ATLANTIS PRESS

Advances in Engineering Research, volume 94

2nd International Conference on Sustainable Development (ICSD 2016)

An Approach to Energy Conservation for Large Scale Compute Systems

Yong-peng LIU*
School of Computer Science, National University of Defense Technology University,
Changsha, 410073, China
e-mail: liuyp@nudt.edu.cn, www.nudt.edu.cn

Wan-qing CHI
School of Computer Science, National University of Defense Technology University,
Changsha, 410073, China

Yong-yan LIU
Information Center, Ministry of Science and Technology,
Beijing, 100008, China
e-mail: liuyy@most.gov.cn

***Abstract*—Active idle nodes in large scale compute systems waste energy because they are not always fully loaded with applications. Noticeably, most modern nodesare equipped with multiple level dynamic sleep mechanisms. By scheduling the sleep states of such idle nodes, a new approach to reasonable energy conservation is proposed. When nodes are released from applications, they are put into appropriate levels of sleep states. To balance between energy consumption and system response speed, the sizes of different node groups with corresponding sleep state are dynamically adjusted according to the need of computation. The experiments demonstrated that our approach can significantly improve energy efficiency of large scale systems.**

Keywords-Large scale system; Energy conservation; Dynamic sleep; Idle node.

I. INTRODUCTION

A large scale compute system consumes a tremendous amount of energy. According to the recent list of fastest supercomputers [1], the average power consumption of Top10 systems is 8.88 MW. The peak power consumption of the most power consuming supercomputer, i.e. the Tianhe-2, reaches 17.808 MW, which equals the power usage of a middle scale town. The electricity consumption of global cloud computing is larger than the 5th largest electricity demand country in the world, i.e. India [2].

On the other hand, the workloads on large scale compute systems vary significantly with time and a large number of nodes are idle in most time [3]. A node can sleep when it is idle and be waken up when it is needed. The deeper anode sleeps, the less power it consumes, but the more time delay are needed to wake it up. In this paper, we propose an approach to managing sleep states of idle nodes in large scale systems to make an effective tradeoff between energy conservation and system response times.

II. RELATED WORKS

Dynamic cluster configuration is a typical power management technique based on node sleep mechanism. Put idle nodes into a sleep state and wake them up on demand. However, most researchers focus on allocating tasks on an appropriate active portion. The idle nodes are simply turned off [3].

Gandhi *et al* investigated the importance of multiple sleep states [4]. However, they do not dynamically manage the sleep depths of idle nodes. Horvath *et al.* [5] propose a policy that exploits the multiple sleep states of idle servers. They predicate the incoming workload and determine the optimal number of spare servers for each state. Differing from them, we adjust the number of nodes in each sleep depths in an adaptive manner rather than predications.

Xue *et al.* [6] also manage a pool of active resources whose size is adjusted dynamically according to the workload demand. However, their spare nodes are simply turned off. Our approach explores the benefits of multiple sleep states to minimize the sacrifices of system performance.

III. DYNAMIC ADJUSTING OF SLEEP STATES

In a large scale compute system, if a node has been allocated to any application, the node is *busy*. Otherwise, it is *idle*. To reduce the energy waste, idle nodes should be put into a low power state. The deeper a node sleeps, the less power it consumes, but the more latency is required to wake it up. Consequently, arbitrary putting an idle node into its deepest sleep state is not the best choice. In the paper, we propose an energy management approach to schedule the sleep or waken up timing of idle nodes.

In our approach, as Figure 1, the idle nodes are classified into several node groups according to their sleep depths. The group of level *i* is composed of all of the nodes with the same power consumption level.

In order to minimize the impact due to wake-up latency, we preferentially allocate nodes from the highest group as many as possible, which has the shortest wakeup latency. When the nodes in the group of the highest level are not sufficient, the nodes in the group(s) of next level(s) are allocated.

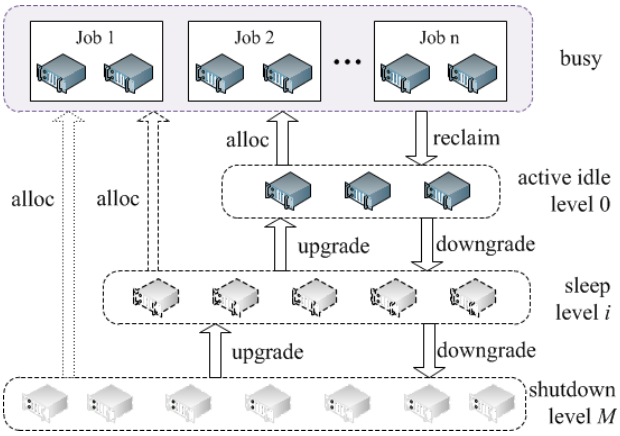

Figure 1. Dynamicstate transition of idle nodes.

