*Computers Elect. Engng* Vol. 19, No. 6, pp. 495–506, 1993
Printed in Great Britain. All rights reserved

0045-7906/93 $6.00 + 0.00
Copyright © 1993 Pergamon Press Ltd

# MODELLING AND ANALYSIS OF THE VARIANCE IN PARALLELISM IN PARALLEL COMPUTATIONS

C. R. M. SUNDARAM† and Y. NARAHARI
Department of Computer Science and Automation, Indian Institute of Science, Bangalore-560 012, India

*(Accepted in final revised form 6 January 1993)*

**Abstract**—In this paper, we introduce an analytical technique based on queueing networks and Petri nets for making a performance analysis of dataflow computations when executed on the Manchester machine. This technique is also applicable for the analysis of parallel computations on multiprocessors. We characterize the parallelism in dataflow computations through a four-parameter characterization, namely, the minimum parallelism, the maximum parallelism, the average parallelism and the variance in parallelism. We observe through detailed investigation of our analytical models that the average parallelism is a good characterization of the dataflow computations only as long as the variance in parallelism is small. However, significant difference in performance measures will result when the variance in parallelism is comparable to or higher than the average parallelism.

*Key Words:* Variance in parallelism, average parallelism, dataflow computations, parallel computations, product form queueing networks, generalized stochastic Petri nets.

## 1. INTRODUCTION

A number of small to medium scale multiprocessing architectures are not in wide use. However, the high cost of building such architectures can be justified only if they are efficiently utilized. Exploiting the parallelism in applications is a viable and effective approach to improve the performance of such systems. For exploiting the parallelism, it is necessary to characterize the parallelism in different applications through some common parameters. Parallelism in applications can be characterized at different levels of detail ranging from the full data dependency graph at one extreme to a single parameter such as the average parallelism, the variance in parallelism at the other extreme. Detailed characterizations will not only be impossible to get due to their possible dependency on data input and partitioning schemes, but also will be difficult to manage in an effective way. So, simple and effective characterizations are needed for this purpose. Such characterizations will be useful for two reasons. First, it will give the user an insight that is otherwise not clear, into the structure of his algorithm, and will facilitate the user in improving the efficiency of his algorithm. Second, these characterizations which are simple and independent of different applications and their structures will help in arriving at efficient resource management strategies and will also give the ability to treat the different applications in a uniform way. Moreover, the overhead due to resource scheduling will be very minimal as the amount of information that has to be dealt by the scheduler is very small.

We characterize the parallelism in applications by a four-parameter characterization, namely minimum parallelism, maximum parallelism, average parallelism and the variance in parallelism. Similar characterizations to ours have been considered [1–4]. Sevcik [1] analyzed the effects of the variance in parallelism through simulation. But simulation models are computationally expensive, and for an approximate and quick feedback, the analytical models are the preferred tools. Jiang *et al.* [2] consider the variation in parallelism (the difference between maximum and minimum parallelism). They make an approximate analytical study by assuming that the instantaneous parallelism of the applications remains constant over sufficiently long intervals of time, before it changes to another value. They analyze the effects of variation in parallelism by repeatedly evaluating an approximate analytical model for different degrees of parallelism and combine the results using a weighted approach. Eager *et al.* [3] consider a single-parameter

†Present affiliation: Department of Computational Science, University of Saskatchewan, Saskatoon, Canada S7N 0W0.

characterization, namely average parallelism, and obtain bounds on speedup, efficiency and knee of the service time–efficiency profile in terms of the average parallelism and the number of processors. Knee of the service time–efficiency profile is defined as the number of processors at which the ratio of the efficiency to the service time with those many processors is minimized. Majumdar *et al.* [4] consider one parameter namely the variability in parallelism (different from the variance in parallelism and the variation in parallelism) in addition to the average parallelism and obtain tighter bounds on speedup, efficiency, etc. than the bounds obtained just using the average parallelism.

In this paper, we are interested in performance evaluation of parallel algorithms on multiprocessing and dataflow architectures. We introduce an analytical technique based on product form queueing networks (PFQNs) and generalized stochastic Petri nets (GSPNs), to carry out the above evaluation. We illustrate our technique for the performance evaluation of data flow computations when executed on the Manchester dataflow architecture. Our technique is quite general, and it could be applied to any multiprocessing architecture, since the performance analysis of multiprocessing and dataflow architectures can be made in a unified way [2,5]. That is, the same models could be used with different interpretations. We also compare our analytical technique with another technique based on a single parameter characterization namely the average parallelism in applications, considered by Ghosal and Bhuyan [5–7]. Their models are closed queueing network (CQN) models, where the closed nature is ensured by assuming that the degree of parallelism exhibited by the dataflow graphs can be taken to be a constant, equal to the average parallelism.

Through extensive numerical experiments of our models, we find that: (i) average parallelism is a good characterization of the data flow computations only as long as the variance in parallelism is small compared to it, and a significant difference in performance will result if the variance in parallelism is comparable to the average parallelism; and (ii) the model proposed here is remarkably accurate.

The rest of the paper is organized as follows. In Section 2, we present the basic definitions and notations used in this paper. In Section 3, we represent the architecture of a prototype of the Manchester machine, then illustrate the importance of the variance in parallelism in dataflow computations through two simple examples. Then, we develop detailed GSPN models for the execution of the dataflow computations on the Manchester machine and point out their state space explosion. In Section 4, we give a brief introduction to the modelling tool considered in this paper. Then, we develop integrated queueing network-Petri net models and compare their efficiency with exact GSPN models. In Section 5, we show the results of the detailed numerical experiments carried out on our models and compare them with CQN models proposed by Ghosal and Bhuyan. In Section 6, we present the conclusions.

