SIMULATION IN THE DESIGN OF UNIT CARRIER MATERIALS HANDLING SYSTEMS

W. Wayne Siesennop
Systems-Design Department
University of Wisconsin-Milwaukee
(formerly of Rex Chainbelt, Inc.)

Fritz A. Callies
Technical Center
Rex Chainbelt, Inc.

Neal S. Campbell
Data Systems Division
A. O. Smith Corporation

ABSTRACT

Unit carriers are used in satisfying many current materials handling needs. A system using this kind of car-on-track hardware can be extensive and complex. This paper discusses the values and benefits of simulation as applied to the design of such a track system. The design of the automatic baggage handling system for the Seattle-Tacoma International Airport is used as an example of such a unit carrier system. Design requirements include input rates of baggage loading stations and car destination patterns over a relatively fixed system geometry. A simulation model is used to evaluate overall performance including empty car availability and minimized network flow. Design of the trackage controls as well as tuning of the entire baggage handling system are aided by use of the model. The results show that simulation is a highly-cost-effective tool for this problem.

INTRODUCTION: A unit carrier is defined as a vehicle used to transport one or more items to a certain destination. In warehouses designed for automated order picking, unit carriers are loaded with several items and deliver these to a specific control point. In institutions and post office factories individual carriers or trays are used in a similar manner. Another common example is a passenger elevator. In a unit carrier system these discrete, individual vehicles carry an item or items along fixed routes, each relatively independent of the other vehicles in the system.

In the case of the unit carrier, all handling equipment interfaces with the standardized carrier rather than with the item carried. This is in contrast to conventional materials handling hardware where each item moves along a conveyor system with its position fixed with respect to other objects. The items interface directly with the conveyor belt chain or rollers and with any control or processing equipment of the conveyor system.

Unit carriers are particularly useful when the items are fragile or of nonuniform size or, in general, when they are difficult to handle by conventional conveyor methods. Because positive control as well as ability to remember are features of the unit carrier design, items can be processed at greater rates and with more reliability and safety than items on a conveyor belt.

Accompanying the advantages of unit carriers however, are the problems associated with the dramatic increase in the level of complexity and sophistication required. Unlike a continuous processing system, the typical unit carrier system contains a large number of relatively independent containers traveling to different destinations through a network containing branch points and merge points. The unit carrier system requires a supply of available carriers at the origin, and a place to store the carriers not in use. In order to design such a system, the control logic must allocate carriers to different areas within the network. The result is a complex track network requiring considerably more sophisticated design techniques than those required for conventional conveyor systems.

This paper concerns the use of simulation in the design of a unit carrier baggage handling system for the Seattle-Tacoma International Airport (1). This baggage system presents problems which are common to many unit carrier materials handling system applications.

SEA-TAC BAGGAGE HANDLING SYSTEM - Airport congestion, due to increasing air travel and larger aircraft, is becoming increasingly familiar. Aircraft congestion slows arrivals and departures. Automobile congestion and limited parking facilities are increasingly troublesome. With this added traffic and congestion have come the problems of handling larger quantities of baggage which must be moved quickly and carefully in ever-increasing volumes, while minimizing damage, pilferage, or mis-direction.

Currently, most airport terminals handle baggage by means of well-designed conveyor systems and ramp vehicles. But for a growing number of airports, particularly those of medium and large size, these are no longer adequate. Unit carrier systems provide a highly flexible, high-capacity and high-quality solution to these baggage handling problems. Several companies now produce such carrier systems for this baggage handling market (2) (3).

The new Ground Transport Express (GTX) system at the expanding Seattle-Tacoma International Airport involves over 1,000 four-wheeled carriers (Fig. 1) operating on an extensive network of 20,000 feet of track and serving some 3500 peak hour passengers (Fig. 2). The track network, which has 80 switch and merge points, connects all of the airlines, including two satellite terminals and four parking lot check-in stations, into one unified system. It facilitates check-in and interairline transfers.

A typical GTX carrier trip is represented schematically in Figure 3. A carrier moves to a loading position from an adjoining storage line. It is loaded with luggage and coded for the proper destination. The carrier then moves through the complex track network to the proper dump station. After discharging the baggage, the empty carrier seeks the nearest available storage line.

In the Sea-Tac system, baggage enters the system at passenger check-in counters in the main terminal and in the parking lot. It proceeds by carrier to different luggage sorting areas where the baggage handling system automatically sorts it by airline or, in some cases, by flight number. The baggage is then transported by ramp vehicle to the airplanes. Inter-and intra-airline luggages is also handled by the GTX system, although at Sea-Tac, deplaning luggage is sent by conventional methods to the baggage claim area.

The unit carrier in the GTX system is a 185 lb. car with a V-shaped pocket formed in the body to provide a secure holding place for almost any size and shape of passenger luggage (Fig. 1). Motive power is supplied to the car by means of a friction drive mechanism built into the track. A probe on the carrier allows it to switch from one track to another upon command. The detailed physical characteristics of the hardware of the carrier system for the Sea-Tac Airport are described in (4).

Specialized equipment allows cars on secondary lines to merge into traffic on a primary line. Inclines, declines, and vertical lifts allow for necessary changes in elevation. The carrier is capable of dynamically discharging its baggage onto a stationary slide or moving belt collection system.

Carrier traffic is controlled by means of magnetic sensors located at decision points along the guideway (5). These sensors interrogate a memory, fixed to each car, which carries a binary code indicating the empty or full condition of the car and its destination. This code is entered at the loading point. After discharge, the memory is changed to indicate an empty carrier.

The system, then, must manage a large number of independently programmed carriers operating simultaneously, traveling from a large number of origins to a large number of destinations on a network of track which is constrained both by cost and by the existing physical track configuration environment. The system must operate efficiently under all circumstances and, in fact, its primary justification is its lower per-unit cost at both low and high activity levels.

SYSTEM DESIGN AND ANALYSIS - An acceptable system must be designed to at least meet the customer's specifications. Regarding baggage movement at the Sea-Tac International Airport, two major flow characteristics were specified:

1. Travel Time - this is a measure of the maximum amount of time required for a bag to travel to its destination in the system. This includes the time needed for an empty carrier to travel to the check-in station, to be loaded, and to travel through the system.
2. Peak Network Flow Rate - this rate specifies the number of bags per minute that each originating station may send to each destination under peak conditions.

These two criteria, in turn, imply other constraints on the operation of the system. For example, the ability to handle a certain peak number of bags per minute from a certain facility implies that there must always be an adequate supply of available carriers at that facility. The maximum travel times from point to point imply that the traffic densities at peak conditions must not be so large as to prevent the merging of traffic at critical points in the network. The restriction on traffic densities at certain points implies that unnecessary movement of empty carriers should be minimized to speed the flow of the full carriers through the network.

Any unit carrier system can be designed with enough redundancy to insure its satisfactory performance under any circumstances. The cost of redundancy, however, is high. It may mean extra elevators, fork trucks, or in the GTX system, carriers and track. The problem, then, is to produce a system which will meet a specified set of performance criteria under anticipated operating conditions with a minimum of redundancy, excess capacity, and unnecessary activity.

The design problem involves finding a way to relate the system's measures of effectiveness to the system's design parameters to allow meaningful cost-benefit trade-offs to be examined. For example, a lack of empty carriers available at a specific loading area may be due to the system logic, the total number of carriers in the system, the physical track configuration, or other aspects of the system. The analysis and design problem is to relate the measure of performance, such as the availability of empty carriers in a particular area, to a system design parameter such as the total number of carriers in the system, in order to allow the designer to appreciate the effects of design changes on performance.

