Traffic Engineering Techniques in Telecommunications

by: Richard Parkinson

Introduction:

The use of mathematical modeling to predict line, equipment, and staff capacities for telephone systems is an accepted technique for fine-tuning existing systems, as well as designing new ones. Through sensitivity analysis, such predictions can also provide a comprehensive overview of a particular design.

Most available literature on traffic engineering either concentrates on obscure and complicated calculations, or alludes to formulas, but does not include them. This paper defines relevant terms, and discusses the most commonly used formulas and their applications. A knowledge of secondary school mathematics is sufficient to do the calculations in this paper.

Traffic engineering techniques are used most often to determine:

- Line and trunk quantities required for a PBX or computer
- Number of DTMF (Dual Tone Multi-frequency) registers, conference trunks, RAN (Recorded Announcement Route) trunks, etc. required
- Traffic capacity of a PBX, given the number of speech paths (simultaneous conversations) available
- Quantities, service levels, and usage of such special service trunks as foreign exchange (FX), discounted toll trunks, and tie trunks (leased lines between PBXs
- Operator staffing levels and performance predictions as well as the impact of system change on staff quantities
- Automatic call distributor (ACD) staffing and service levels

Terminology:

The terms used in traffic engineering are standard telecommunications usage. The following brief glossary lists those terms needed to understand the formulas presented in this paper.

© Copyright INFOTEL SYSTEMS CORP..

Arrival Rate: - The arrival rate is the number of calls that will arrive at a facility during a finite time period. The Greek letter lambda (λ) is generally used to represent arrival rate. The distribution of calls to a server group will vary with the source. People calling to a line group often do so at random, with each call independent of the others. This is called a Poisson arrival process and is the most common assumption used in traffic engineering for the distribution of call arrivals. (Calls to a power utility during a power failure would not be considered a Poisson distribution.)

Blocking: - Blocking occurs whenever the number of calls, in or out, exceeds the number of facilities (lines, trunks, agents, operators) available to support them. A blocked call is given a busy signal, which requires the caller to disconnect, and try again. Blocking probability is expressed as a percentage of denial, e.g. for 1 call in 100 blocked, it is expressed as P.01 (1% of the offered calls will expect to be blocked).

Centum Call Seconds (CCS): - This is a measure of telephone traffic in 100 second increments. It originated in the early days of electromechanical switching, and was developed to make the traffic volume quantities more manageable, For example:

10 minutes of traffic = 600 seconds (60x10)

$$= \frac{\textbf{600 seconds}}{\textbf{100}}$$

$$= \textbf{6 CCS}$$

Erlang: An erlang is defined as a dimensionless unit of traffic intensity. The key to this definition is that, dimensionless means no specific time period. A CCS is exactly 100 seconds, whereas an Erlang is dependent on observation time. The maximum that a facility can be in use is 100% of the time. If the observation time is 10 minutes, and the facility is in use for the full time, then that is 1 Erlang. If the observation time is 1 hour, then 1 Erlang is equal to 1 hour. This is important because certain environments, e.g. call centers, often want to staff to the busiest 10 - 15 minutes, which can be 25% higher than the hourly observation. As you will see, virtually all of the formulas use the erlang values as the traffic load for calculation. Remember a simple rule. The number of facilities required must exceed the number of Erlangs at a minimum. A one to one ratio = 100% utilization. A 3 lane highway at 100% utilization becomes a parking lot.

Holding time: Holding time is the call length, call overhead time, plus queuing time, if any. Overhead includes the activities necessary on the transmit/receive sides of the call. Outgoing calls incur different activities than do incoming calls.

It is important to know the call types and their overheads, because overhead can represent a considerable amount of time. When calculating the Erlang load for the formula you plan to use, make sure you include everything. For example if you are using a telephone bill for the source of traffic load, remember that this does not include dialing time, only time once the call is answered. If you are going to size a trunk group, you must also add the average dialling time to this value, as the trunk holding time per call includes; the dialing time, ring to answer time etc. Table 1 represents the major items:

ITEM	Outgoing call	Incoming call
Dialing time (DTMF)	1 - 7 seconds [1]	1 second [1]
Dialing time (Rotary)	5 - 12 seconds [1]	5 seconds (@ 10 pulse/sec)
Network call setup	1-3 seconds [2]	1-3 seconds
Ringing time	12 seconds (2 rings)	12 seconds (2 rings)
Operator Answer	5-8 seconds	5-8 seconds
Ringing at station	12 seconds (2 rings)	12 seconds (2 rings)
Conversation time	variable	variable

Table 1: Call Functions Possibilities

Note: 1: DTMF tones can be propagated at 100 milliseconds per digit, typical with speed dialling, however when humans dial they are slower, or when dialing an 800 - name number, they have think time to find the L = 5. For rotary dialling, numbers are typically outpulsed at 10 pulses per second. Using 5 as the average digit dialed, that represents .5 seconds per digit, therefore a 10 digit number has a minimum dial time of 5 seconds, even when a switch is sending it.
Note: 2: Network call set-up time assumes the use of SS7 (Signalling System 7), for a domestic call. International calls may take longer. Also if users are charging to a telco credit card, add 3-5 seconds for that process.

Holding Time Distribution: - The holding time is generally considered the talk time, but for trunk sizing should include the items in table 1, and for ACD agents needs to include holding time in the queue, but not post work time (more on this later). Voice calls typically have a distribution pattern known as an exponential distribution. Typically more calls are shorter than the average, than greater than the average. Figure 1 shows an example of this distribution.

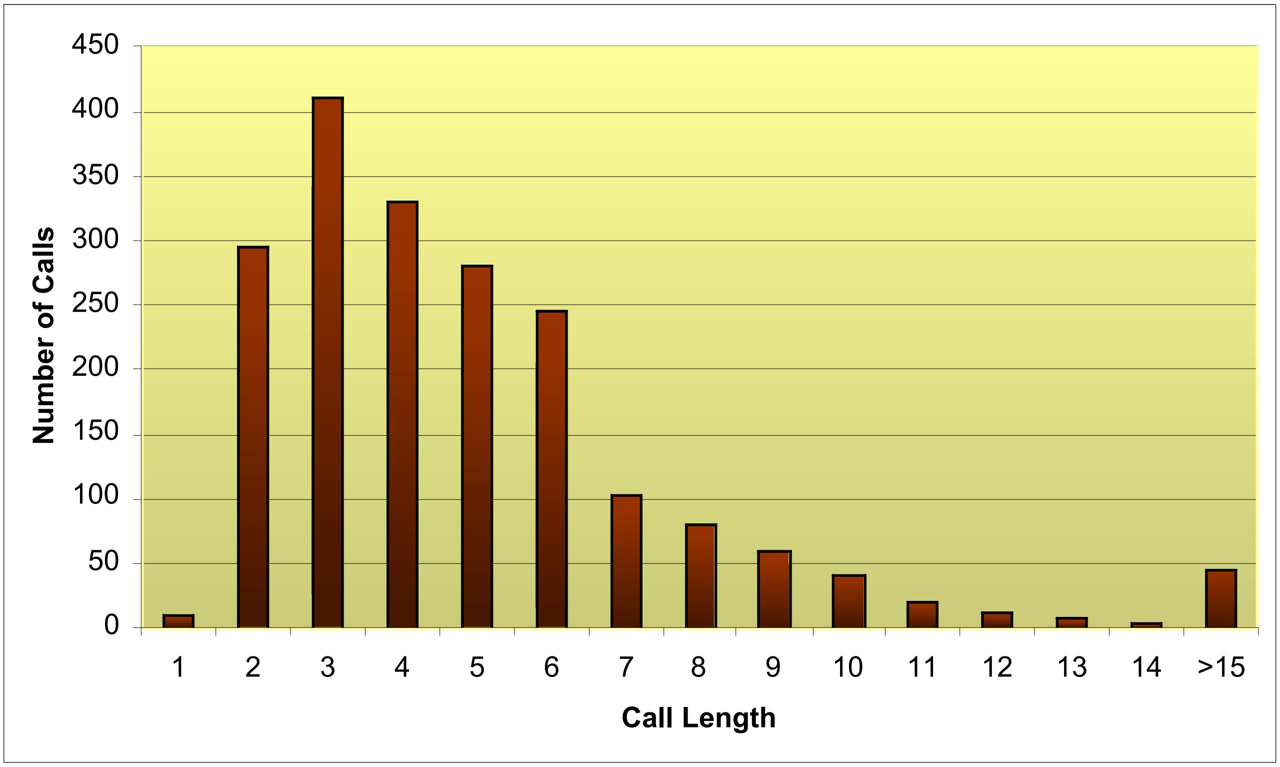


Figure: 1 Call Holding Time Distribution (exponentially distributed)

Queuing: - Queuing is waiting in a holding facility until a server becomes available. When, for example, an ACD (Automatic Call Distributor has more lines than agents (e.g., 50 lines serving 42 agents) and all agents are busy, the extra lines become the holding facility.

