Reducing power usage on demand

G Corbett, A Dewhurst

Science and Technology Facilities Council, Rutherford Appleton Laboratory, Harwell Oxford, Didcot OX11 0QX, UK

E-mail: greg.corbett@stfc.ac.uk, alastair.dewhurst@stfc.ac.uk

Abstract. The Science and Technology Facilities Council (STFC) datacentre provides large-scale High Performance Computing facilities for the scientific community. It currently consumes approximately 1.5MW and this has risen by 25% in the past two years. STFC has been investigating leveraging preemption in the Tier 1 batch farm to save power.

HEP experiments are increasing using jobs that can be killed to take advantage of opportunistic CPU resources or novel cost models such as Amazon's spot pricing. Additionally, schemes from energy providers are available that offer financial incentives to reduce power consumption at peak times.

Under normal operating conditions, 3% of the batch farm capacity is wasted due to draining machines. By using preempt-able jobs, nodes can be rapidly made available to run multicore jobs without this wasted resource. The use of preempt-able jobs has been extended so that at peak times machines can be hibernated quickly to save energy.

This paper describes the implementation of the above and demonstrates that STFC could in future take advantage of such energy saving schemes.

1. Introduction

The STFC datacentre provides large-scale High Performance Computing facilities for the scientific community. In June 2015, the power consumption of the compute resources was 913 kW, with an additional 500 kW for cooling. This means the datacentre has a Power Usage Effectiveness (PUE) of 1.55. Over the last two years, the power consumption of the centre has increased by approximately 25% despite the physical footprint remaining the same. This power increase corresponds to a $£247,000$ increase in running costs over the two years.

This paper documents a pilot scheme to reduce the energy consumption of the STFC datacentre to both save money and reduce environmental impact. The pilot scheme focused on a subset of the datacentre, specifically the worker nodes of the UK Tier 1 batch farm, which forms part of the Worldwide LHC Computing Grid[1]. The batch farm uses HTCondor[2] for workload management. The batch farm consists of approximately 600 physical worker nodes that provide 15500 job slots. When full the power consumption is $\approx$ 200kW.

The motivation behind the pilot scheme was to decide if STFC could take advantage of a UK National Grid scheme[5] where large commercial users are paid to reduce their power usage when necessary during peak winter months. The National Grid scheme is called "Demand Side Balancing Reserve" (DSBR) and was developed in 2013 to protect consumers from experiencing problem with their electricity supply. This was due to the margin between energy generation and demand tightening. The increase in the amount of renewable energy generated also means that the supply is less reliable. Peak power demand is between 4pm and 8pm during winter

months and if activated DSBR will pay users if they can provide a 1MW drop in the power they use within 15 minutes. There are different price bands, ranging from 0.25 per kWh through to 2.50 per kWh, depending on the severity of the problems. Users can choose which price band depending on the cost to them but the higher they pick the less likely they are to be selected. The scheme was first used in November 2015 [6].

There were two main parts of the project. To develop a means of powering down idle machines and powering them back up when required later. To develop a system of identifying preempt-able jobs on the batch farm so that they can be killed if required. The work undertaken for this project could be applied to other parts of the STFC datacentre as well as other datacentres.

2. Preempt-able jobs

HTCondor has features which allow the running of jobs that can be paused if a higher priority job comes along and restarted when resources become available again. However, it requires the software to be written in a specific way, which can be difficult for large collaborations to implement. Due to the security model employed by the LHC experiments, jobs cannot be paused for an extended period of time as the job's authentication credentials will expire. Jobs that are paused at one site cannot be transferred to another site to keep running if capacity is available there. This means that this form of preemption is not widely adopted. It should also be noted that in the past, if an experiments jobs were considered preempt-able at a site, it normally meant they were not the primary user of the site and therefore received worse support.

