FUTURE DEPENDENT EVENTS, AS SHOWN BY THE EXAMPLE OF SLAM

Heimo H. Adelsberger
Institute of Statistics
University of Economics
Franz-Klein-Gasse 1
A-1190, Vienna, Austria

ABSTRACT

Random events in discrete simulation models are classified in ordinary and future dependent events. An event occurring between t_o and T is classified as an ordinary event, if the probability distribution for the realization in time is known at time t_o. If changes of the system after time t_o can affect the probability distribution of an event, it is called a future dependent event. To show the difference we give two examples.

In contrary to ordinary events future dependent events are not well supported by the currently most used languages like GPSS, SIMULA, SIMSCRIPT and SLAM. In taking SLAM as an example, we are going to demonstrate, how to adapt a simulation language to be able to handle future dependent events with as slight an effort as possible.

1. Introduction

One of the aims of simulation languages is to support and to reduce the effort in programming a simulation project.

The user should concentrate on the formulation of the model and concentrate on the crucial points of the problem.

A user should typically not be involved into technical details.

To exemplify this point we regard a typical queuing problem: The main points are

capacity
initial number of entities in the queue
balking / blocking
ranking criterion

A sufficient simulation language should be able to describe it in a simple and adequate way.

But the user should not be bothered with details like

storage allocation
searching algorithm for entities in a file
manipulation of pointers

In this paper we will focus on discrete event simulation.

One of the more important fields, where the user should be supported as much as possible are the manipulations in connection with the event calendar.

Currently, the most used languages for discrete simulation, GPSS, SIMULA, SIMSCRIPT and SLAM, seem to fulfill this requirement in a sufficient way.

We say that this is only the case for "ordinary" events but not for "future dependent" events, which both will be defined in the next section.

1.1 "Ordinary" events

Consider a typical event E, which takes place at time t. It has been caused by another event F at time t_o ($t_o \leq t$), and the last possible time of the realization shall be T.

Example: The end-of-service event is caused by the begin-of-service event.

Random events of the type, that t lies somewhere between t_o and T can be described by a

density function	$f(t)$
distribution function	$F(t) = \int_{t_o}^{t} f(u)du$
hazard rate	$r(t) = \frac{f(t)}{1-F(t)}$

Now we want to define formally ordinary events:

Let $G = \{S_{t_1}, S_{t_2}, \ldots, S_{t_k}\}$ with $t_i \leq T$

be the set of states of the system from beginning to time T. Generally the distribution of the realization of event E can be regarded as a function of S, where S is a subset of G:

$F(t,S) \quad t_0 \leq t \leq T \quad \text{and} \quad S \subseteq G$

Ordinary events are special events for which

$S \{S_i \mid i \leq t_0\}$

holds, i.e. that the distribution F(t,S) depends only on states of the system until time t_0, the first possible outcome of event E.

This definition implies also the important case, that the event only depends on the state of the system at time t_0.

Ordinary events are well supported in simulation languages like GPSS, SIMSCRIPT, SIMULA and SLAM.

In Fig. 1 we list typical statements for scheduling purposes. These statements refer all to ordinary events.

GPSS:	ADVANCE 17,8
SIMSCRIPT:	SCHEDULE A DEPARTURE GIVEN CUSTOMER IN UNIFORM.F(10.,25.,1) MINUTES
SIMULA:	ACTIVATE NEW PASSENGER DELAY NEGEXP (2,U2)
SLAM:	TIM=EXPON(5.,1) CALL SCHDL(1,TIM,A)

Fig.1

1.2 "Future dependent" events

If the probability distribution for t is allowed to depend on states of the system after time t_0, we will call these events "future dependent events".

Or formally:

$F(t,S) \quad t_0 \leq t \leq T; \quad S \subseteq G$

and $\exists S_i \in S$ with $i > t_0$

2. Examples

2.1 Example A: Adjustable machines

Assume two machines M1 and M2 working in a parallel production line. If one machine fails, the repair work starts immediatly. In the meantime the other machine produces with higher capacity to make up for the loss of total production, but with the risk of a higher failure rate. The failure- and repair processes are assumed to be Poisson distributed.

The possible states of one machine and the corresponding rates are

$M = \{L,H,F\} \quad (a,b,c)$

where

L	Production with a low failure rate; rate a
H	Production with a high failure rate; rate b
F	Failure; "repair" rate c

The possible states for the system are

$\{(L,L),(H,F),(F,F)\}$

(Remark: The states (H,F) and (F,H) do not have to be distinguished.)