Servers: - Servers is a generic name for lines, trunks, registers, or people, which receive or originate calls. In many systems, a source encounters two or more server groups. For example, a PBX trunk provides access to the operator. When an attempt is made to transmit a call to a server, the source will either be blocked or queued.

Sources: - Sources originate calls to a system; their number influencing the formula chosen to calculate the effective grade of service of a line group. Sources are considered finite if they number less than 100, and infinite if greater than 100, or the source to server ratio is greater than 8 to 1.

Mathematical Notation

Most symbols used in the formulas for this paper are standard notation. Variables can be assigned any symbol; however, for ease of understanding, characters derived from the variable (e.g., E for erlangs) are used.

Base e Logarithm (e): - This is the base of the natural logarithm 2.71828. A logarithm is the exponent or power to which a base number must be raised to yield a given number. For example, in **base 10**, $10^2 = 100$, the logarithm of 100, to the base 10, is 2, written as $\log_{10} 100 = 2$

Logarithms that employ the **base e**, are referred to as natural or Napierian. In **base e**, the logarithm of 100, written $\log_e 100$, equals 4.605170186

Factorial (!): - This symbol denotes the value of an integer multiplied by all integers below it:

5! = 5x4x3x2x1 = 120

Lambda (λ): - This represents the arrival rate per a unit of time:

e.g. 120 calls per hour, 2 calls per minute

Lemniscate (∞): - The lemniscate denotes infinity.

Mean (x): - The arithmetic mean is the average of a set of numbers and is commonly shown as this formula; numerical example also shown.

Formula	**Example**
$\overline{X} = \sum_{k=1}^{N} X_k * P_k$	$\overline{X} = (4x.25)+(5x.25)+(8x.25)+(4x.25) = 6.5$

Mu (μ): - This represents a service rate, i.e. the number of events that can be handled in a unit of time. The time to service a specific event is referred to as the service time. The arrival rate, divided by the service rate provides total system utilization, often represented by ρ (rho).

$$\frac{\lambda}{\mu} = \rho$$

Sigma (Σ): This symbol indicates the summation of an equation from the value under the symbol to the value above the symbol:

$$\mathbf{P} = \sum_{x=0}^{N} \mathbf{Ex} \quad = \mathbf{E_0+E_1+E_2...E_N}$$

Standard Deviation (s): - Standard deviation represents the degree of dispersion on either side of the mean, and is used to determine how widespread the values are. The formula typically used to calculate standard deviation is:

$$s = \sqrt{\frac{\sum(x-\bar{x})^2}{N}}$$

Thus, for the values of 4, 5, 8, and 9, x = 6.5, and the standard deviation is:

$$s = \sqrt{\frac{(4-6.5)^2+(5-6.5)^2+(8-6.5)^2+(9-6.5)^2}{4}}$$

$$s = \sqrt{\frac{(6.25)+(2.25)+(2.25)+(6.25)}{4}}$$

$$s = \sqrt{\frac{17}{4}} \qquad s = 2.06$$

The preceding formula applies if the values of X represent the total population.

X to the power N(X^N): This notation indicates the value of X to the power of N, or X times itself N times:

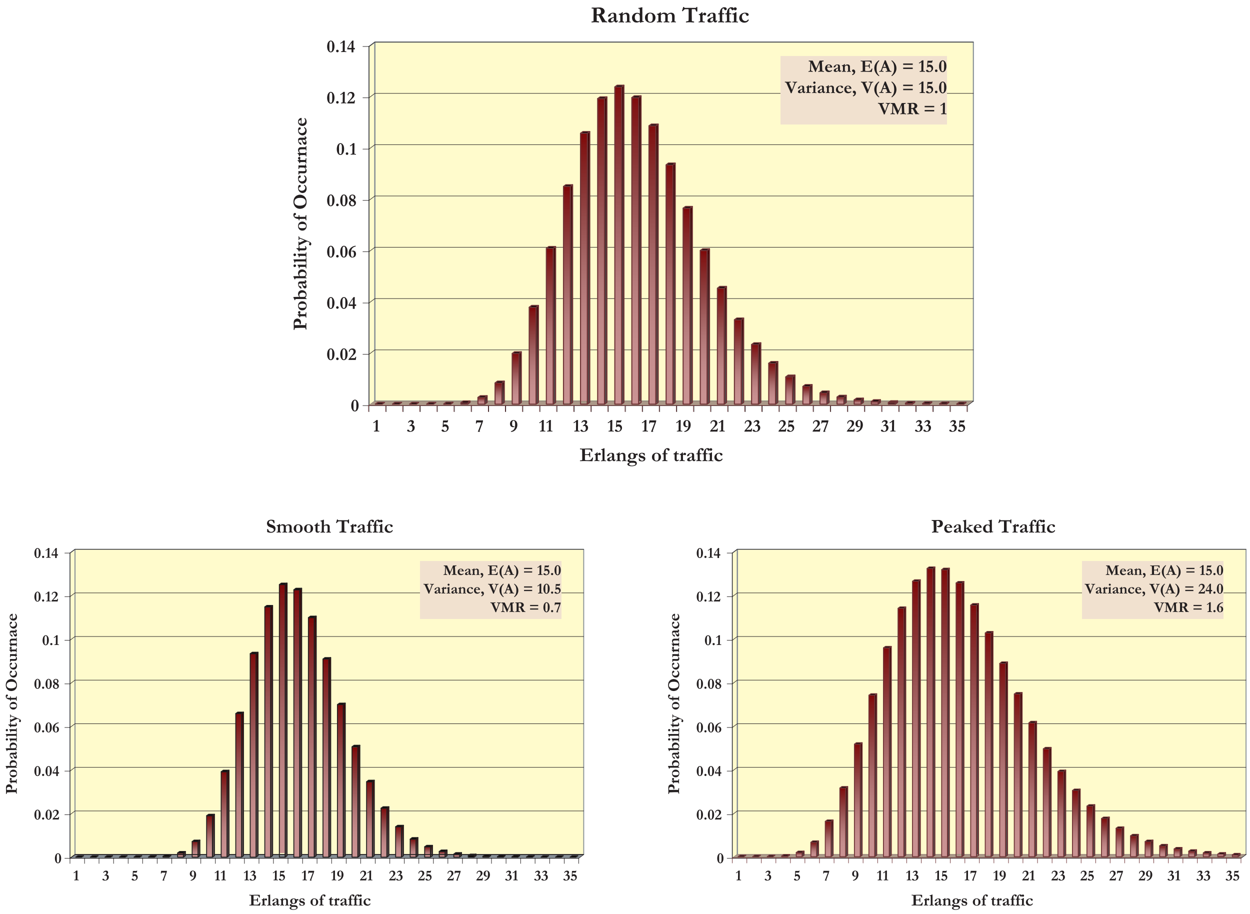
$$\text{If } X=2 \text{ and } N=4, \text{ then } X^N = 2^4 = 2 \times 2 \times 2 \times 2 = 16$$

Variance (σ): - The variance is the square of the standard deviation. As a measure of dispersion, it is useful in many calculations.

Variance-to-Mean Ratio (VMR): - The variance-to-mean ratio measures traffic peakedness (see Figure 2). It is particularly useful for calculating skewness of nonrandom traffic (e.g., overflow route traffic) and is obtained with the following formula:

$$\mathbf{VMR} = \frac{\mathbf{V(A)}}{\mathbf{E(A)}} \quad \begin{matrix}\textbf{(Variance of the offered load)} \\ \textbf{(Average of the offered load)}\end{matrix}$$

The issue here, to be explored later, is the need to know more than just the **average traffic load**. Also needed is the **variance**, and the resulting **Variance to Mean Ratio (VMR)**, to choose the right traffic model. The three traffic distributions shown below, will be expanded on later.



