ANALYSIS OF THE NUMBER OF OCCUPIED PROCESSORS IN A MULTIPROCESSING SYSTEM

Abdelfettah Belghith

February 1987
CSD-870003

UCLA-CSD-870003
February 1987

ANALYSIS OF THE NUMBER OF OCCUPIED PROCESSORS IN A MULTIPROCESSING SYSTEM

by
Abdelfettah Belghith
Leonard Kleinrock

This research was sponsored by the Defense Advanced Research Projects Agency, Department of Defense.

Computer Science Department
School of Engineering and Applied Science
University of California
Los Angeles

ACKNOWLEDGEMENTS

The support by the Advanced Research Projects Agency of the Department of Defense (Contract number MDA-903-77-C-0272) for this work is greatly appreciated, and we owe a debt of gratitude to the many employees of that contract. We would especially like to thank Lily Chien and Jodi Feiner for their administrative assistance.

This Report is essentially a revision of the Technical Report [Belg85].

ABSTRACT OF THE REPORT

Analysis of the Number of Occupied Processors in a Multiprocessing System

by

Abdelfettah Belghith
Leonard Kleinrock
Computer Science Department
School of Engineering and Applied Science
University of California, Los Angeles

We view a multiprocessor system as a set of P cooperating processors, and a computer job as a set of tasks partially ordered by some precedence relationships, and represented by a directed acyclic graph called a Process Graph. Nodes in the process graph represent the tasks and edges represent the precedence relationships between these tasks.

Many parameters are in play to characterize the underlined multiprocessor system. These are: the job arrival process, the process graph description (number of nodes, number of levels, distribution of the tasks among the levels, and the precedence relationships among the tasks), task processing requirements, and the number of processors in the system.

In this report, we investigate the probability distribution of the number of occupied processors, the generating function of this distribution and its first two moments. In particular, the expected number of busy processors is found to be dependent only on the average number of tasks per job, the job average arrival rate, and the task average processing requirement.

Table of Contents

page

List of Figures

page

ANALYSIS OF THE NUMBER OF OCCUPIED PROCESSORS IN A MULTIPROCESSING SYSTEM

One of the major issues in distributed and parallel processing systems is the evaluation of the system concurrency. Concurrency is a measure of the achievable parallelism, and can be thought of as the number of busy resources which can be utilized simultaneously. In this report, we regard a multiprocessor system as a set of cooperating and identical processors, and a parallel job as a set of tasks partially ordered by some precedence relationships and hence represented by a directed acyclic graph called hereafter a *Process Graph.* Many parameters are thus in play to characterize the terrain of the parallel processing system under investigation. These include: the job arrival process, the process graph description, the task processing requirement, and the number of processors involved.

1 Model Description

A computer job is a set of tasks partially ordered by some precedence relationships and represented by a process graph (PG). A node in the process graph represents a given task, and an edge (i,j) between node i and node j represents the precedence relationship between task i and task j. Edge (i,j) is used to prevent the start of task j execution unless task i execution has been completed. The tasks (i.e., nodes) in the process graph are therefore distributed into levels. Tasks at level one are said *starting tasks*, and tasks at the last level in the process graph are said *terminating tasks*. Any two tasks can be executed concurrently (i.e., in parallel) if and only if every predecessor of one task does not include the other task, and vice versa. Figure 1 gives an example of such a process graph, where the edges are implicitly directed downwards.

The multiprocessor system under consideration consists of a set, say P, of identical processors. Each processor is capable of executing any task. The number P of processors can be infinite or finite. Let $\tilde{N}$ denote the random variable counting the total number of tasks in a process graph (i.e., in a job), $\tilde{r}$ denote the random variable counting the number of levels in a process graph $1 \le \tilde{r} \le \tilde{N}$, $\tilde{X}$ denote the random variable representing the task processing requirement, and $\tilde{Y}$ denote the random variable representing the number of occupied processors.

Since each level in the process graph must have at least one task in it, it follows that the total number of process graphs having N tasks and r levels is equal to the number of ways to distribute (N-r) tasks among the r levels. This number of ways *is** :

$$\binom{(N-r)+r-1}{r-1} = \binom{N-1}{r-1} \qquad (1)$$

Proposition 1

For a fixed number of tasks per job, say N, and a fixed number of levels, say r, and for $1 \le n \le N-r+1$ and $1 \le k \le r$, the probability of having n tasks at level k is given by :

1. If r=1 then: $$P\,[\text{n tasks at level 1}] = \begin{cases} 0 & \text{if } n \ne N \\ 1 & \text{if } n = N \end{cases}$$

2. If $r \ge 2$ then: $$P\,[\text{n tasks at level k}] = \frac{\binom{N-n-1}{r-2}}{\binom{N-1}{r-1}}$$

---

* In fact, the ordinary generating function of such a number of ways is :

$$(x + x^2 + + \cdots + x^k + \cdots)^r = x^r\,(1-x)^{-r}$$

since each level can have from 1 to N-r+1 tasks in it, the number of ways to distribute N tasks among the r levels such that no level is left empty is the coefficient of x^N in the generating function. By the binomial theorem [Liu68], we have:

$$(1-x)^{-r} = \sum_{i=0}^{\infty} \binom{r+i-1}{i} x^i \quad \text{and therefore}$$

$$x^r\,(1-x)^{-r} = \sum_{i=0}^{\infty} \binom{r+i-1}{i} x^{r+i} = \sum_{N=r}^{\infty} \binom{N-1}{r-1} x^N$$

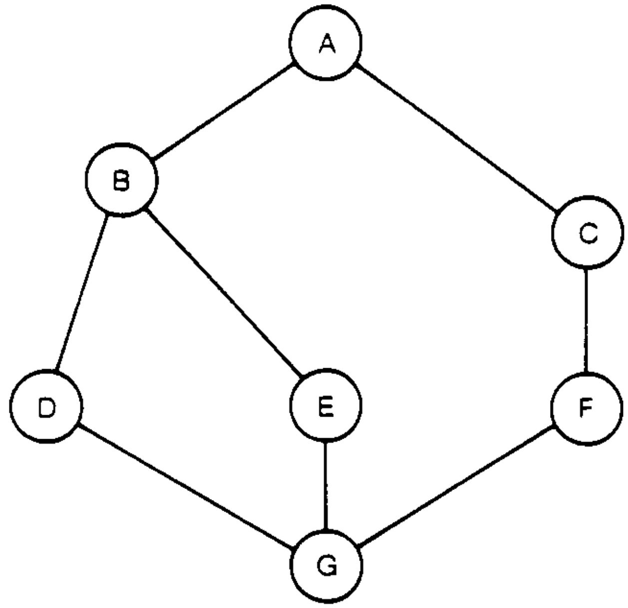


Figure 1: Example of a Process Graph

Proof

The proof of case 1 is trivial since for r=1, all tasks must be at such a level. For N and r fixed and from equation (1), we know that the total number of process graphs that we can have is given by: $\binom{N-1}{r-1}$. Consider now level k, we want to have n tasks at this level where $1 \le n \le N-r+1$. Hence it remains (N-n) tasks for the other (r-1) remaining levels. The number of ways to distribute these (N-n) tasks among the (r-1) levels such that no level is left empty (i.e., the number of process graphs with (r-1) levels and (N-n) tasks), is given by :

$$\binom{\left[(N-n)-(r-1)\right]+(r-1)-1}{(r-1)-1} = \binom{N-n-1}{r-2}$$

This number also represents the number of possibilities for level k, $1 \le k \le r$ to have n tasks out of a total of $\binom{N-1}{r-1}$ possibilities. The probability P[n tasks at level k] is therefore the ratio between them.

∎

Notice that level k can be any level, that is $1 \le k \le r$. The minimum number of tasks any level can have is one, and therefore the maximum number of tasks any level can have is (N-r+1).

Proposition 2

For a fixed number of tasks per job, say N, the probability that a randomly chosen process graph has r levels, $1 \le r \le N$, is given by :

$$P\,[\text{process graph has r levels}] = \frac{\binom{N-1}{r-1}}{2^{N-1}}$$

This conditional probability of having a process graph with r levels given that the number of tasks is fixed to N, is the binomial distribution $b\,(r-1, N-1, \frac{1}{2})$.

Proof

Since the total number of process graphs with N tasks is readily given by $\sum_{r=1}^{N} \binom{N-1}{r-1} = \sum_{r=0}^{N-1} \binom{N-1}{r} = 2^{N-1}$, and since the total number of process graphs with r levels and N tasks is $\binom{N-1}{r-1}$, it follows that:

$$P\,[\text{process graph has r levels} \mid \text{N tasks in it}] = \frac{\binom{N-1}{r-1}}{2^{N-1}} = \binom{N-1}{r-1} \left[\frac{1}{2}\right]^{r-1} \left[\frac{1}{2}\right]^{N-r}$$

$$= b\,(r-1, N-1, \frac{1}{2})$$

∎

Let $\bar{r}$ and σ_r^2 denote respectively the mean number of levels and the variance of the number of levels in a randomly chosen process graph comprising N tasks. From Proposition 2, we readily have:

$$\bar{r} = \frac{N-1}{2} \quad \text{and} \quad \sigma_r^2 = \frac{N-1}{4}$$

This Report is organized into 6 sections. In Section 2, we investigate the case of *Fixed Process Graphs*. All jobs have the same process graph with a fixed number of tasks, say N, and a fixed number of levels, say r. The number of processors is assumed to be infinite, the processing time per task to be constant, say $\bar{X}$, and the job arrival process to be Poisson with parameter λ. The probability distribution, the Z-transform, the average, and the variance of the number of occupied processors are derived. Section 3 deals with *Semi-random Process Graphs* with two or more levels. The case of just one level, being a fixed process graph, has already been treated in Section 2. Each job has a process graph with a fixed number of tasks and a fixed number of levels. The distribution of tasks among the levels, however, may vary from one job to another. The job arrival process is assumed to be Poisson with parameter λ, the number of processors to be infinite, and the processing requirement per task to be constant, say $\overline{X}$. In this Section, we also derive the probability distribution, the Z-transform, the average, and the variance of the number of occupied processors. Section 4 deals with the case of *Random Process Graphs*. Each job has a random process graph with a fixed number of tasks, say N, and a random number of levels (not exceeding the number of tasks). We still assume a Poisson job arrival process with parameter λ, an infinite number of processors, and a constant processing time per task, the same for all the tasks. The Z-transform, the average, and the variance of the number of occupied processors are derived. In Section 5, we generalized our results to random process graphs with a random number of tasks, random number of levels, general task processing requirements, a general job arrival process, and a finite number of processors. In Section 5.1, we further pursue the case of an infinite number of processors, where we first derive a closed form expression of the task arrival process distribution, its mean and variance, and then investigate the average number of occupied processors under the generalized model. In Section 5.2, we consider the finite number of processors case and prove that the average number of occupied processors stays a function of only the job average arrival rate, the task average processing requirement, and the average number of tasks per job. Section 6 concludes the report by providing and discussing profiles of the average number of occupied processors and the system utilization as a function of the number of processors used, and the number of occupied processors and the job average system time as a function of the system utilization factor.

2 Fixed Process Graphs

Throughout this section, the process graph is assumed to have a fixed description, the same for all jobs. All jobs have the same process graph with a fixed number of tasks, N, and a fixed number of levels, r. Moreover, if we let $J(n_1, n_2, \ldots, n_r)$ be the description of the process graph where n_i is the number of tasks at level i in the process graph, then we require that all jobs have the same process graph description. First, we provide an expression for the probability density function of the number of occupied processors. Then, we derive a closed form expression of the Z-transform of the distribution of the number of occupied processors. Closed form expressions for the average and the variance of the number of occupied processors will also be derived.

2.1 Distribution of the Number of Occupied Processors

Let I represent the closed time interval $[t-\bar{X}r, t]$ as depicted in Figure 2. Interval I is divided into r slots of width $\bar{X}$ each (recall that $\bar{X}$ represents the constant service time per task). Jobs arriving before time $(t-\bar{X}r)$ complete execution before time t, and hence do not occupy any processor at time t. On the other hand, a job arriving in slot i of the interval I, participates at time t with the tasks of its ith level. This job will then occupy n_i processors at time t. Hence,

$$P[\tilde{Y}=y] = \sum_{s.t.\ \sum_{i=1}^{r} k_i n_i = y} P[\ k_1 \text{ jobs arrived in slot } 1, \ldots, k_r \text{ jobs arrived in slot r}\ ]$$

Since the job arrival process is Poisson with aggregate rate λ, and the slot width is $\bar{X}$, we obtain:

$$P[\tilde{Y}=y] = e^{-\lambda \bar{X} r} \sum_{s.t.\ \sum_{i=1}^{r} k_i n_i = y} \frac{(\lambda \bar{X})^{(\sum_{i=1}^{r} k_i)}}{k_1! \cdots k_i! \cdots k_r!} \tag{2}$$

Although equation (2) is not an explicit expression, it allows to numerically compute the probability density function of the number of occupied processors in the system. In fact, since $\sum_{i=1}^{r} k_i \le y$, the number of possibilities $(k_1, \ldots, k_i, \ldots, k_r)$ such that $\sum_{i=1}^{r} k_i n_i = y$ is at most equal

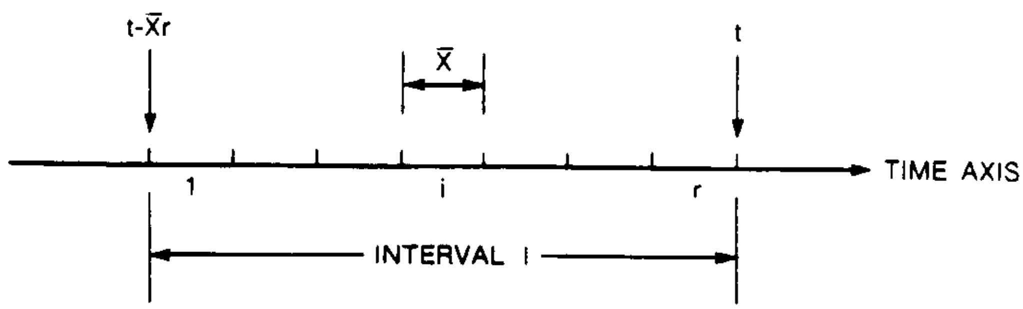


Figure 2: Analysis of Fixed Process Graphs

to the number of ways to distribute y objects among r cells. Hence for a small y, not too many computations are involved in computing $P[\tilde{Y}=y]$. If the job arrival rate, λ, is small, only $P[\tilde{Y}=y]$ for small values of y are of interrest (of any significance). For a large value of λ however, we can see the system as composed of several independent subsystems, say j subsystems, each one comprising an infinite number of processors, and having a Poisson job arrival process with average rate $\frac{\lambda}{j}$. Now the probability density function of the number of occupied processors can be computed using equation (2) and the following:

$$P[\tilde{Y}=y] = \sum_{s.t. \sum_{k=1}^{j} y_k = y} P[\tilde{Y}_k = y_k]$$

where $\tilde{Y}_k$, k=1,...,j represents the number of occupied processors in the kth subsystem.

