1 Introduction

This annotated bibliography covers resource allocation algorithms for high performance computing (HPC) applications. It focuses more on distributed scheduling and ad-hoc approaches to allocating resources. I am less interested in the classical centralized scheduling algorithms where full knowledge is assumed. Seeing the application to distributed systems, the agent-based and market oriented approaches are currently more interesting.

2 Task Categories

There are many different categories of tasks. Tasks are simply work that users would like to accomplish. In some research areas tasks are also referred to as loads, jobs, or work.

2.1 Interactive

Jobs that are actively being manipulated by the user are interactive jobs.

2.2 Indivisible

Jobs not capable of being easily divided usually due to the lack of effort or the tight interconnection of the algorithm are referred to as indivisible jobs.
2.3 Divisible
Usually represented by directed acyclic graphs (DAG) representing the task dependencies. These tasks are usually very structured and well known.

2.4 Arbitrarily Divisible
A task that can be split an (almost) infinite number of times into smaller tasks and executed in parallel.

2.5 Parallel
These tasks are internally parallelized. They utilize message passing, multiple threads, and/or sub-tasks to distribute work across multiple cores and multiple machines while the jobs is running.

3 Computing Platforms
There are various types of high performance computing models used today. The platforms are listed with increasing coupling or connectedness. Interestingly enough they are also listed in decreasing fault tolerance. [Schroeder and Gibson, 2006]

3.1 Grid
Computing grids are very loosely coupled machines. They reside on the same network but can have different operating systems, architectures, and availability. [Li and Buyya, 2009] [Sulistio et al., 2003] [Foster and Kesselman, 2003] [Abramson et al., 2000]

3.1.1 Volunteer Computing
Volunteer computing is a relatively modern computing strategy where computing resources are volunteered to be part of the project for either recognition, fun, and sometimes monetary gain. Popular examples include SETI@home [Anderson et al., 2002] FOLDING@home [Beberg et al., 2009]

3.2 Cloud
Cloud computing has recently become a very popular computing platform with applications to HPC. Cloud computing has some key properties: pay for only what you
use, dynamic/elastic, virtualized networks, storage, and computing, secure, and internet connected. [Antonopoulos and Gillam, 2010] HPC instances have recently been added to Amazon EC2. Clouds utilized economies of scale to minimize the cost of its offerings.

Performance   [Fenn et al., 2009] [Napper and Bientinesi, 2009]

Business model  [Buyya et al., 2008]

3.3 Cluster
Clusters consist of many tightly coupled computing nodes. Typically they are connected with a fast interconnect. Each node runs its own instance of a common operating system. The top 500 supercomputers almost always are the cluster architecture. [Yeo et al., 2006]

Job arrival model  [Lublin and Feitelson, 2003]

3.4 Shared Memory Machines
Large shared memory machines contain a large number of CPUs which all share memory and thus run one large and highly scalable operating system.

4 Resource Allocation
The goal of a resource allocation algorithms is to assign tasks to machines (or collections of machines) while trying to optimize a given objective. Objective functions include minimum time, minimum cost, and most reliable/fault-tolerant. [Sulistio et al., 2004]

4.1 Clustering
Given a collection of tasks, find the clusters of tasks that should be run on the same machines. This is somethings referred to as DAG scheduling. [Hall et al., 2007][Cao et al., 2010][Gerasoulis and Yang, 1992]
4.2 Time Scheduling

This is the classic scheduling problem that is very similar to the NP-Hard bin packing problem. Determine the optimal placement of tasks on machines. [Bhat, 2008] [Benoit et al., 2010] [Brucker, 2007]

4.2.1 Shared Memory

Work stealing  [Reinders, 2007] [Navarro et al., 2009]

Parallel containers  [Buss et al., 2010]

Parallel FOR loops  [Chandra et al., 2001]

Robust  [Wang et al., 2009]

