# A STOCHASTIC MODEL OF MULTIPROCESOR ACCESS TO AN INTERLEAVED MEMORY

by

LEON J. OSTERWEIL
Department of Computer Science
University of Colorado
Boulder, CO 80302

Report #CU-CS-006-72 Oct. 1972

* This work supported by NSF Grant GJ-660.

# A STOCHASTIC MODEL OF MULTIPROCESSOR ACCESS TO AN INTERLEAVED MEMORY

by

Leon J. Osterweil *

ABSTRACT

In this paper we create a model of the way in which processors access a shared central memory. We investigate the way in which overall system efficiency increases as the amount of central memory interleaving is increased. We discover that studies such as this can have profound importance. For example, we see that a two-processor computer system having the characteristics of our model will lose about 70% of the processing power of one of its processors if central memory is only two-way interleaved. This loss of power drops sharply for four-, eight- and sixteen-way interleaving. At the end of the paper we also advance proposals for sharpening this admittedly-crude model and for modeling n-processor systems.

## I. INTRODUCTION

In most large computers, the central memory is divided into modules, often called banks, of words. In such cases each module or bank contains its own addressing hardware. Thus, the several banks are capable of making memory references simultaneously.

Through the years, this architectural feature has been used in many different ways. For instance, by putting a program's instructions in one bank of consecutive memory locations and its data in another bank of consecutive memory locations, it is possible to speed program execution by using a technique called overlap. The general idea here is to fetch a particular instruction from memory location L during machine cycle n, and also compute the address of the instruction's operand at this time. Then during machine cycle n + 1 the operand can be fetched from the program's data bank while the next program instruction is simultaneously fetched from location L + 1 of the instruction bank. The objective here is to roughly double the execution time of a program.

In other machines, the goal of using separate banks is to enable the interleaving of the memory. In an interleaved memory, a bank consists not of a block of consecutive words but rather of a block of words whose locations are the same modulo b, where b is the number of banks (invariably a power of 2). Such a memory is often referred to as a b-way interleaved memory.

In machines such as the ILLIAC II and the CDC-6600, the memory is interleaved because memory reference speed is slow relative to the

speed or processing power of the central processor. In these machines a program's instructions are placed so that consecutive instructions wind up in consecutive banks of memory. Thus it is possible to fetch b consecutive instructions simultaneously. The goal of doing this is to get several instructions into the processor more or less simultaneously where a great deal of decoding and processing can be done while another fetch of the next several words is going on.

With the advent of multiprocessor machines designed to do multiprogramming a new use for interleaved memories has become apparent. This use can be illustrated by the following situation: Let us assume we have a multiprogramming machine with two identical processing units and a central memory shared by them. Suppose there are two users -- each with control of a different processor and each attempting to execute his own program. Let us further assume that the two programs are stored in the central memory in such a way that the two instruction sets reside in the same bank of memory and/or the two data sets reside in the same bank of memory. It is worthwhile to note that this last assumption is not an unreasonable one. In a multiprogramming system, frequently-used systems such as compilers are often implemented using reentrant code so that several concurrent executions of that program will nevertheless require that only one instruction set be resident in central memory. Hence two users using the same reentrant program would actually be using instruction sets located in the same bank of memory (namely, the same instruction set).

This utilization of memory results in serious problems. Since only one processor may access a memory bank during any memory

reference cycle we see that if, say, the instruction sets for two different users share a single memory bank, then any instruction reference by one user can "freeze out" or block any instruction reference by the other user. In the extreme case where each instruction takes exactly one memory reference cycle to execute, this could result in complete blockage of one processor by the other.

It is impractical to alleviate the problem by loading programs into memory in such a way that all user instruction sets go into different banks, and all user data sets go into different banks. Instead the problem is attacked through the use of interleaved central memory.

