Towards an Automatic Modeling Tool for Observed System Behavior

Thomas Begin[*], Alexandre Brandwajn[+], Bruno Baynat[*], Bernd E. Wolfinger[#], and Serge Fdida[*]

[*]Université Pierre et Marie Curie – Lab. LIP6 – CNRS, Paris, France
{Thomas.Begin, Bruno.Baynat, Serge.Fdida}@lip6.fr
[+]University of California Santa Cruz, Baskin School of Engineering, USA
alexb@soe.ucsc.edu
[#]Universitaet Hamburg, Dept. Informatik, Germany
Wolfinger@informatik.uni-hamburg.de

Abstract. Current computer systems and communication networks tend to be highly complex, and they typically hide their internal structure from their users. Thus, for selected aspects of capacity planning, overload control and related applications, it is useful to have a method allowing one to find good and relatively simple approximations for the observed system behavior. This paper investigates one such approach where we attempt to represent the latter by adequately selecting the parameters of a set of queueing models. We identify a limited number of queueing models that we use as Building Blocks in our procedure. The selected Building Blocks allow us to accurately approximate the measured behavior of a range of different systems. We propose an approach for selecting and combining suitable Building Blocks, as well as for their calibration. We are able to successfully validate our methodology for a number of case studies. Finally, we discuss the potential and the limitations of the proposed approach.

Keywords: High-Level Modeling, Automatic Tool, Constructive Modeling, Computer and Communication Systems, Model Calibration, Building Blocks, Performance, Measurements

1 Introduction

1.1 Motivations

Analytic performance modeling of computer and communication systems has numerous applications throughout the life-cycle of such systems, from their design phase to the actual configuration, tuning and capacity planning [11]. A commonly used method, which we refer to as the constructive approach, is to attempt to reproduce in the mathematical model essential aspects of the system structure and operation. This constructive approach has its limits. First, important aspects of large and heterogeneous computer or communication systems, such as modern I/O controllers, or Internet

Service Provider networks, may be largely unknown. Second, extensive knowledge and expertise that may simply not be available may be necessary to correctly identify key system components and features lest the resulting models become unrealistic or intractable in their complexity. These difficulties motivate in part our approach.

In our high-level modeling, we don't necessarily seek to "mimic" the structure of the system under study. Rather, we focus on the observable behavior of the system as given by measurements, and attempt to infer a possible high-level model structure capable of adequately reproducing the observed system. In doing so, we forego the detailed representation of the system in favor of the possibility that a relatively simple model, not necessarily related to the apparent structure of the system, might be able to capture the behavior of the system under consideration (e.g. certain priority systems, cf. Section 2.3). An obvious justification for our approach is that, even in a complex system, it is possible that a small number of components, or a single component, may be the critical bottleneck, effectively driving the system behavior. This idea is by no means novel, and has been frequently employed in the past, e.g. in the case of an Internet path [20, 2], disk arrays [22], time-sharing system [21] and a Web server [7].

Our approach has several objectives. First, it may help discover properties of the system not immediately apparent from the system structure. This may include both the fact that the system performance can be represented by a simple model, or, on the contrary, that no simple model (among the ones examined) will be able to adequately represent the system. As such, our approach can be viewed as helpful for and complementary to constructive system modeling. Second, our approach may provide the performance analyst with a ready-to-use model to generate reliable predictions for system performance at other workload levels, without the expense and the effort of obtaining additional measurements. Finally, for a subsystem embedded in a larger system (with the obvious proviso that measurements be available for the subsystem), our approach may be able to provide a model of the subsystem that can then be incorporated in the overall system model. This latter application has a clear connection with decomposition methods [4].

The advantage of the proposed approach is that it requires a priori little information about the system. Our contribution is to automate the process of model selection and to make it systematic by embedding it into a software tool with an optimization method. As a result, the approach requires no special modeling or queueing theory expertise from the end user.

