# Performance Analysis of Parallel Search Algorithms on Multiprocessor Systems

M. Ghodsi and K. Kant

Computer Science Department
Pennsylvania State University
University Park, PA 16802

**Abstract**

Many applications involve exploring alternate paths in a large state space, and thus could benefit from a parallel implementation of the search. In this paper, we present an exact solution technique for analyzing the performance of such applications. Formally, the problem can be stated as follows: A job arrives at a system with $M$ identical servers in a Poisson fashion. After some initial computation, the job spawns $K$ statistically identical subtasks. All these subtasks can be executed in parallel but only one of them is required to complete in order to complete the entire job. The service times are assumed to be exponentially distributed. The solution is obtained by defining a set of partial generating functions and exploiting their properties. The solution technique appears to be applicable for the general problem, but we have the formal proofs of correctness only for the cases where $K = M$ and when the mean service demands of the initial computation and the subtasks are the same. For these cases, the results show that the average number of busy processors is independent of $K$ while the job response time decreases with $K$. Some conjectures are presented for more general situations.

## 1 Introduction

In this paper we are concerned with the performance modeling of of applications involving parallel search in a large state space. There are many computationally intensive applications that involve exploration of multiple paths in a large or infinite state space and thus could benefit by performing the search in parallel. Some of the examples of such applications are game playing, theorem proving, symbolic manipulations, query execution in a distributed environment, heuristic optimization, VLSI layout, motion planning, etc. For example, in symbolic integration, there are several ways to proceed (e.g., various substitutions, integration by parts, decomposition into components, etc.), but it is not known a priori which of these approaches, if any, would work. Similarly, when doing motion planning in a partially known or unknown environment, one cannot decide a priori which plan will succeed. Resolution theorem proving has exactly the same characteristics. In all of these cases, if we implement the application on a multiprocessor system, a parallel investigation of the promising possibilities appears natural. In this paper, we are concerned with characterizing the benefits of such parallel execution.

The generic component of parallel search applications that we are most interested in has the following structure: there is some preparatory computation, possibly using the results of the previous parallel search, followed by a parallel search. This can be viewed as a 2-level tree structure shown in Figure 1.a. The processing of the application would involve repeated execution of this generic component according to some precedence constraints. In this paper, we ignore such precedence constraints and view each generic component as an independent transaction (or job) driving the multiprocessor system. For simplicity, we also fix the number of parallel searches initiated by each transaction, and denote that number as K. We also make the standard assumptions that allow stochastic modeling of the situation using Markov chains. Thus the problem we wish to consider can be stated as follows: we have a system with M identical single servers (or processors) driven by a Poisson stream with rate $\lambda$ of jobs with structure as in Figure 1.a. For each job, first the header task is processed by some server. After the completion of the header task, all of the subtasks of the job are created. Each of these subtasks is eligible to use an available server, so long as none of them has completed. As soon as any one of the subtasks completes, all of its siblings are removed from the system, and the job departs from the system. The service requirements both the header task and the subtasks are assumed to be exponentially distributed. The mean service rate is $\mu_1$ for the header task, and $\mu_2$ for each of the subtasks. While the assumption of exponentially distributed service time may appear unrealistic, we first wish to look at this simpler situation. It may be possible to extend the results to the nonexponential case either directly or using matrix geometric techniques, but that is beyond the scope of this paper.

We note here that even with the assumptions made above, the problem is by no means easy to solve. Part of the difficulty arises because of the abortion of all other subtasks when one of them finishes – this makes the system nonconservative and even finding the stability condition becomes a nontrivial task. The second difficulty arises due to the precedence constraints between the header task and subtasks. It is interesting to note that in the special case where the header task is null, the problem reduces to one similar to an ordinary M/M/c system, which can be easily solved. But the header task makes the problem much more difficult.

We assume that the jobs are picked according to a FCFS scheduling policy. This still leaves some choices to be made in task scheduling. In particular, if a processor becomes free, we could either schedule the header task of the next job, or a subtask of the oldest partially processed

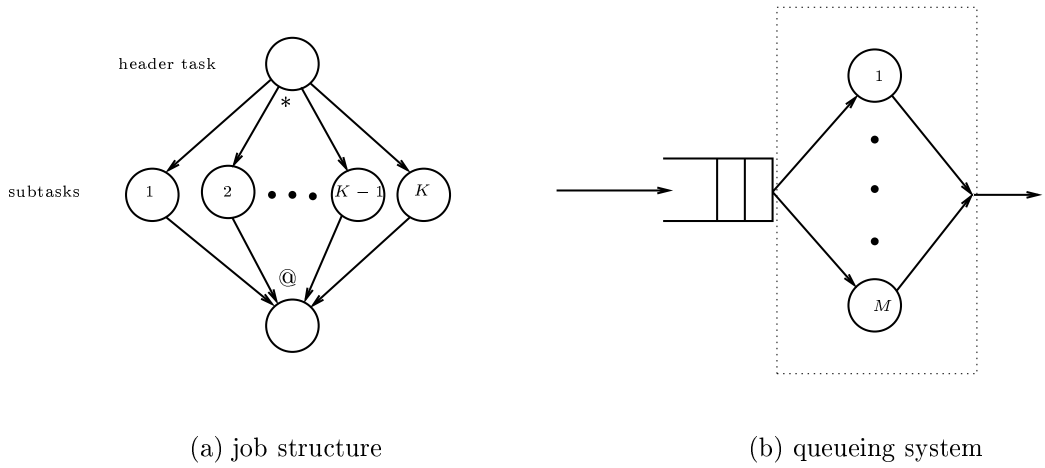


Figure 1: The system model.

job. The first scheme would tend to create lots of partially processed jobs in the system, and may even starve subtasks. Therefore, we choose the second alternative.

The goal of the modeling presented in this paper is to investigate whether parallel execution yields better response time, and to quantify the improvement. We will show that, when the service demands of the header task and the subtasks are the same, the average number of busy processors remain independent of $K$, but the job response time decreases as $K$ increases. This may be surprising when the system load is high.

We are not aware of any previous work on this problem. There is however, some work on the modeling of a superficially similar system, namely the fork-join queue. Flatto [2, 1] has solved this problem without the header task for 2 splits and 2 processors. Other work on the same problem is by Nelson [5].