The arrival of competitively priced large scale cloud computing has significantly changed things. One particular example that has been looked at by ATLAS is Amazon's Elastic Cloud spot pricing[4]. It allows users to bid on the price of CPU resources. If the price rises above what you are prepared to pay your jobs will be killed although you will only be charged for complete CPU hours used. This has lead to the development of the ATLAS Event Service[7] which uploads the results of every event it processes and very little work would be lost in the event the job is killed. The implementation of preempt-able jobs at STFC, is therefore designed to mimic this use case.

3. Power benchmarking and monitoring

The first step was to measure how the power consumption of a worker node varies depending on the load it is put under. A multi-meter was connected to one of the chassis in the datacentre. Each chassis contains four worker nodes. While measuring the power usage each node in the chassis ran identical tests.

HEP-SPEC06 [8] benchmarking tool was used to generate load on the worker nodes. The power usage of the machines was measured when they were idle and with the HEP-SPEC06 benchmark using 8, 16, 24, and 32 threads. The results are shown in Figure 1. An idle worker node uses just over 90W. The CPUs have hyper-threading turned on which is why the power usage increase slows down after 16 cores are being used. HEP-SPEC06 is linearly proportional to the power used. If the worker nodes are made to hibernate, they used less than 5% of their idle power.

An idle machine uses 35% the power of a fully loaded machine. To maximise the energy efficiency of the batch farm, packing jobs on to as few a nodes as possible and hibernating the rest is optimal. It should be noted that in general Grid jobs are not as CPU efficient as the HEP-SPEC06 benchmark. While it is difficult to quantify this, it does mean that an experiment will get more useful work out of a fully loaded machine than the HEP-SPEC06 measurement might suggest. This does not change the conclusion to pack jobs on to as few nodes as possible.

To measure the power usage of the batch farm overall, information was collected from the PDUs in each rack via snmp and aggregated via Cacti[9]. Each rack contains two PDUs and several chassis. Each chassis is connected to both PDUs. Measuring the power usage of the sum

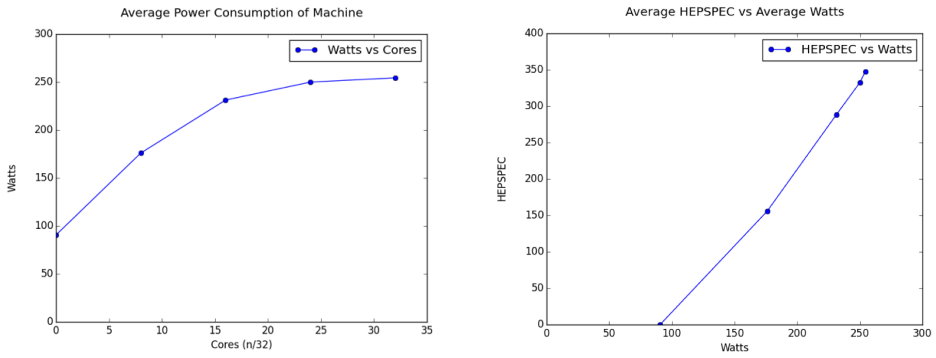

Figure 1. (left) Plot showing how the power usage increases as the number of threads (and hence cores) of the HEP-SPEC06 benchmarking are increased. (right) Plot showing how the measured HEP-SPEC06 changes as a function of energy used.

of the two PDU will give the power usage of each rack, but it is not possible to collect finer granularity information.

4. Hibernating idle machines

As described in the previous section, hibernating idle machines will provide a significant saving in power. Once every 5 minutes, the startd daemon checks the number of jobs a machine is running to determine if it is idle and if so for how long it has been in this state. It also checks how long it has been since the last hibernation. This is done via a shell script which parses the HTCondor logs looking for the last hibernation time. Finally, startd also checks that the machine is in a valid state to run jobs by checking the health status. For the machine to be hibernated it needs to be idle for at least 5 minutes and to have been awake for at least thirty minutes. This latter check prevents a machine from cycling between hibernating and up too quickly, potentially causing damage to the machine. It also must be able to run jobs, this prevents machines that may have been disabled for other reasons being hibernated.

