# Configuration of Mail Delivery System Based on Reinforcement Learning

Soichiro Morishita[†], Kazuyuki Hiraoka[††], Hidetsune Kobayashi[‡], and Taketoshi Mishima[††]
[†]Graduate School of Science and Engineering, Saitama University,
255 Shimo-Okubo, Saitama, 338-8570, JAPAN
Tel. +81-48-858-3723, Fax.: +81-48-858-3723, E-mail: mori@me.ics.saitama-u.ac.jp
[††]Dept. of Information and Computer Sciences, Saitama University,
255 Shimo-Okubo, Urawa, 338-8570, JAPAN
[‡]Dept. of Mathematics, Collage of Science and Technology, Nihon University,
1–8 Kanda Surugadai, Chiyoda-ku, 101–0062, JAPAN

**Abstract**: To solve the internal security problem such as human error and bad faith, the automation of computer system management is significant. For this purpose, we focus attention in the automation of Mail Delivery Service. Today, requirement for reliable mail delivery system becomes larger and larger. However, existing systems are too strict about their configuration. Hence, we propose the method based on Reinforcement Learning (RL) to achieve proper MX record ordering. A measure on validity of the design of system, such as network topology number of servers and so on, is also obtained as side benefit. In order to verify the usability of the presented method, we did on a small model of mail delivery system. As a result, we show that RL is available for determination of the proper MX record ordering. Additionally, we suggest that it is also available for comparing validity of setting of MTA and the network design.

## 1. Introduction

To solve the internal security problem such as human error and bad faith, the automation of computer system management is significant [1]. For this purpose, we focus attention in the automation of Mail Delivery Service.

Electronic mail (E-mail) have been spread widely. It is used for not only personal exchange but also deal on business nowadays. Requirement for reliable mail delivery system become larger and larger. However existing system is too strict about its configuration. Improper configuration easily causes non-delivery of mail. Otherwise, it causes several defects such as delay of mail delivery.

In this paper, we consider to achieve the best configuration of mail delivery system by Reinforcement Learning (RL) towards automation of the system.

The configuration of mail delivery system has following parameters: (a) Mail Exchange (MX) record ordering for domain in the Domain Name Service(DNS) database, (b) setting of Mail Transfer Agent (MTA), and (c) the network design.

MX record describes priority of hosts to which mails for a specified domain are relayed. The ordering of MX record entry should be decided according to formation of servers and network design, though MX record can not force behavior of the other host. Furthermore, there is no established measure to compare validity of setting of MTA and the network design. In such a circumstance, it is too difficult to decide the best configuration of mail delivery system.

We propose a method based on RL to acquire proper MX record ordering and acquire a measure to compare validity of configurations.

## 2. Mail Delivery Service

There are several hosts which provide mail delivery service. MTA is running on each host. There is only one DNS server in our model, therefore all hosts refer to the same DNS database.

When a mail is posted to a MTA, the MTA decides its behavior depending on the domain of destination address of the mail. The behavior is one of the following.

- Accept: If the MTA identified the domain as its own one, it should accept the mail.
- Relay: If the MTA identified the domain as the one which is allowed to relay, it relays the mail to next host. It decides the host according to MX record in the DNS database. We mention the details later.
- Reject: Otherwise, it should reject the mail. The rejected mail should be resent after a while or be

returned to sender.

Hosts can be down by system failure with a certain probability. So commonly the another host which relays mails is builded. The priority of host to which it should relay is described on MX record of DNS database, and distributed to hosts around the world.

However, description of MX record do not force behavior of MTA. It depends on setting of MTA.

## 3. Reinforcement Learning Scheme

In this section, we describe the RL scheme to achieve proper MX record ordering by $Q$-Learning[2].

There are $n$ hosts in the system. State $s_i$ of the system means that a mail is queued to the spool of the host $i$, and action $a_j$ means relaying it to host $j$ (Figure 1). The MTA of host which have received a mail behaves almost like the RL agent. At every time step $t = 1, 2, 3, \cdots$, the agent decides the action $a(t)$ according to the current policy $\pi$, Policy $\pi$ corresponds to MX record for the MTA. It is independent from $s(t)$ because there is only one DNS server in our model. This is a feasible approximation of the real-worlds systems.

