# Fast Computation of Hyper-exponential Approximations of the Response Time Distribution of MMPP/M/1 Queues

Paolo Romano, Bruno Ciciani, Andrea Santoro, Francesco Quaglia
Dipartimento di Informatica e Sistemistica
Sapienza Università di Roma

## Abstract

*Input characterization to describe the flow of incoming traffic in network systems, such as the GRID and the WWW, is often performed by using Markov Modulated Poisson Processes (MMPP). Therefore, to enact capacity planning and Quality-of-Service (QoS) oriented design, the model of the servers that receive the incoming traffic is often described as a $MMPP/M/1$ queue. In a recent work we have provided an approximate solution for the response time distribution of the $MMPP/M/1$ queue, which is based on a hyper-exponential process obtained via a weighted superposition of the response time distributions of $M/M/1$ queues. Compared to exact solution methods, or simulative techniques, the aim of this approximation is to provide the potential for more efficient model solution, so to enable, e.g., real-time what-if analysis in system reconfiguration scenarios. In this paper, we show how fast the computation can be supported in practical settings by ad-hoc techniques allowing the hyper-exponential model to be solved with no iterative or numerical costly steps, which would otherwise be required in order to compute the length of transient phases due to state switches in the MMPP arrival process. An application to the context of performance analysis of a GRID system is also shown, supporting the efficiency of our proposal.*

## 1 Introduction

In the context of queuing theory, a well known model for system evaluation is the $M/M/1$ queue, which is often appreciated for its fast computability. Concerning this model, while several works have shown that the exponential distribution for the service time can well fit specific real world settings (see, e.g., [8, 16]), workload characterization studies of networked systems, such as the GRID and the WWW, have shown that the incoming traffic behavior must rely on models more complex than a simple Poisson process [3, 6, 10, 13, 18, 22]. Specifically, in order to capture the typical features of incoming traffic, such as self-similar and burstiness behaviors, or long range dependency, one of the most used models is the Markov Modulated Poisson Process (MMPP) [17, 19, 20, 21, 24], which is a Poisson process whose mean value changes according to the evolution of a Markov Chain [7]. Hence, it would be important to use the $MMPP/M/1$ queue as a realistic model for networked servers.

In a previous work [5], we have described a technique allowing analytical approximations of the output distributions (i.e. response time and queue length distributions) of the $MMPP/M/1$ queue. This technique consists in approximating that queue as a weighted superposition of different $M/M/1$ queues, which is used to derive statistical upper and lower bounds for the cumulative distribution functions of the $MMPP/M/1$ response time and queue length. As a matter of fact, these bounds have been expressed as hyper-exponential distribution functions, just obtained by an ad-hoc (linear) combination of the exponential distributions characterizing the $M/M/1$ queues.

In this work we show how such an approximate model can be effectively employed in practical settings for supporting, e.g., interactive what-if analysis via real-time computation of the response time distribution, which is particularly important for performance prediction and assessment in case of (dynamic) system reconfiguration events. Actually, as we also quantify, via an experimental study, the main computational intensive step associated with the solution of the approximate model is the evaluation of the length of transient periods when the request arrival process switches between different states of the MMPP. This is due to the need for applying iterative techniques required for determining convergence towards steady state statistics. To tackle this drawback, we present a solving procedure for the approximate model, which completely avoids the need for such iterative approach. Thus it allows saving most of the computational cost for obtaining output statistics from the approximate model. The procedure is based on a further approximation step relying on a detailed analysis showing that the length of the transient phases can be well mapped onto hyperbola equations.

We also provide experimental results quantifying the reduced response time from our model solving approach for a case study related to performance analysis of a GRID system. This study is based on the exploitation of traces of incoming traffic to GRID servers, which have been shown to match the MMPP model [15]. Via these results we point out the viability of our proposal as a computational efficient alternative to techniques based on both exact analytical models and simulation studies of the $MMPP/M/1$ queue, in all the contexts where, beyond accuracy, the response time for model solving plays a relevant role. As sketched above, this might be the case of time constrained decision processes aimed at evaluating the effects of specific reconfiguration scenarios on the system performance, which are important especially in the context of design/maintenance of Quality-of-Service (QoS) oriented systems.

The remainder of this paper is structured as follows. In Section 2, we recall the hyper-exponential approximate model of the $MMPP/M/1$ queue. In Section 3, we provide the fast model solving procedure. Section 4 is devoted to the case study in the GRID environment.

## 2 Recall on the Approximation Technique

The objective of the work in [5] was to derive stochastic processes which approximate the behavior of the $MMPP/M/1$ queue by exploiting well known results in the context of $M/M/1$ queues. This has been done via approximating processes giving rise to statistical lower and upper bound approximations of the queue length and response time Cumulative Distribution Function (CDF) of the $MMPP/M/1$ queue.

The relevance of identifying a lower bound approximation on the CDF of the response time $r$ lies in that any setting ensuring that $CDF_{lower\_bound}(\mathbf{r}) > X$ also ensures that $CDF_{MMPP/M/1}(\mathbf{r}) > X$. Hence, the lower bound approximation can be used for modeling networked servers, without incurring the risk of underestimating the effects of the system load, which might cause violations of any established Service Level Agreement. On the other hand, the upper bound approximation $CDF_{upper\_bound}(\mathbf{r})$ can be employed to determine whether the usage of $CDF_{lower\_bound}(\mathbf{r})$ to perform capacity planning leads to potentially large oversize of the system computing power. Specifically, the lower the difference between $CDF_{upper\_bound}(\mathbf{r})$ and $CDF_{lower\_bound}(\mathbf{r})$, the lower the likelihood of oversize.

