PDRM: A Probability Distribution Based Resource Management for Batch Workloads in Heterogeneous Cluster

Jun Zhou[1,2,3], Dan Feng[1,2,3](✉), and Fang Wang[1,2,3]

[1] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China
{JunZhou,dfeng,wangfang}@hust.edu.cn
[2] Wuhan National Laboratory for Optoelectronics, Wuhan 430074, China
[3] Key Laboratory of Information Storage System, Engineering Research Center of data storage systems and Technology, Ministry of Education of China, Wuhan, China

Abstract. Resource consumption prediction and dynamic resource provision based on historical consumption are common methods to improve cluster resource utilization, however they have to face the challenge of fluctuation in resource consumption for accurate prediction. We propose PDRM, an efficient resource management scheme based on resource consumption probability distribution for batch workloads to deal with this dilemma. Based on the common sense that the same type of tasks have similar resource consumption on the same node, we get the resource consumption probability distribution of each type of task to describe the fluctuations in its resource consumption. Based on the resource consumption distribution function, we can allocate resources precisely for tasks. Experimental results demonstrate that PDRM achieves good performance for various application in the heterogeneous cluster. PDRM can effectively improve resource utilization and reduce job completion time.

Keywords: Resource management · Big data · Gaussian distribution · Heterogeneous

1 Introduction

Low resource utilization is a common issue in cloud platforms. Reiss et al. [5] shows that a Google cluster achieves CPU utilization of 25–35% and memory utilization of 40%. Quasar [3] indicates that the CPU utilization is consistently below 20%, and the memory utilization is (40–50%) on a production cluster at Twitter. Fluctuation of resource consumption and complex heterogeneous environment bring much more challenges to resource allocation in cloud cluster. It is difficult to match resource allocation precisely with resource consumption, and resources are usually over allocated to guarantee task execution.

Published by Springer Nature Switzerland AG 2019
X. Tang et al. (Eds.): NPC 2019, LNCS 11783, pp. 361–365, 2019.
https://doi.org/10.1007/978-3-030-30709-7_34

In this paper, we propose PDRM, a resource management scheme based on task resource consumption probability distribution. The main idea of PDRM is to adopt the probability distribution of resource consumption to quantify the fluctuation of resource consumption for accurate resource allocation, so as to improve resource utilization and reduce task running time. We evaluate PDRM, and results show that it improves job execution efficiency and resource utilization.

2 Related Work

Several recent researches have tried to address the issue of improving efficiency in allocating resources to applications with varying degree of success. (1) **Dynamic Resource Provisioning.** Mohan et al. [4] have proposed dynamic resource management solutions for applications. These research works are used for resource management of long term services, but not suitable for batch workloads. However the batch workload, which consists of a large number of short tasks usually completed in minutes or seconds, is too important to be ignored. (2) **Resource Provisioning with an Appropriate Configuration.** CherryPick [1] builds a performance model with Bayesian Optimization to distinguish the optimal or a near-optimal configuration from the rest. MrMoulder [2] adopts optimization technique to tuning Hadoop configuration parameter settings. They mostly focus on improving of application performance and pay less attention to the resource utilization of the cluster. (3) **Harvesting Idle Resource from Colocated Jobs.** Zhang et al. [6] schedules related batch tasks on servers to colocate with latency-critical jobs. The main idea is harvesting idle resource from other jobs, but it can't harvest idle resources from the job itself. (4) **Characterizing and Classifying Workloads.** Quasar [3] classifies any new incoming application, assign the application proper resources in a datacenter. Classification techniques cannot fully reflect the differences in resource consumption of various tasks.

3 Motivation

Existing research assumes that the resource consumption of the task is the same as the historical consumption. However, the resource consumption of repeated tasks will fluctuate instead of being absolutely the same. The main reason for fluctuations in resource consumption is that the complexity of the algorithm on different input data content is different. It may cause deviations in resource consumption prediction. Therefore, we have conducted in-depth research on the similarity of resource consumption. We repeat running a variety of different batch workloads with different data sets. We extract resource consumption for all types of tasks when the application is running. It shows that the resource consumption of the same tasks on the same cluster node are similar, and the resource consumption fluctuates within a certain range. By counting the number of tasks in different resource consumption intervals, we can obtain the task

resource consumption probability distribution. The probability distribution of resource consumption is in accordance with the Gaussian distribution, and the Gaussian distribution is fitted well.

