Abstract

Cluster computers represent a cost-effective alternative solution to supercomputers. In these systems, it is common to constrain the memory address space of a given processor to the local motherboard. Constraining the system in this way is much cheaper than using a full-fledged shared memory implementation among motherboards. However, memory usage among motherboards may be unfairly balanced depending on the memory requirements of the applications running on each motherboard. This situation can lead to disk-swapping, which severely degrades system performance, although there may be unused memory on other motherboards. A straightforward solution is to increase the amount of available memory in each motherboard, but the cost of this solution may become prohibitive.

On the other hand, remote memory access (RMA) hardware provides fast interconnects among the motherboards of a cluster computer. In recent works, this characteristic has been used to extend the addressable memory space of selected motherboards. In this work, the baseline machine uses this capability as a fast mechanism to allow the local OS to access to DRAM memory installed in a remote motherboard. In this context, efficient memory scheduling becomes a major concern since main memory latencies have a strong impact on the overall execution time of the applications, provided that remote memory accesses may be several orders of magnitude higher than local accesses. Additionally, changing the memory distribution is a slow process which may involve several motherboards, hence the memory scheduler needs to make sure that the target distribution provides better performance than the current one. This dissertation aims to address the aforementioned issues by proposing several memory scheduling policies.

First, an ideal algorithm and a heuristic strategy to assign main memory from the different memory regions are presented. Additionally, a Quality of Service control mechanism has been devised in order to prevent unacceptable performance degradation for the running applications. The ideal algorithm finds the optimal memory distribution but its computational cost is prohibitive for a high number of applications. This drawback is handled by the heuristic strategy, which approximates the best local and remote memory distribution among applications at an acceptable computational cost.

The previous algorithms are based on profiling. To deal with this potential shortcoming we focus on analytical solutions. This dissertation proposes an analytical model that

estimates the execution time of a given application for a given memory distribution. This technique is used as a performance predictor that provides the input to a memory scheduler. The estimates are used by the memory scheduler to dynamically choose the optimal target memory distribution for each application running in the system in order to achieve the best overall performance.

Scheduling at a higher granularity allows simpler scheduler policies. This work studies the feasibility of scheduling at OS page granularity. A conventional hardware-based block interleaving and an OS-based page interleaving have been assumed as the baseline schemes. From the comparison of the two baseline schemes, we have concluded that only the performance of some applications is significantly affected by page-based interleaving. The reasons that cause this impact on performance have been studied and have provided the basis for the design of two OS-based memory allocation policies. The first one, namely on-demand (OD), is a simple strategy that works by placing new pages in local memory until this region is full, thus benefiting from the premise that most of the accessed pages are requested and allocated before than the least accessed ones to improve the performance. Nevertheless, in the absence of this premise for some benchmarks, OD performs worse. The second policy, namely Most-accessed in-local (Mail), is proposed to avoid this problem.