The M/M/1 Fork-Join Queue with Variable Sub-Tasks

Elizabeth Varki Arif Merchant Hui Chen

Abstract

The fork-join queue models parallel resources where arriving jobs divide into various number of sub-tasks that are assigned to unique devices within the parallel resource. Each device in the parallel resource is modeled by $\mathsf{M}/\mathsf{M}/1$ queueing servers. A job completes execution and departs the parallel resource after all its sub-tasks complete execution. This paper analyzes N-server fork-join queues where arriving jobs divide into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks that are assigned to unique servers of the fork-join queue. There is no known closed-form solution for $\mathsf{N} > 2$ fork-join queues. The paper presents an O(log K) algorithm for computing the mean response time pessimistic and optimistic bounds and for computing the mean response time approximation of the fork-join queue. The error bounds for the response time bounds and approximation are presented.

Index Terms: fork-join synchronization, performance evaluation, parallel computer and storage systems.

1 Introduction

Modern computer systems rely on parallel resources, such as multiple processors and disk arrays, to satisfy the performance requirements of its application programs. For example, the response time of an application program is reduced by concurrently executing sub-parts of the program on multiple processors. Similarly, the I/O throughput of a storage system is increased by accessing data from multiple disks. The jobs submitted to a parallel resource are divided into sub-tasks that are each submitted to separate devices within the parallel resource. A job completes execution and departs the parallel resource only after all its sub-tasks complete.

From a performance analysis viewpoint, the N devices of a parallel resource are modeled by N parallel queueing servers jointly referred to as a *fork-join* queue. Figure 1 presents a fork-join queue. Jobs arrive at the fork-join queue at rate λ. Upon arrival, each job divides (at the fork point) into K identical sub-tasks, where $1 \leq \mathsf{K} \leq \mathsf{N}$. Based on a pre-defined allocation policy, each of these K sub-tasks is submitted to a unique server within the fork-join queue. The probability that a particular server is assigned a sub-task of an arriving job is given by $\mathsf{server\_access\_probability}$. The queueing discipline is first-come-first-served. The sub-tasks at each server are serviced at rate μ. When a sub-task completes execution, it will wait (at the join point) until all its sibling sub-tasks complete execution. A job completes execution and departs the fork-join queue after all its sub-tasks complete. In this paper, we analyze fork-join queues with exponential inter-arrival and service times. That is, the N servers of the fork-join model are $\mathsf{M}/\mathsf{M}/1$ queues with synchronized arrivals.

Due to the wide-spread use of parallelism in computer and storage systems, the fork-join queue has been studied extensively. Section 2 summarizes the fork-join literature. An exact analysis of the fork-join queue is presented only for 2-server fork-join queues [2, 7, 18, 23]. There is no known closed-form solution for $\mathsf{N} > 2$ server fork-join queues. Hence, the performance measures of fork-join queues with $\mathsf{N} > 2$ servers is computed using approximation and bounding techniques. This paper also presents mean performance bounds and approximations for the fork-join queue. The **contribution** of this paper is that it is the first to analyze the $\mathsf{M}/\mathsf{M}/1$ N-server fork-join queue where jobs divide into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks. All previous papers on $\mathsf{M}/\mathsf{M}/1$ fork-join queues assume that every job divides into N sub-tasks. It is important to analyze fork-join queues where jobs divide into $\mathsf{K} < \mathsf{N}$ sub-tasks because such fork-join queues model the behavior of real parallel systems. For example, a parallel job may only be executed on some of the

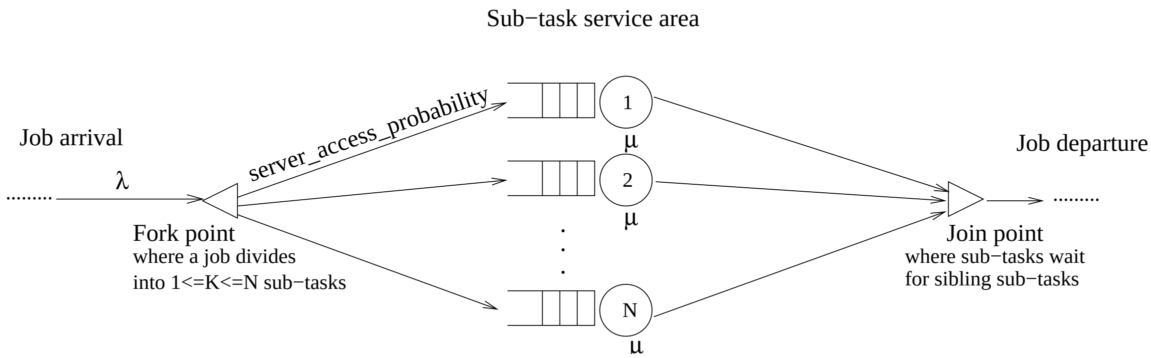

Figure 1: Fork-join queue

processors of a multi-processor system. Similarly, the I/O requests submitted to a disk-array may only access some of the disks in the array.

This paper presents simple pessimistic and optimistic mean response time bounds and a mean response time approximation for the fork-join queue with $\mathsf{K} \leq \mathsf{N}$ sub-tasks. The complexity of the response time bounds and approximation computation is O(log K). This paper also presents the error bound for the response time approximation. Comparison of the mean response time approximations against simulations show an average error of 5% for K (*i.e.*, degree of parallelism) varying from 2 to 100 and parallel resource utilization varying from 0.1 to 0.9.

The remainder of this paper is organized as follows. Section 2 summarizes related work on performance analysis of fork-join queues. Section 3 presents a Markov analysis of the $\mathsf{M/M/1}$ fork-join queue. Sections 4 and 5 use this Markov analysis to derive the mean response time bounds and response time approximation of fork-join queues. Finally, conclusions and future work are presented in Section 6.

2 Related Work

Several papers study parallel (fork-join) queues and propose tools for analyzing their performance. Exact performance measures have been derived only for fork-join queues with two servers [2, 7, 18, 23]. Of these, [2] and [7] derive exact steady state distribution for two server fork-join queues in open networks, and [18] and [23], respectively, derive exact mean response times of two server fork-join queues in open and closed networks. Results in [2] assumes general service time distribution, and results in [7], [18], and [23] assume exponential service time distributions. Due to the difficulty of analyzing fork-join queues exactly, all studies on fork-join queues with three or more servers are approximation or bounding analysis. Heidelberger and Trivedi [8] consider a closed queuing network in which jobs divide into two or more asynchronous tasks. The join synchronization is not modeled. The service centers are of a type described in the BCMP theorem. They develop an iterative method for solving a sequence of product-form models. In [9], the model is expanded to include the join synchronization. Nelson and Tantawi [18] consider a scaling approximation technique to analyze the mean response time of an open homogeneous fork-join queue with exponential service time distributions. They assume that the mean response time increases at the same rate as the increase in the degree of parallelism. Closed-form approximation expressions for the mean response time are developed. An extension of this approximation to heavy traffic, relying on a light traffic interpolation technique, is developed by Makowski and Varma [17]. Kim and Agrawala [10] analyze waiting times for two server open, homogeneous fork-join queues with exponential and 2-stage Erlang service time distributions. In [15, 16], Lui, Muntz, and Towsley present a bounding technique for an open, homogeneous fork-join network with a k-stage Erlang distribution. Response time bounds are obtained for acyclic fork-join queuing networks by Baccelli et. al. [3] using stochastic ordering principles and association of random variables. In [4], Baccelli and Liu propose a new class of queuing models for evaluating the performance of parallel systems. Using the concept of associated random variables, Kumar and Shorey [11] obtain response time bounds for an open fork-join model in which a job forks into a random number of tasks. Service times are drawn from a general distribution. Balsamo, Donatiello, and Van Dijk [5] propose a matrix-geometric algorithmic approach for computing performance bounds of open heterogeneous fork-join systems. Ray [6] uses a dynamic-bubblesort analysis technique to develop a response time bound for fork-join queues with exponential service times. Varki [23] presents a response time approximation for fork-join queues that generalizes the response time expression for single server queues.

There are fewer papers on fork-join queues in closed networks. Almeida and Dowdy [1] propose an iterative technique for obtaining lower performance bounds of closed fork-join networks with exponential service times. No proofs for the technique are presented. Liu and Perros [13, 14] propose an approximation procedure based on decomposition and aggregation for analyzing a closed queuing system with K-sibling fork-join queues. Their method provides an upper bound for mean response time. In [22], Varki develops a mean-value analysis technique for closed fork-join parallel networks. The fork-join structure is studied with relation to parallel storage systems (RAID) in [12, 19, 20].

All but one of the papers on open fork-join queues assume that arriving jobs split into exactly N sub-tasks upon arrival at a N server fork-join queue. In [11], mean performance bounds are computed for M/G/1 fork-join queues where jobs divide into random number of sub-tasks. This is the first paper to analyze $\mathsf{M/M/1}$ fork-join queues where jobs divide into $\mathsf{K} \leq N$ sub-tasks. It is important to compute performance measures of such fork-join queues since these queues model the behavior of parallel computer resources such as disk-arrays and multi-processor computers. Furthermore, the response time computations presented here are computationally simple and scale well with increasing parallelism and increasing load.

