Replication of Data Placement for Uncertain Scheduling

Manmohan Chaubey, Erik Saule
Department of Computer Science
University of North Carolina at Charlotte
Charlotte, USA
Email: mchaubey@uncc.edu, esaule@uncc.edu

***Abstract*—The abstract goes here. DO NOT USE SPECIAL CHARACTERS, SYMBOLS, OR MATH IN YOUR TITLE OR ABSTRACT.**

***Keywords*-component; formatting; style; styling;**

I. Introduction

Many real world scheduling problems related to task allocation in parallel machines are uncertain in nature. This fact makes the class of problem known as 'scheduling with uncertainty' or 'robust' scheduling an important and most studied problem in Scheduling. Often in scheduling models the exact value of parameters such as processing times of tasks, are not known initially, but they have a range outside which their values cannot lie. This paper draws qualitative analysis of the effect of replication in data placement for uncertain scheduling with inexact processing time of a task. Thus our research brings two area of scheduling problem together i.e. task uncertainty and data placement. The scheduling problem related to data placement and task allocation is common in heterogeneous systems and often dealt with common approach. The objective of our reseach is to propose an optimization approach that takes into account effective data placement using replication to schedule tasks in the environment where processing time of tasks are not known exactly but they can be estimated based on some pre estimated values.

Data placement technique can be useful in the scenario of uncertain processing times of tasks. Effective data placement using replication strategy allows better load balancing and hence reduces turnaround job time. This paper is important in the sense that it proposes different models depending upon different scenarios and compares them based on approximation ratio and replication they allow. Replication strategy that allows to place data in a wisely manner offers a faster access to files required by jobs, hence increases the job execution's performance. Replication helps in load balancing but it have certain cost attached with it as it usually increases resource usage[1]. The paper deals with this problem and chooses the scenario in which replication is beneficial.

There is very less literature available for scheduling with estimated processing time of a task. Most of the literature till now focuses on job size estimation with small error. Very less work is done which focuses on coping with less restrictive estimation for task size and which analyzes the effect of estimation on scheduling quantitatively. Wierman and Nuyens [3] introduce $\xi - SMART$ as classification to understand size based policies and draw analytical co-relation between response time and estimated job size in single server problem. [2] provides insight into scheduling behavior when estimation error is present and proposes FSPE+PS based policy to cope with this problem.

Size estimation based policies are widely used in practicality in the area of MapReduce and Hadoop. [4] [5] proposes schedulers which provides robustness in scheduling against uncertain Job Size. [7] provides scheduling heuristics that optimizes both makespan and robustness in scheduling task graph on heterogenous system.

In literature uncertainty problems are also viewed as sensitivity analysis and purturbations in processing time of a task. [8] derives bounds on competitive ratio of Graham's online algorithm in scenario where processing times of jobs either increase or decrease arbitrarily due to pertured processing times of tasks.

[10] presents sensitivity analysis for scheduling trees with communication delays on two identical machines. Wagelmans give supporting evidence for the notion that a good performance of an algorithms is identical with a high degree of sensitivity [9].

The remaining of the paper is organized as follows:we describe system model and notatins in section 2. Sections 3-5 describes different problem models- 1)No replications, 2)replication is done everywhere and 3)replication is done within a group. The respective sections describe derivation of comptetitave ratios for each of these models. Section 6 summarises the 3 models and based on experimental results.

II. Problem Definition

We have set J of n jobs which need to be scheduled to set M of m machines such that makespan, C_{max} is minimized. C^*_{max} denotes optimal makespan of a schedule S. We are considering the problems where the scheduler do not know the processing time of a task exactly before it completes, but we have some estimation of the processing time of a task before it is assigned to a processor. We know that the actual processing time of a task i is between $\frac{1}{\alpha}$and α times of its estimated processing time. p_i denotes the actual processing time and $\tilde{p_i}$ denotes estimated processing time for the task i. We have this estimate:

$$\frac{1}{\alpha} \leq \frac{p_i}{\tilde{p_i}} \leq \alpha \tag{1}$$

[Note:give justification]

