Operations Research
Vol. 59, No. 3, May–June 2011, pp. 536–549
issn 0030-364X | eissn 1526-5463 | 11 | 5903 | 0536

doi 10.1287/opre.1110.0940
© 2011 INFORMS

OR FORUM

Little's Law as Viewed on Its 50th Anniversary

John D. C. Little
MIT Sloan School of Management, Cambridge, Massachusetts 02139, jlittle@mit.edu

Fifty years ago, the author published a paper in *Operations Research* with the title, "A proof for the queuing formula: $L = \lambda W$" [Little, J. D. C. 1961. A proof for the queuing formula: $L = \lambda W$. *Oper. Res.* **9**(3) 383–387]. Over the years, $L = \lambda W$ has become widely known as "Little's Law." Basically, it is a theorem in queuing theory. It has become well known because of its theoretical and practical importance. We report key developments in both areas with the emphasis on practice. In the latter, we collect new material and search for insights on the use of Little's Law within the fields of operations management and computer architecture.

Subject classifications: Little's Law; queuing theory; operations management; computer engineering; computer architecture; operations research.
Area of review: OR Forum.
History: Received February 2011; revision received April 2011; accepted April 2011.

Participate in the OR Forum discussion at http://orforum.blog.informs.org.

1. Introduction

Compared to other fields, most papers in mathematics are rarely cited. The average number of citations for an article in a mathematics/computer journal in the two years following publication is less than one (Amin and Mabe 2000). Little (1961) is a mathematical paper. Its 50th anniversary provides a good occasion to look back at its history and ask why "Little's Law" has become well known and, more importantly, ask what has been its role in the development of theory and practice in queuing.

Little's Law deals with *queuing systems*. See Figure 1 for a schematic diagram. A queuing system consists of discrete objects we shall call *items*, which arrive at some rate to the system. The items could be cars at a toll booth, people in a cafeteria line, aircraft on a production line, or instructions waiting to be executed inside a computer. The stream of *arrivals* enters the system, joins one or more queues and eventually receives *service,* and exits in a stream of *departures*. The service might be a taxi ride (travelers), a bowl of soup (lunch eaters), or auto repair (car owners). In most cases, service is the bottleneck that creates the queue, and so we usually have a *service operation* with a *service time,* but this is not required. In such a case we assume there is nevertheless a *waiting time.* Sometimes a distinction is made between *number in queue* and *total number in queue plus service*, the latter being called *number in system.*

Little's Law says that the average number of items in a queuing system, denoted L, equals the average arrival rate of items to the system, λ, multiplied by the average waiting time of an item in the system, W. Thus,

$$L = \lambda W.$$

For ease of exposition, we frequently shorten "Little's Law" to LL.

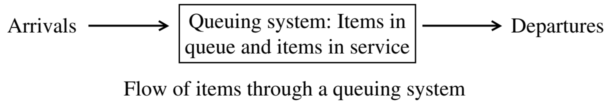
The purpose of the present paper is to summarize major developments in theory and practice over the last 50 years. This is made easier by excellent review papers on the theory by Whitt (1991) and, more recently, Wolff (2011). We shall particularly rely on them for guidance in identifying the most significant developments in queuing theory.

Practice is a different story. By practice we mean problem solving in the "real world," aimed at improving an operation or system for an organization, preferably with measured results. That is what operations research originally tried to do and, of course, many operations researchers continue to do. There is no ready source of written material of this sort for a specialized topic like Little's Law. Nevertheless, the author has collected a few interesting cases to report.

In the course of organizing material for this paper, the author realized that the standard proofs of LL by sample path theory do not address certain issues that regularly occur in practice. Whereas the usual proofs assume that the process runs over a time axis $(0 \leqslant t < \infty)$, practitioners operate over a finite time period $[0, T]$. Furthermore, there are advantages to viewing a queuing system this way.

Accordingly, §2 addresses queuing processes over a finite time period, provides a rationale for doing so, and proves two simple theorems. Each is motivated by an illustrative example from practice. Several corollaries extend the usefulness of the theorems. We discuss their practical implications in a series of remarks. Section 3 discusses the

Figure 1. Schematic diagram of a queuing system.


Arrivals
Queuing system: Items in queue and items in service
Departures
Flow of items through a queuing system

importance of Little's Law to queuing theory and, as mentioned, provides an overview of developments since Little (1961). Section 4 treats the importance of LL to practice. The two biggest fields that make use of LL today are operations management (OM) and computer architecture. At this point, the major OM textbooks devote significant space to explaining and showing how to use LL, complete with pedagogical exercises. In addition, the author has included a few published reports of hands-on experience in the field with all their human and situational issues. Finally, §5 is a note on the author's personal history of involvement with Little's Law.

2. Queuing Processes Over a Finite Time Period

Most sample path proofs of Little's Law (Stidham 1974, Whitt 1991, Wolff 2011) analyze queuing processes over the entire time axis ($0 \leqslant t < \infty$) and show that $L = \lambda W$, subject to the existence and convergence of the three LL parameters, and possibly other conditions.

We argue that proofs of LL for sample paths that are restricted by ($0 \leqslant t \leqslant T$) with T finite are valuable for many applications in the real world. Among the reasons for this are (1) all observations of practice take place in a finite time interval and so we need a way to interpret the numbers obtained; (2) we shall find that the finite time interval guarantees that the relationship $L = \lambda W$ is numerically exact; and (3) models in which the values of $\{L, \lambda, W\}$ are fixed for ($0 \leqslant t < \infty$) do not speak to applications in which, for example, an objective is to be able to control the departure rate in real time.

First, however, we prove Little's Law in two settings, motivating each with an example.

2.1. System Is Empty at 0 and *T*

2.1.1. Example: Supermarket. Little and Graves (2008) consider a supermarket that opens at 7 A.M. and closes at 11 P.M. each day. The "items" are customers, and so the number in the store (system) is zero at the start and finish of the day. By definition, the number of customers in the store, averaged over the day, is L. The average time the customer spends in the store is the wait in the system, and has some average W hours/customer for the day. Customers entering the store to shop have an average arrival rate, λ, customers/hour over the day.

In this application, the time spent in the store is the "waiting time" and so there is no need for a separate service time.

To develop the potential value of the application further, we note that checkout collects a treasure of computer-readable data from each customer. This includes the store's identifying number for each item bought, the price paid for it, whether a coupon was redeemed, the total amount spent in the shopping trip, the time of day, and date. We would like another data source as well. Suppose the store installs RFID (radio frequency identification) on its carts. Then we can find out when each customer enters the store and picks up a cart and also when the customer leaves checkout. The difference gives us each customer's time in store, W_i, and, by averaging over all customers, W. The cash register can read the RFID and link the customer to the checkout data in each case. The number of customers in the store at that instant is inferable because the timing of each arrival and departure is known. The total number of customers during the day is easily determined by tabulating the number of checkouts. By dividing this by the number of hours the store is open, we obtain, λ, the average arrival rate. The only LL parameter not measured directly is L, but it can be calculated quickly from $L = \lambda W$. Thus, all LL parameters are measured for the particular day, and we have rich customer detail as well.

(In the interests of accuracy, we note that putting RFID on carts keeps track of carts, not customers. In some cases, multiple individuals will be associated with a given cart, including people at home in contact by cell phone. We could describe the collection as a "buying unit," but "customer" seems simpler and unlikely to be misunderstood.)

2.1.2. Proof of Little's Law for a System Empty at 0 and *T*. $L = \lambda W$ is a simple formula, and in this case it has a simple proof. Consider a sample path (realization) of a queuing process over a time interval $[0, T]$. A plot of the number of items in the system versus time during the interval might look like Figure 2.

Let

$n(t) =$ number of items in the system at time t
$\lambda =$ average arrival rate in $[0, T]$ (items/time unit)
$N =$ the number of items arriving in $[0, T]$
$L =$ average number of items in the system during $[0, T]$
$W =$ average waiting time of an item during $[0, T]$ (time units)
$A = \int_0^T n(t)\, dt =$ area under $n(t)$ over $[0, T]$ (time units).