3 Markov Analysis

We use Markov state diagrams to analyze the fork-join queue. This analysis is then used in Sections 4 and 5 to derive response time bounds and approximations. Let $\mathsf{P_N}$ represent a N-server fork-join queue where every arriving job divides into N sub-tasks that are assigned to the N servers. The state of $\mathsf{P_N}$ is represented by the vector $(\mathsf{n_1}, \cdots, \mathsf{n_N})$, where $\mathsf{n_i}$ represents the number of sub-tasks at a server of $\mathsf{P_N}$. Since the N service centers are identical, the $\mathsf{n_i}$'s are ordered such that $\mathsf{n_1} \leq \mathsf{n_2} \leq \cdots \leq \mathsf{n_N}$. Thus, $\mathsf{n_N}$ is equal to the number of tasks at the longest server queue. The server queueing discipline is first-come-first-served and each job divides into N sub-tasks, one for each server. Hence, $\mathsf{n_N}$ also represents the total number of jobs in $\mathsf{P_N}$.

Figure 2 presents the Markov diagram of $\mathsf{P_2}$. The diagram maps the states of $\mathsf{P_2}$ when there are 0, 1, 2, and 3 jobs in $\mathsf{P_2}$. Column i $(i = 0, 1, 2, 3)$ of the diagram represents states with i jobs in the fork-join queue. Row i $(i = 0, 1, 2, 3)$ of the diagram represents states with i sub-tasks at the join point. The horizontal transition arcs represent the arrival of jobs at $\mathsf{P_2}$. The downward transition arcs, $\vec{\mathsf{t_1}}$, represent the movement of a sub-task to the join point. The diagonal transition arcs, $\vec{\mathsf{t_2}}$, represent the movement of the last sub-task of a job to the join point at which instant this job departs $\mathsf{P_2}$. The time spent by a job in $\mathsf{P_2}$ can be factored into two phases, namely, $\mathsf{phase_2}$ and $\mathsf{phase_1}$, in order. In $\mathsf{phase_2}$, two sub-tasks of the job are waiting for, or receiving, service at the service centers of $\mathsf{P_2}$. In $\mathsf{phase_1}$, only one sub-task of the job is at the service center while its sibling sub-task waits at the join point.

Next, we analyze the Markov state diagram of $\mathsf{P_3}$ given in Figure 3. The horizontal transition arcs again represent the movement of jobs into $\mathsf{P_3}$ at rate λ. The arcs $\vec{\mathsf{t_k}}$ $(k = 1, 2, 3)$ represent the movement of the k^{th} sub-task of a job to the join point. The response time of a job in $\mathsf{P_3}$ can be factored into three phases. In general, the response time of a job in $\mathsf{P_N}$ can be factored into N phases, namely, $\mathsf{phase_N}, \cdots, \mathsf{phase_1}$, in order. A phase, $\mathsf{phase_k}$, represents the situation when k sub-tasks of the job are at the service centers. A phase, $\mathsf{phase_k}$, ends with the movement of one of

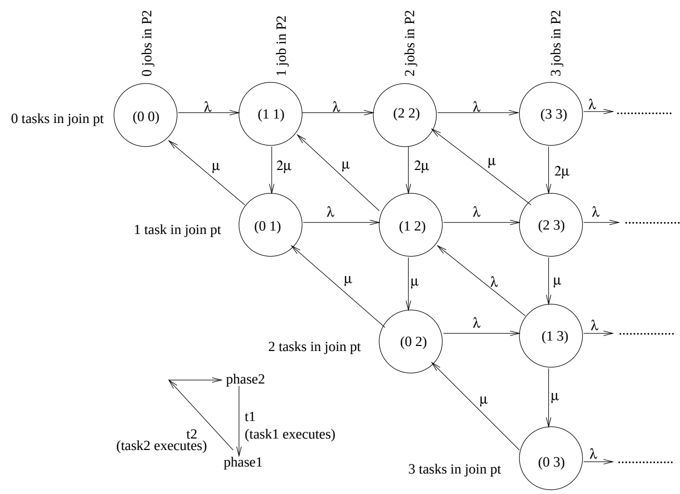

Figure 2: Markov diagram of P_2

the executing sub-tasks to the join point, at which point the corresponding job moves to $\mathsf{phase}_{\mathsf{k}-1}$ of its response time.

The time spent completing each phase of a job's response time in $\mathsf{P_N}$ can be viewed as the time spent getting service at N non-parallel queueing centers $\mathsf{Serial_N}$, $\mathsf{Serial}_{\mathsf{N}-1}, \cdots, \mathsf{Serial}_1$, in order [23]. A job at service center $\mathsf{Serial_k}$ is in $phase_k$ of its response time. Let $\mathsf{Serial\_P_N\_model}$ represent the $\mathsf{Serial_N}$, $\mathsf{Serial}_{\mathsf{N}-1}, \cdots, \mathsf{Serial}_1$ model of $\mathsf{P_N}$. In the $\mathsf{Serial\_P_N\_model}$, let $n_{\mathsf{Serial_k}}$ represents the number of jobs in the $\mathsf{Serial_k}$ queue. By construction, $n_{\mathsf{Serial_k}}$ represents the number of jobs in $\mathsf{P_N}$ with k active sub-tasks. A state $(n_{\mathsf{Serial_N}}; n_{\mathsf{Serial_{N-1}}}; \cdots; n_{\mathsf{Serial_2}}; n_{\mathsf{Serial_1}})$ of $\mathsf{Serial\_P_N\_model}$ is equivalent to the state $(n_{\mathsf{Serial_N}}, n_{\mathsf{Serial_N}} + n_{\mathsf{Serial_{N-1}}}, n_{\mathsf{Serial_N}} + n_{\mathsf{Serial_{N-1}}} + n_{\mathsf{Serial_{N-2}}}, \cdots, n_{\mathsf{Serial_N}} + .. + n_{\mathsf{Serial_2}}, n_{\mathsf{Serial_N}} + .. + n_{\mathsf{Serial_1}})$ of $\mathsf{P_N}$. The next example illustrates this mapping:

Example 1 The state (3,3,3) of P_3 represents the state when all sub-tasks of the 3 jobs within P_3 are at the servers. Thus, all 3 jobs are in the first phase (phase_3) of their response time, which implies that all 3 jobs are at Serial_3. This state is equal to the state (3;0;0) of the serial model.

The state (0,3,4) of P_3 represents the state with job-1 in phase_1 of its response time with 1 active sub-task; job-2, job-3, and job-4 are in phase_2 of their response time with 2 active sub-tasks each. Thus, job-1 is at server Serial_1, job-2, job-3, and job-4 are at server Serial_2. This is equivalent to state (0;3;1) of the serial model. □

There is a 1-1 and onto mapping from the state space of the $\mathsf{Serial\_P_N\_model}$ to the state space of $\mathsf{P_N}$. (That is, every state in $\mathsf{P_N}$ can be mapped to a state in $\mathsf{Serial\_P_N\_model}$, and vice-versa.) Set the rates along the transition arcs in the Markov diagram of $\mathsf{Serial\_P_N\_model}$ to be equal to the rates along the corresponding transition arcs of $\mathsf{P_N}$. Figure 4 presents the Markov diagram of the $\mathsf{Serial\_P_N\_model}$. By construction, $\mathsf{P_N}$ and $\mathsf{Serial\_P_N\_model}$ have identical Markov processes and are equivalent models.

The advantage of the serial model is that the fork-join queue, $\mathsf{P_N}$, can be analyzed from the viewpoint of a job's response time at the parallel queue. If a job arriving at $\mathsf{P_N}$ divides into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks, then the response time of a job can be mapped to the response time of a job at the $\mathsf{Serial\_P_K\_model}$, the serial model equivalent to $\mathsf{P_K}$. In the next section, we use this serial mapping of the fork-join queue to compute mean performance bounds and approximate performance measures of the fork-join queues.

