A DISCRETE BIO SIMULATION - THE POPULATION REGULATION OF TURTLES

Keith P. Tognetti
Australian National University
Canberra, Australia

Summary

The nesting behaviour of sea turtles has been simulated using SIMSCRIPT 1.5. It is known that these nesting turtles may destroy the eggs in other nests in attempting to dig their nests. This simulation has been used to determine a quantitative relationship between the fraction of nests destroyed by the self-induced mortality due to digging and the level of the turtle population.

This paper emphasises the programming and simulation aspects of the study.

Introduction

This simulation is based on the biological work carried out by H.R. Bustard of the Australian National University on a population of green sea turtles (Chelonia mydas).

The model is briefly as follows: A turtle attempts to dig a pit in its nesting beach. If the pit does not collapse it lays its eggs and covers the nest with sand. It may make up to four abortive attempts during the one night before it is successful. It then returns to the nesting beach for up to four more lays at intervals of about a fortnight.

While digging its pit the turtle may destroy other nests. The purpose of this simulation is to provide a discrete model of the digging and laying behaviour of a realistic population of turtles and obtain a quantitative relationship between the fraction of nests destroyed and the population level. This relationship is to be used as a basis for determining whether the self-destruction of nests by the turtles is a limiting factor in the regulation of their numbers.

The paper describes the simulation and programming techniques in an attempt to demonstrate that the modeling of ecological systems is a task that could be readily undertaken by a biologist with the help of a language such as SIMSCRIPT.

Simplified Model and its Extension to the Simulation Model

An oversimplified form for the model would be as follows: A wave of turtles arrives at a beach and lays at random on the beach. A second wave of turtles of the same size arrives and also lays at random. In the process some of the previous nests are destroyed. Thus formulated the problem is equivalent to a sampling process without replacement, i.e. the distribution of nests overlaid for each wave of turtles is according to a hypergeometric distribution. Even this simplification makes the computation of the distribution function for the number of nests destroyed after a given number of nights rather difficult. However a certain structure can be obtained by dealing with expectations (refer Appendix 2).

This structure is lost as soon as the model is extended. Firstly the number of turtles arriving per night for their first nesting is highly variable. Secondly, the number of aborted digs (each of which can result in nest destruction) is a random variable. Thirdly, each turtle has to be scheduled to return to the beach a random number of times at intervals such that the turtles arriving on any given night are a mixture of first, second, third and fourth layers. Additionally there are minor extensions to account for the arrival distribution around high tide and the difference in nest occupancy times between layers and non-layers.

General Organisation of the Simulation

The program is best (and fully explained) by referring to the flow chart (fig 1-8), decision table (fig 9) and the SIMSCRIPT listing (fig 10). Each vertical column in the decision table represents a logical path through the procedure scheduled by the event AR. The entries in the table above the double horizontal line simply represent the decisions to be tested, whereas the entries below indicate the action to be taken if these decisions are satisfied. The numbers on the flow chart are related to the statement numbers on the SIMSCRIPT listing. To approximate the field data, digging sites are selected from a distribution which is uniform parallel to the beach and triangular inland.

The digging activity for a turtle for any one night is according to the branching process detailed in Appendix 1.

The simulation was programmed in SIMSCRIPT 1.5 for the Control Data 3600 at the Commonwealth Scientific and Industrial Research Organisation installation in Canberra, Australia.
An event notice (4 words of storage) is allocated for each active turtle in the system. A temporary event (4 words) is allocated for each nesting site on the beach in which at some stage eggs have been laid.

Turtle Storage. Apart from FR which determines the run characteristics and is entered once only for each run, there is only one other endogenous event AR. AR decides whether the turtle is starting or finishing digging (NTYP). It schedules the number of times it will return to the beach (LAYT) and calls a subroutine (SCRT) to decide how many aborted attempts (ABORT) will be made before laying. AR could have been broken up into many events, one for each begin dig and one for each finish dig. This would have simplified the decision logic somewhat, however the associated increase in the work required to dynamically transfer attributes from one notice to the next far outweighs this. Consequently the simulation is organised with essentially one event notice for each turtle and this remains in the same location in memory for its digging life. This notice is of course rescheduled each time a new start or finish dig is required.

As a result, the organisation of the simulation is similar to G.P.S.S. which has only the one transaction for each entity. In fact, if one wished to go to the trouble AR could have been set out in a block layout equivalent to G.P.S.S.

Laying Site Storage. Sites that have nests are stored in a list ranked on the coordinates. Each site has attributes for coordinates, list processing (successor predecessor), laying status (occupied, eggs, no eggs) and number of sets destroyed.

Whenever a turtle has to select a spot for laying (subroutine SPOT) firstly a site is selected from the bivariate distribution for the laying pattern and then the site list is scanned to determine whether a new addition to the list is required. If the turtle selects a new site but does not lay, this site is removed from the list when it has finished digging.

Burst of Turtles. The program has been written so as to allow bursts of turtles (either normal first arrivals or diggers only) to be scheduled on a particular night. Such bursts may be superimposed on a normal run to assess the destructive worth of a turtle with time. Some care has been taken to ensure that the original sequence of pseudorandom numbers in the run is not affected by the burst.

Discrete Simulation Languages

There are three major reasons for using a special simulation language for discrete modelling based on the following requirements.

The first is the requirement for a list processor so as to store information dynamically and with more flexibility than provided by that of subscripted arrays. The second is to ease the housekeeping associated with scheduling and keeping track of the various events. The third is the requirement to be able to easily extend or modify the program.

