1197

SIMULATION OF SATELLITE COMMUNICATION RANDOM MULTIPLE ACCESS PROTOCOL

Dissertation submitted to

Jawaharlal Nehru University

in partial fulfilment of the

requirements for the award

of the degree of

MASTER OF TECHNOLOGY

IN

COMPUTER SCIENCE AND TECHNOLOGY

by

PYARE LAL MEENA

SCHOOL OF COMPUTER AND SYSTEMS SCIENCES

JAWAHARLAL NEHRU UNIVERSITY

NEW DELHI - 110067

JANUARY, 1995.

CERTIFICATE

This is to certify that the thesis entitled " **SIMULATION OF SATELLITE COMMUNICATION RANDOM MULTIPLE ACCESS PROTOCOL** " being submitted by me to Jawaharlal Nehru University, in partial fulfilment of the requirements for the award of the degree of Master of Technology in Computer Science and Technology is a record of original work done by me under the supervision of Prof. Karmeshu and Mr. S. Madan, School of Computer and Systems Sciences during the Monsoon Semester, 1994.

The result reported in this thesis have not been submitted in part or full to any other University or Institution for the award of any degree etc.

20.01.95

(PYARE LAL MEENA)

Prof. K.K. Bharadwaj
Dean, SC & SS
JNU, New Delhi

Prof. Karmeshu
Professor, SC & SS
JNU, New Delhi

Mr. S. Madan
Asst. Professor, SC & SS
JNU, New Delhi

ACKNOWLEDGMENT

I wish to express my deep sense of gratitude to my project guides **Prof. Karmeshu** and **Mr. S. Madan** of School of Computer and Systems Sciences, J.N.U. New Delhi under whose invaluable supervision and guidance my project work was successfully completed. I gratefully acknowledge the facilities provided by the School of Computer and Systems Sciences.

My thanks are also due to those whose names are not mentioned but who have extended support to me for completion of this work.

20.01.95

(PYARE LAL MEENA)

CONTENTS

CHAPTER 1

INTRODUCTION

The burgeoning technology of satellite communication opens up various horizons in the field of state-of-the-art communictions.The whole globe is brought into unison, just by hopping the data via a satellite. These distances cann't obviously be covered otherwise, by laying down fibre optics cables etc. The wideband data services supported by the satellite communication, encompasses all the needs of technical as well as commercial world.

To make this technology more efficient and cost effective there should be a mechanism which extends this facility to large number of users with a better quality.

In view of the growing needs of day to day communications, the multiple access network suits best for this purpose. It allows a large number of users share the same channel. For this many protocols have been developed to fulfill the requirement. A protocol in contrast, is a set of rules governing the format and meaning of the frames, packets, or messages that are exchanged by the pear entities within a layer[4]. These protocols basically scan through a random access on one end to a pre-assigned access on the other end. The limitation of other networks are given in Chapter 2.

A multiple access scheme proposed by W.W.Wu[2], which has relatively some advantages over other schemes of such as high throughput, simultaneous transmissions toleration and support a large number of users. The scheme called Random Multiple Access is defined as a Time-Frequency (T-F) system utilizing speed spectrum, pulse address and random transmission principles. In this there is no need to check for channel occupancy. In **RMA**, each user is assigned to a unique sequence of code words, which is an entry to a discrete T-F matrix. So the strategy of code sequence generation is very important. It provides controlled overlapping between any two sequences used in the system. It also provides a large number of coded sequences to accommodate large number of traffic users.

The work which is presented in this project in a systematic way as follows:-

(a) Theoretical background of Random Multiple Access Technique proposed by Wu[2]

(b) Computer Program in PASCAL to generate **RMA** code sequences using Wu algorithm.

(c) Simulation of a network where transmission takes place randomly and study the result obtained in terms of channel throughput with varying offered traffic load.

This algorithm can be implemented to VLSI and desired time, number of transistors, execution time etc., can be found out and finally a layout can be generated.

As it is clear from the algorithm that we require adder, shifter, multiplier, B to D converter and D to B converters, filters, comparators etc.

In Chapter 2 comparison of a **RMA** scheme with other schemes and **RMA** code generation algorithm are explained. Chapter 3 gives **RMA** scheme representation works and system modelling and simulation. Chapter 4 is fully devoted to the results of the simulation. Chapter 5 which is most important chapter contains the conclusions. In Appendix 1 the simulation flow charts and programming code in Pascal are attached.

CHAPTER 2

THEORY OF RAMA SCHEME

Using multiple access process, various protocols have been developed. The relative merits and demerits of some ofthose are as follows:

2.1 Comparison with Other Schemes

When the number of users becomes very large and number of packets vary unpredictably, pre-assigned access schemes have some limitations i.e.

(i) In Time division multiple access (TDMA), the number of time slots per frame are restricted by the speed of transmission so, we require timing jitter and guard time.

(ii) Because of the bandwith restriction the number of frequency division are limited. If we increase the frequency bandwidth the oscillator stability problem arises.

(iii) In case of basic demand schemes the source identification, channel conditions, message routing, tracking and control bog down the system processes when the number of users become very large.

(iv) Pure ALOHA is suitable for large users and it in a low duty cycle traffic network but has no tolerance for a simultaneous packet transmissions and has maximum throughput of just 18%[4].

(v) CDMA (Code division multiple access) supports simultaneous users but because of low number of orthogonal sequences it can not entertain large number of users.

A significant improvement in the quality of a transmission channel may be obtained if codes are properly chosen and decoding techniques are efficiently applied **RMA** differs from CDMA and PAMA, in the **RMA** does not use pseudonoise or other direct sequences techniques.

2.2 RMA **Scheme**

A **RMA** code word is defined as a collection of n symbols. The code symbols are the elements of a Time-frequency (T-F) matrix M. It has same spectrum spreading property as code division multiple access (CDMA), and the addressing principle as pulse addressing multiple access (PAMA). The total number of symbols in TF matrix M is given by M=nxf. Where n is the number of time divisions and f is the number of frequency divisions.

The objective is to construct a sequence of m-ary nxf symbol so that a pair of sequences represents binary signaling from a particular station. If the sequences are completely orthogonal, maximum cross-correlation result in a low probability of error. However, the number of such sequences would then be highly limited as in case of CDMA. Hence, one overlapping symbol in considered in order to increase the number of such sequences. This is expected to result in some graceful degradation due to collisions the high traffic cases which have been studied through simulation is the present work.

Code transmission and detection can be illustrated by an example[5]. Let us divide a given system bandwith and time frame into three divisions i.e. n=f=3

$$M = \begin{bmatrix} 0 & 1 & 2 \\ 3 & 4 & 5 \\ 6 & 7 & 8 \end{bmatrix} \begin{matrix} f_2 \\ f_1 \\ f_o \end{matrix}$$
$$\quad t_o \quad t_1 \quad t_2$$

We further assume that a binary '1' represented by the code word (3,1,2) and binary 'O' by (6,4,8) then we have

$$1 = f_1 t_o + f_2 t_1 + f_2 t_2$$

$$O = f_o t_o + f_1 t_1 + f_o t_2$$

A set of frequency synthesizers along with the delay units generate the corresponding codes sequence at the output of the transmitter depending upon binary input received.

The receiver detects this code sequence through a set of filters and delay units and companies the transmitted symbols with stored values to check for any error.

2.3 Principles of Code Generation

The receiver of the system performs the inverse transformation from a sequence of multiple level symbols back to binary digits. The error performance and system efficiency are mostly affected by the construction of the multiple level symbol sequences for any given number of access stations. In this section, therefore, a systematic procedure is presented. This procedure is optimum in the sense that no other sets of sequences of same length can provides a better sequence distance property.

The code generation algorithm uses the properties of combinatorial sets in deriving the optimum code symbols. Combinatorial theory is a branch of applied mathematics that deals with combination permutation arrangement or enumeration of discrete physical objects to achieve the optimal solution[5]. The key to successful application in to match the structure of the set to the physical parameters of the problem is question or at least as closely as possible. The set can provide an instant optimum guideline, where other methods fail combinatorial sets consist of two types of sets.

(i) the additive sets and

(ii) the difference sets

Difference sets are considered more useful from a particle point of view [5].

2.4 Procedure of Sequence Construction Algorithm:

The algorithm begins with two matrices W_o and W_1, with size $(n^2-1)xn$ and $(n+1)xn$ respectively. The row of W_o consist of the elements of Euclidean geometry difference set and its equivalent sets (algorithm to get difference set is in [5]). The elements in the rows of W_1 are so construction that the first element in its 0^{th} row can not be any difference module (n^2-1) among the set of residues. A row matrix R_j is defined in terms of row of time time-frequency matrix M from M and R_j we get B_j. A set of matrices. A_k is constructed and each A_k is used as an ordering transformation the B_j's to obtain a set of matrices U[j,k]. The rows of B_j, U[j,k], and the f rows of M constitute the set of desired sequences. If the sequence lengths is n then algorithm generate n set of sequences of following properties:

* Number of time divisions = n
* Number of symbols = n^2
* Sequence minimum distance = n-1
* Number of frequency divisions = n^2
* Number of sequences = $n^2 (n^2+n+1)$

CHAPTER 3

REPRESENTATION OF RMA SYSTEM

A system is a collection of interacting elements or components that act together to achieve a common goal systems can be studied by direct experimentation by building prototype or by building mathematical/models. The purpose of study through modelling is to aid the analysis understanding design, operation, prediction or control of systems without actually constructing and operating the real thing. Simulation is the process of imitating the behaviour of the real system by constructing and experimenting with a model which is only simplified representation of the system.

3.1 Representing Work

Modelling we mean is the determination of those quantitative features which describe the operation of a system to study the system we take simpler system which resemble with the actual system.

