Simulation Modelling Practice and Theory 19 (2011) 482–493

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

A new window-based job scheduling scheme for 2D mesh multicomputers

Ismail Ababneh [a], Saad Bani-Mohammad [b,*]

[a] Computer Science Department, Jordan University of Science and Technology, Irbid, Jordan
[b] Computer Science Department, Prince Hussein Bin Abdullah College for Information Technology, Al al-Bayt University, Mafraq 25113, Jordan

ARTICLE INFO

Article history:
Received 1 February 2010
Received in revised form 1 August 2010
Accepted 17 August 2010
Available online 21 August 2010

Keywords:
Job scheduling
Mesh
Multicomputer
Contiguous submesh allocation
Average turnaround time
Maximum waiting delay

ABSTRACT

Allocating submeshes to jobs in mesh-connected multicomputers in a FCFS fashion can lead to poor system performance (e.g., long job waiting delays) because the job at the head of the waiting queue can prevent the allocation of free submeshes to other waiting jobs with smaller submesh requirements. However, serving jobs aggressively out-of-order can lead to excessive waiting delays for jobs with large allocation requests. In this paper, we propose a scheduling scheme that uses a window of consecutive jobs from which it selects jobs for allocation and execution. This window starts with the current oldest waiting job and corresponds to the lookahead of the scheduler. The performance of the proposed window-based scheme has been compared to that of FCFS and other previous job scheduling schemes. Extensive simulation results based on synthetic workloads and real workload traces indicate that the new scheduling strategy exhibits good performance when the scheduling window size is large. In particular, it is substantially superior to FCFS in terms of system utilization, average job turnaround times, and maximum waiting delays under medium to heavy system loads. Also, it is superior to aggressive out-of-order scheduling in terms of maximum job waiting delays. Window-based job scheduling can improve both overall system performance and fairness (i.e., maximum job waiting delays) by adopting large lookahead job scheduling windows.

1. Introduction

Mesh-connected interconnection networks have been widely used in recent distributed-memory parallel computers [3,7,10,12,15,23]. This is mainly because these networks are regular, simple, easy to implement, and scalable. Both two-dimensional (2D) and three-dimensional (3D) meshes and tori have been used in recent commercial and experimental multicomputers, such as the Caltech Mosaic [5], the Intel Paragon [19], the IBM BlueGene/L [9,18,21], and the Cray XT3 [11,22].

In most processor allocation policies proposed for mesh-connected multicomputers, a parallel job is allocated its own submesh of processors of the size and shape it has requested [3,7,8,15,20]. This can result in high external processor fragmentation, which occurs when free processors are not allocated to a parallel job because they do not satisfy the job's shape constraint. A recurring outcome in allocation studies is that contiguous allocation suffers from low overall system utilization [3,14,23]. It can reduce system utilization to low levels. Therefore, noncontiguous allocation policies have been proposed so as to increase system utilization by allowing dispersed free processors to be allocated to a parallel job [2,12,23].

Notwithstanding the ability of noncontiguous allocation to reduce, even eliminate, processor fragmentation, an advantage of contiguous allocation is that it isolates jobs from each other, which is useful for security and accounting reasons [4]. For example, contiguous allocation is proposed for use in the IBM BlueGene/L for security reasons [4]. A BlueGene/L

* Corresponding author. Tel.: +962 2 6297000; fax: +962 2 6297051.
E-mail addresses: ismael@just.edu.jo (I. Ababneh), bani@aabu.edu.jo (S. Bani-Mohammad).

1569-190X/$ - see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.simpat.2010.08.007

job is allocated a partition of processors that is isolated from partitions allocated to other jobs because of the sensitive nature of some of BlueGene's applications [4].

