Journal of University of Science and Technology Beijing
Volume 9, Number 5, October 2002, Page 389

Information

A multiserver multiqueue network: modeling and performance analysis

Zhiguang Shan[1], Chuang Lin[1], and Yang Yang[2]

1) Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
2) Information Engineering School, University of Science and Technology Beijing, Beijing 100083, China
(Received 2002-01-11)

Abstract: A new category of system model, multiserver multiqueue network (MSMQN), is proposed for distributed systems such as the geographically distributed Web-server clusters. A MSMQN comprises multiple multiserver multiqueue (MSMQ) nodes distributed over the network, and every node consists of a number of servers that each contains multiple priority queues for waiting customers. An incoming request can be distributed to a waiting queue of any server in any node, according to the routing policy integrated by the node-selection policy at network-level, request-dispatching policy at node-level, and request-scheduling policy at server-level. The model is investigated using stochastic high-level Petri net (SHLPN) modeling and performance analysis techniques. The performance metrics concerned includes the delay time of requests in the MSMQ node and the response time perceived by the users. The numerical example shows the efficiency of the performance analysis technique.

Key words: multiserver multiqueue network (MSMQN); multiserver multiqueue (MSMQ) system; stochastic high-level Petri net (SHLPN); modeling; performance analysis; Web-server clusters

[*This work is financially supported by the National Natural Science Foundation of China (No.90104002 and 60173012), the Projects of Development Plan of the State Key Fundamental Research (No.G1999032707) and the Projects of Development Plan of the State High Technology Research (No. 2001AA112080).*]

As the scale of computer networks and distributed systems grows larger and larger, and the structures become more and more complicated, it is increasingly difficult yet crucial to implement resource management and task scheduling in those systems. In general, various systems exhibit different behaviors, but they may have similar structural models and employ similar strategies for resource management and task scheduling.

In this paper, a new category of system structural model, multiserver multiqueue network (MSMQN), is proposed for distributed systems based on the previous work on multiserver multiqueue (MSMQ) system [1,2]. A MSMQN is composed of multiple MSMQ nodes distributed over the network. Each node comprises multiple server entities that each consists of several priority queues for waiting customers. A typical example of the MSMQN model is the geographically distributed Web-server clusters [3]: each mirrored Web-server cluster [4,5] can be regarded as a MSMQ node, and multiple Web-server clusters connected by the network form a MSMQN. In this system, an incoming HTTP request can be assigned to a waiting queue of any server in any cluster node, according to the routing policy integrated by the node-selection policy at network-level, request-dispatching policy at node-level, and request-scheduling policy at server-level. A response is returned to the end user after service. In general, a Web server has a single processor that selects the requests from different priority queues for service according to the request-scheduling policy.

The modeling and performance analysis technique is also proposed to analyze the performance of a MSMQN based on the Stochastic High-Level Petri Net (SHLPN) [6]. The methods of model abstraction and refinement are used to simplify the system model and reduce the complexity of the model soluton. SHLPN is chosen because it is a powerful graphical and mathematical tool for handling prioritized, concurrent, asynchronous, stochastic and nondeterministic events. In addition, there exist a set of well-established performance analysis techniques for SHLPN. SHLPN has been successfully applied to the performance analysis of some ATM control mechanisms [7].

1 SHLPN modeling of MSMQN

SHLPNs [6] are high-level Petri nets (HLPN) [8] augmented with exponentially distributed transition firing times. A marking of the SHLPN model is a distribution of tokens in each place of the model. A transi-

Corresponding author:Zhiguang Shan **E-mail**:shanzhiguang@ustb.edu.cn

tion represents a class of possible changes of markings and may be associated with an enabling predicate expressed in terms of the token variable or place marking expressions. A transition is enabled whenever the predicates associated with it are satisfied. The state space of a model consists of the set of all markings reachable from the initial marking through the occurrence of transition firing.