We consider various problem models incorporating the above mentioned scenario of uncertain processing times of tasks with an estimate. Depending upon a problem model the scheduling is done in two phases or just in one phase. In Phase 1, we allocate tasks to certain processors or group of processors depending upon a problem model. Phase 1 chooses where data to be replicated using estimated processing time $\tilde{p_i}$, for each of the task i. Phase 1 takes $\tilde{p_i}$, m and α as input and outputs set of machines, $M_j \subseteq M$ where a task j can be scheduled or is replicated. Phase 2 takes output of phase 1 as input and outputs set of task assigned to a machine i, $E_i \subseteq J$. In Phase 2, we choose actual schedule with semi-clairvoyant algorithm which uses only approximate knowledge of initial processing time, and after scheduling the task the actual p_i is known. With objective to find schedule which minimizes makespan, we investigate greedy algorithms for each of the problem models and prove their comptetitive ratios.

We have used Graham's List Scheduling (LS) and Largest Processing Time (LPT) algorithms to derive approximation ratios in different scenarios. The LS algorithm takes tasks one at a time and assign it to the processor having least load at that time. LS is 2-approximation algorithm and is widely used in online scheduling problems. The LPT sorts tasks in decreasing order of processing time and assign them one at a time in this order to the processor with the smallest current load. The LPT algorithm have worst case performance ratio as $4/3 - 1/(3m)$ in offline setting . Depending upon which among these two algorithms suits more for a problem model we have used these algorithms accordingly.

III. Model 1: No replication

In this problem model we have considered situation where each task is restricted to be scheduled on only one machine, i.e. $|M_j| = 1$. We have a set J of n jobs, and a set M of m machines. Let $f : J \leftarrow M$ be a function that assigns each job to exactly one machine. Let us denote E_i as the set tasks which is assigned to a machine i. The restriction that each task can be scheduled to only one machine restrict the problem constrution to phase one only. There is no replication of tasks in this model.

We have considered List scheduling and Longest processing time algorithms and have assigned each task on machine on which they are restricted to. Based on the estimated processing times of tasks loaded on each machine we will derive makespan.

A. Lower Bound

Thoerem 1.1: There is no online algorithm having competitive ratio better than α^2.
Proof: We use adversary technique to prove that there is no online algorithm having competitive ratio better than α^2

Let us consider an instance in which total tasks be λm with $\tilde{p_i} = 1$. After scheduling let the most loaded machine j have B tasks. Obviously, $B \geq \lambda$. In phase 2 the adversary increases the tasks on j by α and decreases the other tasks by α. So, $C_{max} = \alpha B$ and $C^*_{max} \geq \frac{\alpha B + \frac{1}{\alpha}(\lambda m - B)}{m}$. We have,

$$\frac{C_{max}}{C^*_{max}} \leq \frac{\alpha^2 Bm}{\alpha^2 B + \lambda m - B} = \frac{\alpha^2 m}{\alpha^2 + \frac{\lambda m}{B} - 1}$$

From above expression it is clear that smaller the value of B, the value of $\frac{C_{max}}{C^*_{max}}$ decreases. Irrespective of the value of λ,for $B = \lambda$ the value of expression would be minimum and is equal to $\frac{\alpha^2 m}{\alpha^2 + m - 1}$. When m tends to ∞ the expression equals α^2. So, any algorithm cannot give competitive ratio better than α^2.

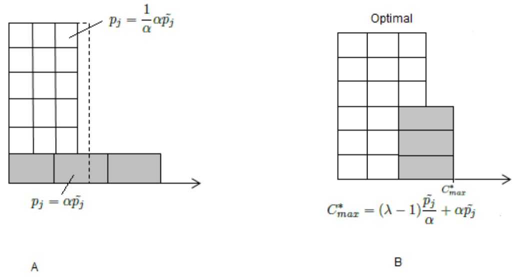
$p_j = \frac{1}{\alpha}\alpha\tilde{p_j}$
$p_j = \alpha\tilde{p_j}$
A
Optimal
C^*_{max}
$C^*_{max} = (\lambda - 1)\frac{\tilde{p_j}}{\alpha} + \alpha\tilde{p_j}$
B

Figure 1. Model 1

B. Algorithm

Theorem 1.2: Using LPT the competitive ratio is $\frac{2\alpha^2 m}{2\alpha^2+m-1}$.

