User's Guide For a Computerized Track Maintenance Simulation Cost Methodology

Richard L. Smith  
Allan I. Krauter  
Joseph Betor

Shaker Research Corporation  
Northway 10 Executive Park  
Ballston Lake NY 12019

February 1982  
Final Report

This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.

U.S. Department of Transportation  
Federal Railroad Administration

Office of Research and Development  
Office of Improved Track Structure Research  
Washington DC 20590
This User's Guide describes the simulation cost modeling technique developed for costing of maintenance operations of track and its component structures. The procedure discussed provides for separate maintenance cost entries to be associated with definable track substructures such as rail, cross-ties, or ballast. In this manner separate tabulations of maintenance expenditures can be obtained from the computerized technique.

This guide describes the background of the technique as well as provides two examples of the application of the costing procedure. The maintenance costing examples provided illustrate the use of maintenance action diagrams representing the system being modeled. The two-example systems involve time-dependent cost estimating and produce costs by year for the class of track component or substructure repaired; type of maintenance operation; as well as by several costing subcategories including labor, material, equipment, delays, scrap, fines, etc.

Although the computer program is tailored specifically for track maintenance cost analysis, user definable flexibility is built into the analysis. Time-dependent aspects of costs, which can vary with track loading MGT, railroad policy, track component quality, and/or Federal Safety Standards, can be entered in the simulation with the aid of user definable functions.
null
Inadequate return on investment by segments of the domestic railroads is an underlying cause of many problems being faced by the industry. One of the consequences of poor financial conditions is an inadequate level of track maintenance resulting in steadily rising track-related accident rates and a decrease in operational efficiency. One study of deferred track maintenance estimates that 123 million ties and 25,000 track miles of rail are in need of replacement at a cost of 17 billion dollars.

The Federal government has attempted to initiate a variety of programs, regulations, and economic incentives to help the industry respond to these problems. One of these programs is the Improved Track Structures Research Program, sponsored by the Federal Railroad Administration (FRA), Office of Research and Development. This program is exploring alternative approaches to achieve a reduction in track-related accidents; a reduction in track-life cycle costs; and improvements in rail service through quality lower-cost maintainability.

This guide is intended to document a simulation cost methodology developed under the program.

This work was conducted for the FRA through the Transportation Systems Center (TSC), Cambridge MA. Robert Smith was the TSC Technical Monitor.

We acknowledge the guidance and cooperation of individuals from two railroads for their assistance in gathering and defining the necessary data for this effort. Robert Tuve, Manager, Quality Control Engineering, Southern Railway System, and Harry Schultz, Assistant Chief Engineer, Delaware and Hudson Railway, contributed significantly.
### Approximate Conversions to Metric Measures

<table>
<thead>
<tr>
<th>Symbol</th>
<th>When You Know</th>
<th>Multiply by</th>
<th>To Find</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>in</td>
<td>inches</td>
<td>2.5</td>
<td>centimeters</td>
<td>cm</td>
</tr>
<tr>
<td>ft</td>
<td>feet</td>
<td>30</td>
<td>centimeters</td>
<td>cm</td>
</tr>
<tr>
<td>yd</td>
<td>yards</td>
<td>0.91</td>
<td>meters</td>
<td>m</td>
</tr>
<tr>
<td>mi</td>
<td>miles</td>
<td>1.6</td>
<td>kilometers</td>
<td>km</td>
</tr>
</tbody>
</table>

### Approximate Conversions from Metric Measures

<table>
<thead>
<tr>
<th>Symbol</th>
<th>When You Know</th>
<th>Multiply by</th>
<th>To Find</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>mm</td>
<td>millimeters</td>
<td>0.04</td>
<td>inches</td>
<td>in</td>
</tr>
<tr>
<td>cm</td>
<td>centimeters</td>
<td>0.4</td>
<td>inches</td>
<td>in</td>
</tr>
<tr>
<td>m</td>
<td>meters</td>
<td>3.3</td>
<td>feet</td>
<td>ft</td>
</tr>
<tr>
<td>km</td>
<td>kilometers</td>
<td>1.1</td>
<td>yards</td>
<td>yd</td>
</tr>
</tbody>
</table>

### Area

<table>
<thead>
<tr>
<th>Symbol</th>
<th>When You Know</th>
<th>Multiply by</th>
<th>To Find</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>in²</td>
<td>square inches</td>
<td>6.4</td>
<td>square centimeters</td>
<td>cm²</td>
</tr>
<tr>
<td>ft²</td>
<td>square feet</td>
<td>0.09</td>
<td>square meters</td>
<td>m²</td>
</tr>
<tr>
<td>yd²</td>
<td>square yards</td>
<td>0.8</td>
<td>square meters</td>
<td>m²</td>
</tr>
<tr>
<td>m²</td>
<td>square miles</td>
<td>2.4</td>
<td>square kilometers</td>
<td>km²</td>
</tr>
<tr>
<td>ac</td>
<td>acres</td>
<td>0.4</td>
<td>hectares</td>
<td>ha</td>
</tr>
</tbody>
</table>

### Mass (weight)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>When You Know</th>
<th>Multiply by</th>
<th>To Find</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>oz</td>
<td>ounces</td>
<td>28</td>
<td>grams</td>
<td>g</td>
</tr>
<tr>
<td>lb</td>
<td>pounds</td>
<td>0.45</td>
<td>kilograms</td>
<td>kg</td>
</tr>
<tr>
<td>st</td>
<td>short tons</td>
<td>0.9</td>
<td>tonnes (1000 lb)</td>
<td>t</td>
</tr>
</tbody>
</table>

### Volume

<table>
<thead>
<tr>
<th>Symbol</th>
<th>When You Know</th>
<th>Multiply by</th>
<th>To Find</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>tsp</td>
<td>teaspoons</td>
<td>5</td>
<td>milliliters</td>
<td>ml</td>
</tr>
<tr>
<td>Tbsp</td>
<td>tablespoons</td>
<td>15</td>
<td>milliliters</td>
<td>ml</td>
</tr>
<tr>
<td>fl oz</td>
<td>fluid ounces</td>
<td>30</td>
<td>milliliters</td>
<td>ml</td>
</tr>
<tr>
<td>c</td>
<td>cups</td>
<td>0.24</td>
<td>liters</td>
<td>l</td>
</tr>
<tr>
<td>pt</td>
<td>pints</td>
<td>0.47</td>
<td>liters</td>
<td>l</td>
</tr>
<tr>
<td>qt</td>
<td>quarts</td>
<td>0.95</td>
<td>liters</td>
<td>l</td>
</tr>
<tr>
<td>gal</td>
<td>gallons</td>
<td>3.8</td>
<td>liters</td>
<td>l</td>
</tr>
<tr>
<td>ft³</td>
<td>cubic feet</td>
<td>0.03</td>
<td>cubic meters</td>
<td>m³</td>
</tr>
<tr>
<td>yd³</td>
<td>cubic yards</td>
<td>0.76</td>
<td>cubic meters</td>
<td>m³</td>
</tr>
</tbody>
</table>

### Temperature (exact)

<table>
<thead>
<tr>
<th>°F</th>
<th>Fahrenheit temperature</th>
<th>°C</th>
<th>Celsius temperature</th>
<th>°F (after subtracting 32)</th>
</tr>
</thead>
</table>

1 °F = 1.8 °C (approximately). For other exact conversions and more detailed tables, see NBS Misc. Pub. 286, Units of Weights and Measures, Price 12.50, SD Catalog No. C13-106-286.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 INTRODUCTION AND OVERVIEW</td>
<td>1</td>
</tr>
<tr>
<td>2.0 OVERVIEW OF TRACK MAINTENANCE SIMULATION COST</td>
<td>4</td>
</tr>
<tr>
<td>2.1 Track Maintenance as a Cost Simulation Problem</td>
<td>4</td>
</tr>
<tr>
<td>2.2 Simulation Technique</td>
<td>8</td>
</tr>
<tr>
<td>2.3 Maintenance Action Diagrams</td>
<td>10</td>
</tr>
<tr>
<td>2.4 Cost Model Limitations</td>
<td>13</td>
</tr>
<tr>
<td>2.5 Data for Model</td>
<td>14</td>
</tr>
<tr>
<td>2.6 Computer Program Structure</td>
<td>16</td>
</tr>
<tr>
<td>3.0 DATA STORAGE CATEGORIES FOR MODELING ANALYSIS</td>
<td>17</td>
</tr>
<tr>
<td>3.1 Maintenance Action Paths and Diagrammed Repair Blocks</td>
<td>17</td>
</tr>
<tr>
<td>3.2 Track Components and FRA Track Classifications</td>
<td>19</td>
</tr>
<tr>
<td>3.3 Cost Descriptor Codes</td>
<td>20</td>
</tr>
<tr>
<td>4.0 TRACK MAINTENANCE ACTION DIAGRAMS FOR COST MODELING</td>
<td>22</td>
</tr>
<tr>
<td>4.1 Maintenance Action Diagram Structuring</td>
<td>22</td>
</tr>
<tr>
<td>4.2 Action Diagramming of Track Inspections &amp; Accidents</td>
<td>25</td>
</tr>
<tr>
<td>4.3 Maintenance Decisions and Deferment Policies</td>
<td>28</td>
</tr>
<tr>
<td>4.4 Manufactured Supplies and Scrap Materials</td>
<td>30</td>
</tr>
<tr>
<td>4.5 Cascading and Reuse of Maintained Materials</td>
<td>30</td>
</tr>
<tr>
<td>5.0 TRACK MAINTENANCE COST DATA</td>
<td>34</td>
</tr>
<tr>
<td>5.1 Model Path Component Quantities</td>
<td>34</td>
</tr>
<tr>
<td>5.2 Cost Data Forms, Sources, and Types</td>
<td>36</td>
</tr>
<tr>
<td>5.3 Concept of System Component Qualities</td>
<td>37</td>
</tr>
<tr>
<td>5.4 Functional Parameters and Their Forms</td>
<td>40</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.0 SAMPLE TRACK MAINTENANCE SIMULATION COST MODEL RUNS</td>
<td>45</td>
</tr>
<tr>
<td>6.1 Background Information on Example Run #1</td>
<td>45</td>
</tr>
<tr>
<td>6.1.1 Inspections</td>
<td>45</td>
</tr>
<tr>
<td>6.1.2 Accident-Related Track Mileage Repaired</td>
<td>50</td>
</tr>
<tr>
<td>6.1.3 Maintenance Capacities of Repair Organization</td>
<td>50</td>
</tr>
<tr>
<td>6.2 Run Input and Output for Example #1</td>
<td>51</td>
</tr>
<tr>
<td>6.3 Sample Run Number 2 Using a Large Data Base File</td>
<td>52</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>56</td>
</tr>
<tr>
<td>APPENDIX A - Operations Manual</td>
<td>A-1</td>
</tr>
<tr>
<td>APPENDIX B - Program Maintenance Manual</td>
<td>B-1</td>
</tr>
<tr>
<td>APPENDIX C - Computer Program Flow Charts</td>
<td>C-1</td>
</tr>
<tr>
<td>APPENDIX D - Computer Program Listing</td>
<td>D-1</td>
</tr>
<tr>
<td>APPENDIX E - Program Subroutines</td>
<td>E-1</td>
</tr>
<tr>
<td>APPENDIX F - Computer Program Control and Operating Data File Structuring</td>
<td>F-1</td>
</tr>
<tr>
<td>APPENDIX G - Input Cost Data Example #1 (For Simplified Track Maintenance Action Diagram #1)</td>
<td>G-1</td>
</tr>
<tr>
<td>APPENDIX H - Input Data Example #2 (For Comprehensive Track Maintenance Action - Diagram #2)</td>
<td>H-1</td>
</tr>
<tr>
<td>APPENDIX I - Output Costs From Program Run Example #1</td>
<td>I-1</td>
</tr>
<tr>
<td>APPENDIX J - Output Costs From Program Run Example #2</td>
<td>J-1</td>
</tr>
<tr>
<td>APPENDIX K - Track Cost Model Maintenance Action Diagrams</td>
<td>K-1</td>
</tr>
<tr>
<td>APPENDIX L - Report of New Technology</td>
<td>L-1</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1-1</td>
<td>Computerized Track Maintenance Programs</td>
</tr>
<tr>
<td>2-1</td>
<td>Simulation Cost Model Information Handling</td>
</tr>
<tr>
<td>2-2</td>
<td>Time Simulation Representation Showing Various Actions Affecting the Track System Component Quality</td>
</tr>
<tr>
<td>2-3</td>
<td>Maintenance Action Diagram Concepts and Notations Used in Numbering Paths and Split Decision Points</td>
</tr>
<tr>
<td>2-4</td>
<td>Concepts and Notations Used by Simulation Cost Methodology in Maintenance Action Diagrams</td>
</tr>
<tr>
<td>3-1</td>
<td>Assigned Data Descriptor Codes</td>
</tr>
<tr>
<td>4-1</td>
<td>Simplified Maintenance Action Diagram of Track Used for Example Run Number One</td>
</tr>
<tr>
<td>4-2</td>
<td>Different Ways of Modeling Various Inspection Processes</td>
</tr>
<tr>
<td>4-3</td>
<td>Conceptual Layout of Simple Schematic Diagram Showing One Diagram for Each Class of Track</td>
</tr>
<tr>
<td>4-4</td>
<td>Details of a Maintenance Repair Block Drawn to Handle Simple Cascading of Rail</td>
</tr>
<tr>
<td>5-1</td>
<td>Sample Input Data Description</td>
</tr>
<tr>
<td>5-2</td>
<td>Component Quality as a Measure of System Condition</td>
</tr>
<tr>
<td>5-3</td>
<td>Portion of Sample Input Data for Use with Programmed Modifying Functions</td>
</tr>
<tr>
<td>5-4</td>
<td>Simulation of Modifications of Inspection Rate Multipliers and Resulting Rail Qualities and Simulation Costs Observed from Computer Run</td>
</tr>
<tr>
<td>5-5</td>
<td>Sample Input Data for Modifying Track Inspections With Time</td>
</tr>
<tr>
<td>6-1</td>
<td>Description of the Railroad Used for Simulation Purposes</td>
</tr>
<tr>
<td>6-2</td>
<td>Showing Functional Relations Used in the Simulation Demonstration</td>
</tr>
<tr>
<td>6-3</td>
<td>Two Examples of Non-Linear Inspection Policies Which Could Be Linked to Rail Quality</td>
</tr>
<tr>
<td>6-4</td>
<td>Summary Cost Model Results with and without Inspection Modifying Function</td>
</tr>
<tr>
<td>Figure</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>E-1</td>
<td>Func. Array Specifications by Column Number</td>
</tr>
<tr>
<td>E-2</td>
<td>Sample Application of a Single Function to Modify the Amount of Track from Track Accidents Being Maintained during Simulation Maintenance Action Diagram Example #1, Path 17</td>
</tr>
<tr>
<td>E-3</td>
<td>Different Functional Forms of Weibull Distribution Resulting from Numeric Adjustment of Shape Parameters A and B</td>
</tr>
<tr>
<td>F-1</td>
<td>Array Storage Structure for Path Data by Array Subscript Number</td>
</tr>
<tr>
<td>G-1</td>
<td>Layout of Data Card or Line File Entry Format</td>
</tr>
<tr>
<td>G-2</td>
<td>Assigned Data Descriptors 20-49 and Their Units of Measure for Coding</td>
</tr>
<tr>
<td>K-1</td>
<td>Track in Use, Schematic Diagram Number 0001</td>
</tr>
<tr>
<td>K-2</td>
<td>Track in Use</td>
</tr>
<tr>
<td>K-3</td>
<td>Repair Operation</td>
</tr>
<tr>
<td>K-4</td>
<td>Repair Operation 1</td>
</tr>
<tr>
<td>K-5</td>
<td>Repair Operation 2</td>
</tr>
<tr>
<td>K-6</td>
<td>Repair Operation 3</td>
</tr>
<tr>
<td>K-7</td>
<td>Repair Operation 4</td>
</tr>
<tr>
<td>K-8</td>
<td>Repair Operation 5</td>
</tr>
<tr>
<td>K-9</td>
<td>Repair Operation 6</td>
</tr>
<tr>
<td>K-10</td>
<td>Repair Operation 7</td>
</tr>
<tr>
<td>K-11</td>
<td>Repair Operation 8</td>
</tr>
<tr>
<td>K-12</td>
<td>Repair Operation 9</td>
</tr>
<tr>
<td>K-13</td>
<td>Repair Operation 10</td>
</tr>
<tr>
<td>K-14</td>
<td>Repair Operation 11</td>
</tr>
<tr>
<td>K-15</td>
<td>Repair Operation 12</td>
</tr>
</tbody>
</table>
## LIST OF ILLUSTRATIONS

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-16</td>
<td>Repair Operation 13</td>
<td>K-17</td>
</tr>
<tr>
<td>K-17</td>
<td>Repair Operation 14</td>
<td>K-18</td>
</tr>
<tr>
<td>K-18</td>
<td>Repair Operation 15</td>
<td>K-19</td>
</tr>
<tr>
<td>K-19</td>
<td>Repair Operation 16</td>
<td>K-20</td>
</tr>
<tr>
<td>K-20</td>
<td>Repair Operation 17</td>
<td>K-21</td>
</tr>
<tr>
<td>K-21</td>
<td>Repair Operation 18</td>
<td>K-22</td>
</tr>
<tr>
<td>K-22</td>
<td>Repair Operation 19</td>
<td>K-23</td>
</tr>
<tr>
<td>K-23</td>
<td>Repair Operation 20</td>
<td>K-24</td>
</tr>
</tbody>
</table>
1.0 INTRODUCTION AND OVERVIEW

The actual total cost of maintaining a segment of railroad track is extremely difficult to ascertain. At the point of manufacture a length of rail, a tie, or a ton of ballast has, at a given instant, a fixed determinable cost. But materials of repair are only a fraction of the overall maintenance expenses.

What about delivery costs, installation effort, and the resulting costs of delaying traffic if the process doesn't proceed on schedule? What about next year's cost? Or a more pertinent question:

WHAT WOULD BE THE COST IMPACT OF SOME PROPOSED MAINTENANCE OPERATION WHICH WOULD ALTER THE STANDARD MAINTENANCE PRACTICE NOW IN USE?

Alterations in a railroad's operating or maintenance procedure can often lead to undesirable economic situations. For example, the conversion to systemwide use of 100-ton hopper cars has increased the rate at which rails sustain fatigue damage.

The task of costing any proposed maintenance practice or policy modification is complicated by:

- The fact that there are almost as many maintenance practices as there are railroads to perform them;
- The constant replacement of system materials;
- The unplanned nature of some maintenance operations; and
- The everchanging demands of business which can alter the physical loading of the track structure themselves.

These problems have prompted several attempts in recent years to get a better grip on what it costs to maintain track. In fact, several of these approaches to track maintenance costing were reviewed recently at a workshop* on the subject of track maintenance. Figure 1-1 is a summary of some of the types of computerized techniques being used today for evaluating track maintenance costs or performance.