As an illustration, let us hypothesize a machine having two processors which share a b-way interleaved central memory. Let us suppose that each processor is attempting to execute the same reentrant program. (Thus the processors share the same instruction set and have different data sets). The two processors may access instructions simultaneously provided that the instructions they fetch have addresses which are different modulo b. Moreover, if on a given machine cycle there is no blockage and each processor executes a nontransfer of control instruction having a c-machine cycle duration then we are assured that no blockage will take place on the next instruction fetch either. Hence, we see that the rate of blockage can be reduced by the use of central memory interleaving.

It is apparent, however, that there are many factors which cause the rate of blockage in such a system to be nonzero. For example,

should a processor execute a transfer of control statement located in bank b then we can no longer assume that it will attempt to draw its next instruction from bank b + 1 (mod b). Hence, its next instruction fetch may block or be blocked by the instruction fetch of another processor. On most machines, moreover, different instructions take varying amounts of time to execute. Thus we cannot assume that all processors will move from bank to bank in unison. Hence one processor may "catch up" to another causing blockage.

In addition, we must recognize the fact that most machines attempt to overlap execution of instructions as described above. Hence each processor will attempt to make up to two memory references during each instruction execution rather than just one. Each of these references may independently result in a blockage and it is important to observe that these blockages may propagate. For example, let us assume that two processors attempt to access instructions from different banks, but both attempt to access data from the same bank. If we assume that the one processor which is blocked sits idle for this cycle, then we must recognize that both processors may now attempt to access instructions from the same bank on the next cycle even though no transfer of control may have been executed. Clearly this resulting blockage may cause a data fetch blockage on the next cycle and so forth.

Thus the problem of determining the probability of blockage in a multiprocessor system utilizing interleaved memory is formidable but it is a significant problem which should be of vital concern to computer architects. A computer with too many processors sharing an

inadequately interleaved central memory may be doomed to needlessly underutilize those processors.

## 2. A TRACTABLE YET INTERESTING MACHINE ARCHITECTURE

We have hypothesized a machine architecture which is not too unreasonable by contemporary standards, yet which is amenable to analysis. Moreover, the analysis of the machine leads to remarkable results.

This hypothetical machine has two identical processors, $P_1$ and $P_2$, sharing a b-way interleaved memory which is, nevertheless, divided into two separate areas. One area of memory is reserved solely for the storage of instructions -- the other is reserved solely for the storage of data. Each of these two areas is b-way interleaved. Thus the memory contains 2b banks. The following assumptions apply to instructions: Each instruction occupies exactly one word; each instruction executes in exactly one unit of time (a machine cycle); and the execution of an instruction is always overlapped with the fetch of the next instruction; the execution of an instruction always requires the fetch of an operand from the data storage area. Hence on this machine each processor must do both an instruction fetch and a data fetch during each machine cycle. We assume that in case of a blockage $P_1$ will be allowed to do both accesses, and $P_2$ will remain idle for one cycle attempting to make the same pair of accesses on the next cycle.

This model does not seem to us to be overly unrealistic. The greatest oversimplifications seem to be the assumption of uniform execution time for instructions and the assumption of a data fetch for every instruction execution. We expect to analyze models of more realistic machines in subsequent research. For now, however, we find this model challenging and provacative.

## 3. A STOCHASTIC MODEL OF THE MACHINE

We have studied the machine described above by means of stochastic modeling techniques. It seems clear that the rate of blockage for our hypothetical machine is strongly dependent upon whether or not the pattern of memory accesses required by the processors is cyclic. For example, if a processor never executes a transfer of control statement, then it will attempt to access locations in the instruction area in a purely sequential fashion. Hence it will attempt to access bank 0, then 1, then 2, ... then b-1, then 0, ... This orderly procession of bank accesses will be interrupted only by the execution of a transfer of control instruction which we shall call an instruction jump. Clearly, the cyclic order of bank accesses is disrupted only if the instruction jump involves a jump of length other than 1 modulo b.

