AN APPLICATION OF SIMULATION TO COMPARE PRODUCTION LINE CONFIGURATIONS WITH FAILURES AND REPAIRS

F. Paul Wyman
Lawrence E. Moberly

Department of Management Science and Organizational Behavior

The Pennsylvania State University
University Park, Pennsylvania

Summary

Three parallel production line configurations are compared by simulation: parallel but independent, dual line without expediting, and a dual line with expediting possible around a failure station. Each facility within a stage draws parts from a common queue in a dual line. Relative performance is studied while varying queue capacity, failure rate, repair rate, and number of stages. The preferability of a dual over independent lines is found to depend upon the degree of expediting possible. Expediting, in turn, depends upon the degree of idle machine capacity built into the work standards.

Introduction

The two most important parameters in production line* design are the output rate and the cost of obtaining the output rate. It is possible to maximize the output rate of a production line at a fixed cost by changing the configuration of the line.

A considerable amount of research has been concerned with production line configurations, especially with configurations along the length of the line. The classic study of this sort is called "assembly-line" balancing in which elementary tasks are grouped together to determine the total work that is to be accomplished by each station. The set of stations along the line is the resulting configuration.[9]

Width, rather than length, is considered here by replacing one station of the ordinary production line with two identical parallel stations. This will hereafter be called a dual production line. (See Figure 1 for a physical model of the system.)

It may be possible with dual production lines to have the capability of expediting parts through a station that is parallel to a failed station. Expediting is done by doubling the service rate of a station. Doubling the service rate of the non-failed station may be accomplished in several situations. In an assembly line each station could be designed to permit workers on both sides of the line while that station's conveyor speed is doubled. Modular machine tools have two to six multi-purpose heads so that more than one worker can be setting up and processing work on a single machine. In general, the actual rate of production in a station would have to be half or less of the absolute maximum rate of production in order to justify the "doubling" assumption in expediting cases. If actual usage were greater than half the maximum rate, say 75%, then obviously, expediting would not double production. However, the simulation in this paper could be replicated using the appropriate factor for expediting. Intuitively, we expect that a smaller factor for expediting would tend to lessen the attractiveness of a dual-expedited line relative to dual or parallel independent lines, but should not affect the runs of the regular dual line without expediting.

At issue in this study is a decision rule for management. Management has the alternatives of arranging two single independent lines or arranging one dual line (or expedited dual line) from facilities available or proposed for purchase. Which of these configurations will give the greatest output rate?

*The term "production line" is used here because it is more precise than the more popular term, "assembly line."

Discussion of Analytical Models

Much of the research that has been done in the area of dual production lines has employed highly restrictive and sometimes unrealistic assumptions. Among the separate tools that have been used are analytical queuing models[7] and reliability models.[10,11] The queuing models deal with the parts flowing through the system and make simplifying assumptions about the parts flowing through the system.

To combine these two tools into a single analytical model and to release the restrictive assumptions used by each would result in a very complex model. Accomplishing the task would be worth the effort if there were no simpler methods.

In addition to the potential complexity of analytical approaches there may be no way to verify the results obtained. Single production lines are in common use, so the analyst can validate his predictions. Dual lines are not commonly in use, so a method which visibly approximates the process is highly desirable.

The Simulation Approach

The approach taken here attempts to encompass most of the essential variables of a production line by using computer simulation.

The production line can be visualized as a multi-dimensional "mathematical space" where each dimension may be a queue capacity, a number of stations in length, a number of stations in width (one or two here), and failure rate, a repair rate, and an output rate. The simulation model is essentially a mathematical function on this space with output rate being the dependent variable. This may be formalized as follows:

$$r = f(Q_i, F_i, R_i, S_i, N, W)$$

where

r = Output Rate
Q_i = Queue Capacity Proceeding Each Station
F_i = Failure Rate for Each Station (distribution)

R_i = Repair Rate for Each Station (distribution)
S_i = Service Rate for Each Station (distribution)
N = Number of Stations in Length
W = Number of Stations in Width

By specifying various independent variables and running the simulation, one can obtain the output rate with a specified degree of certainty. The degree of certainty is established by taking a number of samples of the output rate when the system is in a stable operating state. The stable operating state is achieved when the effects of initial bias are gone.[6] Achieving a stable operating state requires finding a representative state in the system and starting the system at this state.[4] The batch mean method with adjoining batches was used. The effect of autocorrelation was made negligible using an adequate interval size.[3]

Instead of finding the function of output rate, comparisons were made of selected iso-cost points on the "mathematical output-rate space." If two configurations cost the same and the output rate of one is significantly greater than the other, then one configuration must be better than the other. Each test was concerned with dual production lines. One compared dual-expedited to regular dual lines, another compared dual-expedited to single lines, and the remaining six compared dual lines to single lines.