In the analysis in [5], the MMPP modeling the incoming traffic is composed by a generic number of $H$ states ($S_1$ ... $S_H$), and the notation $M_i/M/1$ has been used to refer to a $M/M/1$ queue whose arrival rate is the $\lambda_i$ observed in the generic state $S_i$ of the MMPP. The analytical approximations have been based on pinning the response time and queue length of the $MMPP/M/1$ queue to the steady state values of the $M_i/M/1$ queue as long as the MMPP arrival process does not change its state from $S_i$ to whichever state $S_j$.

As an example, consider a two-state MMPP. The mean number of resident requests at time $t$ in the $MMPP/M/1$ queue is shown in Figure 1.a. The instants $T_k$, $T_{k+1}$ and $T_{k+2}$ represent transitions between the two states of the MMPP. Therefore the evolution of the mean queue length value of the $MMPP/M/1$ queue can be described as follows: each time a state transition occurs there is a transient phase ($t_{12}$ for a transition from $S_1$ to $S_2$ and $t_{21}$ for a transition from $S_2$ to $S_1$) after which the mean queue length may reach the steady state, if any, of the corresponding $M_i/M/1$ queue. As soon as another state transition occurs, a new transient phase starts.

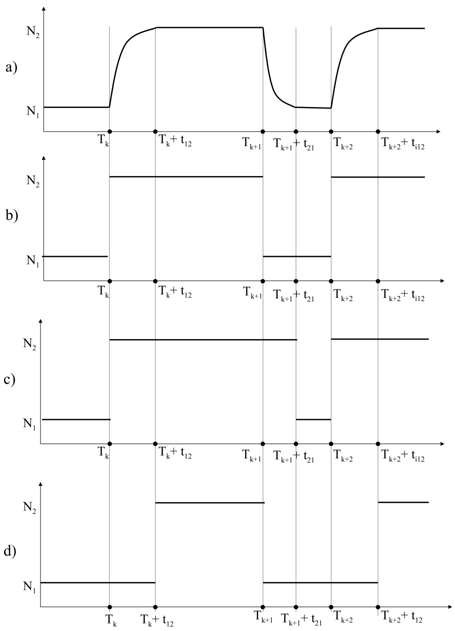

**Figure 1. a) $MMPP/M/1$ behavior; b) Basic approximation; c) Lower bound approximation; d) Upper bound approximation.**

As discussed in [5], the most obvious way to exploit the previous reasoning is to adopt the probabilities for the MMPP to stay in each state $S_i$ as the weights for an approximate representation of the behavior of the $MMPP/M/1$ queue based on weighted superpositions of the $H$ different $M_i/M/1$ queues. Specifically, denoting with $Q_i$ the steady state queue length of $M_i/M/1$ and with $p_i$ the probability for the MMPP to stay in state $S_i$, the mean queue length of the $MMPP/M/1$ queue can be approximated as $Q = \sum_{i=1}^{H} p_i Q_i$, which would correspond to the case shown in Figure 1.b. A similar technique can be applied to derive the mean response time of the $MMPP/M/1$ queue, with only a variation. Specifically, the mean response time of the $MMPP/M/1$ can still be a weighted sum of the mean response times of the $M_i/M/1$ queues, expressed as $R = \sum_{i=1}^{H} w_i R_i$, but the weights $w_i$ do not simply correspond to the state probabilities of the MMPP (as in the case of the average queue length). They must be scaled to keep into account the different arrival rate per each state, which reflects the fact that we are evaluating the response time over samples at discrete time points. Hence

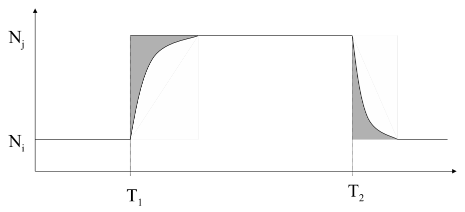

**Figure 2. Difference between the basic approximation and the original $MMPP/M/1$ process.**

$w_i = \frac{p_i \lambda_i}{\sum_{j=1}^{H} p_j \lambda_j}$. Finally, as far as the cumulative distribution and probability density functions of those parameters are concerned, they can also be derived as a weighted superposition of the corresponding functions associated with each $M_i/M/1$ queue, the weights being those described above, respectively.

With the previous simple approximation, the error made vs the real behavior of the $MMPP/M/1$ queue is given by the grayed out areas in Figure 2. One is the area comprised between the real transition curve from state $S_i$ to state $S_j$ of the $MMPP/M/1$ queue, and the stepped transition towards the steady state of $S_j$ (as hypothesized in the approximation). The other one is associated with the counterpart transition between state $S_j$ and state $S_i$. Actually, the two areas would tend to cancel each other since the error introduced when passing to a state with a higher utilization factor is positive, while the error introduced by transitions to states with lower utilization factors is negative. This means that it is not possible to obtain reasonable guarantees of underestimation or overestimation from this approximation.