## 2 Solution Technique

Assume that $\lambda$ is the job arrival rate and the average service rates for the header tasks and subtasks are $\mu_1$ and $\mu_2$ respectively. Also, let $M = cK + r$ for $0 \le r < K$. The solution is obtained by using the Markov chain representation of the system and a set of *partial generating functions* (PGF) for the number of header tasks in the system. These are $O(M^2)$ PGF's which can be obtained from a system of equations we shall provide. The resulting PGF's will include many unknown "initial" probabilities. We will prove that, in addition to the global balance equations, the computation of these initial probabilities requires exactly $M$ extra equations that are independent of the balance equations. One of these extra equations comes from the sum of all state probabilities being equal to one. It would then be expected that the generating functions has exactly $M-1$ poles inside the unit disk. Although, we do not have a proof for it in the general case, for the cases where $\mu_1 = \mu_2$, we can show that the denominator of the generating function can be reduced to a polynomial with degree of $M$ and coefficients that are only dependent on $K$ and $M$. We can further prove that this polynomial has always exactly $M-1$ zeros in the unit disk if $K = M$. For $K < M$, we still do not have a proof, but we believe that the algorithm is correct. In a large number of experimentations no contradiction to this claim has been found.

The stability condition of the problem when $\mu_1 = \mu_2 = \mu$ is shown to be $2\lambda/(M\mu) < 1$. This implies that the average number of busy processors is $2\lambda/\mu$ which is independent of the number of splits $K$. We shall also show that, for a fixed $M$, the system response time decreases as $K$ increases.

## 2.1 Markov Chain Model

A continuous Markov chain (MC) can be constructed for the problem stated above, with state defined by the quadruple $(n_1, n_2, \ell_1, \ell_2)$ where $n_1$ and $n_2$ are the number of header tasks and subtasks in the system, respectively. $\ell_1$ and $\ell_2$ are the number of header tasks and subtasks that are loaded and being served, respectively. Actually, one of the two components, $\ell_1$ and $\ell_2$, is extra in this state description, but we retain it for clarity. We call a state *full* if the servers are fully utilized in that state, i.e., $\ell_1 + \ell_2 = M$, and *partially full* otherwise.

In the following, we only state the needed characteristics of the MC without going into details. These details can be found in [3].

The state diagram for the case where $M = 5$ and $K = 3$ is shown in Figure 2. The numbers inside the states are $(\ell_1, \ell_2)$. The states have been grouped into different *stages*; the states in a stage $k$ are all states with $n_1 = k$. In a stage $n$, states can also be identified by their row and column numbers. In other words, $(n, i, j)$ represents the state at row $i$ and column $j$ of stage $n$. There are exactly $M + 1$ rows in all stages, numbered from 0 to $M$ where the $i$th row consists of states in which $n_2 = i * K$. As shown in the figure, the number of states in a row depends on the row number.

In stage $n \geq M$, all states are full. Therefore, in states $(n, i, j)$ where $i \leq c$, all $iK$ subtasks can be loaded simultaneously and receive service. For $i > c$, on the other hand, at most $M$ out of $iK$ subtasks can be loaded. The number of states in row $i$ and stage $n \geq M$, is $i(K - 1) + 1$ if $i \leq c$ and $M - i + 1$ otherwise. The total number of states in these stages can thus be computed as $W \equiv ((M + 1)^2 - M * c - c * r - r + 1)/2$.

Three transitions are possible out of each state: The arrival of a new job takes the MC to the next stage and completion of a header task takes it to the previous stage. Transitions among the states in a single stage are due to the completion of subtasks.

The stages $n < M$ are referred to as the *initial stages*. Also, we call the states in these stages and their probabilities as the *initial states* and the *initial probabilities*, respectively. In the initial stages, the configuration of states is no longer regular, as illustrated in Figure 2. By identifying the "non-existing" states (those with zero probabilities), however, we can assume that the states in all stages have the same structure. This simplifies the derivation of the balance equations.

Figure 3 shows the structure of the states in a stage. States $(n, i, i(K - 1))$ for $i = 0 \,..\, c$ that are painted in gray are called *special states*. These states "exist" in all stages. We also define the states in one diagonal as below.

**Definition 1** The states in a *diagonal* $d = 1 \,..\, M$ are $(n, i, d - i)$, where $d \div^1 (K - 1) + 1 \leq i \leq d$. Note that the special states are not defined to be in any diagonal.

It can be seen that, for all states in a diagonal $d$ we have $\ell_2 = d$ and $\ell_1 = M - d$.

[1] $a \div b$ is the quotient of two integers $a$ and $b$.

