International Journal of Electrical and Computer Engineering (IJECE)
Vol. 5, No. 3, June 2015, pp. 477~482
ISSN: 2088-8708

Improved Rejection Penalty Algorithm with Multiprocessor Rejection Technique

Prativa Satpathy, Kalyan Das, Jagmohan Padhi
Departement of Computer Science, Sambalpur University Institute of Information Technology,
Burla, Sambalpur, Orissa, India

Article Info

Article history:

Received Feb 22, 2015
Revised Apr 5, 2015
Accepted Apr 20, 2015

Keyword:

Competitive ratio
Improved rejection penalty
Makespan
Off-line scheduling
On-line scheduling
Rejection penalty

ABSTRACT

This paper deals with multiprocessor scheduling with rejection technique where each job is provided with processing time and a given penalty cost. If the job satisfies the acceptance condition, it will schedule in the least loaded identical parallel machine else job is rejected. In this way its penalty cost is calculated. Our objective is to minimize the makespan of the scheduled job and to minimize the sum of the penalties of rejected jobs. We have merged 'CHOOSE 'and 'REJECTION PENALTY' algorithm to reduce the sum of penalties cost and makespan. Our proposed 'Improved Reject penalty algorithm' reduce competitive ratio, which in turn enhances the efficiency of the on-line algorithm. By applying our new on-line technique, we got the lower bound of our algorithm is is 1.286 which is far better from the existing algorithms whose competitive ratio is at 1.819. In our approach we have consider non-preemption scheduling technique.

Corresponding Author:

Prativa Satpathy,
Departement of Computer Science,
Sambalpur University Institute of Information Technology,
Jyotivihar, Burla, Sambalpur, India
Email: prativa.satpathy30@gmail.com

1. INTRODUCTION

Real time scheduling decides which of the processes in the ready queue is to be allocated to the processor and the result depends on both functional accuracy as well as time required to deliver the result.
Types of Scheduling:
i) Online Scheduling ii) Offline Scheduling
Online algorithm processes its input piece-by-piece in a serial fashion, without having the entire input available from the starting. Online scheduling algorithms make their scheduling decisions at runtime, as a result of which there can be significant overheads because of runtime processing where scheduling decisions are based on dynamic parameters may change.

Offline scheduling has complete knowledge of the task sets and its constraints; such as deadlines, computation times, precedence and constraints, much before any decision that is made and it doesn't depend on time. Scheduling decisions are based on fixed parameters and assigned to tasks much before their activation.We have used a criterion to measure the performance of online algorithm called "competitive ratio". This is just like the approximation ratio, comparing the objective value obtained by online algorithm and that of optimal offline algorithm.

Quality of an on-line algorithm is tested by evaluating its worst case analysis so that the competitive ratio is between on-line performances of the algorithm to its optimal off-line performance.

Mathematically: - $\boxed{\text{Competitve Ratio} = \frac{\text{Online performance}}{\text{Optimal offline performnce}}}$

Performance criteria of on-line scheduling depend on CPU utilization, Throughput, Turn-around time (TAT), Waiting time (WT).

R. L. Graham et al. [1] introduced the first deterministic on-line algorithm called as list scheduling online problem in which he gave a competitive ratio of (2-1/m) where 'm' denoted the number of machines. The scheme proposed by D. D. Sleator and R. E. Tarjan [2], addressed about the amortized complexity of Least Recently Used algorithm, proving that the efficiency of the proposed algorithm differs from that of the offline paging rule (Belady's MIN algorithm), and by a factor that depends on the size of fast memory.
D. R. Karger et al. [3] suggested the competitive ratio of upper bound to be at around 1.945.
Y. Bartal et al. [4], introduced a randomized online algorithms for a value of m = 2, where they achieved an optimal competitive ratio of 4/3. The research carried out by S Albers [5], gavethe most popular better bound scheduling problem - the fundamental problem of on-line algorithms, where a sequence of jobs has to be scheduled on 'm'- identical parallel machines to minimize the makespan. S Albers [6] proposed an algorithm called as the M2 algorithm which gave competitive ratio of upper bound as 1.923.

