Abstract

Simultaneous multi-threading (SMT) processors exploit thread level parallelism from multiple applications. One of the major problems with these processors is fair sharing of datapath resources. Today multiple threads can use uncontrolled portion of these resources, and this causes performance and energy problems.

Through the past years, several methods were proposed for SMT resource distribution. Some studies offered static allocation methods, but most studies were based on various dynamic allocation methods.

In this project, we have implemented a dynamic resource allocation method named ARPA-2-bit SMT Processor Distribution, and compared its results with the results of other dynamic resource allocations methods, Hill-Climbing, standard Adaptive Resource Partitioning (ARPA), and also with the results of fixed partition resource allocation. We expected to achieve a new dynamic allocation approach for SMT Processor resource distribution.

In addition, we implemented a hyper-heuristic algorithm used to decide the next heuristic according to performances of threads previous epochs. With this way, we purposed to maximize IPC of each thread.

Motivation

Our approach proposes to improve the processor performance in terms of the throughput IPC (Instruction Per Cycle) metric. To achieve this goal, we implemented a new heuristic and a hyper-heuristic.

Previous Work

While we are implementing our new heuristic, we make use of Hill-Climbing and ARPA heuristics. Especially, we are inspired from the ARPA.

In addition, we used simple random method while we are implementing our hyper-heuristic.

Design & Implementation

As mentioned before, we referenced ARPA for our new heuristic. ARPA calculates efficiency of each thread according to the number of distributed resources. A new heuristic ARPA-2-bit keeps 2-bit information which represent previous and current winner thread of tournaments. Resources are distributed according to the state they are in. The automaton given below controls the mechanism. The leftmost bits represent the current winner thread of the tournament. The rightmost bits represent the previous winner thread. The rightmost bits reserved with division mark represent that resource are distributed or not. If it is equal to 0, it means that resources are not redistributed. Otherwise, resources are redistributed.

The Hyper-heuristic is a selection kind of hyper-heuristic in our project. It selects one of the heuristics (Hill-Climbing or ARPA) according to previo

Tests & Results

We used the M-Sim simulator and conducted 15 tests. In 14 of them we observed the performance of new heuristic ARPA-2-bit. In one test, we observed the performance of the proposes hyper-heuristic. We conducted tests with different epoch sizes (32K, 512K and 1024K cycles) and delta values (Issue Queue, Load/Store Queue and Re-Order Buffer, 2-3-3, 4-6-6 and 6-9-9, respectively).

Our new heuristic ARPA-2-bit gave better results than ARPA in 2 tests. It gave better results than Hill-Climbing in all of the tests.

We run hyper-heuristic test 4 times. It gave better results than other heuristics (Hill-Climbing, ARPA, ARPA-2-bit) in three runs.

Conclusion & Future Works

As a result, our proposed heuristic ARPA-2-bit can be preferable to Hill-Climbing. However, ARPA is still the first choice due to the test results.

Our simple random hyper-heuristic is giving hope. As we discussed in test results, the proposed hyper-heuristic gave better solutions than heuristics.

References