A Threshold Based Dynamic Routing for Jobs with QoS Ranking

Seyed Behrouz Khodadadi[1] and Jafar Razmi[1]

[1] Department of Industrial Engineering, College of Engineering,
University of Tehran, Iran
{ jrazmi,, and bkhodadadi }@ut.ac.ir

Abstract: We consider a set of n heterogeneous servers which differ in processing time and Quality of Service (QoS). Jobs are divided into m levels with regard to their service ranks. We present a Threshold Policy (TP) depending on number of different jobs in the queues as a practical and flexible dynamic routing policy to control the QoS. Two performance measures are discussed: the QoS and the Average Waiting Time (AWT) in the queues. The TP is compared with both a static routing policy which maximizes the QoS level and the Minimum Expected Delay (MED) policy which minimizes the AWT. Numerical example validate that the proposed TP is more effective when both measures are considered. The TP balances the trade-off between QoS and AWT and therefore it is superior to the MED policy and any static routing which keeps the QoS in a certain level.

Keywords: Dynamic Routing, Quality of Service, Heterogeneous Servers, Threshold Base Policy

1 Introduction

Routing policies have widely used to control queuing systems and they can highly improve the system performance. The application of routing policies arise in several areas such as manufacturing industries, computer and communication systems. We deal with dynamic routing of jobs among heterogeneous parallel servers which are ranked by their QoS. Jobs are also divided into several levels of service. Jobs of different quality levels enter to the system with different stream of arrivals and a router must dispatch the jobs to one of the servers immediately after their arrival. Service times are independent of arrival process and job's level.

Routing policies are concerned with the assignment of arrived jobs to the queues of machines. Mostly the assignment decision should be made immediately after job arrival and it is a routing decision based on number of jobs in each queue [1]. Obviously there is no routing policy which can optimize various performance measures all together. There has been presented variety of policies by investigators, which each of them has its own performance measures and cost functions and is appropriate for specific problems. One common objective function which is frequently discussed by researchers is waiting time minimization. In this direction, the JSQ policy for identical parallel machines has been investigated by Winston [2] and

Nelson and Philips [3],[4]. The performance of the Minimum Expected Delay (MED) policy which is derived from JSQ policy and is appropriate for non-identical parallel machines - also called heterogeneous servers - is analyzed by Lui et al. [5].

Routing policies are divided into static policies and dynamic policies. In static scheme no information exchanges are used at decision epoch. Static policies are appropriate when all jobs are available at time zero or situations in which the system characteristics are under the control. For instance the Joining-Shortest-Queue (JSQ) policy assigns each job to the machine with the minimum number of waiting jobs.

Threshold-based policies, which are a kind of dynamic routing, are frequently used to control queuing systems. In this mode, the analysis of heterogeneous servers is mainly limited to two servers because of high complexity of such systems. ([6], [7], [8], [9], [10] and [11]). In this paper we consider several streams of arrivals, parallel queues and several heterogeneous servers.

It is common in manufacturing, service and computer systems to categorize jobs or customers and rank them in order of their importance. Usually the QoS is not at the same level for different servers. In this scheme, more valued jobs are processed by higher quality machines. On the other hand, it is a common practice to provide high quality services even for the low graded jobs as much as possible. An example of this situation is dispatching automobile bodies to paint shops in Iran Khodro Company (IKCO) - the biggest car producer factory in the Middle East. Each paint shop has a specific production rate and QoS due to its technology specifications. Different model of automobiles can be painted in any of paint shops but usually the capacities of higher quality paint shops are assigned to higher valued automobiles. Each automobile body is released from its own body shop. Conveyors carry automobile bodies from body shops to White Body Stocks. Finally automobile bodies are dispatched to paint shops regard to their quality level. To the best of our knowledge, analysis of QoS performance measure has not been studied in dynamic routing policies. Most of researches are focused on AWT and utilization balancing analysis.

The presentation of the rest of this paper is organized as follows. In section 2 we present the problem statement. Section 3 contains the proposed algorithm and the development process. Numerical examples are presented in section 4 and conclusions are stated in section 5.

