Integrated Continuous and Discrete System Simulation Program (CDSP)

Takeaki Akatsuka*, Akinori Kato* and Nobuyuki Yoshida*

Abstract

There are two types of simulation languages at present, namely the continuous system type and the discrete system type. The former is used in the field of engineering. The best-known one is the "CSMP". The other type is used in Operations Research areas. The "GPSS" and the "SIMSCRIPT" are the most typical of the discrete system type. Nevertheless the actual problems which we must solve are usually inter-related with both the discrete and the continuous system.

Therefore the authors have developed a new simulation program which can handle these complex system. This program is a powerful tool with which to analyze actual programs by means of simulation.

1. Intoroduction

It is the purpose of simulation to study change of the system state when time advances. The simulation program can calculate the system status in the computer. Simulation programs can be devided into two groups according to the way of advancing the time, that is, the discrete and the continuous methods. The continuous system simulation program runs continuously with respect to time and is applied to the problems in engineering described by differential equations. The "CSMP" is applied to this type of systems. The discrete system simulation program is calculated only at the time when the system status changes. The "GPSS" and the "SIMSCRIPT" are of the discrete types and are applied to system analyses in the field of Operations Research expressed by sequential and logical elements.

However, real systems which we want to simulate are not so simple since they do not consist separately of the continuous and the discrete systems but are very complicatedly inter-related with a combination of both systems. In Fig.1 the graph of temperature in the reactor stands for the continuous system and the operation sequences stand for the discrete events. When we study such general systems, it is the usual method to separate discrete systems and continuous systems and to simulate each system independently, or one system is approximately expressed by the other system. Therefore the accuracy of simulation is poor; besides it is often diffcult to build the process model by the conventional simulation programs. The authors have developed an "Integrated Continuous and Discrete System Simulation Program (CDSP)" to avoid deficiency of conventional simulation programs and to be able to study the practical system more efficiently and accurately. This new program would expand the application field of the simulation method.

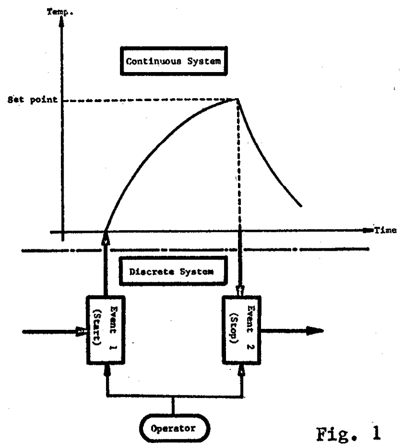


Fig. 1

2. Continuous System and Discrete System

The objects of continuous system simulation are the problems in physical, chemical and engineering processes described by differential equation systems. Therefore time is advanced by integrating the differential equations with respect to time. Integration of the differential equations means the calculation of the future status of the

---

This paper first appeared in Japanese in Joho — Shori (Journal of the Information Processing Society of Japan), Vol. 13, No. 9 (1972), pp, 599 ~ 605.

* Engineering Research Laboratories, Toray Industries, Inc.

system with infinitesimal time steps. It is convenient to show them by graphs since these variables change continuously.

The most important element in continuous system simulation is the "integrator", the function of which is to advance time in such a way that it arrives at the future status of the system as output while its input values are in the present status. The discrete system model is represented by sequential, scheduling and logical relations of each event. It is difficult to express them by mathematical equations.

Future events are forecasted as far as possible by the discrete system model based on the present status. Time and status of each event is queued in the scheduling table according to this discrete model. The timing routine causes an event at the time specified in the scheduling table.

In the discrete system the state would change only when the event occurrs and the state will be held constant between one event and the next. Namely, the state would change discretely with respect to time. The output of simulation are statistical quantities such as the mean, the standard deviation, the maximum, the minimum value of the variables as well as the present value of state variables.

In discrete system simulation the "scheduling table" is the most important element and is equivalent to the "integrator" in the continuous system. The block diagram is the schematic way to describe continuous systems since they are expressed by transfer function or differential equations. This diagram shows the relations between elements. The flow chart is useful to describe discrete systems and is used for computer programming. The flow chart expresses sequential relations of system elements. As shown in Fig.2, the discrete system shown by the flow chart runs by the scheduling table while the continuous system shown by the block diagram advances by integrators. Starting up or stopping of the continuous system is caused by discrete events and discrete events are generated by states of the continuous system. We summarize the features of the continuous and the discrete system in Table 1.

