Richard D. Cuthbert
Manager, Operations Research
Corporate Information Services Division

Larry Peckham
Logistics Planning Specialist
Information Systems Group

Xerox Corporation, Rochester, New York

APL MODELS FOR OPERATIONAL PLANNING OF SHIPMENT ROUTING, LOADING, AND SCHEDULING

ABSTRACT

A vehicle routing algorithm with an imbedded loading heuristic and movement simulation was developed to provide the interactive capability to plan shipment delivery and returns.

The vehicle movement model is a small discrete event fixed interval simulation that uses the APL operators to keep track of loading facility status, truck location, etc. for scheduling purposes.

The models operate in a field environment, but are linked to centralized planning models and data bases.

They are illustrative of APL's almost unique ability to combine simulation modeling with operational restrictions such as interactive operation by non-programmers.

INTRODUCTION

This paper outlines the design of a system of APL models which support Xerox' distribution planning. It starts by describing the operating environment in which the planning occurs. This sets the stage for a discussion of the models' overall design philosophy: The power of simulation should be in the hands of the functional user rather than an operations research specialist. Some details are then given on the routing, loading, and scheduling sub-models to give insight into how they work together to aid development of a viable product delivery plan. This is followed by an explanation of how the truck scheduling simulation is conceived in terms of matrix (APL) operations. The way in which models are conceived in APL leads to a closing analysis of its advantages and disadvantages as a simulation tool, at least as experienced in this system.

SYSTEM OPERATING ENVIRONMENT

Exhibit 1 is a schematic of the system's operating environment. A national distribution staff is located in Rochester, New York. Regional operating staffs are located in five major cities: Chicago, Los Angeles, New York, Dallas, and Washington. The national staff is responsible for development of operating policies, such as what modes of transport can be used. In line with these policies, they negotiate with carriers and develop data bases needed to plan operations, such as freight rates by transport modes. Responsibility for regional movement of the product (Xerox copiers) is in the hands of the regional staffs who must also supply data on how well they are performing to Rochester.

The regional staffs must also develop the actual plans to move the machines in time to notify the carriers. They normally do this once a week and the calculations must be completed in the span of a few hours. They obtain requests for copiers from branches and must decide:

- *How many trucks* from each kind of carrier to use.
- What *routes* should the trucks take.
- How should the trucks be *loaded.*
- When should the trucks be *scheduled* to depart and arrive.