1.2 Structure

The paper is structured as follows. Section 2 describes the general framework in which we cast our approach. We present a subset of the selected models (Building Blocks), as well as our general approach to determining the best set of model parameters and grading the goodness of fit for a given Building Block. Section 3 presents a few examples of application of our tool. All case studies in our paper use measurement data from real life systems. Finally, Section 4 summarizes the main contributions of our approach, as well as its limitations, and outlines possible extensions of our work.

2 General framework

2.1 Terminology

Systems considered in our study may represent a whole computer or communication system, or specific components such as processors, a disk array, an Ethernet network or a WLAN, etc. We use the term requests to refer to the individual entities that are treated by the system, such as packets or frames in the case of networks, I/O requests in the case of storage systems, HTTP requests in the case of web servers, etc. The workload (offered load) includes all the requests that are submitted to the system for treatment. In our view, the system performance changes in response to the workload, and these changes are reflected in the corresponding measurements. More details on workloads for networks can be easily found (e.g. [24] and [8]).

2.2 Measurements of the observed system's behavior

Our approach relies on the availability of measurements of specific system performance parameters. These parameters may include quantities such as the attained throughput of requests processed by the system per time unit, as well as measures of internal system congestion such as the number of requests inside the system. Typical measured performance parameters include throughput, loss probability, average response time and queue length, denoted by $\overline{X}_{mes}$, $\overline{L}_{mes}$, $\overline{R}_{mes}$, and $\overline{Q}_{mes}$, respectively. This is illustrated in Fig. 1. The throughput $\overline{X}_{mes}$ represents the average number of requests that leave the system per unit time (this quantity may differ from the offered workload if the system is subject to losses). $\overline{L}_{mes}$ gives the probability that an arriving request is rejected, i.e., denied entry to the system. $\overline{R}_{mes}$ defines the average sojourn time (waiting for and receiving service) experienced by a request inside the system. Finally, $\overline{Q}_{mes}$ represents the average number of requests in the system. Note that, by Little's law [15], $\overline{Q}_{mes} = \overline{X}_{mes}\overline{R}_{mes}$ so that it suffices to measure any two of these three quantities.

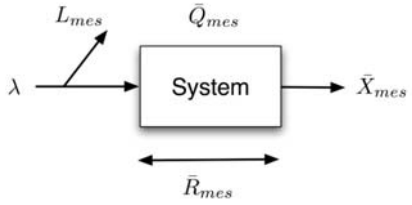

Fig. 1. Performance parameters.

In computer networks, typical performance parameters are the throughput at an interface, the time spent by packets inside the network and the packet loss ratio. In disk arrays, performance parameters may represent the I/O response time, I/O request

throughput, device utilization, etc. Crovella and Krishnamurthy [10], as well as Paxson [17] give additional useful information regarding network measurements.

Each measurement point corresponds to a set of performance parameters that have been measured at a particular state of the load (e.g. $(\overline{X}_{mes}, \overline{R}_{mes})$) and may in general also include input parameters such as the corresponding offered load. A total of n measurement points for the same system constitutes a set of measurements.

Note that in our high-level modeling approach, the measurement set must include measurement points for different load levels. Hence, methods that consist in fitting a model of discrete or continuous distribution to a sample of measurements for a single level of system load are clearly unsuitable for our approach [5].

2.3 Simple and not so simple models

As discussed in Section 1, one of the premises of our approach is that a complex system may exhibit behavior that can be reproduced by a relatively simple queueing model. Consider for example an M/G/1 queue with preemptive-resume priority discipline and three priority levels where level 1 has the highest and level 3 the lowest priority. We denote by λ_i the rate of arrivals to priority level i, by $1/\mu_i$ the mean and by γ_i the coefficient of variation of the service time for level i. We look at the mean response time of the lowest priority level for a number of values of λ_3 with the workload of higher priority levels kept constant. The well-known solution of the M/G/1 priority queue (e.g. [1]) gives the performance curve represented in Fig. 2.

While the analytical formula used to generate this curve is manageable, it may not be obvious that the mean response time for the selected priority level is in fact that of a simple M/G/1 queue with a different (higher) coefficient of variation as shown in Fig. 2. It is interesting to note that, for the parameter values used in this example, a simple M/M/1 queue (as proposed in some approximations) cannot adequately represent the behavior of lower priority levels (cf. [12]).

