Chapter 4

Arrival Time and Queuing Delay Estimation

Queue management and analysis are crucially important for various control actions and resource management. In chapter 3 we have proposed a QoS control architecture based on queueing delay and queue size. It is necessary to measure the queueing delay and queue size on regular intervals. To know the delay of an individual item in a queue, the option is to store the arrival time. But on heavily loaded routers it is neither feasible nor scalable if the queue size is very large and the arrival rate and service rate are too heavy. There will be a huge requirement of computational power and memory space. Hence we need a time and space efficient algorithm which is tunable and scalable to estimate the delay of items in a queue as and when required.

There are some applications which require accuracy in delay estimation and others do not require much accuracy. Similarly there are routers
with different capacities and limitations. We need a strategy which is flexible and suitable to the applications and routers, and is tunable to the requirements and limitations. We propose a tunable strategy that can estimate the arrival time of items in a FIFO queue with an upper limit on the magnitude of Absolute Error in estimation. Larger the upper limit, smaller will be the memory space and computational time required and vice versa. Thus we can tune the space and time complexity as desired. We have based our strategy on FIFO queue, because a multiple set of FIFO queues with separate assigned priorities works very efficiently. It is scalable because with higher arrival rates the computing time and memory space requirements do not increase. It adapts to the available resources and required accuracy.

4.1 The Strategy

In this strategy the queue size and the number of items served are recorded periodically after a fixed time period of choice. Let this time period be $\Delta t$. The number of items served and the queue size are dynamically stored in a table as shown in Table 4.1. We term this table as Queue State Table (QST). After every $\Delta t$ time period a new entry is appended at the end of table QST and a few entries from the top of the table QST are deleted.

4.1.1 Appending the entry in Queue State Table

Let $t_h$ be the time when the last entry in QST was updated and $\Delta t$ be the QST updation interval, then for an integer $l$ we define $t_{h-l}$ as per
equation (4.1).
\[ t_{h-l} = t_h - l \times \Delta t; \quad \forall l \in I \quad \ldots \ldots (4.1) \]

where \( I \) is the set of integers. Let \( t_c \) be the current time. The relationship between \( t_c \) and \( t_h \) is defined as per equation (4.2)

\[ t_c - t_h \leq \Delta t; \quad \ldots \ldots (4.2) \]

Table 4.1 shows Queue State Table (QST) at the time \( t_c \). The \( Q_{h-l} \) is the queue size at time \( t_{h-l} \) and \( S_{h-l+1,h-l} \) is the number of items served during the time period of \( t_{h-l} \) to \( t_{h-l+1} \).

<table>
<thead>
<tr>
<th>Time</th>
<th>Items Served</th>
<th>Queue Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_{h-l} )</td>
<td>( S_{h-l+1,h-l} )</td>
<td>( Q_{h-l} )</td>
</tr>
<tr>
<td>( t_{h-2} )</td>
<td>( S_{h-1,h-2} )</td>
<td>( Q_{h-2} )</td>
</tr>
<tr>
<td>( t_{h-1} )</td>
<td>( S_{h,h-1} )</td>
<td>( Q_{h-1} )</td>
</tr>
<tr>
<td>( t_h )</td>
<td>( S_{c,h} )</td>
<td>( Q_h )</td>
</tr>
<tr>
<td>( t_c )</td>
<td>( S_{c,h} )</td>
<td>( Q_c )</td>
</tr>
</tbody>
</table>

When the current time \( t_c \) becomes as per equation (4.3), a new entry in the table QST is appended as shown in Table 4.2.

\[ t_c - t_h = t_{h+1} - t_h = \Delta t \quad \ldots \ldots (4.3) \]

Equation (4.3) shows that \( t_c = t_{h+1} \). So no item between the time period \( t_{h+1} \) to \( t_c \) would have arrived, because the length of this period is zero \( (0) \). In this case the number of items served \( (S_{c,h+1}) \) during the time period of \( t_{h+1} \) to \( t_c \) and queue size at \( t_{h+1} \) are specified by equation (4.4a) and
At this time the number of items served \((S_{h+1,h})\) during the time period \(t_h\) to \(t_{h+1}\) is as per equation (4.5)

\[
S_{h+1,h} = S_{c,h} \quad \ldots \ldots (4.5)
\]

The \(QST\) updation is as shown in Table 4.2.