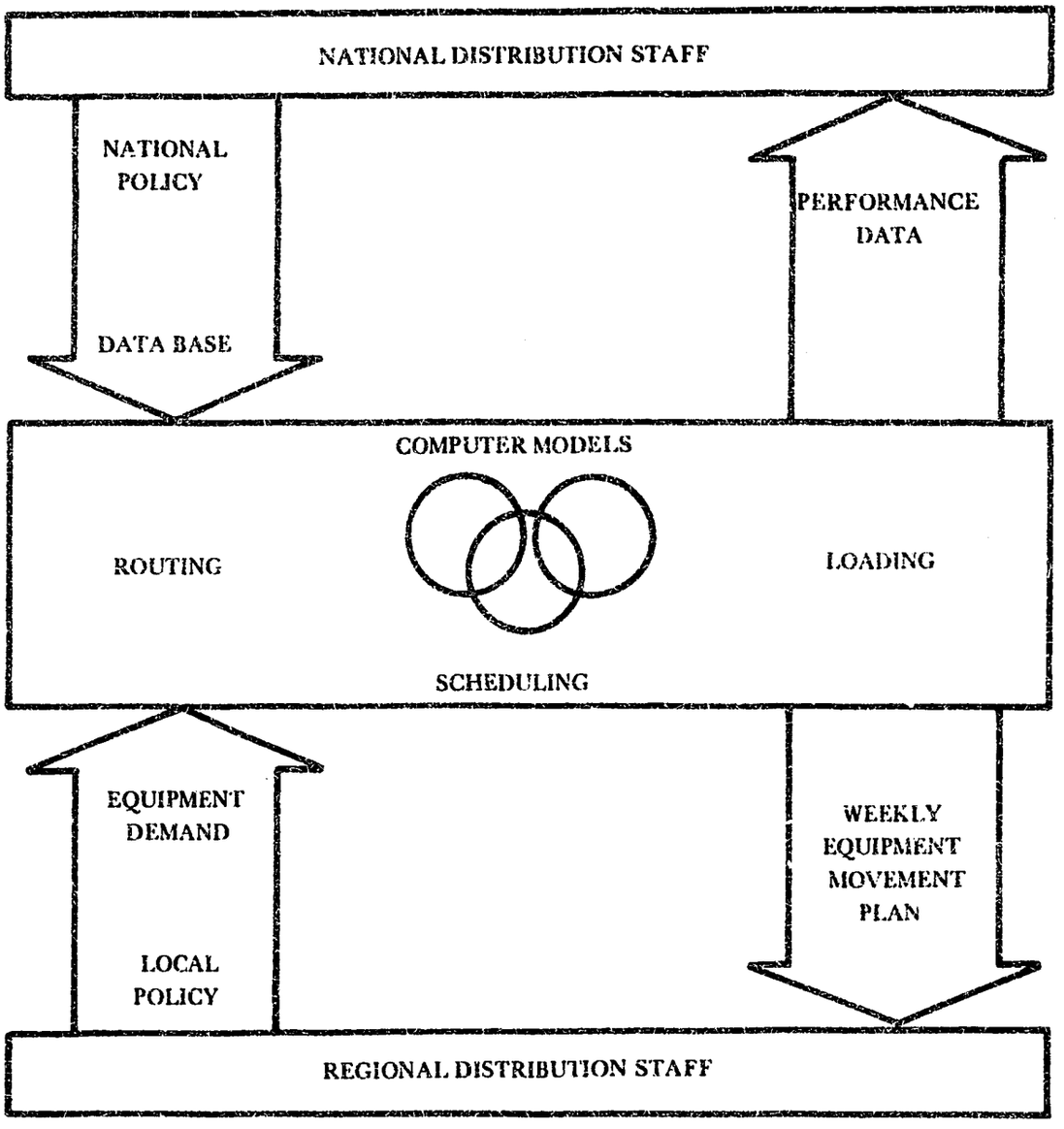

Exhibit 1: System Operating Environment

These decisions must take into account local policies and conditions known only to regional personnel; e.g., a temporary shortage of trucks available to the preferred carrier. The decisions are also interlocked; e.g., how many trucks are needed depends on how they are loaded, which in turn depends on how they are routed, and so on.

It is clear that the typical off-line optimization study by an operations analyst has no place in this environment. What is needed is a system of models that:

- Can be used by non-specialist personnel.
- Has very quick turnaround.
- Is very amenable to operator intervention to reflect local conditions.
- Can be fed data by a central staff of functional experts.
- Can be maintained by a central staff of operations analysts.

OVERALL DESIGN

Exhibit 2 shows the overall design of the system. This design is aimed at answering the needs of the distribution planners just discussed through exploitation of the power of modeling in general and of APL in particular. Later we shall review APL as a vehicle for simulation in the light of this design experience.