Now, we proceed to compute the Z-transform of the number of occupied processors in the system. Let $\tilde{Y}_i$, i=1,...,r be the random variable counting the number of processors occupied at time t by the jobs that arrived in slot i of the interval I, we have :

$$\tilde{Y}=\sum_{i=1}^{r}\tilde{Y}_i \tag{3}$$

since

$$P[\tilde{Y}_i=x] = \sum_{k_i=0}^{\infty} P[\tilde{Y}_i=x \mid k_i \text{ jobs arrived in slot i}].P[k_i \text{ jobs arrived in slot i}]$$

and,

$$P[\tilde{Y}_i=x \mid k_i \text{ jobs arrived in slot i}] = \begin{cases} 1 & \text{iff } x=k_i n_i \\ 0 & \text{otherwise} \end{cases}$$

we obtain:

$$P[\tilde{Y}_i = k_i\, n_i] = P[k_i \text{ jobs arrived in slot i}]$$

and since the job arrival process is Poisson with rate λ, and the slot width is $\bar{X}$ (the same as the task service time), we obtain:

$$P[\tilde{Y}_i = k_i\, n_i] = \frac{(\lambda\bar{X})^{k_i}}{k_i!} e^{-\lambda\bar{X}} \tag{4}$$

Let us define by $Y_i(Z)$ the Z-transform of $\tilde{Y}_i$; that is :

$$Y_i(Z) \triangleq \sum_{j=0}^{\infty} P[\tilde{Y}_i=j]\, Z^j \qquad i=1,...,r$$

From equation (4) we get:

$$Y_i(Z) = \sum_{k_i=0}^{\infty} P[\tilde{Y}_i=k_i n_i]\, Z^{k_i n_i} = \sum_{j=0}^{\infty} \frac{(\lambda\bar{X})^j}{j!} e^{-\lambda\bar{X}} Z^{j n_i}$$

which amounts to:

$$Y_i(Z) = \exp\left\{ -\lambda\bar{X}\left[1 - z^{n_i}\right]\right\} \tag{5}$$

Since the job arrival process is Poisson with parameter λ, then the arrivals and the number of such arrivals in any slot i, i=1, ... ,r are independent random variables. It follows that :

$$Y(Z) = \prod_{i=1}^{r} Y_i(Z)$$

using (5) we obtain :

$$Y(Z) = e^{-\lambda\bar{X}r}\, e^{\lambda\bar{X}\sum_{i=1}^{r} Z^{n_i}} \tag{6}$$

Notice from (6) that $Y(0) = e^{-\lambda\bar{X}r} = P_0$, where $P_0 = P$[no job arrivals in the interval I].

2.2 Average Number of Occupied Processors

We now proceed to derive the average number of occupied processors, we have $\bar{Y} = \frac{d}{dZ} Y(Z)\Big|_{Z=1}$ and using equation (6), we get:

$$\bar{Y} = e^{-\lambda\bar{X}r}\left\{\frac{d}{dZ}\left[\lambda\bar{X}\sum_{i=1}^{r} Z^{n_i}\right] . e^{\lambda\bar{X}\sum_{i=1}^{r} Z^{n_i}}\right\}\Bigg|_{Z=1} = e^{-\lambda\bar{X}r}\left\{\lambda\bar{X}\sum_{i=1}^{r} n_i\, z^{n_i-1} . e^{\lambda\bar{X}\sum_{i=1}^{r} Z^{n_i}}\right\}\Bigg|_{Z=1}$$

$$= e^{-\lambda\bar{X}r}\left\{\left[\lambda\bar{X}\sum_{i=1}^{r} n_i\right] . e^{\lambda\bar{X}r}\right\} = \lambda\bar{X}\sum_{i=1}^{r} n_i$$

Therefore, we have: (7)

$$\bar{Y} = \lambda\bar{X}N$$

2.3 Variance of the Number of Occupied Processors

Now, we proceed to derive the variance of the number of occupied processors, we have $\frac{d^2}{dZ^2}Y(Z)\Big|_{Z=1} = \overline{Y^2} - \overline{Y}$. Using equation (6), we obtain:

$$\frac{d^2}{dZ^2}Y(Z) = \frac{d}{dZ}\left[\frac{d}{dZ}Y(Z)\right]$$

$$= e^{-\lambda\bar{X}r}\left\{\lambda\bar{X}\sum_{i=1}^{r} n_i\left[n_i-1\right]Z^{n_i-2}\right\}e^{\lambda\bar{X}\sum_{i=1}^{r}Z^{n_i}} + e^{-\lambda\bar{X}r}\left\{\lambda\bar{X}\sum_{i=1}^{r} n_i Z^{n_i-1}\right\}^2 . e^{\lambda\bar{X}\sum_{i=1}^{r}Z^{n_i}}$$

and by taking Z=1, we obtain:

$$\overline{Y^2} - \overline{Y} = e^{-\lambda\bar{X}r}\left[\lambda\bar{X}\sum_{i=1}^{r} n_i\left[n_i-1\right]\right]e^{\lambda\bar{X}r} + e^{-\lambda\bar{X}r}\left[\lambda\bar{X}\sum_{i=1}^{r} n_i\right]^2 e^{\lambda\bar{X}r}$$

$$= \lambda\bar{X}\sum_{i=1}^{r} n_i\left[n_i-1\right] + (\lambda\bar{X}N)^2 = \lambda\bar{X}\sum_{i=1}^{r} n_i^2 - \lambda\bar{X}N + (\lambda\bar{X}N)^2$$

Since $\sigma_Y^2 = \overline{Y^2} - \overline{Y}^2 = \frac{d^2}{dZ^2}Y(Z)\Big|_{Z=1} + \overline{Y} - \overline{Y}^2$, and using equation (7), we get:

$$\sigma_Y^2 = \lambda\bar{X}\sum_{i=1}^{r} n_i^2 \tag{8}$$

An upper bound for σ_Y^2 can be found by using the loose inequality $\sum_i x_i^2 \le \left[\sum_i x_i\right]^2$. We get from equation (8):

$$\sigma_Y^2 \le \lambda\bar{X}N^2 \quad \text{or equivalently} \quad \sigma_Y^2 \le N\,\overline{Y}$$

Indeed *, this forms a tight upper bound, for it is accomplished by the process graph comprising only one level (i.e., r=1). On the other hand, the process graph with N levels (i.e., each level has one task) provides the lower bound for the variance of the number of occupied processors; that is $\sigma_Y^2 = \lambda\bar{X}N$. Hence, we have:

---

* Formerly, to obtain an upper bound on σ_Y^2, we have to solve the following maximization problem:

maximize $\left\{\sum_{i=1}^{r} n_i^2\right\}$

subject to: $\sum_{i=1}^{r} n_i = N$ for $1 \le r \le N$

$$\lambda \bar{X} N \leq \sigma_Y^2 \leq \lambda \bar{X} N^2 \quad \text{or equaivalently} \quad \bar{Y} \leq \sigma_Y^2 \leq N\bar{Y} \tag{9}$$

In the above analysis, no restriction was assumed as to the choice of the shape of the PG(N,r). Equations (7), (8), and (9) are valid for any given shape of the process graph PG(N,r). The only restriction is that all jobs have the same process graph description $J(n_1, n_2, \ldots, n_r)$.

Examples

Let us take *the discrete well shaped diamond* process graph denoted by PG(r) (as a function of r only); examples of which are depicted in Figure 3 and Figure 4. Two cases may be distinguished depending on the value of r being odd or even.

Case of an odd number of levels

Figure 3 gives some examples of such a process graph. In this case, the number of tasks at level i, i=1,...,r is given by:

$$n_i = \begin{cases} i & 1 \leq i \leq \frac{r+1}{2} \\ r-i+1 & \frac{r+1}{2} \leq i \leq r \end{cases}$$

and hence, the total number N of tasks in the well shaped process graph with an odd number of levels is :

$$N = \sum_{i=1}^{r} n_i = \left[1+2+\ldots+\frac{r+1}{2}\right] + \left[\frac{r-1}{2}+\ldots+2+1\right] = \frac{1}{2}\,\frac{r+1}{2}\,\frac{r+3}{2} + \frac{1}{2}\,\frac{r-1}{2}\,\frac{r+1}{2}$$

which finally yields:

For a given value of r, the solution to this maximization problem is simply $n_k = N-r+1$, $n_j=1$, j=1,...,N and $j \neq k$. It is also easy to see that r=1 gives the upper bound. Likewise, to obtain a lower bound on σ_Y^2, we have to solve the following minimization problem:

$$\text{minimize} \left\{ \sum_{i=1}^{r} n_i^2 \right\}$$

$$\text{subject to:} \quad \sum_{i=1}^{r} n_i = N \quad \text{for} \quad 1 \leq r \leq N$$

For a given value of r, the solution to this minimization problem is simply $n_i = \frac{N}{r}$ for all i=1,...,r. It is easy to see that r=N gives the lower bound.

$$N = \left[\frac{r+1}{2}\right]^2 \tag{10}$$

replacing N in equation (7) by the above expression, yields:

$$\bar{Y} = \frac{\lambda\bar{X}\left[\frac{r+1}{2}\right]^2}{4} \tag{11}$$

On the other hand, using the known identity $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$, we get :

$$\sum_{i=1}^{r} n_i^2 = 2\sum_{i=1}^{\frac{r-1}{2}} i^2 + \left[\frac{r+1}{2}\right]^2 \left[\frac{r+1}{2}\right]^2 \left[\frac{r}{3}+1\right] - \frac{r}{3}\left[\frac{r+1}{2}\right]$$

Now, by using equation (10) we obtain :

$$\sum_{i=1}^{r} n_i^2 = N\frac{r+3}{3} - \frac{r}{3}\sqrt{N}$$

which along with equation (8) gives:

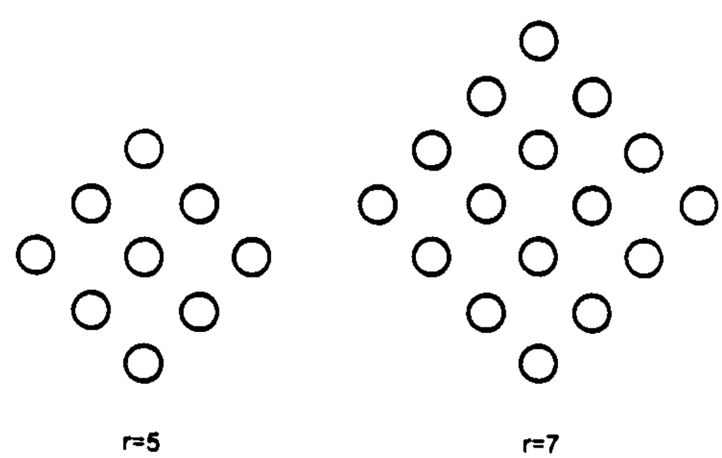


Figure 3: Discrete Well Shaped Process Graph with Odd Number of Levels

$$\sigma_{\bar{Y}}^2 = \lambda\bar{X}N\frac{r+3}{3} - \frac{\lambda\bar{X}r}{3}\sqrt{N} \tag{12}$$

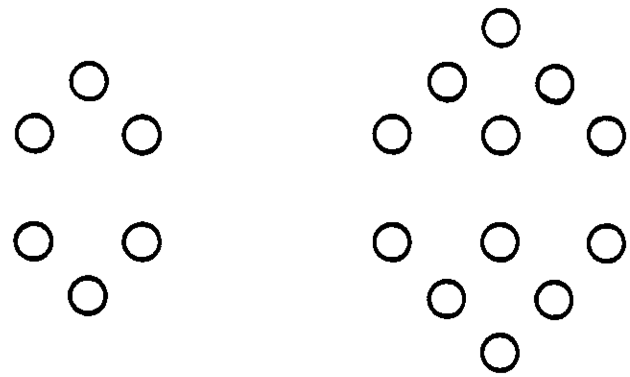


Figure 4: Discrete Well Shaped Process Graph with Even Number of Levels

Case of an even number of levels

Figure 4 gives some examples of such a process graph. In this case, the number of tasks at level i, i=1,...,r is given by:

$$n_i = \begin{cases} i & 1 \le i \le \frac{r}{2} \\ r-i-1 & \frac{r}{2}+1 \le i \le r \end{cases}$$

and hence, the total number N of tasks in the well shaped process graph with an even number of levels is :

$$N = \sum_{i=1}^{r} n_i = \left[1+2+...+\frac{r}{2}\right] + \left[\frac{r}{2}+(\frac{r}{2}-1)+...+2+1\right]$$

which amounts to :

$$N = \frac{r(r+2)}{4} \tag{13}$$

replacing N in equation (7) by the above expression, yields:

$$\bar{Y} = \frac{\lambda \bar{X} r(r+2)}{4} \tag{14}$$

On the other hand, using the known identity $\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$, we get:

$$\sum_{i=1}^{r} n_i^2 = 2\sum_{i=1}^{\frac{r}{2}} i^2 = \frac{r(r+2)}{4}\frac{r+1}{3}$$

now, using equation (13) along with equation (8), we obtain:

$$\sigma_Y^2 = \lambda \bar{X} N \frac{r+1}{3} \tag{15}$$

or equivalently,

$$\sigma_Y^2 = \bar{Y}\frac{r+1}{3}$$

Equations (11) and (14) provide explicit expressions of the average number of occupied processors as a function of the number of levels, for well shaped process graphs having respectively an odd and an even number of levels. Equations (12) and (15) on the other hand, ascertain their variances. Other examples of interrest are studied in [Belg85].

3 Semi Random Process Graphs with two or more Levels

We now proceed to analyze the case of semi random process graphs. Each job has a process graph with a fixed number of tasks, N, and a fixed number of levels, $r \geq 2$. However, jobs do not necessarily have the same process graph description $J(n_1, n_2, \ldots, n_r)$ where n_i i=1,...,r denotes the number of tasks at level i in the process graph. We shall first derive closed form expressions of the probability density function and the Z-transform of the number of occupied processors in the system. Closed form expressions for the average and the variance of the number of occupied processors will also be derived.

3.1 Distribution of the Number of Occupied Processors

Let I define the interval of time [t - $\bar{X}$r,t], $\tilde{Y}_k$ denote the random variable counting the total number of occupied processors at time t given that k jobs arrived in the interval I.