The null hypothesis is that both configurations produce at exactly the same output rate, that is, the difference in their rates is zero.[13] Comparing a single line with a double line is accomplished by specifying that both have the same queue capacities per line, the same failure and repair rates, the same service rates, and the same number of stations. Specifying that the single line has the same number of stations is not accomplished by extending the line, but rather by assuming that one independent line will produce at exactly one-half the output of two equal independent lines. The adjustment for this is made in the estimate of the standard error of the difference.[2] The calculations were made as follows: Sum r_i and Sum r_i^2 were gathered by the program. r_i is the output rate of configuration i. n_i is the sample size.

$$r_i = \frac{\text{Sum } r_i}{n_i} \qquad \text{(Mean)}$$

$$S_i^2 = \frac{\text{Sum } r_i^2 - \frac{(\text{Sum } r_i)^2}{n_i}}{n_i - 1} \qquad \text{(For Standard Error)}$$

$$d = r_D - 2r_I \qquad \text{(Difference between dual and two independent single lines)}$$

$$S_d^2 = \frac{S_D^2}{n_D} + \frac{2^2 S_I^2}{n_I} \qquad \text{(For Standard Error of the Difference)}$$

Obtaining Data

Data for the independent variables required generality, because there is considerable variation of parameters among production lines. The mean service time (the time to process a part through a station) formed the basic time unit (1.0). A normal distribution, truncated at zero, was used, and the standard deviation of the service time was 0.3 for all stations. This was computed from another study[1] that used a coefficient of variation[13] of approximately 30% for service times. Although the standard deviation could have been different for different stations, there was no advantage in attempting this. Since the line is well-balanced the mean service time should be approximately the same for all stations.

Failure rate was not expressed as a rate, but rather as a mean time to next failure AFTER a repair (MTTNFAR). It incorporates not only the failure of equipment at the station, but also the possibility of a stockout of sub-parts that are used by the particular station. Obtaining values for this variable required finding both the probability of equipment failure, and the probability of stockout.[5] Similarly, data for the mean time to repair (MTTR) incorporated both time to repair equipment and time to obtain sub-parts after a stockout. The number of stations (N) was the average number used in other studies.[1,5,9] Queue capacities (Q) were based on a study by Barten[1] in which he found that a satisfactory size was about 4 for a single line.

The above four variables had two values each. One of the values was called a "base" value and was used in all tests but one. In the particular test where the base value was not used, an "altered" value was used. In this manner, it was possible to determine the effect of that variable.

In summary, the values of the variables are:

Base values:
MTTNFAR = 1,000 time units
MTTR = 5 time units
N = 10 stations
Q = 4 parts

Altered values:
MTTNFAR = 50 time units
MTTR = 50 time units
N = 6 stations
Q = 8 parts

Assumptions

Since a production line is fairly simple, it is possible for a simulation program to model most of the "real world" features. The following were assumptions used:

1. As the program operates, the input queue is assumed to be always full. In a production line, the first (arriving) parts are actually sub-parts. A stockout of these parts is incorporated in the failure rates of the first stage of stations and the replenishment of them is incorporated in the repair rates.

2. The output feeds an infinite queue capacity. This assumption may not be valid in the real world, but if output became too overwhelming, the line would be stopped. Of interest here is the line when it is working, not when it has been stopped.

3. The line is perfectly balanced, or in other words, the mean service times are the same for each station. No line is really perfectly balanced, so the problem is deciding which stations should be imbalanced and how much they should be imbalanced. Any choice would be arbitrary, so the choice of no imbalance was chosen.

4. There was no provision made for the length of working days. It can be assumed that workers and repairmen leave the line in exactly the same state that they find it the next morning. It is

possible in the real world that repairs will be implemented and that the stocking of parts will occur when the line is shut down for the night. It is also possible, however, that the line is a 24-hour operation which is assumed here.

