Errata in Chapter 6

Page 335, after solution to problem 54
(changes marked in RED).

Before proceeding onto the next example it is worthwhile to comment on some practical considerations regarding the conclusion of the above problem. In many situations one is faced with this type of a decision. For example, consider fast food restaurants such as those making submarine sandwiches or burritos.

Page 346, Section 6.3.2 ASTA
(changes marked in RED).

Consider a closed Jackson network (with all the usual notation) in steady state. Let $\pi_j(x)$ be the probability that an arriving customer into node j in steady state will see x_k customers in node k for all k so that x is a vector of those x_k values. Notice that $x_1 + x_2 + \ldots + x_N = C - 1$ since we are not counting the arriving customer into node j in the state description.

Page 353, Section 6.3.3 Single-Server Closed Jackson Networks
(changes marked in RED).

Although for the two closed Jackson network examples thus far, ~~the~~ computing $G(C)$ was not particularly difficult, as the number of nodes and customers increase, this would become a very tedious task. There are a few algorithms to facilitate that computation, one of which is provided in the exercises at the end of the chapter. Nonetheless, there is one case where it is possible to circumvent computing $G(C)$. That case is a closed Jackson network where there is a single server at all the nodes. Assume that for all i, there is a single server at node i with service rate μ_i operating FCFS discipline at every node. Then the mean performance measures can be computed without going through the computation of the normalizing constant $G(C)$. For that we first need some notation (for anything not defined here the reader is encouraged to refer to the closed Jackson network notation). Define the following steady-state measures for all $i = 1, \ldots, N$ and $k = 0, \ldots, C$:

- $W_i(k)$: Average sojourn time in node i when there are k customers (as opposed to C) in the closed Jackson network;
- $L_i(k)$: Average number in node i when there are k customers (as opposed to C) in the closed Jackson network;
- $\lambda_i(k)$: Measure of average flow (sometimes also referred to as throughput) across node i ~~in the closed Jackson network~~ when there are k customers (as opposed to C) in the network.

We do not have an expression for any of the above and the objective is to obtain them iteratively. However, before describing the iterative algorithm, we first explain the relationship between those parameters.

On the basis of the arrival theorem described in Remark 14, in a network with k customers (such that $1 \leq k \leq C$) the expected number of customers that an arrival to node i (for any $i \in \{1, \ldots, N\}$) would see is $L_i(k-1)$. Note that $L_i(k-1)$ is the steady state expected number of customers in node i when there are $k-1$ customers in the system. Thereby, the net mean sojourn time experienced by that arriving customer in steady state is the average time to serve all those in the system upon arrival plus that of the customer. Since the average service time is $1/\mu_i$, we have

$$W_i(k) = \frac{1}{\mu_i}[1 + L_i(k-1)].$$

Let a be the solution to $a = aP$ as usual. ~~with the only exception that the a_j values sum to one here. Thus the a_j values describe the fraction of visits that are made into node j. The aggregate sojourn time weighted across the network using the fraction of visits is given by $\sum_{i=1}^{N} a_i W_i(k)$ when there are k customers in the network. One can think of an aggregate sojourn time as the sojourn time for a customer about to enter a node. Hence by conditioning on the node of entry as i (which happens with probability a_i) where the mean sojourn time is $W_i(k)$, we can get the result $\sum_{i=1}^{N} a_i W_i(k)$. Thereby we derive the average flow in the network using Little' law across the entire network as $\lambda(k) = \frac{k}{\sum_{i=1}^{N} a_i W_i(k)}$ when there are k customers in the network. Essentially, $\lambda(k)$ is the average rate at which service completion occurs in the entire network, taken as a whole. Thereby applying Little's law across each node i we get $L_i(k) = \lambda(k) W_i(k) a_i$ when there are k customers in the network.~~ Since a_i, the i^{th} element of a is not unique, we can think of the throughput at node i to be

$$\lambda_i(k) = \nu(k) a_i,$$

where $\nu(k)$ is a constant as $\lambda_i(k)$ also satisfies the flow balance. From Little's law we have $L_j(k) = \lambda_j(k) W_j(k)$, summing over all j we get (using the fact that $\sum_{j=1}^{k} L_j(k) = k$)

$$k = \sum_{j=1}^{k} \lambda_j(k) W_j(k) = \nu(k) \sum_{j=1}^{k} a_j W_j(k).$$

Thus we have

$$\nu(k) = \frac{k}{\sum_{j=1}^{k} a_j W_j(k)} \quad \text{and}$$

$$\lambda_i(k) = \frac{k a_i}{\sum_{j=1}^{N} a_j W_j(k)}.$$

Using the above results we can develop an algorithm to determine $L_i(C)$, $W_i(C)$ and $\lambda(C)$ defined above. The input to the algorithm are N, C, P and μ_i for all $i \in \{1, \ldots, N\}$. For the algorithm, initialize $L_i(0) = 0$ for $1 \leq i \leq N$ and obtain the a_i values for all $i \in \{1, \ldots, N\}$. Then for $k = 1$ to C, iteratively compute for each i (such that $1 \leq i \leq N$):

$$\begin{aligned} W_i(k) &= \frac{1}{\mu_i}[1 + L_i(k-1)], \\ \lambda_i(k) &= \frac{k a_i}{\sum_{j=1}^{N} a_j W_j(k)}, \\ L_i(k) &= \lambda_i(k) W_i(k). \end{aligned}$$

Page 356, Solution to Problem 59
(changes marked in RED).