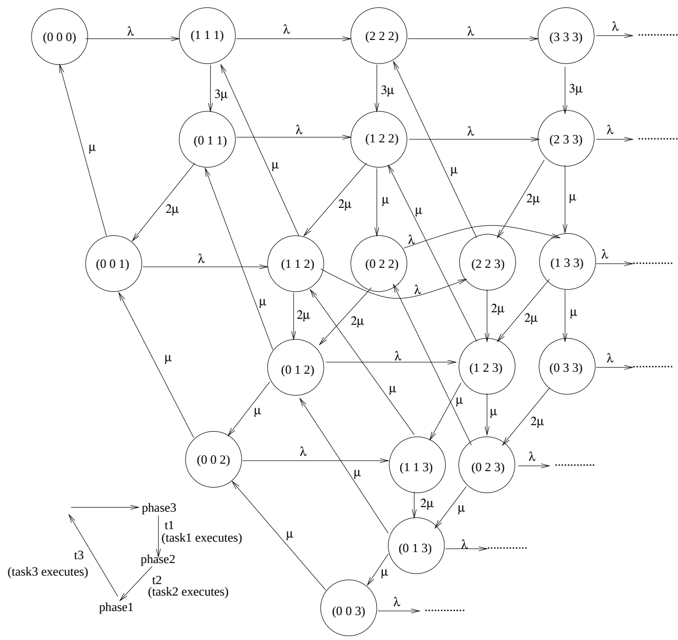

Figure 3: Markov diagram of $\mathsf{P_3}$

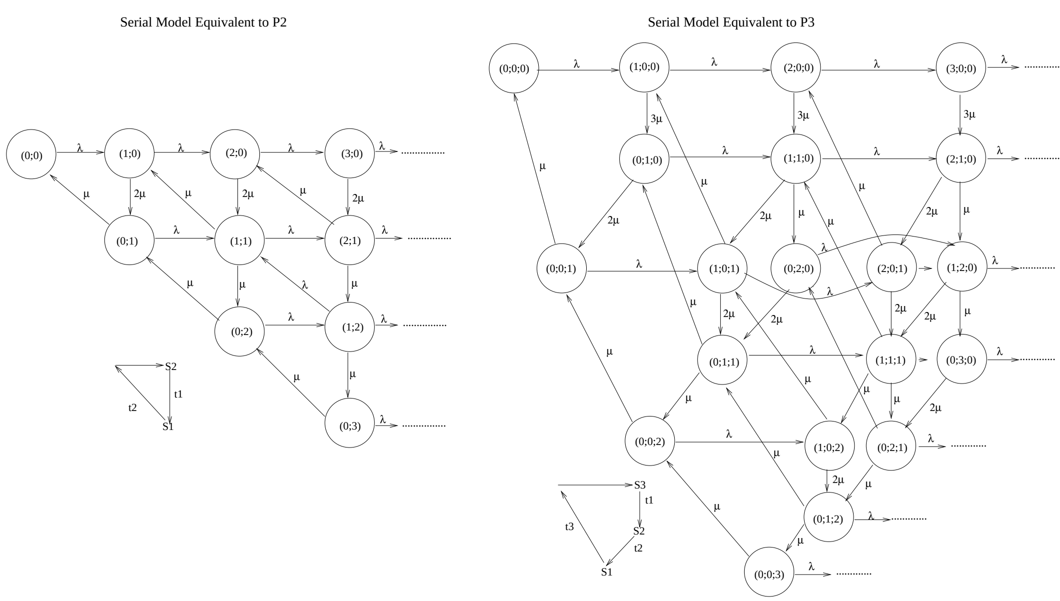

Figure 4: Markov diagram of $\mathsf{Serial\_P_N\_model}$

4 N Sub-Tasks

This section presents the derivation of the response time bounds and the response time approximation for the N-server fork-join queue where every job divides into exactly N sub-tasks. Before presenting this derivation, we briefly explain the harmonic number and the partial sum of a sequence since both are used in the remainder of this paper.

4.1 Harmonic Number and Partial Sums

A well known result in probability theory is that when there are K identical sub-tasks executing concurrently on exponential servers, the mean time taken to finish executing the K sub-tasks is $\mathsf{H}_\mathsf{K}/\mu$, the mean of the K^{th} order statistic of sub-task execution times [21]. Here, $1/\mu$ represents the mean execution time of a sub-task, and

$$\mathsf{H}_\mathsf{K} = 1 + \frac{1}{2} + \cdots + \frac{1}{\mathsf{K}}$$

represents the K^{th} harmonic number. Since the response time of a job at a fork-join queue is the time taken from arrival instant until all the K sub-tasks of the job complete execution, the harmonic number plays a key role in the response time computation of fork-join queues.

We now define the K^{th} partial sum, $\mathsf{Sum}_{a_\mathsf{K}}$, of a sequence a. Consider a sequence $a = \left\langle \frac{1}{a_1}, \frac{1}{a_2}, \frac{1}{a_3}, \cdots, \frac{1}{a_K}, \cdots \right\rangle$. The K^{th} partial sum, Sum_{a_K} of the sequence is given by:

$$\mathsf{Sum}_{a_K} = \frac{1}{a_1} + \frac{1}{a_2} + \cdots + \frac{1}{a_K}$$

The K^{th} harmonic number is the K^{th} partial sum, Sum_K, of the sequence $a =< 1, 1/2, 1/3, \cdots, 1/K, \cdots >$ where $a_\mathsf{K} = \mathsf{K}$. That is,

$$\mathsf{H}_\mathsf{K} = \mathsf{Sum}_\mathsf{K}$$

4.2 Response Time Bounds and Approximation

We first present the mean response time pessimistic and optimistic bounds and then use these bounds to compute the response approximation. The parameter $\rho = \lambda/\mu$ represent the utilization of a server within the fork-join queue. Let R_N represent the mean response time of a N-server fork-join queue. The next theorem presents optimistic and pessimistic bounds of R_N.

Theorem 4.1 *The mean response time, R_N, of a N-server fork-join queue where each arriving job divides into N sub-tasks is bounded by*

$$\frac{1}{\mu}\left(\mathsf{H}_\mathsf{N} + \rho * \mathsf{Sum}_{\mathsf{N}(\mathsf{N}-\rho)}\right) \leq \mathsf{R}_\mathsf{N} \leq \frac{\mathsf{H}_\mathsf{N}}{\mu}\left(1 + \frac{\rho}{1-\rho}\right)$$

where $\rho = \lambda/\mu$ is the utilization of a server in the fork-join queue,

$\mathsf{H}_\mathsf{N} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{\mathsf{N}}$ *is the N^{th} harmonic number, and*

$\mathsf{Sum}_{\mathsf{N}(\mathsf{N}-\rho)} = \frac{1}{1-\rho} + \frac{1}{2}\frac{1}{2-\rho} + \frac{1}{3}\frac{1}{3-\rho} + \cdots + \frac{1}{\mathsf{N}}\frac{1}{\mathsf{N}-\rho}$ *is the N^{th} partial sum of the sequence* $\left\langle \frac{1}{1-\rho}, \frac{1}{2}\frac{1}{2-\rho}, \cdots, \frac{1}{\mathsf{N}}\frac{1}{\mathsf{N}-\rho}, \cdots \right\rangle$

Proof: The pessimistic bound is proved in [18] using associated random variables. Here, we present an informal argument for the pessimistic bound.

Let $\mathsf{R_1}$ represent the mean service time of a single $\mathsf{M/M/1}$ queue. Then, $\mathsf{H_N} * \mathsf{R_1}$ represents the response time of the $\mathsf{M/M/1}$ fork-join queue if the service time of each job at the parallel queue equals $\mathsf{H_N}/\mu$. The service time of a parallel job equals $\mathsf{H_N}/\mu$ only if all the N sub-tasks of the job execute concurrently. In a parallel fork-join queue, however, some sub-tasks of a job may have completed execution and be at the join-point, while other sub-tasks of the job are waiting for service or receiving service. Thus, due to the presence of N independent queues, the sub-tasks of a job in a fork-join queue may not all execute at the same time. This gives

$$\begin{aligned} \mathsf{R_N} &\leq \mathsf{H_N} * \mathsf{R_1} \\ &= \frac{\mathsf{H_N}}{\mu}\left(1 + \frac{\rho}{1-\rho}\right) \end{aligned}$$

The optimistic bound is proved using the Markov analysis presented in the previous section. By construction, the response time of a job in $\mathsf{P_N}$ is equal to the response time of the job in the $\mathsf{Serial\_P_N\_model}$. The response time of a job in the $\mathsf{Serial\_P_N\_model}$ is equal to the sum of the response times at the $\mathsf{Serial_N}$, $\mathsf{Serial_{N-1}}$, $\cdots$, $\mathsf{Serial_1}$ queues. The mean service rate at $\mathsf{Serial_k}$ can equal μ, 2μ, 3μ, $\cdots$, $(k-1)\mu$, or $k\mu$ depending on the number of jobs at queues $\mathsf{Serial_{k-1}}, \cdots, \mathsf{Serial_2}$, and $\mathsf{Serial_1}$. (Refer to Appendix A for details on the service rate at a server in the $\mathsf{Serial\_P_N\_model}$.) Thus, the overall mean service rate at $\mathsf{Serial_k}$ $(k > 1)$ is less than $k\mu$. Let $\mathsf{R_1}(k\mu)$ represent the mean response time of a $\mathsf{M/M/1}$ queue with arrival rate λ and service rate $k\mu$. Since the mean service rate at the $\mathsf{Serial_k}$ queue lies between $[\mu, k\mu]$, the mean response time at $\mathsf{Serial_k}$ $(k > 1)$ is greater than the response time $\mathsf{R_1}(k\mu)$. That is,

$$R\_\mathsf{Serial_k} \geq \mathsf{R_1}(k\mu)$$

This gives,

$$\begin{aligned} \mathsf{R_N} &\geq \mathsf{R_1}(\mu) + \mathsf{R_1}(2\mu) + \mathsf{R_1}(3\mu) + \cdots + \mathsf{R_1}(N\mu) \\ &= \frac{1}{\mu}\left(\mathsf{H_N} + \frac{\rho}{1-\rho} + \frac{1}{2}\frac{\rho}{2-\rho} + \frac{1}{3}\frac{\rho}{3-\rho} + \cdots + \frac{1}{N}\frac{\rho}{N-\rho}\right) \\ &= \frac{1}{\mu}\left(\mathsf{H_N} + \rho * \mathsf{Sum_{N(N-\rho)}}\right) \end{aligned}$$