Proof: In this problem model we are assigning tasks to processors using LPT in offline mode using non-increasing order of estimated processing times of tasks.

$$\tilde{C}_{max} \leq \frac{\sum \tilde{p}_j + (m-1)\tilde{p}_l}{m} \qquad (2)$$

where $\tilde{C}_{max}$ is makespan considering estimated processing times of tasks. Also actual processing time C_{max} must be smaller the $\alpha\tilde{C}_{max}$, we have

$$C_{max} \leq \alpha\tilde{C}_{max} \leq \alpha[\frac{\sum \tilde{p}_j + (m-1)\tilde{p}_l}{m}] \qquad (3)$$

Considering the worst case situation the processor with $\tilde{C}_{max}$ will see its tasks increase by α and the load on the rest of the processors will by shrink $\frac{1}{\alpha}$. The argument behind this statement is that greater the value of ratio $\frac{C_{max}}{\sum p_j}$, the algorithm will give bad approximation. So increasing the load on the machine which have C_{max} and decrasing the rest of the load on other processors will reach worst case scenario. So the total actual processing time is given by the following equation.

$$\sum p_j = \frac{\tilde{p}_j - \tilde{C}_{max}}{\alpha} + \alpha\tilde{C}_{max} \qquad (4)$$

Also the actual optimal makespan have following constraint

$$C^*_{max} \geq \frac{\sum p_j}{m}$$

Substituiting for $\sum p_j$, we have

$$mC^*_{max} \geq \frac{\tilde{p}_j - \tilde{C}_{max}}{\alpha} + \alpha\tilde{C}_{max}$$

$$\Rightarrow mC^*_{max} \geq \frac{\tilde{p}_j - [\frac{\sum \tilde{p}_j + (m-1)\tilde{p}_l}{m}]}{\alpha} + C_{max}$$

$$\Rightarrow mC^*_{max} \geq \frac{m-1}{\alpha m}[\sum \tilde{p}_j - \tilde{p}_l] + C_{max}$$

By the property of LPT $\sum \tilde{p}_j - \tilde{p}_l \geq m(\tilde{C}_{max} - \tilde{p}_l)$, putting this we have,

$$mC^*_{max} \geq \frac{m-1}{\alpha}\left[\tilde{C}_{max} - \tilde{p}_l\right] + C_{max}$$

$$\Rightarrow mC^*_{max} \geq \frac{m-1}{\alpha}\left[\tilde{C}_{max} - \frac{\tilde{C}_{max}}{2}\right] + C_{max}$$

$$\Rightarrow mC^*_{max} \geq \frac{m-1}{2\alpha}\left[\frac{C_{max}}{\alpha}\right] + C_{max}$$

$$\Rightarrow mC^*_{max} \geq \left[\frac{m-1}{2\alpha^2} + 1\right] C_{max}$$

$$\Rightarrow \frac{C_{max}}{C^*_{max}} \leq \frac{2\alpha^2 m}{2\alpha^2 + m - 1}$$

IV. Model 2: replication is done everywhere

In this problem model we consider no restrictaion on task assignment. In the first phase task are replicated everywhere i.e. $\forall j, |M_j| = |M|$. All tasks are allowed to replicate everywhere There are n tasks which need to be assigned to m processors. In the second phase we simplly use the Largest Processing Time algorithm (LPT) using estimated processing times of tasks in non-increasing order as input.

Lemma 2.1: $C^*_{max} \geq \frac{2}{\alpha^2}p_l$ when there are at least two tasks in the machine to which the last task, l is scheduled.

Proof: Suppose l be the last task with estimated processing time $\tilde{p}_l$. Suppose there are at least two tasks in the machine in which l is assigned including l. Let C_{max} be the be makespan of the schedule and C^*_{max} be the optimal makespan.