4.2.2 Distributed Memory

[Bobroff et al., 2008] [Jette et al., 2002] [Dusseau et al., 1996]

Heterogenous  [Kalinov and Klimov, 2005]

Deadline uncertainty  [Yeo and Buyya, 2006a]

Meta-scheduling  [Hamscher et al., 2000]

Integer linear programming  [Pan and Chen, 2005]

4.3 Environmental

Environmental aware schedulers consider environmental factors such as CO$_2$ emissions and electrical energy in the scheduling problem. These objectives may be part of the primary objective or a secondary objective. [Kim et al., 2011] [Al-Daoud et al., 2010] [Buyya et al., 2010]
4.4 Cloud

4.4.1 Instance Allocation

Determine which machine is best suited to host a particular instance. Current schedulers are fairly naive and include round-robin, most used first (to try to cluster instances and thus allow unused machines to sleep), by instance type, and by user. Instance allocation is performed by the cloud provider but may be aided by the user. [Li, 2009]

4.4.2 Instance Scheduling

Determining how many, and of what size, instances are required to execute a particular set of jobs. Also determine when to terminate the running instances when no longer needed. [Li, 2009]

Nash equilibrium [Sun et al., 2010]

Load balancing [Fang et al., 2010]

Over-subscribed [Chard et al., 2010] [Li, 2009]

Local cluster overflow, cloud bursting [de Assuncao et al., 2009]

4.5 Market-Oriented

One of the first problems encountered when dealing with over subscribed systems is how to assign priorities. Market-oriented scheduling policies side-step this problem by using cost as an alternative to artificial priorities. Users then have a finite amount of money or credits assigned to them (or earned) to which they can exchange for computing resources. These schedulers are often called matchmakers because they match offers to buyers of computing services. Cloud computing utilizes a market-oriented scheduling approach. Some general market-oriented scheduling papers include: [Yeo and Buyya, 2006b] [Salehi and Buyya, 2010] [Chard et al., 2010] [Xiao et al., 2008] [Buyya et al., 2008] [Buyya et al., 2009a] [Yeo and Buyya, 2007b] [Yeo and Buyya, 2004]

Utility computing [Yeo et al., 2010] [Buyya et al., 2009b] [Yeo and Buyya, 2007a] [Yeo and Buyya, 2009] [Yeo et al., 2007]

Service level agreement [Yeo and Buyya, 2005]
Agent-based [Veit, 2004] [Sun et al., 2010]

Brokering [Veit, 2004] [Abramson et al., 2002]

Auction [Zhao and Li, 2010]

4.6 Divisible Load Theory

Scheduling of communication heavy, arbitrarily divisible loads to graph of machines. These scheduling techniques are almost always optimal polynomial time algorithms. In addition then are often computable agent-recursively (removing the need for a centralized scheduler). Divisible load theory naturally handles heterogenous clusters of machines. [Bharadwaj et al., 2003] [Shokripour and Othman, 2009] [Lin et al., 2007] [Viswanathan et al., 2007] [Bharadwaj et al., 1996] [Blazewicz et al., 1999] [Robertazzi, 2003] [Benoit et al., 2010]

Linear and bus networks [Sohn and Robertazzi, 1998]

Buffer constraints [Li et al., 2000]

k-dimensional mesh networks [Chang et al., 2007] [Drozdowski and Glazek, 1999]

Tree networks [Beaumont et al., 2005] [Li and Veeravalli, 2010]

General networks [Jia et al., 2007] [Beaumont et al., 2003]

Applications [Altilar and Paker, 2002] [Chin et al., 2009] [Chan et al., 2001] [Viswanathan et al., 2007] [Marchal et al., 2005]

Dishonest machines [Carroll and Grosu, 2006]
5 Bibliography

References


of the 7th IEEE International Conference on Cluster Computing (Cluster 2005), Boston, MA, USA. IEEE Computer Society: Los Alamitos, CA, USA.