THEOREM LL.1 (LITTLE'S LAW). *For a queuing system observed over* $[0, T]$ *that is empty at* 0 *and* T *and has* $0 < T < \infty$, *the formula* $L = \lambda W$ *holds.*

PROOF. Using the notation in the left-hand column above, we see that

$$L = A/T$$

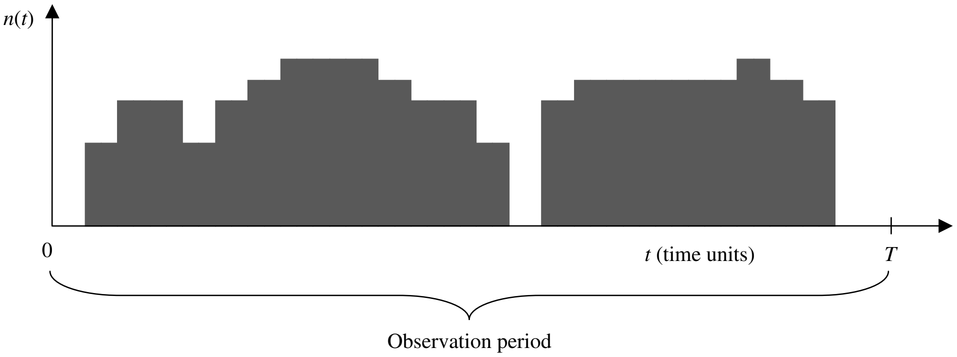
$$\lambda = N/T$$

$$W = A/N,$$

whence

$$L = A/T = (A/T)(N/N) = (N/T)(A/N) = \lambda W. \quad \square$$

Figure 2. $n(t)$ vs. t, showing the area under the curve for a sample path with $n(0) = 0$ and $n(T) = 0$.



2.1.3. The Reason Why Little's Law Is True. The author has long sought a good articulation of why Little's Law is true. The theorem LL.1 is sufficiently simple that it is clear what is happening. Notice that the proof uses the area "A" twice, once to calculate L, the average number of items in the system, and again to calculate W, the average wait for an item. This dual use is possible because of a simple physical fact. *An item in queue is also waiting.* In other words, at the same time that a customer is standing in line and so can be counted, he or she is also accumulating minutes waiting. This is the physical explanation of why $L = \lambda W$.

2.1.4. Remarks About LL.1.

Our proof of LL.1 could be made more formal. Following Wolff (2011), we could define things in terms of a sample path, ω, in a queuing process. For $i \geqslant 1$, item i arrives at $t_i(\omega)$ and leaves at $t_i(\omega) + W_i(\omega)$, where $W_i(\omega)$ is the wait of item i in the system. The arrival times and waiting times combine to produce $n(t, \omega)$, the number in the system at t. To emphasize the finite time period $[0, T]$, we shall sometimes write $L(T)$, $W(T)$, etc. The author has no objection to formality, but in this section the objective was to reduce notation and illustrate with sketches, partly in hopes of reaching a broad audience.

What kind of averages? Each of the quantities in the formula is a different average with different dimensions. L is a time average of the number of items in the system, and so is dimensionless. W is the sample average of the waiting times of individual items, and so is measured in time units. λ is an average rate calculated by counting the points of arrival in $[0, T]$ and dividing by T, and so has the dimensions of items per unit time. All of them are common forms of averages that seem natural.

Under the conditions of Theorem LL.1, the formula $L = \lambda W$ is deterministic and holds exactly. In other words, we assume that we have a particular sample path. From this we can deterministically calculate each of the three LL parameters. LL.1 says that they fit together exactly through $L = \lambda W$. However, let it be noted that the determinism and exactness are after the fact, i.e., the sample path is known. This is not all bad. It just says that we are in the measurement business, not the forecasting business. This doesn't prevent the practitioner from collecting many days of data and using them to make forecasts using standard statistical methods.

LL.1 holds under nonstationary conditions. A stationary process is one whose joint probability distribution does not change with time. Taking our supermarket example, the customer arrival process is certainly nonstationary. Customers are likely to arrive a few at a time in the morning, with a rate that increases and perhaps peaks a little after noon and tapers off at the end of the day. In addition, quite possibly, the types of customer change by time of day with different mixes of housewives, seniors, couples, and students. LL continues to be true. More to the point, we did not have to assume that the underlying arrival process was stationary in order to prove LL.1, and so it is not required.

LL.1 holds independent of queue discipline. Typical queue disciplines include serving items FIFO (first in, first out), LIFO (last in, first out), at random, or by priority classes. No assumption has been made about order of service in deriving LL.1, and so the result is independent of queue discipline.

What often counts most in practice is the sample path. The sample path occurs in real time and is what the managing organization deals with. The organization may wish to control aspects of the processes involved, say, to make a profit or provide customer satisfaction. These activities are separate from Little's Law but are constrained by it.

2.2. Extending LL.1 to Permit Nonzero Starting and Ending Queues

2.2.1. Example: Wine Cellar. Both Wolff (2011) and Little and Graves (2008) use a wine cellar as an illustrative example. An item is a bottle, and so the number of items in the system is the number of bottles in the cellar. The average arrival rate is the average rate of bringing new bottles to the cellar. It is assumed that the average will continue at roughly the same rate for some time, although obviously there are fluctuations. A wine's age (in years) since its bottling is an important indicator of quality for premium wines. The taste changes with age and generally becomes more complex and interesting with new flavors appearing.

The wine cellar example illustrates two phenomena not considered in LL.1: (1) *A queue that may never go to zero in* $[0, T]$; and (2) a *queue for which it would ordinarily be expected that* $n(0) > 0$ *and* $n(T) > 0$.

$W(T)$, the average time in system over $[0, T]$, is likely to be of interest to the wine buff because it measures the average incremental age being added to the wines in the cellar during the interval.

2.2.2. Proof of Little's Law with Permissible Initial and Final Queues in $[0, T]$**.** We can establish a more general result than LL.1 by eliminating the restriction of zero starting and ending queues in the interval $[0, T]$. Figure 3 shows an example of a sample path with nonzero queues at 0 and T and shows the number in the system versus time during the interval.

Theorem LL.2 (Little's Law Over $[0, T]$). *For a queuing system observed over* $[0, T]$ *that has* $0 < T < \infty$, $L = \lambda W$ *holds.*

Proof. In LL.1 we defined N as the total arrivals in $[0, T]$. Here we introduce $S(t) =$ cumulative number of items in the system over $[0, t]$. This includes not only the cumulative arrivals up to t, but also any items that were in the system at $t = 0$. This permits $S(0) = n(0) > 0$ and $n(T) > 0$ in contrast to LL.1. The definition $A = \int_0^T n(t)\,dt =$ area under $n(t)$ over $[0, T]$, however, continues as in LL.1. Otherwise, paralleling the arguments used in LL.1 we obtain

$$L = A/T$$

$$\lambda = S(T)/T$$

$$W = A/S(T),$$

whence $L = [A/T] = [A/T][S(T)/S(T)] = [S(T)/T] \cdot [A/S(T)] = \lambda W$. □

The reader may ask, why prove LL.1 when LL.2 includes it? The answer is that for LL.1, all arrivals, service operations, if any, and departures are contained cleanly in $[0, T]$. Their meaning seems clear in practice. In LL.2 the items in queue seem possibly connected to events prior to 0 and after T, e.g., the items may have a history of waiting prior to 0 and may stay in the system after T. Our algebra is correct, but does using the LL.2 model of the world give us answers to the questions we want to ask? In the case of Example 2 (wine cellar), the author would say yes, but a practitioner should always check his or her model assumptions for relevance to the task at hand. We note that aging is not only for wine bottle populations, but also for humans or polar bears, the question being, as individuals arrive and depart from a system and time passes, what is the average incremental change in age, i.e., what is the average wait in the system? Notice that the wine bottle (or polar bear) populations present at $n(0) > 0$ immediately start accumulating incremental age until one or another of them departs.

2.2.3. Corollaries to LL.1 or LL.2 Extending Their Applicability.

(1) LL.2 holds exactly even if the queuing system is never empty.

Proof. This follows directly from LL.2 because there are no restrictions to the contrary as in LL.1. □

(2) LL.2 holds even if there is no service operation.

Proof. Service is nowhere required (or mentioned) in LL.2, only waiting time. □

This covers both the supermarket and the wine cellar examples.