The need for list processing arises from the uncertainty in the length of queues, sets and other lists that are part of the model. In this simulation, for example, neither the number of turtles active in the system at any time nor the number of laying sites is known. Consequently an overstatement in the storage allocated to either of these reduces the storage available to the other. This difficulty is compounded when the turtles are broken up into lists of begin laying and finish laying types for the first, second, third, etc. time.

An efficient timing routine based on list processing operation on a calendar set with most of the housekeeping kept out of sight of the programmer is a necessity for the efficient modelling of a large stochastic system to ensure that the programmer will not be distracted from his proper business of structuring the system.

Apart from these major reasons for using a discrete simulation language, there are many minor advantages available, such as: automatic provision of routines for calculating probability distribution functions; routines for evaluating mean, variance, etc. of lists; accumulation of time weighted quantities and so on. All of these minor advantages could be provided in the form of subroutine extensions to a FORTRAN program, however none of the major characteristics could be easily provided in this manner.

An attempt was made to translate this SIMSCRIPT program into the exclusively IBM language G.P.S.S./360. It was difficult to make a comparison between these languages in terms of compilation times and memory because apart from the different machines the IBM system does not appear to readily provide these details, but it is known that in such a comparison G.P.S.S. II compared very badly with SIMSCRIPT I requiring twice the storage and being seven times slower. However on the basis
of structure the G.P.S.S. language appears to be deficient for this simulation. Firstly, it is not a general purpose language but rather a special purpose language. It is believed that its special purpose nature makes it reasonable for modelling traffic systems and job shop scheduling. However it does expect lists to be associated with calendar and because of this it sets up the structure of members of this list (transactions) so that they may be used in the timing sets (both future and current). The result is that there is no provision in G.P.S.S. for the list processing of the temporary entities of SIMSCRIPT without paying a severe penalty (9 full words per entity, compared with less than one word in SIMSCRIPT). To represent the site storage in the list processing available in G.P.S.S. required the splitting of transactions and the placing of these transactions on a user chain, rather a messy business. Additionally G.P.S.S. suffers because its block structure is supposed to be adequate for its specific purpose, and it has only a vestigial language which must return everything in fixed point form and requires tedious scaling. The fact that FORTRAN subroutines are not available to G.P.S.S. gives it very little appeal to a programmer who wishes to generate his random samples using analytic inversion. Additionally because of its interpretive nature it would appear that G.P.S.S. is quite unsuitable for interactive on line simulations as it has to be recompiled for each run.

Recently IBM have released their language SEAL which is simply an extension of SIMSCRIPT. Unbelievably this package is claimed to have a memory requirement of 215K whereas the memory available to a 256K 360/50 installation with monitor is 214K! Consequently we are not as yet in a position to use the compiler on the Australian National University installation. However it is certainly structurally very attractive. Apart from free form initialisation and definition the input output format has been extended to character manipulation and logical variables. An extremely attractive feature is the relationship of entities to events.

To overcome the difficulty described earlier of being required to supply a unique event notice entity for each event, SEAL allows the use of compound structure for the event name.

Thus we could have

CREATE TURT

CAUSE ARRIV OF TURT AFTER DEL

Then later after the event ARRIV OF TURT has occurred we could schedule

CAUSE DEP OF TURT AT TIME + DEL

Both events ARRIV and DEP OF TURT refer to the one entity TURT and it is not necessary to reset any activities into new event notices to distinguish between arrivals and departures.

The Conversion of Biological Data into Steps in the Simulation

As an example of the technique for processing biological data and representing the biological structure as procedures within the program consider the turtle's digging activity on any one night.

This information was originally collected and made into a histogram showing the frequency of the first, second and third, etc. attempts to lay. The first step was to represent the information in this histogram by a branching process (Appendix 1) with conditional probabilities allocated to the various outcomes. This branching representation can then be readily coded into SIMSCRIPT using the decision commands available (SUBROUTINE GOSUB(IAR)) which subroutine we will now consider.

The first statement puts a random decimal number between 0 and .999 into XYU. ABORT(IAR) contains the counter for the number of aborted attempts to lay and this gives the subroutine for PLA the probability of laying at the ABORT(IAR)th attempt.

XYU is simply the probability of leaving after this dig. The first decision statement determines whether the random number is greater than the probability of laying plus staying in which case the turtle will leave after digging (ABORT set to 1).

If not, it is tested to see whether it is greater than the probability of laying in which case it is a stayer and the ABORT is incremented by 1 so that the next time it comes into the routine the correct values of the probabilities will be read.

The data for the conditional probabilities are automatically initialised at the beginning of the simulation from the initialisation deck (cards 20 and 21 at end of listing, figure 10). The card following 20 contains the probabilities of laying and the card following 21 contains the probabilities of staying.
Results

A series of simulation runs was carried out over a range of turtle populations representative of and extending the field situation.

Figure 11 is a plot of the percentage of nests destroyed against total turtle population. A linear regression was shown to fit these results with a correlation coefficient of .86.

The regression line has the form fraction of nests destroyed = .005 turtle population + .024.

Figures 12 and 13 show typical output results from the simulation as plotted by the CALCONE plotter.

