Performance-aware Load Balancing for Multiclusters*

Ligang He, Stephen A. Jarvis, David Bacigalupo, Daniel P. Spooner
and Graham R. Nudd

Department of Computer Science, University of Warwick
Coventry, CV4 7AL, United Kingdom
liganghe@dcs.warwick.ac.uk

Abstract. In a multicluster architecture, where jobs can be submitted through each constituent cluster, the job arrival rates in individual clusters may be uneven and the load therefore needs to be balanced among clusters. In this paper we investigate load balancing for two types of jobs, namely *non-QoS* and *QoS-demanding* jobs and as a result, two performance-specific load balancing strategies (called ORT and OMR) are developed. The ORT strategy is used to obtain the optimised mean response time for non-QoS jobs and the OMR strategy is used to achieve the optimised mean miss rate for QoS-demanding jobs. The ORT and OMR strategies are mathematically modelled combining queuing network theory to establish sets of optimisation equations. Numerical solutions are developed to solve these optimisation equations, and a so called *fair workload level* is determined for each cluster. When the current workload in a cluster reaches this pre-calculated fair workload level, the jobs subsequently submitted to the cluster are transferred to other clusters for execution. The effectiveness of both strategies is demonstrated through theoretical analysis and experimental verification. The results show that the proposed load balancing mechanisms bring about considerable performance gains for both job types, while the job transfer frequency among clusters is considerably reduced. This has a number of advantages, in particular in the case where scheduling jobs to remote resources involves the transfer of large executable and data files.

1. Introduction

As grid technologies gain in popularity, separate clusters are increasingly being interconnected to create multicluster computing architectures for the processing of scientific and commercial applications [1][2][5]. These constituent clusters may be located within a single organization or across different geographical sites [3][6]. Load balancing across such architectures is recognised as a key research issue. If users located at different administrative domains submit jobs through domain-specific portals, there may be different submission patterns. Without intervention, this may lead to an unbalanced workload distribution among different domains; and the overall performance may be compromised.

* This work is sponsored in part by grants from the NASA AMES Research Center (administrated by USARDSG, contract no. N68171-01-C-9012), the EPSRC (contract no. GR/R47424/01) and the EPSRC e-Science Core Programme (contract no. GR/S03058/01).

Performance requirements are likely to vary depending on the job type. When the jobs have associated QoS demands (which we call *QoS-demanding jobs* or QDJs), performance is usually used to measure the extent of QoS-demand compliance; when jobs have no associated QoS demands (which we call *non-QoS jobs* or NQJs), a common performance criteria is to reduce the *mean response time* [9][12]. An example of the QoS is job *slack* [10][14]. The QoS of a job is satisfied if the job' waiting time (in the system) is less than its slack [15]; otherwise, the QoS is failed. Mean *miss rate* captures the aggregate slack failure and is therefore used as a performance metric to measure the proportion of jobs whose QoS demands fail.

In this paper, load balancing techniques are addressed for both *QoS-demanding jobs* (QDJs) and *non-QoS demanding jobs* (NQJs) to improve the job type-specific performance requirements in a multicluster. Two multicluster load balancing strategies, *Optimised mean Response Time* (ORT) and *Optimised mean Miss Rate* (OMR), are developed. The aim of ORT is to achieve optimised mean response time for NQJ workloads and the aim of OMR is to achieve the optimised mean miss rate for QDJ workloads. The ORT and OMR strategies are mathematically modelled combining queuing network theory to establish sets of optimisation equations. Numerical solutions are developed to solve the optimisation equation sets and determine a *fair workload level* for each cluster. When the current workload in a cluster reaches the precalculated fair workload level, the jobs subsequently submitted to the cluster are transferred to remote less-loaded clusters for execution.

There are a number of established workload allocation techniques in parallel and distributed systems [12][9]. A static workload allocation strategy is investigated in [12] to achieve the optimised mean response time; this strategy is specifically limited to a single cluster environment. Workload allocation techniques for multiclusters are addressed in [9] where it is assumed that the multicluster has a central entry point for the receipt of submitted jobs. In this paper, we assume that jobs can be submitted through each local cluster, and therefore the further problems brought about by the uneven submission patterns of jobs at the clusters also needs to be considered.

