THE UNIVERSITY OF MICHIGAN
COLLEGE OF LITERATURE, SCIENCE, AND THE ARTS
Computer and Communication Sciences Department

MODELLING AND SIMULATION: AN INTRODUCTORY EXPOSITION OF CONCEPTS

by

Bernard P. Zeigler

MARCH 1978

Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot, Israel

and

Logic of Computers Group
Computer and Communication Sciences Department
Technical Report No. 211

assistance in the preparation of this report
was received through:

National Science Foundation Grant No.
MCS76-04297

MODELLING AND SIMULATION: AN INTRODUCTORY EXPOSITION OF CONCEPTS

by

Bernard P. Zeigler

Department of Applied Mathematics
The Weizmann Institute of Science
Rehovot, Israel

Foreword.

This introductory exposition of modelling and simulation concepts was prepared for first year open university students. The presentation does not require anything in the way of classical mathematics (arithmetic, algebra, calculus) although it makes use of elementary set-theoretic concepts (sets, functions) . These are introduced slowly as the development proceeds. Nevertheless, key concepts in modelling and simulation are introduced which are of practical concern to the working modeller. Thus the interested layman might also derive some understanding about the pervasive activity of modelling from this introduction.

The development begins with a commonly familiar example - the crying pattern of a six month old baby. A certain regularity is hypothesized and formalized as a finite state model. What the latter is, how to build one and what is to be gained are thoroughly discussed. From this characterization of the modelling process, we move on to consider the essentials of computer simulation and the process of model validation. The baby model is always returned to for concrete illustration. Problems are given to amplify the discussion in the text.

Table of Contents.

1. Introduction.

Modelling and Simulation comprises the activities involved in constructing a model of a real world system and simulating it on a computer. Scientists from almost all the disciplines are vitally engaged in making models and, in increasing numbers are employing the computer to simulate them. Certain models, of large scale systems such as the U.S. economy, the ecology of a forest, or of other complex, multifaceted systems, are made feasible only by the computing power of the modern digital computer. Despite the variety and complexity of models, there are certain basic concepts and problems which are common to all such activities. These ideas are surprisingly easy to express (they do not require sophisticated mathematics) and yet are surprisingly helpful in understanding, formulating, manipulating and communicating sophisticated models. Indeed, it is often by stripping away the hairy outer coating of a model that we perceive its essential structure. Put another way, modelling and simulation is not technically doable without mathematics and computers, but both practitioners and laymen can understand the concepts involved in a form requiring only a minimum of mathematical equipment.

2. An Example: What does the baby want?

Imagine the following situation: You have a little baby brother or sister about 6 months old. You notice that there are times when the baby cries and times when it is quiet. When it cries, it sometimes stops if you give it something to eat. But at other times this doesn't work and you have to give it something to play with to make it stop crying. Since giving it the wrong thing makes it more upset, you would like to be able always to know what is the right thing to give it beforehand.

As the days go by, you become more familiar with this aspect of the baby's behavior. You may notice, or think you notice, a certain pattern to it. This pattern seems to be the following:

After the baby finishes eating, it always wants to play.
After it finishes playing, it always wants to sleep.
After it finishes sleeping, it always wants to eat.

Thus the baby seems to want to go through a cycle: eating, playing, sleeping, eating, ..., ... If it is prevented from going through this cycle, it cries until you give it the right thing to let it continue i.e., food, when it finishes sleeping and toys, when it finishes eating.

We shall now try to represent this pattern in a way which allows us to express its features fully and unambiguously. The process of doing so is called modelling and the end result is called a model. There are a number of good reasons for attempting this task and we shall return to them later. For now, it suffices to say that in modelling we gain a better understanding of what it is that we believe might be true about