Our simple Building Blocks include queues such as the M/M/C, M/M/C/K, M/G/1, M/G/1/K [6], as well as the M/G/C approximation [14]. Additionally, we have defined original Building Blocks whose service times are driven by the congestion parameters of an embedded model. These Building Blocks belong to models with load dependent service times, and are not presented in this paper due to lack of space. To represent the fact that, in some systems, the response time comprises a fixed overhead as an additive load-independent component, we expand our Building Blocks to include a fixed "offset" value. Note that this offset does not affect the congestion at the server, and the response time in our Building Blocks is simply the sum of the offset value and the response time at the server. This quantity can be viewed as an irreducible and load independent additive overhead in the response time. This constant offset value is denoted by *Off* in figures and formulas.

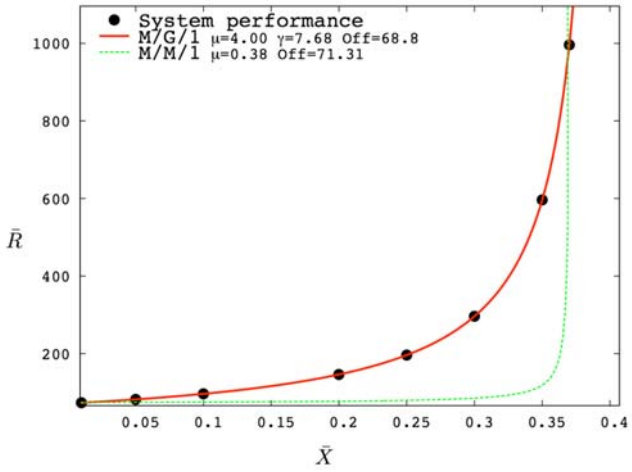

Fig. 2. Behavior of the lowest priority class in an M/G/1 priority system with $\mu_1 = 0.1, \gamma_1 = 2$, $\mu_2 = 0.5, \gamma_2 = 2$ and $\mu_3 = 1, \gamma_3 = 2$ with higher classes workload kept constant.

In the example shown in Fig. 2, to find an M/G/1 block that matches the behavior of a lower priority level in an M/G/1 priority queue, we need to determine the appropriate values for the first moment of the service time, its coefficient of variation, as well as the additional offset value. These three quantities are the parameters of this particular Building Block. We note that, in our approach we are unable to derive the values of its parameters directly from the underlying model, as would be the case in constructive modeling. This limits the predictive power of our approach. However, the fact that an M/G/1 queue (in this example) is a good fit, and the M/M/1 is not, may be valuable in the search of a simple constructive model. Interestingly, several authors [21, 20] have contemplated the use of the M/G/1 queue to model general queueing networks.

2.4 Error criterion

We need a way to measure the goodness of fit of a given model versus the measurement set. This is the role of the error criterion, referred to as ϕ. The goal of the function ϕ is to provide a convenient way to compare fairly various models. There are many reasonable ways to define such a function. In our implementation, we have selected the sum of the deviations between mean sojourn time obtained from measurements and the one obtained from the model for values of throughput equal to the measured throughput as illustrated by the Fig. 3. We use the subscript th to denote values obtained from a model. Thus, let $\bar{R}_{mes,i}, i = 1,...,n$ be the measured mean response time values, and $\bar{R}_{th,i}, i = 1,...,n$, the corresponding mean response times obtained from a model. ϕ can be formally expressed as:

$$\phi = \sum_{i=1}^{n} \left| \overline{R}_{th,i} - \overline{R}_{mes,i} \right| \quad (1)$$

It is worthwhile noting that using a different definition for the function ϕ may affect the results of our approach. In particular, the selected definition, while simple to implement, may introduce an undue bias for points near system saturation where a small visual distance between two curves may result in a very large error value (see Fig. 3). Some adjustments to the definition of ϕ are possible. As an example, one can take into account absolute and relative components for deviations.

