A GASP IV SIMULATION OF AN AUTOMATED WAREHOUSE

Garth L. Jarvis, Ph.D.
Pattern Analysis & Recognition Corporation
On the Mall, Rome, New York

Miles Waugh
Philip Morris, USA
Richmond, Virginia

Recently Philip Morris built a new manufacturing center in Richmond, Virginia. In an effort to utilize the latest in available technology, much of the manufacturing center is dominated by the use of computers. One aspect of this computerization is demonstrated by the Finished Goods Warehouse where the final product is received from manufacturing, sorted by brand, stored in a high density stacker storage area, and eventually shipped to local customers or other distribution warehouses; all of which is controlled by a hierarchy of five minicomputers.

Although the warehouse is only a few years old, sales and subsequently production have increased toward the upper bound of the design specifications. Sales projections over the next several years indicate that the capacity of the warehouse must be increased beyond its present capacity. Among the steps taken to determine necessary system changes to accommodate this production increase has been the development of a simulation model of the warehouse.

This paper will describe the simulation of a portion of the warehouse. Figure 1 illustrates the general case flow through the "Case Input System." The cases are first received from the manufacturing floor via a single conveyor with case brands arriving intermixed at varying production rates. They pass a "Tier Select" laser scanner which reads a binary bar code to determine on which of two tiers each product is being accumulated. The cases then travel on two long conveyors to the warehouse where they pass one of the "Input Scanners." Here each case is identified by its binary code and the lane in which it is to be accumulated is determined. The case is then tracked along the "Input Belt" by the photocells until it arrives at its assigned lane. The computer then diverts the case into the lane where it travels toward the output end to join the queue of other cases of the same brand.

When a sufficient number of cases have been accumulated in a lane (a full pallet load plus five extra cases), the lane is set ready to meter out a load for automatic palletizing. If the "Output Belt" is available, the cases are conveyed to the "Verification Scanner" where each case is verified for the correct binary code. If a wrong code or an unreadable code is detected, a substitute case is ordered. If at the end of the last case of the load the number of substitutes does not exceed five, they are metered out of the lane and past the scanner. After the entire load is past the scanner, the next lane in the priority chain that is ready is allowed to meter out. The "Output Belt" is set available if no other lanes are ready, allowing the first lane ready to meter out a load to do so.

As the case train reaches the "Contingency Diverter," provision is made to merge both tiers to a single palletizer in the event of a palletizer failure. All cases that were not properly verified at the Output Scanner are diverted to a manual palletizing area. The verified cases are then fed to the palletizer where they are automatically palletized in a brand-dependent pattern. Considerable logic and a number of conveyor belts between the contingency diverter and the palletizers insure space between cases in several areas where case count is important, and insure the merging at case train boundaries in the contingency mode.

The following objectives provided the impetus for the simulation:

- To determine the maximum realizable system capacity.
- To determine where system "bottlenecks" occur and what can be done to eliminate them.
- To determine the effect of different production schedules on system capacity.
- To determine what system modifications will be required for projected increased production.
- To determine the best way to implement system modifications with respect to optimum throughput.

After investigating both the GPSS V and the GASP IV simulation languages, it was decided to use GASP IV. The main reasons for this decision