Let us now examine the biological implications of this result. Consider a situation in which the population has been operating in a steady state for some time. By steady state is meant that the number of female adults at the time of nesting is constant as is the number of hatchlings. The probability of one of these hatchlings surviving to adulthood and returning to the beach for nesting (about 8 years) would be constant as would be the survival probability of the original population of females. Also the population of the next cycle of females and hatchlings would be the same as in the first cycle. Now assume that this process was subjected to a sudden fluctuation which resulted in the hatchlings being increased by a factor of p. Then assume that the other factors regulating the population are constant and that the number of turtles returning to the beach as adults will be increased by a factor less than p (as the contribution from the surviving adults is unaffected). The number of hatchlings produced by this increased population of females would certainly be larger than before the fluctuation. However it would be subjected to the self induced mortality due to digging and this would apply to each cycle, being most pronounced on the first cycle and gradually diminishing, resulting in the peak due to the fluctuation being gradually flattened out. On this basis, digging mortality can be regarded as a self regulating mechanism in the control of the turtle population.

This simulation as it stands cannot say anything about the relative importance of the digging mortality compared with other limiting factors. However, if the nature of these other limiting factors were known this model could be readily extended to include these factors. As was mentioned earlier, a distinct advantage of a special simulation language such as SIMSCRIPT is that the program can be much more readily modified or extended in complexity compared with a program written in FORTRAN.

This model is now being extended to include the predation of the hatchlings as they swim across the reef to the open sea. Only a small amount of data and some speculation is available for this region. Across the reef and beyond is mostly speculation.

APPENDIX 1

BRANCHING PROCESS FOR DIGGING ACTIVITY
Appendix 2

Simple Formulation of Model

$s =$ no. of sites for nesting
$r =$ no. of nests containing eggs
$j =$ no. of arrivals per night

It is assumed that each turtle digs once and lays. The laying site is selected from a uniform distribution. The site selection procedure for each turtle is independent of other turtles except that it will not disturb other turtles laying for that night i.e. sampling without replacement.

Thus the probability of $k$ sites being destroyed is

$$P_k = \frac{\binom{r}{k} \binom{s-r}{j-k}}{\binom{s}{j}}$$

The distribution is hypergeometric. Consider the expectation for this distribution

$$E = \frac{rj}{s}$$

Let $E_n =$ Expected number of nests after nth night

$$ED_n =$ Expected number of nests destroyed on nth night

$$p = \frac{j}{s}, q = 1 - p$$

$$TD_n =$ Expected total nests destroyed after n nights

First Night

$$EN_1 = j$$
$$ED_1 = 0$$

Second Night

$$ED_2 = EM_1 j/s = pEM_1 = pj$$
$$EN_2 = j + EM_1 - pEM_1 = j(1 + q)$$

Thus

$$ED_n = pj (1 + q + \ldots q^{n-2})$$
$$ED_n = j (1 - q^n - 1)$$

$$EN_n = j (1 + q + \ldots q^{n-1})$$
$$EN_n = s(1 - q^n)$$

Also

$$TD_n = \sum_{k=1}^{n} ED_k = s(np + q^n - 1)$$
$$TD_n = nj - s(1 - q^n)$$

Expected Fraction of Nests Destroyed

$$= \frac{TD_n}{nj}$$
$$1 - \frac{(1 - q^n)/pn}{1 - \frac{1}{n} \frac{1 - (1 - np + n(n - 1)p/2 - \ldots) - (n-1)p}{np}}$$

As $n \rightarrow \infty$

$$EN_n \rightarrow s, ED_n \rightarrow j, TD_n \rightarrow nj,$$

fraction destroyed $\rightarrow 1$

If only a fraction $f$ of the nests contain eggs, the above formula (1) to (3) are multiplied by $f$ with $p = \frac{j}{s}$ unchanged.

Now for small $p$

$$\frac{1 - q^n}{pn} \sim 1 - \frac{1 - (1 - np + n(n - 1)p/2 - \ldots) - (n-1)p}{pn}$$

Thus the expected fraction of nests destroyed is directly proportional to the turtle population for small $p$ and large $n$. 

350
**Fig. 1**

General Causal Relationship of Events

**Fig. 2**

Endogenous Event Fir

**Fig. 3**

Schedule for Turtle Arrivals (In Fir)

**Fig. 4**

Endogenous Event Ar (1 of 3)
FIG. 5

ENDOGENOUS EVENT AR (2 of 3)

FIG. 6

ENDOGENOUS EVENT AR (3 of 3)

FIG. 7

SUBROUTINE SPOT
Positions turtle in laying zone, increment nests destroyed counter and laying status. Also tags turtle with location of the spot.

FIG. 8

SUBROUTINE SCRT
Determines laying sequence for one night. Returns detail of path according to the branching process detailed in Appendix 1.
<table>
<thead>
<tr>
<th>DECISION</th>
<th>START</th>
<th>FINISH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last dig this wave</td>
<td>Y N Y Y Y Y Y Y Y Y Y N N N N N N N</td>
<td></td>
</tr>
<tr>
<td>Layer?</td>
<td>Y Y N N N N N N N Y Y Y Y Y N N N N</td>
<td>N N N</td>
</tr>
<tr>
<td>New spot selected?</td>
<td>Y N Y Y N Y N Y N Y N Y N Y N Y N</td>
<td>Y N Y</td>
</tr>
<tr>
<td>Last dig tonight?</td>
<td>Y Y N N Y Y Y Y Y Y Y Y Y Y Y Y</td>
<td>Y Y N</td>
</tr>
<tr>
<td>Dig next wave?</td>
<td>Y Y Y Y N N N N N N N N N N N N</td>
<td>Y N N</td>
</tr>
<tr>
<td>Any turtles active?</td>
<td></td>
<td>Y Y N N Y Y N N</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ACTION</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Select spot</td>
<td></td>
<td>* *</td>
</tr>
<tr>
<td>Increment actual lays</td>
<td></td>
<td>*</td>
</tr>
<tr>
<td>counter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schedule finish</td>
<td></td>
<td>* *</td>
</tr>
<tr>
<td>&quot;laying&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Increment wave counter</td>
<td></td>
<td>* * * *</td>
</tr>
<tr>
<td>Status = &quot;Eggs Laid&quot;</td>
<td></td>
<td>* *</td>
</tr>
<tr>
<td>Reschedule next dig</td>
<td></td>
<td>* *</td>
</tr>
<tr>
<td>Remove spot from list</td>
<td></td>
<td>* * * *</td>
</tr>
<tr>
<td>Status = &quot;Eggs Destroyed&quot;</td>
<td></td>
<td>* * * *</td>
</tr>
<tr>
<td>Reschedule next dig</td>
<td></td>
<td>* * * *</td>
</tr>
<tr>
<td>tonight</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schedule for next wave</td>
<td></td>
<td>* * * *</td>
</tr>
<tr>
<td>Remove turtle from system</td>
<td></td>
<td></td>
</tr>
<tr>
<td>decrement active count</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Finish this run</td>
<td></td>
<td>* * * *</td>
</tr>
<tr>
<td>Return</td>
<td></td>
<td>* * * *</td>
</tr>
</tbody>
</table>