The transition probabilities are

$$\begin{tikzcd}(L,L) \arrow[r, shift left, "2a\,dt"] & (H,F) \arrow[l, shift left, "c\,dt"] \arrow[r, shift left, "b\,dt"] & (F,F) \arrow[l, shift left, "2c\,dt"]\end{tikzcd}$$

and the transition matrix is

$$P = \begin{pmatrix} 1 - 2a\,dt & 2a\,dt & 0 \\ c\,dt & 1 - (b+c)dt & b\,dt \\ 0 & 2c\,dt & 1 - 2c\,dt \end{pmatrix}$$

The limiting probability distribution of this Markov chain (i.e. the eigenvector of P' corresponding to the eigenvalue 1) is

$$\begin{array}{ccc} \dfrac{c^2}{c^2+2ac+ab} & \dfrac{2ac}{c^2+2ac+ab} & \dfrac{ab}{c^2+2ac+ab} \\ (L,L) & (H,F) & (F,F) \end{array}$$

2.2 Example B: Non-adjustable machines

As before both machines of example A are allowed to be in states L,H and F with the rates a,b and c.

Now one machine remains without change in state L or H until it fails. When the machine takes up the production after repair, the state of the other machine determines the new state of the machine:

L if the other machine works
H if the other machine fails

This set up implies, that adjustments can only be done at the starting time of the machines.

Therefore we have five states:

$\{(L,L), (L,H), (L,F), (H,F), (F,F)\}$

with the following transition probabilities (the factor dt is omitted, since it does not effect the eigenvector):

$$\begin{tikzcd}
(L,L) \arrow[d, shift right, "2a"'] & (L,H) \arrow[dl, "b"'] \arrow[d, shift right, "c"'] \\
(L,F) \arrow[u, shift right, "c"'] \arrow[d, "a"'] & (H,F) \arrow[u, shift right, "a"'] \arrow[dl, shift left, "b"] \\
(F,F) \arrow[ur, shift left, "2c"] &
\end{tikzcd}$$

This implies the transition matrix

$$P = \begin{pmatrix} 1-2a & 0 & 2a & 0 & 0 \\ 0 & 1-a-b & b & a & a \\ c & 0 & 1-a-c & 0 & a \\ 0 & c & 0 & 1-b-c & b \\ 0 & 0 & 0 & 2c & 1-2c \end{pmatrix}$$

The limiting distribution is given by

(L,L)	bc^3/D
(L,H)	$2a^2c^2/D$
(L,F)	$2abc^2/D$
(H,F)	$2a^2c(a+b)/D$
(F,F)	$a^2b(a+b+c)/D$

with $D=bc^3+2a^2c^2+2abc^2+2a^2c(a+b)+a^2b(a+b+c)$.

2.3 Comments on examples A and B

Example B can easily be programmed with discrete simulation languages, but this is not valid for example A. The reason is, that both events in example B (begin-of-repair, end-of-repair) are ordinary events. In example A, however, the begin-of-repair event is a future dependent event.

To program example A some tricks are needed by which the lack of a conceptual framework for future events in these languages manifests itself.

The structures of the above mentioned languages are in principle the same. Therefore it is possible to concentrate only to the language SLAM.

SLAM being at the disposal as a FORTRAN program, it gives the advantage to the user to insert extensions into the language without any difficulties.

In section 3 we give the SLAM program for example B and compare the results of a simulation run with the theoretical results.

In section 4 we will extend the "next event logic" of SLAM for future dependent events.

Finally we show how to program example A for this extended version of SLAM.

3. The SLAM program for example B

3.1 Definitions

Events and attributes:

BEGREP is the event no. 1 (begin-of-repair)
ENDREP is the event no. 2 (end-of-repair)

We use three attributes:

ATRIB(1) number of machine (1 or 2)
ATRIB(2) event time of latest ENDREP event
ATRIB(3) event time of latest BEGREP event

The following variables are employed in the simulation:

Variable	Definition	Initial Value
XX(1)	state of machine M1 0 M1 works 1 M1 out of order	0.
XX(2)	state of machine M2	
XX(3)	state "both machines work" 1 XX(1)=0. and XX(2)=0. 0 else	1.
XX(4)	state "one works, one out of order" 1 XX(1)=0. and XX(2)=1. or v.v. 0 else	0.
XX(5)	state "both out of order" 1 XX(1)=1. and XX(2)=1. 0 else	0.
NR	number of broken machines (0 or 1 or 2)	0
XMY(1)	the expected length of a working periode in state L XMY(1)=1/a	
XMY(2)	the expected length of a working periode in state H XMY(2)=1/b	
REPTIM	the expected length of a repair periode REPTIM=1/c	