To cope with this issue, a lower bound approximation of the queue length and of the response time CDF has been derived in [5] by systematically overestimating the queue length and the response time during transient periods (this is shown in Figure 1.c for what concerns the queue length). The generation of this approximation employs the same aforementioned technique, except that it requires to modify the probabilities $p_i$ to reflect the different ratios among the average times spent in each of the MMPP states. By the same considerations, the lower bound process on the response time can be also derived using that same aforementioned approximation approach, except that the weights $w_i$ are changed according to the modified $p_i$.

To generate the upper bound approximation of the queue length and of the response time CDF, the point of view can be simply inverted. Specifically, we must systematically underestimate the queue length during transient periods, as shown in Figure 1.d. Also in this case the upper bounds for the output parameters are derived by modifying the $p_i$ and $w_i$ to reflect this new perspective on the ratio between the average times spent in each of the MMPP states.

Overall, assuming that the following parameters are known: (i) $\alpha_{jk}$ - the transition rates between every $S_j$ and $S_k$ of the MMPP, (ii) $\lambda_i$ - the arrival rate of incoming requests to the queue when the MMPP is in state $S_i$, (iii) $\mu$ - the average service rate, the procedure proposed in [5] to compute lower and upper bound approximations is the following:

**1.** Evaluate the equilibrium probabilities vector $\pi = (p_1, \ldots, p_H)$ for each state of the MMPP. This requires solving the following linear system, expressing the balance and normalizing equations:

$$\pi Q = 0 \;\; ; \;\; \pi \cdot e = 1 \tag{1}$$

where $Q$ is the MMPP generator matrix and $e$ is a column vector with $H$ elements each of which is unity.

**2.** Evaluate the length of the transition period $T_{tr_{i,j}}$ associated with each transition from state $S_i$ to state $S_j$ in the MMPP arrival process, i.e., the minimum time $t$ for which the mean queue length $N(t)$ (denoting with $t = 0$ the occurrence time of the state switch) differs from the steady state mean queue length of the $M_j/M/1$ queue associated with state $S_j$, which we denote as $N_{S_j}$, by no more than an arbitrarily small value $\epsilon$. Formally:

$$T_{tr_{i,j}} = min\{t \in \mathbb{R}^+ : |N(t) - N_{S_j}| < \epsilon\} \tag{2}$$

Denoting with $P_k(t)$ the probability for the $M/M/1$ queue to contain $k$ requests at time $t$, with $P_{h,k}(t)$ the probability to contain $k$ requests at time $t$ given that it contained $h$ requests at time $t = 0$, with $P_h(0)$ the probability of $h$ queued requests at time $t = 0$, by using basic queuing theory results [11], and results in [1, 12] on continuous time analysis, $N(t)$ can be rewritten as:

$$\begin{aligned} N(t) =& \sum_{k=0}^{\infty} k P_k(t) = \sum_{k=0}^{\infty} k \sum_{h=0}^{\infty} P_h(0) P_{h,k}(t) = \sum_{k=0}^{\infty} k \sum_{h=0}^{\infty} {\rho_i}^h (1-\rho_i) P_{h,k}(t) = \\ &= \sum_{k=0}^{\infty} k \sum_{h=0}^{\infty} {\rho_i}^h (1-\rho_i) [{\rho_j}^{\frac{(k-h)}{2}} e^{-(\rho_j+1)\mu t} (I_{h-k} - I_{h+k}) + \\ &+ {\rho_j}^{-h-1} P_{0,h+k+1}(t)] \end{aligned} \tag{3}$$

where $I_n$ is the modified Bessel function of first kind having argument $2\mu\sqrt{\rho_j}t$, and $P_{0,m}(t)$, namely the probability to have $m$ requests in the queue at time $t$ given that the queue is empty at time $t = 0$, is defined as:

$$\begin{aligned} P_{0,m}(t) \;\; = \;\; & \rho_j^m e^{-(\rho_j+1)\mu t} \sum_{n=m}^{\infty} \frac{[(1+\rho_j)\mu t]^n}{n!} \\ & \sum_{u=0}^{\lfloor \frac{n-m}{2} \rfloor} \frac{n+1-2u}{n+1} \binom{n+1}{u} \frac{\rho_j^u}{(1+\rho_j)^n} \end{aligned} \tag{4}$$

**3.** For the lower bound case, evaluate the modified probabilities $p'_i$ by using the formula:

$$\begin{aligned} p'_i \quad = \quad & p_i(1 + \sum_{j}^{\lambda_i > \lambda_j} \frac{p_j \alpha_{ij}}{\sum_k^{\lambda_k > \lambda_j} p_k \alpha_{kj}} \frac{min(T_{S_j}, T_{tr_{i,j}})}{T_{S_j}} + \\ & - \sum_{j}^{\lambda_j > \lambda_i} \frac{p_j \alpha_{ji}}{\sum_k^{\lambda_k > \lambda_i} p_k \alpha_{ki}} \frac{min(T_{S_i}, T_{tr_{j,i}})}{T_{S_i}}) \end{aligned} \tag{5}$$

where $T_{S_i}$ is the expected permanence time of the MMPP arrival process in the generic state $S_i$. In other words, the modified probability $p'_i$ is generated by adding to each probability $p_i$ contributions associated with transitions from state $S_i$ to whichever state $S_j$ having lower request arrival rate, and subtracting contributions associated with transitions to state $S_i$ from whichever state $S_j$ having higher request arrival rate.