Similarly if consecutive instructions attempt to access consecutive data words, then the pattern of data bank references will be orderly (cyclic). Otherwise, we shall say the processor has executed a data jump. Clearly a processor can execute either, neither, or both types of jumps on any given instruction execution. Moreover, it seems

perfectly reasonable to consider the execution of an instruction jump and the execution of a data jump to be independent events in the probability sense.

We note that if $b > 1$, and both processors have begun simultaneous execution, then if neither processor ever executes a data or instruction jump, there is no possibility of subsequent blockage. The execution of jumps can, on the other hand, lead to blockages. Hence in order to determine the probability of processor blockage, we must study the way in which jumps can produce blockages. We do this by means of stochastic modeling techniques.

Let us assume that for each processor, the probability of executing an instruction jump on any machine cycle is $i$, and that the probability of executing a data jump on any machine cycle is $d$. As noted previously, it is reasonable to regard these events as independent, and we shall do so. On any machine cycle, the machine may be in one of four states:

1. There is an instruction blockage and a data blockage,
2. There is an instruction blockage, but no data blockage,
3. There is no instruction blockage, but there is a data blockage,
4. There is no blockage of either type.

The probability of blockage is thus the fraction of time the machine spends in the first three states or 1 - the fraction of time spent in state 4.

This fraction can be computed by standard methods for manipulating Markov Chains (for example, see [1]) provided transition probabilities can be computed between all the states. We can compute these probabilities as functions of b, i and d.

We shall denote by T the matrix of transitions between these four states. By $T_{ij}$ we denote the probability that the machine will move from state i to state j. $T_{ij}$ will be the i, $j^{th}$ entry in T. The matrix T is given by:

$T_{11}$: $\frac{id}{b^2}$

$T_{12}$: $\frac{i}{b}\left(1 - \frac{d}{b}\right)$

$T_{13}$: $\left(1 - \frac{i}{b}\right)\frac{d}{b}$

$T_{14}$: $\left(1 - \frac{i}{b}\right)\left(1 - \frac{d}{b}\right)$

$T_{21}$: $\frac{i}{b}\left(\frac{d}{b} + \frac{1}{b-1}(1 - d)\right)$

$T_{22}$: $\frac{i}{b}\left(\frac{b-1}{b}d + \frac{b-2}{b-1}(1 - d)\right)$

$T_{23}$: $\left(1 - \frac{i}{b}\right)\left(\frac{d}{b} + \left(\frac{1}{b-1}\right)(1 - d)\right)$

$T_{24}$: $\left(1 - \frac{i}{b}\right)\left(\frac{(b-1)d}{b} + \frac{b-2}{b-1}(1 - d)\right)$

$T_{31}$: $\left(\frac{i}{b} + \frac{1}{b-1}(1 - i)\right)\frac{d}{b}$

$T_{32}$: $\left(\frac{i}{b} + \frac{1}{b-1}(1 - i)\right)\left(1 - \frac{d}{b}\right)$

$T_{33}$: $\left(i - \frac{i}{b} + \frac{b-2}{b-1}(1 - i)\right)\frac{d}{b}$

$T_{34}$: $\left(i - \frac{i}{b} + \frac{b-2}{b-1}(1 - i)\right)\left(1 - \frac{d}{b}\right)$

$T_{41}$: $\left(\frac{i}{b}\right)(2 - i)\frac{d}{b}(2 - d)$

$T_{42}$: $\frac{i}{b}(2 - i)\left[(1 - d)^2 + \left(1 - \frac{1}{b}\right)(2d - d^2)\right]$

$$T_{43}: \quad [(1 - i)^2 + (1 - \frac{1}{b}) (2i - i^2)] \frac{d}{b} (2 - d)$$

$$T_{44}: \quad [(1 - i)^2 + (1 - \frac{1}{b}) (2i - i^2)] [(1 - d)^2 + (1 - \frac{1}{b}) (2d - d^2)]$$

Let us denote $T^T$ by M. We can now use this matrix to produce results concerning the probability of blockage. Let:

$$X = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

be a column vector where $x_i$ is the probability that the machine is in state i during any cycle. If we assume that the machine has reached an equilibrium condition with both processors involved in the execution of a reentrant program, then we can say that $MX = X$ or $(M - I)X = 0$, where I is the identity matrix of degree four. We can now solve this matrix equation for X. The probability that $P_2$ is blocked on any machine cycle is then given by $1 - x_4$.