The design of a dynamic system typically has two major phases. The first phase is a static or steady-state phase; it essentially asked what the maximum loading conditions are like, on average, and uses this average loading to produce a preliminary design. This phase can use the conventional methods of analysis much like those used in balancing assembly lines. Assuming that the system must be able to operate at steady-state at some maximum rate, the capacities of each part of the system can be determined from these input rates. This is really a mass-flow analysis: cars in equals cars out.

In the past, this static analysis has been sufficient for many systems. Tuning of the system was done by adjusting either the real system after it was built or an actual physical model. This tuning corresponds to the second major phase of systems analysis and design. Using the preliminary design, the second phase investigates system response to dynamic loading conditions. This dynamic analysis deals with the system response to such factors as random elements in the loading, system start-up, system shut-down, or a net cross-flow of carriers in the system. The random nature of arrivals to to the airport may for short periods transform a low average demand rate per hour to a demand well in excess of the design criteria. Rapid start-up may cause temporary shortages of empty cars. Certain random loading conditions may produce an unbalanced cross-flow of carriers in the system. This can result in shortages of empty carriers if the control logic does not respond adequately to compensate for this flow condition. Several of these conditions may occur at once, compounding the problem.

More than the conventional static design tools are required to intelligently design for this kind of system loading. The usual continuous modeling techniques used in many engineering control system design applications do not allow for probabilistic inputs. Queueing models allow for certain kinds of random elements in the system but become unmanageable for large systems. In addition, the queueing theory assumptions often become unrealistic and confining for such problems, and formulation of queueing model segments require data which are not always readily available.

Use of analytic models requires that the problem be divided into small components which can be dealt with and for which clear relationships among design variables can be defined. Analysis relying entirely on these small units may lead to a system in which each component is designed independently. This can result in good component design and poor system design. Optimizing subsystems does not necessarily optimize the total system.

To cope with such large system problems, computer simulation is regarded as the most cost-effective tool. Use of a computer simulation model allows the designer to gain experience with the system as the system design and model evolve. He then has a clearer understanding of system performance relationships while the design process is still going on. In this way the designer's understanding of the significance of different measures of effectiveness can guide the progress of the design by providing fast, efficient feedback as changes in the system are made.

A simulation analysis usually has credibility not only with the designers, but also with management and decision makers. The manager can be shown what the computer is doing at each stage of the process, and the manager can verify that the computer model does represent the process being modeled. The model can be coded in as much detail as is necessary to reflect the real system, without worrying about fitting the constraints of a particular theoretical model. The simulation approach allows examination of the effects of many different and unusual loadings. Control parameters and alternative configurations are readily changed. System design then becomes an iterative process with the model builder searching for the best solution in terms of his evolving understanding of the problem at hand.

SIMULATION PROJECT PLANNING AND MANAGEMENT: With an iterative design process for a large system, project planning and management are critical factors. A lack of careful planning and management can allow project costs to become greater than necessary without significantly improving the results. The tendency to include more detail than necessary can increase data collection, model development, running, and data analysis costs. A model structure which does not take advantage of system modularity can result in higher development, testing, validation, and running costs. Poor choice of a simulation language can result in higher model development costs or running costs. With these problems in mind, the GTX model was carefully planned and budgeted.

For the Sea-Tac System, the possible project conclusions included the following:

1. Confirming that the original system design was satisfactory
2. Revealing that the original design, or portions of it, was unworkable
3. Discovering control logic changes and additions which would improve system performance
4. Finding ways of cutting costs without impairing system performance

Each of these outcomes was valuable, and each was present in some degree. The combined value of these results was the value that was relevant to the development of the project development and design.

Based on a preliminary examination of the project as a whole, a specified task, a specific completion date, and limited resources were assigned to the effort. The project design was thus controlled by both available resource inputs and the required outputs.

The next step in the planning process was to allocate the available resources to produce the desired results. The demands to be placed on the model and the inputs to the model were defined. How the model would be used and what kinds of experiments would be run were important design considerations. The resulting design stressed simplicity of output and modeling, modular design, and testing of modules prior to assembly of the total model. Associated with the modular design was a project plan with milestones and review points. These were important factors in the economic as well as design success of the project.

ACTUAL MODEL - Having analyzed the problem and established the project budget, completion time and environment in which the model was to operate, the final level to consider was the model itself. One of the most important aspects of any model planning is the choice of a simulation language. The language chosen for modeling the baggage handling system was General Purpose Simulation System (GPSS). This choice was made in order to reduce model development time at the expense of somewhat longer running times and larger core requirements. It was anticipated that model development would be the most expensive and timeconsuming portion of the project. With discrete carriers flowing through a network of tracks, this system was a good application for GPSS, which is oriented toward flow type systems.

For this baggage handling system, the system geometry was fixed (Fig. 2) before the simulation was initiated. The total baggage handling system network, consisting of about four miles of guideway, was divided into five subsections. Each section was programmed, debugged, and tested independently and then the segments, which are too highly interactive to give meaningful results independently, were joined into one total system. This modular construction minimized the cost, time, and computer charges required for model programming, testing, and debugging.

The design choices were assumed fixed for such design parameters as number of carriers in the system, line speeds, and storage bank capabilities. The latter was largely influenced by architectural and constructural considerations. Therefore, the model was not organized for easy alteration of these parameters.

Speeds and distances were converted to delay-times for each piece of equipment (lifts, inclines, turntables, lines, etc.) involved in the system. These were grouped into about 140 line segments, each with the appropriate transit time (Fig. 4). Attempts to model randomness in these transit times were a refinement deemed unnecessary and one that would complicate debugging and checking of the simulation output.

With this planning completed, the actual coding began. The Sea-Tac system model, which contained about 1500 GPSS statements, was developed in an elapsed time of two months, and represented about a three man-month effort. This included time required to plan the project, collect data, build and test the modules of the system, and asemble the complete model. Then one month was spent in an iterative process revising both system and model logic to achieve a reasonably satisfactory level of operation. At this point the simulation could be considered a finished tool representing an operable system. An additional five months were then spent using this tool to develop and refine the logic of empty car control, to evaluate the effects of specific control hardware, and to work out solutions to problem areas. Working with the simulation was a great aid to imaginative innovations; however, the project time and budget constraints regulated the degree to which these could be pursued.

The simulation model, once constructed, was used to understand the system by asking "what if" questions. Using different rates of baggage arrivals at different parts of the network, the response of the system was carefully monitored. The number of cars in each section was monitored, as was the rate of flow of traffic at critical points, the availability of empty carriers, and the cross-traffic from one section to the other. The system was examined during system startup in the morning, peak operating levels during the day, and as the system activity decreased later in the evening. Arrival rates were changed at different sections of the system to test the ability of the system to respond to uneven loading in various parts of the network. Each of these experiments served to provide a deeper understanding of the system's performance, and point out critical areas in the system.

Some of the relatively fixed design features were changed as the simulation pointed out problem areas which needed to be corrected. In most cases where changes were originally anticipated, the input was programmed for easy changes. Control logic at each switch was represented by a predefined true-false Boolean variable, a set of logical conditions which, when satisfied, cause the carrier to switch. These Boolean variables evaluated such things as whether the carrier was loaded and, if so, its destination. For an empty carrier, the levels of relative need in various storages would influence its path.

The generators which created GPSS transactions to represent the arrival of baggage were controlled by values which could be initialized or changed at any predetermined time in the course of a simulation run to reflect various system loadings. Destination of the carrier from various input locations was controlled by random number generators which would produce distributions to match predetermined but easily changed functions. Thus the system was represented by a mathematical model including both the internal system logic and the inputs arriving from outside the system.