FIGURE 9 DEcision Table for AR
FIG. 10
SIMSCRIPT LISTING

DEFINITION CARDS,

* N AR 4 N NTYP 31/4 I 1RN1 I US GC XIRNK L
* N FIN 2 N LAYT 32/4 I 2RN2 F UL
* N LAYA 41/2 I 3RN3 I US
* N ID 42/2 I 4RN4 I US
* T COM 4 T FOC 11/2 I 51OX I
* T SOC 12/2 I 610Y I
* T I RNK 2 T FOC 7FC 1 FO
* T I STAT 3 T 6LOC I
* N DSD 4 I 9LNG I
* N ABR10 33/4 I 10KFL I
* N ISVR 34/4 I 11KF I
* N D2F 12FKF F
* N D1TSD I
* N D1TL F
* N D1KPA I
* N D1NIT I
* N D1XNIT F 1 F
* N D1IOG I
* N D9LPL I
* N D2PLA F 1 F
* N D2PST 1 F
* N D2TID F
* N D2KOUT I
* N D2KTB F
* N D2INORR I
* N D2IBU RR I

EVENTS
1. EXOGENOUS
   1. BEGIN
      2. ENOGENOUS
      AR
      FIN

END
EXOGENOUS EVENT BEGIN

TEMPORARY ATTRIBUTES,

NTYP = ENTITY TYPE ( 1 = ARRIVE ) ( 2 = FINISH LAYING)

(3 = HAS DUG ONLY ON NEW SPOT,...,DESTROY THAT SITE AT END)
LAYA = ACTUAL NO. OF LAYS PER TURTLE,SO FAR
LAYT = TOTAL NO. OF LAYS PER TURTLE
ID = IDEAT OF COR ( SITE OCCUPIED)
I RNK = X,Y COORDINATE FOR RANKING
I STAT = COORDINATE STATUS ( 1 = OCCUPIED ) ( 2 = EGGS )
   ( 3 = EGGS DESTROYED )
NSD = NO. OF SETS DESTROYED AT THIS POSITION
ABORT = STATUS FOR ABORTED LAYS
   1 = WILL LAY
   2 = WILL DIG AND LEAVE
   2 = WILL DIG AND ENTER SECOND TEST
   N = WILL DIG AND ENTER N-TH TEST
PERMANENT ATTRIBUTES:

1 - RN1 - TURTLES PER NIGHT (1)
2 - RN2 - TIME OF ARRIVAL ABOUT HIGH TIDE (1), HOURS
3 - RN3 - LAYS PER TURTLE (1)
4 - RN4 - DAYS BETWEEN LAYS (1)
5 - IOX - XCOORD ON BEACH (1, TRI)
6 - IOD - YCOORD ON BEACH (1, RECT)

(THE ABOVE IS NO LONGER READ IN AS A FUNCTION)

7 - FOC - FIRST POINTER (FOR QUEUE QC)
8 - LOC - LAST POINTER (FOR QUEUE QC)
9 - LNS - LENGTH OF KFL (10)

ABOVE IS THE LARGEST NUMBER OF WAVES OF TURTLES,

10 - KFL() - COUNTER FOR I-th WAVE
11 - KF - COUNTER FOR TOTAL NO. OF TURTLES, ACTIVE
12 - FK - TOTAL NO. OF NEW TURTLES
13 - ITSD - TOTAL SETS DESTROYED
14 - TTL - TIME BETWEEN LAYINGS (IN DAYS)
15 - KPA - ACTIVE NO./TIME PLOT (0 OR BLANK = PLOT)
16 - NIT - LENGTH OF XNIT

ABOVE IS THE NUMBER OF NIGHTS OF NEW ARRIVALS

17 - XNIT() - MEAN FACTOR FOR THE NO. OF TURTLES ON I-th NIGHT
18 - LNC - COUNT OF THE NO. OF OCCUPIED SITES
19 - LNL - LENGTH OF PLA AND PST

20 - PLQ() - PROB OF LAYING AFTER 1-1 TRIES
21 - PST() - PROB OF STAYING AFTER 1-1 TRIES

HENCE [1 - (PLQ() * PST())] IS THE PROBABILITY OF

LEAVING AFTER 1-1 ATTEMPTS AT LAYING;

22 - TTD - TIME TO DIG (IN HOURS)
23 - PLW - LAYING PLOT (0 OR BLANK = DO PLOT)
24 - TAB - TIME AT BEGINNING OF SIMULATION
25 - INORH - INDICATES NORMAL RANDOM ROOT SEQUENCE
26 - IBURH - INDICATES BURST TURTLE RAND SEQUENCE
27 - BURSTS REFERS TO PROVISION FOR SCHEDULING A BURST OF

TURTLES AT ANY TIME TO DETERMINE THEIR DESTRUCTIVE WORTH.