4 PDRM Design

By extracting resource consumption of big data applications, we can get the task consumption probability distribution. Based on the distribution function of resource consumption, we propose an accurate resource allocation scheme, called PDRM.

We use $Task_0$ to denote a type of task, the resource allocation vector of $Task_0$ is expressed as $[ra_{10}, ra_{20}, ..., ra_{m0}]^T$ and the resource consumption vector of $Task_0$ is expressed as $[rc_{10}, rc_{20}, ..., rc_{m0}]^T$, where m is the total number of the types of resource, $ra_{i0}(i \in N, 1 \leq i \leq m)$ is the amount of the class i resource allocated to $Task_0$ and $rc_{i0}(i \in N, 1 \leq i \leq m)$ is the class i resource consumed by $Task_0$. We perform a Gaussian fitting on the class i resource consumption of $Task_0$, and the Gaussian distribution satisfied by rc_{i0} is expressed as $N_{io}(\mu_{io}, \sigma_{io}^2)$, where μ_{io} is the mean of rc_{i0} and σ_{io}^2 is the variance of the probability distribution of rc_{i0}. The cumulative distribution function of the probability distribution of rc_{i0} is

$$F_{i0}(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi\sigma_{i0}^2}} e^{-\frac{(x-\mu_{i0})^2}{2\sigma_{i0}^2}} dx. \tag{1}$$

The probability that rc_{i0} is less than ra_{i0} can be denoted as $P_{i0} = F_{i0}(ra_{i0})$. The success ratio of $Task_0$ can be expressed as $min\{P_{10}, P_{20}, ..., P_{m0}\}$. Conversely, only when the resource allocated for the class i resource ra_{i0} is not less than $F_{i0}^{-1}(P_{\text{success}})$, the success rate of $Task_0$ could reach P_{success}.

The resource allocation of class i resource for $Task_j$ is ra_{ij}. P_{success} is the probability that $Task_j$ can be successfully completed. The failed task will be restarted with the default resource allocation which is much larger than the actual consumption of the task to ensure successful execution. The Expectation resource allocation of class i resource for $Task_j$ expressed as $E(ra_{ij}) = ra_{ij} + (1 - P_{\text{success}})ra_{ij\_default}$, where $ra_{ij\_default}$ is the default resource allocation. The average resource utilization of class i resource on a node is $\overline{ut_i} = \frac{\sum_{j=1}^{n} \mu_{ij}}{\sum_{j=1}^{n} E(ra_{ij})}$.

When the derivative of $\overline{ut_i}$ is 0, $\overline{ut_i}$ takes the maximum value. By solving the formula $(\overline{ut_i})' = 0$, we can get the solution of optimal resource utilization and set the values of P_{success}. Then, we get the resource allocation vector for each type of task.

5 Evaluation

We implement PDRM as a component on Hadoop Yarn. In this section, we demonstrate the effectiveness of our approach on a heterogeneous cluster.

We choose four representative applications on Hadoop to show different resources requirement: Terasort, WordCount, TextSearch, and TriangleOfOriented. We select 6 physical nodes to build a heterogeneous environment, named NODE0-NODE5. NODE0 is the master node, NODE1-NODE5 are the slave nodes. NODE0-NODE3 have the same physical configuration (two Intel Xeon E2620 6x cores 2.1 GHz CPUs, 16 GB memory). The number of virtual cores available for container allocation on each node is 8. As a comparative instance of CPU heterogeneity, NODE4 has two Intel Xeon E5620 4x cores 2.4 GHz CPUs. As a comparative instance of Memory heterogeneity, the memory available for container allocation on NODE5 is 3 GB, while the memory available on other nodes are 8 GB.

5.1 Job Completion Time of Heterogeneous Applications

In this experiment, we evaluate the effectiveness of PDRM for reducing job completion time. Figure 1 shows the job completion times with different resource allocation schemes. It can be observed that PDRM reduces the job completion time by 30.4%, 24.3%, 25.1%, 24.7% compared to the default for Terasort, WordCount, TextSearch, TriangleOfOrineted, respectively. PDRM resource allocation schemes can effectively reduce job completion time.