The problem of uneven job arrival patterns in different resources is addressed in a number of load balancing techniques [8][13]. A load balancing mechanism is presented in [8] for multi-domain environments. The mechanism identifies the least loaded computer among all domains and when a job is submitted to the system it is scheduled on that computer, whichever domain the job is actually submitted to. Hence, a job has to be transferred to a remote domain if the local domain does not contain the current least loaded computer.

In this paper however, a *fair workload level* is calculated for each cluster. Only when the current workload in a cluster exceeds its specified fair workload, does the cluster transfer the newly submitted job to a remote cluster. Although the load balancing technique presented in this paper may not achieve the best possible response time for a specific job, theoretical analysis and experimental studies show that considerable performance gains are still achieved in terms of the jobs' mean response time. Moreover, the job transfer frequency among clusters is dramatically reduced. This is desirable when jobs require the transfer of large executable and data files.

The workload allocation and load balancing techniques referenced above are applied to non-QoS jobs (NQJs). In this paper, techniques for allocating QoS-demanding jobs (QDJs) in multiclusters are also presented. The technique is similar to

that for NQJs, except that a fair workload level for each cluster is otherwise calculated to obtain the optimised mean miss rate for the QDJs.

The rest of the paper is organized as follows. The system and workload model is discussed in Section 2. Section 3 presents the load balancing techniques for NQJs and QDJs in multiclusters. The performance of these techniques is evaluated in Section 4. Section 5 concludes the paper.

2. System and workload model

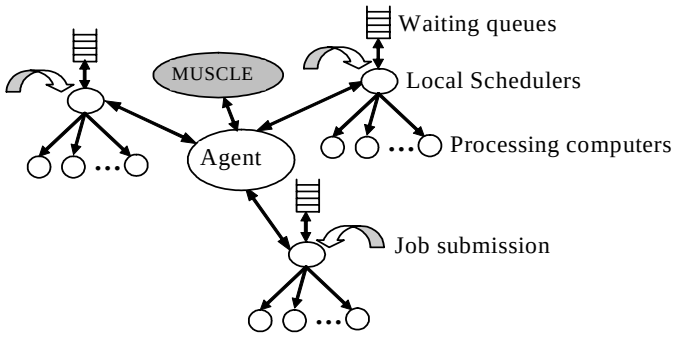

Fig. 1. The multicluster architecture

The multicluster architecture assumed in this paper is shown in Fig.1. The system consists of n different clusters, where each cluster comprises a set of homogeneous computers. Cluster i ($1 \le i \le n$) is modelled using an M/M/m_i queue, where m_i is the number of computers in cluster i. Jobs can be submitted through each local cluster, where they are queued for scheduling. The clusters in the multicluster are interconnected through an agent system [7], which is able to monitor the job submission in each cluster and determine the mean arrival rate of the jobs submitted to the entire multicluster. The mean arrival rate is utilized by the multicluster workload manager (which we call MUSCLE) to calculate the fair workload level for each cluster through the ORT and OMR strategies (for NQJs and QDJs, respectively).

The fair workload level for each cluster is measured by the mean number of jobs in its waiting queue. The local scheduler in each cluster is informed of its fair workload level. When the current number of jobs in the waiting queue in a cluster reaches its specified fair workload level, subsequent jobs are then transferred by MUSCLE (and the supporting agent system) to other suitable clusters. Each local scheduler processes locally submitted and remotely transferred jobs based on a First-Come-First-Served basis. The scheduling itself is non-preemptive. The jobs investigated in this paper are independent and each QDJ has a slack which follows a uniform distribution in $[s_l, s_u]$.

3. Load Balancing Techniques

Suppose that the mean arrival rate of the jobs submitted to the multicluster is λ. The overall performance of the job execution depends on the workload distribution among the clusters. The fair workload level for each cluster, measured by the mean number of jobs in its waiting queue, is determined in this section. The approaches for NQJs and QDJs differ as they have different performance requirements.