□

Let $\mathsf{R_{opt}}$ and $\mathsf{R_{pes}}$, respectively, represent the optimistic and pessimistic response time bounds computed in Theorem 4.1. That is,

$$\mathsf{R_{opt}} = \frac{1}{\mu}\left(\mathsf{H_N} + \rho * \mathsf{Sum_{N(N-\rho)}}\right)$$

$$\mathsf{R_{pes}} = \frac{\mathsf{H_N}}{\mu}\left(1 + \frac{\rho}{1-\rho}\right) = \frac{1}{\mu}\left(\mathsf{H_N} + \rho * \mathsf{Sum_{N(1-\rho)}}\right)$$

The difference between $\mathsf{R_{pes}}$ and $\mathsf{R_{opt}}$ is

$$\begin{aligned} \mathsf{R_{diff}} &= \mathsf{R_{pes}} - \mathsf{R_{opt}} \\ &= \frac{\rho}{\mu}(\mathsf{Sum_{N(1-\rho)}} - \mathsf{Sum_{N(N-\rho)}}) \\ &= \frac{1}{\mu}\frac{\rho}{1-\rho}\left(\frac{1}{2(2-\rho)} + \frac{2}{3(3-\rho)} + \frac{3}{4(4-\rho)} + \cdots + \frac{\mathsf{N}-1}{\mathsf{N}(\mathsf{N}-\rho)}\right) \end{aligned}$$

The maximum error in the spread of the response time bounds is given by

$$\text{Maximum error} = \frac{\mathsf{R_{diff}}}{\mathsf{R_{pes}} + \mathsf{R_{opt}}} * 100$$

For a given ρ, the value of $\mathsf{R_{diff}}$ is non-decreasing for increasing values of N. For a given N, the value of $\mathsf{R_{diff}}$ is increasing for increasing ρ. The bounds are tight for $\rho \leq 0.6$. If $\lambda = 1$ time unit, $\mathsf{R_{diff}} = 0.6$ time unit when $\rho = 0.1$ and $\mathsf{N} = 1000$; $\mathsf{R_{diff}} = 3.05$ time units when $\rho = 0.5$ and $\mathsf{N} = 1000$; $\mathsf{R_{diff}} = 5.5$ time units when $\rho = 0.6$ and $\mathsf{N} = 1000$. The relative difference between the bounds increases for $\rho > 0.6$. $\mathsf{R_{diff}} = 10.2$ time units when $\rho = 0.7$ and $\mathsf{N} = 1000$; $\mathsf{R_{diff}} = 20.2$ time units when $\rho = 0.8$ and $\mathsf{N} = 1000$; $\mathsf{R_{diff}} = 33.2$ time units when $\rho = 0.9$ and $\mathsf{N} = 100$; $\mathsf{R_{diff}} = 51.8$ time units when $\rho = 0.9$ and $\mathsf{N} = 1000$. To address this spread in the bounds for $\rho > 0.6$, we present a response time approximation that is relatively invariant to the values of ρ and N. The next corollary presents the response time approximation, computed from the response time optimistic and pessimistic bounds.

Corollary 4.1

$$\begin{aligned} \mathsf{R_N} &\approx \frac{\mathsf{R_{opt}} + \mathsf{R_{pes}}}{2} \\ &= \frac{1}{\mu}\left(\mathsf{H_N} + \frac{\rho}{2(1-\rho)}\left(\mathsf{Sum_{N-\rho}} + (1-2\rho) * \mathsf{Sum_{N(N-\rho)}}\right)\right) \end{aligned}$$

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

$\mathsf{H_N} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{\mathsf{N}}$,

$\mathsf{Sum_{N-\rho}} = \frac{1}{1-\rho} + \frac{1}{2-\rho} + \frac{1}{3-\rho} + \cdots + \frac{1}{\mathsf{N}-\rho}$,

$\mathsf{Sum_{N(N-\rho)}} = \frac{1}{1-\rho} + \frac{1}{2}\frac{1}{2-\rho} + \frac{1}{3}\frac{1}{3-\rho} + \cdots + \frac{1}{\mathsf{N}}\frac{1}{\mathsf{N}-\rho}$.

Let $\mathsf{R_{approx}}$ represent the mean response time approximation computed in Corollary 4.1. The tightness of the response time approximation is verified by comparing against simulation results for N varying from $2, 3, 4, \cdots, 99, 100$ and ρ varying from 0.1 to 0.9. The simulated response time, $\mathsf{R_{simulation}}$, is accurate within 1% at 95 percent confidence. Figure 5 plots the mean simulated response times, the mean response time bounds, and the mean approximate response times for N-server fork-join queues where every job divides into N sub-tasks. There are three graphs in the figure corresponding to ρ values of 0.1, 0.5, and 0.9. The error in the response time approximation is given by

$$\text{Relative approximation error} = \frac{\mathsf{R_{approx}} - \mathsf{R_{simulation}}}{\mathsf{R_{simulation}}} * 100$$

Tables 1 and 2 present the approximate and simulated mean response time values and the error in the approximation. The errors increase for increasing values of ρ and are maximum ($\approx 13\%$) for $\rho = 0.9$. An interesting point is that the errors are relatively invariant to the value of N. That is, for a fixed ρ, the relative error remains approximately constant as N increases. On analyzing the response times for $\rho = 0.1, 0.2, 0.3, \cdots, 0.9$, we observe that the approximate response time is a pessimistic bound for $\rho < 0.5$ and becomes an optimistic bound for $\rho \geq 0.5$. This suggests that at $\rho \approx= 0.5$ the approximate response time matches the exact response time, though mathematical analysis is required to verify this observation.

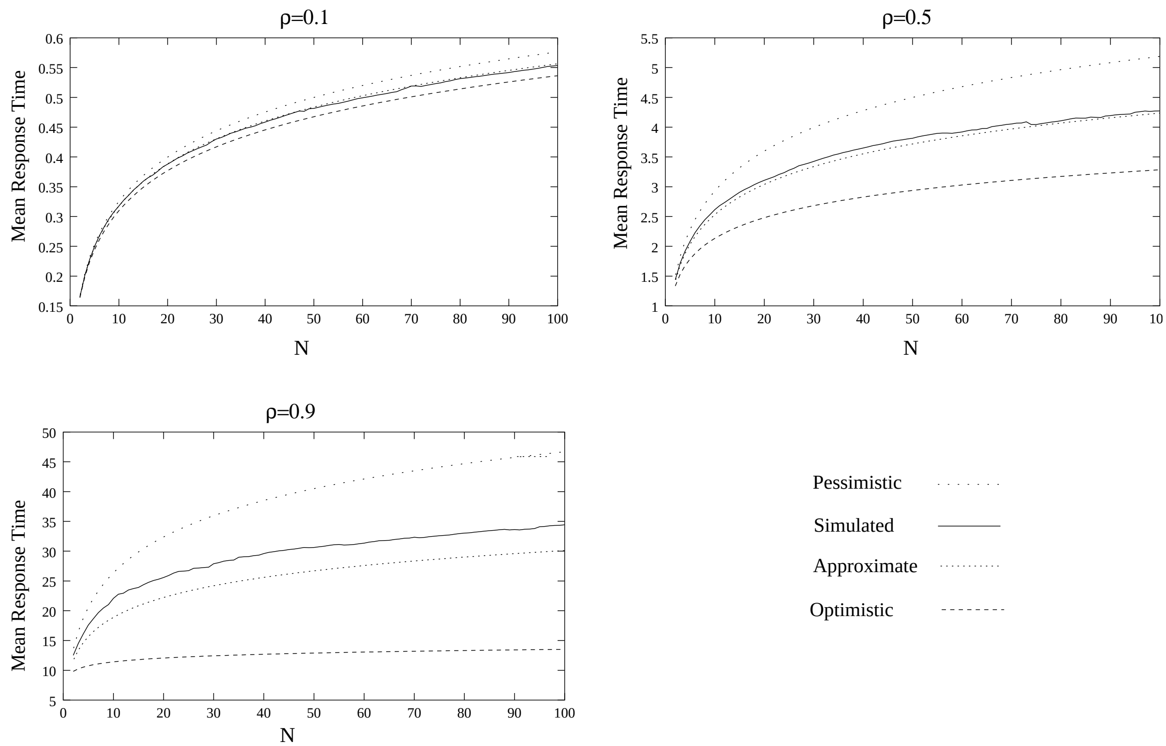

Figure 5: Model and simulated mean response time values when $\mathsf{K} = \mathsf{N}$

