 1: Historical Compute/Idle Time and Energy Cost

Next, each machine's speed was benchmarked. This was carried out with the Broad Institute's DISCOVAR benchmark package, which performs a compute-intensive genome assembly algorithm several times to generate an average compute time [1]. This test was chosen because genome assemblies make up much of Moana's workload. The results of the benchmark appear in Figure 2.

Node	Benchmark Compute Time (seconds)
FAST	391.17
SLOW1	586.43
SLOW2	644.75

Figure 2: DISCOVAR Benchmark Results.

Two observations were immediately apparent from this preliminary data: the cluster's overall workload is relatively light (that is, the proportion of idle time is high), and the FAST node is superior both in energy efficiency and in computing speed. Based on these facts, scheduling algorithms were implemented in simulation for the purpose of exploring different strategies to reduce energy consumption.

4. Simulation

The first simulation scenario, "allfast", mirrors an arrangement in which the two SLOW nodes are powered down. All jobs are assigned to the FAST node. Implementing such a policy required some significant assumptions. Each job's compute time is known *for the node on which it originally ran*. This necessitates a formula for estimating a job's compute time on FAST based on its compute time on SLOW_1 or SLOW_2. The situation is further complicated by the fact that some jobs requested all 48 processors on a SLOW node; FAST has only 32 CPUs.

The wide diversity of jobs submitted to the cluster means that accurately implementing this function would require extensive micro-benchmarking. As such an undertaking was beyond the scope of this study, an effort was made to make conservative assumptions. The time for a job to run on FAST was simply assumed to be equal to its SLOW runtime. This ignores the fact that the former node is significantly faster than the other two. However, the assumption also ignores the effect of reducing the number of CPUs from 48 to 32 in the case of high-core-usage jobs. It is hoped that this runtime estimate overestimates most (if not all) jobs' actual FAST runtime, but it must be emphasized that this assumption has not been tested and is fundamental to the simulation results.

The results for “allfast” (and all other algorithms) are summarized in Figures 3-5. While the algorithm successfully reduces energy consumption, it drastically raises the maximum wait time and significantly increases the average wait time. This effect is likely unacceptable to the users of Moana, so two more algorithms were implemented.

Each of these algorithms favors “greener” nodes in assigning jobs. They differ in that one of them, “greenfirst2nodes”, leaves the least efficient node powered off. The other, “greenfirst3nodes”, uses all three nodes but only assigns jobs to the least efficient one when the other two are busy.

Because the historical job submission data was of a sparse nature, two further input sets were developed. One of them simply duplicates every other job in the original record; the other duplicates every single job. These are referred to as “x1.5” and “x2”, respectively. Each algorithm was tested with these datasets as well as the original. All simulator code and datasets are available online at https://github.com/BrianReallyMany/scheduling_simulator.

5. Results and Discussion

The results of running each algorithm with a variety of inputs are summarized in Figures 3-5.

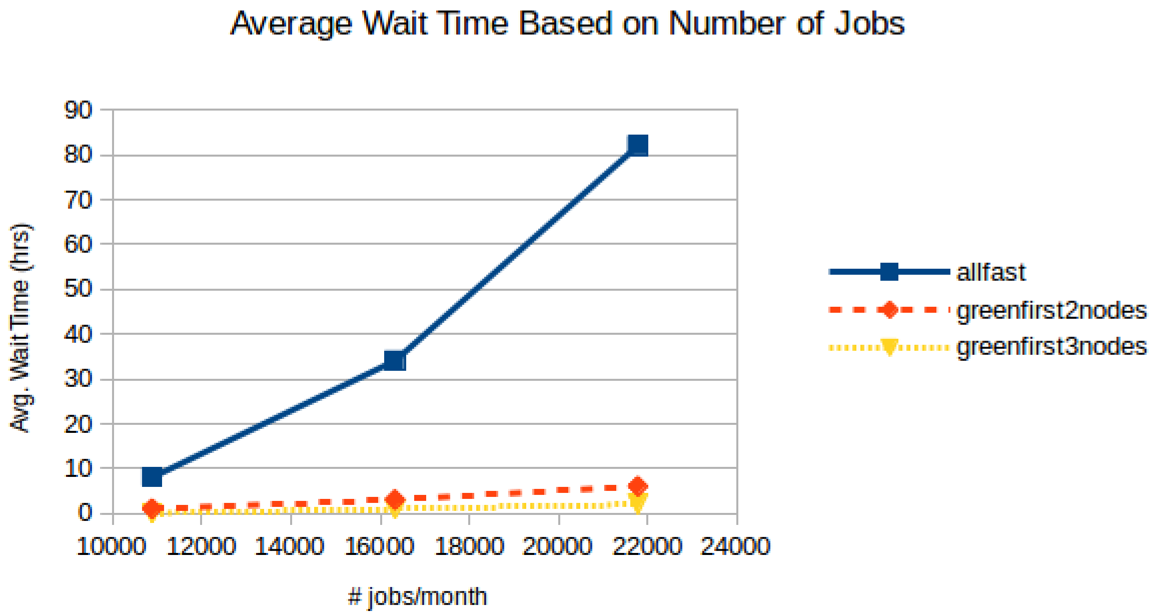

Figure 3: Average Wait Time. All three algorithms exhibit an increase in average wait time as job density is increased, but the increase is slight where more than one node is available. The three data points represent the number of jobs submitted in a month for the historical data, “x1.5” and “x2”.