3.1 ORT (Optimised mean Response Time) strategies for NQJs

The ORT strategy aims to optimise the mean response time of the NQJs in the multicluster. The response time of a job is defined as the time from when the job arrives at the system until it is completed. The following analysis first establishes the optimisation equations for the mean response time, then a numerical solution to the optimisation equations is developed and the fair workload level is determined.

The response time of a job is its waiting time in the queue plus its execution time. Hence, the average response time of the jobs in cluster i, denoted as R_i, can be computed using Eq.1, where W_i is the mean waiting time of the jobs in cluster i and u_i is the service rate of the computers in cluster i.

$$R_i = W_i + \frac{1}{u_i} \tag{1}$$

Cluster i, containing m_i computers, is modelled using an M/M/m_i queue ($1 \le i \le n$). According to queueing theory [11], the mean waiting time of jobs, W_i, is computed using Eq.2, where ρ_i is the utilization of cluster i and W_{0i} is the mean remaining execution time of the jobs in service when a new job arrives.

$$W_i = \frac{W_{0i}}{1-\rho_i} \tag{2}$$

The formula for W_{0i} is given in Eq.3 [4], where P_{mi} is the probability that the system has no less than m_i jobs.

$$W_{0i} = \frac{P_{mi}}{m_i u_i} \tag{3}$$

Suppose that in the total workload in the entire multicluster, the fraction of workload allocated to cluster i is α_i, then,

$$\rho_i = \frac{\alpha_i \lambda}{m_i u_i} \tag{4}$$

P_{mi} in Eq.3 is given in Eq.5 [4][11].

$$P_{mi} = \frac{(m_i\rho_i)^{m_i}}{(1-\rho_i)m_i!\left[\sum_{k=0}^{m_i-1}\frac{(m_i\rho_i)^k}{k!} + \frac{(m_i\rho_i)^k}{(1-\rho_i)m!}\right]} \tag{5}$$

Using Eqns.1-5, the formula for R_i, in terms of α_i, is derived and is shown in Eq.6.

$$R_i = \frac{m_i u_i (\frac{\alpha_i \lambda}{u_i})^{m_i}}{\left[m_i! \sum_{k=0}^{m_i-1} \frac{(\frac{\alpha_i \lambda}{u_i})^k}{k!} + \frac{(\frac{\alpha_i \lambda}{u_i})^{m_i}}{(1-\frac{\alpha_i \lambda}{m_i u_i})} \right] (m_i u_i - \alpha_i \lambda)^2} + \frac{1}{u_i} \quad (6)$$

Thus, the mean response time of the incoming jobs over these n clusters, denoted by R, can be computed - see Eq.7.

$$R = \sum_{i=1}^{n} R_i \alpha_i \quad (7)$$

Hence, in order to achieve the optimal mean response time of the job stream in the multicluster, the aim is to find a workload allocation $\{\alpha_1, \alpha_2 \ldots, \alpha_n\}$ that minimizes Eq.7 subject to $\sum_{i=1}^{n} \alpha_i = 1$ and $0 \le \alpha_i \le \frac{m_i u_i}{\lambda}$ (the constraint $\alpha_i \le \frac{m_i u_i}{\lambda}$ is used to ensure that cluster i does not become saturated). This is a constrained-minimum problem and according to the Lagrange multiplier theorem, solving this problem is equivalent to solving the following equation set.

$$\begin{cases} \sum_{i=1}^{n} \alpha_i = 1, \quad 0 \le \alpha_i \le \frac{m_i u_i}{\lambda} & (a) \\ \frac{\partial}{\partial \alpha_k} (\sum_{i=1}^{n} R_i \alpha_i) - v \frac{\partial}{\partial \alpha_k} (\sum_{i=1}^{n} \alpha_i - 1) = 0 \quad 1 \le k \le n & (b) \end{cases} \quad (8)$$

Since α_i is the only unknown variable in the expression of R_i, Eq.8 can be reduced to Eq.9 by solving the partial differential equations in Eq.8.b.

$$\begin{cases} \sum_{i=1}^{n} \alpha_i = 1, \quad 0 \le \alpha_i \le \frac{m_i u_i}{\lambda} & (a) \\ \frac{\partial}{\partial \alpha_k} (R_k \alpha_k) = v \quad 1 \le k \le n & (b) \end{cases} \quad (9)$$

It is impossible to find the general symbolic solution $\{\alpha_1, \alpha_2 \ldots, \alpha_n\}$ from Eq.9, however, we identify a property of Eq.9.b that enables us to develop a numerical solution. The property is shown in Theorem 1 (the proof is omitted). Based on this property, we develop a numerical solution to solve Eq.9 and therefore derive the optimised workload allocation $\{\alpha_1, \alpha_2 \ldots, \alpha_n\}$. The numerical solution is shown in Algorithm 1.

