Introduction to Simulation

This Hypercard stack was prepared by:
Dennis L. Bricker,
Dept. of Industrial Engineering,
University of Iowa,
Iowa City, Iowa 52242
e-mail: dbricker@icaen.uiowa.edu

author

Definitions

System: collection of *entities* which act & interact toward accomplishment of some end.

State of a system: collection of variables necessary to describe the status of the system at any time

Definitions

Monte Carlo simulation (static simulation) representation of a system at a particular point in time

Dynamic simulation representation of a system as it "evolves" in time

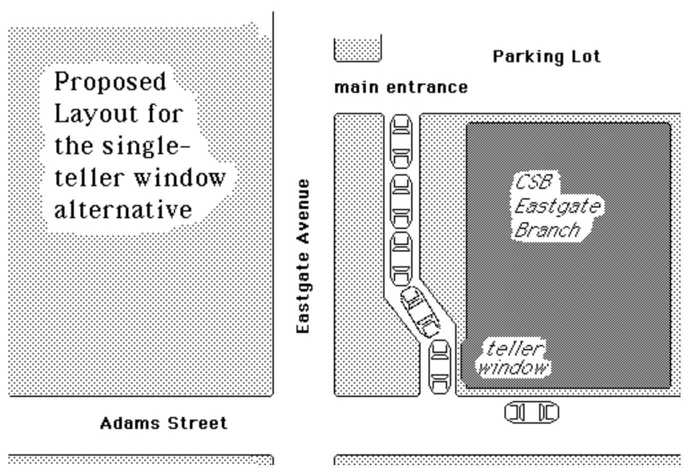
Example of Dynamic Simulation Model

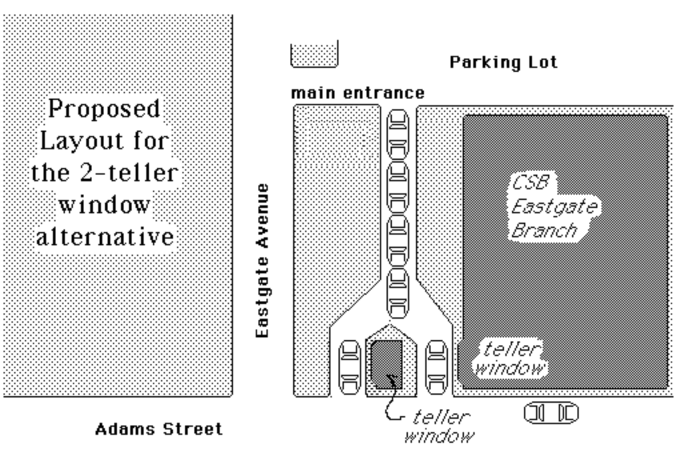
Centerville State Bank plans to include drive-in teller service at a new branch location on Eastgate Avenue.

Should 1 or 2 drive-in windows be included?

To be considered:

- single window saves construction costs and reduces the number of tellers required
- board of directors feels that customers should not have to wait more then 5 minutes for service.
- site layout provides room in the driveway for only 4 waiting cars (in addition to the car or cars being served.) Any additional cars would back up into the main entrance of the parking lot.



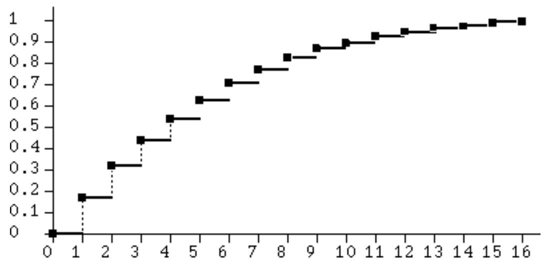


Data was collected on 100 customers arriving at an existing branch office:

Time between arrivals (min.)	Frequency	Cumulative Frequency
1	17	17
2	15	32
3	12	44
4	10	54
5	9	63
6	8	71
7	6	77
8	6	83
9	4	87
10	3	90
11	3	93
12	2	95
13	2	97
14	1	98
15	1	99
16	1	100