We define four variables for statistical purposes

STAT,1	"M1 WORKING" the effective length of a working periode for machine M1
STAT,2	"M2 WORKING"
STAT,3	"M1 REPAIR" the effective length of a repair periode for machine M1
STAT,4	"M2 REPAIR"

3.2 The SLAM input statements

The SLAM input statements are given below.

```
GEN,ADELSBERGER,2 MACHINES B,4/15/1981,1 ;
LIM,1,3,50;
INIT,0,100000;
STAT,1,M1 WORKING;
STAT,2,M2 WORKING;
STAT,3,M1 BROKEN;
STAT,4,M2 BROKEN;
TIMST,XX(1),STATE M1;
TIMST,XX(2),STATE M2;
TIMST,XX(3),STATE M00;
TIMST,XX(4),STATE M01;
TIMST,XX(5),STATE M11;
FIN;
```

3.3 The FORTRAN subroutines

The FORTRAN subroutines EVENT, BEGREP and ENDREP are given below.

```
      SUBROUTINE EVENT(I)
      GOTO(1,2)I
1     CALL BEGREP
      RETURN
2     CALL ENDREP
      RETURN
      END

      SUBROUTINE BEGREP
      DIMENSION NSET(100)
      COMMON/SCOM 1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
C
C EVENT 1: BEGIN-OF-REPAIR
C
C INCREASE NUMBER OF BROKEN MACHINES
C
      NR=NR+1
C
C SET SYSTEM VARIABLES
C
      XX(MACHNR)=1
      CALL XXX(XX)
C
C SAVE EVENT TIME
C
      ATRIB(3)=TNOW
C
C COLLECT STATISTICS OF LENGTH OF WORKING PERIODE
C
      CALL COLCT(TNOW-ATRIB(2),MACHNR)
C
C SCHEDULE END-OF REPAIR EVENT
C
      TIM=EXPON(REPTIM,2)
      CALL SCHDL(2,TIM,ATRIB)
      RETURN
      END

      SUBROUTINE ENDREP
      DIMENSION NSET(1000)
      COMMON/SCOM1/ ATRIB(100)DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
C
C EVENT 2: END-OF-REPAIR
C
C DECREASE NUMBER OF BROKEN MACHINES
C
      NR=NR-1
C
C SET SYSTEM VARIABLES
C
      XX(MACHNR)=0
      CALL XXX(XX)
C
C SAVE EVENT TIME
C
      ATRIB(2)=TNOW
C
C COLLECT STATISTICS OF LENGTH OF REPAIR PERIODE
C
      CALL COLCT(TNOW-ATRIB(3),MACHNR+2)
C SCHEDULE BEGIN-OF-REPAIR EVENT
C
      TIM=EXPON(XMY(NR+1),1)
      CALL SCHDL(1,TIM,ATRIB)
      RETURN
      END
```

3.4 Results for example B

The limiting distribution for the parameters

$$(a,b,c)=(1./100., 2./100, 1./100.)$$

can be computed (see section 2.2) as

$(L,L)\cup(L,H)$	0.182
$(H,F)\cup(L,F)$	0.454
(F,F)	0.364

The results of a simulation run over a periode of 100000 time units is given by Fig.2.

The average values of variables M00, M01 and M11

0.202 / 0.432 / 0.366

correspond to the above given figures.