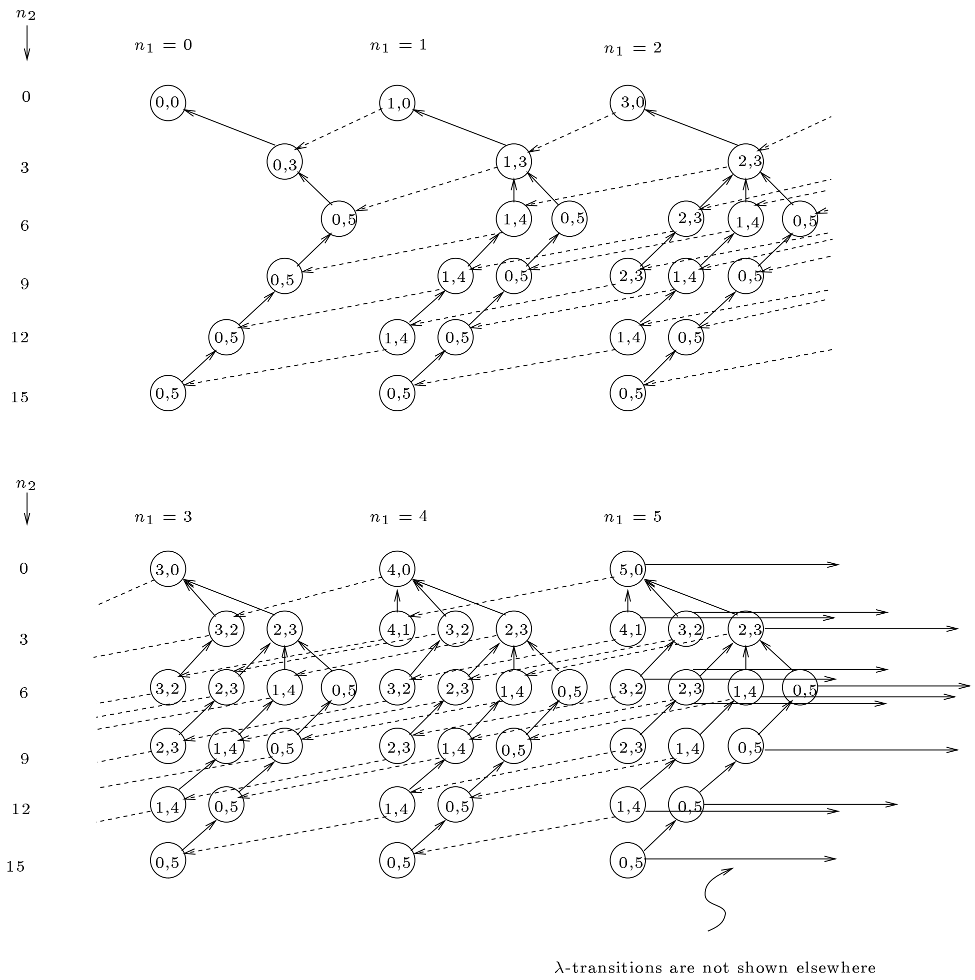


Figure 2: The state diagram for a system with $M = 5$ and $K = 3$.

## 2.2 Balance Equations

In this section, we present the balance equations without going into details of their derivations. The balance equations can be classified into five forms for all possible state configurations which are shown in Figure 4.

## 2.3 Partial Generating Functions

The solution technique is based on the computation of the partial generation functions defined for the number of header tasks in the system. For states $(n, i, j)$, we define the $(i, j)$th generating function as

$$G_{i,j}(z) = \sum_{n=0}^{\infty} z^n P(n, i, j). \tag{6}$$

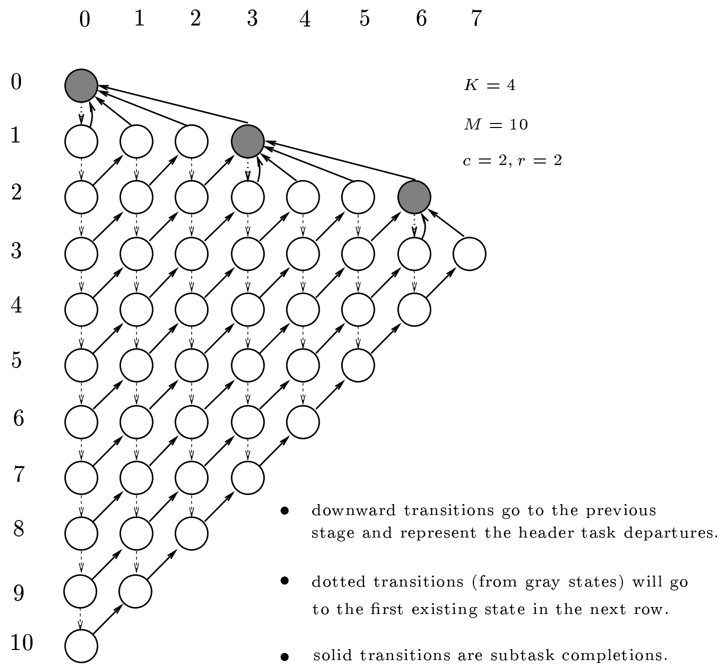


Figure 3: The general structure of a stage.

The number of these PGF's is $W$, the maximum number of states in a stage which is $O(M^2)$. The overall generating function for the number of header tasks in the system is then the sum of $G_{i,j}(z)$ over all possible values of $i$ and $j$.

The computation of the PGF's is resulted from the balance equations presented in the last section. The uniformity of these equations for full states, enables us to write one equation for $G_{i,j}$ from the balance equations written for states $(n, i, j)$. This system of equations can then be symbolically solved to get the $G$ functions. At this point, this solution may seem very expensive. However, it is shown that the complexity can be greatly reduced by exploiting the properties of these equations.

For full states, only the balance equations (2), (4), and (5) of Figure 4 apply. First, consider equation (2) written for the special states. In this equation, all state probabilities are positive for $i < c$. For $i = c$, however, states $(n, c+1, c(K-1)+k)$ exists only for $k = 0 \mathinner{..} r-1$,since the number of states at row $c+1$ is $M-c-1$. If we multiply both sides of this equation by $z^{n+1}$ and sum over all $n \geq M - ik$, we have

$$
\begin{aligned}
z((M-iK)\mu_1 + iK\mu_2 - \lambda z)G_{i,i(K-1)} &= \\
\sum_{k=0}^{\alpha}(iK+k)\mu_2 G_{i+1,i(K-1)+k-1} &+ A_{i,i(K-1)}(z),
\end{aligned}
\tag{7}
$$

where $\alpha = K-1$ for $i < c$ and $\alpha = r-1$ for $i = c$. $A_{i,i(K-1)}(z)$ is a polynomial in $z$ with coefficients in terms of $\lambda$, $\mu_1$, $\mu_2$, and the initial probabilities.

Similar equation can be obtained for the balance equations of (4), and (5) by multiplying

1. For special state $S = (n, i, j)$ (i.e., $i \leq c$ and $j = i(K-1)$)

   (a) if $n + iK \leq M$,

$$(n\mu_1 + iK\mu_2 + \lambda)P(n,i,j) = (n+1)\mu_1 P(n+1, i-1, j-K+1) + \lambda P(n-1,i,j) + \sum_{k=0}^{K}(iK+k)\mu_2 P(n, i+1, j+k-1) \tag{1}$$

   (b) if $n + iK > M$,

$$((M - iK)\mu_1 + iK\mu_2 + \lambda)P(n,i,j) = \lambda P(n-1,i,j) + \sum_{k=0}^{K}(iK+k)\mu_2 P(n, i+1, i+k-1) \tag{2}$$

2. For states $(n, i, j)$ in rows $i \leq c$ and with $(i-1)(K-1) < j < i(K-1)$,

   (a) if $j = M - n - i$

$$((M-i-j)\mu_1 + (i+j)\mu_2 + \lambda)P(n,i,j) = (i+j)\mu_2 P(n, i+1, j-1) + \lambda P(n-1,i,j) + (n+1)\mu_1 P(n+1, i-1, (i-1)(K-1)) \tag{3}$$

   (b) else $(j > M - n - i)$

$$((M-i-j)\mu_1 + (i+j)\mu_2 + \lambda)P(n,i,j) = \lambda P(n-1,i,j) + (i+j)\mu_2 P(n, i+1, j-1) \tag{4}$$

3. For other states,

$$((M-i-j)\mu_1 + (i+j)\mu_2)P(n,i,j) = \lambda P(n-1,i,j) + (M-i-j)\mu_1 P(n+1, i-1, j) + (i+j)P(n, i+1, j-1) \tag{5}$$

Figure 4: The balance equations.