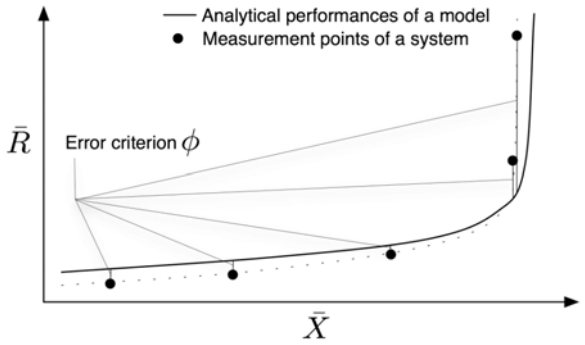

Fig. 3. Error criterion.

2.5 Search for an adequate model among the Building Blocks

Our high-level approach uses a set of generic models - the Building Blocks - that we attempt to automatically calibrate. By calibration of a Building Block we mean the search for a set of values of model parameters that minimizes the error criterion ϕ. In general, this leads to a non-linear numerical regression problem. Such a search must be efficient since it is repeated for each Building Block. Clearly, because of its inherent exponential complexity, we must exclude exhaustive search of the parameter space. Liu et. al. [16] propose an efficient and robust solution method based on a quadratic programming. Unfortunately, their method does not appear usable for our application since it is specifically tailored to open Kelly-type queueing networks [13], and it appears restricted to end-to-end delays and server utilization. We avoid as well algorithms based on derivatives of ϕ. In most cases, computing the derivative of ϕ, if at all possible, is time consuming and it is specific to each Building Block making the inclusion of new blocks difficult.

We cast the calibration of a Building Block as a numeric optimization problem, and we choose to employ an iterative descent technique in order to find a minimum of the error function. Our tool is based on Derivative Free Optimization (DFO) methods [18] and [9]. These methods have the advantage that no derivatives are invoked or estimated. They are not specific to a particular Building Block, so that the introduction

of a new Building Block is an easy task. In our specific implementation, we use a local quadratic approximation, which implies a low computational cost while speeding up the convergence. A drawback of DFO methods is that they require that all parameters be continuous. To treat all Building Block parameters as continuous, we define intermediate models in which discrete parameters are replaced by their corresponding continuous extensions. These intermediate models coincide with "standard" models when their extension parameters have integer values.

We note in passing that for a given Building Block and a set of measurements, certain bounds on the values of the Building Block (such as related to the stability of an open queue) must be taken into account in the search procedure.

The results of our experiments indicate that the proposed search method tends to be robust and very fast for Building Blocks with a limited number of parameters (say, up to 5 or 6). With a larger number of parameters, the complexity of the method leads to excessive search times. There exist several other DFO methods, some of which might outperform the one we use.

In our search for an adequate model, we start by the simplest Building Blocks (in terms of the number of parameters and their computational complexity) and move on to more complex ones only if no good calibration has been found for a simpler model.

2.6 Requirements for the methodology

Measurements represent a key component for our approach. To be of use, the sets of measurements must satisfy certain common sense conditions.

First, the different measurement points from a particular set must come from the same system, and correspond to varying load levels. As a result, it makes sense to require that key parameters of the system stay identical for every measurement point or vary in a "non-random" way as a function of the workload.

Second, in our view, the system resources can be shared by two types of traffic: the one directly captured in the available measurements (captured traffic) and the uncaptured or background traffic. In a large computer network, a significant part of the traffic may be processed without being directly captured by measurements. Since the background traffic competes for shared system resources, the common sense condition discussed above requires that the background traffic be either negligible, constant, or in a clear relationship to the captured (measured) traffic for all measurement points.

Third, the available measurement data must adequately capture the salient features of system behavior in the range of interest. Clearly, for instance, if the system response exhibits an inflection point and this inflection point is not present in the measurement data, there is little chance that the model proposed by our approach will correctly reproduce such a behavior.

3 Case Studies

3.1 Preliminaries

The proposed approach aims at finding a model, referred to as the laureate model, whose performance parameters match as closely as possible those known from system measurements (in terms of the error function described in Section 2.4). As discussed before, the laureate model is chosen from a set of pre-defined more or less simple Building Blocks.