Let *A* be the number of nodes required by an incoming application and the number of nodes in level *i* is *Ni*. The lowest level that can cover the need of A nodes is denoted by *Level(A)*. Formally, we have that

$$Level(A) = l \Leftrightarrow \left(\sum_{i=0}^{l-1} N_i < A \le \sum_{i=0}^{l} N_i \right) \quad (1)$$

The overall wakeup latencyforan allocation is determined by the largest wakeup latency of the allocated nodes. To satisfy the requirement on response time, enough nodes should be reserved in the groups whose sleep depths are lower than l. Therefore, we set a reserve capacity threshold, for each level of group to control its minimum number of nodes. Whenever the number of nodes is less than the threshold, nodes in the lower group will be upgraded to fill the reserve capacity.

When nodes are released by applications, they will be reclaimed as active idle. The number of active idle nodes becomes more than the required, hence some of them will be downgraded recursively to a sleep state as deep as possible.

Generally speaking, the less the reserve capacity thresholds, the more idle nodes are downgraded, and hence the more energy is saved. However, the wakeup latency is also more significant. Thus, a trade-off between energy consumption and performance must be made by assigning appropriate values to the thresholds. Because workload on large scale systems is unpredictably time varying, fixed values of the thresholds is not optimal. Therefore, our approach dynamically adjusts the threshold according to the difference between the allocation node requirement and the original reserve capacity.

IV. EVALUATION

In our evaluation, each idle node has four states: S0 is the active idle state, and S1, S3, S4 are sleep states ranking on sleep depth. Five different scenarios are simulated. In four of them, the idle nodes are always at a fixed state S0, S1, S3 and S4, respectively. The other scenario uses our proposed approach, denoted as "Our". The benchmarks are from Parallel Workload Archive [8].

The effects in the five management scenarios on the 8 systems are shown in Figure 2. In comparison with scenario S0, in which all idle nodes are active, on average of 8 workloads, our approach reduces overall energy consumption by 50.93%at the cost of increasing the average job execution time by 3.49%.The energy efficiency is improved by 49.32%atthe metric of EEIR [7].

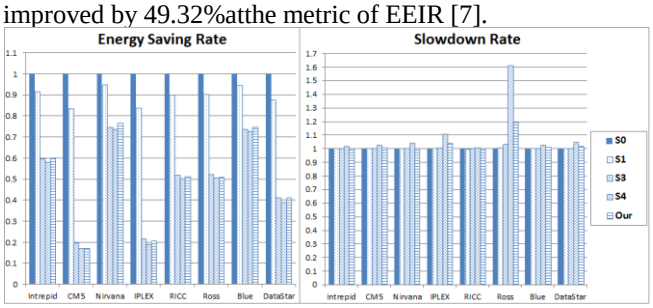

Figure 2. The results of different management scenarios.

In scenario S4, all idle nodes are always put into the deepest sleep state. Thus is consumes the least energy, i.e. 47.62% of S0. However, its job execution time is much higher, 111.25% of that in S0 scenario. In comparison with the S4 scenario, we improve the energy efficiency by 4.21%.

V. CONCLUSIONS

The paper proposes a new approach to balance between energy consumption and system response speed by dynamic adjusting the sleep depths of idle nodes. Idle nodes are classified into different groups according to their sleep depths. The nodes with lower sleep depth are allocated firstly. The state of an idle node is dynamically upgraded to a lower sleep depth group to be ready for wakeup with a shorter latency, or downgraded to a deeper sleep depth in order to save energy. Our simulation experiments demonstrated that our approach is an effective solution to optimize the system energy efficiency.

ACKNOWLEDGMENT

The research work was supported by the National High Technology Research and Development Program of China (863 Program) under grant No.2012AA01A301.

REFERENCES

[1] Top500. http://www.top500.org/. Jun. 2016.

[2] Gary Cook. *How Clean is Your Cloud?*. Greenpeace International. Apr. 2012.

[3] Andrew Krioukov, Prashanth Mohan, Sara Alspaugh, et al. *NapSAC: Design and implementation of a power-proportional web cluster*. ACM SIGCOMM Computer Communication Review, 2011, 41(1):102-108.

[4] Anshul Gandhi, Mor Harchol-Balter, Michael A. Kozuch. *The case for sleep states in servers*. Proc. of the HotPower '11, Cascais, Portugal, 2011.

[5] Tibor Horvath, Kevin Skadron. *Multi-mode Energy Management for Multi-tier Server Clusters*. Proc. of the 17th International Conference on Parallel Architecture and Compilation Techniques, Toronto, Canada, 2008:270-279.

[6] Zhenghua Xue, Xiaoshe Dong, Siyuan Ma, et al. *An energy-efficient management mechanism for large-scale server clusters*. Proc. of the 2007 IEEE Asia-Pacific Services Computing Conference. 2007:509-516.

[7] Yongpeng Liu, Hong Zhu, Kai Lu And Xiaoping Wang. *Self-Adaptive Power Management of Idle Nodes in Large Scale Systems*. International Journal of Next-Generation Computing, Vol. 2, No. 4, July 2013.

[8] Parallel Workloads Archive. http://www.cs.huji.ac.il/labs/parallel/workload/l_anl_int/ANL-Intrepid-2009-1.swf.gz. Apr. 2016