*Track Maintenance Planning Workshop, Penn State University, June 1980.
<table>
<thead>
<tr>
<th>ORGANIZATION</th>
<th>PROGRAM USE</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Canadian National Railways</td>
<td>Manpower Planning</td>
<td>Data of past maintenance efforts are converted through regression procedures to forecasting formulas for estimating manpower needs by geographic location in the Canadian National Railway System.</td>
</tr>
<tr>
<td>P.A. International</td>
<td>Job Task Planning</td>
<td>Continuous tabulation and updating of projected work for periods of one to four weeks. Identifies through listings the need for people and equipment by regions.</td>
</tr>
<tr>
<td>Conrail</td>
<td>Track Maintenance Management System</td>
<td>Keeps tabs on manhours by maintenance operation during past year. Data from sample 9 miles of track have been kept in detail on a trial basis.</td>
</tr>
<tr>
<td>Southern Railway</td>
<td>Maintenance Planning from track quality indices</td>
<td>Track geometry parameters are measured and correlated with track performance (derailments, rail defects, train delays) and track is identified where scheduled maintenance can be economically justified.</td>
</tr>
<tr>
<td>Bessemer and Lake Erie Railroad</td>
<td>Organization of maintenance operations data</td>
<td>Tabulates and records for future reference track repair histories under 110 different accounting codes.</td>
</tr>
<tr>
<td>Pugh Roberts</td>
<td>Analysis of track related safety options</td>
<td>Uses a collection of mathematical equations that describe the various inter-relationships among the safety operational elements of the railroad.</td>
</tr>
<tr>
<td>Shaker Research</td>
<td>Simulates maintenance actions and tabulate cost of track repair</td>
<td>Allows user to simulate his own maintenance practices and obtain cost accounting outputs with time. Provides for cost comparison simulations under user defined maintenance practice modifications.</td>
</tr>
</tbody>
</table>

FIGURE 1-1. COMPUTERIZED TRACK MAINTENANCE PROGRAMS
This guide presents a methodology for computer simulation of track maintenance procedures and costs. The methodology and FORTRAN IV computer program contained herein (Appendix D) provide for:

- Graphic representation of the system being simulated through the use of a maintenance action diagram;
- Costing proposed alternate maintenance action policies or procedures;
- Listing all annual track maintenance costs; and
- Maintenance cost outputs at any desired instant during the time simulation.

This guide has three purposes. These are to present the simulation costing methodology for track maintenance, to cover the details of the computer program developed, and to give examples of its use.

Section 2 of the report is concerned with introducing the methodology. An introduction is given to the use of maintenance action diagrams for setting up a simulation in graphic form. Also discussed are the types of data needed to perform a simulation as well as costs that are handled (and not handled) by the program in its present state.

Section 3 presents details of the specific data categories under which the program holds cost information.

Section 4 deals with the details of representing maintenance actions in graphic form. Sample maintenance diagrams are covered and alterations of diagrams for costing different maintenance operations and schemes are shown.

Sections 5 and 6 are included in order to give data entry examples and to provide the reader with two example simulations from two separate modeled track maintenance systems. The Appendices contain the computer program, flow charts, cost data examples for test runs, and tabulated output results.
2.0 OVERVIEW OF TRACK MAINTENANCE SIMULATION COST METHODOLOGY

2.1 Track Maintenance as a Cost Simulation Problem

Track maintenance costs can be defined as the annual direct charges for the upkeep of track property and equipment. Track property includes rail, cross ties, ballast, and associated hardware fastenings. Equipment refers to those pieces of machinery used for inspecting, installing, and keeping the track in satisfactory operating condition. In early stages of the work for this contract it became apparent that there are some commonly identifiable aspects to these costs and the associated maintenance operations which are performed by every railroad. It also became apparent that there are as many different ways to perform some of the major tasks of track repair as there are track systems in use. Each railroad has its own maintenance policy which has usually been derived from many years of practical repair operations experience.

In developing a methodology for determining track maintenance costs it was desired that:

- the technique be applicable to a variety of railroads' maintenance of way operations,
- the procedure allow for cost comparison of alternate maintenance operations, either real or imaginary in concept,
- the procedure allow for evaluation of the cost impact of proposed safety standards.

The approach taken in the work centers around a general computerized simulation procedure which can handle any user defined maintenance system. The program uses a standard fourth order iteration technique and a time stepping procedure to simulate defined maintenance actions. As a result, non-linear costing information can be addressed by the procedure. Figure 2-1 is a display which represents the cost information handling of the computer technique developed.
Track Maintenance Cost Inputs From

- Accounting Records
- Policy Definitions
- Business Practices
- Accidents
- Component Failures
- Maintenance Actions
- Safety Compliance
- Component Life Expectancies

Computer Model

Maps Input Data to Operating Arrays

- Performs Desired Cost Simulation
- User Can Control
  - Maintenance System Definition
  - Data File Values
  - Component Defect/Failure Rates
  - Cost Output Forms
  - Frequency of Output
  - Simulation Timing

Provides Tailored Cost Outputs

Track Maintenance Cost Outputs For

- Any Time Interval
- FRA Track Class
- Track Component
- Labor Categories
- Material (New/Scrap)
- Handling Expenses
- Delays/Deferments
- Fines
- Inspection Resources

FIGURE 2-1. SIMULATION COST MODEL INFORMATION HANDLING
In any modeling procedure, one must agree in advance to what the model will represent and include. Determining which costs to include as maintenance costs in the modeling process is not simply resolved. Defining what "is" or "is not" a track maintenance cost is often ambiguous and should be decided carefully in advance. An attempt has been made in the present study to define a manageable set of cost input data that will provide a useful set of output information about the modeled track maintenance system.

In the model developed the user is able to include or exclude whatever costs he wants. The present limitations of the simulation cost model are determined in part by the cost information or data available to the program. If a cost item is to be included in the model there must be some data to describe the maintenance expense.

For example purposes, the present technique of track maintenance modeling described in this report has been set up to allow at least for those costs listed below.

**Maintenance Cost Items Included in Model Reported**

- Labor on repairs of five track structural components. For example purposes the track was defined to consist of the three major substructures or components (referred to by numbers 1 through 5 in computer outputs):
  - rail
    - 1 - jointed
    - 2 - welded
  - cross ties
    - 3 - wood
    - 4 - concrete
  - ballast
    - 5 - ballast
- Present-day material costs of these new components installed during repair.
- Cost to keep electrical isolation and/or contact throughout rail structure.
- Cost of track inspection labor and necessary equipment used in the inspection process.
- Supervisory labor needed in monitoring subcontracted maintenance operations.
. Fines accrued for not keeping the track components within safe standards.
. Delays of trains caused directly by the maintenance repair crews.
. Delays from slow orders imposed as a result of insufficient maintenance upkeep.
. Return costs from scrapping components of system.
. Subcontracted maintenance of the five basic components.
. Delivery costs of transporting new (renewed) components to the repair sites.
. Travel and living associated with getting to and from the repair location if paid by the railroad.
. Cost of heavy equipment needed to perform repairs on basic track structure.
. Cost of fueling or powering certain pieces of equipment such as welders, etc.
. Accident costs.
. Rework or refurbishment costs (for example, conversion of used jointed rail to welded rail).
. Track idle or closure costs.

Simulation cost modeling can address questions of:

- How much does it cost to maintain track?
- What are the cost breakdowns within the maintenance system by defined structural component (i.e., rail, cross ties, or ballast)?
- Where in the maintenance system are the most costly procedures?
- What savings in annual maintenance can be expected if certain operations or policies are altered?
- What will be future maintenance costs if the system work volume or procedures are changed?

Maintenance costs are available for output at any time point in the simulation process. This time frozen "snap-shot" of maintenance repair costs and system rates of repair can be used to compare with past or present accounting records of the railroad system being modeled.
Single simulation time point costs can be printed and broken out into the following categories:

1. Simulation maintenance action diagram path number.
2. Track component (i.e., rail, tie, ballast, etc.).
3. FRA track class (1-6).
4. Major repair block or type (such as rail laying, surfacing, etc. For example purposes, 20 separate types were identified.)
5. Cost code (for the present example consists of about 15 codes including):
   a. Material costs
   b. Equipment costs
   c. Fines
   d. Delays
   e. Scrap return costs
   f. Contracted costs
   g. Delivery costs
   h. Travel and living
   i. Six levels of labor

2.2 Simulation Technique

The Simulation Cost Modeling (SCM) technique consists of a qualitative and quantitative representation of the maintenance actions associated with the track system being modeled. The qualitative representation consists of a maintenance action diagram which is usually constructed by the user and describes the maintenance actions and their relationship to one another. The quantitative representation is the data set associated with the action diagram. Linking the two representations is the computer program. This program quantifies the modeled actions shown in the diagram, implements the associated data set, and provides a selected set of cost outputs.

The computerized time stepping routine used in the simulation of maintenance actions is a fourth order iteration procedure. Being a rate of change structured technique this algorithm takes into account the rate at which the various track components are replaced and the rate at which they naturally generate observable defects requiring maintenance work to be performed.

Figure 2-2 shows this time stepping action and information processing scheme for keeping tabs on the condition or quality of the various track components.
FIGURE 2-2. TIME SIMULATION REPRESENTATION SHOWING VARIOUS ACTIONS AFFECTING THE TRACK SYSTEM COMPONENT QUALITY
2.3 Maintenance Action Diagrams

The representation of the maintenance system takes the form of a pictorial action diagram which describes how the railroad maintains its track. The maintenance action diagram is analogous to a diagram of a water pipe network. Water systems can be represented by drawings which show the various connecting or branching points as well as their distribution pattern. In a similar fashion the cost model maintenance action diagram displays the system being represented. Each of these diagrams has inherent characteristics which must be understood in advance in order to be used. The piping diagram, for example, through a developed notational convention can reveal pipe lengths, sizes used, elevation of layout, valves, and many other features.

The simulation cost model action diagram also carries with it certain special meanings which are discussed here in order that the reader can gain a more thorough understanding of the details in this pictorial view of the maintenance system. Figures 2-3 and 2-4 display some of the shorthand notation used in maintenance action diagramming. All maintenance action diagrams contain paths or lines with arrowheads which are used for showing the association between various maintenance actions or stand for a maintenance action itself. This shorthand method of displaying the maintenance system pictorially is in one sense convenient and in another misleading. It is convenient in that the overall system may be viewed in a glance, and if the notation is understood, can give an impression of the complexity of its structure. This method is misleading from the standpoint that many of the details of the system cannot be contained fully in the display, and may lead the viewer to believe that the model is too simple.

Figure 2-3 shows the numbering technique used in diagramming. The path numbers given on the diagram are used for reference in the computer handling of the cost information input and output. Maintenance costs by path number can be selected by the user.

Diagram numbered circles represent maintenance decision points. These decisions define alternate routes whereby two separate actions are defined
Paths are used to represent maintenance "actions" or simply to act as connecting links that show action sequencing and hierarchy.

Each path may have many non-zero scalars associated with its action. These scalars define:
- Components that are represented by this path
- Track classes that are represented by this path
- Amount of component on path or amount of component which is being transferred by path
- Rate at which maintenance action is performed
- Labor rate costing of maintenance work
- Quality of components to which work is applied
- Functional operators controlling user defined relationships which can be used to modify system scalars

Numbered circles represent system maintenance decision points that affect downstream actions.

These numbered split points can also represent alternate maintenance policy actions.

This path represents the balance of the components from path "1" not transferred downstream by path "2".

This path represents that portion of the components associated with path "1" transferred downstream to another point in the maintenance system.

FIGURE 2-3. MAINTENANCE ACTION DIAGRAM CONCEPTS AND NOTATIONS USED IN NUMBERING PATHS AND SPLIT DECISION POINTS
Path represents amount of track reviewed by inspectors for potential defects

Path represents track approved for use by inspection processes

Point 1 represents decision of changing track operating status. Apply "slow order" or other operating restriction.

Path represents track with operating restrictions and under consideration for maintenance.

Point 2 represents decision to replace defective materials such as rail or ties from track being maintained

Path and triangle represent action of scrapping or removal of material such as rail or ties from system

Path and triangle represent action of returning track to normal operating status

Path represents action of doing maintenance work on track

Point represents combining of track under repair with new component materials needed for repair or maintenance

Path and triangle represent entry of new component material from manufacturer to point of maintenance

FIGURE 2-4. CONCEPTS AND NOTATIONS USED BY SIMULATION COST METHODOLOGY IN MAINTENANCE ACTION DIAGRAMS
on the track that is conceptually sent beyond this point in the maintenance system. In addition, the maintenance action diagram often has other notational characteristics needed for representing track maintenance. Some of these features are shown in Figure 2-4 and include:

- The "in use" or "in service" portion of the track system,
- The inspection-of-track loop,
- The removal or scrapping of defective system components,
- The act of maintaining the defective track found from a single inspection process, and
- The supplying or bringing of new components to replace those being scrapped from the system.

2.4 Cost Model Limitations

The present maintenance cost model is constrained mainly by the information or data available to be supplied to the program. One could find he does not have sufficient time or resources to collect and enter the data required for a maintenance system whose scope is too inclusive. An attempt has been made in the present study to limit the scope of the required input information while at the same time providing a technique that will yield a useful set of output information about the modeled track maintenance system.

In limiting the scope of the examples used for explaining the model, certain costs have been excluded from the costing procedure. In general, the effects of most of the excluded costs can be obtained, if needed, by operating on the output costs obtained from the simulation cost model. An example would be costs of inflation. Once costs of maintenance have been obtained from the model an inflationary factor might be used to establish final costs including inflation effects.

Although they could be handled by the model, the following cost categories have not been explicitly provided for in the present simulation cost model.
of the track maintenance system,

- Taxes
- Cost of money, interest
- Cost of inflation
- Costs of equipment storage such as housing and land
- Cost of bridge maintenance
- Cost of signal excitation and transmitting electronics and housing for such
- Failure sensing and detection equipment
- Opportunity costs
- Penalties paid to customers for not delivering goods as a result of track condition or track maintenance practices
- Business cost to customers for not delivering goods on time as a result of shipping delays due to track condition or track maintenance practices
- Loss of business due to unreliable rail service to customers as a result of poor track maintenance.

2.5 Data for Model

The data base required for running a useful cost model is typically very large. Since a large effort in time is needed in collecting data, it was decided that the data coding should:

1. Provide the model user with an annotated description of the data coded;
2. Allow the user to update the data quickly as new (or more reliable) information about the system becomes available; and
3. Let the user add (delete) associations between data elements without restructuring the whole body of data already established.

Data input structuring for use by the computer program are explained in Section 5. Examples of data entry and formats are included in Appendices G and H.
Data here are loosely defined as those numbers needed by the computer program to successfully simulate the maintenance system being analyzed. In general, the data required fall into six separate categories:

1. Computer program control numbers.
2. Cost and performance rates for every maintenance action diagrammed.
3. Initial annual amounts of components handled on each path of the action diagram.
4. Condition (quality) of each component at the start of the simulation.
5. Defect generation rates for each component being handled.
6. Formulas which control maintenance actions.

The computer control numbers are described in detail in Appendix F. Examples for their entry are given also in Appendix F as well as in Appendix G. Control numbers are used by the program to sort incoming data, define the simulation procedure, and set the type of cost outputs, as well as their frequency during the execution of the program.

Control inputs to data files named "IFILE", "RFILE", and "INODE", allow the user to define and run a simulation for any maintenance action diagram as long as it is a closed network of paths similar to those depicted schematically in Figures 2-3 and 2-4. Two examples of these kinds of maintenance action networks are drawn out in detail in Appendix K. The "IFILE" data set defines for the computer the size and complexity of the maintenance network. The "RFILE" contains real numbers used in the computational cycle, the time length of the simulation, and frequency during the cycle of printing the cost outputs. The information contained in "INODE" defines explicitly the path connections of the maintenance action diagram used in the simulation. This file also contains (see Appendix F) path type codes for describing the various allowable path linkage forms that may be desired in general maintenance simulations.

Costs, items 2 through 4 above (including initial component amounts
and qualities), are all contained in the path by path cost data entries coded as shown in Appendix G. These data initialize maintenance action diagram path information and contain a commentary entry on the right side. Insertion and/or deletion on a line by line (path by path of the maintenance diagram) basis is allowed for and is convenient for updating the cost data should new information become available.

Track structures such as rail and ties tend to develop defects as they age. These defect generation rates are formed by the program from the Weibull distribution and the data supplied to the "WEIBL" file. This distribution is very general and is well suited for many engineering components which have finite wear-out lives. The attributes of this distribution and its potential for representing cyclic failure modes are discussed in References 9 and 11. Details of data entries to "WEIBL" are included in Appendix F.

Formula for functional relationships of maintenance actions and/or costs can be entered by the user to modify a programmed simulation. In general, any formula can be written for altering costs and/or path elements. For user convenience several families of curves have been programmed which can be used for this purpose. Entry of special parameter values to the "DATA" file will provide many functional operators for the user. Examples of functional use are discussed in Sections 5 and 6 as well as in Appendix E (see explanation of "FUNCTS" subroutine).

2.6 Computer Program Structure

The simulation program designed for implementing the costing methodology is contained in Appendix D. The listing is over 2500 lines in length and has been broken down into 36 separate subroutines. The full subroutine list by name and file number and in alphabetical order is given in Appendix E.

The program is written in standard FORTRAN IV language. The present version has been set up to operate in the batch mode with all run options being selected before running through the control data files "TFILE" and "RFILE". For two example inputs, see Appendices G and H.
3.0 DATA STORAGE CATEGORIES FOR MODELING ANALYSIS

Each path of the maintenance action diagram is intended to portray some element of repair work or service action. In order for the costing information to be of use the analysis must provide for the tabulation of expenses under several different headings.

The computerized cost data are therefore kept in the analysis under separate categories. These categories are, in general, expandable by the user (see Appendices F, G and H). The subcategories used in the model examples discussed here include maintenance expenses by:

- Maintenance Action Diagram Path Number
- Major Repair Block Operation (usually a collection of paths)
- Track Structural Component
- FRA Track Classification
- User Identified Cost Descriptor Codes

3.1 Maintenance Action Paths and Diagrammed Repair Blocks

Cost data for the simulation are entered and keyed to the paths of the maintenance action diagram drawn for the simulation. For convenience the paths of the maintenance action diagram are assigned a number when the drawing is constructed. These paths numbers can be assigned in any way by the user who wishes to perform a simulation. It must be remembered, however, that the program stores the input data in matrix form with one of the matrix subscripts being comprised of the path numbers labeled on the action diagram. Thus, in order to conserve computer space (required matrix size), the path numbering should be sequential starting from one. If path numbering is not sequential (often convenient for inserting paths in the diagram at a future time), this will simply increase the program running time.

In drawing the maintenance action diagram it is convenient to encircle or block out regions of the diagram which represent a single repair operation concept. In this way, the total expenses of this "block" or repair operation might be followed. For an example, see the diagram included in Appendix K. In the second example run for the present study, cost summaries were obtained for twenty separate repair or "block" operations (see Appendix J). The
twenty repair (maintenance) operation "blocks" were numbered and included:

1. Laying New Welded Rail (Method 1)
2. Laying New Welded Rail (Method 2)
3. Laying New Welded Rail (same wt. welded)
4. Rail Change Out (jointed)
5. Transpose Rail
6. Transpose Rail (dummy rail method)
7. Track Panel Installation
8. Surfacing
9. Smoothing
10. Timber and Surfacing
11. Tie Renewal
12. Joint Repairs
13. Turn-out Repairs
14. Rail Buildup
15. Rail Grind
16. Ballast Cleaning
17. Brush and Weed Control
18. Ditching
19. Snow/Ice Removal
20. Slides/Washouts

These identified track maintenance repair operations need not remain fixed, but can be modified as required by the user. By expanding the number of paths of the diagram a larger or different set of maintenance repair operations could be handled. Also, if a given track system does not use one or more of the included maintenance practices, then the path data for those operations could simply be zero.