The results of the simulation are no more accurate than the inputs to the simulation. Therefore data must be gathered or generated with great care. In the case of the Sea-Tac simulation, a complete engineering study was conducted prior to the simulation (7). Baggage input spectrums were derived from that report and from the customer specifications. Figure 5 is an example of the magnitude and distribution of total airport baggage arrivals used for input data in the computer simulated system.

With clear measures of cost effectiveness available, the required level of detail in the system was initially established. More detail was added later in the project, but only after it became clear that the added detail would have a significant effect on the system performance at key points in the network. For example, it was initially decided that all merges would be simulated as working without physical constraints. When areas were detected where rates were exceeding equipment capabilities, merge suppression was modeled. By adding this detail only at key points, however, significant costs were avoided without jeopardizing the project results.

MACRO SUBROUTINE EXAMPLES - Exploiting program modularity through the use of GPSS MACROS, or subroutines, further reduced coding and debugging time and expense. Macros used in the system simulation were designed to standardize and document the modeling of similar physical situations, and provide the basic framework around which the rest of the system model was built.

The proposed system of releasing empty cars from storage banks was modeled in a macro; eventually several different macros were used to represent the different methods of ordering cars. Macros were also programmed for each kind of station, carrier storage queue, and other facilities which occurred at several points in the network, as well as for various initiation routines. These macros were then called at the appropriate point in the program. A discussion of one of these macros follows, to illustrate this capability.

The Load Macro (Fig. 6) describes a portion of the system where bags are loaded into carriers. This simplified diagram, in which each symbol represents one GPSS statment, illustrates the one-to-one relationship between GPSS statements and system logic. The macro models the process of loading empty cars and initiating calls for additional cars to be released for loading if there are both empty cars and more bags waiting.

In the Load macro (Fig. 6) empty cars enter the loader and occupy the loading facility A. The car, represented by an GPSS transaction, then moves into two successive ASSIGN blocks which store the number of the origin B and the number of the destination C in parameters associated with the car. The transaction, the car, then moves into an UNLINK block which removes one bag from the bag queue D and loads it into the car.

Having performed these bookkeeping functions in zero simulated time, the transaction then enters the ADVANCE block where it spends the loading time E (typically 3.7 seconds for a station rate of 17 cars/min.). When the loading time has elapsed, the transaction enters a TEST block and tests whether or not there are empty cars waiting to be loaded and bags waiting in the queue. If this is true, the transaction moves into the second UNLINK block and releases one empty car from the storage bank H. If there are no bags waiting to be loaded, or no empty cars available to be sent out, the transaction goes to the RELEASE block G. It leaves the loading facility A, and the transaction representing the full car leaves the loading area.

<u>OUTPUTS IN RELATION TO GTX SYSTEM DESIGN</u> - The philosophy of adding details only where required was followed in specifying simulation output. Initially the only output was the standard GPSS statistics. Once the model had been debugged and verified, it furnished a broad overview of the overall performance of the system. In addition, it provided the ability to examine in any desired depth of detail the areas of special concern. As key problem areas in the system became apparent, additonal tables were added.

Problem areas common to unit carrier installations have been addressed previously (6) and determine overall system effectiveness. Certain of these problems, including excessive line flow rates, insufficient empty car supply, and overall system imbalance, were examined in the Sea-Tac system design. System imbalance can result from many occurrences, including high activity within one particular area or a disproportionate number of loaded carriers sent from one station in the system to another. Both have the tendency to deplete certain areas of empty cars, resulting in lack of containers in thoses areas and causing high line desnities in other sections. Lack of empty cars at a check-in area is one of the most serious system problems. If empty containers are absent, the airline passenger is delayed and the airplane may be detained. This not only results in a poor customer relationship, but may also mean additional expense for the airline. It was a prime design consideration.

One of the primary indications of the performance of the baggage handling system was the length of the baggage waiting queues. Baggage waiting queues formed at loading stations when the system response was such that an insufficient number of empty cars was available to handle the rate of baggage arrivals. Figure 7 shows the amount of excess baggage that was not removed from certain stations under the baggage input conditions given in Figure 5. However, as the system responded and sent the necessary empty cars to those stations, the baggage waiting queues decreased. Important variables shown are the amounts of baggage in the queues and the time it takes for the queues to be relieved.

For a better understanding of why the system handled or did not handle baggage in various areas, it was necessary to study the availability and movement of the empty carriers. Several types of output contributed to this analysis, the most important being the tabulated conditions of empty carrier storages (Fig. 8). Shown in the figure is the activity of each empty car storage area in the system, including a number designating each storage and the empty car capacity for that storage, current contents of empty cars, and the rate of cars entering the storage at this particular point in time.

In general, a low level in a storage bank at any given time is not always significant. Perhaps no carriers are needed in a certain storage during normal running conditions; this is the case for ST080. Maybe the carriers needed are already enroute. Likewise, a moderately high level could still be dangerously low. In the case of long tunnels, a number of carriers may have entered previously and not yet emerged. Therefore they are not available for immediate use. Or, perhaps an inadequate flow may be presently arriving which may cause a future shortage. The average level over some small period is more meaningful.

A second measure of empty car availability can be used. Other tables exist which show the number of carriers in line at the exit of the storage area. But again, a low level is not necessarily significant. If a carrier always arrives just when required, it will never stay in the storage, so the number of carriers stopped and waiting could be small even with a large number of carriers enroute. The problems inherent in these two methods of counting empty car storage contents resulted in a detailed study of the type of electrical sensors to be installed at various points in the physical system and their effect on system logic.

Other graphical portrayals help analyze the condition of critical storages over time. Figure 9 is one example showing the activity in the north tunnel bank, the main empty carrier reservoir in the north half of the GTX system. The return route from the North Satellite contains a storage bank line for 261 carriers and a high speed line which allows empty carriers to bypass this storage when they are needed immediately elsewhere. The figure shows that more cars than necessary were entering the storage and then immediately leaving, when in fact they should have been bypassing the storage altogether. In later simulation runs different control methods were employed to reduce this flow.

In order to analyze where the carriers were going to and coming from, it was necessary to study the block counts of activity through each logic block of the model. From these a plot of traffic density and carrier location distribution everywhere in the system could be prepared. This was of great value, particularly in analyzing traffic at merges.

Line flow rate is limited by the car velocities and by the processing rates of such equipment as inclines and elevators. If the line density becomes too high at the limiting speed, merges become bottlenecks. At merge points, cars on the secondary track are halted in a queue until an adequate gap occurs in the primary line to allow merging. The higher the flow rate on the primary line, the less opportunity for cars to merge, thereby resulting in a longer waiting line. Each holding position in the track requires some specialized hardware to stop, retain, and then advance the carriers In addition, when the number of positions in the queue is filled, the secondary line must shut down until the queue is relieved. The number of queue positions at a merge point is a matter of economics as well as geometrical restriction. Minimizing line flow rate minimizes merge problems. For example, the traffic density was such that it seemed advisable to add to the model a simulation of the merge at the output of the north tunnel storage bank. The efforts expended in simulating this area were also applicable to developing the actual merge control used in the physical Sea-Tac system.

Studies of traffic densities also pointed out other problems. Carriers were diverted by some preceding station or logic test before reaching their required destination, or they were trapped in a loop. The latter type of problem is illustrated with the loop shown in Figure 4. After the first run with one set of logic, it was noted that there was abnormally heavy traffic in the lines in this loop. A review of the logic for switches SWB02 and SWB05 revealed an inconsistency which was preventing cars from leaving the loop. After this was corrected, traffic was reduced to a reasonable level.