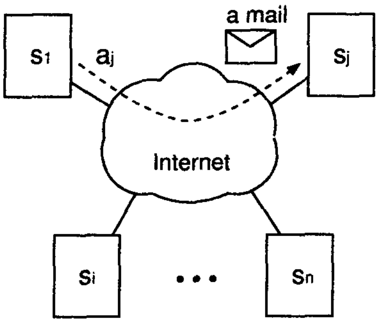

Figure 1. *Reinforcement Learning Scheme on the simplest model of mail delivery system. State $s_i$ means that a mail is queued to the host i, and action $a_j$ relay to host j.*

We define return $V(t)$ at the time step $t$ as follows:

$$V(t) = \sum_{k=0} \gamma^k r(t+k),$$

where $\gamma$ is discounted factor, and $r(t)$ is reward at the time $t$. $Q^*(a)$ denotes the expected value of $V(t)$. Though $Q^*$ cannot be observed directly its estimation $Q(a(t))$ can be incrementally computed as

$$Q(a(t)) \leftarrow (1-\alpha)Q(a(t)) + \alpha \left[r(t) + \gamma \max_a Q(a)\right],$$

where $\alpha$ is learning factor, and $\max_a Q(a)$ is maximum $Q$-value of all actions $a_j (j = 1, 2, \cdots, n)$. As well as policy $\pi$, $Q$-value is independent from $s(t)$. In this case, $Q(a)$ means weighted average of $Q(s, a)$

Finally, when a mail have reached to the destination host, the agent gets reward. The reward is computed as follows:

$$r(t) = \frac{\hat{T}(t) - T + 1}{\hat{T}(t)},$$

where $T$ demotes the time taken for delivery, and $\hat{T}(t)$ denotes weighted average of $T$. Weighted average $\hat{T}$ is updated as follows:

$$\hat{T}(t+1) = \begin{cases} (1-\beta)\hat{T}(t) + \beta T & \text{(A mail has reached)} \\ \hat{T}(t) & \text{(A mail has not reached)} \end{cases},$$

where $\beta$ is forgetting coefficient of $\hat{T}$.

While learning, the agent select action with $\varepsilon$-greedy selection[3]. The agent selects action randomly with probability of $\varepsilon$, otherwise it selects the action which have largest $Q$-value.

After enough learning, $Q$-value converges to the optimal value $Q^*(a)$. Sorting actions $a_j$ by $Q$-value in descending order, we can achieve a proper ordering of MX record.

## 4. Experimental Results

In order to verify the usability of the presented method, we did some experiments on a small model of mail delivery system.

There are hosts A, B, and C. Host A accepts the mail, and host B and C permit to relay the mail.

When a mail relayed, the transferring time of the mail increase 1 count. If the mail was rejected, or the host which to connect is down, MTA waits 240 counts and tries a new action.

The following table shows setting of the experiments.

| | |
|---|---|
| $n$ (number of hosts) | 3 |
| $\alpha$ (learning factor) | 0.001 |
| $\gamma$ (discounted factor) | 0.1 |
| $\beta$ (forgetting coefficient of $\hat{T}$) | 0.01 |
| $\varepsilon$ (probability of random selection) | 0.5 |

### 4.1 Convergence of Q-value

After enough learning, $Q(a)$ converged to desired value. Figure 2 shows change of $Q$-value of each action. Host A accepts mail. Host B and C relayes mail.

$Q(a_A)$ is max. $Q(a_B)$ and $Q(a_C)$ are converged to the same value. This order of $Q$-value is desired one corresponding to the roles of host A, B, and C.

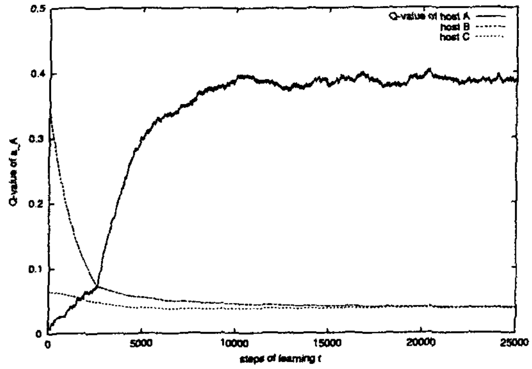

Figure 2. *Q-value of each action in one trial. Host A accepts mail. Host B and C relayes mail. $Q(a_A)$ is max. $Q(a_B)$ and $Q(a_C)$ are converged to the same value.*