Theorem 1. $\frac{\partial}{\partial \alpha_k} (R_k \alpha_k)$ is a monotonically increasing function of α_k.

Algorithm 1 Computation of workload allocation among clusters for optimised mean response time

```
1.   Let lower and upper limits of the mean response
time be v_lower and v_upper;
2.   while (v_lower≤v_upper)
3.        v_mid=(v_lower+ v_upper)/2;
4.        for each cluster i (1≤i≤n) do
```

```
5.            if (v_mid< ∂/∂αi (Riαi)|αi=0 )
6.                αi=0;
7.          else if (v_mid> miui/λ )
8.              v_upper=v_mid;
9.            continue;
10.            while (α_lower≤α_upper)
11.                α_mid=(α_lower+α_upper)/2;
12.              v_cur= ∂/∂αi (Riαi)|αi=α_mid ;
13.              if (the difference between v_cur and
v_mid is less than v_valve)
14.                  αi=α_mid;
15.              if (v_cur is less than v_mid)
16.                  α_lower=α_mid;
17.              else
18.                  α_upper=a_mid;
19.      end for
20.      α_sum= Σ_{i=1}^{n} αi ;
21.      if (the difference between α_sum and 1 is less
than α_valve)
22.          the current set of αi (1≤i≤n) is the cor-
rect workload allocation;
23.      else if (α_sum is less than 1)
24.           v_lower= v_mid;
25.      else
26.           v_upper= v_mid;
27.      end while
```

Since a binary search technique is used to search for v and α_i in their respective search spaces [v_lower, v_upper] and [α_lower, α_upper], the time complexity of Algorithm 1 is $O(n \log k_v \log k_\alpha)$, where k_α and k_v are the number of elements in the search space of v and α_i, which equal $\frac{v\_upper - v\_lower}{\varphi}$ and $\frac{\alpha\_upper - \alpha\_lower}{\gamma}$, respectively ($\varphi$ and γ represent the precision in the calculation). Since φ and γ are predefined constants, the time complexity is linear with the number of clusters, that is n.

The feasibility and effectiveness of Algorithm 1 are shown in Theorem 2.

Theorem 2. The workload allocation strategy $\{\alpha_1, \alpha_2 \ldots, \alpha_n\}$ computed by Algorithm 1 minimizes the average response time of the incoming job stream in a multicluster system of n clusters.

After α_i ($1 \le i \le n$) is determined, the mean number of jobs in the waiting queue of cluster C_i, denoted by N_i, can be calculated using Eq.10 [11]; where W_i is the jobs' mean waiting time. W_i can be calculated using the first item to the right of Eq.6.

$$N_i = \lambda \alpha_i \times W_i \quad (10)$$

$\lceil N_i \rceil$ is regarded as the *fair workload level* for cluster *i*. When the current number of jobs in the waiting queue of cluster *i* is less than $\lceil N_i \rceil$, the arriving jobs are scheduled locally; otherwise, the local scheduler of cluster *i* transfers the arriving jobs to the supporting agent system where they are further dispatched to the cluster with the least load (defined by Eq.11) among those clusters whose current number of jobs in the waiting queue is less than its fair workload level (if such a cluster does not exist, the jobs are scheduled to the cluster with the least load among all clusters).

$$load_i = \frac{the\ \ job\ number\ in\ the\ waiting\ \ queue + m_i}{m_i u_i} \quad (11)$$