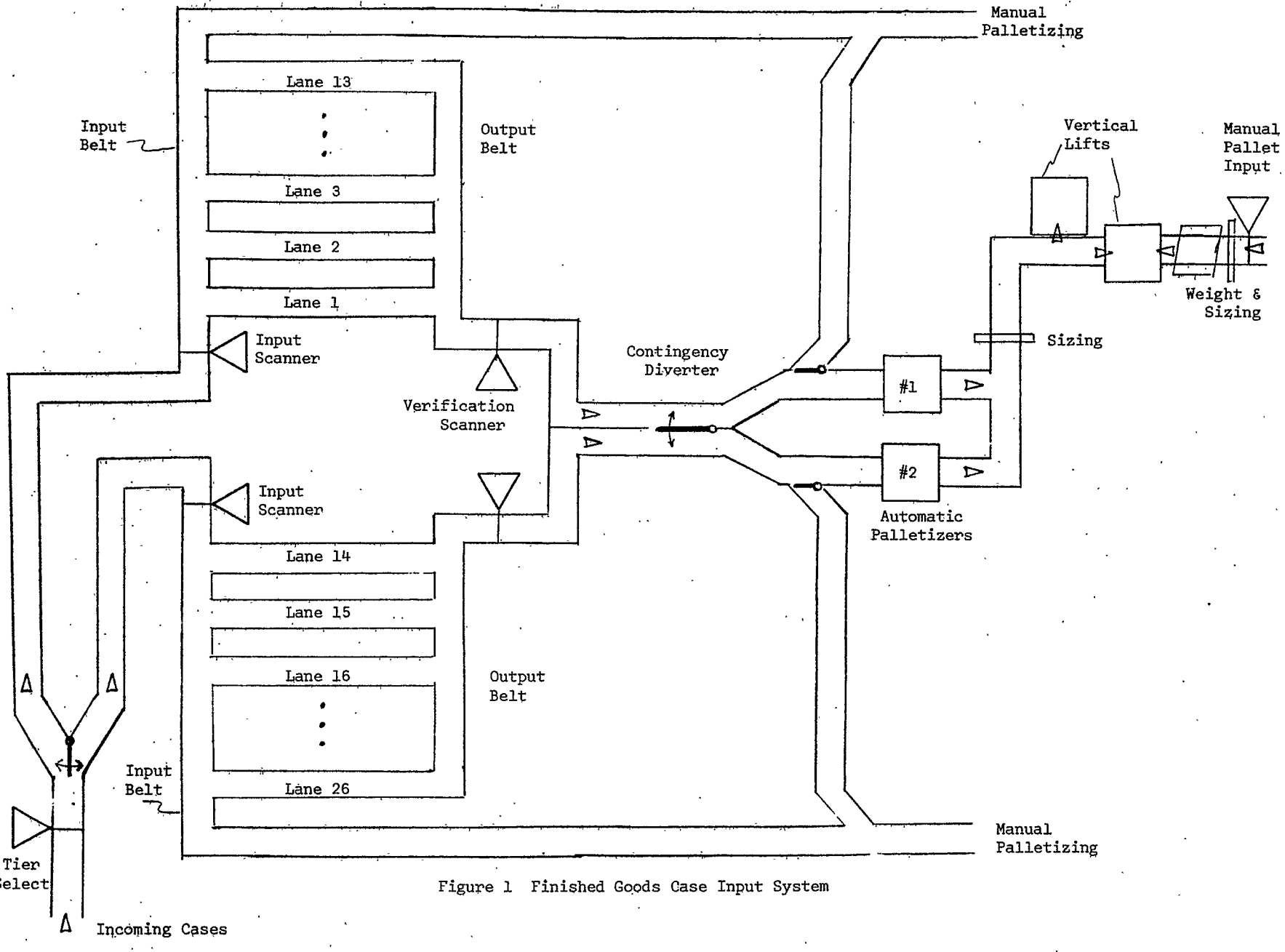

Figure 1 Finished Goods Case Input System

are listed below:

- GASP is written in FORTRAN allowing modifications and program maintenance without special software training of personnel.
- GASP is generally faster than GPSS in execution speeds.
- GASP is modular in structure allowing modifications to one area without the necessity of modifying the entire program.
- GASP allows a combined discrete and continuous simulation, facilitating the simulation of belt conveyors which frequently stop and start under normal operation.

A description of the entire simulation model would be too lengthy and repetitive to discuss here. Therefore, this paper will focus on the techniques used for simulating different types of conveyors and on modifications made to the GASP subroutine. A small subsection of the model will then be presented as an example. Since this is an application of the commercially available GASP IV software [1], reader familiarity with it will be assumed.

Conveyor Simulation

There are two generic types of conveyors which were simulated. This first type will be referred to as roller conveyors. These generally run continuously but allow cases to stack up or queue at the output end of the conveyor as they have a relatively low coefficient of friction with the cases. Maintaining distances between cases on this type of conveyor is unimportant. The second type of conveyor will be referred to as belt conveyors. This type of conveyor has a high coefficient of friction and thus does not allow cases to slip with respect to conveyor speed. They are used when maintaining intercase distance is important. When the lead case on this type of conveyor reaches a point beyond which it cannot continue, the entire belt is turned off, stopping all cases on it. Cases upstream from such a conveyor must either queue or be stopped in a similar manner.

