A CASED GOODS CONVEYOR SIMULATOR

Donald A. Heimburger
Math & Computing Section
THE PROCTER & GAMBLE COMPANY
Cincinnati, Ohio

Abstract

The SIMSCRIPT Conveyor Program (SIMCON) models the flow of cased goods through an accumulating powered belt conveyor system. The program is almost totally general with all configuration descriptions and parameters being read into the model as input data. The model has been used to test proposed changes to existing systems, to design totally new systems, to study the effects of several variables, and to model new concepts in cased goods merging. The quantitative results have been used in the decision process for many multi-million dollar conveyor networks.

SUMMARY

At the Procter & Gamble Company, powered accumulating belt conveyors are used in manufacturing plants to transport high-volume, low-cost consumer cased goods. The SIMSCRIPT Conveyor Simulator (SIMCON) is an operational tool used to provide quantitative assistance in the decision making process concerning the design of these conveyors. Although an earlier conveyor simulator written in GPSS was available, SIMCON was developed because of the cost, accuracy, and model concept advantages made possible by using SIMSCRIPT. The model has been highly successful and has given quantitative direction to decisions

previously "engineered" by committee. In the first three months of use SIMCON minimized the possibility of design error that could have cost upward of $500,000 (roughly 100 times its development cost) either in over-design or post startup corrections. Since that time, the program was used to justify the use of an existing system where an addition would have cost roughly $1.5 million. SIMCON can be run in either the batch mode or via conversational terminals with prompting. It is also operational on TSO without prompting.

INTRODUCTION

Multiple packing lines for cased goods are often located in a single room. This room is typically a long distance (200' - 2000') from the cased goods warehouse. Rather than a separate conveyor from each packing line to the warehouse, the cases are usually merged onto a single conveyor that carries all the cases to the warehouse. Experience has shown that even the best thought-out plans for merging cases together have occasionally resulted in massive jams and costly re-design. A method was needed to predict, before the fact, what size build-ups could be expected for particular configurations running under many sets of packing line feed rates and conveyor belt speeds.

OBJECTIVE

The objective of the study was to devise a means to provide information on queue lengths and gate utilization for cased goods conveyor networks. The system had to be easy to access and use with little or no modeling or computing knowledge. The tool was planned to be used by both designers and engineers for new systems and evaluating proposed changes to existing networks.

Because of the diversity of all the users, the program was supposed to be available via conversational terminals as well as batch job submission methods. The conversational version was to be extremely user-oriented and self-explanatory. Because the program was intended to be run frequently, cost was also a major consideration.

PHYSICAL SYSTEM DESCRIPTION

An accumulating powered belt conveyor system for cased goods consists of two basic elements. The first element is the packing line, and the second is the merging unit. When two packing lines feed a single merging unit, the result is a single stream of cases. Several additional lines may feed in via more merging units to form a large network.

The Packing Line

Each packing line ejects cases of a known length. The length can vary from line to line, so queues are measured in feet rather than cases. The line averages a given number of cases per minute over the course of a day, and the ejection pattern can be measured.

The Merging Unit

The merging unit is usually called a gate or a traffic cop. The name is adopted because the unit regulates case flow as a traffic cop would regulate automobile flow through an intersection. The operation of the gate can best be described by the series of Figures 1 - 4.

In Figure 1 the case A1 has just arrived on the A side of the traffic cop. The spring-loaded arm swings open as the case passes and mechanically locks the other arm from opening. In Figure 2 a case has arrived at the arm on the B side, but cannot pass until the A arm shuts. Since no more cases arrive on the A side before the spring-action return is completed, the arm swings shut and locks (Figure 3). In the meantime, another case B2 arrived at the queue on the B side. As soon as the A arm locks the B arm is free to open, and the cases B1 and B2 pass through (Figure 4). The cycle continues, each side clearing its entire queue and then releasing the other side.

Two types of major problems can occur at the gate. The first happens when one side holds an arm open indefinitely. An extremely large queue results on the opposite side. The other problem occurs when the belt speeds are too slow and the queues continue to grow indefinitely.

The Total System

The total system is made up of groups of lines and gates. Since each gate merges two lines into a single stream, there is always one fewer gate than lines. A fairly typical cased goods conveying system is shown in Figure 5. This network consists of 12 packing lines and 11 merging gates.

Every configuration consists of the same basic units (i.e., gates and lines). The only distinguishing features from one system to the next are (1) the feed rates, (2) the case lengths, (3) the arm closing times, (4) the belt speeds, and (5) the sources and exits of the gates.

When designing a new accumulating powered belt conveyor network, the first three (3) items mentioned above (feed rated, case lengths, and arm closing times) are usually known. Item 5, the actual configuration of gates and lines, is frequently limited to no more than four choices. Figures 6 and 7 show two ways of

merging six lines into a single conveyor stream.

Once the alternate arrangements of gates and lines are determined, several conveyor belt speeds can be examined to see how fast the belts must travel to avoid frequent and long jams.

METHOD OF SOLUTION

Thought was given to several different approaches to the problem. One method of analyzing a particular configuration would be to build a pilot facility. Howeve., this method is hardly practical for large networks, very costly, and does not meet all of the desired objectives.

Another approach to the problem is the use of analytical techniques. Knowing the various belt speeds and gate operations, the length of the queue at the gate could be calculated. This method was in fact useful in helping to understand the workings of a gate, but is completely deterministic ignoring the consideration of randomness of case arrival and the case-length mix downstream.

