A Neural Network Approximation for the Expected Waiting Time in a GI/G/m Queue

Stephen Farris, Manuel Rossetti, Ph.D., P. E.
Department of Industrial Engineering
University of Arkansas
Fayetteville, AR 72701, USA

Abstract

The goal of this research is to develop a good approximation for the expected wait time in a GI/G/m queue. We use neural networks to close the gap between the output of Whitt's [6] GI/G/m approximation, and the results obtained via simulations of a GI/G/m queue. Once the neural network has been trained, we will be able to feed in the parameters and resulting output of Whitt's GI/G/m approximation, along with additional information describing the arrival and service distributions of the queue, and yield an acceptably accurate approximation for the expected wait time in a GI/G/m queue.

Keywords: queueing, approximation, neural networks, GI/G/m

1 Introduction

Analysis of large queueing networks, an essential part of logistics modeling and communications network modeling, can be a computationally challenging task. Full-scale simulation is often infeasible especially if the simulation is to be embedded within an optimization procedure. This has prompted research into hybrid techniques combining analytical methods and simulation. These hybrid techniques use approximations to compute the service metrics for individual queues in the system, and then simulate the flow of customers among the queues. This approach relies heavily on approximations that adequately model the individual queues and the network of queues within the system. While many closed-form queueing approximations do exist, most rely on simplifying assumptions that can render them inappropriate for modeling the complexity found in real-world situations. Clearly an approximation is needed that does not require such simplifying assumptions; the queue in question is the GI/G/m, and there are no closed-form, exact analytical solutions for this queue at this time. This research seeks to satisfy this need by combining an existing approximation that is "close" to the GI/G/m, along with induction learning techniques. We use the GI/G/m approximation presented in Whitt [6], along with a neural network that will be trained on Whitt's approximation, other parameters specifying the service and arrival distributions for the queue, as well as a simulation model. Our initial goal for this research is to develop an approximation that shows a significant improvement over Whitt's GI/G/m approximation in terms of absolute relative error. A secondary goal is to achieve an approximation with an average absolute relative error of less than 10%.

2 Literature Review

In Whitt's [6], approximations are presented for the GI/G/m queue. The model presented has unlimited waiting room, *m* identical parallel servers, and the queue operates on first-come first-served basis. All service and interarrival times are independent, identically distributed random variables with general distributions that are specified by their first two moments. The arrival process is specified by λ and c_a^2, which are the arrival rate and the squared coefficient of variation of the interarrival time. The service process is specified by τ and c_s^2, which are the mean service time and the squared coefficient of variation of service time. Other assumptions made by Whitt [6] are that traffic intensity, ρ, is less than 1, and that mean service time, τ, is normalized equal to 1. The traffic intensity, ρ, is equal to $\lambda\tau/m$. Whitt [6] focuses on approximations for the expected waiting time of a customer before beginning service, or EW. The formula he presents for EW is:

$$EW\left(\rho, c_a^2, \mathrm{c}_s^2, \mathrm{m}\right) \approx \phi\left(\rho, c_a^2, c_s^2, m\right)\left(\frac{c_a^2 + c_s^2}{2}\right) EW\left(M/M/m\right) \tag{1}$$

where

$$\phi(\rho, c_a^2, c_s^2, m) = \begin{cases} \left(\dfrac{4(c_a^2 - c_s^2)}{4c_a^2 - 3c_s^2}\right)\phi_1(m,\rho) + \left(\dfrac{c_s^2}{4c_a^2 - 3c_s^2}\right)\psi\left((c_a^2 + c_s^2)/2, m, \rho\right) & c_a^2 \geq c_s^2 \\ \left(\dfrac{c_s^2 - c_a^2}{2c_a^2 + 2c_s^2}\right)\phi_3(m,\rho) + \left(\dfrac{c_s^2 + 3c_a^2}{2c_a^2 + 2c_s^2}\right)\psi\left((c_a^2 + c_s^2)/2, m, \rho\right) & c_a^2 \leq c_s^2 \end{cases} \quad (2)$$

and EW(M/M/m) is the expected waiting time in the standard M/M/m queueing system. In the preceding equations, ψ and φ denote functions of m and ρ that can be found in Whitt [6]. Measures of interest other than expected waiting time, such as the number of busy servers (*E[B]*), the expected queue length (*E[Q]*), the expected number of customers in the system at any given time (*E[N]*), and the expected time in system (*E[T]*), can be found through analytical relationships between these measures and the expected waiting time.