LOCAL VARIABLES...(IN EVENT FIX)

KBUR - TYPE OF TURTLE BURST
0 = DIGS ONCE AND GOES
1 = NORMAL TURTLE

NBUR - NUMBER OF TURTLES IN BURST

TUR - TIME OF BURST

MULTPLICATION FACTOR FOR RANDOM NO. OF TURTLES

IF VALUE IS LESS THAN ZERO, TERMINATE THE PROGRAM

CALLS TO SUBROUTINE PLOT REFER TO A LIBRARY PLOT ROUTINE FOR

GRAPHICAL OUTPUT TO A CALCOPM PLOTTER

WRITE ON 61, NIT
FORMAT(*, NIT = *4, 16)
DO, FOR I = 1, NIT
WRITE ON 61, 1, XNIT(I)
FORMAT(* NIT, N, *12, * = *, D, 6)
END
WRITE ON 61, TTD
WRITE ON 61, 1, TTD
FORMAT(* TIME TO DIG = *4, M, 4, 3, 3)
WRITE ON 61
FORMAT(S14, SHAPE(I) + STA(I)$ = TOTAL$
D0, FROM I = (1)(LPL)
LET XYZ = SHAPE(I) + PST(I)
WRITE ON 61, SHAPE(I), PST(I), XYZ
FORMAT(* NO,*, SHAPE + SUM $= D6,4,*, D6,4,*, D6,4,*)
LOOP
C SET ORIGINS AND SCALES FOR ANY REQUIRED PLOTS
IF KOUT = 0, CALL PLOT(0,0,0,1)
IF KPA EQ 0, CALL PLOT(0,0,0,1,4)
IF KOUT EQ 0, CALL PLOT(0,0,0,2)
IF KPA EQ 0, CALL PLOT(0,0,0,2,4)
CREATE PLOT
CAUSE PLOT AT TIME
RETURN
END

ENDogenous EVENT FIR

C THIS FOLLOWING EVENT DEPENDS ON THE SIMULATION
C
WRITE ON 61, TIME
FORMAT(* IN FIR AT TIME $= M4,3,3$

C NB. DATA CARDS MUST CONTAIN DUMMY EXOGENOUS EVENT CARD (1,1,15)

C THIS ALLOWS DATA TO BE READ IN FROM EVENT FIR
C
LET TAB = TIME
READ FROM 60, RN, KOUT, KPA, NBUR, TBUR, KBU
FORMAT(04,5,16,18,12,03,1,12)
WRITE ON 61, NBUR, TBUR, KBU, RN, KOUT, KPA, TAB
FORMAT(* NO, IN BURST $= 18, TIME OF BURST $= D6,1$
C, KBU $= 18$, NEW RN $= D6,4$, KOUT $= 14$, KPA $= 14$,
C, START AT TIME $= M4,3,3$
C SET PEN TO ORIGIN (IF REQUIRED)
IF KOUT EQ 0, CALL PLOT(0,0,0,3)
IF KPA EQ 0, CALL PLOT(0,0,0,4)
C IF VALUE OF RN NEGATIVE, INDICATES TERMINATION OF PROGRAM
IF RN LS 0, STOP
IF KOUT NE 0, GO TO 61
C PLOT AXES
C PLOT (0,0,0,4)
CALL PLOT(0,0,0,1)
CALL PLOT(0,0,0,1,4)
CALL PLTO(30,0,0,3)
CALL PLOT(0,0,0,4)
61 IF KPA NE 0, GO TO 60
CALL PLOT(0,1000,0,3,4)

CALL PLOT(0,0,0,1,4)
60 IF QC IS EMPTY, GO TO 50
REMOVE FIRST IV FROM QC
DESTROY COR CALLED IV
GO TO 60
C SCHEDULE BURSTS
C LET IRS = RANDR
C RESTART ALL INDEX STORAGES
DO, FOR I = (1)(NB)
LET KFL(I) = 0
LOOP
LET I = 0
LET IRS = 0
LET KF = 0
C ADJUST ATTRIBUTES ETC., AND SCHEDULE BURST OF TURTLES
DO, FOR I = (1)(NB)
CREATE AR
STORE 1 IN TYP (AR)