The method of analysis that overcame the objectives to the pilot facility and the analytical approach but still gave the desired results was computer simulation. Once the physical description was used to create a model, ideally any number of "what if" questions could be asked to see the effects of each of the operating parameters. Alternate conveyor configurations with the same case-merging requirement (Figures 6 and 7) could also be tested. Simulation was selected as the means to provide the needed answer in a short time at a low cost.

SIMULATION LANGUAGE

The conveyor model was originally written in GPSS, since it was the only simulation language available in-house at Procter & Gamble.[2] However, several distinct drawbacks soon became apparent.

The biggest problem is that GPSS is an interpretive language, and as such, cannot be compiled. Every model was a different program, even though series of MACRO's were used to describe lines and gates. This became a cost problem and also made interface with conversational facilities difficult.

The conversational program, written in RUSH*, which is also interpretive, would (1) create a file consisting of GPSS statements, (2) schedule a GPSS job using the file as input and another file as output, and (3) selectively read results from the output file. Because different size configurations would result in a different

*RUSH - a Remote Use of Shared Hardware is a trademark of Allen-Babcock Computing, Inc.

number of MARCO's, the output could be fairly small or very large.

Explaining the GPSS model to the user was difficult because the source code was very hard to follow for the laymen. Good reports had to be generated using the RUSH program because GPSS did not provide flexible report writing facilities. To be able to provide the level of accuracy necessary, the execution time became quite long and costs soared to upwards of $60-$100/simulation.

Shortly after the GPSS model was completed, SIMSCRIPT II.5Ⓣ became available in-house. Because of the problems with the GPSS model, a new model in SIMSCRIPT seemed attractive.

Some of the reasons for changing to SIMSCRIPT were as follows:

1. The physical system was easily modeled using the concept of entities, attributes, sets and events.
2. The SIMSCRIPT model, once compiled, could be used indefinitely directly from the stored machine code, hence saving from 40-80% of the cost of running each time. Only the data was used to build each separate model.
3. The core requirements were much less than with GPSS, and the total cost was reduced by 80-90%.
4. The user could easily understand the source code. Figure 8 is an example of one of the seven events used in the model.
5. The conversational interface was easy, since the data was just entered directly into a file, and the report that was desired was printed into an output file.
6. SIMSCRIPT made it possible to build into the model several tracing features that are only used when problems occur, or to trace out the step-by-step simulation to verify correct operation. To the user the trace is transparent, and no suppression cards are necessary.

Ⓣ trademark and service mark of Consolidated Analysis Centers Inc.

THE SIMULATION MODEL

SIMSCRIPT views the world in terms of entities ("objects" in the system), attributes (describers of the "objects"), sets (groups the "objects" may be part of) and events (points in time that mark the starting and stopping of activities or changes in status).[1] The SIMSCRIPT conveyor model (SIMCON) was easy to conceptualize in these terms.

Permanent Entities

Permanent entities are objects in the model that usually exist throughout the entire simulation, such as machines or loading docks. In this model there are two types of permanent entities. The quantity of each type is specified when the model is run.

The first permanent entity is the LINE. It has attributes of feed rates, case lengths, and a set (queue) in which to place the cases. The second permanent entity is the GATE. Since the GATE receives cases from two sides, the designation of A and B is used. Each GATE has attributes of a source of cases, a conveyor speed and an arm closing for each side. It also has a designation of the queue to send the case on to.

These two types of permanent entities specify and create each specific SIMCON model, linking the proper lines and gates together in the correct sequence.

Temporary Entities

Temporary entities are objects in the model that may be created or discarded at will. In SIMCON, the CASES are the only temporary entities and their attributes are their length and a serial number for tracing.

Sets

In order to accumulate statistics, the cases are entered and removed from various sets. Each GATE has three sets associated with it. These are an A.QUEUE, a B.QUEUE, and a JUNCTION. Depending on which side the CASE approaches the GATE, it is placed in the A.QUEUE or the B.QUEUE. When a CASE is removed from a queue and is passing by the arm pivot, it is placed in the JUNCTION. The CASE is then sent on to the correct queue for the next GATE, or in the case of the last GATE, it is destroyed.

Events

There are only seven events in the model. The relationship of the events is shown in Figure 9. Each event has a single argument specifying either the LINE number (in the EJECT.CASE) or the GATE number (all the other events).

The EJECT.CASE event creates a temporary entity, assigns it the correct length and serial number, and places it in the queue of the specified GATE. It then schedules an OPEN.GATE for that GATE on either the A or B side, depending on which side the CASE is placed.

The OPEN.GATE(A or B) checks for three conditions. If any condition fails, nothing transpires. First the GATE must not be busy (no CASE in the JUNCTION); second the opposite arm must not be open (or this side would still be locked); third, the queue must have at least one CASE in it to open the GATE. If all conditions are met, the CASE is removed from the queue and placed in the JUNCTION. A PASS.GATE (A or B) is then scheduled in the time it takes the CASE to move its trailing edge to the arm pivot point.

The PASS.GATE takes the CASE from the JUNCTION and sends it on to the next queue scheduling an OPEN.GATE there. If there are no more GATES, it simply destroys the CASE. If there are more CASES in this GATE'S A.QUEUE (if the last CASE came from A.QUEUE), another OPEN.GATE is scheduled immediately, and the cycle continues until the queue is empty. When the queue becomes empty, the SHUT.ARM (A or B) is scheduled in the number of seconds specified by the input data. Historically, this runs from 1-2 seconds.