2 Problem Statement

The problem regarded in this paper consists of a set of n parallel heterogeneous servers and m level of jobs. All servers can perform the same operation, but with different performances i.e. they have different service time and QoS. Servers are arranged in order of their quality ranking. The quality ranking is in increasing order from 1 to n whereby top ranked server indicates that the server has greater quality. In the same manner jobs are ranked in order of their quality level from 1 to m. There is a separate stream of arrivals for each level. Arrivals of each level of jobs are independently distributed. The arrival of a new job occurs only after the current job is dispatched. The dispatched job waits in the queue of corresponding server until the server becomes idle. Queues are assumed to have infinite capacity and their discipline

is FCFS. No preemption or jockeying is allowed. Each job is processed in only one server. The service times are independently distributed and they are also independent of the job's quality level. Table 1 summarizes the main parameters of the model.

Table 1. Model parameters

Parameter	Description	Range
i	Job rank	1,2..m
j	Server rank	1,2..n
t_i	Inter arrival time of jobs with rank i	> 0
T_j	Service time of server j	> 0
T	Processing period	> 0
P_i^T	Total number of processed jobs with rank i in period T	0,1..∞
$p_{i,j}^T$	Processed jobs with rank i in server j in period T	0,1..Pi
$q_{i,j}$	Number of jobs with rank i waiting in queue of server j	0,1..∞

The necessary condition for stability is as follows ([12]).

$$\sum_{j=1}^{n} \frac{1}{T_j} > \sum_{i=1}^{m} \frac{1}{t_i} \tag{1}$$

3 Proposed Method

3.1 Threshold Policy

We propose a TP based on the number of jobs with different levels in each queue. When a job with quality level *i* arrives, the number of jobs with levels 1 to *i* which currently await service in the queue of the first ranked server are counted and the summation is compared with a predefined threshold. If the summation is smaller than the threshold, then the job is dispatched to the server; otherwise the summation of jobs with levels 1 to *i* await service in the queue of the second ranked server is compared with its corresponding threshold. The above routine continues until a server is selected and the job is dispatched. The procedure of the proposed threshold routing is depicted in procedure 1.

```
Procedure 1
S = ∅
For j = 1 to n-1
      If  ∑_1^i q_{i,j} < c_{i,j}  then S=j and break
End for
If S = ∅ then S=n
```

Where S denotes the selected server. $c_{i,j}$ is a constant threshold and it is a cell of matrix C . Note that the last machine is a reserved machine in case of heavy traffic conditions. If none of j=1 to n-1 servers are selected for a job then the job is dispatched to the last server. Thus we have excluded the last column of matrix C. If we consider $c_j=(c_{0,j}, c_{1,j}, \ldots, c_{m,j})^T$ then $c_{i,j}$ values are integer and c_j is in non-increasing order. Our main idea to find effective thresholds is to solve the problem for deterministic situation and use the same solutions for stochastic model. In this section we present an algorithm for the deterministic model which determines the thresholds so that the total QoS is maximized. In this direction the following statement determines a feasible solution.

$$\forall\{j, j' < j\}\ if \lim_{T\to\infty} utilization(j) > 0\ then \lim_{T\to\infty} utilization(j') = 100\% \qquad (2)$$

Statement (2) denotes that the lower quality machines should not be used unless the higher quality ones are completely utilized. In deterministic model, t_i and T_j are assumed to be constant values. Using the Kendall [13]'s notation the model can be described as *D/D/n* with m arrival streams. The following statement represents an optimal assignment for the deterministic model.

$$\forall i, j, T\ if\ p_{i,j}^T > 0\ then \sum_{x=i+1}^{x=m} \sum_{y=1}^{y=j-1} p_{x,y}^T = 0 \qquad (3)$$

Statement (3) implies that the high ranked jobs should be prior in assigning to higher quality machines. In the other word, if a server with a short service time has low quality, it is not allowed to dispatch any jobs to it unless constraint (2) is satisfied. Thus, statement (3) is sufficient to consider a solution as optimal. It is noteworthy that an easy way to reach optimal solutions is to use zero or big integer values for the thresholds to force the system to dispatch jobs to a specific server or avoid dispatching to some other servers. Such thresholds actually function like static rules and therefore reduce the flexibility of the routing method. The main reason for development of the proposed threshold policy is to construct a flexible routing policy for stochastic situation. Therefore it is not appropriate to use extreme values.