Developing of model involves two tasks (i) developing a representation of the system, and (ii) developing a representation of the work done by the system, these are interested. Some considerations are as follows:

3.1.1 Variability

Most of the problems resources; this contention causes work to be quenched for or blocked from execution, and system performance may suffer as a result.

3.1.2 Choosing a Distribution

Determining what distribution use to represent most different variable is one of the most difficult aspects of the simulation modelling.

The most important distribution in the analysis of queuing system is the exponential distribution Random arrivals or random service, we mean that the inter-arrival times or the service times are exponentially distributed. We take poison arrivals because of the relationship between piosson and exponentially distribution.

An important property of the Poisson process is that it is in employees; the probability of an arrival in interval 't' depends only on the duration of t, and not on the previous arrival occurrence. Another merit of Poisson process is * that the super position of Poisson process is itself a Piosson process.

3.1.3 Sampling from Distribution

A particular value of a model variable is determined by generating a random sample from the distribution specified for that variable we use a uniform random number generator to generate these samples; the uniform random number generator is a numerical algorithm which produces a deterministic sequence of values distributed between 0 and 1.

3.2 Simulation model of RMA Codes Scheme

In **RMA** scheme, the user's messages are coded and an **RMA** sequenced are obtained. The messages are transmitted randomly and depend on their availability. So this could result in message overlapping but as seen from the figure 2, every overlapping does not conclude in collision. This is because of the controlled overlapping provided by the orthogonality is coded sequences by the **RMA** code generation algorithm. However, we have limitation of the code length, it is not very large and one overlapping symbol has been allowed designing code generation there is district probability that two or more overlapping messages have the same code word and same time to result in a clash.

The various possible cases are shown in figure 2 to clear the condition of the clash.

It is clear from the figure that some part or all message which have same code word and same time of overlapping cause a clash.

3.3 Message Arrival Process:

For the purpose of simulation, a treatment carried out as follows. Th message are generated using Poisson probability distribution with exponential inter-arrival time (IAT). The probability that no message arrive before a time T is given by the probability density function [4]

$$P_r\ [T > t] = \lambda\ e^{-\lambda t}$$

where λ is a positive constant. The distribution function F(t) is the integral of density function i.e.

$$F(t) = \int_{o}^{t} \lambda e^{-\lambda t}\, dt = 1 - e^{-\lambda t}$$

The average or expected value of t is given by

$$E(t) = \int_{o}^{\infty} t \quad e^{-\ t}\, dt$$

$$= 1/\lambda$$

So we define λ as the inverse of mean inter-arrival time (MIAT). Since the distribution function F(t) has a value varying between '0' and '1', this property can be used to derive exponentially distributed inter-arrival times utilising the inverse transformation function, as in figure 3.

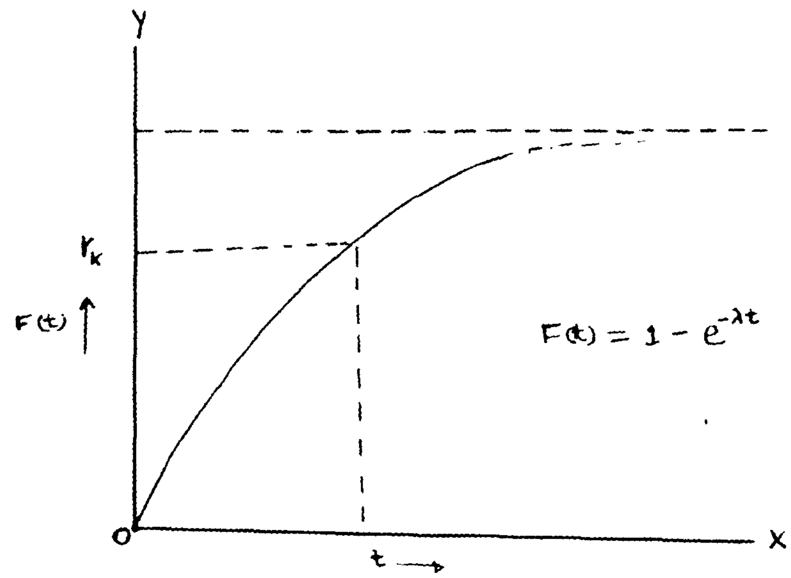


Fig 3 Arrival distribution function

Let r be the random number between 0 and 1 using one to one correspondence between this random number and the distribution function, we get

$$r = 1 - e^{-\lambda t}$$

or $$-\lambda t = \ln (1-r)$$

$$t = - \ln (1-r)/\lambda$$

$$t = F^{-1}(r)$$

Since (1-r) has similar as r its randomness. we replace it by r,

$$t_k = - \frac{1}{\lambda} \ln r_k$$

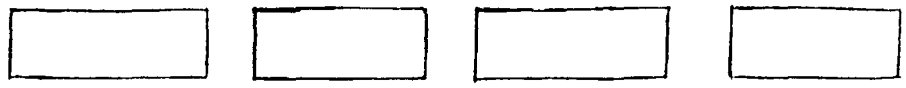
therefore, for a given r_k, a corresponding t_k can be found if know the arrival rate λ.

So the arrival process is modelled through a Poisson point process with exponential distribution for interarrival times. Random numbers between '0' and '1' are used to determined the interarrival distribution function.

3.4 Channel Observation Process

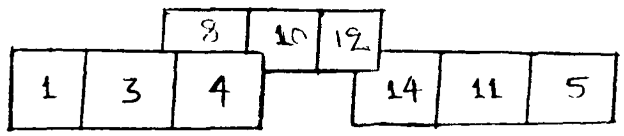
The channel observation process is involved very much here is necessary, but not a sufficient condition for message clash. It is therefore required to actually compare the contents of the overlapped portion of the message to determine a clash. A Monte-Carlo technique is used to generate random streams of data bits in each message. Then depending upon the binary digit '0' or '1' a corresponding **RMA** code sequence is assigned to each message, to achieve this the programme for the code sequence generation is made separately and the code sequence for various values of the code sequence length n; are determines.

These are stored in data file (**RMA**CODE) which is read by the simulation program while assigning these code sequences; the channel observation process compares the overlapping message bit by bit to check the clash.



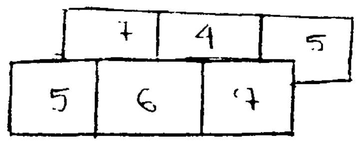
No Packet overlap — No Clash

(a)



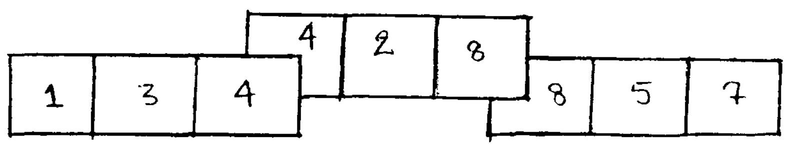
overlapping but code words are not same — No Clash

(b)



Overlapping with same code words but they are not exist simultaneous — No clash

(c)



Overlapping with same code words and they exist simultaneous — Clash

(d)

Fig 2 CLASH CONDITION FOR RIMA

Due to the random arrived of the message, it is possible that there could be multiple overlaps i.e. message no.1 could have overlap with message nos. 2,3 & 4. Each of these messages has different contents and are generally orthogonal to each other. It is however, quite likely that message no.1 may not clash with message no.2 but it may clash with message no.3 or 4.

The channel observation process thus has to keep a track of correct number of distinct message collisions, and not to count a collided message twice.

3.5 Traffic Model

Mean Interarrival Time (MIAT) which determines the value of is based upon the actual traffic statistics of the application in mind. Let we consider the following traffic model.

No. of nodes or user = N

No. of message/node = M

No. of bits/message = W

Considering t hrs working day in which all the M messages in each node has to be cleared transmission bit rate required is given by

$$\text{Bit rate} = \frac{N \times M \times W}{t \times 3600} \text{ Kbps}$$

average number of message arriving at the satellite channel per second is given by

$$= \frac{N \times M}{3600 \times t}$$

if the traffic per node is higher or lower then this average value of would according be higher (say 20) and lower (say 1).

The above figures represent a realistic network of a typical thin traffic, large user application. However, due to computational constraints and for easiness of simulation while keeping the same values. The following model is chosen

No. of users = 100 = N

No. of message/node/day = 3000 = M

No. of bits/message = 100 = W

Transmission bit rate = 0.8328 kbps

In the above figures, though transmission bit rate is not required for simulation purpose, it is kept for the sake of completeness and to indicate the typical value for the application is mind.

3.6 Throughput Computation

Let us define the "message time" [4] as the amount of time needed to transmit a fixed length message (i.e. message lengths divided by the bit rate) on the traffic model chosen, we have computed approximately 10 message per second which for the chosen transmission bit rate would led to one message per message time. However, if we increased or decreased the traffic loading and kept the same transmission bit rate. The number of message per message time would also increase or decrease respectively.

The number of message offered per message time is defined as the offered traffic Z. Throughput TP is computed at different offered traffic, which is varied by changing the value of mean inter arrival time.