3.2 OMR (Optimised mean Miss Rate) strategy for QDJs

The QoS of a QDJ fails if its waiting time is greater than its slack. The performance criterion for evaluating the scheduling of QDJs differs from that for NQJs in that it typically aims to minimize the fraction of jobs that miss their QoS, termed the *miss rate*. In this subsection, a workload allocation strategy called OMR is developed, whose aim is to optimise the mean miss rate of the submitted QDJs in the multicluster. Every QDJ has some slack that follows a uniform distribution. Its probability density function $S(x)$ is given in Eq.12, where s_u and s_l are the upper and lower limits of the slack, respectively.

$$S(x) = \frac{1}{s_u - s_l} \quad (12)$$

We continue to model cluster *i* (of m_i computers) as an M/M/m_i queue ($1 \le i \le n$). As in queuing theory [11], in an M/M/m_i queue the probability distribution function of the job waiting time, $P_w(x)$ (which means that the probability that the job waiting time is less than x), is given by Eq.13 [11], where ρ_i and P_{mi} are the same variables as those found in Eq.2 and Eq.3.

$$P_w(x) = 1 - P_{mi} e^{-m_i u_i (1-\rho_i) x} \quad (13)$$

Using the probability density function of slack, the miss rate of the QDJs allocated to cluster *i*, denoted by MR_i, and can be computed by Eq.14.

$$MR_i = \int_{s_l}^{s_u} S(x)(1 - P_w(x))dx \quad (14)$$

$$MR_i = \frac{m_i u_i (\frac{\alpha_i \lambda}{u_i})^{m_i} [e^{-(m_i u_i - \alpha_i \lambda) s_l} - e^{-(m_i u_i - \alpha_i \lambda) s_u}]}{\left[m_i! \sum_{k=0}^{m_i - 1} \frac{(\frac{\alpha_i \lambda}{u_i})^k}{k!} + \frac{(\frac{\alpha_i \lambda}{u_i})^k}{(1 - \frac{\alpha_i \lambda}{m_i u_i})} \right] (m_i u_i - \alpha_i \lambda)^2 (s_u - s_l)} \quad (15)$$

Applying Eq.12 and Eq.13 and solving the integral, Eq.14 becomes Eq.15, where the workload fraction α_i for cluster i is the only unknown variable.

The mean miss rate (denoted by MR) of the QDJs over these n clusters can be computed using Eq.16.

$$MR = \sum_{i=1}^{n} MR_i \times \alpha_i \tag{16}$$

Similar to the case of minimizing the mean response time, this is a constrained-minimum problem. This requires identifying a workload allocation that minimizes MR in Eq.16 subject to $\sum_{i=1}^{n} \alpha_i = 1$ and $0 \le \alpha_i \le \frac{m_i u_i}{\lambda}$. This is equivalent to solving the following equation set.

$$\begin{cases} \sum_{i=1}^{n} \alpha_i = 1, \quad 0 \le \alpha_i \le \dfrac{m_i u_i}{\lambda} & (a) \\ \dfrac{\partial}{\partial \alpha_k}(MR_k \times \alpha_k) = v \quad 1 \le k \le n & (b) \end{cases} \tag{17}$$

In the previous subsection, we state that the numerical solution to Eq.9 is based on the property that $\frac{\partial}{\partial \alpha_k}(R_k \alpha_k)$ is a monotonically increasing function of α_k. Theorem 3 is introduced to establish the case that $\frac{\partial}{\partial \alpha_k}(MR_k \times \alpha_k)$ in Eq.17 also monotonically increases over α_k. The proof of the theorem is omitted for brevity. With this property, a numerical solution is also developed to solve Eq.17. The solving algorithm is similar to that found in Algorithm 1 and the proof of the algorithms effectiveness is similar to Theorem 2. Hence, they are omitted in the paper.

Theorem 3 $\frac{\partial}{\partial \alpha_k}(MR_k \times \alpha_k)$ is a monotonically increasing function of α_k.

As in the case of NQJs, the mean number of jobs in the waiting queue of cluster i (i.e. N_i) can be obtained using Eq.10. The *fair workload level* for cluster i for QDJs can be subsequently determined. If the current number of jobs in the waiting queue of cluster i is greater than $\lceil N_i \rceil$, then the arriving jobs are transferred to the cluster with the least miss rate among those clusters whose number of jobs in the waiting queue is less than its fair workload level (if such clusters do not exist, the jobs are scheduled to the cluster with the least miss rate among all clusters).