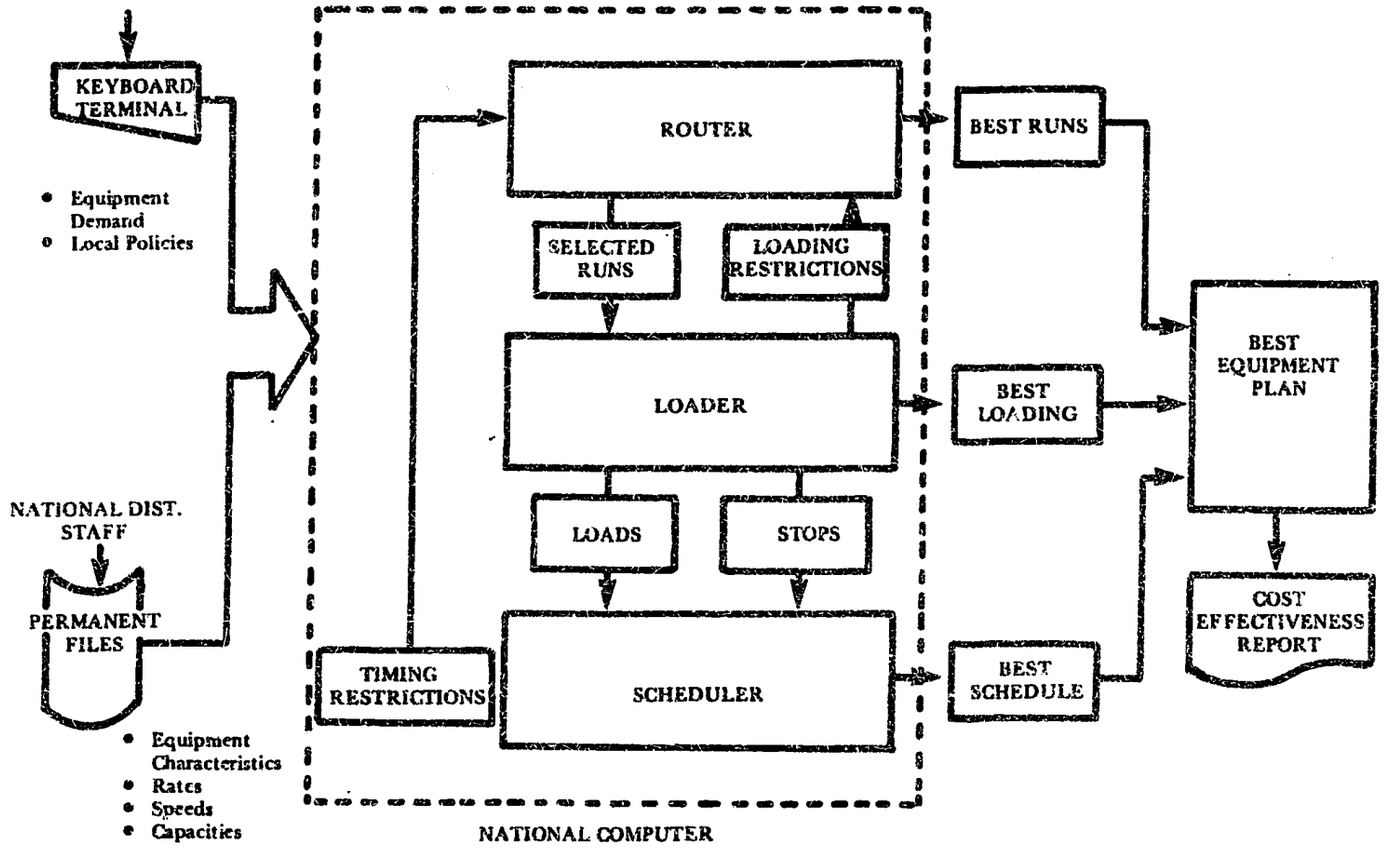

Exhibit 2: Overall Systems Design

The first feature of the planning system that should be noted is that the computer, the models and all data files are centrally located. The work was initially done on an IBM 370-145 owned by an outside time sharing service. It is currently being converted to our own XDS Sigma-7 in Rochester (an APL interpreter has only recently become available on that machine). Central location enables us to keep national level control over model refinements and data base updates. All interfacing with model including data entry is done by teletypes over phone lines, so actual physical location of the computer is immaterial except for phone charges.

The national distribution staff maintains files on product characteristics, such as shipping weight, freight rates, vehicle capacities, speeds, etc. They enter this data via file maintenance procedures over a teletype. The data is stored permanently on disk packs at the computer center. Updating is done on an as needed basis.

The regional distribution staff enters weekly branch demand for copiers, and also through answering questions, determines the local operating policies under which planning is done.

The local planner seeks to develop a best equipment plan composed of best (near cost optimum) runs or routes, best (highest feasible load factor) loading, and best (minimum vehicle usage) truck schedule.

In this task he is aided by three interlocking sub-models: a router, a loader, and a scheduler. The router examines a file of available routes and queries the loader as to how many trucks would be required to meet input demand. The corresponding delivery cost is also calculated. The router selects the cheapest routes and transport modes and ignores scheduling problems. The interaction between the router and loader is entirely automatic; i.e., occurs without operator intervention. What is produced is the cheapest routing consistent with truck capacities, but which may be infeasible from a scheduling viewpoint.

The human operator then intervenes to determine if the cheapest routing is feasible from a scheduling viewpoint. He inputs departure times for the cheapest runs consistent with his dock loading capacity. A truck movement simulator traces the movement of each vehicle and informs him of all their activities. If the trucks' arrival times are not satisfactory, he can manually input new departure times until they are; then his job is done.

If, however, no departure times can be found to make the cheapest runs feasible, he may instruct the routing routine to modify its recommendations; e.g., restrict use of one or more runs.

This iterative process is repeated until the operator is convinced he has the cheapest route consistent with all operating constraints even those the model is not directly aware of. In other words he conditions the computer solution to make it feasible in light of non-quantifiable constraints.

In all cases the model informs the operator of the cost of all alternatives. If, for example, a branch requests crash service not within the optimal schedule, he can evaluate the cost and make a decision whether to approve the variance from plan.

This segmented design approach was taken because it represented a good mix of computer and human talents under the technical limitations of APL. The calculation burden of searching thousands of routing alternatives was taken off the shoulders of the planner. The human supplied full recognition of rapidly changing local conditions. How much the technical limitations of APL influenced the design tradeoff is a later topic of discussion.

TRUCK ROUTING, LOADING, AND SCHEDULING CONCEPTS

Exhibits 3, 4, and 5 show in conceptual terms how truck routing, loading, and scheduling are accomplished.