1. Write the balance equations for all states in the stages 0.
2. For stages $n = 1 .. M-1$, write the balance equations for:
   (a) all states in diagonal $M-n$, and for
   (b) all special states of $(n, i, i(K-1))$, for $i = 0 .. c - (n+K-r-1) \div K - 1$.

Figure 5: The complete system of initial balance equations.

both sides of these equations by $z^n$ and $z^{n+1}$, respectively and summing over all possible $n$ that these equation apply. For $i < c$ and $(i-1)K < j < i(K-1)$ and also for $i = c$ and $(c-1)K < j \leq M-c+1$,

$$((M-i-j)\mu_1 + (i+j)\mu_2 + \lambda - \lambda z)G_{i,j} = (i+j)G_{i+1,j-1} + A_{i,j}(z). \tag{8}$$

And for the rest of the values of $i, j$,

$$\begin{aligned} z((M-i-j)\mu_1 + (i+j)\mu_2 + \lambda - \lambda z)G_{i,j} = \\ (M-i-j)\mu_1 G_{i-1,j} + (i+j)\mu_2 z G_{i+1,j-1} + A_{i,j}(z). \end{aligned} \tag{9}$$

The detailed expressions for $A_{i,j}(z)$'s in above equations are given in [3]. These expressions appear in the PGF's and involve a number of unknown initial probabilities which must be computed. The balance equations are not sufficient, however, for computations of these probabilities.

**Lemma 1** *The computation of the initial probabilities in this system, requires exactly $M$ extra equations that are linearly independent of the balance equations.*

**Proof:** We omit the proof which involves a detailed examination of the state characteristics in the initial stages. The idea is to come up with a system of initial balance equations which include $k$ probabilities for states whose balance equations have not been written. Moreover, this system of balance equation must include all initial probabilities that appear in the generating functions. It should be proven that the minimum value of $k$ is $M$. Here, we only present the final system of balance equations shown in Figure 5. □

For the state diagram of Figure 2, the balance equations are written for all states on stage 0; for state $(1,0,0)$ and three states in diagonal 4 of stage 1; for state $(2,0,0)$ and two states in diagonal 3 of stage 2; for states $(3,1,1)$, and $(3,2,0)$ in stage 3; and for state $(4,1,0)$ in stage 4.

The number of equations in the system of initial balance equations is an important factor in the complexity of this algorithm. The system of equations of Figure 5 has $O(M^2)$ equations which the smallest possible.

As stated before, one of the $M$ additional equations comes from the sum of all state probabilities being equal to one. We will show that the other $M-1$ extra equations can be written using the poles of the generating functions. The idea comes from the fact that the partial generating function defined in equation (6) will have a bounded value for complex

1. For $i = 0$ and $j = 0$,

$$yMG_{0,0} = \sum_{k=1}^{K} kG_{1,k-1} + A_{0,0} \tag{10}$$

2. For $i = 1 \mathinner{..} c - 1$ and $j = i(K-1)$,

$$yMG_{i,j} = \sum_{k=0}^{K} (iK + k)G_{i+1,j+k-1} + A_{i,j} \tag{11}$$

3. For $i = c$ and $j = c(K-1)$,

$$yMG_{i,j} = \sum_{k=0}^{r} (iK + k)G_{i+1,j+k-1} + A_{i,j} \tag{12}$$

4. For $i > 0$ and $j = 0$,

$$zyMG_{i,0} = (M - i + 1)G_{i-1,0} + A_{i,0} \tag{13}$$

5. For $i = 1 \mathinner{..} c$ and $(i-1)(K-1) < j < i(K-1)$, and also for $i = c$ and $(c-1)(K-1) < j < M - c$,

$$yMG_{i,j} = (i + j)G_{i+1,j-1} + A_{i,j} \tag{14}$$

6. For other values of $i$ and $j$,

$$zyMG_{i,j} = (M - i - j + 1)G_{i-1,j} + z(i + j)G_{i+1,j-1} + A_{i,j} \tag{15}$$

Note: $b = \lambda/(M\mu)$ and $y = 1 + b - bz$.

Figure 6: The system of equations for the computation of the PGF's when $\mu_1 = \mu_2 = \mu$.

values of $z$ on or inside the unit disk, i.e., $|z| \leq 1$. If the denominator of the PGF's has a zero in the unit disk, that pole should also be a zero of the numerators of the PGF's. In the next section, we will show that this is in fact true; that there exist exactly $M-1$ poles for the PGF's inside the unit disk. For the cases where $\mu_1 = \mu_2$, we will prove that the denominator of the PGF's can be reduced to a polynomial with degree of $M-1$ and with coefficients that are independent of the arrival and the service rates. We present a formal proof that the zeros of such polynominals are all in the unit disk for the cases where $K = M$. We will also prove that for each zero of this reduced polynomial, there is exactly one pole for the PGF's inside the unit disk.

## 2.4 Poles of the Generating Functions

In this section we discuss the results regarding the number of poles of the partial generating functions when $\mu \equiv \mu_1 = \mu_2$.