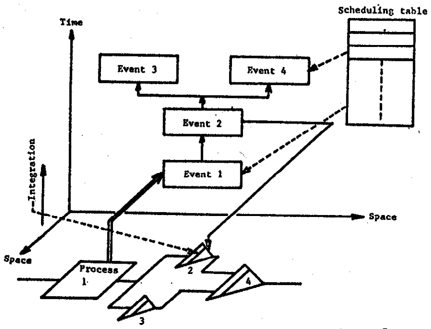


Fig. 2

	Continuous system	Discrete system
Object	Physics, chemistry engineering	Operations research
Model	Differential equation	Sequence, logics
Element	Integrator	Scheduling table
Output	State variables	Statistical data
Schematics	Block diagram	Flow chart
Variable change	Simultaneously	Sequentially

Table 1

3. Program Structure

We chose a method to serve basic element subroutines written in FORTRAN and to build up a model calling these subroutines. Our simulation program executes events in the users' assigned sequence. Therefore with few exceptions users can apply FORTRAN statements without any restriction. As basic elements not only continuous and discrete system elements but also connective elements of both systems are necessary to describe real system models. We want to start integration of continuous system by an event, for example the operator starts the reaction when he has fed raw materials to the reactor.

Though we can easily write the system model with these elements, it is indispensable to have the timing routine which controls the computation flow of the users model to advance time correctly. Then users can run simulation automatically without having to worry about these complicated timing and sequential relationships among the various elements. The system program consists of this timing routine, the standard data input and the standard output printing routines. The sequential flow of simulation is shown in Fig.3. As soon as simulation starts, it reads the standard input data. Then the starting program written by the user reads the data for his model.

After initialization the timing routine starts simulation. In this routine the time and certain state variables are checked step by step to determine whether they are equal to the predetermined values. If conditions are satisfied the discrete event previouly assigned in the scheduling table is executed. If conditions are not satisfied the program calculates the continuous model and integrates it based on the present value. We use the Runge-Kutta-Gill method for integration. The Predictor-Corrector method with variable step size would not be appropriate because of poor convergence when the system has discrete elements.

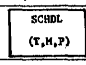
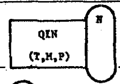
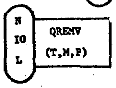
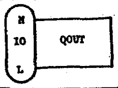
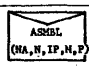
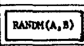
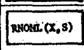
Element	Function	Description	Symbol
Call SCHDL (T,M,P)	Scheduling an entity	Writes scheduled time of the future event T, the event no. M and event informations P(array) in the scheduling table.	
Call CANCL (X,IP,T,M,P, L)	Cancellation of a previously scheduled entity	Removes the entity of which IP-th event information is equal to X from the scheduling table, and gets the scheduled time in T, the event number in M and the event information in P(array) where L=1 succeeded in finding the entity L=0 otherwise	
Call QIN(N,T,M,P)	Registering an entity in a queue	Registers an entity in the N-th queue table with the reference value T, the destination event no. M when taken out of the queue, and the information P(array)	
Call QREMV(N,IO, T,M,P,L)	Taking an entity out of the queue by the reference value	*Takes an entity of the max. reference* value of T when IO=1 or an entity of the min. reference value of T when IO=-1 out of the N-th queue and sets L=1. L=0 if the queue is empty.	
Call QOUT(N,IO,L)	Taking an entity out of a queue by the reference value and scheduling.	Takes an entity of the max. reference value of T when IO=1 or an entity of the min. reference value of T when IO=-1 out of the N-th queue, schedules it and sets L=1. L=0 if the queue is empty.	
Call CATCH (N,M,L)	Catching storages	Catches M units of the N-th storage. Then sets L=1. If it fails to catch,sets L=0.	
Call RELES(N,M)	Releasing storages	Releases M units of the N-th storage.	
Call MATCH(MAT,IC, M,P)	*Waiting for the other entity* (Synchronizing two corresponding entities)	Seeks the conjugate entity with the same data as P(IC) in the MAT-th matching table and if it finds one,schedules. Otherwise registers this in the matching table and waits.	
Call ASMBL(NA,N,IP ,M,P)	Joining entities	Joins N-entities of which information P(IC) are equal.	
Call STATS (N,DATA)	*Statistical Data Collection*	Calculates the average,standard deviation, max.,min.,and frequency of entries of the data X and writes them in the statistical table.	
Call HSTGR (N,DATA,S,DS)	Tabulating in the histogram	Increases the frequency in the histogram by the value of DATA.	
RANDM (A,B)	Uniform random number	Uniform random number in a range (A,B).	
RNOML (X,S)	Random number of normal distribution	A random number of normal distribution with the expected value X and the standard deviation S.	