In addition, tables were specified to record the origin and destination of all carriers passing certain points. Fig. 10 gives the origin table ORB11 for line LNB11 and the destination table DEB12 for LNB12. Table ORB11 shows the origins of cars passing on LNB11 during a given time; the column Upper Limit indicates car origin station. Similarly, Table DEB12 gives the destinations of cars passing on LNB12, the column Upper Limit representing the destination stations, and Observed Frequency indicates the number of carriers passing through LNB12 destined for those stations. Destination zero indicates an empty carrier.

A more detailed study of destination and origin tables aided in locating GTX system control logic errors and in redesigning workable control logic. In an initial model run, these tables revealed that very few of the empty carriers required on LNB13 came from LNB12 (Fig. 4). Most empties entered storage bank 70, then left again on LNB23 as required. The result was a large number of carriers in dynamic storage, i.e., on their way into and out of bank 70, rather than bypassing via LNB12. A set of bypass logic was devised to prevent the carriers from going into this loop. Analysis of the effects of the new logic revealed that under certain circumstances it resulted in starving station 10. Consequently, a reserve level had to be established for storage bank 70, below which it could have priority. All these considerations were eventually integrated into a set of final logic which minimized undesirable system responses.

EMPTY CAR MANAGEMENT - Many of these problem areas are related to each other and are really symptoms of the overall empty car flow management problem. The trip of a full, coded carrier is deterministic in the sense that the path of travel is known, and the trip times can be estimated, within certain limits, depending on system traffic levels. However, empty car flow is much more complicated. Once the car has been unloaded, it is coded as empty and sent into the system to seek a home in an empty car storage line. The car may pass through a number of switches before reaching an empty car storage line.

When directing empty cars, the switches can be generalized as either area switches or storage line switches. An area switch allows the carrier to move into a particular area which contains a multitude of storage lines; a storage switch allows the carrier to enter a particular storage line (Fig. 11). When an empty car reaches an area switch, the switch directs it into either area A or area B, depending on the relative number of empty cars in storage in these respective areas and/or the priority of the one area over the other. After a carrier is in an area, it approaches a storage switch (A-1, Fig. 11). If storage A-1 has room and there are no priority needs downstream, the car enters. Otherwise it passes, traveling on to switch A-2.

Another aspect of empty car management enables certain empty car storage lines to call for empties when their level becomes critically low. If the level of empty cars in storage A-2 is low, a signal can be sent to storage A-1 to release a certain number of cars. Upon reaching switch A-2 these cars will satisfy the needs of storage A-2. Another form of signal can also be initiated when storage A-2 is low. The signal will block Switch A-1, preventing any empties from entering storage A-1. It may even block the area switch preventing any empty carriers from being diverted to area B. Therefore all empties will go into A-2 until its needs are met.

Figure 12 is a simple model of only a small part of the overall system. In reality, there are 39 storage lines scattered throughout the Sea-Tac installation with capacities ranging from two carriers to 276. Their placement and capacities are, like the track itself, subject to architectural constraints.

CONCLUSIONS: The Sea-Tac simulation was useful in several ways. It clearly showed that the static preliminary design alone was insufficient for a complex unit-carrier system, and that simulation was a very helpful tool in design of dynamic systems. This was demonstrated not only in an engineering sense, but also in an economic sense--doing the analysis quickly and at a relatively low cost. The modular design approach to development was a central part of this cost effectiveness. A set of macros was developed and then used extensively. In addition, building and testing the entire system in pieces and then assembling these saved many hours and dollars.

With this simulation tool available, it will be comparatively easy in the future to model similar systems with relatively small changes to the design approach.

From an engineering standpoint, the major result was a good final design. On a more specific level, it became clear that the symptoms by which the system's performance would be measured were all part of empty carrier management. This area could be systematically analyzed and workable solutions developed.

In designing the GTX system for empty car management, then, it became evident that:

1. A priority must be established among storage lines
2. The overall capacity of each line must be set within geometric constraints
3. Operating levels and critical levels must be determined for most storage lines
4. Depending on storage lines priorities and their desired performance parameters, the automatic call logic must be designed.

These points constitute the controlling elements of empty car management. They are highly interactive, and adjustments at any one point may have far-reaching ramifications throughout the network. Thus a computer simulation becomes a necessary part of the design process in order to develop the best logic system.

In some cases there is not one best logic. A perfect control system would have to anticipate demand changes ahead of time. Varying input of bags at different stations results in different requirements on the system. It is only possible to set relative priorites on the goals and to strike a compromise between such conflicting requirements as immediate car availability and minimum non-essential traffic. However, the simulation allows the designer to try various schemes and precisely monitor their effect throughout the system while other parameters remain constant, an accomplishment which may never be possible in the real life system.

REFERENCES

1. "Airport's $125-Million Expansion Squeezes Around Old Terminal", Engineering News Report, Feb. 18, 1971, pp. 26-31.

2. "Automated Baggage System Tested for Use by Airlines", Aviation Week & Space Technology, Oct. 20, 1969, pp. 150-161.

3. "Airport 'Roller Coaster' Gives Your Bags a Fast Ride", Popular Science, May 1971, pp. 34-36.

4. G. J. Eggert and W. W. Siesennop, "The Use of Destination-Coded Unit Carriers to Solve Airport Baggage Handling Problems", presented at the Joint IEEE/ASME Materials Handling Conference, Milwaukee, Wisconsin, Oct. 1971, IEEE Paper #71 CP 731-IGA.

5. G. J. Eggert and R. J. Patton, "Using an Array of Magnets for an Escort Memory", Automation, January 1972, pp. 50-51.

6. L. W. Hillman, "An Order Picking and Shipping Model", Proceedings of the Third Conference on Applications of Simulation, Los Angeles, California, December 1969.

7. "Engineering Performance Study Report, Baggage Handling System, Seattle-Tacoma International Airport", Seattle, Washington, 1969.