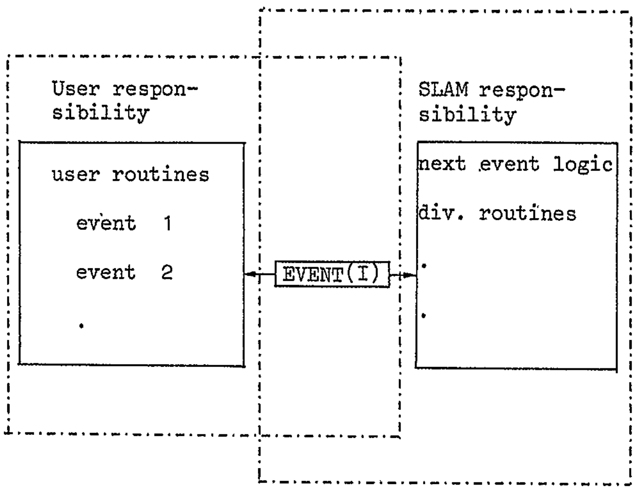
```
                                S L A M   S U M M A R Y   R E P O R T

                          SIMULATION PROJECT 2 MACHINES B              BY ADELSBERGER

                          DATE  4/15/1981                              RUN NUMBER     1 OF      1

                          CURRENT TIME   0.1000E+06
                          STATISTICAL ARRAYS CLEARED AT TIME  0.0

                          **STATISTICS FOR VARIABLES BASED ON OBSERVATION**

                  MEAN         STANDARD      COEFF. OF     MINIMUM       MAXIMUM       NUMBER OF
                  VALUE        DEVIATION     VARIATION     VALUE         VALUE         OBSERVATIONS

M1 WORKING     0.7111E+02    0.8031E+02    0.1129E+01    0.2305E+00    0.5624E+03        586
M2 WORKING     0.6913E+02    0.7597E+02    0.1099E+01    0.3906E-02    0.4576E+03        606
M1 BROKEN      0.9969E+02    0.1055E+03    0.1059E+01    0.1719E+00    0.8212E+03        585
M2 BROKEN      0.9600E+02    0.9591E+02    0.9990E+00    0.6250E-01    0.6520E+03        605

                          **STATISTICS FOR TIME-PERSISTENT VARIABLES**

                  MEAN         STANDARD      MINIMUM       MAXIMUM       TIME          CURRENT
                  VALUE        DEVIATION     VALUE         VALUE         INTERVAL      VALUE

STATE M1       0.5833E+00    0.490E+00     0.0           0.1000E+01    0.1000E+06    0.1000E+01
STATE M2       0.5811E+00    0.4934E+00    0.0           0.1000E+01    0.1000E+06    0.1000E+01
STATE M00      0.2021E+00    0.4016E+00    0.0           0.1000E+01    0.1000E+06    0.0
STATE M01      0.4315E+00    0.4953E+00    0.0           0.1000E+01    0.1000E+06    0.0
STATE M11      0.3664E+00    0.4818E+00    0.0           0.1000E+01    0.1000E+06    0.1000E+01

                          **FILE STATISTICS**

FILE     ASSOCIATED    AVERAGE     STANDARD     MAXIMUM     CURRENT     AVERAGE
NUMBER   NODE TYPE     LENGTH      DEVIATION    LENGTH      LENGTH      WAITING TIME

  1                    0.0         0.0          0           0           0.0
  2                    1.9998      0.0246       2           2           83.8834
```

Fig. 2

4. The extension of SLAM for simulating models with future dependent events

4.1 The basic concept of SLAM for discrete event simulation

First we will display the basic conception of SLAM



An analysis of the manipulations concerning the event calendar shows, that there are three components, where the first two are covered by SLAM

(1) the chronological ordering of the events in the event calendar

(2) at event time to execute the right user-written event subroutine

The third component, the

(3) scheduling of events

belongs to the user's responsibility. If future dependent events should be included in SLAM, this third component has to be managed by SLAM too.

4.2 Scheduling of future dependent events

The scheduling of future dependent events has to be done in special subroutines called

tevent1, tevent2, ... , teventn

These subroutines are called by subroutine TEVENT in the same manner as in subroutine EVENT the subroutines event1, event2, ...

The assumption thereby is, that time is TNOW and event i has not been realized until TNOW. Instead of the SLAM provided subroutine SCHDL the new subroutine TSCHDL has to be used; for ordinary events the new subroutine NSCHDL (See APPENDIX). The parameter list for these subroutines are the same as for SCHDL, but this subroutine itself must not be used anymore.

4.3 The extended next event logic

For the simulation of future dependent events we have to extend the standard next event logic of SLAM. Therefore we develop the subroutine EXNEL, which is called before leaving the user-written SLAM subroutine EVENT. (A small adjustment must be done to this subroutine.)

The flow-chart of the extended next event logic is shown in Fig.3, the source code in the appendix.

The extended next event logic uses the logical array LRISK(I). The user has to define, which events (future dependent events and ordinary events!) can cause system changes, so that the probability distribution for future dependent events in the event calendar can be influenced. This is done by setting the logical variable LRISK(I)= .TRUE.

