Analysis for instances’ staying-time distribution in the workflow

Liu Sheng\(^1,2\), Fan Yushun\(^3\)

\(^1\)Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China
\(^2\)Graduate School of Chinese Academy of Sciences, Beijing 100086, China
\(^3\)Department of Automation, Tsinghua University, Beijing, China

liusheng@sia.cn, fanyus@tsinghua.edu.cn

Abstract:

Many workflow applications have time constraints. The serving time of the servers and the arrival intervals of users’ service requests in a workflow are often stochastic. From the statistic angle, we can find a tiny proportion of service requests will be executed beyond the deadline in any case. Thus people can only require an acceptable proportion of service requests to be finished within the deadline normally. We try to determine its probability density function at a workflow network so that we will know the accurate proportion of requests that can be executed without delay. We also present a method to improve the proportion of the undelayed with the lowest additional cost. An experiment illustrates our method can be effectively utilized in practice.

Introduction:

A workflow is an abstraction of an automated and computerized business process. It consists of a set of activities that are interconnected by control flows of a workflow. Each activity is a naturally defined task in a workflow and has associated servers that are either humans or executions of programs commonly called processes in the UNIX world [4]. Each service request triggers a workflow processing. Many business processes have time constraints called deadlines in their corresponding workflows. Actually the servers’ serving time and the arrival intervals of users’ service requests in a workflow are often stochastic. There is a tiny proportion of service requests will be executed beyond the deadline in any case. But we can say a business process is successful if the proportion of requests that are executed within the deadline is large enough. In this paper we discuss the performance problem of the workflow with deadline and proportion constraints.

The performance analysis problem for time-constrained workflow has been paid great attentions in recent years [1-5]. A method is brought out to compute the average time for which a user’s service request is served in a workflow where there is only one server for each activity [12]. The time for which users’ service requests remain in a workflow, called requests’ executed time, including the waiting time and the time for which they are served, shows the workflow traffic conditions for the requests. When many requests exist in a workflow simultaneously, the requests should queue for the services provided by the servers for each activity. Requests’ executed time depends on their arrival rate, the servers’ number and service rates for each activity, and the activities’ interconnected relations. It’s very difficult to analyze requests’ executed time in a workflow quantificationally. In order to improve the proportion within the deadline with lowest additional cost in a workflow, precise quantification analysis of requests’ executed time in a workflow is necessary and effective. This paper will discuss it systematically.

In this paper we assume that servers for every activity in a workflow system have exponential service time and the arrival process of users’ service requests is a Poisson process. Therefore the workflow system can be modeled as an M/M/C queuing network where each activity is an independent M/M/C queuing system.

A workflow is an activity network where activities are interconnected by four workflow control structures, i.e., sequence, concurrency (AND), alternative (OR) and iteration (LOOP) (Lawrence, 1997). We call the activity network as workflow structure. A workflow instance presents an actual process in execution. The arrival and departure processes of all activities in the
M/M/C queuing network can be stated as in Fig. 1.

![Workflow queuing network](image)

**Fig. 1. Workflow queuing network**

When the arrival rate of users' service requests, the service rate of the servers for each activity, the number of servers for each activity and the workflow structure are specified during a workflow definition, the distribution of requests' executed time in the workflow can be worked out in consequence.

In M/M/C, $\lambda$ is an arrival rate, $\mu$ is a service rate, and $c$ is the number of the servers. Probability density function for time distribution is a tool to analyze time distribution precisely. Let $fr(t)$, $fw(t)$ and $fs(t)$ denote the probability density function of executed, waiting and served time distribution respectively, then we will work out $fr(t)$ at a workflow.

Since activities in a workflow are interconnected by four control structures, requests’ $fr(t)$ at a workflow may be worked out by three steps: to work out $fr(t)$ at each activity, to work out $fr(t)$ at each control structure, to work out $fr(t)$ at the whole workflow.

In this paper how to work out $fr(t)$ at an activity will be discussed first.

**How to work out requests’ $fr(t)$ at an activity**

It is supposed there are $c_i$ servers for activity $i$ and $m_i$ requests in activity $i$. Each server serves for one queue. Each request will enter the shortest queue when it arrives. Let $l_i$ denote the requests number in the shortest queue, then $l_i = \text{Round}(m_i/c_i)$. So when a new request arrives, if $m_i < c_i$, it needn’t wait, if $m_i \geq c_i$, it has to wait. Let $P_{m_i}$ denote the probability with which the requests number in the activity $i$ is $m_i$. Let $P_{\min(l_i)}$ denote the probability with which the requests number in the shortest queue is $l_i$. Let $fr(t|l_i)$ denote probability density of last requests’ executed time distribution in the queue with $l_i$ requests in. As is known, last requests’ executed time distribution is an Erlang distribution. Then about activity $i$, $fr(t) = \mu e^{-\mu t}$