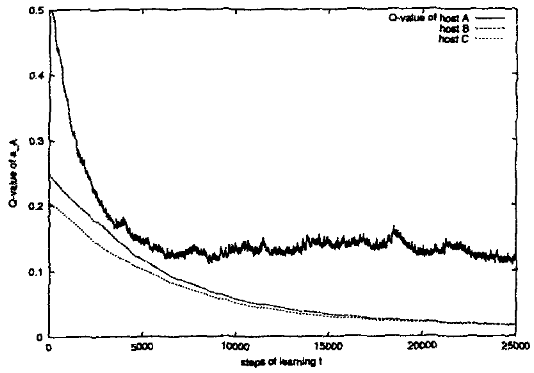

Figure 3. *Q-value of each action in one trial. Host A accepts mail. Host B relayes mail. Host C rejects mail. Though $Q(a_B)$ and $Q(a_C)$ should be differ each other, they converged to the same value.*

By the way, when the setting of host C was changed to *'reject the mail'*, it is not appeared that the difference between $Q(a_B)$ and $Q(a_C)$ 3. The reason for it is how to give rewards to agent. Because only the host which accept the mail gets the reward, it is not reflected in $Q$-values of other hosts. The expected effect would be derived to give reward to agent retrospectively when the mail arrived.

## 4.2 Q-value as a index of reliability

So far at least, we assumed there is no system failure on any hosts. To examine validity of proposed method, we did experiments on the situation that system failure occurs on each host at random. Then, $Q(a)$ converged to the value less than the value of the case which is assumed that no system failure occurs. This means that we can regard $Q$-value as measure of reliability of mail transfer system.

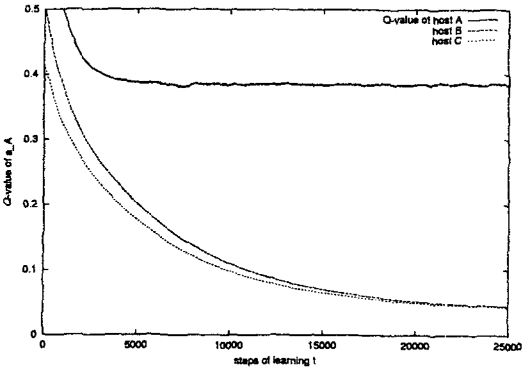

Figure 4. *Average in 20 trials of Q-value of each action.*

After this, we consider max value of $Q$-value as index of reliability of the system. To clarify tendency of $Q$-value, we use average of $Q$-value in 20 trials. Figure 4 shows average of $Q$-value of each action in 20 trials. With this operation, the transition of value become stable.

## 4.3 Simulation with system failure

As a measure of the system availability, we introduce MTTR and MTBF. MTTR is defined as the mean time to repair. MTBF is defined as the mean time before failure. System failure rate is found as follows:

$$[\text{System failure rate}] = \frac{[\text{MTTR}]}{[\text{MTTR}] + [\text{MTBF}]}$$

Figure 5 shows average of $Q(a_A)$ in 20 trials for each MTTR of host A. The larger MTTR become, the smaller $Q(a_A)$ become. On the other hand, Figure 6 shows average of $Q(a_A)$ in 20 trials for each MTTR of host B. Host B relays a mail, but do not accept. These are have similar tendency. However, the case that there are failure with host A(the host which accept the mail), $Q$-value is lower than the case with host B(the host which only relay the mail). This result is acceptable, so $Q$-value is valid as a index of reliability.

Figure 7 shows correlation between MTTR of host A and host B.