The following general assumptions for the MSMQN are made:

(1) Incoming task requests can be classified into n categories and assigned n priority levels accordingly. Priority i is higher than priority k, for $1 \le i < k \le n$. Requests with priority i are denoted by r_i;

(2) The MSMQN is composed of k MSMQ nodes, and each node comprises m clustered servers (in fact, different node can have different number of servers). Each server contains n waiting queues of requests, and one for each priority level. Requests in the same waiting queue are scheduled in a first-come-first-served (FCFS) manner. All the n waiting queues contest the single processor of the server for service. A threshold is set for each queue to restrict the maximum number of waiting requests;

(3) The client requests are assumed to be generated according to the Poisson distribution with mean arrival rate λ_i for requests r_i. An incoming request can be distributed to any of the k nodes and further dispatched to any of the m servers of a node. A server can not accept any more requests of a priority level if the corresponding waiting queue has reached its capacity;

(4) The service times of requests in different waiting queues are assumed to be exponentially distributed with different means.

In **figure 1**, a SHLPN model of the MSMQN is proposed. In the model, the rectangles denote timed transitions and the black bars denote immediate transitions; circles denote places, and they contain tokens that denoted by black dots. In this model, requests' arriving and servicing are represented by timed transitions, which are associated with exponentially distributed firing delay. Requests entering the waiting queues or competing for mutual exclusive server resources are represented by immediate transitions, which fire in zero time and can be associated with firing probabilities. In general, immediate transitions have higher firing priorities than timed transitions. Tokens in this SHLPN model represent either service requests or processors, and different classes of requests can be represented by tokens of different colors. In the model, the first subscript of the symbols represents the category of requests, while the second subscript represents the destination server. The superscript of the symbols represen

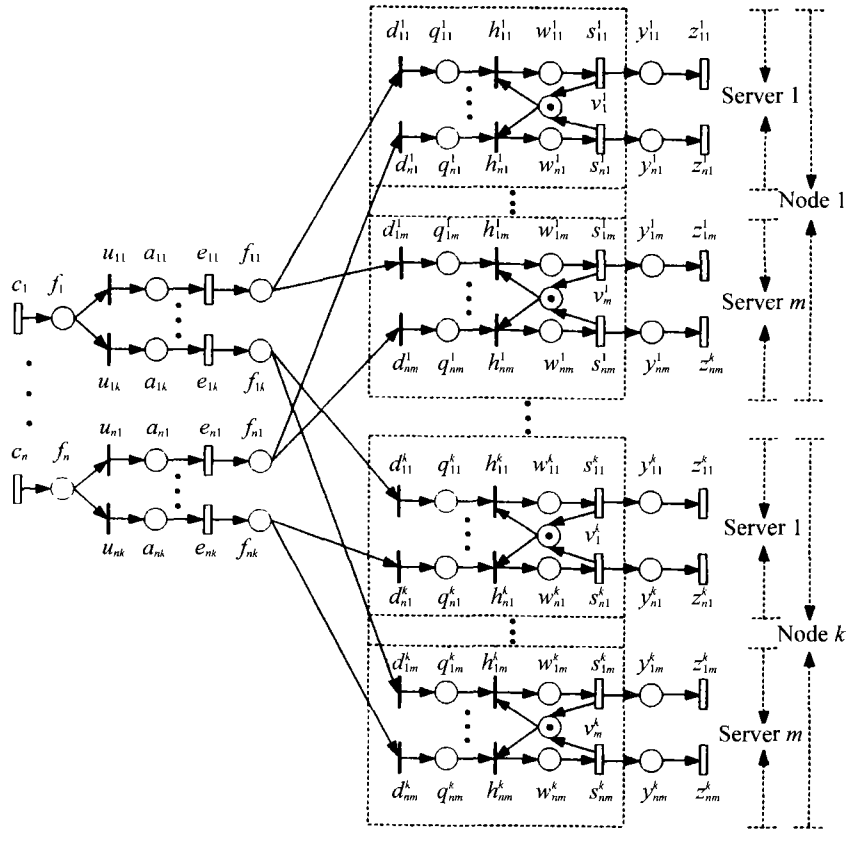

Figure 1 A SHLPN model of the MSMQN.

ts the corresponding MSMQ node. If there is only one subscript, it either indicates the category of requests or differentiates the servers of a node.