Mean value was 5.00 minutes

Interarrival Time Cumulative Distribution Function

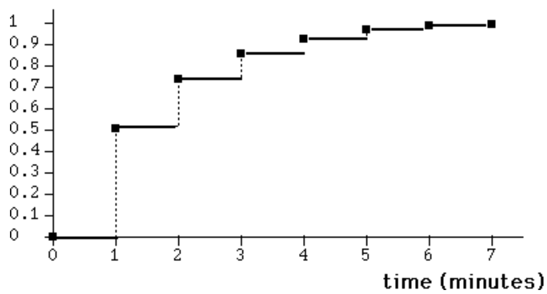


Data collected on 100 customers arriving at an existing branch office:

Service Time (min.)	Frequency	Cumulative Frequency
1	51	51
2	23	74
3	12	86
4	7	93
5	4	97
6	2	99
7	1	100

Mean value was 2.00 minutes

Service Time Cumulative Distribution Function



Using the inverse transformation method, the inter-arrival & service times for 25 cars were randomly generated:

i	T_i^a	T_i^s	i	T_i^a	T_i^s
1	5	2	14	1	1
2	6	4	15	2	4
3	1	3	16	5	1
4	2	1	17	6	4
5	2	2	18	3	1
6	1	4	19	6	1
7	3	1	20	7	1
8	13	3	21	11	1
9	4	1	22	1	7
10	5	1	23	4	2
11	4	1	24	1	1
12	8	1	25	3	2
13	4	2			

T_i^a = *time between arrivals of customers i-1 and i*

T_i^s = *service time for customer i*

Events in this simulation are either

- arrival of a car
- departure of a car

We will maintain two tables:

- a "log" of events which have occurred
- a schedule of events to occur in the future

When an event is "logged", this may trigger the scheduling of other events:

Event being logged	Event to be schedulted
arrival of car #i	arrival of car #i+1 if server was not previously busy, the departure of car #i
departure of car #i	departure of car #i+1

Car i	T_i^a	T_i^s
1	5	2
2	6	4
3	1	3
4	2	1
5	2	2
6	1	4
7	3	1
8	13	3
9	4	1
10	5	1
11	4	1
12	8	1
13	4	2
14	1	1
15	2	4
16	5	1
17	6	4
18	3	1
19	6	1
20	7	1
⋮	⋮	⋮

event log

event #	event type	clock time	server status	Queue Length
0	initialize	0	idle	0

initialize the event log

enter event

event schedule

time	event
5	#1 arrives

Car i	T_i^a	T_i^s
1	5	2
2	6	4
3	1	3
4	2	1
5	2	2
6	1	4
7	3	1
8	13	3
9	4	1
10	5	1
11	4	1
12	8	1
13	4	2
14	1	1
15	2	4
16	5	1
17	6	4
18	3	1
19	6	1
20	7	1
⋮	⋮	⋮

event log

event #	event type	clock time	server status	Queue Length
0	initialize	0	idle	0
1	#1 arrives	5	busy	0

Remove the first event from the schedule, and log it. Then schedule next arrival & next departure

event schedule

time	event
5	#1 arrives
7	#1 departs
11	#2 arrives

Car i	T_i^a	T_i^s
1	5	2
2	6	4
3	1	3
4	2	1
5	2	2
6	1	4
7	3	1
8	13	3
9	4	1
10	5	1
11	4	1
12	8	1
13	4	2
14	1	1
15	2	4
16	5	1
17	6	4
18	3	1
19	6	1
20	7	1
⋮	⋮	⋮

event log

event #	event type	clock time	server status	Queue Length
0	initialize	0	idle	0
1	#1 arrives	5	busy	0
2	#1 departs	7	idle	0

Remove the next event from the schedule, and log it.

event schedule

time	event
5	#1 arrives
7	#1 departs
11	#2 arrives

Car i	T_i^a	T_i^s
1	5	2
2	6	4
3	1	3
4	2	1
5	2	2
6	1	4
7	3	1
8	13	3
9	4	1
10	5	1
11	4	1
12	8	1
13	4	2
14	1	1
15	2	4
16	5	1
17	6	4
18	3	1
19	6	1
20	7	1
⋮	⋮	⋮

event log

event #	event type	clock time	server status	Queue Length
0	initialize	0	idle	0
1	#1 arrives	5	busy	0
2	#1 departs	7	idle	0
3	#2 arrives	11	busy	0

Remove the next event from the schedule, and log it.