<table>
<thead>
<tr>
<th>Time</th>
<th>Items Served</th>
<th>Queue Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
</tr>
<tr>
<td>(t_{h-1})</td>
<td>(S_{h-1,h-1})</td>
<td>(Q_{h-1})</td>
</tr>
<tr>
<td>(\ldots)</td>
<td>(\ldots)</td>
<td>(\ldots)</td>
</tr>
<tr>
<td>(t_{h-2})</td>
<td>(S_{h-2,h-2})</td>
<td>(Q_{h-2})</td>
</tr>
<tr>
<td>(t_{h-1})</td>
<td>(S_{h,h-1})</td>
<td>(Q_{h-1})</td>
</tr>
<tr>
<td>(t_h)</td>
<td>(S_{h+1,h})</td>
<td>(Q_h)</td>
</tr>
<tr>
<td>(t_{h+1} = t_c)</td>
<td>(S_{c,h+1} = 0)</td>
<td>(Q_{h+1} = Q_c)</td>
</tr>
<tr>
<td>(t_c)</td>
<td>(S_{c,h+1} = 0)</td>
<td>(Q_c)</td>
</tr>
</tbody>
</table>

When the current time \(t_c\) is as per equation (4.6), The table \(QST\) is as shown in Table 4.3.

\[
0 < t_c - t_{h+1} < \Delta t \quad \ldots \ldots (4.6)
\]

In this case the number of items served \((S_{c,h+1})\) during the time period of \(t_{h+1}\) to \(t_c\) and queue size at \(t_{h+1}\) will be specified by equation (4.7a) and
In this way a row in the table is appended and updated after every $\Delta t$ time period. By using QST it is possible to calculate the arrival time with an upper limit on the magnitude of Absolute Error in estimation.

### 4.1.2 Arrival Time Calculation

**Theorem**: By using the Queue State Table (QST) it is possible to evaluate the arrival of an item in a queue with an upper limit of $\Delta t$ on the magnitude of Absolute Error.

**Proof**: Let us assume that we have to calculate the arrival time of $n^{th}$ item ($I_n$) from the rear of the queue. The queue state table (QST) is as per Table 4.1. The number of elements arrived during the time period of $t_c$ to $t_h$ are say $N_{c,h}$. Then $N_{c,h}$ will be evaluated as per equation (4.8).

$$N_{c,h} = Q_c - Q_h + S_{c,h} \quad \ldots \ldots (4.8)$$
According to the value of $N_{c,h}$, there are two cases as per equation (4.9)

\[
\begin{align*}
\text{Case 1:} & \quad N_{c,h} \geq n \\
\text{Case 2:} & \quad N_{c,h} < n
\end{align*}
\]

**Case 1 ($N_{c,h} \geq n$):** In case $N_{c,h} \geq n$ then this means that more than $n$ elements have arrived during the time interval between $t_c$ and $t_h$. Or the $n^{th}$ item $I_n$ has arrived before $t_c$ and after $t_h$.

We perfectly know that the item $I_n$ had arrived during the time interval between $t_c$ and $t_h$. If $T_n$ is the actual arrival time of the item $I_n$ then $T_n$ will satisfy the equation (4.10)

\[
t_h \leq T_n \leq t_c \quad \text{........(4.10)}
\]

In this case the arrival time ($ATE_n$) of $I_n$ will be estimated as per equation (4.11)

\[
t_h \leq ATE_n \leq t_c \quad \text{........(4.11)}
\]

The magnitude of the Absolute Error in Arrival Time estimation ($e_n$) will be as per equation (4.12)

\[
e_n = |T_n - ATE_n| \quad \text{........(4.12)}
\]

As $t_c - t_h \leq \Delta t$ (as per equation (4.2)), it is clear that the upper limit on $|T_n - ATE_n|$ is as per equation (4.13a). So the upper limit on $e_n$ is
specified in equation (4.13b).

\[
\begin{align*}
\left| T_n - ATE_n \right| & \leq t_c - t_h \leq \Delta t \quad \text{(4.13a)} \\
|e_n| & \leq t_c - t_h \leq \Delta t \quad \text{(4.13b)}
\end{align*}
\]

Thus the upper limit on the magnitude of Absolute error is $\Delta t$.