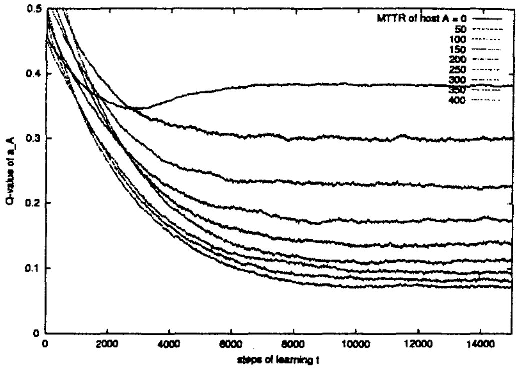

Figure 5. *average of $Q(a_A)$ in 20 trials for each MTTR of host A. ($MTBF = 1000$)*

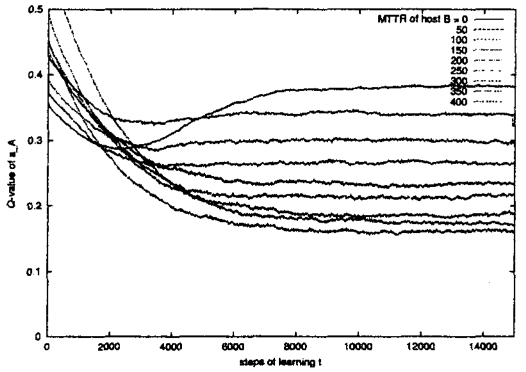

Figure 6. *average of $Q(a_A)$ in 20 trials for each MTTR of host B. ($MTBF = 1000$)*

### 4.4 Setting ratio of MTTR to MTBF constant

Figure 8 shows the learning curve when the ratio of MTTR to MTBF is a constant 0.5. It is estimated that there are some relation between $Q$-value and wait time of host when the action is failed, but any relations have not found in this experiments.

## 5. Conclusion

We pointed out difficulty of achieving the proper configuration of mail delivery system. Hence, we proposed the method based on RL to achieve proper MX record ordering and determining measure to compare validity of configuration.

In order to verify the usability of the presented method, we did some experiments applying the method with a small model of mail delivery system.

As a result, we showed that RL is available for determination of the proper MX record ordering. Additionally, we suggested that it is also available for comparing validity of setting of MTA and the network design.

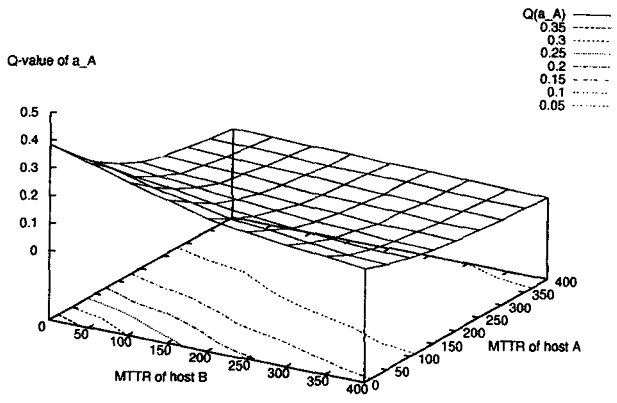

Figure 7. *Correlation between MTTR of host A and host B. ($MTBF = 1000$)*

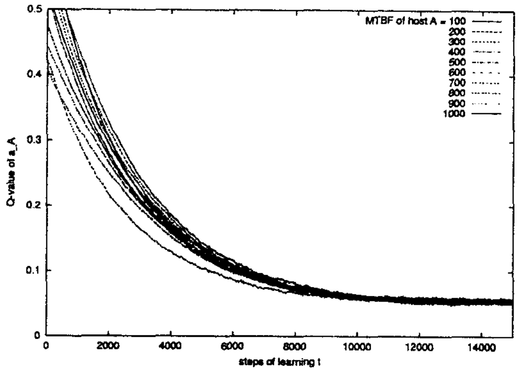

Figure 8. *the learning curve when the ratio of MTTR to MTBF is a constant* 0.5.

## References

[1] Zhanfei Zhou, Soichiro Morishita, Hiroshi Mikami, and Taketoshi Mishima, "Groupware administration system based on UNIX", TECHNICAL REPORT OF IEICE FACE2001-16(2001-11) pp.5-8, Tokyo, Japan, IEICE, Dec. 7, 2001.

[2] Watkins, C. J. C. H. and Dayan, P., Technical Note: Q-Learning, Machine Learning 8, pp. 279–292, 1992.

[3] Sutton, R. S. and Barto, A., Reinforcement Learning: An Introduction, A Bradford Book, The MIT Press, 1998.