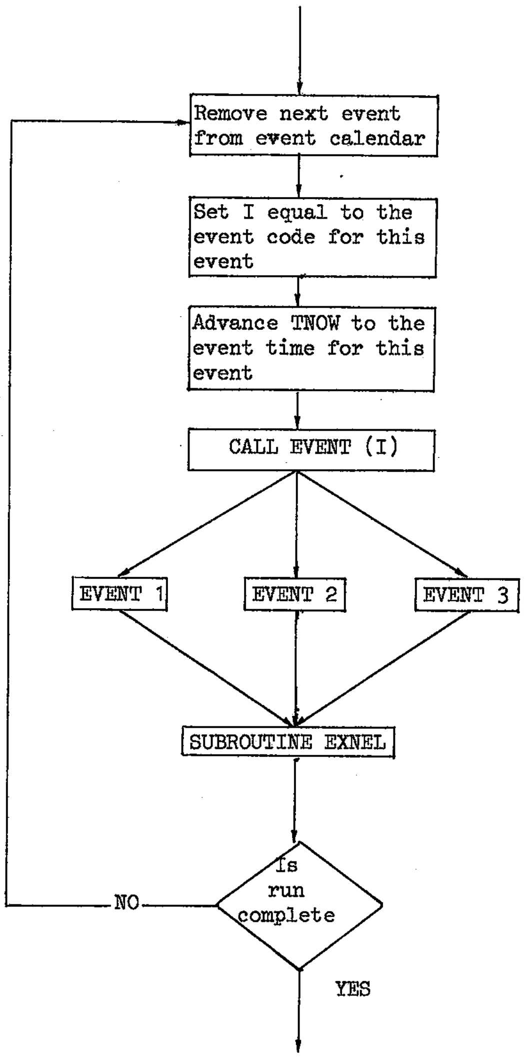

Remove next event from event calendar
Set I equal to the event code for this event
Advance TNOW to the event time for this event
CALL EVENT (I)
EVENT 1
EVENT 2
EVENT 3
SUBROUTINE EXNEL
Is run complete
NO
YES

Fig.3a

SUBROUTINE EXNEL

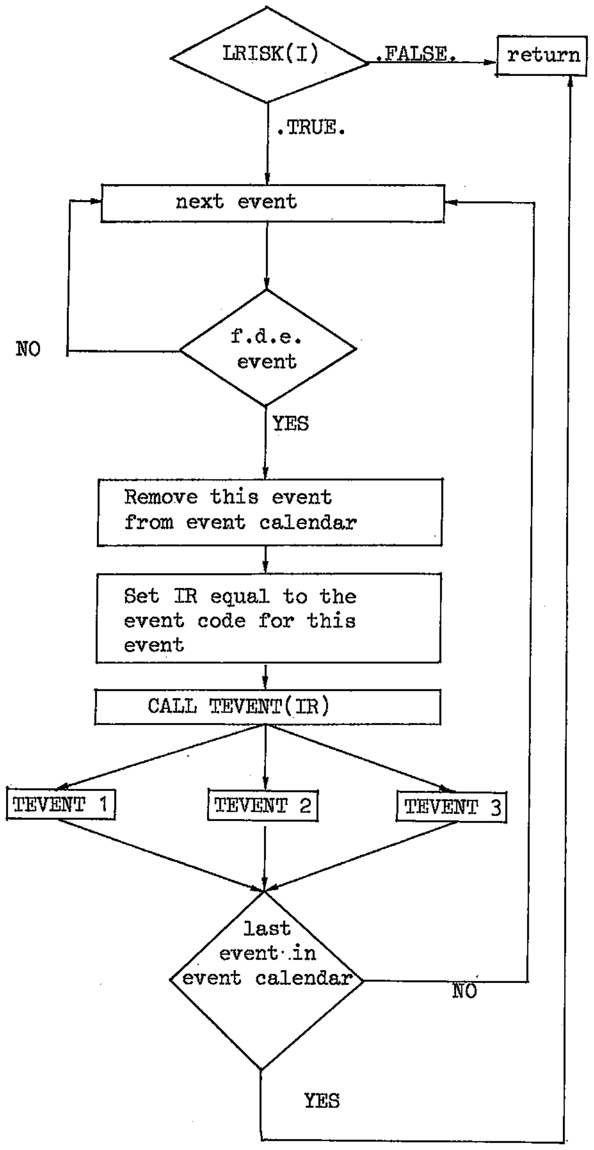
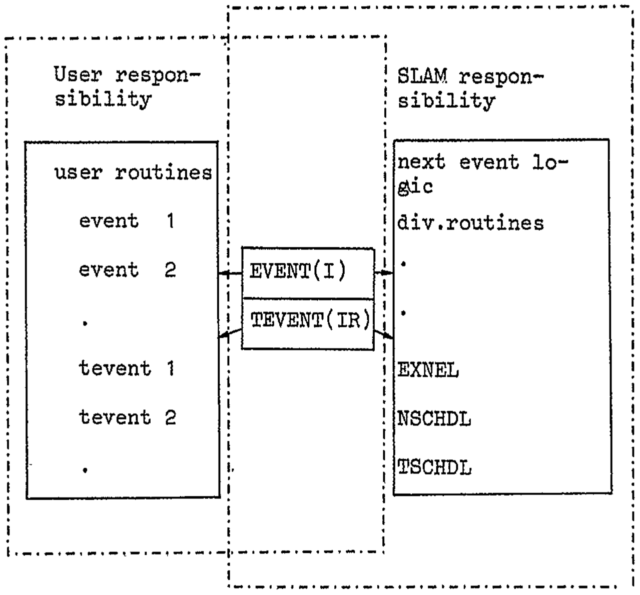

LRISK(I)
.FALSE.
return
.TRUE.
next event
NO
f.d.e. event
YES
Remove this event from event calendar
Set IR equal to the event code for this event
CALL TEVENT(IR)
TEVENT 1
TEVENT 2
TEVENT 3
last event in event calendar
NO
YES

Fig.3b

A graphical display of the extended concept is given below.