The SHUT.ARM checks only to see if another CASE arrived during the swing time. If a CASE has arrived, nothing is changed, since the next CASE will also schedule a SHUT.ARM. If no CASE arrived, the OPEN.GATE is scheduled for the opposite side.

The model is allowed to run for a time specified by the user. It then generates a report, part of which is shown in Figures 10 and 11.

Assumptions

Two basic assumptions are made for the model. The first is that case acceleration is almost instantaneous. Since most cases pass back-to-back in a continuous slug, acceleration effects are almost negligible. The second assumption is that cases turning corners pass through at the same rate as their driving belt. Actual timed studies have shown that single cases turn the corners somewhat slower than if they pass straight through, but the difference varies from gate to gate. Again, if the cases are part of a slug, this is not very important.

VALIDATION

In order to test the analytic techniques employed by the model, a scaled-down physical system was constructed out of paper. Cases were moved through the system by half second intervals. The results were compared to the printout produced by the computer simulation (using the built-in trace routines). One minor change pertaining to conversion of time units had to be incorporated to produce identical results.

SIMCON was then validated on an existing network, and predicted maximum queue build-ups varied from actual build-ups by an average of less then one foot. Over 100 simulations have been made and several systems physically constructed as a result. None of these systems have produced significant deviation from the model's predictions.

It was found that when starting the simulation, downstream gates had lower utilization until the cases were flowing throughout the entire system. For this reason the model allows cases to flow for a user-specified warm-up period, clears all statistics, and continues from that point.

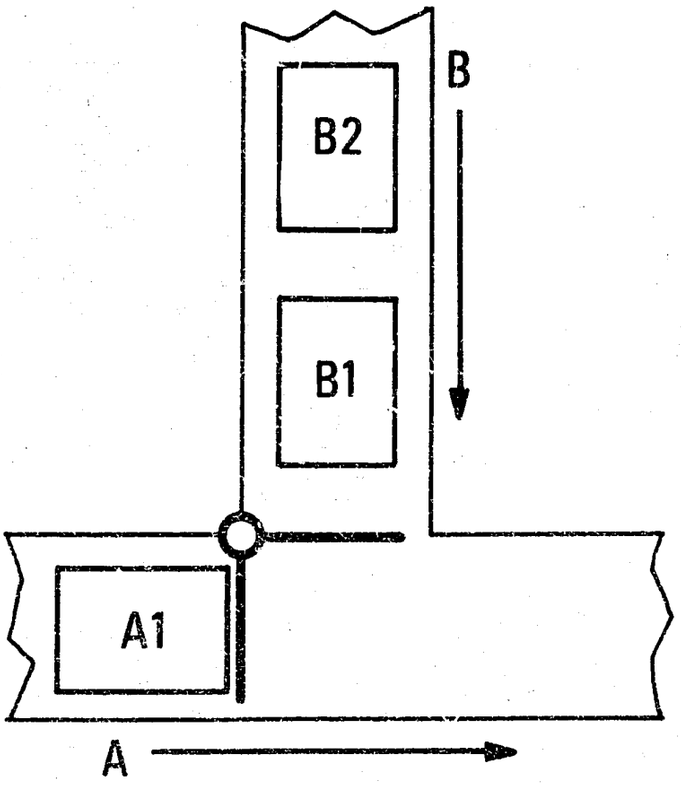
DISCUSSION

Upper management felt from the very beginning of this project that close coordination and communication were very important between the analytical team constructing the model and the intended users (Finished Products Handling Group). Both written and verbal contact were maintained through every step of development, testing and implementation; no unilateral decisions were made without consultation between both groups. As a result, all persons concerned with the project felt joint responsibility and accomplishment for the planned goals and solution. Acceptance has been 100%, and SIMCON is used as standard practice on every significant conveyor project.

The design and implementation of the conveyor simulator was forecast to require 1.5 man-months of analytical effort and roughly $1500 for computer charges. The total development cost was within that estimate. As previously stated, within three months the program minimized the possibility of design error that could have cost upwards of $500,000 in either over-design or post start-up corrections.

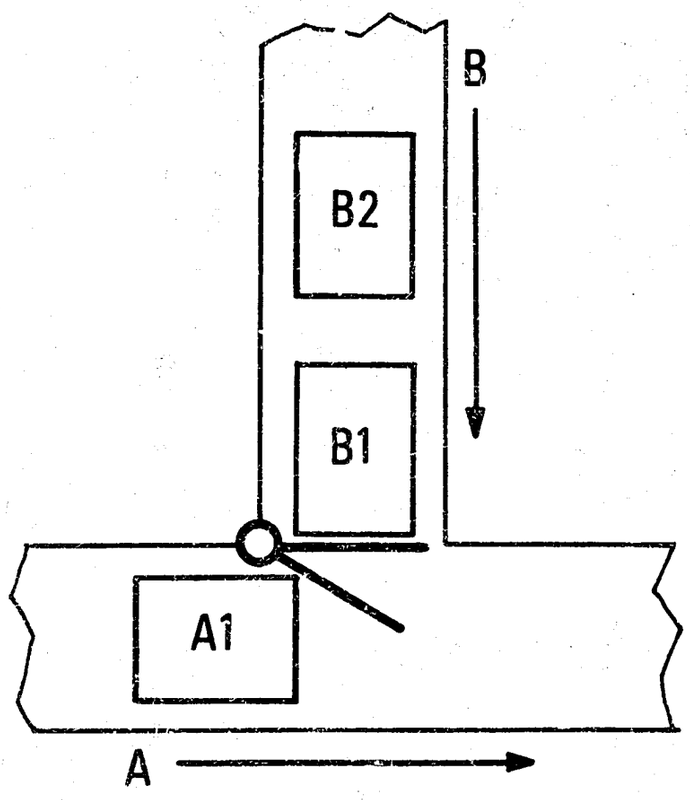
SIMCON has been in use for less than one year, but already has been the primary quantitative tool for making decisions on several powered conveyor configurations costing upwards of $500,000 with the largest being roughly $3,500,000. The program is often run in the batch mode to evaluate many alternatives in a short time. It is also run from many different points in the country via conversational terminals to test effects of changes to existing networks.