(3) Classes of items or market segments. Let $k = 1, 2, \ldots, K$ index sets of mutually exclusive items. Let $\{L_k, \lambda_k, W_k\}$ be the LL parameters for a class k item. Then

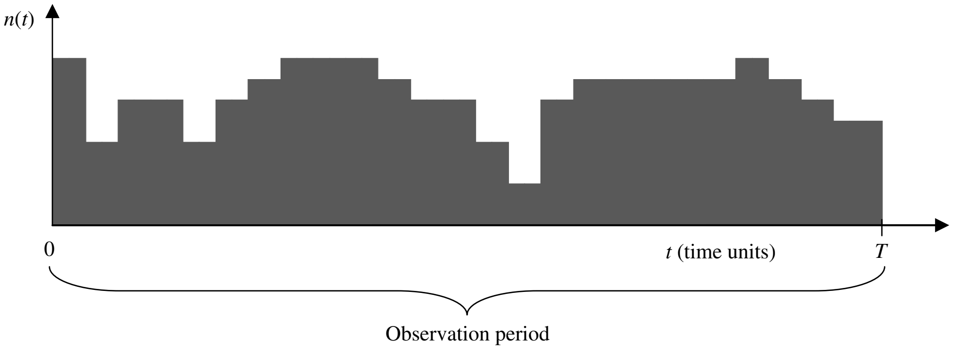
(a) $L_k = \lambda_k W_k$ and

(b) letting $\lambda = \Sigma_k \lambda_k$, $L = \Sigma_k L_k$, and $W = \Sigma_k (\lambda_k/\lambda) W_k$, it follows that

$$L = \lambda W.$$

Proof. (a) is a special case of LL.2 where we only record data about arrival time, departure time, and waiting time for type k items. (b) follows from the definitions of λ, L, and W and by noting that $W = \Sigma_k (\lambda_k/\lambda) W_k = \Sigma_k L_k/\lambda = L/\lambda$. □

Figure 3. $n(t)$ vs. t, showing the area under the curve for a sample path with $n(0) > 0$ and $n(T) > 0$.



The practical meaning of (3) is that, for example, in a machine shop with many types of goods being produced, we can look at an individual type, e.g., k, in isolation, and use LL. Of course, other activities going on may affect the values of L_k, W_k, and λ_k, but not the LL relationship. In the example of the supermarket, we can look at a particular market segment k, say, deli buyers, defined by purchases from a specific set of deli products. This segment can be compared with the remainder of the customers in terms of average time spent in store (W_k), average number in the store over the day (L_k), and average arrival rate during the day (λ_k). Furthermore, the store can collect data having nothing to do with LL (e.g., checkout data), but that make it possible to evaluate profitability of the segment and the success of any promotional activities that may have been run during some time period of interest. LL provides a structure for the analysis and its own consistent three measures of performance.

LL.2 encourages flexibility. Consider, for example, dynamic control in a job shop. Suppose we have a machine shop that takes on jobs that entail making widgets of various types. One particular job may be of special interest. Suppose a job consists of some fixed number of widgets painted blue. Management is particularly interested in this job and manually decides when to start each widget, depending on the number in process, the current throughput, and delays so far. Management can apply LL.2 over $[0, t]$ to monitor and manage the progress on this job. Little's Law tells what has happened up to t, and management can then decide what to do next.

2.3. Why Is Little's Law Useful in Practice?

Although §4 provides further discussion of Little's Law in practice with several real-world case studies, we introduce three general points here. These are true whenever LL holds, not just for the finite T case.

If you know two of {L, λ, W}, you can quickly calculate the third. This is perhaps the most common and also the most valuable use of LL and seems to turn up regularly in case studies and illustrative examples. Sometimes one of the three may be difficult or expensive to measure, and yet important to know in a particular application. Little and Graves (2008) illustrate each of the three possibilities with plausible operations management scenarios.

L, λ, and W *are three quite different and important measures of effectiveness of system performance, and LL insists that they must obey the "law."* Some system managers may seek high average throughput (large λ), e.g., to produce many airplanes/month; others may want short average waits (low W) to give good customer service; and still others may desire to reduce the average number in the system (low L) to reduce inventory costs. LL locks the three measures together in a unique and consistent way for any system in which it applies. LL will not tell the managers how to handle trade-offs or provide innovations to improve their chosen measures, but it lays down a necessary relation. As such, it provides structure for thinking about any operation that can be cast as a queue and suggests what data might be valuable to collect. Note that if λ is prespecified, the only way to reduce inventory (L) is to find a way to reduce average wait/item (W).

In engineering design, back-of-the-envelope calculations can often give early estimates of average queues and waiting times, e.g., for queues in the processes inside computers or in the processes for the manufacture of products. Design decisions require judgmental engineering calls. That's fine. It's what engineers often do. With LL they can use their knowledge of the physical systems to estimate two of the three parameters and immediately fill in the third as they try to scope out possibilities and rule out ineffective designs quickly.

2.4. Sample Path vs. Stationary Proofs

So far we have dealt with sample path proofs, but there is another world of stationary proofs. Circa 1960 the author taught a course in queuing largely based on Morse (1958) and knew only stationary queuing models. We shall take up the distinction in §3.

3. Importance to Queuing Theory

Little's Law and its extensions have stimulated a remarkable number of theoretical papers, especially considering their modest ancestry in a simple three-parameter formula. Whitt (1991) and, more recently, Wolff (2011) have written extensive reviews. El-Taha and Stidham (1999) have written a book that covers Little's Law and its extensions and much new material, all approached by sample path analysis. The present paper provides an overview of major theoretical developments, not a review, and tries to describe content without repeating proofs. Readers interested in the proofs will find many of them in the three references above.

Queuing theory, as a field and growing body of knowledge, is primarily concerned with relationships among long-run averages and is the subject of this section. Correspondingly, queuing theory is not focused on what happens in finite intervals. Little's Law, as exemplified by Little (1961) and studied by many others, deals with the long-term average arrival rate, number in the system, and wait per item. However, finite interval analyses involving the formula $L = \lambda W$ will often be valuable in practice for engineering design and operational problem solving, as discussed in §§2 and 4. In the author's view, these two intellectual efforts support each other.

Listed below are several theoretical developments that have some of their roots in Little's Law. Under each major topic, we state the main results and then discuss them. The notation is that of Wolff, except for a few symbols for which a slightly different notation seems standard.

Before embarking on the list, we distinguish between *two types of proof* for Little's Law and its extensions: sample path and stationary. Whitt (1991, p. 236) says, "There are two frameworks for expressing the results. The first is a *deterministic framework* involving averages over sample paths. The second is a *stationary framework* involving steady-state distributions." He goes on to say, "The deterministic framework is appealing because it requires only elementary arguments... and lays bare the essential ideas.... .The stationary framework is appealing because it leads beyond the particular issue [at hand]... to a full investigation of the concept of statistical equilibrium or steady state."

3.1. The Role of Little's Law in Queuing Theory

The first really rigorous proof of $L = \lambda W$ in the generality we know today was a sample path version given by Stidham (1972). He followed this with a simpler but equally rigorous sample path proof in Stidham (1974) under the title, "A last word on $L = \lambda W$." Later, in Stidham (2002, p. 212), he added, "Of course it was not a last word, but just the beginning of a twenty-five year research program concerned with applying sample path analysis to a wide variety of problems in queuing theory and related fields... ."

3.1.1. Main Results. Let

$$L = \lim_{T\to\infty}(1/T)\int_0^T N(t)\,dt \quad W = \lim_{n\to\infty}(1/n)\sum_{i=1}^{n} W_i$$

$$\lambda = \lim_{t\to\infty}\frac{\Lambda(t)}{t}$$

where $N(t) =$ *the number of items in the system at* t; $W_i =$ *the waiting time of item* i; *and* $\Lambda(t) =$ *the cumulative number of arrivals up to* t.

Little's Law (sample path version):

Theorem. *If limits* λ *and* W *exist and are finite,* L *exists and is finite, and*

$$L = \lambda W.$$

Little's Law (**Stationary Version**). In the sample path version, it is only necessary to know that there is some sample space and that the sample path comes from it. For a stationary (steady-state) version, we need some sort of probabilistic structure. For example, a classical $G/G/s/\infty$ queuing system is such a structure, although more restrictive than might at first glance appear. A more general approach was developed by Franken et al. (1982), in which, starting from a jointly stationary interarrival time and waiting time process, they construct a stationary process $\{N(t)\}$. Furthermore, they show that

Theorem. (1) W_i *has the same distribution for all* i.
(2) $N(t)$ *has the same distribution for all* t.
(3) $E(N(t)) = \lambda E(W_i)$.
Here $E(\cdot)$ *is the expectation operator.*

3.1.2. Remarks. Little's Law holds under remarkably general conditions. It applies to individual queues and also networks and subnetworks. It holds for subclasses of items as well as the whole population and is independent of queue discipline. It holds and is useful in many operations management settings, for example, when the managerial goal is to obtain high throughput (λ) and control processes are superimposed on the queuing model.