The matrix M is a stochastic matrix, hence M - I is a singular matrix. Hence X is not uniquely determined. However, it is clear that $\sum_{i=1}^{4} x_i = 1$. Thus, it is possible to solve $(M - I)X = 0$ for X subject to the condition that $\sum_{i=1}^{4} x_i = 1$.

## 4. SOME RESULTS OBTAINED

As already noted, we are interested in studying R, the probability of blockage for our machine. We noted that $R = 1 - x_4$ and is

therefore a function of the three variables b, i and d. Since $x_4$ is determined by solving the matrix equation $(M - I)X = 0$ where each entry of M is a function of b, i and d, it seems that obtaining an explicit formula for the blockage is difficult. On the other hand, we can certainly evaluate all sixteen entries in M for a particular triple of values for b, i, and d. Under these circumstances we can readily compute a numeric value for $x_4$, and hence the probability of blockage. Using a computer we have generated the results tabulated in Table I and II.

The results in Table II do not immediately suggest the functional relationship between R and its parameters b, i, and d. They do, however, seem to suggest some interesting conclusions. For example, we see that R seems to be strongly dependent upon b, but remarkably far less dependent upon fluctuations in the values of i and d.

These computations seem to indicate that two-way interleave is quite inadequate for our machine. Four-way interleave seems to be far better. Eight- and sixteen-way interleave offer further but less dramatic improvements.

These observations are, of course, no substitute for an explicit formula for R in terms of b, i, and d. Such a formula might be obtainable through the symbolic solution of the matrix equation $(M - I)X = 0$. This task appears forbidding, however. Alternatively, some sort of curve fitting technique might be applied to the data of Table II in an effort to obtain an approximating formula.

TABLE I

EVALUATION OF $x_4$, or 1 - R(b, i, d)

For b = 2

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .3322 | .3311 | .3279 | .3268 | .3247 | .3236 | .3226 |
| .2 | .3311 | .3289 | .3226 | .3205 | .3165 | .3145 | .3125 |
| .3 | .3300 | .3268 | .3165 | .3145 | .3086 | .3058 | .3030 |
| .5 | .3279 | .3226 | .3077 | .3030 | .2941 | .2899 | .2857 |

For b = 4

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .8776 | .8322 | . 7373 | .7161 | .6857 | .6760 | .6697 |
| .2 | .8322 | .7936 | .7109 | .6919 | .6645 | .6555 | .6497 |
| .3 | .7945 | .7611 | .6880 | .6709 | .6458 | .6375 | .6320 |
| .5 | .7373 | .7109 | .6512 | .6368 | .6152 | .6079 | .6029 |

For b = 8

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .9477 | .9258 | .8768 | .8655 | .8495 | .8448 | .8422 |
| .2 | .9258 | .9054 | .8595 | .8488 | .8337 | .8293 | .8268 |
| .3 | .9068 | .8877 | .8444 | .8343 | .8199 | .8157 | .8133 |
| .5 | .8768 | .8595 | .8202 | .8109 | .7977 | .7938 | .7916 |

For b = 16

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .9752 | .9644 | .9399 | .9343 | .9264 | .9243 | .9233 |
| .2 | .9644 | .9540 | .9303 | .9248 | .9172 | .9151 | .9142 |
| .3 | .9550 | .9449 | .9218 | .9165 | .9091 | .9071 | .9061 |
| .5 | .9399 | .9303 | .9083 | .9032 | .8961 | .8941 | .8933 |