S. S. Seiden [7], introduced a new concept of online randomized multiprocessor scheduling in which he gave a randomized algorithm for a value of $m \geq 3$ number of machines. In [8] the authors introduced a new version of multiprocessor scheduling with the special feature where the proposed jobs might be rejected at a certain penalty. The objective of the scheduling technique was to minimize the makespan of the schedule for an accepted job as well as to minimize the sum of the penalties for rejected jobs. The algorithm $RP(\alpha)$ hinted at a lower bound of 1.819 for m=3 machine. In [9] authors introduced a deterministic constant competitive online algorithm which scheduled a sequence of jobs on a processor running at variable speed so as to minimize the cost of power consumption and the total flow time for all the jobs.Tamas Nemet and Csana dImreh [10] proposed an algorithm called 'CHOOSE' which gave better return values for the parameter α, and, in turn reduced the makespan of the algorithm with Multi-processor-rejection technique. In [11] authors found a new lower bound for minimization of makespan for a few number of uniformly related machine.

2. RESEARCH METHOD

In the existing $RP(\alpha)$ algorithm, there is an open problem area for improving the (α) value so that makespan is minimized, resulting in the automatic reduction of the competitive ratio as both the parameters are directly proportional to each other. The objective of the paper is to minimize the makespan, which is defined as the total completion time for all accepted jobs. It also minimizes the sum of the penalties of all rejected jobs.

Notation and preliminaries:

J = Set of jobs (Each job has its own processing time and penalty = (p_j, w_j)
$p_j =$ Processing time of each job.
w_j = Penalty of each job.
If $w_j \ll p_j$then the job is of rejected job otherwise accepted job.
m = number of machines.
M (A) = Length of summation of all processing time of chosen heavily loaded machine
$\emptyset$ = Golden Ratio = 1.618
$\alpha = \emptyset - 1 = 1.618 - 1 = 0.618$.
p_l= Largest Processing time between all jobs.
c = Competitive Ratio =$\frac{Z^{ON}}{Z^{OPT}}$.

Previous algorithms:

$RP(\alpha)$Algorithm:

Each job is having a processing time and a penalty value, i.e. Job = (p_j, w_j) .
α = 0.618 that is $\emptyset$-1= α, which is a parameter used to get accepted and rejected jobs.
Taking 'w'as the penalty of each job, a job $j = (p_j, w_j)$ becomes available,
The job is rejected if $w_j \leq \alpha . p_j$, else it is accepted and scheduled on a least loaded machine.

$Z^{ON}$is the Online Makespan Performance which is equivalent to the Highest Load of summation of accepted jobs on the processing time of each machine + summation of Penalties of rejected jobs.
Z^{OPT} = M (A) + (1-1/m) = p_l<= M (A) + (1-1/m) Z^{OPT}
c = Competitive Ratio =$\frac{Z^{ON}}{Z^{OPT}}$.

CHOOSE Algorithm:
For given I=1, 2…10 Jobs, Generate one element from the intervals [(i-1)/10, i/10] by uniform distribution of processing times.
Store all the values on Set {S1}.
Using $RP(\alpha)$technique, where $\boldsymbol{\alpha}$ =0.618, find smallest cost among I, denote it as $\boldsymbol{\alpha}$***.**
In the same way generate one element from the interval [α - i/100 , α - (i-1)/100] ,[α + (i -1)/100, α + i/100] , [$\alpha *$ - i/100 , $\alpha *$ - (i-1)/100] ,[$\alpha *$ + (i-1)/100, $\alpha *$ + i/100] for I=1,2….10.
Store all the values on Set {S2}.
Using $RP(\alpha)$ algorithm, from {S1}, {S2}, {α} and{ $\alpha *$}, get the smallest cost value for new alpha parameter.