356
STORE 9 IN ISVR(AR)
STORE 1 IN ABORT(AR)
STORE 0 IN LAYA(AR)
      IF KF M 10, GO TO 10
STORE 1 IN LAY(AR)
GO TO 32

10 LET LAY(AR)= RN3
    CALL SORT(AR)
12 CAUSE AR AT TIME + TBUR + RN2/24, 0
    LET KF = KF + 4
    LOOP
LET INHR = RANDR
LET HDR = IRRSAV
C LOOP FOR NO OF NIGHTS
DO TO 4, FOR II=1(INI)
   LET MOLD = RN1
C RN1 IS NO OF TURTS PER NIGHT
C WRITE ON 61, III, MOLD, XNIT(II)
   FORMAT* NO, II, XNIT(II)
   LET IR=RN* MOLD, XNIT(II)
   WRITE ON 61, IR
   FORMAT* (TURTS PER NIGHT) *
   LET TIM=II
GO TO 5, FOR IJ=1(INI)
C RN2 IS TIME OF ARRIV OF TURTS
LET XY=RN2
LET CH = XY/24,
   LET TY = TIM + CH
CREATE AR
STORE 1 IN ABORT(AR)
    CALL SORT(AR)
STORE 1 IN NTYP(AR)
C NTYP IS ENTITY TYPE 1 ARRIV TYPE 2 FIN LAYING
C RN3 IS NO OF LAYS PER TURT
C LET KF = KF + 1
STORE RN3 IN LAY(AR)
STORE 0 IN LAYA(AR)
C WRITE ON 61, AR, NTYP(AR), LAYT(AR), LAYA(AR), KF, CH, TY
C, ABORT(AR)
C FORMAT* AN NTYP LAYT LAYA KF CH TY*, 518, D6, 4
   C,M,3,3, * ABORT = *, I8)
    CAUSE AR AT TIME + TY
3 LOOP
4 LOOP

WRITE ON 61, KF
FORMAT* TOTAL NO, OF TURTLES,..., I8)
LET KF = KF
LET INDR = RANDR
RETURN
END

---
C THIS EVENT CAUSES ARRIVALS OF TURTLES FOR LAYING
C
C DIMENSION AN(10)
C WRITE ON 61, TIME, AR, NTYP(AR), LAYT(AR), ID(AR), LAYA(AR)

C, ABORT(AR)
C FORMAT* OIN AR,..., TIME, AR, NTYP, LAYT, ID, LAYA, ABORT,..., I8)
C, M, 3, I8, 618)
IF NTYP(AR) NE 1, GO TO 10
C TURTLE ARRIVES TO DIG OR LAY
IF LAYA(AR) NE 0, GO TO 90
IF ABORT(AR) GR 1, GO TO 90
LET TATA = TIME + TAB
IF KPA EQ 0, CALL PROF(1, TATA)

90. STONE 2 IN NTYP(AH)
   IF ABORT(AR) LE 1, LET LAYA(AR) = LAYA(AR) + 1
   IF LSRD(AR) EQ 9, GO TO 92
   LET RANDR = INORR
   CALL SPOT(AR)
   LET INORR = RANDR
   GO TO 93
92. LET RANDR = IBURR
   CALL SPOT(AR)
   LET IBURR = RANDR
93. IF ABORT(AR) GR O, GO TO 12

C RESCHEDULE AFTER LAYING
   LET XY = TIME *.TIL
   CAUSE AR AT XY
   GO TO 70

C RESCHEDULE AFTER DIGGING
   12. LET XY = TIME *.TID
   CAUSE AR AT XY
   LET L0 = L0
   70. WRITE ON 61, XY
   C FORMAT(* TIME TO FINISH DIG OR LAY *,M4,3,3)
   RETURN

C TURTLE FINISHED DIGGING OR LAYING
   10. IF ABORT(AR) LS 2, LET KFL(LAYA(AR)) = KFL(LAYA(AR)) + 1
   IF ABORT(AR) GT 2, GO TO 13
   C WRITE ON 61
   C FORMAT(* LAY AND FINISH *)
   STORE 2 IN ISTAT(I0D(AR))
   STORE 1 IN ABORT(AR)

C SCHEDULES NEXT WAVE
   15. LET L0 = L0
   15. WRITE ON 61, KFL(LAYA(AR))
   C FORMAT(* NO. OF TURTS IN LAYA WAVE KFL *, I8)
   IF LAYT(AR) GE LAYA(AR), GO TO 20
   CALL SCRT(AR)
   LET XY=M4
   LET RND = TIME *.TIL + .5
   LET INT = ROUND
   LET WRT = INT
   LET TOA = RN2/24.
   LET XZ = XY + WAIT + TOA
   C WRITE ON 61, XY, XZ, TOA

C FORMAT(* TIME TO AND AT NEXT LAY AND RANDOM TOA *,2M4,3,3)
C C,D6,4)
   STORE 1 IN NTYP(AR)
   CAUSE AR AT XZ
   RETURN

C

C IF NEW SPOT REMOVE FROM G0

13. IF NTYP(AR) EQ 3, GO TO 35
   STONE 3 IN ISTAT(I0D(AR))
36. IF ABORT(AR) EQ 1, GO TO 14
   WRITE ON 61, ABORT(AR)
   C FORMAT(* DIG AND TEST..,.ABORT = *,I8)
   C
C TURTLE IS TESTED FOR (ABORT)*TH ATTEMPT
   CALL SCRT(AR)
   IF ABORT(AR) EQ 5., LET ABORT(AR) = 0
   STORE 1 IN NTYP(AR)
   CAUSE AR AT TIME