The process begins with the planner entering the weekly demand for product pickups and deliveries through a 'START' routine. After this data is edited and filed, the START routine calls the ROUTE routine to determine a good set of truck runs. The planner can force the computer

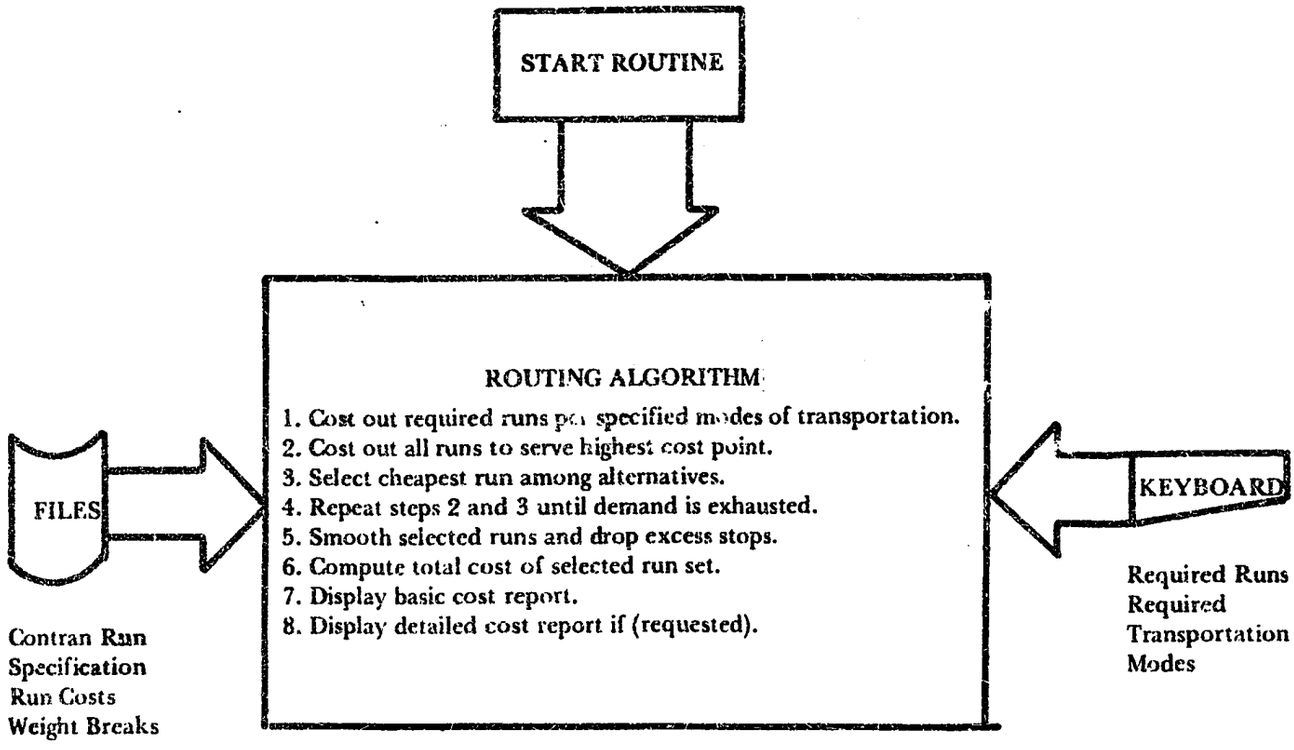

Exhibit 3: Truck Routing Concept

to make some runs for operating reasons and these will be routed and loaded first. The ROUTE routine depends on the LOAD routine to tell it how many trucks are needed and what is their load factor.

After all required runs are completed, the computer examines all branches and determines which is most expensive to serve with direct (1 stop) service. It then searches all filed multistop runs (CONTRAN) that go to that point and selects the cheapest. The machine demand is decremented by the truck's load and the process is repeated until all demand at all points is exhausted.

Then, because the above heuristic is "greedy," loads are shifted among the chosen runs to smooth the loading. This may eliminate some excess charges incurred for very heavy loads and also some unnecessary stops.

The ROUTE routine also displays the routing's total cost and run costs if desired.

The loading concept as in Exhibit 4 is interesting in that it deals with products that are relatively large compared with the vehicle. For example, only three of our large machines can be fit side by side on a truck. Products of different sizes are normally shipped together. This means space can be wasted if the loading pattern is not well laid out.

The routing logic gives the loader data on the machine demand at the potential stopping points. The loader loads the stops in reverse order since there are doors only at the end of the truck.

Big machines (duplicators) are loaded first since they are the hardest to fit. They are loaded in from front to back until they are exhausted. Then middle size machines, known to fit, are filled in beside them. Next, consoles and desktops are put in as sets of 3, 4, 5 until the first deck of the truck is exhausted. A second deck is then gone to until the truck's overall capacity is exhausted. Legal weight limits are also observed.