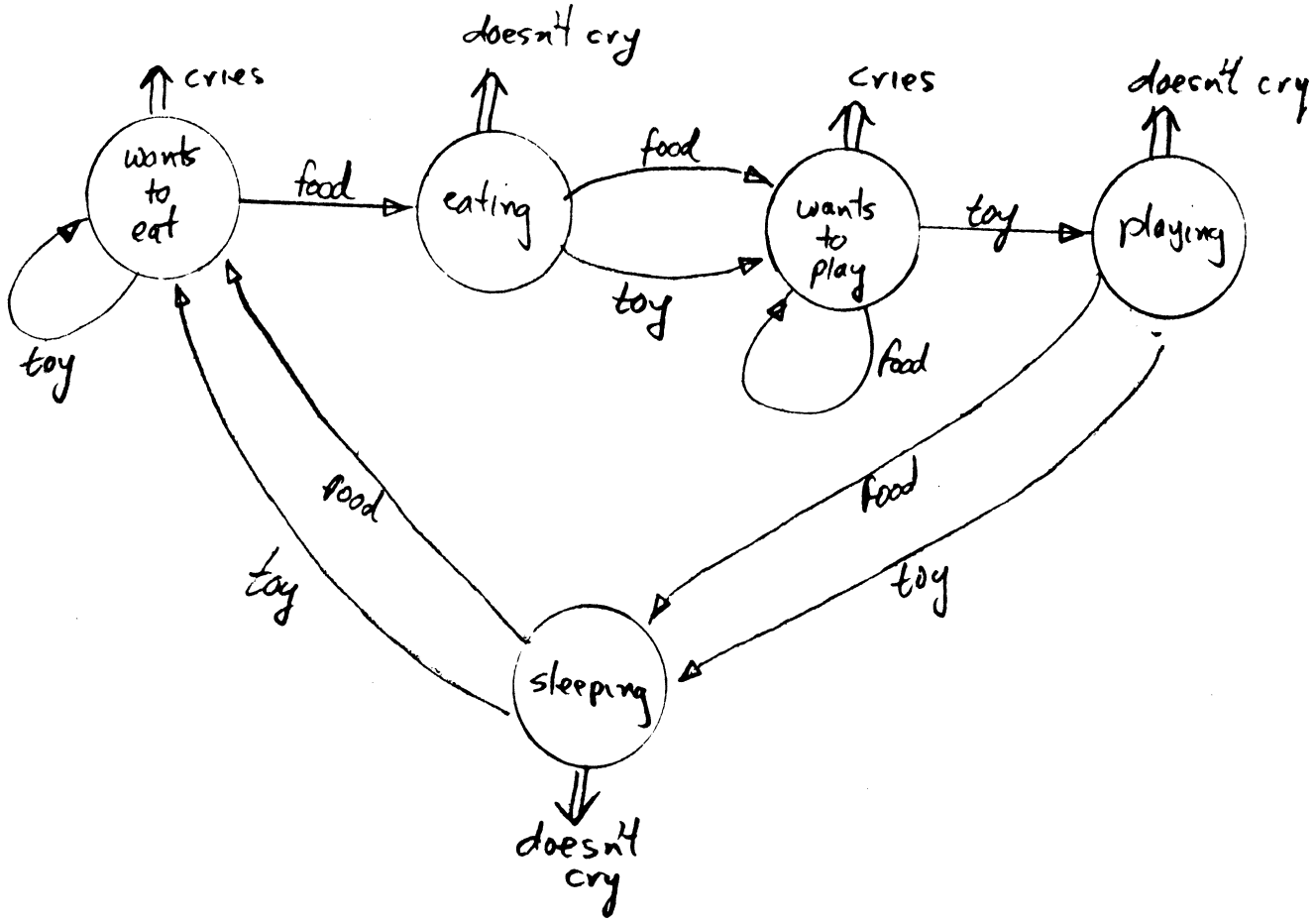


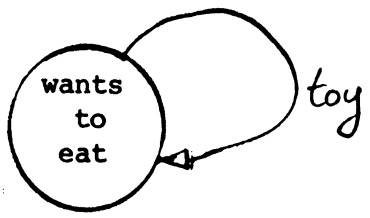
Figure 1. The Baby Model

the real system under study. In our present example, we want to understand more clearly what the baby's pattern seems to be.

3. Finite State Model of the Baby.

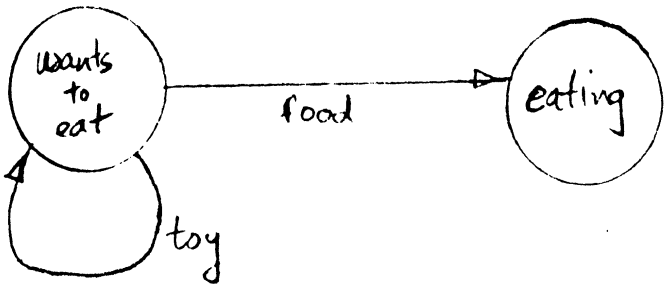
The model we shall discuss is called a finite state model. Basically we shall be saying that the baby can be in different states at different times. A state is not something you can measure, feel or see directly, so it may be difficult to imagine existing in reality. Indeed it is better not to try. This is because the concept of finite state model is an abstract concept . We can better understand this concept if we learn to visualize it by means of diagrams on paper. The diagram for the model we shall discuss is presented in Figure 1.

In the diagram, states appear as large circles. Let us start with the state labelled "wants to eat". The model says that if the baby is given a toy in this state then it stays in this state. This is shown by the arrow starting from the circle and returning to it thus:



We interpret this model rule as saying that if the baby wants to eat and if it is given a toy then it still wants to eat.

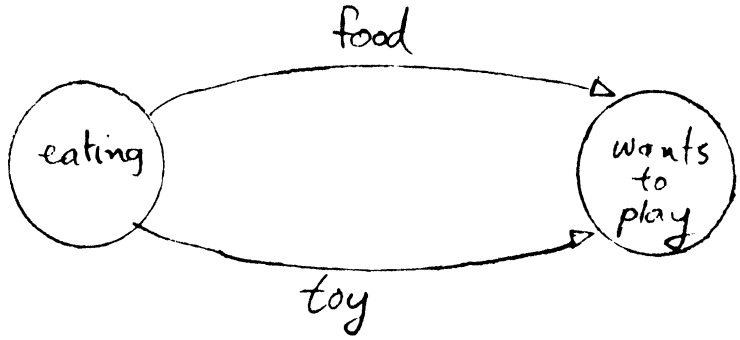
The model also says that in the state "wants to eat", if the baby is given some food it will go to another state labelled "eating". This is shown again by an arrow, this time going from "wants to eat" to "eating" thus:



Note that we have just introduced two new concepts. The first is that there are stimuli to which the baby reacts. In modelling terms, these stimuli are called inputs. In our model there are just two kinds of of inputs - "toy" and "food".

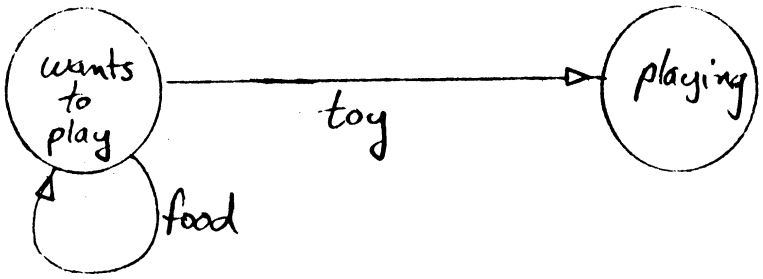
The second concept is that the model indicates how the baby reacts to receiving these inputs. Indeed it says that how the baby reacts to an input depends on what state the baby is in at the time. The way it reacts to is by changing or not changing its state; if a change in state is indicated, the model must say what the new state will be.

Let us see if these ideas are clear. What happens in the state "eating" is diagrammed as:

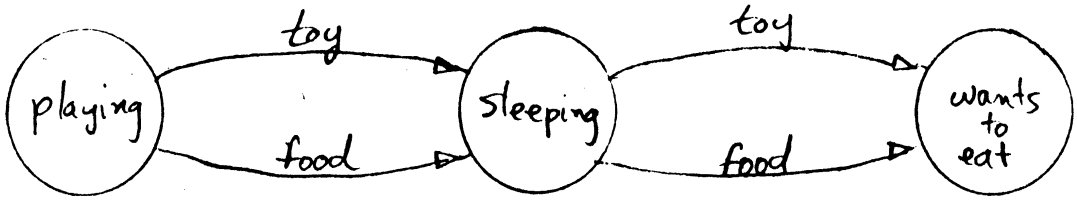


Since both arrows (labelled "food" and "toy") point to the same state "wants to play", our model says that while it is eating the baby does not pay any attention to whether it is given a toy or is given some food; when it finishes eating, it always goes into the state "wants to play". (If you think about it, you may not agree with what the model says at this point. Exercise 3.2 will help you consider some more realistic possibilities.)

Now you should be able to interpret what the model says for state "wants to play". Contrast it with state "wants to eat".



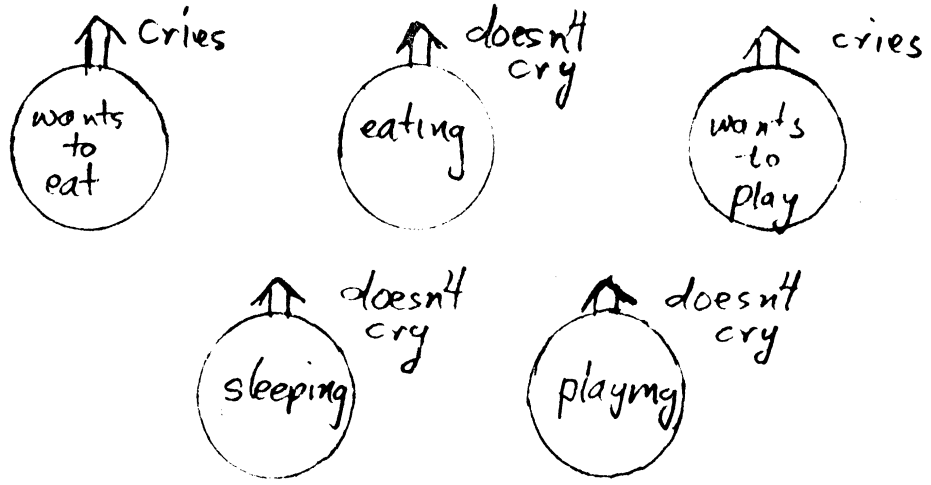
Similarly, interpret the following:



We have now examined all the model's state transition rules i.e., for each state and each input what change in state is brought about.