FIGURE 1

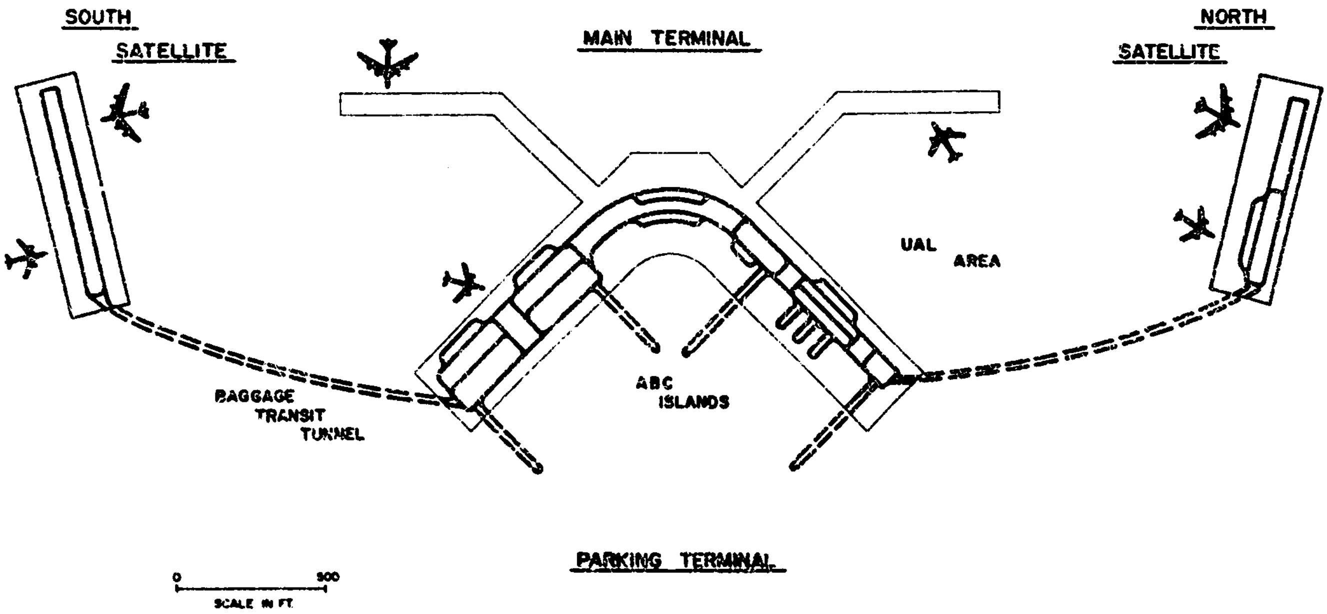

FIGURE 2
Simplified Schematic of the
GTX Baggage Handling System

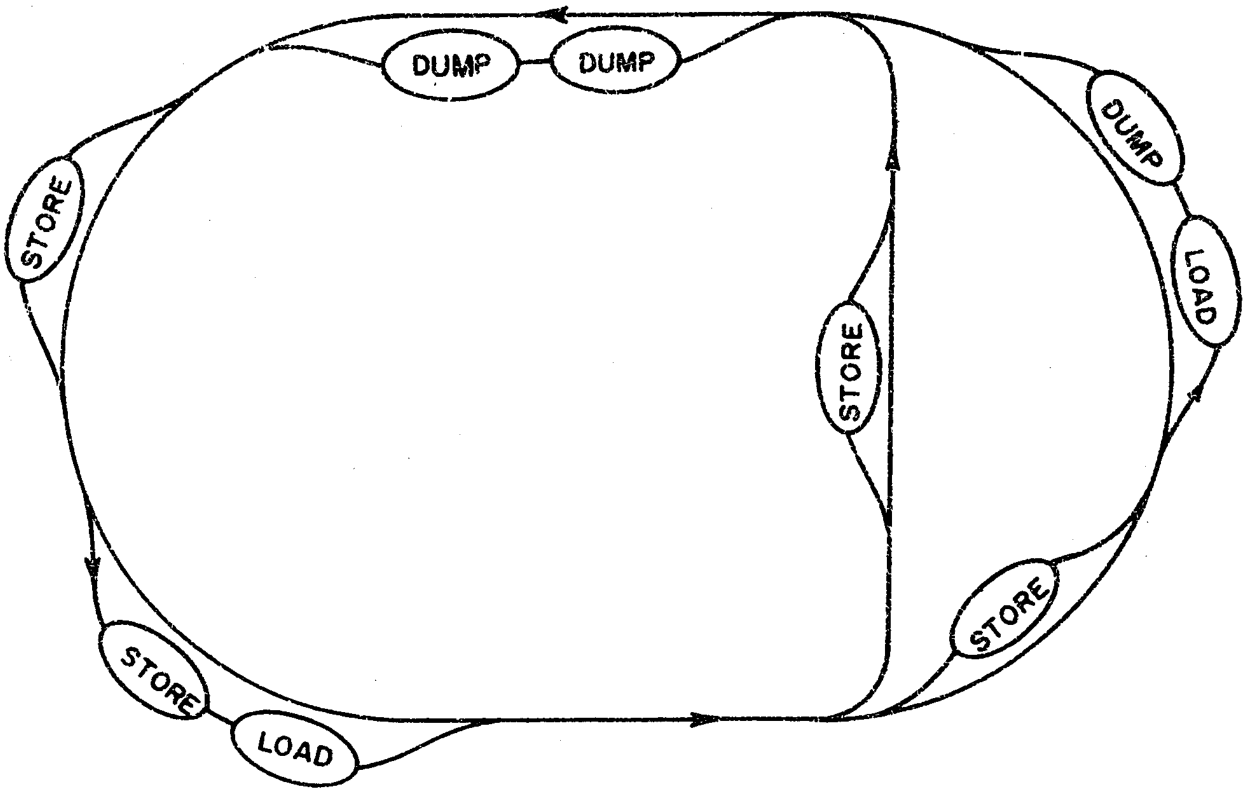

FIGURE 3
Basic Carrier Network

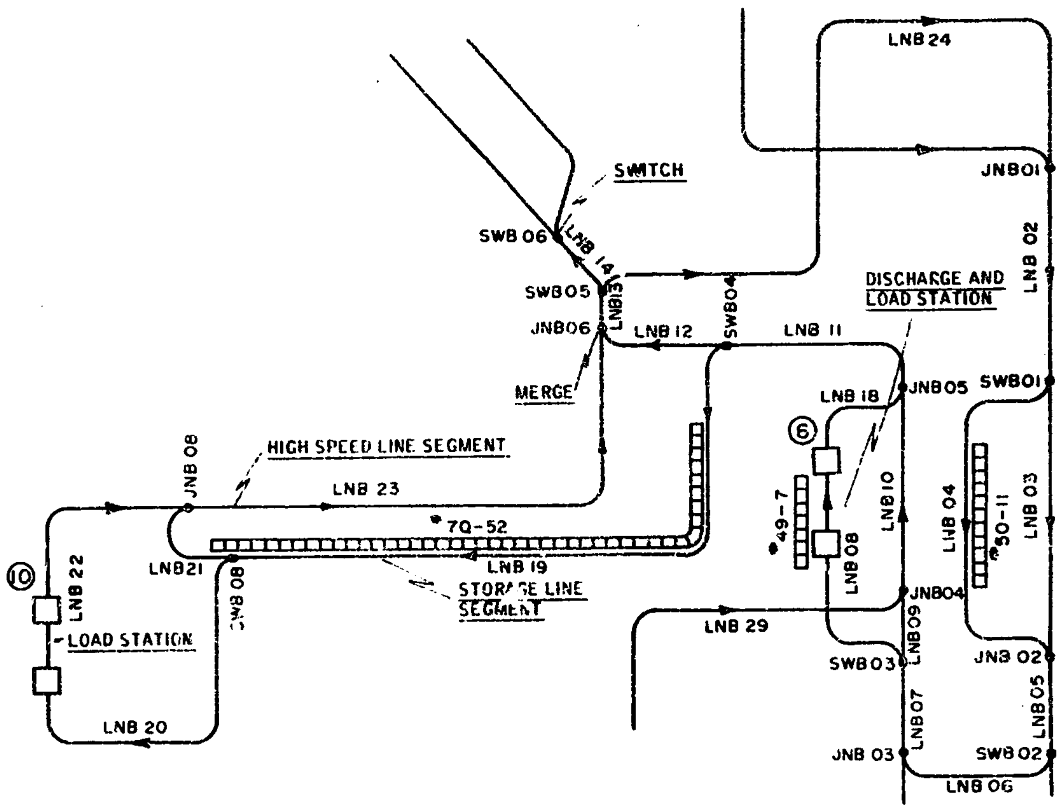

FIGURE 4
Detailed Network of a Portion of the Simulation Model

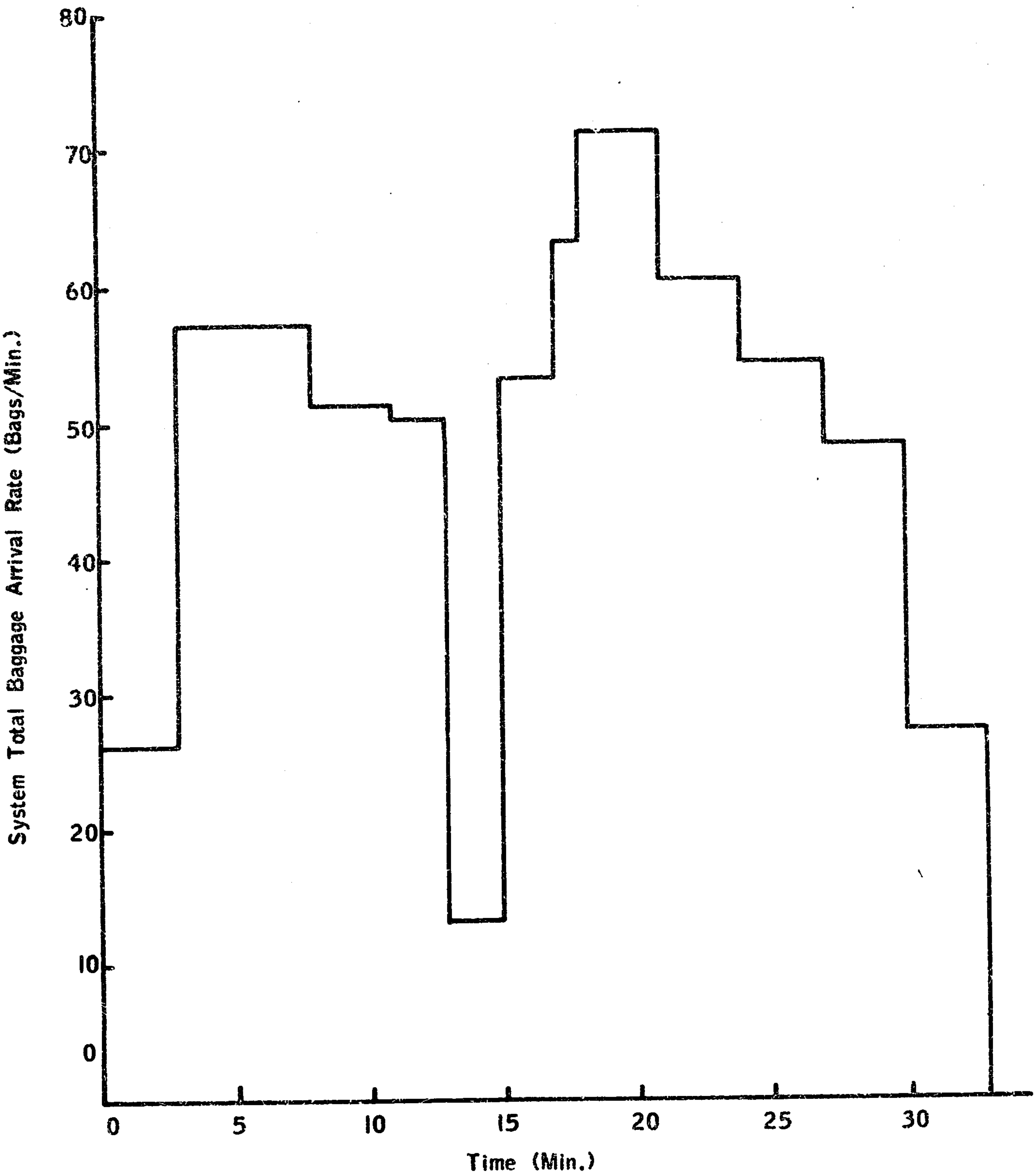

FIGURE 5
Total System Baggage Input Spectrum

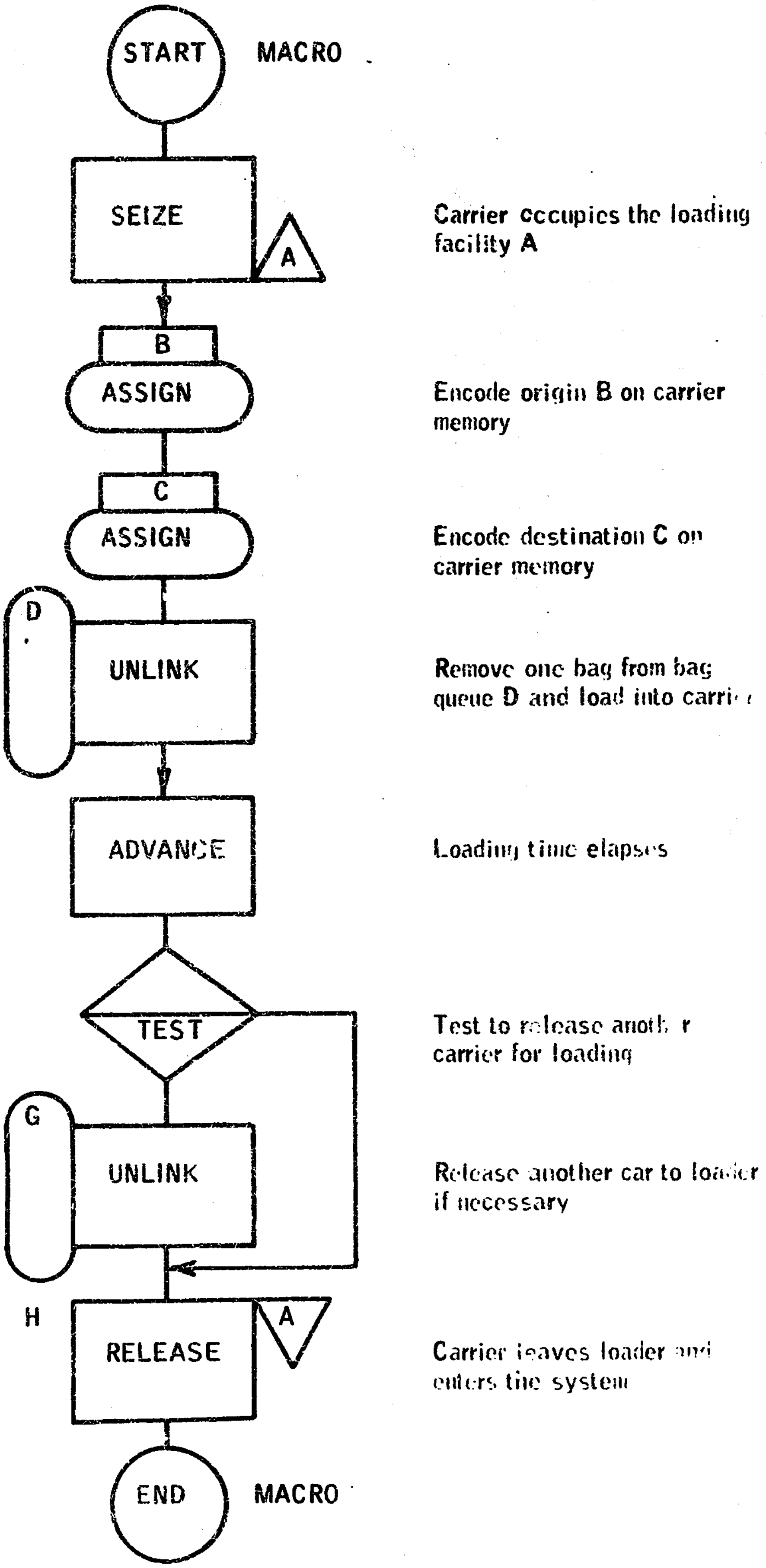

FIGURE 6
Model of a Carrier Loader — Load Macro

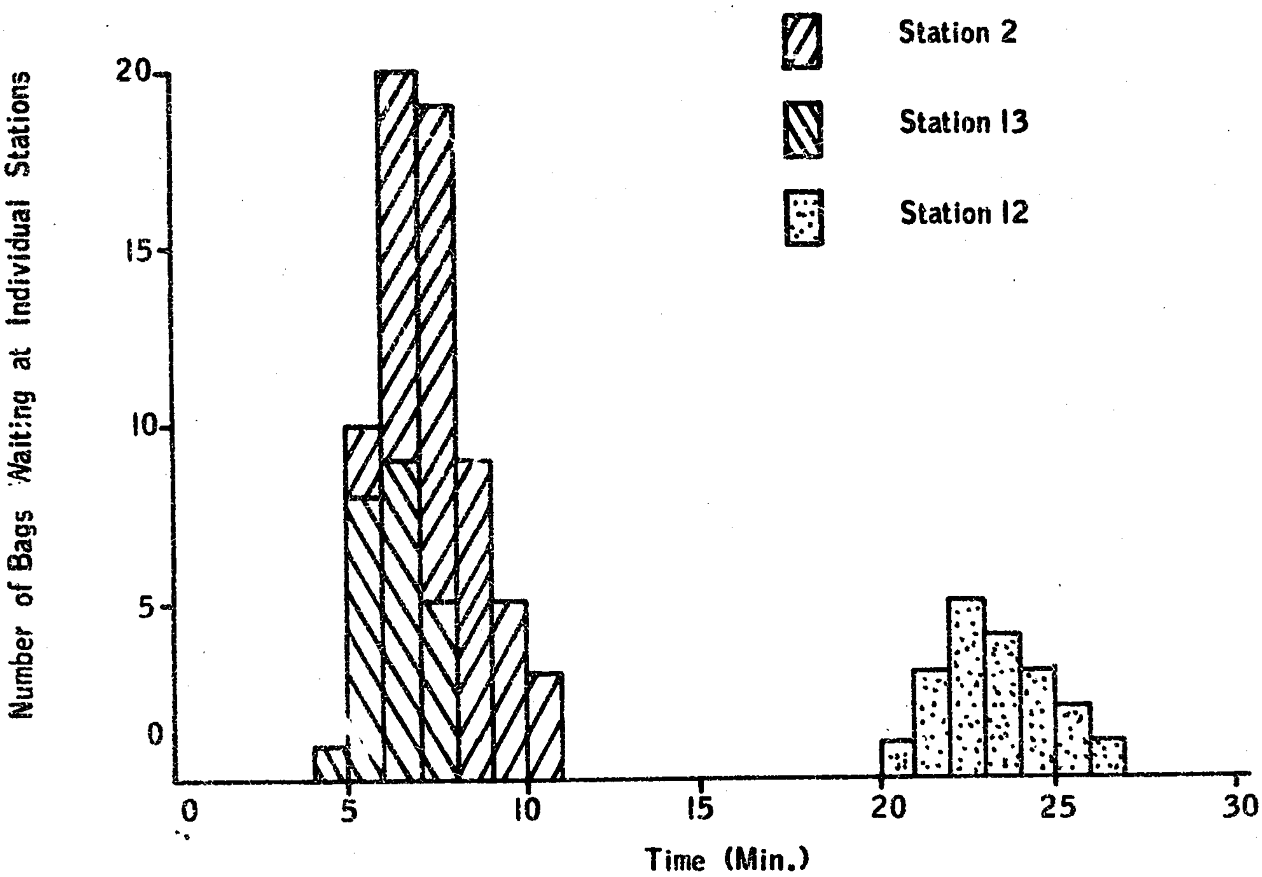

FIGURE 7
Number of Waiting Bags for Input Spectrum of Fig. 5

STORAGE NUMBER	CAPACITY	AVERAGE CONTENTS	AVERAGE UTILIZATION	ENTRIES	AVERAGE TIME/TRANS	CURRENT CONTENTS	MAXIMUM CONTENTS
LNL17	6	1.374	0.2251	81	30.543	2	4
LNA02	11	9.990	0.5447	150	71.907	4	11