Schedule departure & next arrival.

event schedule

time	event
5	#1 arrives
7	#1 departs
11	#2 arrives
15	#2 departs
12	#3 arrives

Car i	T_i^a	T_i^s
1	5	2
2	6	4
3	1	3
4	2	1
5	2	2
6	1	4
7	3	1
8	13	3
9	4	1
10	5	1
11	4	1
12	8	1
13	4	2
14	1	1
15	2	4
16	5	1
17	6	4
18	3	1
19	6	1
20	7	1
⋮	⋮	⋮

event log

event #	event type	clock time	server status	Queue Length
0	initialize	0	idle	0
1	#1 arrives	5	busy	0
2	#1 departs	7	idle	0
3	#2 arrives	11	busy	0
4	#3 arrives	12	busy	1

Remove the next event from the schedule, and log it. Schedule arrival.

time	event
5	#1 arrives
7	#1 departs
11	#2 arrives
15	#2 departs
12	#3 arrives
14	#4 arrives

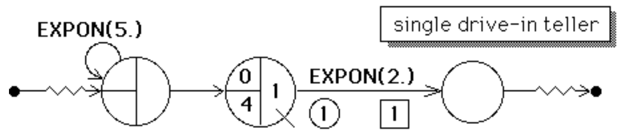
By running this simulation model, the bank can estimate

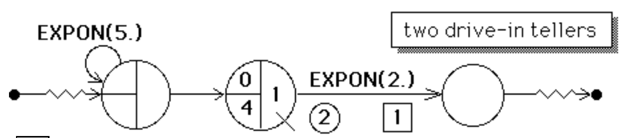
- *the frequencies of customer waiting times*
- *the frequency with which the main entrance will be blocked due to overflow of the queue*

Simulation models may be programmed and executed on a computer, using either a general-purpose language (e.g., Pascal or Fortran), or languages designed specifically for simulation, e.g.,

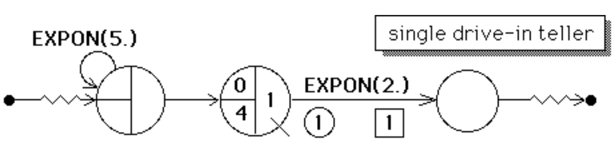
SLAM
SIMSCRIPT
GPSS
SIMULA
SIMAN
DYNAMO
& many others

SLAM utilizes network model of the system.





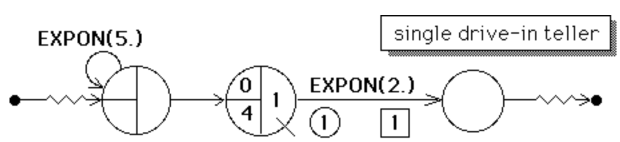
Goodness-of-Fit Tests for Dist'ns




```
GEN,BRICKER,BANKTELLERS,2/1/1993,,,,,,,72;
LIM,1,1,50;
INIT,0,480;
NETWORK;
       CREATE,EXPON(5.0);
       QUE(1),0,4;
       ACT(1)/1,EXPON(2.0);
       TERM;
       END;
FIN;
```

Simulate 480 minutes of operation of the drive-in bank window

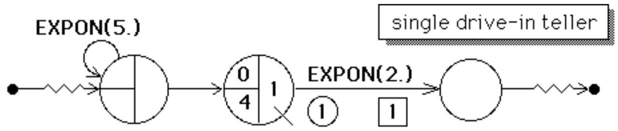
(assumes exponential distribution for both interarrival & service times.)




```
**INTERMEDIATE RESULTS**

***WARNING - ENTITY LOST AT TIME   0.4081E+03***
            FILE   1 IS FULL.
***WARNING - ENTITY LOST AT TIME   0.4120E+03***
            FILE   1 IS FULL.
```

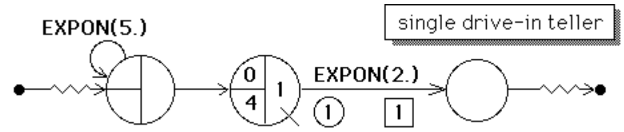
At 408.1 and 412 minutes into the simulation, two cars arrived while the waiting line is full (with 4 cars)!



FILE STATISTICS

FILE NUMBER	LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	QUEUE	0.359	0.856	4	0	1.580
2	CALENDAR	1.448	0.497	3	1	2.594

(The maximum waiting time was not reported.)