The meanings of the transitions and places in figure 1 are described as follows, and for all the symbols below, $1 \le i \le n$, $1 \le j \le m$, $1 \le x \le k$.

Places: f_i is the place that holds requests r_i instantaneously and decides the destination node according to the enabling predicates associated with u_{ix}. a_{ix} is the place that models the transmission link from the client i to the node x. f_{ix} is the place that models the request dispatcher of the node x, and it holds the incoming requestsr_i instantaneously and decides the destination server according to the enabling predicates associated with d_{ij}^x. q_{ij}^x is the place that models the waiting queue with priority i in the server j of the node x, and the capacity of q_{ij}^x is b_{ij}^x that represents the thresholds of the waiting queue. As well as the aggregate of all the waiting queues in the server j of the node x is represented by q_j^x. w_{ij}^x is the place that models the running state of a request of priority i at the server j of the node x, and its capacity is '1' since only one service request at a time can be in this state due to one processor. v_j^x is the place that models the processor resource of the server j in the node x, and the token in v_j^x represents the single processor of the server j. y_{ij}^x is the place that models the transmission link from the server j in the node x to the client i.

Transitions: c_i is the timed transition modeling the arrival of requests r_i, and its mean firing rate is at λ_i. u_{ix} is the immediate transition that models distributing requests r_i to the node x according to the node-selection algorithm. e_{ix} is the timed transition that models the actions of probing the delay information from client i to the node x for selecting the best cluster node. d_{ij}^x is the immediate transition that models dispatching requests r_i to the server j of the node x. h_{ij}^x is the immediate transition that models scheduling requests r_i to be served by the server j of the node x. s_{ij}^x is the timed transition that models serving requests r_i by the server j of the node x, and its mean service rate is at μ_{ij}^x. z_{ij}^x is the timed transition that models the transfer of response documents *via* the link from the server j of the node x to the client i.

In figure 1, the n waiting queues in a server are arranged in the order of priority, *i.e.*, the requests in q_{1j}^x have the highest priority for service and the requests in q_{nj}^x the lowest.

2 Model refinement

A SHLPN is homomorphic to a continuous time Markov Chain (MC), and there is one-to-one relationship between marking in the SHLPN and states in the MC. In general, to analyze the performance of the SHLPN model in figure 1, the corresponding MC of the model can be constructed. Based on the MC and state transition rates, the state transition matrix can be constructed and all the steady-state probabilities can be obtained to compute the performance metrics. Software package SPNP (Stochastic Petri Net Package) [9] adopted in this paper for performance analysis is developed based on this idea. However, the MC of the model in figure 1 is generally with $m \times n \times k$ dimensions. The number of states of the MC grows exponentially with the increase of b_{ij}^x, m, n and k. The state space is so large that solving the state equations is impossible for the practical computing systems.

In order to handle the problem of state-space explosion, abstraction and refinement can be used before analysis to simplify the system model and reduce the complexity of the model solution. Model refinement can develop compact models and also reveal the independence and interdependence relations of the submodels in the original model. The method of refinement is to delete the immediate transitions and transfer the enabling predicates. In this way, the complicated structure and relations among the original SHLPN model is simplified, and the state space of the model is reduced.