LNA05	8	3.151	0.5251	146	30.802	2	6
LNAL8	7	3.310	0.4729	142	41.998	4	7
ST030	20	15.559	0.7780	97	491.351	13	20
ST031	7	7.000	1.0000	7	1800.000	7	7
ST032	6	5.977	0.9961	9	1195.333	6	6
ST033	6	5.993	0.9988	7	1541.000	6	6
ST034	18	17.284	0.9602	25	1244.440	18	18
ST035	28	15.596	0.5570	51	550.431	16	27
ST036	21	11.384	0.5421	46	445.457	20	20
ST037	24	8.805	0.3669	49	323.449	14	15
ST040	13	9.707	0.7467	17	1027.765	8	13
ST041	7	6.734	0.9621	18	673.444	7	7
ST042	7	6.580	0.9400	22	538.364	7	7
ST043	7	6.014	0.8592	40	270.690	6	7
ST044	7	6.716	0.9594	17	711.059	7	7
ST049	7	6.626	0.9466	20	596.350	7	7
ST050	13	11.270	0.8669	13	1560.462	7	13
ST051	15	1.252	0.0834	9	250.333	0	9
ST052	33	10.412	0.3155	21	892.429	7	16
ST053	19	13.342	0.7022	19	1264.000	3	19
ST054	10	7.004	0.7004	10	1260.800	2	10
ST060	30	28.869	0.9623	114	455.825	29	30
ST061	5	1.074	0.2148	11	175.727	2	3
ST070	40	12.016	0.3004	29	865.160	3	20
ST071	15	14.766	0.9844	31	857.387	14	15
ST075	33	7.265	0.2202	37	353.432	7	9
ST076	14	6.538	0.4670	37	318.054	6	8
ST080	261	0.000	0.0000	0	0.000	0	0
ST081	42	42.000	1.0000	42	1800.000	42	42
ST091	276	107.700	0.3902	138	1425.441	96	138
ST092	7	7.000	1.0000	7	1800.000	7	7
ST094	2	2.000	1.0000	2	1800.000	2	2
ST095	2	2.000	1.0000	2	1800.000	2	2
ST096	5	5.000	1.0000	5	1800.000	5	5
ST097	14	14.000	1.0000	14	1800.000	14	14
POR6	2	0.389	0.1944	15	46.667	1	2
POR5	2	0.306	0.1528	11	50.000	0	1
POR1	2	0.461	0.2303	17	48.769	0	2
POR2	2	0.957	0.4783	35	49.200	1	2
POR3	2	0.728	0.3639	26	50.385	0	2

