Appears in *Proc. Third International Conf. on Parallel and Distributed Information Systems,* Austin, TX, Sept. 28-30, 1994, pp. 7-16

Exposing I/O Concurrency with Informed Prefetching

R. Hugo Patterson†, Garth A. Gibson*

†Department of Electrical and Computer Engineering
*School of Computer Science
Carnegie Mellon University, Pittsburgh PA 15213

Abstract

Informed prefetching provides a simple mechanism for I/O-intensive, cache-ineffective applications to efficiently exploit highly-parallel I/O subsystems such as disk arrays. This mechanism, dynamic disclosure of future accesses, yields substantial benefits over sequential readahead mechanisms found in current file systems for non-sequential workloads. This paper reports the performance of the Transparent Informed Prefetching system (TIP), a minimal prototype implemented in a Mach 3.0 system with up to four disks. We measured reductions by factors of up to 1.9 and 3.7 in the execution time of two example applications: multi-file text search and scientific data visualization.

1: Introduction

Reducing program execution time is commonly the reason for purchasing new, faster processors. However, for programs that process stored data, faster processors do not linearly decrease execution time unless they are coupled with proportionately faster storage systems. Because storage performance is increasing more slowly than processor performance, data-intensive programs do not benefit as much as one might expect from a faster processor. To directly combat this limitation, new storage systems are increasing disk parallelism, usually in the form of Redundant Arrays of Inexpensive Disks (RAID) [Patterson88, Gibson91].

RAID subsystems exploit data striping to provide high throughput: high data rate for large parallel transfers and I/O concurrency and disk load balancing for large numbers of small accesses [Kim86, Livny87]. Unfortunately, RAID subsystems cannot reduce the access latency of isolated small reads and can increase small write latencies [Chen90, Stodolsky93]. The situation is analogous to that of parallel processors which are effective when applied to large jobs distributed over the processors and to many independent small jobs running in parallel. But, parallel processors do not reduce the execution time of a serial program any more than RAIDs reduce the latency of a small access. Since serial streams of small accesses dominate many important workloads, access latency is an increasingly important component of overall system performance. Distributed file systems further increase its importance by adding transfer and server overheads to the storage access time [Sandberg85, Satya85].

This work was supported in part by the National Science Foundation under grant number ECD-8907068 and by an IBM Graduate Fellowship.

Caching recently used file blocks can provide fast access when a program's workload is small or has high locality. But, growing file sizes and the large volume of read-once data limit the effectiveness of file caching [Baker91]. Beyond caching, prefetching soon-to-be-needed file blocks is the best method of reducing storage access time [Feiertag71, McKusick84].

To be most successful, prefetching should be based on the *knowledge* of future accesses often available within applications. By passing hints to the file system, applications can disclose this information to lower levels of the system. There, it may be combined with global knowledge of the competing demands for system resources. Thus informed, a file system can transparently prefetch needed data and optimize resource utilization. We call this *informed prefetching*.

As presented previously, informed prefetching reduces application execution time through three mechanisms [Patterson93].

Exposure of an application's I/O concurrency: The primary advantage of informed prefetching is its ability to perform multiple I/O accesses in parallel so that applications and users spend less time waiting for these accesses to complete. Because informed prefetching systems know what to prefetch, they can utilize resources less timidly than uninformed prefetching systems. It is this ability of informed prefetching to expose and exploit I/O concurrency that enables it to convert the high throughput of parallel I/O technologies to the lower access latencies these storage technologies cannot directly provide.