## 2. BASIC DEFINITIONS AND NOTATIONS

The average parallelism of a job denoted by $A$ is defined as the average number of busy processors when an unlimited number of them are allocated for the execution of the job. Putting in mathematical terms

$$A = \sum_{j=m}^{j=M} j \times p_j$$

where $p_j$, $m$ and $M$ are the fraction of time $j$ processors are simultaneously busy, the minimum number of processors that are always busy, the maximum number of processors that are simultaneously busy, respectively, when an unlimited number of processors are available. Average parallelism can also be equivalently defined as $A = T_1/T_\infty$, where $T_1$ and $T_\infty$ are the total execution times of the job on a single processor and on an unlimited number of processors. The variance in parallelism denoted by $V$ is defined as:

$$V = \sum_{j=m}^{j=M} (j - A)^2 \times p_j.$$

The fraction of the sequential portion of the job denoted by $f$ can be computed using:

$$f = \frac{T_\infty \times p_1}{T_1} = \frac{p_1}{A}.$$

The variability in parallelism denoted by $\omega(A)$ is defined as the ratio between the service times with $A$ processors and when the number of processors is unbounded. It can be computed using $\omega(A) = A/S(A)$. The variation in parallelism of a job denoted by $V_1$ is defined as the difference between the minimum and the maximum parallelisms of the job, i.e. $V_1 = M - m$.

## 3. THE VARIANCE IN PARALLELISM IN DATAFLOW COMPUTATIONS

The closed queuing network models presented by Ghosal and Bhuyan for the execution of the dataflow graphs on the Manchester machine assume that the degree of parallelism in the dataflow computations takes a constant value, namely $A$, the average parallelism. However, in an actual execution, the degree of parallelism varies with time, and this variation is found to have a significant impact on their performance in this study. In this section, we illustrate the importance of the variation in parallelism in dataflow computations through two simple examples.

First, we give below a brief overview of the Manchester machine. A prototype of the Manchester dataflow system is shown in Fig. 1. An I/O switch acts as an interface between the host unit and the ring. The tokens from the host contain the values to initialize the arcs of the underlying dataflow graph. The tokens from the ring contain both final and intermediate results. The final results are sent to the host whereas the intermediate results are sent to the token queue. The token queue is a FIFO buffer. Its main function is to smooth the flow of tokens in the ring structure. The tokens from the token queue are sent to the matching unit. Its main function is to match the incoming tokens from their partner tokens. Unmatched partner tokens are written into the overflow unit. Matched tokens are sent to the node store unit whose function is to form the executable packet containing the instruction and the operands. The processing unit contains a number of functional units. Executable packets formed by the node store unit is executed in one of the functional units. The functional units generate result tokens which are sent to the I/O switch for further processing.

The number of active instructions at any time in an execution of a dataflow graph is the degree of parallelism at that time instant. Consider the dataflow graphs shown in Figs 2 and 3. Figure 2 shows the dataflow graph for the evaluation of the expression

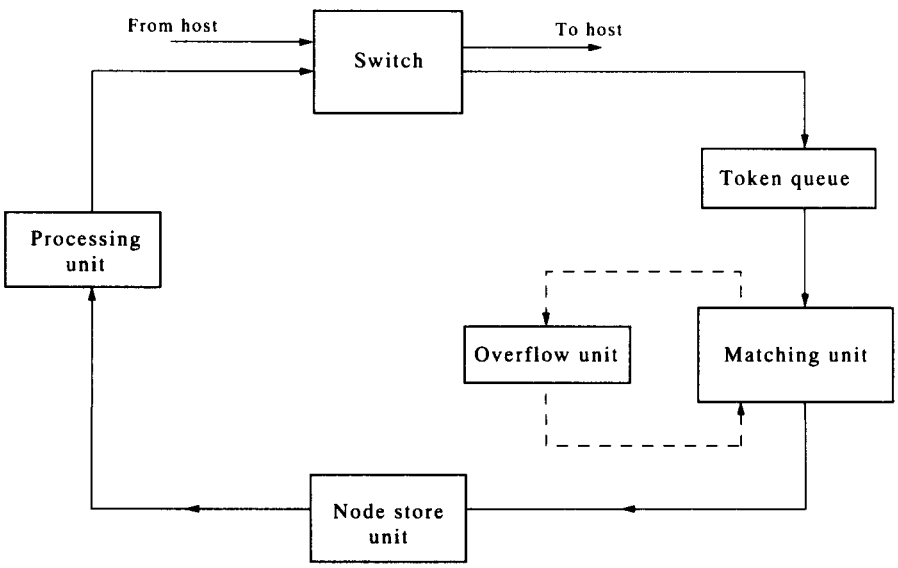



Fig. 1. The Manchester dataflow computer.