3.2. An Important Generalization: $H = \lambda G$

$L = \lambda W$ turns out to be a special case of the more general $H = \lambda G$, as defined below.

3.2.1. Main Result. For each item, i, define a function $f_i(t)$, $t \geqslant 0$, where $\int_0^\infty |f_i(t)|\,dt < \infty$, and for some finite $l_i > 0$, $f_i(t) = 0$ for $t \notin [t_i, t_i + l_i]$. (Often $l_i = W_i$.) Define

$$G_i = \int_0^\infty f_i(t)\,dt, \quad i \geqslant 1, \quad \text{and} \quad H(t) = \sum_{i=1}^{\infty} f_i(t), \quad t \geqslant 0.$$

Further define

$$G = \lim_{n\to\infty}\frac{1}{n}\sum_{i=1}^{n} G_i \quad \text{and} \quad H = \lim_{T\to\infty}\frac{1}{T}\int_0^T H(t)\,dt \quad \text{and}$$

λ as defined in 3.1.1.

Theroem ($H = \lambda G$). *If limits* λ *and* G *exist and are finite, and technical condition* $l_i/t_i \to 0$ *as* $i \to \infty$ *holds, then* H *exists and*

$$H = \lambda G.$$

3.2.2. Remarks. This result is due to Brumelle (1971) and Heyman and Stidham (1980).

$H = \lambda G$ permits the practitioner (or theorist) to place a different weighting function on the time spent in the system by each item. $L = \lambda W$ is the special case in which $f_i(t) = 1$ for $t \in [t_i, t_i + l_i]$ and $l_i = W_i$. In the general case, $f_i(t)$ could be used in a variety of interesting applications, for example, the dollar rate of return on the ith asset in a portfolio of assets, or the unique cost rate of an item i in an inventory. At the same time, we still have $L = \lambda W$ for the underlying queuing process. Therefore, we have available the standard measures L and W simultaneously with the new H and G. In the financial assets example, we would know not only the time average dollar return H and average dollar return per asset G, but also the average number of assets L and the average time of holding an asset, W.

We note that the holding cost concept has been extended by Glynn and Whitt (1989) to include lump costs.

An important application of $H = \lambda G$ is the concept of work, defined as the sum of the remaining service times for all the items in the system at t. This creates a new process in its own right.

$H = \lambda G$ is related to the Rate Conservation Law (RCL) of Miyazawa (1994). Although the approaches are different, Sigman (1991) has shown that the sample path versions of these results are the same.

3.3. A Distributional Form of Little's Law (DLL)

It is natural to ask whether there is a distributional form of Little's Law, i.e., whether the stationary distributions of N and W are related in some way by λ.

3.3.1. Main Result. We assume (a) items depart FIFO from the queuing system, (b) $\{W_i\}$ is stationary, and (c) for every i, W_i is independent of the arrival process after item i has arrived; in particular, it is independent of $t_{i,}$, which tells "how long ago" the item arrived. Under these assumptions, *N has the same distribution as $\Lambda(W)$, where $\{\Lambda(t)\}$ and W are independent and $\Lambda(t)$ is the cumulative arrivals up to t.*

3.3.2. Remarks. The conditions (a) through (c) are quite restrictive. The result is due to Haji and Newell (1971). Other distributional forms have been investigated. A fairly recent discussion appears in Bertsimas and Nakazato (1995).

3.4. Growth of Sample Path Analysis of Queuing Systems

The general sample path proof of Little's Law in Stidham (1974) confirmed the power of the sample path approach. It has grown since then and has been given a further boost by the book *Sample-Path Analysis of Queueing Systems* (El-Taha and Stidham 1999).

4. Importance to Practice

The two biggest fields in which Little's Law regularly comes into play are operations management (OM) and computer architecture ("computers"). OM covers a multitude of practical applications both in the manufacture and distribution of goods, and also in providing services. Computers have shown extraordinary growth in power, complexity, and applicability. With the advent and explosion of the Internet and its invasion of almost every aspect of science, engineering, commerce, and daily life, not to mention the huge growth of social networking and the imaginary lives in computer games, computers have become, arguably, the factories of the future.

4.1. Computer Architecture

To be able to talk about Little's Law and computer architecture with some specificity, we focus on servers, which are ubiquitous. As individuals we connect to them when we draw money out of an ATM machine, or have a credit card purchase approved, or visit Facebook to leave a message or find one from someone else. Servers are a huge and competitive market. They are made, or at least marketed and sold, by HP, IBM, Dell, Acer, Hitachi, Lenovo, and others. Any large operation—a bank, hospital, auto manufacturer, university, or social networking site—has one or more systems with one or more banks of servers. Some of these contain thousands of servers.

Where are the queues in these computers? Where does Little's Law offer design insight or connect to operational control variables? We shall try to shed light on these questions.

In truth, there are many queues in a single computer system. We shall focus on the top and the bottom and find queues in both places. At the top, we consider the value proposition offered by the manufacturers (HP et al.) to their customers (Facebook et al.). Then we shall look down into the hardware to find queues in the execution of instructions in individual threads. At this point we can discuss possible design insights for computer architects. HP and Facebook will be our placeholders for the many other players in the real world of 2011.

For notation and nomenclature we shall use standard queuing notation:

$L =$ average number of items in the system (items).
$\lambda =$ average arrival rate of items to the system (items/unit time).
$W =$ average wait in the system (time units/item).

However, we are in a computer server world and should connect to the terminology used there. An item will be a "request" to retrieve a piece of data needed for a response that Facebook wants to make to a member or "post" data that is derived from new activity by members. Therefore, the connections are

$L =$ average number of requests in process
$\lambda =$ average arrival rate of requests (requests/second)
$W =$ average response time per request (seconds) = latency (seconds).

Latency is a key performance measure for computers (low is good) and has an unspoken "average" implicitly attached, but *response time* seems more meaningful for a layperson.

The value proposition that Facebook would like from HP is a *low average response time* at a reasonable price so that its many members can communicate easily with one another. Communications in Facebook can take many forms: changes in the member's profile are reported to friends via the news feed. The news feed also brings to the member changes in friends' profiles and other events or commentaries. Friends can be added or subtracted. A member can post a message on the wall of another member. Photo albums can be uploaded, etc. The task of Facebook's main application program is to turn members' activities into a stream of requests to the server. In the terminology presented above, Little's Law tells us that

$$\textit{ave. response time to a request (seconds)} \\ = \frac{\textit{ave. no of requests in process}}{\textit{ave. arrival rate of requests (requests/second)}}.$$

A computer architect looking at this would notice that the arrival rate of requests is fixed by Facebook members' activities so that to reduce response time, the average requests in process (i.e., the average queue of requests) should ideally be reduced.

But where are those requests in the machine? The answer is: in computer memory.

Computer memory comes in various *tiers* with different inherent response times. The tiers range from CPU cache (fast), to DRAM (dynamic random access memory) (not quite as fast), to RAM (random access memory) (a little slower), all the way to solid-state drives and finally to network attached storage (banks of hard disk drives). The computer architect must trade off speed (quick response) with cost, so as to make the server cost effective for the potential customer.

Although hard drives have increased in speed and decreased in response time over the years, they have been vastly outstripped by the increased speed and decreased response time of computer chips. There the growth in performance has been exponential (Moore's Law).

Consider a typical hierarchy of memory devices laid out in order of decreasing speed as shown in Table 1. Note that response time ranges from about a nanosecond up to a glacial four milliseconds. As might be expected, the cost per unit of storage goes in the opposite direction, although there is room for ingenuity in engineering design.

One of the author's puzzlements has been, "where are the queues, physically?" If we look deeply enough inside the machine, we shall eventually find a "von Neumann computer" and, indeed, many of them. A von Neumann computer is an elementary machine that executes a set of computer instructions serially. In other words, if you list a set of instructions on a page and start at the top, the machine does them one at a time until it reaches the end of that particular piece of code and thereby completes its current task.