The system of equations for the PGF's given above (equations (8), (8), and (9)) greatly simplify when $\mu_1 = \mu_2 = \mu$. The resulting equations are summarized in Figure 6, where we

1. Find the expression for $G_{i,0}$ for $i = 1 \,..\, M$ and in the same order, in terms of $G_{0,0}$ using equation (13).
2. For $j = 1 \,..\, M - c - 1$, let $v = j \div (K-1)$, i.e., $v(K-1) \leq j < (v+1)(K-1)$. Then,
   (a) If $j > v(K-1)$, find the expressions for $G_{v,j}$ and for $G_{i,j}$, $i = v+1 \,..\, M-j$ using equations (14) and (15), respectively. These expressions are in terms of $G_{i,i(K-1)}$ for $i = 0 \,..\, v$.
   (b) If $j = v(K-1)$, using equation (14), compute the expression for $i = v+1 \,..\, M-j$ in terms of $G_{i,i(K-1)}$ for $i = 0 \,..\, v$.
3. Solve the system of equations for $G_{i,i(K-1)}$, $i = 0 \,..\, c$ (equations (10), (11), and (12)) substituting the other $G$'s with the expressions found in above steps. These equations form a lower Hessenberg system of equations.

Figure 7: The order in which the system of equations is solved for the PGF's.

have assumed that $b = \lambda/(M\mu)$ and $y = 1 + b - bz$. In these equations, the argument $z$ of the $G_{i,j}(z)$ and $A_{i,j}(z)$ have been dropped for clarity.

The computation of the PGF's can be carried out symbolically in $O(M^5)$ with the help of a series of back-substitutions shown in Figure 7. The delails of the complexity analysis can be found in [3]. Using the correspondence between $G_{i,j}$ and the state $(n, i, j)$, computation of the PGF's in terms of $G_{i,i(K-1)}$'s is carried out in the same order as visiting the corresponding states in column-row order.

**Theorem 1** *The denominator of the partial generating functions satisfying the system of equations of Figure 6 is a $M$ degree polynomial in $x = zy^2$ whose coefficients are all independent of $\lambda$ and $\mu$. One of the zeros of this polynomial is $x = 1$.*

The proof of this theorem is rather involved. We first follow the solution order shown of Figure 7. In doing so, we will show how the expressions of the PGF's evolve into the final expression for the denominator.

**Lemma 2** *Let $x = zy^2$ and $v = j \div (K-1)$ for $j = 0 \,..\, M - c - 1$. Following the solution order of Figure 7, the partial generating function $G_{i,j}$, $j \neq i(K-1)$, can be expressed in terms of $G_{w,w(K-1)}$ for $w = 0 \,..\, v$ as,*

$$G_{i,j} = \sum_{w=0}^{v} \frac{C_{w,i,j}\; G_{w,w(K-1)}}{(yz)^{i-w}\; x^{j-w(K-1)}} + \frac{B_{i,j}}{(yz)^i x^j} \tag{16}$$

*where the coefficients $C_{w,i,j}$ are positive constants (dependent on $M$ and $K$, but not on $b$) and $B_{i,j}$ is a polynomial in $z$, $b$, and the initial probabilities.*

**Proof:** We use induction on $j$ and $i$ for the proof. The expression for $G_{i,0}$, which is computed using equation (13), is the basis for induction.

$$G_{i,0} = \frac{\prod_{k=1}^{i}(M-k+1)}{(Myz)^i} G_{0,0} + \frac{B_{i,0}}{(yz)^i} \qquad i = 1 \,..\, M \tag{17}$$

For the induction steps, assume that the claim is true for $j < k$ and for all possible values of $i$. We will prove that it is also true for $G_{i,k}$, for $i = v+1 \,..\, M-j$, where $v = k \div (K-1)$, i.e., $v(K-1) \leq k < (v+1)(K-1)$ (corresponding to all states in column $k$ except the special state).

First we shall prove the claim for $i = v+1$. There are two cases to consider: (1) $k = v(K-1)$, and (2) $v(K-1) < k < (v+1)(K-1)$. For the first case, we use equation (15) and find the expression for $G_{v+1,k}$ as

$$G_{v+1,k} = \frac{(M-v-k)G_{v,k}}{Myz} + \frac{(v+1+k)G_{v+2,k-1}}{My} + \frac{A_{v+1,k}}{Myz}. \tag{18}$$

Using the induction hypothesis,

$$G_{v+2,k-1} = \sum_{w=0}^{v-1} \frac{C_{w,v+2,k-1}\, G_{w,w(K-1)}}{(yz)^{v+2-w}\; x^{k-1-w(K-1)}} + \frac{B_{v+2,k-1}}{(yz)^{v+2} x^{k-1}}. \tag{19}$$

Substituting this equation in equation (18) results in

$$\begin{aligned} G_{v+1,k} &= \frac{(M-v-k)G_{v,k}}{Myz} + \\ &\quad \sum_{w=0}^{v-1} \frac{(v+1+k)C_{w,v+2,k-1}\, G_{w,w(K-1)}}{M(yz)^{v+1-w}\; x^{k-w(K-1)}} + \frac{B_{v+1,k}}{(yz)^{v+1} x^{k}} \\ &= \sum_{w=0}^{v} \frac{C_{w,v+1,k}\, G_{w,w(K-1)}}{(yz)^{v+1-w}\; x^{k-w(K-1)}} + \frac{B_{v+1,k}}{(yz)^{v+1} x^{k}}. \end{aligned} \tag{20}$$

Similarly, if $v(K-1) < k < (v+1)(K-1)$, then $G_{v+1,k}$ can be expressed as follows using equation (14). Using the induction hypothesis, $G_{v+1,k}$ is expressed in the claimed form.

$$\begin{aligned} G_{v+1,k} &= \frac{(v+1+k)G_{v,k}}{My} + A_{v+1,k}/My \\ &= \sum_{w=0}^{v} \frac{(v+1+k)C_{w,v+1,k}\, G_{w,w(K-1)}}{M(yz)^{v+1-w}\; x^{k-w(K-1)}} + \frac{B_{v+1,k}}{(yz)^{v+1} x^{k}} \end{aligned} \tag{21}$$

Next, we use induction on $i$ to prove the claim for $G_{i,k}$, $i = v+2 \,..\, M-k$. From equation (15),

$$G_{i,k} = \frac{(M-i-k+1)G_{i-1,k}}{Myz} + \frac{(i+k)G_{i+1,k-1}}{My} + \frac{A_{i,k}}{My}. \tag{22}$$

