Adaptive Software Cache Management for Distributed Shared Memory Architectures

John K. Bennett[*]
John B. Carter[**]
Willy Zwaenepoel[**]

[*]Department of Electrical and Computer Engineering
[**]Department of Computer Science
Rice University
Houston, TX 77251-1892

Abstract

An *adaptive* cache coherence mechanism exploits semantic information about the expected or observed access behavior of particular data objects. We contend that, in distributed shared memory systems, adaptive cache coherence mechanisms will outperform static cache coherence mechanisms. We have examined the sharing and synchronization behavior of a variety of shared memory parallel programs. We have found that the access patterns of a large percentage of shared data objects fall in a small number of categories for which efficient software coherence mechanisms exist. In addition, we have performed a simulation study that provides two examples of how an adaptive caching mechanism can take advantage of semantic information.

1 Introduction

We are developing Munin [4], a system that will allow programs written for shared memory multiprocessors to be executed efficiently on distributed memory machines. What distinguishes Munin from previous distributed shared memory systems [6, 12, 14] is the means by which memory coherence is achieved. Instead of a single memory coherence mechanism for all shared data objects, Munin will employ several different mechanisms, each appropriate for a different category of shared data object. We refer to this technique of providing multiple coherence mechanisms as *adaptive caching*. Adaptive caching maintains coherence based on the expected or observed access behavior of each shared object and on the size of cached items. We contend that adaptive caching provides an efficient abstraction of shared memory on distributed memory hardware. Since coherence in distributed shared memory systems is provided in software, we expect the overhead of providing multiple coherence mechanisms to be offset by the increase in performance that such mechanisms will provide.

For adaptive caching to perform well, it must be possible to characterize a large percentage of all accesses to shared data objects by a small number of categories of access patterns for which efficient coherence mechanisms can be developed. In a previous paper [4], we have identified a number of categories, and described the design of efficient coherence mechanism for each. In this paper, we show that these categories capture the vast majority of the accesses to shared data objects in a number of shared memory parallel programs. We also show, through simulation, the potential for performance improvement of adaptive caching compared to static coherence mechanisms.

In Section 2 of this paper, we briefly reiterate the main results of our previous paper [4]. We describe the categories of access patterns, provide examples, and give a brief description of how each category can be handled efficiently. Section 3 describes the programs that we study in this paper, our technique for logging the accesses to shared memory by these programs, the method by which we analyze these logs to discover common access patterns, and the results of our logging study. Section 4 describes a simulation study that provides two examples of how an adaptive caching mechanism can take advantage of semantic information. We discuss previous work in this area in Section 5. Finally, we draw conclusions in Section 6.

This work was supported in part by the National Science Foundation under Grants CDA-8619893 and CCR-8716914.