The server CPU contains microprocessors, each of which contains cores, perhaps 8 or 16 of them. Each is a von Neumann computer. Now we are coming to the queues. The piece of code the computer is executing is aptly called a thread (of instructions). The author visualizes it as a list of instructions running down his penciled code sheet. Oh, oh, the instruction calls for a piece of data that is somewhere else in memory, electronically far away from the thread. The thread STOPS and sends out a request for that data. This may take a relatively long time. It is really a whole new retrieval operation with a possibly large response time. *The stopped threads are in queue*. (In standard queuing terminology, the stop requires a service operation, which will have a corresponding service time. The service time is a random variable.)

Table 1. A set of memory tiers and their average response times (latencies) in microseconds (μs).

L1 L2 L3	DRAM	RAM	Solid state drive	Network attached storage
Cache 0.001 μs	0.05 μs	1 μs	250 μs	Disk drives 4,000 μs

There are a very large number of threads executing instructions in real time on a server running Facebook operations. Because of the speed of modern nanosecond chips compared to the speed of a Facebook member who is making requests in seconds, we can work in a massively parallel way (even though each thread is actually serial). We can "time slice" or "multiprocess," moving from a stopped thread to a ready thread, although the CPU must know how to find the location of the stopped thread so that it can go back there. Note that this will cost the CPU some time and will require some memory. Time slicing and multiprocessing introduce a factor of, say, 10 in the number of threads that can be handled, thereby creating 100s or 1,000s or 100,000s of threads being worked on at once in terms of a Facebook member's time scale. For the input rate of all online members, let $\lambda =$ the average rate of total thread processing required after all members' requests have been broken down into whatever detailed subtasks are required. Let L be the average number of stopped threads in queue during some relevant time. Then $W =$ average response time, measured, say, in seconds.

HP is trying to sell servers to Facebook on the basis of low average response time. Component vendors are trying to sell HP add-in modules to improve W even more (or, alternatively, permit the server to handle a greater input rate, λ). Like any good model, Little's Law ignores most of the details of the operation (the rest of the code!) and focuses on the important design details that will affect the measures that HP's customers care most about. Similarly, vendors selling add-in modules to HP will focus on memory tiers that will improve response time. It turns out, as we shall see, that Little's Law must be combined with other models to understand what is going on and enable engineers to make sensible design decisions.

4.1.1. Queue-Related Overhead.

A model is needed to describe an important second-order effect that shows up in many real-world queuing systems, including those in computers. We shall call it *queue-related overhead* or, with more words, describe it as *an adverse nonlinearity in response time as queues become large.* In operations management for manufacturing, for example, as queues increase, someone has to keep track of the items. This incurs a cost. It may be extra record keeping and the people to do it, or it may be keeping track of physical objects and finding a space to store them temporarily and moving them there and back. The combined costs (partly time, partly dollars) represent an overhead associated with large queues. Computers have an analogous issue: they deal with pieces of data, and it is necessary to know where every data item in the queue is located in memory, itself a costly resource. If the average queue, L, is large, the extra cost will be large. Furthermore, the CPU itself must devote extra time to keeping track of everything, and this effort too is part of the overhead. The extra cost, whatever is its source, can be nontrivial—more memory is required, more switching takes place among threads. This cost is not a consequence of LL but is a second-order effect requiring a new submodel. In particular, switching among threads and remembering where the process was when the thread stopped will take precious nanoseconds and slow down the average service rate. This will feed back to increase the queue, creating a larger queue in a nonlinear way with adverse effects on response time (Flynn 2010).

Another important design characteristic of a server is the physical capacity of its various memory tiers. Cache has very low latency, and so if a request can be satisfied there, it will be quick, but cache has little capacity. It fills up quickly, and the request is sent on to slower and larger capacity tiers.

We like the computer architecture example for illustrating the overhead phenomenon because performance measurements are routinely made on computers and we can see the nonlinear effects quantitatively in plots and tables and understand why they are taking place, as well as see to what degree they are hurting performance. Commercial services are available to do benchmark tests of servers by feeding them with a simulated client load of requests under various hardware and load conditions. The data plotted below comes from performance tests at a Microsoft FAST Search Farm. This is a third-party service offered by Microsoft to hardware vendors and their customers. Suppose Intel is trying to sell servers to a large hospital for storage and retrieval of medical records. Intel and the hospital study the nature of the flow of requests and devise a load simulator. This can then be run on various configurations of processors and memory tiers. The situation may be competitive: HP may be interested in the same business. Microsoft FAST Search Farm is a disinterested party that runs performance analysis scenarios as a paid service.

Figure 4(a) shows the *measured* effect on latency (response time) at the aggregate level during a benchmark test for a particular server configuration designated as Scenario M8. Plotted is latency versus the arrival rate, λ, of user requests (simulated to resemble closely the client's setup). The measurements were done in November 2010.

As λ, the average requests/second, increases in the load test, L, the average queue of requests goes up approximately linearly and then climbs steeply, indicating failure. After about 18 requests/second further requests are essentially dumped.

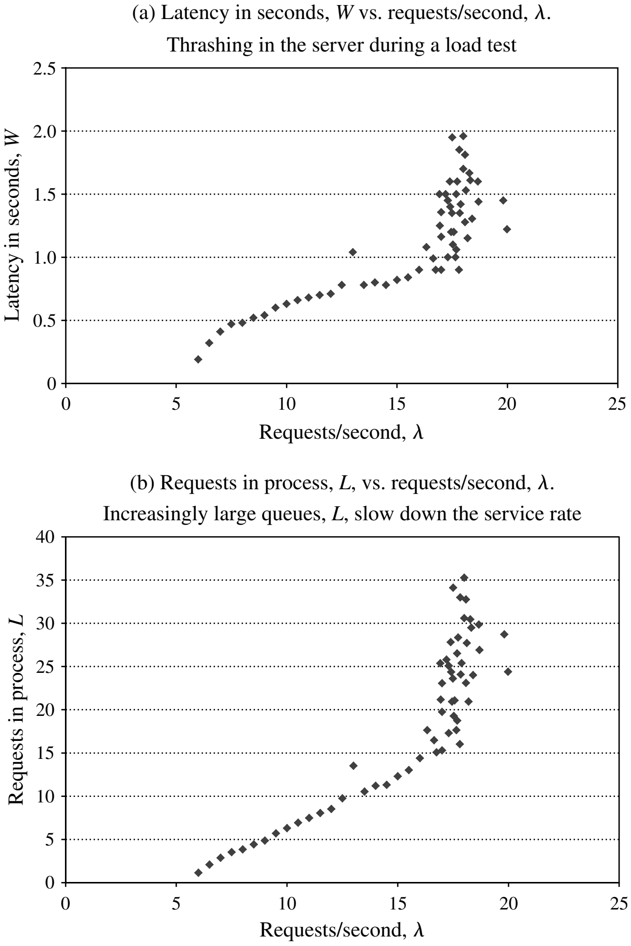
Some comments.

(i) These are actual numbers from the behavior of an Intel server under increasing retrieval/storage loads in simulation of client operations.

(ii) At high client loads, performance seems to be hitting a brick wall.

(iii) The author's interpretation of this, following his discussion with Flynn (2010), is that large queues (L) are taking up storage and represent "overhead" that is *slowing the average service time,* creating almost a singularity.

Figure 4. Increasing the load (requests/second, λ) on the server eventually causes it to fail.


(a) Latency in seconds, W vs. requests/second, λ. Thrashing in the server during a load test

(b) Requests in process, L, vs. requests/second, λ. Increasingly large queues, L, slow down the service rate

(iv) We could model the nonlinearity analytically and fit it to data, but the plot tells the story better.

(v) We have used Little's Law to go from the measured latency, W, in the top graph to the average queue, L, in the bottom. This is more useful and interesting than it might appear at first glance. The client, Intel, is most interested in measuring response time (latency, W). This is rather easy to do. To generate a point on Figure 4 (a) the simulation system, using the designated rate, λ, generates, say, 1,000 queries. These go out at rather random intervals, as would be expected of a cluster of users making requests. The measurement process records the response time of each request as it is completed and keeps a running total. After the requests are all back, dividing by 1,000 gives the average latency, W, and a point on Figure 4(a). LL gives us the corresponding point for L on Figure 4(b). As an alternative, a direct measurement of the average queue in the same time period would require keeping track of all the ups and downs in the queue during that period, then somehow integrating to find the area under the curve and dividing by the elapsed time. It is much easier to use LL.

