TWO-SERVER QUEUE SIMULATION

Two Servers in Series

- Assumptions: homogeneous \( \lambda \) Poisson arrivals;
  service at server 1, then by server 2 service for each customer;
  service times are RVs with distributions \( G_1 \) and \( G_2 \);
  no customers after final arrival time \( T \).
- Examples: airline checkin, doctor’s office, restaurant.
- Variables: time \( t \); counters \( N_A, N_D \);
  system state \( (n_1, n_2) \) = customer #s at server 1,2;
  output \( (A_1(i), A_2(i), D(i)) \) customer \( i \)
    arrival (for each server) and departure times;
  next events at \( t_A \), or \( t_1 \), or \( t_2 \), next arrival and completion times.
TWO SERVERS in SERIES CONTINUED

- Two Servers in Series Simulation Algorithm:
  Initialize: $t = N_A = N_D = 0$, $(n_1, n_2) = (0, 0)$;
  generate $T_0$, set $t_A = T_0$, $t_1 = t_2 = \infty$.

Update the system state using the following cases:

1. $t_A = \min(t_A, t_1, t_2)$ (a new arrival at time $t_A$)
   a) reset $t = t_A$, $N_A = N_A + 1$, $n_1 = n_1 + 1$;
   b) generate $T_t$ and reset $t_A = T_t$ (next arrival time);
   c) if $n_1 = 1$ generate $Y_1 \sim G_1$ and reset $t_1 = t + Y_1$;
   d) collect output data $A_1(N_A) = t$.

2. $t_1 = \min(t_A, t_1, t_2)$ (a departure from 1 at time $t_1$)
   a) reset $t = t_1$, $n_1 = n_1 - 1$, $n_2 = n_2 + 1$;
   b) if $n_1 > 0$ generate $Y_1 \sim G_1$ and reset $t_1 = t + Y_1$; else set $t_1 = \infty$ (Q1 empty);
   c) if $n_2 = 1$ generate $Y_2 \sim G_2$ and reset $t_2 = t + Y_2$;
   d) collect output data $A_2(N_A - n_1) = t$.

3. $t_2 = \min(t_A, t_1, t_2)$ (a departure from 2 at time $t_2$)
   a) reset $t = t_2$, $n_2 = n_2 - 1$, $N_D = N_D + 1$;
   b) if $n_2 > 0$ generate $Y_2 \sim G_2$ and reset $t_2 = t + Y_2$; else set $t_2 = \infty$ (Q2 empty);
   c) collect output data $D(N_D) = t$.

Cases 1, 2, and 3 are used until $t_A > T$; then cases 2 and 3 are used until $n_1 = 0$;
then case 3 is used until $n_2 = 0$; then $T_p = \max(t - T, 0)$. 
• End of “run” results
  – Times \((A_1(1), A_2(1), D(1)), \ldots, (A_1(N_A), A_2(N_A), D(N_A))\)
    provide \((A_2 - A_1), (D - A_2), (D - A_1)\) averages.
  – Time \(T_p\) is server overtime.
  – \((\text{Event}, \text{Time})\) data \((n_1(t_i), n_2(t_i), t_i)\) provides history.

• Averages over many runs give expected total customer time, server overtime, and other statistics.

• Generalizations possible to several servers in series.

• Example Matlab Function for Two Servers in Series

  \begin{verbatim}
  function [A1 A2 D Tp Ev] = sriesv(T, lam, l1, l2)
  \end{verbatim}

