Dynamic Cluster Resource Allocations For Jobs With Known Memory Demands

D. A. Vidhate¹, A. K. Patil² and D. V. Guleria³
¹Head, ²Asstt.Prof. ³Lecturer, Dept. of Information Technology, P.D.V.V.P.C.O.E. Ahmednagar
E-mail: ¹dvidhate@yahoo.com, ²anitamdhoke@yahoo.com, ³deepa.guleria@gmail.com

ABSTRACT
The cluster system we consider for load sharing is a compute farm which is a pool of networked server nodes providing high-performance computing for CPU-intensive, memory intensive, and I/O active jobs in a batch mode. Existing resource management systems mainly target at balancing the usage of CPU loads among server nodes. With the rapid advancement of CPU chips, memory and disk access speed improvements significantly lag behind advancement of CPU speed, increasing the penalty for data movement, such as page faults and I/O operations, relative to normal CPU operations. Aiming at reducing the memory resource contention caused by page faults and I/O activities, we have developed and examined load sharing policies by considering effective usage of global memory in addition to CPU load balancing in clusters. This paper describes memory demands are known in advance or predictable. Conducting different groups of trace-driven simulations, we show that our proposed policies can effectively improve overall job execution performance by well utilizing both CPU and memory resources.

SUMMARY
We summarize our study on load sharing with known job memory demands as follows:
- The performance of a load sharing policy considering both CPU or memory resources is robust for all traces in this part of the study and is much better than the performance of a load sharing policy considering only CPU or only memory resource, particularly when the memory access interactions are intensive.
- The reason that CPU_MEM-based policies perform well is that these policies effectively reduce the paging time and queuing time. Meanwhile CPU policy suffers large paging overhead and MEM policy could not reduce queuing time.
- The high-performance approach is slightly more effective than the high throughput approach for all traces in this part.

CONCLUSION
This paper aims at providing effective load sharing strategies to improve overall performance of cluster systems by coordinating the usage of CPU and memory resources. We first evaluate our strategies on a real workload with an assumption that jobs memory demands are known in advance. We show that the proposed load sharing policies considering both CPU and memory resources are robust & significantly outperform resources, particularly when the memory access interactions are intensive. Our trace-driven simulations consistently show the effectiveness of the policies by comprehensively considering the online information of CPU memory & I/O resources.

REFERENCES