Random Traffic

Smooth Traffic

Peaked Traffic

Figure: 2 Traffic Distribution Curves

TRAFFIC THEORY:

In voice or data communications, sources generate calls to a facility, or servers. When a call arrives at a group of servers, and one is available, the call is handled. When all servers are busy (depending on system design), the caller can:

- Receive a busy signal requiring the caller to hang up and try later
- Automatically route to another facility
- Queue (wait) in a holding facility until a server is available
- Queue for some tolerable time interval, then disconnect if not served

The disposition of a call when all servers are busy has the greatest influence on which formula to use. The formulas explained in subsequent sections therefore cover those situations in which sources are either blocked or queued, when all servers are busy. For both cases, the formulas generally calculate the probability of all servers being busy. Because the sum of all probabilities is one, the probability of being served is; one, minus the probability of all servers being busy. For example, if the probability of all servers being busy simultaneously is 10 percent, then the probability of being served is 1 minus 0.10, which equals 0.90, or 90 percent. Thus, 10 percent of the callers would encounter a busy signal, and 90 percent would be served.

The validity of the formula output parallels the accuracy of the input data and the fit of the assumptions. Given that assumptions are not always accurate, the development of very sophisticated techniques to improve accuracy becomes academic, especially for good grades of service—the goal of most systems designers.

The remainder of this paper discusses the formulas in common use, their assumptions and applications.

FORMULAS

The factors dictating the formula that best applies to a given situation include **source population** (finite or infinite), **holding time distribution** (constant or exponential), and **call disposition** when all servers are busy (blocked or queued). Figure 3 provides a basic decision tree. The three most widely used formula types are:

- **blocking** formulas that assume **infinite sources**,
- **blocking** formulas that assume **finite sources**, and
- **delay formulas.**

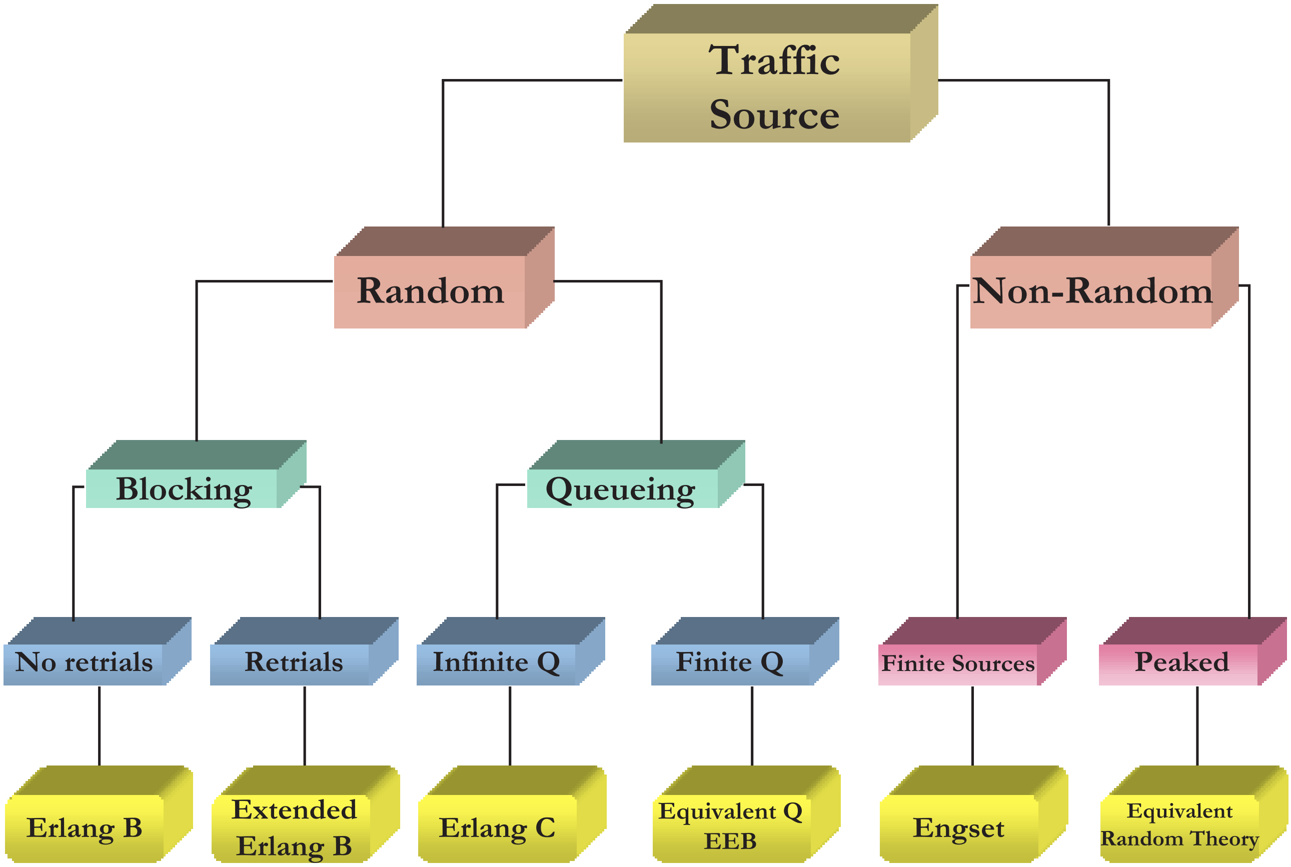


Figure: 3 Formula Use Decision Tree

Although an in-depth discussion of each formula is beyond the scope of this paper, formulas are provided along with at least one solved example to enable the reader to recreate the results from any given input. The Extended Erlang B (EEB), and Erlang C formulas are emphasized because they have fairly universal application.

Before starting with specific telephone models, it is worth reviewing the Poisson formula.

Poisson: The Poisson formula, developed by the French mathematician Siméon-Denis Poisson, (1781-1840) states that for nonoverlapping events, arriving at an average rate λ, the probability of **x** arrivals in time **t** equals:

$$P(k) = \frac{(E(n)*t)^{*k}\; e^{-E(n)*t}}{k!}$$

Where: P(k) = Probability of arrivals
E(n) = Average Arrival Rate
t = Average Holding Time
e = 2.71828

This formula allows calculation of the probability of having n arrivals, during some time interval. e.g. 1 second, 1 minute, etc. The graph below shows the probability of arrivals from 0 - 13, with an average arrival rate of 4. Notice how the average can be deceiving, i.e. the probability of having more than 4 arrivals is 1-(.018+.073+.147+.195+.195) = 1-(.628), or 37.2%

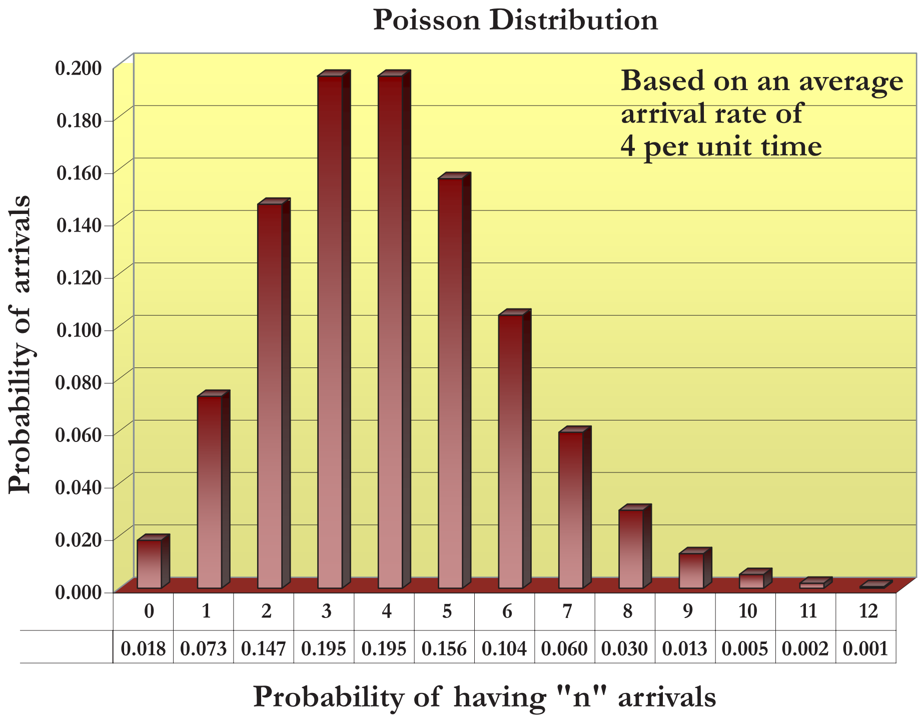


Figure: 4 Poisson Distribution

An example of the value of this calculation came when a vendor was proposing a PBX system capable of 14,000 Busy Hour Call Attempts (BHCA) for a requirement of 11,000 BHCAs, or approx. 79% utilization at installation. 14,000 BHCA represents 3.88 call arrivals per second. The question here then was whether this slim margin was workable. Using this Poisson formula, with an arrival rate of 4 per second, it could be shown that there was a 37.2% potential that the arrival rate would exceed the 100% utilization of the system in the busy hour. This was the end of this vendor from consideration.