(vi) The upward-sloping part at the beginning of the curves in Figure 4 is normal steady-state queuing theory. The average queue and delay grow as the arrival rates increase. However, at a fairly abrupt point on this machine, the large queues slow the service rate and the machine begins to fail. One could accurately say that the traffic intensity (usually denoted by ρ) goes to 1, but the system is not like an ordinary $M/M/1$ queue with a constant service rate, μ, and $\rho = \lambda/\mu$. An important feature is that increasing λ increases L, which in turn decreases μ. Thus, a deleterious feedback is initiated, hastening failure. The computer architect wants to understand what is going on so as best to design the system. With enough data, we could presumably model the process.

(vii) On the client side, Facebook or whoever does not want failure at all and so buys ample capacity. However, in order to do this, the client needs to know at what λ the failure is likely.

(viii) It is characteristic of applications of LL in practice that it is used in conjunction with other models and other measurements to look for system improvements.

4.1.2. Computer System Analysis and Computer System Performance. We have taken a look at the outside and inside of a server as it might be used by Facebook or a similar firm that is managing many user requests. We find not only Little's Law, but other phenomena, that can potentially be modeled and used with it to understand and improve performance. As we shall find in operations management in the next section, Little's Law plays a sufficiently distinct yet integrated role in understanding and solving engineering problems that it appears in major textbooks. An example of this in computer architecture is Lazowska et al. (1984) with their text on computer system analysis using queuing network models. Another is Gunther (2010), who addresses the narrower task of analyzing computer system performance.

4.2. Operations Management (OM)

Operations management (OM) covers a large arena of practice. Little's Law is discussed in several current OM texts, including Hopp and Spearman (2000), Cachon and Terwiesch (2004), and Chhajed and Lowe (2008). Much of the introductory material in §2 applies directly, as do our illustrative examples there. Queuing theory and OM generally use different notations for Little's Law parameters, and so we introduce that issue first. In a desire to get close to people who are actually using LL to support decision making, we also take up three field cases drawn from people with whom the author has communicated directly.

4.2.1. Most Commonly Used Notations.

(a) Queuing theory. $L = \lambda W$

$L =$ average number of items in the system (items)
$\lambda =$ average arrival rate of items to the system (items/unit time)
$W =$ average wait in the system (time units/item)

(b) Operations management (Hopp and Spearman 2000)

$TH =$ WIP/CT
$TH =$ throughput (items/unit time)
$WIP =$ work in process (items)
$CT =$ cycle time (time units/item)

(c) Operations management (Cachon and Terwiesch 2004)

Average Inventory = Average Flow Rate × Average Flow Time, where the flow unit is chosen to fit the application (item, customer, vehicle, widget...), and so the dimensions of the terms in the LL equation are

Average Inventory (flow units)
Average Flow Rate (flow units/unit time)
Average Flow Time (time units/flow unit).

4.2.2. OM Has a Different Orientation from Queuing Theory. Little and Graves (2008) make a few observations about OM versus queuing theory: It is easy to see that $TH = WIP/CT$ is equivalent to $\lambda = L/W$. However, there is a more fundamental difference, in that for OM, the law is stated in terms of the average output or departure rate or flow rate for the system, rather than the arrival rate. This reflects the perspective of a typical operating manager, especially in manufacturing. Output is the primary focus, because it is nominally its raison d'etre. As stated in (b), we see that any increase in output requires either an increase in work-in-process inventory or a reduction in cycle time or both, unless some change is made in the manufacturing process itself.

Furthermore, in many contexts the output rate is determined exogenously and is given to the manufacturing system; it reflects orders, actual sales, and/or a forecast of sales. The manufacturing system must then manage its operations to achieve this rate. It will need to determine how to release work to the operation so as to meet the target output. In effect, the arrival process is endogenous. The operations manager decides the arrivals based on desired outputs. There is extensive research in the operations literature on how best to set the work release (or arrival process) to achieve the output targets. The best policies are dynamic and depend on the state of the manufacturing shop, e.g., depend on the work in process.

We now turn to actual case studies in which the author has personally communicated with the people involved to understand better how things work out.

4.2.3. Case Studies.

(a) Staffing emergency departments in hospitals. Hospitals are full of opportunities to apply OM methods. Nowhere is this more evident than in the emergency department (ED), where the unexpected is the norm. Mark Harris, MD, published a paper (Harris 2010) that he entitled "Little's Law: The Science Behind Proper Staffing." To learn more about this work, the author talked to Harris for over an hour. He works for a company, TeamHealth, which has a major business in outsourcing emergency departments.

It operates over 500 EDs, some at large hospitals. Harris is the chief designer of the staff-scheduling systems.

An ED may be viewed as a queuing system with an arrival rate of patients who receive medical service and leave (possibly sent home or perhaps admitted to the hospital proper). A patient arriving at an ED potentially receives a broad variety of services. She/he will quite likely see a physician and almost certainly will see a nurse (RN), may occupy a bed, see a hospitalist, have an X-ray, lab work, etc. A key measure of ED performance is the *average length of stay* (LOS), because this says something about efficient use of expensive resources and also about moving the patient toward recovery.

LOS fits easily into Little's Law: it is W, the average time spent by a patient in the system, here an ED. The average arrival rate of patients is not ordinarily considered controllable and has the same meaning as λ in queuing. L, the average number in the system in queuing, translates into "average number of patients in the system" or borrowing OM terminology, "average patients in process," and can be given the acronym PIP. So, rewriting Little's Law in ED-relevant terms gives

$$\text{(average length of stay (LOS) in ED)} = \frac{\text{(average no. of patients in process (PIP))}}{\text{(average arrival rate of patients, } \lambda)}.$$

TeamHealth has developed algorithms for doing the scheduling of all the various kinds of staff and resources. We can think of the passage of the patient through the ED as a series of queues. Each of these involves one of the ED resources, such as MDs, RNs, hospitalists, beds, etc. Each contributes to the average length of stay (LOS). A typical overall LOS goal for the busiest eight hours of the day might be to keep LOS under three hours. The average patient arrival rate is reasonably predictable by time of day and, because the queues are in series, is the same for all types of ED resources. Not all resources are used by all patients, but the theory inherently deals with averages.

Consider, for example, MD resources. Little's Law tells us

$$\text{(average length of stay (LOS) in MDs)} = \frac{\text{(average no. of patients in process (PIP) with MDs)}}{\text{(average arrival rate of patients, } \lambda)}.$$

Because of the serial arrangement of the queues, the average arrival rate of patients, λ, is the same for all resources, and so one constant of proportionality between LOS and PIP applies to all.

Now we need to relate LOS to our control variables, which are the numbers of MDs, RNs, etc., to be used in the staffing plan. From historical records, the ED can determine the average *producer productivity* for MDs, defined as the average number of patients/hour that an MD can handle. Call this PP(MD) patients per hour. Thus, we can find the minimum number of MDs needed by dividing this into the patient arrival rate, i.e., λ/PP(MD). For example, if MDs can treat 2.5 patients/hour and the patient arrival rate is 10 per hour, the minimum number of MDs needed is 4.

We say minimum. The reason is basic queuing theory. There is variability in the interarrival times between successive ED patients and also in the service times of MDs, RNs, and other resources. We know from queuing theory (e.g., the $M/M/1$ queue) that, as the arrival rate approaches the service rate, the average number of patients in process (PIP) and average patient length of stay (LOS) both increase, slowly at first and then rapidly. Therefore, we shall need to build in extra resources (MDs, RNs, etc.) to meet LOS targets, even though this means that there will be idle resources part of the time. Harris suggests 10%–20% extra staffing compared to the minimum, as a rule of thumb. We now have a basic staffing strategy because we can set the number of MDs, RNs, etc., to hold the total LOS across all of the resources to be in the target range.