Recompilation of the program has not been necessary since it was installed and released for use.



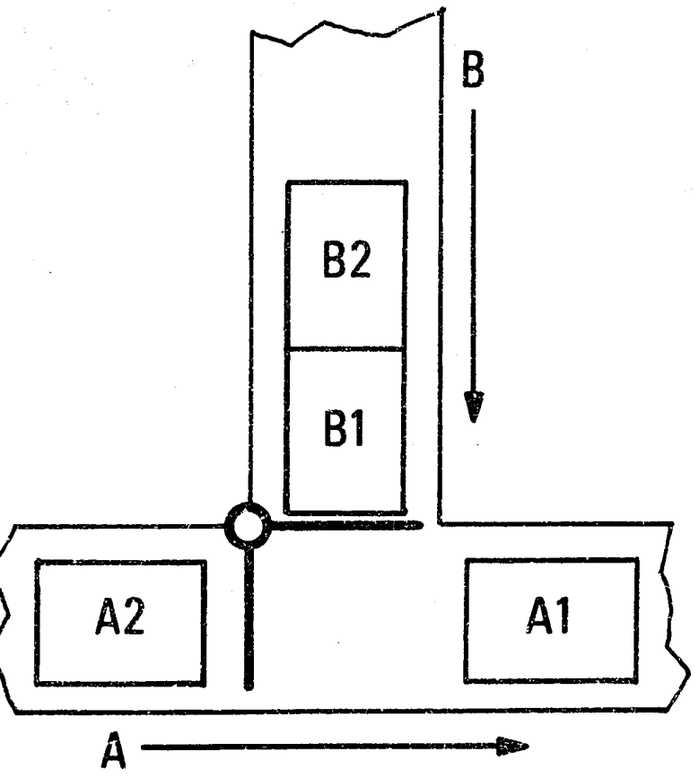
CASE A1 ARRIVES, A.ARM OPENS

Fig. 1



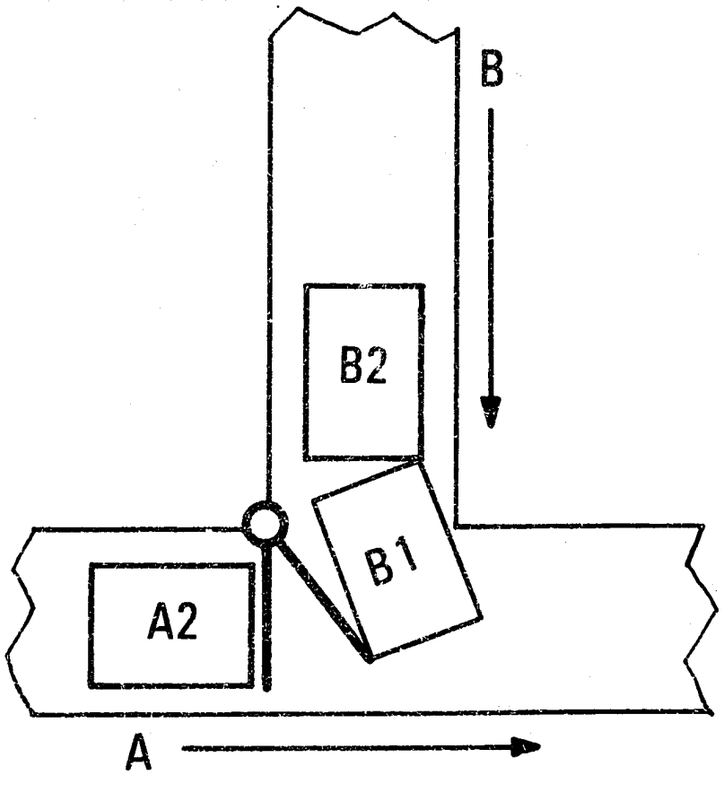
CASE B1 ARRIVES

Fig. 2



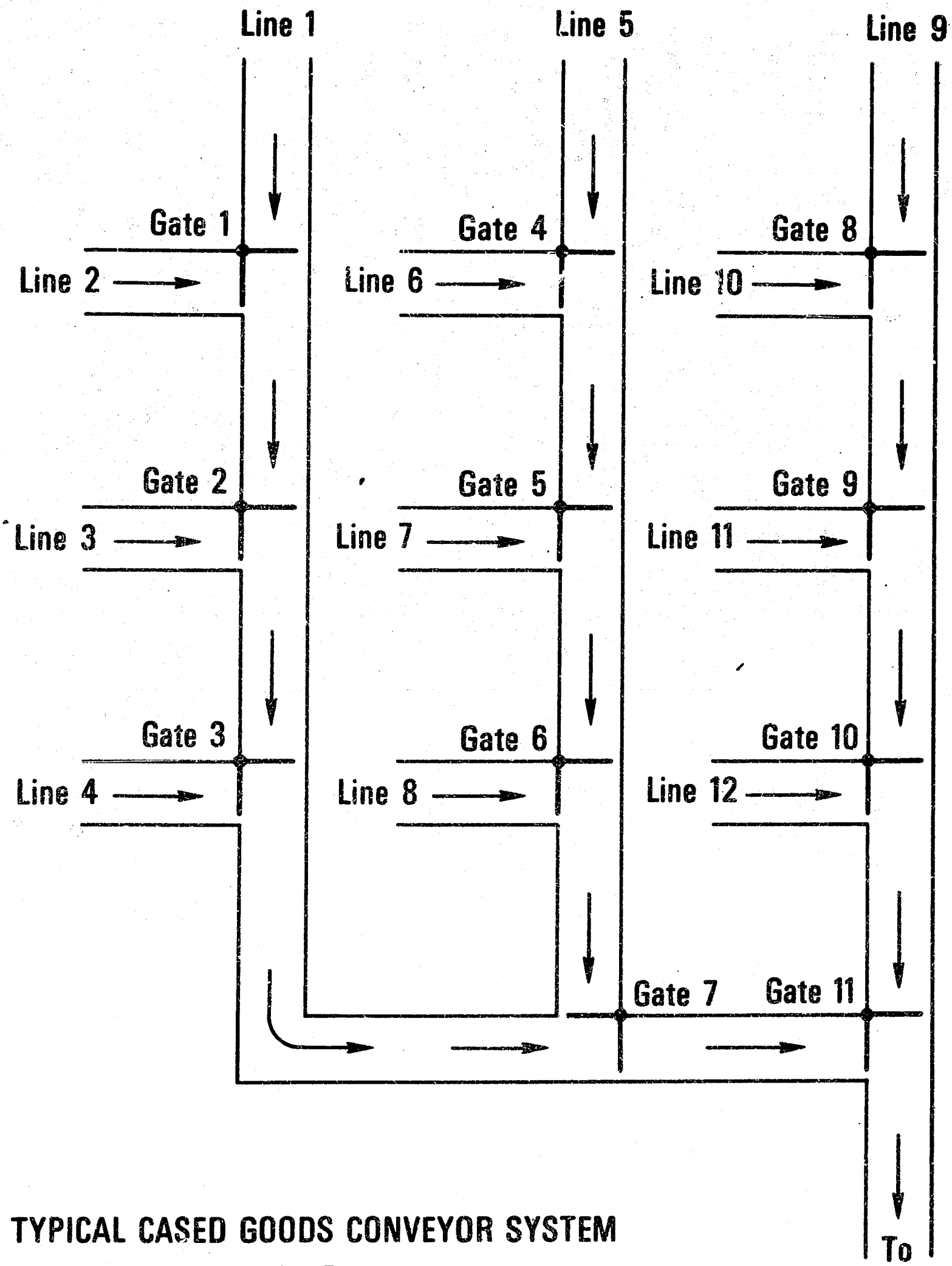
CASE B2 ARRIVES, A.ARM SHUTS

Fig. 3



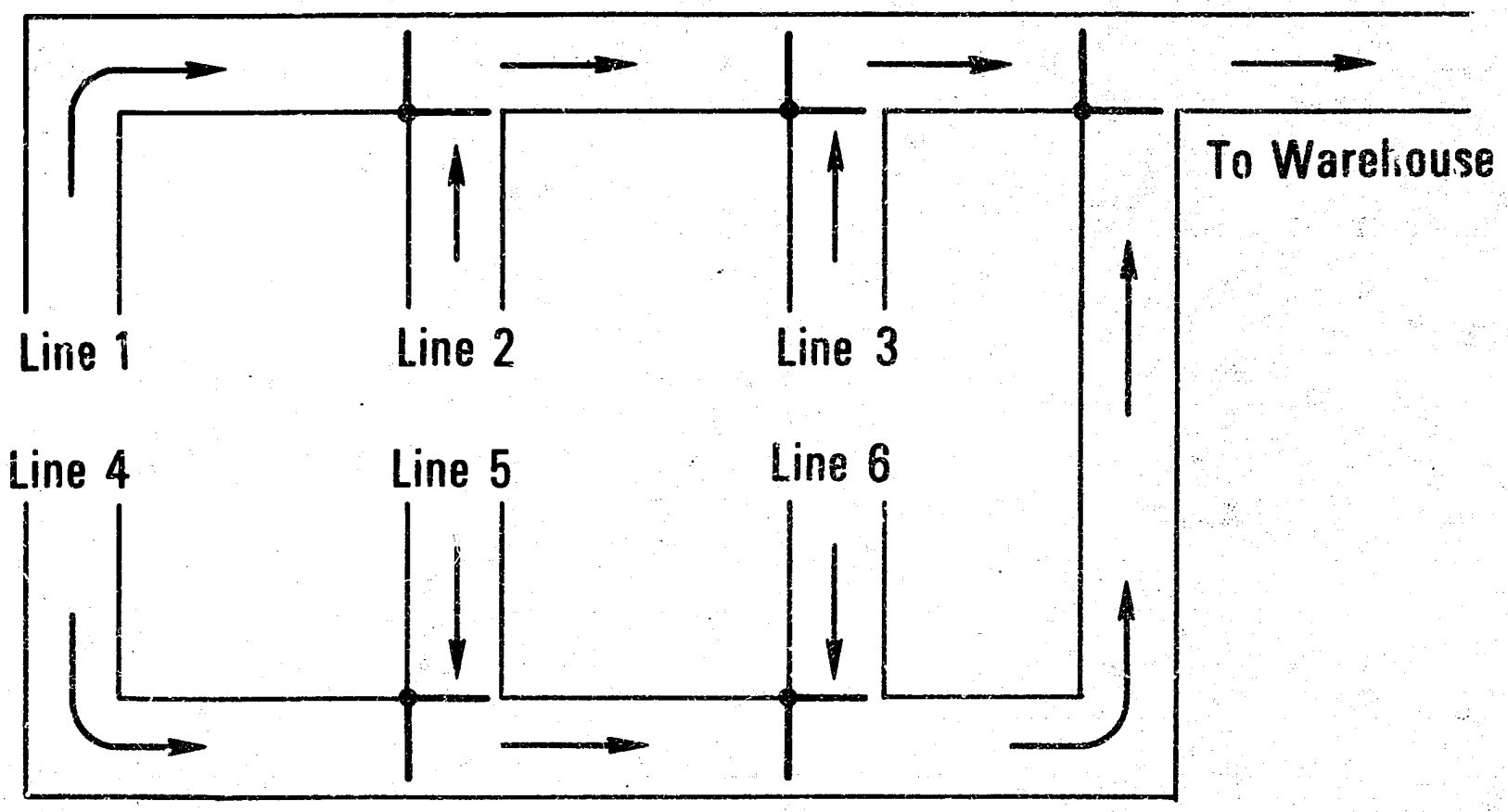
CASE A2 ARRIVES, B.ARM OPENS

Fig. 4



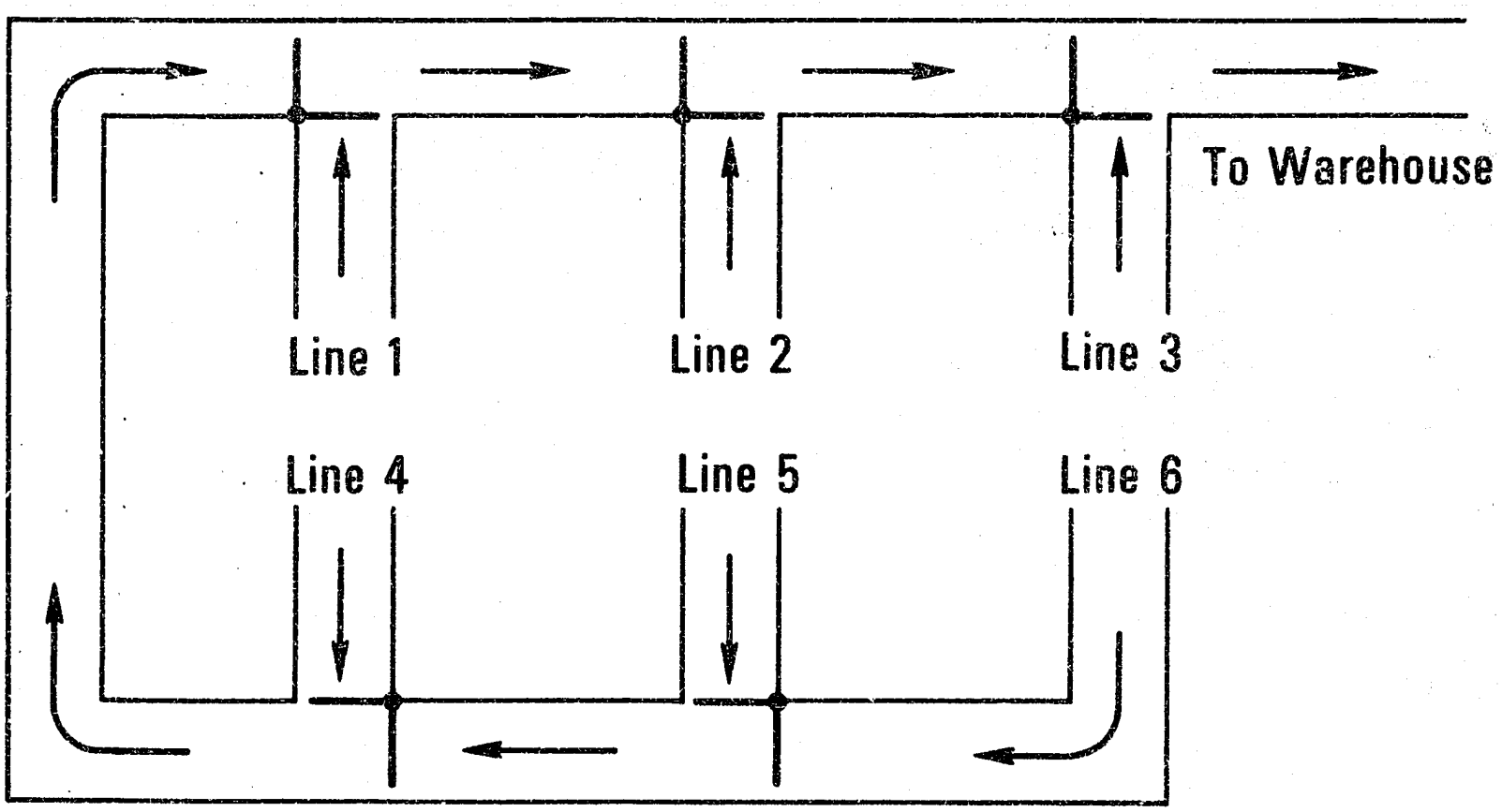
TYPICAL CASED GOODS CONVEYOR SYSTEM

Fig. 5



PARALLEL-TYPE ACCUMULATING CONVEYOR SYSTEM

Fig. 6



SERIES-TYPE ACCUMULATING CONVEYOR SYSTEM

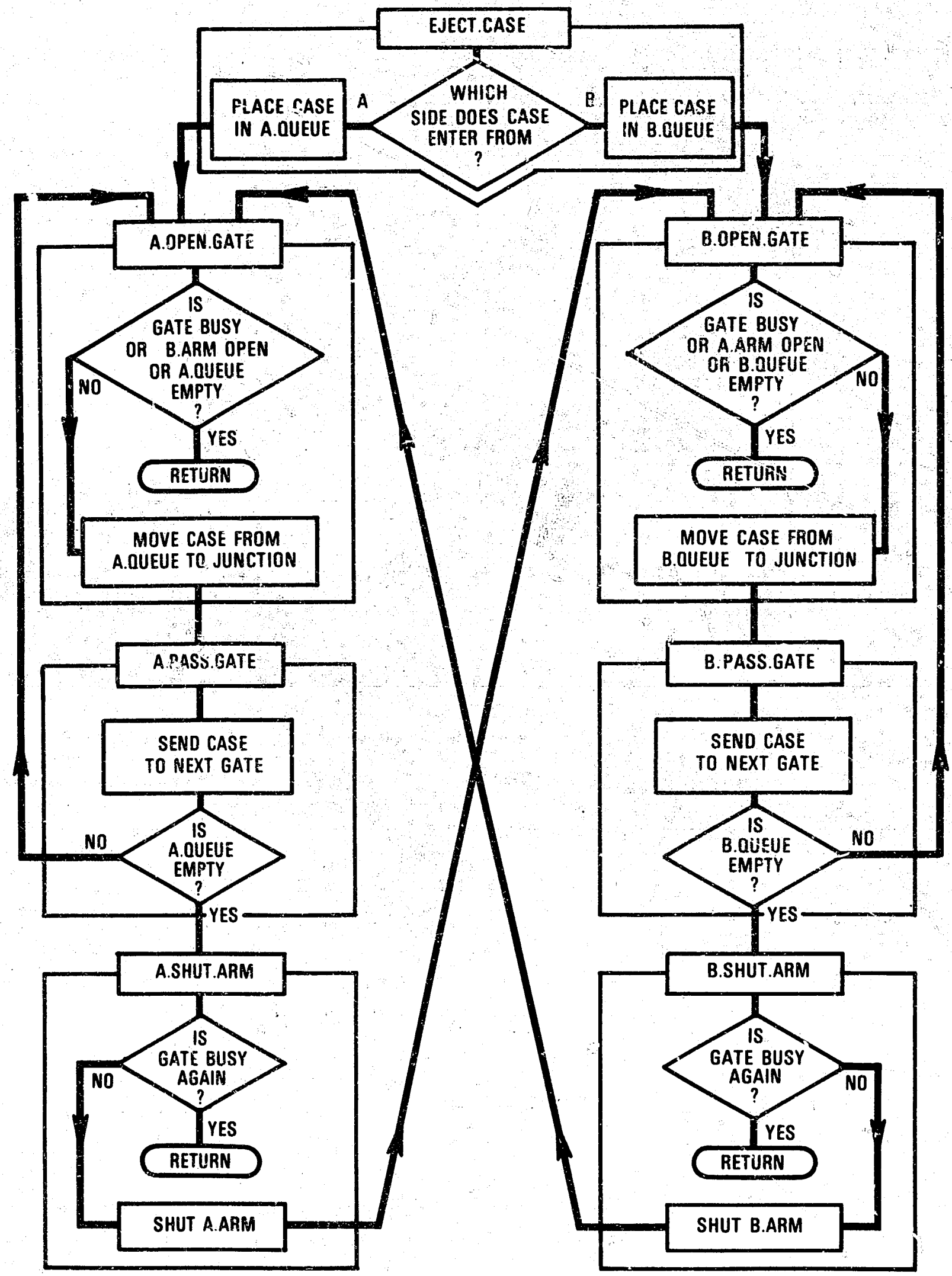
Fig. 7

```