Also, from the induction hypothesis, and the proof for $i = v+1$,

$$G_{i-1,k} = \sum_{w=0}^{v} \frac{C_{w,i-1,k}\, G_{w,w(K-1)}}{(yz)^{i-1-w}\; x^{k-w(K-1)}} + \frac{B_{i-1,k}}{(yz)^{i-1} x^{k}} \tag{23}$$

$$G_{i+1,k-1} = \sum_{w=0}^{\alpha(k)} \frac{C_{w,i+1,k-1}\, G_{w,w(K-1)}}{(yz)^{i+1-w}\; x^{k-1-w(K-1)}} + \frac{B_{i+1,k-1}}{(yz)^{i+1} x^{k-1}} \tag{24}$$

where $\alpha(k) = v-1$ if $k = v(K-1)$ and $\alpha(k) = v$ if $v(K-1) < k < (v+1)(K-1)$. Substituting in equation (22), we have

$$\begin{aligned} G_{i,k} &= \sum_{w=0}^{v} \frac{(M-i-k+1)C_{w,i-1,k}\ G_{w,w(K-1)}}{M(yz)^{i-w}\ x^{k-w(K-1)}} + \\ &\quad \sum_{w=0}^{\alpha(k)} \frac{(i+k)C_{w,i+1,k-1}\ G_{w,w(K-1)}}{M(yz)^{i-w}\ x^{k-w(K-1)}} + \frac{B_{i,k}}{(yz)^i x^k} \\ &= \sum_{w=0}^{v} \frac{C_{w,i,k}\ G_{w,w(K-1)}}{(yz)^{i-w}\ x^{k-w(K-1)}} + \frac{B_{i,k}}{(yz)^i x^k}. \end{aligned} \tag{25}$$

This completes the proof of the lemma. □

At this point, the expressions for the PGF's that are in the form of equation (16) can be substituted in the equations written for $G_{i,i(K-1)}$, $i = 0 \mathinner{..} c$ (equations (10), (11), and (12)). This results in a system of $c+1$ equations that can be solved for the final expressions of the PGF's. Starting from the following notations, we investigate the properties of this system of equations to complete the proof of the theorem.

*poly*$(x,n)$ A degree $n$ polynomial in $x$ and with constant coefficients that are dependent only on $M$ and $K$.

*poly*$^+(x,n)$ Same as above polynomial but with positive coefficients.

Using this definition and after proper simplifications, equations (10) and (11) can be written concisely as

$$\begin{aligned} &\sum_{j=0}^{i-1} \frac{poly^+(x,K-1)\ G_{j,j(K-1)}}{(yz)^{i-j}\ x^{(i-j+1)(K-1)}} - \left[1 - \frac{poly^+(x,K-2)}{x^{K-1}}\right] G_{i,i(K-1)} + \\ &\quad \frac{\alpha_{i,i+1}\ G_{i+1,(i+1)(K-1)}}{y} = \frac{D_{i,i}}{(yz)^i x^{(i+1)(K-1)}} \qquad i = 0 \mathinner{..} c-1 \end{aligned} \tag{26}$$

where $\alpha_{i,i+1} = (i+1)K/M$ and $D_{i,i}$ is another polynomial in $z$ which does not play any role in the expression for the denominator. For $i = c$,

$$\sum_{j=0}^{c-1} \frac{poly^+(x,r)\ G_{j,j(K-1)}}{(yz)^{c-j}\ x^{(c-j)(K-1)+r}} - \left[1 - \frac{poly^+(x,r-1)}{x^r}\right] G_{c,c(K-1)} = \frac{D_{c,c}}{(yz)^c x^{c(K-1)+r}} \tag{27}$$

where $poly^+(x, r-1) = 0$ for $r = 0$.

The system of equations (26) and (27) can also be written as $Qg = d$ where $Q$ is a lower Hessenberg $(c+1) \times (c+1)$ matrix and $g$ is the vector of unknown $G_{j,j(K-1)}$ functions. The denominator of PGF's is, hence, the numerator of the determinant of $Q$.

To prove that the denominator is a polynomial only in $x$, we multiply the $i$th row of $Q$ by $y^{-i}$ and its $j$th column by $y^j$. These multiplications create a new matrix $R$ that has the same determinant as $Q$ and only involves $x$. To see this, the following is the system of equations after these multiplications.

$$
\begin{array}{c|cccc}
 & 0 & 1 & 2 & 3 \\
\hline
0 & \frac{poly(x,K-1)}{x^{K-1}} & \alpha_{0,1} & & \\
1 & \frac{poly^{+}(x,K-1)}{x^{2K-1}} & \frac{poly(x,K-1)}{x^{K-1}} & \alpha_{1,2} & \\
2 & \frac{poly^{+}(x,K-1)}{x^{3K-1}} & \frac{poly^{+}(x,K-1)}{x^{2K-1}} & \frac{poly(x,K-1)}{x^{K-1}} & \alpha_{2,3} \\
3 & \frac{poly^{+}(x,r)}{x^{3K+r}} & \frac{poly^{+}(x,r)}{x^{2K+r}} & \frac{poly^{+}(x,r)}{x^{K+r}} & \frac{poly(x,r)}{x^{r}}
\end{array}
$$

$\alpha_{i,i+1} = (i+1)K/M$

Figure 8: Matrix $R$ for $c = 3$.

$$\sum_{j=0}^{i} \frac{poly(x,K-1)\ G_{j,j(K-1)}}{x^{(i-j)K+K-1}} + \alpha_{i,i+1} G_{i+1,(i+1)(K-1)} = D'_{i,i} \qquad i = 0 \mathinner{..} c-1 \tag{28}$$

$$\sum_{j=0}^{c} \frac{poly(x,r)\ G_{j,j(K-1)}}{x^{(c-j)K+r}} = D'_{c,c} \tag{29}$$

Figure 8 shows the elements of the resulting matrix $R$ for $c = 3$.