3.2 Initial Solutions

By solving a variety of problems, we concluded a general initial solution for threshold values which has advantages over random ones. Procedure 2 shows the initialization procedure. The proposed initial solution not only makes the algorithm reaching the optimal solution earlier, but also increases the flexibility of routing policy under uncertainty.

```
Procedure 2
For x = 1 to m
  For y = 1 to n-1
      If x <= m-n+j then cx,y = n-j+1 else cx,y = m-x+1
```

```
  End for
End for
```

3.3 Objective Function Value

We carried out a simulation model to evaluate the performance of the proposed routing policy against the optimal assignment. The simulation model evaluates the OFVs and indicates the critical servers which have discrepancies in relation to optimal ones. The OFV is defined for each machine as stated in procedure 3. The routine counts the number of discrepancies between the optimal assignment and the dynamic routing policy for each server. The objective of the proposed algorithm is to minimize the discrepancies of all servers ($\sum_{j=1}^{n} OFVc_j$=0). The simulation model also determines the critical jobs for each critical server. Jobs which are assigned wrongly to a critical server are called excessive jobs. Jobs which are not assigned to the correct server are called lacking jobs.

```
Procedure 3
For i=1 to m
```

$$\text{if } p_{i,j}^{T} > 0 \text{ t en } OFVc_j = OFVc_j + \min\left(\sum_{x=i+1}^{x=m}\sum_{y=1}^{y=j-1} p_{x,y}^{T}\ , p_{i,j}^{T}\right)$$

```
End for
```

$$\text{if } OFVc_j > 0 \text{ t en } j \in \{\text{critical servers}\}$$

3.4 Performance Measures

Two performance measures are considered, first the percentage of incorrect assignments to total assignments which is denoted by $FV = 100 \times \sum_{j=1}^{n} OFVc_j / \sum_{i=1}^{m} P_i^T$, and second the AWT. The proposed threshold policy tends to maximize the QoS but it does not deal with other performance measures such as waiting time. Thus, a policy should be developed to reach a desirable level for both QoS and AWT.

3.5 Tune-up

An algorithm is developed to tune up the threshold policy. The algorithm consists of an initialization stage and four repetitive steps. The proposed algorithm is as follows.

Step 0: procedure initialization

Step 1: if the solution is feasible and $\sum_{j=1}^{n} OFVc_j = 0$ then C is an optimal solution, stop. Else if the solution was infeasible go to step 2. Else go to step 3

Step 2: select server y={min(j)|$j \neq n$ and utilization(j)<100%} . select job x={min(i)|$p_{i,y+1}$>0}. Make an increase of 1 in cell $c_{x,y}$ of matrix C. update values of $c_{i,y}$ for i<x and go to step 1.

Step 3: select the first critical server (y= min{critical servers}). For the selected server select the first lacking job (x=min{lacking jobs of server y}). make an increase of 1 in cell $c_{x,y}$ of matrix C. if the $OFVc_y$ is decreased update values of $c_{i,y}$ for $i>x$ so and go to step 1 else $c_{x,y}$-- and go to step 4.

Step 4: select the first critical server (y= min{critical servers}). For the selected server select the last excessive job (x=max{excessive jobs of server y}). make a decrease of 1 in cell $c_{x,y}$ of matrix C. if the $OFVc_y$ was decreased update values of $c_{i,y}$ for $i<x$ and go to step 1 else $c_{x,y}$++ and go to step 1.

In the first step the OFVs are calculated and the critical servers and critical jobs are determined. If the current solution is both feasible and optimal the algorithm is stopped and the current solution is regarded as optimal solution, otherwise the algorithm tries to overcome the discrepancies of job assignment in a top-down order. If the current solution is not feasible (i.e. there is at least one server that its utilization is not equal to 100% excluding the last server), then step 2 aims to reach a feasible solution by increasing the corresponding thresholds. There are two reasons that may cause a solution not to be optimal. First, a job with a low quality level has been wrongly assigned to a high quality server and second, a job with a high quality level has been wrongly assigned to a low quality server. To resolve these mismatches, the algorithm decreases/increases the threshold of corresponding jobs (step 3 and 4).

