4. Admission Control Algorithms and Traffic Models

From the baseline strategy in Sec 3.3, we first drive **static** admission control algorithm in Sec. 4.1, and then **adaptive** admission control algorithm in Sec 4.2. To study the behaviors of proposed static and adaptive admission control, we consider traffic models with stationary arrivals, nonstationary arrivals, and nonhomogeneous arrivals, which are described in Sec 4.3.

4.1 Static time-based admission control

Let the busy status Q(t) be the number of unfinished service requests remaining in service queue, i.e., $Q(t) = N_q(t)$. Then, the busy threshold T_b limits the number of requests that can be accepted without any constraint. Then *the number-based admission control without buffering request* essentially is the baseline admission control strategy in Sec 3.3 with the following parameters:

$Q(t) \leftarrow N_q(t)$

$T_b \leftarrow$ A positive integer value

$T_a \leftarrow 0$

$T_d \leftarrow 0$

Note that the value of T_b gives rise to a worst-case minimum rate guarantee.

Let *L* be the forward link air-time required to complete service of a new request

or a head request in waiting queue, and $S=L \times (N_q(t^-)+1)$ be defined as the associated channel resource. With this definition, S can be viewed as an approximate estimate of scheduler latency at time t for the request. Then, *the time-based admission control with number-based busy threshold* is the baseline strategy in Sec 3.3 with the following parameters

$Q(t) \leftarrow N_q(t)$

$T_b \leftarrow$ A positive integer value

$T_a \leftarrow$ A nonnegative integer value

$T_d \leftarrow$ A nonnegative real value

Note that if the blocking threshold is set to $T_d = 0$, the system has no waiting queue and hence does not buffer any request not accepted into service queue.

Also note that if T_a is set to the minimum guaranteed service rate for all admitted users, then it is essentially the rate-based admission control [9] with request queue (i.e. buffering requests).

The flow chart of the algorithm is illustrated in Figure 4. 1 where the busy status is the number of unfinished requests in service queue.