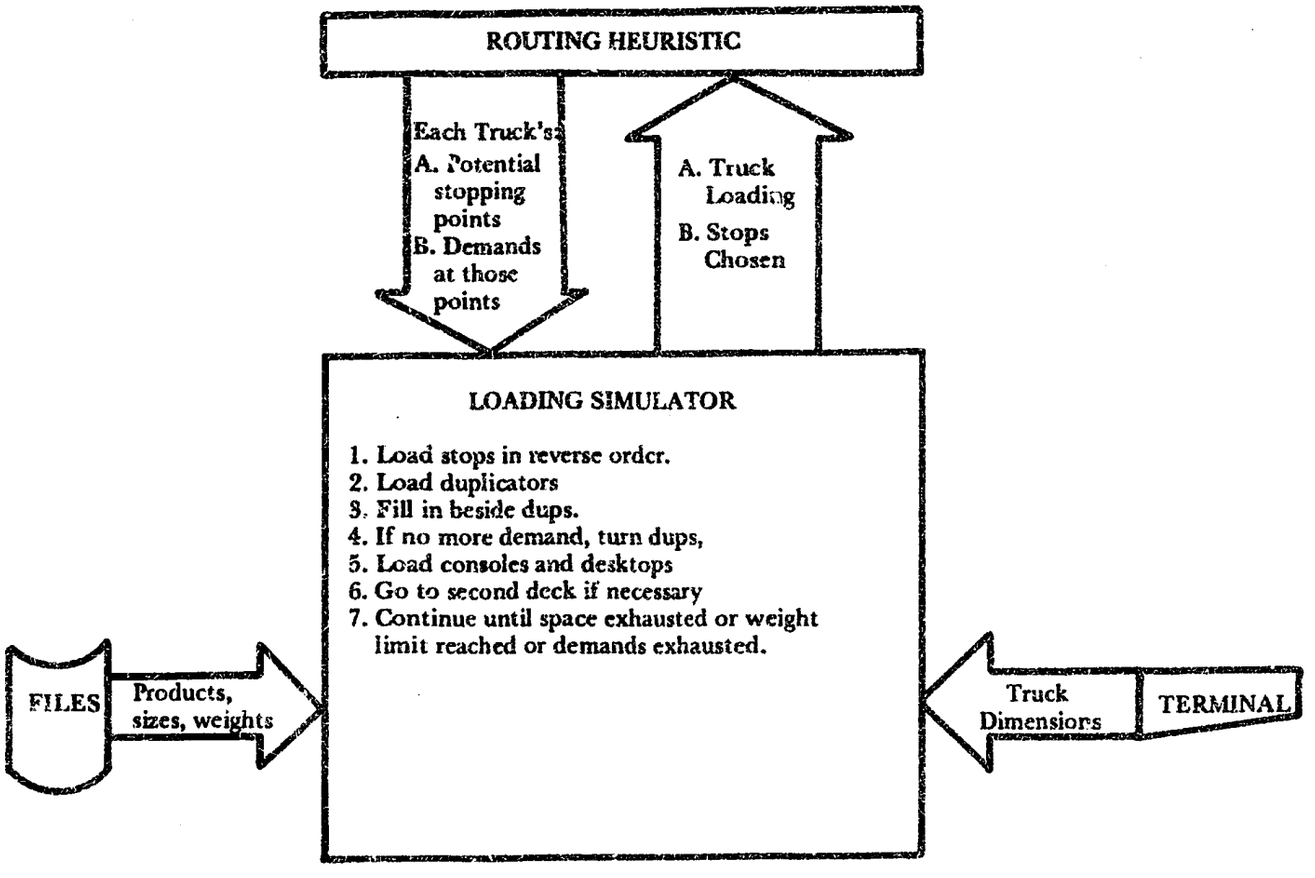

Exhibit 4: Truck Loading Concept

This loading process is essentially a simulation of human dock loading practice and was tested and refined through competition with experienced dock personnel.

Note, however, that the point is not to "beat" the man on the dock, but to make the planner aware of how the dock man would behave if he were told to load a given block of machines. This enables the planner/computer to try many different routings and loadings to determine the cheapest.

Exhibit 5 shows the scheduling concept and how it relates to the rest of the system.

The scheduler is called upon to determine the truck departure and arrival times for the runs initially selected by the routing process.

The scheduler accepts manual starting times and simulates truck movements taking into account truck speeds, inter-city mileages, legal restrictions on driving hours, and the operating schedules of the delivery point facilities. How this simulation works in detail is discussed in a moment to give a better feel for APL model formulation.

The computed arrival times are displayed so that the planner can determine if the schedule is satisfactory. If it is, the planning job is completed. If it is not, he can juggle departure times to avoid dock congestion or weekend runs until he arrives at a satisfactory schedule. Occasionally he is forced to redo the routing process with restrictions on the use of runs that cannot be satisfactorily scheduled. In other words, the routing-scheduling process is a feedback iterative process controlled by the systems operator.