Remember it was said that states are abstract, not directly observable entities. This being so, there must nevertheless be some means by which different states are distinguishable by observation. Otherwise it wouldn't matter what state the real system was in, we wouldn't be able to tell the difference.

Thus we come to a fourth basic concept. For every state, the model indicates what output the observer would detect if the real system were in that state. In our diagram the output is symbolized by a double arrow as in in the following:.



We interpret this as follows. There are two possible outputs indicated for each state: "cries" and "doesn't cry". The model indicates that in state "wants to eat" the output is "cries". It thus says that when the baby wants to eat, it signals this by crying. Better put, it says that what we detect about the state "wants to eat" is not the state itself but only the observable consequence - the baby's crying.

The importance of this concept is further brought out when we consider that the output from state "wants to play" is also "cries". Thus the model says that the two states "wants to eat" and "wants to cry" cannot be distinguished by direct observation -- if we just came into the room and heard the baby crying we would not be able to tell (from this information alone) whether the baby wants to eat or wants to play. Indeed, this is just the problem we started with, have we made any progress at all! We

will return to this question in a moment.

To finish our presentation of the model, let us note that the states "eating","sleeping" and "playing" all give the same output "doesn't cry". This means that just from the information that the baby is quiet we cannot tell whether it is eating, playing or sleeping. Of course you may feel that, in contrast to the "wants to eat" and "wants to play" states, these states are distinguishable by observation. To do this however, would require that the outputs are suitably expanded. Exercise 3.4 helps you to consider this issue.

Exercises

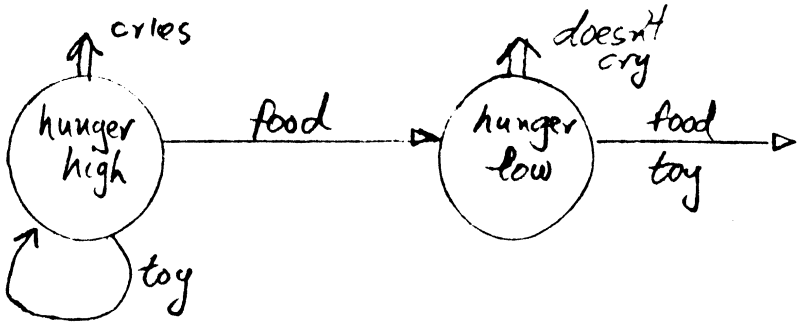
3.1. Draw finite state model diagrams to express the following description:

a. A certain pushbutton switch model has two inputs: "press" and "no press". It has two states:"on" and "off". When the "press" input is received (a person presses the button) the state changes (goes from "on" to "off", or from "off" to "on"); when the "no press" input is received (a person is not pressing the button) the state does not change. The switch controls a light which shines when the switch is "on" and is dark otherwise.

b. A sensor on the tracks triggers the lowering of the gate at a railway crossing. So long as sensor is depressed by a railraod car, the gate remains down. The gate is raised otherwise. Hint: let the inputs be "sensor depressed" "sensor not depressed" and the states and outputs be "gate up" and "gate down".

c. After finishing shopping, a customer goes to the checkout counter. If there is no other customer there, he is served by the cashier; if not, he waits in line until his turn comes. Hint: let the inputs to the customer be "no one ahead" and "someone ahead". Let the states be "head for counter", "wait in line" and "turn to be served". Choose outputs appropriate to observation by the cashier i.e., "serve this customer", and "don't serve this customer".

3.2. In the baby model, suppose the "eating" state is replaced by the following:



Give an interpretation in words of this modification. Suppose also, that if in the "hunger low" state, the baby receives a toy it goes directly into the "playing state". Draw the new model with both the above modifications. Do you feel it is more realistic than the original? Can you suggest further changes that might better reflect your conception of the real baby's behavior.

3.3. A good illustration of the distinction between state and outputs is derived by imagining an automatic soda can dispenser. If one allows for the possibility that someone has put in some coins amounting to less than the total required, then when one first approaches the machine, one can't tell whether it is ready for the full amount or only some part of it. Construct a model of a machine selling 25¢ cans, which accepts nickels, dimes and quarters adding up exactly to 25¢. Hint: let the inputs be 0¢, 5¢, 10¢ and 25¢, the states: "received 0¢", "received 5¢", etc.; and the outputs "release can" and "no release of can". (When a coin is received which would cause overpayment, it is rejected but this is not observed in the output set suggested.) Which states are not distinguished by the "no release of can" output?

3.4. This problem concerns choosing outputs which enable us to distinguish the "eating", "playing" and "sleeping" states. Consider the outputs "moves hands and mouth", "moves hand but not mouth", "moves mouth but not hand", "moves neither hands nor mouth". Assign the outputs to the above states so that each state receives a unique output. What assumptions must be made to justify this assignment?

4. Finite State Modelling Formalism

Let us now return to examine the progress we have made. We have succeeded in expressing the pattern we thought we perceived in the baby's behavior in more full and unambiguous form. This form is our finite state model diagrammed in Figure 1. This form is more complete than our original one because it considers more of the possibilities that would arise if we were asked to describe the pattern in more detail. It is less ambiguous because it leaves less doubt about what happens for each of the possibilities it considers. Indeed in trying to construct a finite state model we are forced to consider more possibilities than we might have originally thought of and we are forced to deal with these possibilities in a clear manner. This is because in finite state modelling we must adhere to the following principles:

1. Choose finite sets of inputs, states and outputs
2. For each input and state combination, choose one and only one transition i.e., the state to which the model will go when it receives the given input while it is in the given state.
3. For each state, choose one and only one output.

If these principles are satisfied we have succeeded in specifying a complete and unambiguous finite state model. Complete because each input-state combination has been considered in 2. and each state has been considered in 3. (It is important to note that the choice of next state in 2. must be one of the set chosen in 1. Similarly in 3.,the output chosen must be one of the ones indicated in 1.)

Unambiguous, because in each choice, only one state (in 2.) or output (in 3.) can be chosen (for example, for the combination "wants to eat" and "toy", we chose "wants to eat", we could have chosen any one of the 5 states but not two or more states as next states).