**Case2** ($N_{c,h} < n$): In case $N_{c,h} < n$ then this implies that the $I_n$ had not arrived during the time interval of $t_c$ to $t_h$. The number of items arrived ($N_{c,h}$) before $t_c$ and after $t_h$ are as per equation (4.14):

\[
N_{c,h} = Q_c - Q_h + S_{c,h} \quad \text{(4.14)}
\]

The number of items arrived ($N_{h,h-1}$) before $t_h$ and after $t_{h-1}$ are as per equation (4.15):

\[
N_{h,h-1} = Q_h - Q_{h-1} + S_{h,h-1} \quad \text{(4.15)}
\]

Similarly the number of items arrived during the time before $t_{h-l+1}$ and after $t_{h-l}$ are as per equation (4.16).

\[
N_{h-l+1,h-l} = Q_{h-l+1} - Q_{h-l} + S_{h-l+1,h-l} \quad \text{(4.16)}
\]

The number of items arrived ($N_{c,h-l}$) before $t_c$ and after $t_{h-l}$ is as per equation (4.17):

\[
N_{c,h-l} = N_{c,h} + \sum_{i=1}^{l} N_{h-i+1,h-i} \quad \text{(4.17)}
\]
Solving the equation (4.17) with the help of equations (4.14), (4.15) and (4.16) we get the equation (4.18)

\[
N_{c,h-l} = Q_c - Q_h + S_{c,h} + \sum_{i=1}^{l} (Q_{h-i+1} - Q_{h-i} + S_{h-i+1,h-i}) \quad \ldots (4.18)
\]

Further solving equation (4.18) we get equation (4.19)

\[
N_{c,h-l} = Q_c - Q_{h-l} + S_{c,h} + \sum_{i=1}^{l} S_{h-i+1,h-i} \ldots (4.19)
\]

In this way we find a value of \( l \) such that the following equation (4.20) is satisfied.

\[
N_{c,h-l+1} < n \leq N_{c,h-l} \ldots (4.20)
\]

According to equation (4.20), the number of elements arrived after time \( t_{h-l+1} \) are less than \( n \) whereas the number of items arrived after the time \( t_{h-l} \) are more than or equal to \( n \). This means it is certain that the item \( I_n \) had arrived before \( t_{h-l+1} \) and after \( t_{h-l} \). If the the actual arrival time of item \( I_n \) is \( T_n \) then \( T_n \) will satisfy the equation (4.21)

\[
t_{h-l} \leq T_n < t_{h-l+1} \ldots (4.21)
\]

In this case an arrival time \( (ATE_n) \) of \( I_n \) will be estimated as per equation (4.22)

\[
t_{h-l} \leq ATE_n < t_{h-l+1} \ldots (4.22)
\]
The magnitude of the Absolute Error in Arrival Time estimation ($e_n$) will be as per equation (4.23)

$$e_n = |T_n - ATE_n| \quad \ldots (4.23)$$

It is clear that the upper limit on $|T_n - ATE_n|$ is as per equation (4.24a). So the upper limit on $e_n$ is specified in (4.24b).

$$\begin{align*}
|T_n - ATE_n| &\leq t_{h-l+1} - t_{h-l}; \quad \ldots (4.24a) \\
e_n &\leq t_{h-l+1} - t_{h-l}; \quad \ldots (4.24b) \\
t_{h-l+1} - t_{h-l} &= \Delta t \quad \ldots (4.24c)
\end{align*}$$

Thus the upper limit on the Absolute error is $\Delta t$.

Hence it is proved that for any element ($I_n$) in queue, it is possible to estimate the Arrival Time with an upper limit of $\Delta t$ on the magnitude of Absolute Error ($e_n$).

### 4.1.3 Removing entries from the top of QST

It is clear that at a specified time say $t_c$ there is no need to estimate the arrival time or delay of the items already served before $t_c$ and are not in the queue. For the element at the front of the queue, equation (4.20) will be as per equation (4.25):

$$N_{c,h-r+1} < Q_c \leq N_{c,h-r} \quad \ldots (4.25)$$
Using equation (4.19) we get $N_{c,h-r}$ as per equation (4.26)

$$N_{c,h-r} = Q_c - Q_{h-r} + S_{c,h} + \sum_{i=1}^{r} S_{h-i+1,h-i} \ldots \ldots \text{(4.26)}$$

Thus after the time $t_c$ the records in the table QST which are above the record containing $Q_{h-r}$ i.e recorded before the time $t_{h-r}$ will not be used to evaluate the arrival time of any item in the queue. These records are deleted and the memory is freed. This process of deletion is done at the same time when a new record is appended in the table.