4. Experimental Evaluation

An experimental simulator is developed to evaluate the performance of the proposed workload allocation techniques under a wide range of system settings and workload levels. Two types of job stream (NQJs and QDJs) are generated using the same parameters, with one exception, in that every QDJ has an additional slack metric which follows a uniform distribution. Each job stream includes 500,000 independent jobs. The job arrival follows a Poisson process and a job is submitted to the multicluster through a randomly selected cluster. The run of the first 100,000 jobs is considered as

the initiation period, allowing the system to achieve a steady state, and the run of the last 100,000 jobs is considered the ending period. Statistical data are collected from the middle 300,000 jobs. The job size follows an exponential distribution. The mean size of the incoming jobs is set to be the inverse of the average of the speeds of all processing computers multiplied by the average of the number of computers in each cluster, that is,

$$\frac{n^2}{\sum_{i=1}^{n} u_i \sum_{i=1}^{n} m_i} (\text{sec})$$

Based on the mean job size, the job arrival rate at which the system becomes saturated, can be computed. The incoming workload levels in the experiments are measured using the percentage of the saturated arrival rate.

An intuitive load balancing strategy, the *weighted* strategy [12][9], takes into account the heterogeneity of the clusters' performance. In this strategy, the workload fraction α_i allocated to cluster i ($1 \le i \le n$) is proportional to its processing capability, $m_i u_i$. Hence, α_i is computed as Eq.18.

$$\alpha_i = \frac{m_i u_i}{\sum_{i=1}^{n} m_i u_i} \quad (\mathbf{18})$$

Consequently, under the weighted strategy the corresponding fair workload level for each cluster can be determined using Eq.10.

The *Multi-domain Load Balancing mechanism* (MLB) of [8], which is based on a dynamic least load algorithm, can be used as the ideal bound of the mean response time obtained by the ORT strategy. Using this approach the arriving jobs are scheduled on the computer with the least load. Hence a job is transferred to a remote domain while the local domain does not contain the least loaded computer. In the load balancing mechanism presented in this paper, the job transfer frequency among clusters can be dramatically reduced so as to improve the scheduling cost. Similarly, a dynamic least miss-rate (DLM) strategy is used as the upper bound of the mean miss rate of QDJs in the multicluster. The DLM strategy schedules newly arriving QDJs to the cluster with the least miss rate.

These five load balancing strategies (ORT, OMR, Weighted, MLB and DLM) are evaluated in these experiments. The performance metrics used in the experiments include the *mean response time* (for NQJs) and the *mean miss rate* (for QDJs). Each point in the performance curves are plotted as the average result of 5 independent runs of the job streams with different initialisation random numbers.

4.1 Workload

Table 1. System setting in Figure 2

	Cluster 1	Cluster 2	Cluster 3	Cluster 4
m_i	3	5	7	9
u_i	20	16	12	8

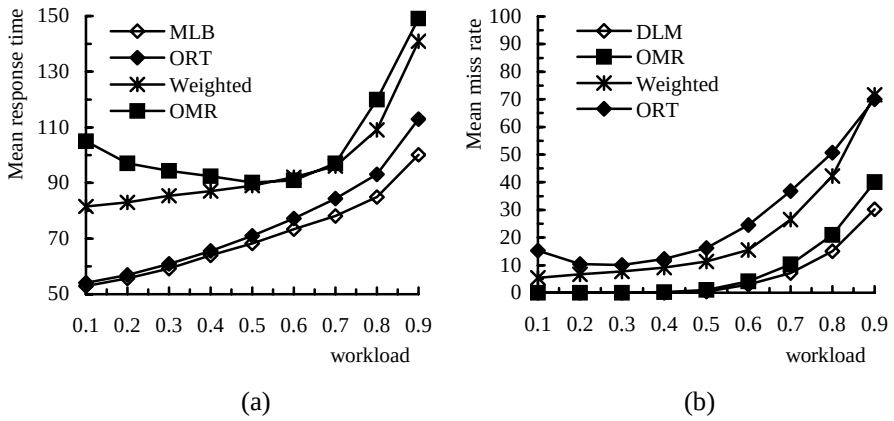

Fig. 2. Impact of the incoming workload levels on a) mean response time and b) mean miss rate