The principles above govern the construction of finite state models. There are quite a few other principles which govern the construction of other types of models. The name formalism has been given to refer to such principles. Thus the finite state formalism is governed by principles 1, 2, and 3, above. We shall discuss other formalisms for model expression later.

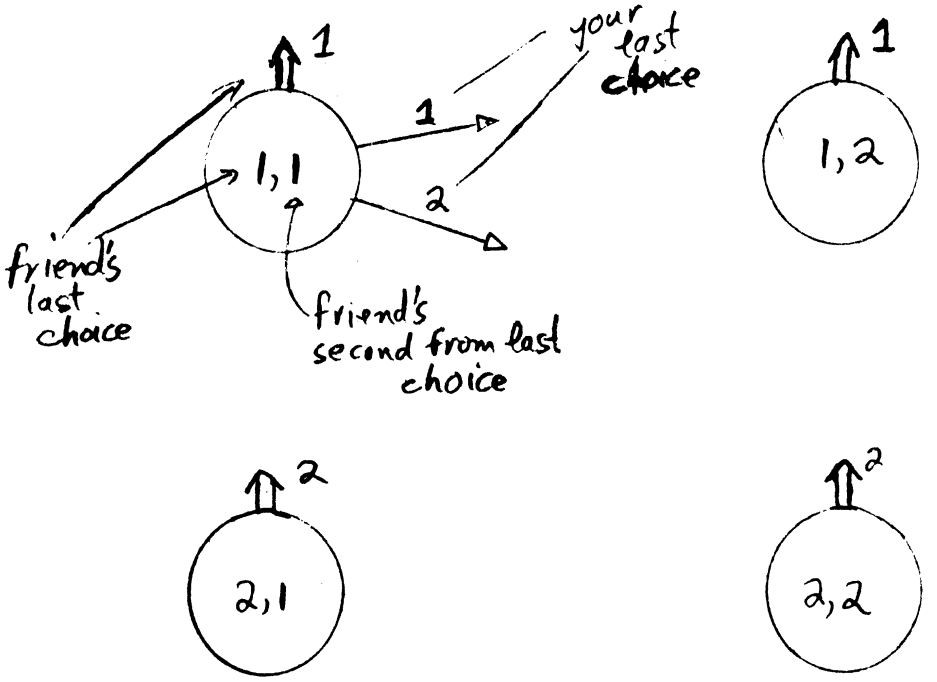
5. Modelling: Means to Express Hypotheses

Having a complete and unambiguous model puts us in a position to better understand the system being modelled. For one thing, the process of modelling gives us a certain familiarity and understanding of our original ideas. This comes from being forced to concentrate on expressing our ideas within a formalism.

Secondly, in making the specific choices required to construct a full and unambiguous model we often don't know beforehand which is the proper choice. The choices that we make without full confidence are our hypotheses about how the real system works. We don't know that they are correct and we have to be able to test our model to find out. Thus a second kind of understanding comes about from our ability to deduce the consequences of our model hypotheses once they are expressed in an appropriate formalism. We shall discuss this idea now for finite state models.

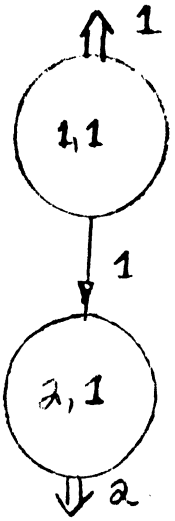
Exercises

5.1. You have been playing the following game with a friend: on each round, each person puts out one or two fingers; if each does the same thing (i.e., the total number of fingers is even), you win; otherwise he wins. After playing a while you notice your friend seems to be following a pattern where he determines his next choice on the basis or your previous choice and his last two choices. You make a model of your friend's strategy using inputs, states, and outputs apparent in the following diagram :

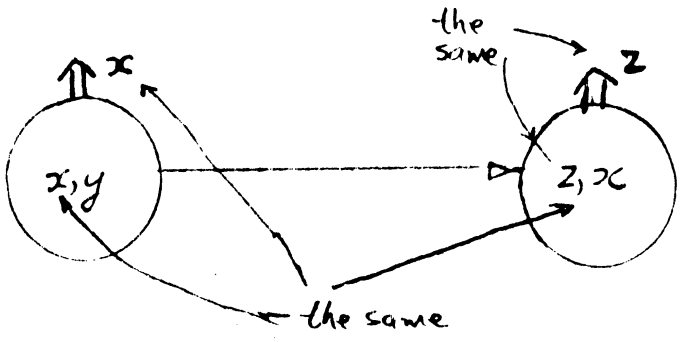


Your next task is to fill in the transitions of the model.

For example, if you write the following :



you are saying that when your friend put out 1 finger in both previous rounds (state 1, 1) and you put out 1 finger on the last round (input to friend is 1) then on this round he will put out 2 fingers. Note that to make sense, the transitions must always be of this form:



Each of the following hypotheses enables you to fill in a complete and unambiguous model. Do so.

a) Your friend's next move is just what your last move was (independent of his previous moves)

b) Your friend's next move is the opposite of his last move (independent your last move and his second of last move)

c) Your friend adds up your last move and his last two moves. If the result is even he puts out 2 fingers; if it is odd he puts out 1 finger.

Each of the following hypotheses may not enable you to fill in a complete and unambiguous model. Try them. Add your own hypotheses, if necessary, to construct a complete and unambiguous model.

d) Your friend repeats your previous move except when he has put out the same number of fingers twice in a row.

e) Your friend sometimes alternates his moves and sometimes repeats them depending on your last move.

6. The Behavior of a Model

Suppose we ask the question: How would the baby respond, according to the model, if when we come into the room, it is in its "wants to eat" state and we give it a toy and then some food, then a toy and some food. Looking at Figure 1, we can see that when it gets the first toy it will remain in the "wants to eat" state. When it gets the food, it will accept it, go into the "eating" state and stop crying. When it gets the second toy, it will in any case go into the "wants to play" state and will cry. The second food offering will keep it in the "wants to play" state and crying.

This description can be summarized by saying that if the model receives the input sequence

toy, food, toy, food

and it starts in the state "wants to eat", then it follows the state sequence

"wants to eat", "wants to eat", "eating", "wants to play", "wants to play" and the observer sees the output sequence

cries, cries, doesn't cry, cries, cries

Such sequences exhibit the behavior of the model. Loosely, behavior refers to what the model says the systems does over extended time periods. When we talk about deducing the consequences of the model hypotheses, we mean working out such behavioral sequences. There are important reasons for wanting to know these consequences which we shall discuss in a moment. First we shall discuss how to work out behavioral sequences in a more orderly fashion. Indeed, we shall recognize that the process is algorithmic so that it can be done by a computer, a fact of fundamental importance, as we shall see.