Hibernating machines can be woken up via the HTCondor Rooster daemon[3]. To wake up hibernating machines, there needs to be jobs waiting in the queue for more than 15 minutes that could be run on the hibernating machines (i.e. HTCondor would have matched a job to that worker node if it had been awake). One machine is woken up per Rooster cycle which is configured to run every 5 minutes. The machine chosen is based on number of largest number of CPUs. Machines with more CPUs are newer so should be more energy efficient. The machine being woken up needs to have been hibernating for at least 30 minutes.

The automatic hibernation of idle machines was enabled at the start of December 2015. Over the Christmas period there was a dip in the number of jobs being sent to batch farm and some machines became idle. Figure 2 shows the used and idle capacity of the batch farm over this period as well as the energy usage of a generation of worker nodes. As the farm started to drain these worker nodes became idle and were powered down. A mistake in the configuration of the HTCondor Rooster (the configuration file was only put on one of the two head nodes) caused a delay in the worker nodes being woken up when the farm subsequently filled again.

One concern with repeatedly hibernating idle machines is that this may increase the chance of hardware failure. As a result only the older worker nodes - those that are over 3 years old have the ability to be hibernated enabled. This is around 40% of the worker nodes. The intention is to monitor the hardware failure rates over the next few years to see if hibernation has any impact.

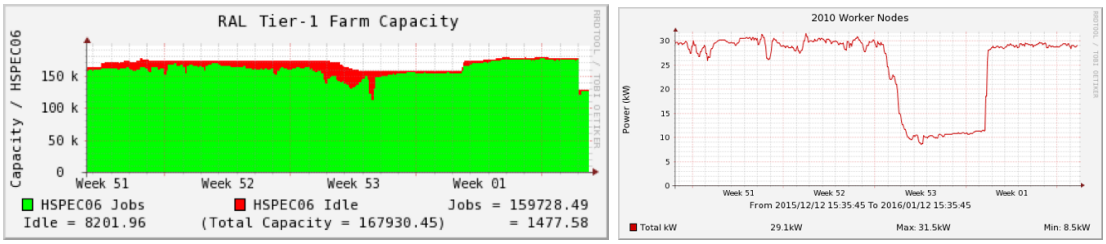

Figure 2. (left) Plot showing the total used and idle capacity of the Tier 1 batch farm (right) Plot showing how the measured HEP-SPEC06 changes as a function of energy used.

From the observed power saving shown in Figure 2, and from previous years data on the amount of time the batch farm is not completely full, we estimate that there is an energy saving of around 15000kWh a year from hibernating idle nodes.

5. Preempt-able jobs on draining worker nodes

The batch farm runs a mixture of single and multi-core (normally 8 core) jobs. HTCondor is configured to allow dynamic provisioning of multi-core jobs. If the batch farm is running at or near capacity, it is highly unlikely that several CPU slots will become available on the same node at the same time to allow a multi-core job to start. In this situation HTCondor needs to select some nodes to drain (i.e. stop starting new jobs) while it allows single core jobs to start on other nodes. Previously a node that was being drained would have wasted resources. With the introduction of preempt-able jobs these can be used to fill worker nodes while they are being drained. Worker nodes also need to be drained periodically so that they can be rebooted for Kernel and other updates.

Worker nodes are drained via a dedicated python script which replaces the HTCondor defrag daemon. When a worker node needs to be drained an attribute is added to the machine ClassAd so that only preempt-able jobs can be started. Once the required number of slots on the machine are either empty of running preempt-able jobs then the preempt-able jobs are killed.