As illustrated in the example, matrix $R$ has a nice recursive structure such that eliminating the first $h$ rows and columns (numbered $0 \mathinner{..} h-1$) of a $n-$dimensional $R$, gives the matrix for $c = n - h + 1$. Exploiting this property, we will prove that the numerator of $det[R]$, which is the denominator of the PGF's, is a polynomial with degree $M$. For this, let us define $R^{(n)}$ to be a $n-$dimensional matrix $R$ (i.e., $c = n - 1$) and let $R^{(n)}_{i,j}$ to denote the matrix resulting from eliminating the $i$th row and the $j$th column of $R^{(n)}$. The recursive property of $R$ can be stated as: $R^{(n-1)} = R^{(n)}_{0,0}$. It must be mentioned that this equality is only in terms of the degrees of the polynomials involved in these matrices and does not apply to the coefficients of these polynomials. We use this concept of the matrix equality in this theorem, because we are only interested in the degree of these polynomials.

Using induction on $c$, we will prove that

$$det[R^{(c+1)}] = \frac{poly(x, cK + r)}{x^{cK+r}}. \tag{30}$$

The basis for this induction is trivially shown for $c = 0$, because

$$det[R^{(1)}] = \frac{poly(x, r)}{x^r}. \tag{31}$$

For the induction steps, assume that the claim is true for $R^{(c)}$, we prove that it is also true for $R^{(c+1)}$. From the definition of the determinant of a matrix,

$$det[R^{(c+1)}] = \frac{poly(x, K-1)}{x^{K-1}} det[R^{(c+1)}_{0,0}] - \alpha_{0,1} det[R^{(c+1)}_{0,1}]. \tag{32}$$

From the recursive property of $R$ and the induction hypothesis,

$$det[R^{(c+1)}_{0,0}] = \frac{poly(x, (c-1)K + r)}{x^{(c-1)K+r}}. \tag{33}$$

For the second determinant in equation (32), notice that the matrix $R^{(c+1)}_{0,1}$ is the same as $R^{(c+1)}_{0,0}$ if first column of the latter is divided by $x^K$. Therefore, $det[R^{(c+1)}_{0,1}] = det[R^{(c+1)}_{0,0}]/x^K$. Substituting into equation (32) proves the claim.

$$\begin{aligned} det[R^{(c+1)}] &= \frac{poly(x, cK + r - 1)}{x^{cK+r-1}} - \alpha_{0,1} \frac{poly(x, (c-1)K + r)}{x^{cK+r}} \\ &= \frac{poly(x, cK + r)}{x^{cK+r}} \end{aligned} \tag{34}$$

We now show that $x = 1$ is one of the poles of the generating functions. This can be easily shown because at $z = 1$, the system of equations is reduced to a homogeneous system of balance equations. This implies that the numerators and the denominators of all the PGF's must be zero at $z = 1$. Recall that $x = zy^2 = z(1 + \lambda/(M\mu)(1 - z))^2$ which has the value of 1 for $z = 1$. This completes the proof of the theorem. □

For the rest of this analysis, let $p_{k,m}(x)$ be the denominator of a system with $K = k$ and $M = m$ which is a $m$-degree polynomial with the variable $x$. We give a formal proof that: (1) $p_{m,m}(x)$ has exactly $m - 1$ zeros in the unit disk, and (2) for any value of $x$ in the unit disk and $b < 1$, *exactly* one root of the equation $x = z(1 + b - bz)^2$ lies inside, and the rest are outside the unit disk.

**Lemma 3** *In a system where $K = M = m$, for all positive values of $\lambda$ and $\mu$, the denominator of the partial generating functions, $p_{m,m}(x)$, has exactly $m - 1$ zeros inside the unit disk.*

**Proof:** Starting from equations (26) and (27), we have:

$$\left[1 - \frac{poly^+(x, K-2)}{x^{K-1}}\right] G_{0,0} + G_{1,K-1}/y = \frac{D_{0,0}}{x^{K-1}} \tag{35}$$

$$G_{1,(K-1)} + \frac{C_{0,1,K-1}\, G_{0,0}}{(yz)x^{K-1}} = \frac{D_{1,1}}{(yz)x^{K-1}} \tag{36}$$

which simplifies to

$$\left[1 - \frac{poly^{+}(x, K-1)}{x^K}\right] G_{0,0} = \frac{xD_{0,0} + D_{1,1}}{x^K}. \tag{37}$$

This implies that the denominator of $G_{0,0}$ and all other PGF's is,

$$p_{m,m}(x) = x^m - a_{m-1}x^{m-1} - a_{m-2}x^{m-2} - \ldots - a_1 x - a_0 \tag{38}$$

where $a_i > 0$. Since $x = 1$ is one of the roots of this polynominal, $\sum_{i=0}^{m-1} a_i = 1$. Therefore,

$$p_{m,m}(x) = (x-1)q(x) = (x-1)(x^{m-1} + b_{m-2}x^{m-2} + \ldots + b_1 x + b_0) \tag{39}$$

where $b_0 = a_0$ and $b_i = b_{i-1} + a_{i-1}$ for $0 < i \leq m-2$. Thus, $q(x)$ is a polynomial with positive coefficients such that $0 < b_0 < b_1 < \ldots < b_{m-1} < 1$. According to Kakeya-Eneström's Theorem [4] the zeros of $q(x)$ are all inside the unit disk. □

We strongly believe that this lemma is true for other values of $K < M$ as well. Although, we do not have a formal proof for this claim, we have carried out a large number of experimentations and found no contradictory case. Therefore, we conjecture that,

**Conjecture 1** *Lemma 3 holds for other values of $K < M$.*

We now show that for each zero of the polynomial $p_{m,k}(x)$ that is inside the unit disk, we have exactly one pole for the PGF's that is also inside the unit disk.

