Proceedings of the 36th Annual Meeting Transportation Research Forum

Volumes
1 and 2

November 3-5, 1994

Daytona Beach, Florida

Published and Distributed by:

Transportation Research Forum
1730 North Lynn Street, Suite 502
Arlington, VA 22209

An Expert System for Node-Link-Node Application

*William H. VanMarter, Jr.**

ABSTRACT

Railroad management requires node-link-node applications such as mileage masters and train routing systems for car hire accounting, operations analysis, and internal costing. Existing computer programs and files often include thousands of complex, individually maintained records. It has proved difficult to update, verify, or modify these systems given today's flexible operations and "real time" information needs. We present an easily-updated expert system which automatically generates train routing between any two stations on the railroad. The general logic can also be applied to a mileage master or any other node-link-node system. Processing and file layout is simple enough to be done on a PC spreadsheet.

INTRODUCTION

Node-link-node applications such as train routing systems and mileage masters are essential accounting and decision support tools for railroad management. A railroad with 1000 stations, small for a Class 1, requires 1 million computer records to cover all possible station to station moves in either direction. Even if the file is set up on a segment to segment basis, there are usually thousands of combinations which must be separately maintained. At a time when operations and even physical layouts are changing rapidly ,it has become difficult if not impossible to update, verify, or adapt the old individually maintained file structures.

We have developed an expert system (called Automated Route File or ARF) that on request generates the train routing sequence between any two stations on the railroad. When operations and/or physical layout changes, or when doing comparative routing analyses, the knowledge-based maintenance file can easily be updated. Any further processing (online or batch) of from/to station pair points will automatically reflect the new assumptions. If the expert system logic were applied to a mileage master, the entire mileage master could be "remiled" to reflect the changes. This avoids laborious and error-prone updating of individual records, and ensures consistent routings between similar moves.

ARF currently generates train routing using expert logic, but looks up miles for each train run from a conventionally maintained mileage master. An unexpected benefit has been to compare the total of the intermediate miles generated as a by-product of the expert system processing, with the endpoint to endpoint miles read off the conventional mileage master (which itself necessarily has inherent routing assumptions). Any significant difference indicates a probable error either in the expert system knowledge base or in the conventional mileage master. A warning message on the screen prompts the user to investigate. This reciprocal

verification allowed for a high degree of confidence at startup of ARFF, and although most discrepancies have been found, continues to serve as an audit check.

ARF was programmed in COBOL on a mainframe because of its system-wide application. The logic is simple enough to be included in a PC spreadsheet using macros, IF/THEN logic and lookup tables.

The expert system is illustrated by presenting key files and concepts, and then walking through the logical process. Reference is made to a schematic of an "Illustrative Railroad" in Exhibit 1, and to key file layouts in Exhibit 2. Stations 023, 320, and 710 are highlighted for purposes of discussion.

KEY FILES AND CONCEPTS

* Processing is in three basic steps; each accessing a key file

(1) Origin Station to Near Node (Station to Node file)
(2) Near Node to Far Node (Node to Node file)
(3) Far Node to Destin Station (Station to Node file)

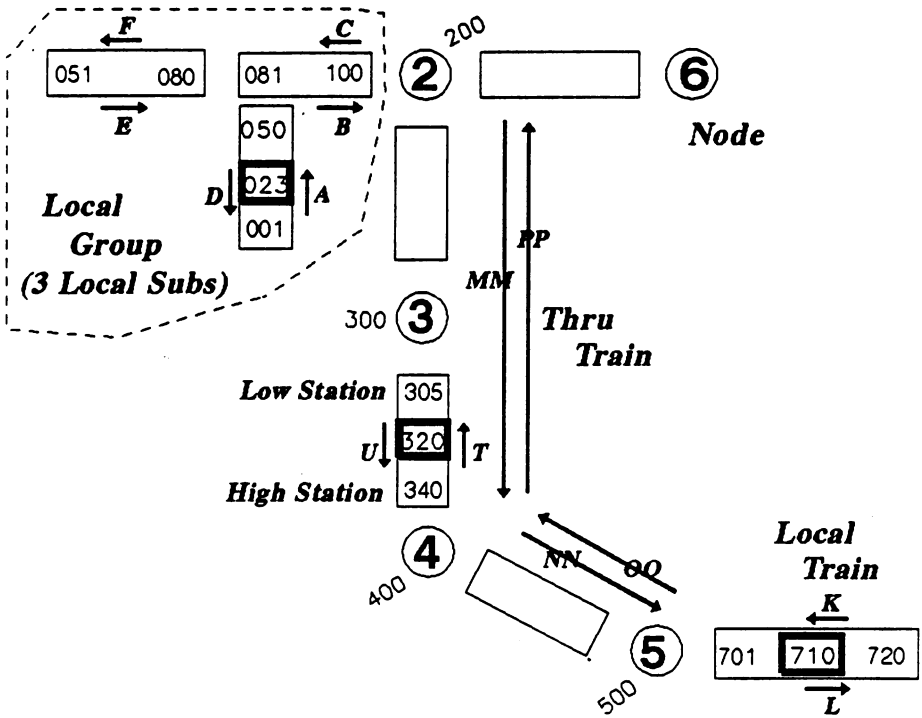
* The smallest building block is the Local Sub, which is a numerically sequential set of stations all handled by the same train(s). Local Subs are represented by the rectangular "links" in Exhibit 1. The Local Sub low-high stations constitute the key fields in the Station to Node file. This allows a number of stations to be handled in one computer record, which is a marked improvement over having a record for every station.