Figure 2 shows the refined SHLPN model of the MSMQN. The model in figure 1 has been decomposed

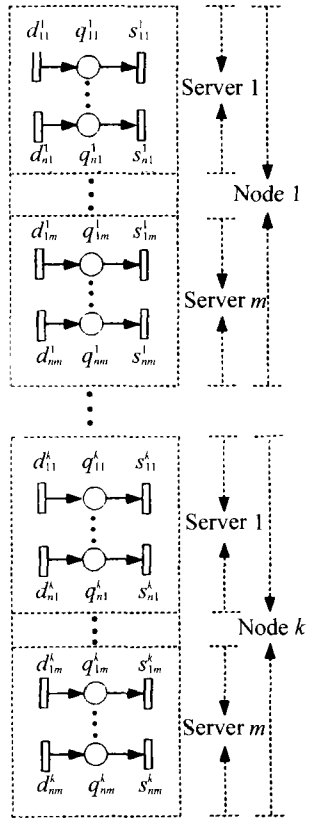

Figure 2 The refined SHLPN model of the MSMQN.

into multiple submodels, and each submodel is an independent part in structure containing one waiting queue. The timed transitions of delay probing and response transferring together with the relevant places and arcs from the model in figure 1 have been removed, for these delay can be considered directly in the programming of SPNP without description in the model. The structure of the SHLPN model in figure 1 has been simplified by using transition enabling predicates and rate functions. The enabling predicates and firing rates of transition d_{ij}^x and s_{ij}^x in figure 2 have been modified. Whether the transition d_{ij}^x and s_{ij}^x in figure 2 are enabled or not, it depends not only on the marking of its waiting queue q_{ij}^x, but also the markings of other queues in other servers of the same node or even other nodes. In this way, a complicated model is equivalently transformed into a compact model.

3 Policy description and performance evaluation

In this section, the routing strategies involved in the MSMQN model is described, which is integrated by the node-selection policy, request-dispatching policy, and request-scheduling policy. The performance evaluation method is also specified.

3.1 Policy description

(1) Request-scheduling policy.

Request-scheduling policies are implemented at the server to select requests from different waiting queues for service. The Weighted Priority Scheduling policy is considered in this paper.

Weighted Priority Scheduling (WPS). This policy schedules requests based on their weighted importance, *i. e.*, requests with higher priority have higher firing probability to be scheduled. The priority value is used to represent the weight of importance. The goal of this policy is to provide differentiated quality of service (QoS) to different classes of service requests. The policy can be described by the enabling predicate and firing probability of transition s_{ij}^x in the SHLPN model of figure 1.

The enabling predicate of transition s_{ij}^x is $M(q_{ij}^x)>0$, where $M(q_{ij}^x)$ denotes the marking of q_{ij}^x that represents the number of waiting requests in q_{ij}^x.

Assume the priority value of r_i is α_i. The firing probability of transition s_{ij}^x is:

$$P_s(M(q_{ij}^x))=\begin{cases}1 & \text{if } i\in \mathrm{WPS}_1(M);\\ \beta_i & \text{if } i\in \mathrm{WPS}_2(M);\\ 0 & \text{otherwise.}\end{cases} \tag{1}$$

Where $\mathrm{WPS}_1(M) = \{t|M(q_{ij}^x)>0$ and $M(q_{yj}^x) = 0, \exists t, 1\le t\le n$, for $\forall y, 1\le y\ne t\le n\}$, $\mathrm{WPS}_2(M) = \{t|M(q_{ij}^x)>0$ and $M(q_{yj}^x)> 0$, for $\forall t, 1\le t\le n; \exists y, 1\le y\ne t\le n\}$, and β_i is:

$$\beta_t=\alpha_t/(\alpha_t+\sum_y \alpha_y) \tag{2}$$

(2) Request-dispatching policy.

Request-dispatching policies are used by the request dispatcher in a cluster node to distribute an incoming request to the best available server. The Minimum Expected Delay policy is concerned in this paper.

Minimum Expected Delay (MED). For a given MSMQ node, this policy selects the server in which an incoming request will experience the minimum expected delay as the destination. More specifically, the expected delay of a request if assigned to a server is measured by two elements: one is the number of pending requests in the waiting queues, the other is the mean service time of the server for each class of requests. The server with the minimum product of those two items is selected as the best server. This policy is based on the server conditions and requires a feedback mechanism to dynamically report the status of server load and capacity.