For the upper bound approximation, the above formula still holds, with the only need to invert the "$\lambda_i < \lambda_j$" and "$\lambda_i > \lambda_j$" conditions in the summations.

Finally, for both lower and upper bound approximations, the $w'_i$ values are immediately computable once $p'_i$ values are available.

# 3 On Computational Costs

## 3.1 Discussion

Among the three steps required by the above recalled approximate method for calculating lower and upper bound response time distributions of the $MMPP/M/1$ queue, Step 2 is by far the most onerous one in terms of execution time required for its completion. In fact, Step 1 only implies the solution of a linear equations' system of order $n = H$, whose asymptotic complexity is known to be upper bounded by $O(n^{2.52})$ [2] and for which a large number of efficient numerical implementations can be found in common mathematical libraries (such as Mathematica [25] or Matlab [23]). On the other hand, the computation of Step 3 requires $O(H^2)$ scalar manipulations and can be completed even more rapidly.

Conversely, the determination of the transient duration $T_{tr_{i,j}}$ in Step 2 is drastically more costly. In fact, to the best of our knowledge, no analytic closed form is known for $N(t)$ in expression (3), therefore solving expression (2) requires the following two steps:

2.1 Numerically computing $N(t)$, which implies calculation of nested sums, each one over a theoretically infinite number of terms.

2.2 Using a numerical root-finding algorithm to determine the time value satisfying the inequality in expression (2).

In order to quantify the expected execution latency required for computing Step 2, so to actually assess whether this step is the real hurdle to fast solution of the approximate model of the $MMPP/M/1$ queue presented in [5], we have developed a prototype model solver in Mathematica [25]. The solver has been based on the following two algorithmic approaches for Steps 2.1 and 2.2:

2.1 The external sum in $P_{0,m}(t)$ is stopped at iteration $i$, with value denoted as $X_i$, if the condition $|X_i - X_{i-1}| < (10^{-4} \times X_{i-1})$ is verified, where $X_{i-1}$ is the summation value at iteration $i - 1$. Hence, the computation is stopped as soon as the relative variation of the summation value is 4 orders of magnitude lower than the current value. On the other hand, the two outermost summations in expression (3) are stopped as soon as both $\sum_k P_k(t)$ and $\sum_h P_{h,k}(t)$ reach the value 0.9999, which means having an almost complete coverage of state space probability values.

2.2 The minimum finding problem in expression (2) is solved after having set $\epsilon$ to the 10% of the difference between the steady state average queue lengths associated with $S_i$ and $S_j$, namely $\epsilon = 0.1 \times |N_{S_j} - N_{S_i}|$. This is done by numerically computing the smallest root of the equation:

$$N(t) - N_{S_j} - 0.1 \times (N_{S_j} - N_{S_i}) = 0 \tag{6}$$

To this purpose we have developed a specialized search algorithm which first determines two time instants $t'$ and $t''$ such that $t' < t''$ and that $t''$ satisfies expression (2) whereas $t'$ does not, and then iteratively restricts the $[t', t'']$ interval, while ensuring that only one of the new extremes keeps on satisfying expression (2). The determination of $t'$ and $t''$ is performed via a variant of the *Secant* method [4], where the value for the next iteration is chosen twice as distant as the one predicted by the original Secant method, so to reduce the number of iterations required for finding a time value falling within distance $\epsilon$ from $N_{S_j}$ (recall that $N(t)$ asymptotically converges to $N_{S_j}$). The initial guess for $t'$ is set to $\frac{|N_{S_j} - N_{S_i}|}{\lambda_j}$ for ramp-up transitions (i.e. transitions associated with an increase in the arrival rate), and to $\frac{|N_{S_j} - N_{S_i}|}{\mu}$ for ramp-down transitions (i.e. transitions associated with a decrease in the arrival rate). These two values represent lower bounds on transient durations, which are determined on the basis of the assumption that, during the transient phase, either no request completion (this holds for ramp-up transitions), or no request arrival (this holds for ramp-down transitions) takes place. On the other hand, the initial guess for $t''$ is set to 5 times the guessed value of $t'$, as such a choice was experimentally found to typically enable termination of the Secant method in a very small number of iterations.

Next, the values of $t'$ and $t''$ found via the Secant method are used as the input for the *Regula Falsi Method*, an effective root finding algorithm combining

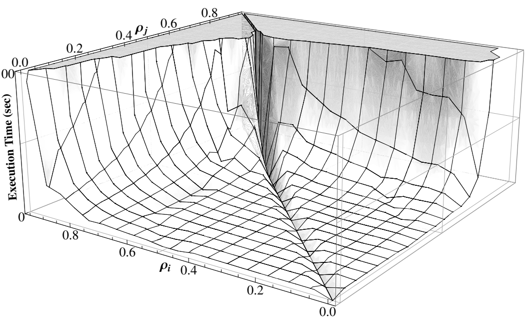

**Figure 3. Execution times for the computation of transient duration $T_{tr_{i,j}}$ ($\mu$ set to 1).**

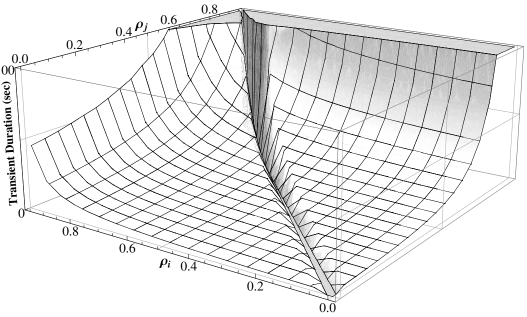

**Figure 4. Transient duration $T_{tr_{i,j}}$ ($\mu$ set to 1).**

features of both the Bisection method and the Secant method [4], which is stopped as soon as it determines a solution for expression (6) with at least 1% accuracy.