* Captive stations (023,710) have no choice in their Near Node. Decision stations (320) require special logic to determine which Near Node applies to a particular move. Including this special logic (less than/greater, AND/OR connectors) in the file structure rather than in the working program simplifies application and permits a great deal of flexibility and creativity.

* The Node to Node file allows reading the specific train routing from the Near Node directly to the Far Node, and avoids the difficult logic of piecing together trains across the many interim nodes. Exhibit 2 shows a Node to Node file record for Node 2 to Node 5. Again, the expert knowledge is found in the easily-maintained file structure rather than in the program itself.

* The Node to Node file size increases geometrically. Thirty nodes require 900 records if reverse direction is assumed, and adding the 31st node creates 62 more records to be maintained. Should the number of nodes exceed 25 to 30 or so, it may be more practical to maintain two or more separate Node to Node files connecting end-to-end, rather than have one oversized file.

EXHIBIT 1
Illustrative Railroad



* A Decision Station to Decision Station move means both Near Node and Far Node vary depending on the other. It is necessary to "fix" or default the initial Far Node to allow the Near Node to choose the proper routing. This is accomplished by the "D to D" field in the Station to Node file (Exhibit 2).

* A large city may have more than one node to account for operational or physical differences, i.e. runthru trains to various i/ch roads.

* Stations in a large city or on small spurs may be seldom-used and/or out-of-numerical sequence. A pesky housekeeping problem is easily solved by using the station to node file to dummy these stations to their near node.

EXHIBIT 2

Station to Node File

KEY FIELDS							DECISION STATION LOGIC									
Low Station	Hi Station	Local Group	Local Sub	Rec #	Decision or Captive	D to D	LESS/ EQUAL GRT'R/	Far Node	AND /OR	LESS/ EQUAL/ GRT'R/	Far Node	Near Node	Station to Node Train	Station to Node Station	Node to Station Train	Node to Station Station
001	050	3	1	1	Captive	2						2	A	081	C	081
													B	200	D	xxx
051	080	3	2	1	Captive	2						2	E	081	C	081
													B	200	F	xxx
081	100	3	3	1	Captive	2						2	B	200	C	xxx
701	720	9	1	1	Captive	5						5	K	500	L	xxx
305	340	5	1	1	Decision	3	LE	3	OR	EQ	6	3	T	300	U	xxx
305	340	5	1	2	Decision	3	G	3	AND	NQ	6	4	U	400	T	xxx

Node to Node File

From Node	To Node	Train	Station
2	5	MM	400
		NN	500
5	2	OO	400
		PP	200

PROCESSING ILLUSTRATION

Overview of processing sequence to create Output File (Exhibit 3)

- Look up origin and destin station numbers in Station to Node file (Exhibit 2) to see if the stations have the same Local Sub or Local Group numbers. Stations 023 and 710 are in different Local Groups 3 and 9. If stations were in the same Local Group, the Near Node would not be involved in the move. Processing would then pass to either Local Group files (keyed on Local Sub 1 to 2, 1 to 3,etc.), or Local Sub files (keyed on low to high or high to low station). These auxiliary files are necessary, but are not discussed further.

- Reading Station to Node File, find that Station 023 is captive to Near Node 2, and takes Local Train "A" to Station 081, then Local Train "B" to Station 200 (Node 2). Write this data to Output file.

- Reading Station to Node File, find that Station 710 is captive to Far Node 5. Therefore Node to Node move is 2 to 5.

- Looking up 2 to 5 in Node to Node file, find that Thru Train "MM" operates Node 2 (Station 200) to station 400, then Thru Train "NN" operates Station 400 to Node 5. Write this data to Output File.

- Reading Station to Node file for Station 710, for Node to Station direction, find Local Train "L" from Station 500 (Node 5) to Station 710 ("xxx" in the file). Write this data to Output File.

In actual practice, the program automatically finds and writes routing in both forward and reverse directions to the Output File.

EXHIBIT 3

Output File

023 710	Origin Station Destin Station	From Station	Train	To Station
	Station to Node Processing	023 081	A B	081 200
	Node to Node Processing	200 400	MM NN	400 500
	Node to Station Processing	500	L	710

The example above considered a Captive Station to Captive Station move. Captive Stations are those that must pass through a specified Near Node before going onto the rest of the system.

Station 320 is a Decision Station, because its Near Node will be either 3 or 4 depending on final destination. Processing of a decision station requires the further step of choosing the Station to Node record which satisfies the less-than,equal, or greater-than criteria with respect to the Far Node. For example, to go from Station 320 to Station 710, the program will determine that Far Node (Station 710) is 5. Browsing the Station to Node records relating to Station 320 (Exhibit 2), the program finds that Far Node 5 is "Greater than 3 and not equal to 6", and writes to Output File "Local Train "U" to Station 400". The Node to Node file would then be read to generate the train routing for Node 4 to Node 5. Finally, the Station to Node file is read for Node 5 to Destin Station 710.

POTENTIAL RAILROAD USES

- Generate expected train-station-train routings between any two stations in PC or mainframe,online or batch form. Allows backtrack, alternative routings due to hazmats or interchange connections, and other unusual situations. This is particularly useful for prospective routing needs such as internal costing and what-if analyses.

- By adding train schedule lookups and/or train size/capacity data in conjunction with fuzzy logic, would provide ability to figure probability of making connections, alternative connections, and transit times.

- Suitable for railroad (or highway) mileage masters. Can accomodate different types of miles (operating vs tariff/short line vs rate basis miles) by adding fields to the basic records and "toggle switch" option to the input request.

ENDNOTE

* The author is Manager Contract Analysis, Development Department, CP Rail System.