In order to measure the accuracy of the approximations, Whitt [6] uses absolute difference and relative percentage error as performance metrics. As long as one of those values is below a designated threshold, Whitt contends that the approximation is accurate. He defines an adjusted measure of error (*AME*) to combine the effects of the two performance measures:

$$AME = \min\left\{A\left|exact - approx.\right|, 100\left(\left|exact - appprox.\right|\right)/exact\right\} \quad (3)$$

where A is a constant determined by the user that weights the importance of the two performance metrics. Whitt does not present an exact value A for use in all calculations. By evaluating the approximations based on the performance metrics outlined above, Whitt [6] is able to draw the conclusion that the approximations work best when the probability distributions for service time and interarrival time are not too irregular. The approximations are less accurate when c_a^2 and c_s^2 are large. The approximations overestimate the actual values both when m is large and ρ is small, however, the approximation works better than earlier results when traffic is heavy (ρ is large). Whitt compared the approximation for different queueing models. He concluded that the approximations work well for GI/M/m queues when $0< c_a^2<1$, as in an E_4/M/m model. It also performs well for E_2/E_2/m models but is not always better than earlier approximations for G/H_2/m models. He identifies room for improvement in the approximations when $c_a^2>1> c_s^2$ and also when $c_a^2<1$.

Whitt's approximation involves characterizing the arrival process and the service process of the queue by the first two moments of the distributions. Approximations are then developed based on the parameters, $\vec{x} = (\lambda, c_a^2, \tau, c_s^2, m)$. This set of input parameters involves only second moment information plus the number of servers so that the approximation is not a full specification of the queueing system. It should be clear that the mapping ability is limited for approximations based upon only the first two moments. There are a wide variety of more complicated approximations, but our point here is that analytical models can be used as building blocks.

Another way to build an approximating function for the expected waiting time would be to fit an induction model via some technique (least squares, non-linear regression, neural networks, GMDH, etc.) directly to the output of the system over a wide range of input parameters and distributions. In this case, we let $(\vec{x}_1, \vec{y}_1), \ldots, (\vec{x}_k, \vec{y}_k)$ be input/output pairs from the system where $\vec{x}_k$ represents the *k*th input and $\vec{y}_k$ the *k*th output and $\vec{y}_k = f(\vec{x}_k)$. Let us consider the response variable y_k univariate for simplicity of discussion, but these techniques are not limited to that case. While it is possible to build an induction model based on observations of the actual system, it will be impractical due to the large amount of data required; however, an induction model could be built based on input/output pairs from a detailed simulation. One can think of this approach as a form of simulation meta-modeling, see Friedman [1] for example. Other similar approaches have been taken by Kimura [2] where interpolations between analytical models are used to build new approximations.

The work closest related to ours is that of Shin, Sargent, and Goel [4]. Shin, Sargent, and Goel [4] present a systematic approach to queueing approximation via meta-modeling using Gaussian radial basis functions, which are specialized feed-forward networks having 3 layers that map an input space onto a desired output space, and use Gaussian basis functions. They use the SG algorithm to determine the number of basis functions, the basis function centers), and the basis function weights. For a detailed explanation of the SG algorithm, we refer the reader to Shin and Goel [3,4]. An additional parameter specifies the coverage of the input space, with 100% coverage requiring one basis function for every input value, permitting direct control over the model complexity. They demonstrate their approach on the M/M/1 queue, with a data set created by fixing the arrival rate and varying the service rate. They vary the basis function widths to create a variety of different meta-models for which they compute both the fitting and test mean squared errors (MSE) defined by the equation:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2 \tag{4}$$