  Parameter \textbf{lam} for homogeneous Poisson arrivals;
  parameters \textbf{l1}, \textbf{l2} for exponential service times.
function [A1 A2 D Tp Ev] = sriesv( T, lam, l1, l2 )
% Two Servers in Series Q Simulation for total time T;
%   input parameter lam for exponential arrivals
%   input parameters l1, l2 for server time distributions
%   output arrays A1, A2, D for server arrival, departure times
%   output Tp for overtime and Ev for event list

    t = 0; na = 0; nd = 0; n1 = 0; n2 = 0; j = 0;
    ta = -log(rand)/lam; t1 = inf; t2 = inf;
    while ta <= T % time left
        if ta <= min( t1, t2 ) % new arrival
            t = ta; n1 = n1 + 1; na = na + 1;
            ta = t - log(rand)/lam; A1(na) = t;
            if n1 == 1, t1 = t + G(l1); end
        elseif t1 <= t2 % departure from Q1
            t = t1; n1 = n1 - 1; n2 = n2 + 1; A2(na-n1) = t;
            if n1 > 0, t1 = t + G(l1); else, t1 = inf; end
            if n2 == 1, t2 = t + G(l2); end
        else % departure from Q2
            t = t2; n2 = n2 - 1; nd = nd + 1; D(nd) = t;
            if n2 > 0, t2 = t + G(l2); else, t2 = inf; end
        end, j = j + 1; Ev(j,:) = [ n1 n2 t ];
    end % no more arrivals
% sriesv continued
%

while n1 > 0 % empty Q1
    if t1 <= t2, t = t1; n1 = n1 - 1; n2 = n2 + 1;
        if n1 > 0, t1 = t + G(l1); else, t1 = inf; end
        if n2 == 1, t2 = t + G(l2); end, A2(na-n1) = t;
    else, t = t2; n2 = n2 - 1; nd = nd + 1; D(nd) = t;
        if n2 > 0, t2 = t + G(l2); else, t2 = inf; end
    end, j = j + 1; Ev(j,:) = [ n1 n2 t ];
end % Q1 is empty
while n2 > 0 % empty Q2
    t = t2; n2 = n2 - 1; nd = nd + 1; D(nd) = t;
    if n2 > 0, t2 = t + G(l2); else, t2 = inf; end
    j = j + 1; Ev(j,:) = [ n1 n2 t ];
end % Q2 is empty, find Tp
Tp = max(t-T,0);
% end sriesv
function Y = G(a), Y = -log(rand)/a; % Exponential(a) RV
Two servers in series

Sample runs:

```matlab
[A1, A2, D, Tp, Ev] = sriesv(.5, 6, 5, 4); disp([A1; A2; D]), disp(Ev)
```

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>0.0079643</td>
<td>0.031342</td>
<td>0.29468</td>
<td></td>
</tr>
<tr>
<td>A2</td>
<td>0.021433</td>
<td>0.083831</td>
<td>0.56838</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>0.035904</td>
<td>0.90509</td>
<td>0.91373</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>n1</th>
<th>n2</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0.0079643</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.021433</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.031342</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0.035904</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.083831</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0.29468</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>0.56838</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0.90509</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0.91373</td>
</tr>
</tbody>
</table>

for i = 1:1000

```matlab
[A1, A2, D, OT(i), EV] = sriesv(9, 6, 5, 4); S(i) = mean(D-A1);
end, disp([mean(S), mean(OT)]) ...
```

```matlab
3.4929 5.8376
```

for i = 1:1000

```matlab
[A1, A2, D, OT(i), EV] = sriesv(9, 6, 6, 7); S(i) = mean(D-A1);
end, disp([mean(S), mean(OT)]) ...
```

```matlab
1.4232 1.8973
```
TWO SERVERs in PARALLEL

Two Servers in Parallel

- Assumptions: homogeneous or nonhomogeneous $\lambda(t)$ Poisson arrivals; arrivals form single queue, service at server 1 or 2 as available; service times are RVs with distribution $G_1$ and $G_2$; no customers after final arrival time $T$.

- Examples: airline checkin, doctor’s office, restaurant.

- Variables: time $t$; counters $N_A$, $N_D$, $(C_1, C_2)$ #s of customers served by servers 1, 2; system state $SS = (n, i_1, i_2)$, $n$ customers in system, customer $i_j$ at server $j$; output $(A(i), D(i))$ customer $i$ arrival and departure times; next events at $t_A$ or $t_1$ or $t_2$, next arrival and completion times.