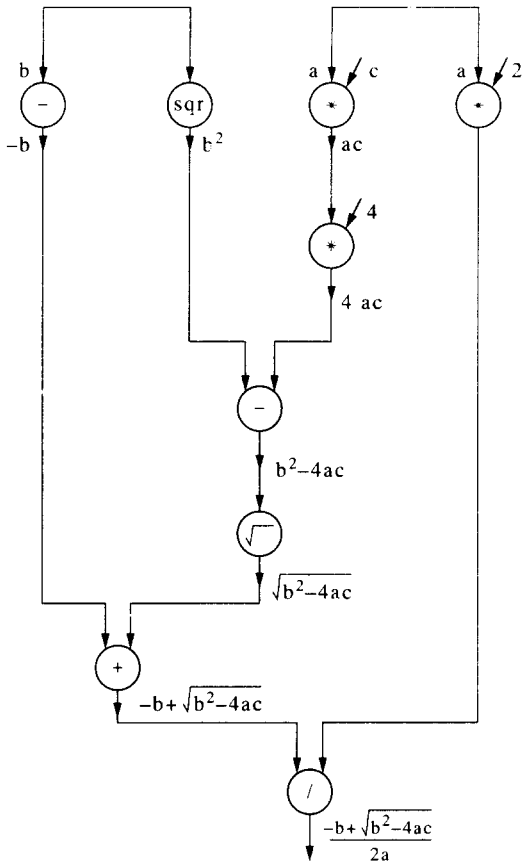



Fig. 2. Example dataflow Graph 1.

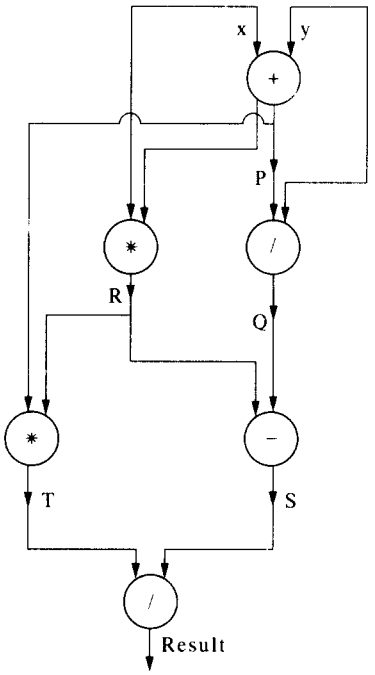



Fig. 3. Example dataflow Graph 2.

$(-b + \sqrt{b^2 - 4 \times a \times c})/2 \times a$, whereas Fig. 3 shows the dataflow graph for the evaluation of the following program segment:

$$P = X + Y,$$
$$Q = \frac{P}{Y}$$
$$R = X \times P,$$
$$S = R - Q,$$
$$T = R \times P,$$
$$\text{Result} = \frac{S}{T}.$$

If such dataflow graphs are executed on the Manchester machine even if all the instructions get executed in more or less the same time, the degree of parallelism varies due to the structure of the computation. The degree of parallelism can also vary due to the difference in the execution time of instructions and the availability of functional units. Moreover, this variation in parallelism with time can occur not only during the execution of a single dataflow graph, but also due to the presence of dissimilar dataflow graphs in the job queue of Fig. 1.

Now, we develop a GSPN model which incorporates this variation in parallelism. The GSPN model presented below is based on the following assumptions: (i) the service times at all the service centers, namely, matching unit, node store unit and processing unit are exponentially distributed; (ii) the matching store is large enough so that the overflow unit is not required; (iii) the degree of parallelism in a dataflow graph can be characterized by a four-parameter characterization namely, the minimum, maximum, average and the variance in parallelism; and (iv) the degree of parallelism changes in steps of 1 only, i.e. it either increases by 1, or decreases by 1. This assumption is justified since the execution time of functional units is exponentially distributed and the

probability of two or more activities with exponentially distrubuted durations ending simultaneously is zero.

Figure 4 shows the GSPN model for the execution of dataflow graphs on the Manchester machine. The reader is referred to Ref. [8] for GSPN details. The interpretation of the places, and the transitions is given in Table 1. Figure 4 becomes self-explanatory with Table 1. The random switch $t_8$, $t_9$, $t_{10}$ models the variance in parallelism. The firing of the transition $t_9$ increases the degree of parallelism by 1, and the firing of $t_{10}$ decreases the degree of parallelism by 1. A change in parallelism of $+1$ is due to the processing unit generating two result tokens and a change in parallelism of $-1$ is due to the processing unit generating no tokens. In these GSPN models, the loss of tokens is not necessarily equal to the gain of tokens, which makes the parallelism of the underlying dataflow graph vary. The degree of parallelism $d$ in a marking $M$ of the GSPN model is given by:

$$d(M) = \sum_{i=1, i\neq 4, i\neq 8}^{10} M(P_i).$$

Consider the execution of a dataflow graph having an average parallelism $A$, a minimum parallelism $A - V_1$, and a maximum parallelism $A + V_1$. Here, $V_1$ is the variation in parallelism from the average parallelism. For modelling execution of such a dataflow graph, the place $p_1$ is initialized with $A$ tokens. The probability $q_1$ is set to zero when $d$ equals $A + V_1$. This restricts the increase of parallelism above $A + V_1$. The probability $q_2$ is set to zero when $d$ equals $A - V_1$. This