7. Computation of Model Behavior

The first step toward an orderly computation of behavior is to represent the finite state model in tabular form. Table 1 illustrates this form of representation for our baby model of Figure 1. In Table 1a, we list the five states of the model on the leftmost column. The next two columns are for listing the state transitions caused by the inputs "toy" and "food" respectively. For example, the first row of the table tells what happens if the model is in state "wants to eat" and the input "toy" is presented (the next state is "wants to eat") or the input "food" is presented (the next state is "eating"). To see if you understand this, check some of the entries in the table to see if they agree with the diagram of Figure 1. For obvious reasons, Table 1a is called a state transition table.

Table 1. Tables for Baby Model

Transition Table

present state	next state: input = toy	next state: input = food
wants to eat	wants to eat	eating
eating	wants to play	wants to play
wants to play	playing	wants to play
playing	wants to sleep	wants to sleep
sleeping	wants to eat	wants to eat

a)

Output Table

state	output
wants to eat	cries
eating	doesn't cry
wants to play	cries
playing	doesn't cry
sleeping	doesn't cry

b)

Table 1b, on the other hand, is called output table. It lists the output observed for each state. Again check that the table agrees with Figure 1.

Now let us note that the two forms of finite state model representation we have now seen are equivalent in the sense of conveying the same information. The diagram form sometimes has advantages in enabling us to visualize the model as a whole, especially for relatively simple models. The tabular form enables us, or a computer, to more readily compute behavioral sequences.

Actually this computation can be itself thought of as filling in a table. For example, going back to our alternating input sequence case, let us construct the following table

input sequence	toy	food	toy	food
state sequence	wants to eat			
output sequence				

Looking up Table 1b, we see that the output from state "wants to eat" is "cries". Thus filling the appropriate square gives

input sequence	toy	food	toy	food
state sequence	wants to eat			
output sequence	cries			

Looking up Table 1a, we see that the next state from "wants to eat" with input "toy" is "wants to eat". Thus we get

input sequence	toy	food	toy	food
state sequence	wants to eat	wants to eat		
output sequence	cries			

From Table 1b, the output from "wants to eat" is cries so we have

input sequence	toy	food	toy	food
state sequence	wants to eat	wants to eat		
output sequence	cries	cries		

Proceeding in this way, we fill out the table

input sequence	toy	food	toy	food	
state sequence	wants to eat	wants to eat	eating	wants to play	wants to play
output sequence	cries	cries	doesn't	cries	cries

Check that the new entries are correct. Note tnat the last column has state and output entries even though there is no input specified. In particular, the state entry "wants to play" is the state that the model indicates the baby is in after receiving the input sequence toy, food, toy, food. Suppose that we added a toy at the end of this sequence, then the table would become

input sequence	toy	food	toy	food	toy	
state sequence	wants to eat	wants to eat	eating	wants to play	wants to play	playing
output sequence	cries	cries	doesn't cry	cries	cries	doesn't cry

To make sure you understand how to compute state and output sequences given an initial state and an input sequence, you are asked to do some examples in Exercises 7.3 and 7.4.

Exercises

7.1. Make transition and output tables for each of the models in Exercise 3.1.

7.2. Make transition and output tables for the model in Exerice 3.3.

7.3. Using the baby model in Table 1, fill in the following:

input sequence	toy	toy	toy	toy	toy	toy
state sequence	wants to eat					
cutput sequence						

input sequence	food	food	food	food	food	food
state sequence	wants to eat					
output sequence						

7.4. Using the table made in Exercise 7.1 fill in the following:

input sequence	press	no press	no press
state sequence	off		
output sequence			

(Model 3.1a)

input sequence	sensor not depressed	sensor depressed	sensor depressed	sensor not depressed
state sequence	gate up			
output sequence				

(Model 3.1b)

input sequence	someone ahead	someone ahead	someone ahead	no one ahead
state sequence	head for the counter			
output sequence				

(Model 3.1c)

7.5. Using the tables made in Exercise 7.2 check whether the input sequences "10¢ 10¢" and "5¢ 5¢ 5¢ 5¢" always produce the same final result i.e., either releasing a can or not when starting in the same state.

8. The Structure of a Model

Let us state the concepts we have introduced just now in more abstract and general terms.

The state transition table, of which Table 1a is an example, generally takes the form of Table 2a. This is the transition table of a model which has n states $S_1,\ldots,S_n$ and m inputs $I_1,\ldots,I_m$. The entry in the i th row and j th column is denoted transition (S_i, I_j) and this is, in any particular model, some element of $S_1,\ldots,S_n$. For example, if

transition $(S_i, I_j) = S_k$ then the model indicates that in state S_i when input I_j is received, the next state is S_k. Check that Table 1a has this form for the values $n = 5$ and $m = 2$.

Somewhat more abstractly, transition (. , .) is a function, sometimes called the transition function. This means that to every state-input combination S_i , I_j it assigns a unique element called transition (S_i, I_j). The tabular form is one way to represent such a function, but sometimes it may be more efficient to represent it in a more compact way. For example, suppose that in particular model, the effect of input I_j on state S_i is to put the model in state S_k , where $k = i+j$. Then it would make sense to represent the function by the rule

$$\text{transition } (S_i, I_j) = S_{i+j}$$

rather than write out the equivalent table.

Similarly, the output table, of which Table 1b is an example takes the general form of Table 2b. Check that Table 1b takes this form.

Again, we can think of output (.) as a function, call it the output function. Recall, this function tells for every state S_i , what would be observed, namely output (S_i).

To summarize, we have arrived at a more computational form of describing a finite state model. Let us rephrase the principles of finite state modelling (Section 4) in the new terms:

1. Choose finite sets of inputs, states and outputs
2. Specify the transition function by table or otherwise
3. Specify the output function by table of otherwise.

Table 2. General Form of Finite State Table

Transition Table

present state	Input = I_2	Input = I_2	.... Input = I_i	Input I_m
S_1	transition (S_1, I_1)	transition (S_1, I_2)		
S_2	transition (S_2, I_2)	transition (S_2, I_2)		
. . .	. . .	. . .		
S_i			transition (S_i, I_j)	
. . .				
S_n				

(a)

Output Table

state	output
S_1	output (S_1)
S_2	output (S_2)
. . .	
S_i	output (S_i)
. . .	
S_n	output (S_n)

(b)

9. Simulation of Models

Now we can use our new form of model description to define precisely what we mean by behavioral sequences and to compute them algorithmically i.e., as a computer could. The tables of Section 7 are examples of behavioral tables which take the general form shown in Table 3. In such tables, the input sequence $i_1, i_2, \ldots, i_n$ is specified by the modeller according to his desire to know what the model would do in response to this sequence. So must the initial state s_1 be specified if a unique answer is to be produced. In our example of Section 7, the input sequence i_1, i_2, i_3, i_4 = toy, food, toy, food and the initial state s_1 = "wants to eat".

But from here on all the entries in the table are uniquely determined by the model. Here is an algorithm for filling in behavior tables. We call it a simulation algorithm and will explain why in a moment:

Simulation algorithm

Step 1. Decide on the length of the input sequence, N and fill in the entries $i_1, i_2, \ldots i_N$ in the input sequence row.

