

INFORMATION RETRIEVAL ON THE INTERNET: A REALISTIC QUEUEING MODEL FOR SOUJOURN TIME ANALYSIS

A. B. WOLPERT, J. R. KENEVAN

School of Computer Science and Telecommunications,
Roosevelt University, Chicago, IL
{jkenevan,awolpert}@roosevelt.edu

Abstract: In this paper we design a model of a complex process known as the WEB information retrieval. An open queuing network represents the Internet. The TCP channel delivery and multi-facility requirements to serve an information retrieval request are incorporated into the model. We further use this model to determine analytical solutions for mean values of sojourn times needed to respond to an information retrieval request.

Keywords: Modeling and Simulation of the Internet, Communication Networks, Client-Sever Systems, and Queuing Networks

1. INTRODUCTION

The main purpose of this paper is to design realistic model of a complex process known as WEB information retrieval. This model must allow evaluating the mean time that it takes the Internet to serve a request.

In this paper it is assumed that all requests are directed to one server. Due to limited processing capabilities, the server can run a predefined maximum number of threads (processes) simultaneously. Besides the limitation on the total number of threads is also limited in the number of threads of each service application that can run but on the server simultaneously. Each WEB request requires to run simultaneously a predefined number of threads of different service applications (i.e., WEB

database server, http server, mail server, address translation server, etc.) Therefore, if a request arrives that requires more threads of certain type that are currently available then the service is denied and the request rejected. Consequently we consider the system to be a multi-facility propagation network with blocking service.

Here we assume that all requests from the same source require the same facilities

Apparently, a reasonable model of the service process should incorporate the model of underlying TCP sessions with a subsequent evaluation of a TCP sojourn time and a given probability of connection establishment. The system under consideration in this paper is given on Fig. 1.1, i.e., a number of sources s_1, \dots, s_n generate requests for an access to a Threaded Server TS. Each source s_i generate requests according to a Poisson process with the rate λ_i . Each generated request must travel through the network cloud to reach a Server TS where it will be served. A request may be split into a number of different Transport Protocol Data Units (TPDU's) that could travel separately over different routes. The number of routes available for a source s_i depends only on a source. TPDU's are recombined into a request before delivering it to the TS server. Therefore, recombined TPDU's that form the request must be delivered in the order they are transmitted. Apparently each route has its own propagation time that is assumed to be exponentially distributed with the mean service time of k^{th} route being $T_k = 1/\mu_k$. A TPDU is directed into a k^{th} route with a probability p_k . In this paper we assume that every source request is delivered to the destination with a probability p_d that is the same for all sources and routes. Maximum of number of request retransmissions N_{max} is allowed. After maximum number of retries is reached TCP connection is dropped.

The server TS is a multi-facility server that can run simultaneously c_1 threads of application A_1 , c_2 threads of application A_2 ... c_L threads of application A_L . Denote a thread capacity vector of the server TS by $\mathbf{c} = \{c_1, c_2, \dots, c_L\}$. Each request from a source s_i requires certain number of threads of applications from the set $\{1, \dots, L\}$. This demand can be represented by a vector $\mathbf{d}_i = \{d_{i1}, \dots, d_{iL}\}$ where d_{ij} is the number of servers of type j (threads of application A_j) required to serve the request from the source s_i . Obviously $\mathbf{D} = [d_{ij}]_{1 \leq i \leq n, 1 \leq j \leq L}$ is a demand matrix of the mix.

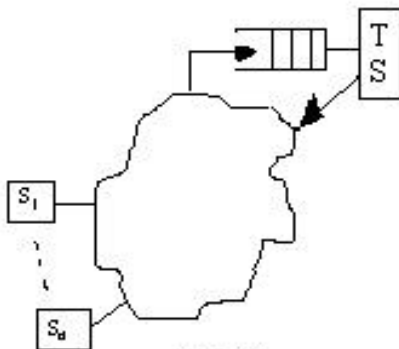


Fig 1.1

When a request from the source s_i finally arrives at the multi-facility server TS it generates d_{i1}, \dots, d_{iL} to servers of types A_1, \dots, A_L (threads of corresponding applications) if the corresponding number of servers is available. If the required number of servers (threads) is not available for at least one the

