J. Parallel Distrib. Comput. 73 (2013) 926–938

Contents lists available at SciVerse ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Job scheduling with adjusted runtime estimates on production supercomputers

Wei Tang [a,*], Narayan Desai [b], Daniel Buettner [b], Zhiling Lan [a]

[a] Illinois Institute of Technology, Chicago, IL 60616, USA
[b] Argonne National Laboratory, Argonne, IL 60439, USA

HIGHLIGHTS

- We studied the inaccuracy of user runtime estimates in large amount of job traces.
- We proposed a set of runtime adjusting schemes to better the estimation accuracy.
- We refined our schemes to avoid impact of too much adjusting (underestimates).
- We used real job trace to evaluate our schemes and got positive results.

ARTICLE INFO

Article history:
Received 29 August 2011
Received in revised form
4 February 2013
Accepted 12 February 2013
Available online 6 March 2013

Keywords:
Job scheduling
Runtime estimates
Walltime prediction

ABSTRACT

The estimate of a parallel job's running time (walltime) is an important attribute used by resource managers and job schedulers in various scenarios, such as backfilling and short-job-first scheduling. This value is provided by the user, however, and has been repeatedly shown to be inaccurate. We studied the workload characteristic based on a large amount of historical data (over 275,000 jobs in two and a half years) from a production leadership-class computer. Based on that study, we proposed a set of walltime adjustment schemes producing more accurate estimates. To ensure the utility of these schemes on production systems, we analyzed their potential impact in scheduling and evaluated the schemes with an event-driven simulator. Our experimental results show that our method can achieve not only better overall estimation accuracy but also improved overall system performance. Specifically, the average estimation accuracy of the tested workload can be improved by up to 35%, and the system performance in terms of average waiting time and weighted average waiting time can be improved by up to 22% and 28%, respectively.

1. Introduction

In a supercomputing systems, the job runtime estimate, also called requested walltime, is an important job attribute provided by users at job submission. Although this value was originally used by resource managers to kill a job at its expiration, the value is also heavily used in job scheduling. Backfilling [15], for example, needs to know the expected runtime of both running and waiting jobs so that it can fill short jobs into backfilling windows, reducing fragmentation without delaying high-priority jobs. Some schedulers favor short jobs in order to achieve improved average response time [23]; they need to know the runtime estimates of the waiting jobs when sorting the queue. Moreover, job runtime estimates are essential to other resource management strategies, such as advance reservation [11], queuing time prediction [7,20], and walltime-aware job allocation reducing fragmentation on torus-connected systems [24].

However, user estimates of job running time have been repeatedly demonstrated to be highly inaccurate [3,30,2]. Indeed, a large number of jobs consume only a small portion of the walltime requested. A number of studies have been done to investigate whether such inaccuracy can impact job scheduling performance. Surprisingly controversial results have been reported. On one hand, some claimed inaccuracy is helpful. For example, Mu'alem et al. [15] reported that the inaccurate runtime estimates have the potential to be beneficial because of backfilling; such results have led to the suggestion that estimates should be doubled [34] or randomized [17] to make them even less accurate. On the other hand, some others suggested accuracy is more favorable. Studies have shown that using more accurate runtime estimates can improve system performance far more significantly than previously suggested [2,21,28].

In this paper, we present a set of walltime adjustment schemes that can be used by large-scale production systems directly. First,

* Corresponding author.
E-mail address: wtang6@iit.edu (W. Tang).

0743-7315/$ – see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jpdc.2013.02.006

we studied workload characteristics based on a large amount of historical data (275,000 jobs in 30 months) from a leadership-class computer. Next, we proposed a set of walltime adjustment schemes to produce more accurate estimates, and we discussed how to configure each scheme for real computer systems. We evaluated the performance of our walltime adjustment schemes on production machines using simulations with real workloads. Our experimental results show that our method can achieve not only better overall estimation accuracy but also improved overall system performance. Specifically, the average and median of estimation accuracy of the tested workload can be improved by up to 35% and 42%, respectively. Moreover, the system performance in terms of average waiting time and weighted average waiting time can be improved by up to 22% and 28%, respectively.

In this paper, several terms regarding job runtime are used repeatedly. For example, we use job *actual runtime* (t_{act}) for job execution time. We use *user-requested walltime* (t_{req}), or simply walltime, to represent the runtime estimates provided by users at job submission; the resource manager kills jobs when this time expires. We use t_{sched} to represent the job walltime used by scheduler for prioritizing and backfilling jobs. Usually, t_{sched} equals t_{req}; but in this work, t_{sched} can be other adjusted values. In this context, the term "walltime adjustment" refers to the effort of a system to adjust the user's estimates to create possibly more accurate walltime estimates. The term "walltime estimate" refers to the runtime estimate either provided by users or adjusted by the system.