Step 2. Decide on the initial state s_1 and mark it as the first entry in the state sequence row.

Step 3. For i running from 1 to N, do the following:

a) Apply the output function to the i ith entry in the state sequence and mark the result as the i th entry in the output sequence, i.e., o_i = output (s_i).

b) Apply the transition function to the i th entry in the state sequence and the i th entry in the input sequence and mark the result as the i+1 th entry of the state sequence, i.e.,

Table 3. Computation of Behavior

input sequence	i_1	i_2	i_3	...	i_N
state sequence	s_1			...	
output sequence				.	

a) Initial appearance of table

input sequence	i_1	i_2	i_3	...	i_{N-1}	i_N
state sequence	s_1	s_2	s_3	.		
output sequence	o_1	o_2		.		

b) Appearance after 2nd step

input sequence	i_1	i_2	i_3	....	i_{N-1}	i_N	
state sequence	s_1	s_2	s_3	....	s_{N-1}	s_N	s_{N+1}
output sequence	o_1	o_2	o_3	....	o_N	o_N	$N+1$

c) Final appearance

s_{i+1} = transition (s_i, i_i) .

Check that algorithm describes the procedure we used in Section 7, by applying it to the input sequence toy,food,toy,food.

The algorithm is called a simulation algorithm because it is an example of algorithms used by computers to simulate models. We can now think of a model as a set of instructions for generating behavioral sequences.

A computer simulates a model by carrying out the instructions of the model. It generates these sequences at the request of the modeller by carrying out a simulation algorithm like the one just described.

By the structure of a model we mean the instructions themselves, as opposed to the behavior of the model, which refers to the sum total of all the state and output sequences that can be generated with these instructions. For finite state models, the structure of a model is established according to principles of Section 8 i.e., by specifying input, state and output sets, and transition and output functions. The behavior of the model is computed on an instance by instance basis using the simulation algorithm.

Exercises

9.1. (For those who know a computer language e.g., FORTRAN.) Write a simulation program that accepts models of up to 10 inputs, 10 states and 10 outputs. When given an initial state and an input sequence, the program produces the corresponding state and output sequences. Hint: Store the transition and output tables as 10 x 10 and 10 x 1 arrays. Write code which implements the simulation algorithm just given.

10. Goals of Modelling: Understanding, Prediction, Control

We stated before (Section 5) that a model embodies hyotheses about how a real system works. To check these hypotheses we have to be able to deduce some of the consequences. We can now rephrase these ideas as shown in Figure 2. In the modelling process, we try to convert our hypotheses into a complete and unambiguous model structure. This structure (or set of instructions) may be used by a computer to generate state and output sequences of the model behavior. These behavior sequences are interpreted by us as consequences of our original hypotheses.

Why go through all this procedure? For one thing, we want to check whether our hypotheses are right. We can do this by comparing the output sequences generated by the model with corresponding observations on the real system. This process is called validation and we shall discuss it further in a moment.

A second thing we are interested in is predicting what would happen according to our model if we took certain actions. In this case we formulate our actions as an input sequence. The model will produce a state sequence and a corresponding output sequence depending on what its initial state is. If we don't know which state characterizes the real system currently, the model may predict many different behavioral sequences. Thus the more certainly we can determine the present state of the real system and the more confident we are in the validity of our model (the correctness of our hypotheses), the more confident we can be that what the model predicts will be true (see Exercise 10.1).

Finally, we want to predict what would happen if we took certain

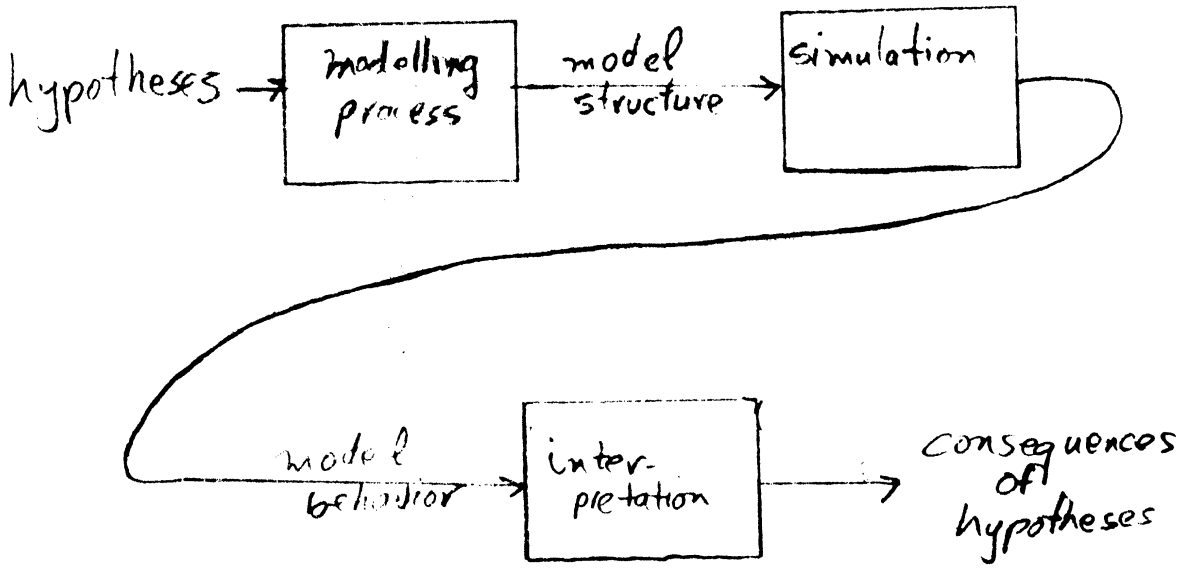


Figure 2. Formalization and Testing of Hypotheses

actions. This is called the control problem. Suppose we have a number of policies which have been proposed to deal with a real system. We formulate each policy as an input sequence. Starting our model from the same state, we generate the behavior sequences corresponding to the input sequence. We choose that input sequence which produces the most desirable behavior sequence as the one to apply in practice (see Exercise 12.1).

Of course, in practice there are many difficulties in deciding what action to take, not the least of which is constructing a model on which we can rely. Models aimed at predicting well into the future tend to be much more complex then the example we have been considering. The problem of gaining confidence in such a model's hypotheses is a very real and difficult one. The essence of the validation process however will be readily stated in the terms we have already developed.