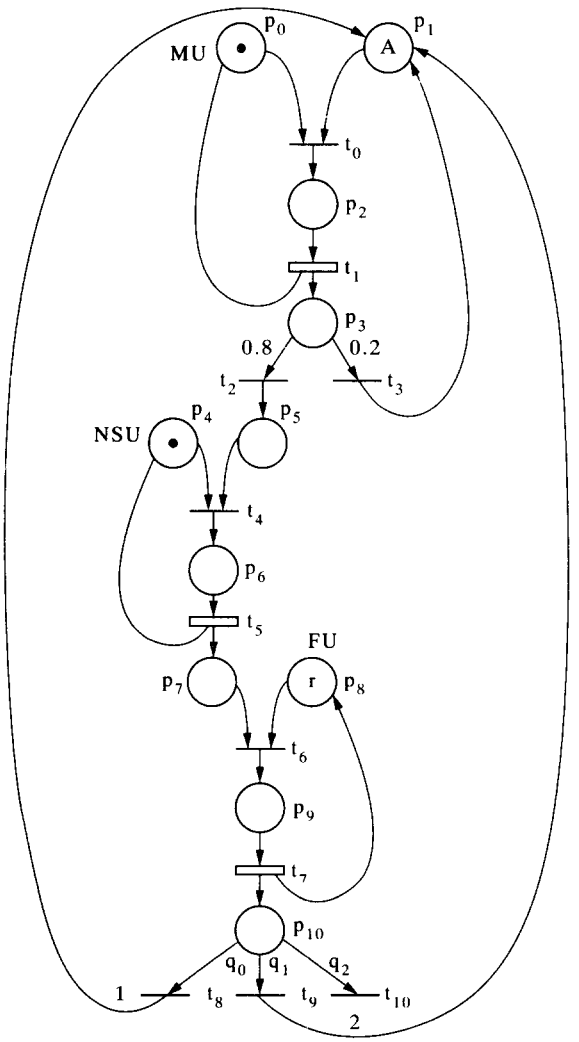



Fig. 4. GSPN model of the prototype incorporating time variance of parallelism.

Table 1. Legend for GSPN model in Fig. 4

**Places:**

$P_0, P_4, P_8$: availability of MU, NSU and PU
$P_1, P_5, P_7$: Tokens waiting for MU, NSU and PU
$P_2, P_6, P_9$: Tokens getting executed in MU, NSU and PU
$P_3, P_{10}$: Matching and processing operations have just finished

**Immediate transitions:**

$t_0, t_4, t_6$: starting of the operations of MU, NSU and PU
$t_2$: Matching operation is successful
$t_3$: Matching operation is unsuccessful
$t_8, t_9, t_{10}$: Processing unit generates no tokens, exactly one token and two tokens

**Exponential transitions:**

$t_1, t_5, t_7$: Execution of tokens by MU, NSU and PU

**Random switch: dynamic**

$t_8 - q_0; t_9 - q_1; t_{10} - q_2$:

**Initial marking:**

$P_0 - 1; P_1 - A; P_4 - 1; P_8 - r$; where $r$ is the number of functional units

does not allow the decrease of parallelism below $A - V_1$. Thus the degree of parallelism varies between $A - V_1$ and $A + V_1$. More formally:

$$q_1(M) = 0 \quad \text{if } d(M) = A + V_1.$$

$$q_2(M) = 0 \quad \text{if } d(M) = A - V_1.$$

When $d$, the degree of parallelism, is between $A - V_1$ and $A + V_1$, the probabilities $q_1$ and $q_2$ take positive values. The values taken by them depend on $V_1$ due to the following reason. The sojourn times are very low in the markings having a degree of parallelism close to $A + V_1$ (because of the high throughputs in these markings). Similarly, the sojourn times are very high in the markings having a degree of parallelism close to $A - V_1$ (because of the low throughputs in these markings). So, this makes the systems spend more time in the states with a low degree of parallelism, than the states with a high degree of parallelism, and hence the average parallelism decreases with an increase in the variation in parallelism. So, we adjust the probabilities $q_0$, $q_1$ and $q_2$ so that the average parallelism $A$ remains the same for all variations. For an average parallelism 10, Table 2 shows the probabilities $q_0$, $q_1$ and $q_2$ so that the average parallelism $A$ remains the same for all the variations in parallelism. The variance in parallelism $V$ in a marking $M$ of the GSPN model can be calculated by:

$$V = \sum_{M \in S} [d(M) - A]^2 \times \text{SSP}(M),$$

where SSP($M$) is the steady state probability of marking $M$, $S$ is the reachable set of the GSPN model and $d(M)$ is the degree of parallelism in the marking $M$. The values of $V$ are also shown in Table 2 for different values of $V_1$. As expected, the variance in parallelism increases with an increase in $V_1$.

The cardinality of the state space of the exact GSPN model for the execution of a dataflow graph with an average parallelism $A$ and a variation $V_1$ is given by:

$$\sum_{i=A-V_1}^{A+V_1} \frac{(i+2)!}{i!2!}.$$

Table 2. The values of the probabilities for different values of the variations in parallelism

| $V_1$ (Variation in parallelism) | $q_1$ (Probability that parallelism increases) | $q_2$ (Probability that parallelism decreases) | $A$ (Average parallelism) | $V$ (Variance in parallelism) |
|---|---|---|---|---|
| $10 \pm 7$ | 0.2543 | 0.2457 | 10.00 | 19.2958 |
| $10 \pm 6$ | 0.2530 | 0.2470 | 9.99 | 13.8154 |
| $10 \pm 5$ | 0.2521 | 0.2479 | 10.00 | 9.5059 |
| $10 \pm 4$ | 0.2514 | 0.2486 | 10.00 | 6.1745 |
| $10 \pm 3$ | 0.2509 | 0.2491 | 9.99 | 3.6245 |
| $10 \pm 2$ | 0.2505 | 0.2495 | 9.99 | 1.7782 |
| $10 \pm 1$ | 0.2500 | 0.2500 | 9.99 | 0.6000 |

Average parallelism = 10. No. of functional units = 5.