However, there is another danger. If, for whatever reason we simply have too few RNs currently in the ED to achieve the desired LOS for RNs, we may think we can achieve it for the overall ED by returning to our formulas and increasing, say, the MDs. Sorry. What happens is that the RNs become a bottleneck and patients pile up in front of them, leaving the extra MDs idle. Because everybody tries to help out in an ED, some of the MDs may pitch in and do work that RNs normally do. Harris reports that this can cause confusion and make matters worse. The answer is "balanced" staffing, in which the RN-to-MD ratio and similar ratios are set up correctly at the start. It is best to be balanced and suffer a temporary increase in the overall LOS until adequate resources can be found.

Harris and TeamHealth and a number of similar firms have developed quite elaborate models and calibrated them on historical data collected at ED installations. Such models can, for example, take into account specific work rules at a particular ED. However, these complicated systems are black boxes for the CEOs of hospitals and hard to understand. Harris reports that the simple explanations of staffing basics built up from Little's Law and basic queuing concepts are much more useful for communicating the reasons for formal models to CEOs than trying to explain complicated black boxes or simply saying "Trust me."

(b) Michael George: Lean Six Sigma and Little's Law. Michael George, usually called Mike George, founded a consulting company, Converge Consulting Group Inc., with a practice centered on Lean Six Sigma in manufacturing (George 2002). A key feature in the methodology of the firm's practice was the use of Little's Law. I met Mike in my office at MIT in October 2008 and was joined by Steve Graves to discuss our mutual interests in Little's Law. Mike connects the latter to Lean Six Sigma, starting from his basic principle (George 2002).

> The Principle of Lean Six Sigma: The activities that cause the customer's critical-to-quality issues and create the

> longest time delays in any process offer the greatest opportunity for improvement in cost, quality, capital, and lead time. (p. 4)

George offers a methodology for improving organizational effectiveness that starts with "The First Law of Lean Six Sigma for Supply Chain Acceleration" (p. 47). This has to do with finding the time delays (potential time traps) in each workstation of the process. "The Second Law" (p. 51) is the 80/20 rule, which says that 80% of the delays are caused by less than 20% of the workstations. The Third Law is Little's Law, (p. 49), which he expresses as

$$\textit{Process Lead Time} = \frac{(\textit{Number of "Things" in Process})}{(\textit{Completions per Hour})}.$$

If we rewrite this equation in the OM words and notation of Hopp and Spearman (2000), it becomes

$$\textit{Cycle Time}\colon CT = \frac{\textit{Work in Progress: WIP}}{\textit{Throughput: TH}}.$$

One of the characteristics of OM applications of Little's Law is to draw the vocabulary for the application from jargon of the client setting. Hopp and Spearman also offer alternate words and expanded definitions: Throughput (*TH*) is "the average output of a production process (machine, workstation, line, plant) per unit time," work in process (*WIP*) as "the inventory between the start and end points of a product routing," and cycle time (*CT*) as "the average time from release of a job at the beginning of the routing until it reaches an inventory point at the end of the routing (that is, the time the part spends as *WIP*.)" George (2002) also has alternative descriptors, calling *TH* the velocity of products through the plant in items per hour, *WIP* the average number of items in process in the plant, and *CT* the process lead time, i.e., the average hours each item spends in the plant.

Note that Little's Law is not in charge of changing these quantities—that is the responsibility of the lean-six-sigma activities. By eliminating "time traps" in the operation of individual workstations, *CT* can be reduced. By improving the precision of certain parts, buffer stock may be decreased, and with it total WIP. Mike's book reports a variety of case studies in considerable detail. The role of Little's Law is to provide a structure that helps the client, working with the consultants, define the three rather different measurements of LL. After process improvements have been made, the same measurements validate the results. Mike George's financial success and accolades from his clients testify to his Little's Law methodology.

Although we have talked about a plant and thereby implied a manufacturing facility, nothing changes but the vocabulary if we think in terms of a service process with multiple stages. Mike has, for example, worked with municipalities to improve their paperwork processing.

Postscript: George's firm grew to considerable size and about 2006 he sold it to the large consulting firm Accenture, where the practice continues to operate as a division. In March 2010, as the author was starting work on this 50th anniversary paper, he received an e-mail from Mike showing a photograph of a white board that he described as "Little's Law in Russian," evidently from an international consulting engagement by his former company.

(c) Flow analysis of observation units in a hospital. In the current period of rising health costs and increasing demand for hospital services, hospitals are under continuous pressure to provide more efficient service without expensive expansion of facilities. Bill Lovejoy of the University of Michigan recently completed a study entitled "Little's Law Flow Analysis of Observation Unit Impact and Sizing" (Lovejoy and Desmond 2011).

Observation care is defined by Medicare and Medicaid as a well-defined set of specific, clinically appropriate services. These include ongoing short-term treatment, assessment, and reassessment before a decision is made whether a patient requires further in-patient treatment or can be discharged from the hospital. Observation services are commonly ordered for patients who come to the ED.

Congested hospitals frequently find that a considerable number of observation care patients are occupying expensive in-patient beds in the hospital. This suggests the possible advantage of constructing lower-cost observation care units within the ED. Lovejoy and Desmond (2011) have conducted an analysis of this strategy using a Little's Law flow analysis. For the large hospital they were dealing with, their analysis indicated that a substantial observation care unit of 14 beds would be warranted, taking into account the cost and revenue flows. The resulting net annual revenue attributable to the proposed unit was on the order of $5.8 million.

Bill Lovejoy also e-mailed to the author what he thought "in a nutshell" were the key advantages of Little's Law in this setting:

1. "*Check on the raw data*: It is surprising how often data gathered in hospitals is inaccurate or inconsistent. Data is gathered in a very decentralized fashion by a range of personality types each of which can make their own assumptions about how things are defined. I have frequently been handed spreadsheets with data that do not add up. Little's Law provides a reality check."
2. "*Computing backfill patients for freeing up inpatient beds:* If the patients being removed (e.g., going to observation status) are different than backfill patients (e.g., patients qualifying for regular admission) then the backfill flow has to account for the change in lengths of stay."
3. "*Rough cut sizing*: Observation patients stay on average 1.14 days, so for any given flow rate we know what the inventory will be from Little's Law, and dividing this by the target utilization (say 90%) suggests how many beds to install."

Another comment he made was that the simplicity of Little's Law (a three-parameter equation) made it easy for physicians to understand the patient flow analysis and see where the results were coming from.

4.3. Concluding Thoughts

4.3.1. Looking Back. Simple ideas can be extremely powerful. Little's Law seems to be one of them. It is a mathematical relationship among three averages in a queuing process. As a sometime physicist, the author has been pleased to realize that a key to Little's Law is that a person in queue is also waiting. This physical simultaneity is the glue that creates the equality in $L = \lambda W$.

Meanwhile, the queuing theorists have extended their mathematics in important new directions such as $H = \lambda G$, and have made connections to other forms of stochastic systems and mathematical structures.

Within hands-on practice, operations management and computer architecture have led the way. Most of this seems to have been in the last 20 years of the 50 since Little (1961). The amount of good practice still appears to be increasing. Little's Law can usually be expressed in words that are adapted to be descriptive of the task at hand, e.g., scheduling MDs to treat patients in a hospital emergency department. Such developments make the ideas easier to implement and help explain why Little's Law has become well known over the past 50 years.

4.3.2. Looking Ahead. In terms of practice, $H = \lambda G$ seems full of potential. The model says that in an ongoing queuing process with items indexed by i, instead of each one merely countable, it can be assigned its own unique numerical value, e.g., cost or profit. The ensuing averages have new meaning, The applications await someone's imagination.

A favorite thought of the author is that LL could serve senior levels of management concerned with ongoing operations, e. g., the senior manager of a pharmaceutical firm's new product development program. Managers of ongoing operations collect, or are given, piles of data. LL relates three quite different average statistics $\{L, \lambda, W\}$. Each of these is actually a measure of effectiveness of the process: λ is throughput, the average rate at which "items" are passing through "the system." L is the average number of items in the system at an arbitrary time. W is the average time spent in the system by an item. These are all quite different and measure different aspects of the process, and yet are tied together by LL. We argue that managers should seriously consider collecting and displaying the LL measures for whatever stream of "items" they are managing.

If something is amiss, most likely it will show up in one or more of the three measures. We cannot, a priori, tell the manager how to fix a problem, but all three measures are interlocked by LL. If the manager fixes a problem, it is likely to show in one or more of the measures. Managers should also note the phenomenon that we have called "queue-related overhead." Large average queues tend to generate inefficiencies having nothing to do with LL per se, but LL helps measure them. Little's Law does valuable work on the factory floor but might be ready for new challenges.