LINE  CACI SIMSCRIPT II.5 REL/VERS 6C                    07/11/72

   1  EVENT A.OPEN.GATE GIVEN A.GATE.NUMBER
   2
   3        ''  THE FOLLOWING "DEFINE TO MEAN" STATEMENTS ARE NORMALLY ''
   4        ''  FOUND IN THE PREAMBLE.  THEY ARE INSERTED IN THIS ROU- ''
   5        ''  TINE SO THAT ANY UNEXPLAINED STATEMENTS OR PHRASES WILL''
   6        ''  BE EASILY UNDERSTOOD.                                  ''
   7        DEFINE TR2 TO MEAN IF TRACE NE 2, JUMP AHEAD   ELSE
   8        DEFINE IS.BUSY TO MEAN NE 0
   9        DEFINE IS.OPEN TO MEAN NE 0
  10        DEFINE PUT TO MEAN FILE
  11        DEFINE SECONDS TO MEAN UNITS
  12        DEFINE BUSY.A TO MEAN 1
  13        DEFINE ONE TO MEAN 1
  14        DEFINE EQUAL TO MEAN =
  15
  16        LET GATE EQUAL A.GATE.NUMBER
  17        LET TEMP.NO EQUAL A.GATE.NUMBER
  18  TR2   '' LEVEL 2 TRACE  ''
  19        PRINT ONE LINE WITH FCT.SECONDS(TIME.V),
  20                            GATE,
  21                            STATUS AND
  22                            B.ARM AS FOLLOWS
  EVENT A.OPEN   **.** SECONDS   GATE **   STATUS = *   BARM = *
  23  HERE  ''  END OF LEVEL 2 TRACE  ''
  24        IF STATUS IS.BUSY
  25             OR B.ARM IS.OPEN
  26                  OR A.QUEUE IS EMPTY, RETURN
  27                       OTHERWISE
  28                       REMOVE THE FIRST CASE FROM THE A.QUEUE
  29                       LET INCHES EQUAL LENGTH(CASE)
  30                       SUBTRACT INCHES FROM A.LENGTH
  31                       PUT THE CASE IN THE JUNCTION
  32                       ADD ONE TO COUNTER
  33                       LET STATUS EQUAL BUSY.A
  34                       SCHEDULE AN A.PASS.GATE GIVEN TEMP.NO IN
  35                            PASS.ARM(A.SPEED,INCHES) SECONDS
  36                            ''  PASS.ARM IS A FUNCTION THAT CALCULATES THE
  37                            ''  TIME NECESSARY FOR THE CASE TO PASS A GIVEN
  38                            ''  POINT KNOWING THE SPEED AND LENGTH OF CASE.
  39  RETURN  ''AND''  END
```