$$P_{m_i} = \sum_{m=0}^{c_i} \frac{(\lambda / \mu)^m}{m!}$$

$$P_{\min(l_i)} = \sum_{m_{\min(l_i)}}^{c_i} P_{m_i}$$

When $l_i \geq 1$, $fw(t)$ of activity $i$

$$fw(t) = \sum_{l_i=0}^{\infty} P_{\min(l_i)}fr(t|l_i) = \sum_{l_i=1}^{m} \frac{\sum_{m_{\min(l_i)}}^{c_i} P_{m_i} \mu_i (\mu_i t)^{(l_i-1)} e^{-\mu_i t}}{(l_i-1)!}$$

It can be rewritten as

$$fw(t) = \sum_{l_i=1}^{m} \frac{P_0 (\mu_i t)^{(l_i-1)} e^{-\mu_i t}}{(c_i-1)! * (\mu_i * c_i - \lambda_i)}$$

Let

$$A_i = \frac{P_0 (\mu_i t)^{(l_i-1)} e^{-\mu_i t}}{(c_i-1)! * (\mu_i t)^{(l_i-1)} e^{-\mu_i t}}$$

$$B_i = \left(\frac{\lambda_i}{\mu_i * c_i}\right)^{l_i}.$$
\[ f_{W_i}(t) = \sum_{j=1}^{n_i} \left( A_i \cdot B_i^{j-1} \cdot e^{-\mu_i^j \cdot t} \cdot \frac{(\mu_i^j)^{i-1}}{(i-1)!} \right) \]

\[ = A_i B_i e^{(B_i - 1) \mu_i} \]  

When a request arrives, there is some probability of waiting if \( l_i = 0 \) and some probability of being served if \( l_i > 0 \). So

\[ \int f_r(t) \, dt = \int f_{W_i}(t) \int f_{S_i}(w) \, dw \, dt \]

\[ + P_{min(0)} \int f_{S_i}(t) \, dt \]

\[ = \int f_{W_i}(t) \int \mu_i e^{-\mu_i^j \cdot t} \, dw \, dt \]

\[ + \sum_{m=0}^{c-1} P_{m_i} \mu_i e^{-\mu_i^j \cdot t} \]

The differential equation about \( x \) of Eq. (7) is

\[ f_r(x) = \mu_i e^{-\mu_i^j \cdot x} \int f_{W_i}(t) e^{\mu_i^j \cdot t} \, dt \]

\[ + \sum_{m=0}^{c-1} P_{m_i} \mu_i e^{-\mu_i^j \cdot x} \]

\[ = \mu_i e^{-\mu_i^j \cdot x} \int A_i B_i e^{(B_i - 1) \mu_i} \, dt \]

\[ + \sum_{m=0}^{c-1} P_{m_i} \mu_i e^{-\mu_i^j \cdot x} \]

\[ = A_i e^{(B_i - 1) \mu_i^j} + \left( \mu_i \sum_{m=0}^{c-1} P_{m_i} - A_i \right) e^{-\mu_i^j \cdot x} \]

(8)

Specially, if there is only one server for activity \( i \),

\[ f_r(x) = (\mu_i - \lambda_i) e^{(\lambda_i - \mu_i) \cdot x} + \left( \mu_i \sum_{m=0}^{c-1} P_{m_i} - \lambda_i \right) e^{-\lambda_i \cdot x} \]

\[ = (\mu_i - \lambda_i) e^{(\lambda_i - \mu_i) \cdot x} \]

Thus we work out requests’ \( f_r(t) \) at an activity.

How to work out requests’ \( f_r(t) \) at four control structures

Because requests’ \( f_r(t) \) at an activity is worked out, if the relation of requests’ \( f_r(t) \) at an activity and four control structures is known, we will be able to work out requests’ \( f_r(t) \) at four control structures. Let \( f_{r_{n_1 \ldots n_m}}(t) \) denote requests’ \( f_r(t) \) at a control structure made of activity \( n_1 \), activity \( n_2 \) ... and activity \( n_m \). And the relation will be analyzed in the following part. Normally, requests’ executed time at each activity is independent.