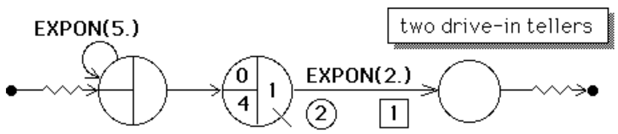


SERVICE ACTIVITY STATISTICS

ACT NUM	ACT LABEL OR START NODE	SER CAP	AVERAGE UTIL	STD DEV	CUR UTIL	AVERAGE BLOCK	MAX IDL TME/SER	MAX BSY TME/SER	ENT CNT
1	QUEUE	1	0.448	0.50	0	0.00	17.35	29.23	109

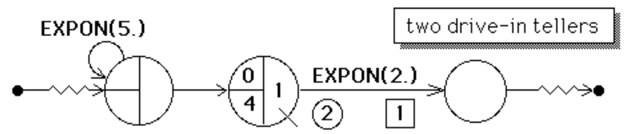
Fortran STOP

The teller was busy 44.8% of the time.




```
GEN,BRICKER,BANKTELLERS,2/1/1993,,,,,,,72;
LIM,1,1,50;
INIT,0,480;
NETWORK;
        CREATE,EXPON(5.0);
        QUE(1),0,4;
        ACT(2)/1,EXPON(2.0);
        TERM;
        END;
FIN;
```

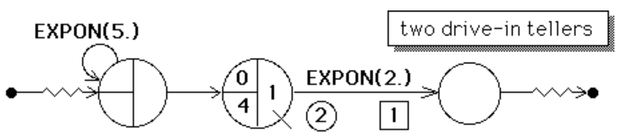
Now we wish to simulate the system with TWO tellers.



FILE STATISTICS

FILE NUMBER	LABEL/TYPE	AVERAGE LENGTH	STANDARD DEVIATION	MAXIMUM LENGTH	CURRENT LENGTH	AVERAGE WAIT TIME
1	QUEUE	0.024	0.155	2	0	0.096
2	CALENDAR	1.519	0.683	4	3	2.883

At no time did the queue have more than 2 cars (not including those being served).



SERVICE ACTIVITY STATISTICS

ACT NUM	ACT LABEL OR START NODE	SER CAP	AVERAGE UTIL	STD DEV	CUR UTIL	AVERAGE BLOCK	MAX IDL TME/SER	MAX BSY TME/SER	ENT CNT
1	QUEUE	2	0.519	0.68	2	0.00	2.00	2.00	116

Fortran STOP

Average # of busy tellers was 0.519, so that each teller was busy about $0.5 \times 0.519 = 26\%$ of the time.

Simulation

- Creating a model of a real or proposed system for the purpose of evaluating the system's behavior for various conditions
- Allows the analyst to draw inferences about new systems without building them, or make changes to existing systems without disturbing them.
- Allows system interactions (system integration) to be analyzed
- Permits managers to visualize the operation of a new or existing system under a variety of conditions.
- Helps understand how various components interact with each other, and how they effect overall system performance.

Why Simulate?

- Provide general insight into the nature of a process
- Identify specific problems or problem areas with a system
- Develop specific policies or plans for a process
- Test new concepts and/or systems prior to implementation
- Improve the effectiveness of a system

Simulation...

- cannot optimize... it can only describe the results of "What-if" questions
- cannot give accurate results if the data are inaccurate
- cannot describe system characteristics that have not been explicitly modeled
- cannot solve problems... it can only provide information
- cannot provide easy answers to complex problems

Project Management: The 'Players'

Simulation project team

System design team

Data/information sources

Implementation team

Contractors

Decision-makers/ management

The Simulation Process

DEFINE -- functional specification

FORMULATE -- the simulation model

VERIFY/VALIDATE -- input from all players

ANALYZE -- statistical evaluation

RECOMMEND -- alternatives to the decision-maker(s)

Why Have a Functional Specification?

- Defines the problem completely
- Requires system understanding from the start
- Provides vision of the task
- Defines how the simulation will be used
- Defines all assumptions of the simulation model
- Identifies data requirements
- Identifies required output statistics and analysis

A Functional Specification

- Objectives
- Assumptions
- Inputs
- Outputs
- Control logic
- Level of detail
- Flexibility
- Analysis

Model Formulation

One or more analysists

Data structure requirements

Model control logic

Level of detail

Flexibility

Statistical requirements

Verification & Validation

VERIFICATION: Ensuring that the model behaves in the way it was intended

VALIDATION: Ensuring that the model behaves the same as the real system

Requires:

- Involvement of all the players
- Use of animation and data
- Reasonable and robust model