5. Each of the single lines has a queue capacity equivalent to that of the dual line. This means that two single lines have twice the capacity of a dual line. It was implicitly assumed that the cost of doubling queue capacity is equal to the cost of constructing common queues (as in a dual line) between lines.

6. Each of the stations had the same failure rates, repair rates, and queue capacities. It is unlikely that this would be the situation in the real world, but it had to be assumed to find the general effect of these variables.

The Programming Language

The selection of SIMSCRIPT II as the programming language was based on familiarity with the language[8,14]. The model was complex enough that algorithmic languages were ruled out, but not so complex that the selection of a particular simulation language was critical.

However SIMSCRIPT was preferred to GPSS for explicit reasons. Changing the number of stations did not require recompilation since the number of entities is read in at execution time in SIMSCRIPT. Re-coding and debugging checks would have been required in GPSS for changes in the system structure. Also the output of SIMSCRIPT was more flexible and could be more judiciously extracted than with GPSS. Furthermore, the SIMSCRIPT II preamble permits an English-like description of the system structure, providing self-documentation.

It was a simple matter to make the program more general than was used in this study. With the same program, it is possible to investigate dual, dual-expedited, and single production lines. Each station can have a different mean failure time, mean repair time, mean service time and variance, or a different preceeding queue capacity. There can be any number of stations along the length of the line. The size of interstage queues can also be varied.

With minor program changes it would be a simple matter to determine the idle time of each of the stations, change the distributions, or have some stations with parallel facilities, and others with a single facility.

The Program Model

The program model is conceptually simple (See Figure 2). There are four events called FAIL, REPAIR, SERVICE, and COMPLETE. Each of these four "working" events has arguments denoting the particular STATION and QUE (permanent entities) with which it is involved. The number of unique working events is therefore four times the number of stations.

Representing the state of the system are the attributes of the entities. These attributes are changed by the events. Attributes that are changed are: SIZE.OF (QUE), TIME.OF.COMPLETION(STATION), ABORT(STATION), FAIL.MODE(STATION), HOLD(STATION), and CYCLE.MODE(STATION). ABORT is utilized whenever it is desired to end the use of that station in the program. FAIL.MODE prevents a SERVICE or a COMPLETE from occuring whenever the particular station is in a state of failure. HOLD registers whether the station is holding a part. It becomes 1 when service begins and becomes 0 whenever service is completed and the queue following the station is able to take the part. It also prevents parts from disappearing when a station fails. CYCLE.MODE is used in those configurations where expediting of parts through a station adjacent to a failed station occurs, i.e., in dual expedited lines.

The parts moving through the system are not "entities" in the program. It was more convenient to keep track of parts by using attributes of STATION and QUE. Since the input queue is always full, parts are automatically generated when needed (see assumption 1). As a part goes into a station, HOLD(STATION) changes from 0 to 1. The converse occurs when a part leaves a station. When a part is placed in an interstage queue (by a station), SIZE.OF(QUE) is increased by 1. When a part is taken from an interstage queue the converse occurs. The last station passes parts into the variable PARTS which is used in computing the output rate (Figure 1 may aid in visualizing this process).

A station is temporarily independent of other stations when the queue that preceeds the station is not empty and when the queue that follows the station is not full. A station will become dependent on its predecessor if the preceeding queue is empty (SIZE.OF (QUE) = 0). It must wait until its predecessor completes service on a part and places the part in the queue. Once the part is in the queue, the station can begin service on it. The time that the station may begin service on the part is therefore dependent on the TIME.OF.COMPLETION of the preceeding station. This value may be found with the routine SEEK.TIME which finds the minimum TIME.OF.COMPLETION of the preceeding stage of stations.

A station also will become dependent on its successor if the succeeding queue is full. It must wait until its successor draws a part from that queue, making the queue available for parts. The successor will draw a part from the queue whenever it disposes of the part that it is servicing. Therefore, TIME.OF. COMPLETION of the succeeding station is the time that a COMPLETE may be scheduled for the station waiting for queue space. In this case the routine SEEK.TIME finds the minimum TIME.OF.COMPLETION of the succeeding stage.

