Contents lists available at SciVerse ScienceDirect

Simulation Modelling Practice and Theory

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

On the performance of non-contiguous allocation for common communication patterns in 2D mesh-connected multicomputers

Saad Bani-Mohammad *, Ismail Ababneh

Department of Computer Science, Prince Hussein Bin Abdullah College for Information Technology, Al al-Bayt University, Mafraq 25113, Jordan

ARTICLE INFO

Article history:
Available online 4 October 2011

Keywords:
Non-contiguous processor allocation
Turnaround time
System utilization
Communication
Contention

ABSTRACT

The communication pattern used by applications can have a major influence on the performance of non-contiguous processor allocation in multicomputers. In this paper, the performance of well-known non-contiguous allocation strategies for 2D mesh multicomputers is re-visited considering several important communication patterns. These are the Near Neighbour, Ring, Divide and Conquer Binomial Tree (DQBT), Fast Fourier Transform (FFT), and Random communication patterns. The allocation strategies investigated are the Greedy Available Busy List (GABL), Multiple Buddy Strategy (MBS), Adaptive Non-contiguous Allocation (ANCA), and Paging(0). They are compared using detailed flit-level simulations. The results show that GABL is overall superior to the remaining non-contiguous allocation strategies. It produces superior average turnaround times and mean system utilization.

1. Introduction

Various multicomputer architectures have used the mesh interconnection network because of its simplicity, structural regularity, partitionability, flexible routing, and ease of implementation [1,4,6,9–15,18,20,24–26,28,32]. Also, the mesh topology is suitable for a variety of applications, including matrix computations, and image processing [1]. Both two-dimensional (2D) and three-dimensional (3D) meshes and tori have been used in recent commercial and experimental multicomputers, such as the iWARP [5], the IBM BlueGene/L [31], the Delta Touchstone [17], the Caltech Mosaic [30], the Intel Paragon [16], and the Cray XT3 [7].

Efficient processor allocation and job scheduling are critical if the full computational power of large-scale multicomputers is to be harnessed effectively [4,12]. Processor allocation is responsible for selecting the set of processors on which parallel jobs are executed, whereas job scheduling is responsible for determining the order in which jobs are executed [4,12,18,25].

In most processor allocation strategies proposed in the literature for mesh-connected multicomputers, a parallel job is allocated a distinct contiguous sub-mesh of processors of the size and shape it has requested. This contiguous allocation can result in high external processor fragmentation, which occurs when free processors are not allocated to a parallel job because of the shape constraint. A recurring outcome in allocation studies is that contiguous allocation suffers from low overall system utilization [10,12,28]. It can reduce system utilization to levels unacceptable for government-audited systems in the USA [8]. Therefore, non-contiguous allocation strategies have been proposed with the goal of increasing system utilization by allowing dispersed free processors to be allocated to a parallel job [6,13,25,28].

Although non-contiguous allocation increases message contention in the network, several previous studies have shown that lifting the contiguity condition can be expected to improve overall system performance [6,18,25,28]. It is the introduction of wormhole switching [27] that has lead researchers to consider non-contiguous allocation in multicomputers with

* Corresponding author.
E-mail addresses: bani@aabu.edu.jo (S. Bani-Mohammad), ismael@aabu.edu.jo (I. Ababneh).

1569-190X/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.simpat.2011.08.007

long communication distances, such as the 2D mesh [6,25,28]. This is because one of the main advantages of wormhole switching over earlier switching techniques, e.g. store-and-forward, is that message latency depends less on the distance messages traverse from sources to destinations.

In this paper, we compare, using detailed simulations, the performance of existing well-known non-contiguous allocation strategies considering several important communication patterns. The allocation strategies investigated are the GABL [25], MBS [28], ANCA [6,24], and Paging(0) [28] strategies. These strategies have been selected because they have been shown to perform well in [6,13,25,28]. The communication patterns we consider are the Ring [22,28], Divide and Conquer Binomial Tree [21,27–29], Fast Fourier Transform [2,19,27,28], Random [18,25,28], and Near Neighbour patterns. Unlike previous related works [13,18,20,24,25,28], we consider the Near Neighbour communication pattern in this paper to study the effect of this important pattern on the performance of non-contiguous allocation. The communication patterns considered in this paper have been selected because they have been used in related works [13,18,20,24,25,28] and because they are common. The details of these communications patterns will be provided in Section 4. In the simulation experiments, communication is simulated at the flit-level. The results show that GABL is overall better than the previous non-contiguous allocation strategies, in terms of the average turnaround time and mean system utilization performance parameters.

The rest of the paper is organized as follows. Section 2 provides some preliminaries. Section 3 contains a brief overview of the non-contiguous allocation strategies considered in this study. Simulation results are presented in Section 4. Finally, Section 5 concludes this paper.

2. Preliminaries

The target system is a $W \times L$ 2D mesh, where W is the width of the mesh and L is its length. Every processor is denoted by a pair of coordinates (x, y), where $0 \leqslant x < W$ and $0 \leqslant y < L$ [25]. Each processor is connected by bidirectional communication links to its neighbouring processors, as depicted in Fig. 1. This figure shows a 4×4 2D mesh, where allocated processors are denoted by shaded circles and free processors are denoted by white circles. Given the system allocation state shown in Fig. 1, if a job requests the allocation of sub-mesh of size 2×2, contiguous allocation fails because no 2×2 sub-mesh of free processors is available, however four free processors can be allocated to the job if allocation is non-contiguous.

The following definitions have been adopted from [25].

Definition 1. A sub-mesh $S(w, l)$ of width w and length l, where $0 < w \leqslant W$ and $0 < l \leqslant L$ is specified by the (x, y, x', y'), where (x, y) are the coordinates of the base of the sub-mesh and (x', y') are the coordinates of its end. For example $(0, 0, 2, 1)$ represents the 3×2 sub-mesh S in Fig. 1. The base node of the sub-mesh is $(0, 0)$, and its end node is $(2, 1)$. The size of $S(w, l)$ is $w \times l$ processors.

Definition 2. An allocated sub-mesh is one whose processors are all allocated to a parallel job.

Definition 3. A free sub-mesh is one whose processors are all not allocated.

Definition 4. A suitable sub-mesh $S(w, l)$ is a free sub-mesh that satisfies the conditions: $w \geqslant \alpha$ and $l \geqslant \beta$ assuming that the allocation of $S(\alpha, \beta)$ is requested, where α and β are the side lengths of the allocation request.

3. Processor allocation strategies

Advances in switching techniques, in particular wormhole switching [27], have made communication latency less sensitive to the distance between communication end nodes [6,25]. This has made allocating a job to non-contiguous processors

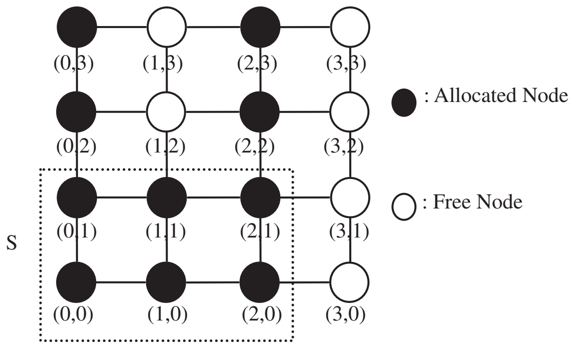

Fig. 1. An example of a 4×4 2D mesh.