When selecting a node to drain the aim is to identify the node that will drain the fastest. This minimises the wasted resources as well as allowing the multi-core job (which should have the highest priority) to run. The python script has an algorithm which it uses to decide which node should be drained. From the experience we have at STFC, the LHC experiments do not accurately specify the length of time they think a job will take. All jobs are normally submitted with the absolute maximum wall-time specified. Using this to calculate when jobs are likely to finish is not useful. Instead, a simpler algorithm which assumes that all jobs are equally likely to finish at any time is used. Therefore, the more running jobs there are relative to the number of slots required, the faster that machine is likely to be drain. This algorithm was modified to take into account the preempt-able jobs that are now being run:

$$\text{Rank} = \frac{[\text{Total Slots}] - [\text{Preemptable Jobs}] - [\text{Free Slots}]}{[\text{CPUs requested}] - [\text{Preemptable Jobs}] - [\text{Free Slots}]}.$$

Machines with the highest Rank will be drained next. If $[\text{CPUs requested}] - [\text{Preemptable Jobs}] - [\text{Free Slots}] \leq 0$ there is no need to drain a node (as their are enough slots already available) so the Rank will not be calculated.

Under normal operating conditions there are around 15 worker nodes ($\approx 0.5\%$ of batch farm capacity) being drained so that multi-core jobs can be started. Worker nodes are also rebooted around 6 times a year for kernel and security patching. To completely drain a worker node takes

around 36 hours. This has previously meant that up to 3% of the batch farm capacity is wasted due to worker nodes being drained. These resources can now be used by preempt-able jobs.

6. Killing jobs to reduce power usage

The final aim of the project was to see if we could reduce the energy usage on the batch farm rapidly (within 15 minutes) assuming we had received a signal as part of an energy saving scheme. A large number of test preempt-able jobs were submitted to the farm and allowed to start running. A script was then run which would kill all preempt-able jobs and prevent any more from starting. From the time the script was run it took just under 11 minutes for all the preempt-able jobs to be killed. It took under 20 minutes for the energy usage to drop to its lowest point. While some further optimisation may be required this does show that it should be possible to take part in schemes like this. While it was not monitored for this test, it can be assumed that any energy saving from a reduction in the amount of work being done by the air-conditioning would take significantly longer to manifest.

The average energy usage drop observed per job killed was 0.1W. The test was done when the batch farm was busy so jobs were being run on worker nodes which were effectively full. None of the nodes were full with preempt-able jobs, so none of them went into hibernation and the energy reduction was simply due to the fact that the worker nodes were less loaded. Figure 1 shows that this is the least effective way to save energy. Preempt-able jobs are preferentially matched to older hardware, which is the least energy efficient. In order to make significant savings, we estimate that at least 40% of the running jobs should be preempt-able. With this volume of preempt-able jobs the batch system can easily fill entire nodes with them, which allows for the maximum energy saving. Currently there are a limited number of ways experiments can take advantage of preempt-able slots but this is growing.

7. Summary and future work

This paper described the successful implementation of both preempt-able jobs and the hibernation of idle worker nodes on the STFC Tier 1 batch farm. The hibernation of idle worker nodes should save 15000kWh per year while the implementation of preempt-able jobs allows 3% of the batch farm's capacity that was previously being wasted to be utilised. This paper also described a test to reduce power usage on demand. While very small scale it did demonstrate that we could reduce the power usage within the 15 minute time frame requested. We will be encouraging experiments to submit more preempt-able jobs in future.

We will be monitoring hardware failure rates on the farm to see if the hibernation of machines causes it to increase. We also aim to run preempt-able jobs on the idle cloud resources available at STFC.

References

[1] WLCG: http://wlcg.web.cern.ch/
[2] HTCondor: https://research.cs.wisc.edu/htcondor/
[3] http://research.cs.wisc.edu/htcondor/manual/v8.4/3_15Power_Management.html
[4] Amazon Spot Pricing: https://aws.amazon.com/ec2/spot/
[5] Demand Side Balancing Reserve: http://www2.nationalgrid.com/UK/Services/Balancing-services/System-security/Contingency-balancing-reserve/
[6] http://www.theguardian.com/business/2015/nov/04/national-grid-issues-urgent-call-for-extra-power
[7] P Calafiura and [ATLAS Collaboration] 2015 The ATLAS Event Service: A new approach to event processing *J. Phys.: Conf. Series* **664** 062065
[8] HEP-SPEC06: https://w3.hepix.org/benchmarks/doku.php
[9] Cacti: http://www.cacti.net/