CHAPTER 4

RESULTS OF THE PROGRAM

RMACODE SEQUENCES ARE AS FOLLOWS

w0

1	3	4	12
2	4	5	13
3	5	6	14
4	6	7	0
5	7	8	1
6	8	9	2
7	9	10	3
8	10	11	4
9	11	12	5
10	12	13	6
11	13	14	7
12	14	0	8
13	0	1	9
14	1	2	10
0	2	3	11

w1

0	5	10	15
1	6	11	15
2	7	12	15
3	8	13	15
4	9	14	15

M

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15
16	17	18	19
20	21	22	23
24	25	26	27
28	29	30	31
32	33	34	35
36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	55
56	57	58	59

B0 =

4	12	16	48
5	13	17	49
6	14	18	50
7	15	19	51

U(0,1)

4	13	18	51
7	12	17	50
6	15	16	49
5	14	19	48

U(0,2)

4	15	17	50
6	12	19	49
5	14	16	51
7	13	18	48

U(0,3)

4	14	19	49
5	12	18	51
7	13	16	50
6	15	17	48

B1 =

8	16	20	52
9	17	21	53
10	18	22	54
11	19	23	55

U(1,1)

8	17	22	55
11	16	21	54
10	19	20	53
9	18	23	52

U(1,2)

8	19	21	54
10	16	23	53
9	18	20	55
11	17	22	52

U(1,3)

8	18	23	53
9	16	22	55
11	17	20	54
10	19	21	52

B2 =

12	20	24	56
13	21	25	57
14	22	26	58
15	23	27	59

U(2,1)

12	21	26	59
15	20	25	58
14	23	24	57
13	22	27	56

U(2,2)

12	23	25	58
14	20	27	57
13	22	24	59
15	21	26	56

U(2,3)

12	22	27	57
13	20	26	59
15	21	24	58
14	23	25	56

B3 =

16	24	28	0
17	25	29	1
18	26	30	2
19	27	31	3

U(3,1)

16	25	30	3
19	24	29	2
18	27	28	1
17	26	31	0

U(3,2)

16	27	29	2
18	24	31	1
17	26	28	3
19	25	30	0

U(3,3)

16	26	31	1
17	24	30	3
19	25	28	2
18	27	29	0

B4 =

20	28	32	4
21	29	33	5
22	30	34	6
23	31	35	7

U(4,1)

20	29	34	7
23	28	33	6
22	31	32	5
21	30	35	4

U(4,2)

20	31	33	6
22	28	35	5
21	30	32	7
23	29	34	4

U(4,3)

20	30	35	5
21	28	34	7
23	29	32	6
22	31	33	4

B5 =

24	32	36	8
25	33	37	9
26	34	38	10
27	35	39	11

U(5,1)

24	33	38	11
27	32	37	10
26	35	36	9
25	34	39	8

U(5,2)

24	35	37	10
26	32	39	9
25	34	36	11
27	33	38	8

U(5,3)

24	34	39	9
25	32	38	11
27	33	36	10
26	35	37	8

B6 =

28	36	40	12
29	37	41	13
30	38	42	14
31	39	43	15

U(6,1)

28	37	42	15
31	36	41	14
30	39	40	13
29	38	43	12

U(6,2)

28	39	41	14
30	36	43	13
29	38	40	15
31	37	42	12

U(6,3)

28	38	43	13
29	36	42	15
31	37	40	14
30	39	41	12

B7 =

32	40	44	16
33	41	45	17
34	42	46	18
35	43	47	19

U(7,1)

32	41	46	19
35	40	45	18
34	43	44	17
33	42	47	16

U(7,2)

32	43	45	18
34	40	47	17
33	42	44	19
35	41	46	16

U(7,3)

32	42	47	17
33	40	46	19
35	41	44	18
34	43	45	16

B8 =

36	44	48	20
37	45	49	21
38	46	50	22
39	47	51	23

U(8,1)

36	45	50	23
39	44	49	22
38	47	48	21
37	46	51	20

U(8,2)

36	47	49	22
38	44	51	21
37	46	48	23
39	45	50	20

U(8,3)

36	46	51	21
37	44	50	23
39	45	48	22
38	47	49	20

B9 =

40	48	52	24
41	49	53	25
42	50	54	26
43	51	55	27

U(9,1)

40	49	54	27
43	48	53	26
42	51	52	25
41	50	55	24

U(9,2)

40	51	53	26
42	48	55	25
41	50	52	27
43	49	54	24

U(9,3)

40	50	55	25
41	48	54	27
43	49	52	26
42	51	53	24

B10 =

44	52	56	28
45	53	57	29
46	54	58	30
47	55	59	31

U(10,1)

44	53	58	31
47	52	57	30
46	55	56	29
45	54	59	28

U(10,2)

44	55	57	30
46	52	59	29
45	54	56	31
47	53	58	28

U(10,3)

44	54	59	29
45	52	58	31
47	53	56	30
46	55	57	28

B11 =

48	56	0	32
49	57	1	33
50	58	2	34
51	59	3	35

U(11,1)

48	57	2	35
51	56	1	34
50	59	0	33
49	58	3	32

U(11,2)

48	59	1	34
50	56	3	33
49	58	0	35
51	57	2	32

U(11,3)

48	58	3	33
49	56	2	35
51	57	0	34
50	59	1	32

B12 =

52	0	4	36
53	1	5	37
54	2	6	38
55	3	7	39

U(12,1)

52	1	6	39
55	0	5	38
54	3	4	37
53	2	7	36

U(12,2)

52	3	5	38
54	0	7	37
53	2	4	39
55	1	6	36

U(12,3)

52	2	7	37
53	0	6	39
55	1	4	38
54	3	5	36

B13 =

56	4	8	40
57	5	9	41
58	6	10	42
59	7	11	43

U(13,1)

56	5	10	43
59	4	9	42
58	7	8	41
57	6	11	40

U(13,2)

56	7	9	42
58	4	11	41
57	6	8	43
59	5	10	40

U(13,3)

56	6	11	41
57	4	10	43
59	5	8	42
58	7	9	40

B14 =

0	8	12	44
1	9	13	45
2	10	14	46
3	11	15	47

U(14,1)

0	9	14	47
3	8	13	46
2	11	12	45
1	10	15	44

U(14,2)

0	11	13	46
2	8	15	45
1	10	12	47
3	9	14	44

U(14,3)

0	10	15	45
1	8	14	47
3	9	12	46
2	11	13	44

B15 =

0	20	40	60
1	21	41	61
2	22	42	62
3	23	43	63

U(15,1)

0	21	42	63
3	20	41	62
2	23	40	61
1	22	43	60

U(15,2)

0	23	41	62
2	20	43	61
1	22	40	63
3	21	42	60

U(15,3)

0	22	43	61
1	20	42	63
3	21	40	62
2	23	41	60

B16 =

4	24	44	60
5	25	45	61
6	26	46	62
7	27	47	63

U(16,1)

4	25	46	63
7	24	45	62
6	27	44	61
5	26	47	60

U(16,2)

4	27	45	62
6	24	47	61
5	26	44	63
7	25	46	60

U(16,3)

4	26	47	61
5	24	46	63
7	25	44	62
6	27	45	60

B17 =

8	28	48	60
9	29	49	61
10	30	50	62
11	31	51	63

U(17,1)

8	29	50	63
11	28	49	62
10	31	48	61
9	30	51	60

U(17,2)

8	31	49	62
10	28	51	61
9	30	48	63
11	29	50	60

U(17,3)

8	30	51	61
9	28	50	63
11	29	48	62
10	31	49	60

B18 =

12	32	52	60
13	33	53	61
14	34	54	62
15	35	55	63

U(18,1)

12	33	54	63
15	32	53	62
14	35	52	61
13	34	55	60

U(18,2)

12	35	53	62
14	32	55	61
13	34	52	63
15	33	54	60

U(18,3)

12	34	55	61
13	32	54	63
15	33	52	62
14	35	53	60

B19 =

16	36	56	60
17	37	57	61
18	38	58	62
19	39	59	63

U(19,1)

16	37	58	63
19	36	57	62
18	39	56	61
17	38	59	60

U(19,2)

16	39	57	62
18	36	59	61
17	38	56	63
19	37	58	60

U(19,3)

16	38	59	61
17	36	58	63
19	37	56	62
18	39	57	60

A1

0	1	2	3
3	0	1	2
2	3	0	1
1	2	3	0

A2

0	3	1	2
2	0	3	1
1	2	0	3
3	1	2	0

A3

0	2	3	1
1	0	2	3
3	1	0	2
2	3	1	0

4.2

Sequence Length = 3
Packet Transmission time = 0.1 sec
Channel speed = 1000 Bits/Sec.
Number of Packet = 60

Nclash	Lamda	Z	TP
7	1	0.1	0.08
5	2	0.2	0.11
13	3	0.3	0.13
24	4	0.4	0.17
29	5	0.5	0.20
27	6	0.6	0.22
38	7	0.7	0.24
41	8	0.8	0.26
43	9	0.9	0.27
38	10	1.0	0.27
39	11	1.1	0.26
42	12	1.2	0.25
43	13	1.3	0.21
46	14	1.4	0.20
48	15	1.5	0.15
48	16	1.6	0.14
49	17	1.7	0.14
53	18	1.8	0.14
53	19	1.9	0.13
54	20	2.0	0.12

Sequence Length = 4
Packet Transmission time = 0.1 sec
Channel Speed = 1000 bits/Sec
Number of Packets = 60

Nclash	Lamda	Z	TP
7	1	0.1	0.09
9	2	0.2	0.17
10	3	0.3	0.24
13	4	0.4	0.33
18	5	0.5	0.39
21	6	0.6	0.46
20	7	0.7	0.49
23	8	0.8	0.53
24	9	0.9	0.55
21	10	1.0	0.60
25	11	1.1	0.64
27	12	1.2	0.65
31	13	1.3	0.68
38	14	1.4	0.69
38	15	1.5	0.71
38	16	1.6	0.73
38	17	1.7	0.76
39	18	1.8	0.82
39	19	1.9	0.83
39	20	2.0	0.84

Sequence Length = 4
Packet transmission time = 0.1 sec
Channel speed = 1000 bits/sec
Number of Packets = 120

