An Efficient All Shapes Busy List Processor Allocation Algorithm for 3D Mesh Multicomputers

Saad Bani-Mohammad, Al al-Bayt University, Department of Computer Science, Mafraq, Jordan

ABSTRACT

Contiguous processor allocation is useful for security and accounting reasons. This is due to the allocated jobs are separated from one another, where each sub-mesh of processors is allocated to an exclusive job request, and the allocated sub-mesh has the same size and shape of the requested job. The size and shape constraint leads to high processor fragmentation. Most recent contiguous allocation strategies suggested for 3D mesh-connected multicomputers try all possible orientations of an allocation request when allocation fails for the requested orientation, which reduces processor fragmentation and hence improves system performance. However, none of them considers all shapes of the request in the process of allocation. To generalize this restricted rotation, we propose, in this paper, a new contiguous allocation strategy for 3D mesh-connected multicomputers, referred to as All Shapes Busy List (ASBL for short), which takes into consideration all possible contiguous request shapes when attempting allocation for a job request. ASBL depends on the list of allocated sub-meshes, in the method suggested in (Bani-Mohammad et al., 2006), for selecting an allocated sub-mesh. The performance of the proposed ASBL allocation strategy has been evaluated considering several important scheduling strategies under a variety of system loads based on different job size distributions. The simulation results have shown that the ASBL allocation strategy improves system performance in terms of parameters such as the average turnaround time of jobs and system utilization under all scheduling strategies considered.

KEYWORDS

All Shapes, Busy List, Contiguous Processor Allocation, Job Scheduling, Processor Fragmentation, Rotation, System Utilization, Turnaround Time

INTRODUCTION

Multicomputers, comprising of many processing elements that are associated through an interconnection network. Among the different multicomputer designs, those based on the mesh network have gotten much consideration because of its simplicity, scalability, structural regularity, partition-ability, and ease of implementation of this network (Yoo & Das, 2002; Chang & Mohapatra, 1998; Sharma & Pradhan, 1996; Kim & Yoon, 1998; Chiu & Chen, 1999; Ababneh & Bani-Mohammad, 2011; Ababneh, 2006; Ababneh, 2008; Ababneh, 2001; Ababneh, 2009; Ababneh et al., 2015; Ababneh et al., 2010; Bani-Mohammad & Ababneh, 2013; Bani-Mohammad et al., 2010; Bani-Mohammad et al., 2011; Bani-Mohammad et al., 2015; Bani-Mohammad et al., 2007a; Bani-Mohammad et al., 2008; Bani-Mohammad et al., 2006; Bani-Mohammad et al., 2007b; Bani-Mohammad et al., 2009). Meshes are reasonable to a variety of applications, including matrix computations, image processing and problems with task graphs that can be embedded naturally into the mesh.
Efficient processor allocation and job scheduling are critical to accomplish and harness the full computational power of vast scale multicomputers (Yoo & Das, 2002; Chiu & Chen, 1999; Ababneh, 2006; Bani-Mohammad et al., 2007a; Bani-Mohammad et al., 2006; Yoo et al., 1997). The aim of processor allocation is to choose the arrangement of processors on which parallel jobs are executed, while the aim of job scheduling is to choose the following job to be executed (Yoo & Das, 2002).

In distributed memory multicomputers, separate contiguous processor sub-meshes are assigned to jobs for the duration of their execution (Yoo & Das, 2002; Ababneh, 2006; Ababneh, 2001; Ababneh, 2009; Ababneh et al., 2015; Ababneh et al., 2010; Bani-Mohammad et al., 2006; Bani-Mohammad et al., 2007b; Bani-Mohammad et al., 2009; Zhu, 1992). Both two-dimensional (2D) and three-dimensional (3D) meshes and tori have been utilized in recent experimental and commercial multicomputers. Most contiguous processor allocation strategies that are proposed in the previous studies are for 2D meshes (Chiu & Chen, 1999; Ababneh, 2006; Ababneh, 2009; Ababneh et al., 2010; Chuang & Tzeng, 1994; Zhu, 1992). In spite of the fact that the 2D mesh has been used in a various of parallel machines, such as the Cray XE6m (Cray, 2014b), the iWARP (Peterson et al., 1991) and the Touchstone Delta system (Intel, 1991), current multicomputers, such as the K-computer (Riken & Fujitsu, 2014), the Cray XE6 (Cray, 2014a) and the IBM BlueGene/L (Horn et al., 2014) use the 3D mesh or the 3D tori (Cray, 2014a; Horn et al., 2014; Riken & Fujitsu, 2014) as an interconnection network due to their characteristics such as lower diameter and lower average communication distance (Athas & Seitz, 1988).

The vast majority of the existing processor allocation strategies use contiguous sub-mesh allocation (Yoo & Das, 2002; Sharma & Pradhan, 1996; Gabrani & Mulkar, 2005; Kim & Yoon, 1998; Chiu & Chen, 1999; Choo et al., 2000; Ababneh, 2006; Ababneh, 2001; Ababneh, 2009; Ababneh et al., 2015; Ababneh et al., 2010; Bani-Mohammad et al., 2006; Bani-Mohammad et al., 2007b; Bani-Mohammad et al., 2009; Zhu, 1992), where a distinct contiguous sub-mesh of processors is allocated to a parallel job for the duration of its execution. This can lead to high external processor fragmentation that happens when the processor allocation strategy cannot allocate available sub-mesh of processors to an incoming job because the available processors are not contiguous or they do not have the same topology as the interconnection system network. The previous studies on allocation (Chiu & Chen, 1999; Choo et al., 2000; Ababneh, 2006; Lo et al., 1997) have shown that contiguous allocation is prone to low system utilization. Accordingly, non-contiguous allocation has been suggested with the goal of increasing system utilization by allowing dispersed free processors to be allocated to a parallel job (Chang & Mohapatra, 1998; Ababneh, 2008; Bani-Mohammad et al., 2015; Bani-Mohammad et al., 2007a; Lo et al., 1997) instead of always waiting until a single sub-mesh of the requested size and shape is available.

Even though non-contiguous processor allocation can avoid processor fragmentation, an advantage of contiguous allocation is that it separates jobs from each other; consequently, it is useful for security and accounting reasons (Aridor et al., 2005). Contiguous allocation is, for example, suggested for use in the IBM BlueGene/L for security reasons (Aridor et al., 2005). In a BlueGene/L, a partition of processors is allocated to a parallel job that is isolated from partitions allocated to other jobs (Aridor et al., 2005). In this way, the allocated computing and communication resources are not shared.

Another approach can be used to improve system utilization. It is a job scheduling that is not strictly First Come First Served (FCFS). Rather than always serving the allocation to the oldest job request first, allocation to more recent job requests is considered in order to reduce the number of idle processors and improve the overall system performance (Mu’alem & Feitelson, 2001; Ababneh, 2001). However, these scheduling strategies have several weaknesses. In (Ababneh, 2001), job requests are considered for allocation without waiting for the head of the queue or earlier job requests. This is an Out-of-Order (OO) scheduling and it can lead to extreme waiting delays, including indefinite
On the use of the Hybrid Cloud Computing Paradigm
www.igi-global.com/chapter/use-hybrid-cloud-computing-paradigm/66234?camid=4v1a

Control Engineering for Scaling Service Oriented Architectures
www.igi-global.com/chapter/control-engineering-scaling-service-oriented/52239?camid=4v1a