SIMSCRIPT EVENT SOURCE CODE

Fig. 8



EVENT RELATIONSHIPS

Fig. 9

LENGTH OF SIMULATION
1 HRS, 0 MINS
SAMPLE RUN FOR SIMCON WITH 6 LINES AND 5 GATES

SIMCON CONVEYOR SIMULATOR

LINE INPUT INFORMATION:

LINE NUMBER	RATE CASES/MIN	CASE LENGTH INCHES	FEEDS GATE	SIDE OF GATE
1	10.00	28.00	1	A
2	4.00	28.00	1	B
3	2.50	28.00	2	B
4	1.00	28.00	3	B
5	4.00	28.00	4	B
6	6.00	24.00	5	B

GATE INPUT INFORMATION:

GATE NUMBER	FEEDS GATE	SIDE A SOURCE	SIDE A SPEED PAST ARM	SIDE A ARM CLOSING TIME	SIDE B SOURCE	SIDE B SPEED PAST ARM	SIDE B ARM CLOSING TIME
1	2	L1	100 FT/MIN	2.00 SEC	L2	125 FT/MIN	2.00 SEC
2	3	G1	100 FT/MIN	2.00 SEC	L3	125 FT/MIN	2.00 SEC
3	4	G2	100 FT/MIN	2.00 SEC	L4	125 FT/MIN	2.00 SEC
4	5	G3	110 FT/MIN	2.00 SEC	L5	120 FT/MIN	2.00 SEC
5	99	G4	110 FT/MIN	2.00 SEC	L6	120 FT/MIN	2.00 SEC

SIMULATION RESULTS:

GATE NUMBER	MAXIMUM QUEUE LENGTH (IN FEET) SIDE A	MAXIMUM QUEUE LENGTH (IN FEET) SIDE B	AVERAGE QUEUE LENGTH (IN FEET) SIDE A	AVERAGE QUEUE LENGTH (IN FEET) SIDE B
1	2.33	2.33	.17	.16
2	2.33	2.33	.18	.26
3	2.33	2.33	.08	.31
4	4.67	2.33	.55	.45
5	9.33	14.00	.81	2.32

GATE NUMBER	PERCENT UTILIZATION	NUMBER OF CASES PROCESSED	PRESENT QUEUE LENGTH SIDE A	PRESENT QUEUE LENGTH SIDE B
1	77.27	840	0. FT	0. FT
2	87.95	992	0. FT	0. FT
3	91.34	1053	0. FT	0. FT
4	97.11	1293	0. FT	0. FT
5	99.99	1653	0. FT	4.0 FT

LENGTH OF SIMULATION • SIMCON REPORT

Fig. 10

FREQUENCY TABLES

		UPPER LIMIT, IN FEET																			OVER-
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	FLOW
GATE 1	SIDE A:	0	0	602	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	SIDE B:	0	0	237	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GATE 2	SIDE A:	0	0	840	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	SIDE B:	0	0	152	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GATE 3	SIDE A:	0	0	993	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	SIDE B:	0	0	60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GATE 4	SIDE A:	0	0	986	0	66	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	SIDE B:	0	0	241	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GATE 5	SIDE A:	0	0	1112	0	140	0	39	0	0	2	0	0	0	0	0	0	0	0	0	0	0
	SIDE B:	0	196	0	97	0	44	0	19	0	3	0	1	0	1	0	0	0	0	0	0	0

FREQUENCY TABLE • SIMCON REPORT

Fig. 11

REFERENCES

1. Kiviat, P. J., R. Villanueva and H. M. Markowitz, The SIMSCRIPT II Programming Language, Prentice-Hall, Inc. Englewood Cliffs, N.J. 1968

2. Stuebing, R. C., "Monte Carlo Simulation of Full Case Conveying Systems", Digest of the Second Conference on Applications of Simulation 1968