It is easy to see from Figure 5 that all jobs which arrived before time (t-r$\bar{X}$) had finished before time t. Such jobs will not occupy any processor at time t. Those jobs which occupy some

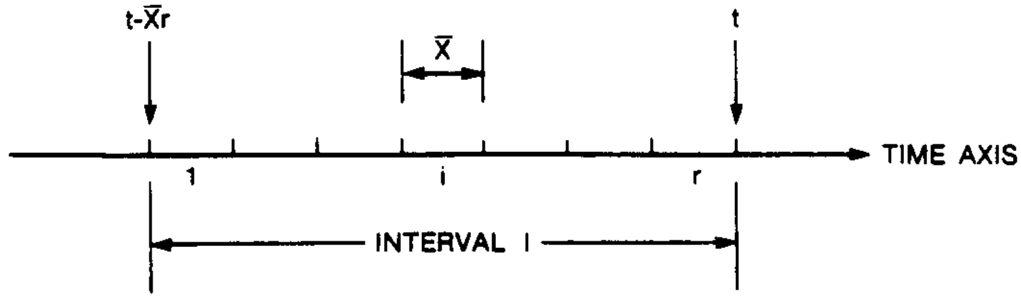


Figure 5 Analysis of Semi Random Process Graphs

processors at time t, are the jobs that arrive in the interval I. Therefore :

$$P[\tilde{Y}=y] = \sum_{k=0}^{\infty} P[\tilde{Y}_k = y] P[k \text{ jobs arrived in I}]$$

Since, if no job arrived in the interval I, then no processor will be occupied at time t, it follows that:

$$P[\tilde{Y}=0] = e^{-\lambda \bar{X} r} \tag{16}$$

On the other hand, if k jobs arrived in the interval I, then at least k processors will be occupied at time t. The number of job arrivals in the interval I and the number of occupied processors at time t are therefore related by the following double sided inequality:

$$k \le y \le k(N-r+1)$$

and hence for $y \geq 1$, we have:

$$P[\tilde{Y}=y] = \sum_{k=1}^{y} P[\tilde{Y}_k = y] P[k \text{ jobs arrived in I}] \qquad y \geq 1 \tag{17}$$

To explicitly express the probability density function of the number of occupied processors in the system, we need to evaluate $P[\tilde{Y}_k = y]$ for the values $1 \le k \le y$. We have:

$$P[\tilde{Y}_k = y \text{ s.t. } 1 \le k \le y] = \sum_{\substack{\text{s.t. } \sum_{j=1}^{k} n_j = y \\ 1 \le n_j \le N-r+1,\ \forall j=1,\dots,k}} P[\text{job j participates with } n_j]$$

Proposition 1 readily gives the probability of having n tasks at a given level k given that we have N tasks and r levels in such process graph n=1,...,N-r+1 ,and k=1,...,r. We then obtain:

$$P[\tilde{Y}_k = y \text{ s.t. } 1 \le k \le y] = \left[\frac{1}{\binom{N-1}{r-1}}\right]^k \sum_{\substack{\text{s.t. } \sum_{j=1}^{k} n_j = y \\ 1 \le n_j \le N-r+1,\ \forall j=1,\ldots,k}} \prod_{j=1}^{k} \binom{N-n_j-1}{r-2} \tag{18}$$

Let us define the following quantities: L=y-k, M=N-2, and R=r-2. Since $r \ge 2$, $N \ge 2$, and $k \le y \le k(N-r+1)$, it follows that $M \ge 0$, $R \ge 0$, and $0 \le L \le (k-1)(N-r+1)$. Using these quantities, equation (18) may be rewritten as:

$$P[\tilde{Y}_k = y \text{ s.t. } 1 \le k \le y] = \left[\frac{1}{\binom{N-1}{r-1}}\right]^k \sum_{\substack{\text{s.t. } \sum_{j=1}^{k} n_j = L \\ n_j \ge 0,\ \forall j=1,\ldots,k}} \prod_{j=1}^{k} \binom{M-n_j}{R}$$

where by definition $\binom{n}{i} \triangleq 0$ whenever $n<i$. Let the bivariate function $a_{k,l}$ be defined as:

$$a_{k,l} = \sum_{\text{s.t. } \sum_{j=1}^{k} n_j = l} \prod_{j=1}^{k} \binom{M-n_j}{R}$$

which amounts to :

$$a_{k,l} = \sum_{i=0}^{l} \left\{ \binom{M-i}{R} \sum_{\substack{\text{s.t. } \sum_{j=1}^{k-1} n_j = l-i \\ \forall n_j \ge 0,\ j=1,\ldots,k-1}} \prod_{j=1}^{k-1} \binom{M-n_j}{R} \right\}$$

Notice that the inner summation is exactly $a_{k-1,l-i}$, and hence we obtain the following recurrence relation on the bivariate function $a_{k,l}$:

$$a_{k,l} = \sum_{i=0}^{l} \binom{M-i}{R} a_{k-1,l-i} \qquad 2 \le k \le y \tag{19}$$

with the following boundary condition for k=1:

$$a_{1,l} = \binom{M-l}{R} \qquad \forall l \ge 0 \tag{20}$$

since this is the case where only one job arrived in the interval I. Finally, using equations (17), (19), and (20), and the fact that the job arrival process is Poisson with aggregate rate λ, we obtain:

$$P[\tilde{Y}=y] = \sum_{k=1}^{y} \left[\frac{\lambda\bar{X}r}{\binom{M+1}{R+1}}\right]^k \frac{1}{k!} e^{-\lambda\bar{X}r} \sum_{i=0}^{L} \binom{M-i}{R} a_{k-1,L-i} \qquad y\geq 1 \tag{21}$$

Equation (16) and equation (21) provide then the probability density function of the number of occupied processors in the system. In the sequel, we derive the Z-transform of the number of occupied processors, denoted hereafter by Y(Z). Since by definition $Y(Z) \triangleq \sum_{y=0}^{\infty} P[\tilde{Y}=y]Z^y$, we can use equations (21) and (16) to derive such a Z-transform. In the sequel, however, we shall take a rather simpler and more elegant way. Let $\tilde{X}_i$, i=1,...,k denote the random variable counting the number of occupied processors at time t, and by the ith job arriving in the interval I. Since only the jobs that have arrived in the interval I will occupy some processors at time t, we have:

$$\tilde{Y}_k = \sum_{i=1}^{k} \tilde{X}_i$$

and from Proposition 1, we already have:

$$P[\tilde{X}_i=n_i] = \frac{\binom{N-n_i-1}{r-2}}{\binom{N-1}{r-1}} \qquad \forall\ n_i=1,\ldots,N-r+1$$

Let $X_i(Z)$ denote the Z-transform of the random variable $\tilde{X}_i$, and $Y_k(Z)$ denote the Z-transform of the random variable $\tilde{Y}_k$. We therefore have:

$$X_i(Z) \triangleq \sum_{j=1}^{N-r+1} P[\tilde{X}_i=j]\, Z^j$$

which amounts to:

$$X_i(Z) = \frac{1}{\binom{N-1}{r-1}} \sum_{j=0}^{N-r} \binom{N-j-2}{r-2} Z^{j+1} \tag{22}$$

Since the random variables $\tilde{X}_i$'s, i=1,...,k are independent and identically distributed, we may drop the index i in $X_i(Z)$, and we get :

$$Y_k(Z) = \left[X(Z)\right]^k$$

and since the job arrival process is Poisson with an average rate λ, we get :

$$Y(Z) = \sum_{k=0}^{\infty} \left[X(Z)\right]^k \frac{(\lambda\bar{X}r)^k}{k!} e^{-\lambda\bar{X}r}$$

which amounts to:

$$Y(Z) = e^{-\lambda \bar{X} r \left[1 - X(Z)\right]} \tag{23}$$

Finally, by using the expression of X(Z) as given by equation (22) into equation (23), we obtain the following expression for the Z-transform Y(Z) of the number of occupied processors:

$$Y(Z) = \exp\left\{-\lambda \bar{X} r \left[1 - \frac{Z \sum_{j=0}^{N-r} \binom{N-j-2}{r-2} Z^j}{\binom{N-1}{r-1}}\right]\right\} \tag{24}$$

3.2 Average Number of Occupied Processors

We now proceed to derive the average number of occupied processors in the case of semi random process graphs. We have $\bar{Y} = \frac{d}{dZ} Y(Z)\Big|_{Z=1}$, where Y(Z) is as given by equation (24). Let b(Z) and a be defined as:

$$b(Z) = \sum_{i=0}^{M-R} \binom{M-i}{R} Z^i \tag{25}$$

and,

$$a = \frac{\lambda \bar{X} r}{\binom{M+1}{R+1}} \tag{26}$$

Therefore, the Z-transform of the number of occupied processors given by equation (24) can be rewritten as follows:

$$Y(Z) = \exp\left\{-\lambda \bar{X} r + aZb(Z)\right\} \tag{27}$$

from the above equation, the average number of occupied processors is then given by:

$$\bar{Y} = ab(1) + a \frac{d}{dZ} b(Z)\Big|_{Z=1} \tag{28}$$

Let us first compute b(1) and $b'(1) = \frac{d}{dZ} b(Z)\Big|_{Z=1}$. From equation (25), we get:

$$b'(1) = \sum_{i=0}^{M-R} i \begin{bmatrix} M-i \\ R \end{bmatrix} \tag{29}$$

Now let us compute b(1), from equation (25), we get :

$$b(1) = \sum_{i=0}^{M-R} \begin{bmatrix} M-i \\ R \end{bmatrix} = \begin{bmatrix} M \\ R \end{bmatrix} + \begin{bmatrix} M-1 \\ R \end{bmatrix} + \begin{bmatrix} M-2 \\ R \end{bmatrix} + \cdots + \begin{bmatrix} R \\ R \end{bmatrix}$$

which amounts to:

$$b(1) = \sum_{i=0}^{M-R} \begin{bmatrix} R+i \\ R \end{bmatrix}$$

Consider now the function $\beta(x)$ defined by :

$$\beta(x) = (1+x)^R + (1+x)^{R+1} + \cdots + (1+x)^M$$

since the coefficient of x^R in $(1+x)^{R+i}$ *is* $\begin{bmatrix} R+i \\ R \end{bmatrix}$, it follows that the coefficient of x^R in $\beta(x)$ is exactly b(1). First, let us rewrite the function $\beta(x)$ as :

$$\beta(x) = \frac{(1+x)^{M+1} - (1+x)^R}{x}$$

Thus, the coefficient of x^R in $\beta(x)$ is $\begin{bmatrix} M+1 \\ R+1 \end{bmatrix}$, and therefore we obtain :

$$b(1) = \begin{bmatrix} M+1 \\ R+1 \end{bmatrix} \tag{30}$$

by replacing M and R by their respective values, it can easily be seen that $X_i(Z)\big|_{Z=1} = 1$. Now, by using equations (28), (29), and (30) we get:

$$\frac{d}{dZ} Y(Z)\Big|_{Z=1} = \lambda \bar{X} r \left[1 + \frac{b'(1)}{\begin{bmatrix} M+1 \\ R+1 \end{bmatrix}} \right] \tag{31}$$

Now, let us compute $b'(1)$, we have:

$$b'(1) = \sum_{i=0}^{M-R} i \begin{bmatrix} M-i \\ R \end{bmatrix} = 0 \begin{bmatrix} M \\ R \end{bmatrix} + 1 \begin{bmatrix} M-1 \\ R \end{bmatrix} + \cdots + i \begin{bmatrix} M-i \\ R \end{bmatrix} + \cdots + (M-R) \begin{bmatrix} R \\ R \end{bmatrix}$$

$$= \sum_{i=0}^{M-R} (M-R-i) \begin{bmatrix} R+i \\ R \end{bmatrix} = (M-R) \sum_{i=0}^{M-R} \begin{bmatrix} R+i \\ R \end{bmatrix} - \sum_{i=0}^{M-R} i \begin{bmatrix} R+i \\ R \end{bmatrix}$$

$$= (M-R)b(1) - \sum_{i=0}^{M-R} i \begin{bmatrix} R+i \\ R \end{bmatrix}$$

which by using equation (30), amounts to:

$$b'(1) = (M-R) \begin{bmatrix} M+1 \\ R+1 \end{bmatrix} - \sum_{i=0}^{M-R} i \begin{bmatrix} R+i \\ R \end{bmatrix} \tag{32}$$

Now we proceed to derive a closed form expression for the summation in the right hand side of the above equation. Let us define the following :

$$a_n = \sum_{i=0}^{n} i \begin{bmatrix} R+i \\ R \end{bmatrix} \qquad n \geq 0 \tag{33}$$

which then results in the following recurrence relation:

$$a_n = a_{n-1} + n + \begin{bmatrix} R+n \\ R \end{bmatrix} \qquad n \geq 1 \tag{34}$$

with the boundary condition:

$$a_0 = 0 \tag{35}$$

Define the generating function of a_n by A(x); that is $A(x) \triangleq \sum_{n=0}^{\infty} a_n x^n$. Thus, equation (34) yields:

$$\sum_{n=1}^{\infty} a_n x^n = \sum_{n=1}^{\infty} a_{n-1} x^n + \sum_{n=1}^{\infty} n \begin{bmatrix} R+n \\ R \end{bmatrix} x^n$$

That is,

$$A(x) - a_0 = xA(x) + x \sum_{n=0}^{\infty} n \begin{bmatrix} R+n \\ R \end{bmatrix} x^{n-1} = xA(x) + x \frac{d}{dx} \left\{ \sum_{n=1}^{\infty} \begin{bmatrix} R+n \\ R \end{bmatrix} x^n \right\}$$

and using equation (A.5) of Appendix (A), yields:

$$A(x) - a_0 = xA(x) + x \frac{d}{dx} \left\{ \frac{1}{(1-x)^{R+1}} - 1 \right\} = xA(x) + x + \frac{R+1}{(1-x)^{R+2}}$$

which finally amounts to:

$$A(x) = \frac{(R+1)x}{(1-x)^{R+3}}$$

We can rewrite A(x) as :

$$A(x) = \frac{R+1}{(1-x)^{R+3}} - \frac{R+1}{(1-x)^{R+2}} \tag{36}$$

Equation (36) can be inverted [Klei75] to give :

$$a_n = (R+1) \binom{n+R+2}{R+2} - (R+1) \binom{n+R+1}{R+1}$$

and, by using the well known formula $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$, we get :

$$\binom{n+R+2}{R+2} = \binom{n+R+1}{R+2} + \binom{n+R+1}{R+1}$$

hence, we obtain :

$$a_n = (R+1) \binom{n+R+1}{R+2} \tag{37}$$

Now, using equation (32) and equation (37), we get:

$$\frac{d}{dZ} b(Z)\Big|_{Z=1} = (M-R) \binom{M+1}{R+1} - (R+1) \binom{M+1}{R+2} \tag{38}$$

and using equations (31) and (38), we obtain :

$$\frac{d}{dZ} Y(Z)\Big|_{Z=1} = \lambda \bar{X} r \, \frac{M+2}{R+2}$$

Finally, since M=N-2 and R=r-2 we obtain :

$$\bar{Y} = \lambda \bar{X} N \tag{39}$$

3.3 Variance of the Number of Occupied Processors

Now, we proceed to derive the variance of the number of occupied processors, denoted by $\sigma_{\bar{Y}}^2$. We have:

$$\sigma_{\bar{Y}}^2 = \frac{d^2 Y(Z)}{dZ^2}\Big|_{Z=1} + \frac{dY(Z)}{dZ}\Big|_{Z=1} - \left[\frac{dY(Z)}{dZ}\Big|_{Z=1} \right]^2$$

Using equation (27), we obtain:

$$\frac{d^2 Y(Z)}{dZ^2}\Big|_{Z=1} = 2ab'(1) + ab''(1) + \left[ab(1) + ab'(1) \right]^2 \tag{40}$$

To evaluate the above expression, we need to compute $b''(1) = \frac{d^2 b(Z)}{dZ^2}\Big|_{Z=1}$. From equation (25), we obtain :

$$b''(1) = \sum_{i=0}^{M-R} i(i-1) \binom{M-i}{R}$$

$$= 0.\binom{M}{R} + 0.\binom{M-1}{R} + 2.\binom{M-2}{R} + 6.\binom{M-3}{R} + \cdots + (M-R)(M-R-1).\binom{R}{R}$$

$$= \sum_{i=0}^{M-R} (M-R-i)(M-R-i-1) \binom{R+i}{R}$$

which can be rewritten as:

$$b''(1) = (M-R)(M-R-1) \sum_{i=0}^{M-R} \binom{R+i}{R} - 2(M-R) \sum_{i=0}^{M-R} i \binom{R+i}{R} + \sum_{i=0}^{M-R} i(i+1) \binom{R+i}{R} \tag{41}$$