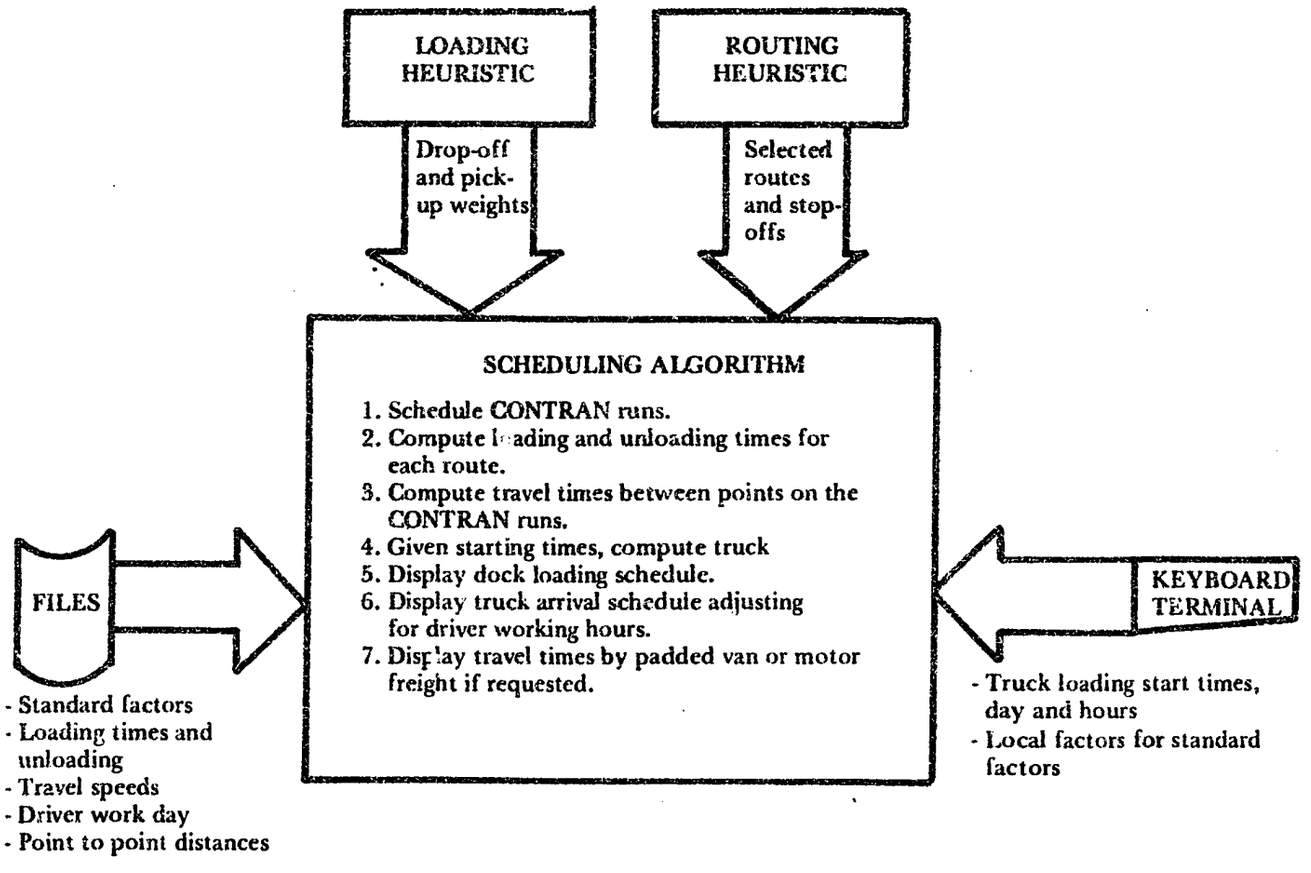

Exhibit 5: Truck Scheduling Process

APL FORMULATION OF THE TRUCK MOVEMENT SIMULATION

The SCHEDULE routine is required to determine the arrival time of trucks given departure times, truck speeds, inter-city distances, etc.

Exhibit 6 shows how this problem was formulated for APL coding. First, each trip was conceived of as occurring without delays due to facility shutdowns or logged driving hour limitations. The truck was loaded, rolled to the first stop, unloaded, rolled to the next stop, etc.

Inter-city distances and driving speeds were drawn from files and travel times were computed from their ratios. Shipment weight at each stop was drawn from the routing logic and multiplied by a loading factor per pound. The sum of the loading time at the prior stop and the inter-city travel time between each stop and the prior stop, gives the time between arrivals in working hours.

Input supplies working and driving hour limits and facility open hours. This information is used to set up facility and crew status vectors.