Blocking Formulas—Infinite Sources

There are three formulas in this group; Molina, Erlang B, and Extended Erlang B (EEB), which are used to calculate line quantities on such telephone equipment as PBXs and ACDs. The assumptions shared are **infinite sources**, **constant or exponential holding time**, and **lost calls are blocked** (the calls receive a busy signal).

The formulas differ in what happens to the calls blocked; although, for good grades of service (i.e., a less than 5% probability of being blocked), the difference in results is small. In North America, a formula known as **Molina** was once popular by telephone companies, but has since fallen out of favor. **Erlang B** is more prevalent today when blocked calls overflow, or disappear, or **Extended Erlang B**, a modification of Erlang B, which assumes that some of the callers who are blocked, will retry. The following sections discuss each formula and its predominant application.

Molina: Though commonly called the Poisson formula, Molina is actually an application of Poisson's formula. It was developed by AC. Molina, a researcher for Bell Laboratories during the 1920s.
Given that this formula overstates the number of facilities required, its error rate is too high, therefore it is no longer used, so it will not be discussed further.

Erlang B: The Erlang B formula was developed by Agner Krarup Erlang (1878 - 1929), a Danish mathematician who is credited with the first systematic study of telephone traffic characteristics. This formula is recommended for use by the ITU-T in Recommendation E.520.

Erlang B is used primarily for determining trunk quantities in first-choice trunk groups in which, if all trunks are busy, a call overflows to another group, or never returns. It shares with Molina the assumptions of **infinite sources** and a **holding time** distribution that can be **constant or exponential**. The major difference from Molina is that in Erlang B's assumption, lost calls leave the system (overflow, or die).

The Erlang B formula is:

$$\mathbf{Pb} = \frac{\dfrac{A^N}{N!}}{\displaystyle\sum_{X=0}^{N} \frac{A^X}{X!}}$$

Where:

A = Offered Traffic

N = Number of servers (lines)

Pb = Probability of blocking

Assumptions:

- Poisson arrivals (random traffic).
- Call holding times are of fixed length or are exponentially distributed.
- Blocked calls are cleared.

Inserting the values: A=3 and N=6, yields a probability of blocking (Pb) of .0522, implying 5% of callers would be blocked. Generally for end user access lines or PBX trunks, 1% blocking is considered an optimum design target.

Extended Erlang B: - The **EEB** formula is an enhancement to the accuracy of Erlang B when a percentage of callers try again on receipt of a busy signal. It was developed in the mid-1970s by James Jewitt and Jacqueline Shrago—principals in Telco Research Corporation—after considerable research comparing various techniques available with actual user data and simulation modeling.

Of the three formulas used to calculate line quantities when callers are blocked, EEB is generally the most accurate for a wide range of configurations. It is based on the premise that in most situations, on hearing a busy signal, callers will try again. This retrial behavior increases the offered traffic. Thus, as the blocking percentage increases, so does the offered as well as carried traffic. To calculate the probability of blocking using EEB, the telecommunications manager needs the total offered erlangs, number of lines, and percentage of blocked calls that will be attempted again (0 to 100 percent). This is a flow chart of the logic behind EEB

.

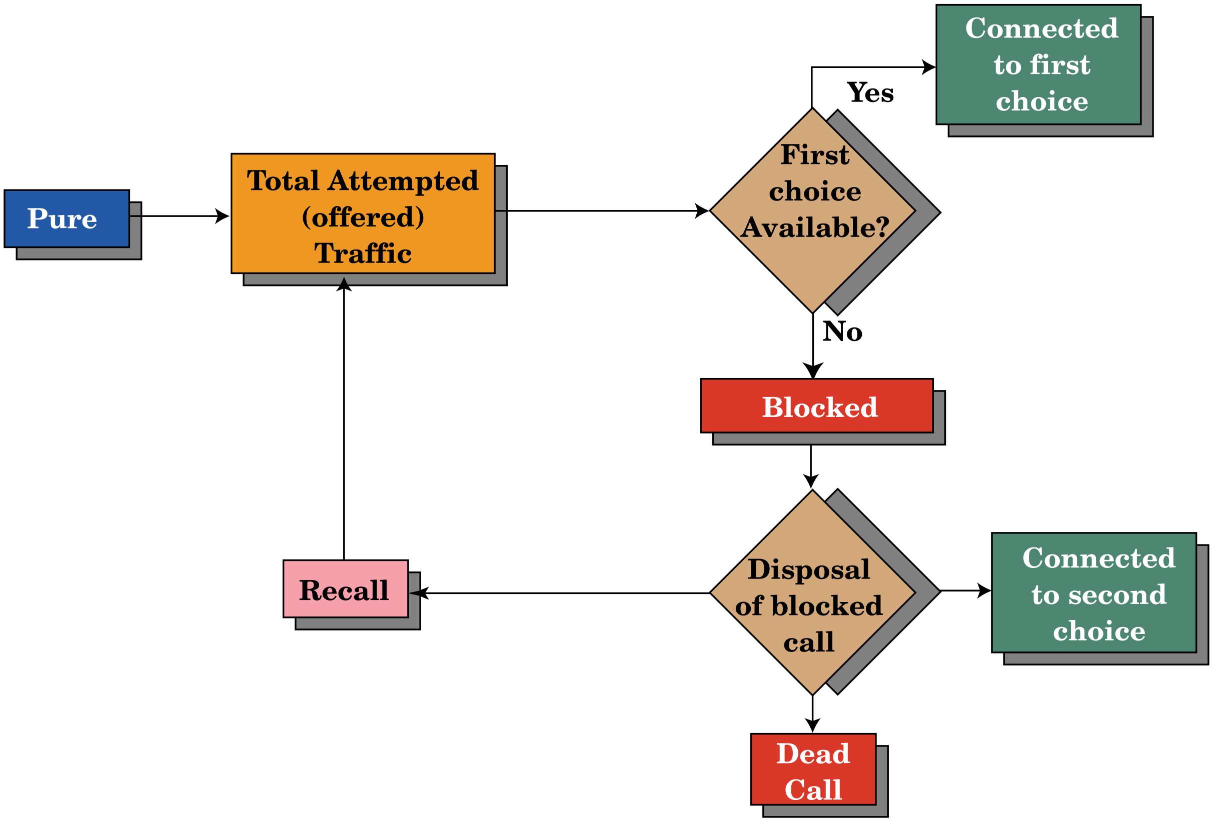

Connected to first choice
Yes
Pure
Total Attempted (offered) Traffic
First choice Available?
No
Blocked
Recall
Disposal of blocked call
Connected to second choice
Dead Call

Figure: 5 EEB Flow Chart

To use EEB perform the following steps:

1. Calculate the probability of blocking using Erlang B (e.g. A=3, N=6) using the formula on the previous page:

2. Calculate the following values as shown in figure 6:

Be = N * Pb (e.g. 3 erlangs * Pb {.0522})
B = Be * Recall factor (e.g. .1566 * .5)
C = (N-Be)+R (e.g. 3 - .1566 Erlangs + .0782 {50% of Be})
R = Be * Recall factor (e.g. .1566 * .5 = .0782)
C+B = the carried traffic, plus the traffic that never returns
N+R = the original traffic, plus the recall traffic, which becomes the new offered load in the next iteration

Offered Load: 3 Erlangs, 6 Trunks, 50% recall factor

A	N	ERL-B		Be =N*Pb	B =Be*.5	C =(N-Be)+R	R =Be*.5	C+B	N+R
Iteration	Offered Load	No. of Trunks	Pb	Blocked Erlangs	Overflow Traffic	Carried Traffic	Recall Traffic	Carried & Overflow	Recall & Orig. Load
1	3.0000	6	.0522	.1566	.0782	2.8435	.0782	2.9218	3.0782
2	3.0782	6	.0565	.1740	.0870	2.9042	.0870	2.9912	3.0870
3	3.0870	6	.0570	.1760	.0880	2.9110	.0880	2.9990	3.0880
4	3.0880	6	.0571	.1762	.0881	2.9117	.0881	2.9999	3.0881
5	3.0881	6	.0571	.1764	.0882	2.9118	.0882	3.0000	3.0882

Figure: 6 EEB Example

Using the values of 3 for the offered load (A), and 6 for the number of lines (N), and a 50% recall factor, you can see that EEB improves the accuracy over Erlang B, when there are retrials. However for good grades of service, i.e. P05 or better, the accuracy improvement is minimal. An often asked question is how do you determine the recall factor. If you don't know, 50% is a reasonable guess. If, on the other hand you know that virtually every blocked call will retry, e.g. calling to see the status of your income tax refund, then for the incoming tax lines, we should assume close to 100% recall.