Consider first the roller type of conveyor. Because these conveyors are not frequently turned on and off, the unimpeded travel time of a case traveling a known distance may be calculated when the case moves onto the conveyor. Thus a time event scheduled at a Δt, equal to the travel time, can be used to simulate this type of conveyor. When cases queue at the end of such a conveyor, a simulated case does not actually reach the end of the conveyor when its time event occurs; rather, it butts up against the last case already in the queue. Therefore, time events are used to file the cases in First-In-First-Out (FIFO) files. Cases are removed from these files by routines controlling the next downstream conveyor.

Belt conveyors, on the other hand, are frequently turned off and on due to downstream conditions; thus the travel time cannot be calculated when the case starts its travel. These conveyors are simulated using state variables. Each state variable is used to represent a belt position rather than a case position due to the number of cases on the belt at any one time. The derivatives of the state variables are set to the belt speed when running and to zero when the respective belt is off. The position of the belt is set to zero at the start of the simulation and the maximum position the belt is allowed to reach is 10,000 feet. When it reaches this value, it is reset to zero and all related distances used to determine case position are reset equivalently.

A distance array with two circular pointers is used in conjunction with each state variable for determining when cases reach a destination of interest. One of the pointers is used to indicate the oldest, or leading, case on the conveyor, the other to indicate the next available position in the array for the next case moving onto the belt. The value stored in the array when a case enters a belt is set equal to the current position of the belt plus the distance to the end of the belt, or to some other position of interest along the belt, minus one-half of the tolerance. For example, consider a belt (B1) a distance D1 long. A case entering the belt would cause the following:

```
DISTB1(INEW(IB1)) = SS(IB1) + D1 - TOL/2

INEW(IB1) = INEW(IB1) + 1

IF (INEW(IB1) .GT. MAXDIM(IB1)) INEW(IB1)=1
```

where IB1 is the state variable number for belt B1, SS(IB1) is then the current value of the state variable, and MAXDIM(IB1) is the dimension of the array DISTB1 representing the distance for the case to travel before its state event occurs. The case is then filed in a FIFO file used for holding case attribute information while the case is on the belt. The subroutine SCOND, called at the advance of each simulation clock interval in which state events occur, is checked to determine if the position of the belt has passed the value in the distance array corresponding to the oldest case. Thus, only the leading case on each such conveyor is checked to determine if it has reached its destination.

Subroutine GASP

A basic flow chart of the modified GASP subroutine is shown in Figure 2. The general flow is the same as with the published version of GASP [1]. However, because the conveyors represented by state variables had constant speeds, the state variable integration was replaced with the following linear equation for each state variable, I.

```
SS(I) = SSL(I) + DD(I) * DTFUL
```

where DTFUL is the integration step size, SSL(I)