With this process of labeling groups of operations, two or more similar maintenance operations could be compared by diagramming each operation, supplying the separate cost data, and rating the separately tabulated expenses on the outputs from the model. This was done, for example, in the laying of welded rail (methods 1 and 2 above. See output examples of Appendix J.
3.2 Track Components and FRA Track Classifications

For purposes of the examples in this guide, it has been assumed that the track system is comprised of five basic components upon which maintenance repair operations are performed. These five components are:

<table>
<thead>
<tr>
<th>Major Substructure</th>
<th>Computerized Component Number</th>
<th>Description</th>
<th>Units Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rail</td>
<td>1</td>
<td>Jointed</td>
<td>Miles</td>
</tr>
<tr>
<td>Rail</td>
<td>2</td>
<td>Welded</td>
<td>Miles</td>
</tr>
<tr>
<td>Cross Tie</td>
<td>3</td>
<td>Wood</td>
<td>Miles</td>
</tr>
<tr>
<td>Cross Tie</td>
<td>4</td>
<td>Concrete</td>
<td>Miles</td>
</tr>
<tr>
<td>Ballast</td>
<td>5</td>
<td>Ballast</td>
<td>Miles</td>
</tr>
</tbody>
</table>

Therefore, the model provides for storing or tabulating data under categories for any one of the above components for each maintenance action path diagrammed.

Furthermore, maintenance actions may be performed on various classes of track. Since different amounts and degrees of maintenance may be needed for each of the different FRA track classifications, cost coding of the present methodology provides for associating expenses with any one of the presently required FRA classifications. Classification of track is primarily by speed of operation where:

<table>
<thead>
<tr>
<th>Class</th>
<th>Operating Speed for Freight Trains is</th>
<th>Operating Speed for Passenger Trains is</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 track</td>
<td>10 m.p.h.</td>
<td>15 m.p.h.</td>
</tr>
<tr>
<td>Class 2 track</td>
<td>25 m.p.h.</td>
<td>30 m.p.h.</td>
</tr>
<tr>
<td>Class 3 track</td>
<td>40 m.p.h.</td>
<td>60 m.p.h.</td>
</tr>
<tr>
<td>Class 4 track</td>
<td>60 m.p.h.</td>
<td>80 m.p.h.</td>
</tr>
<tr>
<td>Class 5 track</td>
<td>80 m.p.h.</td>
<td>90 m.p.h.</td>
</tr>
<tr>
<td>Class 6 track</td>
<td>110 m.p.h.</td>
<td>110 m.p.h.</td>
</tr>
</tbody>
</table>

However, several tolerance limits of track geometry must also be held by FRA class (see Reference 10). If data are not available by track class or
require other components to be handled, this can be done. By changing two numbers in the "IFILE" data set, the number of track components and track classifications handled by the program can be altered. Dimensioned program arrays may have to be enlarged if more than the above data subcategories are needed for a given cost simulation. For resizing of program arrays, see Appendix F on file structuring.

3.3 Cost Descriptor Codes

Since maintenance expenses are often tabulated in many accounting practices under such headings as materials, labor and equipment, the simulation methodology includes a set of user definable cost data descriptor codes. See codes 30 through 49 of Figure 3-1. Every cost item entered to the program is referenced to these descriptor codes (user account numbers). Summary output costs are then categorized (if desired and selected) under these user defined cost codes (see outputs in Appendices I and J).
<table>
<thead>
<tr>
<th>COMPONENTS</th>
<th>00</th>
<th>01</th>
<th>02</th>
<th>03</th>
<th>04</th>
<th>05</th>
<th>06</th>
<th>07</th>
<th>08</th>
<th>09</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jointed Rail</td>
<td>00</td>
<td></td>
<td>02</td>
<td></td>
<td>03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Welded Rail</td>
<td>01</td>
<td></td>
<td></td>
<td></td>
<td>04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wood Ties</td>
<td>02</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>05</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Concrete Ties</td>
<td>03</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>06</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ballast</td>
<td>04</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>07</td>
<td></td>
</tr>
<tr>
<td></td>
<td>05</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>08</td>
</tr>
<tr>
<td></td>
<td>06</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>09</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TRACK CLASS</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0</td>
<td></td>
<td></td>
<td>10</td>
<td></td>
<td></td>
<td>11</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 1</td>
<td></td>
<td>11</td>
<td></td>
<td></td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Class 2</td>
<td></td>
<td></td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>13</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 3</td>
<td></td>
<td></td>
<td></td>
<td>13</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14</td>
</tr>
<tr>
<td>Class 4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>14</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Class 6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>16</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PATH INFO</th>
<th>20</th>
<th>21</th>
<th>22</th>
<th>23</th>
<th>24</th>
<th>25</th>
<th>26</th>
<th>27</th>
<th>28</th>
<th>29</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantity of Component</td>
<td>20</td>
<td></td>
<td>21</td>
<td></td>
<td>22</td>
<td></td>
<td>23</td>
<td></td>
<td>24</td>
<td></td>
</tr>
<tr>
<td>Quality of Component</td>
<td></td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td>21</td>
<td></td>
<td></td>
<td></td>
<td>22</td>
</tr>
<tr>
<td>Amount of Bad Components</td>
<td></td>
<td></td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td>21</td>
<td></td>
<td></td>
<td>22</td>
</tr>
<tr>
<td>Amount of Good Compnt</td>
<td></td>
<td></td>
<td></td>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td>21</td>
<td></td>
<td>23</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>COST CATEGORIES</th>
<th>30</th>
<th>31</th>
<th>32</th>
<th>33</th>
<th>34</th>
<th>35</th>
<th>36</th>
<th>37</th>
<th>38</th>
<th>39</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Quality in Costing</td>
<td>30</td>
<td></td>
<td>31</td>
<td></td>
<td>32</td>
<td></td>
<td>33</td>
<td></td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>Material Costs</td>
<td></td>
<td>30</td>
<td></td>
<td>31</td>
<td></td>
<td>32</td>
<td></td>
<td>33</td>
<td></td>
<td>34</td>
</tr>
<tr>
<td>Equipment Costs</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>Fines</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>Delay Costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td>33</td>
</tr>
<tr>
<td>Scrap Return Costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
<td></td>
<td></td>
<td>34</td>
</tr>
<tr>
<td>Contracted Costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
<td></td>
<td>35</td>
</tr>
<tr>
<td>Delivery Costs</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
<td>36</td>
</tr>
<tr>
<td>Travel and Living</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MISC. CODING</th>
<th>40</th>
<th>41</th>
<th>42</th>
<th>43</th>
<th>44</th>
<th>45</th>
<th>46</th>
<th>47</th>
<th>48</th>
<th>49</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trackmen Labor</td>
<td>40</td>
<td></td>
<td>41</td>
<td></td>
<td>42</td>
<td></td>
<td>43</td>
<td></td>
<td>44</td>
<td></td>
</tr>
<tr>
<td>Mechanic Labor</td>
<td></td>
<td>40</td>
<td></td>
<td>41</td>
<td></td>
<td></td>
<td>42</td>
<td></td>
<td>43</td>
<td></td>
</tr>
<tr>
<td>Machine Operator Labor 1</td>
<td></td>
<td></td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42</td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>Machine Operator Labor 2</td>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42</td>
<td>44</td>
</tr>
<tr>
<td>Foremen Labor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td></td>
<td></td>
<td>41</td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>Supervisor Labor</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40</td>
<td></td>
<td></td>
<td>41</td>
<td>43</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ISOLATED PARAMETERS</th>
<th>50</th>
<th>51</th>
<th>52</th>
<th>53</th>
<th>54</th>
<th>55</th>
<th>56</th>
<th>57</th>
<th>58</th>
<th>59</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td>52</td>
<td></td>
<td>53</td>
<td></td>
<td>54</td>
<td></td>
</tr>
<tr>
<td>Federal Safety Standards</td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td></td>
<td>52</td>
<td></td>
<td></td>
<td>53</td>
</tr>
<tr>
<td>Speed</td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td></td>
<td>52</td>
<td></td>
<td>53</td>
</tr>
<tr>
<td>Load</td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td></td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Mgt.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td>52</td>
<td></td>
</tr>
<tr>
<td>Accidents</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td>52</td>
</tr>
<tr>
<td>Weibull</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td>53</td>
</tr>
<tr>
<td>Shipping Rates</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>Income</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>55</td>
</tr>
<tr>
<td>Design Factors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>56</td>
</tr>
<tr>
<td>Track Gradient Factors</td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td></td>
<td>52</td>
<td></td>
<td>53</td>
</tr>
<tr>
<td>Track Curvature Factors</td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td>52</td>
<td></td>
<td>53</td>
<td></td>
</tr>
<tr>
<td>Weather Factors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>51</td>
<td>54</td>
</tr>
<tr>
<td>Crossing Factors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td></td>
<td>55</td>
</tr>
<tr>
<td>Switch Factors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>56</td>
<td>57</td>
</tr>
<tr>
<td>Available Manpower</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
<td>58</td>
</tr>
</tbody>
</table>

**FIGURE 3-1. ASSIGNED DATA DESCRIPTOR CODES**
4.0 TRACK MAINTENANCE ACTION DIAGRAMS FOR COST MODELING

4.1 Maintenance Action Diagram Structuring

In simulating maintenance costs for the model, the user represents pictorially the way in which a railroad (or the industry as a whole) maintains its track. This schematic diagram consists of a set of maintenance actions which comprise those performed by the typical track maintenance of way department. Furthermore, these actions are linked with connecting paths (or lines) which show with arrows the order in which these actions are carried out. In addition to representing the order in which the actions are done, the diagram also shows the exact order in which the computer carries out its cost computations. The simulation cost modeling procedure calculates the cost for each individual maintenance action represented and thus the total cost to maintain the track system under consideration. Refer to References 12 and 13 for diagrams of other systems.

Track maintenance-related actions chosen for an example simulation are:

- Inspections and other processes which result in a need for maintenance repair work,
- Decisions to carry out maintenance work,
- Maintenance repair operations,
- Maintenance deferments,
- Manufacturing supply of new materials (for instance rail, cross ties, etc.)
- Renew or rework shop operations, and
- Material scrapping.

Each of these maintenance-related actions involves a complex network of tasks performed by several people of a typical maintenance of way staff. The present simulation methodology involves establishing the maintenance policies and/or procedures as they are actually done and in terms of how much (or at what rate) defined work is performed. The various maintenance actions modeled are defined explicitly by the labels on the action diagram; whereas, the rates of doing the work are defined by the data and functions associated with the paths.
Therefore, in order to complete the costing analysis it is essential that cost data for each repair action modeled be identified and available for input. Layed out in diagrammed form the above listed maintenance actions could look as shown in Figure 4-1. This diagram contains the essential features of the maintenance actions listed above. This figure and the more comprehensive one shown in Appendix K are presented for the purpose of demonstrating the simulation costing methodology. Each maintenance action diagram contains numbered paths which connect various points of the action diagram to one another. Each line (path) either represents an element of maintenance repair work or just links two or more actions to one another. During the simulation the calculations proceed as depicted in the diagram in that upstream paths are processed first.

Circled junctions (or nodes) of three connecting paths indicate or represent various decision making processes in the maintenance-related work. For example, circled junction 3 in Figure 4-1 represents an inspector's decision as to whether he accepts the track he has inspected (path 1) and allows its continued use (path 2) or labels it as being in need of repair (path 18) by placing a "slow order" or other restriction on its use.

Junction number 5 in Figure 4-1 represents the decision making process whereby all track identified as being in need of work (path 3) is either passed to the point in the system where it is maintained (path 5) or where work is deferred (path 4).

Areas of the diagram which are shaded represent major repair operations which usually require several paths for their description. The maintenance repair block shown in Figure 4-1 has two incoming and three outgoing paths which pass its boundaries. The connections of these paths are shown and merely represent the overall nature of performing work on track components which have renew (or rework) shops, manufacturing, and scrap yards associated with them.

In Figure 4-1 junctions 6, 7, and 8 represent repair actions of adding new parts to the track system and removing worn or degraded defective parts. Paths 6, 7, 8 and 10 are used to keep tabs on the various rates at which
FIGURE 4-1. SIMPLIFIED MAINTENANCE ACTION DIAGRAM OF TRACK USED FOR EXAMPLE RUN NUMBER ONE
the following maintenance costs are expended:

Path 6  - New parts for materials installed
Path 7  - Labor, equipment, and other costs
Path 8  - Costs of converting degraded materials to reusable ones
  (for example, making old jointed rail into continuous welded rail)
Path 10 - Scrap return costs from materials removed from use.

Setting out in block form the various types of repair work has several advantages in simulating maintenance costs. One advantage is that each repair block may be as complex or simple in construction as necessary and the internal details of the blocked actions might not be shown at all. Examples might include remote rework shops where components are sent for repair. If the repair rates and costs are known the details of the maintenance work in the facility might be skipped and the applied costs be attached only to the maintenance actions as one lump sum or cost per component repaired. This blocking procedure allows the major maintenance actions to be initially simulated and then updated in more detail at a later time when more detail of the repair operation is learned and/or desired.

4.2 Action Diagramming of Track Inspections and Accidents

The need for maintenance repairs on a given section of track can be generated through inspection reviews of the track or through accidentally occurring incidents. The upkeep and maintenance of track which has been torn up from accidents, washouts, or other environmentally caused events can be costed by the simulation technique. An example of one way to account for expenses of maintenance as a result of accidents is depicted in Figure 4-1 with the linking path number 17. This path is intended for the tabulation of costs required by putting track into acceptable repair after an accident/incident. Data needed for tabulating these repair expenditures can be obtained from FRA reports of annual national reports of all accidents which occur.

Periodic inspections performed by track personnel are made in accordance with schedules set forth in the FRA Track Safety Standards. These inspections are intended to identify potentially unsafe track conditions, as well as yield
information about the relative state of deterioration of the track as judged by the people doing the survey. These inspection judgments can often be used for planning or scheduling maintenance work when the track degrades to the limiting condition of compliance with set safety standards or increasing costs.

In complying with either federal inspection standards or with company standards (which can oftentimes be more stringent), unscheduled maintenance repairs must sometimes be performed. Existing standards require repair action be initiated immediately if track is found to be outside the limits of the standard (see Reference 10). In some instances, where repair cannot be performed immediately, a slow-order is placed on that section of track.

Track inspection processes may be diagrammed in a simple way as in Figure 4-1 or in a more complex manner as shown in Figure 4-2. The simplified approach is the single inspection review process practiced by many roads. In such a road the inspector will travel the road on a timely basis (once a day, week, or month) and note various track deficiencies. Within a short time the noted required repairs are made by the maintenance of way staff. Thus, the equivalent of the sent "to repair" operations and the other repair block features of the maintenance system would be accomplished.

Railroads (or specific track structures) may require a more involved inspection process. For example, some roads employ automatic systems for rapid scanning inspections of the rail system; and then back-up the measurements with an on-site personal inspection to establish the exact need or amount of repair. This type of procedure would be more like one of the two or three step inspection processes shown in the diagram of Figure 4-2. Many other examples might be given. Appendix K shows a simulation diagram which has eleven types of inspections, all in the simple one loop arrangement.

The kinds of inspections shown in Appendix K, which include:

1. Federal visual inspections
2. State visual inspections
FIGURE 4-2. DIFFERENT WAYS OF MODELING VARIOUS INSPECTION PROCESSES
3. Scheduled track reviews
4. Federal inspection cars
5. Sperry rail cars
6. Sonirail inspection cars
7. Track geometry cars
8. Rail wear cars
9. Track inspected after accidents
10. Track reviewed after weather damages, and
11. Joint inspection cars

are not so much a model of one road's inspection practices as they are
a representation of the nation's rail system as a whole. The intent of the
inspection portion of the diagram is to account for the costs of inspection.
In doing so, each of these represented "inspection loops" could have various
amounts of track, ties, or ballast associated with it annually.

Furthermore, each of these "inspection loops" can be separately used to
account for separate inspections on all or just some of the six FRA track
classes modeled. Thus, each inspection loop may have in the modeled simu-
lation data up to thirty (5x6) separately specified inspection rates. The
model allows for separate inspection rates for each of the five components
(i.e., jointed rail, welded rail, wood cross ties, concrete cross ties,
and ballast) as well as for a separate inspection rate for each of the
six classes of track.

4.3 Maintenance Decisions and Deferment Policies

Most maintenance decisions are modeled with diagrammed branching points.
These circled points in the maintenance system provide for alternate decisions
to be made on the components represented by the paths drawn. These split
points could represent managerial actions or policies which might cause, for
example, all track inspected and found defective with cracks to be replaced,
or up to 5% of the ties in use to be replaced annually, or any other set of
conditions to be placed on the repair and up-keep of the track. It is not
the intention here to examine all the various maintenance practices of the
rail industry. It is sufficient to say that there are many ways to keep a
track maintained in such a way as to keep the roadbed structures (rails,
ties, etc.) in acceptable operating condition both from a business standpoint
as well as from a safety standpoint.

28
The present simulation cost methodology models maintenance policymaking procedures (if they are known for a given railroad maintenance system) through the use of:

- Simple split points in the maintenance action diagram; or
- Functional equations which describe the policy that causes a given repair operation diagrammed to be executed.

Examples of functions being used in the simulation are given in Section 6.

The deferring of maintenance work can be handled in the present methodology in several different ways. In simulating a specific railroad more than one of these methods may have to be used in order to represent accurately all deferring policies and procedures of that road.

The modeled maintenance system shown previously in Figure 4-1 takes the direct approach of setting aside one path which represents the rate at which maintenance work is delayed for a later period. The costs tabulated on this path would represent the net savings (or loss) from two economic factors. On one hand, there would be a savings since some real amount of maintenance labor is eliminated. On the other hand, repeated maintenance deferment could cause the track structure to degrade to the point where business related revenues may be adversely affected. Degraded track conditions might cause:

- delays in customer deliveries,
- more costly deliveries,
- more frequent train equipment breakdown,
- more expensive repairs when they are finally performed.

Additional approaches to getting an estimate of deferred maintenance costs could be taken with the simulation cost methodology. Although they have not been employed in the costing demonstrations discussed here, these would involve:

1. Setting up several maintenance repair blocks which represented the same repair action but would have costs based on the amount of deferred work accumulated.
4.4 Manufactured Supplies and Scrap Materials

As mentioned earlier, the present simulation cost methodology is in some ways analogous to the flow of fluid in a complex network of water pipes. Supplying and scrapping of materials in the cost analysis have their counterparts in sources and sinks of the water pipe analogy.

The schematic diagram of the simplified maintenance cost system previously shown in Figure 4-1 has three points associated with material origination or removal. The box labelled "component manufacturer" and its attached path (path 6) is where those purchase costs of new materials would be tabulated. Scrap yard paths (path 10) in this case are used to acknowledge and accumulate material return costs realized from selling materials removed from use (e.g., metals).