In addition to simply matching the data points in the measurement set, we would want the laureate model to be able to correctly predict the performance of the system within some reasonable domain. Therefore, in the case studies that follow we deliberately remove one or more data points from the measurement sets. Having found the laureate model for a given data set, we then test the ability of this model to predict the system performance at the removed data points.

The data sets used in this paper have been measured in operational real-life systems such as wireless and Ethernet networks.

3.2 Broadband Wireless Network

We start by considering the high-speed wireless network for which Quintero et al. give in [19] a set of performance measurements. The measurement points, shown in Fig. 4, relate packet throughput to queueing delays experienced by packets in high load scenarios. As mentioned before, we remove some number of measurement points from the measurement set during the search for the laureate model. It is apparent from the shape of the delay time curve in Fig. 4 that, in this case, the point for the highest load level is likely to be most difficult to reproduce accurately. We elect to remove precisely this point from the measurement set in order to test the predictive capabilities of the laureate model.

Fig. 4 shows that a simple M/G/1 queue with adequate parameters determined by our approach, viz. $\mu = 20.03$, $\gamma = 5.5$ and $Off = 0.44$, closely approximates the observed performance for this system. We have also represented in Fig. 4 the results of the "best" (in terms of smallest error) M/M/C and M/M/C/K queues (these two curves are so close that they are difficult to tell apart). We notice that neither of these two queueing models is able to correctly reproduce the measured system behavior. The laureate M/G/1 model provides also a reasonable prediction for the removed point. When comparing the expected sojourn times for the throughput level of the removed point, we observe a relative error of 15%, while a comparison of expected throughputs at the same mean sojourn time for the removed point yields a relative difference of less than 1%. Given the steep slope of the performance curve in the vicinity of the removed point, we view the attained accuracy as more than reasonable. Not surprisingly, we note that the performance predictions of the M/M/C and M/M/C/K Building Blocks are poor. If we remove other, randomly selected points, and repeat the calibration procedure, we find that the laureate M/G/1 model yields predictions whose relative errors are all below 5%.

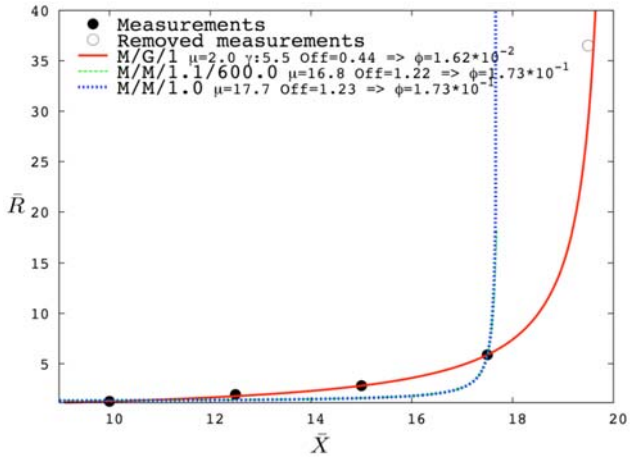

Fig. 4. Broadband Wireless Network.

It may be of interest in building a constructive model of the average performance parameters for the wireless network considered that neither the M/M/1 (M/M/C) nor the M/M/1/K (M/M/C/K) models appear adequate. Our results show that even with the best possible combination of parameters these two Building Blocks fall short from matching the observed performance.

3.3 Ethernet Network

In this example, we consider the Ethernet network with a nominal rated capacity of 10 Mbps described and measured by Wang and Keshav [23]. The performance of this network has been measured for three packet sizes: 64, 512 and 1500 bytes. Thus we have three measurement sets, one per packet length. The data points in each set give the expected sojourn time and the corresponding average packet throughput in the network.

As could be expected, the behavior of the Ethernet network considered depends on the packet size. Since the service time of a packet in this network (such as in many other communication and computer systems) includes a fixed incompressible overhead, using shorter packets reduces the transfer time of a packet but also the achievable network throughput. This tradeoff between minimizing delay versus maximizing network throughput has been extensively studied.

