

Efficient Management of Idleness in Systems

Ningfang Mi
College of William and Mary
ningfang@cs.wm.edu

ABSTRACT

Temporal dependence has been identified as an important characteristic in workloads processed by multi-tier architectures, disk drives, and communication networks. In this paper, we discuss how to use the knowledge of temporal dependence in flows to forecast the length of idle intervals in storage systems. We design a new background scheduling policy to determine when and for how long idle times can be used for serving background tasks, without violating pre-defined performance targets of foreground jobs. Our analysis shows that if idle times have low variability, then it is not necessary to idle wait before starting a background job. Only if idle times are highly variable, then idle waiting is necessary to minimize the impact of background activity on foreground performance. We further show that if there is burstiness in addition to high variability in idle intervals, then it is possible to predict accurately the length of incoming idle times and use that information to serve more background jobs without affecting foreground performance.

1. INTRODUCTION

As computer systems operate 24 hours a day, 7 days a week, it is becoming common to schedule maintenance jobs during idle times. These jobs are considered *background jobs*. Background jobs are used for improving system reliability, availability, and consistency, to enhancing system performance. Completion of background jobs is critical to system operation, yet their priority is not as high as that of foreground jobs, i.e., the regular jobs of the system users. In addition, scheduling of background activities should not compromise the performance of foreground jobs.

We focus on the general problem of how to bin-pack non-preemptive background jobs during system idle times. Idleness is considered as an additional system resource that needs to be effectively managed but there is a trade-off between maximizing the completions of non-preemptive background jobs while minimizing their impact on foreground performance. Previous work focuses on monitoring the performance of foreground and background job to base background scheduling decisions. Idle waiting has been proposed to delay scheduling of a background job during an idle interval. This delay is fixed and equal to the average demand of a background job.

Here, in addition to monitoring the performance of fore-

ground and background jobs, measurements of the empirical distribution of idle times are also collected. Resource management of idle times is now done in a dynamic way, using statistical information not only on the foreground and background job demands, but also on the idle intervals of the system. This statistical information is collected online while the system is in operation, and is incorporated on-the-fly into scheduling policies. Detailed analysis of various systems with different statistical characteristics of foreground/background jobs and idle times shows that the effectiveness of idle wait critically depends on the variability in the empirical distribution of idle times. In systems with low variability of idle times, limiting the idle wait to zero is beneficial for system performance, the opposite holds for idle times of high variability. We use the cumulative data histogram of idle times to dynamically determine the length of idle wait. Additionally, we show that estimating the number of background jobs to be served in any given idle interval is an effective way to meet foreground performance targets. Furthermore, we show how to take advantage of the burstiness (if any) in idle intervals to improve background scheduling.

2. SCHEDULING BACKGROUND JOBS

To evaluate the effectiveness of idle waiting, the following scheduling policies are considered.

Mean-based: This serves as a base-line comparison. When an idle interval occurs, no background job is scheduled during a delay period which is statically defined as the mean service time of background jobs. After the delay period elapses, the system starts serving background jobs until a foreground job arrives.

CDF-based: Similar to mean-based, this policy starts serving background jobs after an idle wait until a foreground job arrives. Different from the mean-based policy, the CDF-based policy continuously monitors the empirical cumulative histogram of idle intervals and the mean of background service times to dynamically calculate the idle wait. If idle times have low C.V., then the system does not idle wait, i.e., it schedules background jobs immediately. If idle times have high C.V., then the idle wait is calculated based on the empirical CDF.

CDF/w-estimates: This policy estimates the idle wait the same way as the CDF-based policy but is more conservative by limiting the number of background jobs to be served in an idle interval according to the following equation:

$$T \cdot \frac{90^{th} \text{ percentile of idle intervals} - \text{idle wait}}{\text{Average background service time}}$$