As explained, in any marking $M$, the degree of parallelism $d(M)$ varies between $A - V_1$ and $A + V_1$. When the degree of parallelism takes a value of $i$, the $i$ tokens can be distributed across the queues of three service centers of the Manchester machine in $(i + 2)!/i!2!$ ways. Then, the total number of states is the sum over the number of states having a parallelism of $i$, where $i$ varies from the minimum parallelism $(A - V_1)$ to the maximum parallelism $(A + V_1)$.

As the expression shows, studying the effects of the variance in parallelism in dataflow computations through exact GSPN models will not be practically feasible, as the number of states grows exponentially with an increase in the average parallelism and the variation in parallelism.

## 4. INTEGRATED QUEUEING NETWORK–PETRI NET (IQP) MODELS FOR THE VARIANCE IN PARALLELISM

The IQP modelling originated by Balbo *et al.* [9,10] is based on the concept of flow-equivalence. The basic idea is to isolate the product form and non-product form portions of a given system, evaluate them independently and combine them through flow-equivalents. The overall model (also called the high-level model) of a given system will then have some subsystems of it represented by the flow-equivalents. These flow-equivalents of the subsystems (also called the submodels) are computed by evaluating the throughput of these subsystems in isolation for all possible populations. The throughputs thus computed together with associated populations constitute the flow-equivalent. The IQP modelling can be of two types: (i) IQP model with PFQN as the high-level model; and (ii) IQP model with GSPN as the high-level model. If the high-level model is a PFQN model, then some of its stations will be the flow-equivalent representations (derived using GSPNs) of the submodels that incorporate non-product form features. If the high-level model is a GSPN model, then some of its timed transitions will be the flow-equivalent representations (derived using PFQNs) of the submodels that are product form solvable. Since exact representations can be derived for the flow-equivalents, the only source of error in this technique could be due to the interaction between the product form and the non-product form portions. When the interaction is weak, this technique will produce remarkably accurate results.

The development of IQP models for the execution of dataflow computations on the Manchester machine are motivated by the following two observations: (i) the non-product form feature, namely the variation in parallelism cannot be analyzed through product form queueing networks. GSPNs can elegantly model the variation in parallelism. But, analyzing the GSPN models is practically infeasible as the cardinality of the models grows exponentially; and (ii) the service rates of the matching and node store units are 10 times more than the processing capacity of the functional units. Consequently, between two interactions by the processing unit with the rest of the system, the subsystem containing the node store unit and the matching unit would have reached a steady state. In such a case, the interaction is weak, and the integrated technique would give much less error.

In these models the non-product form feature, namely the variation in parallelism, is captured by a GSPN model and the rest of the product form features by a PFQN model. The two models interact through a flow-equivalent server. The GSPN part is the high-level model and the PFQN part is the submodel. The PFQN part is represented in the GSPN part by a marking-dependent timed transition. Figure 5a shows the high-level GSPN model. Figure 5b shows the PFQN part of the model. As in the exact GSPN model of Fig. 4, the random switch $t_2$, $t_3$, $t_4$ captures the variation in parallelism. In the GSPN high-level model, the sub-PFQN model is represented by the marking-dependent timed transition "$t_5$". The firing rate of this transition will vary from marking to marking. The rate of the transition, "$t_5$" in a marking $M$ is obtained by solving the PFQN model of Fig. 5(b) with a population of $M$ $(P_4)$ customers. As the number of tokens in the place "$P_4$" can vary from $A - V_1$ to $A + V_1$, $2 \times V_1$ evaluations of the PFQN part will be required. These $2 \times V_1$ evaluations together with the associated populations constitute the flow-equivalent. Along the same lines as the exact GSPN model, the cardinality of the GSPN high-level model for execution of a dataflow graph with an average parallelism, and a variation $V_1$ is given by:

$$\sum_{A-V_1}^{A+V_1} (i + 1).$$

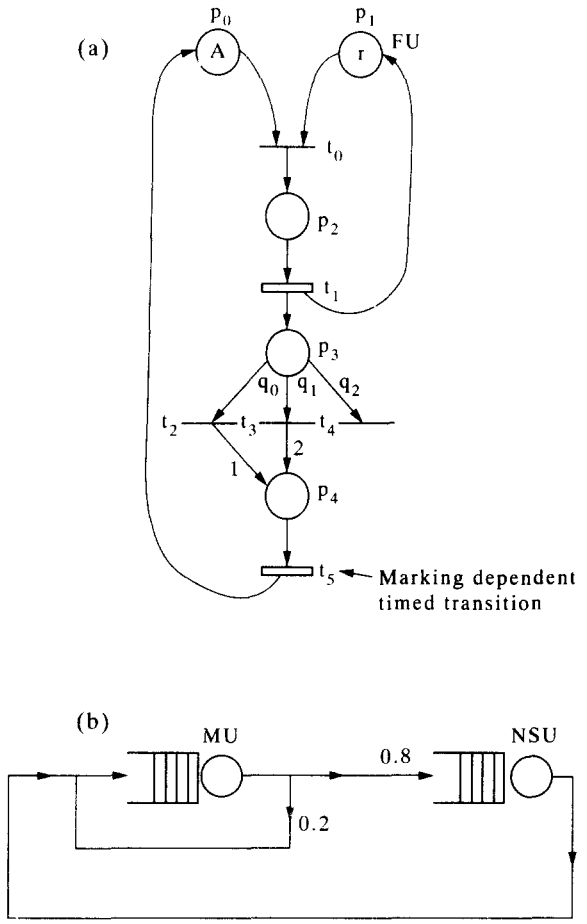



Fig. 5. (a) GSPN part of the integrated model. (b) PFQN parts of the integrated model.