5 $1 \leq \mathsf{K} \leq \mathsf{N}$ Sub-Tasks

This section presents the response time optimistic and pessimistic bounds and the response time approximation for N-server fork-join queues where arriving jobs divide into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks. Let $\mathsf{P}_\mathsf{N}^\mathsf{K}$ represent such a fork-join queue. The workload is single class so all jobs divide into K sub-tasks that each have a mean service time of $1/\mu$. The arrival rate to $\mathsf{P}_\mathsf{N}^\mathsf{K}$ is λ. The arrival rate to servers of $\mathsf{P}_\mathsf{N}^\mathsf{K}$, however, is less than λ. The server arrival rate can be computed from the value of K and the sub-task allocation policy. For example, if the K sub-tasks of a job can only be submitted to K adjoining service centers of the fork-join queue, then the arrival rate to service centers is given by $\mathsf{K}/\mathsf{N} * \lambda$. Let $\mathsf{server\_access\_probability}$ represent the probability that an arriving job's sub-tasks are submitted to a server. In the above example, $\mathsf{server\_access\_probability} = \mathsf{K}/\mathsf{N}$. If all arriving jobs divide into N sub-tasks then $\mathsf{server\_access\_probability} = 1$. In general, if we assume a allocation policy that treats each server uniformly, then $\mathsf{server\_access\_probability} = \mathsf{K}/\mathsf{N}$. The next theorem shows that the response time of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ with arrival rate λ is equal to the response time of P_K with arrival rate $\lambda * \mathsf{server\_access\_probability}$.

Theorem 5.1 *The response time, $\mathsf{R}_\mathsf{N}^\mathsf{K}$, of a N-server fork-join queue $\mathsf{P}_\mathsf{N}^\mathsf{K}$ with arrival rate λ and service rate μ and where each arriving job divides into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks is equal to the response time, R_K, of a K-server fork-join queue P_K with arrival rate $\lambda * \mathsf{K}/\mathsf{N}$ and service rate μ and where each arriving job divides into K sub-tasks.*

Proof: Consider $\mathsf{P}_\mathsf{N}^\mathsf{K}$ (a N-server fork-join queue where each arriving job divides into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks) with arrival rate lambda and service rate mu. The probability that a server queue in $\mathsf{P}_\mathsf{N}^\mathsf{K}$ is assigned a sub-task is K/N. Therefore, the arrival rate to a server queue of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ is $\lambda * \mathsf{K}/\mathsf{N}$. That is, each server of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ is a $\mathsf{M}/\mathsf{M}/1$ queue with arrival rate $\lambda * \mathsf{K}/\mathsf{N}$ and service rate μ.

Now, consider P_K (a K-server fork-join queue where each arriving job divides into K sub-tasks) with arrival rate $\lambda * \mathsf{K}/\mathsf{N}$ and service rate μ. Thus, each server of P_K is a $\mathsf{M}/\mathsf{M}/1$ queue with arrival $\lambda * \mathsf{K}/\mathsf{N}$ and service rate μ.

Hence, a $\mathsf{M/M/1}$ queue of the $\mathsf{P}_\mathsf{N}^\mathsf{K}$ fork-join queue is identical to a $\mathsf{M/M/1}$ queue of the P_K fork-join queue.

In both fork-join queues $\mathsf{P}_\mathsf{N}^\mathsf{K}$ and P_K, the response time of a job is the time taken for all K sub-tasks of the job to finish. The response time of a job at the $\mathsf{P}_\mathsf{N}^\mathsf{K}$ fork-join queue is only dependent on the state of the K queues it is assigned to. To a job arriving at the $\mathsf{P}_\mathsf{N}^\mathsf{K}$ fork-join queue, the K $\mathsf{M/M/1}$ queues assigned to the job's sub-tasks are statistically identical to the K $\mathsf{M/M/1}$ queues of the P_K fork-join queue. Hence, the response time of a job at the $\mathsf{P}_\mathsf{N}^\mathsf{K}$ fork-join queue is identical to the response time of a job at the P_K fork-join queue.

□

An implication of Theorem 5.1 is that by setting $\rho = (\lambda * \mathsf{server\_access\_probability})/\mu$, and $\mathsf{N} = \mathsf{K}$, Theorem 4.1 and Corollary 4.1, respectively, can be used to compute the response time bounds and the response time approximation of the $\mathsf{P}_\mathsf{N}^\mathsf{K}$ fork-join queue.

Corollary 5.1 *The mean response time, $\mathsf{R}_\mathsf{N}^\mathsf{K}$, of a N-server fork-join queue, $\mathsf{P}_\mathsf{N}^\mathsf{K}$, where each arriving job divides into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks is bounded by*

$$\frac{1}{\mu}\left(\mathsf{H}_\mathsf{K} + \rho * \mathsf{Sum}_{\mathsf{K}(\mathsf{K}-\rho)}\right) \leq \mathsf{R}_\mathsf{N}^\mathsf{K} \leq \frac{\mathsf{H}_\mathsf{K}}{\mu}\left(1 + \frac{\rho}{1-\rho}\right)$$

The mean response time $\mathsf{R}_\mathsf{N}^\mathsf{K}$ of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ is approximated by

$$\mathsf{R}_\mathsf{N}^\mathsf{K} \;\approx\; \frac{1}{\mu}\left(\mathsf{H}_\mathsf{K} + \frac{\rho}{2(1-\rho)}\left(\mathsf{Sum}_{\mathsf{K}-\rho} + (1-2\rho) * \mathsf{Sum}_{\mathsf{K}(\mathsf{K}-\rho)}\right)\right)$$

where $\rho = \frac{\lambda * \mathsf{K}}{\mu * \mathsf{N}}$ is the utilization of a server in the fork-join queue,

$\mathsf{H}_\mathsf{K} = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{\mathsf{K}}$ *is the K^{th} harmonic number,*

$\mathsf{Sum}_{\mathsf{K}(\mathsf{K}-\rho)} = \frac{1}{1-\rho} + \frac{1}{2}\frac{1}{2-\rho} + \frac{1}{3}\frac{1}{3-\rho} + \cdots + \frac{1}{\mathsf{K}}\frac{1}{\mathsf{K}-\rho}$ *is the K^{th} partial sum of the sequence* $\left\langle \frac{1}{1-\rho}, \frac{1}{2}\frac{1}{2-\rho}, \cdots, \frac{1}{\mathsf{K}}\frac{1}{\mathsf{K}-\rho}, \cdots \right\rangle$,

and

$\mathsf{Sum}_{\mathsf{K}-\rho} = \frac{1}{1-\rho} + \frac{1}{2-\rho} + \frac{1}{3-\rho} + \cdots + \frac{1}{\mathsf{K}-\rho}$. *is the K^{th} partial sum of the sequence* $\left\langle \frac{1}{1-\rho}, \frac{1}{2-\rho}, \cdots, \frac{1}{\mathsf{K}-\rho}, \cdots \right\rangle$.

The mean response time bounds and approximation of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ is equal to the mean response time bounds and approximation of P_K. The Markov state diagrams of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ and P_K are, however, different. The difference between the Markov diagrams of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ and P_K account for the differences in the throughput and the queue length of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ and P_K. The throughput of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ equals λ while the throughput of P_K equals $\lambda * \mathsf{K}/\mathsf{N}$. Subsequently, the queue length (*i.e.*, number of jobs) at $\mathsf{P}_\mathsf{N}^\mathsf{K}$ and P_K are different. The queue length can be computed using Little's Law.

We validate our model response time against simulated response times. The simulated response times are accurate within 1% at 95% confidence. Figures 6, 7, and 8 plots the model and simulated response times for $(\rho * \mathsf{N})/\mathsf{K} = 0.1, 0.5, 0.9$. Each figure has three graphs that correspond to $\mathsf{N} = 10, 50, 100$. In each case, the number of sub-tasks vary from $\mathsf{K} = 1, 2, \cdots, \mathsf{N}$. Since λ/μ is held constant, the server utilization ρ varies as K varies.