4.4 The extended concept from the user's point of view.

If an user has to model future dependent events, the following tasks have to be done:

(1) To write the subroutine TEVENT and all subroutines tevent 1, tevent 2, ...

(2) To adapt the subroutine EVENT (see example next section)

(3) To use TSCHDL for future dependent events and NSCHDL for ordinary events; the original SLAM subroutine SCHDL must not be used

(4) To set up the logical array LRISK (e.g. in subroutine INTLC)

(5) To increase the parameter MATR, the number of attributes, by one. This has to be done by the LIMITS statement. (ATRIB(MATR) is used in EXNEL, NSCHDL and RSCHDL.)

5. The SLAM program for exemple A

We use the same terms and expressions as in example B in section 3.

Event 1, the begin-of-repair event is a future dependent event, therefore TSCHDL must be used to schedule this event.

Event 2, the end-of-repair event is an ordinary event, therefore subroutine NSCHDL must be used.

Since both events have a feed-back on the probability distribution of event 1, LRISK(1) and LRISK(2) has to be provided as .TRUE.

Additionally MATR in the LIMITS statement must be set to four.

The rest of the program setup can be taken from example B.

5.1 The SLAM input statement

```
GEN,ADELSBERGER,2 MACHINES A,4/15/1981,1 ;
LIM,1,4,50;
INIT,0,100000;
STAT,1,M1 WORKING;
STAT,2,M2 WORKING;
STAT,3,M1 BROKEN;
STAT,3,M2 BROKEN;
TIMST,XX(1),STATE M1;
TIMST,XX(2),STATE MOO;
TIMST,XX(3),STATE MOO;
TIMST,XX(4),STATE MO1;
TIMST,XX(5),STATE M11;
FIN;
```

5.2 The FORTRAN subroutines

```
      DIMENSION NSET(1000)
      COMMON/SCOMI/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      NNSET=1000
      NCRDR=5
      NPRNT=6
      NTAPE=7
      CALL SLAM
      STOP
      END

      SUBROUTINE INTLC
      DIMENSION NSET(1000)
      COMMON/SCOMI1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100)?SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1)
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1).MACHNR)
      COMMON /ZCOMI/ LRISK
      LOGICAL LRISK(100)
C READ PARAMETER A,B,C  (--> XMY(1),XMY(2),REPTIM)
C
      READ(11,900) XMY,REPTIM
900   FORMAT(3F5.0)
      WRITE
910   FORMAT(' EXAMPLE A ',3F9.i)
C
C SET  INITIAL VALUES
C
      NR=0
      XX(1)=0
      XX(2)=0
      CALL XXX(XX)
C
C SCHEDULE BEGIN-OF-REPAIR EVENTS FOR BOTH MACHINES
C AND SET VARIABLES LRISK(1) AND LRISK(2)
C
      DO 10 MM=1,2
      ATRIB(2)=TNOW
      MACHNR=MM
      LRISK(MM)=.TRUE.
      CALL BEGRT
10    CONTINUE
      RETURN
      END
      SUBROUTINE XXX(XX)
      DIMENSION XX(1)
      K=XX(1)+XX(2)+.1
      XX(3)=0.
      XX(4)=0.
      XX(5)=0.
      XX(K+3)=1.
      RETURN
      END

      SUBROUTINE EVENT(I)
      GOTO (1,2),I
1     CALL BEGREP
      GOTO 100
2     CALL ENDREP
100   CALL EXNEL(I)
      RETURN
      END

      SUBROUTINE BEGREP
      DIMENSION NSET(1000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
C
C EVENT 1: BEGIN-OF-REPAIR
C
C INCREASE NUMBER OF BROKEN MACHINES
C
      NR=NR+1
C
C SET SYSTEM VARIABLES
C
      XX(MACHNR)=1.
      CALL XXX(XX)
C
C SAVE EVENT TIME
C
      ATRIB(3)=TNOW
C
C COLLECT STATISTICS OF LENGTH OF WORKING PERIODE
C
      CALL COLCT(TNOW-ATRIB(2),MACHNR)
C
C SCHEDULE END-OF-REPAIR EVENT
C
      CALL ENDRT
      RETURN
      END

      SUBROUTINE ENDREP
      DIMENSION NSET(1000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENVE (NSET(1),QSET(1))
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
C
C EVENT 2: END-OF-REPAIR
C
C DECREASE NUMBER OF BROKEN MACHINES
C
      NR=NR-1
C
C SET SYSTEM VARIABLES
C
      XX(MACHNR)=0
      CALL XXX(XX)
C
C SAVE EVENT TIME
C
      ATRIB(2)=TNOW
C
C COLLECT STATISTICS OF LENGTH OF REPAIR PERIODE
C
      CALL COLCT(TNOW-ATRIB(3),MACHNR+2)
C
C SCHEDULE BEGIN-OF-REPAIR EVENT
C
      CALL BEGRT
      RETURN
      END

      SUBROUTINE TEVENT(I)
      GOTO (1,2),I
1     CALL BEGRT
      GOTO 100
2     CALL ENDRET
      GOTO 100
100   RETURN
      END

      SUBROUTINE BEGRT
      DIMENSION NSET(1000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EUQIVALENCE (NSET(1),QSET(1)
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
      TIM=EXPON(XMY(NR+1),1)
      CALL TSCHDL(1,TIM,ATRIB)
      RETURN
      END

      SUBROUTINE ENDRT
      DIMENSION NSET(1000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
      TIM=EXPON(REPTIM,2)
      CALL NSCHDL(2,TIM,ATRIB)
      RETURN
      END
```