The label "renew shop" is used for expenses related to putting degraded components such as jointed rail into a usable condition as continuous welded rail. This procedure involves transportation and delivery costs, as well as shop welding expenses in this simplified maintenance picture.

4.5 Cascading and Reuse of Maintained Materials

If rail cascading is handled in a cost simulation, then some rail which is removed from a higher class (say class 6) could be reused or installed in a part of the rail system represented by a lower class (say class 3 as displayed in Figure 4-3). In this example the rail leaving the class 6 maintenance repair block along the "reuse" path would appear or be the source "cascade" for the rail being installed in the FRA class 3 maintenance block. Figure 4-4 details one way the "Maintenance Repair Block #1" might be layed out for systems that contain rail cascading in their maintenance operations.

This, of course, is only one way to handle cascading of track components through the use of the simulation cost model. Another way might involve the drawing of a single large schematic diagram which had "regions" or
FIGURE 4-3. CONCEPTUAL LAYOUT OF SIMPLE SCHEMATIC DIAGRAM SHOWING ONE DIAGRAM FOR EACH CLASS OF TRACK
FIGURE 4-4. DETAILS OF A MAINTENANCE REPAIR BLOCK DRAWN TO HANDLE SIMPLE CASCADING OF RAIL
"blocks" of paths which are tailored separately to each class of track maintained. Then the movement of cascaded materials would be followed from "class" to "class" with separately drawn lines or paths.
A complete set of costing data for the track maintenance cost model will consist of the following information:

1. Annual starting number of components handled on each path of the Maintenance Action Diagram.
2. Quality information on components.
3. Unit cost on each path for the maintenance action cost categories required for the simulation diagrammed.
4. Functional parameters for the above entries needing parametric modifying relationships.

5.1 Model Path Component Quantities

Since the paths of the simulation model represent maintenance actions, these paths may have various amounts of each track component being acted upon. The simulation process requires the annual amount of each component repaired, worked upon, or otherwise being handled by each path at the start of the simulation. The quantity of each path component is entered as shown in Figure 5-1 (see data descriptor code "20").

As in the case of the fluid pipe analogy it is possible to under-define the system flows by specifying the flows within too few pipes of the system. In the case of the cost model, an insufficient data set would require the user to provide more data. A necessary minimum data set provides component quantities for:

1. Each path directly attached to or emanating from every circled split (branch) point in the diagram, and
2. Each manufacturing or scrap path represented.

It is not necessary to specify quantities on paths which emanate from summation points in the diagram unless these paths are connected directly to a branch point.

From these initial quantities the computer program can compute and store all system annual repair rates (flows) for every path diagrammed and can store them for simulation cycling. The subroutine entitled "RAWDAT" uses this particular
FIGURE 5-1. SAMPLE INPUT DATA DESCRIPTION
data entry format. However, alternate entry formats could be generated if the user wanted to take input data directly from his own computer stored accounting records without retyping the information.

5.2 Cost Data Forms, Sources, and Types

Costs for simulating maintenance work can be found in many forms and can be obtained from many different sources. Such sources as:

- Standard accounting records,
- Policy definitions,
- Business practices,
- Accident records,
- Component failure histories,
- Standard maintenance practices,
- Safety compliance standards, and
- Published reports

can be used to determine various input cost numbers for simulating track maintenance operations.

Cost data, however, must be entered for simulation purposes as shown in Figure 5-1. These initializing costs are all entered directly after codes "30" to "49". Each cost set entered is associated with the data descriptor of the same number listed in Figure 3-1 and detailed further in Appendix G (in particular refer to Figure G-2). Costs are entered on a per unit basis. Each assigned cost designator code has its own unit of measure. If the cost entry is material based then the three cost numbers needed are: number of material units per mile, dollars per material unit, and a multiplying factor (usually one (1.0)). If the cost entry can be based upon hours then the three inputs required for each descriptor are: hours per mile, dollars per hour, and number of people (or a multiplying factor).

The entries in each case provide a unit price rate (dollars per mile as it now stands) for every maintenance action expensed on that path of the diagram. By entering the costs on a unit basis they can be easily scanned at a future time and updated automatically where appropriate with the aid of the computer.
editing system. For example, if the labor rates (dollars per hour) increase then the data file can be called up and changed as needed.

5.3 Concept of System Component Qualities

Track substructures (i.e., rails, ties, etc.) are considered defective for modeling purposes if some identifiable work needs to be performed on that track in order to bring it within an acceptable level of use. These "acceptable levels of use" are typically defined by some safety standard under which the railroad operates. A set of track structural qualities has been chosen as a numeric way to define the fraction of the track system that needs repair work. Two alternate definitions of quality have been defined and either can be used. One's precise definition need only be made clear when setting or adjusting the numeric parameters in the computer simulation.

At each time step the maintenance simulation model provides for the replacement of defective components with new ones, such as installation of new track for worn out track. The modeling procedure, however, also allows for maintenance practices that put track structures back into a "like-new" user condition without component replacement. Track realignment, surfacing, and brush control operations are examples of this type of maintenance practice.

Track defect rates of growth are analytically generated during the simulation with a set of Weibull* (Refs. 9, 11) defect distributions for each track substructure modeled. Track system degradation is in the present simulation procedure derived from accumulating defects that are generated from natural causes during a time step, as well as from neglecting to replace all defects that are present in the system at the beginning of the time step (maintenance deferment). Each component is modeled with its own failure rate distribution. The rate at which defects are created can be modeled as either constant or changing with time. In the present examples the Weibull parameters were chosen to keep the defect generation rate constant with time. Data entry examples for failure rate generation of each component for each FRA track class are given in Appendix F (p.F-16) and the Weibull distribution given explicitly in Appendix E (p.E-3).

*The Weibull distribution is an extreme value distribution for the smallest time to failure from a large sample of times from a given failure distribution.
Track component quality (Q) in the present model is defined as a ratio of two measures, the total defective unit amounts (miles) of a given component divided by the total unit amounts (miles) of that component in the system. This measure has the advantage of simply separating components into two defined "states"; good and defective. This ratio measuring concept of quality is displayed in Figure 5-2. As depicted, this particular concept does not portray the defect concept measure of "how bad is it". Since some repairs or levels of repair may require a measure of defect severity, other "quality" or track component condition measures may have to be developed. This would not be difficult as long as the concept of "quality" as defined for the simulation remains consistent throughout the process of data gathering and output interpretation of the results.

One alternate concept of quality which is possible defines it as a frequency of occurrence of defects per unit measure (mile) of that component. This measure has the advantage of indicating the defect severity on the basis of defect density (number/per mile) of the average defect level for that component's population.

One way of obtaining the advantage of either measure of quality; i.e., defect proportion based or frequency based measure, is to adopt a minimum unit length (e.g., 39 ft.) which is associated with each defect. In doing so, each definition of quality can be obtained from the other; i.e.,

\[
\frac{\text{Defective Proportion}}{\text{Based Quality}} = \left[ \frac{\text{Frequency Based}}{\text{Quality}} \right] \times \left[ \text{A Constant of Propportionality} \right]
\]

In other words,

\[
\text{Defective Proportion} = \left[ \frac{\text{Number of Defects}}{\text{Standard Length, Mi.}} \right] \times \left( \frac{\text{Minimum Defect Length, ft.}}{5280 \text{ ft. mi.}} \right)
\]

*In the present form of the cost simulation methodology, the defect proportion based definition of quality has been used. At the present time, this requires the user to structure his track "quality" data in terms of that definition.
Figure 5-2. Component quality as a measure of system condition.
5.4 Functional Parameters and Their Forms

The majority of cost models developed to date lack a method of evaluating the cost effects of certain parametric alternatives. The cost advantages (or disadvantages) in terms of track maintenance expenditure caused by higher traffic densities, changes in Federal Safety Standards, or an alternate component design, are not usually handled in the traditional cost modeling procedures. The present model addresses these potential variations in the operating system through the identification and use of functional relations.

If a cost or rate of track repair has a known dependence which can be written in terms of either the state variables (component population and component quality) or time, then use can be made of several preprogrammed families of functions which will carry out the definable dependencies during the simulation. Seven families of curves have been already defined (refer to Appendix E). Families of curves; i.e., linear, power, exponential, and others have been programmed for activation and can be "turned-on" through the use of various data codes. Figure 5-3 shows an example of data entries which makes use of a modifying function (number 7, Appendix E). This equation is used in one of the sample computer runs discussed in Section 6.

In order to "activate" the preprogrammed functions the user must enter two lines of data, as shown in Figure 5-3, directly after the data elements (costs or flows or qualities) to be modified by the functional expression. This procedure allows basic data elements to be modified in different functional ways by simply entering new functional parameters rather than new data values themselves. Overriding functions onto base data can be easily "added" or "taken-out" of the analysis by simply putting in or deleting the two data lines required per function applied.

If the programmed equation has a user selectable dummy variable (see first five equations, Appendix E) then the dependent function can be linked to the independent variable "X" through the data designator codes 50-79 listed in Figure 3-1. Although simulation time (code "50") is the only code presently
PORTION OF SAMPLE INPUT DATA
FROM EXAMPLE SIMULATION RUN

Isolated Parameter Data Descriptor Code

005 2 20 50 / TO REPAIR
005 2 20 7 2.0 39.0 / TO REPAIR

Always path number to which data is referred

Enter up to five real parameter constants A, B, C, X₀ and Y₀ for programmed Equation "7".

In this case Equation Number 7 is to be evaluated with A = 2.0, B = 39.0, C = X₀ = Y₀ = 0

"7" means Apply parameters entered to programmed equation number 7. See "FUNCT" subroutine.

"20" means Apply factor computed from this equation activation to the quantity (flow) of the component this entry is attached.

"2" means Entries are for activation of programmed functions

FIGURE 5-3. PORTION OF SAMPLE INPUT DATA FOR USE WITH PROGRAMMED MODIFYING FUNCTIONS
updated during the simulation it is intended that other isolated parameters such as speed, load, millions of gross tons (MGT), etc., could also be step-wise defined as independent variables.

For example, if one wanted to "modify" the annual amount of rail inspected over a ten year period as depicted by the dashed line on the uppermost graph of Figure 5-4, then he would enter the two data lines as shown in Figure 5-5.

The net results to the simulated systems "rail quality" and cumulative maintenance expenditures caused by this "modifying" function on inspection rates is shown in the lower two graphs of Figure 5-4.
FIGURE 5-4. SIMULATION OF MODIFICATIONS OF INSPECTION RATE MULTIPLIERS AND RESULTING RAIL QUALITIES AND SIMULATION COSTS OBSERVED FROM COMPUTER RUN
Two data lines, if inserted, will "modify" rail inspections as shown in previous figure over simulation time span. Isolated parameter "50" simulation "time" will be used as independent variable through Equation 5 of FUNCT subroutine.

**FIGURE 5-5. SAMPLE INPUT DATA FOR MODIFYING TRACK INSPECTIONS WITH TIME**
6.0 SAMPLE TRACK MAINTENANCE SIMULATION COST MODEL RUNS

Two example case runs of the computer simulation model were chosen for inclusion in this report. They include:

Example #1. A ten-year time based simulation which shows a comparison of two alternative inspection policies which are known to vary with time, and

Example #2. A test run which depicts a single time point costing (zero year) of a large hypothetical maintenance of way system.

The first example is intended to portray a small railroad whereas the second would be a larger, more complex system, such as that which makes up the full 200,000 miles contained within the United States. These maintenance systems were generated for example purposes; however, run number 2 contains many base line cost elements (see comment line inputs of Appendix H) which were drawn from References 1 through 7.

6.1 Background Information on Example Run #1

A hypothetical Alpha Railroad maintenance system was defined. This system is detailed by the information given in Figures 6-1, 6-2, Appendix G, and the previously presented maintenance action diagram of Figure 4-1. The example run consists of a set of base data plus three overriding functions which pertain to:

1. The amount of track inspected per year (function linked to time of simulation).
2. The number of accidents which can occur (function linked to defects in track).
3. The maintenance capacity of the maintenance of way department after the rail has been identified for repair (function linked to repair resources available at start of simulation).

6.1.1 Inspections

The inspection rate of rail can often be placed on a time basis (see Reference 10). This cycle time is usually on a weekly or monthly basis but can be more often in typical operations. The major expense of
### Figure 6-1: Description of the Railroad Used for Simulation Purposes

<table>
<thead>
<tr>
<th>Road Name: ALPHA RAILROAD</th>
<th>Schematic Diagram Used: 0001</th>
<th>FRA TRACK CLASSES 1, 2, 3</th>
<th>FRA TRACK CLASSES 4, 5, 6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SYSTEM TOTALS</td>
<td>Component 1</td>
<td>Component 2</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>Jointed Rail</td>
<td>Welded Rail</td>
</tr>
<tr>
<td>Track Systems Size</td>
<td>1000 miles</td>
<td>700 miles</td>
<td>50 miles</td>
</tr>
<tr>
<td>Track Accident Rate (Initial)</td>
<td>.01/mile</td>
<td>.011/mile</td>
<td>.02/mile</td>
</tr>
<tr>
<td>Track Accident Rate (During Simulation)</td>
<td>(per class only)</td>
<td>[Variable based on rail quality function]</td>
<td>.02/mile</td>
</tr>
<tr>
<td>Initial Track Quality</td>
<td>.1</td>
<td>.1</td>
<td>.1</td>
</tr>
<tr>
<td>Repair/Inspect Policy</td>
<td>100% Defects</td>
<td>100% Defects</td>
<td>100% Defects</td>
</tr>
<tr>
<td>Maintenance Deferment Policy Assumed</td>
<td>(per class only)</td>
<td>10%</td>
<td>&quot;16%</td>
</tr>
<tr>
<td>Maintenance Overload Capacity</td>
<td>(per class only)</td>
<td>[Variable based on simulation function]</td>
<td>Total</td>
</tr>
<tr>
<td>Average Component Age</td>
<td>20 years</td>
<td>20 years</td>
<td>20 years</td>
</tr>
<tr>
<td>Characteristic Component Life</td>
<td>32 years</td>
<td>32 years</td>
<td>32 years</td>
</tr>
<tr>
<td>Weibull Slope</td>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Rail Reuse Realized</td>
<td>(per class only)</td>
<td>80%</td>
<td>80%</td>
</tr>
<tr>
<td>Rail Scrap Fraction</td>
<td>(per class only)</td>
<td>Net Unused</td>
<td>Net Unused</td>
</tr>
</tbody>
</table>

FIGURE 6-1. DESCRIPTION OF THE RAILROAD USED FOR SIMULATION PURPOSES
FIGURE 6-2. SHOWING FUNCTIONAL RELATIONS USED IN THE SIMULATION DEMONSTRATION
of inspection is the transportation and labor of performing the task. In some operations the inspector also has "immediate repair" obligations if they can be handled with the small amount of equipment he carries. The present model has two assumptions imposed upon the inspection cycle. First of all, the inspection loop sends toward the maintenance repair block all the "defective" track found during the inspection cycle. Secondly, the amount of track inspected during the simulation of this example is escalated on the schedule shown previously in Figure 5-4 of the last section. This variable amount of inspection based on time in this example, portrays the program's ability to alter maintenance action path flows through functional parameters (see input data for path 1, Appendix G). In addition, this inspection escalation with time follows one of two scenarios expected if certain future plans to rejuvenate the track system are executed, (Reference 8).

In a recent report by Dr. Orringer (8) of the Transportation Systems Center, it has been recognized that different remedial inspection procedures could be implemented in order to rejuvenate substandard track. Two alternate inspection policies were reviewed as new approaches to implementing modifications in standards such as the one outlined above.

These inspections procedures were labeled as the:

1. Interim Revision Plan, and

2. The Defect-Rate-Based Revision Plan

Either of these plans, if implemented, would have a time-varying impact on present day inspection resources. The net expected impact on these resources was depicted graphically in previously shown Figure 5-4 of Section 5.

In general, any formula which generates a simulated inspection procedure could be used. Two examples not exercised in the present program which could be implemented are shown in Figure 6-3. These inspection rates are shown to be linked to the quality of the rail in the system, but in general, could be associated with other variables such as traffic density or average car loading.
Two Non-Exercised Hypothetical Inspection Policies Which Might Be Based on Rail Qualities

FIGURE 6-3. TWO EXAMPLES OF NON-LINEAR INSPECTION POLICIES WHICH COULD BE LINKED TO RAIL QUALITY
Each is depicted as increasing the amount of inspection as the rail gets worse. In particular, the discontinuous inspection schedule (plot "B", Figure 6-3) would have a tendency to create step-like changes in the amount of track repaired, if the track passed toward the maintenance block were based on the amount of inspection performed. This type of simulation would imitate the case where the probability of finding defects in a system would be proportional to the amount of effort spent in reviewing the system.

Other policy concepts might also be added to alter the amount of rail sent toward the maintenance block from the "inspection loop". Such practices of sending (or finding) 50% of all defective rail in the system could be implemented. This type of simulation would represent an automated inspection car that has the feature of only finding some fraction of the defects because of its design limitations.

6.1.2 Accident-Related Track Mileage Repaired

For simulation, the Alpha Railroad was assumed to have accident-associated track and the mileage of that track was assumed to depend on rail quality. This dependence of the accident-associated track mileage was used only for jointed rail in FRA track classes 1-3. The dependence is given in Figure 6-2 as the ratio of the accident-associated track mileage to the initial mileage at the start of the simulation. The figure shows that the accident-associated track miles repaired goes down linearly as the condition of the rail improves (see Reference 14). The accident-associated track mileage for perfect rail condition (quality=0) is that associated with accidents not caused by the track but nevertheless requiring track maintenance.

6.1.3 Maintenance Capacities of Repair Organization

The example run #1 contains a function which provides for varying amounts of track to be deferred during the simulation. The selected function was chosen to reflect the fact that certain maintenance of way organizations have limited resources.
For this example, a simple limiting factor of two (2) over and above the originally defined rail defect repair capacity was chosen. The lower portion of Figure 6-1 shows the simulated association between the amount of rail designated as in need of repair and the amount which is passed along to the maintain path. As the amount of rail needing repair goes up, more rail is then deferred on the argument that the maintenance system is being stretched to more of a limit than it can respond.

6.2 Run Input and Output for Example #1

The input data for example run #1 of the simulation model are the contents of Appendix G. Shown in Appendix I are the outputs which include:

- Program Run Information
  - Data File Used
  - Simulation Description
  - Weibull Parameters Used
  - Active Functions with Parameters
- Annual Maintenance Costs by Simulation Time Point
  - Five Time Points (0-2 years)
- Track Maintenance Cost Summary Table
  - Costs by Designator Codes 31-49
  - Costs by Track Class
  - Costs by Track Component
  - Costs by Major Maintenance Repair Blocks

The program outputs for this simulation period reveal that:

- Maintenance expenditures per year decrease as the time of the simulation proceeds.
- The simulated track condition ultimately improves as time proceeds. The average qualities of each rail type by class drop to separate levels even though they were initialized to the same starting value.
In order to get an impression of how some features of the simulation influence the results, a similar run to the above was made. This alternate run was made with one change in the input data. Equation number five which modifies the inspection cycle was deleted. All other inputs were kept constant.