Fig. 5 illustrates the results obtained for this Ethernet network with 64, 512 and 1500 byte blocks. Note that the throughput in Fig. 5 is expressed in requests per time unit and not in bits or bytes per time unit.

To represent the effect of the size of packets on the network behavior, we assume that the intrinsic service time in our Building Blocks $1/\mu$ can be expressed as

$$1/\mu = S_0 + U/capa \qquad (2)$$

where U is the length of a packet in bits (units of work considered), S_0 is the fixed overhead expressed as a time, and *capa* denotes the treatment capacity of the server in terms of units of work per time unit. A similar representation of the service time may be of interest in other applications such as I/O subsystems, virtual memory, file systems, etc. In our case, two parameters, S_0 and *capa*, are required to define the service-time for a given packet size. Clearly, the use of a formula like (2) implies some knowledge of the system and a bit of constructive modeling.

This case study has two objectives. First, we show that a simple model can adequately represent the performance of this Ethernet system. Second, we show that, if we use only two of the three measurement sets, our laureate model is able to correctly predict the performance of this network for the third packet size, not used to calibrate the laureate model.

As shown in Fig. 5, the first objective is fully achieved. We observe that a simple M/G/1 with ($capa = 9.7 \times 10^{+6}$, $S_0 = 1.4 \times 10^{-4}$ and $\gamma = 6.0 \times 10^{-1}$) is well suited to reproduce the measured system behavior for different throughputs and packet sizes. As before, we tested the predictive capability of the laureate model by randomly removing some number of measurement points from the search and calibration process. These results are not presented for the sake of clearness, but the laureate M/G/1 yielded accurate predictions for the removed data points.

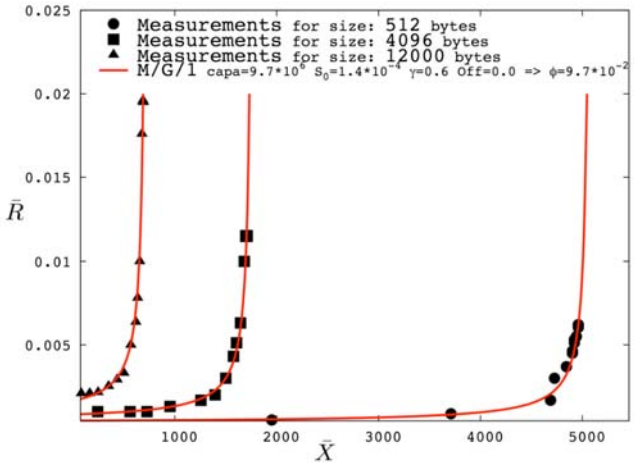

Fig. 5. Ethernet Network.

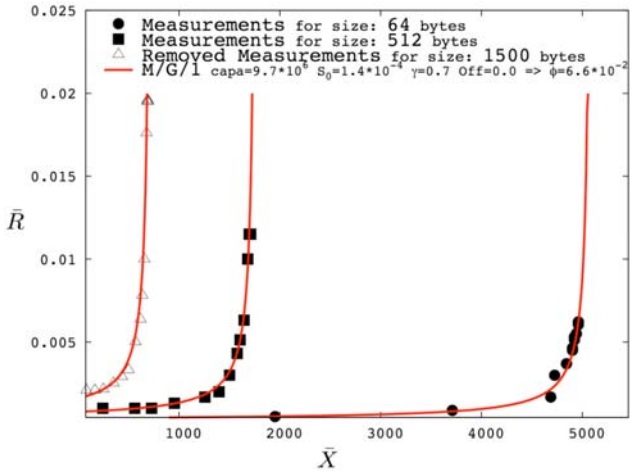

Fig. 6. Ethernet network – only sets for 64 and 512 bytes packets are used for calibration.

To illustrate our second objective, we remove one of the data sets (corresponding to one of the packet sizes) from the model search and calibration procedure. As an example, we remove the measurement set for 1500 byte packets, and we search for the "best" model. We find the same laureate as before, viz. an M/G/1 queue with ($capa = 9.7 \times 10^{+6}$, $S_0 = 1.4 \times 10^{-4}$ and $\gamma = 5.3 \times 10^{-1}$). Therefore, it is not surprising that the laureate model correctly predicts the performance of the network for the "missing" packet size of 64 bytes. This is illustrated in Fig. 6. Similar experiments where we remove the measurement set for 512 and 1500 bytes, respectively, yield the same result (not presented to be more concise).