The remainder of this paper is organized as follows. Section 2 discusses some related work. Section 3 presents our study of historical job traces. Section 4 presents our walltime adjustment schemes and analytical evaluation. Section 5 presents our analysis of the impact of imperfect prediction and an enhancement for utilizing walltime adjustment. Section 6 presents a performance evaluation of scheduling using enhanced walltime adjustment. Section 7 summarizes our conclusions.

2. Related work

In this section we review some related studies that have focused on various aspects of runtime estimation, including accuracy and impact on job scheduling, and we present schemes for improving the accuracy.

2.1. Inaccuracy of user estimation

User-provided runtime estimates are known to be inaccurate. For example, Cirne and Berman [3] showed that in four different traces, 50%–60% of jobs used less than 20% of their requested time. Ward et al. [30] reported that jobs on the Cray T3E used on average only 29% of their requested time. Chiang et al. [2] studied a certain workload and found that users grossly overestimated their job runtime, with 35% of jobs using less than 10% of their requested time. Similar patterns are seen in other workload analyses [15,21]. We studied a large amount of data from a production Blue Gene/P system [1] and found that although the accuracy is better than previously reported, the user estimates are still highly inaccurate: half the jobs use less than 50% of their requested walltime.

2.2. Impact of user runtime estimates on job scheduling

Considerable work has been done on backfilling job scheduling and the dependence on runtime estimation. Many results suggest that using more accurate requested runtime has only minimal impact on system performance [15,20,34]. Additional results in [15] show that doubling the user-requested runtime slightly improves, on average, the slowdown and response time for IBM SP workloads using FCFS–backfill. Other results are conflicting. Chiang et al. [2] examined this question on the NCSA Origin 2000 (O2K) and showed that more accurate requested runtime can improve system performance much more significantly than suggested in previous studies. Srinivasan et al. [21] studied the effect of various backfilling schemes on different priority policies and observed that inaccurate estimates can significantly deteriorate the overall performance. On the other hand, Tsafrir et al. [29,26] offered two reasons that inaccuracy could better the scheduling: (1) an invalid model (F-model) is used in modeling user estimates, and (2) the improved performance is due to the "heel and toe" effect—that is, the FCFS–backfilling is switched into a short-job-first type of policy. Zhang et al. [33] showed that even though the average job behavior is insensitive to the average degree of overestimation, individual jobs can be affected; under common backfilling schemes, users who provide more accurate runtimes are favored over ones that do not.

Our work also reveals the relationship between the accuracy and scheduling performance, but we do not use artificial or modeled inaccuracy. Instead, we evaluate the performance change using the workload after applying real walltime adjustment schemes, where the requested walltime is adjusted based on the real inputs.

2.3. Motivation for improving user accuracy

Besides the goal of improving the job scheduling performance directly [22,28], a wide range of related work shares the common motivation of improving the accuracy of runtime estimation. One example is advance reservations for grid allocation and collocation, shown to benefit considerably from better accuracy [11,20,14]. Another is scheduling moldable jobs that may run on any number of nodes [7,20,4]. The scheduler's goal is to minimize response time, considering whether waiting for more nodes to become available is preferable over running immediately. Thus, a reliable prediction of how long it will take for additional nodes to become available is crucial. Recently, Yuan et al. [32] proposed PV-EASY backfilling; this scheme, used to guarantee strict fairness and protect the interests of blocked top-priority jobs, also needs more accurate walltime estimates. Similarly, our previous work [24] proposed a walltime-aware job allocation strategy that tries to pack jobs with similar size and length together to reduce fragmentation on torus-connected system; its performance improves with better walltime estimation accuracy.

Our primary motivation is to better the backfilling and queue sorting. But, with more accurate estimates, our schemes benefit all the other motivating problems that requiring more accurate estimates.

2.4. Efforts to improve user estimations

Numerous efforts have been devoted to improving the accuracy of user runtime estimates. Lee et al. [13] tried to improve user estimation by removing the threat of job killing at walltime expiration and providing tangible reward for accurate estimates. However, experiments showed that their method leads to only insubstantial improvement in the overall average accuracy. Considerable research has focused on using system-generated prediction to better the estimation accuracy. Suggested prediction schemes include using the top of a 95% confidence interval of job runtime [8], a statistical model based on the (usually) long uniform distribution of runtime [7], using the mean plus 1.5 standard deviations, genetic algorithms [20,19], instance-based learning [18], rough set theory [12], and three-phase adaptive prediction [9,10]. Tsafrir et al. [28] proposed a runtime predictor that averages the runtime of the last two jobs by the same user. Wu et al. [31] proposed an adaptive hybrid method (AHModel) for Grid load prediction within confidence windows.