Fig.2.a and Fig.2.b show the impact of the incoming workload levels on the mean response time and the mean miss rate of the incoming jobs under these load balancing strategies. The multicluster in this experiment consists of 4 clusters whose configurations are listed in Table 1. For QDJs, the job slacks follow a uniform distribution in the range [0, 30]. In order to gain insight into the difference of the load balancing behaviours between OMR and ORT, the ORT strategy is also used to balance QDJs, and OMR is used to balance NQJs.

It can be observed from Fig.2.a that the ORT strategy performs significantly better than the weighted strategy in terms of the mean response time. Furthermore, the performance difference increases as the workload decreases. This trend can be explained as follows. The weighted strategy allocates the same fraction of workload to a cluster even if the workload varies. However, the waiting time accounts for a lower proportion of the response time as the workload decreases. Hence, in order to reduce the response time, a higher proportion of the incoming workload should be allocated to the cluster with the greater u_i (the number of computers m_i in each cluster has less impact). The ORT strategy is able to satisfy this allocation requirement. Fig.2.b shows the impact of the incoming workload on the mean miss rate. It can be observed that the OMR strategy outperforms the weighted strategy at all incoming workload levels.

In Fig.2.a, although the MLB outperforms ORT, the performance difference is small, especially when the workload is low. A similar pattern can be observed between the DLM and OMR strategies. This suggests that applying the ORT and OMR schemes will achieve competitive performance with relatively low cost, especially when the system load is low.

It can be observed from Fig.2.a and Fig.2.b that in most incoming workload levels, the OMR strategy obtains the worst performance in terms of mean response time while the ORT strategy obtains the worst performance in terms of mean miss rate. These results suggest that the performance-specific load balancing strategies are necessary to achieve good respective performance.

4.2 Computer speed

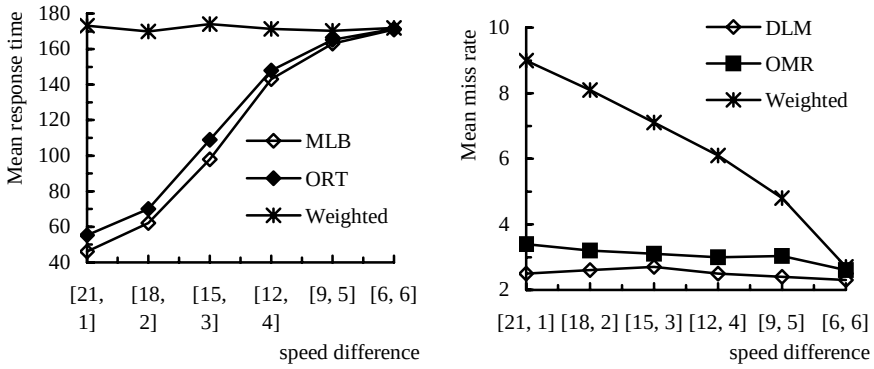

Fig. 3. The impact of speed difference on a) the mean response time and b) the mean miss rate; the arrival rate is 50% of the saturated arrival rate

Fig.3 demonstrates the impact of the difference of computer speed. Here the multicluster consists of 4 clusters and the number of computers in each cluster is set to be 4. The speed of the computers in cluster 1 varies from 21 to 6 with a decrement of 3, while the speed of all computers in the other three clusters increases from 1 to 6 with an increment of 1. Thus, the multicluster ranges from a highly heterogeneous system to a homogeneous system, while the average speed of all computers remains constant (i.e., 6). The slack of the QDJs follows a uniform distribution in [0, 10].

Fig.3.a shows the impact of the difference of computer speed on the mean response time. It can be observed in this figure that the mean response time decreases significantly under the ORT strategy as the speed difference increases, while it remains approximately the same under the weighted strategy. This is because as the speed difference increases, despite the average computer speed remaining constant, a higher proportion of the workload is allocated to cluster 1 under the ORT strategy (higher than $m_1u_1/\sum_{i=1}^{n} m_iu_i$), while the weighted strategy does not make full use of the computing power of cluster 1. This suggests that under the ORT strategy, the speed difference among the clusters is a critical factor for the mean response time.