We then need to compute $\sum_{i=0}^{M-R} i(i+1) \binom{R+i}{R}$. Let:

$$a_n = \sum_{i=0}^{n} i(i+1) \binom{R+i}{R} \qquad n \geq 0 \tag{42}$$

which results into the following recurrence relation :

$$a_n = a_{n-1} + n(n+1) \binom{R+n}{n} \qquad n \geq 1 \tag{43}$$

with the boundary condition

$$a_0 = 0 \tag{44}$$

Let A(x) define the generating function of a_n; that is $A(x) \triangleq \sum_{n=0}^{\infty} a_n x^n$. Therefore, equation (43) yields:

$$\sum_{n=1}^{\infty} a_n x^n = \sum_{n=1}^{\infty} a_{n-1} x^n + \sum_{n=1}^{\infty} n(n+1) \binom{R+n}{R} x^n$$

which gives :

$$A(x) = \frac{x}{1-x} \sum_{n=1}^{\infty} n(n+1) \binom{R+n}{R} x^{n-1} \tag{45}$$

On the other hand, we have:

$$\sum_{n=1}^{\infty} n(n+1) \begin{bmatrix} R+n \\ R \end{bmatrix} x^{n-1} = \frac{d^2}{dx^2} \left\{ \sum_{n=1}^{\infty} \begin{bmatrix} R+n \\ R \end{bmatrix} x^{n+1} \right\} = \frac{d^2}{dx^2} \left\{ x \sum_{n=1}^{\infty} \begin{bmatrix} R+n \\ R \end{bmatrix} x^{n} \right\}$$

Now, using equation (A.5) of Appendix (A), we obtain:

$$\sum_{n=1}^{\infty} n(n+1) \begin{bmatrix} R+n \\ R \end{bmatrix} x^{n-1} = \frac{d^2}{dx^2} \left\{ x \left[\frac{1}{(1-x)^{R+1}} - 1 \right] \right\} = \frac{R(1+R)x + 2(1+R)}{(1-x)^{R+3}}$$

hence equation (45) becomes:

$$A(x) = \frac{R(1+R)x^2 + 2(1+R)x}{(1-x)^{R+4}}$$

which can be rewritten as

$$A(x) = \frac{R(1+R)}{(1-x)^{R+2}} - \frac{2(1+R)^2}{(1-x)^{R+3}} + \frac{(1+R)(2+R)}{(1-x)^{R+4}} \tag{46}$$

The above equation can be easily inverted [Klei75] to obtain:

$$a_n = 2(1+R) \begin{bmatrix} R+n+1 \\ R+2 \end{bmatrix} + (1+R)(2+R) \begin{bmatrix} R+n+1 \\ R+3 \end{bmatrix} \qquad n \geq 1 \tag{47}$$

Let us now return to the expression of $b''(1)$ given by equation (41). Using equations (30), (37), and (47), we obtain:

$$b''(1) = (M-R)(M-R-1) \begin{bmatrix} M+1 \\ R+1 \end{bmatrix} - 2(M-R)(R+1) \begin{bmatrix} M+1 \\ R+2 \end{bmatrix}$$

$$+ 2(1+R) \begin{bmatrix} M+1 \\ R+2 \end{bmatrix} + (1+R)(2+R) \begin{bmatrix} M+1 \\ R+3 \end{bmatrix}$$

which after some algebra yields:

$$b''(1) = (M-R)(M-R-1) \begin{bmatrix} M+1 \\ R+1 \end{bmatrix} + 2(1+R)(1+R-M) \begin{bmatrix} M+1 \\ R+2 \end{bmatrix} + (1+R)(2+R) \begin{bmatrix} M+1 \\ R+3 \end{bmatrix} \tag{48}$$

Now, let us return to the computation of σ_Y^2; using equation (39) and equation (40), we obtain:

$$\sigma_Y^2 = ab(1) + 3ab'(1) + ab''(1) \tag{49}$$

where a is given by equation (26), b(1) is given by equation (30), $b'(1)$ is given by equation (38), and $b''(1)$ is given by equation (48). Using the facts that

$$\frac{\begin{bmatrix} M+1 \\ R+2 \end{bmatrix}}{\begin{bmatrix} M+1 \\ R+1 \end{bmatrix}} = \frac{M-R}{2+R} \quad \text{and} \quad \frac{\begin{bmatrix} M+1 \\ R+3 \end{bmatrix}}{\begin{bmatrix} M+1 \\ R+1 \end{bmatrix}} = \frac{(M-R)(M-R-1)}{(2+R)(3+R)}$$

and after some algebra, we obtain:

$$\sigma_Y^2 = \lambda \bar{X} r + \lambda \bar{X} r (M-R) \left\{ M-R+2 + \frac{1+R}{2+R}(2R-2M-1) + \frac{1+R}{3+R}(M-R-1) \right\}$$

replacing M and R by their respective values as a function of N and r, we obtain:

$$\sigma_Y^2 = \lambda \bar{X} r + \lambda \bar{X} r (N-r) \left\{ N-r+2 + \frac{r-1}{r}(2r-2N-1) + \frac{r-1}{r+1}(N-r-1) \right\} \tag{50}$$

Equation (50) provides then the variance of the number of occupied processors as a function of both the total number of tasks, N, and the number of levels, r. Moreover, from the previous section, we already know that the upper bound and the lower bound on σ_Y^2 are obtained respectively by the process graph having just one level, and the process graph having N levels. Figure 6 depicts the variance σ_Y^2 for the value N=10, and for $\lambda \bar{X}=1$, and as a function of the number of levels $1 \le r \le 10$. When r=1, we observe that the variance gets its highest value of 100 as expected by equation (9). For r=10 on the other hand, the variance gets its lowest value of 10. We purposely joined the points in Figure 6 by straight lines to shed light on the slope of the decrease in the variance when we move from r=1 to r=10. We observe that for small value of r, the decrease in the variance is very substantial, and as r approaches the number of tasks N, the decrease in the variance (respectively the increase in the slope) gets smaller.

4 Random Process Graphs

We now proceed to analyze the case of random process graphs. Each job has a random process graph with a fixed number of tasks, N, and a random number of levels $\tilde{r}$, $1 \le \tilde{r} \le N$. Moreover, two jobs having the same number of levels do not necessarily have the same process graph description $J(n_1, \ldots, n_i, \ldots, n_r)$, where n_i, i=1,...,r denotes the number of tasks at level i. We shall first derive a closed form expression of the Z-transform of the number of occupied processors in the system. Closed form expressions for the average and the variance of the number of occupied processors will then be derived.

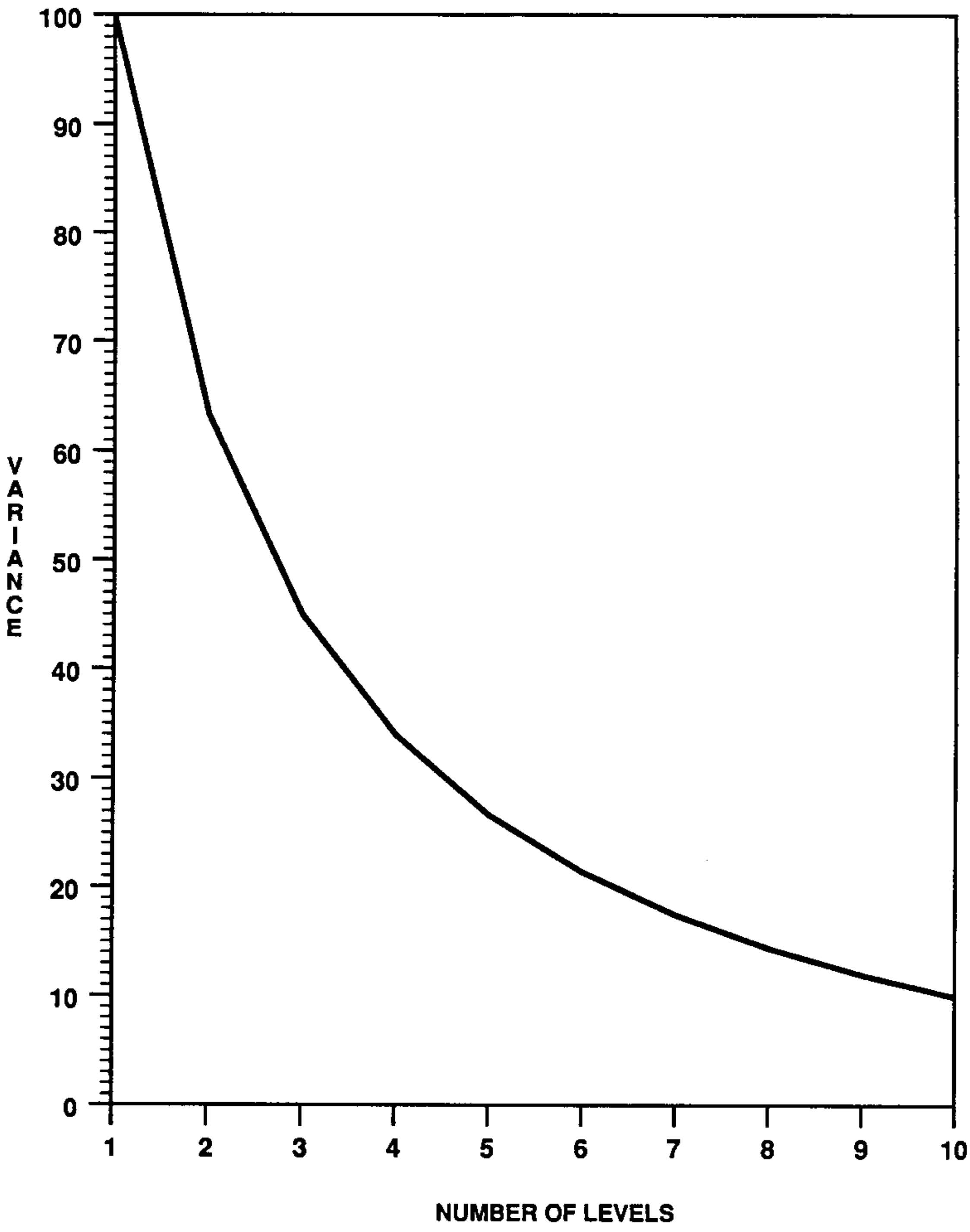


Figure 6: Variance of the Number of Occupied Processors versus the Number of Levels; for Semi Random Process Graphs

4.1 Distribution of the Number of Occupied Processors

From Proposition 2, we know that the probability of an incoming job to have r levels follows the Binomial probability distribution given by :

$$P[\tilde{r}=r] = \frac{\binom{N-1}{r-1}}{2^{N-1}} = b(r-1, N-1, \frac{1}{2})$$

Proposition 1, on the other hand, gives the probability of having n tasks at a given level k given that the job has r levels and N tasks, where n=1,...,N-r+1, k=1,...,r and r=1,...,N.

From Figure 7, we see that any job that had arrived before time (t-N$\bar{X}$) would not participate (occupy any processor) at time t. On the other hand, a job arriving in the interval I=(t-N$\bar{X}$,t) will participate if and only if it has enough levels. Let us divide the interval I into N equal slots of duration $\bar{X}$ units of time each, equal to the processing time of one task. We number such slots by 1,2,...,N (see Figure 7). It follows that a job arriving in slot number i, will occupy some processors at time t if and only if it has at least i levels, where $1 \le i \le N$.

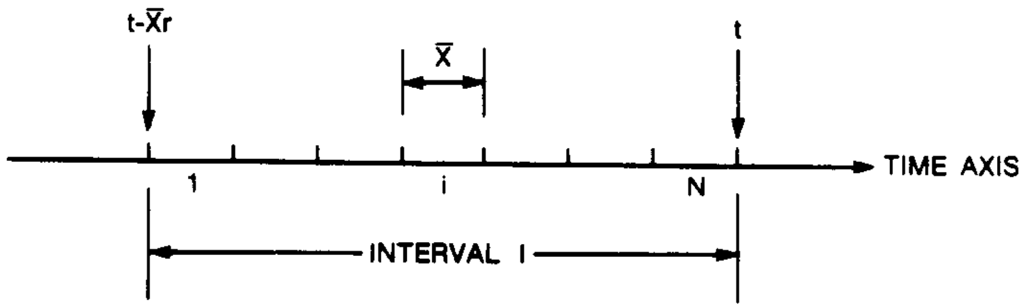


Figure 7 Analysis of Random Process Graphs

Proposition 3

The probability of a job arriving in slot i, $1 \le i \le N$ to occupy some processors at time t is given by:

$$P\,[\text{job arriving in slot i occupies some processors at time t}] = \frac{\sum_{j=0}^{i-1} \binom{N-1}{j}}{2^{N-1}}$$

Proof

A job arriving in slot i, $1 \le i \le N$ occupies some processors at time t if and only if it has at least i levels. Hence, from Proposition 2 we get:

$$P\,[\text{job arriving in slot i occupies some processors at time t}] = \sum_{r=i}^{N} \frac{\begin{bmatrix} N-1 \\ r-1 \end{bmatrix}}{2^{N-1}} = \frac{1}{2^{N-1}} \sum_{j=1}^{i} \begin{bmatrix} N-1 \\ N-j \end{bmatrix}$$

and since $\begin{bmatrix} N-1 \\ N-j \end{bmatrix} = \begin{bmatrix} N-1 \\ j-1 \end{bmatrix}$, we obtain:

$$\sum_{j=1}^{i} \begin{bmatrix} N-1 \\ N-j \end{bmatrix} = \sum_{j=1}^{i} \begin{bmatrix} N-1 \\ j-1 \end{bmatrix} = \sum_{j=0}^{i-1} \begin{bmatrix} N-1 \\ j \end{bmatrix}$$

IIII

Let $\tilde{Y}_k$ denote the random variable counting the total number of occupied processors at time t given that k jobs arrived in the interval I, $\tilde{Y}_{(k_1,\ldots,k_N)}$ denote the random variable counting the total number of occupied processors at time t given that k_i jobs arrived in slot i, i=1,...,N, $\tilde{X}_{i_{k_i}}$ denote the random variable counting the number of occupied processors at time t due to jobs that arrived in slot i and given that k_i jobs arrived in slot i, i=1,...,N, and $\tilde{X}_{i,j}$ denote the random variable counting the number of occupied processors by the jth job that arrived in slot i, i=1,..,N. From these definitions, we readily have :

$$P\,[\tilde{Y} = y] = \sum_{k=0}^{\infty} P\,[\tilde{Y}_k = y\,]\, \frac{(\lambda \bar{X} N)^k}{k!}\, e^{-\lambda \bar{X} N} \tag{51}$$

$$P\,[\tilde{Y}_k = y] = \sum_{s.t.\ \sum_{i=1}^{N} k_i = k} P\,[\tilde{Y}_{(k_1,\ldots,k_N)} = y] = \frac{k!}{k_1!\ldots k_N!} \left[\frac{1}{N}\right]^k \tag{52}$$

$$\tilde{Y}_{(k_1,\ldots,k_N)} = \sum_{i=1}^{N} \tilde{X}_{i_{k_i}} \tag{53}$$

$$\tilde{X}_{i_{k_i}} = \sum_{j=1}^{k_i} \tilde{X}_{i,j} \tag{54}$$

Also, define the following Z-transforms of the above defined random variables.