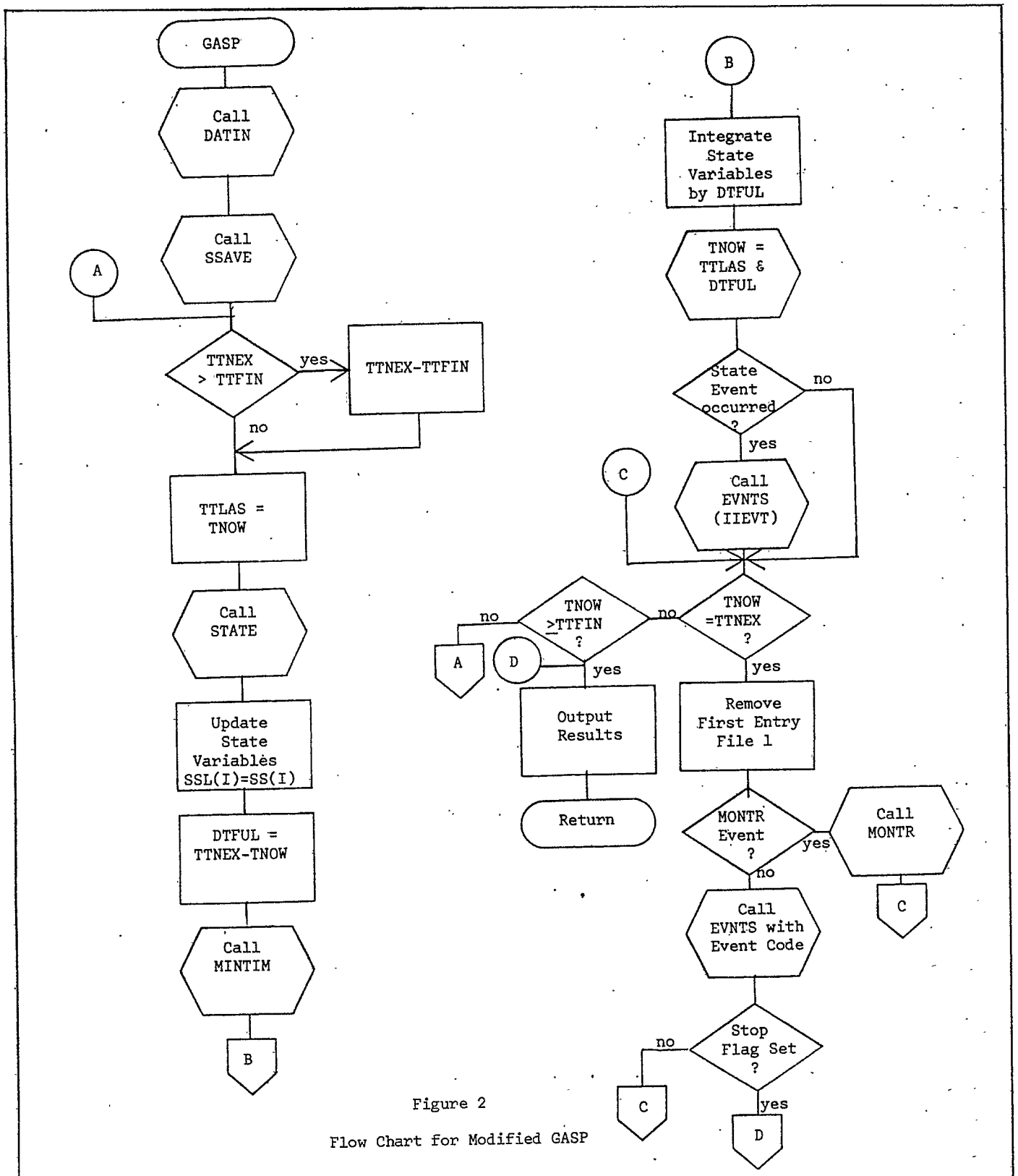

Figure 2

Flow Chart for Modified GASP

is the current state variable value, DD(I) the derivative, and SS(I) the value of state variable I at the end of the time increment.

This procedure greatly reduced the time required for integration from that required by the more general purpose Runge-Kutta-England algorithm. Furthermore, because the equations were linear, the step size could be solved for in close form, thus allowing the step size to be set at the maximum value not exceeding the tolerance of the next state event or the time of the next time event. This was achieved by setting the step size (DTFUL) to the time increment of the next time event (TTNEX-TNOW) and calling the subroutine MINTIM. This routine simply found the minimum time of all scheduled state events by solving for the time required for the leading case on each state variable conveyor to reach its destination. For example,

TIME(IB1) =

{DISTB1(IOLD(IB1))-SS(IB1)+TOL}/DD(IB1)

solves for the time for the leading case on belt B1 to travel the remainder of the distance which was set when the case moved onto the belt. Note that since the distance the case had to travel was decreased by one-half the tolerance, the case will arrive at its destination within ± one-half of the tolerance and thus should statistically average to a mean equal to the correct distance.

Example of Simulation

Since the simulation model consists of 25 user subroutines, 8 of which are state event initiated and 18 time event initiated (one being called from either mode), 49 files, and 10 state variables, only the small subsection shown in Figure 3 will be discussed here.

Figure 4 illustrates the flow chart for the simulation logic initiated by a case reaching the end of one of the two full load belts (Point A, Figure 3). Assuming that both palletizers are operational and therefore not in contingency mode, a check is made to determine if the belt after the contingency diverter is stopped. If it is, the full load belt is also stopped. Any cases already past the contingency diverter effectively queue at Point B on the appropriate tier. A flag in user common is set to indicate to the routine which turns the belt after the contingency diverter on, that the full load belt must also be turned on.

If the Post-Contingency belt is on, the event of the case arriving at Point B is scheduled. Since this area runs continuously, either a time event or a state event could be used. A time event can simply be scheduled at a future time increment equal to the travel time from Point A to Point B on the appropriate tier. A state event, which was actually used, is set for at the distance from A to B minus one-half of the 6-inch tolerance.