By expressions (2), (3) and (4), the transient duration only depends on the values of $\rho_i$, $\rho_j$ and $\mu$. Hence, to evaluate the run-time cost for Step 2 we have repeatedly executed the model solver by setting $\mu = 1$ and varying the utilization factors $\rho_i$, $\rho_j$ in the whole plausible domain $[0, 1) \times [0, 1)$, using the value 0.05 as the basic step for determining discrete samples in that domain. The outcoming execution times, obtained by running the model solver on a Windows 2003 Server machine equipped with an Intel Xeon 2.0 GHz CPU and 4 GB of RAM, are shown in Figure 3.

The reported data clearly highlight how direct numerical solution methods for the computation of transient durations can be slow, especially when we need to evaluate transient durations for transitions from/to states associated with high utilization factors. In fact, execution times remain lower than 10 seconds only when transitions occur between states $S_i$ and $S_j$ of the MMPP arrival process, such that the corresponding utilization factors $\rho_i$ and $\rho_j$ are both within the range [0,0.5]. On the other hand, computation time exceeds 100 seconds when $\rho_i$ and $\rho_j$ are in the range $[0.75, 0.85]$, and gets even longer than several hours when they fall in the range $[0.9, 1)$ (this range is not explicitly shown in the plots).

By profiling the model solver implementation, we found out that such an increase in the execution times is due to a very rapid increase of the cost to compute Step 2.1, when transitions involving states with high utilization factors occur. The reason is that transient durations get very long for transitions involving MMPP states with high utilization factors, as confirmed by the experimental data shown in Figure 4 (obtained by means of the same model solver), which causes a large increase of the number of iterations required to compute the mean queue length in Step 2.1. Conversely, the above described solution strategy for Step 2.2 turned out to never require more than 10 iteration steps, and to terminate, on the average, after 5 iterations.

### 3.2 Speeding up the Model Solving Procedure

By the results in the previous section, we get that the approximate model of the $MMPP/M/1$ queue presented in [5] cannot be used in combination with classical model solving techniques in case fast output is required in order to support performance modeling and prediction. As pointed out this might be extremely useful in, e.g., interactive what-if analysis applications and real-time autonomic dynamic reconfiguration schemes.

To bypass this drawback, just due to the need for explicit computation of the transient durations associated with state switches in the MMPP arrival process, we have envisaged an alternative way, which is based on a further approximation step, this time involving the transient duration value. Specifically, we rely on the idea to provide an approximate (although accurate) description of transient durations vs main parameters characterizing the $MMPP/M/1$ queue, which can be immediately used to speed up computation of Step 2 in the aforementioned procedure.

As already highlighted (see expressions (2)-(4)), the transient duration when a state switch occurs from $S_i$ to $S_j$ in the MMPP arrival process, exclusively depends on the values of $\rho_i$, $\rho_j$ and $\mu$. Hence, in order to provide an accurate approximation for the transient duration value in generic settings, we have performed an extended sensitivity study where, beyond $\rho_i$ and $\rho_j$ values, we have also varied the value of the service rate $\mu$. In Figure 5 we plot the duration $T_{tr_{i,j}}$ for both ramp-up and ramp-down transitions as a function of $\mu$ when considering a set of $\Delta\rho = |\rho_i - \rho_j|$ values widely spanning in the whole plausible range $[0, 1)$.

From these plots we can observe that once the values of $\rho_i$ and $\rho_j$ are fixed, the duration of transient periods rapidly decreases vs increasing values of $\mu$. More precisely, we found that $T_{tr_{i,j}}$ can be very closely fitted by means of a hyperbola of equation $\frac{k}{\mu}$, whose $k$ parameter exclusively depends on $\rho_i$ and $\rho_j$. In particular, the value of $k$ is equal to the value of the transient duration associated with a tran-

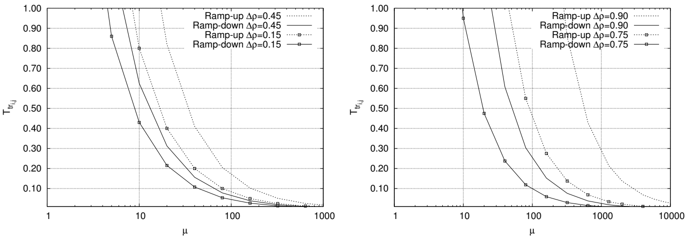

**Figure 5. Transient duration $T_{tr_{i,j}}$ vs $\Delta\rho$ and $\mu$.**

sition from utilization factor $\rho_i$ to utilization factor $\rho_j$, as observed for the case of $\mu = 1$. This reference transient duration value will be denoted as $T^*_{tr_{i,j}}$. Actually, the fitting procedure determining the value of the hyperbola's $k$ parameter, has been based on the nonlinear least-squares (NLLS) Marquardt-Levenberg algorithm, which converged after a few iterations and showed negligible residual error. The outcoming fitting curves are shown in Figure 6.