TABLE II

PROBABILITIES OF BLOCKAGE, $R(b, i, d) = 1 - x_4(b, i, d)$

For b = 2

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .6678 | .6689 | .6721 | .6732 | .6753 | .6764 | .6774 |
| .2 | .6689 | .6711 | .6774 | .6795 | .6835 | .6855 | .6875 |
| .3 | .6700 | .6732 | .6835 | .6855 | .6914 | .6942 | .6970 |
| .5 | .6721 | .6774 | .6923 | .6970 | .7059 | .7101 | .7143 |

For b = 4

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .1224 | .1678 | .2627 | .2839 | .3143 | .3240 | .3303 |
| .2 | .1678 | .2064 | .2891 | .3081 | .3355 | .3445 | .3503 |
| .3 | .2055 | .2389 | .3120 | .3291 | .3542 | .3625 | .3680 |
| .5 | .2627 | .2891 | .3488 | .3632 | .3848 | .3921 | .3971 |

For b = 8

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .0523 | .0742 | .1232 | .1345 | .1505 | .1552 | .1578 |
| .2 | .0742 | .0946 | .1405 | .1512 | .1663 | .1707 | .1732 |
| .3 | .0932 | .1123 | .1556 | .1657 | .1801 | .1843 | .1867 |
| .5 | .1232 | .1405 | .1798 | .1891 | .2023 | .2062 | .2084 |

For b = 16

| i \ d | .1 | .2 | .5 | .6 | .8 | .9 | 1.0 |
|---|---|---|---|---|---|---|---|
| .1 | .0248 | .0356 | .0601 | .0657 | .0736 | .0757 | .0767 |
| .2 | .0356 | .0460 | .0697 | .0752 | .0828 | .0849 | .0858 |
| .3 | .0450 | .0551 | .0782 | .0835 | .0909 | .0929 | .0939 |
| .5 | .0601 | .0697 | .0917 | .0968 | .1039 | .1059 | .1067 |

We note that the values of R for b = 2 appear to approximately satisfy the equation $R(i, d)\Big|_{b=2} = \frac{1}{3} - \frac{id}{10}$. On the other hand, the values for b = 4, 8 and 16 do not appear to be linear in either i or d. Hence it appears that the relationship between R and its parameters is not a simple one.

## 5. FUTURE RESEARCH AREAS

It seems clear that there are interesting and surprising results to be obtained in this area of investigation. Moreover, the importance of such work should be clear. Computer system efficiency seems to depend in important and subtle ways upon the relationship between numbers of processors and extent of memory interleaving. Results in this area should be available to computer hardware system designers.

The results given here are just a beginning. Much more work is needed in this area. Perhaps the most obvious extension of this work would be to consider computer systems with n processors, where $n > 2$. This would entail the analysis of larger stochastic matrices.

Another important extension of this work would be to change the model of the machine we are analyzing. It is unrealistic in terms of present day architecture to model a machine in which all instructions execute in the same length of time and in which each instruction must reference an operand in memory. The model should be altered so that these restrictions are removed. Since we are already dealing with a stochastic system, this could be done by parameterizing the probability of an operand reference and the probability of execution

of a multicycle instruction. It is reasonable to guess that blockage rates for such machines might be radically different from those obtained here and perhaps might be strongly dependent upon such parameters.

Finally, it seems that further attempts at determining the functional relationships between R and the various parameters should be made. The symbolic solution of the stochastic matrices developed here could be pursued perhaps through the use of a symbol manipulation package on a computer. Such a course of action may be reasonable for the small 4 x 4 case considered here, but it unlikely to be a fruitful approach when dealing with the subsequent research suggested in this section. Some sort of curve fitting scheme is more likely to be successful in determining the desired relationships.

cah

## REFERENCES

1. W. Feller, An Introduction to Probability Theory and its Applications, Volume 1, John Wiley and Sons, Inc., New York, 1950.

LJO:cah