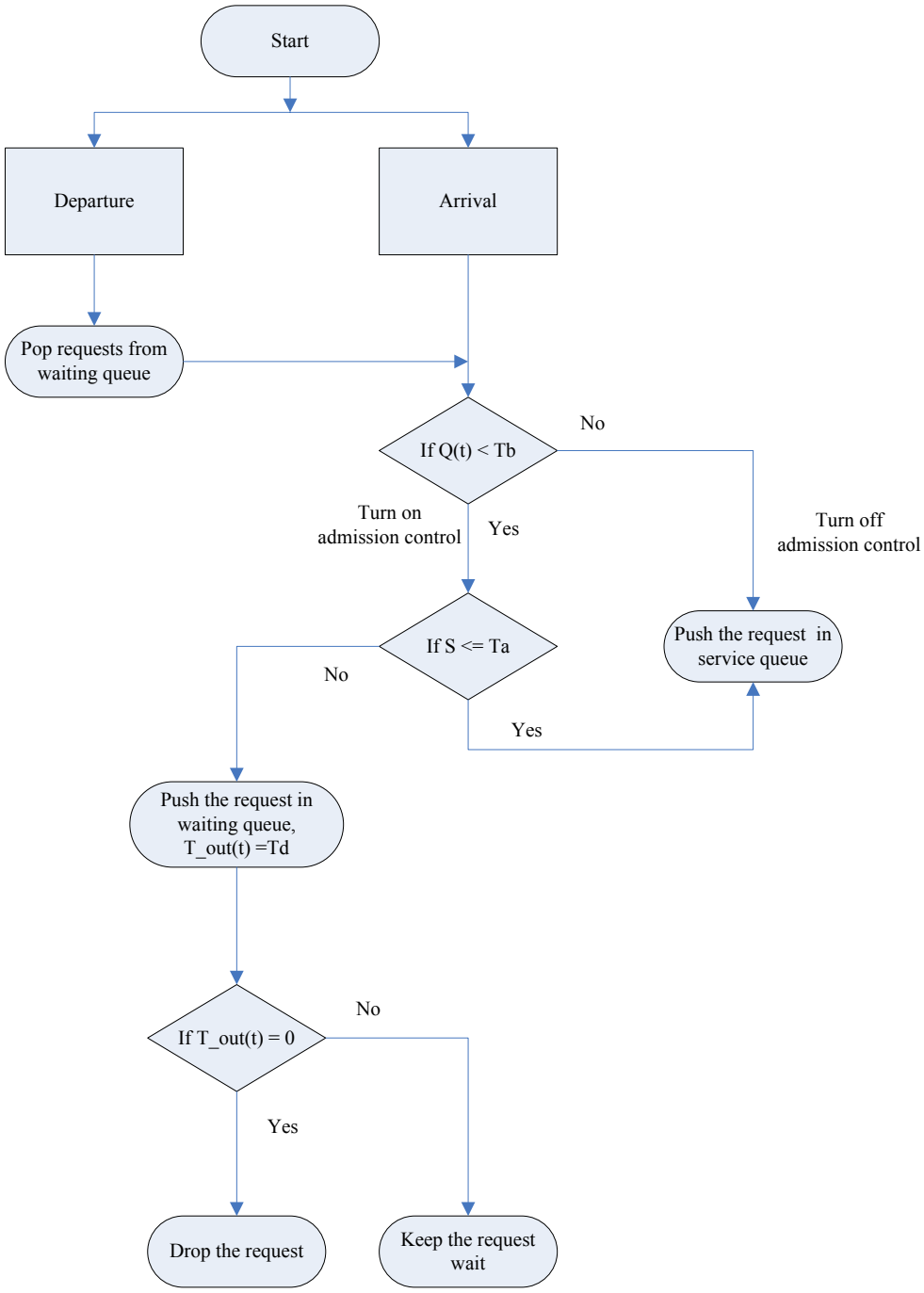

Figure 4. 1 The flow chart of time-based admission algorithm

Additionally, the state transition diagram for the event of a request arrival is illustrated in Figure 4. 2. The system of time-based admission control can be described in simple procedures as depicted in Figs 4.3 (a) to (d), where main procedures are showed in Figure 4.3(a), the algorithm to make admission decision is described in

Figure 4.3(b), round-robin service scheduling is presented in Figure 4.3(c), and the algorithm to determine whether to accept the head of queued requests in waiting queue or to discard it when buffering timer timeouts is illustrated in Figure 4.3(d).

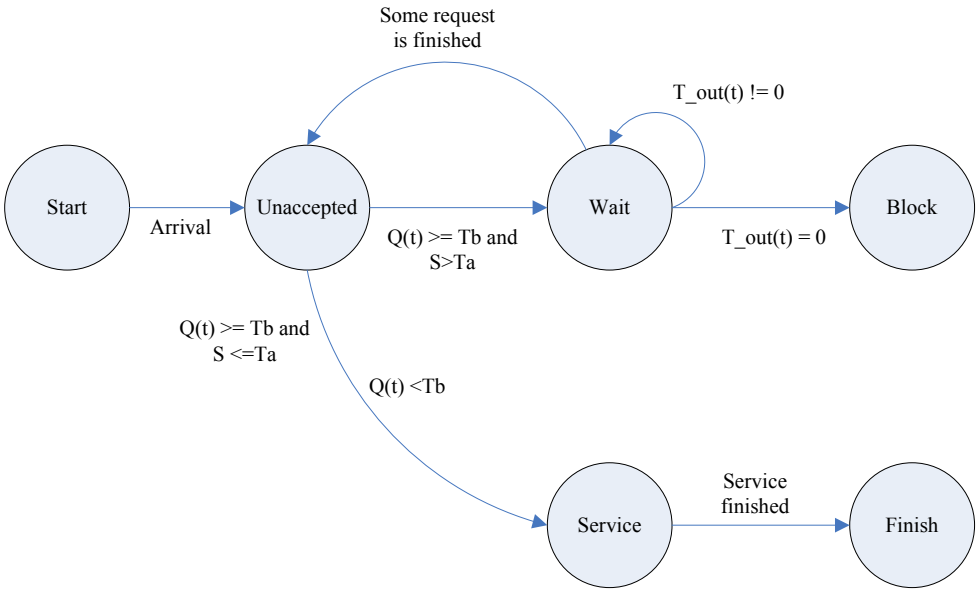

Figure 4. 2 The state transition diagram for a request.

```
New Request N arrives
Decide(N);
Service();
Drop();
```