Define the function $\mathrm{ED}(q_j^x)$ below as the measurement of the expected delay for an incoming request in the server j of the node x:

$$\mathrm{ED}(q_j^x)=\sum_{i=1}^{n}\frac{M(q_{ij}^x)}{\mu_{ij}^x} \tag{3}$$

Where $M(q_{ij}^x)$ represents the number of waiting requests in q_{ij}^x, and μ_{ij}^x is the mean service rate of requests r_i by the server j in the node x. It is assumed that the dispatcher can dynamically obtain these information to implement load balancing.

The MED policy can be described by the enabling predicate and firing probability of transition d_{ij}^x in figure 1.

The enabling predicate associated with d_{ij}^x is:

$(M(q_{ij}^x)<b_{ij}^x)\wedge((\mathrm{ED}(q_j^x)\le \mathrm{ED}(q_t^x), \forall t, 1\le t\le m \text{ and } t\ne j)\vee(M(q_{it}^x)=b_{it}^x))$.

The firing probability of d_{ij}^x is:

$$P_d(M(q_{ij}^x))=\begin{cases}\dfrac{1}{\|\mathrm{MED}(M)\|} & \text{if } j\in\mathrm{MED}(M)\\ 0 & \text{otherwise}\end{cases} \tag{4}$$

Where $\mathrm{MED}(M) = \{t|\mathrm{ED}(q_t^x) = \min(\mathrm{ED}(q_1^x), \mathrm{ED}(q_2^x), \cdots, \mathrm{ED}(q_m^x))$ and $M(q_{it}^x)<b_{it}^x\}$.

(3) Node-selection policy.

Node-selection policies are used by the clients to se-

lect the best MSMQ node for a given request. The Minimum Predicted Response Time policy is considered in this paper.

Minimum Predicted Response Time (MPRT). Based on the request-scheduling policy WPS and the request-dispatching policy MED, this algorithm selects the cluster node with the minimum predicted response time experienced by the users as the best destination for request distribution. The clients are assumed to be able to dynamically obtain the following information *via* probing approach for computing the predicted response time (PRT): the current available network bandwidth, the transmission latency (RTT), and the expected delay at the server. Thus, the function of PRT can be given by:

$$\mathrm{PRT}=\mathrm{PTT}+\frac{\text{Document size}}{B_{\text{avail}}}+\mathrm{ED} \tag{5}$$

Where B_{avail} is the available bandwidth measured by probing, RTT is the round trip time in transmission, and ED is the expected delay at the server described in (2). The predicted response time of a given request is calculated for all candidate cluster nodes and the node with the minimum value is chosen.

The MPRT policy can be described by the enabling predicate and firing probability of transition u_{ix} in figure 1.

The enabling predicate associated with u_{ix} is:

$$\left(\sum_{j=1}^{m}M(q_{ij}^{x})<\sum_{j=1}^{m}b_{ij}^{x}\right)\wedge((\forall y,\ 1\le y\le k \text{ and } x\ne y,\ (\mathrm{PRT}_{ix})\le(\mathrm{PRT}_{iy}))\vee\left(\sum_{j=1}^{m}M(q_{ij}^{y})=\sum_{j=1}^{m}b_{ij}^{y}\right) \tag{6}$$

Where PRT_{ix} is the predicted response time of distributing request r_i to the node x, and it can be specified by:

$$\mathrm{PRT}_{ix}=\mathrm{RTT}_{ix}+\frac{\text{Document size}}{B_{\text{avail\_}ix}}+\sum_{x=1}^{n}\sum_{j=1}^{m}\frac{M(q_{ij}^{x})}{\mu_{ij}^{x}}.$$

Where RTT_{ix} and $B_{\text{avail\_}ix}$ are respectively the round trip time and available bandwidth from client i to the node x, and it is assumed the same bandwidth in both directions, *i.e.* $B_{\text{avail\_}ix}=B_{\text{avail\_}xi}$.