At the end of the time required for an unimpeded case to travel from Point A to Point B, the routine ECTGY, whose flow chart is shown in Figure 5 is called. If there are no cases queued at Point B, the case is scheduled to move a case length onto the Post-Contingency belt. If cases are queued at this point, the case simply stays in the First-In-First-Out file representing the queue. Because the PCB conveyor frequently stops and starts, it is represented as a state variable, the derivative of which is set equal to the belt speed; zero if off.

When the PCB state variable has increased a distance such that the case is completely on the belt, allowing for some slipping initially, the routine ONPCB is called (Figure 6). The distance the belt is required to travel in order for the case to reach the end of this belt is used to

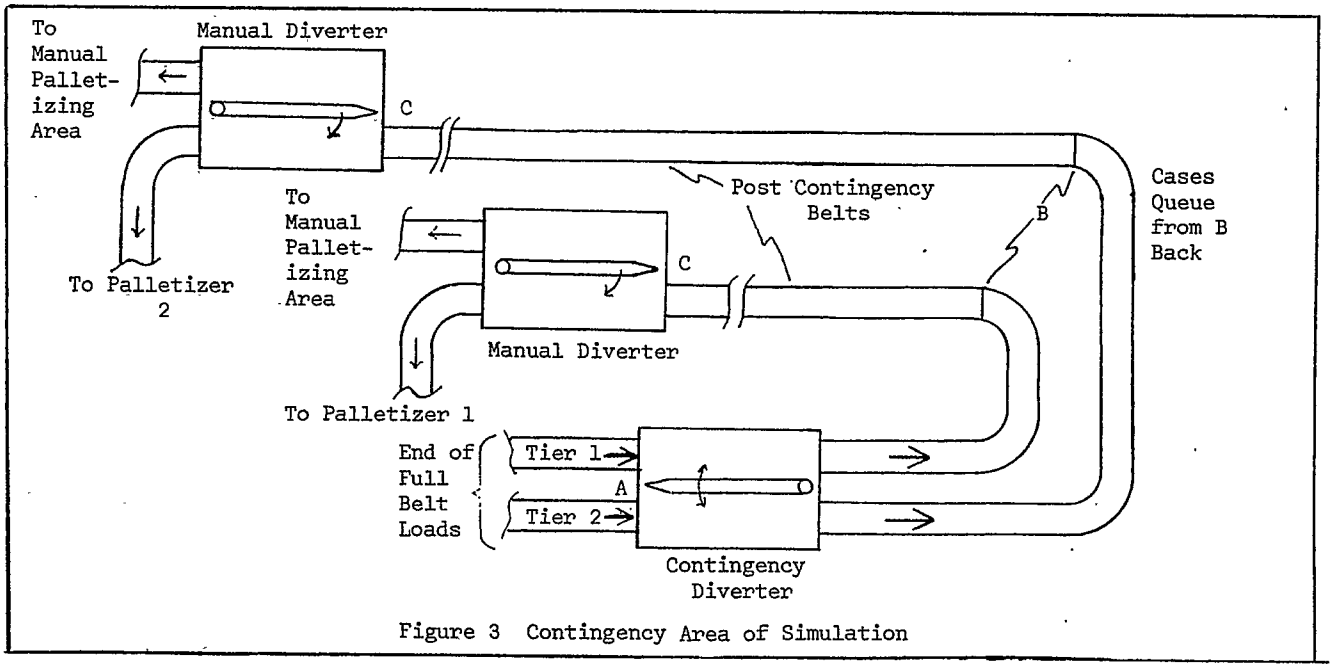

Figure 3 Contingency Area of Simulation

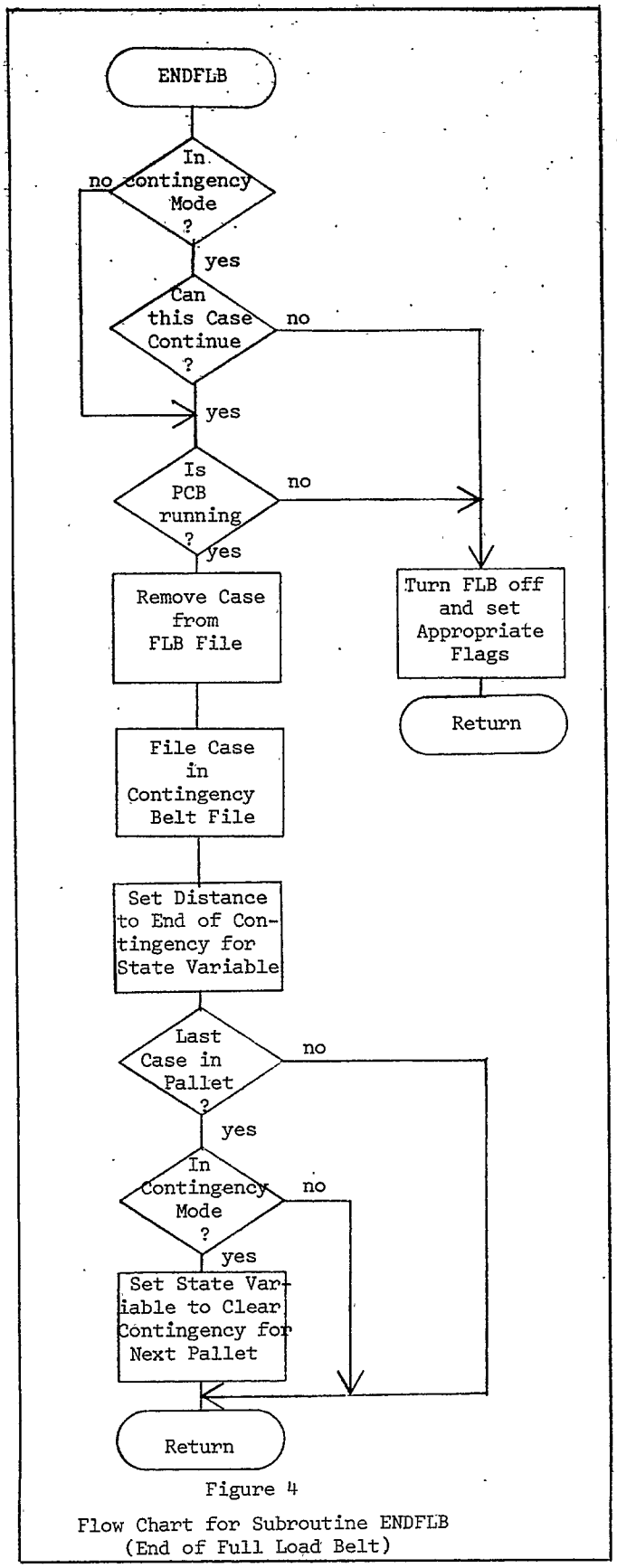

Figure 4

Flow Chart for Subroutine ENDFLB
(End of Full Load Belt)

schedule the state event of the case arriving at Point C of the appropriate tier. If any cases remain the queue at Point B, the next case is scheduled to move a case length onto the PCB conveyor. When a case arrives at Point C, a time event is scheduled for the case to arrive at the next point downstream.