### 4.1.4 Memory Access Time

When an item in a queue arrives $Q_c$ is updated, and when an item is served $Q_c$ as well as $S_{c,h}$ are updated. To minimize the memory access time in this activity these two variables may be kept in the registers or in a cache memory. After the time interval of $\Delta t$, these variables are updated in QST. Thus the table QST is accessed and updated periodically after the fixed time period $\Delta t$.

### 4.1.5 Size of QST

Let there be 'a' number of bytes required to store the queue size in the record and 'b' number of bytes required to store the number of items served. In such case the number of bytes required to store a single record in the table QST are 'a + b'. From equation (4.25) and (4.26) it can be calculated that the item at the front of the queue would have arrived between the time $t_{h-r}$ and $t_{h-r+1}$. The delay of the item at the
front of the queue \((\text{Delay}_{\text{Front}})\) is as per equation (4.27)

\[(t_c - t_{h-r}) > \text{Delay}_{\text{Front}} > (t_c - t_{h-r+1}) \quad \ldots \ldots \quad (4.27)\]

Solving equation (4.27) with the help of equation (4.1) and (4.2) we get equation (4.28)

\[r \Delta t + t_c - t_h > \text{Delay}_{\text{Front}} > (\Delta t(r - 1) + t_c - t_h) \quad \ldots \ldots \quad (4.28)\]

subtracting \((t_c - t_h)\) from each side and then divide the equation (4.28) by \(\Delta t\) we get equation (4.29)

\[r > \frac{\text{Delay}_{\text{Front}} - (t_c - t_h)}{\Delta t} > (r - 1) \quad \ldots \ldots \quad (4.29)\]

Further solving equation (4.29) we get the equation (4.30)

\[r < \frac{\text{Delay}_{\text{Front}} - (t_c - t_h)}{\Delta t} + 1 \quad \ldots \ldots \quad (4.30)\]

We know that the minimum value of \(t_c - t_h\) is 0. By putting it in the equation (4.30) we get equation (4.31),

\[r < \frac{\text{Delay}_{\text{Front}}}{\Delta t} + 1 \quad \ldots \ldots \quad (4.31)\]

It is clear from equation (4.25) that the number of the records in the table are \(r + 1\) excluding the record which contains the values of \(t_c\) and \(S_{c,h}\). The upper limit on the size of Queue State Table (QST) is as per equation (4.32)

\[QST_{Size} \leq r + 1 \quad \ldots \ldots \quad (4.32)\]
'a + b' is the size of a record in $QST$. Thus using equation (4.31) in (4.32) we get the upper bound on the size of $QST$ as per equation (4.33):

$$QST \text{Size} \leq \left( \frac{\text{Delay}_{\text{Front}}}{\Delta t} + 2 \right) \times (a + b) \quad \ldots \ldots (4.33)$$

Thus the size of the Queue State Table ($QST$) is limited as per equation (4.33).

### 4.2 Algorithm

The algorithm is divided into two procedures. One is to update the table $QST$ and the other one is to evaluate the arrival time. The procedure to update the table $QST$ is shown in Figure 4.1.

![Procedure QST Updation](image-url)

**Figure 4.1**: Procedure $QST$ Updation
It starts with updating the table $QST$. It appends the entry at the bottom of $QST$ as per section 4.1.1 and also removes the entries from the top of the table $QST$ as per section 4.1.3. The two most preferred data structures to store the table are either linked list or a circular array implementation of queues. After updating the table a timer of the duration of $\Delta t$ starts. At this point the procedure waits for one of the following events to occur and then act accordingly

**Event1-Timer $\Delta t$ elapsed:** In this case the control of the procedure goes back to update the table $QST$ and to restart the timer $\Delta t$.

**Event2-An Item Arrived:** In this case the queue size $Q_c$ is incremented by one. After this updation the control goes back in the state of waiting for events.

