CS 672
System Level Performance Models of Computer Systems

Dr. Daniel A. Menascé
http://www.cs.gmu.edu/faculty/menasce.html
Department of Computer Science
George Mason University

Part V: Learning Objectives

Characterize system-level models
Present State Transition Diagram (STD) technique
Show general solution to STDs
Show how to obtain performance metrics from the solution of STDs

System-level Models

- System is seen as a black box.
- Only its input-output characteristics are considered.
- Inputs: arrivals of requests
- Output: throughput.
System-level Example

- A Web server receives 10 requests/sec.
- The maximum number of requests in the server is 3.
- Requests that arrive and find three requests being processed are rejected.

System-level Example

- The measured throughput as a function of the number of requests is:

<table>
<thead>
<tr>
<th>Number of requests</th>
<th>Throughput (req/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
</tr>
</tbody>
</table>
System-level Example: a few questions

- Q1: What is the probability that an incoming request is rejected?
- Q2: What is the average number of requests in execution?
- Q3: What is the average throughput of the Web server?
- Q4: What is the average time spent by an HTTP request in the Web server?

System-level Example

- Characterize the Web server by its state, i.e., the number $k$ of requests in the Web server.
- Assumptions made:
  - homogeneous workload: all requests are equivalent
  - memoryless: how the system arrived at system $k$ does not matter.
  - operational equilibrium: no. requests at beginning of interval = no. request at the end.
System-level Example

Assume we are able to find the values of:
– $P_k$ = probability that there are $k$ requests in the Web server.

• Question: can we answer all the questions posed before as a function of the $P_k$’s?
System-level Example: a few questions

• Q1: What is the probability that an incoming request is rejected?
• A: It is the probability that an arriving HTTP request finds 3 requests already being processed. The answer is then $P_3$. 

System-level Example: a few questions

• Q2: What is the average number of requests in execution?
• A: using the definition of average:

\[ n_{\text{req}} = 0 \times P_0 + 1 \times P_1 + 2 \times P_2 + 3 \times P_3 \]
System-level Example: a few questions

- Q3: What is the average throughput of the Web server?
- A: again, using the definition of average:

\[ X = 0 \times P_0 + 12 \times P_1 + 15 \times P_2 + 16 \times P_3 \]

throughput value at each state
System-level Example: a few questions

- Q4: What is the average time spent by an HTTP request in the Web server?
- A: It will be a function of the average number of requests, $n_{req}$, and the average throughput $X$. More on this later...

System-level Example: computing the $P_k$’s

- use the flow in = flow out principle: the flow into a set of states is equal to the flow out of this set of states in equilibrium.
System-level Example: computing the $P_k$'s

$flow\ in = flow\ out$

$12 \times P_1 = 10 \times P_0$

System-level Example: computing the $P_k$’s

Putting it all together:

\[ 12 \times P_1 = 10 \times P_0 \Rightarrow P_1 = \frac{10}{12} P_0 \]
\[ 15 \times P_2 = 10 \times P_1 \Rightarrow P_2 = \frac{10}{15} P_1 = \frac{10 \times 10 P_0}{15 \times 12} \]
\[ 16 \times P_3 = 10 \times P_2 \Rightarrow P_3 = \frac{10}{16} P_2 = \frac{10 \times 10 \times 10 P_0}{16 \times 15 \times 12} \]
System-level Example: computing the $P_k$’s

- Putting it all together:
  \[ P_1 = \frac{10}{12} P_0; \quad P_2 = \frac{10 \times 10}{15 \times 12} P_0; \quad \text{and} \]
  \[ P_3 = \frac{10 \times 10 \times 10}{16 \times 15 \times 12} P_0 \]

- But, the Web server has to be in one of the four states at any time. So,
  \[ P_0 + P_1 + P_2 + P_3 = 1. \]

System-level Example: computing the $P_k$’s

- Solving for $P_0$ and then for the other $P_k$’s we get:

<table>
<thead>
<tr>
<th>$k$</th>
<th>$P_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.365</td>
</tr>
<tr>
<td>1</td>
<td>0.305</td>
</tr>
<tr>
<td>2</td>
<td>0.203</td>
</tr>
<tr>
<td>3</td>
<td>0.127</td>
</tr>
</tbody>
</table>
System-level Example: answering the questions

• Q1: What is the probability that an incoming request is rejected?
  A: It is the probability that an arriving HTTP request finds 3 requests already being processed. The answer is then
  \[ P_3 = 0.127 = 12.7\% . \]


System-level Example: answering the questions

• Q2: What is the average number of requests in execution?
  A: using the definition of average:

\[
    n_{\text{req}} = 0 \times 0.365 + 1 \times 0.305 + 2 \times 0.203 + 3 \times 0.127 \\
    = 1.091 \text{ requests}
\]
System-level Example: answering the questions

- Q3: What is the average throughput of the Web server?
- A: again, using the definition of average:

\[
X = 0 \times 0.365 + 12 \times 0.305 + 15 \times 0.203 + 16 \times 0.127 \\
= 8.731 \text{ requests/sec.}
\]


System-level Example: answering the questions

- Q4: What is the average time spent by an HTTP request in the Web server?
- A: It is a function of the average number of requests, \( n_{\text{req}} \), and the average throughput \( X \). We need Little’s Law to answer this question.

Little’s Law

\[
\text{avg. number requests in the server} = \frac{\text{avg. departure rate from the server}}{\text{avg. time spent at the server}} \times \text{avg. time spent at the server}
\]

\[
= \frac{1.091 \text{ req}}{8.731 \text{ req/sec}} \times 8.731 \text{ sec}
\]

System-level Example:
answering the questions

- Q4: What is the average time spent by an HTTP request in the Web server?
- A: From Little’s Law,

\[
R = \frac{n_{\text{req}}}{X} = \frac{1.091}{8.731} = 0.125 \text{ sec.}
\]
Practice Drill
Using Models for Decision Making

• What happens if the maximum number of allowed TCP connections changes from 3 to 10?
• What if the load on the server doubles?
• What is the impact of a threefold increase in the server’s capacity?

Types of System-level Models

• Population Size:
  – infinite
  – finite
• Service Rate:
  – fixed
  – variable
• Maximum Queue Size:
  – unlimited
  – limited
Types of System-level Models (population size)

- Infinite Population: the number of clients is very large. The rate at which requests arrive to the system does not depend on the number of requests in the system.
  - e.g., requests arriving from the Internet to a public Web server.

Types of System-level Models (infinite population)

arrival rate (requests/sec)  

SERVER  

completed requests
Infinite Population/Infinite Queue

\[ \text{flow in} = \text{flow out} \]

\[ \lambda P_0 = \mu P_1 \]
Infinite Population/Infinite Queue

\[
\begin{align*}
\lambda P_0 &= \mu P_1 \\
\lambda P_1 &= \mu P_2 \\
\lambda P_{k-1} &= \mu P_k
\end{align*}
\]

flow out = flow in

Infinite Population/Infinite Queue Example

- A DB server receives 30 req/sec. Each request takes 0.02 sec on the average. Find:
  - Fraction of requests in the DB server?
  - Average response time.
  - Average response time for a server twice as fast.
  - Average response time for a server twice as fast for twice the arrival rate.

Types of System-level Models
(maximum queue size)

- Unlimited Queue Size: all arriving requests are queued for service. No requests are rejected!
- Limited Queue Size: requests that find more than W requests waiting for service are rejected.
Infinite Population/Finite Queue

- arriving requests that find the server in state W are lost.

Infinite Population/Finite Queue Example

- A DB server receives 30 req/sec. Each request takes 0.02 sec on the average. At most 4 request can be queued. Find:
  - Fraction of requests in the DB server?
  - Average response time.
  - Average response time for a server twice as fast.
  - Average response time for a server twice as fast for twice the arrival rate.
Infinite Population/Finite Queue Example (cont’d)

- What is the maximum value for the maximum number of requests queued so that less than 1% of the requests are rejected?