Assigning a value to TIME.OF.COMPLETION may be done in the events SERVICE, COMPLETE, or FAIL. If a station has a part in service, it does not change the status of preceeding or following queues until service is complete, so a TIME.OF.COMPLETION is assigned for the time that a service is to be completed. If a station has completed a service, but a following queue is full, or if a station is trying to accomplish a service, but the preceeding queue is empty, then the TIME.OF.COMPLETION of successor or predecessor stations becomes the TIME.OF.COMPLETION of the station. If a station fails, then TIME.OF.COMPLETION becomes the time the station is repaired, plus any time required to finish a part that had been in service.

An event TRANSIT was used to periodically gather statistics, to cause a transition of the program into another configuration, and to activate ABORT for all stations, thereby ending the program.

A set GROUP was used. The GROUP was owned by a QUE and consisted of parallel STATIONS which were co-members. The GROUP served the purpose of identifying stages of stations and was used in conjunction with the attribute SIDE.OF(STATION) to identify a particular station.

Several attributes in the program are unchanging, such as FAIL.RATE, and are read-in at the start of the program. Certain routines were written when particular implementations of SIMSCRIPT II were found to be unavailable locally.

Program Development

Before going into the results obtained from the study, it is worthwhile to review the development of the program and the examination of it for proper functioning.

Using figures from the local accounting system, the program of 228 lines cost $158.27 to develop. The principal component of this cost was computer time. The cost of running each of the 12 configurations was in the range of $7.00. Approximately, a man-month of human resources were required to develop and gather the results.

There were six errors in the original program, four syntactical and two logical.

To insure proper functioning of the program, print statements were inserted in every routine and event. The program was then examined carefully to make sure that the results were a measure of the assumed process.

Results

Table I is a summary of the statistics gathered for certain variables and configurations, including the sum of output rates, the sum of output rates squared, the mean output rate, and the variance of the output rate. The sample size in all cases was 40. The null hypothesis was: The difference in output rates of the two configurations is zero or less.

$$d \leq 0$$

The standard normal deviate was computed as,

$$K = \frac{d - 0}{S_d}$$

There are three possible configurations: Two independent single lines (2 x I), dual line (D), and dual line with expediting around failed stations (DE). The variables are failure rate (MTTNFAR), repair rate (MTTR), number of stations (N), and queue capacity (Q). Table II shows the configuration pair being compared, the variable that was altered from the base, the difference in the output rates of the two configurations, the standard error of the difference, and the standard normal deviate resulting from the assumption that the difference is zero.

The type I error allowed was 10% and the type II error was 0.06% when rejection did not occur. In the one dual versus single comparison, in which the base variables were used (test 4), there was no significant difference in the output rates of a dual line and two independent single lines. If another test with the same base variables, excepting one altered variable, has a significant difference in output rates, we can conjecture that the altered variable has the effect of making the dual line superior to the two single lines.

In test 2, the altered variable was the number of stations (N). Decreasing the number of stations from ten to six resulted in a significant difference in the output rates. The dual line's output rate was approximately 2% better than the two single lines. Therefore, we can conclude that the advantage of dual lines is greater for shorter lines. If all other variables remain the same, the advantage is likely to be lost for longer lines.

In test 3, the altered variable was queue capacity, which increased from 4 to 8. Increasing queue capacity also increased the relative advantage of dual lines with the dual line's output rate being about 28% better than the single line's. This result must be viewed in the light of assumption 5. It should also be noted that increasing queue capacity for the single line from 4 to 8 (Run 8 and Run 6) did not greatly increase the output rate of the single line, confirming Barten's study.[1]

Increasing the failure rate, or equivalently, decreasing the time between failures (test 5) showed that failure rate influences the relative advantage of dual lines. By changing the time between failures (MTTNFAR) from 1,000 to 50, the output rate of the dual line became approximately 7.5% better than the single lines.

Increasing the time it takes to repair a station from 5 to 50 time units resulted in a significant advantage for dual lines. The dual line's output rate was about 39% better, indicating that if it generally takes a long time to repair stations, then it may be advantageous to use a dual line.

When both severe failure and repair conditions afflict a production line as in case 1, dual production lines are significantly better with output rate of the dual line being over 200% better than the single line.

In tests 7 and 8, expedited dual lines were used. Expedited dual lines may cost more than regular dual lines. Any improvement in output rate should be compared with the added cost of having a system that can expedite. The previous comparisons were made with systems that were assumed to cost the same.