Figure 4. 3 (a) Main procedures.

```
Decide(R)
Begin
    New Request N arrives
    If the number of requests in system is less than Busy Threshold
        Then accept N
    Else if the length of N is less than Admission Threshold
        Then accept N
    Else push N in waiting queue W
End
```

Figure 4.3 (b) The algorithm to determine admission.

```
Service( )
Begin
    Serve one request in system for one time slot by turns
    If the request is totally served
        Then it is finish
End
```

Figure 4.3 (c) Round-robin service discipline.

```
Drop( )
Begin
    Repeat
        Pop a request R in waiting queue W
        If the waiting time of R is more than Dropping Threshold
            Then block R
        Else if there is a finished request
            Then Decide(R)
        Else push R in waiting queue W
    Until all requests in waiting queue W have been pop in this time slot
End
```

Figure 4.4 (d) The algorithm to determine whether to accept a buffered request or discard the request.

4.2 Adaptive time-based admission control

One of fundamental issues in time-based admission control is what value the admission threshold should be optimally set. In our experience, the system delay latency is a convex function of admission threshold under some given traffic arrival rate and request size. Therefore, there exists an optimal admission threshold regarding minimizing system delay latency. However, traffic arrival rates and the distribution of each request size are practically unknown and tend to be nonstationary. This gives rise to a time-varying optimal admission threshold, instead of the static one in the previous section. To resolve this issue, we propose an adaptive time-based admission control algorithm which autonomously adjusts admission threshold closely to optimal values all the time. Essentially, the adaptive algorithm makes the value of admission threshold *drift along negative gradient direction* at each update of estimate. In the following, we present the approach to adjust the admission threshold.

Let n, n=1, 2, 3, …, be the monitored time slot and ω be a window size for obtaining an estimate of sample means. Then, we have the time sequence, 1, 2, …, ω, ω+1, …, 2ω, 2ω+1, …. The window size ω should be chosen to reflect expected response time when traffic statistics change. Let $I_{\{E\}}[n]$ be an indicator function with value 1 if E is true in time slot n and 0 otherwise. Furthermore, define the average

$$D_a[m-1]=\frac{\left(\sum_{i=1}^{\omega} I_{\{Ed\}}[(m-1)\omega+i]\times D[(m-1)\omega+i]\right)}{\sum_{i=1}^{\omega} I_{\{Ed\}}[(m-1)\omega+i]},\quad m=1,2,\cdots \qquad (4.1)$$

where $\{Ed\}$ represents a service departure event and D is system delay time for the service departure.

The adaptive algorithm for admission threshold:

1. Let $T_a[0]$ be a given initial admission threshold.

   Obtain $D_a[0]$, after one window time

2. $T_a[1]=T_a[0]+\delta$, where δ is a given constant.

   Obtain $D_a[1]$, after another window time

3. Update the value of admission threshold after each window time,

   according to the following recursive formula, for m>0,

$$\begin{cases} g[m]=\dfrac{D_a[m]-D_a[m-1]}{T_a[m]-T_a[m-1]} \\ T_a[m+1]=T_a[m]-\beta\times g[m]\times\left|T_a[m]-T_a[m-1]\right| \end{cases} \qquad (4.2)$$

where β is a positive constant.

Remarks: We set $D_a[m]=D_a[m-1]$ if no departure event occurs in m^{th} time window. Besides, we set ω=100000 time slots, δ=5 time slots, and β=5 in simulations. T_a takes a continuous value, in the unit of time slot.

4.3 Arrival traffics in simulations

In simulations, traffic arrival is Poisson with rate λ. Each arrival is a service request with data size geometrically distributed. The location of each request is uniformly distributed in the coverage area of a BST. We consider a single cell and four discrete feasible rates, as described in Sec 3.2. They are based on the path loss exponent 4, which characterizes the radio environment.

To study the behaviors of proposed time-based admission control, we consider both stationary and nonstationary traffics, as well as nonhomogeneous traffics:

- In stationary traffic model, service requests arrive according to the Poisson process with constant rate λ.
- In nonstationary traffic model, service requests arrive according to the Poisson process with rates alternating between 0.9λ for 0.9 PT and 1.9λ for 0.1 PT, where PT is the length of a time period.
- In nonhomogeneous traffic model, service requests arrive according to the Poisson process with rate alternating between different values but constrained to a fixed loading; That is, fix $\lambda E[\sigma]$ while vary λ periodically.