Although EEB is the most complicated of the three formulas discussed, it is also sensitive to input data accuracy. If the three required variables (i.e., total busy hour erlangs, number of servers, and percentage of retries) are acquired from realistic statistics, EEB will provide reasonable results. Figure 7 compares various erlangs and line quantities and the probabilities of loss using Erlang B, EEB with 50% and 100% recall factors. You can see that for low blocking, there isn't much difference between them.

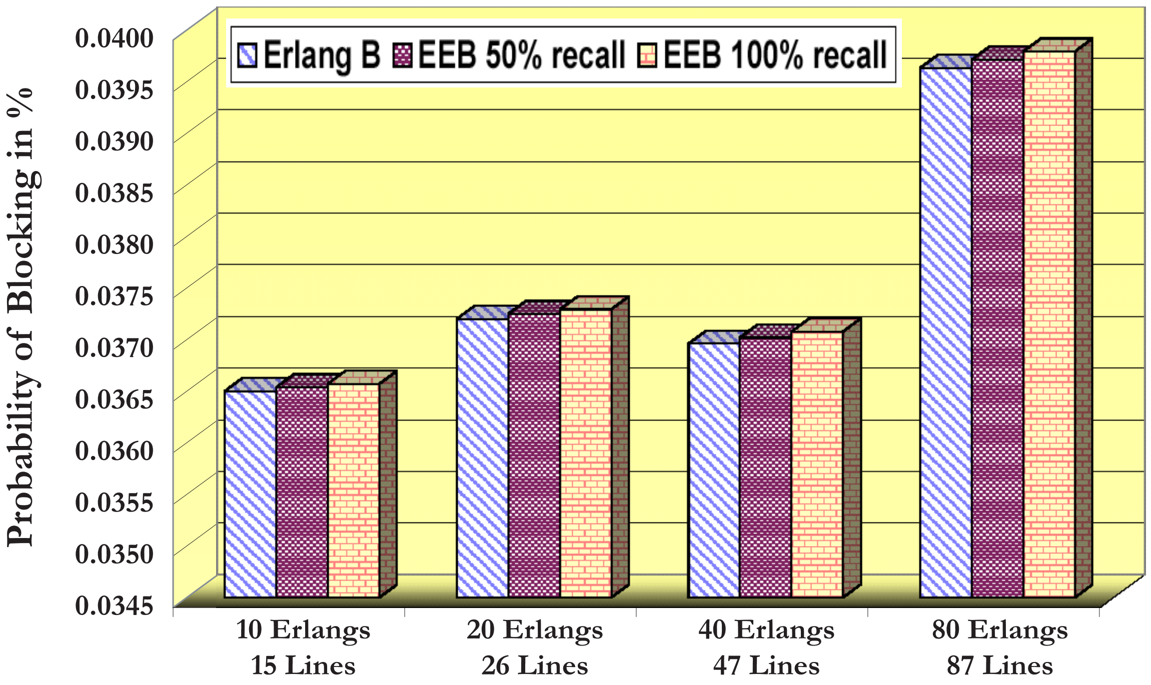


Figure: 7 Comparison of Erlang B to EEB

In summary, the disposition of lost calls is the most important factor in determining which of the three formulas to use for infinite sources. EEB gives the most consistent and accurate results when there is retrial behavior. When there are no retrials, Erlang B should be used.

Equivalent Random Theory: - Erlang B and EEB are very good at sizing trunk groups when their assumptions are valid, i.e. infinite sources, Poisson (random) arrival rate, and blocked calls are cleared (Erlang-B) or retry (EEB). However there is one traffic type where these two models will underestimate the number of trunks required. When traffic can't be served by the initial trunk group attempted, some networks support alternative trunk groups as overflow groups. The design problem is that the traffic to the overflow group is no longer random, causing peaks of activity. This is analogous to the side roads of a freeway, suddenly having to carry more cars due to congestion on the freeway. This overflow characteristic is no longer random.

To adjust for this, two Bell Laboratories researchers, R.I. Wilkinson and S.R. Neal developed in the mid 1950s, the concept know as **Equivalent Random Theory (ERT)**. The basis of ERT is that peaked traffic can be modelled as overflow traffic from a trunk group that has been offered random traffic. What is needed then is to estimate the original offered traffic from the overflow traffic erlangs.

The problem is to determine the number of trunks required, when the traffic is peaked, i.e. a Variance To Mean Ratio (VMR) greater than 1. The solution was a model better than Erlang B, designed to solve trunk sizing when the traffic is random (VMR=1).

Consider figure 8 below. Random traffic is offered to a first attempt trunk group, some or most is carried, and the rest overflows to the overflow group. What is known is the overflowed amount, but what is needed to be known is the original offered load

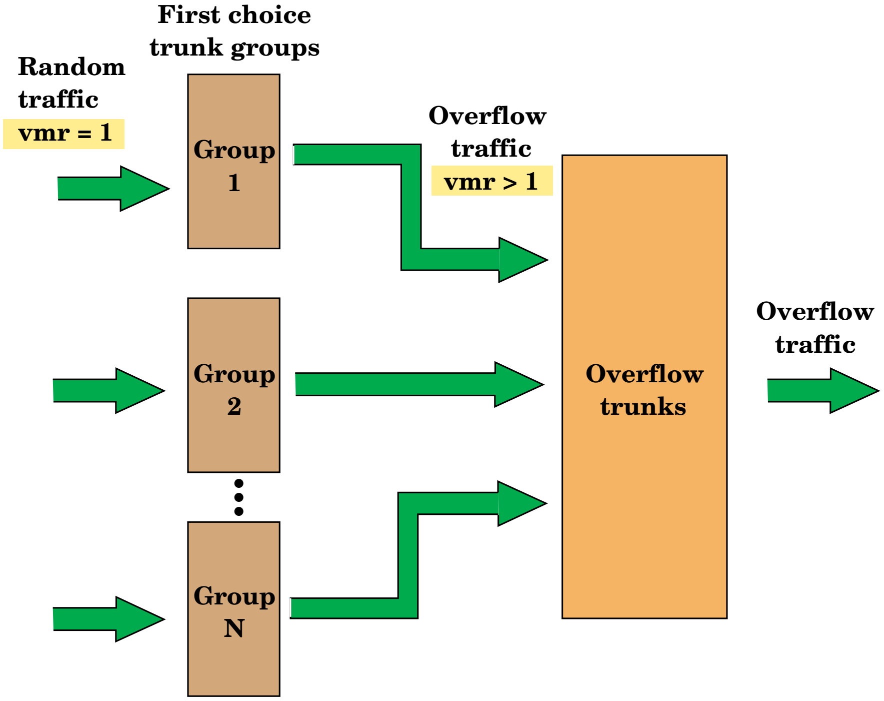


Figure: 8 ERT Concept Flow Chart

The Equivalent Random Theory model requires the VMR of the offered traffic to be specified. If the offered traffic is overflow from a trunk group that has been sized using the Erlang B model, the following relationships apply:

$$\text{Avg Overflow} = \mathbf{E(A,N)*A}$$
$$\text{Variance of overflow} = \mathbf{Avg * \left(1\text{-}Avg+ \frac{A}{N+1+Avg\text{-}A}\right)}$$

Where:

E(A,N)= Probability of blocking from Erlang B model
A = Offered Traffic
N = Number of Trunks
AVG = Avg of Overflow Traffic

For example if:

A = 5 Erlangs
N = 4 Trunks

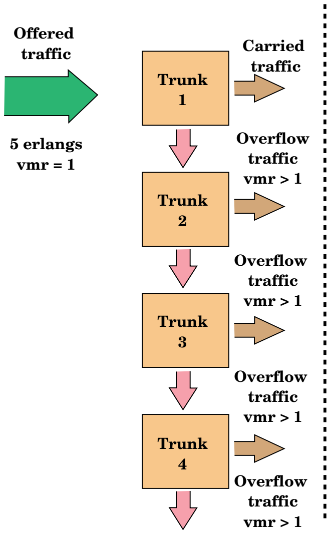
- From Erlang B model probability of blocking is .398
- Average of overflow traffic is 5*.398 = 1.99

which yields:

$$\textbf{Variance of overflow traffic} = 1.99^{*}\left(1\text{- }1.99 + \frac{5}{4+1+1.99-5}\right) = 3.03$$

$$\textbf{VMR} = \frac{3.03}{1.99} = 1.52$$

Now consider this example graphically:



Consider this example:

5 Erlangs is presented to a 4 trunk, first attempt group. The previous formulas allow you to callculate the VMR and overflow for each trunk. Note that the overflow from the firt 2 trunks is 3.38 Erlangs, and the VMR is 1.25.