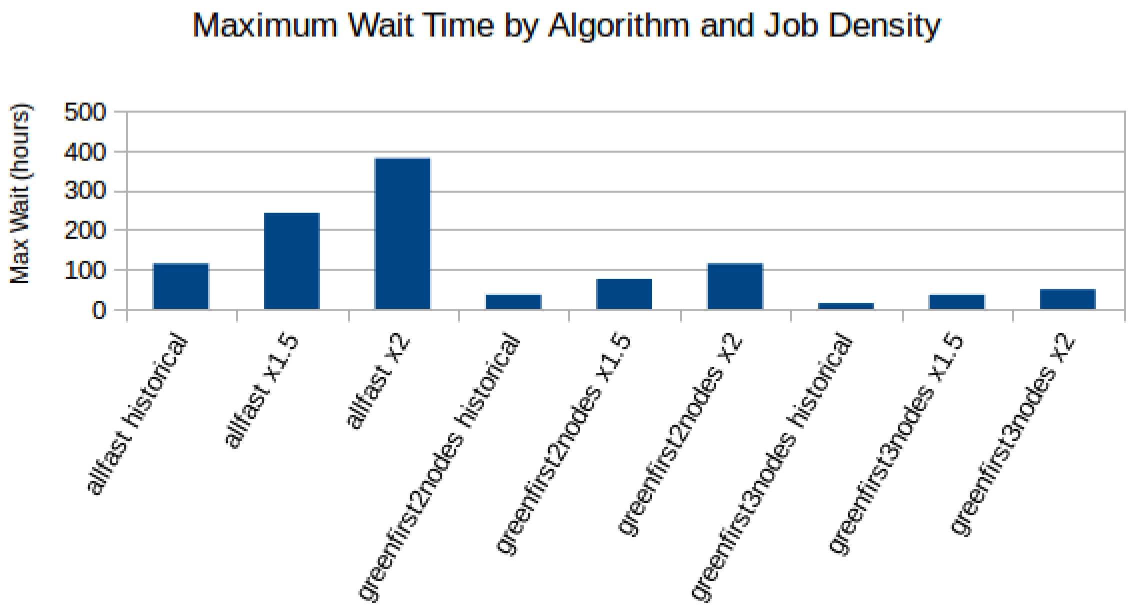

Figure 4: Maximum Wait Time. This statistic is quite high for “allfast”, but greatly reduced with the multi-node algorithms. This compares favorably with the actual historical job submission data, for which the maximum wait time was nearly 16 hours.

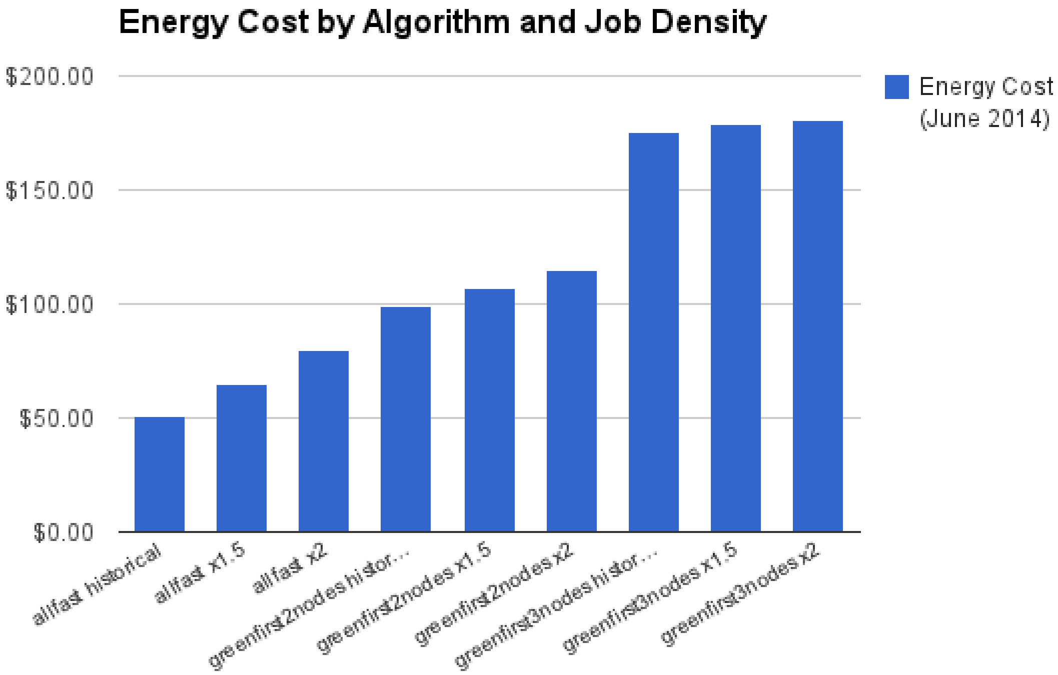

Figure 5: Energy Cost. Job density has less of an effect on overall cost than number of nodes enabled; the total range of possible savings is less than $150.