$$Y_k(Z) \triangleq \sum_{y} P\,[\tilde{Y}_k = y] Z^y$$

$$Y_{(k_1,\ldots,k_N)}(Z) \triangleq \sum_y P[\tilde{Y}_{(k_1,\ldots,k_N)}=y]Z^y$$

$$X_{i_{k_i}}(Z) \triangleq \sum_x P[\tilde{X}_{i_{k_i}}=x]Z^x$$

$$X_{i,j}(Z) \triangleq \sum_x P[\tilde{X}_{i,j}=x]Z^x$$

Since the random variables $\tilde{X}_{i,j}$'s are independent and identically distributed $\forall$ j=1,..., k_i, then using equation (54), we obtain:

$$X_{i_{k_i}}(Z) = \prod_{j=1}^{k_i} X_{i,j}(Z)$$

Let us denote by $X_i(Z) \triangleq X_{i,j}(Z)$ since the $\tilde{X}_{i,j}$'s are i.i.d. . Thus we get:

$$X_{i_{k_i}}(Z) = \left[X_i(Z)\right]^{k_i}$$

and from equation (53), we get:

$$Y_{(k_1,\ldots,k_N)}(Z) = \prod_{i=1}^{N} \left[X_i(Z)\right]^{k_i}$$

and from equation (52) we obtain:

$$Y_k(Z) = \sum_{s.t.\ \sum_{i=1}^{N} k_i=k} \frac{\left[X_1(Z)\right]^{k_1}}{k_1!} \cdots \frac{\left[X_N(Z)\right]^{k_N}}{k_N!}\, k!\left[\frac{1}{N}\right]^k = \left[\frac{\sum_{i=1}^{N} X_i(Z)}{N}\right]^k$$

and finally using equation (51), we obtain:

$$Y(Z) = \sum_{k=0}^{\infty} \left[\frac{\sum_{i=0}^{N} X_i(Z)}{N}\right]^k \frac{(\lambda\bar{X}N)^k}{k!} e^{-\lambda\bar{X}N}$$

which amounts to:

$$Y(Z) = e^{-\lambda\bar{X}N} \,.\, e^{\lambda\bar{X}\sum_{i=1}^{N} X_i(Z)} \tag{55}$$

4.2 Average Number of Occupied Processors

We now proceed to derive the average number of occupied processors in the case of random process graphs. We have $\bar{Y} = \frac{d}{dZ} Y(Z)\Big|_{Z=1}$, where the Z-transform Y(Z) of the number of processors is as given by equation (55). Hence, we obtain:

$$\bar{Y} = \lambda \bar{X} \sum_{i=1}^{N} \left\{ \frac{dX_i(Z)}{dZ}\Big|_{Z=1} \right\} \tag{56}$$

We need now to evaluate the Z-transform $X_i(Z)$; concentrating on slot i, we have:

$$P[\tilde{X}_{i,j} = x] = \sum_{r=1}^{N} P[\tilde{X}_{ij} = x / \tilde{r}=r] P[\tilde{r}=r]$$

where for $i \geq 2$, we have:

$$P[\tilde{X}_{i,j}=x/\tilde{r}=r] = \begin{cases} 0 & \text{if } x \neq 0 \quad 1 \leq r \leq i-1 \\ 1 & \text{if } x = 0 \quad 1 \leq r \leq i-1 \\ \dfrac{\binom{N-x-1}{r-2}}{\binom{N-1}{r-1}} & \text{if } x \geq 1 \quad i \leq r \leq N \end{cases} \tag{57}$$

and for i=1, we have:

$$P[\tilde{X}_{1j}=x/\tilde{r}=r] = \begin{cases} 0 & \text{if } x \neq N \quad r=1 \\ 1 & \text{if } x = N \quad r=1 \\ \dfrac{\binom{N-x-1}{r-2}}{\binom{N-1}{r-1}} & x \geq 1 \quad r \geq 2 \\ 0 & \text{otherwise} \end{cases} \tag{58}$$

Since we know that $P[\tilde{r}=r] = \dfrac{\binom{N-1}{r-1}}{2^{N-1}}$, and by using equation (57) and Proposition 3, we obtain for the case of $i \geq 2$:

$$P[\tilde{X}_{i,j}=x] = \begin{cases} \dfrac{\sum_{r=1}^{i-1} \begin{bmatrix} N-1 \\ r-1 \end{bmatrix}}{2^{N-1}} & x=0 \\ \dfrac{\sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix}}{2^{N-1}} & x\neq 0 \end{cases} \qquad i\geq 2 \tag{59}$$

Therefore, we obtain:

$$X_{i,j}(Z) = \frac{1}{2^{N-1}} \left\{ \sum_{r=1}^{i-1} \begin{bmatrix} N-1 \\ r-1 \end{bmatrix} + \sum_{x\geq 1} \left\{ \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\} Z^x \right\} \qquad i\geq 2 \tag{60}$$

and for the case of i=1, and by using equation (58), we obtain:

$$P[\tilde{X}_{1j}=x] = \begin{cases} 0 & \text{if } x=0 \\ \dfrac{1}{2^{N-1}} & \text{if } x=N \\ \sum_{r=2}^{N} \dfrac{\begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix}}{2^{N-1}} & \text{if } x\geq 1 \end{cases} \tag{61}$$

which then yields:

$$X_{1j}(Z) = \frac{1}{2^{N-1}} \left\{ Z^N + \sum_{x\geq 1} \left\{ \sum_{r=2}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\} Z^x \right\} \qquad i=1 \tag{63}$$

Now, using equations (60) and (63), we get:

$$\sum_{i=1}^{N} \left\{ \frac{dX_i(Z)}{dZ}\bigg|_{Z=1} \right\} = \sum_{i=2}^{N} \frac{1}{2^{N-1}} \sum_{x\geq 1} x \left\{ \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\} + \frac{1}{2^{N-1}} \left\{ N + \sum_{x\geq 1} x \left\{ \sum_{r=2}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\} \right\}$$

Define the quantities A and B by the following expressions:

$$A = \sum_{i=2}^{N} \frac{1}{2^{N-1}} \sum_{x\geq 1} x \left\{ \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

Since we have $X_{ij}(Z)\big|_{Z=1}=1$, we get from equation (60):

$$\sum_{x\geq 1} \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} = 2^{N-1} - \sum_{r=1}^{i-1} \begin{bmatrix} N-1 \\ r-1 \end{bmatrix} \qquad i\geq 2 \tag{62}$$

$$B = \sum_{x \geq 1} x \left\{ \sum_{r=2}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

Hence equations (56) becomes:

$$\overline{Y} = \lambda \overline{X} \left\{ A + \frac{1}{2^{N-1}} \left[N + B \right] \right\} \tag{64}$$

The quantities A and B are evaluated in Appendix (B), where A is given by equation (B.9); that is:

$$A = N - 2 + \left[\frac{1}{2} \right]^{N-1}$$

and B is given by equation (B.10); that is:

$$B = 2^N - (N+1)$$

and therefore equation (64) becomes:

$$\overline{Y} = \lambda \overline{X} N \tag{65}$$

finally, from equation (56) and equation (65), we deduce that:

$$\sum_{i=1}^{N} \left\{ \frac{dX_i(Z)}{dZ} \bigg|_{Z=1} \right\} = N \tag{66}$$

4.3 Variance of the Number of Occupied Processors

Now, we proceed to derive a closed form expression for the variance of the number of occupied processors, denoted by σ_Y^2. We have:

$$\sigma_Y^2 = \frac{d^2 Y(Z)}{dZ^2} \bigg|_{Z=1} + \overline{Y} - \overline{Y}^2$$

where $\overline{Y}$ is the average number of occupied processors and is given by equation (65). Using equation (55), we obtain:

$$\frac{d^2 Y(Z)}{dZ^2} = \frac{dY(Z)}{dZ} \frac{d}{dZ} \left\{ \lambda \overline{X} \sum_{i=1}^{N} X_i(Z) \right\} + Y(Z) \frac{d^2}{dZ^2} \left\{ \lambda \overline{X} \sum_{i=1}^{N} X_i(Z) \right\}$$

where $X_i(Z)$ is given by equation (60) for $i \geq 2$ and by equation (63) for i=1. Now, using equation (66), the expression of the variance of the number of occupied processors becomes:

$$\sigma_Y^2 = \bar{Y} + \lambda\bar{X}\sum_{i=1}^{N} \left\{ \frac{d^2}{dZ^2} X_i(Z)\Big|_{Z=1} \right\}$$

and using equations (60) and (63), we obtain:

$$\sigma_Y^2 = \bar{Y} + \sum_{i=2}^{N} \frac{\lambda\bar{X}}{2^{N-1}} \left\{ \sum_{x\geq 1} x(x-1) \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\} + \frac{\lambda\bar{X}}{2^{N-1}} \left\{ N(N-1) + \sum_{x\geq 1} \sum_{r=2}^{N} x(x-1) \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

Define the quantities C and D by the following expressions:

$$C = \sum_{i=2}^{N} \frac{1}{2^{N-1}} \sum_{x\geq 1} x^2 \left\{ \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

$$D = \sum_{x\geq 1} x^2 \left\{ \sum_{r=2}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

It follows then that σ_Y^2 becomes:

$$\sigma_Y^2 = \bar{Y} + \lambda\bar{X}C + \frac{\lambda\bar{X}D}{2^{N-1}} + \frac{\lambda\bar{X}N(N-1)}{2^{N-1}} - \sum_{i=2}^{N} \frac{\lambda\bar{X}}{2^{N-1}} \left\{ \sum_{x\geq 1} x \sum_{r=i}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

$$- \frac{\lambda\bar{X}}{2^{N-1}} \sum_{x\geq 1} x \left\{ \sum_{r=2}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$$

$$= \bar{Y} + \lambda\bar{X}C + \frac{\lambda\bar{X}D}{2^{N-1}} + \frac{\lambda\bar{X}N(N-1)}{2^{N-1}} - \lambda\bar{X} \left\{ \sum_{i=1}^{N} \left\{ \frac{dX_i(Z)}{dZ}\Big|_{Z=1} \right\} - \frac{N}{2^{N-1}} \right\}$$

using equations (56) and (66), we obtain:

$$\sigma_Y^2 = \lambda\bar{X} \left\{ C + \frac{D}{2^{N-1}} + \frac{N^2}{2^{N-1}} \right\} \quad (67)$$

The quantities C and D are evaluated in Appendix (B), where C is given by equation (B.17); that is:

$$C = 3N - 10 + \frac{2N+5}{2^{N-1}}$$

and D is given by equation (B.18); that is:

$$D = 3.2^N - N^2 - 2N - 3$$

Therefore equation (67) becomes:

$$\sigma_Y^2 = \lambda\bar{X} \left\{ 3N - 4 + \left[\frac{1}{2}\right]^{N-2} \right\} \quad (68)$$

From the above equation, we observe that for a large value of N (e.g., $N>5$), the variance of the number of occupied processors is:

$$\sigma_Y^2 \approx 3\bar{Y} - 4\lambda\bar{X} \qquad \text{for } N>>1$$

and finally for the value N=1, equation (68) verifies that $\sigma_Y^2 = \lambda\bar{X}$ as provided by equation (7) of the fixed process graph case.

5 Average Number of Busy Processors - The General Case

In the previous sections, we have considered the case of multiprocessor systems with infinite number of processors, and have investigated the distribution of the number of occupied processors for three different process graph models. We have assumed that the task service time is constant, the same for all the tasks, that the process graphs have a fixed and prescribed total number of tasks, and that the job arrival process is Poisson. We have found a rather interresting result stating that the average number of occupied processors in the system is only a function of the job average arrival rate, the task constant service time, and the fixed number N of tasks forming the process graph. The question naturally arises as to what extent can this result be generalized.. This is the aim of the current section.

Our multiprocessor system can be generalized by relaxing all the assumptions made in the previous sections. We shall then consider the infinite and the finite number of processors cases, an arbitrary distribution of the number of tasks per job, an arbitrary conditional distribution for the number of levels in the process graph, arbitrary repartitions of the tasks among the levels, arbitrary task service time distribution, perhaps different service requirements for the different tasks, and general job arrival process. Under these rather general conditions, we shall prove that the average number of occupied processors, in both the infinite number of processors case and the finite number of processors case, remains a function of only the job average arrival rate, the task average service requirements, and the average number of tasks per job.

In Section 5.1, we further pursue the case of an infinite number of processors, where we first derive a closed form expression of the task arrival process distribution, its mean and variance, and then investigate the average number of occupied processors under the above mentioned generalized conditions. In Section 5.2, we consider the finite number of processors case, and prove that the average number of processors in the system stays a function of only the job average arrival rate, the task average service requirements, and the average number of tasks per job. For the finite number of processors case, we assume throughout this section that the system is in equilibrium.

5.1 Infinite Number of Processors

In this section, we pursue the infinite number of processors case. We shall provide a theorem stating that under the generalized model, the average number of occupied processors in the system is only a function of the job arrival average rate, the task average service times, and the average number of tasks per job. First, we investigate the distribution of the task interarrival times to the system.

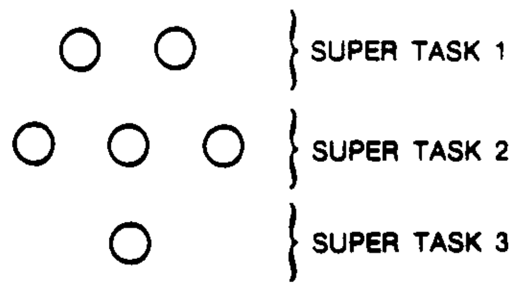
5.1.1 The Task Interarrival Time Process

Since a job is composed of a set of tasks, the task interarrival process is then different from the job arrival process. These two arrival processes are identical only in the case where jobs are composed of only one task. In this section, we derive a closed form expression of the distribution of the task interarrival time process. Throughout the section, we assume that the number of available processors is infinite, the job arrival process is Poisson with average rate λ, and the task service time is constant equal to $\bar{X}$, the same for all tasks. Jobs are represented by the same process graph with N tasks and $1 \le r \le N$ levels. The average and the variance of the task interarrival time will also be derived.