**Lemma 4** *For all real and positive values of $b = \lambda/(M\mu)$ and complex values of $x$ with $|x| < 1$, function $F(z) = z(1+b-bz)^2 - x$ has always exactly one zero inside and two zeros outside the unit disk.*

**Proof:** We use Rouché Theorem for the proof. We choose, $f(z) = z(1+b)(1+b-bz) - x$ and $g(z) = -bz^2(1+b-bz)$. Then $F(z) = f(z) + g(z)$. At $|z| = 1$, $|g(z)| = b|1+b-bz|$, and

$$|f(z)| \geq (1+b)|1+b-bz| - |x| > (1+b)|1+b-bz| - 1. \tag{40}$$

Since $|1+b-bz| \geq 1$, this implies that at $|z| = 1$, $|f(z)| > |g(z)|$. According to the Rouché Theorem, the number of the zeros of $F$ inside the unit disk is the same as that for $f$. We will show that the number of zeros of $f$ inside the unit disk is exactly one. For this we use the same theorem one more time. Let $h_1(z) = z(1+b)^2 - x$ and $h_2(z) = -b(1+b)z^2$. Clearly, $f(z) = h_1(z) + h_2(z)$. At $|z| = 1$ again, $|h_1(z)| > (1+b)^2 - 1 \geq |h_2(z)| = b(1+b)$. Also, $x/(1+b)^2$ is the only zero of $h_1(z)$ which is clearly inside the unit disk. Therefore, $f$ and subsequently $F$ have exactly one zero inside the unit disk. Moreover, $F$ cannot have $|z| = 1$ as its zero, since for any such $z$, $|F(z)| \geq |1+b-bz| - |x| \geq 1 - |x| > 0$. □

Next, we prove the stability condition and provide the processor utilization of the system.

**Lemma 5** *The stability condition of the system, when $\mu_1 = \mu_2 = \mu$, is $2\lambda < M\mu$.*

**Proof:** We have proved (formally for $K = M$) that out of $3M$ poles of the PGF's, $M-1$ lie inside the unit disk for all values of $b = \lambda/M\mu$. According to the solution technique, the initial probabilities are computed such that these poles are also zeros of the PGF's. Among other poles, $2(M-1)$ are always outside the unit disk and the remaining poles are the roots of $x = z(1+b-bz)^2 = 1$ which are $z = 1$,

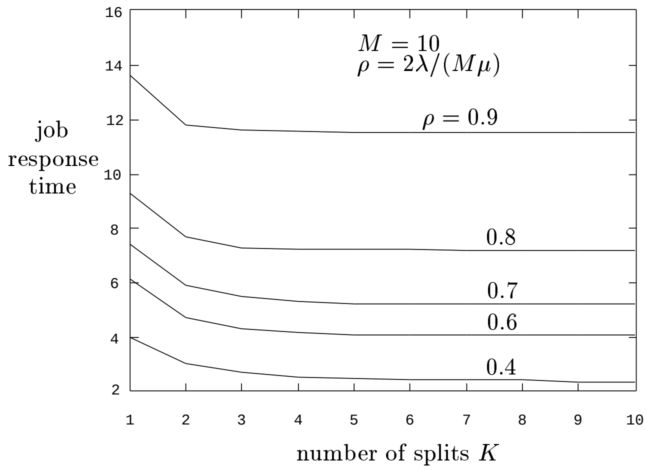


Figure 9: Job response time when $\mu_1 = \mu_2$ at different values of $K$ and $\rho$.

$$\begin{aligned}\beta_1 &= (b + 2 + \sqrt{b(b+4)})/2b, \text{ and} \\ \beta_2 &= (b + 2 - \sqrt{b(b+4)})/2b.\end{aligned}$$

For all $b > 0$, we have $\beta_1 > 1$. But, $\beta_2 \geq 1$ for $b \leq 1/2$ and $0 < \beta_2 \leq 1$ for $b \geq 1/2$. Therefore, eliminating the common poles and zeros, we can say that the simplified PGF's have all of their poles outside the unit disk except for $\beta_2$ which moves in and out of the unit disk depending on the value of $b$. We now prove that $\beta_2$ cannot be a zero of the PGF's. From this and the final value theorem, we can prove the stability condition.

The system is stable iff the PGF's are analytic for $|z| < 1$ and according to the final value theorem, $\lim_{z\to 1}(z-1)G(z) \to 0$. If $\beta_2$ is a zero of the PGF's, then the system would be stable even for $\lambda \geq M\mu$. But we know that, in a stable system, the arriving rate of service request is at least $\lambda$ (when there is no splits) must be less than $M\mu$, the total service capacity of the processors. This is a contradiction! Therefore, $\lim_{z\to\beta_2} G_{i,j} \to \infty$. Since $0 < \beta_2 \leq 1$ for $b \geq 1/2$, we conclude that the system is stable iff $b < 1/2$. □

From this result, we conclude that the average number of busy processors is $2\lambda/\mu$, independent of $K$.

Figure 9 shows the plot of job response time in terms of $K$ for different values of $b$ when $M = 10$. It clearly shows that the job response time reduces as the number of subtasks increases. It also indicates that there is no gain in increasing the number of subtasks after a particular point (4 in this case).

## 3 Conclusions

In this paper we have obtained an exact solution to the problem of characterizing performance of applications involving parallel search implemented on a M-processor system. The solution requires that $M - 1$ poles of the generating function be inside the unit disk. While we have not been able to formally show if this requirement is met in general, it is believed to be the case. We showed that the response time decreases with $K$, but there is a gradual reduction in the rate of decrease. Thus, increasing $K$ beyond some value may not be worthwhile.

In this paper, we assumed exponentially distributed service times. Exact solution to the problem with hyper or hypoexponential service time distribution is still an open problem, and is likely to be very difficult. We conjecture that the response time always decreases with $K$, and the rate increases with the coefficient of variation of service time.

Because of the complexity of solving this and related problems, it may be worthwhile to investigate approximate solutions. One approach that we are investigating is to divide up the available processors among the header tasks and subtasks, so that the processing of the two proceeds independently. The two subproblems are then trivial to solve. The crucial point of this approach is how to do the division of the processors. At this stage we only have some partial results concerning this aspect.

## References


[1] L. Flatto, *Two parallel queues created by arrivals with two demands II*, SIAM J. Applied Math, 45 (Oct. 1985), pp. 861–878.

[2] L. Flatto and S. Hahn, *Two parallel queues created by arrivals with two demands I*, SIAM J. Applied Math, 44 (Oct. 1984), pp. 1041–1053.

[3] M. Ghodsi and K. Kant, *Performance modeling of parallel search algorithms*, Tech. Report CS-89-34, Computer Science Dept., Penn State University, Univ Park, PA 16801, Oct 1989.

[4] P. Henrici, *Applied and Computational Complex Analysis*, vol. 1, Wiley Interscience, 1988.

[5] R. Nelson, D. Towsley, and A. N. Tantawi, *Performance analysis of parallel processing systems*, IEEE Transactions on Software Engineering, (April 1988), pp. 532–540.