Test 7 compared expedited dual lines with single lines. The expedited line was an improvement over the single lines by less than 2% in this test. With the same variables a regular dual line showed no improvement (test 4).

Test 8 compared an expedited dual line with a regular dual line. It was found to improve the output rate about 2.5%, but this figure is fairly consistent, since the standard error is low.

Conclusions

In one test, a dual production line was found to be no better than two single independent production lines on an equal cost basis.

Dual production lines were found to be better than two single independent production lines when the following changes in mutual variables occurred:

(a) An increase in interstage queue capacities.
(b) An increase in time to repair.
(c) An increase in failure rate.
(d) A decrease in number of stations.

When two of the above changes occurred (b, c), the advantage of dual lines over single lines increased greatly.

In experimentation with expedited dual lines it was found that:

(e) An expedited dual line is better than a regular dual line.

(f) With the same variables, an expedited dual line was found to be better than two single lines while a regular dual line was not.

In single assembly-line balancing, there is a point at which it becomes too costly to increase output rate by increasing line length due to the resulting imbalance. At this point it would be advisable to consider constructing a dual line rather than another single line to increase output rate. A dual line has the added advantage of possibly becoming an expedited dual line.

It is suggested that further experimentation be conducted to find a general function for predicting the percentage improvement that a dual line would bring to a pair of single lines when the variables of each are the same.

The simulation program developed for this study can be used for either further research or for application in actual production line situations. It would be especially useful for determining the validity of analytical models that attempt to incorporate the variables used in this study.

Bibliography

1. Barten, Kenneth, "A Queuing Simulator for Determining Optimum Inventory Levels in a Sequential Process," Journal of Industrial Engineering, XIII, no. 4 (July-August, 1962), pp. 245-52.

2. Chou, Ya-lun, Statistical Analysis, Holt, Rinehart and Winston, Inc., New York, 1969.

3. Conway, R. W., "Some Tactical Problems in Digital Simulation," Management Science, Vol. 10, no. 1 (October, 1963), p. 47.

4. Conway, R. W., Johnson, R. M., and Maxwell, W. L., "Some Problems of Digital Machine Simulation," Management Science, Vol. 4, no. 1 (October, 1959).

5. Gavett, J. William, Production and Operations Management, Harcourt, Brace and World, New York, 1968.

6. Gordon, Geoffrey, System Simulation, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1969.

7. Hillier, Frederick S. and Lieberman, Gerald J., Introduction to Operations Research, Holden-Day Inc., San Francisco, 1968.

8. Kiviat, P. J. et al., The SIMSCRIPT II Programming Language, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1968.

9. Mastor, A. A., "Experimental Investigation and Comparative Evaluation of Production Line Balancing Techniques," Management Science, 16 (July, 1970), pp. 728-746.