Generalized System-level Models

\[\begin{align*}
\lambda_0 & \quad \lambda_1 & \quad \lambda_2 & \quad \lambda_3 \\
\mu_1 & \quad \mu_2 & \quad \mu_3 & \quad \mu_4
\end{align*}\]

*Generalized System-level Models can be solved using the flow in = flow out principle!*

Generalized System-level Models

\[ p_k = \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \]

\[ p_0 = \left[ \sum_{k=0}^{\infty} \prod_{i=0}^{k-1} \frac{\lambda_i}{\mu_{i+1}} \right]^{-1} \]

Types of System-level Models (population size)

- Finite Population: the number of clients is limited. The rate at which requests arrive to the system depends on how many have already arrived.
  - e.g., requests arriving to an intranet Web server from a known number of clients within the organization.
Types of System-level Models
(finite population)

- Fixed Service Rate: the throughput does not vary with the number of requests being processed.
Types of System-level Models (service rate)

- Variable Service Rate: the throughput depends on the number of requests being processed.

![Throughput (requests/sec) vs. requests in the system]

Types of System-level Models

<table>
<thead>
<tr>
<th>Population</th>
<th>Service Rate</th>
<th>Queue Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>infinite</td>
<td>fixed</td>
<td>unlimited</td>
</tr>
<tr>
<td>infinite</td>
<td>fixed</td>
<td>limited</td>
</tr>
<tr>
<td>infinite</td>
<td>variable</td>
<td>unlimited</td>
</tr>
<tr>
<td>infinite</td>
<td>variable</td>
<td>finite</td>
</tr>
<tr>
<td>finite</td>
<td>fixed</td>
<td></td>
</tr>
<tr>
<td>finite</td>
<td>variable</td>
<td></td>
</tr>
</tbody>
</table>

System-level Models

Example

A Web server receives 30 requests/sec. Its throughput function is given below. The server queue is limited to five requests. What is the server utilization, avg. throughput, avg. no. requests, avg. response time, and fraction of lost requests?

<table>
<thead>
<tr>
<th>No. of requests</th>
<th>Throughput (req/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>35</td>
</tr>
<tr>
<td>3 or more</td>
<td>50</td>
</tr>
</tbody>
</table>

From Little’s Law,

\[
\text{avg. response time} = \frac{\text{avg. no requests}}{\text{avg. throughput}} = \frac{1.85}{28.4} = 0.065 \text{ sec.}
\]
System-level Models
Example (cont’d)

<table>
<thead>
<tr>
<th>Max. Queue Size</th>
<th>3</th>
<th>5</th>
<th>10</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Number of Requests</td>
<td>1.433</td>
<td>1.850</td>
<td>2.212</td>
<td>2.264</td>
</tr>
<tr>
<td>Server Utilization</td>
<td>79.81%</td>
<td>82.7%</td>
<td>83.9%</td>
<td>83.96%</td>
</tr>
<tr>
<td>Avg. Server Throughput (req/sec)</td>
<td>24.8</td>
<td>28.4</td>
<td>29.9</td>
<td>30</td>
</tr>
<tr>
<td>Average Response Time (sec)</td>
<td>0.058</td>
<td>0.065</td>
<td>0.074</td>
<td>0.075</td>
</tr>
<tr>
<td>Fraction of Lost Requests</td>
<td>0.173077</td>
<td>0.05343</td>
<td>0.003869</td>
<td>0.000299</td>
</tr>
</tbody>
</table>

System-level Models
Example (cont’d)

Fraction of Lost Requests vs. Max Queue Size

Summary

System-level models view a server as a black box. Only its arrival process and throughput functions are relevant.

State Transition Diagrams (STDs) can be used to find the probability that $k$ requests are in the server. Use the $flow\ in = flow\ out$ principle.

Little’s Law can be used to compute the response time from the average number of requests and from the throughput.