Integrated Processor Scheduling for Multimedia

Jason Nieh and Monica S. Lam

Computer Systems Laboratory
Stanford University
Stanford, CA 94305

Sun Microsystems Laboratories
2550 Garcia Avenue, MTV29-110
Mountain View, CA 94043

The advent of multimedia ushers forth a growing class of applications that must manipulate digital audio and video within well-defined timeliness requirements. Existing processor schedulers are inadequate in supporting these requirements. They fail to allow the integration of these continuous media computations with conventional interactive and batch activities. By observing that all tasks do in fact have timeliness requirements, we have created a new scheduler that provides integrated processor allocation for all classes of computational activities. By exploiting information on the relative importance of different activities to the user, the scheduler seeks to maximize the value the system delivers to the user, and gracefully degrades activities in order of importance when all timeliness requirements can not be met. The solution is unique in the degree to which it allows users control over the allocation of processing resources, and in that it provides an integrated scheduling solution in supporting multimedia applications.

1 Introduction

Applications that manipulate digital audio and video represent a new class of computations executed by workstation users. This new class of computations is known as *continuous media*. Continuous media activities are characteristic of applications that manipulated sampled digital media, such as television or teleconferencing. These often compute-intensive activities must process and transport media samples within well-defined timeliness requirements. Their integration into the workstation environment requires that the operating system manage resources to meet their time constraints, while at the same time supporting the interactive and batch activities found in conventional applications today. The advent of multimedia should not reduce the workstation to a single function system, like an embedded system or single-tasking PC. Instead, the workstation operating system must manage resources in such a manner that all classes of applications can continue to function correctly. In particular, as processor cycles are often the most oversubscribed resource, effective processor scheduling is of paramount importance.

Anticipating that processor scheduling based on traditional timesharing would not be suitable for the support of multimedia applications, attempts have been made to adapt static real-time schemes to support the timeliness requirements of multimedia applications [6][11][13]. Not only do they rely on static predictability and forced low resource utilization, but they require painful hand-tuning by the user to allow conventional interactive and batch activities to run. These ailments make such schemes unacceptable for the highly dynamic multimedia environment.

Because of the difficulty of scheduling conventional interactive and batch activities together with real-time continuous media activities, proposed solutions with actual implementations have predominantly been two-level schedulers [6]. These schedulers support separate scheduling policies for conventional tasks and real-time tasks on top of a base-level scheduling mechanism [1][3][4]. No matter how sophisticated the policies may be, the scheduler's overall effectiveness is limited by a typically static base-level mechanism that artificially constrains the range of behavior the system can provide. Such static schemes of resource partitioning are notoriously ineffective at best, and at worst can lead to pathological behavior, with runaway real-time activities causing basic system services to lock up and the user losing control over the machine [12].

2 An Overview of our Scheduler

As existing schemes have proven inadequate, a new approach to processor scheduling is required to provide support for multimedia applications. Our solution to the difficult problem of how to schedule time-critical continuous media activities with conventional activities is to allow *time constraints* to be provided on both types of activities. For those activities that have easily defined deadlines, their time constraints can be simply specified as in traditional dynamic deadline real-time systems. Many activities cannot be assigned meaningful deadlines in any reasonable manner. However, since all activities must make forward progress to be of any value to the users, a reasonable means of specifying the time constraints for such activities is a minimally acceptable rate of forward progress. This desired (minimum) computation rate is expressed in terms of the desired fraction of the system's processing resources across a given interval of time. This rate specification can be translated into a series of deadlines at run time. With the concept of minimum execution rates, all computational activities, whether they be continuous media, interactive, or batch, can be scheduled along common lines. Our scheduler attempts to meet the given time constraints at all times using an earliest-deadline first discipline.

While time constraints are sufficient when the system is not overloaded, more user input are desirable to control the scheduler's behavior otherwise. The user specify an activity's *priority*. If the system is overloaded, lower priority activities are shed to allow higher priority activities to meet their time constraints. Activities of the same priority, on the other hand, may both run at degraded performance. The user can bias the resource allocation decisions by specifying different *resource shares*.

The scheduler does not demand information from the user which is impractical (or perhaps impossible) for the user to provide. Default scheduling parameter values are assigned whenever an activity does not provide an explicit value; an activity which does not provide any information to the scheduler is placed at the median priority level, and is assigned to have the highest rate of progress. If all activities assume these default parameters, the scheduler behavior defaults to a fixed, equal quantum, round-robin scheduler. In addition, with this new scheduling facility it is possible to achieve the desirable aspects of such scheduling policies as: static priority, dynamic deadline, fair share, etc.