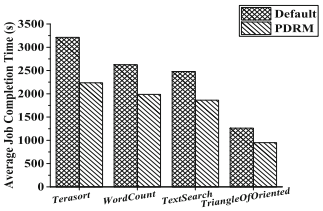

Fig. 1. Job completion time.

5.2 Resource Allocation Ratio and Resource Consumption Ratio of Heterogeneous Cluster

In this experiment, we compare the resource allocation ratio and the resource consumption ratio of different nodes in heterogeneous clusters under the default and PDRM. We normalize the node available resources to 1. We run Terasort on the heterogeneous cluster, the resource allocation ratio and resource consumption ratio of NODE1, NODE4 and NODE5 are shown in Figs. 2 and 3, respectively. It can be seen that the resource allocation ratio of PDRM is less than the that of the default, but the resource consumptions ratio of PDRM are greater that of the default.

The performance of NODE1 and NODE4 are limited by the CPU resources. With the PDRM resource allocation scheme, the CPU resources of NODE1 and NODE4 can achieve higher utilization. The CPU processing power of NODE4 is lower than that of NODE1. The NODE4 CPU can maintain high utilization, but the NODE4 memory resource utilization is less than NODE1. The performance of NODE5 is limited by the memory resources. NODE5 has less memory resources than NODE1. With PDRM resource allocation scheme, the memory resources of NODE5 can achieve higher utilization, far greater than that of NODE1. PDRM can improve resource utilization, and the scarce resources of nodes can be efficiently utilized in heterogeneous clusters.

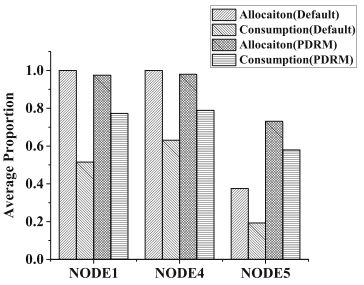

Fig. 2. CPU resource ratio.

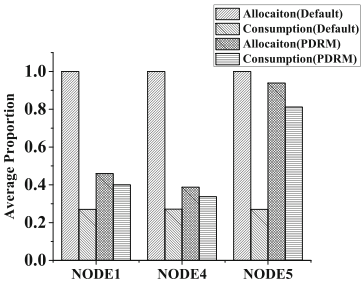

Fig. 3. Memory resource ratio.

6 Conclusion

The resource consumption probability distribution of the task can well describe the fluctuation of resource consumption. We propose PDRM, a resource allocation scheme based on the probability distribution of task resource consumption. Through experimental verification, PDRM can reduce job completion time by over 25%. What's more, PDRM can minimize the gap between resource allocation and resource consumption, and make efficient use of scarce resources in heterogeneous clusters.

References

1. Alipourfard, O., Liu, H.H., Chen, J., Venkataraman, S., Yu, M., Zhang, M.: CherryPick: adaptively unearthing the best cloud configurations for big data analytics. In: 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 2017), pp. 469–482. USENIX Association, Boston (2017)
2. Cai, L., Qi, Y., Wei, W., Wu, J., Li, J.: mrMoulder: a recommendation-based adaptive parameter tuning approach for big data processing platform. Future Gener. Comput. Syst. **93**(1), 570–582 (2019)
3. Delimitrou, C., Kozyrakis, C.: Quasar: resource-efficient and QoS-aware cluster management. In: Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 127–144. ACM, New York (2014)
4. Mohan, A., Kaseb, A.S., Lu, Y., Hacker, T.: Adaptive resource management for analyzing video streams from globally distributed network cameras. IEEE Trans. Cloud Comput. 1 (2018)
5. Reiss, C., Tumanov, A., Ganger, G.R., Katz, R.H., Kozuch, M.A.: Heterogeneity and dynamicity of clouds at scale: Google trace analysis. In: Proceedings of the Third ACM Symposium on Cloud Computing, pp. 7:1–7:13. ACM, New York (2012)
6. Zhang, Y., Prekas, G., Fumarola, G.M., Fontoura, M., Goiri, I.n., Bianchini, R.: History-based harvesting of spare cycles and storage in large-scale datacenters. In: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation, pp. 755–770. USENIX Association, Berkeley (2016)