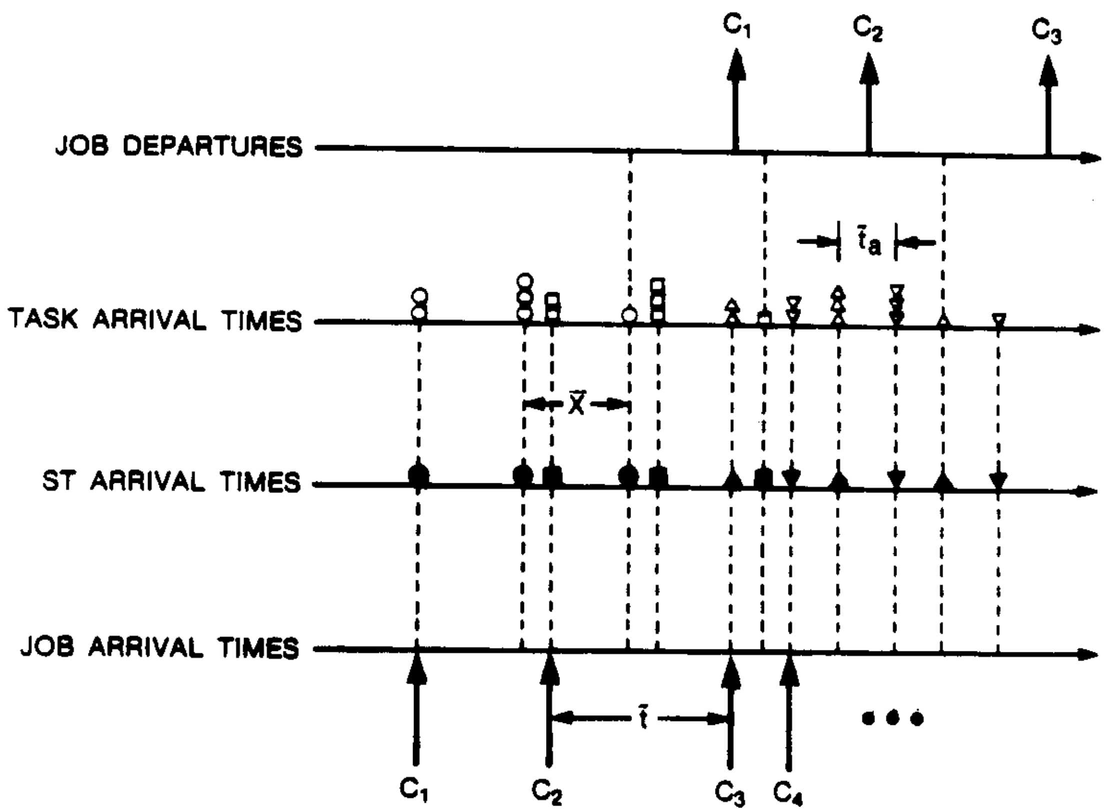
Our multiprocessor system can be viewed as an FCFS queueing system with an infinite number of processors. Let $J = (n_1, n_2, \ldots, n_r)$ denote the process graph description, where n_i is the number of tasks at level i, i=1,..,r, and thus we have $\sum_{i=1}^{r} n_i = N$. Jobs may thus be regarded as a vertical string of super-tasks (ST); where super-task ST_i, i=1,...,r comprises n_i tasks representing the set of tasks at level i in the process graph. Since the number of processors is infinite, the n_i, i=1,...,r tasks forming super-task i are then executed in parallel. Upon the arrival of a job to the system, its starting tasks (i.e., the tasks forming its first super-task) are ready-for-service and thus start execution immediately. Upon the completion of its first super-task, the job feeds back all the tasks forming its second level (i.e., its second super-task), which immediately start their execution. Each $\bar{X}$ seconds thereafter and until completion, the job creates all the tasks of its next level as shown in Figure 8. Figure 8:(b) represents a time diagram of the job arrival process, and the corresponding super-tasks arrival process, the task arrival process, and the job departure process. The process graph description used is depicted in Figure 8:(a). Let us first consider the interarrival time of super-tasks to the system.

Proposition 4

Provided the task constant service time $\bar{X}$ is strictly positive, no simultaneous super-task arrivals can occur.



(a): Process Graph Description



(b): Arrival Processes Time Diagram

Figure 8: Job Arrival and Task Arrival Time Diagram

Proof

Super-task arrivals from the same job are separated exactly by $\bar{X}$ seconds. Since $\bar{X}>0$, then no simultaneous super-task arrivals from the same job can occur. On the other hand, to have simultaneous super-task arrivals from 2 different jobs, the arrival of these two jobs must be separated exactly by $i\bar{X}$ seconds where $0 \le i \le r-1$. But since the job arrival process is Poisson with parameter λ, we have:

$$P\left[\text{job arrival in } [\, t, t+dt \,] \text{ and job arrival in } [t+i\bar{X}\,,\, t+i\bar{X}+dt]\right]$$

$$= P\left[\text{job arrival in } [t, t+dt]\right] . P\left[\text{job arrival in } [t+i\bar{X}\,,\, t+i\bar{X}+dt]\right]$$

$$= (\lambda dt + O(t)).(\lambda dt + O(t)) \;=\; \lambda^2 dt^2 + O(t) \;=\; O(t)$$

The above can also be seen by noticing that the probability of having two arrivals separated by exactly $i\bar{X}$ seconds is the same as the probability of having simultaneous arrivals.

IIII

Proposition 4 says that to characterize the distribution of the tasks interarrival time process, we need to find :

1. the distribution of the super-task interarrival times, and then

2. the distribution of the super-task size

First, we proceed to find the distribution of the super-task interarrival times. Recall that jobs are represented by a process graph with r levels, and thus comprising r super-tasks. Let $\tilde{t}$ denote the random variable measuring the interarrival time between jobs, and $\tilde{t}_a$ denote the random variable measuring the interarrival time between super-tasks. Our objective is to find the probability distribution of the random variable $\tilde{t}_a$; that is $P[\tilde{t}_a \le t]$. In the sequel, we distinguish two cases depending on the value of the number of levels (i.e., the number of super-tasks) in the process graph.

(1): Case r=1

Since each job creates just one super-task and this is exactly upon its arrival to the system, then the distribution of the interarrival time between super-tasks is the same as the job interarrival time distribution. We then have :

$$P[\tilde{t}_a \le t] = 1 - e^{-\lambda t} \qquad t \ge 0 \tag{69}$$

(2): Case r ≥2

This is the case of two or more super-tasks per job. Depending on whether the interval of time t is less or equal to the task constant service time $\bar{X}$, we distinguish the two following cases.

(2.1): Case where $0 \le t < \bar{X}$

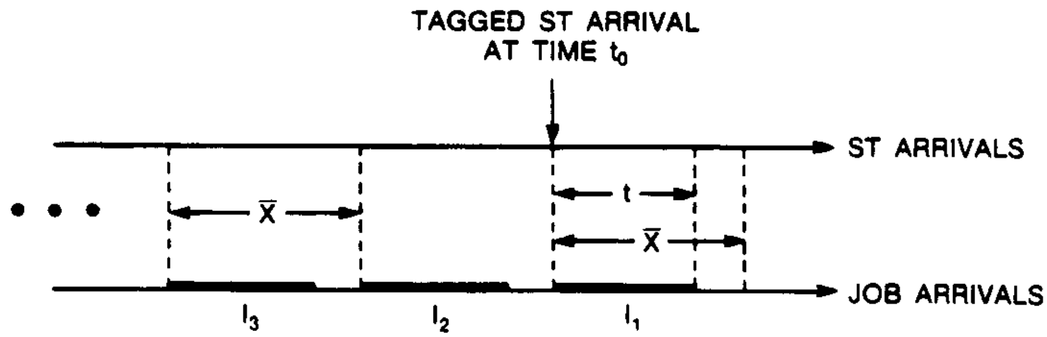


Figure 9: Super-task Interarrival Time Diagram, case of $0 \le t < \bar{X}$

Since $P[\tilde{t}_a \le t] = 1 - P[\tilde{t}_a > t]$, let us first compute $P[\tilde{t}_a > t]$. As depicted in Figure 2.3, let us place ourselves at the tagged super-task arrival time t_0, and compute the probability of no super-task arrivals during the interval of time t. The intervals I_i, i=1,...,r are of the same time length and are equal to t. Therefore, we have:

$$P[\tilde{t}_a > t] = P[\text{no ST arrivals during the interval of time t}]$$

$$= P[\text{no job arrivals in the intervals } I_1, I_2, I_3, \ldots, I_r] = \prod_{i=1}^{r} e^{-\lambda t}$$

which yields:

$$P[\tilde{t}_a \le t] = 1 - e^{-\lambda r t} \qquad 0 \le t < \bar{X} \tag{70}$$

(2.2): Case where $t \ge \bar{X}$

Since $t \ge \bar{X}$, we must distinguish whether $t = \bar{X}$ or $t > \bar{X}$.

(2.2.1): Case where $t=\bar{X}$

Let us position ourselves at a super-task arrival instant, say t_0. From Proposition 4, we know that the next arriving super-task, if any, must belong to the same job as the tagged super-task. Since $t=\bar{X}$ then:

$$P[\tilde{t}_a \le t] = P[\tilde{t}_a \le \bar{X}] = 1 - P[\tilde{t}_a > \bar{X}]$$

on the other hand,

$$P[\tilde{t}_a > \bar{X}] = P[\text{no ST arrivals in the interval } [t_0, t_0+\bar{X})\text{, and no ST arrival at time } (t_0+\bar{X})]$$

and since both events are disjoint we obtain:

$$P[\tilde{t}_a > \bar{X}] = P[\text{no ST arrivals in the interval } [t_0, t_0+\bar{X}]].P[\text{no ST arrivals at time } (t_0+\bar{X})]$$

Finally, from case (2.1) where $0 \le t < \bar{X}$, we get from equation (70):

$$P[\text{no ST arrivals in the interval } [t_0, t_0+\bar{X})] = e^{-\lambda r \bar{X}}$$

and by application of Proposition 4, we obtain:

$$P[\text{no ST arrivals at time } (t_0+\bar{X})] = P[\text{ST is the last super-task of its job}]$$

on the other hand, since a job has r super tasks, and each super-task takes $\bar{X}$ seconds of processing time, it follows that:

$$P[\text{job is executing its ith super-task} \mid \text{job is in the system}] = \frac{1}{r} \qquad i=1,\ldots,r$$

and therefore, we obtain:

$$P[\tilde{t}_a > \bar{X}] = \frac{e^{-\lambda r \bar{X}}}{r} \qquad t=\bar{X} \qquad (71)$$

(2.2.2): Case where $t>\bar{X}$

Let us place ourselves at a super-task arrival instant, say t_0, as indicated in Figure 10. Since the interval of time t is strictly larger than $\bar{X}$, we already know that no job arrivals occur in the interval $(t_0, t_0+\bar{X})$, and that the system becomes empty at time $t_0+\bar{X}$. We therefore have:

$$P[\tilde{t}_a > t] = P[\text{no ST arrivals during } t] = P[\text{no ST arrivals in } [t_0+\bar{X}, t_0+t]]$$

$$= P[\text{no job arrivals during } (t-\bar{X})]$$

which amounts to:

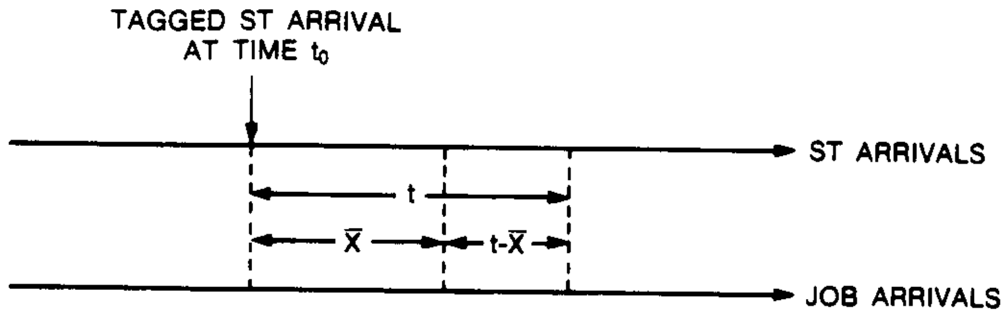


Figure 10: Super-task Interarrival Time Diagram, case of $t>\bar{X}$

$$P[\tilde{t}_a>t] = e^{-\lambda(t-\bar{X})} \qquad t>\bar{X} \tag{72}$$

Finally, putting the two cases together, that is for $t\geq\bar{X}$, we get:

$P[\tilde{t}_a\leq t] = 1 - P[\tilde{t}_a>t]$

$= 1-P$ [no ST arrivals during t]

$= 1-P$ [no ST arrivals during t | no ST arrivals during $\bar{X}$] . P [no ST arrivals during $\bar{X}$]

$- P$ [no ST arrivals during t | ST arrivals during $\bar{X}$] . P [ST arrivals during $\bar{X}$]

Since P [no super-task arrivals during t | super-task arrivals during $\bar{X}$] = 0, then using equation (71) and equation (72), yields:

$$P[\tilde{t}_a\leq t] = 1 - \frac{e^{\lambda\bar{X}(1-r)}}{r}\, e^{-\lambda t} \qquad t\geq\bar{X} \tag{73}$$

Equation (70) along with equation (73) provide an explicit closed form expression of the probability distribution function of the super-task interarrival time process.

5.1.1.1 Average Interarrival Time Between Super Tasks

Let $\bar{t}_a$ denote the average interarrival time between super-tasks. Therefore:

$$\bar{t}_a = \int_0^\infty \left[1 - P[\tilde{t}_a\leq t]\right] dt$$

and using equations (70) and (73), we get:

$$\bar{t}_a = \int_0^{\bar{X}} e^{-\lambda r t}\, dt + \int_{\bar{X}}^{\infty} \frac{e^{\lambda \bar{X}(1-r)}}{r}\, e^{-\lambda t}\, dt$$

which amounts to:

$$\bar{t}_a = \frac{1}{\lambda r} \qquad r \geq 2$$

since for the case of r=1, and from equation (69) we have $\bar{t}_a = \frac{1}{\lambda}$, therefore, we obtain:

$$\bar{t}_a = \frac{1}{\lambda r} \qquad r \geq 1 \tag{74}$$

For any process graph with N tasks and r levels, $1 \leq r \leq N$, and for the case of an infinite number of processors and constant task service time, $\bar{X}$, the average number of jobs occupying some processors is $\lambda r \bar{X}$. This can be seen by noticing that the only jobs which occupy some processors at any given time t, must have arrived in the interval of time (t-r$\bar{X}$,t). On the other hand, since the expected number of busy processors in such a case is $\lambda N \bar{X}$, it follows that the jobs which occupy some processors at any given time t, participate on the average by $\frac{N}{r}$ tasks.

5.1.1.2 Variance of the Interarrival Time Between Super Tasks

Let $\sigma_{t_a}^2$ represent the variance of the interarrival times between super-tasks. We have:

$$\sigma_{t_a}^2 = E[\,\bar{t}_a^{\,2}\,] - E[\,\bar{t}_a\,]^2$$

Using equations (70) and (73), and after some algebra, we obtain:

$$\sigma_{t_a}^2 = \frac{1 - 2(1-r)\, e^{-\lambda \bar{X} r}}{(\lambda r)^2} \qquad r \geq 1 \tag{75}$$

Notice that in the special case where r=1, equation (75) reduces to the variance of the job arrival process (i.e., the variance of the exponential distribution with parameter λ).

In the sequel, we shall find the distribution of the super-task size. Recall that a job is represented by the process graph description $J = (n_1, n_2, \cdots, n_r)$, where n_i is the number of tasks at level i, i=1,...,r. Let $\tilde{S}$ denote the random variable representing the size of a super task, and $\delta_k(i)$ denote the binary function, which is equal one if i=k and equal zero otherwise.

Proposition 5

the distribution of the size of a super-task, given that all jobs have the same fixed process graph description, is given by :

$$P[\tilde{S}=k] = \frac{1}{r} \sum_{j=1}^{r} \delta_k(n_j) \qquad\qquad 1 \le k \le N-r+1$$

Proof

Consider the arrival process of super-tasks to the system (see Figure 8). Take any arrival and call it the tagged arrival. This tagged arrival belongs to a given job, call such a job the tagged job. Hence we have:

P [tagged arrival is the jth ST of the tagged job] =

$$P[\text{tagged job is executing its jth level} \mid \text{tagged job is in the system}] = \frac{1}{r}$$

and therefore:

$$P[\tilde{S}=k] = \frac{1}{r} \cdot \left[\text{number of levels having k tasks}\right]$$

the proof is complete by using the binary function $\delta_k(i)$. The restriction on the value of k is due to the fact that the maximum number of tasks at any level cannot exceed N-r+1.