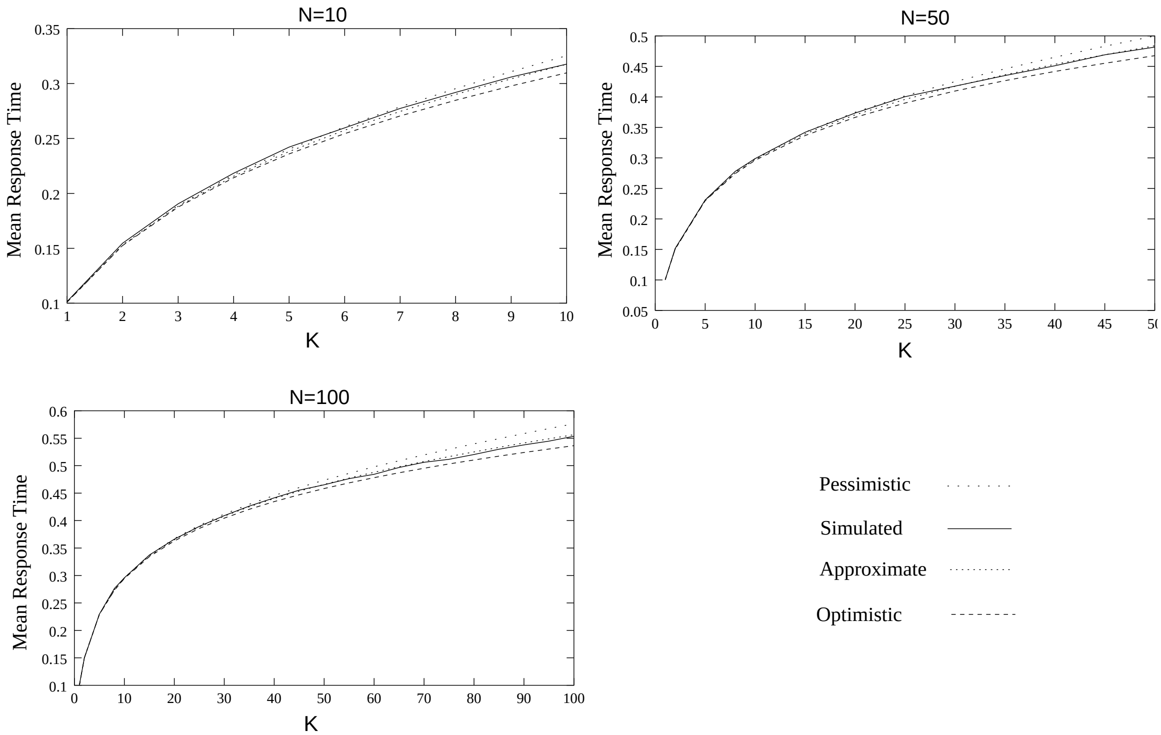

Figure 6: Model and simulated mean response time values when $\mathsf{K} \leq \mathsf{N}$ and $(\rho * \mathsf{N})/\mathsf{K} = 0.1$

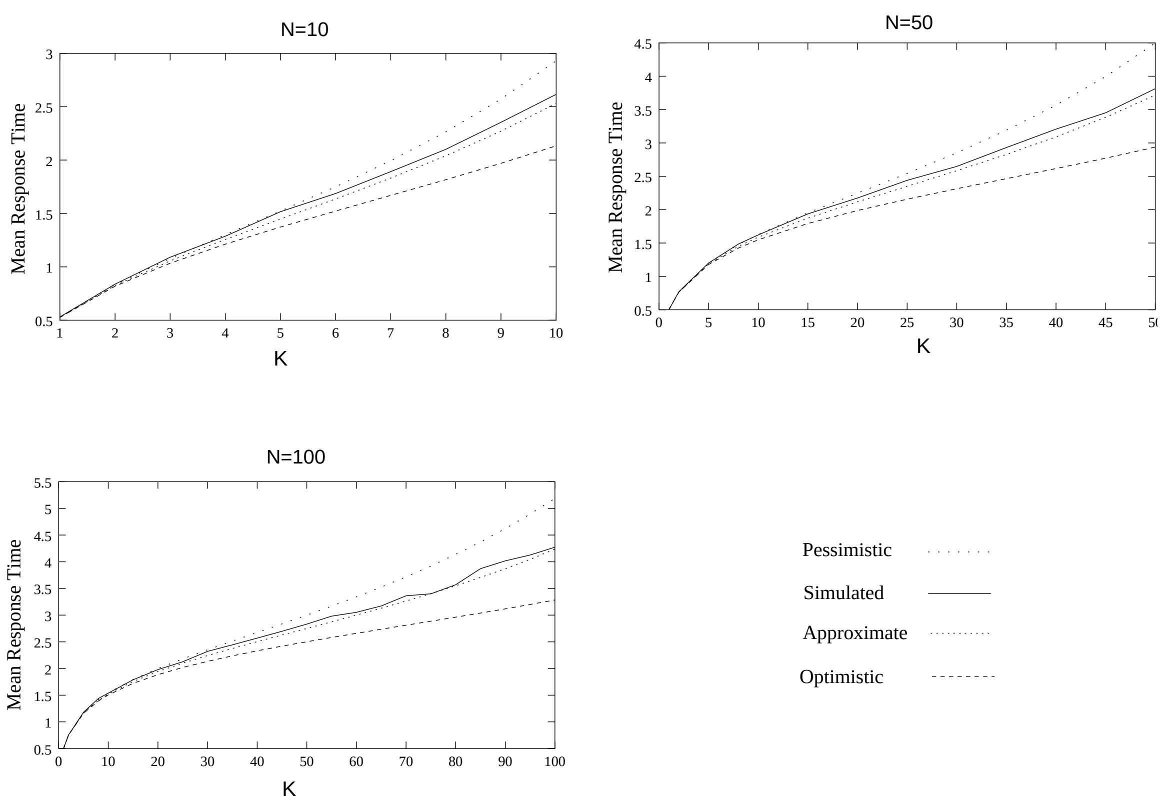

Figure 7: Model and simulated mean response time values when $\mathsf{K} \leq \mathsf{N}$ and $(\rho * \mathsf{N})/\mathsf{K} = 0.5$

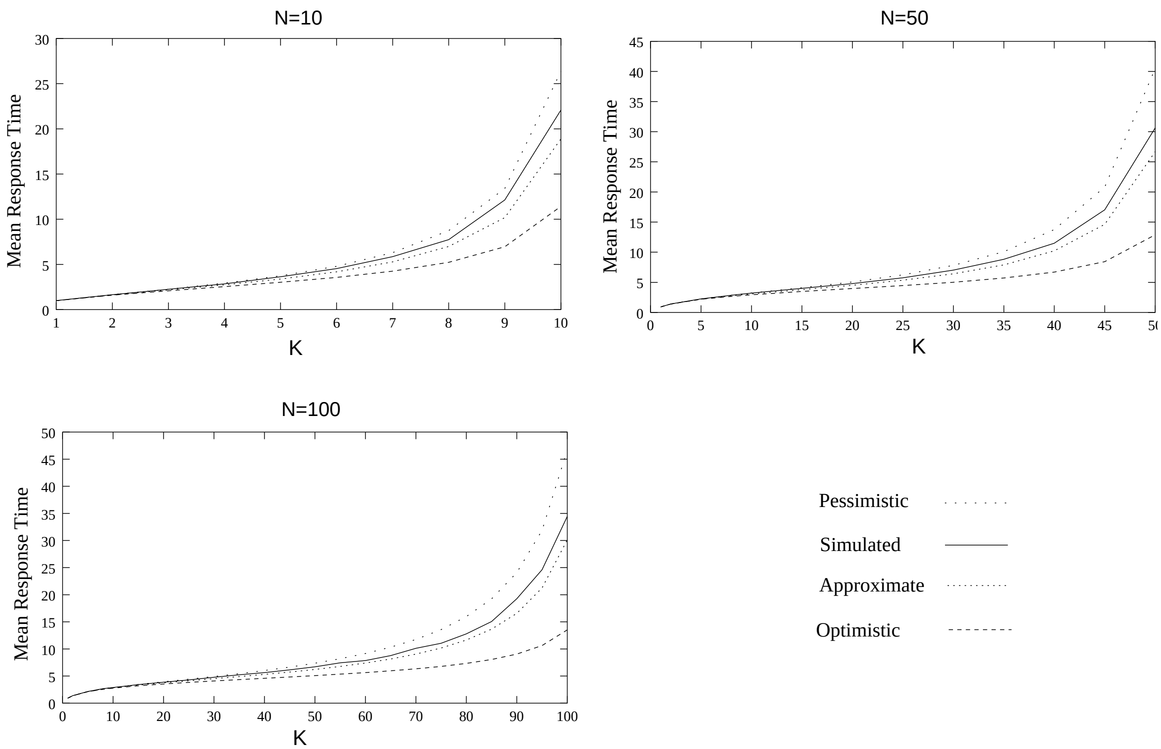

Figure 8: Model and simulated mean response time values when $\mathsf{K} \leq \mathsf{N}$ and $(\rho * \mathsf{N})/\mathsf{K} = 0.9$

6 Conclusions

This paper derives mean response time bounds and approximations for $\mathsf{M/M/1}$ N-server fork-join queues where arriving jobs divide into $1 \leq \mathsf{K} \leq \mathsf{N}$ sub-tasks. This work is notable for two key reasons, namely: a) the response time bounds and approximation are computationally simple and are presented as close-form equations, and b) the paper shows that the mean response time of $\mathsf{P}_\mathsf{N}^\mathsf{K}$ (a N-server fork-join queue where jobs divide into K sub-tasks) with arrival rate λ and service rate μ is equal to the mean response time of P_K (a K-server fork-join queue where jobs divide into K sub-tasks) with arrival rate $\lambda * \mathsf{K}/\mathsf{N}$ and service rate μ. The relative error in the approximation, as compared to simulated values, is less than 5% for $\mathsf{N} \leq 100$ and ρ varying from 0.1 to 0.9. Moreover, the relative error in the approximation remains approximately constant for fixed ρ as N varies from 2 to 100.