358
C NOTE... TIME DELAY IS NOT CONSIDERED.
RETURN

35 REMOVE ID(AH) FROM QC
DESTROY CDR CALLED ID(AH)
GO TO 36

14 LET LO = LO
C 14 WRITE ON 61
C.. FORMAT(* DIG AND FINISH,*
GO TO 15
C THIS TURTLE FINISHED........
20 LET KF = KF
C WRITE ON 61,KF
C FORMAT(* THIS TURTLE FINISHED, NO,ACTIVE=*,18)
IF KPA NE 0, GO TO 21
LET TATA = TIME = TAB
CALL PROF(-1,TATA)
21 DESTROY AR
IF KF EQ 0, GO TO 30
RETURN
C
C FINISH OF SIMULATION,...STATISTICS ETC.,
C
30 LET XZ = TIME = TAB
WRITE ON 61,TIME,XZ
FORMAT(* END OF SIMULATION,...,2M4,6,3)
LET KK = 0
LET AN(I) = 0,FOR 1 =17(10)
WRITE ON 61
FORMAT(* XY COORDINATE DESTROYED,,NO. OF SETS DESTROYED*)
LET KNL = 0
IF KOUT EQ 0, THEN PLOT
DO, FOR EACH I IN QC
LET IYX = NSD(I)
C IF KOUT EQ 0, THEN CALL OUT(IANK(I),IYX)
IF IYX EQ 0, GO TO 80
C ARRAY AN GIVES AGGREGATES OF THE NUMBER OF SETS DESTROYED ON
C INDIVIDUAL SITES,.....
LET AN(IYX) = AN(IYX) + 10
WRITE ON 61,1ANK(I),1STAT(I),IYX
FORMAT(* =19,17,54,18)
80 LET KNL = KNL + IYX + 1
IF IYX(EQ 3, LET KNL = KNL + 1
LET KK = KK + 1