10. Sadler, G. H., System Reliability Engineering, Prentice-Hall, Inc., Englewood Cliffs, N. J., 1963.

11. Shooman, Martin L., Probablistic Reliability: An Engineering Approach, McGraw-Hill Book Company, New York, 1968.

12. Taylor, J. and Jackson, R., Operations Research Quarterly, Vol. 5, 1954, p. 95.

13. Wallis, W. Allen and Roberts, Harry V., Statistics: A New Approach, The Free Press, New York, 1956.

14. Wyman, Forrest Paul, Simulation Modeling: A Guide to Using SIMSCRIPT, John Wiley & Sons, Inc., New York, 1970.

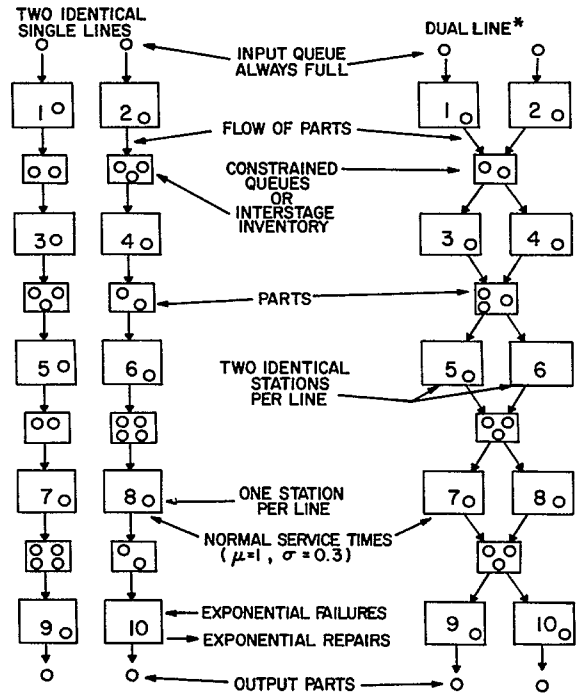

FIGURE I
PHYSICAL MODEL

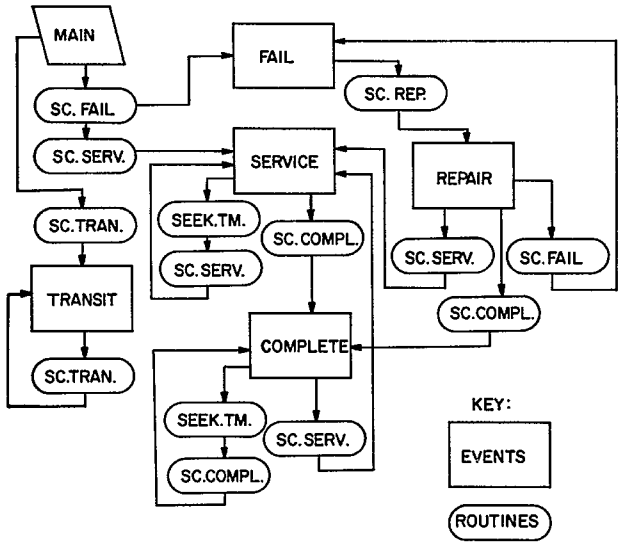

FIGURE 2
SIMULATION FLOW DIAGRAM

Table I

SUMMARY STATISTICS

Run	Variables Altered	Configurations	Sum r_i	Sum r_i^2	Mean r_i	Var r_i
1	MTTNFAR = 50 MTTR = 50	D	14.460	10.220	0.362	0.12802
2	MTTNFAR = 50 MTTR = 50	I	1.960	0.459	0.049	0.00931
3	N = 6	D	77.380	149.894	1.935	0.00519
4	N = 3	I	37.380	36.102	0.934	0.00589
5	Q = 8	D	77.960	152.084	1.949	0.00359
6	Q = 8	I	37.920	36.200	0.948	0.00646
7	None	D	75.220	141.700	1.880	0.00638
8	None	I	37.860	35.999	0.946	0.00422
9	MTTNFAR = 50	D	60.380	92.341	1.509	0.03070
10	MTTNFAR = 50	I	28.040	20.676	0.701	0.02615
11	MTTR = 50	D	65.540	112.265	1.638	0.12507
12	MTTR = 50	I	23.600	20.322	0.590	0.16431
13	None	DE	77.14	148.871	1.928	0.00273