As there is at least one task j before l in the machine to which l is assigned, we have

$$C^*_{max} \geq p_l + p_j$$

As actual processing time of a task must be greater than $\frac{1}{\alpha}$ times of its estimated value, we have

$$C^*_{max} \geq \frac{1}{\alpha}\tilde{p}_l + \frac{1}{\alpha}\tilde{p}_j$$

As j is scheduled before l using LPT on estimated values of processing times, $\tilde{p}_j \geq \tilde{p}_l$ holds true for tasks l and j. Using this, we have

$$C^*_{max} \geq \frac{2}{\alpha}\tilde{p}_l$$

$$\Rightarrow C^*_{max} \geq \frac{2}{\alpha^2}p_l \tag{5}$$

Theorem 2.1: $\frac{C_{max}}{C^*_{max}} \leq 1 + (\frac{m-1}{m})\frac{\alpha^2}{2}$

Proof: The optimal makespan, C^*_{max} must be at least equal to the average load on the m machines. We have

$$C^*_{max} \geq \frac{\sum p_j}{m} \tag{6}$$

By the property of LPT the load on each machine i is greater than the load on the machine which reach C_{max} before the last task l is scheduled. So for each machine i, $C_{max} \leq \sum_{j \in E_i} p_j + p_l$ holds true. Summing for all the machines we have

$$mC_{max} \leq \sum p_j + (m-1)p_l$$

$$C_{max} \leq \frac{\sum p_j}{m} + \frac{(m-1)}{m}p_l \tag{7}$$

Using (6) and (7), we have

$$\frac{C_{max}}{C^*_{max}} \leq 1 + \frac{m-1}{m}\left(\frac{p_l}{C^*_{max}}\right)$$

As $C^*_{max} \geq \frac{2}{\alpha^2}p_l$, We have

$$\frac{C_{max}}{C^*_{max}} \leq 1 + \left(\frac{m-1}{m}\right)\frac{\alpha^2}{2}$$

Hence the theorem follows.

V. Model 3: Replication in groups

In this model the first phase is in offline mode and each task is pre-assigned to a particular group of processors. In the second phase the tasks are scheduled within the group they are assigned to in first phase. We have a set J of n jobs. There are k groups and size of each group is equal and have m/k processors within each group. We have considered task allocation in a group such that each task can be assigned to only one group, i.e. $\forall j$, $|M_j| = m/k$. In phase 2 each task is scheduled to a particular processor within the group it was allocated in phase 1. We propose List Sheduling algorithm in both the phases. In phase 1 using LS we pre-assign the tasks in different groups. In phase 2 we use online LS to schedule tasks to processors within each group.

Theorem 3.1: When the number of groups is k the approximation ratio is $\frac{k\alpha^2}{\alpha^2+k-1}[1+\frac{k-1}{m}] + \frac{m-k}{m}$

Proof: We have k groups of m/k machines. We have restriction that each task can be assigned to only one of these groups. In Phase 1, each task is assigned to different groups. In Phase 2, tasks are scheduled online within repective group.

Let us consider that C_{max} comes from group $G1$. Also, taking the property of List Scheduling that the load difference between any two groups cannot be greater than the largest task. So, for any group $Gl \neq G1$, We have

$$|\sum_{i \in G1}\tilde{p}_i - \sum_{i \in Gl}\tilde{p}_i| \leq max_{i \in T}\tilde{p}_i$$

for all, $l = 2, 3, ..., k$

Adding for all values of l, We have

$$|(k-1)\sum_{i \in G1}\tilde{p}_i - \sum_{l=2}^{k}\sum_{i \in Gl}\tilde{p}_i| \leq (k-1)max_{i \in T}\tilde{p}_i$$

$$\Rightarrow \sum_{l=2}^{k}\sum_{i \in Gl}\tilde{p}_i \geq (k-1)[\sum_{i \in G1}\tilde{p}_i - max_{i \in T}\tilde{p}_i]$$

As the actual processing time of tasks can vary within a factor α and $\frac{1}{\alpha}$ of their estimated processing time, the following inequality holds

$$\alpha\sum_{l=2}^{k}\sum_{i \in Gl}p_i \geq (k-1)[\frac{1}{\alpha}\sum_{i \in G1}p_i - \alpha max_{i \in T}p_i]$$

$$\sum_{l=2}^{k}\sum_{i \in Gl}p_i \geq (k-1)[\frac{1}{\alpha^2}\sum_{i \in G1}p_i - max_{i \in T}p_i] \tag{8}$$

In phase 2, We are applying LS on online mode. We assume that C_{max} comes from $G1$. Using LS property we can write,