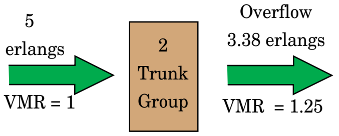
Traffic Summary

Trunk	Offered	Prob. of blocking	Overflow
Trunk 1	5 erl vmr = 1	.83	4.17 erl vmr = 1.12
Trunk 2	4.17 erl vmr = 1.12	.68	3.38 erl vmr = 1.25
Trunk 3	3.38 erl vmr = 1.25	.53	2.65 erl vmr = 1.39
Trunk 4	2.65 erl vmr = 1.39	.4	1.99 erl vmr = 1.52

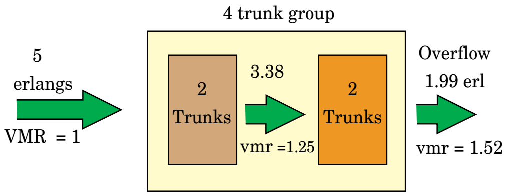
Figure: 9 Graphical logic of ERT

Then consider:

From the previous example we see that a load of 3.38 erlangs with a VMR of 1.25 is the overflow when 5 erlangs with VMR = 1 are applied to a 2 trunk group.



We can therefore use the erlang B model as follows :



Remember
Blocking =
overflow / offered traffic

Probability of blocking
1.99 / 3.38 = .59

Note: The 3.38 Erlangs & vmr=1.25 applied to a 2 trunk group, gives the same overflow as a 5 Erlang & vmr=1 load to a 4 trunk group. Given the overflow traffic (e.g. 3.38) as original offered traffic, what we want to calculate is the the equivalent original offered load and trunk groupsize. Once known we can use Erlang B to calculate blocking.

Figure: 10 Graphical logic of ERT (Continued)

The equations you need to solve are:

Given :

A = **Peaked load (erlangs)**

Z = **VMR**

Aeq = $V + Z * (Z-1) * (2 + G^{B})$

G = $(2.36 * Z - 2.17) \log (1 + (Z -1) / (A * (2 + 1.5)))$

B = $Z / (1.5 * A +2 * Z - 1.3)$

V = $A * Z$

Teq = $Aeq * (A + Z) / (A + Z -1) - A - 1$

Aeq = Equivalent random load
V = Variance of peaked load
G = Empirically derived parameter
B = Empirically derived parameter
Teq = Equivalent trunk group

Reference: ITU-T Recommendation E.524

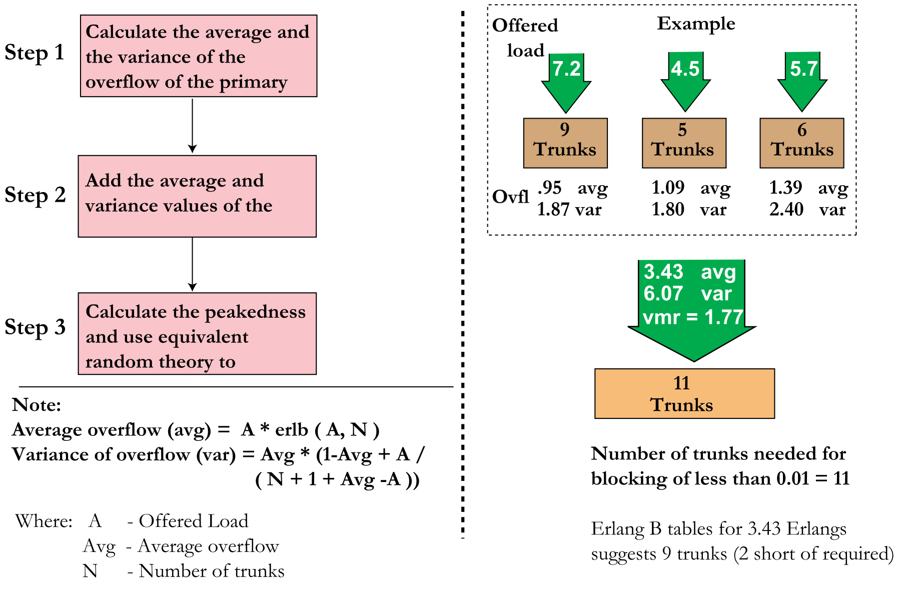
These are numerical values for these equations as an example:

A = 10 erlangs
Z = 2
V = 10 * 2 = 20
G = (2.36 * 2 - 2.17) log (1+(2-1) / (10*(2+1.5))) = .031
B = 2 / (1.5 * 10+2 * 2 - 1.3) = .113
Aeq = $20 + 2 * (2-1) * (2+ .031^{.113}) = 25.352$
Teq = 25.352 * (10+2) / (10+2 - 1) - 10 - 1 = 16.66

A = Offered load
Z = Variance to Mean Ration (VMR)
V = Variance of peaked load
G = Empirically derived parameter
B = Empirically derived parameter
Aeq = Equivalent random load
Teq = Equivalent trunk group

Note that the equivalent trunk value (Teq) is not an integer. This is solved in practice by rounding Teq to an integer value and adjusting Aeq or by using an alternate form of the Erlang B model that allows non integer trunk values.

Finally, this last figure shows some results


Step 1
Calculate the average and the variance of the overflow of the primary
Step 2
Add the avergage and variance values of the
Step 3
Calculate the peakedness and use equivalent random theory to
Offered load
Example
7.2
4.5
5.7
9 Trunks
5 Trunks
6 Trunks
Ovfl
.95 avg
1.87 var
1.09 avg
1.80 var
1.39 avg
2.40 var
3.43 avg
6.07 var
vmr = 1.77
11 Trunks

Note:
Average overflow (avg) = A * erlb (A, N)
Variance of overflow (var) = Avg * (1-Avg + A / (N + 1 + Avg -A))

Where: A - Offered Load
Avg - Average overflow
N - Number of trunks

Number of trunks needed for blocking of less than 0.01 = 11

Erlang B tables for 3.43 Erlangs suggests 9 trunks (2 short of required)

NOTE: Assumes we are designing just the overflow trunk group, with no knowledge of the original offered load

Figure: 11 Numerical Example using ERT

The interpretation of these results is that ERT suggests 2 additional trunks over what Erlang-B would suggest, given the 3.43 Erlangs in isolation. However to calculate these numbers, you either need tables, or a program, which is available from ***http://www.infotel-systems.com***. Lastly, the need for ERT today is waning, given that when trunk groups are added, they are generally in groups of 24/30, 480/672, or higher, so a popular solution is to simply over engineer initially, then adjust once firm data is known.

Blocking Formulas—Finite Sources

Finite source formulas have fewer applications than the infinite formulas. The **Engset** and **Binomial** formulas and their applications are presented in the following subsections along with an example worked out for each.

Engset: - The Engset formula, named for its developer, Tore Olaus Engset, is used for **finite sources** when **blocked calls are cleared** (overflow or die). When the number of sources is small, i.e. the number of sources to facilities is less than 8 to 1, an effect called **"limited source gain"** occurs. Assume a purchasing department has five employees that are constantly calling out, and collectively generate 3.75 erlangs of traffic in busy hour. Using the **infinite source** formula EEB with 50% recall, and P.01 grade of service, EEB would specify the need for 9 lines. But in this case we have a **finite source** (5 callers), so we wouldn't need more than one line per caller, or 5 lines. This is the limited source gain effect. Engset would be more appropriate for this scenario.
The required input is:

- total offered erlangs from all sources,
- the number of sources and lines, and
- the desired probability of blocking.