Comparing this expression with that for the exact GSPN model, one can easily observe that there will be a drastic reduction in the size of the state space. To see the computational advantage obtained, we show in Table 3 a comparison of the cardinalities of the exact GSPN models and the integrated analytical models for a value of 10 for $A$ and values of the variation in parallelism, $V_1$ ranging from 1 to 9. Effectively we have reduced an evaluation of a GSPN model into $2 \times V_1$ evaluations of a 2-node PFQN, for whose performance measures a closed form expression can be obtained, followed by an evaluation of a GSPN whose cardinality of the state space is much smaller than that of the exact GSPN model. Now, it would be very easy to study the effect of the variation in parallelism as the experimentation is required on a small-sized GSPN model.

Table 3. A comparison of the cardinalities of the exact GSPN model and the integrated analytical model

| Variation in parallelism | Variance in parallelism | Cardinality of exact GSPN model | Cardinality of integrated analytical model |
|---|---|---|---|
| $10 \pm 1$ | 0.600 | 199 | 33 |
| $10 \pm 2$ | 1.779 | 335 | 55 |
| $10 \pm 3$ | 3.627 | 476 | 77 |
| $10 \pm 4$ | 6.179 | 624 | 99 |
| $10 \pm 5$ | 9.513 | 781 | 121 |
| $10 \pm 6$ | 13.825 | 949 | 143 |
| $10 \pm 7$ | 19.309 | 1130 | 165 |
| $10 \pm 8$ | 27.564 | 1326 | 187 |
| $10 \pm 9$ | 36.057 | 1539 | 209 |

Average parallelism = 10.

Table 4. Processing power of the Manchester machine

| Variation in parallelism | Variance in parallelism | Average parallelism model | Variance model (integrated analytical model) | %difference in processing power |
|---|---|---|---|---|
| $20 \pm 3$ | 3.6173 | 8.774 | 8.743 | 0.353 |
| $20 \pm 7$ | 17.747 | 8.774 | 8.659 | 1.311 |
| $20 \pm 11$ | 43.666 | 8.774 | 8.442 | 3.784 |
| $20 \pm 15$ | 85.273 | 8.774 | 8.025 | 8.537 |
| $20 \pm 19$ | 160.672 | 8.774 | 7.190 | 18.053 |

Average parallelism = 20; No. functional units = 10.

## 5. NUMERICAL RESULTS

In this section, we study the effects of the variance in parallelism in dataflow computations. For this purpose, we evaluated the integrated queueing network–Petri net models for values of 5, 10, and 20 for $A$. In each case $V_1$, the variaiton in parallelism, is varied over a wide range of values.

### *5.1. Processing power*

Processing power of the Manchester machine (i.e. the average number of busy functional units) is an important performance measure. Table 4 gives the processing power of the machine for $A = 20$ and various values of $V_1$ ranging from 3 to 19. The variance in parallelism $V$ is also presented for various values of $V_1$. The third entry in the table is obtained by the evaluation of CQN models proposed by Ghosal and Bhuyan. We observe from this table that there is close agreement in the processing power when the variance in parallelism is small. However, there is a significant decrease in the processing power when the value of $V$ increases. The decrease in the processing power is due to the following reason: the degree of parallelism of the underlying dataflow graph varies from $A - V_1$ to $A + V_1$. The utilization of the functional units drops rapidly when the system visits states having a degree of parallelism close to $A - V_1$, compared to the increase obtained when the system visits the states having a degree of parallelism close to $A + V_1$. This occurs because, as the degree of parallelism increases, the matching unit and node store unit become bottleneck centers and the utilization of functional units more or less saturates. But, the probability that the system visits a state having a degree of parallelism $A - k$, $1 \leqslant k \leqslant V_1$ is same as that for the state $A + k$, $1 \leqslant k \leqslant V_1$.

Similar results are shown in Table 5, which shows processing power for a value of 5 for $A$, and $V_1$ from 1 to 5. But this table shows a much more significant decrease than the previous table, since the average parallelism itself has a low value. For example, when $V_1 = 4$, the variance in parallelism is approx. 1.45 times the value of average parallelism. Even for such a low ratio of the variance in parallelism to average parallelism, we get 16.3% difference in the processing powers predicted by the average parallelism model and our model.