∎

On the other hand, if jobs are described by semi-random process graphs, the distribution of the super-task size is readily given by Proposition 1; that is:

$$P[\tilde{S}=k] = \frac{\binom{N-k-1}{r-2}}{\binom{N-1}{r-1}} \qquad\qquad r \ge 2\,,\ 1 \le k \le N-r+1$$

and,

$$p[\tilde{S}=k] = \begin{cases} 0 & \text{if } k \ne N \\ 1 & \text{if } k = N \end{cases} \qquad\qquad r=1$$

5.1.2 Expected Number of Busy Processors

We have shown in the previous sections that $\bar{Y}=\lambda N\bar{X}$, where we have assumed for the most general case studied that the job arrival process to the system is Poisson with fixed rate λ, that the total number of tasks per job is fixed to N, and that the task average service time is constant equal to $\bar{X}$, the same for all the tasks. In this section, we show that such a result still holds for the more general case. If we still assume a Poisson job arrival process and a constant task service time, we can see the multiprocessor system as an M/G/∞ queueing system. For this system, it is readily shown [Klei75], that the probability of having k jobs in the system in steady state is given by:

$$P[k] = \frac{\left(\lambda \bar{r} \bar{X}\right)^k}{k!} e^{-\lambda \bar{r} \bar{X}}$$

Since each job in the system participates on the average by $\frac{\bar{N}}{\bar{r}}$ tasks, it follows that $\bar{Y} = \lambda \bar{N} \bar{X}$. We can further generalize our multiprocessor system by relaxing the Poisson job arrival process assumption. Let $\bar{N}$ denote the average number of tasks per job, $\bar{C}_j$ denote the average concurrency per job over all jobs, $\bar{K}$ denote the average number of jobs present in the system, and T be the average time a job spends in the system.

Theorem 1

The expected number of busy processors $\bar{Y}$ in the case of:

1. an infinite number of processors,
2. random service time per task (possibly different service requirement and distribution for each task) with an overall average $\bar{X}$,
3. random job arrival process with average arrival rate λ (but independent job arrivals), and
4. random process graph, that is,
   - N random
   - r random, r=1,...,N
   - random repartition of tasks among levels, and
   - random precedence relationships among levels

is given by:

$$\bar{Y} = \lambda \bar{N} \bar{X}$$

Proof

Since the average number of occupied processors, $\overline{Y}$, represents the average concurrency in the system, it follows that

$$\overline{Y} = \overline{K}\ \overline{C}_j$$

Notice that $\overline{KC_j} = \overline{K}\ \overline{C}_j$, due to the fact that P is infinite. By using Little's formula [Litt61], we have:

$$\overline{K} = \lambda T$$

where the job average system time T can be written as:

$$T = \frac{\overline{N}\overline{X}}{\overline{C}_j}$$

It then follows that:

$$\overline{Y} = \lambda T\overline{C}_j = \lambda \frac{\overline{N}\overline{X}}{\overline{C}_j}\overline{C}_j = \lambda\overline{N}\overline{X}$$

∎

5.2 Finite Number of Processors

In this section, the number of processors in the system is finite, say P. We shall prove that the average number of occupied processors, $\overline{Y}$, is still given by $\overline{Y} = \lambda\overline{N}\overline{X}$. Throughout this section, we assume that the multiprocessor system is in equilibrium.

Theorem 2

If the multiprocessor system is in equilibrium and work-conservative, then the average number of occupied processors $\overline{Y}$ for the case of:

1. finite number of processors, say P,

2. random service time per task (possibly different service requirement and distribution for each task) with overall average $\overline{X}$,

3. random job arrival process with average arrival rate λ (but independent job arrivals), and

4. random process graph per job, that is for each job:
   - N random

- r random, r=1,...,N
- random repartition of tasks among levels, and
- random precedence relationships among levels

is given by:

$$\overline{Y} = \lambda \overline{N}\overline{X}$$

Moreover, if the system is overloaded then

$$\overline{Y} = P$$

Proof

For p=1,...,P, let $\overline{n}_p$ denote the average number of tasks per job processed by processor p, ρ_p denote the utilization factor of processor p, and ρ be the system total utilization factor. The equilibrium condition is then $\forall$ p=1,...,P $\rho_p < 1$ and that $\rho = \sum_{p=1}^{P} \rho_p < 1$. We have:

$$\rho_p = \lambda \overline{n}_p \overline{X} \quad \text{and} \quad \overline{N} = \sum_{p=1}^{P} \overline{n}_p$$

$$\overline{Y} = \sum_{p=1}^{P} \rho_p = \sum_{p=1}^{P} \lambda \overline{n}_p \overline{X} = \lambda \overline{N}\overline{X}$$

If the system is overloaded (i.e., the system utilization factor ρ is greater than one) then it is easy to see that $\overline{Y} = P$ since all the processors are being used all the time.

∎

6 Conclusion

In this report, we have proved that the average number of occupied processors in a multiprocessor system with P=1,2,3,... processors is given by $\overline{Y} = \lambda \overline{N}\overline{X}$, where $\overline{N}$ and $\overline{X}$ represent respectively the average number of tasks per job and the average service time per task. It is interesting to note that the average number of occupied processors does not depend on the jobs description (e.g., the distribution of the number of tasks per job, the distribution of the number of levels in the process graph, the repartition of the tasks among the levels, the precedence relationships among the levels inside the process graph, the distribution of the task service time, the distribution of the job arrival process and the number of processors in the system given that such multiprocessor system is in equilibrium). More importantly, in the case of finite number of processors, the average number of occupied processors is independent of any processor scheduling provided the multiprocessor system is work-conservative.

Figure 11 and Figure 12 provide a pictorial profile of the system utilization and the average number of occupied processors in the system as a function of the total number of processors. In Figure 11, we have $\lambda\overline{NX} < 1$, that is the utilization factor of the system, when P=1, is less than unity. In Figure 12, we have $\lambda\overline{NX} \geq 1$, that is the utilization factor of the system, when P=1, is greater than unity. Notice that whenever $\rho < 1$, the expected number of busy processors is $\overline{Y} = \lambda\overline{NX}$; whereas for $\rho \geq 1$, $\overline{Y} = P$.

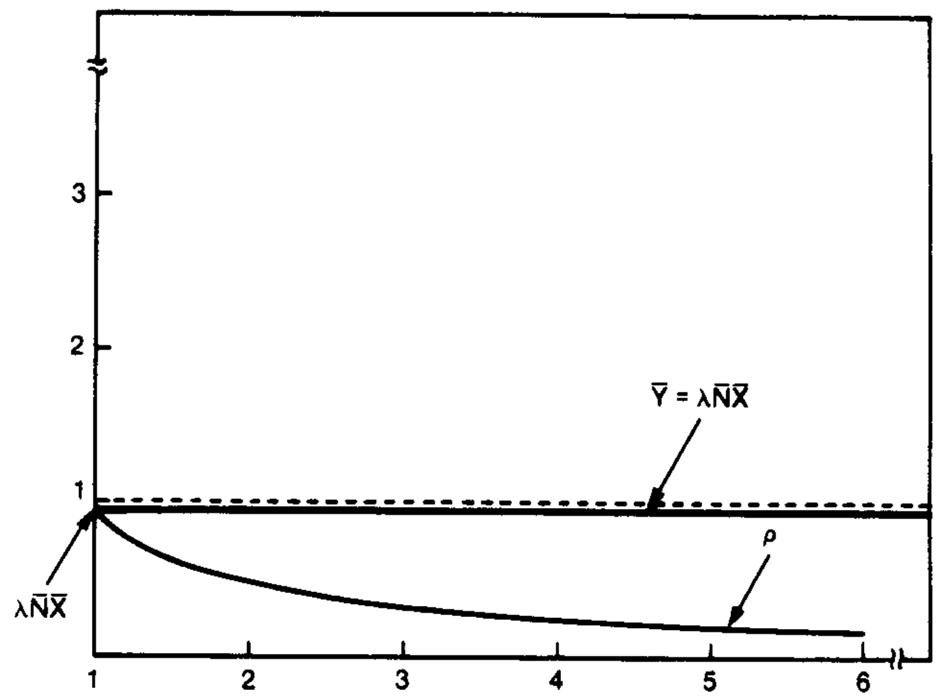


Figure 11: System Utilization and Average Number of Occupied Processors versus P, $\lambda\overline{NX} < 1$

Figure 13 provides a pictorial profile of the average number of occupied processors $\overline{Y}$ and the average system time T as a function of the job arrival rate λ, and for a given number P of processors. In the region where $\rho < 1$, we observe that the expected number of busy processors grows linearly with the number of processors used, and at a constant slope equal to $\overline{NX}$. At $\lambda = \frac{P}{\overline{NX}}$, the system total utilization factor ρ reaches the value one, which results in an average number of occupied processors equal to $\overline{Y} = P$, and an infinite job average system time.

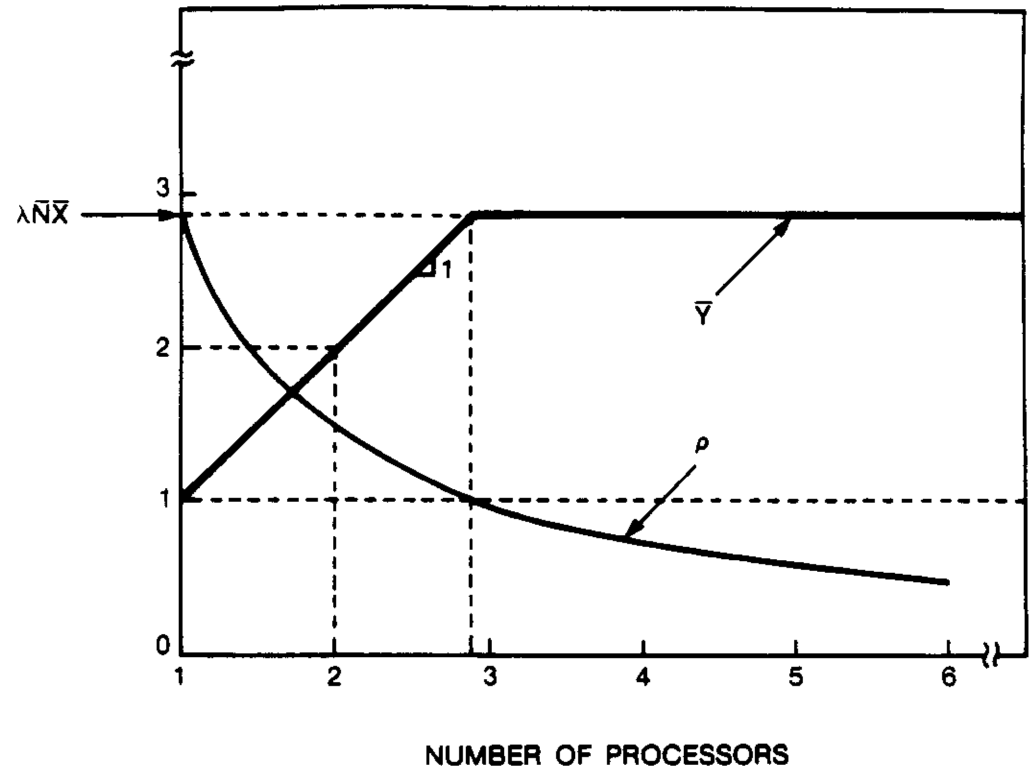


Figure 12: System Utilization and Average Number of Occupied Processors versus P, $\lambda\bar{N}\bar{X} \geq 1$

Speedup Factor

The achievable parallelism (Speedup) can be thought of as the number of busy resources which can be utilized simultaneously. The expected number of busy processors readily ascertain such a measure. The best we can achieve is for the concurrency (equivalently the speedup factor) to grow linearly with P. Indeed, the two previous Theorems witness such a behavior, and prove that for any finite number of processors, the speedup factor is a linear function in P for any value of the system total utilization factor; namely $\bar{Y} = \rho P$.

In practice however, the speedup is much less since some processors are idle at a given time because of conflicts over memory access or communication paths, and inefficient algorithms for properly exploiting the natural parallelism in the computing problems [Mura71, Kuck72, Kuck74, Kuck77, Kuck84].

In the early days of parallel processing, Minsky and Papert [Mins71] provided a depressingly pessimistic form of the speedup factor known as *Minsky's Conjecture*; namely that the speedup factor is equal to the base 2 logarithm of the number of processors used.

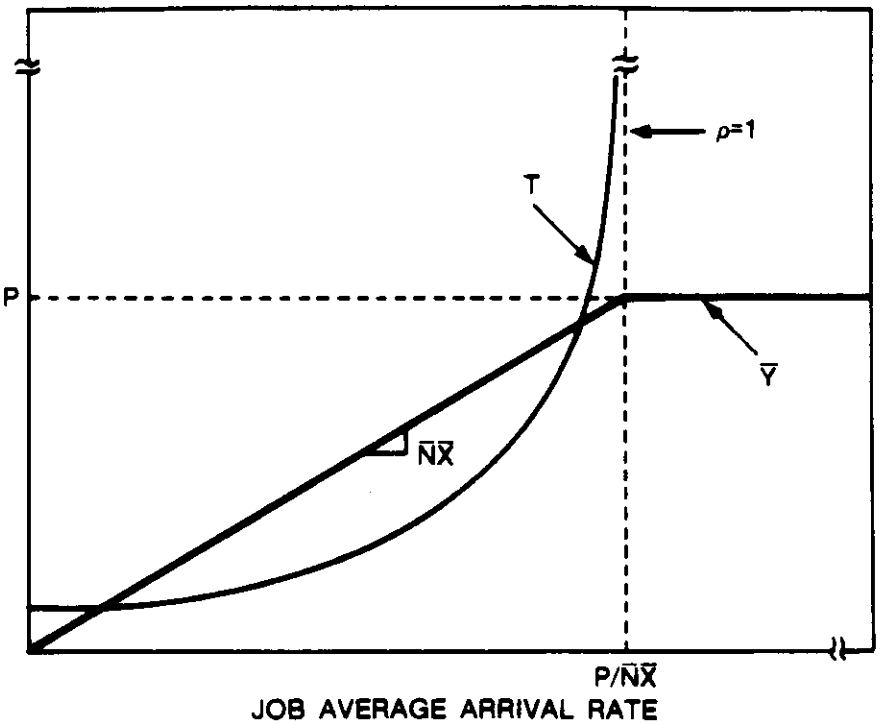


Figure 13: Average System Time and Average Number of Occupied Processors versus λ

Although the expected number of busy processors in a multiprocessor system provides a measure of how much resources can be utilized simultaneously, it does not accurately ascertain how much faster a job can be processed using multiple processors, as opposed to using a single processor. In [Belg86], we properly defined the speedup measure as the ratio of the job average system time using a uniprocessor system to the job average system time using a multiprocessor system; and as a function of the system utilization factor, the number of processors, and the scheduling strategy used. Consequently, we investigated the gain in the job average response time achieved by using a multiprocessing system relative to using a uniprocessor system.

APPENDIX A

Derivation of $\sum_{n=0}^{\infty} \binom{n+k}{k} x^n$

Let the bivariate function $a_k(n)$ be defined as:

$$a_k(n) = \binom{n+k}{k} x^n \qquad \forall k \geq 0\,,\ n \geq 0 \tag{A.1}$$

where the absolute value of x is less than unity. From the above equation, we obtain the following recurrence relation on the bivariate function $a_k(n)$:

$$a_k(n) = \frac{n+k}{k} a_{k-1}(n) \qquad k \geq 1 \tag{A.2}$$

with the following boundary condition for k=0:

$$a_0(n) = x^n \qquad n \geq 0 \tag{A.3}$$

Define the normal generating function of $a_k(n)$ by $A_k(Z)$, $\forall k \geq 0$; that is $A_k(Z) \triangleq \sum_{n=0}^{\infty} a_k(n) Z^n$.