3. PROPOSED ALGORITHM

Our Proposed 'Improved Rejection Penalty (IRP)' algorithm is a combination of both 'CHOOSE' algorithm and 'RP (α)' algorithm.
Input to our algorithm is number of Jobs with each job having its processing time denoted by (p_j) and penalty (w_j), so $j = (p_j, w_j)$.
When 'CHOOSE' algorithm is applied, it will generate one element from the given interval [(i-1)/10, i/10] for i= {1, 2… 10) number of jobs where processing time with penalty cost of each job (penalty will be is calculated by multiplying processing time of the job with α (0.618), taken from RP (α) algorithm. After putting all the jobs in the above interval, the result is stored in Set {S1}, and a value for α*(smallest cost value of {S1}) is generated.
Another interval [α - i/100 , α - (i-1)/100], [α + (i-1)/100, α + i/100], [$\alpha *$ - i/100 , $\alpha *$ - (i-1)/100] ,[$\alpha *$ + (i-1)/100, $\alpha *$ + i/100] is applied for the same job and then the value is stored in another Set {S2}.
From Set {S1}, {S2}, {α}, {α*}, which is having less cost value, that is taken as new alpha (α1).
After getting new alpha value from 'CHOOSE' algorithm, check the condition using new alpha value, If $w_j \leq \alpha 1 . p_j$, Reject the job, otherwise accept it and schedule it in the least loaded machine.
Then calculate the On-line makespan that is: - On-line makespan (Z^{ON}) = Sum of processing time of heavily loaded machine + Sum of Penalties of all rejected jobs.
Z^{OPT} =M (A) + (1 – 1 / m) Pi < = M (A) + (1 – 1 /m) Z^{OPT}this equation is taken from existing algorithm; we have taken the same off-line technique.
$c = Z^{ON}/_{Z^{OPT}}$ = Competitive Ratio. (For finding the values of optimal offline, no rejection is required)

Pseudo Code for our proposed algorithm:

```
Input:
    Number of jobs (i1, i2, i3…in)
    Randomly generated processing time and penalties for each jobs $(p_j, w_j)$
    Fixed number of machines (m)
Output:
    To minimize Competitive-ratio
    To minimize Makespan
    To minimize Penalties
Method:
At the beginning use 'choose' algorithm and after calculating all intervals, find smallest cost value and denotes it as α1.
Give the random processing time and the penalty to each job and then apply it to the rejection technique,
    If ($w_j \leq \alpha 1 . p_j$),
        Reject the job
    Else
        Accept the job and schedule it to the least loaded machine 'm'
Calculate the sum of processing time of accepted jobs of heavily loaded machine and penalties sum of rejected jobs.
On-line makespan ($Z^{ON}$) = Sum of $p_j$ of heavily loaded machine + Sum of $w_j$ of rejected jobs.
```

Calculate Off-line makespan (Z^{OPT})
Calculate c = Competitive Ratio $=\frac{Z^{ON}}{Z^{OPT}}$.

4. RESULTS AND ANALYSIS

For our experiment we have generate random processing time and penalties for each jobs. We have applied nearly about 50000 jobs and calculate the penalties, makesapn, competitive ratio of both Existing $RP(\alpha)$ algorithm and our proposed IRP algorithm. Improved Reject penalty (IRP) gives less competitive ratio, makespan and penalties cost.

Initially we have generated interval for CHOOSE algorithm, after calculating the entire interval for from {S1}, {S2} by uniform distribution where {S1} and {S2} representing two sets, and given penalties for I= {1, 2, 10} jobs, the lowest cost value is taken for the new parameter '$\alpha 1$', which varies in different penalties sequence is then processed with CHOOSE algorithm. For one given input of penalties, the new alpha $'\alpha 1'$ parameter is calculated as '0.090' which is better than existing alpha = 0.618. Appling new alpha value in $RP(\alpha)$algorithm we found the summation of penalties is minimum in our case, which reduces the makespan and competitive ratio in our proposed algorithm. For a given range of job with random processing time and penalties, we found our competitive ratio is lower than existing $RP(\alpha)$ for fixed number of machine.