FCFS. A single server (route) R_i has a exponential service time distribution with the mean $1/\mu_i$. Upon the arrival from route R_i a call from the source s_j joins the buffer RB_j to await for a service completion of all calls that entered the system before it. With a probability p_d a call is delivered to the plain

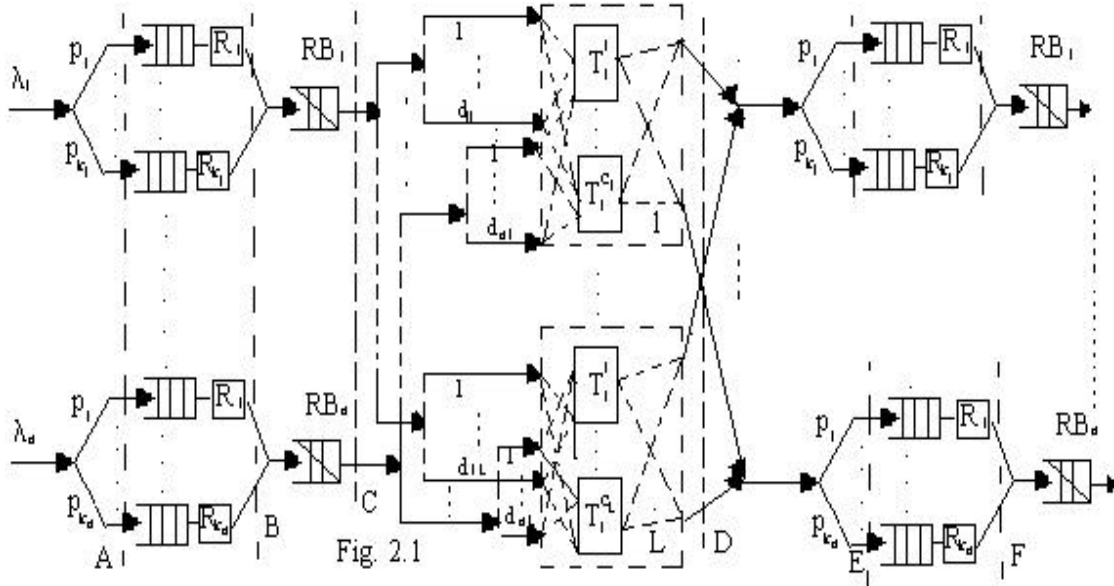


Fig. 2.1

types A_1, \dots, A_L , the service is denied and the request is rejected. The situation is not unlike allocation of channels in circuit switched systems. To extend the analogy, if a service is denied to a request from a source s_i , a source can try to repeat the request with a given probability $p_p(i)$ (a.k.a. persistence probability of a source) or it can abandon the request.

2. The Model

Each pair (source, the network cloud) is represented by set of parallel queues with infinite capacity buffer each attended by a separate server (route in the cloud). Each queue represents possible route through the cloud. For simplicity purposes we assume that splitting of a request into calls (TPDU's) is instantaneous, therefore, assuming that a request from each source is represented by a set of calls from this source. Each call from a source s_i can be directed into one of k_i possible routes. Upon arrival calls from the same source are re-sequenced in accordance with the time order in which they were generated. Again, for simplicity purposes we assume that recombination of a request from calls is instantaneous. Hence, the requests from the same source arrive in a server TS in the order they were transmitted. Requests arrive from a source in the according to a Poisson process with the rate λ_i , and are directed into a r^{th} route with a probability p_r , $0 < r < k_i$, $\sum_{r=1}^{k_i} p_r = 1$. Equivalently, a request from j^{th} source join $(j,r)^{\text{th}}$ queue with probability p_r . The service discipline for each queue is

FCFS. A single server (route) R_i has a exponential service time distribution with the mean $1/\mu_i$. Upon the arrival from route R_i a call from the source s_j joins the buffer RB_j to await for a service completion of all calls that entered the system before it. With a probability p_d a call is delivered to the plain

FCFS. A single server (route) R_i has a exponential service time distribution with the mean $1/\mu_i$. Upon the arrival from route R_i a call from the source s_j joins the buffer RB_j to await for a service completion of all calls that entered the system before it. With a probability p_d a call is delivered to the plain

source they are re-sequenced, instantaneously recombined and delivered to a source.