They then show how each of these meta-models perform against the known closed-form equation for the expected wait time for the M/M/1 queue. Unlike the approach taken by Shin, Sargent, and Goel [4], we use a custom neural network created by off-the-shelf software and tailored to our problem specifications. In addition, our approach uses back propagation instead of the SG algorithm to set the weights. When evaluating our approximation, we consider only test data sets, not fitting data sets, to get an idea of how valuable our approximation is in the real world. To that end, we consider average absolute relative error, instead of mean square error (which is an un-scaled error metric) because it is more relevant to real-world applications. Finally we present an approximation for the GI/G/m for which there is no closed-form analytical model to test against, requiring simulation to obtain the "true" values for the expected wait time in queue. To reduce the time required to train our neural network to approximate the GI/G/m queue, we utilized Whitt's GI/G/m approximation, as our starting point. We train our neural network on $\vec{x} = \left(\lambda, c_a^2, \tau, c_s^2, m\right)$ as well as the output from Whitt's approximation and an exact simulation model.

Our research is intended as the first step in developing hybrid analytical/simulation techniques that are robust enough for general use in the real world. In his research on modeling queueing networks, Whitt [5] presented the Queueing Network Analyzer, or QNA, software. The purpose of QNA is to approximate congestion measures for a network of queues. The software is intended to be used on non-Markov networks, so arrival processes do not have to be Poisson and service times do not have to be exponential. He assumes that each node in the network is stochastically independent. The arrival process to each node is partially characterized by a few parameters, which are represented by linear equations. The system of linear equations is solved to determine the internal flow parameters. Once these parameters have been defined and a routing matrix has been provided, each of the nodes is analyzed separately. Calculus transforms the parameters that define each queue and node to represent operations such as merging, splitting, and departure. Currently QNA uses just the first two moments of the arrival and service distributions for each queue. Future research will involve expansion of QNA to include higher moment information so it could be used with our neural network approximation. Our ultimate goal is to integrate the ideas of QNA, Neural Networks, and hybrid techniques to reduce the computational requirements of large-scale simulations of these systems.

3 Methodology

To perform the neural network fits we needed a set of test cases for which the wait time in queue was known. To generate these test cases we randomly select one of the six following distributions (independently) for both the arrival and service distributions: (exponential, uniform, triangular, gamma, Weibull, and lognormal). Once the distributions had been selected, values for the distribution parameters were selected based on the information in the table below:

Table 1: Distribution Parameters

Distribution	Parameters		
Exponential	λ ~uniform(1,25)		
Uniform	min~uniform(1,25)	max~uniform(min,25)	
Triangular	min~uniform(1,25)	mode~uniform(1,25)	max~uniform(1,25)
Gamma	α ~uniform(1,5)	β ~uniform(1,5)	
Weibull	α ~uniform(1,25)	β ~uniform(1,25)	
Lognormal	E[X]~uniform(1,25)	$\sqrt{V[X]}$ logstdev~uniform(1,25)	

These parameters were used to compute λ, μ, and the offered load for the queue. The offered load and ρ (a randomly generated number from 0.05 to 0.95 inclusive) were then used to compute the required number of servers.

Table 2: Test-case Parameter Statistics

	m	λ	μ	c_a^2	c_s^2
Mean	1.99721	0.111011	0.133516	2.15944	2.56615
Stdev	1.67144	0.101712	0.119208	5.53683	6.1999
Sample size	3225	3225	3225	3225	3225
Minimum	1	0.04	0.04	0	0
Q1	1	0.05	0.06	0.22	0.09
Median	1	0.07	0.09	1	0.87
Q3	2	0.13	0.16	1	1
Maximum	27	0.98	0.97	49.07	49.59

Table 3: Frequency of Distributions in Training Data Set

Distribution	Arrival (%)	Service (%)
Exponential	32.217	24.434
Uniform	7.504	9.3023
Triangular	8.4341	9.0233
Gamma	14.7597	16.031
Weibull	7.1938	9.5504
Lognormal	28.91	31.6899