Data set:

In Our approach we have randomly generate processing time between 1 to 500 and penalties between 1 to 200 which is applied on both existing and proposed IRP method for 3 machines and the calculated makespan, penalties and competitive ratio, are given in Table 1. Figure 1 compares the Competitive ratio between RP and IRP. Table 2 shows the calculated competitive ratio using existing and IRP algorithms with 3, 10, 100 machines and 10, 100, 1000 jobs and in Table 3 comparative result of lower bound in general for 3 machines is shown.

Table 1. Results of makespan, penalties and competitive ratio of 3 machines using RP and IRP

NUMBER OF JOBS	EXISTING RP ALGORITM (Competitive ratio)	PROPOSD IRP ALGORITHM (Competitive ratio)	RP ALGORIM (Penalties sum)	IRP ALGORITM (Penalties sum)	RP ALGORITHM (on-line makespan	IRP ALGORITHM (on-line makespan)
10	1.8218	1.224	125	21	317	213
100	16.05	8.24	1672	197	3194	1720
500	82.95	49.54	7749	1089	16599	9930
1000	167.87	99.97	15829	2262	33576	19674
10000	1619.54	958.224	153384	21154	323457	198789
50000	7.97316e+03	4.714030+03	759366	107539	1594633	942806

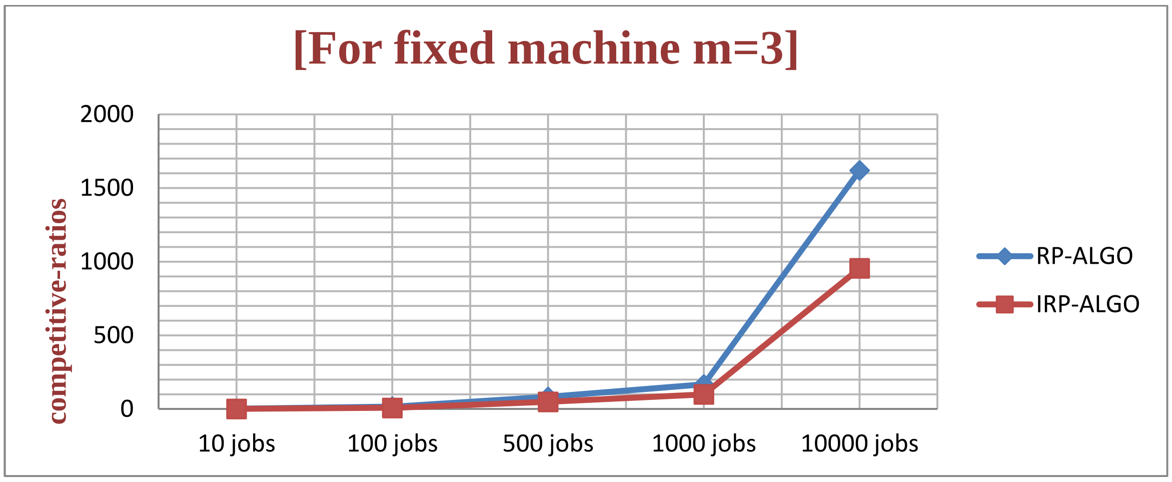

Figure 1. Comparison of Competitive ratio between RP and IRP

Table 2. Comparison of competitive ratio using RP and IRP algorithms with 3, 10, 100 machines and 10, 100, 1000 jobs

Fixed no. of machine	RP-for 10jobs	IRP-for 10jobs	RP-for 100jobs	IRP-for 100jobs	RP-for 1000jobs	IRP-for 1000jobs
M= 3	1.821	1.224	16.05	8.24	167.87	99.97
M=10	1.494	1.047	11.130	3.723	106.12	38.235
M=100	1.435	1.012	9.361	1.954	82.41	14.525