Nclash	Lamda	Z	TP
8	1	0.1	0.09
23	2	0.2	0.16
39	3	0.3	0.23
52	4	0.4	0.29
42	5	0.5	0.37
47	6	0.6	0.42
50	7	0.7	0.46
63	8	0.8	0.47
61	9	0.9	0.51
61	10	1.0	0.53
64	11	1.1	0.55
66	12	1.2	0.56
68	13	1.3	0.58
68	14	1.4	0.59
72	15	1.5	0.59
72	16	1.6	0.60
73	17	1.7	0.63
74	18	1.8	0.64
76	19	1.9	0.66
76	20	2.0	0.68

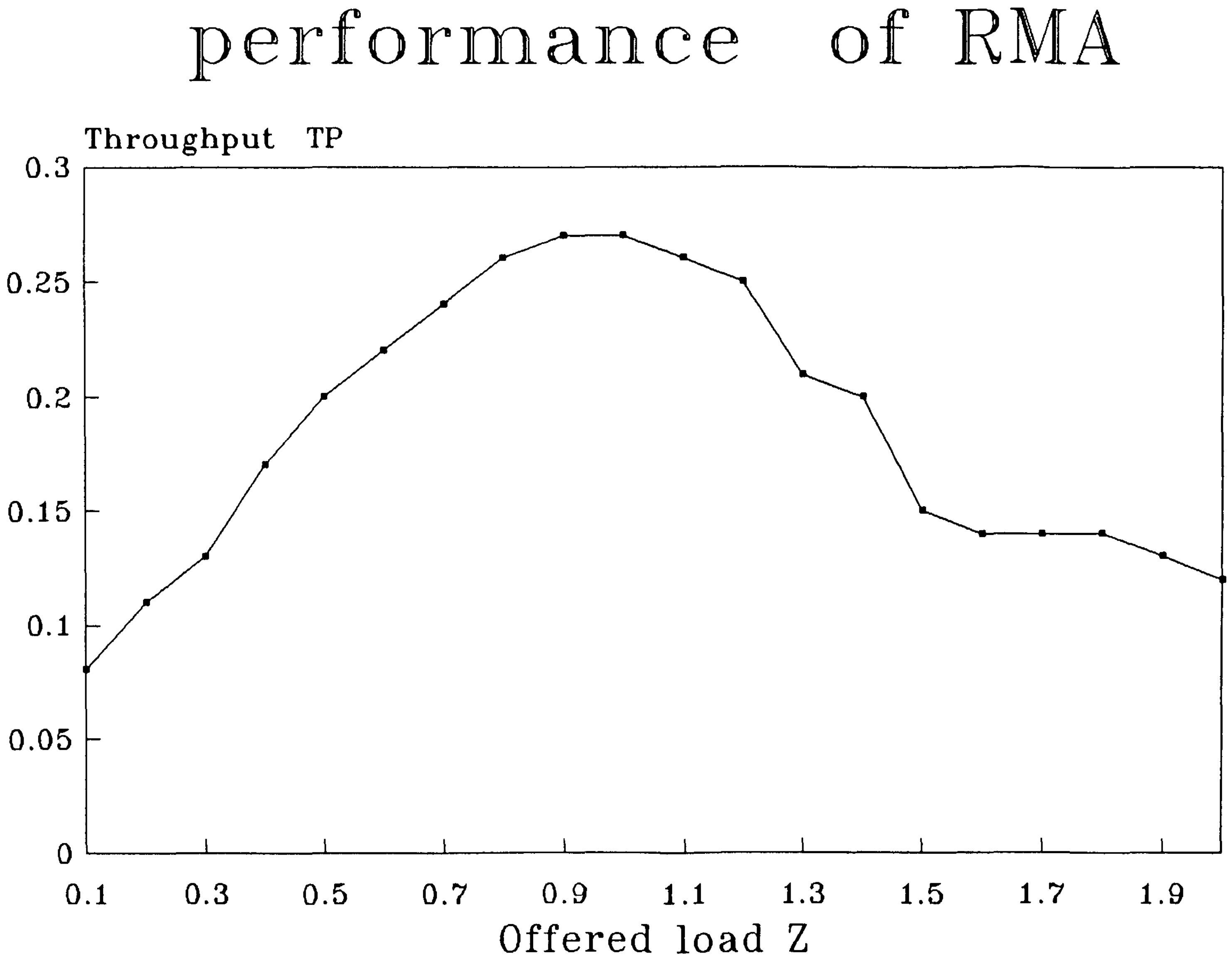
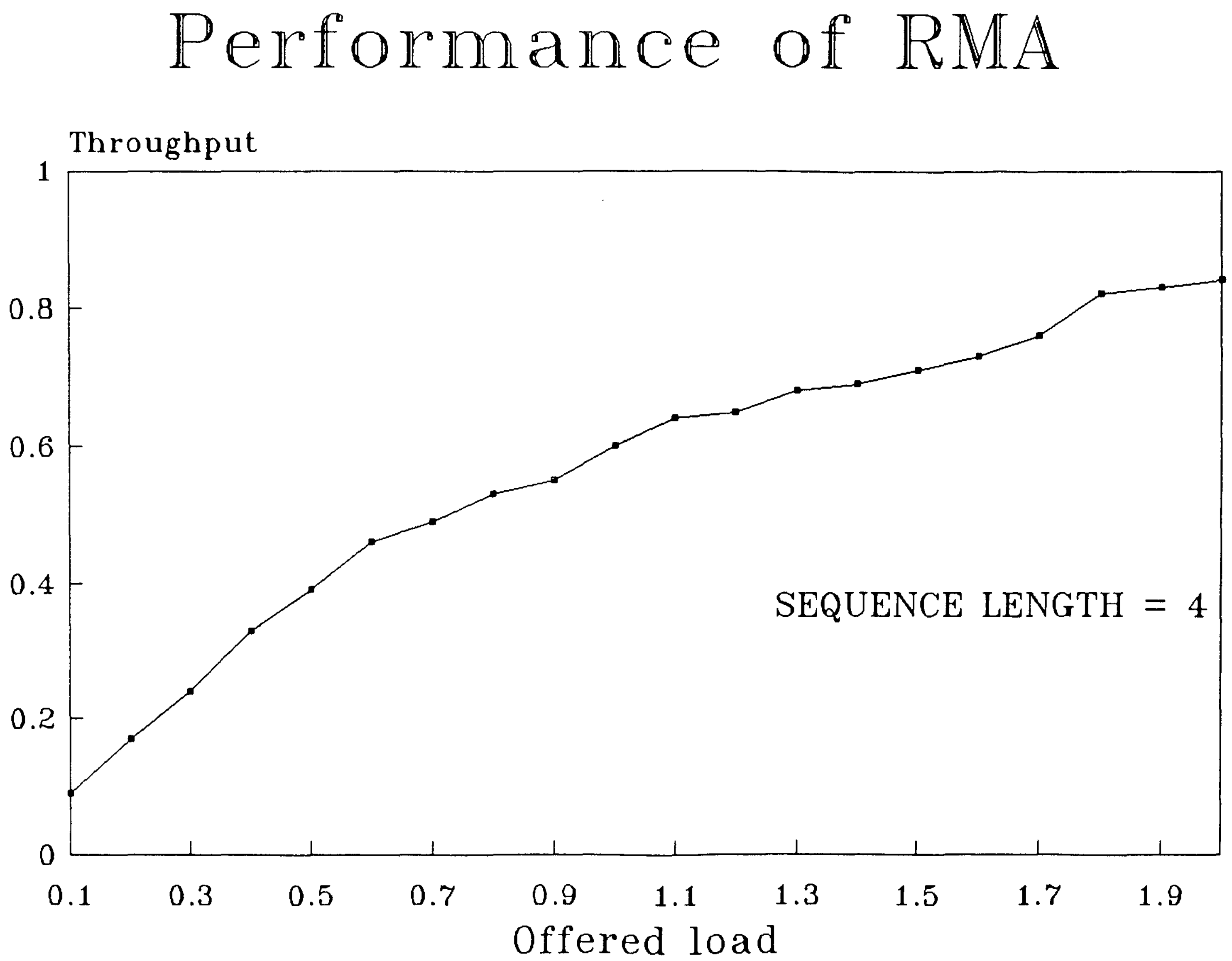

performance of RMA
Throughput TP
0.3
0.25
0.2
0.15
0.1
0.05
0
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
Offered load Z

FIG 4.a For n = 3



Performance of RMA
Throughput
1
0.8
0.6
0.4
0.2
0
SEQUENCE LENGTH = 4
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
Offered load