For simulation, we used Arena 7.01 from Rockwell Software. The model used VBA to import the table containing the test cases created in Access. At the start of each replication, a new test case was loaded into the model and run, with the resulting wait time in queue (and other related statistics) automatically written back out to the database by the VBA code in the Arena model. For the neural network fitting process we used NeuroSolutions 4 from NeuroDimension. First we selected the desired use of the neural network: approximating a function. We used the file containing the test cases as the input to the neural network, and selected the inputs and the outputs for the fitting process. Next we selected the neural network settings. These included neural network complexity ranging from Low to High (we selected high), and data set aside for cross-validation (20%). The software automatically generated a neural network for us with the following topology (back propagation not shown):

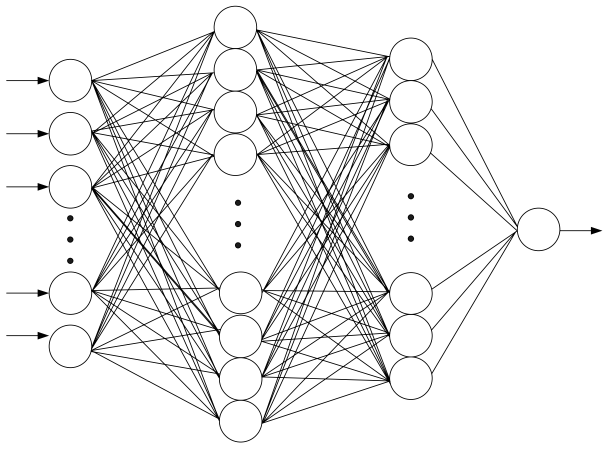

Figure 1: Neural Network Topology

A hyperbolic transfer function was used in the hidden layer, and back propagation was used to train the network. To select the best set of inputs for training the neural network, we started with: $\left(\lambda, \mu, c_a^2, c_s^2, m, W_q^a, W_q^s\right)$ where W_q^a

and W_q^s are the expected waiting time values from Whitt's approximation and from the simulation respectively. We then added the second, third, and forth moments and with the skew of the arrival and service distributions as inputs. We trained the network for 10,000 epochs (internal software time steps). We then tested the neural network on a file containing a new set of test cases. The testing output file showed the desired wait time in queue side-by-side with the wait time in queue from the neural network. We computed the following error metrics for both the neural network and Whitt's GI/G/m approximation for each test case:

Table 4: Error Metric Definitions

Error Metric	**Definition**
Error	$W_q^s - W_q^a$
Absolute Error	$\left\lvert W_q^s - W_q^a \right\rvert$
Relative Error	$\dfrac{W_q^s - W_q^a}{W_q^s}$
Absolute Relative Error	$\left\lvert \dfrac{W_q^s - W_q^a}{W_q^s} \right\rvert$

W_q^s is the expected wait time in queue (WQ) as computed by the Arena simulation. W_q^a is the output of the approximation under evaluation. We then computed the averages of these metrics so that we could compare our approximation to Whitt's GI/G/m approximation. We selected *Absolute Relative Error* as the most logical criterion for this comparison.

4 Conclusions

The table below shows a side-by-side comparison of our neural network-based approximation with Whitt's GI/G/m approximation for the test cases run.

Table 5: Summary of Approximation Error Metrics

	Whitt Approximation				Neural Network Approximation			
	Error	Absolute Error	Relative Error	Absolute Relative Error	Error	Absolute Error	Relative Error	Absolute Relative Error
Average	-1.16990	1.76220	-1.6E-01	0.195159	0.24516	1.07652	0.036738	0.166515
Stdev	6.62107	6.48855	0.266897	0.240814	6.16899	6.07925	0.392490	0.357299
Sample size	3185	3185	3185	3185	3185	3185	3185	3185
Minimum	-134.550	0	-1.87655	0	-23.066	0	-4.92096	0.00003
Q1	-1.440	0.160	-0.26930	0.03352	-0.361	0.192	-0.07852	0.03666
Median	-0.350	0.620	-0.06540	0.09513	0.055	0.478	0.01083	0.08321
Q3	0.020	1.635	0.00581	0.27948	0.574	1.006	0.08679	0.16219
Maximum	233.660	233.660	0.64036	1.87655	282.597	282.597	6.95870	6.95870