On the basis of the previous results, we can derive a computational effective approach for the evaluation of the duration of transient phases associated with a switch in the Modulating Markov Process characterizing the $MMPP/M/1$ queue from state $S_i$ to state $S_j$, which can be adopted for whichever value of the service rate $\mu$. The approach is based on the assumption that the $n \times n$ matrix $M_n$ associated with the transient duration for the reference case $\mu = 1$ is available (i.e. it has been pre-computed una-tantum). In particular, the entry $< i, j >$ of this matrix stores, for the case $\mu = 1$, the duration of the reference transient period $T_{tr_{i,j}}$ associated with a switch from utilization factor $\rho_i = \frac{i-1}{n}$ to a state with utilization factor $\rho_j = \frac{j-1}{n}$, where $n$ is a fixed number of discrete samples within the interval $[0, 1)$.

Below we describe such a computational effective approach:

- To compute the duration of a ramp-up transition (case $\rho_i < \rho_j$), divide the entry in position $< \lfloor \rho_i n \rfloor + 1, \lceil \rho_j n \rceil + 1 >$ of the matrix $M_n$ associated with the reference case $\mu = 1$ by the real $\mu$ value characterizing the service rate.

- To compute the duration of a ramp-down transition (case $\rho_i > \rho_j$), divide the entry in position $< \lceil \rho_i n \rceil + 1, \lfloor \rho_j n \rfloor + 1 >$ of the matrix $M_n$ associated with the reference case $\mu = 1$ by the real $\mu$ value characterizing the service rate.

It is straightforward to see that, assuming the availability of the pre-sampled $M_n$ matrix, the above approach is characterized by time complexity $O(1)$.

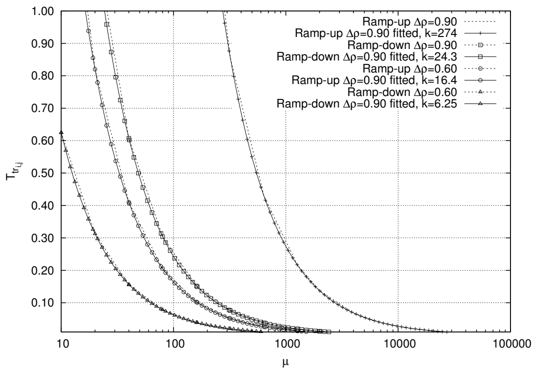

**Figure 6. Fitting $T_{tr_{i,j}}$ through hyperbolas of equation $\frac{k}{\mu}$.**

## 4 The Case Study

In this section we aim at evaluating the performance benefits from the proposed model solving approach in realistic settings for what concerns the parameters space of the $MMPP/M/1$ queue, representative of a GRID network server.

To this end, we compare the latency of our fast model solving technique, whose implementation still relies on Mathematica, with the latency for obtaining the queue statistics via either simulative analysis or by applying the most efficient exact solution technique (to the best of our knowledge) presented in [9], for which we have developed an implementation using again the Mathematica library. Actually, the main computational steps required by this exact solution are the following:

1. Determine the MMPP equilibrium probabilities by numerically solving a system of $H$ linear equations (where $H$ is the number of states of the MMPP process).

2. Use the spectral expansion method [14] to derive the steady-state probability distribution of the queue. This implies (1) numerically computing the eigenvalues/eigenvectors of a sparse square matrix having size $2H \times 2H$, and (2) numerically solving a linear equations system of size $(2H + 1) \times (2H + 1)$

3. Compute the Laplace transform of the response distribution. This requires (1) $O(H)$ symbolic operations (i.e. additions and multiplications), as well as two symbolic inversions, involving $H \times H$ polynomial matrices, and (2) symbolically reducing the Laplace transform of the response time distribution into partial fractions.

4. Pattern match each term resulting from partial fraction decomposition in order to constructively compute the

Laplace anti-transform and obtain the response time distribution in the time domain. This can be done via a single iteration over the $O(H^2)$ terms deriving from the partial fraction decomposition.

On the other hand, concerning the simulative approach, we have developed an optimized discrete event simulation program for the $MMPP/M/1$ queue, exclusively relying on C technology, whose execution latency is determined by stopping the run as soon as the incrementally computed statistics on the simulated response time CDF vary by no more than $1\%$.

We set the MMPP arrival process parameters on the basis of the results reported in [15]. This work has shown, via real traces analysis, the feasibility to model incoming traffic to a GRID server just by means of the MMPP model. Specifically, according to the data reported in [15], the incoming traffic of the analyzed GRID server can be modeled by a two-state MMPP, where the transition rate $\alpha_{12}$, from state $S_1$ to state $S_2$ is 0.17, while the transition rate $\alpha_{21}$ from state $S_2$ to state $S_1$ is 0.08. Also, the request arrival rates $\lambda_1$ and $\lambda_2$, associated with states $S_1$ and $S_2$ are 22.1 and 7.16.