FIG 4.b

Performance of RMA

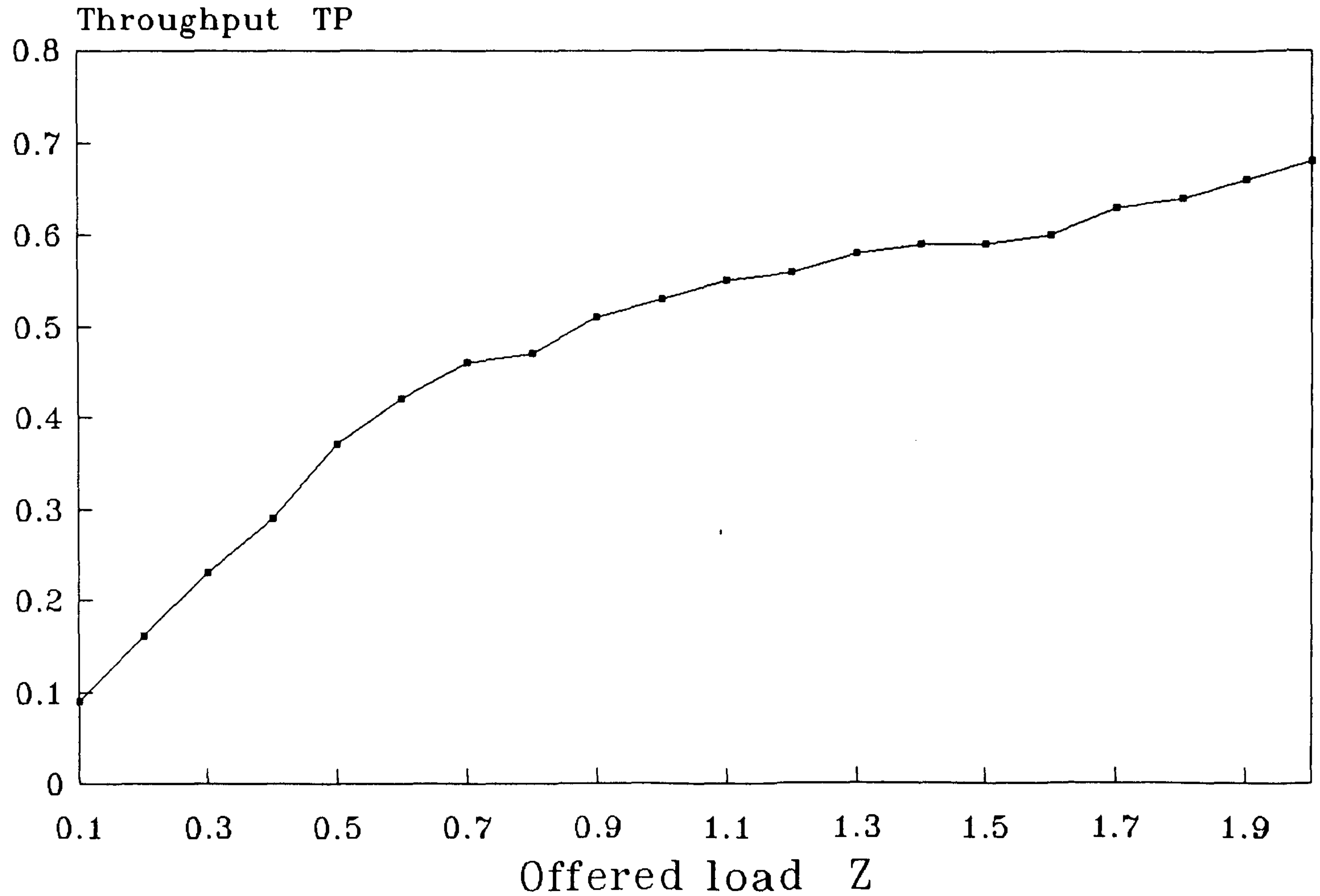


FIG 4.c

For n = 4 , Packets 120

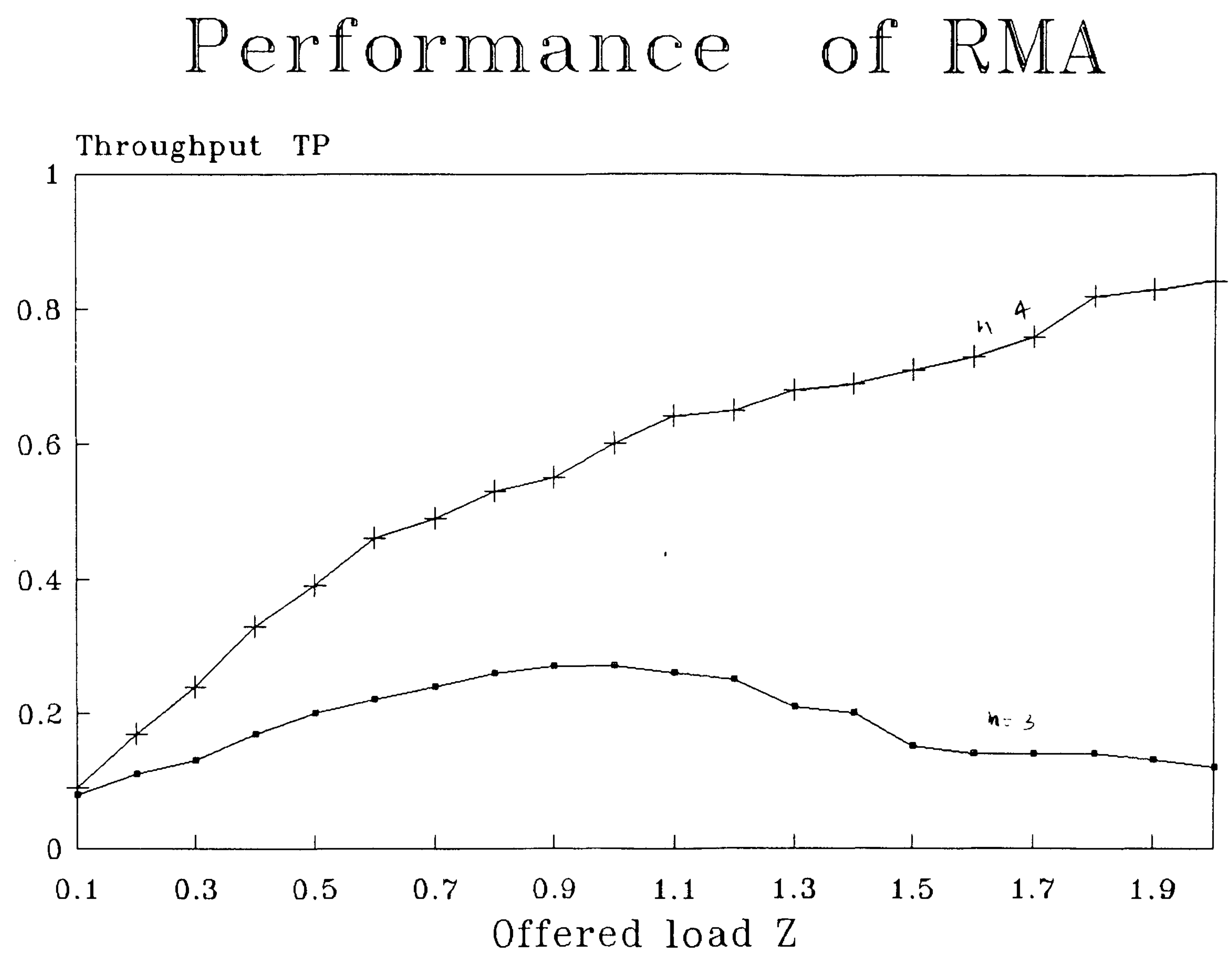
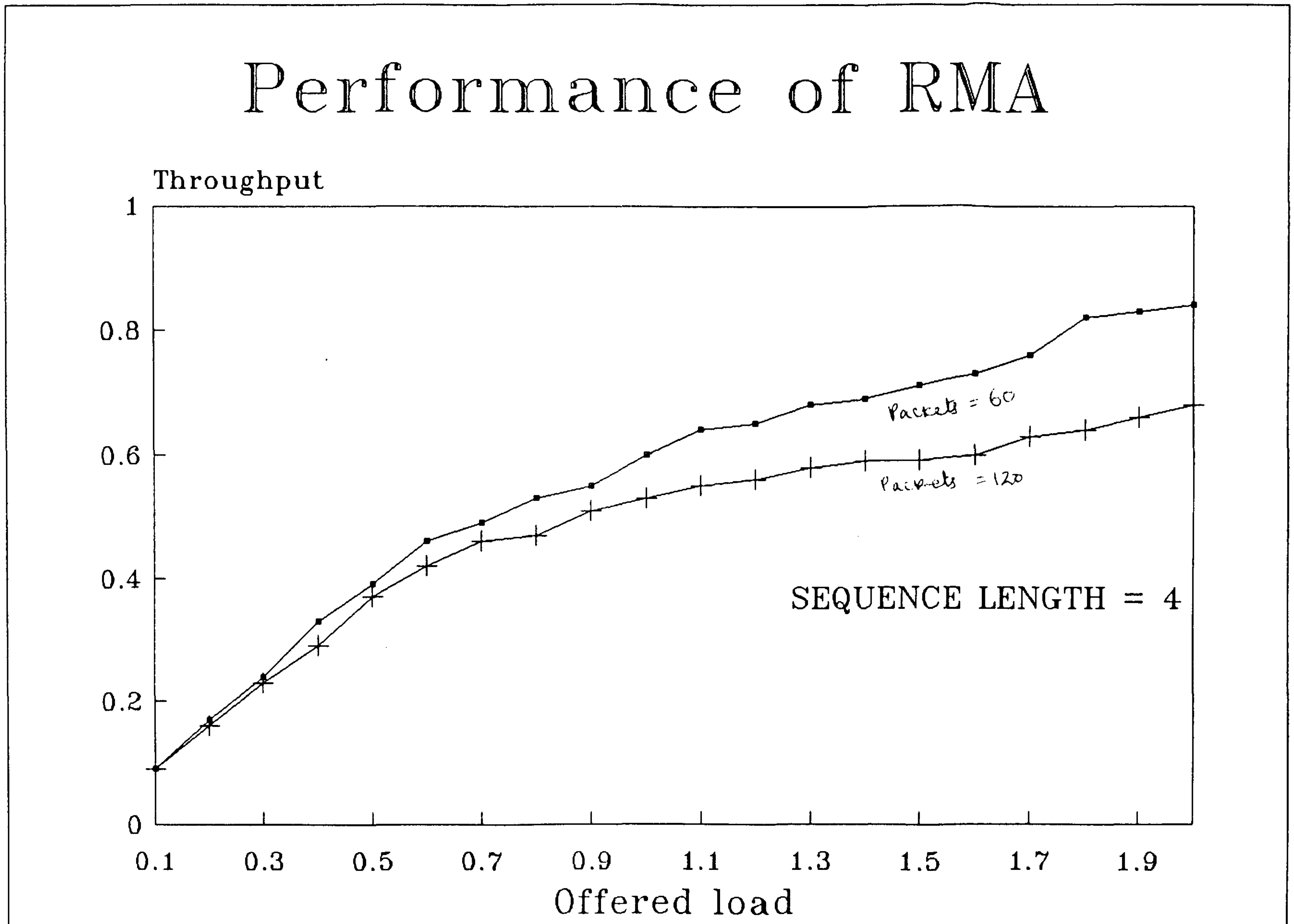

Performance of RMA
Throughput TP
1
0.8
0.6
0.4
0.2
0
n 4
n=3
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
Offered load Z

FIG 4.d


Performance of RMA
Throughput
1
0.8
0.6
0.4
0.2
0
0.1
0.3
0.5
0.7
0.9
1.1
1.3
1.5
1.7
1.9
Offered load
Packets = 60
Packets = 120
SEQUENCE LENGTH = 4
FIG 4.e

CHAPTER 5

CONCLUSION

Using the **RMA** codes sequences, the simulation is carried out and performance evaluation of the system is done by calculating the channel throughput. The simulation programs in written in Pascal.