Exercises

10.1. Consider the problem of predicting the baby's response to food or toys. Based on our model, we can make unique predictions if we know the baby's state. What can we predict if we know the baby was just crying and continued to cry when given food? Hint: Find the possible state or states X which agree with following table. Use the subsequent state or states Y to make the prediction.

input sequence	food	
state sequence	X	Y
output sequence	cries	cries

11. The Validation Process

We just now said that the validation process involves comparison of output sequences of the model with real system observations. Note that we said output sequences and not state sequences since as we indicated early on (Section 3) the states are in general not directly observable. Thus in Table 3c we omit the state sequence row to obtain the input sequence - output sequence form shown in Table 4a. For example, for our baby model, starting from state "wants to eat", the table of Section 7 becomes

input sequence	toy	food	toy	food	
output sequence	cries	cries	doesn't cry	cries	doesn't cry

The pair of input and output sequences displayed in such tables are called input-output pairs or I/O pairs for short.

It is important to note that the only way to generate input-output pair tables is to use the simulation algorithm to fill in the tables like Table 3a and then throw away the state sequence. In particular this means that we may get a different output sequence for each state we initially place in Table 3a. For example starting from each of the 5 states in the baby model we obtain the 5 I/O pairs shown in Table 4b) each having the same input sequence. Exercise 11.1. asks you to generate Table 4b.

Each of the output sequences is a possible response of the model. Which one should we see in the real system? The answer is that in general, the best we can say is that we expect to observe one of them but we don't know which. On the other hand, we can say that if we do not observe one of these

Table 4. Input-Output Sequence Pairs

input sequence	i_1	i_2	i_3	...	i_{N-1}	i_N	
output sequence	o_1	o_2	o_3	...	o_{N-1}	o_N	o_{N+1}

a)

	input sequence				output sequence				
1	toy	food	toy	food	cries	cries	doesn't cry	cries	doesn't cry
2	toy	food	toy	food	doesn't cry	cries	cries	doesn't cry	cries
3	toy	food	toy	food	cries	doesn't cry	doesn't cry	doesn't cry	cries
4	toy	food	toy	food	doesn't cry	doesn't cry	cries	cries	doesn't cry
5	toy	food	toy	food	doesn't cry	cries	doesn't cry	cries	cries

b)

output sequences when experimenting with the real system, then our model is definitely not correct.

This leads to the following:

Validation Procedure

0. Start with confidence rating set to 0.
1. Choose an input sequence to try next (this could be one already tried).
2. If the input sequence has never been tried before, generate each output sequence starting from the various states of the model (as in Table 4b). Save the resulting table.

If the input sequence has been tried before, retrieve the table of associated output sequences.

3. Apply the input sequence to the real system and record the output sequence observed.
4. Compare the experimental output sequence (observed in step 3.) with the model output sequences (in the table generated in step 2.)

If the experimental sequence agrees with one of the model output sequences, increase the confidence rating by 1 and go to step 1; otherwise (there is no agreement) stop (the model is not valid).

Figure 3 represents the validation process in more general terms.

The first thing to note is that the above validation procedure is not an algorithm. How to choose a next input sequence to try (step 1) has been left unspecified. Wisely choosing a next input sequence is a major issue which may always need human guidance. It is possible however, to specify

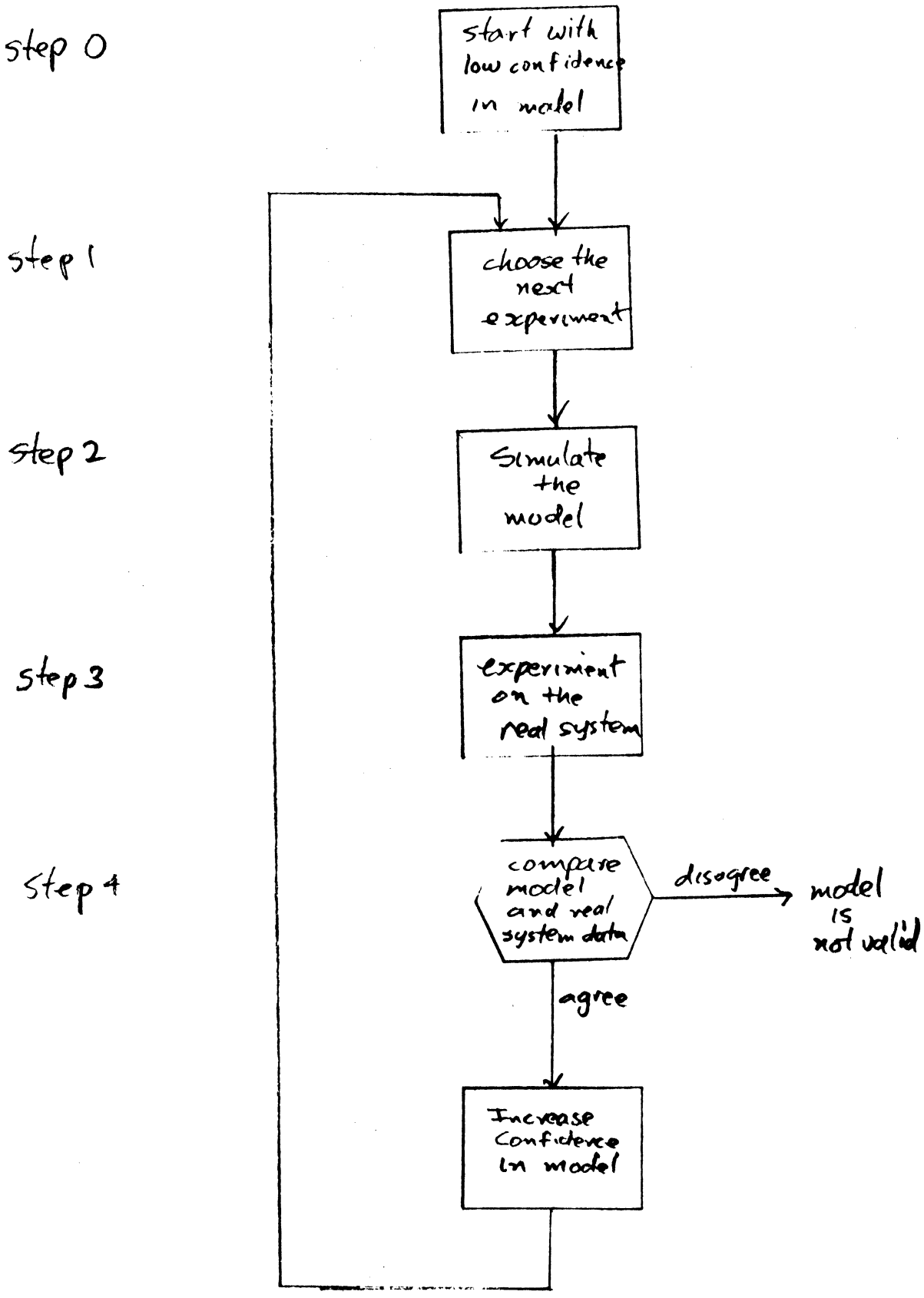


Figure 3. The Validation Process

algorithms which chose input sequences in some arbitrary way such as choosing new sequences in order of length, running through all those of a given length and then going on to the next longer length. To ensure repetition of input sequences, all the ones previously used might be repeated before introducing a new one. Why is repetition desirable? Because we must apply the same input sequence to the real system in its different states.