We have used these parameters to build a test scenario where the performance of the GRID server is evaluated in case of (i) a single source of jobs, (ii) two uncorrelated job sources, and (iii) three uncorrelated job sources. In all the cases the job sources are described on the basis of the previously mentioned trace based study. Also, while case (i) represents a basic performance analysis scenario, case (ii) and case (iii) may be representative of more critical settings where different job sources need to be de-routed to a single GRID site due to, e.g., critical events in the GRID infrastructure, and the final performance achievable after derouting towards that single GRID site must be assessed. In terms of MMPP arrival process, the aforementioned cases correspond to situations where the number of MMPP states is equal to 2, 4 and 8, respectively. Also, the GRID server processing time has been set to achieve a scenario where the server capacity is saturated at the $75\%$ when the three job sources simultaneously exhibit their peak rate.

As a final preliminary observation, we also want to show whether gains (if any) in execution speed for solving the approximate model of the $MMPP/M/1$ queue via our fast method arise at the expense of excessive accuracy loss (due to either the original approximation in the $MMPP/M/1$ model or the approximation step associated with our fitting-based approach to the determination of transient duration) compared to both the simulative study and the aforementioned exact solution approach. In other words, we want to evaluate whether synergic adoption of the original approximate model in [5] and the associated model solving method provided in this paper can be effective for both computation latency and accuracy. Hence, beyond performance results, we also show the output statistics for the $MMPP/M/1$ queue obtained via the three compared solutions.

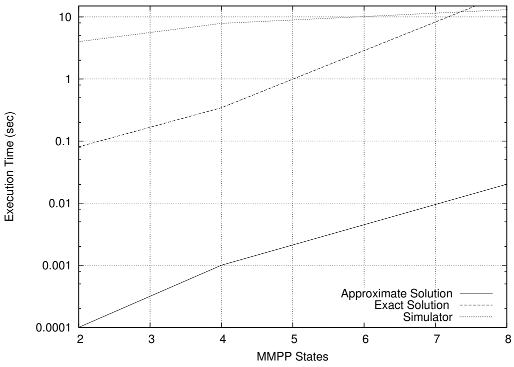

**Figure 7. Execution times for the three compared approaches.**

Data related to the latency of each solution, while varying the number of states of the MMPP arrival process, are shown in Figure 7. They have been evaluated on the same platform (i.e. a machine with a Xeon 2 GHz CPU and 4 GB RAM) used for the experimental study in Section 3.1. By these data we get that our fast model solving procedure always shows execution time lower than (or at most up to) 11 milliseconds, while the simulative solution shows latency from 5 to 10 seconds. Concerning the exact method in [9], it shows execution time on the order of 100 milliseconds only for the MMPP with 2 states. On the other hand, its performance rapidly decreases when the number of states in the MMPP arrival process gets increased. Specifically, for 8 states the latency of the exact solution method is even greater than the one provided by the optimized simulation approach. Overall, our proposal reveals a definitely more efficient and scalable alternative, having the ability to support fast (e.g. real-time) performance analysis of servers modeled via the $MMPP/M/1$ queue.

As said above, we want to also show that the whole approach, based on both the approximate model presented in [5] and the model solving solution introduced in this paper, provides accurate results. This is supported by the data in Figure 8 and in Figure 9 showing, respectively, the response time CDF for 2/4 states and for 8 states in the MMPP arrival process. By these results, the bounds achieved via the approximate model and its associated fast solving procedure are quite close to the results achieved via the exact method (simulative results are not shown since they are practically identical to those achieved via the exact method). This denotes adequate accuracy from the approximate approach for such a realistic, representative test case related to GRID oriented infrastructures.

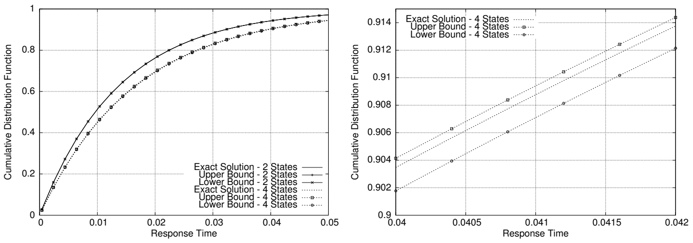

**Figure 8. CDFs for the MMPPs with 2 and 4 states (left) - Zoom on the CDF for the 4 states case (right).**

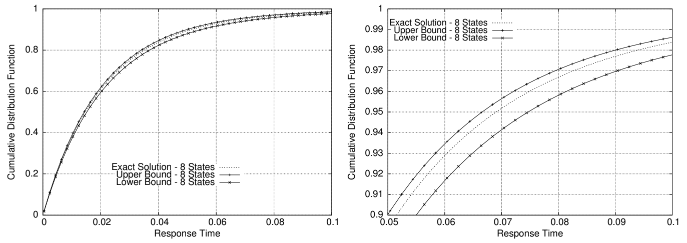

**Figure 9. CDF for the MMPP with 8 states (left) - Associated zoom (right).**

## 5 Conclusions

In this article we have complemented one of our previous studies related to approximate solutions for the $MMPP/M/1$ queue, which is a type of queue particularly interesting to model the performance of networked servers. Specifically, we have introduced a fast model solving procedure associated with the original approximate approach, which is based on ad-hoc fitting techniques allowing the avoidance of most of the computational costs associated with the approximate solution. Via an experimental study in the context of a networked GRID application, we have also quantified the increased performance and scalability of our model solving approach when compared to both classical simulative approaches and exact solution methods for the $MMPP/M/1$ queue.