5.1 RMA **code generation**

We generated **RMA** code for sequence length 3 and 4. The **RMA** code sequences which are attached to this work tally with the given one for sequence length 4 [5]. The results for n=3 also tally.

5.2 Simulation

This is the main task of this work. We calculated channel throughput for varying offered load. We get the results (chapter 4). We see that results are better than Pure ALOHA[4]. In this work channel throughput for n=3,4 at varying load is obtained and graphs are also drawn between throughput and offered load (Fig.4.a). We have taken the above for the different packet numbers i.e. packet number 60 and 120. It is observed from these graphs that the varying load with a graceful degradation for high value of traffic rate. We see that as the number of packet increases the number of clashes increases. As the number of packet increase the throughput decreases (Fig.4.e). As the offered load values increases throughput increases. We see that curve does not change the slope from positive to negative (Fig.4.b and 4.c). As the sequence length increases number of clashes increases and throughput decreases (Fig.4.a, Fig.4.b).

Conclusion

The **RMA** CODE algorithm which is given by W.W.Wu has great potential in improvement the satellite communication application where it can be entertained by large number of users. For sequence length four, we get 336 coded seqeunces (Chapter 4).

Hence, we tried to obtain the channel throughput as a performance measure at varying traffic load and different packet numbers and also at different sequence length.

We can implement this to VLSI and transistor level simulation can be done.

REFERENCES

1. Kuriacose Joseph and Dipankar Raychaudhari "Simulation Models for Performance Evaluation of Satellite multiple Access Protcols" IEEE JSAC Vol.6, No.1, January, 1988.

2. Wu et al. "Coding for Satellite Communication IEEE JSAC, Vol. SAC.5, No.4, May 1987.

3. D.Raychaudhari, "Performance Analysis of Random Access Packet Switched Code Division Multiple Access System". IEEE Trans Com. Vol. Com.29 June, 1981.

4. Andrew S.Tanenbaum, Computer Networks Prentice Hall of India Private Limited, New Delhi-1991.

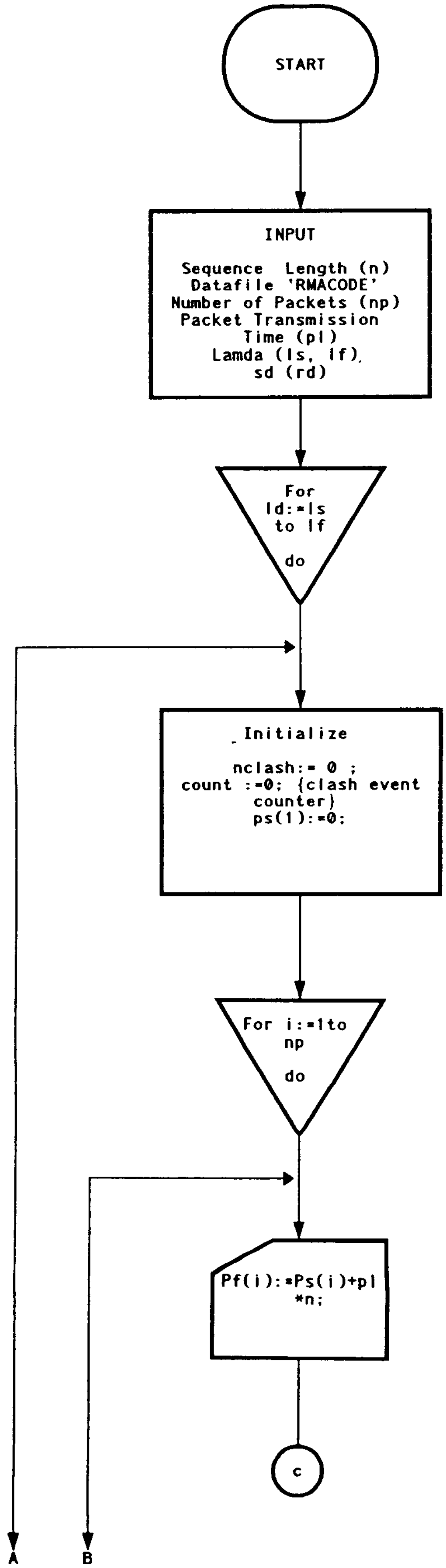
5. W.W.Wu "Element of Digital Satellite Communications Vol.II Channel Coding Integrated Services Digital Satellite Networks". Rockville, M.D. Computed Science, 1985.

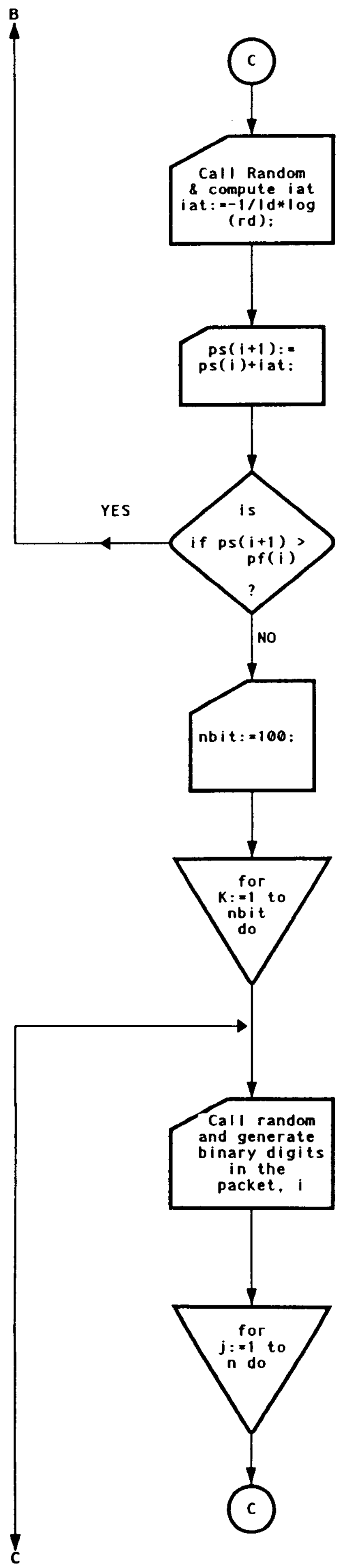
6. I.J.Nagrath and M.Gopal, "System Modelling and Analysis" Tata McGraw Hill, New Delhi.

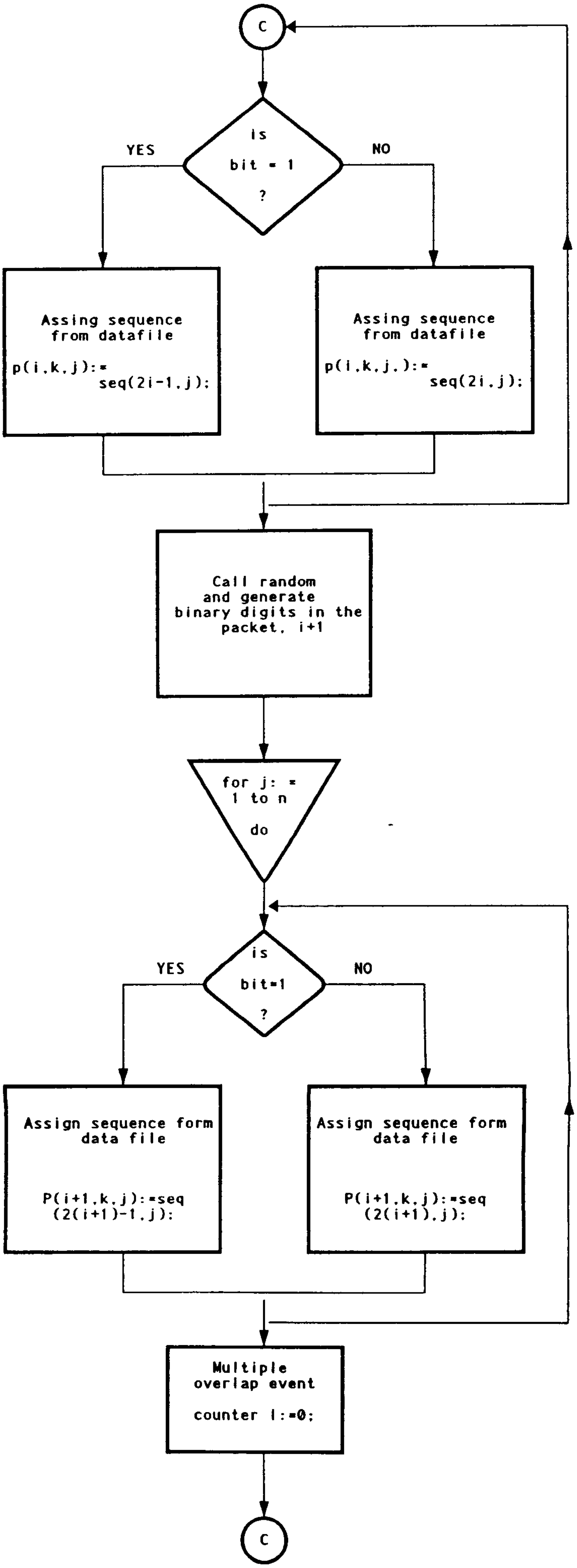
7. Pierre A. Humblet, Rajiv Ramaswami, An Efficient Communication Protocl for High Speed Packet Switched Multichannel Networks, IEEE JSAC, Vol. II, No.4, May 1993.