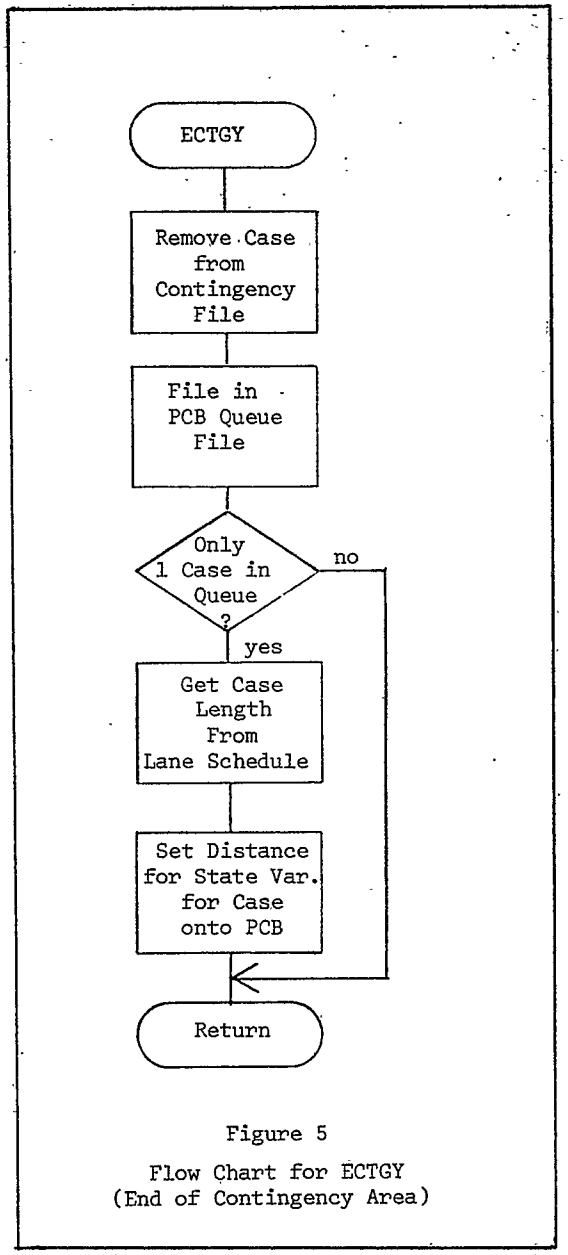

Figure 5

Flow Chart for ECTGY
(End of Contingency Area)

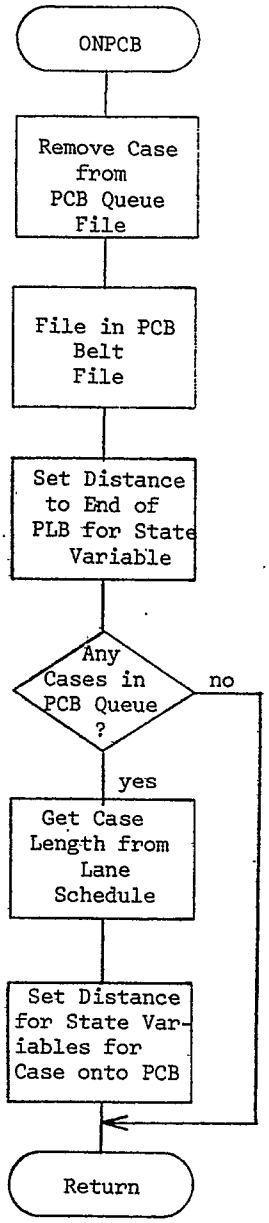

Figure 6

Flow Chart for ONPCB

(Case on Post-Contingency Belt)

Conclusions

The results of the simulation indicated several minor changes that could be made to the system to eliminate some existing difficulties. Some of these were subsequently made and were verified by actual operation. For example, the simulation indicated that the maximum utilization obtainable from the palletizers due to conveyor operation was approximately 84% each when both were operational. Additionally, when contingency mode was necessary due to a breakdown of one palletizer, the maximum utilization dropped to 78% when Tier 1 palletizer was down and 63% when Tier 2 palletizer was down. This decrease was due to necessary spacing between pallet loads to insure that cases merging to one tier at the contingency diverter, did so at pallet load boundaries. By adding two "zone clear" photocells in the area of the contingency diverter to shorten this introduced gap between pallet loads, the simulation indicated that the palletizer maximum utilization could be increased to 80% when Tier 1 palletizer was down and 81% when Tier 2 palletizer was down. Additionally, by moving another photocell 4 feet and reducing an associated time from 4 to 2 seconds, the maximum utilization of the palletizers could be increased to over 96% and possibly to approaching 100% when not in contingency mode and approximately 90% when in contingency mode. These changes were subsequently made. Although the increase of maximum utilization had no noticeable effect when not in contingency due to lower than maximum production rates, the improvement in contingency mode operation was well received by supervisors in the warehouse.

The model also indicated that several major modifications will be necessary to accommodate future production, particularly if equipment failures are considered. These results were used to justify the ordering of a third palletizer and additional accumulation lanes. A model is presently being developed to help in the design and layout of the connecting conveyor for interfacing these additions with the existing configuration. Additional expansion of the model will eventually simulate the entire Finished Goods Warehouse. It is hoped that this model will not only provide the necessary information for system expansion in the near future but will be maintained and used continually whenver modifications are necessary, or to help in production planning, so that the impact of different schedules on the warehouse can be measured.

References

[1] Pritsker, A. Alan B., The GASP IV Simulation Language, John Wiley & Sons, Inc., 1974.