Table II

CONFIGURATION COMPARISON

Test	Configurations	Altered Variables	Difference, $d = r_{i_2} - r_{i_1}$	S_d	Standard Normal Deviate
1	D and 2 x I (1 - 2)	MTTNFAR = 50 MTTR = 50	0.2635	0.06427	4.0997
2	D and 2 x I (3 - 4)	N = 6	0.0405	0.02681	1.5108
3	D and 2 x I (5 - 6)	Q = 8	0.0530	0.02712	1.9542
4	D and 2 x I (7 - 8)	None	-0.0125	0.02411	-0.5184
5	D and 2 x I (9 - 10)	MTTNFAR = 50	0.1075	0.05816	1.8483
6	D and 2 x I (11 - 12)	MTTR = 50	0.4585	0.13985	3.2786
7	DE and D (13 - 7)	None	0.0480	0.01510	3.1798
8	DE and 2 x I (13 - 8)	None	0.0355	0.02214	1.6035

```
// EXEC SIM2CLG,PARM.ASM='NE,B,NL,NED,NX'
//SOURCE.INPUT DD *
PREAMBLE
NORMALLY, MODE IS REAL AND DIMENSION IS 0
PERMANENT ENTITIES
  EVERY QUE HAS A CAPACITY.OF, A SIZE.OF, AND OWNS A GROUP
    DEFINE CAPACITY.OF AND SIZE.OF AS REAL VARIABLES
  EVERY STATION HAS A CYCLE.MODE, A SIDE, A TOC, AN ABORT, A FAIL.MODE, A HOLD,
      A FAIL.RATE, A REP.RATE, AND BELONGS TO A GROUP
    DEFINE SIDE, ABORT, FAIL.MODE, AND HOLD AS INTEGER VARIABLES
    DEFINE FAIL.RATE, CYCLE.MODE, TOC, AND REP.RATE AS REAL VARIABLES
EVENT NOTICES INCLUDE TRANSIT
  EVERY FAIL HAS A F.QUE AND A F.STATION
    DEFINE F.QUE AND F.STATION AS INTEGER VARIABLES
  EVERY REPAIR HAS A R.QUE AND A R.STATION
    DEFINE R.QUE AND R.STATION AS INTEGER VARIABLES
  EVERY SERVICE HAS A S.QUE AND A S.STATION
    DEFINE S.QUE AND S.STATION AS INTEGER VARIABLES
  EVERY COMPLETE HAS A CO.QUE AND A CO.STATION
    DEFINE CO.QUE AND CO.STATION AS INTEGER VARIABLES
PRIORITY ORDER IS TRANSIT, FAIL, REPAIR, SERVICE, AND COMPLETE
DEFINE DIRECTION, PLACE, OTHER.SIDE, K, AND AD AS INTEGER VARIABLES
DEFINE OUTPUT.RATE, PARTS, L.PARTS, L.TIME, SUM.RATES, SUM.SQR.RATE, B, C, F, R,
  NEXT.TIME, T.SERVICE, T.FAIL, T.COMPLETE, T.REPAIR, AND T.TRANSIT AS
  REAL VARIABLES
DEFINE GROUP AS A FIFO SET WITHOUT P, M, AND N ATTRIBUTES AND
  WITHOUT FF, FB, FA, AND R ROUTINES
END



 MAIN
 READ N.QUE
 CREATE EACH QUE
 FOR EACH QUE, DO
   READ SIZE.OF(QUE), CAPACITY.OF(QUE)
 LOOP
 READ N.STATION
 CREATE EACH STATION
 FOR EACH STATION, DO
   LET ABORT(STATION) = 0
   LET CYCLE.MODE(STATION) = 1
   LET FAIL.MODE(STATION) = 0
   LET HOLD(STATION) = 0
   LET TOC(STATION) = 0
   READ FAIL.RATE(STATION), REP.RATE(STATION)
 LOOP
 LET K = 1
 FOR EACH QUE, DO
   FOR STATION = 2*QUE-1 TO 2*QUE, DO
     LET SIDE(STATION) = 1*K
     LET K = -1*K
     FILE THE STATION IN THE GROUP(QUE)
     LET F = FAIL.RATE(STATION)
     LET T.FAIL = EXPONENTIAL.F(F,1)
     CALL SC.FAIL
     LET T.SERVICE = 0
     CALL SC.SERVICE
   LOOP
 LOOP
 LET T.TRANSIT = 50
 CALL SC.TRANSIT
 LET QUE = 1
 LET STATION = 1
 START SIMULATION
 STOP END
```

```
ROUTINE TO SC.TRANSIT
CREATE TRANSIT
LET TIME.A(TRANSIT) = T.TRANSIT
LET EUNIT.A(TRANSIT) = 0
FILE TRANSIT IN EV.S(I.TRANSIT)
RETURN END



EVENT TRANSIT
IF TIME.V GT 2000,
  FOR EACH STATION, DO
    LET ABORT(STATION) = 1
  LOOP
  GO T.OUT
ELSE