3. Mean reaction time: retrieved information delivery.

Since thinning of a Poisson process by a Bernoulli process results in a set of independent Poisson processes (see [1]), the set of queues between planes A and B fig 2.1 behave as a set of independent M/M/1 systems. Therefore, if we denote $T_i^j(t)$ the distribution of sojourn time for a request from the source s_j directed into i^{th} route (queue) at the time when it reaches plane B it is simply (see for example [2])

$$(3.1) \quad T_i^j(t) = 1 - e^{-(\mathbf{m}_i - p_i \mathbf{I}_j)t}$$

As it was shown in [3] at the time when a call from a source s_j reaches the plain C the distribution of sojourn time becomes weighted sum of its components

$$(3.2) \quad T^j(t) = \sum_{i=1}^{k_j} p_i T_i^j(t)$$

and the expected time to reach plain D for request from source s_j is

$$(3.3) \quad E[T^j] = \sum_{i=1}^{k_j} \frac{p_i}{\mathbf{m}_i - p_i \mathbf{I}_j}$$

At this point TPDU is either delivered or dropped. If a delivery probability for request is the same and is equal p_d then with a probability $1-p_d$ retry is needed.

The probability of exactly K retries for delivery of the request is

$$(3.4) \quad P(K) = (1-p_d)^{K-1} p_d$$

Therefore the mean number of retries is

$$(3.5) \quad E[N_{rt}] = \sum_{i=1}^{N_{max_i}} P(i) = \sum_{i=1}^{N_{max_i}} (1-p_d)^{i-1} p_d$$

We assume that the travel of rejection notification takes one call and travels through the cloud the same time. Therefore the mean time to reach plain C fig 2.1 is computed as

$$(3.6) \quad E[t_d] = E[N_{rt}] \times 2E[T^j] = \sum_{i=1}^{N_{max_i}} (1-p_d)^{i-1} p_d \times 2 \left(\sum_{i=1}^{k_j} \frac{p_i}{\mathbf{m}_i - p_i \mathbf{I}_i} \right)$$

Finally, when a request is recombined from its calls at the plain C its mean delivery time to the server TS becomes

$$(3.7) \quad E[t_j] = \frac{1}{sp_j \left(\sum_{k=1}^{sp_j} k E[t_d] \right)} = \frac{1}{sp_j \left(\sum_{k=1}^{sp_j} k \sum_{i=1}^{N_{max_i}} (1-p_d)^{i-1} p_d \times 2 \left(\sum_{i=1}^{k_j} \frac{p_i}{\mathbf{m}_i - p_i \mathbf{I}_i} \right) \right)}$$

Consider now the multi-unit multi-facility server TS as a separate entity. Let us describe the state of the TS by a vector $\mathbf{n}=(n_1, \dots, n_L)$ where n_i is a number of busy servers of type i (i.e., the number of active threads of service application i .) Then the state space of the system is defined by $\text{State}(\mathbf{c}) = \{ \mathbf{n} | \mathbf{0} \leq \mathbf{n} \leq \mathbf{c} \}$. Here the $\mathbf{0}$ denote all 0's vector. For such a system under the following assumptions:

- The service time for each type of server is exponentially distributed with a mean service time t_i^s .
- The request inter-arrival time is distributed according to a general distribution with the mean time between arrivals of calls of the type r being T^{sa}_r ,

the equilibrium probability distribution is given by a well-known product form (see, e.g., [4]):

$$(3.8) \quad p(\mathbf{n}) = \frac{1}{\sum_{\mathbf{n} \in \text{State}(\mathbf{c})} \prod_{i=1}^L \frac{(\mathbf{r}_i)^{n_i}}{n_i!}} \prod_{i=1}^L \frac{(\mathbf{r}_i)^{n_i}}{n_i!}$$