**Event3-An Item Served:** In this case $Q_c$ is decremented by one and $S_{c,h}$ is incremented by one. After this updation the control goes back in the state of waiting for events.

The procedure for evaluating the arrival time is shown in Figure 4.2. This procedure accepts an argument specifying the position of the item in the queue for which the arrival time is to be evaluated. If the value of this argument is $n$, this implies that the arrival of the item ($I_n$) at $n^{th}$ position from the rear of the queue is to be evaluated. In case the arrival of the element at the front of the queue is to be evaluated then $n = Q_c$. If $n > Q_c$ the procedure transfers its control to the exception handler because this means that a request has been made to evaluate the arrival of an item which is not present in the queue. If $n \leq Q_c$
the procedure begins to calculate the arrival time. In the first step it calculates the lower bound and upper bound. To calculate the arrival time it assumes that all the items which arrived during the time period from lower bound to upper bound would have arrived uniformly. If the condition of case 1 of the theorem holds true then from equation (4.11) we get the lower bound \( (LB_n) \) and the upper bound \( (UB_n) \) on the arrival time estimation are as per equation (4.34).

\[
\begin{align*}
LB_n &= t_h \\
UB_n &= t_c
\end{align*}
\] .......(4.34)

In this case arrival time of \( I_n (EvArT_n) \) is evaluated as per equation (4.35)

\[
EvArT_n = LB_n + \frac{N_{c,h} - n}{N_{c,h} + 1} \times (UB_n - LB_n) .......(4.35)
\]

If the condition of case2 of the theorem is satisfied then the lower bound \( (LB_n) \) and the upper bound \( (UB_n) \) on arrival time estimation are calculated by using the equations (4.22) and are as per equation (4.36).

\[
\begin{align*}
LB_n &= t_{h-l} \\
UB_n &= t_{h-l+1}
\end{align*}
\] .......(4.36)

Thus in this case the arrival time of \( I_n (EvArT_n) \) is evaluated as per equation (4.37)

\[
EvArT_n = LB_n + \frac{N_{c,h-l} - n}{N_{h-l+1,h-l} + 1} \times (UB_n - LB_n) .......(4.37)
\]
Thus the arrival time evaluated using the equations (4.35) and (4.37) is returned by the procedure.

![Flowchart of Procedure Arrival Time Estimation](image)

Figure 4.2: Procedure Arrival Time Estimation

### 4.3 Simulation and Results Analysis

We have divided the simulation experiments in two sets. Four experiments were conducted in each set. The updation period of the Queue State Table (QST) is different for the experiments in the first set (Table 4.4) and the average arrival rate is different for the experiments.
in the second set (Table 4.5). All the experiments were conducted on a FIFO queue with the Poisson arrival process and the exponential service time. The arrival rate and the service rate are equal. The simulation period was 50,000 s in all the experiments. The sample size shows how many times the experiment of arrival time estimation was conducted. Following results were observed.

- The magnitude of Absolute Error in arrival time estimation ($|e_n|$) always remains less than the upper limit. As per the theorem this upper limit is QST Updation Period ($\Delta T$). A control on Absolute Error governs the space and computational requirements. Thus it can be tuned accordingly.

- The size of Queue State Table (QST) is always less than the upper limit specified in equation (4.33). This means we can effectively check the QST size and set the upper limit suiting to the requirements.

- The computational time is directly proportional to the frequency of the QST updation. The frequency is inversely proportional to the QST Updation Period ($\Delta T$). The upper limit on the magnitude of Absolute Error in arrival time estimation is $\Delta T$ and so it is inversely proportional to the computational time requirements.

- Thus this strategy is tunable with respect to computational time requirement. Higher the upper limit lower will be the computational time and vice versa. In Table 4.4, the upper limit on Absolute Error in experiments Exp2, Exp3, and Exp4 is 5 times.
times, and 20 times respectively of the upper limit in the experiment Exp1. The results show that the magnitude of the Maximum Absolute Error is proportional to the respective limits and is within these limits.

