Aimuel Emmanuel
16/sci01/005

1) Concept of Operational Laws in Computer Systems- Operational laws are equations which may be used as an abstract representation of a model of the average behavior of any system. They are general and make no assumptions about the behavior of the random variable that characterize the system. Using these abstractions, these laws can be applied to any device and gradually build a more complex system.
2) Operational Laws
a) Little's Law- Little's Law is a theorem that determines the average number of items in a stationary queuing system based on the average waiting time of an item within a system and the average number of items arriving at the system per unit of time. The law provides a simple and intuitive approach for the

assessment of the efficiency of queuing systems. The concept is hugely significant for business operations because it states that the number of items in the queuing systems primarily depends on two key variables, and it is not affected by other factors such as the distribution of the service or service order. Little's Law can only be used in queuing systems. In addition, the theorem can be applied in different fields, from running a small coffee shop to the maintenance of the operations of a military airbase.

Formula for Little's Law

$L=\lambda xW$

L – the average number of items in a queuing system

λ – the average number of items arriving at the system per unit of time W – the average waiting time an item spends in a queuing system

b) Space-Time Product Laws- it states that the throughput is

equal to average amount of memory in use divided by average space-time product. Space-time products are often used to evaluate program performance and assign accounting charges to programs in virtual memory systems. Essentially, a program's space-time product is equal to its execution time multiplied by the average amount of money allocated to it during its execution. Since space-time products, response time, and throughput are all used as indicators of system performance, it is interesting to examine the manner in which these quantities are related.

c) Forced-Flow Law- The Forced-Flow Law (FFL) relates throughputs at individual resources within a system to the overall system throughput. It is the average no of visits that a system level job makes to that resource. The general residence time law is the sum of the product

of its average residence time at each resource and the number of visits it makes to that resource.
The Forced-Flow Law:
$\lambda k = V k\lambda$.
The average arrival rate to resource k is the total system arrival rate times the expected number
of visits made to resource k.
Because of the Conservation Law, we could also state the FFL in terms of output rates, Λ and Λk.

d) Conservation Law- What goes in must (normally) come out.
Consider a system with arrival rate of λ and an output rate of Λ.
If the system is not overloaded and no customers are created or destroyed inside the system, then $\lambda = \Lambda$.
Creating customers in a system is called forking. Destroying customers is called joining. It's possible to model systems with these behaviors, but usually difficult, so we won't see any

examples until the end of class.

e) Utilisation Law- It states that, The utilization of a resource is equal to the product of the throughput of that resource and the average service requirement at that resource. If we know the amount of processing that each job requires at a resource then we can calculate the utilisation of the resource. The total amount of service that a system job generates at the ith resource is called the service demand,

Di : Di = SiVi

The utilisation of a resource, the percentage of time that the ith resource is in use processing to a job, is denoted Ui

Ui = XiSi = XD

f) Interactive Response Time Law- The name of this law dates back to the time when most of the systems which were being modelled were mainframes processing both interactive jobs and batch jobs. The think time, Z,

was quite literally the length of time that a programmer spent thinking at his terminal before submitting another job. More generally interactive systems are those in which jobs spend time in the system not engaged in processing, or waiting for processing: this may be because of interaction with a human user, or may be for some other reason. The think time represents the time between processing being completed and the job becoming available as a request again. Thus the residence time of the job, as calculated by Little's law as the time from arrival to completion, is greater than the system's response time. The interactive response time law reflects this: it calculates the response time, R as follows:

$R = N=X - Z$

The response time in an interactive system is the residence time minus the think time.

Note that if the think time is zero, $Z = 0$ and $R = W$, then the interactive response
time law simply becomes Little's law.

g) General Residence Time Law- One method of computing the mean residence or response time per job in a system is to apply Little's law to the system as a whole. However, if the mean number of jobs in the system, N, or the system level throughput, X, are not known an alternative method can be used. Applying Little's law to the ith resource we see that $N_i = X_i W_i$, where N_i is the mean number of jobs at the resource and W_i is the average response time of the resource. From the forced flow law we know that $X_i = X V_i$. Thus we can deduce that
$N_i/X = V_i W_i$.
The total number jobs in the system is clearly the sum of the number of jobs at each resource, i.e. $N = N_1 + \cdots + N_M$ if there are

M resources in the system. We know from Little's law that W = N/X and from this we arrive at the general residence time, or general response time law:

The average residence time of a job in the system will be the sum of the product of its average residence time at each resource and the number of visits it makes to that resource.

3)

4) Basic queuing models-

Basic Queuing Disciplines-

1. First-in-first-out (FIFO)- this means the oldest inventory items are recorded as sold first but do not necessarily mean that the exact oldest physical object has been tracked and sold. In other words, the cost associated with the inventory that was purchased first is the cost expensed first.
2. Last-in-first-out (LIFO)- this describes a method for accounting for inventories. Under this

system, the last unit added to an inventory is the first to be recorded as sold.
3. Service in random order (SIRO)- Under this type of queue structure, the customer is chosen for service randomly and hence all the customers are equally likely to be selected. Therefore, the time of arrival of the customer has no consequence on the selection of the customer.
4. Shortest processing time first (SPT)- Its principle is to order jobs according to their duration and
schedule them by beginning by the shorters.
5) How to resolve basic queuing problems
6) ?