There are several extensions of this work. One extension is to consider multiple-class workloads where the value of K is not constant across all jobs and where the service requirement μ is different for each job class. Another interesting extension to this work is to consider fork-join queues with phase-type service time distributions and variable sub-tasks.

A Appendix

The mean service rate at $\mathsf{Serial_k}$ of the $\mathsf{Serial\_P_N\_model}$ is dependent on the number of jobs at $\mathsf{Serial_{k-1}}$, $\cdots$, $\mathsf{Serial_2}$, and $\mathsf{Serial_1}$. By analysing the Markov diagram of $\mathsf{P_N}$, we find that the rate varies according to the following rule: if there is at least one job in service center $\mathsf{Serial_{k-1}}$, the service rate at $\mathsf{Serial_k}$ equals μ, else if there is at least one job in service center $\mathsf{Serial_{k-2}}$, the service rate at $\mathsf{Serial_k}$ equals 2μ, else if there is at least one job in service center $\mathsf{Serial_{k-3}}$, the service rate at $\mathsf{Serial_k}$ equals 3μ, $\cdots$, else if there is at least one job in service center $\mathsf{Serial_1}$, the service rate at $\mathsf{Serial_k}$ equals $(k-1)\mu$, else (if there are no jobs in $\mathsf{Serial_{k-1}}, \mathsf{Serial_{k-2}}, \cdots, \mathsf{Serial_1}$), the service rate at $\mathsf{Serial_k}$

		rho = 0.1			rho = 0.5			rho = 0.9	
k	apprRT	simRT	err%	apprRT	simRT	err%	apprRT	simRT	err%
2	0.16520	0.16504	0.09695	1.41667	1.43928	1.57092	11.65909	12.52336	6.90126
3	0.20096	0.20090	0.02987	1.68333	1.71854	2.04883	13.37338	14.50522	7.80298
4	0.22767	0.22770	0.01318	1.87976	1.92015	2.10348	14.64354	16.10673	9.08434
5	0.24899	0.24971	0.28833	2.03532	2.08383	2.32793	15.65329	17.58570	10.98853
6	0.26672	0.26787	0.42931	2.16411	2.22879	2.90202	16.49153	18.63964	11.52442
7	0.28191	0.28275	0.29708	2.27400	2.34337	2.96027	17.20816	19.69006	12.60484
8	0.29518	0.29666	0.49889	2.36983	2.44826	3.20350	17.83404	20.46240	12.84483
9	0.30697	0.30781	0.27290	2.45480	2.53071	2.99955	18.38959	21.02895	12.55108
10	0.31758	0.31766	0.02518	2.53111	2.61548	3.22579	18.88904	22.08428	14.46839
11	0.32721	0.32762	0.12514	2.60038	2.68423	3.12380	19.34269	22.77161	15.05787
12	0.33605	0.33566	0.11619	2.66378	2.73650	2.65741	19.75823	22.94005	13.87015
13	0.34419	0.34465	0.13347	2.72224	2.79350	2.55092	20.14157	23.47786	14.21037
14	0.35176	0.35145	0.08821	2.77648	2.85122	2.62133	20.49735	23.71785	13.57838
15	0.35882	0.35921	0.10857	2.82705	2.90611	2.72048	20.82927	23.90498	12.86640
16	0.36544	0.36526	0.04928	2.87443	2.95269	2.65046	21.14032	24.36873	13.24817
17	0.37166	0.36982	0.49754	2.91899	2.99210	2.44343	21.43298	24.76445	13.45263
18	0.37754	0.37621	0.35353	2.96106	3.03529	2.44557	21.70929	25.07899	13.43635
19	0.38311	0.38318	0.01827	3.00089	3.07339	2.35896	21.97100	25.29888	13.15426
20	0.38840	0.38798	0.10825	3.03871	3.10882	2.25520	22.21956	25.58520	13.15464
21	0.39344	0.39286	0.14764	3.07471	3.13841	2.02969	22.45623	25.90421	13.31050
22	0.39825	0.39785	0.10054	3.10907	3.16828	1.86884	22.68210	26.31574	13.80786
23	0.40285	0.40155	0.32375	3.14192	3.20728	2.03786	22.89812	26.60038	13.91807
24	0.40725	0.40631	0.23135	3.17339	3.23581	1.92904	23.10510	26.64745	13.29339
25	0.41148	0.41004	0.35119	3.20359	3.27767	2.26014	23.30377	26.73399	12.83093
26	0.41555	0.41360	0.47147	3.23263	3.30973	2.32950	23.49477	27.12726	13.39055
27	0.41947	0.41689	0.61887	3.26058	3.35315	2.76069	23.67868	27.16729	12.84121
28	0.42324	0.42010	0.74744	3.28753	3.37493	2.58968	23.85600	27.25337	12.46587
29	0.42689	0.42556	0.31253	3.31354	3.39886	2.51025	24.02719	27.31246	12.02847
30	0.43041	0.43013	0.06510	3.33868	3.42525	2.52741	24.19265	27.88652	13.24608
31	0.43382	0.43265	0.27043	3.36301	3.45339	2.61714	24.35276	28.07468	13.25721
32	0.43713	0.43617	0.22010	3.38657	3.48081	2.70742	24.50786	28.31394	13.44242
33	0.44033	0.43985	0.10913	3.40941	3.50426	2.70671	24.65824	28.44451	13.31107
34	0.44344	0.44224	0.27135	3.43158	3.53273	2.86322	24.80419	28.51275	13.00667
35	0.44646	0.44526	0.26951	3.45311	3.55145	2.76901	24.94596	28.97331	13.90021
36	0.44940	0.44822	0.26326	3.47404	3.57518	2.82895	25.08378	29.06224	13.68945
37	0.45225	0.45004	0.49107	3.49441	3.59181	2.71172	25.21786	29.09154	13.31549
38	0.45503	0.45229	0.60581	3.51423	3.61442	2.77195	25.34841	29.25261	13.34650
39	0.45774	0.45580	0.42563	3.53355	3.63170	2.70259	25.47561	29.33367	13.15233
40	0.46039	0.45843	0.42755	3.55238	3.65081	2.69611	25.59962	29.59205	13.49156
41	0.46296	0.46157	0.30115	3.57074	3.66868	2.66963	25.72060	29.79437	13.67295
42	0.46548	0.46366	0.39253	3.58867	3.69162	2.78875	25.83869	29.90492	13.59719
43	0.46794	0.46633	0.34525	3.60618	3.70603	2.69426	25.95403	30.05418	13.64253
44	0.47034	0.46924	0.23442	3.62329	3.72220	2.65730	26.06674	30.13776	13.50804
45	0.47269	0.47189	0.16953	3.64002	3.74472	2.79594	26.17695	30.26644	13.51163
46	0.47498	0.47454	0.09272	3.65639	3.76540	2.89504	26.28475	30.35382	13.40546
47	0.47723	0.47707	0.03354	3.67240	3.77731	2.77737	26.39026	30.47955	13.41650
48	0.47943	0.47668	0.57691	3.68808	3.78821	2.64320	26.49356	30.62197	13.48186
49	0.48159	0.48085	0.15389	3.70344	3.80357	2.63253	26.59475	30.58877	13.05714
50	0.48370	0.48150	0.45691	3.71849	3.81371	2.49678	26.69392	30.62528	12.83698

Table 1: Model and simulated mean response time values for $\mathsf{N} = 2, \cdots, 50$