LET OUTPUT.RATE = (PARTS-L.PARTS)/(TIME.V-L.TIME)
LET L.PARTS = PARTS
LET L.TIME = TIME.V
LET SUM.RATES = SUM.RATES + OUTPUT.RATE
LET SUM.SQR.RATE = SUM.SQR.RATE + (OUTPUT.RATE)**2
PRINT 1 LINE WITH OUTPUT.RATE, SUM.RATES, SUM.SQR.RATE, TIME.V THUS
OUTPUT.RATE***.*** SUM.RATES*****.*** SUM.SQR.RATE**********.*** TIME.V****.*
LET T.TRANSIT = T.TRANSIT + 50
CALL SC.TRANSIT
IF TIME.V NE 1000, GO T.OUT
ELSE
PRINT 1 LINE WITH PARTS THUS
PARTS ********.*
FOR EACH STATION, DO
  LET CYCLE.MODE(STATION) = 1
  IF SIDE(STATION) = 1,
    LET ABORT(STATION) = 1
    PRINT 1 LINE WITH STATION THUS
    ABORT STATION **
    LET TOC(STATION) = 2051
  ELSE
LOOP
LET SUM.RATES = 0
LET SUM.SQR.RATE = 0
'T.OUT' RETURN END



 ROUTINE TO SC.FAIL
 CREATE FAIL
 LET F.QUE(FAIL) = QUE
 LET F.STATION(FAIL) = STATION
 LET TIME.A(FAIL) = T.FAIL
 LET EUNIT.A(FAIL) = 0
 FILE FAIL IN EV.S(I.FAIL)
 RETURN END



 EVENT FAIL(F.QUE, F.STATION)
 LET QUE = F.QUE
 LET STATION = F.STATION
 IF ABORT(STATION) = 1, GO F.OUT
 ELSE
 PRINT 1 LINE WITH STATION,QUE,TIME.V THUS
 S **  Q **  TIME FAILED ****.***
 LET FAIL.MODE(STATION) = 1
 LET OTHER.SIDE = SIDE(STATION) + STATION
 LET CYCLE.MODE(OTHER.SIDE) = 1
 LET R = REP.RATE(STATION)
 IF HOLD(STATION) = 1,
   LET TOC(STATION) =
   TOC(STATION) + EXPONENTIAL.F(R,1)
   GO SCHED
 ELSE
   LET TOC(STATION) =
     TIME.V + EXPONENTIAL.F(R,1)
 'SCHED' LET T.REPAIR = TOC(STATION)
         CALL SC.REPAIR
 'F.OUT' RETURN END
```

```
ROUTINE TO SC.REPAIR
CREATE REPAIR
LET R.QUE(REPAIR) = QUE
LET R.STATION(REPAIR) = STATION
LET TIME.A(REPAIR) = T.REPAIR
LET EUNIT.A(REPAIR) = 0
FILE REPAIR IN EV.S(I.REPAIR)
RETURN END


EVENT REPAIR(R.QUE, R.STATION)
LET QUE = R.QUE
LET STATION = R.STATION
IF ABORT(STATION) = 1, GO R.OUT
ELSE
PRINT 1 LINE WITH STATION,QUE,TIME.V THUS
S **  Q **  TIME REPAIRED ****.***
LET FAIL.MODE(STATION) = 0
LET OTHER.SIDE = SIDE(STATION) + STATION
LET CYCLE.MODE(OTHER.SIDE) = 1
LET F = FAIL.RATE(STATION)
LET T.FAIL = EXPONENTIAL.F(F,1) + TIME.V
CALL SC.FAIL
IF HOLD(STATION) = 1,
  LET T.COMPLETE = TIME.V
  CALL SC.COMPLETE
  GO R.OUT
ELSE
LET T.SERVICE = TIME.V
CALL SC.SERVICE
'R.OUT' RETURN END


ROUTINE TO SC.SERVICE
CREATE SERVICE
LET S.QUE(SERVICE) = QUE
LET S.STATION(SERVICE) = STATION
LET TIME.A(SERVICE) = T.SERVICE
LET EUNIT.A(SERVICE) = 0
FILE SERVICE IN EV.S(I.SERVICE)
RETURN END


EVENT SERVICE(S.QUE, S.STATION)
LET QUE = S.QUE
LET STATION = S.STATION
IF ABORT(STATION) = 1, GO S.OUT
ELSE
IF FAIL.MODE(STATION) = 1, GO S.OUT
ELSE
IF QUE = 1, LET SIZE.OF(QUE) = 1
ELSE
IF SIZE.OF(QUE) = 0
  LET DIRECTION = -1
  CALL SEEK.TIME
  LET TOC(STATION) = NEXT.TIME
  LET T.SERVICE = TOC(STATION)
  CALL SC.SERVICE
  GO S.OUT
ELSE
LET SIZE.OF(QUE) = SIZE.OF(QUE) - 1
LET HOLD(STATION) = 1
LET C = EXPONENTIAL.F(1.0,1)
LET T.COMPLETE =
        C/CYCLE.MODE(STATION) + TIME.V
LET TOC(STATION) = T.COMPLETE
CALL SC.COMPLETE
'S.OUT' RETURN END
```