Table 2

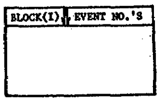
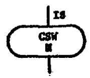
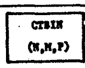
Element	Function	Description	Symbol
Call BLOCK(I)	Block identification	Defines the program between block statements as the I-th block in the continuous system.	
Call BCNTL(I,IS)	Block control	Starts the I-th block if IS=1 stops if IS=-1 Holds if IS= 0	
Call CSW(N,IS)	Connection switch	Transfers to the discrete system when IS=1, but neglects when IS=0.	
Call CTBIN(N,M,P)	Registering in the connection table	Registers the transfer information in the connection table where N is the C-sw. no.; M is the destination event and P(array) is the information.	
J=IEQUL (X,S,IS)	Judging whether the state variable is equal to the set point.	J=1 if X=S J=0 if X≠S where S is the set point and if I=1, under the condition $\delta X/\delta t>0$, I=-1 under the condition $\delta X/\delta t<0$, and I=0 unconditionally.	

Table 3

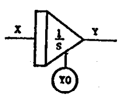
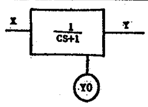
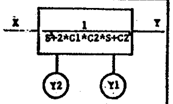
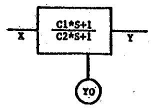
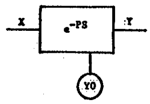
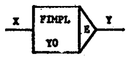
Element	Function	Description	Symbol
Y=FINTE (Y0,X)	Integrator	$Y=\int_{t_s}^{t} X\,dt + Y0$ where t: time, ts: the start time of integration, Y0: value of Y at t=ts	
Y=REALP (Y0,C,X)	First-order lag	$C\frac{dY}{dt} + Y = X$ t: time, Y0: value of Y at the start time of integration	
Y=CMPXP (Y1,Y2,C1, C2,X)	Second-order lag	$\frac{d^2Y}{dt^2} + 2\cdot C1\cdot C2\frac{dY}{dt} + (C2)^2Y = X$ where t: time, $Y=Y1,\ \frac{dY}{dt}=Y2$ at the start time of integration	
Y=FLDLG (Y0,C1,C2,X)	Lead/lag	$C2\frac{dY}{dt} + Y = C1\frac{dX}{dt} + X$ where t: time, $Y = Y0 + \frac{C1}{C2}X$ at the start time of integration	
Y=DELAY (Y0,P,X)	Dead time	$Y=X\ (t-P)$ if $t\geq P$, $Y=Y0$ if $t<P$	
Y=FIMPL (Y0,E,X)	Implicit function	Iterates the computation unless $\lvert Y-X\rvert<E$ when $\lvert Y\rvert<1$ or $\lvert(Y-X)/Y\rvert<E$ when $\lvert Y\rvert\geq 1$ where Y0= is initial value & E is the convergence criterion.	