Note: if $n$ customers are in the system, and $j = \max(i_1, i_2)$, the customer at the head of the queue has number $j + 1$. 
• Two Servers in Parallel Simulation Algorithm:
  Initialize: $t = N_A = N_D = C_1 = C_2 = 0,$
  $$\text{SS} = (n, i_1, i_2) = (0, 0, 0);$$
  generate $T_0,$ set $t_A = T_0,$ $t_1 = t_2 = \infty.$

  Update the system state using the following cases:

  1. $t_A = \min(t_A, t_1, t_2)$ (a new arrival at time $t_A$)
     reset $t = t_A, N_A = N_A + 1;$
     generate $T_t$ and reset $t_A = T_t$ (next arrival time);
     collect output data $A(N_A) = t.$
     if $\text{SS} = (0, 0, 0),$ reset $\text{SS} = (1, N_A, 0)$ (use server 1),
        generate $Y_1 \sim G_1$ and reset $t_1 = t + Y_1;$$
     else if $\text{SS} = (1, i_1, 0),$ reset $\text{SS} = (2, i_1, N_A)$ (use server 2),
        generate $Y_2 \sim G_2$ and reset $t_2 = t + Y_2;$$
     else if $\text{SS} = (1, 0, i_2),$ reset $\text{SS} = (2, N_A, i_2)$ (use server 1),
        generate $Y_1 \sim G_1$ and reset $t_1 = t + Y_1;$$
     else reset $\text{SS} = (n + 1, i_1, i_2).$
TWO SERVERs in PARALLEL CONTINUED

Two Servers Parallel Simulation Algorithm Cases 2 and 3

2. \( t_1 < t_A, t_1 \leq t_2 \) (a departure from 1 at time \( t_1 \))
   reset \( t = t_1, C_1 = C_1 + 1 \); collect output data \( D(i_1) = t \);
   if \( n = 1 \) reset \( SS = (0, 0, 0), t_1 = \infty \);
   else if \( n = 2 \) reset \( SS = (1, 0, i_2), t_1 = \infty \);
   else reset \( SS = (n - 1, \max(i_1, i_2) + 1, i_2) \),
   and generate \( Y_1 \sim G_1 \) and reset \( t_1 = t + Y_1 \).

3. \( t_2 < t_A, t_2 < t_1 \) (a departure from 2 at time \( t_2 \))
   reset \( t = t_2, C_2 = C_2 + 1 \); collect output data \( D(i_2) = t \);
   if \( n = 1 \) reset \( SS = (0, 0, 0), t_2 = \infty \);
   else if \( n = 2 \) reset \( SS = (1, i_1, 0), t_2 = \infty \);
   else reset \( SS = (n - 1, i_1, \max(i_1, i_2) + 1) \),
   and generate \( Y_2 \sim G_2 \) and reset \( t_2 = t + Y_2 \).

Cases 1, 2, and 3 are used until \( t_A > T \).
Then cases 2 and 3 are used until \( n = 0 \),
Then \( T_p = \max(t - T, 0) \).
• End of “run” results
  – Times \((A(1), D(1)), \ldots, (A(N_A), D(N_A))\);
    \((D - A)\) average provides average service time.
  – Server overtime \(T_p\), customers served \(C_1, C_2\).
  – (Event,Time) data \((n(t_j), i_1(t_j), i_2(t_j), t_j)\) provides history.

• Averages over many runs give expected total customer time,
  server overtime, and other statistics.

Some Matlab Test results where input \(N\) controls total \# of customers:

\[
\begin{array}{l}
\begin{array}{c}
[A \ D \ C1 \ C2] = prllsv(1000, 6, 4, 3);
disp([mean(D-A) \ C1/(C1+C2)])
\end{array}\\
\begin{array}{cc}
0.81528 & 0.566
\end{array}
\end{array}
\]

\[
\begin{array}{l}
\begin{array}{c}
[A \ D \ C1 \ C2] = prllsv(1000, 6, 4, 3);
disp([mean(D-A) \ C1/(C1+C2)])
\end{array}\\
\begin{array}{cc}
0.945 & 0.602
\end{array}
\end{array}
\]

• Generalizations to several servers in parallel and combinations with series servers.