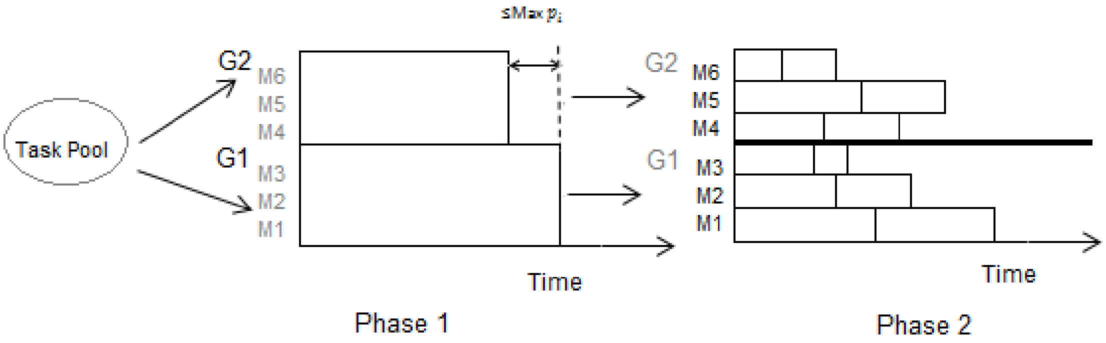

Figure 2. Model 3

$$C_{max} \leq \frac{\sum_{i \in G1} p_i}{m/k} + \frac{m/k - 1}{m/k} p_{max} \tag{9}$$

Also, C^*_{max} must be greater than the average of the loads on machines.

$$C^*_{max} \geq \frac{\sum_{i \in T} p_i}{m}$$

$$\Rightarrow C^*_{max} \geq \frac{\sum_{i \in G1} p_i + \sum_{l=2}^{k} \sum_{i \in Gl} p_i}{m}$$

from (11), we derive

$$\Rightarrow C^*_{max} \geq \frac{\sum_{i \in G1} p_i + (k-1) \left[\frac{1}{\alpha^2} \sum_{i \in G1} p_i - max_{i \in T} p_i\right]}{m}$$

$$\Rightarrow m\alpha^2 C^*_{max} + \alpha^2 (k-1) max_{i \in T} p_i \geq \alpha^2 \sum_{i \in G1} p_i + (k-1) \sum_{i \in G1} p_i$$

$$\Rightarrow \frac{\alpha^2}{\alpha^2 + k - 1} \left[m C^*_{max} + (k-1) max_{i \in T} p_i\right] \geq \sum_{i \in G1} p_i \tag{10}$$

Using (12) and (13), We have

$$C_{max} \leq \frac{k\alpha^2}{\alpha^2 + k - 1} \left[C^*_{max} + \frac{(k-1)}{m} max_{i \in T} p_i\right] + \frac{m/k - 1}{m/k} p_{max}$$

As $C^*_{max} \geq max_{i \in T} p_i \geq p_{max}$, we have

$$C_{max} \leq \frac{k\alpha^2}{\alpha^2 + k - 1} \left[C^*_{max} + \frac{k-1}{m} C^*_{max}\right] + \frac{m-k}{m} C^*_{max}$$

So we have approximation ratio,

$$\frac{C_{max}}{C^*_{max}} \leq \frac{k\alpha^2}{\alpha^2 + k - 1} \left[1 + \frac{k-1}{m}\right] + \frac{m-k}{m}$$

Corollary 3.1: When the number of groups is 2 the approximation ratio is $1 + \frac{2}{1+\alpha^2}(\alpha^2 - \frac{1}{m})$

VI. Summary

The Table I summarizes the results in term of approximation ratio given by the three models.

Replication	Approximation ratio
$\lvert M_j \rvert = 1$	$\frac{C_{max}}{C^*_{max}} \leq \frac{2\alpha^2 m}{2\alpha^2 + m - 1}$ [Th. 1.2] No approximation better than α^2 [Th. 1.1]
$\lvert M_j \rvert = \lvert M \rvert$	$\frac{C_{max}}{C^*_{max}} \leq 1 + (\frac{m-1}{m})\frac{\alpha^2}{2}$ [Th. 2.1]
$\lvert M_j \rvert = m/k$	$\frac{C_{max}}{C^*_{max}} \leq \frac{k\alpha^2}{\alpha^2+k-1}\left[1 + \frac{k-1}{m}\right] + \frac{m-k}{m}$ [Th. 3.1] $\frac{C_{max}}{C^*_{max}} \leq 1 + \frac{2}{1+\alpha^2}\left(\alpha^2 - \frac{1}{m}\right)$ when $k = 2$ [Col. 3.1]