		rho = 0.1			rho = 0.5			rho = 0.9	
k	apprRT	simRT	err%	apprRT	simRT	err%	apprRT	simRT	err%
51	0.48577	0.48352	0.46534	3.73324	3.83598	2.67832	26.79113	30.72810	12.81228
52	0.48781	0.48540	0.49650	3.74771	3.85288	2.72965	26.88648	30.82968	12.79027
53	0.48980	0.48720	0.53366	3.76191	3.86712	2.72063	26.98002	30.96477	12.86866
54	0.49176	0.48874	0.61792	3.77584	3.88111	2.71237	27.07183	31.07436	12.88049
55	0.49368	0.48990	0.77159	3.78952	3.89668	2.75003	27.16197	31.12176	12.72354
56	0.49556	0.49170	0.78503	3.80295	3.89999	2.48821	27.25049	31.03390	12.19122
57	0.49742	0.49366	0.76166	3.81615	3.90009	2.15226	27.33746	31.05607	11.97386
58	0.49924	0.49592	0.66946	3.82912	3.89688	1.73883	27.42293	31.12450	11.89279
59	0.50103	0.49793	0.62258	3.84187	3.90858	1.70676	27.50694	31.24641	11.96768
60	0.50279	0.49929	0.70100	3.85440	3.91969	1.66569	27.58956	31.34717	11.98708
61	0.50452	0.50080	0.74281	3.86673	3.93788	1.80681	27.67081	31.51585	12.20034
62	0.50622	0.50215	0.81051	3.87886	3.95266	1.86710	27.75076	31.60768	12.20248
63	0.50790	0.50395	0.78381	3.89080	3.95517	1.62749	27.82944	31.75880	12.37251
64	0.50955	0.50512	0.87702	3.90255	3.97587	1.84412	27.90688	31.79126	12.21839
65	0.51118	0.50656	0.91203	3.91411	3.97864	1.62191	27.98313	31.82286	12.06595
66	0.51278	0.50833	0.87542	3.92551	4.00802	2.05862	28.05822	31.95715	12.20049
67	0.51435	0.50949	0.95390	3.93673	4.02072	2.08893	28.13220	32.03017	12.16968
68	0.51591	0.51305	0.55745	3.94779	4.03205	2.08976	28.20508	32.16927	12.32291
69	0.51744	0.51575	0.32768	3.95868	4.04604	2.15915	28.27691	32.20213	12.18932
70	0.51895	0.51903	0.01541	3.96942	4.05400	2.08633	28.34770	32.33649	12.33526
71	0.52044	0.51921	0.23690	3.98001	4.06758	2.15288	28.41750	32.25241	11.89031
72	0.52190	0.51831	0.69264	3.99045	4.06968	1.94684	28.48633	32.30628	11.82417
73	0.52335	0.52006	0.63262	4.00075	4.09051	2.19435	28.55422	32.42987	11.95087
74	0.52478	0.52160	0.60966	4.01091	4.04455	0.83174	28.62118	32.50519	11.94889
75	0.52618	0.52311	0.58687	4.02093	4.04414	0.57392	28.68726	32.59632	11.99234
76	0.52757	0.52434	0.61601	4.03082	4.05956	0.70796	28.75246	32.65664	11.95524
77	0.52895	0.52639	0.48633	4.04058	4.07151	0.75967	28.81681	32.71402	11.91297
78	0.53030	0.52788	0.45844	4.05022	4.08450	0.83927	28.88034	32.84456	12.06964
79	0.53164	0.52987	0.33404	4.05973	4.09528	0.86807	28.94307	32.94905	12.15810
80	0.53296	0.53142	0.28979	4.06913	4.10653	0.91074	29.00501	33.03771	12.20635
81	0.53426	0.53244	0.34182	4.07840	4.12366	1.09757	29.06618	33.07987	12.13333
82	0.53555	0.53342	0.39931	4.08757	4.14053	1.27906	29.12661	33.18181	12.22115
83	0.53682	0.53468	0.40024	4.09662	4.15241	1.34356	29.18630	33.26090	12.25042
84	0.53808	0.53537	0.50619	4.10557	4.15200	1.11826	29.24529	33.36544	12.34856
85	0.53932	0.53660	0.50690	4.11441	4.15070	0.87431	29.30358	33.43354	12.35275
86	0.54055	0.53788	0.49639	4.12315	4.16797	1.07534	29.36120	33.50060	12.35620
87	0.54176	0.53902	0.50833	4.13179	4.16559	0.81141	29.41815	33.60166	12.45031
88	0.54296	0.53987	0.57236	4.14032	4.16098	0.49652	29.47445	33.66284	12.44218
89	0.54415	0.54078	0.62317	4.14877	4.18753	0.92561	29.53012	33.56976	12.03357
90	0.54532	0.54177	0.65526	4.15712	4.19412	0.88219	29.58517	33.62919	12.02533
91	0.54648	0.54315	0.61309	4.16537	4.20692	0.98766	29.63961	33.56958	11.70694
92	0.54763	0.54416	0.63768	4.17354	4.21308	0.93851	29.69347	33.67562	11.82502
93	0.54877	0.54544	0.61052	4.18162	4.21775	0.85662	29.74674	33.71413	11.76774
94	0.54989	0.54632	0.65346	4.18961	4.22130	0.75072	29.79945	33.80737	11.85517
95	0.55100	0.54755	0.63008	4.19752	4.25127	1.26433	29.85160	34.10062	12.46024
96	0.55210	0.54885	0.59215	4.20535	4.26028	1.28935	29.90320	34.14585	12.42508
97	0.55319	0.55032	0.52151	4.21309	4.27167	1.37136	29.95428	34.25186	12.54700
98	0.55427	0.55198	0.41487	4.22076	4.26439	1.02312	30.00483	34.31641	12.56419
99	0.55534	0.55273	0.47220	4.22835	4.27091	0.99651	30.05487	34.33054	12.45442
100	0.55639	0.55353	0.51668	4.23586	4.27306	0.87057	30.10441	34.43731	12.58199

Table 2: Model and simulated mean response time values for $N = 51, \cdots, 100$

equals $k\mu$. The service rate at server Serial_1 is equal to μ. Thus, service rates at all queues $\mathsf{Serial}_\mathsf{k}$, $k = N, \cdots, 2$ are dependent on the states of the queues $\mathsf{Serial}_{\mathsf{k}-1}, \cdots, \mathsf{Serial}_2$, and Serial_1. Only queue Serial_1 is state independent.

References

[1] Almeida, V.A.F., Dowdy, L.W., "A reduction technique for solving queueing network models of programs with internal concurrency", Proceedings of the 3^{rd} International Conference on Supercomputing, Boston, May 1988.

[2] Baccelli, F. "Two parallel queues created by arrivals with two demands: The M/G/2 symmetrical case", Report INRIA, 426, July 1985.

[3] Baccelli, F., Massey, W. A., Towsley, D. "Acyclic fork-join queueing networks", Journal of the ACM, 36, 3, July 1989, pp. 615 – 642.

[4] Baccelli, F., Liu, Z. "On the execution of parallel programs on multiprocessor systems – A queueing theory approach", Journal of the ACM, 37, 2, April 1990, pp. 373 – 414.

[5] Balsamo, S., Donatiello, L., Van Dijk, N.M., "Bound performance models of heterogeneous parallel processing systems", IEEE Transactions on Parallel and Distributed Systems, Vol. 9, No. 10, October 1998.

[6] Chen, R.J., "A hybrid solution of fork/join synchronization in parallel queues", IEEE Transactions on Parallel and Distributed Systems, Vol. 12, No. 8, August 2001.

[7] Flatto, L., Hahn, S. "Two parallel queues created by arrivals with two demands", SIAM J. Appl. Math., 44, Oct. 1984, pp. 1041 – 1053.

[8] Heidelberger, P., Trivedi, S. K. "Queueing network models for parallel processing with asynchronous tasks", IEEE Trans. on Computers, 31, Nov. 1982, pp. 1099 – 1109.

[9] Heidelberger, P., Trivedi, S. K. "Analytic queueing models for programs with internal concurrency", IEEE Trans. on Computers, 32, Jan. 1983, pp. 73 – 82.

[10] Kim, C., Agrawala, A.K. "Analysis of the fork-join queue", IEEE Transactions on Computers, 38, 2, Feb. 1989, pp. 250 – 255.

[11] Kumar, A., Shorey, R. "Performance analysis and scheduling of stochastic jobs in a multicomputer system", IEEE Trans. on Parallel and Distributed Systems, 4, 10, Oct. 1993, pp. 1147 – 1164.

[12] Lee, E.K., Katz, R.H., "An analytic performance model of disk arrays", Proceedings of ACM SIGMETRICS, 1993.

[13] Liu, Y. C., Perros, H. G. "Approximate analysis of a closed fork/join model", European Journal of Operational Research, 53, 1991, pp. 382 – 392.

[14] Liu, Y. C., Perros, H. G. "A decomposition procedure for the analysis of a closed fork/join queueing system", IEEE Transactions on Computers, 40, 3, Mar. 1991, pp. 365 – 370.

[15] Lui, J.C.S., Muntz, R.R., Towsley, D, "Bounding the response time of a minimum expected delay routing system: an algorithmic approach", IEEE Trans. on Computers, vol 44, no. 5, May 1995, pp. 1371 – 1382.

[16] Lui, J.C.S., Muntz, R.R., Towsley, D, "Computing performance bounds of fork-join parallel programs under a multiprocessing environment", IEEE Trans. on Parallel and Distributed Systems, vol. 9, no. 3, March 1998, pp. 295-311.

[17] Makowski, A., Varma, S., "Interpolation approximations for symmetric fork-join queues", Performance Evaluation 20, 1994, pp. 145 – 165.

[18] Nelson, R., Tantawi, N., "Approximate analysis of fork/join synchronization in parallel queues", IEEE Transaction on Computers, 37, 6, June 1988, pp. 739 – 743.

[19] Thomasian, A., Tantawi, A.N., "Approximate solutions for M/G/1 fork-join synchronization", Proc. 1994 Winter Simulation conf., Orlando, Fla., Dec 1994, pp. 361 – 368.

[20] Thomasian, A., Menon, J., "Approximate analysis for fork-join synchronization in RAID5", Computer Systems: Science and Eng., 1997.

[21] Trivedi, K.S., *Probability and Statistics with Reliability, Queuing, and Computer Science Applications*, Prentice-Hall, 1982.

[22] Varki, E., "Mean value analysis of fork-join parallel networks", Proceedings of ACM/SIGMETRICS. Also, Performance Evaluation Review, May 1999.

[23] Varki, E., "Response time analysis of parallel computer and storage systems", IEEE Transactions on Parallel and Distributed Systems, Vol. 22(11), pp. 1146 – 1161, November 2001.