This solution is unique in the degree to which it allows users control over the sharing of processing resources, and in that it provides an integrated solution to scheduling computational activities with and without well-defined time constraints, such as those found in multimedia applications. We have implemented it in the Solaris operating system [2] to demonstrate its effectiveness on real applications on real systems. Due to space constraints, our scheduling algorithm and implementation results are not presented here; the algorithm and experimental results can be found in [12].

3 Conclusions

This paper introduces a novel solution to the difficult problem of scheduling multimedia applications, which have a mix of activities that have very different expected performance characteristics, resource requirements, and value to the user. Our solution handles mixes of conventional interactive and batch activities and real-time continuous media activities in a unified and tightly integrated manner. This means that the system is able to dynamically generate a more nearly optimal schedule for the activity mix at hand.

Our scheduling approach seeks to maximize the value the system delivers to the user by taking full advantage of user-specified timeliness and importance information. In other approaches, these dimensions, if they exist at all, are frequently prematurely collapsed to a single priority value. In so doing, significant information, required to schedule diverse collections of activities properly, is lost. At the same time, the scheduler does not impose draconian demands on the user for information he does not have or does not choose to provide; intelligent defaults are available.

Through the specification of time constraints and importance, the user can effectively control the behavior of the applications within the system. Through integrated processor scheduling, the system can deliver high value and high utilization for the user in support of multimedia applications.

4 References

1. AT&T: UNIX System V Release 4 Internals Student Guide, Vol. I, Unit 2.4.2., AT&T, 1990.

2. J. R. Eykholt, S. R. Kleiman, S. Barton, R. Faulkner, et. al.: *Beyond Multiprocessing...Multithreading the SunOS Kernel*, USENIX Summer 1992, San Antonio, Texas.

3. D. B. Golub: *Operating System Support for Coexistence of Real-Time and Conventional Scheduling*, Technical Report CMU-CS-94-212, School of Computer Science, Carnegie Mellon University, November 1994.

4. J. G. Hanko: *A New Framework for Processor Scheduling in UNIX*, Abstract talk from the Fourth International Workshop on Network and Operating Systems Support for Digital Audio and Video, November 1993.

5. J. P. Lehoczky, L. Sha, J. K. Strosnider: *Enhanced Aperiodic Responsiveness in Hard Real-Time Environments*, Proceedings of the IEEE Real-Time Systems Symposium, December 1987.

6. R. Levin, E. Cohen, W. Corwin, F. Pollack, W. Wulf: *Policy/Mechanism Separation in Hydra*, Proceedings Fifth Symposium on Operating Systems Principles, ACM, November, 1975.

7. K. J. Lin, J. W. S. Liu, K. B. Kenny, S. Natarajan: *FLEX: A Language for Real-Time Systems Programming*, Technical Report UIUCDCS-R-90-1634, Department of Computer Science, University of Illinois at Urbana-Champaign, October 1990.

8. C. L. Liu, J. W. Layland: *Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment*, JACM 20(1), January 1973.

9. C. D. Locke: *Best-Effort Decision Making for Real-Time Scheduling*, Ph.D. Thesis, Department of Computer Science, Carnegie Mellon University, May, 1986.

10. C. W. Mercer, S. Savage, H. Tokuda: *Processor Capacity Reserves: Operating System Support for Multimedia Applications*, Proceedings of the IEEE International Conference on Multimedia Computing and Systems, May 1994.

11. J. Nieh, J. G. Hanko, J. D. Northcutt, G. A. Wall: *SVR4 UNIX Scheduler Unacceptable for Multimedia Applications*, Proceedings of the Fourth International Workshop on Network and Operating Systems Support for Digital Audio and Video, November 1993.

12. J. Nieh, M. S. Lam, J. G. Hanko, J. D. Northcutt: *Integrated Processor Scheduling in Support of Multimedia Applications*, submitted for publication.

13. S. Ramos-Thuel, J. P. Lehoczky, *On-Line Scheduling of Hard Deadline Aperiodic Tasks in Fixed-Priority Systems*, Proceedings of the IEEE Real-Time Systems Symposium, December 1993.