Table 3. Comparative result of lower bound in general for fixed machine m=3

LOWER BOUND OF RP-ALGORITHM	LOWER BOUND OF IRP-ALGORITHM
1.819	1.286

5. CONCLUSION

The on-line algorithm with less competitive ratio and makespan gives a better performance for real time data's in different real time on-line scheduling. A new on-line algorithm called "Improved Rejection Penalty" is introduced to enhance the on-line efficiency as compare to the existing algorithm for fixed number of machine (3) with randomly generated jobs, processing time and penalties. We get lower bound of our proposed IRP algorithm as 1.286 which is much better than the lower bound of existing algorithm is ie. 1.819 and it is simultaneously minimizing three objective factors, such as competitive ratio, makespan and penalties as compared to the existing RP algorithm.

The future work can be carried out in the following directions:

- Lower bound of the algorithm can be further reduced by applying better technique.
- Offline algorithm heuristics technique can be improve.
- It can be scheduled with arbitrary number of machines.
- Reducing time and space complexity for millions of jobs.
- It can be implemented using preemption techniques.

REFERENCES

[1] R.L. Graham, "Bounds for Certain Multiprocessor Anomalies", Bell System Technical Journal, Nov. 1966.

[2] D.D. Sleator and R.E. Tarjan, "Amortized Efficiency of List Update and Paging Rules", Comm. Association Computing Machinery, 1985.

[3] D.R. Karger, S.J. Phillips and E. Torng, "A Better Algorithm for an Ancient Scheduling Problem", Journal of Algorithms, 1996.

[4] Y. Bartal, A. Fiat, H. Karloff and R. Vohra, "New Algorithms for an Ancient Scheduling Problem", Journal of Computer and System Sciences, 1995.

[5] S. Albers, "Better bounds for online scheduling", *SIAM Journal on Computing*, 1999.

[6] S. Albers, "Better Bounds for Online Scheduling", Proceedings of the 29th Annual ACM Symposium on Theory of Computing, ACM, 1999.

[7] S.S. Seiden, "Online Randomized Multiprocessor Scheduling", Algorithmica, 2000.

[8] Y. Bartal, et al., "Multiprocessor Scheduling with Rejections", Proc. of the 10th *Conference* on Computability in Europe (CiE) 2001.

[9] S. Albers and H. Fujiwara, "Energy-efficient algorithms for flow time minimization", *Proc. 23rd International Symposium on Theoretical Aspects of Computer Science (STACS)*, Springer LNCS, 2006.

[10] T. Nemeth and C.Imreh,"Parameter Learning on-line algorithm for Multiprocessor Scheduling with Rejection", Acta Cybernetica 2009.

[11] J. Schwartz, "Lower bounds for online makespan minimization on a small number of related machines", Springer 2013.

BIOGRAPHIES OF AUTHORS

Prativa Satpathy received B.Tech degree from Biju Patnaik University of Technology, Rourkela in 2012. She is currently pursuing her M.Tech from Sambalpur University Institute of Information Technology, Burla. Her research interests are in wireless sensor network and online scheduling algorithm.

Kalyan Das received the B.Tech degree from Berhampur University, Berhampur in 2005. He received M.Tech degree from People Education Society Institute of Technology, Bangalore in 2014. He was working as as associate system engineer in IBM India Pvt. Ltd. Currently he is working as an assistant professor in Sambalpur University Institute of Information Technology, Burla.
His research interests are in wireless sensor network and online scheduling algorithm.

Jagamohan Padhi received M.Sc degree from Sambalpur University Institute of Information Technology, Burla. He is currently pursuing his PhD in Electronics from Sambalpur university Institute of Information Technology, Burla. His research interests are in bioinformatics, online algorithm.