Deletion of the inspection modifying function provides an output for comparison between the alternate inspection policies assumed. See Figures 6-4 and graphs of previously discussed Figure 5-4 for comparison of results. In comparing the maintenance costs with (modified) and without (base) inspection modifiers it can be noted that:

- The modified Inspection Plan brings the track to a "better condition" more rapidly than the reference base simulation (see first two columns Figure 6-4).
- The modified simulation results in a track condition which is ultimately better than the base simulation maintenance practice (see Figure 5-4).
- The annual cost to maintain the track under the modified simulation conditions is slightly greater than the base maintenance case (see Figure 5-4).
- The total cumulative difference in maintenance expense after ten years with and without the imposed inspection modification is 3.1%.

In summary, these simulation comparison runs with time have shown some of the modeling capabilities of the present simulation cost methodology. The interrelated effects of changing inspection procedures and the realizable quality of the rail and the net associated cost results are shown.

6.3 Sample Run Number 2 Using a Large Data Base File

Maintenance cost operations data obtained from references one through seven were used to generate the elements of the input data. These input data are listed in Appendix H. The data values are keyed to the complex maintenance action diagram #2 shown in Appendix K.
### FIGURE 6-4. SUMMARY COST MODEL RESULTS WITH AND WITHOUT INSPECTION MODIFYING FUNCTION

<table>
<thead>
<tr>
<th>YEAR</th>
<th>SYSTEM RAIL QUALITIES</th>
<th>TRACK MAINTENANCE COSTS</th>
<th>ANNUAL RATES</th>
<th>CUMULATIVE</th>
<th>% Increased Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Class 1,2,3 Track</td>
<td>Class 4,5,6 Track</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Jointed</td>
<td>Welded</td>
<td>Jointed</td>
<td>Welded</td>
<td>Base Run</td>
</tr>
<tr>
<td>0</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>1.000</td>
<td>5312979</td>
</tr>
<tr>
<td>.5</td>
<td>.0761</td>
<td>.0759</td>
<td>.0737</td>
<td>.0737</td>
<td>4009673</td>
</tr>
<tr>
<td>1.0</td>
<td>.0651</td>
<td>.0640</td>
<td>.0585</td>
<td>.0585</td>
<td>3337871</td>
</tr>
<tr>
<td>1.5</td>
<td>.0597</td>
<td>.0568</td>
<td>.0497</td>
<td>.0497</td>
<td>2974604</td>
</tr>
<tr>
<td>2.0</td>
<td>.0569</td>
<td>.0509</td>
<td>.0446</td>
<td>.0446</td>
<td>2773460</td>
</tr>
<tr>
<td>2.5</td>
<td>.0555</td>
<td>.0456</td>
<td>.0416</td>
<td>.0416</td>
<td>2660741</td>
</tr>
<tr>
<td>3.0</td>
<td>.0548</td>
<td>.0415</td>
<td>.0399</td>
<td>.0399</td>
<td>2597199</td>
</tr>
<tr>
<td>3.5</td>
<td>.0544</td>
<td>.0389</td>
<td>.0389</td>
<td>.0389</td>
<td>2561299</td>
</tr>
<tr>
<td>4.0</td>
<td>.0542</td>
<td>.0376</td>
<td>.0384</td>
<td>.0384</td>
<td>2541037</td>
</tr>
<tr>
<td>4.5</td>
<td>.0541</td>
<td>.0370</td>
<td>.0380</td>
<td>.0380</td>
<td>2529659</td>
</tr>
<tr>
<td>5.0</td>
<td>.0540</td>
<td>.0369</td>
<td>.0379</td>
<td>.0379</td>
<td>2523342</td>
</tr>
<tr>
<td>10</td>
<td>.0539</td>
<td>.0435</td>
<td>.0376</td>
<td>.0376</td>
<td>2518858</td>
</tr>
</tbody>
</table>
This complex diagram is intended to portray a composite of several railroads since this model contains:

- Eleven separate modes of inspection, and
- Twenty individual major maintenance repair operation blocks.

This model might be used to represent, if desired, all railroads within the U.S. as a composite. Although some single railroads could be simulated with the complex diagram shown most roads do not have many of the features incorporated in this complex model.

For example purposes then, some of the data contained in Appendix H was devised for a hypothetical road of 200,000 miles in length and assumed to be comprised of "all" features of the complex diagram. Much of the maintenance gang work laid out in the various major repair blocks (see twenty separate operations of Appendix K) were drawn up from conversations with railroads consulted during the project. Individual manpower performances as well as material quantities and unit costs entered to the data file of Appendix H have been obtained from references 1 through 7 (see commentary of data file).

A summary output of the single time period from this hypothetical maintenance system is given in Appendix J. Notable features of this example run are listed both in the heading of the computer run output, as well as in the summary (last page of Appendix K).

Run assumptions include:

- This system is represented by a maintenance action diagram with about 575 paths and 201 diagram junction points.
- The system has 5 components:
  
  Jointed Rail
  Welded Rail
  Cross Ties
  Concrete Ties, and
  Ballast
• Due to the complexity of this data set only one FRA track class was assumed.

• No overriding functions were used in this example run to modify any of the initial data values.

• The total mileage was assumed to be roughly split 50-50 between welded and jointed rail.

Output features include:

• Summary costs of maintenance by path number of the modeled maintenance action diagram.

• Materials and labor make up the majority of the maintenance repair costs for this single year's expenses (see cost codes 31 and 40 to 45 of the Track Maintenance Cost Summary Table; last page of Appendix J).

• Rail is the most expensive of the five components to keep up.

• Maintenance repair blocks one through four account for most of the maintenance costs (these represent rail laying operations).

The execution of this program simulation requires about one to two minutes wall clock time, as opposed to the same time for twenty time steps used in example simulation run number 1 discussed previously. This time is for a PRIME 350 run with data being held on a hard disk drive unit. Each of these program simulations have been performed in less time with an altered "PATHIN", "PATOUT" subroutine than that shown in the listing of Appendix D. The altered subroutines used direct addressing schemes, to and from the computer disk which eliminated the required large computational "RARAY" used for storing the data elements during program run.
REFERENCES


6. Based on consultation with small railroads.

7. Based on consultation with large railroads.


10. FRA Safety Standards


1.0 GENERAL INFORMATION

1.1 Summary
The CONTRL program is designed to simulate the operation of a railroad track maintenance system and compute the changes in track inventory population and costs over a desired time period. The program has the capability of producing a base case for the system and outputting the state of the system at selected time steps.

1.2 Environments
This program was developed for DOT by Shaker Research Corporation under DOT-TSC-1594.

1.3 References
See Section 7.

2.0 OVERVIEW

2.1 Problem and Solution Method
Refer to example program runs 1 and 2 with data explanation in Section 6.

2.2 Program Inventory

2.2.1 Main Program: CONTRL
"CONTRL" is an extremely modular program for calling required subroutines as required through data selections specified.

2.2.2 Subroutines
Refer to Appendices C, D and E which consist of flow charts, listings and further explanations by name.
2.3 File Inventory

CONTROL/6 : Source Program (Appendix D)
IFLAG     : Diagnostic control file
IFILE     : Integer Controls for Program Options
RFILE     : Real Controls for Program Options
INODE     : Diagram Path Linkage Designators
RNODE     : Split Fractions for all junctions Array
FUNCT     : Function evaluation Parameter Storage
STATE     : System quantity and qualities
WEIBL     : Component Failure Rate Distributions
DATA      : User input cost operations information
OUTPUT    : Receiving file for output storage.

3.0 DESCRIPTION OF RUNS

3.1 Run Inventory

1. Base Run: Given a set of system flow inputs and node decision parameters, the run will produce the corresponding system flows and costs by path and code.

2. Time Simulation Run: Given a starting population state, a time step, and an ending time, the run will use the Runge-Kutta method to determine the change in state for each time step. The technique requires sufficiently small time steps in order to maintain accuracy.

3.2 Run Progression

In all cases the Base Run is done first. To select other desired run options the control indice values stored in the IFILE and RFILE must be altered. See Appendices F, G, and H for examples.

3.3 Run Description

3.3.1 Control Inputs

The first eight files are read in through subroutine DATAIN on FORTRAN
units 8, 9, 12, 13, 14, 15, 17, and 16, respectively. DATA is read in the RAWDAT subroutine on a file unit set from IFILE, so alter the IFILE data accordingly. Output is written to a unit set in IFILE as well.

3.3.2 Operating Information

a. Base Run: see input/outputs of Appendices H and J.

b. Run Time: 2.0 minutes CPU.
   Turn Around Time: 3.5 minutes total on PRIME 350.

c. Operator messages and commands: NONE

d. Trouble Shooting: All control is through IFILE and RFILE.

See example explanations in text and appendices or contact the authors.
1. General Information

1.1 Summary

This program was written under standard FORTRAN IV conventions and consists of a main program, subroutines called by the main program only, subroutines called by the main program and other subroutines, and subroutines accessed solely from other subroutines. There are no recursive calls. The program is supported by two types of data files distinguished by purpose, those with program control values, the rest with calculation data. The software package of main program, subroutines, and data files is completely self-contained except for the standard FORTRAN functions which must be provided by the user's compiler.

1.2 Environment

This program was developed for DOT by Shaker Research Corporation under DOT-TSC-1594.

1.3 References


2. Program Descriptions

2.1 Program Description - Control

2.1.1 Problem and Solution Method. The problem was to set up a mathematical model to simulate the maintenance system for track. The solution was to set up a diagram of nodes and paths. The paths represent various steps and operations of the maintenance process. The nodes, depending on type, either combine the flows on two incoming paths and send the
combined flow down the single outgoing path, or determine how to split up the flow of an incoming path between two outgoing paths. Beginning with the information from some starting path, the program sweeps through the diagram, filling in the flows and qualities for all downstream paths. There are four basic path types, in-use, internodal, manufacture, and scrap. There is only one in-use path, it contains the system population, and quality for all components. Internodal paths represent maintenance operations or movement of components between nodes. Manufacture paths enter the system: the flow on each such path is always set equal the defective portion of the flow in the path it is summed with. The scrap path permits unrepairable units to leave the system. Any path that leaves, enters, or in any way alters the system quality or population, is monitored to keep track of total system change.

A single sweep through the diagram using equilibrium values for the start path, will produce a base run. By taking the system change in state from a sweep, and applying the Runge-Kutta technique, the program simulates operation through time.

2.1.2 Input (See Appendices G and H for listings). Input is from nine data files, read in the following order: IFILE, WEIBL, INODE, RNODE, FUNCT, STATE, RFLE, IFLAG, and DATA. The first eight are read in by the DATAIN subroutine. DATA is read in the subroutine RAWDAT. (NOTE: These file names correspond to those of the sample run provided with this manual. The file names may be changed to suit the user without affecting the program as long as they are read in the proper order.) Files IFILE, RFLE and IFLAG are of the control file type described in Section 1.1. DATA is actually a combination control and cost information with a control data line for every block of flow and costs. Detailed file descriptions and formats are presented separately in Appendix F.
2.1.3 Processing

a. Processing Logic

The main program is faced with two primary tasks. It must first read in, sort, process and store data to be used in the second task. It then must simulate the operation of a track maintenance system with the processed data. To accomplish these goals, the main program acts as a central control area (hence the name CONTRL) calling subroutines as required. The course of action taken is heavily dependent on the first subroutine called, DATAIN. DATAIN reads and stores most of the control variable values used (see Section 2.1.2 and Operations Manual Section 2.2 for greater detail). DATAIN also reads in precalculated population and node split values if base path data is not to be used.

If base data are used, subroutines INTDAT, RAWDAT, and NDVAL are called respectively to clear data structures, to process the base data, and to calculate node split values. This completes the first task, and if the data structures are permanent, the program can be halted at this point and restarted later. (Data structures will be altered to best suit the users machine by simply changing the PATHIN and PATOUT subroutines.)

The second task of actually simulating the system operation is handled by calling the SWEEP subroutine (called MAIN in early versions of the program). SWEEP handles all aspects of node sweeps, time simulation, and output. In the batch mode version, SWEEP will be the final main program call before CALL EXIT.

b. Data Structures and Linkages

All data, with the exception of control variables, are
stored and referenced by the corresponding class-component. Class-component always goes by column, with nodes, paths, and Weibull factors referenced by row. The large real data storage structure for path data may be envisioned as having blocks of columns for each class-component (even though it may be stored as a single sequential file with a calculated positioning).

There is only one instance of a linked or threaded list. In DATA2, FPLACE, and LNKLST functions are stored as read in, in a temporary file. A pointer array is sorted and after the last function for that class-component is read, the pointer array is used to store the function data in the FUNCT array in the proper order.

c. Variables and Constants

Any non-declared names come directly under the control of the IMPLICIT statement that leads the main program and every subroutine. This program was written by several people, with many key subroutines being started before nomenclature conventions were established. This resulted in some cases of the same variable having many different names in different subroutines. Despite this, several important variable names retain their meaning through most of the program.

**IMPORTANT VARIABLES AND CONSTANTS**

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISTND</td>
<td>Starting node, usually set to 1, but may be set higher for partial sweeps.</td>
</tr>
<tr>
<td>LSTND</td>
<td>Last node checked.</td>
</tr>
<tr>
<td>ISTPH</td>
<td>Lowest path number. Paths need not be numbered sequentially.</td>
</tr>
<tr>
<td>LSTPH</td>
<td>Highest path number.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ISTCL</td>
<td>First class number, like ISTND may be greater than 1 for partial runs.</td>
</tr>
<tr>
<td>LSTCL</td>
<td>Last class checked.</td>
</tr>
<tr>
<td>ISTCP</td>
<td>First component.</td>
</tr>
<tr>
<td>LSTCP</td>
<td>Last component.</td>
</tr>
<tr>
<td>LSTFL</td>
<td>Number of flow property columns in a class-component block of RPATH arrays.</td>
</tr>
<tr>
<td>ISTCT</td>
<td>Number of columns preceding first cost column in a class-component block (equivalent to LSTFL).</td>
</tr>
<tr>
<td>LSTCT</td>
<td>Number of cost columns following first.</td>
</tr>
<tr>
<td>IDM</td>
<td>IPATH array dimension.</td>
</tr>
<tr>
<td>IRDM</td>
<td>RPATH array dimension.</td>
</tr>
<tr>
<td>IBDM</td>
<td>Class-component block size in RPATH arrays.</td>
</tr>
<tr>
<td>NDOUT</td>
<td>Final out path indicator.</td>
</tr>
<tr>
<td>NREFP, NRPTH</td>
<td>In-use path number.</td>
</tr>
<tr>
<td>IUNIT</td>
<td>Output unit.</td>
</tr>
<tr>
<td>IRP</td>
<td>RPATH array reference index for present class-component block. Calculated in RCOLUM subroutine.</td>
</tr>
<tr>
<td>IRCOL</td>
<td>STATE, RNODE, and WEIBUL array reference column for current class-component. Calculated in RCOLUM subroutine.</td>
</tr>
<tr>
<td>STIME</td>
<td>Simulation start time.</td>
</tr>
<tr>
<td>DTIME</td>
<td>Simulation time step.</td>
</tr>
<tr>
<td>LTME</td>
<td>Simulation end time.</td>
</tr>
<tr>
<td>LNZHST</td>
<td>Number of STATE array columns.</td>
</tr>
<tr>
<td>IP1, IP2, IP3</td>
<td>Path numbers from INODE.</td>
</tr>
<tr>
<td>IBSH</td>
<td>Branch node indicator.</td>
</tr>
<tr>
<td>ICDEF</td>
<td>Simple of defective decision indicator.</td>
</tr>
<tr>
<td>IREP</td>
<td>Branch node reference path indicator.</td>
</tr>
<tr>
<td>IPDT1, IPDT2, IPDT3</td>
<td>Population change path indicators.</td>
</tr>
<tr>
<td>ISGN1, ISGN2, ISGN3</td>
<td>Population increase or decrease indicators.</td>
</tr>
<tr>
<td>IQDT1, IQDT2, IQDT3</td>
<td>Quality change path indicator.</td>
</tr>
<tr>
<td>IMFG1, IMFG2, IMFG3</td>
<td>Manufacture path indicators.</td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>RCP, RCP1, RCP2, RCP3</td>
<td>Population rate of change.</td>
</tr>
<tr>
<td>RCQ, RCQ1, RCQ2, RCQ3</td>
<td>Quality rate of change.</td>
</tr>
<tr>
<td>ICOST, ICST1, ICST2, ICST3</td>
<td>Cost calculation indicators.</td>
</tr>
<tr>
<td>IPRNT, IPRT1, IPRT2, IPRT3</td>
<td>NORMAL print output indicators.</td>
</tr>
<tr>
<td>ICL</td>
<td>Current class.</td>
</tr>
<tr>
<td>ICF</td>
<td>Current component.</td>
</tr>
<tr>
<td>IIROW</td>
<td>Current node.</td>
</tr>
<tr>
<td>CDE</td>
<td>Branch node split value.</td>
</tr>
<tr>
<td>PCOST1</td>
<td>Total sweep cost.</td>
</tr>
<tr>
<td>DP, DPOP</td>
<td>Change in population.</td>
</tr>
<tr>
<td>DB, DBAD</td>
<td>Change in defective population.</td>
</tr>
<tr>
<td>USEQ</td>
<td>User quality indicator.</td>
</tr>
<tr>
<td>BAD, BAD1, BAD2, BAD3</td>
<td>Defective flow.</td>
</tr>
<tr>
<td>QUAL, QUAL1, QUAL2, QUAL3</td>
<td>Flow quality.</td>
</tr>
<tr>
<td>FCOST</td>
<td>Function value.</td>
</tr>
<tr>
<td>IND1, INDL, I1, I2, I3</td>
<td>Do loop parameters.</td>
</tr>
<tr>
<td>J, L, K, etc.</td>
<td>Indexing, etc.</td>
</tr>
</tbody>
</table>

d. Formulas
   Formulas and a full explanation of each may be found in Appendix E.

e. Error Handling
   The program has built-in logic to avoid errors like divide by zero. Other errors, such as incorrect path numbers, will call the ERRORS subroutine allowing the program to continue. Major errors such as format/mismatch will cause the program to halt.
e. Limits

The program, as set up, cannot handle more than:

- 30 class-components
- 300 nodes (path junctions)
- 600 paths
- 2 path properties (flow and quality)
- 19 cost codes per class-component

To exceed these limits, the affected arrays must be expanded. If more path properties are desired, the program will have to be redesigned.

The program is also limited to one run type at a time. To change run type, the user must alter the control values in IFILE and RFILE to the appropriate run set up.

2.1.4 Output

Output is primarily done through subroutines OUTPT and HEADNG. OUTPT is called from many other subroutines to output path flow and costs, system states, and system costs. HEADNG is called from SWEEP prior to any output to write the output file heading. All other output, with one exception in NORMAL, will be diagnostic and in the users version could be commented out. Output device selection will be through variable IUNIT, which is set from the IFILE file. (See Appendices I and J for output examples.)

2.1.5 Interfaces

The program will input data from a maximum of nine different units, and output at most through two more. The base path data file, the diagnostic outputs, and the result outputs are set by values selected by the user and carried in the control file IFILE. The remaining data files are read by the DATAIN subroutine, on constant unit numbers; IFILE on unit 8, WEIBL on unit 9, INODE on Unit 12, RNODE on unit 13,
FUNCT on unit 14, STATE on unit 15, RFILE on unit 17, and IFLAG on unit 16.