Table 6 shows the processing power of the Manchester machine for a value of 10 for $A$ and various values of $V_1$ from 1 to 7, and for various numbers of functional units. The decrease in the processing power is more pronounced when the number of functional units is high, than the case when the number of functional units is low. This is because, for a given value of $A$ and variation $V_1$, when the system visits states having a degree of parallelism close to $A + V_1$, the increase in the processing power will be more or less the same, in cases when the number of functional units is high as well as when it is low. However, when the system visits states having a degree of parallelism close to $A - V_1$, the decrease in the processing power is more pronounced when the number of functional units is high than when it is low. But, the states having a parallelism above

Table 5. Processing power of the Manchester machine

| Variation in parallelism | Variance in parallelism | Average parallelism model | Variance model (integrated analytical model) | %difference in processing power |
|---|---|---|---|---|
| $5 \pm 1$ | 0.6005 | 2.92 | 2.894 | 0.891 |
| $5 \pm 2$ | 1.8067 | 2.92 | 2.830 | 3.082 |
| $5 \pm 3$ | 3.9205 | 2.92 | 2.675 | 8.391 |
| $5 \pm 4$ | 7.3557 | 2.92 | 2.445 | 16.268 |

Average parallelism = 5. Number of functional units = 3.

Table 6. Processing power of the Manchester machine

| No. of functional units | $10 \pm 1$ | $10 \pm 2$ | $10 \pm 4$ | $10 \pm 5$ | $10 \pm 6$ | $10 \pm 7$ |
|---|---|---|---|---|---|---|
| 1 | 0.999 | 0.999 | 0.999 | 0.999 | 0.999 | 0.999 |
| 3 | 2.999 | 2.999 | 2.995 | 2.989 | 2.975 | 2.940 |
| 5 | 4.922 | 4.905 | 4.828 | 4.749 | 4.611 | 4.400 |
| 7 | 6.252 | 6.187 | 5.949 | 5.755 | 5.505 | 5.176 |
| 9 | 6.724 | 6.617 | 6.314 | 6.097 | 5.820 | 5.458 |

Average parallelism = 10.

$A$ are visited with the same probabilities as those below $A$. So, on the whole, there is a greater decrease in the processing power when the number of functional units is high than when it is low.

We summarize the results presented above into the following observations: (i) for a given number of functional units, and a given average parallelism, there will be a decrease in the processing power as the variance in parallelism increases; (ii) for a given ratio of variance in parallelism to the average parallelism, and a given number of functional units, the decrease in the processing power will be more when the average parallelism is high than it is low; and (iii) for a given average parallelism and a given variance in parallelism, the decrease in processing power will be more when the number of functional units is high than when it is low.

So, there will be a very high decrease in the processing power when the average parallelism is low, the ratio of the variance in parallelism to the average parallelism is high, and the number of functional units is also high. This decrease will be much more than 18.1% which is the percentage of difference in the processing powers predicted by the average parallelism model and our model, for $A = 20$, $V = 160.7$ and 10 functional units (Table 4).

### *5.2. Mean queue lengths*

In this section, we study the effects of the variance in parallelism on the mean queue lengths at various service centers of the Manchester machine. Table 7 shows the change in the mean queue lengths at the Matching unit, the node store unit and the processing units for $A = 5$, and various values of $V_1$ and 5 functional units.

The variation in the mean queue length of the processing unit is due to the following reason. When $V_1$ takes values 1,2 or 3, the decrease in the mean queue length when the system visits states having a degree of parallelism 9,8 or 7, is more than the increase obtained when the system visits having a degree of parallelism of 11, 12 or 13. So, on the whole, there is a net decrease in the mean queue length. However, when $V_1$ takes values 4, 5, 6 or 7, there is no decrease in the mean queue length due to the system visiting states having a degree of parallelism 3, 4, 5 or 6, as there are 5

Table 7. Mean queue lengths of the PU, MU and NSU of the Manchester machine

| Variation in parallelism | Processing unit mean queue length | Matching unit mean queue length | Node store unit mean queue length |
|---|---|---|---|
| 0 | 3.25 | 0.514 | 0.335 |
| 1 | 3.09 | 0.661 | 0.364 |
| 2 | 3.04 | 0.711 | 0.378 |
| 3 | 3.05 | 0.723 | 0.380 |
| 4 | 3.08 | 0.715 | 0.373 |
| 5 | 3.12 | 0.694 | 0.360 |
| 6 | 3.16 | 0.662 | 0.342 |
| 7 | 3.17 | 0.619 | 0.318 |

Average parallelism = 10. Number of functional units = 5.

Table 8. A comparison of the values of the average parallelism and the variance in parallelism computed by the exact GSPN models and the integrated analytical models

| Variation in parallelism | Average parallelism (exact GSPN model) | Average parallelism (integrated analytical model | Variance in parallelism (exact GSPN model) | Variance in parallelism (integrated analytical model |
|---|---|---|---|---|
| $10 \pm 1$ | 9.999 | 9.999 | 0.600 | 0.600 |
| $10 \pm 2$ | 9.999 | 10.000 | 1.779 | 1.778 |
| $10 \pm 3$ | 10.000 | 9.999 | 3.627 | 3.624 |
| $10 \pm 4$ | 10.000 | 9.998 | 6.179 | 6.174 |
| $10 \pm 5$ | 10.000 | 9.996 | 9.513 | 9.506 |
| $10 \pm 6$ | 9.999 | 9.992 | 13.825 | 13.815 |

functional units. But, there is an increase in the mean queue length due to the system visiting states having a degree of parallelism 14, 15, 16 and 17. So, there is a net increase in the mean queue length when $V_1$ takes values above 3.