Here $\mathbf{r}_i = t_i^s / (T^{sa}_i)$. Based on this result (see [5]) blocking probability for a call of type i can be computed as:

$$(3.9) \quad p_0(i) = 1 - \frac{\sum_{\mathbf{n} \in \text{State}(\mathbf{c})} \frac{1}{\prod_{i=1}^L \frac{(\mathbf{r}_i)^{n_i}}{n_i!}}}{\sum_{\mathbf{n} \in \text{State}(\mathbf{c})} \prod_{i=1}^L \frac{(\mathbf{r}_i)^{n_i}}{n_i!}} \times \prod_{i=1}^L \frac{(\mathbf{r}_i)^{n_i}}{n_i!}$$

Here $\mathbf{I}_i = (0, 0, \dots, 0, 1, 0, \dots, 0)$ is a vector pointing in the direction i .

The TS subsystem (the subsystem on fig. 2.1 between plane C and plane D) adhere to the definition above:

- Each server has exponentially distributed with $t_i^s = 1/\mu_i^s$,
- We argue that we know mean inter-arrival time because the mean time between departure of two calls from a system with exponential arrivals and service (time of generating responses) is the same as mean time between arrival of calls. The re-sequencing do not change this relation. Hence

$$T^{sa} = 1/I_j.$$

Therefore, formula 3.9. Apply.

As in circuit switched systems the rejected request can be repeated with persistence probability $P_p(j)$ or it can be dropped with a probability $1-P_p(j)$. Let $P_s(j)=1-P_0(j)$ be the probability that a request was after all delivered. Obviously the probability that the request was served in exactly m tries is

$$(3.10) \quad P_m(j) = (P_0(j)P_p(j))^{m-1}(1-P_0(j))$$

Therefore the mean number of attempts due to the fact that server is busy is computed as

$$(3.11) \quad \begin{aligned} E(N_{RTK}(j)) &= \sum_{i=1}^{\infty} i(P_0(j)P_p(j))^{i-1}(1-P_0(j)) = \\ &= (1-P_0(j)) \left[\sum_{i=1}^{\infty} (i-1)(P_0(j)P_p(j))^{i-1} + \sum_{i=1}^{\infty} (P_0(j)P_p(j))^{i-1} \right] = \\ &= (1-P_0(j)) \left[\sum_{i=0}^{\infty} i(P_0(j)P_p(j))^i + \sum_{i=0}^{\infty} (P_0(j)P_p(j))^i \right] = \\ &= \frac{1-P_0(j)}{(1-P_0(j)P_p(j))^2} \end{aligned}$$

Therefore, the mean time T_j^{comp} between the request of type j arrival in the system and completion of service of this request in the server TS can be computed as follows:

$$(3.12) \quad \begin{aligned} T_j^{comp} &= E(N_{rr}(j))E(t_j) + \max_{j \in \{r|d_{ir} \neq 0\}} \frac{1}{m_j^s} = \\ &= \frac{1-P_0(j)}{(1-P_0(j)P_p(j))^2} E(t_j) + \max_{j \in \{r|d_{ir} \neq 0\}} \frac{1}{m_j^s} \end{aligned}$$

Here $\max_{i \in \{r|d_{ir} \neq 0\}} [1/\mu_i^s]$ is a maximal mean service time of all services that are required (i.e., have nonzero d_{ji}).