APPENDIX

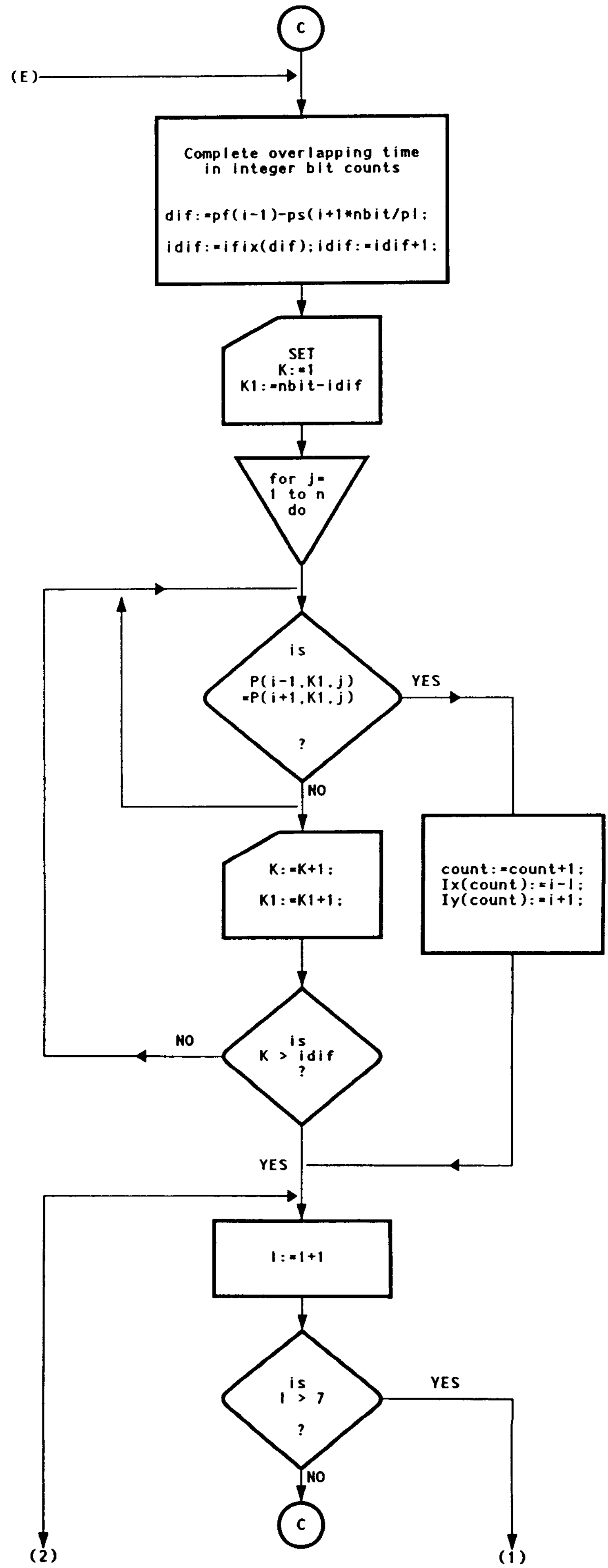
FLOW CHART FOR RMA SIMULATION PROGRAM

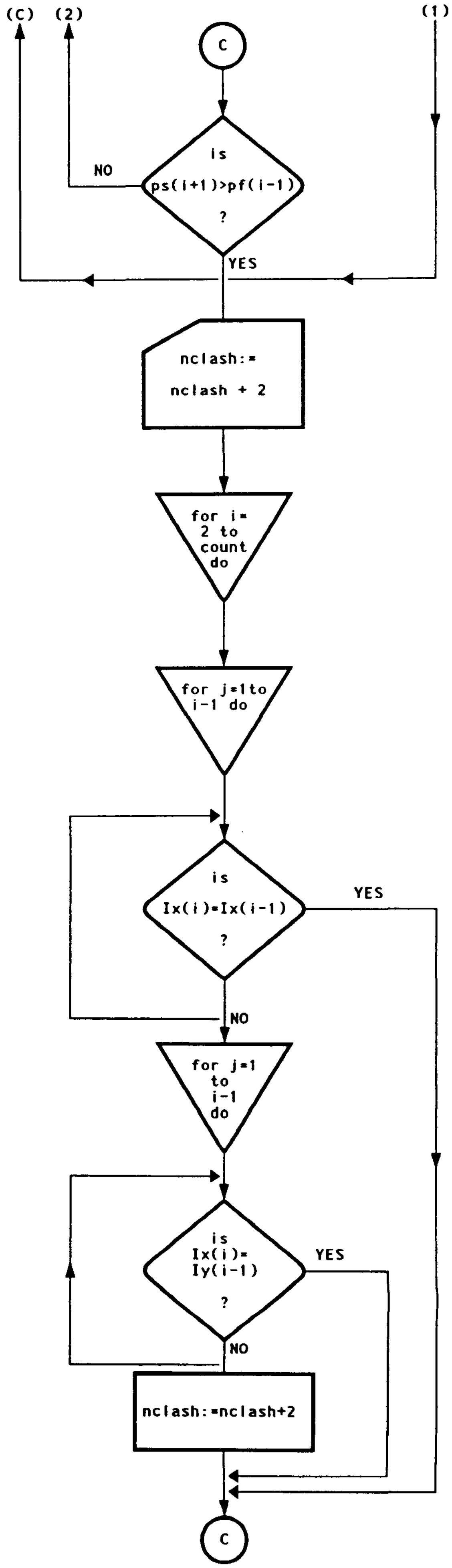



B
C
Call Random
& compute iat
iat:=-1/ld*log
(rd);
ps(i+1):=
ps(i)+iat;
YES
is
if ps(i+1) >
pf(i)
?
NO
nbit:=100;
for
K:=1 to
nbit
do
Call random
and generate
binary digits
in the
packet, i
for
j:=1 to
n do
C
C

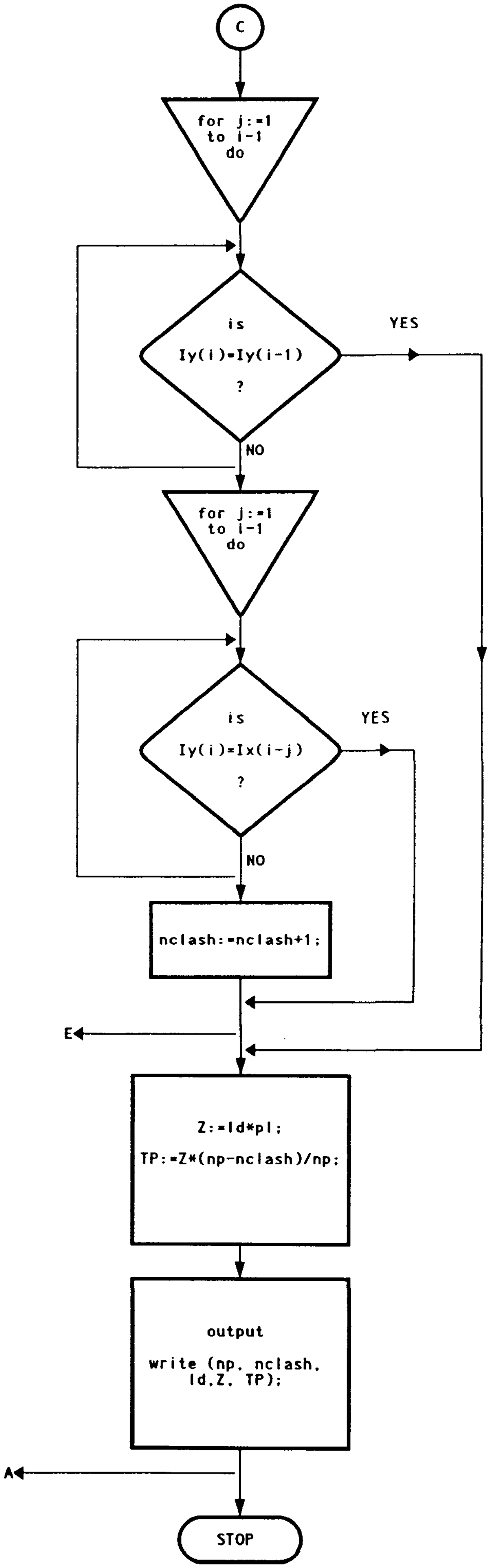


C
is
bit = 1
?
YES
NO
Assing sequence
from datafile
p(i,k,j):=
seq(2i-1,j);
Assing sequence
from datafile
p(i,k,j,):=
seq(2i,j);
Call random
and generate
binary digits in the
packet, i+1
for j: =
1 to n
do
is
bit=1
?
YES
NO
Assign sequence form
data file
P(i+1,k,j):=seq
(2(i+1)-1,j):
Assign sequence form
data file
P(i+1,k,j):=seq
(2(i+1),j):
Multiple
overlap event
counter l:=0;
C