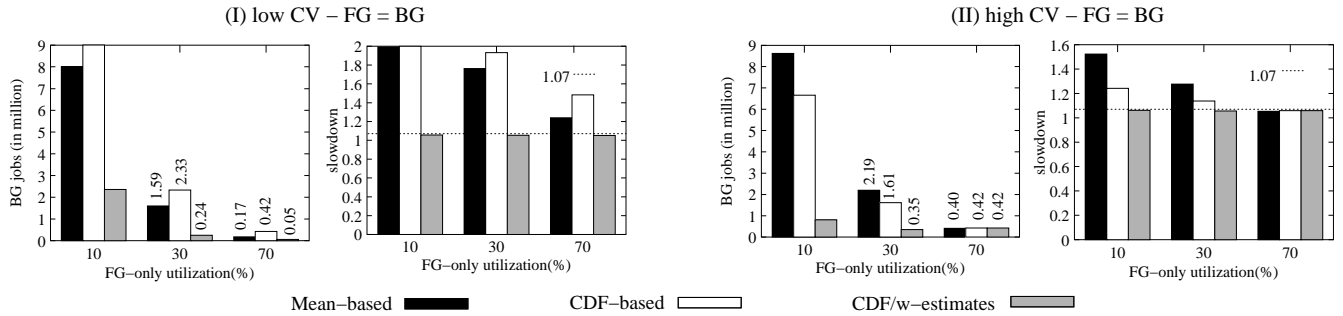


Figure 1: Overall system performance measured by number of completed background jobs in millions and slowdown of the foreground jobs attributed to background activity (the horizontal line corresponds to 7% slowdown).

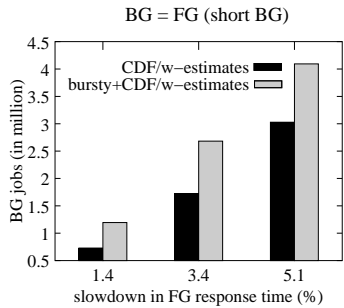


Figure 2: Number of completed background jobs when under bursty idle intervals. Three different foreground slowdowns are considered, i.e., 1.4%, 3.4% and 5.1%.

T is a parameter that weighs the estimated number of background jobs assuming that the interval is large (i.e., equal to the 90th percentile). This parameter controls the performance degradation of foreground jobs, i.e., the impact on foreground performance increases as T increases. The maximum value given to T is 1. This parameter is self-adjusted to reflect variability in the distribution of idle intervals. T is close to 1 under idle intervals of high variability and less than 1 for low variability intervals.

All of the above policies are non-preemptive. The three policies are evaluated via simulation of a single server queue. We assume that there is no limit on the waiting queue capacity and the service process is FCFS. We also assume that there are *always* background jobs waiting for service. The acceptable slowdown of foreground jobs due to background jobs is set to 7%. Service times of background jobs are exponentially distributed and service times of foreground jobs are drawn from a Lognormal distribution. Both foreground and background jobs have the same mean service time.

Figure 1 illustrates the performance of the three policies under idle intervals of low variability and of high variability. The graph shows that the CDF/w-estimates policy consistently meets the performance target of foreground jobs while serving a large number of background jobs. Under low foreground-only utilizations there is more room to exploit idle times and serve large quantities of background jobs with small foreground performance degradation. Similar results are observed for systems where the average background time is larger (as much as seven times) than the average foreground time [1].

When there is burstiness in addition to high variability in idle intervals, the CDF/w-estimates policy is further augmented into a new version called Bursty+CDF/w-estimates

policy: predict the length of incoming idle times and serve more background jobs if the next idle interval is predicted to be “long”. Figure 2 presents the number of completed background jobs as a function of foreground slowdown when short background jobs are served. There are more background jobs completed with Bursty+CDF/w-estimates than with CDF/w-estimates. The relative performance gap between the two policies increases as foreground slowdown decreases. If the requirements on foreground slowdown are relaxed, then the difference between the two policies diminishes. In general, overall system utilization improves with Bursty+CDF/w-estimates.

3. SUMMARY

We show that monitoring the stochastic characteristics of idle times is as important as monitoring the characteristics of foreground and background jobs. In particular, if idle intervals have low variability, then idle waiting is not effective. However, if idle times are highly variable, then idle waiting remains effective for scheduling background jobs without delaying foreground ones. For idle intervals with high variability, we propose to compute the length of idle wait dynamically using the cumulative histogram of the observed idle times in the system.

Apart from managing effectively idle intervals by distinguishing between low and high variability, we have also identified correlation as a source of additional information to improve idle time utilization. The analysis shows that if correlation exists in the observed idle interval lengths, then it can be used to predict the near future. For more details, we direct the reader to [1], where we also show the effectiveness of the methodology using actual measurements from disk drive traces. The proposed scheduling policies can be used to schedule background activities, scrubbing and intra-disk parity updates, and dramatically improves storage system reliability while keeping average user performance degradation within pre-defined targets [2].

4. REFERENCES

- [1] N. Mi, A. Riska, Q. Zhang, E. Smirni, and E. Riedel. Efficient management of idleness in systems. Technical report, WM-CS-2008-01, William and Mary, 2008.
- [2] E. Smirni, N. Mi, A. Riska and E. Riedel. Enhancing data availability through background activities. In *DSN*, 2008.