Probably the biggest future winners have yet to be conceived.

5. A Note of Personal History

[*This is an updated version of a section originally appearing in Little and Graves* (2008).]

How did a sensible young OR Ph.D. like me get involved in an unlikely field like this? From 1957–1962, I taught operations research at the Case Institute of Technology in Cleveland (now Case Western Reserve University). I was asked to teach a course on queuing. OK. Initially I used my own notes, but when Morse (1958) came out, I used his book extensively. (I had been Morse's doctoral student but had done my thesis on optimal operation of hydroelectric systems.) At Case, queuing was taken by most of the OR graduate students and, indeed, one of these, Ron Wolff, went on to become a first-class queuing theorist at Berkeley (Wolff 1989).

One year the class was at the point when we had done the basic Poisson-exponential queue and moved through multiserver queues, and some other general cases. I remarked, as many before and after me probably have (and Morse does in his book), that the often reappearing formula $L = \lambda W$ seemed very general. In addition, I gave the heuristic proof that is essentially Figure 2 of Little and Graves (2008). After class I was talking to a number of students and one of them (Sid Hess) asked, "How hard would it be to prove it in general?" On the spur of the moment, I obligingly said, "I guess it shouldn't be too hard." Famous last words. Sid replied, "Then you should do it!"

The remark stuck in my mind and I started to think about the question from time to time. Clearly there was something fundamental going on, since, when you draw the picture you don't really seem to need any detailed assumptions about interarrival times, service times, number of servers, order of service, and all the other ingredients that go into the standard queuing models I had been teaching. You only seemed to need a process that goes up and down in unit amounts and some guarantee of steady state and conservation of items. In addition, because I could see there were end effects in the picture, there needed to be a way to get rid of them in the limit. It seemed to me I was in the general arena of stationary stochastic processes. I am not a mathematician by training, and so I bought copies of measure-theoretic stochastic process books like Doob (1953), which mentioned stationary processes and ergodic theorems.

My family's habit at the time was to go to Nantucket in the summer where my wife's family had a small summer house. We would load our children into a station wagon, drive to Woods Hole, take the ferry, and spend a couple months away from the world. Since the beach was our babysitter, I was able to split off solid blocks of time to work on research as a good assistant professor should. I always brought a pile of books and projects with me.

$L = \lambda W$ was one of them. I soon ran into problems that required more than looking up theorems in my new books, but I worked out approaches to the roadblocks and eventually wrote everything up, giving it my best shot. I sent the paper off to *Operations Research.* It was accepted!

Nevertheless, I had learned my lesson. I decided that Borel fields and metric transitivity were not going to be my career and retired from queuing. That was in 1961. My retirement held until 2004 when I was accosted by e-mail and in person at an INFORMS meeting by Tim Lowe to write a chapter on "Little's Law" for an OM book. Even then, as he will tell you, I resisted re-entrapment, saying, "I don't know anything about OM and I haven't looked at $L = \lambda W$ for 40 years." Being always susceptible to a new challenge and, more importantly, thanks to much help from Steve Graves, who really does know OM, I took a run at holding up my end of the chapter. It was a wonderful experience, and I learned much. So I started to look around for ways to use Little's Law in marketing and gave a few talks that were embellishments of the supermarket application in Little and Graves (2008). Interestingly, perhaps because of our chapter, my e-mail and telephone traffic about Little's Law picked up and I had a few visitors.

In a piece of remarkable serendipity, I received an e-mail in December 2009 from Ron Wolff, with whom I had had no contact for years, saying that he had been asked by John Wiley to write an article on "Little's Law and Related Results" for the *Wiley Encyclopedia of Operations Research and Management Science*. In February 2010, I suddenly realized that 2011 would be the 50th anniversary of Little (1961) and mentioned it to Steve Graves, who promptly told David Simchi-Levi, editor of *Operations Research*, who then asked me whether I would like to do a retrospective piece for the journal to coincide with the anniversary. I quickly agreed. Ron and Steve have since been trusted advisors and the readers of my drafts. I owe them many thanks but remain responsible for whatever is written here.

References

Amin, M., M. Mabe. 2000. *Perspectives in Publishing.* Elsevier, Oxford, UK.

Bertsimas, D., D. Nakazato. 1995. The distributional Little's Law and its applications. *Oper. Res.* **43**(2) 298–310.

Brumelle, S. 1971. On the relation between customer and time averages in queues. *J. Appl. Probab.* **8** 508–520.

Cachon, G. P., C. Terwiesch. 2004. *Matching Supply with Demand: An Introduction to Operations Management*. McGraw-Hill Custom Publishing, New York.

Chhajed, D., T. J. Lowe, eds. 2008. *Building Intuition: Insights from Basic Operations Management Models and Principles*. Springer Science + Business Media LLC, New York.

Doob, J. L. 1953. *Stochastic Processes*. John Wiley & Sons, New York.

El-Taha, M., S. Stidham. 1999. *Sample-Path Analysis of Queueing Systems*. Kluwer Academic Publishers, Boston.

Flynn, David. 2010. Personal communication (David Flynn, Fusion-io, Salt Lake City, UT).

Franken, P., D. Konig, U. Arndt, V. Schmidt. 1982. *Queues and Point Processes*. John Wiley, New York.

George, M. L. 2002. *Lean Six Sigma: Combining Six Sigma Quality with Lean Speed.* McGraw-Hill, New York.

Glynn, P. W., W. Whitt. 1989. Extensions of the queueing relations $L = \lambda W$ and $H = \lambda G$. *Oper. Res.* **37**(4) 634–644.

Gunther, N. 2010. *Analyzing Computer System Performance with Perl::PDQ.* Springer-Verlag, Heidelberg, Germany.

Haji, R., G. F. Newell. 1971. A relation between stationary queue and waiting time distributions. *J. Appl. Probab.* **8** 617–620.

Harris, M. D. 2010. Little's Law: The science behind proper staffing. *Emergency Physicians Monthly* (Feb.).

Heyman, D. P., S. Stidham Jr. 1980. The relation between customer and time averages in queues. *Oper. Res.* **28**(4) 983–994.

Hopp, W. J., M. L. Spearman. 2000. *Factory Physics: Foundations of Manufacturing Management*, 2nd ed. Irwin/McGraw-Hill, New York.

Lazowska, E. D., J. Zahorjan, G. S. Graham, K. C. Sevckik. 1984. *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*. Prentice Hall, Englewood Cliffs, NJ.

Little, J. D. C. 1961. A proof for the queuing formula: $L = \lambda W$. *Oper. Res.* **9**(3) 383–387.

Little, J. D. C., S. C. Graves. 2008. Little's Law. D. Chhajed, T. J. Lowe, eds. *Building Intuition: Insights from Basic Operations Management Models and Principles.* Springer Science + Business Media LLC, New York.

Lovejoy, W. S., J. S. Desmond. 2011. Little's Law flow analysis of observation unit impact and sizing. *Acad. Emergency Medicine* **18** 1–7.

Miyazawa, M. 1994. Rate conservation laws: A survey. *Queueing Systems: Theory Appl.* **15**(1–4) 1–58.

Morse, P. M. 1958. *Queues, Inventories and Maintenance*. John Wiley & Sons, New York.

Sigman, K. 1991. A note on a sample-path rate conservation law and its relationship to $H = \lambda G$. *Adv. Appl. Probab.* **23** 662–665.

Stidham, S., Jr. 1972. $L = \lambda W$: A discounted analogue and a new proof. *Oper. Res.* **20**(6) 1115–1126.

Stidham, S., Jr. 1974. A last word on $L = \lambda W$. *Oper. Res.* **22**(2) 417–421.

Stidham, S., Jr. 2002. Analysis, design, and control of queueing systems. *Oper. Res.* **50**(1) 197–216.

Whitt, W. 1991. A review of $L = \lambda W$ and extensions. *Queueing Systems* **9**(3) 235–268.

Wolff, R. W. 1989. *Stochastic Modeling and the Theory of Queues*. Prentice-Hall, Inc., Englewood Cliffs, NJ.

Wolff, R. W. 2011. Little's Law and related results. J. J. Cochran, ed. *Wiley Encyclopedia of Operations Research and Management Science*, Vol. 4. John Wiley & Sons, Hoboken, NJ, 2828–2841.