5.3 Results for example A

The limiting distribution for the parameters

$(a,b,c) = (1./100., 1./50., 1./100.)$

can be computed (see section 2.1) as

(L,L) 0.2   (H,F) 0.4   (F,F) 0.4

The results of a simulation run over a periode of 100000 time units are shown in Fig.4.

The average values of variables M00, M01 and M11

0.219 / 0.403 / 0.378

correspond to the above given figures.

```
                                        S L A M   S U M M A R Y   R E P O R T

                                  SIMULATION PROJECT 2 MACHINES A            BY ADELSBERGER

                                  DATE  4/15/1981                            RUN NUMBER    1 OF     1

                                  CURRENT TIME   0.10000E+06
                                  STATISTICAL ARRAYS CLEARED AT TIME  0.0

                                  **STATISTICS FOR VARIABLES BASED ON OBSERVATION**

                MEAN          STANDARD      COEFF. OF     MINIMUM       MAXIMUM       NUMBER OF
                VALUE         DEVIATION     VARIATION     VALUE         VALUE         OBSERVATIONS

M1 WORKING     0.6973E+02    0.7652E+02    0.1097E+01    0.3750E+00    0.4999E+03        580
M2 WORKING     0.7201E+02    0.7461E+02    0.1036E+01    0.0           0.7063E+03        604
M3 BROKEN      0.1027E+03    0.1038E+03    0.1010E+01    0.1406E+00    0.8212E+03        579
M2 BROKEN      0.9343E+02    0.9779E+02    0.1047E+01    0.6250E-01    0.6520E+03        604

                                  **STATISTICS FOR TIME-PERSISTENT VARIABLES**

                MEAN          STANDARD      MINIMUM       MAXIMUM       TIME          CURRENT
                VALUE         DEVIATION     VALUE         VALUE         INTERVAL      VALUE

STATE M1       0.5956E+00    0.4908E+00    0.0           0.1000E+01    0.1000E+06    0.1000E+01
STATE M2       0.5643E+00    0.4958E+00    0.0           0.1000E+01    0.1000E+06    0.0
STATE M00      0.2187E+00    0.4134E+00    0.0           0.1000E+01    0.1000E+06    0.0
STATE M01      0.4027E+00    0.4904E+00    0.0           0.1000E+01    0.1000E+06    0.1000E+01
STATE M11      0.3786E+00    0.4850E+00    0.0           0.1000E+01    0.1000E+06    0.0

                                  **FILE STATISTICS**

FILE    ASSOCIATED   AVERAGE       STANDARD      MAXIMUM     CURRENT     AVERAGE
NUMBER  NODE TYPE    LENGTH        DEVIATION     LENGTH      LENGTH      WAITING TIME

   1                   0.0           0.0           0           0           0.0
   2                   1.9998        0.0242        2           2          62.1244
```

Fig. 4

6. Summery

Example A can not be programmed straightforward in the currently most used languages for discrete simulation. This shows, that these languages lack a conceptual framework for future dependent events.