Another approach to improving system utilization is using job scheduling policies that are not strictly first-come-first-served. Rather than always accommodating the oldest allocation request first, allocation to more recent requests is considered in order to decrease the number of idle processors and improve overall system performance [1,8,16,20,24]. These schemes, however, have several shortcomings. In [1], jobs are considered for allocation without ever waiting for the head of the queue or earlier requests. This greedy out-of-order scheduling can lead to excessive waiting delays, including indefinite postponement, for large jobs. Because small jobs are easier to accommodate, they can block allocation to a large job that has arrived earlier. Other schemes [8,16] place small bounds on the ability of the non-FCFS scheme to bypass the head of the waiting queue. Results in [3] indicate that such bypassing bounds should increase with the system load, and they should be much larger than those considered in [8,16]. Finally, the approach proposed in [24] assumes that job execution time estimates are available upon job submission. Although many current systems require users to submit such estimates, users often provide inaccurate estimated job execution times [17]. Therefore, we have limited ourselves to the case where execution time estimates are assumed to be unavailable.

In this paper, we propose a non-FCFS job scheduling scheme for use in mesh-connected multicomputers. The scheme aims to bound job waiting delays, while reducing processor fragmentation. In the proposed scheme, it is possible to bypass the head of the waiting queue of jobs awaiting allocation, but this ability is limited to a window of consecutive jobs that starts with the head of the waiting queue. This scheduling window corresponds to the lookahead capability of the job scheduler. Bypassing the head of the waiting queue has for goal reducing processor fragmentation by finding jobs that can be accommodated, which improves system utilization and average job turnaround times. Limiting the ability to bypass the oldest waiting job to a window has for goal bounding job waiting delays. Using detailed simulations, we have evaluated the proposed scheme and compared it with previous schemes. The results show that the proposed scheme can yield good average job waiting delays (i.e., good average turnaround times) and system utilization, and superior maximum job waiting delays.

This paper is organized as follows. The following section contains a review of previous related job scheduling schemes. Section 3 contains some preliminaries and the system model assumed. The proposed scheme is presented in Section 4. Simulation results are presented and discussed in Section 5. Finally, conclusions are given in Section 6.

2. Previous job scheduling schemes

Typically, traditional first-come-first-served (FCFS) job scheduling was assumed in research studies of contiguous processor allocation in mesh multicomputers [2,7,13–15,27]. In FCFS scheduling, jobs are served strictly in their arrival order. The queue where jobs wait is FIFO, and the job at the head of the waiting queue is considered for allocation before any subsequent job. An advantage of this scheme is its fairness. However, processor submeshes may remain unallocated because they are not large-enough for the head waiting job, and although they can satisfy other waiting allocation requests. As a consequence, contiguous allocation that assumes FCFS job scheduling suffers from low system utilization and poor performance. In order to improve system performance, several non-FCFS job scheduling policies that consider allocation to subsequent jobs when allocation fails for the head of the waiting queue were proposed and evaluated. Below, we include a brief review of previous job scheduling policies proposed for mesh-connected multicomputers.

2.1. k-Lookahead processor allocation

This scheme aims to retain large free mesh regions for large allocation requests. In k-lookahead allocation, the waiting requests are scanned and the largest k requests are selected. In making the allocation decision for the head of the waiting queue, an attempt is made to leave sufficient free mesh regions for the largest k waiting requests. Using simulations, the performance of 1-lookahead (i.e., $k = 1$) was evaluated. The results show that some moderate performance gains can be obtained [8]. In the general case, this scheme would require examining as many as $k!$ allocation cases. Also, $k = 1$ limits the scheduling lookahead window to two jobs (the oldest waiting job and the largest waiting job).

2.2. Reservation-based non-FCFS job scheduling

In this scheme [16], a waiting allocation request has a counter that contains the number of times that it has been overtaken by subsequent requests. When a new job arrives, allocation is attempted for the job provided that the counter of the current FIFO waiting queue head does not exceed some fixed value, called MAX_PRI. If this allocation attempt fails, reservation of a large-enough submesh is attempted for the new job, and if this allocation/reservation process fails the new job is appended to the waiting queue and its counter is set to zero. An issue with this policy is that reservation can decrease system utilization. Therefore, reservation degrades performance when the load is heavy, as can be seen in the simulation results in [16]. Another issue is that only small MAX_PRI values were considered because high MAX_PRI would mean excessive waiting delays for large jobs and unfairness to such jobs. A problem with this reasoning is that large MAX_PRI values can improve overall system utilization and waiting delays, which may also yield good maximum job waiting delays.