論文の内容の要旨

STUDY ON SHARED CACHE MANAGEMENT METHOD WITH SMALL HARDWARE FOR MULTI-CORE PROCESSORS

（マルチコアプロセッサ向け省資源共有キャッシュ管理の研究）

氏名　　小川　周吾

We propose a method of shared-cache management to avoid conflict misses in multi-core processors with less hardware in this thesis. Decreased performance in multi-core processors, which is caused by conflict misses in several cores in shared caches, is a problem. There is also another problem as the frequency of conflict misses increases in proportion to the number of cores. These two problems prevent us from proportionally increasing the number of cores in each processor. Therefore, decreasing the frequency of conflict misses in shared caches is important so that we can achieve high-performance scalable multi-core processors in proportion to the number of cores.

In this thesis, we propose History-Free Cache Allocation (HFCA) method for reducing conflict misses in shared cache by allocating cache ways dynamically to each core, and then we aim to achieve high-performance scalable multi-core processors. HFCA divide shared cache into private blocks for each core and one shared block while many previous cache partitioning methods do not have a shared block. HFCA decide the size of private blocks for each core according to conflicts detected in a shared block. Therefore HFCA do not require large additional hardware for counting the number of cache hits in any blocks. We evaluate HFCA on the effect for improving IPC (Instructions per clock) and additional hardware size. Then we showed that HFCA improves IPC by 13.5% on a 2-core processor and 17.2% on a 4-core processor. We also showed that HFCA reduces additional hardware size to less than 0.1% when a 4-core multi-core processor has one 32-way set-associative shared cache at the cost of 1.4% IPC decrease in a 2-core processor, and 1.5% IPC decrease in a 4-core processor.

This thesis also explains two methods for reducing conflict misses in a shared cache, which have been studied until the proposal of HFCA. The HFCA derived from these methods. At first, we describe Thread-Set Selection (TSS) for reducing conflict misses in shared cache by choosing simultaneously executing threads. The frequency of conflict misses increase or decrease by the combination of parallel threads. TSS counts the number of cache misses for each combination of parallel threads by using performance monitoring counters in processors. Then TSS chooses simultaneously executing threads according to the result of counting misses for each combination. We evaluate TSS concerning the effect on reducing conflict misses in shared cache among cores. We implemented TSS on process scheduler of Linux kernel, and evaluated TSS on multi-threaded processor which has shared cache. Then the result of evaluation indicates that TSS has a limited effect on reducing conflict misses because TSS cannot evict cache conflicts

among cores directly. Therefore, to reduce cache conflicts efficiently, a new method for avoiding cache conflicts directly is required.

Next, we describe Partial Trial-Based Runtime Partitioning (PTRP) for reducing cache misses in shared cache by dynamic cache partitioning. PTRP resolves problems of TSS above mentioned. Several cache partitioning methods are previously proposed for reducing conflict misses by dividing one shared cache into private cache blocks for each core. These previous methods have a problem which these methods require large additional hardware to count the number of hits for minimizing cache misses. On the other hand, PTRP requires smaller additional hardware than previous methods because PTRP repeats modifying cache partition in a part of shared cache sets for reducing cache misses. We evaluate PTRP on the effect for improving processor performance and additional hardware size. Then we showed that PTRP reduces cache misses by 31% in a 2-core processor, and 19% in a 4-core processor at the maximum. We also showed that PTRP reduces additional hardware size to less than 1% from one of previous proposed cache partitioning methods (UCP) at the cost of 1.5% miss increase in a 2-core processor, and 3% miss increase in a 4-core processor. However, PTRP requires either software overhead or increase of additional hardware resources for management because of repeating trials for finding optimal partitions. Moreover, a certain pattern of cache accesses in a core obstructs finding the optimal partitions. Therefore, to resolve these problems above mentioned, a new method for avoiding cache conflicts is required.

The purpose of HFCA proposed in this thesis is to reduce conflict misses in shared cache for multi-core processors. Prior cache partitioning methods for reducing conflict misses require large hardware to count the number of cache hits. On the other hand, HFCA reduces the size of required hardware by detecting cache conflicts instead of counting hits in a shared cache. Then HFCA contribute to achieve high-performance and scalable multi-core processors in proportion to the number of cores.