A time counter is advanced starting at the input beginning point. The truck is loaded for as many hours as it takes to fill it. The crew's time is not taken up in the initial loading. Once the truck is full, the crew's status is marked as working and driving. This status is kept as each hour passes until an arrival is due. The crew's log book or status vector is scanned each hour to determine how long they have worked. When they have worked their limit, they are forced to sleep. If the truck arrives at a facility whose status is closed, time advances and the crew is forced to "sleep" again.

The effect of this process is to interject dead time into the arrival time schedule originally computed. In other words, we convert working hours into calendar hours by taking into account departure time, sleeping time, and time spent waiting at shut facilities. There is no way these dead times could be calculated without tracing the prior history of truck movement, hence the procedure is rightfully called a simulation.

The advantage of the approach is that in general individual entities and events need not be traced in the coding. For example, the array ARIV is a single matrix that is computed in a few steps without iterations or scanning to trace an individual truck making individual stops. Matrix operations (x ÷ etc.) permit doing the job in one pass.

Other operations are used to flag facilities as open or shut to all trucks. In other words, as much homework as possible is done before getting into the time scanning (which does involve a FORTRAN type loop). In addition, historical scans of the facility and crew status vectors are done directly with single operators. This prevents loops occurring within loops.

These steps are taken because:

- It is conceptually easier to use the APL primitive operators (after you get used to the idea).
- APL is interpretive and loops should be avoided since each line would be retranslated over and over.

When formulating APL problems for the experienced coder, it is often not necessary to go into much more detail than Exhibit 6. This enables the analyst and programmer to converse on the conceptual level rather than the coding level.

APL AS A SIMULATION MODELING TOOL

Exhibit 7 summarizes our experience with APL as a simulation modeling tool. In systems development, the main thing that stands out is the sheer coding speed of APL over FORTRAN. For heuristics, models, and simulations, we estimate it to be perhaps five times faster. This savings comes from the fact that it is possible, as we have seen with the truck movement simulator, to conceive the coding in terms of matrix operations that are conceptually very close to the basic processes to be modeled.