2.1.6 Arrays

Arrays are dimensioned somewhat oversized to allow for system expansion without changing the program. Most of the arrays can be separated into general categories, those with node oriented data, those with path oriented data, and those with system oriented data. Under the category of path data, some versions of the program have the RARRAY and IARRAY arrays in COMMON. These two arrays are necessary if no direct access file scheme is used to handle the large quantity of path flow and cost data.

### Node Arrays

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Where Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>INODE (6,300)</td>
<td>Inode has one column for each node. The first three rows hold the path numbers of the associate paths. The fourth row holds the sum/branch indicator, the fifth row is the decision type, and the sixth row indicates which path is the node reference path on branch paths. Kept in common block 17.</td>
<td>SWEEP, ABNRML, NORMAL, DATAIN, NDVAL</td>
</tr>
<tr>
<td>RNODE (30,300)</td>
<td>RNODE holds the flow split fraction for branch nodes. Each column represents the corresponding node with the one row per class-component. In the case of a sum node a zero is stored.</td>
<td>SWEEP, ABNRML, NORMAL, DATAIN, NDVAL</td>
</tr>
</tbody>
</table>

### Path Arrays

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Where Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPATH, IPATH1, IPATH2, IPATH3 (15)</td>
<td>An IPATH array by any other name would do just as much. Array holds path types and operation indicators.</td>
<td>NORMAL, ABNRML, OUTPT, RAWDAT, DUMMY</td>
</tr>
<tr>
<td>RPATH, RPATH1, RPATH2, RPATH3, PATH (750)</td>
<td>Holds path flow and/or cost data. Array is divided up into equal sized blocks, one for each class-component. The first LSTFL columns of each block holds flow data, or are disregarded</td>
<td>NORMAL, ABNRML, OUTPT, BRANCH, SUMS, PDOT, QDOT, PCOSTS, ADDRCH, CDFUNC, FUNCTS, RAWDAT, DUMMY</td>
</tr>
</tbody>
</table>
when costs only are handled. The remaining columns hold unit costs, or cost function pointer, or path costs, the last when cost only are stored.

**FUNCT (10,10)**

Holds path function data. The first three columns hold the path number, function number, and data code respectively. The fourth column records how many FUNCT lines following the current one are associated with the current function, the fifth column points to the location in the RF.ILE array where the independent variable is stored. The remaining columns are function parameters.

**VALUE (50)**

Array is used to sum and temporarily hold path data during base data processing. Each column corresponds to a data code. Unused codes are left blank. When data for a class-component block is all readin, VALUE is used to fill the appropriate spaces in RPATH.

**RL, REAL2, REALA, REALB, REALC, READD**

Buffer arrays into which base/data is first read, prior to sorting.

**System Arrays**

**STATE (20,30)**

STATE Stores the system population and quality for each class-component at the start of and at various points of each time step. It also contains the rates of change computed during the time step. The array can be subdivided into five blocks of four rows each. The first and third row of each block hold the population and quality respectively of the class-components for that point in the time step calculations.

**IFILE (100)**

IFILE is the integer control value array. Almost all the constants and subroutines.

Used in almost all
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Where Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>IFILE</td>
<td>control values used are stored in IFILE. Detailed descriptions of the array can be found in the description of the IFILE file used to fill the array (Section 2.1.3).</td>
<td>SWEEP, ABNRM, FUNCTS</td>
</tr>
<tr>
<td>RFILF (100)</td>
<td>RFILE is the real value counterpart to IFILE. Many of the values contained correspond to either time or path functions.</td>
<td>BRANCH, SUMS, FUNCTS</td>
</tr>
<tr>
<td>IFLAG (10)</td>
<td>IFLAG is a diagnostic indicator and control array used in development of the program. IFLAG controls calls to the TRACE subroutine.</td>
<td>ERRORS</td>
</tr>
<tr>
<td>IERROR (10)</td>
<td>The array keeps track of the calls to the ERRORS subroutine.</td>
<td></td>
</tr>
<tr>
<td>ACCUM (750)</td>
<td>ACCUM sums up system costs by cost coded node sweep.</td>
<td>NORMAL, ACOSTS, OUTPT</td>
</tr>
<tr>
<td>REVISN, SCHMDG,</td>
<td>These arrays hold alphanumeric strings describing various aspects of a run.</td>
<td>DATAIN, RAWDAT, OUTPT,</td>
</tr>
<tr>
<td>DATAFL, PLACE,</td>
<td></td>
<td>HEADNG</td>
</tr>
<tr>
<td>RUNN, DATES</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### 3.1

**a. Processor**

The program was developed to run in double precision mode utilizing 64 bits per number. The CPU used was a Prime 350 with 16 bit, 2 byte words. The internal memory required for running is approximately 132 K bytes exclusive of that used for storing the path flow and cost data. The data structures, whether large arrays or direct access files, require close to an additional three megabytes for the maximum size model.

**b. Output and Input Files**

The input files require about 15 K bytes of memory. The data may be stored in sequential files on disk or magnetic tape without alteration. Card storage requires the files to be integrated according to the program reading
order.

The output file, for a base run, would require very little disk memory with memory used increasing proportionally when time simulations are run.

c. Data transmission devices

None. The program and data files are non-interactive and are self-contained.

3.2 Support Software

3.2.1 Operating System. The program is run in batch mode and should be compatible with any system equipped to run FORTRAN IV programs.

3.2.2 Compiler/Assembler. Requires a compiler for standard FORTRAN IV.

3.2.3 Other Software. If FORTRAN functions such as DABS( ) are not inserted by the compiler, then a FORTRAN IV math library will also be required.

3.3 Data Base

See Operations Manual, Section 2.3 and Program Maintenance, Section 2.1.2 and Section 4.5.

4. Maintenance Procedures

4.1 Programming Conventions

The program consists of a main program, CONTRL, and 35 subroutines. Subroutines follow the main program in the listing.

Variable and constants follow the IMPLICIT statement and the standard FORTRAN naming conventions. Mnemonic attributes may be found in most
names but this should not be assumed to always be true as some names were generated at random.

Strings are kept in double precision arrays, and are used to print out headings for the output. String names are mnemonic in nature, but are not distinguished as strings. The remaining arrays, most of which serve to hold large amounts of data, are similarly mnemonic in nature, i.e., if "node" is part of the name, the data stored are node oriented.

4.2 Verification Procedures

Tested and verified data will be provided with the program along with the corresponding output. After installation and whenever the program becomes suspect, the data can be run to verify integrity of the program.

4.3 Error Correction Procedures

Almost all errors will be data related. Be sure that data values read through free formats are separated by blanks or commas. Carefully check the control file to insure against improper indicator and control parameters. If data are correct, confirm that the data files have been opened on the proper reading unit. If the data are all in order, check the following features of the program:

- Are the arrays too small for the model?
- Are there the correct number of repetitions in the loops?
- Are reals and integers being mixed?
- Are all the subroutines present?
- Are all the subroutine arguments present?
- Is the program compiled correctly?

4.4 Special Maintenance Procedures

Data modifications are necessary for different types of runs. Some modifications are specified in the Operations Manual Section 3.3, 3.4 and 3.5.
Program modifications will revolve around four main points, string handling, data functions, I/O units and path flow/cost data structures. Each of these points is machine dependent and must be accounted for before using the program.

Alpha fields are used in DATAIN and RAWDAT to input strings and in OUTPT to write strings. These fields must be altered to conform to the characters per word of the user's machine. The program revision number is assigned to an array in the main program with enclosure by apostrophes. This must also be brought into line.

The date function is found in the OUTPT subroutine. In PRIME versions, this will be found as DATE$A. In DEC10 versions, it will be DATE. Variations for other machines will probably be similar. If the user's machine does not have a date function, it would be a simple matter to input a date by modifying the DATAIN subroutine to read another alpha line from the IFILE file.

As mentioned previously, Output units are set through the IFILE file, as is the READ unit for the base path data. All other inputs are through the subroutine DATAIN with constants for unit numbers. If these constants aren't compatible with the user's machine, the user can simply change DATAIN READ statements to more suitable units. More extensive modification are necessary when card input is required. In this case the user must consolidate all data files into sequential cards, and alter FORMAT and READ statements accordingly. (NOTE: DATAIN inputs precede RAWDAT input.)

The path flow/cost data structures are dependent upon the machine's ability to handle extremely large arrays. Normally path data would be held in two large array IARAY and RARAY which are dimensioned 15 by 600 and 750 by 600 respectively to allow for the maximum of 600 paths, 30 class-components, and flow/cost blocks of 25 rows each. To handle this, some machines will require special compiling procedures. Some machines will be unable to handle it at all. In the latter instance, a direct access
file setup may be able to handle the large amounts of data. As all handling of the data structure is through PATHIN and PATOUT, it is possible to replace these subroutines with two data manipulating subroutines in character with the user's computer. Retain names and arguments in the program but remove RARAY and IARAY from the COMMON block in the main program.
APPENDIX C

Computer Program Flow Charts
MAIN PROGRAM

CALL DATAIN TO INITIALIZE CONTROL INDICES AND CONSTANTS

IF RAW DATA REQUIRED
  Y
  CALL INTDAT TO INITIALIZE DIRECT ACCESS DATA STORAGE TO ZERO
  CALL RAWDAT TO READIN, SORT, ACCUMULATE, AND STORE PATH DATA
  CALL NDVAL TO CALCULATE BRANCH NODE FLOW SPLIT FRACTION FROM INITIAL PATH DATA
  N

IF NODE SWEEP REQUIRED
  Y
  CALL SWEEP FOR NODE DIAGRAM SWEEP
  N

END
SWEEP

Initialize Control Indices from RFILE and IFILE Arrays

Call HEADING to print output Title and Heading

Call SETAAPR to initialize ACCUMAE and STATE arrays

Call NORMAL for actual node sweep

If STIME GE Time

N

Y

Reset indices and call ABNRM for time step calculations

Increment simulation Time (STIME = STIME + DTIME) and STORE

If Final Output Required

Y

N

Call OUPPT for writing final output

RETURN
SETARR

Initialize control indices from IFILE array

Call PATHIN to get system population data from reference path (NREFP)

DO I = 1, IRDM

ACCUM(I) = 0.0D-00

DO I = 1, LTHST

STATE(5,I) = STATE(N1,I)
STATE(6,I) = 0.00D-00
STATE(7,I) = STAT(N2,I)
STATE(8,I) = 0.00D-00

Set rest of Ith column to 0.00

DO I = 1, LNTST

Reset populations and qualities on reference path

Call PATOUT to replace updated population path

RETURN
Initialize control indices and constants from RFILE and IFILE arrays

DO NPASS = 4
  1, 17

Set simulation time and call DRate to calculate population change

DO IIROW 1
  ISTND, LSTND

If output required and time for output

Y

Call OUTPT to write ABNRML output

N

Increment output counter

Calculate the total population and quality changes

RETURN

Set path indices from INODE array

Call PATHIN three times to get path data on all paths associated with node IIROW

Set index from IPATH array

DO ICL = 1
  ISTCL, LSTCL

DO ICP = 1
  ISTCP, LSTCP

Do path Calculations
DRATE

Initialize control indices from arguments IFILE array

DO I = 1, LTHST

Get interim populational and quality changes and reset population and quality for next ABNRML loop

RETURN
Set control indices and constants from IFILE and RFILE arrays

DO IIROW = 1
  ISTND, LSTND

Set path pointer from INODE. Call PATHIN three times to get path data associated with node II-ROW. Set control indices from INODE and IPATH

Print intermediate cost if required, then call WEIBUL to estimate aging and hazard effects

DO ICL = 1
  ISTCL, LSTCL

RETURN

If computed flows are to be stored

Y

Restore any function addresses overwritten with ADDRCH and call PATOUT to save data

N

Call RCOLUMN to calculate the reference index for the RPATH array for class ICL and component ICP

DO ICP = 1
  ISTCP, LSTCP

Calculate path flows for node IIROW. Note, some paths have been calculated at previous node and will not be calculated here, though they will be used to determine unknown paths. The programming logic of this section is detailed in User's Manual.
WEIBUL

Set constants and control indices from argument list

DO ICL = 1
ISTCL, LSTCL

RETURN

DO ICP = 1
ISTCP, LSTCP

Call RCOLUM to get STATE reference column for Class ICL, component ICP

Convert change in number of bad units to change in population quality

Calculate Weibull effect on population quality and store
Datain

Read in data from appropriate files: IFILE and RFILE will contain control indices. INODE will contain the node map and node types. WEIBUL will contain class component data for calculating Weibull effects. IFLAG will hold trace and error indicators. RNODE, STATE, and FUNCT will hold preset values to be used if raw base data is not provided.

Set control indices and constants from IFILE array

Return
TRACE

Increment Trace Message Count

If Trace Count Exceed or if no path to be traced

Write trace message and argument list on unit ITUNIT

IFLAG(1) = ∅

RETURN
PATHIN

Open file directory for reading and write

Set file pointer

Position and open sequential file for reading

* Binary read: IPATH

* Binary read: RPATH

Close file

Close directory

RETURN

* PATOUT is exactly the same except that these instructions substitute a write for read.
Initialize constants and control variables from argument list

If RPATH2 is node reference path

Store RPATH2 flow function address

If flow function present

Call CDFUNC to calculate new split fraction

If defective based split

1

2
Calculate RPA1
tract from RPA
get RPATH3 flc
and RPATH3 qua
of RPATH1.

1

Determine number of bad units on RPATH1. Determine number of bad to go down node reference path. Calculate total quality of flow of good and remaining bad units.

2

If split based on RPATH3

N

Y

Set fraction equal to fraction minus one

Q

Calculate RPATH2 flow subtract from RPATH1 flow to get RPATH3 flow. Set RPATH2 and RPATH3 qualities to that of RPATH1.

Set RPATH3 flow to one and quality to remainder

Set RPATH3 flow to remainder

Set RPATH2 flow to bad and quality to one

Set RPATH2 flow to bad and quality to one

RETURN
Initialize control indices from argument list

If F1 is a manufacturing path
  Y
  Set F1 flow equal to F2 flow
  N

If F2 is a manufacturing path
  Y
  Set F2 flow equal to F1 flow
  N

Sum F1 and F2 flows and calculate the combined flow quality. Assign the results to F3.

RETURN
PDOT

Initialize control indices from argument list

Calculate whether population change will be positive or negative

Determine change in defective population

Sum defective and total changes in STATE arrays

RETURN
QDOT

Initialize control indice from argument list

Determine change in defective population

Set path quality to zero

Sum defective change in STATE array

RETURN
Determine the RPATH array's reference index for the component ICP, class ICL, BLOCK

Determine the RNODE array reference column for component ICP, class ICL

RETURN
PCOSTS

Initialize control indices from argument list

DO $M = 1$

L, K1

Calculate actual path cost for cost code associated with path(M).
Store result in RPATH(M)

Sum cost with PCOST

RETURN
ACCUME

Initialize control indices from argument list

DO M = 1, K, L

Sum RPATH cost columns in ACCUM array

RETURN
FUNCTS

Initialize control indices from argument list

Call ADDRCH to store RPATH functions that will be overwritten

DO K = 1, I2, I3

RETURN

If function to be evaluated

Evaluate appropriate function

Multiply FCOST by function value output function if needed

Assign FCOST to RPATH

If another function

N

Y
CDFUNC

Initialize control indices from argument list

If function not necessary

Call FUNCTS to evaluate the function

Calculate new split fraction

Restore function address

If using reference path

Y

N

Maintain split fraction between one and zero inclusive

RETURN
ADDRCH

Initialize Control
Index from Argument List

If Addresses
are to be saved

Y

DO K= 1
IND1, INDL

Store Addresses
From RPATH in RFC

N

DO K= 1
IND1, INDL

Restore Addresses
From RFC to RPATH

Return

C-24
OUTPT

Initialize control indices from IFILE and RFILE arrays, and from Argument List

IF INTERMEDIATE OUTPUT

IF HEADING NEEDED

Output Heading

Initialize Local Indices.

DO I = I1, I2

Sum Path Costs

Sum Path Cost to Total Cost

IF PATH FLOW EQUAL ZERO

1

IF COST TABLE REQUIRED

Alter Heading

Output Cost Tables by cost code by component, by class, and by repair block

2

3

Y N

IF FINAL OUTPUT

Alter Heading

Y N
HEADNG

Initialize Indices from IFILE array and from Argument List

Output Top Heading of Program Revision, Node Map Data File, Number of Classes Components, Paths and Nodes, Start time, End Time and Time Step

DO ICL = 1, LSTCL

Output Heading Function

IF FUNCTION TO BE OUTPUTTED

Y

Output Function Parameters

N

Increment Function Number

Return
Set Local Indices From INODE Array

If J is a sum node

Call PATHIN 3 times for each path associated with Node J.