Table 4

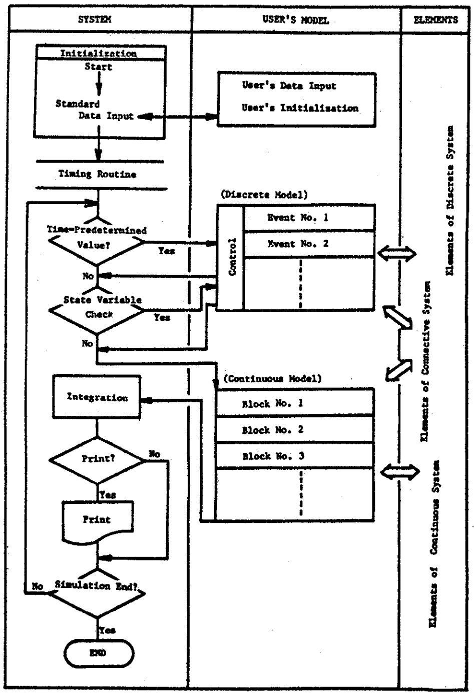


Fig. 3

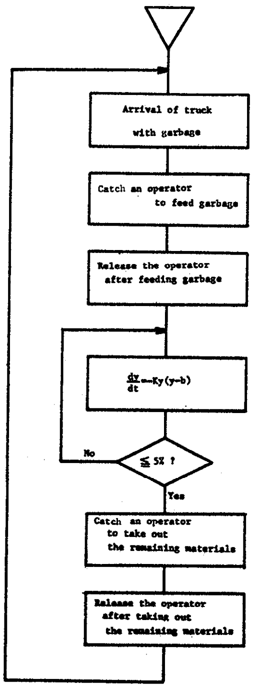


Fig. 4

The program goes to the type out routine if it is type-out time. Otherwise it returns to the beginning of the timing routine after integration by one step. Iterating this computation cycle simulation advances. Each user of this program should write his models by combining element subroutines.

We begin to program a loop in the block diagram at the integrator or at the implicit function since a loop contains the integrator or the implicit function. The discrete system is expressed in the event-oriented method. We divide the flow chart at time elapsing points and consider each block as an event. These events consist of combinations of discrete elements. The simulation program controls computation flow to execute these events at the appropriate time.

4. Elements

We apply most elements in our discrete and continuous systems from conventional "GPSS", "SIMSCRIPT" and "CSMP". The unique connective elements are classified into two groups; one to transfer from continuous system to the discrete system and the other to control the continuous system by the discrete system. The main elements are shown in Tables 2, 3, and 4.

5. Data Files

The transfer of data between the system program and user's models must be carefully designed as every element of this simulation program is written in FORTRAN. We applied labelled COMMON file for variables used only in the system program, blank COMMON for variables among all element subroutines and the user's model, and arguments for variables between a certain element subroutine and the user's model. Blank COMMON also transfers the data within the user's model. There are the scheduling table, the matching table, the assembling table and the queue tables as entity files. These files, along with a blank space are list-structured in order to write and read data quickly and easily. The program has a storage status file to obtain the utilization rate.

6. Input and Output

There are two types of input data: the standard data input for the system program and the optional data input for the user's model. The system program reads the standard data in the fixed format and the assigned sequence. These data are the job number, simulation end time, integration step-size, reporting interval and the storage numbers and unit numbers for use in simulation. Users should write a start routine with read statements in it for their model data.

The output routine for simulation results has two functions: to print the standard report for every time interval and to print the values of continuous system variables. It is able to write arbitrary variables to the printer and/or the external memory during the time interval set by input data. These results are available to the X-Y plotter.

The print out items in the standard report are the reporting time, the contents of scheduling, matching, assembling and queuing tables, maximum and average length of queues, the statistical tables and histograms of the assigned data, efficiency and utilization of storages and values of integrators. The standard report is printed out automatically at the time intervals read in input data. This simulation program can trace the computation by printing out the time and event no. before executing event when the trace subroutine is called.

7. Sample Problem

Let us consider a problem of the garbage furnace process as an example.

(1) Description of Process

The full load garbage trucks arrive at the garbage plant with a normal distribution of one every 15 minutes on the mean and a standard deviation of 5 minutes. There are three garbage furnaces with inlets capable of treating a truckful of load each at one time. The truck goes to an inlet which is not in use, and it takes 5 minutes. The operator gives priority to the furnace which has the least total supply. It takes 10 minutes for him to feed the garbage. The combustion rate is expressed by the following equation:

$$\frac{dy}{dt} = -Ky(b-y)$$

where K is the combustion rate constant
y is the quantity of garbage in the furnace
b is the volume of the furnace.

When the operator has supplied four truck-loads of garbage, he stops garbage supply and lets the furnace keep burning until garbage in the furnace is 5 %

of total supply; then he takes out the remaining material. If trucks arrive late and the amount of garbage is burnt to 5 % of total supply, the operator also takes the remaining content out of the furnace. It takes 10 minutes to take the remaining out of the furnace.

(2) Purpose

We want to discuss by simulation how much garbage can be treated, how busy each operator is and the sequence flow of operations such as how long trucks wait.