4 Numerical Examples

To evaluate the performance of the justified TP presented in section 3.3, it was compared with a static routing policy, the modified version of static policy and the MED policy. The static routing policy maximizes the quality of service whereas the MED policy minimizes the AWT. The modified static policy keeps the quality in a lower level in comparison with the static policy. Instead, in stochastic models the system is stable by using the modified version. We consider three types of uncertainty: *stochastic arrivals, stochastic service times* and *unexpected breakdowns*. To put it more clearly, a numerical experiment is addressed as follows

Table 2. Simulation results.

	D/D/n		M/D/n		D/M/n		M/M/n		M/M*/n	
Policy	OFV	AWT	OFV	AWT	OFV	AWT	OFV	AWT	OFV	AWT
Static	0	274.8	0	55280.2	0	40214.2	0	98901.7	0	∞
MS	8.1	551.53	8.1	1308.06	8.1	1638.68	8.1	2556.6	8.1	11553.8
TP	0	641.4	11.7	742.4	11.6	828.4	17.1	1236.7	23.8	3931.8
MED	16.3	233.0	26.6	428.4	28.9	453.9	28.4	775.4	32.4	1893.0

As it is shown in table 2, five different systems are considered. The exponential distribution is used for stochastic systems. m, n and $E(T_j)$ are identical in each of five systems. To keep the traffic volume in a constant level while a variety of problems to be evaluated, ten random combinations of arrival times were generated so that $\sum_{i=1}^{m} \frac{1}{E(t_i)}$ was considered fixed. The system parameters are (m,n)=(4,3), the

deterministic Tj is {288,576,432}, the stochastic Tj is {expo(288), expo(576), expo(432)}, {mttf, mttr} = {expo(7200), expo(600)} and $\sum_{i=1}^{m} \frac{1}{E(t_j)} = 6.366 \times 10^{-3}$.The simulation model was run for each of five systems for 10000 hours with 2 hours warm-up period. The average OFVs and the average of AWTs for four policies are shown in table 2.

As it has been mentioned before, the static routing is suitable for the deterministic situation. Although in the MED policy the AWT is about 15% smaller than the static policy but the OFV is about 16% greater. In contrast, the static policy is not appropriate for stochastic models. Because the servers are planned to be fully utilized the system becomes unstable and therefore modifications are needed if intended to be used in such systems. In the modified static (MS) policy, the assignment is scheduled with 90% of capacity of each server to prevent instability. However, in this paper we focus on the dynamic routing policies. A Wilcoxon signed rank test for ten samples of each group shows that the TP in average provides jobs with higher QoS in comparison with the MED policy. For example in the M/M*/n model which arrival times of jobs and the s times, mean time to failure (MTTF) and mean time to repair (MTTR) of servers are exponentially distributed, the hypothesis $\mu_{TP} = \mu_{MED}$ is rejected with P-value 0.005 against $\mu_{TP} < \mu_{MED}$. The TP is superior to MED policy in QoS level and it is superior to static routing policies in AWT. Table 2 also implies that if the average waiting time is decreased, some quality priorities would be ignored.

Table 3. Variation of thresholds with different mean of arrival time

	t_2=125		t_2=150		t_2=175		t_2=200		t_2=225		t_2=250		t_2=275	
	j=1	j=2	j=1	j=2	j=1	j=2	j=1	j=2	j=1	j=2	j=1	j=2	j=1	j=2
i=1	3	2	3	2	3	2	3	2	3	3	4	4	3	3
i=2	3	2	3	2	3	2	3	2	3	3	4	4	3	3
i=3	2	2	2	2	2	2	2	2	2	3	2	3	2	2
i=4	1	1	1	1	1	1	1	1	1	1	1	1	1	1

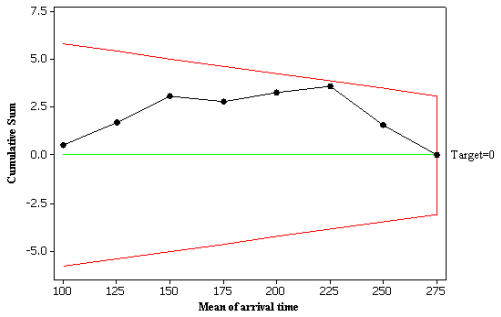

Fig. 1. Cumulative-sum control chart with two-sided V-mask (h:4.0, k:0.5) for OFV.