Requests’ \( f_r(t) \) at concurrent control structure

Activity 4 and activity 5 in Fig. 1 are connected by a concurrent control structure. Requests’ executed time at a concurrent control structure is the longest executed time at its branches. A request at a concurrent control structure with \( n \) branches can be divided into \( n \) independent sub-requests which may run in parallel. As to Activity 4 and activity 5 in Fig. 1,

\[ \int t \cdot f_{r_{45}}(t) \, dt = \int u \cdot f_{r_4}(u) \int f_{r_5}(v) \, dv \, du \]

\[ + \int u \cdot f_{r_5}(u) \int f_{r_4}(v) \, dv \, du \]

The differential equation about \( x \) of equation (10) is

\[ f_{r_{45}}(x) = f_{r_4}(x) \int f_{r_5}(v) \, dv + f_{r_5}(x) \int f_{r_4}(v) \, dv \]

When a concurrent control structure is made of activity 1, activity 2 ... and activity \( n \),

\[ f_{r_{12 \ldots n}}(x) = \sum_{i=1}^{n} \left( \int f_{r_i}(x) \prod_{k=1}^{n} f_{r_k}(t) \, dt \right) \]

Requests’ \( f_r(t) \) at sequential control structure

Requests’ executed time at a sequential control structure is the sum of requests’ executed time at all of its branches. Let \( u_1 \), \( u_2 \) ... and \( u_n \) denote requests’ executed time at activity 1, activity 2 ... and activity \( n \) connected by a sequential control structure, \( t \) denote requests’ executed time at the structure. We know \( t = u_1 + u_2 + \cdots + u_n \). So

\[ \int f_{r_{12 \ldots n}}(t) \, dt \]

\[ = \int \int \cdots \int f_{r_1}(u_1 + u_2 + \cdots + u_n) \, du \, dt \]

\[ = \int \int \cdots \int \prod_{i=1}^{n} f_{r_i}(u_i) \, du_1 \, du_2 \cdots du_n \]

Requests’ \( f_r(t) \) at a sequential control structure can
be achieved according to equation (13).

Requests’ $fr(t)$ at alternative control structure

Each request at an alternative control structure is exclusively served at branch $i$ with probability $P_i$. When an alternative control structure is made of activity 1, activity 2... and activity $n$, Requests’ $fr(t)$ at alternative control structure

$$fr_{12...n}(t) = \sum_{i=1}^{n} P_i fr_i(t)$$

(14)

Activity 6 and activity 7 in Fig.1 are connected by a loop control structure. Requests return to be served again at activity 6 via activity 7 with probability $q$ after they pass it. So a request is served for $n$ times by the servers for activity with probability $P_n = q^{(n-1)}(1-q)$. The loop control structure shown in Fig.2 (a) is equivalent to the alternative structure shown in Fig. (b).

Fig.2 Loop control structure and its equivalent structure

In fact the probability with which a request is served for 3 or more times in loop control structure is so small that may be ignored. Here we regard the loop control structure as an alternative structure with 3 branches. Let $fr_6(t)$, $fr_{67}(t)$ and $fr_{676}(t)$ denote requests’ $fr(t)$ at the first three branches of equivalent alternative structure in Fig.2(b). According to equation (14),

$$fr_{loop}(t) = (1-q)^2 fr_6(t) + q(1-q)^2 fr_{67}(t) + q^2 (1-q) fr_{676}(t)$$

(15)

Because branch 2 is sequential control structure as well as branch 3, according to equation (13),

$$\int fr_{676}(t) dt =$$

$$\int \int fr_6(w) fr_7(v) fr_6(u) dwdvdu$$

(16)

Requests’ $fr(t)$ at loop control structure

How to work out requests’ $fr(t)$ at a workflow

When we have worked out requests’ $fr(t)$ at every activity of a workflow, we can work out requests’ $fr(t)$ at every control structure of a workflow according to equation (12), equation (13), equation (14) and equation (15). Then we regard every control structure as an activity at which requests’ $fr(t)$ is the same as the one at it. After this step, the workflow network is simplified with reduced quantity of activities. Then we work out requests’ $fr(t)$ at every control structure of the simplified workflow network and simplify it again until only one activity is left. Requests’ $fr(t)$ at the activity
is requests’ \( f_r(t) \) at the workflow. Thus we work out requests’ \( f_r(t) \) at a workflow.

How to improve proportion within the deadline with lowest additional servers’ cost

If we increase the number of servers for a workflow, the proportion of requests that are executed within the deadline will increase too. But the cost and effect to the executed time of each server is different. Let \( C_i \) denote the cost of one server for activity \( i \), \( P_c \) denote current proportion within the deadline, \( P_n \) denote required proportion within the deadline, and \( P_{r;i} \) denote proportion within the deadline when a server for activity \( i \) is added into the workflow. When \( P_c < P_{re} \), if \((P_{r;i} - P_c)/C_i = \max((P_{r;i} - P_c)/C_i) \), a server will be added into the workflow for activity \( k \) prior to other activities. The same effort will be repeated until \( P_c = P_{re} \) (Namely the proportion is OK).

Performance analysis experiment

With the arrival rate of users’ service requests, the service rate of each server for each activity, the number of servers for each activity and the probability of each branch in alternative and loop control structures in a workflow definition, requests’ \( f_r(t) \) at the workflow can be worked out. Table 1 lists the necessary data about the workflow described in Fig.1.