The firing probability of u_{ix} is:

$$P_{u}(x)=\begin{cases}\dfrac{1}{\|\mathrm{MED}(M)\|} & \text{if } x\in\mathrm{MED}(M)\\ 0 & \text{otherwise}\end{cases} \tag{7}$$

Where $\mathrm{MED}(M)=\{y \mid \mathrm{PRT}_{iy}=\min(\mathrm{PRT}_{i1},\mathrm{PRT}_{i2},\cdots,\mathrm{PRT}_{ik}))$ and $\sum_{j=1}^{m}M(q_{ij}^{x})<\sum_{j=1}^{m}b_{ij}^{x}\}$.

Note that the enabling predicate associated with d_{ij}^{x} in figure 2 is the intersection of the enabling predicates associated with d_{ij}^{x} and u_{ix} in figure 1, and the firing probability of s_{ij}^{x} in figure 2 is the same as that of s_{ij}^{x} in figure 1.

3.2 Performance evaluation

In SHLPN models, the performance measurements can be given based on the steady-state probabilities. The steady-state probabilities of the marking M is denoted by $P[M]$. The metrics adopted in the performance evaluation includes the delay time of requests spent at the MSMQ node and the response time experienced by the end users.

In SHLPN models, the mean value $D(q)$ of the markings in a waiting queue q with capacity b in steady state is:

$$D(q)=\sum_{y=1}^{b}y\times P[M(q)=y] \tag{8}$$

The delay time DT_{ij}^{x} in the submodel A_{ij}^{x} (formed by transition d_{ij}^{x}, s_{ij}^{x} and place q_{ij}^{x} in figure 2) is:

$$\mathrm{DT}_{ij}^{x}=D(q_{ij}^{x})/T(s_{ij}^{x}) \tag{8}$$

Where $T(s_{ij}^{x})$ is the throughput, *i.e.* the number of requests that have been served, of submodel A_{ij}^{x}, and it can be obtained from the SPNP programs.

The delay time DT_i of requests r_i for all the MSMQ nodes is:

$$\mathrm{DT}_{i}=\frac{\sum_{j=1}^{m}\sum_{x=1}^{k}D(q_{ij}^{x})}{\sum_{j=1}^{m}\sum_{x=1}^{k}T(s_{ij}^{x})} \tag{10}$$

The delay time DT for all classes of requests in all the cluster nodes is:

$$\mathrm{DT}=\frac{\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{x=1}^{k}D(q_{ij}^{x})}{\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{x=1}^{k}T(s_{ij}^{x})} \tag{11}$$

The response time RT_i of requests r_i perceived by the users is:

$$\mathrm{RT}_{i}=\sum_{x=1}^{k}\mathrm{Pr}_{ix}(\mathrm{RTT}_{ix}+1\%\cdot\mathrm{Tt}_{xi})+\mathrm{DT}_{i}+\sum_{x=1}^{k}Pr_{ix}\cdot\mathrm{Tt}_{xi} \tag{12}$$

There are three items on the right of equation (12). The first item represents the time that request r_i spent before it is assigned to a node, including round trip latencies and probing time. Pr_{ix} is the probability of request r_i being distributed to the node x, and it can be expressed based on throughput as follows:

$$Pr_{ix}=\frac{\sum_{j=1}^{m}T(s_{ij}^{x})}{\sum_{x=1}^{k}\sum_{j=1}^{m}T(s_{ij}^{x})} \tag{13}$$

Tt_{xi} is the transfer time of response documents from the node x to the client i, and it is given by:

$$\mathrm{Tt}_{xi}=\frac{\text{Document size}}{B_{\text{avail\_}xi}} \tag{14}$$

$1\%Tt_{xi}$ represents the time cost in probing the network and system states, for the overhead of lightweight probes can be restricted to one percent of the time needed in transferring the response documents [10]. The second item is the delay time of request r_i spent at the node as described in equation (3). The third item is the time needed in transferring the response documents of request r_i from the node x back to client i.