Table 4.4: Results: Experiment Set 1

<table>
<thead>
<tr>
<th>Simulation Period</th>
<th>Exp 1</th>
<th>Exp 2</th>
<th>Exp 3</th>
<th>Exp 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Items Arrived</td>
<td>523</td>
<td>523</td>
<td>523</td>
<td>523</td>
</tr>
<tr>
<td>Items Served</td>
<td>490</td>
<td>490</td>
<td>490</td>
<td>490</td>
</tr>
<tr>
<td>$\Delta T$</td>
<td>10 s</td>
<td>50 s</td>
<td>100 s</td>
<td>200 s</td>
</tr>
<tr>
<td>Sample Size</td>
<td>5,461</td>
<td>5,461</td>
<td>5,461</td>
<td>5,461</td>
</tr>
<tr>
<td>Max. Delay at Front</td>
<td>3312 s</td>
<td>3272 s</td>
<td>3272 s</td>
<td>3266 s</td>
</tr>
<tr>
<td>Upper Limit on $</td>
<td>e_n</td>
<td>\rangle$</td>
<td>10 s</td>
<td>50 s</td>
</tr>
<tr>
<td>Max. $</td>
<td>e_n</td>
<td>\rangle$</td>
<td>9 s</td>
<td>49 s</td>
</tr>
<tr>
<td>Upper Limit on $QST Size$</td>
<td>333</td>
<td>67</td>
<td>34</td>
<td>18</td>
</tr>
<tr>
<td>Max. $QST Size$</td>
<td>333</td>
<td>67</td>
<td>34</td>
<td>18</td>
</tr>
</tbody>
</table>

- The space required is directly proportional to the size of Queue State Table ($QST$). The size of $QST$ is inversely proportional to $QST$ updation period ($\Delta T$) which is also the upper limit on the magnitude of Absolute Error in arrival time estimation. So the memory space required is inversely proportional to the magnitude of Absolute Error.

- Thus this strategy is tunable with respect to the memory space requirements. In Table 4.4, the upper limit on Absolute Error in
experiments Exp2, Exp3, and Exp4 is 5 times, 10 times, and 20 times respectively of the upper limit in the experiment Exp1. The results show that the maximum size of QST is inversely proportional to these limits.

Table 4.5: Results: Experiment set 2

<table>
<thead>
<tr>
<th></th>
<th>Exp 5</th>
<th>Exp 6</th>
<th>Exp 7</th>
<th>Exp 8</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Simulation Period</strong></td>
<td>50,000 s</td>
<td>50,000 s</td>
<td>50,000 s</td>
<td>50,000 s</td>
</tr>
<tr>
<td><strong>Items Arrived</strong></td>
<td>523</td>
<td>2,482</td>
<td>4,888</td>
<td>9,914</td>
</tr>
<tr>
<td><strong>Items Served</strong></td>
<td>490</td>
<td>2,400</td>
<td>4,842</td>
<td>9,802</td>
</tr>
<tr>
<td><strong>ΔT</strong></td>
<td>50 s</td>
<td>50 s</td>
<td>50 s</td>
<td>50 s</td>
</tr>
<tr>
<td><strong>Sample Size</strong></td>
<td>5,461</td>
<td>23,446</td>
<td>17,793</td>
<td>26,935</td>
</tr>
<tr>
<td><strong>Max. Delay at Front</strong></td>
<td>3272 s</td>
<td>2298 s</td>
<td>986 s</td>
<td>684 s</td>
</tr>
<tr>
<td>**Upper Limit on</td>
<td>(</td>
<td>e_n</td>
<td>)**</td>
<td>50 s</td>
</tr>
<tr>
<td>**Max.</td>
<td>(</td>
<td>e_n</td>
<td>)**</td>
<td>49 s</td>
</tr>
<tr>
<td><strong>Upper Limit on QSTSize</strong></td>
<td>67</td>
<td>47</td>
<td>21</td>
<td>15</td>
</tr>
<tr>
<td><strong>Max. QSTSize</strong></td>
<td>68</td>
<td>47</td>
<td>21</td>
<td>15</td>
</tr>
</tbody>
</table>

Size of QST is directly proportional to the delay at the front of the queue as defined in the equation (4.33). The upper limit on the size of QST is determined by the delay at front and QST updation period (ΔT). In table 4.5, ΔT is constant and the upper limit on QST is determined by the maximum delay at front. The results show that the maximum size of QST is within this limit. Because the delay at front is also limited by the maximum delay in the queue, so we can say that the limiting factors of the QST size are ΔT and maximum delay in the queue. The volume of traffic does not affect the QST size. Thus this strategy is scalable.