<table>
<thead>
<tr>
<th>Activity</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Servers number</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( \mu )</td>
<td>16</td>
<td>20</td>
<td>20</td>
<td>36</td>
<td>24</td>
<td>40</td>
<td>10</td>
</tr>
<tr>
<td>( \lambda ) (deduced)</td>
<td>8</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>20</td>
<td>2</td>
</tr>
</tbody>
</table>

We cannot ensure that every service request shall be executed before the deadline because the service rate of every server is stochastic. But we can ensure an acceptable proportion of users’ service requests shall be executed before the deadline. Let \( P(0 < t < T) \) denote the proportion of requests shall be executed before the deadline \( T \). Let \( E(t) \) denote the average executed time of users’ service requests. So

\[
P(0 < t < T) = \int_0^T f_r(t) dt \tag{18}
\]

\[
E(t) = \int_0^T t \cdot f_r(t) dt \tag{19}
\]

In the experiment, experiences tell us that the users are satisfied if 98% of requests can be executed within 1 time unit. We want to know \( P(0 < t < 1) \) and \( E(t) \) of users’ service requests. First we must work out requests’ \( f_r(t) \).

According to equation (8), we can know

\[
f_{r_1}(t) = 8e^{-8t}, f_{r_2}(t) = 12e^{-12t}, f_{r_3}(t) = 10e^{-10t},
\]

\[
f_{r_4}(t) = 26e^{-26t}, f_{r_5}(t) = 14e^{-14t}, f_{r_6}(t) = 20e^{-20t},
\]

\[
f_{r_7}(t) = 8e^{-8t}.
\]

Activity 4 and activity 5 make up of a concurrent structure (called structure 1). According to equation (12),

\[
f_{r_{15}}(t) = 26e^{-26t} + 14e^{-14t} - 40e^{-40t}.
\]

Activity 6 and activity 7 make up of a loop structure (called structure 2). According to equation (15), (16) and (17),

\[
f_{r_{17}}(t) = e^{-8t}(1.26 + 2.96t) + e^{-20t}(16.74 - 18.07t + 17.78t^2)
\]

Activity 3 and structure 1 make up of a sequential structure (called structure 3). According to equation (13),

\[
f_{r_{135}}(t) = \frac{160e^{-40t} - 195e^{-26t} - 420e^{-14t} + 455e^{-10t}}{12}.
\]

Activity 1 and activity 2 make up of a sequential structure (called structure 4). According to equation (13),

\[
f_{r_{12}}(t) = 24e^{-8t} - 24e^{-12t}.
\]

Structure 3 and structure 4 make up of an alternative structure (called structure 5). According to equation (14),

\[
f_{r_{12345}}(t) = \frac{5}{108} \left( 160e^{-40t} - 195e^{-26t} - 420e^{-14t} + 455e^{-10t} \right).
\]

\[
+ \frac{32}{3} \left( e^{-8t} - e^{-12t} \right).
\]

Structure 2 and structure 5 make up of a sequential structure (called structure 6). According to equation (13),

776
\[ f_{1234567}(t) = e^{-8t}(7.32 + 26.38t + 15.78t^2) + \\
- e^{-20t}(-2.86 + 15.19t + 7.87t^2) \\
- 6.86e^{-40t} + 31.39e^{-26t} - 45.47e^{-14t} \\
- 18.46e^{-12t} + 34.65e^{-10t} \] (20)

Equation (20) is requests’ \( f_r(t) \) at the workflow shown in Fig. 1. Fig. 3 shows that requests’ executed time distribution at the workflow is an approximate normal distribution. According to equation (18) and equation (19),

\[
P(0 < t < 1) = \int f_{1234567}(t) dt = 99.8\% > 98\% ,
\]

\[
E(t) = \int t * f_{1234567}(t) dt = 0.265.
\]

Obviously the users are satisfied. Otherwise we must add servers for the most busy activity and compute the \( f_r(t) \) again.

**Conclusion**

Workflow management systems have gained increasing attention and have been applied in many areas. Many workflow applications have time constraints so that an acceptable proportion of service requests need to be finished within given deadline. In this paper we have presented a method to work out the probability density function (called \( f_r(t) \)) of requests’ executed time distribution at a workflow. First we work out requests’ \( f_r(t) \) at each activity. Then we work out requests’ \( f_r(t) \) of each control structure. And last we work out requests’ \( f_r(t) \) at the workflow. According to it we will know the average requests’ executed time and can analyze whether the users are satisfied with the workflow processing. In the last we give a suggestion to improve the service level if the users are not satisfied with the proportion within the deadline.

**References**