The first observation from Fig.3.b is that the OMR strategy performs better than the weighted strategy under all speed combinations, as is to be expected. A further observation is that under OMR, the mean miss rate remains approximately the same as the speed difference varies. The experimental results for other incoming workload levels show similar patterns. This suggests that under OMR, the speed difference among the clusters is not an important parameter for the mean miss rate. This differs from the characteristic of the ORT for mean response time. This divergence may originate from the difference between the expressions of the response time and the miss rate (see Eq.6 and Eq.15): we note the occurrence of $1/u_i$ in Eq.6 while this is absent in Eq.15.

5. Conclusion

Two load balancing strategies (ORT and OMR) for multicluster architectures are proposed that deal with different types of jobs. The ORT strategy can optimize the mean response time of NQJs, while the OMR strategy can optimize the mean miss rate of QDJs. The effectiveness of these proposed load balancing strategies is demonstrated through theoretical analysis. The proposed strategies are also evaluated through extensive experimental studies. The results show that the ORT and OMR strategies can achieve considerable performance gain with relatively low overhead.

References

1. O. Aumage.: Heterogeneous multi-cluster networking with the Madeleine III. International Parallel and Distributed Processing Symposium (IPDPS 2002), 2002.
2. S. Banen, A.I.D. Bucur, and D.H.J. Epema.: A Measurement-Based Simulation Study of Processor Co-Allocation in Multicluster Systems. Ninth Workshop on Job Scheduling Strategies for Parallel Processing, D.G. Feitelson, L. Rudolph and U. Schwiegelshohn (eds), 2003.
3. M. Barreto, R. Avila, and P. Navaux.: The MultiCluster model to the integrated use of multiple workstation clusters. Proc. of the 3rd Workshop on Personal Computer-based Networks of Workstations, 2000, pp. 71–80.
4. G. Bolch.: Performance Modeling of Computer Systems. 2002.
5. A.I.D. Bucur and D.H.J. Epema.: The maximal utilization of processor co-allocation in multicluster Systems. Int'l Parallel and Distributed Processing Symp. (IPDPS 2003), 2003, pp. 60-69.
6. R. Buyya and M. Baker.: Emerging Technologies for Multicluster/Grid Computing. Proceedings of the 2001 IEEE International Conference on Cluster Computing, 2001.
7. J. Cao, D. J. Kerbyson, and G. R. Nudd.: Performance Evaluation of an Agent-Based Resource Management Infrastructure for Grid Computing. Proceedings of 1st IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid'01), 2001.
8. S.T. Chanson, W. Deng, C. Hui, X. Tang, M. To.: Multidomain Load Balancing. 2000 International Conf. on Network Protocols, Japan, 2000.
9. L. He, S.A. Jarvis, D.P. Spooner, G.R. Nudd.: Optimising static workload allocation in multiclusters. Proceedings of 18th IEEE International Parallel and Distributed Processing Symposium (IPDPS'04), 2004.
10. B. Kao and H. Garcia-Molina.: Scheduling soft real-time jobs over dual non-real-time servers. IEEE Trans. on Parallel and Distributed Systems, 7(1): 56-68, 1996.
11. L. Kleinrock.: Queueing system, John Wiley & Sons, 1975.
12. X.Y. Tang, S.T. Chanson.: Optimizing static job scheduling in a network of heterogeneous computers. 29th International Conference on Parallel Processing, 2000.
13. M. Wu.: On Runtime Parallel Scheduling for Processor Load Balancing. IEEE Transaction on Parallel and Distributed Systems, 8(2): pp.173-186, Feb. 1997.
14. W. Zhu.: Scheduling soft real-time tasks on cluster. In proc. of 1999 Annual Australian Parallel and Real-Time Conference, 1999.
15. W. Zhu and B. Fleisch.: Performance evaluation of soft real-time scheduling on a multicomputer cluster. The 20th International Conference on Distributed Computing Systems (ICDCS 2000), 2000.