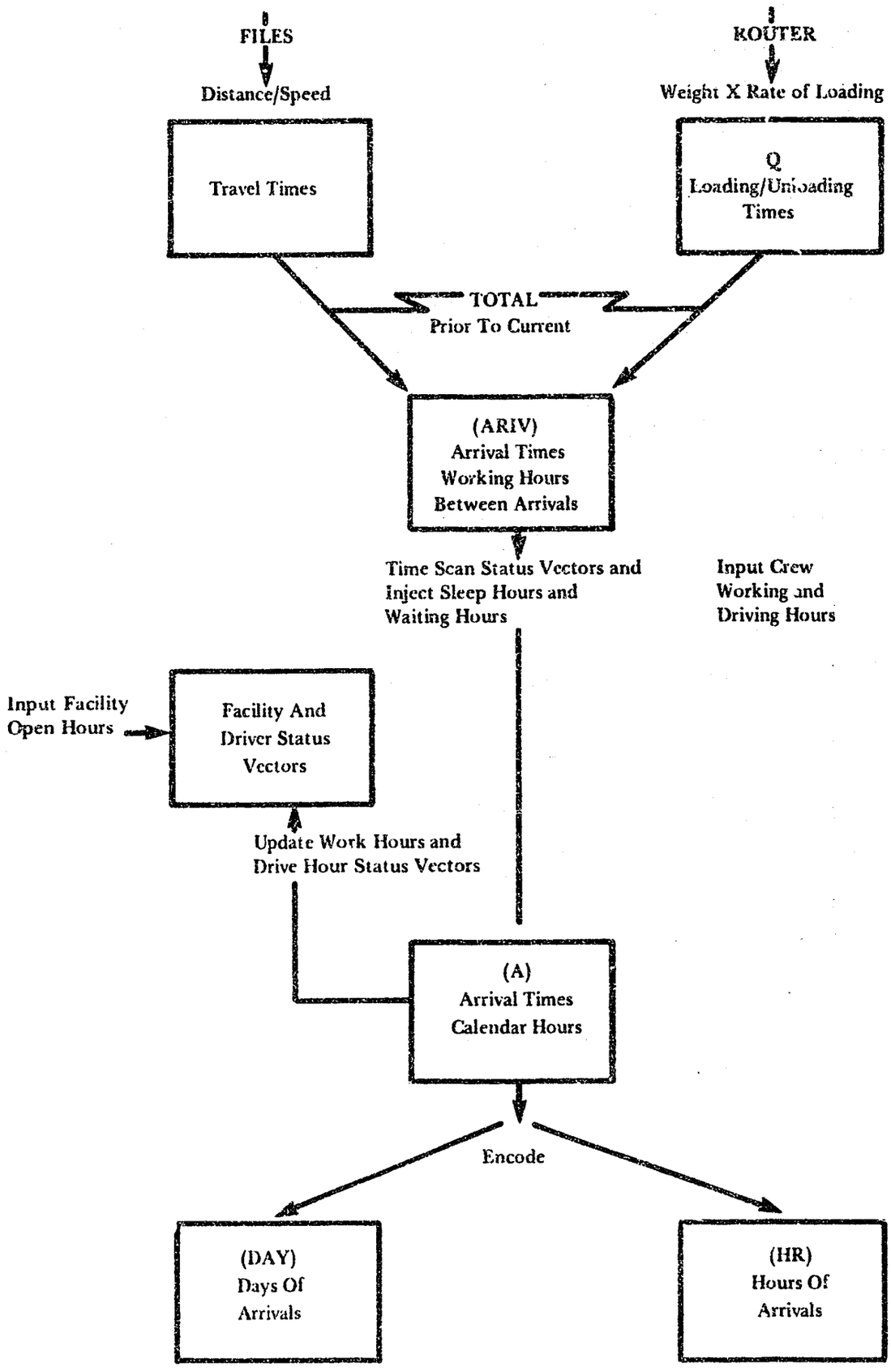

Exhibit 6: Scheduling Simulation APL Formulation

Besides reducing the coding bottleneck, APL's speed has another major influence on the project. It made it possible to experiment with alternative heuristics on an ongoing basis; i.e., to conduct research as part of the planning system design. This turned out to be very important because we were working in an area where operations research was required before the final design could be selected. For example, we originally thought linear weight and area constraints would be sufficient to determine product loading feasibility. Experiments comparing the loads obtained against dock loading practice showed it was necessary to simulate the fitting of each machine on board the truck.

If coding speed is APL's strength, then its biggest weakness in our experience was the small available workspace. The total storage requirement of the system exceeded the workspace available, forcing us to segment the routing and scheduling into two separate workspaces. We don't believe that this hurt the project too badly in terms of performance of the system because the interface between these two processes probably will always require manual intervention anyway. However, under current APL software, program swapping or chaining is not possible at the speeds necessary for algorithmic searches or large scale simulations. This is a major limitation of the language which is currently being addressed by software

SYSTEM CHARACTERISTIC			INDICATOR OR EXPERIENCE		COMMENTS
			Elapsed Time (Months)	Effort (Man Months)	
Development	Design		1.0	2.0	Time largely unaffected by APL characteristics
	Coding		1.0	2.0	Major reduction in time over FORTRAN due to APL coding speed.
	Test And Implementation		3.0	9.0	Major reduction in time over FORTRAN due to rapid rewrite capability and terminal editing.
Operating	Time	CPU	3 - 4 Minutes		Much heavier CPU use not practical on interactive basis.
		Terminal Connect	20 - 40 Minutes		Much longer elapsed times not practical on interactive basis.
	Size	Coding	Almost 48,000 bytes or 6,000 characters		Coding is stored and interpreted - not compiled.
		Total	About 75,000 bytes		Small workspace size (48,000 bytes) forced use of program segmenting.
	Cost	CPU	$30		Estimated payoff per use: $200
		Terminal Connect	$5 - 10		Relatively small proportion of total cost for APL application. Depends on distance to computer.

Exhibit 7: Summary of System Characteristics

groups. Disk files for data storage are available and were used extensively.

In terms of running time, APL is faster than we originally anticipated, but slow enough to force us to limit the amount of route searching done in the system.

APL is an interpretive language; i.e., it retranslates source coding on a line-by-line basis. This imposes a translation overhead which can be very severe in highly repetitive search processes. It is not only desirable to conceive models in terms of matrix operations, it is mandatory if speed is a consideration. Carrying out processes as in FORTRAN row-by-row or column-by-column is not practical.

If the model is to operate in an interactive mode (as do most APL models), the amount of calculation that is practical is limited. In our system the user, at some points, must wait up to 10 minutes elapsed time for calculations to be complete. Phone connections are not reliable enough to push that figure much higher.

Fortunately we found that the system helped the planner reduce the distribution costs significantly within the limits imposed by the technology. The direct savings came out about 5-10 times system operating costs.

CLOSING SUMMARY

Summarizing our experience with APL in this system, we were generally impressed. We are converting most of our modeling efforts over to the language because of its coding speed advantages. We are also requesting our software groups to address the program chaining and workspace size issues on our own machine. We anticipate a 50% increase in workspace size.

Perhaps the best thing that can be said in APL's favor is that it makes it technically possible to put models where they belong (in the planner's hands) in an amazingly short period of time.

He then can participate in the testing process and give much more rapid feedback on an approach's relevance to the real problem. This goes a long way in overcoming the implementation gap that often occurs with operations research based projects.