## References

[1] J. Abate and W. Whitt, "Transient Behavior of the M/M/1 Queue Via Laplace Transforms", Advances in Applied Probability, Vol.20, No.1, 1988, pp.145-178.

[2] A. Bojańczyk, "Complexity of Solving Linear Systems in Different Models of Computation", SIAM Journal on Numerical Analysis, Vol.21, No.3, 1984.

[3] L. Breslau, P. Cao, L. Fan, G. Phillipps and S. Shenker, "Web Caching and Zipf-like Distributions: Evidence and Implications", Proc. of IEEE INFOCOM, 1999.

[4] R.L. Burden, J.D. Faires (2000), "Numerical Analysis, (7th Ed)", Brooks/Cole.

[5] B. Ciciani, A. Santoro and P. Romano, "Approximate Analytical Models for Networked Servers Subject to MMPP Arrival Processes", Proc. 6th IEEE International Symposium on Network Computing and Applications (NCA), 2007, pp.25-32.

[6] M. Crovella and A. Bestavros, "Self-similarity in World-Wide-Web traffic: Evidence and possible causes.", IEEE/ACM Transactions on Networking, Vol.3, No.3, 1994, pp.226-244.

[7] W. Fischer and K. Meier-Hellstern, "The Markov-modulated Poisson process (MMPP) cookbook", Performance Evaluation, Vol.18, No.2, 1993, pp.149-171.

[8] Y. Fujita, M.Murata and H. Miyahara, "Analysis of Web Server Performance Toward Modeling and Performance Evaluation of Web Systems", Proc. of IEEE SICON, 1998.

[9] P.G. Harrison and H. Zatschler, "Sojourn Time Distributions in Modulated G-Queues with Batch Processing", International Conference on Quantitative Evaluation of Systems (QEST), 2004, pp.90-99.

[10] A. Horvath and M. Telek, "A Markovian Point Process Exhibiting Multifractal Behavior and Its Application To Traffic Modeling", Proc. of MAM4, Adelaide, Australia, 2002.

[11] L. Kleinrock, "Queuing Systems", Volume I: Theory, John Wiley & Sons, 1975.

[12] W. Leguesdron, J. Pellaumail, G. Rubino and B. Sericola, "Transient analysis of the M/M/1 queue", Advances in Applied Probability, No.25, 1993.

[13] W. Leland, M.S. Taqqu, W. Willinger and D.V. Wilson, "On the self-similar nature of the Ethernet traffic (extended version)", IEEE/ACM Transactions on Networking, Vol.2, No.1, 1994, pp.1-15.

[14] I. Mitrani,"Spectral Expansion Solutions for Markov-Modulated Queues", Performance Evaluation of Complex Systems: Techniques and Tools, Performance Tutorial Lectures, Springer-Verlag, 2002, pp.17–35.

[15] H. Li, M. Muskulus and L. Wolters, "Modeling Job Arrivals in a data-intensive Grid", Proc. 12th Workshop on Job Scheduling Strategies for Parallel Processing, 2006.

[16] Z. Liu, N. Niclausse and C. Jalpa-Villanueva, "Traffic Model and Performance Evaluation of Web Servers", Performance Evaluation Journal, Vol.46, No.2-3, 2001, pp.77-100.

[17] A. Nogueira, P. Salvador, R. Valadas and A. Pacheco, "Fitting self-similar traffic by a superposition of MMPPs modeling the distribution at multiple time scales", IEICE Transactions, Vol.E87-B, No.3, 2004, pp. 678-688.

[18] V. Paxson and S. Floyd, "Wide Area Traffic: the Failure of Poisson Modeling", IEEE/ACM Transactions on Networking, Vol.3, No.3, 1995, pp. 226-244.

[19] A. Riska, M. Squillante, S. Yu, Z. Liu and L. Zhen, "Matrix-Analytic Analysis of a MAP/PH/1 Queue Fitted to Web Server Data", Proc. of Int. Conference on Matrix Analytic Methods in Stochastic Models, 2002.

[20] P. Rodriguez, C. Spanner and E.W. Biersack, "Analysis of Web Caching Architectures: Hierarchical and Distributed Caching", IEEE/ACM Transactions on Networking, Vol.9, No.4, 2001, pp.404-418.

[21] P. Salvador, R. Valadas and A. Pacheco, "Multiscale Fitting Procedure using Markov Modulated Poisson Processes", Telecommunication Systems, Springer, Vol.23, No.1-2, 2003, pp. 123-148.

[22] W. Willinger, M.S. Taqqu, R. Sherman and D.V. Wilson, "Self-similarity through high-variability: Statistical analysis of Ethernet LAN traffic at the source level", IEEE/ACM Transactions on Networking, Vol.5, No.1, 1997, pp 71-86.

[23] The MathWorks, "MATLAB 7", 2007.

[24] T. Yoshihara, S. Kasahara and Y. Takahashi, "Practical Time-Scale Fitting of Self-similar traffic with Markov Modulated Poisson process", Telecommunication Systems, Vol.17, No.1-2, 2001, pp 185-211.

[25] Wolfram Research Inc., "Mathematica Edition: Version 6.0", 2007.