Solving the Engset formula involves iteration, in that to obtain the answer to this problem of blocking, the answer must already known. The user makes an initial guess of Pb, and runs the formula using that guess. The process is repeated with the guess adjusted until it, and the answer (value of Pb) converge, and are equal. The formula is:

$$\mathbf{Pb} = \frac{\left[\dfrac{(S-1)!}{N!*(S-1-N)!}\right]*M^{N}}{\displaystyle\sum_{X=0}^{N}\left[\dfrac{(S-1)!}{X!*(S-1-X)!}\right]*M^{X}}$$

Where:

A = Offered Erlangs from all sources
S = Number of Sources
N = Number of servers
Pb = Probability of blocking

$$M = \frac{A}{S-A*(1-Pb)}$$

Binomial: - The Binomial finite source formula differs from Engset in its use of traffic per source rather than total traffic from all sources. This formula also assumes that some calls hang around, (sort of a retrial equivalent assumption), and are eventually served. The formula is:

$$\mathbf{Pb} = \sum_{x=N}^{S-1} \frac{(S-1)!}{x!(S-1-X)!} A^{x}(1-A)^{(S-1-X)}$$

Where:

A = Offered Erlangs per sourc
S = Number of sources
N = Number of servers
Pb = Probability of blocking

Table 2 below, shows the blocking probabilities for both Engset and Binomial for the same number of lines, sources, and offered erlangs the same number of sources, lines, and Erlangs. Note that Binomial's "hang around" assumption, provides a higher blocking probability than Engset, albeit small.

Engset				
Number of sources	10	20	30	40
Number of lines	5	5	5	5
Total Erlangs	2	2	2	2
Probability of Blocking	.0175	.0294	.0335	.0336
Percentage Blocked	1.75	2.94	3.35	3.56

Binomial				
Number of sources	10	20	30	40
Number of lines	5	5	5	5
Total Erlangs per source	.2000	.1000	.0667	.0500
Total Erlangs	2	2	2	2
Probability of Blocking	.0196	.0352	.0409	.00438
Percentage Blocked	1.98	3.52	4.09	4.38

Table 2: Engset and Binomial Results Example

The choice of which to use, is based on whether callers, when blocked, leave the system (Engset) or hang around (Binomial). These formulas are used in such applications as small key telephone systems or PBX systems in which a finite number of users have dial access to a special service trunks, Foreign Exchange (FX) trunks, or tie-trunk group.

Delay Formulas—Infinite Sources:

Erlang C and **Equivalent Queue, Extended Erlang B (EQEEB)** are two possible delay formulas for infinite sources. Delay formulas apply to those situations in which the caller waits in a queue until a server is available when all servers are busy. The time spent waiting is more important than the probability of being blocked. Erlang C and EQEEB can be used in telephone queuing situations to:

- Determine staffing levels for an Automatic Call Distributor (ACD) or Automatic Call Sequencer (ACS)
- Determine staffing levels for PBX operator positions
- Determine outgoing line quantities on a PBX with either on-hook or off-hook queuing
- Perform sensitivity analysis to answer "what if?" questions for staffing or line quantities using different traffic volume.

Erlang C: - The Erlang C delay formula is presented as a group of equations in several different notations.

The **assumptions** for Erlang C **are** a **Poisson arrival process; exponential service times; infinite sources; a FIFO queue; a single server queue**, in which calls are directed to the first available server; **no calls leave the queue**; and the **waiting area** (queue) is as large as necessary (**infinite**).

The Erlang C formula will be presented in two ways; the first the traditional formula, as typically shown in traffic engineering texts, and then as a series of equations that provide a wide variety of useful insight. The traditional view of Erlang C is this formula group:

$$\mathrm{P(>0)} = \frac{\dfrac{A^{N}}{N!} * \dfrac{N}{N-A}}{\displaystyle\sum_{X=0}^{N-1} \dfrac{A^{X}}{X!} + \dfrac{A^{N}}{N!} * \dfrac{N}{N-A}}$$

$$\mathrm{D1} = \mathrm{P(>0)} * \frac{H}{N-A}$$

$$\mathrm{D2} = \frac{H}{N-A}$$

$$\mathrm{P(>T)} = \mathrm{P(>0)} * e^{(-(N-A)*T/H)}$$

Where:

A = Offered Traffic

N = Number of Servers

D1 = Average Delay on All Calls

D2 = Average Delay on Delayed Calls

H = Average Call Time

P(>0) = Probability of a Delay >0

P(>T) = Probability of a Delay >T

T = Time

This does not allow as much insight as what follows. The following is Erlang C, and companion equations, that provide more practical information. The information **input required** is the **arrival rate per unit time**, **service rate per unit time**, and **number of servers**.

The formulas are:

1. System utilization. The value of P is the minimum number of servers needed.

$$\mathbf{P} = \frac{\lambda}{\mu}$$

Where: P = Total System Utilization
λ= Arrival Rate Per Unit Time
μ = Service Rate Per Unit Time

The arrival rate, and service rate are particularly useful in determining the minimum number of servers required. For example, if λ = 1.2 per minute, and μ = 0.5 per minute, then:

- Example:

- For 72 calls per hour: λ **= 72 ÷ 60 (min.) = 1.2 calls per minute**

- For 2 minute service time: μ **= 60 (min.) ÷ 2 = 30 per hour ÷ 60**
= .5 per minute

$$\mathbf{P} = \frac{1.2}{.5} \qquad \mathbf{P = 2.4}$$

- Total utilization of all servers is 2.4. This value identifies the minimum number of servers you need. Given that you can't have 2.4 people, the minimum number of servers required would be 3.
- Utilization per server is 2.4 ÷ 3 = 80%

2. **The probability that all servers are idle**:

$$\mathbf{P_0} = \left[\sum_{k=0}^{n-1} \frac{\rho^k}{k!} + \frac{\rho^n}{n!\ \ 1-\frac{\rho}{n}} \right]^{-1}$$

- Where: ρ = Total System Utilization
n = Number Of Servers

3.The probability that all servers are busy:

$$P_b = \frac{\rho^{n}\ P_0}{n!\ \ 1 - \frac{\rho}{n}}$$

4.The average number of callers in the queue:

$$L_q = \frac{\rho\ P_b}{n - \rho}$$

5.The average number of callers in the system (waiting or being served):

$$L = L_q + \rho$$

6.The average wait time in the queue:

$$Tq = \frac{L_q}{\lambda}$$

7.The average flow time through the system:

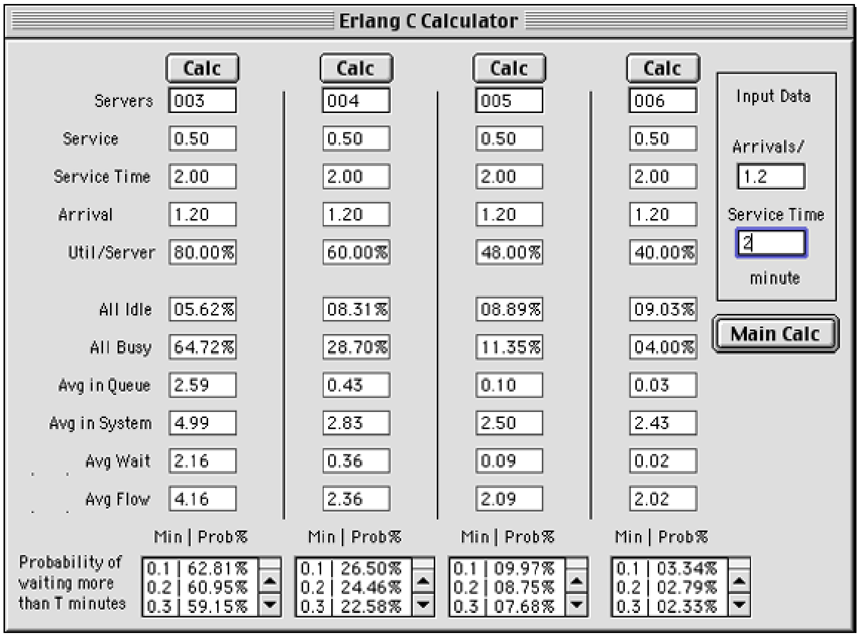
$$T = \frac{L}{\lambda}$$

This is the sum of the service and wait times.

8.The probability of waiting longer than "t" time:

$$P(t) = P_b\ e^{-}(n\mu - \lambda)t$$

By varying the value of "t", it is possible to obtain any percentile (e.g. the probability of wait less than 90%. See figure 12 below.