Sum Incoming Paths and Assign to Out-flowing Path

C-28
DO J = 1 ISTND, LSTND

Return

If J is a Branch Node

Y

Set Local Indices From INODE

Call PATHIN to get associated path data

DO ICL = 1 ISTCL, LSTCL

DO ICP = 1 ISTCP, LSTCP

Calculate Split Fraction for Node J

N
INTDAT

DO \( T = 1 \) to IIDM

IPATH(1) = 0  
RPATH(1) = 0.0D-00

DO \( I = 1 \) to IROM

RPATH(I) = 0.0D-00

DO \( I = 1 \) to LSTPH

IPATH(1) = I

Call PATOUT to initialize Data Storage for Path I

Return
Initialize control indices & constants from IFILE & RFILE arrays & from the Argument List

Set IFTLINK, RPATH and FRMP arrays to zero

Call ASSIGN to set RL, value, and realz arrays to zero

Read: DATAHL, the data file number or description

Read: First data line; path number, data code, comp, class, IPATH data

Call PATHIN for path IPH to protect existing data
Call RESET to assign RL array values to SCALAR variables. Then call RCLEAR to zero out RL

Read Next data line: IPH, ICD, and RL array

Call RCOLUMN to determine RPATH reference index, IRP

If ICD .EQ. 1

Y

Call DATA1 to process RL array data & sum it up in the value array

N

Call REALCL to zero out the REALA, REALB, REALC, REALD & REAL2 arrays

Read Function Data into REALA
If more function data

Read Function Data into REALB

If more function data

Read Function Data into REALC

If more function data

Read Function Data into REALD

Call REFILL to more REALA, REALB, REALC, & REALD, data into REAL2
Call DATA2 to process and store the function data in the REAL2 array

Read Next Data Line

If
not end of new class component

Y

N

Call DUMMY to fill IPATH & RPATH arrays

DO I = 1, ITMP

Zero out remaining addresses in value arrays

Update function array indices and move functions from FTMP to FUNCT

11
7
5
4
10
9
Call PATOUT to store process path data for program use

DO I = 1, 20

Reset IPTLNK Linkage array

Reset Linkage indicators

If end of data

Y

Return
APPENDIX D

Computer Program Listing
C******************************************************************************
C      PROGRAM CONTROL
C******************************************************************************

CONTROL IS A RAIL ROAD TRACK MAINTENANCE COST MODEL DESIGNED
BY SHAKER RESEARCH CORPORATION FOR THE D.O.T. THE ORIGINAL
VERSION ONE WAS AN INTERACTIVE PROGRAM PRESENTING OPTIONS TO
THE USER AND ACTING UPON SUBSEQUENT INPUT. VERSIONS THREE
THROUGH SIX ARE BATCH MODE PROGRAMS, CONTROLLED BY THE IFILE
AND RFILE DATA FILES. BOTH IFILE AND RFILE INCLUDE DESCRIP-
TIONS FOR THE CONTROL VALUES CONTAINED

PROGRAM HISTORY

ORIGINAL VERSION .............. R. SMITH, H. LIPSTEIN, J. BE-
TOR SHAKER 07/79 - 02/80 *

CONTRL/2 ...................... R. SMITH, J. BETOR SHAKER *

02/80 - 03/80

CONTRL/3 ...................... R. SMITH, J. BETOR SHAKER *

06/80

CONTRL/4 ...................... R. SMITH, J. BETOR SHAKER *

06/80

CONTRL/5 ...................... R. SMITH, J. BETOR SHAKER *

06/80 - 07/80

CONTRL/6 ...................... R. SMITH, J. BETOR SHAKER *

07/16/80

C******************************************************************************
C</C>
C**************************************************************
C SUBROUTINE SWEEP, ISTHD, LSTDH, ISTCP, LSTCP,
C * ISTCL, LSTCL, LSTFL, ISTCT, LSTCT, I IDM, IRDN,
C * IBDM, IUNIT, RNUM, PLACE
C IMPLICIT REAL*s(A-H,0-2)
C THIS SUBROUTINE COMPUTES SYSTEM COSTS AND FLOWS
C THROUGH TIME BY CALLING THE SUBROUTINE NORMAL
C FOR GIVEN TIME STEPS.
C
C ARGUMENT LIST
C SEE IFFILE AND RFFILE FOR MORE
C COMPLETE DESCRIPTIONS.
C
C ISTHD - STARTING NODE
C LSTDH - LAST NODE
C ISTCP - INITIAL COMPONENT
C LSTCP - LAST COMPONENT
C ISTCL - INITIAL CLASS
C LSTCL - LAST CLASS
C LSTFL - NUMBER OF FLOW COLUMNS
C ISTCT - NUMBER OF COLUMNS FROM REFERENCE TO FIRST COST COLUMN
C LSTCT - NUMBER OF COST COLUMNS FOLLOWING FIRST
C I IDM - DIMENSION OF IAPTH
C IDM - DIMENSION OF IAPTH ARRAYS
C IRDN - DIMENSION OF CLASS/COMPONENT BLOCK IN IPATH ARRAY
C IUNIT - FORTRAN OUTPUT UNIT DEVICE NUMBER
C RUNN - RUN IDENTIFICATION
C PLACE - USER IDENTIFICATION
C
C COMMON/I1/IERORC(10)
C COMMON/I2/IFILE(100)
C COMMON/I3/IFLAG(10)
C COMMON/I7/INODE(6,300)
C COMMON/R1/ACCUM(750)
C COMMON/R2/FUNCT(10,10)
C COMMON/R3/RNODE(30,300)
C COMMON/R4/STATE(20,30)
C COMMON/R5/RFILE(100)
C COMMON/R6/WEIBL(3,30)
C
C DIMENSION IPATH(750), RUNN(10), PLACE(10)
C C'/="/ Initialize loop parameters and loop independent control indices
C
C S TIME = RFFILE(1)
(0125)  DTIME = RFILE(2)
(0126)  TIMEL = RFILE(3)
(0127)  N1 = IFILE(31)
(0128)  N2 = IFILE(32)
(0129)  LNSTHST = IFILE(24)
(0130)  NDOUT = IFILE(29)
(0131)  IFMT = IFILE(36)
(0132)  RFILE(50) = RFILE(1)
(0133)  C  INITIATE ACCUM AND STATE ARRAYS AND BEGIN TIME LOOP
(0134)  C  CALL EVERYTHING
(0135)  CALL HEADNG(IUNIT,DTIME,TIMEL,STIME,LSTCP,LSTCL,LSTPH,LSTND)
(0137)  IF (STIME <= TTIME) GO TO 150
(0138)  N1 = IFILE(33)
(0139)  N2 = IFILE(34)
(0140)  CALL ABKLML(ISTND,LSTND,LSTCP,LSTCL,LSTCL,IIDM)
(0141)  *IRDM,IBDM,IUNIT,RUNN,PLACE,DTIME,STIME,LSTCT)
(0142)  STIME = STIME + DTIME
(0143)  RFILE(50) = STIME
(0144)  GO TO 100
(0145)  C  CALL OUTPUT FOR FINAL PRINT OUT IF INDICATED
(0146)  IF (NDOUT .EQ. 1) CALL OUTPUT(HDOUT,NODE,IP,ICL,ICP,RPATH,IRDM)
(0147)  *IIDM,RUNN,IFMT,PLACE,ARGN.IUNIT,LSTCL,LSTCT,LSTCP
(0148)  *LSTCL,LSTCL,IBDM,STIME,PCOST1)
(0149)  RETURN
(0150)  END

PROGRAM SIZE:  PROCEDURE - 000322  LINKAGE - 006007  STACK - 000070
0000 ERRORS [<Sweep>FTN-REV16.6]
C******************************************************************************
C******************************************************************************
C******************************************************************************
0161   SUBROUTINE SETARR(IHEAD,H1,H2,UNIT,IRDM,IBDM,IIDM)
0162   IMPLICIT REAL*8(A-H,O-Z)
0163   C THIS WILL RESET THE VALUES IN THE ACCUM AND STATE ARRAYS FOR
0164   C USE WITH NORMAL AND ABNORMAL
0165   
0166   C
0167   C
0168   C
0169   C
0170   C    IHEAD = INDICATOR OF HEADING FOR NORMAL POPULATION OUTPUT*
0171   C    H1 = ROW IN STATE TO SET PRESENT POPULATION FROM*
0172   C    H2 = ROW IN STATE TO SET PRESENT QUALITY FROM*
0173   C    IRDM = DIMENSION OF RPATH AND ACCUM*
0174   C    IUNIT = FORTRAN OUTPUT UNIT DEVICE NUMBER*
0175   C    IBDM = CLASS/COMPONENT BLOCK SIZE*
0176   C    IIDM = IPATH DIMENSION*
0177   C******************************************************************************
0178   COMMON/I2/FILE(100)
0179   COMMON/R1/ACCUM(50)
0180   COMMON/R4/STATE(20,30)
0181   C
0182   DIMENSION RPATH(750),IPATH(15)
0183   C******************************************************************************
0184   C INITIALIZE ARRAYS AND VARIABLES*
0185   C******************************************************************************
0186   NREFP = IFILE(13)
0187   LTHST = IFILE(24)
0188   CALL PATH(K,NREFP,IPATH,IIDM,RPATH,IRDM)
0189   C******************************************************************************
0190   C RESET ACCUM ARRAY*
0191   C******************************************************************************
0192   DO 100 I = 1,IRDM
0193   ACCUM(I) = 0.00
0194  100   CONTINUE
0195   C******************************************************************************
0196   C RESET STATE ARRAY*
0197   C******************************************************************************
0198   DO 300 J = 1,LTHST
0199      STATE(5,J) = STATE(N1,J)
0200      STATE(6,J) = 0.00
0201      STATE(7,J) = STATE(N2,J)
0202      STATE(8,J) = 0.00
0203      DO 200 K = 9,20
0204      STATE(K,J) = 0.00
0205  200   CONTINUE
0206   300   CONTINUE
0207   IHEAD = 1
C //INITIALIZE IN USE PATH ON DISK.

C //INITIALIZE IN USE PATH ON DISK.

DO 500 I = 1, LNHST

IRP = (I-1)*IIDM + 1
IRP2 = IRP + 1
RPATH(IRP) = STATE(5, I)
RPATH(IRP2) = STATE(7, I)
CALL PATOUT(HREFP, IPATH, IIDM, RPATH, IRDM)

500 CONTINUE

RETURN

END

PROGRAM SIZE: PROCEDURE - 000223 LINKAGE - 005763 STACK - 000042
0000 ERRORS [SETARR]FTN-REV16.6]
SUBROUTINE ABHRLM(LSTDH,LSTND,LSTCP,LSTCT,ISM,ISM,IRDM,IBDM,IRUNH,PLACE,DT,STIME,LSTCT,LSTCT,ISTRD,IRDM,IBDM,IRUNH,PLACE,DT,STIME,LSTCT,LSTCT)

IMPLICIT REAL*(A-H.O-Z)

** ABHRLM WILL MAKE THREE TIME SIMULATION SWEEPS THROUGH THE NO-**
** DAL ARRAY TO INTERPRET POPULATION AND QUALITY TRENDS. ABHRLM**
** IS A CONDENSED FORM OF THE SUBROUTINE NORMAL WITH NO COST**
** COMPUTATION AND EXTRANEOUS COMMENTS AND SUBROUTINE CALLS ELIMINATED.**

** ARGUMENT LIST **
(SEE IFILE & RFILE FOR MORE DETAIL)

ISTND - FIRST NODE
LSTND - LAST NODE
ISTCP - FIRST COMPONENT
LSTCP - LAST COMPONENT
ISTCL - FIRST CLASS
LSTCL - LAST CLASS
IIDM - DIMENSION OF IPATH
IRDM - DIMENSION OF RPATH
IBDM - DIMENSION OF CLASS/COMPONENT BLOCK
IRUNH - FORTRAN PRINT DEVICE NUMBER
RUNH - RUN NUMBER
PLACE - GROUP RUNNING PROGRAM
DT - TIME STEP IN YEARS
STIME - CURRENT SIMULATION TIME
ISTCT - NUMBER OF COLUMNS FROM REFERENCE TO FIRST COST
IHRM - COLUMN
LSTCT - NUMBER OF COLUMNS FROM FIRST TO LAST COST CO-

COMMON/I2/IFILE(100)
COMMON/I13/IFLAG(10)
COMMON/I17/INODE(6,300)

COMMON/R1/ACCUM(750)
COMMON/R2/FUNCT(10,10)
COMMON/R3/RNODE(30,300)
COMMON/R4/STATE(20,30)
COMMON/R5/RFILE(100)
COMMON/R6/EWIL(3,30)

DIMENSION IPATH1(15),IPATH2(15),IPATH3(15)
DIMENSION RUNH(10),IPATH1(750),IPATH2(750),IPATH3(750)
DIMENSION RPATH(750),PLACE(10)
C SET CONTROL INDICES.

IDSIM = IFILE(27)
OUTLD = RFILE(6)
IFMTS = IFILE(28)
TIMSTP = 0.00
NRPTH = IFILE(13)
DMP = 0.5
LNTHST = IFILE(21)

C BEGIN TIME SIMULATIONS

DO 2000 HPASS = 9,17,4
DMP1 = DMP * DT
CALL DRATE(DMP1, HPASS)
RFILE(50) = STIME + DMP1
DO 1900 IROW = ISTD, LSTD
IFLAG(4) = IROW
IP1 = INODE(1, IROW)
IP2 = INODE(2, IROW)
IP3 = INODE(3, IROW)
CALL PATHIN(IP1, IPATH1, IDM, RPATH1, IRDM)
CALL PATHIN(IP2, IPATH2, IDM, RPATH2, IRDM)
CALL PATHIN(IP3, IPATH3, IDM, RPATH3, IRDM)

C SET NODE DEPENDENT INDICES

IBRH = INODE(4, IROW)
ICDEF = INODE(5, IROW)
IREFP = INODE(6, IROW)

C SET PATH DEPENDENT INDICES FOR THIS NODE

IPDT1 = IABS(IPATH1(11))
IPDT2 = IABS(IPATH2(11))
IPDT3 = IABS(IPATH3(11))

C LOOP THROUGH CLASSES

DO 1800 ICL = ISTCL, LSTCL

C
C LOOP THROUGH COMPONENTS.
DO 1700 ICP = ISTCP, LSTCP
CALL RCOLUMN (ICL, LSTCL, ICP, LSTCP, IRP, IRCOL, IBDM)
C*************************************************************************
C CALCULATE FLOWS AND CHANGES.
C*************************************************************************
IF(IPDT1 .EQ. 1) CALL PDDT (IRP, IRCOL, RPATH1, 
   IRDM, DTMP, ISGN1, NPASS) 
IF(IBRSH .EQ. 1) CALL BRANCH (TCDEF, TREP, CDE, 
   IRP, RPATH1, RPATH2, RPATH3, IRDM, 
   NPASS, NRPTH, IP1) 
IF(IPDT2 .EQ. 1) CALL PDDT (IRP, IRCOL, RPATH2, 
   IRDM, DTMP, ISGN2, NPASS) 
IF(IQDT2 .EQ. 1) CALL QDDT (IRP, IRCOL, RPATH2, 
   IRDM, DTMP, ISGN2, NPASS) 
IF(IBRSH .EQ. 0) CALL SUMS (IMFG1, IMFG2, IRP, 
   RPATH1, RPATH2, RPATH3, IRDM) 
IF(IPDT3 .EQ. 1) CALL PDDT (IRP, IRCOL, RPATH3, 
   IRDM, DTMP, ISGN3, NPASS) 
IF(IQDT3 .EQ. 1) CALL QDDT (IRP, IRCOL, RPATH3, 
   IRDM, DTMP, ISGN3, NPASS) 
1700 CONTINUE
1800 CONTINUE
1900 CONTINUE
CALL WEIBUL (ISTCP, ISTCL, LSTCP, LSTCL, NPASS, IBDM, DT)
TIMSTP = DTMP + TIMSTP
DTMP = TIMSTP
2000 CONTINUE
IF (RFILE(5) .LT. OUTLMD) GO TO 10
RFILE(5) = 0.0
IF (10SIM .EQ. 3) CALL OUTPT (10SIM, NODE, IP, ICL, ICP, RPATH, 
   IRDM, IDM, RUNN, IFNTS, PLACE, ARGN, IUNIT, 
   ISTCT, LSTCT, ISTCP, LSTCP, ISTCL, LSTCL, IBDM, 
   STIME, PCOST) 
2010 RFILE(5) = RFILE(5) + 1.0
DO 2100 I = 1, LNGTHST
   RCPI = STATE (6, I)
   RCQ1 = STATE (8, I)
   RCQ2 = STATE (12, I)
   RCQ3 = STATE (16, I)
   RCQ4 = STATE (18, I)
   RCQ4 = STATE (20, I)
   RCPF = (RCPI + 2.0*RCQ2 + 2.0*RCQ3 + RCQ4)/6.000
2100 CONTINUE
"
(0370)  \[ RCQF = \frac{RCQ1 + 2.0 \cdot RCQ2 + 2.0 \cdot RCQ3 + RCQ4}{6.00} \]
(0371)  \[ \text{STATE}(17, I) = \text{STATE}(5, I) + (DT \cdot \text{RCPF}) \]
(0372)  \[ \text{STATE}(19, I) = \text{STATE}(7, I) + (DT \cdot \text{RCQF}) \]
(0373)  2100 CONTINUE
(0374)  RETURN
(0375)  END

PROGRAM SIZE:  PROCEDURE - 001067  LINKAGE - 027716  STACK - 000100
0000 ERRORS [\textless ABRML\textgreater FTN-REV16.6]
C GET CONTROL INDICES

ICOST = IFILE(35)
NPASS = IFILE(19)
NRPTH = IFILE(13)
IOUT = IFILE(15)
PCOST1 = 0.0
IYAL = 0.0

C CHECK FOR PRINT TIME.

IPRINT = 0.0
IF RFILe(24) LT RFILE(25) GO TO 155
IPRINT = IFILE(25)
RFILE(24) = 0.0
RFILE(24) = RFILE(24) + 1.00

155 RFILE(24) = RFILE(24) + 1.00

C BEGIN NODE SWEEP.

C START NORMAL CALCULATIONS

DO 1900 IIR0 = 1STND, LSTND
    FORMAT('NODE:', I4)
    IFLAG(4) = IIR0
    IP1 = INODE(4, IIR0)
    IP2 = INODE(2, IIR0)
    IP3 = INODE(3, IIR0)
    CALL PATHINC(IP1, IPATH1, IIDM, RPATH1, IRDM)
    CALL PATHINC(IP2, IPATH2, IIDM, RPATH2, IRDM)
    CALL PATHINC(IP3, IPATH3, IIDM, RPATH3, IRDM)
    IF (IP1 .NE. IPATHK1) CALL ERRORS(IP1)
    IF (IP2 .NE. IPATH2) CALL ERRORS(IP2)
    IF (IP3 .NE. IPATH3) CALL ERRORS(IP3)

C SET NODE CALL INDICES

C SET PATH DEPENDENT CALL INDICES FOR THIS NODE.