Table 3 shows the threshold variation in response to changes in its mean arrival time of a level of jobs in a 4×3 M/M/n system. The corresponding cumulative-sum control chart with two-sided V-mask for OFV of this system after 10000 hours simulation is illustrated in figure 1. V-masks are used to warn about abnormal

conditions. The sensitivity analysis implies that algorithm 3.3 justifies thresholds so that the QoS is kept under the control.

5 Conclusions

A threshold policy was presented for dynamic routing of jobs with service ranking among servers with different QoS. In contrast with traditional dynamic routing policies which have not considered the QoS, the proposed policy assigns jobs to servers with QoS considerations. The algorithm, which is developed on the basis of deterministic situation, tunes-up the thresholds in order to increase the QoS. The tune-up procedure is simple to implement and it reaches certain thresholds for each set of input parameters. Numerical experiments show that the adjusted TP keeps the QoS in a high level while maintains the stability of system. A sample problem was solved by the proposed algorithm to compare it with other policies. The TP balances the trade-off between QoS and AWT and therefore it is more effective than the MED policy which only minimizes the AWT and any static routing which keeps the QoS in a certain level. The proposed method is useful to control the QoS in production, service and communication systems. Distinguished advantages of the threshold policy are simplicity and flexibility. The decision maker can change the thresholds manually to change the utilization of servers or to root a level of jobs in a specific way. Mathematical analysis of the system is a problem yet to be resolved.

References

1. Down, D. G., Lewis, M. E.: Dynamic load balancing in parallel queueing systems: Stability and optimal control, European Journal of Operational Research, 168 509--519 (2006)
2. Winston W.: Optimality of the shortest line discipline. J Appl Prob, 14 181--9 (1977)
3. Nelson R. D., Philips T. K.: An approximation for the mean response time for shortest queue routing with general interarrival and service times. Perform Eval, 17 123--39 (1993)
4. Nelson R. D., Philips T. K.: An approximation to the response time for shortest queue routing. Perform Eval, 17 181--9 (1989)
5. Lui J. C. S., Muntz R. R., Towsley D.: Bounding the mean response time of a minimum expected delay routing system: an algorithmic approach. IEEE Trans Comput, 1371--82 (1995)
6. Lin W., Kumar, P.R.: Optimal control of a queueing system with two heterogeneous servers, IEEE Trans. Automat. Control, 29 696--703 (1984)
7. Koole, G.: A simple proof of the optimality of a threshold policy in a two-server queueing system, Systems & Control Letters, 26 301--303 (1995)
8. The, Y. C.: Critical Thresholds for Dynamic Routing in Queueing Networks, Queueing Systems, 42 297--316 (2002)
9. Feng, W., Adachi, K., Kowada, M.: A two-queue and two-server model with a threshold-based control service policy, European Journal of Operational Research, 137 593--611 (2002)
10. Sun, W., Guo, P., Tian, N.: Equilibrium threshold strategies in observable queueing systems with setup/closedown times, Springer-Verlag, (2009), DOI: 10.1007/s10100-009-0104-4
11. Altman, E., Nain, P.: Optimality of a Threshold Policy in the M/M/1 Queue with Repeated Vacations, Mathematical Methods Operations Research, 44 75--96 (1996)
12. Thomas, M. U., Wilson, G. R.: Applications of queuing theory, Industrial engineering handbook, Fifth edition, McGRAW-HILL, Vol2 11.67--11.99 (2001)
13. Kendall, N. K.: Stochastic processes occurring in the theory of queues and their analysis by the method of imbedded Markov Chains, Annals of Mathematical Statistics, 24 360--369 (1953)