FIGURE 8
Table of Contents of Empty Car Storages

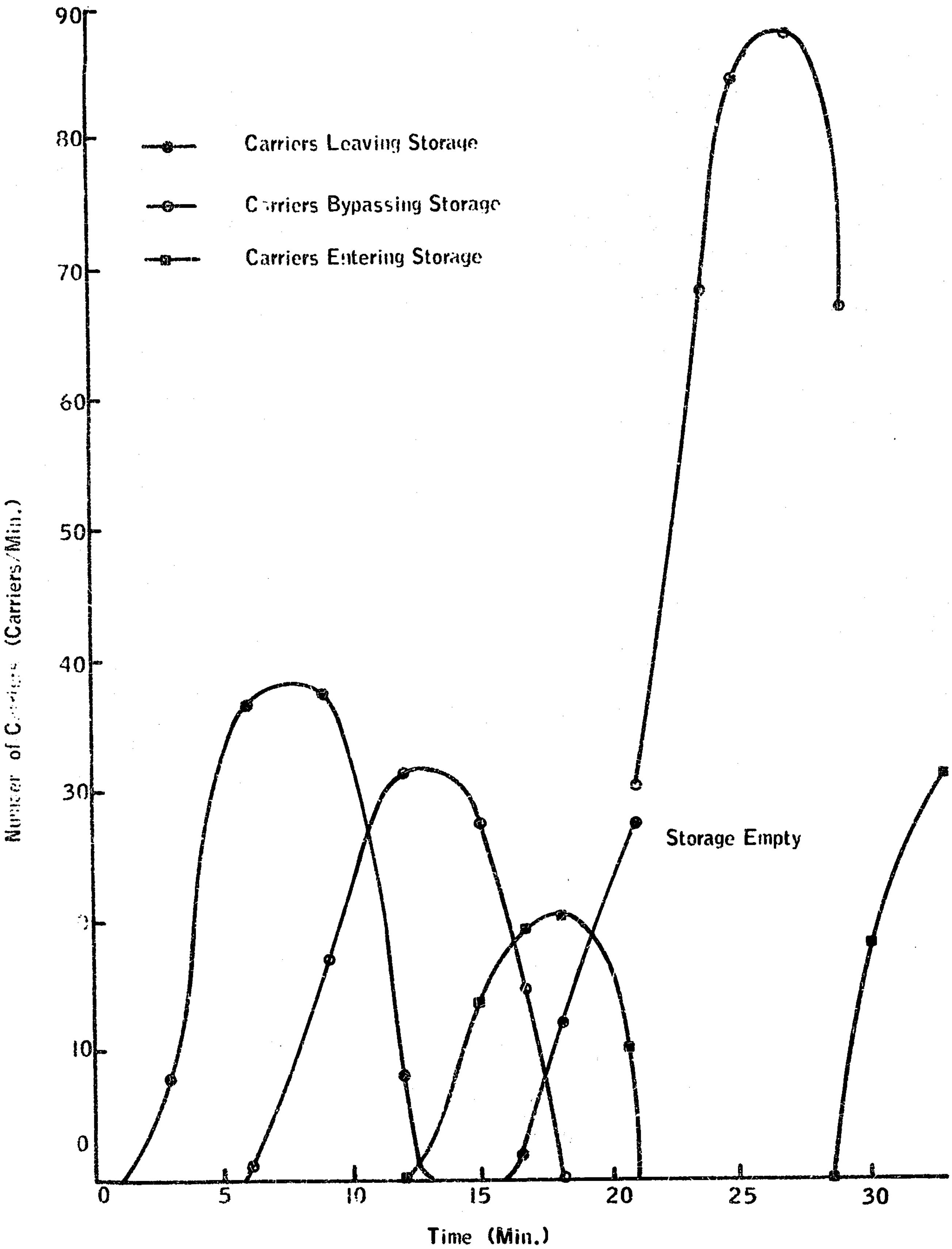

FIGURE 9
North Tunnel Traffic Activity

TABLE NUMBER ORG11

ENTRIES IN TABLE	MEAN ARGUMENT	STANDARD DEVIATION	SUM OF ARGUMENTS	
24	43.750	31.744	1050.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PERCENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
23	9	37.50	37.5	62.5	0.526	-0.654
25	0	0.00	37.5	62.5	0.571	-0.591
27	0	0.00	37.5	62.5	0.617	-0.528
29	0	0.00	37.5	62.5	0.663	-0.465
31	0	0.00	37.5	62.5	0.709	-0.402
33	0	0.00	37.5	62.5	0.754	-0.339
35	0	0.00	37.5	62.5	0.800	-0.276
37	0	0.00	37.5	62.5	0.846	-0.213
39	0	0.00	37.5	62.5	0.891	-0.150
41	0	0.00	37.5	62.5	0.937	-0.087
43	0	0.00	37.5	62.5	0.983	-0.024
45	0	0.00	37.5	62.5	1.029	0.039
47	0	0.00	37.5	62.5	1.074	0.102
49	4	16.67	54.2	45.8	1.120	0.165
51	1	4.17	58.3	41.7	1.166	0.228
53	2	8.33	66.7	33.3	1.211	0.291
55	0	0.00	66.7	33.3	1.257	0.354
57	0	0.00	66.7	33.3	1.303	0.417
59	0	0.00	66.7	33.3	1.349	0.480
61	0	0.00	66.7	33.3	1.394	0.543
63	0	0.00	66.7	33.3	1.440	0.606
65	0	0.00	66.7	33.3	1.486	0.669
67	0	0.00	66.7	33.3	1.531	0.732
69	0	0.00	66.7	33.3	1.577	0.795
71	0	0.00	66.7	33.3	1.623	0.858
73	0	0.00	66.7	33.3	1.669	0.921
75	0	0.00	66.7	33.3	1.714	0.984
77	0	0.00	66.7	33.3	1.760	1.047
79	0	0.00	66.7	33.3	1.806	1.110
81	8	33.33	100.0	0.0	1.852	1.173

REMAINING FREQUENCIES ARE ALL ZERO

TABLE NUMBER DES12

ENTRIES IN TABLE	MEAN ARGUMENT	STANDARD DEVIATION	SUM OF ARGUMENTS	
21	2.667	5.295	56.000	NON-WEIGHTED

UPPER LIMIT	OBSERVED FREQUENCY	PERCENT OF TOTAL	CUMULATIVE PERCENTAGE	CUMULATIVE REMAINDER	MULTIPLE OF MEAN	DEVIATION FROM MEAN
0	12	57.14	57.1	42.9	0.000	-0.504
1	0	0.00	57.1	42.9	0.375	-0.315
2	3	14.29	71.4	28.6	0.750	-0.126
3	2	9.52	81.0	19.0	1.125	0.063
4	2	9.52	90.5	9.5	1.500	0.252
5	0	0.00	90.5	9.5	1.875	0.441
6	0	0.00	90.5	9.5	2.250	0.630

OVERFLOW, WITH AVERAGE VALUE= 16.00

FIGURE 10
Origin and Destination Tables

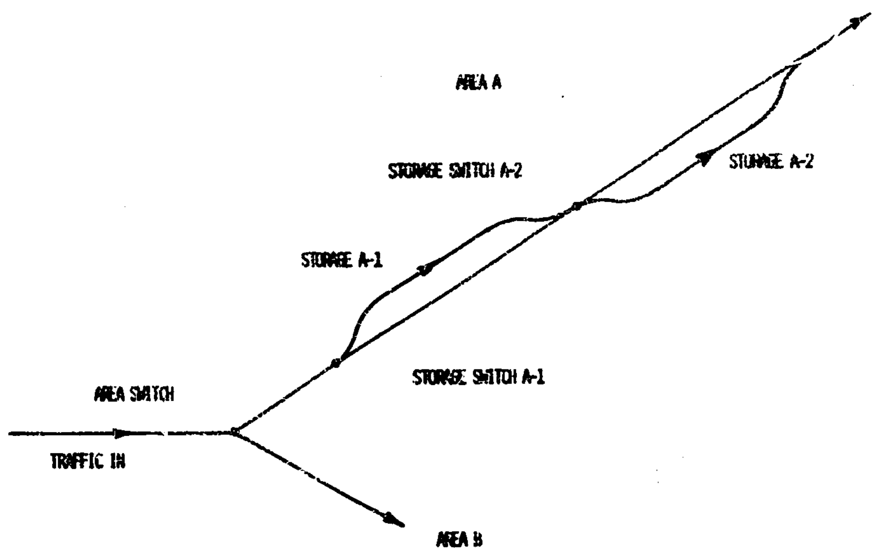

FIGURE 11
Simplified Switch Diagram