(3) Results

We show the flow chart of this problem in Fig.4 by unit of event. Each event has several elements of continuous and/or discrete systems. An example of the standard report is shown in Fig.5. Queue tables are empty since there happens to be no queue at this reporting time. However the statistical data of past queues are printed out as shown.

Fig.6 shows the burning status in each furnace drawn by the X-Y plotter. The computation time of this example for 24 hour simulation is 150 seconds with integration steps of 0.12 minutes calculated by the computer shown in the following section.

(4) Implemented Computer

We implemented this example in TOSBAC-3400/31 with 16 KW core memory. We repeated overlay to run this simulation because of the small core size. This "Integrated Continuous and discrete System Simulation Program" is written in FORTRAN IV. CDSP provides a file size of 100 entities and 50 kinds of storages at present. The number of cards of CDSP source program is approximately 3000.

8. Conclusion

CDSP is the most useful simulation program for the problems described by differential equation models inter-related with operator's actions and operation sequences including start up or, shut-down of plants, batch reaction, traffic control and automatic warehouse. In these problems events are caused by time or the status of the system. CDSP can simulate more efficiently for these types of problems than conventional simulation programs and it can also compute more flexible for continuous systems.

```
          JOB NO.   1   CLOCK   8.000

SCHEDULE TABLE
  TIME     EVENT NO.        INFORMATION
  8.000     1000       0.000    0.000
  8.011        4     250.000    1.000
  8.049        1     250.000    0.000

C-TABLE
  CDSW NO.     EVENT NO.   INFORMATION
     2            5       2.000    0.000
     3            5       3.000    0.000
     1            5       1.000    0.000

MATCH TABLE
  MATCH NO.  MATCH INF.  EVENT NO.  INFORMATION
   EMPTY

ASSEMBLE TABLE
  ASSEMBLE NO.  ASSEMBLE INF.  ASSEMBLE NO.  INFORMATION
   EMPTY

QUEUE TABLE   1
  TIME      EVENT NO.     ,INFORMATION
   EMPTY

Q-NO.   AVERAGE     MAXIMUM     AVERAGE
        LENGTH      LENGTH      TIME/TRANS.
  1      0.087         1          0.139
  2      0.003         1          0.023
  3      0.058         1          0.465

STORAGE
  STORAGE  CAPACITY  AVERAGE   MAXIMUM   AVERAGE     ENTRIES
    NO.              CONTENTS  CONTENTS  UTILIZATION
    1         1       0.62        1       0.6236       25
    2         1       0.37        1       0.3750        4

OUTPUT OF INTEGRATOR
  F(1)=0.  F(2)=0.7232E02  F(3)=0.4100E03
```

Fig. 5

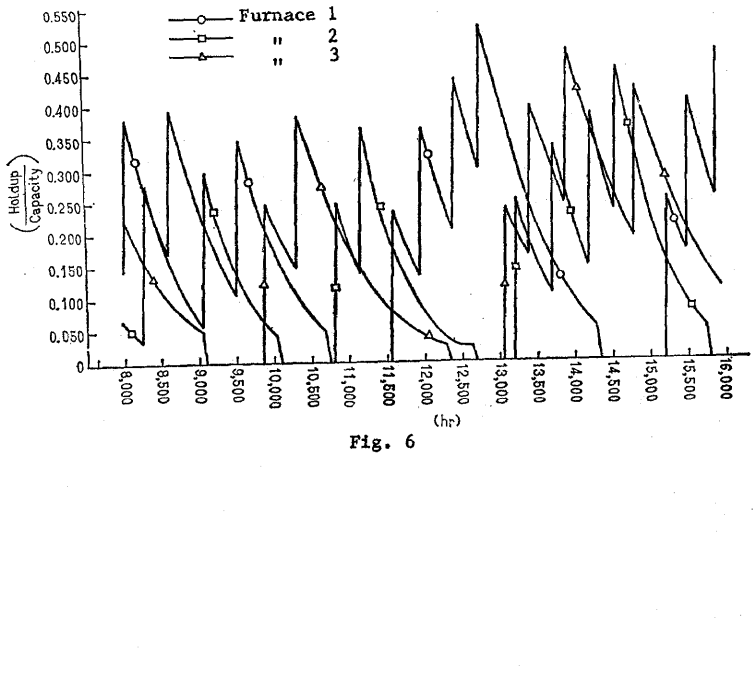


Fig. 6