It is important to note that, in general, the execution times for our approach are quite short. Thus, (to the extent that the network can be adequately represented as one of the Building Blocks, and we have measurement sets for two different packet sizes), the laureate model provides a convenient way to approximately determine the optimal packet size in a specific application.

Clearly, in some systems, the real dependence of the service time on the request size may be more complex than the one given in formula (2). The results presented here, suggest that, at least for this type of system, formula (2) is adequate.

4 Conclusions

We have presented a high-level modeling approach based on measurement data. Unlike in constructive modeling, we don't seek to represent "explicitly" the structure of the system being studied. We focus on the measurement results, and attempt to discover a more or less elementary model that might correctly reproduce the observed behavior. We identify a few obvious classical queueing models as possible Building Blocks for our approach. Using several sets of measurements from real computer and

communication systems, we have shown that our Building Blocks are not only able to reproduce the observed system behavior, but have also some predictive power.

We embed the search for a best fitting model in an efficient Derivative Free Optimization procedure. The speed and the efficiency of this approach allow us to automate the search for the best fitting Building Block. Note that, owing to the use of DFO methods, our approach is not limited to the particular performance measures used in this paper. Other performance measures could be used as long as the Building Blocks considered can be solved for the selected performance indices.

Our main contribution lies in the automation of the search for the laureate model. Since the search for a laureate model has been automated, performance analysts with a minimal queueing network background can use the resulting tool. It is worthwhile noting that, in addition to the laureate model, our tool can produce the next best candidate (from another Building Block), which may be of interest in some situations.

The laureate models obtained from our approach are useful to predict performance at workload levels for which measurements may not have been obtained. Hence, our approach may be of help in predicting whether the system considered fulfills or fails to fulfill a quality of service requirement. For instance, based on a projection of the growth in workload, the laureate model provides a quick answer whether a given allowed threshold-value for the average response time will be satisfied or not by the system. Unlike the "classical" constructive approach (such as a proposed framework for e-business applications [3] or disk arrays [22]), our approach reaches this goal without investigating the internal behavior of the system. The nature of the best-fitting Building Block may also be of help for constructive modeling of the system. Indeed, it may provide guidance in the search for simple approximations, by indicating which Building Block may and which ones may not work.

Our approach has several limitations. Since it is based on measurement data, the system considered (or a detailed constructive simulation model of the system) must exist, and there must be a sufficient number of measurement points to adequately capture the behavior of the system. In general, there is no guarantee that our approach will find an adequate model, and a failure of the approach does not necessarily imply that there is no adequate simple model for the given system.

As mentioned before, the laureate model determined by our approach may be a good starting point for constructive modeling or for a search for a good approximation. The potential drawback of our approach is that there is in general no clear readily seen relationship between the parameters of the laureate model and the "natural" parameters of the corresponding constructive model. This limits also the predictive application of the laureate model in that it is not typically clear how the parameters of the laureate should be modified to reflect a change in the characteristics of the system being modeled. However, we believe that, packaged as a ready-to-use tool, our approach can be of significant value both to the performance analyst in capacity planning situation, and to the performance modeler in general.

Acknowledgments. We would like to thank Safia Kedad and Francis Sourd from LIP6 for their valuable help in designing the specific Derivative Free Optimization method we implemented in our automatic model calibration tool.