As expected, each algorithm sees increases in average and maximum wait time as well as energy consumption as job density is increased. As noted above, the “allfast” algorithm offers significant energy savings -- more than $100 over the course of a month. However, it nearly quadruples the average wait time and raises the maximum wait time from 15 hours to more than four days.

The “greenfirst2nodes” algorithm was intended to mitigate this increase in wait times. Its success is so thorough as to be peculiar; its average wait time is actually less than that for the historical data in spite of making use of only two nodes. This strange result may be explained by the “queue” system in use on Moana. Users may request the FAST node or one of the SLOW nodes specifically, and if the node is overloaded at the time, the job simply waits. The simulated algorithm will always schedule a job if either node is available.

Another observation concerns trends as job density increases. It is apparent that increasing, even doubling, the number of jobs submitted has only a small effect on energy usage, regardless of the algorithm chosen. However, wait times quickly increase. This is explained by the fact that the nodes’ power consumption in the Idle state is relatively close to their Working consumption. As discussed earlier, a scheduling policy which powers down inactive nodes might hope to see more significant gains in efficiency.

6. Conclusions and Future Work

A number of tentative conclusions are suggested by the data. It is apparent that the SLOW_1 node is the least efficient in the cluster, and that removing it entirely saves money without greatly increasing wait times. An experimental policy of keeping this node powered down until the cluster is under a heavy load may be advisable.

At the same time, it should be noted that the power savings offered by these alternative scheduling policies are likely insignificant when compared with the cost of operating the facility which houses the cluster. Cooling expenses are not considered in this study, yet they are likely to cost more than half of the cost of running the cluster itself [6]. Considering this cost as well as the costs for lighting the building, powering workstations, lab equipment, etc., a savings of $100 is simply not significant.

Still, the sparseness of historical jobs does seem to indicate that the ability to power nodes on and off remotely (or even automatically by the scheduler itself) might be exploited to maximize savings while having negligible effect on wait times. This possibility is left for future studies.

Another conclusion which cannot be emphasized enough is that better benchmarking data is required in order to conduct more realistic simulations. Rather than prioritize gathering this data, though, I will recommend an experimental approach in which the slowest node is disabled and wait times are monitored. If they stay within acceptable bounds, the node may become a candidate for repurposing. One possibility is that the node be powered on for new user training sessions or for running programs which are interactive yet compute-intensive.

Yet another conclusion that may be drawn from this data is that Moana is ready to accommodate more users and/or jobs. Rather than decommission or repurpose nodes to counterbalance a fairly sparse workload, administrators may choose to court new users and encourage existing users to take greater advantage of the computing facilities. SLOW_2 may be an energy hog when compared with its rack mates, but the cost of running it is low relative to overall facility costs. All things being equal, it may be preferable to feed it data rather than see it gather dust.

7. References

[1] The Broad Institute. “Benchmarking | DISCOVAR”. http://www.broadinstitute.org/software/discovar/blog/?page_id=187.
[2] Goiri, I.; Kien Le; Haque, M.E.; Beauchea, R.; Nguyen, T.D.; Guitart, J.; Torres, J.; Bianchini, R., "GreenSlot: Scheduling energy consumption in green datacenters," *High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for* , vol., no., pp.1,11, 12-18 Nov. 2011.
[3] Hawaii Energy. “Get the Facts.” http://www.hawaiienergy.com/get-the-facts.
[4] Hawaiian Electric Company, Inc. “Effective Rate Summaries” for June 2014. http://www.heco.com/vcmcontent/StaticFiles/FileScan/PDF/EnergyServices/Tarrifs/HECO/EFFRATESJUN2014.pdf.

[5] “Moana@PBARC”. http://moana.dnsalias.org/wordpress/.
[6] Patel, C.; Bash, C.; Sharma, R.; Beitelmal, M. “Smart Cooling of Data Centers,” in *Proceedings of IPACK,* July 2003.
[7] Pinheiro, E; Bianchini, R; Carrera, E; Heath, T. “Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems,” *Workshop on compilers and operating systems for low power,* pp 182-195. Nov. 2001.
[8] Stress Project Page. http://people.seas.harvard.edu/~apw/stress/.