The mean queue lengths for the matching and node store units exhibit a reverse variation. This is because, the mean queue lengths in the matching unit, the node store unit, are complementary in nature. When the mean queue length in the processing unit increases, the matching and node store units exhibit a decrease and vice-versa.

### *5.3. Accuracy of results*

Since we have used IQP models for all our experiments, we investigated the accuracy of this technique. Table 8 shows the average parallelism and the exact GSPN models at various levels of variation in parallelism. Since there is such a close agreement in the values of the average parallelism and the variance in parallelism between these two models, all the performance measures such as processing power, throughput, mean queue lengths at various service centres computed by the integrated models are found to be very accurate.

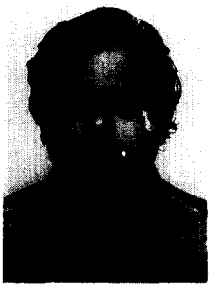
## 6. CONCLUSIONS

In this paper, we have made a performance analysis of dataflow computations when executed on the Manchester machine. The parallelism in dataflow computations has been characterized using a four-parameter characterization. An analytical technique based on queueing networks and Petri nets has been introduced to carry out the performance analysis. This technique is applicable for the performance analysis of parallel computations on any general multiprocessor architecture, as the same models could be used with different interpretations in the case of multiprocessors. By comparing the complexity of analysis with that of exact GSPN models, we have found it to be very efficient. The performance measures computed using the integrated models were also found to be very accurate, and their interpretations presented in the previous section, we conclude that the variance in parallelism affects the performance measures such as the processing power, mean queue lengths, when it is comparable to or higher than the average parallelism. We believe that this observation is also true for parallel computations. Even though we have conducted experiments for a small set of values, our interpretations of the results indicate that similar behaviour will be exhibited at all levels of the average parallelism. These investigations on the effects of the variance in parallelism in the dataflow computations on their performance have been possible, because the integrated models provided us with efficient solutions having an accuracy comparable with that of the exact GSPN modelling approach.

## REFERENCES

1. K. C. Sevcik, Characterizations of parallelism in applications and their use in scheduling. *Perform. Eval. Rev.* **17,** 171–180 (1989).
2. H. Jiang, L. N. Bhuyan and D. Ghosal, Approximate analysis of multiprocessing task graphs. *Proc. Nineteenth Parallel Processing Conf.* (1990).
3. D. L. Eager, J. Zahorjan and E. D. Lazowska, Speedup versus efficiency in parallel systems. *IEEE Trans. Comput.* **38,** 408–423 (1989).
4. S. Majumdar, D. Eager and R. B. Bunt, Characterisation of programs for scheduling in multiprogrammed parallel systems. *Perform. Eval.* (1993).
5. D. Ghosal and L. N. Bhuyan, Performance analysis of the MIT tagged token dataflow architecture. *Proc. Seventeenth Int. Conf. on Parallel Processing*, pp. 680–682 (1988).
6. D. Ghosal and L. N. Bhuyan, Analytical modelling and architectural modifications of a dataflow architecture. *Proc. Fourteenth Annual Symp. on Computer Architecture*, pp. 81–89 (1987).
7. D. Ghosal and L. N. Bhuyan, Performance evaluation of a dataflow architecture. *IEEE Trans. Comput.* **39,** 615–627 (1990).
8. M. Ajmone Marsan, G. Balbo and G. Conte, A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems. *ACM Trans. Comput. Syst.* **2,** 93–122 (1984).
9. G. Balbo, S. C. Bruell and S. Ghanta, Combining queueing network and generalized stochastic Petri net models for the analysis of software blocking phenomena. *IEEE Trans. Software Engng* **12,** 561–576 (1986).
10. G. Balbo, S. C. Bruell and S. Ghanta, Combining queueing networks and generalized stochastic Petri nets for the solution of complex models of computer systems. *IEEE Trans. Comput.* **17,** 1251–1268 (1988).
11. D. Ghosal, S. K. Tripathi, L. N. Bhuyan and H. Jiang, Analysis of computation–communication issues in dynamic dataflow architectures. *Proc. Sixteenth Annual Symp. on Computer Architecture*, pp. 325–333 (1989).
12. J. R. Gurd, I. Watson and C. C. Kirkham, The Manchester prototype dataflow computer. *Commun. ACM* **28,** 34–52 (1985).

## AUTHORS' BIOGRAPHIES



**C. R. M. Sundaram**—C. R. Meenakshi Sundaram completed the Batchelor and Master's degrees in computer science and engineering from the Department of Computer Science and Automation at the Indian Institute of Science, Bangalore in 1988 and 1990, respectively. Currently, he is a doctoral student at the department of computational science at the University of Saskatchewan, Saskatoon, Canada. His research interests are in the areas of modeling and analysis of parallel and distributed systems, and computer communication networks.

**Y. Narahari**—Dr Y. Narahari is an Assistant Professor at the Indian Institute of Science where he teaches and researches on performance modeling and analysis, and analysis of manufacturing systems. He has a Ph.D. from the Indian Institute of Science and he has written several papers on Petri net models of manufacturing systems, deadlock analysis in flexible manufacturing systems, and performability analysis of fault-tolerant systems. He has recently co-authored a book entitled *Performance Modeling of Automated Manufacturing Systems*, published by Prentice-Hall, Englewood Cliffs. During July 1992–January 1993, he visited the Massachusetts Institute of Technology, Cambridge under the INDO–U.S. Science and Technology Fellowship Program for collaborative research.