Thus equation (A.2) yields:

$$A_k(Z) = \frac{Z}{k} \frac{d}{dZ} A_{k-1}(Z) + A_{k-1}(Z) \qquad k \geq 1 \tag{A.4}$$

Now, we proceed to show, by induction on the index k, that $A_k(Z) = \left[A_0(Z)\right]^{k+1}$.

Basis step

From equation (A.2), we have $a_0(n) = x^n$, and consequently its generating function $A_0(Z)$ is given by: $A_0(Z) = \frac{1}{1-xZ}$. On the other hand, for the value of k=1, we have $a_1(n) = (n+1)x^n$, and hence its generating function $A_1(Z)$ is given by: $A_1(Z) = \frac{1}{(1-xZ)^2}$. Therefore we readily have: $A_1(Z) = \left[A_0(Z)\right]^2$.

Inductive Step

Now, suppose that we have:

$$A_j(Z) = \left[A_0(Z)\right]^{j+1} \qquad 1 \le j \le k-1$$

and let us show it for the value k. We have:

$$\frac{d}{dZ}A_{k-1}(Z) = xk\left[A_0(Z)\right]^k$$

thus, using equation (A.4), we indeed obtain $A_{k+1}(Z) = \left[A_0(Z)\right]^{k+1}$.

Finally, since $A_0(Z) = \frac{1}{1-xZ}$, and putting Z=1, we obtain:

$$\sum_{n=0}^{\infty} \binom{n+k}{n} = \left[\frac{1}{1-x}\right]^{k+1} \qquad k \ge 0 \qquad \text{(A.5)}$$

∎

APPENDIX B

Derivation of $A = \sum_{i=2}^{N} \frac{1}{2^{N-1}} \sum_{x \ge 1} x \left\{ \sum_{r=i}^{N} \binom{N-x-1}{r-2} \right\}$

To evaluate the quantity A, notice that for any given value of i, i=2,...,N correspond a range of values for x, and a range of values for r. For i=j for example, we have r=j,j+1,...,N, and x=1,2,...,N-j+1. This value of i accounts with the following in the expression of the quantity A:

$$1.\left\{ \binom{N-2}{j-2} + \cdots + \binom{N-2}{N-2} \right\} + 2.\left\{ \binom{N-3}{j-2} + \cdots + \binom{N-3}{N-3} \right\} + \cdots + (N-j+1).\left\{ \binom{j-2}{j-2} \right\}$$

In the above expression, we purposely factored out the values of x. Let A_x denote the participation of the value x in the quantity A. Therefore, for any value of x, x=1,...,N-1, we obtain:

$$A_x = x \left\{ \binom{N-x-1}{0} + 2\binom{N-x-1}{1} + 3\binom{N-x-1}{2} + \cdots + (i+1)\binom{N-x-1}{i} + \cdots + (N-x)\binom{N-x-1}{N-x-1} \right\}$$

Which amounts to:

$$A_x = x \sum_{i=0}^{N-x-1} (i+1) \binom{N-x-1}{i} \qquad x=1,\dots,N \tag{B.1}$$

since x varies from 1 to N-1, it follows that:

$$A = \sum_{x=1}^{N-1} A_x$$

and hence by using equation (B.1), we obtain:

$$A = \frac{1}{2^{N-1}} \sum_{x=1}^{N-1} x \sum_{i=0}^{N-x-1} (i+1) \binom{N-x-1}{i} \tag{B.2}$$

on the other hand, we have:

$$\sum_{i=0}^{N-x-1} (i+1) \begin{bmatrix} N-x-1 \\ i \end{bmatrix} = \sum_{i=0}^{N-x-1} \begin{bmatrix} N-x-1 \\ i \end{bmatrix} + \sum_{i=0}^{N-x-1} i \begin{bmatrix} N-x-1 \\ i \end{bmatrix}$$

and since by using the binomial theorem [Liu68], we have:

$$\sum_{i=0}^{N-x-1} \begin{bmatrix} N-x-1 \\ i \end{bmatrix} = 2^{N-x-1} \quad \text{and,} \quad \sum_{i=0}^{N-x-1} i \begin{bmatrix} N-x-1 \\ i \end{bmatrix} = (N-x-1)\, 2^{N-x-2}$$

therefore equation (B.2) becomes:

$$A = \frac{1}{2^{N-1}} \sum_{x=1}^{N-1} x\,(N-x+1)\, 2^{N-x-2} \tag{B.3}$$

Now, define the quantities A1 and A2 by:

$$\mathrm{A1} = \sum_{x=1}^{N-1} x \left[\frac{1}{2}\right]^{x-1}$$

$$\mathrm{A2} = \sum_{x=1}^{N-1} x\,(x-1) \left[\frac{1}{2}\right]^{x-2}$$

Therefore equation (B.3) becomes:

$$A = \frac{1}{8} \left\{ 2N\mathrm{A1} - \mathrm{A1} \right\} \tag{B.4}$$

now, we proceed to evaluate the expressions of A1 and A2. Let $y = \frac{1}{2}$; we have:

$$\mathrm{A1}(y) = \sum_{x=1}^{N} x\, y^{x-1} = \frac{d}{dy} \sum_{x=1}^{N-1} y^x = \frac{d}{dy} \left[y\, \frac{1-y^{N-1}}{1-y} \right]$$

which amounts then to:

$$\mathrm{A1}(y) = \frac{\left[1-Ny^{N-1}\right](1-y) + y - y^N}{(1-y)^2} \tag{B.5}$$

replacing the dummy variable y by its value $\frac{1}{2}$, we obtain:

$$\mathrm{A1} = 4 - (N+1) \left[\frac{1}{2}\right]^{N-2} \tag{B.6}$$

Now let us evaluate A2; we have:

$$\mathrm{A2}(y) = \sum_{x=1}^{N-1} x\,(x-1)\, y^{x-2} = \frac{d^2}{dy^2} \sum_{x=1}^{N-1} y^x$$

which amounts then to:

$$A2(y) = \frac{N\,(N-1)\,y^{N-2}}{y-1} + \frac{2}{1-y} A1(y) \qquad \text{(B.7)}$$

replacing the dummy variable y by its value $\frac{1}{2}$, we obtain:

$$A2 = 16 - \left\{N^2 + N + 2\right\}\left[\frac{1}{2}\right]^{N-3} \qquad \text{(B.8)}$$

Finally, by using equations (B.4), (B.6) and (B.8), we obtain:

$$A = N - 2 + \left[\frac{1}{2}\right]^{N-1} \qquad \text{(B.9)}$$

IIII

Evaluation of $B = \sum_{x\geq 1} x \left\{\sum_{r=2}^{N} \binom{N-x-1}{r-2}\right\}$

As before, let A_x denote the total participation of the value x, x=1,...,N-1 in the expression of the quantity B. For the value x=i for example, we have:

$$A_i = i\binom{N-i-1}{0} + i\binom{N-i-1}{1} + \cdots + i\binom{N-i-1}{N-i-1} = i2^{N-i-1}$$

it follows then that the expression of B becomes:

$$B = \sum_{x=1}^{N-1} x 2^{N-x-1} = 2^{N-2} \sum_{x=1}^{N-1} x \left[\frac{1}{2}\right]^{x-1} = 2^{N-2}\, A1(y)$$

and by using equation (B.6) and replacing the dummy variable y by its value $\frac{1}{2}$, we get:

$$B = 2^N - (N+1) \qquad \text{(B.10)}$$

IIII

Derivation of $C = \sum_{i=2}^{N} \frac{1}{2^{N-1}} \sum_{x\geq 1} x^2 \left\{\sum_{r=i}^{N} \binom{N-x-1}{r-2}\right\}$

From the evaluation of the quantity A earlier in this Appendix, and by following the same exact steps, it is not hard to see that the quantity C may be written as:

$$C = \frac{1}{2^{N-1}} \sum_{x=1}^{N-1} x^2 (N-x+1)\, 2^{N-x-2} \tag{B.11}$$

Now, define the quantities A3 and A4 by:

$$A3 = \sum_{x=1}^{N-1} x^2 \left[\frac{1}{2}\right]^{x-2}$$

$$A4 = \sum_{x=1}^{N-1} x^3 \left[\frac{1}{2}\right]^{x-3}$$

Therefore equation (B.11) becomes:

$$C = \frac{1}{16} \left\{ 2(N+1)A3 - A4 \right\} \tag{B.12}$$

Now, we proceed to evaluate the expressions of A3 and A4. To evaluate A4, we need first to evaluate the following expression:

$$A5 = \sum_{x=1}^{N-1} x(x-1)(x-2) \left[\frac{1}{2}\right]^{x-3}$$

Let $y = \frac{1}{2}$, the quantity A5 can then be written as:

$$A5(y) = \sum_{x=1}^{N-1} x(x-1)(x-2) y^{x-3} = \frac{d^3}{dy^3} \left\{ \sum_{x=1}^{N-1} y^x \right\} = \frac{d}{dy} A2(y)$$

using the expression of A2(y) as given by equation (B.7), we obtain:

$$A5(y) = \frac{d}{dy} \left\{ \frac{N(N-1)y^{N-2}}{y-1} + \frac{2}{1-y} A1(y) \right\}$$

where the expression of the quantity A1(y) is given by equation (B.5). Define the quantities A6(y) and A7(y) by:

$$A6(y) = \frac{d}{dy} \left\{ \frac{N(N-1)y^{N-2}}{y-1} \right\}$$

$$A7(y) = \frac{d}{dy} \left\{ \frac{2A1(y)}{1-y} \right\}$$

Therefore the quantity A5(y) can be rewritten as:

$$A5(y) = A6(y) + A7(y) \tag{B.13}$$

After derivation and some algebra and by replacing the dummy variable y by its value $\frac{1}{2}$, we obtain:

$$A6 = -N(N-1)^2 \left[\frac{1}{2}\right]^{N-4} \tag{B.14}$$

Now, we proceed to evaluate the quantity A7(y); we have:

$$A7(y) = \frac{d}{dy}\left\{\frac{2A1(y)}{1-y}\right\} = \frac{2}{1-y}\frac{d}{dy}A1(y) + \frac{2}{(1-y)^2}A1(y)$$

Since from the definitions of the quantities A1(y) and A2(y), we have $\frac{d}{dy}A1(y) = A2(y)$, and after some algebra and replacing the dummy variable y by its value $\frac{1}{2}$, we obtain:

$$A7 = 96-(N^2+2N+3)\left[\frac{1}{2}\right]^{N-5} \tag{B.15}$$

Returning now to the expression of the quantity A5, and using equations (B.13), (B.14), and (B.15), we obtain:

$$A5 = 96 - (N^3+5N+6)\left[\frac{1}{2}\right]^{N-4} \tag{B.16}$$

Let us now return to the evaluation of the quantity C. Since:

$$A4 = A5 + 3\sum_{x=1}^{N-1} x^2\left[\frac{1}{2}\right]^{x-3} - 2\sum_{x=1}^{N-1} x\left[\frac{1}{2}\right]^{x-3}$$

using equations (B.12), (B.13), (B.14), (B.15) and (B.16), and after some algebra, we obtain:

$$C = \frac{1}{16}\left\{2(N-2)\sum_{x=1}^{N-1} x(x-1)\left[\frac{1}{2}\right]^{x-2} + 4N\sum_{x=1}^{N-1} x\left[\frac{1}{2}\right]^{x-1} - A5\right\}$$

$$= \frac{1}{16}\left\{2(N-2)A2 + 4NA1 - A5\right\}$$

and therefore by using the expressions of the quantities A1, A2, and A5, which are given respectively by equations (B.6), (B.8), and (B.16), we obtain:

$$C = 3N - 10 + \frac{2N+5}{2^{N-1}} \tag{B.17}$$

IIII

Evaluation of $D = \sum_{x\geq 1} x^2 \left\{ \sum_{r=2}^{N} \begin{bmatrix} N-x-1 \\ r-2 \end{bmatrix} \right\}$

From the evaluation of the quantity B earlier in this Appendix, and by following the same exact steps, it is not hard to see that the quantity D may be written as:

$$D = \sum_{x=1}^{N-1} x^2 2^{N-x-1} = 2^{N-3} \sum_{x=1}^{N-1} x^2 \left[\frac{1}{2}\right]^{x-2}$$

$$= 2^{N-3} \left\{ \sum_{x=1}^{N-1} x(x-1) \left[\frac{1}{2}\right]^{x-2} + 2 \sum_{x=1}^{N-1} x \left[\frac{1}{2}\right]^{x-1} \right\}$$

$$= 2^{N-3} \left\{ \text{A2} + 2\text{A1} \right\}$$

and therefore using the expressions of the quantities A1 and A2 given respectively by equations (B.6) and (B.8), and after some algebra, we obtain:

$$D = 3.2^N - N^2 - 2N - 3 \tag{B.18}$$

■

References

[Belg85] A. Belghith and L. Kleinrock, "Analysis of the Number of Occupied Processors in a Multiprocessing system," *Technical Report No. UCLA-CSD-850027, School of Engineering and Applied Science, Computer Science department, University of California, Los Angeles*, August, 1985.

[Belg86] A. Belghith, "Response Time and Parallelism in Multiprocessing Systems with Certain Synchronization Constraints," *Ph.D. Dissertation, School of Engineering and Applied Science, Computer Science Department, University of California, Los Angeles*, December 1986.

[Klei75] L. Kleinrock, *Queueing Systems, Vol 1: Theory*, New York: Wiley-Interscience, 1975.

[Kuck72] D.J. Kuck, Y. Muraoka, and S.C. Chen, "On the Number of Operations Simultaneously Executable in Fortran-Like Programs and Their Resulting Speedup," *IEEE Transactions on Computers*, Vol. C-21, No. 12, December 1972, pp. 1293-1310.

[Kuck74] D.J. Kuck, P.P. Budnik, S.C. Chen, D.H. Lawrie, R.A. Towle, R.E. Strebendt, E.W. Davis, Jr., J. Han, P.W. Kraska, and Y. Muraoka, "Measurments of Parallism in Ordinary FORTRAN Programs," *IEEE Computer Magazine*, January 1974, pp. 37-45.

[Kuck77] D.J. Kuck, "A Survey of Parallel Machine Organization and Programming," *ACM Computing Surveys*, Vol. 9, No. 1, March 1977, pp. 29-59.

[Kuck84] D.J. Kuck and et al.,, "The Effects of Program Restructuring, Algorithm Change and Architecture Choice on Programs," *Proceeding of the International Conference on Parallel Processing*, 1984.

[Litt61] J. Little, "A Proof of the Queueing Formula $L = \lambda W$," *Operations Research*, Vol. 9, No. 2, March 1961, pp. 383-387.

[Liu68] C.L. Liu, *Introduction to Combinatorial Mathematics*, New York: McGraw Hill, 1968.

[Mins71] M. Minsky and S. Papert, *On Some Associative, Parallel and Analog Computations*: Associative Information Technologies, (Elsevier North Holland, New York), 1971.

[Mura71] Y. Muraoka, ''Parallelism Exposure and Exploitation in Programs,'' *Ph.D. Dissertation Technical Report No. 71-424, Department of Computer Science, University of Illinois at Urbana-Champaign*, February 1971.