LOOP
LET XY = 0,0
WRITE ON 61,A,AN(1),AN(2),AN(3),AN(4)
C,AN(5),AN(6),AN(7),AN(8),AN(9),AN(10)
FORMAT(* VALUES OF AN(I)=10D4,0)
LET XY = XY + AN(I),FOR I = 1(10)
LET KY4 = KNL - ITSU
LET XY4 = KY4
LET X4 = X4/FKY
WRITE ON 61, KY4,XY4
FORMAT(*0 NO. OF SETS SURVIVING =*,18,*, NESTING EFFICIENCY =*
C,5D4,4)
LET XIOC = IGC
LET P = ITSU
LET FKNL = KNL
LET COUNT = KK
LET PUS = P/COUNT
LET POC= XIOC/COUNT
LET POCLU = P/FKNL
LET XY2 = P/XY
WRITE ON 61,ITSU,POUS,POCC
FORMAT(* TOTAL SETS DESTROYED =*,18,* PROXY OF SITE DESTRUCTION
NN = \*U2,7,\* RATIO OF OCCUPIED SITES =*,D2,6
WRITE ON 61,X2,KNL,POCLU
FORMAT(* COND, MEAN *,
\*U2,6,\* TOTAL LAYS \*I5,\* PROB OF CLUTCH DESTRUCTION \*D2,6)
C WAVE DATA FOLLOWS
DO , FGM 1=(1)(LNG)
WRITE ON 61,11,KFL(II)
FORMAT(* WAVE NO., \*13,\* CARRIED \*14,\* TURTLES,\*) LOOP
C PRINT THE TOTAL NUMBER OF TURTLES IN EACH WAVE
WRITE ON 61,11K
FORMAT(* TOTAL NO. OF NESTS \*I8 )
LET NEXTDAY = (TIME + 10.07/10.0
LET HOUR = NEXTDAY = 10
CREATE FIR
CAUSE FIR AT HOUR
RETURN END
SUBROUTINE SPOT (IFL)
POSITIONS TURTLE IN LAYING ZONE
C C SETS ISTAT(COR) TO 1 FOR OCCUPANCY
C SETS SITE ADDRESS TO ID(AR)
C UPDATES OCCUPANCY COUNTER IOC
C UPDATES NO OF SITES DESTROYED NSD
C UPDATES TOTAL SETS DESTROYED ITSD
C C WRITE ON 61
C FORMAT(* IN _SPUT _*)
25 LET IX=30, SRTF(RANDM)
LET IY=100,RANDM
LET IR=10000,IX + IY
C WRITE ON 61,IX,IY,IR
C FORMAT(* IX ,IY ,IR*, 319)
C FIND FIRST , FOR EACH I IN GC, WITH IR EQ IRNK(I),
WHERE IJ SATISFIES THIS, IF NONE GO TO 30
IF ISTAT(IJ) EQ 1, GO TO 20
IF ISTAT(IJ) EQ 2, GO TO 30
C OCCUPYING OLD EMPTY NEST
40 STORE IJ IN ID(IFL)
C WRITE ON 61,ISTAT(IJ), IJ,ABORT(IFL),IFL
C FORMAT(* ISTAT,IJ,ABORT,IFL *, 418)
STORE 1 IN ISTAT(IJ)
RETURN C IF OCCUPIED SELECT ANOTHER RANDOM SITE
20 LET IJ = LC
C WRITE ON 61
C FORMAT(* _OCCUPIED _*)
C LET IOC = IOC + 1
GO TO 25
30 LET ITSD = ITSD + 1
C WRITE ON 61
C FORMAT(* LAYING OVER *)
C LET NSD(IJ)=NSD(IJ)+1
GO TO 40
50 CREATE COR
C CREATE A NEW LAYING SITE
STORE IR IN IRNK(COR)
STORE I IN ISTAT(COR)
STORE ID IN NSD(COR)
FILL COR IN GC
STORE COR IN ID(IFL)
IF ABOERT(IFL) GR 0 , STORE 3 IN NTYP(IFL)
RETURN END
SUBROUTINE SORT(IAR)
C CHECKS FOR N*TH TEST AND...
C SETS ABO RT(IAR) INDEX AS FOLLOWS...
C 0 = IF TURTLE LAYS
C 1 = IF TURTLE DIGS AND LEAVES
C N+1 = IF TURTLE DIGS UNSUCCESSFULLY AND GOES ON TO N+1 -TH TEST
C
C RESULTING ACTION DEPENDS ON A RANDOM NC, FALLING IN THESE RANGES...
C RANDM < PLA ...LAY NOW
C PLA < RANDM < PLA+STA ...DIG AND TRY AGAIN
C RANDM > PLA+STA ... DIG AND LEAVE
C
C LET XYU = RANDM
C LET XYZ = PLA(ABORT(IAR))
C LET XYI = XYZ + 1ST(ABORT(IAR))
C IF XYU GT XYZ , GO TO 10
C IF XYU GT XYI , GO TO 12
C STORE 0 IN ABORT(IAR)
C GO TO 20
C 10 STORE 1 IN ABORT(IAR)
C GO TO 20
C 12 LET ABORT(IAR) = ABORT(IAR) + 1
C 20 LET LO = LO
C 20 WRITE ON 61, IAR, X,Y,XY,XY,ABORT(IAR)
C FORMAT* IN SCRT ..., IAR, RAN, SUM, ABORT *, IM, 206, 6, 18
C RETURN
C END
C
C FORTRAN SUBROUTINES
C
C SUBROUTINE OUT(I,J)
C PLOTS NESTING PATTERN *****
C THE NO. PLACED ON THE GRAPH INDICATES THE NO. OF SETS...
C DESTROYED AT THAT LOCATION,
C
C IN = 1/10000
C I = IN
C CALL PLOT(V, W, J)
C INK = SHIFT(J, 42)
C CALL TEXT(INK, 1, 1)
C RETURN
C END
C SUBROUTINE PROF(X,R)
C MAINTAINS A PLOT OF THE TOTAL NO. OF TURTLES ACTIVE
C IN THE MODEL AT A GIVEN TIME
C DATA(WHERE = 0, J)
C WHERE = WHERE + X
C CALL PLOT(R, WHERE, 4, 4)
C RETURN
C END
C
C INITIALIZATION CARDS,.....
C
C NUMBER IN COLUMNS 3-4 REFERS TO PERMANENT ATTRIBUTES
C AS SPECIFIED ON DEFINITION CARDS,
C
C | 123456789 | 123456789 | 123456789 | 123456789 | 123456789 | 123456789 | 123456789 | 123456789 | 123456789 |
C | 1 26 |
<table>
<thead>
<tr>
<th></th>
<th>R</th>
<th>S</th>
<th>I</th>
<th>US I</th>
<th>4(D1,6;12)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.025000</td>
<td>10.030000</td>
<td>20.032000</td>
<td>30.035000</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>0.040</td>
<td>0.45</td>
<td>0.60</td>
<td>0.92</td>
<td>70.063</td>
</tr>
<tr>
<td>3</td>
<td>0.08</td>
<td>0.6</td>
<td>1.0</td>
<td>100.0</td>
<td>110.08</td>
</tr>
<tr>
<td>4</td>
<td>0.063</td>
<td>1.35</td>
<td>0.05</td>
<td>1.40</td>
<td>0.45</td>
</tr>
<tr>
<td>5</td>
<td>0.035</td>
<td>1.70</td>
<td>0.032</td>
<td>1.80</td>
<td>0.03</td>
</tr>
<tr>
<td>6</td>
<td>0.0000</td>
<td>2.0000</td>
<td>0.057</td>
<td>1.750</td>
<td>0.0271</td>
</tr>
<tr>
<td>7</td>
<td>0.2045</td>
<td>0.5000</td>
<td>2.7840</td>
<td>0.250</td>
<td>0.363</td>
</tr>
<tr>
<td>8</td>
<td>0.6795</td>
<td>1.000</td>
<td>0.7370</td>
<td>1.250</td>
<td>0.7922</td>
</tr>
<tr>
<td>9</td>
<td>0.9481</td>
<td>2.5000</td>
<td>0.9708</td>
<td>2.750</td>
<td>0.9470</td>
</tr>
<tr>
<td>10</td>
<td>0.4044410</td>
<td>2.7111120</td>
<td>2.4592630</td>
<td>0.6370440</td>
<td>0.0148195</td>
</tr>
<tr>
<td>11</td>
<td>0.001454</td>
<td>0.80</td>
<td>0.042159</td>
<td>0.90</td>
<td>0.07376100</td>
</tr>
<tr>
<td>12</td>
<td>0.035927120</td>
<td>1.03664130</td>
<td>2.19178140</td>
<td>0.22706819</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>0.170</td>
<td>0.66160</td>
<td>0.097998170</td>
<td>0.064117180</td>
<td>0.01369912</td>
</tr>
<tr>
<td>14</td>
<td>0.019937200</td>
<td>0.6032221</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>0.9</td>
<td>10</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>12</td>
<td>Z</td>
<td>Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>14</td>
<td>H</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>15</td>
<td>R</td>
<td>35</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>17</td>
<td>1</td>
<td>35</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>0.2286</td>
<td>0.3571</td>
<td>0.4587</td>
<td>0.6143</td>
<td>0.7429</td>
</tr>
<tr>
<td>21</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
<td>0.98</td>
</tr>
<tr>
<td>22</td>
<td>0.98</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>24</td>
<td>Z</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>25</td>
<td>Z</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>26</td>
<td>0</td>
<td>Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>30</td>
<td>Z</td>
<td>Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>15</td>
<td>U</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>123456789</td>
<td>123456789</td>
<td>123456789</td>
<td>123456789</td>
<td>123456789</td>
</tr>
</tbody>
</table>

362
**FIG. 11**
Percentage of nests destroyed against total turtle population (linear regression).

**FIG. 12**
Nesting pattern on beach site. Numbers indicate nests destroyed.

**FIG. 13**
Active turtles versus time.