By subroutine EXNEL we provide a suitable concept. To increase the performance of this subroutine, instead of the SLAM subroutines COPY, REMOVE and SCHDL, the SLAM array QSET and the SLAM subroutines ULINK and LINK can be used directly. We have delayed this adaption until the SLAM II version is available for us.

It can be thought to develop a concept which is based on hazard rates. This leads to combined discrete-continous models on which we will focus in a seperate paper.

Appendix: The subroutines NSCHDL, TSCHDL and EXNEL

```
      SUBROUTINE NSCHDL(N,T,AF)
      DIMENSION NSET(1000)
      COMMON/SCOM1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON/GCOM1/ JJCDR,KKNN,LLFIL,LLRNK,LLTRY,
     1MFEX,NNAM1,NNAM2,NNAM3,NNAPO,NNAPT,NNATR,
     2NNFIL,NNTRY,TTBEG,TTCLR,TTFIN,TTSET,XXI(100)
     3,TTTS,TTTF
      DIMENSION AF(1)
      AF(NNAM2)=-3.
      CALL SCHDL(N,T,AF)
      RETURN
      END
      SUBROUTINE EXNEL(IEVENT)
      DIMENSION NSET(1000)
      COMMON/SCIM1/ ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON /MEIN/ XMY(2),REPTIM,NR
      EQUIVALENCE (ATRIB(1),MACHNR)
      COMMON /ZCOM1/ LRISK
      LOGICAL LRISK(100)
      COMMON/GCOM1/ JJCDR,KKNN,LLFIL,LLRNK,LLTRY,
     1MFEX,NNAM1,NNAM2,NNAM3,NNAPO,NNAPT,NNATR,
     2NNFIL,NNTRY,TTBEG,TTCLR,TTFIN,TTSET,XXI(100)
     3,TTTS,TTTF
      DIMENSION AHELP(100)
      IF(.NOT.LRISK(IEVENT)) RETURN
C
C SAVE ARRAY ATRIB
C
      DO 10 O=1,NNATR
10    AHELP(I)=ATRIB(I)
C
C LENGTH OF THE EVENT CALENDAR
C
      NMAX=NNQ(NCLNR)
C
C LOOP
C     I=0
20    I=I+1
21    IF(I.GT.NMAX) GOTO 30
C
C GET NEXT EVENT
C
      CALL COPY(I,NCLNR,ATRIB)
C
C CHECK FOR FUTURE DEPENDEND EVENT
C
      IF(ATRIB(NNAM2+1).LT.1.) GOTO 20
      IF(ATRIB(NNAM2).LT.O. .OR.
     1ATRIB(NNAM2).EQ.TNOW) GOTO 20
C
C RESCHEDULE THE FUTURE DEPENDENT EVENT
C
      CALL RMOVE(I,NCLNR,ATRIB)
      IR=ATRIB(NNAM2+1)
      ATRIB(NNAM2)=TNOW
      CALL TEVENT(IR)
      GOTO 21
C
30    CONTINUE
C
C END OF LOOP
C
C RESTORE ARRAY ATRIB
C
      DO 40 I=1,NNATR
40    ATRIB(I)=AHELP(I)
      RETURN
      END
```

```
      SUBROUTINE TSCHDL(N,T,AF)
      DIMENSION NSET(1000)
      COMMON/SCOM1 ATRIB(100),DD(100),DDL(100),
     1DTNOW,II,MFA,MSTOP,NCLNR,NCRDR,NPRNT,NNRUN,
     2NNSET,NTAPE,SS(100),SSL(100),TNEXT,TNOW,
     3XX(100)
      COMMON QSET(1000)
      EQUIVALENCE (NSET(1),QSET(1))
      COMMON/GCOM1/ JJCDR,KKNN,LLFIL,LLRNK,LLTRY,
     1MFEX,NNAM1,NNAM2,NNAM3,NNAPO,NNAPT,NNATR,
     2NNFIL,NNTRY,TTBEG,TTCLR,TTFIN,TTSET,XXI(100)
     3,TTTS,TTTF
      DIMENSION AF(1)
      AF(NNAM2)=TNOW
      CALL SCHDL(N,T,AF)
      RETURN
      END
```

References:

Chiang, L.C., An Introduction to Stochastic Processes and Their Applications, Robert E. Krieger 1980

Pritsker, A.A.B. and Pedgen, C.D., Introduction to Simulation and SLAM, John Wiley, 1979

Schriber, T., Simulation Using GPSS, John Wiley, 1974

SIMULA VERSION 1 Reference Manual, Publication No. 30234800, Control Data Corporation, Sunnyvale CA

SIMSCRIPT Reference Manual, Version 3, Publication No. 60358500, Control Data Corporation. Sunnyvale CA