There is something more fundamental to note however. Even if we formulate the validation process as an algorithm, it is a non-terminating algorithm, i.e., it may never stop. This in the very nature of the validation process. In any experiment we may find a discrepancy between model and real system thus bringing us to a stop. But if our model has passed all the comparisons until now, we are not justifed in stopping because the model may yet fail in the next experiment. Since there are an infinite number ofinputs to try, we never can stop by exhausting all the experiments. Of course, as long as the model doesn't fail, the confidence rating keeps increasing and at some point we may be sufficiently confident in the model to stop the validation process.

We can see why the validation process is as important as the modelling process and is also very difficult. There are some aspects we should mention that are beyond the scope of this discussion. These are: what are reasonable ways of choosing next experiments? Do all output sequences for all the initial states have to be simulated? Do we have to keep repeating the same experiment on the real system indefinitely? What standards of comparison should we use to compare the model and real system output sequences? How do we modify the model when we find that it disagrees with experiment?

Although we can never be sure of it, we can imagine a situation in which a model never fails. In such a case, we say that it is a valid model of the real system. In other words, a model is valid for a real system, if the complete set of input-output sequence pairs of the model equals the set of all input-output sequence pairs observable in the real system. Thus we strive to build a valid model although we can only hope to achieve high, but not complete, confidence in its validity.

Exercises

11.1. Generate Table 4b). Each line in the table is associated with a different initial state - the order of states is that appearing in Table 1.

12.1. You have observed the following input-output pairs in playing the game of Exercise 5.1.

input sequence	1	1	1	1	
output sequence	1	2	2	1	2

which, if any, of the models constructed in Exercise 5.1 would pass the validation test for this I/O pair? Of those which pass, which would still survive after the following experiment:

input sequence	2	2	2	2	
output sequence	2	1	1	1	1

If none survive, pick one of them to modify so as to agree with both I/O pairs? Suggest a third experiment to test the modified model?

12. Validating the Baby Model

Returning to our baby model, we can see that the validation process would be made much easier if we recognized that our ability to make observations about the baby go beyond its crying or not crying. In fact, we can quite confidently distinguish the model's eating, playing and sleeping states by correlating them to distinct actions of the baby (recall Exercise 3.4.). On the other hand, the "wants to eat" and "wants to play" states are not distinguishable by any observations which would normally be available to us. Assuming this, let us see how we might go about validating the model.

Suppose the baby is now sleeping. Then we choose "sleeping" as the initial state of our model. The model says the next state will be "wants to eat" and that we will observe the baby to cry. If the baby does cry, we continue, if not the model is invalid. Now the model says that no matter how many toys we give the baby, it will not stop crying until we give it some food. In other words, we can expect the following input-output pairs:

input sequence	output sequence
food	cries, doesn't cry
toy, food	cries, cries, doesn't cry
toy, toy, food	cries, cries, cries, doesn't cry
.	.
.	.
.	.
toy,...,toy,toy,food	cries,cries,...,cries, doesn't cry

Applying one of the input sequences (with due concern for the baby's discomfort), if we observe the corresponding output sequence, we continue. If not the model is invalid. For example, if in fact, the baby sometimes, or always wants to play after sleeping, then our model predictions would not be borne out. In like way, we continue to carry out the validation of the model.

Suppose that after following the above procedure for some time, we are quite confident in the model's validity. Of what use is the model? We have mentioned the precise understanding of the baby's behavior it affords. We have also mentioned the prediction and control uses of a model. Here for example, assume that our goal is to minimize the baby's crying. Then our model predicts that the following policy is best: If the baby has just finished eating, give it a toy; if it has just finished sleeping, give it food, otherwise it doesn't matter what you give it.

Of course this policy is not a surprising one and we could have come up with it without building our model. But remember that our model allows us to test out policies to see how well they would do when applied to the real system (the usefulness of this of course depends on the validity of the model).

In real life policy decision making there are many conflicting criteria and factors to take into account. Formulating policies and testing them out are thus greatly facilitated if based on complete and unambiguous models in addition to human intuition. Of course such models are likely to be much more complex than our baby model (see Exercise 13.1.).

Exercises

12.1. In playing the game of Exercise 5.1. you naturally want to win as may rounds as possible. Indeed, if your model can uniquely predict your friend's moves you can always do this by putting out the same number of fingers as your model predicts. Suppose the game has begun as shown. Assuming the model partially validated in Exercise 11.2, is indeed valid, and you use the above policy, complete the table.

round	1	2	3	4	5	6	7
your move	1	2					
friend's move	2	1					

13. Other Modelling Formalisms

Earlier, we indicated that there are other modelling formalisms besides the finite state one. These formalisms usually come into play, when trying to build complex models in which the number of states of a finite state model would be very high. While full treatment is beyond the scope of this introduction, some of the common formalisms are briefly described below:

Discrete event models: The modelling principles allow one to specify how long the model stays in its states. For example, the baby model typically spends a lot of time in the "sleeping" state compared to the "playing" state. The state transitions are called "events". Between successive events nothing significant happens in the model.

Discrete time models: The modelling principles force one to specify a fixed time step for the model. Discrete time models may be viewed as a subclass of the discrete event models since the model stays the same length of time in each state. Discrete time models include, but are more general than, finite state models since there is no restriction on the number of states.

Differential equation models: The modelling principles require one to write equations for the rates at which various events occur in the model. For example, the rate at which a chemical reaction occurs might be specified as a certain function of the concentrations of the reactants.

General system models: The modelling principles enable one to specify the model as a general system, i.e., as an interconnected network of components. This is essential for large models. For example, an airport might be described in terms of passengers, ticket counters, clerks, gates, runways and airplanes.

Stochastic models: The modelling principles do not force one to specify exactly one next state but enable one to specify a set of possible states to be selected from, according to a given probability distribution. For example "wants to eat" and "wants to play" might be specified as equally likely next states to "sleeping".

Exercises

13.1. Starting wtih the extensions of the baby model discussed under the discrete event and stochastic models above, discuss ways in which the baby model may have to be refined to better serve as a basis for deciding on a policy for giving it food or toys. Be sure to bring in other criteria by which a policy may be judged such as "exposure to intellectual stimulation", "feelings of security", etc.

14. Summary.

We have discussed two essential processes in modelling and simulation. The modelling process consist of expressing a model within the guiding principles of a modelling formalism. These principles enable the model to be understood as an unambiguous and complete set of instructions for generating input-output behavior. The validation process consists of comparing the model's input-output behavior with that observed in the real systems to be modelled. The goal is to achieve a valid model, one whose behavior is indistinguishable from that of the real system. The actual generation of model's behavior is called simulation and for large models the computer must be employed to do this task. Having attained some confidence in a model's validity, we may use it to predict future real system behavior and/or to ascertain a suitable course of action in dealing with the system (control).