C
(E)
Complete overlapping time
in integer bit counts
dif:=pf(i-1)-ps(i+1*nbit/pl;
idif:=ifix(dif);idif:=idif+1;
SET
K:=1
K1:=nbit-idif
for j=
1 to n
do
is
P(i-1,K1,j)
=P(i+1,K1,j)
?
YES
NO
K:=K+1;
K1:=K1+1;
count:=count+1;
Ix(count):=i-l;
Iy(count):=i+1;
NO
is
K > idif
?
YES
l:=l+1
is
l > 7
?
YES
NO
C
(2)
(1)

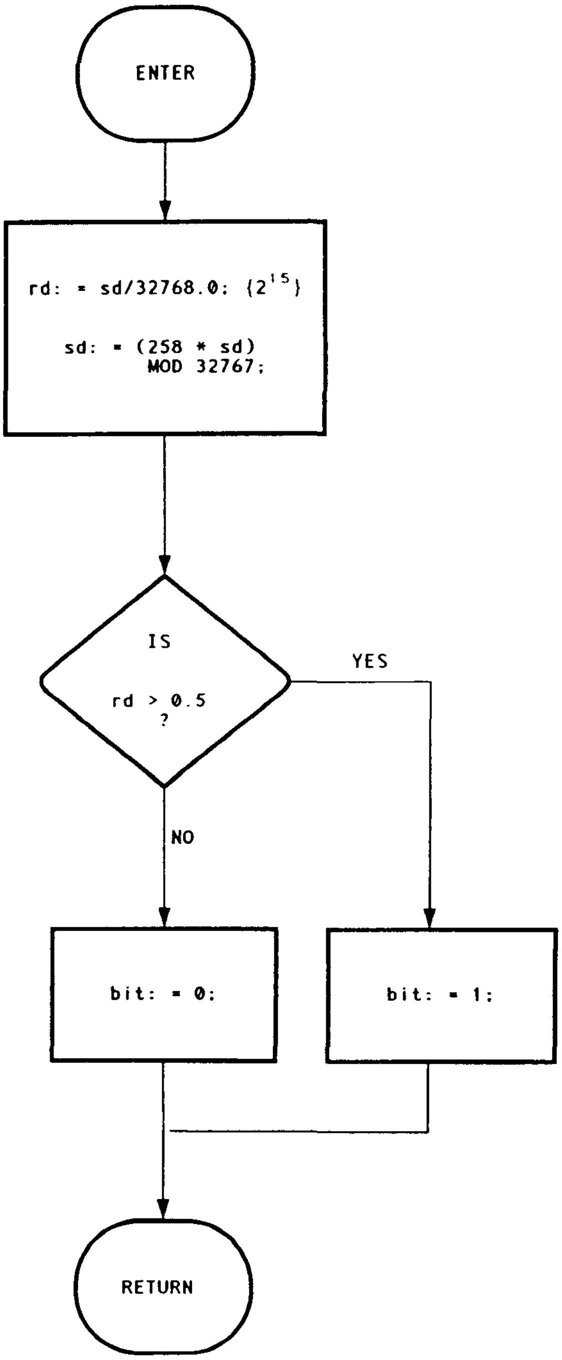


(C)
(2)
(1)
C
is
ps(i+1)>pf(i-1)
?
NO
YES
nclash:=
nclash + 2
for i=
2 to
count
do
for j=1to
i-1 do
is
Ix(i)=Ix(i-1)
?
YES
NO
for j=1
to
i-1
do
is
Ix(i)=
Iy(i-1)
?
YES
NO
nclash:=nclash+2
C



C
for j:=1
to i-1
do
is
Iy(i)=Iy(i-1)
?
YES
NO
for j:=1
to i-1
do
is
Iy(i)=Ix(i-j)
?
YES
NO
nclash:=nclash+1;
E
Z:=ld*pl;
TP:=Z*(np-nclash)/np;
output
write (np, nclash,
ld,Z, TP);
A
STOP

PROCEDURE RANDOM



```
{ This program  generate RMA coded sequences, which

 are  to be used further for the simulation purpose. }

Program RMACODE(input,output,fout);

const n=4;   (* sequence lenth *)
      d0=1 ; d1=3 ; d2=4 ; d3=12 ; (* these are euclidean distances *)


var i,j,p,k,s  :integer;
    q,r,m1,t,v :integer;
    w0         :array [0.. (n*n+n-1),0..n-1] of integer;
    m          :array[0..n*n-1,0..n-1] of integer;
    b          :array[0..n*n+n-1,0..n-1] of integer;
    a,a1,b1,u  :array[0..n-1,0..n] of integer;
    fout       :text;


 procedure order(s,p,q,r:integer);

 begin
      u[0,0] :=b[s,0];  u[1,0] :=b[p,0];
      u[2,0] :=b[q,0];  u[3,0] :=b[r,0];
      u[0,1] :=b[r,1];  u[1,1] :=b[s,1];
      u[2,1] :=b[p,1];  u[3,1] :=b[q,1];
      u[0,2] :=b[q,2];  u[1,2] :=b[r,2];
      u[2,2] :=b[s,2];  u[3,2] :=b[p,2];
      u[0,3] :=b[p,3];  u[1,3] :=b[q,3];
      u[2,3] :=b[r,3];  u[3,3] :=b[s,3];
 end;


begin
  assign(fout,'rmacode');
  rewrite(fout);
  w0[0,0]:=d0;  w0[0,1]:=d1;
  w0[0,2]:=d2;  w0[0,3]:=d3;
  writeln(fout,'RMACODE SEQUENCES ARE AS FOLLOWS');
  writeln(fout,'w0');
  writeln(fout,'-----------------------------------');

(*-------------To generate W0 matrix-------------------*)

  for i:=1 to (n*n-1) do
  begin
    writeln(fout);
     for j:=0 to (n-1) do
      begin
```

```
      w0[i,j] :=(w0[0,j]+i) mod (n*n-1);
      write(fout,w0[i-1,j],'   ');
     end;
   end;

    writeln(fout); writeln(fout);
    writeln(fout); writeln(fout);


(*------------- To generate the W1 matrix---------------*)

    writeln(fout,'w1');
    writeln(fout,'----------------------------------');
    for i :=(n*n-1) to (n*n+n-1) do
    w0[i,n-1] :=(n*n-1);
    p:=0;
    for i:=(n*n-1) to (n*n+n-1) do
     begin
       for j:=0 to n-2 do
        begin
          w0[i,j] := (j*(n+1)+p) mod(n*n-1);
          write(fout,w0[i,j],'   ');
        end;
          writeln (fout,w0[i,n-1]);
          p:=p+1;
     end;

(*----------------To construct M matrix-----------------*)

      writeln(fout);
      writeln(fout,'M');
      writeln(fout,'-------------------------------');

      for i:=0 to n-1 do
       m[0,i]:=i;

       for i:=1 to (n*n-1) do
        begin
          for j:=0 to n-1 do
           begin
             m[i,j]:=m[i-1,j]+n;
             write(fout,m[i-1,j],'   ');
           end;
             writeln(fout);
        end;

 (*-----To get sequences from the combination of above-----------*)
```

```
for t:=0 to (n*n+n-1) do
 begin
    writeln(fout,'B',t,' =');
    for v:=0 to n-1 do
     begin
       for j:=0 to (n-1)  do
        begin
          b[v,j] :=n*w0[t,j]+v;
          write(fout,b[v,j],'   ');
        end;
          writeln(fout);
     end;


   for m1:=1 to n-1 do
    begin
        case m1 of
             1  : begin
                   s:=0; p:=3;
                   q:=2; r:=1;
                   order(s,p,q,r);
                  end;

             2  : begin
                   s:=0; p:=2;
                   q:=1; r:=3;
                   order(s,p,q,r);
                  end;

             3  : begin
                   s:=0; p:=1;
                   q:=3; r:=2;
                   order(s,p,q,r);
                  end;

            end;   (*end of case*)


  writeln(fout,'U(',t,',',m1,')');

  for i:=0 to n-1 do
   begin
     for j:=0 to n-1 do
     begin
       write(fout,u[i,j],'  ');
     end;
       writeln(fout);
   end;
end;
```

```
end;

   writeln(fout);


(*--------To construct the permutation matrices------------------*)


  k:=1;
  if k=1 then
   begin
    writeln(fout,'A',k);
     for i:=0 to n-1 do
     begin
      a[0,i] :=i;
      a1[0,i]:=i;
     end;


      for i:=1 to n-1 do
       begin
         for j:=0 to n-1 do
         a[i,j+1]:=a[i-1,j];
         a[i,0]  :=a[i-1,n-1];
       end;
   end;


 for i:=0 to n-1 do
  begin
     for j:=0 to n-1 do
     write (fout,a[i,j],' ');
     writeln(fout);
  end;



 writeln(fout);

   for k:=2 to n-1 do
    begin
     writeln(fout,'A',k);
      begin
        i:=0;
        for  j:=1 to n-2 do
        a[i,j+1]:=a1[0,j];
        a[i,0]  :=0;
        a[i,1]  := n-k+1;
        for  j:=0 to n-1 do
```

```
    a1[0,j] :=a[0,j];
   end;

   for i:=1 to n-1 do
    begin
     for j:=0 to n-1 do
      a[i,j+1]:= a[i-1,j];

      a[i,0]  := a[i-1,n-1];
    end;

      for i:=0 to n-1 do
       begin
        for j:=0 to n-1 do
         write (fout,a[i,j],'  ');
         writeln(fout);
       end;
  end;

(*------------------------------------------------------*)

 close(fout);

end.
```