At this point the retrieved information is traveling back to a source along one of the same routes it traveled to the TS server. The only difference is that the inter-arrival time between responses to server s_j requests is no longer exponentially distributed. The sojourn time distribution problem for a system of n parallel GI/M/1 queues with re-sequencing buffer was solved in [6] where it was shown that for such system has mean time computed as

$$(3.13) \quad E(T^j) = \sum_{i=1}^n p_i \frac{2 - r_i(1 - s^2 m_i^2)}{2(m_i - \frac{p_i}{t_j})}$$

Here t_j is the

mean inter-arrival time of the call from the j^{th} source, m_i is a service rate of the i^{th} GI/M/1 queue, s_j is its standard deviation and $r_i = p_i/m_j t_j$. This formula would apply if we knew the mean inter-arrival time. Again one can argue that this time is known because the mean time between departure of two calls from a system with general arrivals and exponential service (time of generating responses) is the same as mean time between arrival of calls. Therefore, taking $t_j = 1/I_j$ and $s_j = 2m_i$ and applying (3.13) we get the following time at the plain F of fig 2.1.:

$$(3.14) \quad E(T^j) = \sum_{i=1}^n p_i \frac{2 - r_i(1 - 4m_i^2)}{2(m_i - p_i I_i)}$$

Finally, a response to a call from a source s_i generates anywhere between a and asp_i calls. The number of calls is a random variable that takes values $\{a, \dots, asp_i\}$ with a

$$\frac{1}{a(sp_i - 1)}$$

probability

Therefore, after recombination the mean delivery time from Ts to a source s_i is

$$(3.15) \quad \begin{aligned} E(t_i^{rec}) &= \frac{1}{asp_i \left(\sum_{j=a}^{asp_i} j E(T^j) \right)} = \\ &= \frac{1}{asp_i \left(\sum_{j=a}^{asp_i} j \sum_{i=1}^n p_i \frac{2 - r_i(1 - 4m_i^4)}{2(m_i - p_i I_i)} \right)} \end{aligned}$$

To

complete our investigation we need to take into account both (3.12) and (3.15) to receive mean sojourn time of a request from the source s_j in the system

$$(3.16) \quad \begin{aligned} E(T_{system}^j) &= \left(\frac{1-P_0(j)}{(1-P_0(j)P_p(j))^2} \right) \times \\ &\times \left(\frac{1}{asp_j \left(\sum_{k=a}^{asp_j} k \sum_{i=1}^n p_i \frac{2 - r_i(1 - 4m_i^4)}{2(m_i - p_i I_i)} \right)} \right) + \\ &+ \max_{j \in \{r|d_{ir} \neq 0\}} \frac{1}{m_j^s} + \sum_{i=1}^n p_i \frac{2 - r_i(1 - 4m_i^4)}{2(m_i - p_i I_i)} \end{aligned}$$

4. Conclusions

As stated in the introduction we in this paper we obtained mean sojourn times for an Internet information retrieval. The importance of this result comes from the fact that to the best of our knowledge this is the first attempt to model the system in its entirety. However, in conclusion we'd like to mention a number simplifying assumptions that are fundamental for this model but do not always hold in a system under consideration:

- All distributions that are model parameters are exponential which is not always or even predominantly the case in reality;
- A source is expected to generate more requests before receiving a retrieved information. The latter assumption allows us to use open queuing network as a system model as opposed to more realistic model of closed queuing network;
- The assumption of the negative confirmation TPDU travel time from a re-sequencing buffer to a source being the same as a original call travel time is a simplification that only hold if a server re-sequencing policy is FCFS which is usually not the case.

Therefore, further work is needed to remove the aforementioned limitations.

References

- [1] Feller W. An Introduction to Probability Theory and its Applications. V.2. Wiley.1970
- [2] Kleinrock L. Queuing Systems, vol. 1: Theory. Wiley, 1975
- [3]Baccelli, F., Makowski, A.M., Shwartz, A. The fork join Queue and related systems with synchronization constraints. Adv. Appl. Prob., v.21, N. 9, 1989, pp. 967-989
- [4]Burman, D.Y., Lehoczky, J.P., Lim, Y. Insensitivity of blocking probabilities in a circuit switching network. Adv. Appl. Prob., v.16, N.8., 1984, pp. 850-859.
- [5] Kelly F.P. Blocking probabilities in large circuit-switched networks. Adv. Appl. Prob., v. 18., N.5, 1986, pp. 473-505
- [6] Jean-Marie A., Gun L. Parallel queues with resequencing. Journal of the ACM, v.40, N.5, 1993, pp. 1188-1208.