References

1. Allen, A. O. Probability, Statistics, and Queueing Theory with Computer Science Applications. Academic Press, 2nd edition, 1990.
2. Alouf, S., Nain, P., and Towlsey, D. F. Inferring network characteristics via moment-based estimators, in *INFOCOM*, 2001, pp. 1045-1054.
3. Bacigalupo, D. A., Turner J. D., Graham R. N., and Dillenberger, D. N. A dynamic predictive framework for e-business workload management, in *7th World Multiconference on Systemics,Cybernetics and Informatics (SCI2003) Performance of Web Services Invited Session*, Orlando, Florida, USA, 2003.
4. Brandwajn, A. Equivalence and decomposition in queueing systems- a unified approach. *Performance Evaluation*, Vol. 5, 1985, pp. 175-186.
5. Burnham, K. P., and Anderson, D. R. Model Selection and Multi-Model Inference. Springer-Verlag, 2nd edition, 2002.
6. Brandwajn, A., and Wang, H. A. Conditional Probability Approach to M/G/1-like Queues. Submitted for publication, available as a technical report, 2006.
7. Cao, J., Andersson, M., Nyberg, C., and Kihl, M. Web server performance modeling using an M/G/1/K*PS queue. In *ICT'2003: 10th International Conference on Telecommunications*, Vol. 2, 2005, pp. 1501-1506.
8. Cong, J., and Wolfinger, B.E. A unified load generator based on formal load specification and load transformation. In Proceedings of ValueTools 2006: International Conference on Performance Evaluation Methodologies and Tools, 2006, Pisa, Italy.
9. Conn, A.R., K. Scheinberg, K, and Toint, P.L. Recent progress in unconstrained nonlinear optimization without derivatives. *Mathematical Programming*, Vol. 79, 1997, pp. 397-414.
10. Crovella, M., and Krishnamurthy, B. *Internet Measurement: Infrastructure, Traffic and Applications.* John Wiley & Sons, Inc, 2006.
11. Heidelberger, P., and Lavenberg, S.S. Computer performance evaluation methodology. *IEEE Trans. Computers*, Vol. 33, 1984, pp. 1195-1220.
12. Kaufman, J.S. Approximation Methods for Networks of Queues with Priorities. Performance Evaluation, Vol. 4, 1984, pp. 183-198.
13. Kelly, F.P. *Reversibility and Stochastic Networks*. John Wiley & Sons, 1979, New York.
14. Kimura, T. Approximations for Multi-Server Queues: System Interpolations. Queueing Systems: Theory and Applications, Vol. 17, 1994, pp. 347-382 .
15. Kleinrock, L. *Queueing Systems*, Volume 1: Theory. John Wiley & Sons, 1975.
16. Liu, Z., Wynter, L., Xia, C.H., and Zhang, F. Parameter inference of queueing models for IT systems using end-to-end measurements. *Performance Evaluation*, Vol. 63, 2006, pp. 36-60.
17. Paxson, V.E. *Measurements and Analysis of End-To-End Internet Dynamics.* Doctoral Thesis, University of California at Berkeley, 1998.
18. Powell, M.J.D. Unconstrained minimization algorithms without computation of derivatives. *Bollettino della Unione Matematica Italiana*, Vol. 9, 1974, pp.60–69.
19. Quintero, A., Elalamy, Y., and Pierre, S. Performance evaluation of a broadband wireless access system subjected to heavy load. *Computer Communications*, Vol. 27, 2004, pp. 781-791.
20. Salamatian, K., and Fdida, S. A framework for interpreting measurement over Internet. In *Proceedings of the ACM SIGCOMM workshop on models, methods and tools for reproducible network research*, Karlsruhe, Germany, 2003, pp. 87-94.
21. Scherr, A. *An Analysis of Time-Shared Computer Systems.* MIT Press, Cambridge, MA, 1967.
22. Varki, E., Merchant,A., Xu, J., and Qiu, X. Issues and challenges in the performance analysis of real disk arrays. *IEEE Trans. Parallel Distrib. Syst.*, Vol.15, 2004, pp. 559-574.

23. Wang, J., and Keshav, S. Efficient and accurate Ethernet simulation. In 24th Annual IEEE International Conference on Local Computer Networks, 1999, Boston, MA, pp. 182–191.
24. Wolfinger, B.E., Zaddach, M., Heidtmann, K.D., and Bai, G. Analytical modeling of primary and secondary load as induced by video applications using UDP/IP. *Computer Communications*, Vol. 25, 2002, pp. 1094-1102.