Table I
Summary Table

The figure **??** shows ratio- replication graphs for each of the models for different values of α. We vary the value of α while keeping the number of machines fixed, $m = 210$. For replication everywhere scenario the number of replication is full and is always equal to

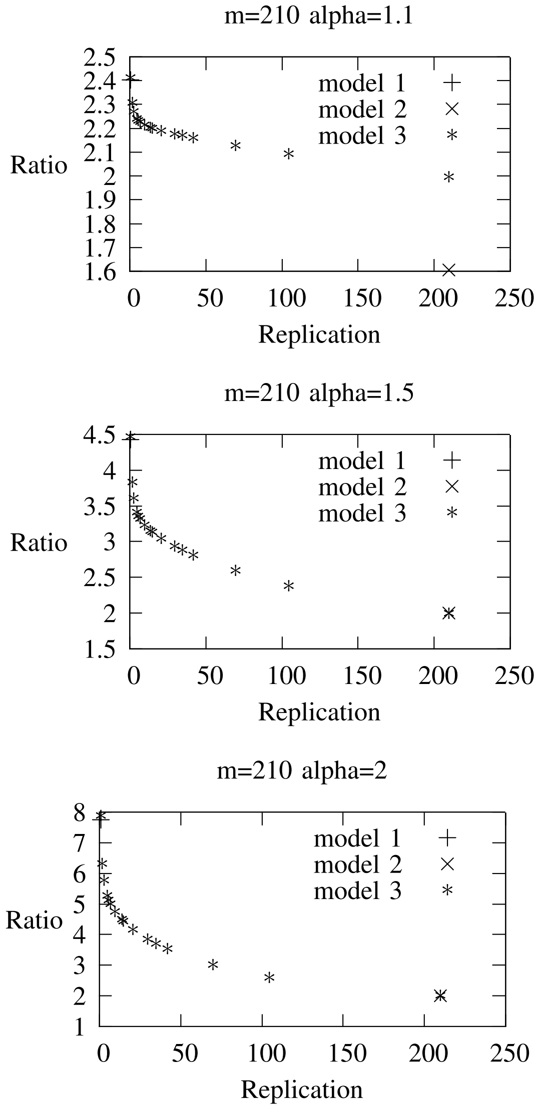

Figure 3. Ratio-Replication graph with m=210 and value of α as 1.1, 1.5 and 2

total number of machines. For no replication scenario the approximation ratio is fixed for each value of α. For the model where replication is allowed within the group the approximation ratio decreases significantly for small replication. We observe that after a significant number of replications the ratio is not much improved further.

References

[1] Da Wang, Gauri Joshi, Gregory Wornell, *Efficient Task Replication for Fast Response Times in Parallel Computation*

[2] Matteo Dell'Amico, Damiano Carra, Mario Pastorelli, Pietro Michiardi, *Revisiting Size-Based Scheduling with Estimated Job Sizes*

[3] A. Wierman, M. Nuyens, *Scheduling despite inexact job-size information*

[4] J. Wolf, D. Rajan, K. Hildrum, R. Khandekar, V. Kumar, S. Parekh, K.- L. Wu, and A. Balmin, *FLEX: A slot allocation scheduling optimizer for MapReduce workloads*

[5] M. Pastorelli, A. Barbuzzi, D. Carra, M. Dell'Amico, and P. Michiardi, *HFSP: size-based scheduling for Hadoop*

[6] B. Schroeder and M. Harchol-Balter, *Web servers under overload: How scheduling can help*

[7] Louis-Claude Canon and Emmanuel Jeannot, *Evaluation and Optimization of the Robustness of DAG Schedules in Heterogeneous Environments*

[8] Michael Gatto and Peter Widmayer, *On the robustness of Graham's algorithm for online scheduling*

[9] Wagelmans A.,*Sensitivity Analysis in Combinatorial Optimization*

[10] Frédéric Guinand, Aziz Moukrim and Eric Sanlaville, *Sensitivity Analysis of Tree Scheduling on Two Machines with Communication Delays*