Erlang C Calculator

	Calc	Calc	Calc	Calc
Servers	003	004	005	006
Service	0.50	0.50	0.50	0.50
Service Time	2.00	2.00	2.00	2.00
Arrival	1.20	1.20	1.20	1.20
Util/Server	80.00%	60.00%	48.00%	40.00%
All Idle	05.62%	08.31%	08.89%	09.03%
All Busy	64.72%	28.70%	11.35%	04.00%
Avg in Queue	2.59	0.43	0.10	0.03
Avg in System	4.99	2.83	2.50	2.43
Avg Wait	2.16	0.36	0.09	0.02
Avg Flow	4.16	2.36	2.09	2.02
	Min | Prob%	Min | Prob%	Min | Prob%	Min | Prob%
Probability of waiting more than T minutes	0.1 | 62.81% 0.2 | 60.95% 0.3 | 59.15%	0.1 | 26.50% 0.2 | 24.46% 0.3 | 22.58%	0.1 | 09.97% 0.2 | 08.75% 0.3 | 07.68%	0.1 | 03.34% 0.2 | 02.79% 0.3 | 02.33%

Input Data

Arrivals/ 1.2

Service Time 2

minute

Figure: 12 Erlang-C equations output

9.The probability of K callers in the queue:

$$P_k = \left[\begin{array}{ll} P_0 \text{ x } \dfrac{(np)}{k!} & \textbf{Where } k <= n \\ & \\ P_0 \text{ x } \dfrac{p^k n^n}{n!} & \textbf{Where } k => n \end{array}\right]$$

- Where:

P_0 = Equation 2.

n = Number of callers

λ= Arrival Rate Per Unit Time

μ = Service Rate Per Unit Time

Note: $p = \dfrac{\lambda}{n\mu}$ and must be less than 1

Erlang C is used primarily for ACD and PBX operator position staffing and closely parallels realistic situations. For an ACD system in which a detailed simulation model is written, Erlang C can predict the number of agents required, very close to a simulation model output. In situations in which a long wait is acceptable (e.g., call centers for organizations that don't care), Erlang C will predict a level of service worse than the actual level. However given that most people want to design for a good grade of service, e.g. 90% of all calls answered with 18 seconds, the error Erlang C introduces is small. Also some systems want to prioritize certain call types, e.g. calls to an 800 number cost the call center by the minute, so you might want to answer them ahead of local free calls. This violates the FIFO assumption of Erlang C. Again, for a good grade of service, this introduces an acceptable error into the results.

Equivalent Queue, Extended Erlang B (EQEEB): - **EQEEB,** more a process than a formula**,** was developed by James Jewitt of Telco Research Corporation, as a technique for determining line quantities in queuing situations. This formula can not only be more accurate than Erlang C, it provides traffic per line utilization as well. Because Erlang C unrealistically assumes that no calls leave the queue, EQEEB is more accurate when poor grades of service (i.e., when more than 10 percent of callers wait more than one minute for service) is acceptable. For good service levels, the error of this assumption is small.
The main problem with EQEEB today is that it is based on the assumption that people will wait specific times, up to 10 minutes, before they overflow, when making outgoing calls over a trunk group. This may have been reasonable when line costs were high, but today, queuing on outgoing lines is rarely done, so we won't dwell on EQEEB any further.

These formulas, i.e. Erlang C and EQEEB, and their output must be viewed as approximations and estimates of realistic situations and treated as such. Users often do not follow the theory as closely as designers would like, and user behavior strongly influences the real world experience.

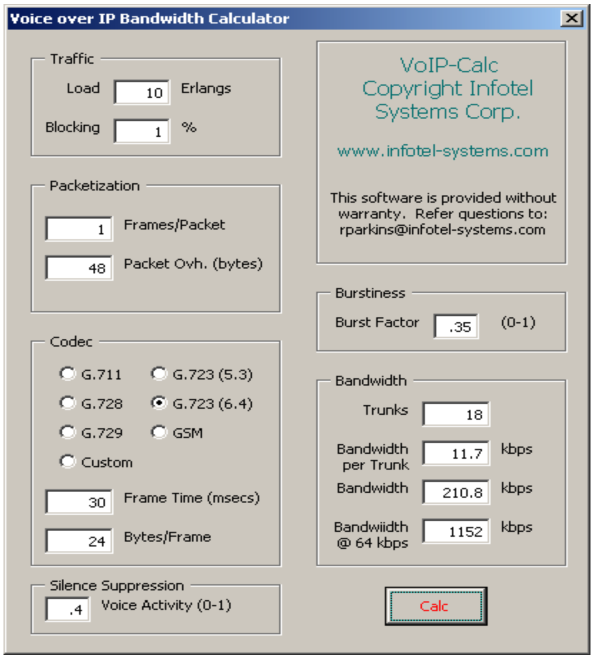
Voice over IP (VoIP)

In 1995, VocalTec Communications Ltd. (***http://www.vocaltec.com/***), was the first company to introduce a voice over the internet applications. Since that time, VoIP has become the new darling of the telecommunications environment. It legitimized packetized voice. The key benefits of packetized voice is its potential reduction of bandwidth, from 2 main sources:

- the ability to use a codec, other than ITU-T G.711, to digitize speech at a lower bit rate
- the ability to take advantage of the half duplex nature of conversation over a full duplex link, by utilizing the approx. 50%-60% idle path time when one direction is active.

Today the design issues focus on two main networks, Frame Relay, and some IP based network, whether internet or intranet. The whole issue of VoIP design is a paper unto itself, and you will find a tutorial and a model at: ***http://www.infotel-systems.com/tutorials&tools.htm***

The following figure is a dialog box from the VoIP calculator tool:



Voice over IP Bandwidth Calculator

Traffic

Load 10 Erlangs

Blocking 1 %

Packetization

1 Frames/Packet

48 Packet Ovh. (bytes)

Codec

○ G.711 ○ G.723 (5.3)

○ G.728 ● G.723 (6.4)

○ G.729 ○ GSM

○ Custom

30 Frame Time (msecs)

24 Bytes/Frame

Silence Suppression

.4 Voice Activity (0-1)

VoIP-Calc
Copyright Infotel
Systems Corp.

www.infotel-systems.com

This software is provided without warranty. Refer questions to: rparkins@infotel-systems.com

Burstiness

Burst Factor .35 (0-1)

Bandwidth

Trunks 18

Bandwidth per Trunk 11.7 kbps

Bandwidth 210.8 kbps

Bandwiidth @ 64 kbps 1152 kbps

Figure: 13 VoIP-Calc Dialog Box

SUMMARY

The formulas identified are those most commonly used worldwide. All formulas are easily programmed in Excel. (The user should check the range of precision and the largest factorial number the machine accommodates. To exceed its range, the user must convert all formulas to logarithms, or use recursion.) There are many applications for each formula. Users should consult the decision tree discussed, to ascertain the appropriate formula for a given application.

To obtain accurate results, the telecommunications manager must observe certain cautions in using the formulas. The manager must check assumptions for reasonableness; include all time components in the holding time call length; express arrival and service rates in the same time unit; and convert service times to service rates. He or she must also remember that, despite its four decimal places, the output result is only as good as the accuracy of the input data. Lastly, and most importantly, choose a formula that fits the application.

With these cautions in mind, the telecommunications manager will find traffic engineering techniques a useful tool in system design and planning.

About the Author:

Richard Parkinson is Vice President of Infotel Systems Corporation, a telecommunications consulting firm specializing in data and voice systems design management. Infotel also offers several telecommunications related seminars. Visit Infotel's web site at: ***http://www.infotel-systems.com*** for more information

Bibliography

Introduction to Teletraffic Engineering (1974) - Ramses R. Mina - Telephony Publishing

A Practical Guide to Teletraffic Engineering and Administration (1983) - Robert W. Lawson - Telephony Publishing

Lee's abc of the telephone - Traffic Series - **Tables for Traffic Management and design - Trunking** - Lee's abc of the Telephone Publishing

Reference Manual for Telecommunications Engineering (1984) - Roger L. Freeman - John Wiley & Sons ISBN 0-471-86753-5

Engineering and Operations in the Bell System (1984) - R.F. Rey - Editor - AT&T Bell Laboratories

Systems Analysis for Data Transmission (1972) - James Martin Prentice Hall ISBN 0686981006

Index of Figures

Index of Tables

Keywords Index

Keywords Index (Continued)