We can obtain $a = [a_1\ a_2\ a_3\ a_4\ a_5]$ by solving for $a = aP$ and $a_1 + a_2 + a_3 + a_4 + a_5 = 1$ to get a particular case of vector a as

$$a = [1/2\ 1/8\ 1/8\ 1/8\ 1/8].$$

We are also given in the problem statement $\mu = [\mu_1\ \mu_2\ \mu_3\ \mu_4\ \mu_5] = [1/6\ 1/17\ 1/28\ 1/13\ 1/23]$.

By initializing $L_i(0) = 0$ for $i = 1, 2, 3, 4, 5$ we can go through the algorithm of iteratively computing for $k = 1$ to C (and all i such that $1 \leq i \leq 5$) using the following steps: $W_i(k) = \frac{1}{\mu_i}[1 + L_i(k-1)]$; $\lambda_i(k) = \frac{ka_i}{\sum_{j=1}^{N} a_j W_j(k)}$; and $L_i(k) = \lambda_i(k)W_i(k)$. These computations are tabulated in Table 6.3. From the last row of that table notice that with $C = 16$, the expected number of connections at nodes 1, 2, 3, 4 and 5 are 3.6023, 1.327, 7.208, 0.7815 and 3.0812 respectively. Also the throughput of the database-server system is indeed $\lambda_1(16)$ which is 0.1360 transactions per millisecond. If we were to increase C to 25, 50, 75 and 100, the corresponding values are described in Table 6.4. Notice in the table that the throughput $\lambda_1(C)$ has practically leveled off and increasing C is mostly contributing only to longer queues in node 3 (i.e. disk 2) which is the bottleneck in this system (and the throughput tends toward μ_2, the processing rate of disk 2). Also notice how all the other nodes have not only scaled very well but the contribution to the overall number in the system is becoming negligible. This will be the basis of bottleneck-based approximations that we will consider in the next chapter.

k	$L_1(k)$	$L_2(k)$	$L_3(k)$	$L_4(k)$	$L_5(k)$	$\lambda_1(k)$
1	0.2286	0.1619	0.2667	0.1238	0.2190	0.0381
2	0.4631	0.3102	0.5570	0.2294	0.4403	0.0628
3	0.7018	0.4452	0.8714	0.3195	0.6621	0.0800
4	0.9433	0.5674	1.2102	0.3962	0.8829	0.0924
5	1.1860	0.6776	1.5737	0.4615	1.1013	0.1017
6	1.4284	0.7765	1.9620	0.5173	1.3158	0.1089
7	1.6692	0.8650	2.3754	0.5649	1.5255	0.1146
8	1.9073	0.9439	2.8138	0.6057	1.7294	0.1191
9	2.1414	1.0142	3.2772	0.6406	1.9266	0.1228
10	2.3706	1.0766	3.7657	0.6706	2.1165	0.1258
11	2.5940	1.1321	4.2790	0.6964	2.2985	0.1283
12	2.8109	1.1812	4.8169	0.7187	2.4723	0.1304
13	3.0207	1.2246	5.3792	0.7379	2.6376	0.1321
14	3.2228	1.2631	5.9654	0.7546	2.7942	0.1336
15	3.4167	1.2970	6.5752	0.7690	2.9421	0.1349
16	3.6023	1.3270	7.2080	0.7815	3.0812	0.1360

C	L_1	L_2	L_3	L_4	L_5	$\lambda_1(C)$
25	4.8794	1.4786	13.8299	0.8418	3.9704	0.1409
50	5.9284	1.5435	37.0910	0.8660	4.5711	0.1428
75	5.9972	1.5454	61.9915	0.8667	4.5992	0.1429
100	5.9999	1.5455	86.9880	0.8667	4.6000	0.1429

Page 374, Exercise 6.10
(changes marked in RED).

Compute the in-process inventory of the number of products in the system. Assume infinite waiting area and exponential service times.

Page 374, Exercise 6.11 (b)
(changes marked in RED).

Consider a closed queueing network with two identical stations and C customers in total. The service times at both stations are according to exp(μ) distribution. Each customer at the end of service joins either stations with equal probability. Let p_i be the steady state probability there are i customers in one of the queues (and $C - i$ in the other). Is the following TRUE or FALSE?

$$p_i = \binom{C}{i} \frac{1}{2^C} \quad \text{for } i = 1, 2, \ldots, C$$