The response time RT of all classes of requests perceived by the users is:

$$RT=\frac{\sum_{i=1}^{n} RT_i \cdot T_i}{\sum_{i=1}^{n} T_i} \tag{15}$$

Where T_i is the throughput of request r_i for all the nodes, and it is described as follows:

$$T_i=\sum_{j=1}^{m}\sum_{x=1}^{k} T(s_{ij}^x) \tag{16}$$

4 Numerical results

The software package SPNP [9] is used to analyze the SHLPN model of MSMQN in figure 2 under the WPS, MED and MPRT policies. To avoid the problem of state-space explosion and simplify the model solution, without loss of generality, a MSMQN is only considered to consist of two MSMQ nodes that each comprises two servers. Requests are divided into two categories. Class 1 requests are attached with a high priority and class 2 requests are attached with a low priority. The priority values of the two classes of requests are $\alpha_1=4$ and $\alpha_2=1$. The available bandwidth and RTT from the client i to the node x are defined in **table 1**. The size of response document of request r_1 is 500 kbit and that of request r_2 is 1 Mbit. For the two servers in each node, the thresholds of the waiting queues are $b_{11}^1=b_{12}^1=b_{11}^2=b_{12}^2=6$ and $b_{21}^1=b_{22}^1=b_{21}^2=b_{22}^2=2$. It is assumed that the servers in each cluster node are homogeneous but heterogeneous in different clusters. A server can serve different classes of requests at different service rates. The mean service rates of transitions s_{ij}^x are at $\mu_{11}^1=\mu_{12}^1=60.0$ times/s, $\mu_{21}^1=\mu_{22}^1=48.0$ times/s, and $\mu_{11}^2=\mu_{12}^2=40.0$ times/s, $\mu_{21}^2=\mu_{22}^2=32.0$ times/s. The request arrival rates are $\lambda_1=\lambda_2$, and they monotonically increase from 20.0 times/s to 160.0 times/s ranging from underload to overload.

Figure 3 shows the numerical results of delay time in the MSMQ nodes for request r_1, r_2 and all categories of requests (*i.e.* DT_1, DT_2 and DT respectively). It shows that DT_1 is always less than DT_2 with the increase of the request rate, and the decrement is by 17.87% at least and by 44.61% at most according to the numerical results. This shows that at the server the high-priority requests can experience less delay thus better QoS performance than the low-priority requests.

Figure 4 shows the numerical results of the response time of request r_1, r_2 and all categories of requests (*i.e.* RT_1, RT_2 and RT respectively) perceived by the users. In figure 4, the response time RT_1 of the high-priority request r_1 is always much less than RT_2 of the low-priority requests. This shows that high-priority requests can experience much less response time thus achieve better QoS performance than the low-priority requests.

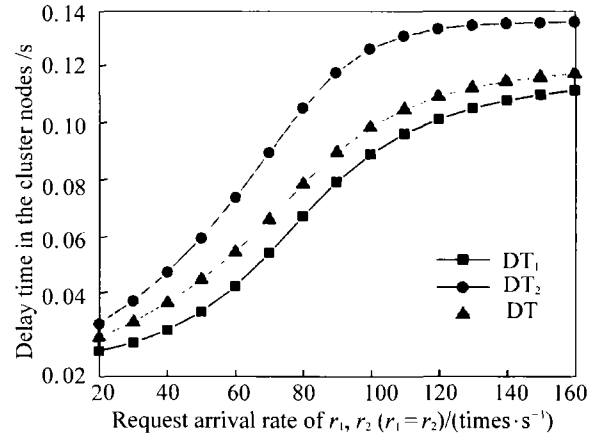

Figure 3 Delay time in the cluster nodes.