As the table shows, our approximation outperformed Whitt's GI/G/m approximation with an average absolute relative error of 16.65% versus 19.52%. Below are box-plots comparing the error and absolute relative error of our approximation and Whitt's GI/G/m approximation:

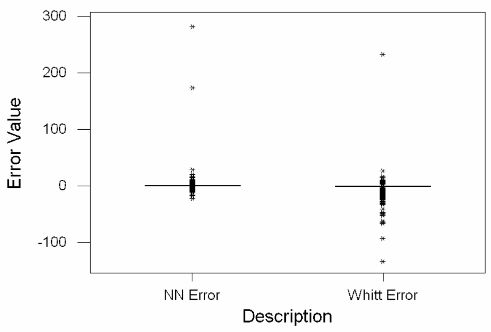

Figure 2: Error Boxplot

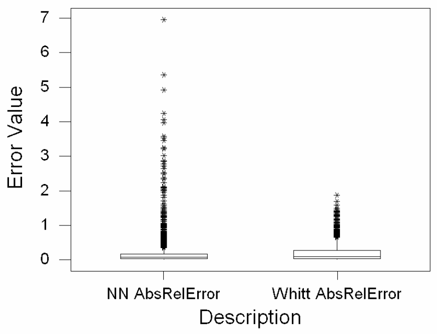

Figure 3: Absolute Relative Error Boxplot

Notice that in the plots for both error and absolute error (Figure 2), our approximation error is clustered tighter around zero than Whitt's GI/G/m approximation error, confirming our conclusion. The large number of outliers in Figure 3, in contrast to the tightness in Figure 2, suggests that relative and absolute relative error do not provide meaningful results when the expected wait times in queue become very small. In the future we plan to use the expected time in the system which has consistently larger values (bounded by the mean service time) and thus should prove to be a better overall metric. We are investigating the worst cases to improve performance for those cases. These results are preliminary and our research in this area is ongoing in an effort to meet our secondary goal of 10% or lower absolute relative error; however, the method we presented shows promise by demonstrating a better general-case approximation than the GI/G/m presented in Whitt [6]. While our approximation outperformed the GI/G/m for the test cases run, there are a number of directions in which the research presented here could be extended to achieve a greater reduction in error. These include: considering different network topologies; running multiple simulation replications per test case to achieve better "true" values of expected wait times; forcing simulation replications to run until a desired wait time half-width is reached; presenting bounds to the neural network as inputs; identifying outliers and treating them separately; applying a bias correction to the neural network output.

References

1. Friedman, L. W., 1996, "The Simulation Metamodel," Kluwer Academic Publishers, Dordrecht, The Netherlands.
2. Kimura, T., 1994, "Approximations for Multiserver Queues—System Interpolations," Queueing Systems, 17(3-4) 347-382.
3. Shin, M., and A. L. Goel, 1998, "Radial Basis Function Model Development and Analysis Using the SG Algorithm," Technical Report 98-5, Department of Electrical Engineering and Computer Science, Syracuse University, Syracuse, New York 13244.
4. Shin, Sargent, and Goel, 2002, "Gaussian Radial Basis Functions For Simulation Metamodeling", The Proceedings of the 2002 Winter Simulation Conference, eds. E. Yücesan, C.-H. Chen, J. L. Snowdon, and J. M. Charnes, 483-488.
5. Whitt W., 1983, "The Queueing Network Analyzer", The Bell System Technical Journal, Vol. 62, No. 9, November.
6. Whitt, W., 1993, "Approximations for the GI/G/m queue," Productions and Operations Management. 2(2) 114-161.