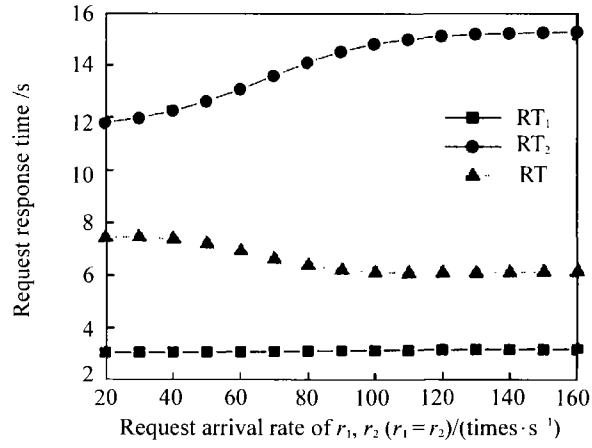

Figure 4 Request response time.

Table 1 Network parameters

Sending Client-receiving node	Bandwidth type	Available bandwidth/Mbps	RTT/ms
Client 1-node 1	Large	1.35	50
Client 1-node 2	Medium-large	0.9	130
Client 2-node 1	Medium-narrow	0.7	190
Client 2-node 2	Narrow	0.4	280

5 Conclusions

In this paper, a new category of system model, MSMQN, has been proposed for distributed systems such as the geographically distributed Web-server clusters. The routing policy for an incoming request to be distributed to a waiting queue of any server in any node in a MSMQN is integrated by the node-selection policy MPRT at network-level, request-dispatching policy MED at node-level, and request-scheduling policy WPS at server-level. The policy is investigated using SHLPN modeling and performance analysis techniques. The numerical example shows the efficiency of the performance analysis technique. Moreover, the MSMQN models, the integrated scheme of request routing and the performance analysis technique can be applied to performance evaluation of this class of various complex systems.

References

[1] Z. Shan, C. Lin, and Y. Yang, *et al.*, Performance modeling and approximate analysis of multiserver multiqueue systems with poisson and self-similar arrvials [J], *Journal of University of Science and Technology Beijing*, 8(2001), No.2, p. 145.

[2] C. Lin and S. Yang, Decomposition and approximate performance analysis for the multiserver multiqueue system model [J], *Journal of Software* (in Chinese), Supplement (1999), p.6.

[3] V. Cardellini, M. Colajanni and P. S. Yu, Geographic load balancing for scalable distributed Web systems, [in] *Proceedings of the Eighth International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS2000)* [C], IEEE Computer Society, San Francisco, 2000, p.20.

[4] Z. Shan, Q. Dai, and C. Lin, *et al.*, Integrated schemes of Web request dispatching and selecting and their performance analysis [J], *Journal of Software* (in Chinese), 12 (2001), No. 3, p.355.

[5] C. Lin, Z. Shan, and Y. Yang, Integrated schemes of request dispatching and selecting in Web server clusters, [in] *Proceedings of International Conference on Software. Theory and Practice. 16th World Computer Congress (WCC2000)* [C], Beijing: Publishing House of Electronics Industry, 2000, p.922.

[6] C. Lin and D.C. Marinescu, Stochastic high-level Petri nets and applications [J], *IEEE Transactions on Computers*, 37 (1988), No. 7, p. 815.

[7] C. Lin and S. T. Chanson, ATM admission models of stochastic high level Petri nets based on hierarchical modeling, [in] *Proceedings of 1995 International Conference on Network Protocols* [C], Tokyo, 1995, p.144.

[8] T. Murata, Petri nets: properties, analysis and applications [J], *Proceedings of the IEEE*, 77(1989), No.4, p. 541.

[9] G. Ciaodo, J. Muppala and K.S. Trivedi, SPNP: stochastic petri net package, [in] *Proceeding of the Petri Nets and Performance Models* [C], kyoto, 1989, p.142.

[10] R. L. Carter and M. E. Crovella, Server selection using dynamic path characterization in wide-area networks, [in] *Proceedings of IEEE INFOCOM1997* [C], Kobe, 1997, p.1014.