IFG1 = IPATH1(13)
IFG2 = IPATH2(13)
IFG3 = IPATH3(13)
ICST1 = IPATH1(10) * IFG1 * ICOST
ICST2 = IPATH2(10) * (IPRINT + IFG2) * ICOST
ICST3 = IPATH3(10) * ICOST
IFCT1 = ICST1
IFCT2 = ICST2
CALLS FOR PATH 1

IF (IBRSH .EQ. 1) CALL BRANCH (ICDF, IREFP, CDE, IPR, 
                   RPATH1, RPATH2, RPATH3, IRDM, IBDM, 
                   NPASS, NRPATH, IP1)

IF (IBRSH .EQ. 0) CALL SUMS (IMFG1, IMFG2, IPR, RPATH1, 
                   RPATH2, RPATH3, IRDM)

IF (IFCT1 .EQ. 1) CALL FUNCTS (IPR, ISTC, LSTCT, 
                   RPATH1, IRDM, IP, RFC1, IBDM, NPASS, 
                   IUNIT, IVAL, IPC, ICL)

IF (IPDT1 .EQ. 1) CALL PDOT (IPR, IRCOL, RPATH1, IRDM, 
                   DT, ISGN1, NPASS)

IF (IQDT1 .EQ. 1) CALL QDOT (IPR, IRCOL, RPATH1, IRDM, 
                   DT, ISGN1, NPASS)

IF (ICST1 .EQ. 1) CALL PCOSTS (IPR, ISTC, LSTCT, 
                   RPATH1, PCOST, RPATH, IRDM)

IF (IACM1 .EQ. 1) CALL ACCUME (IPR, ISTC, LSTCT, RPATH, 
                   IRDM)

IF (IPRT1 .EQ. 1) CALL OUTPT (NDOUT, IROW, IP1, ICL, 
                   ICP, RPATH1, IRDM, IBDM, RUNN, IHEAD, 
                   PLACE, ARGN, IUNIT, ISTCT, LSTCT, 
                   ISTCP, LSTCP, ISTCL, LSTCL, IBDM, 
                   STIME, PCOST1)

CALLS FOR PATH 2

IF (IBRSH .EQ. 1) CALL BRANCH (ICDF, IREFP, CDE, IPR, 
                   RPATH1, RPATH2, RPATH3, IRDM, IBDM, 
                   NPASS, NRPATH, IP1)

IF (IBRSH .EQ. 0) CALL SUMS (IMFG1, IMFG2, IPR, RPATH1, 
                   RPATH2, RPATH3, IRDM)

IF (IFCT2 .EQ. 1) CALL FUNCTS (IPR, ISTC, LSTCT, 
                   RPATH2, IRDM, IP, RFC2, IBDM, NPASS, 
                   IUNIT, IVAL, IPC, ICL)

IF (IPDT2 .EQ. 1) CALL PDOT (IPR, IRCOL, RPATH2, IRDM, 
                   DT, ISGN2, NPASS)

IF (IQDT2 .EQ. 1) CALL QDOT (IPR, IRCOL, RPATH2, IRDM, 
                   DT, ISGN2, NPASS)

IF (ICST2 .EQ. 1) CALL PCOSTS (IPR, ISTC, LSTCT, 
                   RPATH2, PCOST, RPATH, IRDM)

IF (IACM2 .EQ. 1) CALL ACCUME (IPR, ISTC, LSTCT, RPATH, 
                   IRDM)

IF (IPRT2 .EQ. 1) CALL OUTPT (NDOUT, IROW, IP2, ICL, 
                   ICP, RPATH2, IRDM, IBDM, RUNN, IHEAD, 
                   PLACE, ARGN, IUNIT, ISTCT, LSTCT, 
                   ISTCP, LSTCP, ISTCL, LSTCL, IBDM, 
                   STIME, PCOST1)

CALLS FOR PATH 3

CALLS FOR PATH 1

IF (IBRSH .EQ. 1) CALL BRANCH (ICDF, IREFP, CDE, IPR, 
                   RPATH1, RPATH2, RPATH3, IRDM, IBDM, 
                   NPASS, NRPATH, IP1)

IF (IBRSH .EQ. 0) CALL SUMS (IMFG1, IMFG2, IPR, RPATH1, 
                   RPATH2, RPATH3, IRDM)

IF (IFCT1 .EQ. 1) CALL FUNCTS (IPR, ISTC, LSTCT, 
                   RPATH1, IRDM, IP, RFC1, IBDM, NPASS, 
                   IUNIT, IVAL, IPC, ICL)

IF (IPDT1 .EQ. 1) CALL PDOT (IPR, IRCOL, RPATH1, IRDM, 
                   DT, ISGN1, NPASS)

IF (IQDT1 .EQ. 1) CALL QDOT (IPR, IRCOL, RPATH1, IRDM, 
                   DT, ISGN1, NPASS)

IF (ICST1 .EQ. 1) CALL PCOSTS (IPR, ISTC, LSTCT, 
                   RPATH1, PCOST, RPATH, IRDM)

IF (IACM1 .EQ. 1) CALL ACCUME (IPR, ISTC, LSTCT, RPATH, 
                   IRDM)

IF (IPRT1 .EQ. 1) CALL OUTPT (NDOUT, IROW, IP1, ICL, 
                   ICP, RPATH1, IRDM, IBDM, RUNN, IHEAD, 
                   PLACE, ARGN, IUNIT, ISTCT, LSTCT, 
                   ISTCP, LSTCP, ISTCL, LSTCL, IBDM, 
                   STIME, PCOST1)
(0611) C * IF(IBRSH .EQ. 1) CALL BRANCH(ICDEF, IREFP, CDE, IRP, RPATH1, RPATH2, RPATH3, IRDM, IBDM, HPASS, IRPTH, IP1)
(0613) C * IF(IBRSH .EQ. 0) CALL SUMS(IMFG1, IMFG2, IRP, RPATH1, RPATH2, RPATH3, IRDM)
(0614) C * IF(IFCT3 .EQ. 1) CALL FUNCTS(IRP, ISTCT, LSTCT, RPATH3, IRDM, IP, RFC3, IBDM, HPASS, IUNIT, IVAL, IP, ICL)
(0616) C * IF(IPDT3 .EQ. 1) CALL POUTCIRP, IRCOL, RPATH3, IRDM, DT, ISGN3, HPASS)
(0619) C * IF(IDT3 .EQ. 1) CALL QDOTCIRP, IRCOL, RPATH3, IRDM, DT, ISGN3, HPASS)
(0621) C * IF(IACM3 .EQ. 1) CALL ACME(IRQ, ISTCT, LSTCT, RPATH3, IRDM)
(0626) C * IF(IPRT3 .EQ. 1) CALL OUTPT(NDOUT, IROW, IP3, ICL, ICP, RPATH3, IRDM, IBDM, RUNN, RHEAD, PLACE, ARGN, IUNIT, ISTC, LSTC, IACM3, IBDM, PCOST, IDM)
(0631) C * IF(IPRT3 .EQ. 1) CALL ADDRCH(RPATH3, RFC3, IRP, ISTCT, LSTCT, IRDM)
(0637) C /////////////////////////////////////////////////////////////////////////////////////////////
(0638) RFC3(1) = 1.0
(0639) IF(IOUT .EQ. 1) CALL ADDRCH(RPATH2, RFC2, IRP, ISTCT, LSTCT, IRDM)
(0640) * IF(IOUT .EQ. 1) CALL PATOUT(IP2, IPATH2, IBDM, RPATH2, IRDM)
(0644) * IF(IOUT .EQ. 1) CALL ADDRCH(RPATH3, RFC3, IRP, ISTCT, LSTCT, IRDM)
(0645) * IF(IOUT .EQ. 1) CALL PATOUT(IP3, IPATH3, IBDM, RPATH3, IRDM)
(0646) * IF(IOUT .EQ. 1) WRITE(IUNIT, 352) PCOST1
(0648) C /////////////////////////////////////////////////////////////////////////////////////////////
(0649) 1900 CONTINUE
(0650) 352 FORMAT(1X,'64(1H-)',/,'45X,GHTOTOL $',112)
(0651) PCOST1 = 0.0
(0652) CALL WEIBUL(IACM3, IP, LSTC, IBDM, DT)
(0653) RETURN
(0654) END

PROGRAM SIZE: 19800  PROCEDURE = 002074  LINKAGE = 051306  STACK = 000100
0000 ERRORS [[NORMAL>FTH-REV16.6]]
C****************************************************************************************************
C****************************************************************************************************
C****************************************************************************************************
SUBROUTINE WEBUL(IETCP,ISTCL,LSTCP,LSTCL,NPASS,IBDM,DT)
IMPLICIT REAL*8,0-Z
CWEBUL IS USED TO COMPUTE AND APPLY THE WEIBULL DISTRIBUTION
TO THE CHANGE IN SYSTEM QUALITY.

ARGUMENT LIST

ISTCP - THE FIRST COMPONENT
ISTCL - THE FIRST CLASS NUMBER
LSTCP - THE NUMBER OF COMPONENTS
LSTCL - THE NUMBER OF CLASSES
HPASS - THE STATE ARRAY REFERENCE ROW
IBDM - CLASS/COMPONENT BLOCK SIZE
DT - TIME SIMULATION INCREMENT

COMMON/R4/STATE(20,30)
COMMON/R6/WEBUL(3,30)

C
N2 = HPASS + 1
N3 = HPASS + 2
N4 = HPASS + 3
CLOOP THROUGH CLASSES AND COMPONENTS.
DO 100 ICL = ISTCL,LSTCL
DO 50 ICP = IETCP,LSTCP
RCLUDM(IP,C,LSTCL,ICL,LSTCP,IRP,IRCOL,IBDM)
CCONVERT CHANGE IN BAD TO CHANGE IN QUALITY.
QNOW = STATE(N3,IRCOL)
POP = STATE(NPASS,IRCOL)
DBAD = STATE(N4,IRCOL)
DPOP = STATE(N2,IRCOL)
IF(POP .EQ. 0.0) GO TO 50
DG = DBAD/POP * QNOW*DPOP/POP
C
CCALCULATE WEIBULL EFFECT ON QUALITY.
ANOW = WEBUL(1,IRCOL)
WBSLP = WEBUL(2,IRCOL)
ALF = WEBUL(3,IRCOL)
IF(CHLF .EQ. 0.0) CHLF = 1.0E-10
H2RD = (WBSLP/CHLF) * (ANOW/CHLF)**(WBSLP-1.0)
DG = DG + H2RD*(1.0 - QNOW)
STATE(N4,IRCOL) = DG

100 CONTINUE
50 CONTINUE
(0705) 50 CONTINUE
(0706) 100 CONTINUE
(0707) RETURN
(0708) END

PROGRAM SIZE: PROCEDURE - 000257
0000 ERRORS [<WEIBUL>FTN-REV16.6]

LINKAGE - 000111
STACK - 000046
C**********************************************************************
C**************************************************************************
C                                                                
C SUBROUTINE DATAIN(ISTDH,LSTHD,ISTPH,LSTPH,ISTCP,LSTCP,
C * ISTCL,LSTCL,LSTFL,ISTCT,LSTCT,ITDM,IRDM,
C * IDNM,HDOUT,UNIT,ICODE,NREFP,PLACE,
C * RHN)
C**************************************************************************
IMPLICIT REAL*8(A-H,O-Z)
C**************************************************************************
C DATAIN READS DATA INTO ARRAYS FROM DATA FILES THAT HAVE BEEN
C OPENED FOR READING PRIOR TO RUNNING THE PROGRAM. IT THEN IN-
C ITIALIZES SEVERAL CONTROL VARIABLES WITH VALUES THAT HAVE BEEN
C READ INTO THE IFILE ARRAY.
C**************************************************************************
C ARGUMENT    DESCRIPTION
C             (SEE RFIL AND IFILE FOR DETAIL)
C ISTDH       FIRST NODE
C LSTND       LAST NODE
C ISTPH       FIRST PATH
C LSTPH       LAST PATH
C ISTCP       FIRST COMPONENT
C LSTCP       LAST COMPONENT
C ISTCL       FIRST CLASS
C LSTCL       LAST CLASS
C ISTCT       FIRST COST
C LSTCT       LAST COST
C ITDM        DIMENSION OF PATH INTEGER NUMBERS
C IRDM        DIMENSION OF PATH REAL NUMBERS
C IBDM        DIMENSION OF THE COST-COMPONENT BLOCK
C IDNM        FORTRAN UNIT DEVICE NUMBER
C ICODE       -
C**************************************************************************
COMMON/I1/ERROR(10)
COMMON/12/IFILE(100)
COMMON/13/IFLAG(10)
COMMON/17/INODE(6,300)
COMMON/R1/ACCUM(750)
COMMON/R2/FUNCT(10,10)
COMMON/R3/RNODE(30,300)
COMMON/R4/STATE(20,30)
COMMON/R5/RFILE(100)
COMMON/R6/UEIBL(3,30)
COMMON/R7/REVISN(8),SCHMD(8),DATAFL(8)
DIMENSION RHN(10),PLACE(10)
C**************************************************************************
C READ IN DATA FROM FILES.
C**************************************************************************
READ(8,199) (REVISN(J),J=1,7)
(0806) C*********************************************************************
(0807) C*********************************************************************
(0808) C*********************************************************************
(0809) SUBROUTINE ERRORS(IER)
(0810) C
(0811) COMMON/I1/IERROR(10)
(0812) COMMON/I2/IFILE(100)
(0813) COMMON/I3/IFLAG(10)
(0814) C
(0815) ITUNIT = IFILE(23)
(0816) 100 FORMAT(17ERROR LOCATED AT ,1015/)
(0817) N = 0
(0818) 200 DO 400 I = 1,10
(0819)  IF(N .EQ. 1) GO TO 400
(0820)  IF(IERROR(I) .EQ. 0) GO TO 300
(0821)  GO TO 400
(0822) 300 IERROR(I) = IER
(0823)  WRITE(ITUNIT,100) IERROR(I)
(0824)  N = 1
(0825) 400 CONTINUE
(0826) RETURN
(0827) END

PROGRAM SIZE: PROCEDURE - 000102 LINKAGE - 000037 STACK - 000020
0000 ERRORS [ERRORS]FTN-REV16.6]
(0878) 999 RETURN
(0879) END
PROGRAM SIZE: PROCEDURE - 000777
0000 ERRORS [<TRACE >FTH-REV16.6]
LINKAGE - 000047 STACK - 000074
C***SUBROUTINE PATHIN(IFILE, IPATH, IIDM, RPATH, IRDM)

C***IMPLICIT REAL*8(A-H,0-Z)

COMMON/I0/IARRAY(15,600)
COMMON/R8/RARRAY(750,600)
DIMENSION IPATH(15), RPATH(750)

DO 100 I = 1, IIDM
   IPATH(I) = IARRAY(I,IFILE)

100 CONTINUE

DO 200 I = 1, IRDM
   RPATH(I) = RARRAY(I,IFILE)

200 CONTINUE

RETURN

END

PROGRAM SIZE:  PROCEDURE - 000073   LINKAGE - 000027   STACK - 000034
0000 ERRORS [(PATHIN)FTN-REV16.6]
SUBROUTINE PATOUT(IFILE, IPATH, IIDM, RPATH, IRDM)
IMPLICIT REAL*8(A-H,O-Z)
COMMON/I8/IARAY(15,600)
COMMON/R8/RARAY(750,600)
DIMENSION IPATH(15), RPATH(750)
DO 100 I = 1, IIDM
   IARAY(I,IFILE) = IPATH(I)
100 CONTINUE
DO 200 I = 1, IRDM
   RARAY(I,IFILE) = RPATH(I)
200 CONTINUE
RETURN
END

PROGRAM SIZE:  PROCEDURE - 000075   LINKAGE - 000027   STACK - 000034
0000 ERRORS [(PATOUT)]
C**************************************************************
C**************************************************************
C SUBROUTINE BRANCH(NODI, CD, I, RPATH1, RPATH2, RPATH3, IRDM,
C IBDM, NPATH, NPATH, IPH)
C**************************************************************
IMPLICIT REAL*8(A-H.O-Z)
C**************************************************************
C BRANCH WILL DETERMINE THE OUT GOING FLOWS AND QUALITIES ON A BRANCH.
C NODE A BRANCH NODE WILL INVOLVE A SIMPLE OR A DEFECTIVE DECISION.
C DEPENDING UPON THE TYPE OF DECISION THE CALCULATIONS WILL BE WORKED.
C BY ONE OF TWO BLOCKS. EACH BLOCK REQUIRES A NODE SPLIT FRACTION. TO
C DETERMINE THE REFERENCE PATH FLOW, THE FRACTION IS SENT FROM NORMAL.
C AND THEN COFUNC IS CALLED TO ALTER THE FRACTION AS NECESSARY. AFTER
C FLOWS AND QUALITIES ARE COMPUTED, A REWORK CHECK IS MADE. IF THIS
C IS THE CASE, QUALITY IS SET TO ZERO ON THE APPROPRIATE PATHS.
C**************************************************************

C (SEE IFILE AND RFILE FOR COMPLETE DESCRIPTIONS)

C 1 = INDICATOR OF SIMPLE OR DEFECTIVE DECISION
C NODI = REFERENCE PATH INDICATOR
C CD = NODE SPLIT FRACTION
C I = REFERENCE COLUMN INDEX OF FLOW
C RPATH1 = NODE INCOMING PATH NUMBERS
C RPATH2 = NODE OUTGOING PATH NUMBERS
C RPATH3 = NODE OUTGOING PATH NUMBERS
C IRDM = DIMENSION OF PATH REAL NUMBERS VECTOR
C COMMON/I2/IFILE(100)
C COMMON/I3/IFLAG(10)
C COMMON/R5/RFILE(100)
C COMMON/R5/RFILE(100)
C DIMENSION RPATH1(750), RPATH2(750), RPATH3(750)
C C INITIALIZE CONTROL INDICES AND VARIABLES.
C K = I + 1
C L = I + 4
C ADDR = 0.0
C CDTEMP = CD
C C COMPUTE PROPER NODE SPLIT FRACTION AND CD TO APPROPRIATE BLOCK
C IF(NODI.EQ.0) ADDR = RPATH2(I3)
C IF(NODI.EQ.1) ADDR = RPATH3(I3)
C IF (ADDR .LT. 10.0*47) AND (ADDR .GE. 10.0**20)
C *CALL COFUNC, RPATH, IRDM, CD, IBDM, NPATH, NPATH, IPH, ADDR
C IF(J.GT.1)CALL ERRORS(1005)
IF(J .EQ. 1)GO TO 200
C//SIMPLE SPLIT DECISION.
C//IF(NOD1 .EQ. 1) 
RPATH2(I) = RPATH1(I) * CD
RPATH3(I) = RPATH1(I) - RPATH2(I)
IF(RPATH3(I) .LT. 0.0) CALL ERRORS(1010)
RPATH2(K) = RPATH1(K)
RPATH3(K) = RPATH1(K)
IF(IFLAG(1) .EQ. IFLAG(4)) CALL TRACE(1010,
* I, J, K, L, IRDM,
* CD, RPATH1(I), RPATH1(K), RPATH1(L), RPATH3(I),
* RPATH3(K), RPATH3(L))
GO TO 300
C//DEFECTIVE BASED BRANCH DECISION.
200 BAD1 = RPATH1(I) * RPATH1(K)
GOOD1 = RPATH1(I) - BAD1
BAD2 = BAD1 * CD
GOOD2 = 0.
ALL2 = BAD2 + GOOD2
BAD3 = BAD1 - BAD2
GOOD3 = GOOD1 + GOOD2
ALL3 = BAD3 + GOOD3
IF(ALL2 .EQ. 0.0) GO TO 235
IF(ALL2 .LE. 0.0) CALL ERRORS(1020)
QUAL2 = BAD2/(BAD2 + GOOD2)
GO TO 240
235 QUAL2 = 0
GO TO 245
IF(ALL3 .EQ. 0.0) GO TO 245
IF(ALL3 .LE. 0.0) CALL ERRORS(1030)
QUAL3 = BAD3/(BAD3 + GOOD3)
GO TO 250
245 QUAL3 = 0
250 IF(NOD1 .EQ. 1) GO TO 260
RPATH2(I) = ALL2
RPATH3(I) = ALL3
RPATH2(K) = QUAL2
RPATH3(K) = QUAL3
GO TO 275
RPATH2(I) = ALL3
RPATH3(I) = ALL2
RPATH2(K) = QUAL3
RPATH3(K) = QUAL2
275 IF(IFLAG(1) .EQ. IFLAG(4)) CALL TRACE(1030,
* I, J, K, L, IRDM,
* CD, BAD1, GOOD1, BAD2, GOOD2,
* BAD3, GOOD3, QUAL2, QUAL3, 0.0)
C RESET QUALITY TO ZERO IF REWORK INDICATOR = 1

IF(RPATH2(L) .GE. 1) RPATH2(K) = 0.0
IF(RPATH2(L) .LT. 0) CALL ERRORS(1040)
IF(RPATH3(L) .GE. 1) RPATH3(K) = 0.0
IF(RPATH3(L) .LT. 0) CALL ERRORS(1050)

IF(IFLAG .EQ. IFLAG(4)) CALL TRACE(1050).
* I, J, K, L, IRDM,
* CD, RPATH1(I), RPATH1(K), RPATH1(L), RPATH2(I),
* RPATH2(K), RPATH2(L), RPATH3(I), RPATH3(K), RPATH3(L)
CD = CDTEMP

END
275 QUAL3 = 0.0
C ASSIGN COMBINED FLOW AND QUALITY TO OUT GOING PATH.
C
250 F3(I) = FLOW3
F3(K) = QUAL3
IF(IFLAG(I) .EQ. IFLAG(4)) CALL TRACE(1060,
* I, 0, K, L, IRDM,
* 0.0, BAD1, GOOD1, BAD2, GOOD2,
*BAD3, GOOD3, 0.0, QUAL3, 0.0)
C
RETURN
END

PROGRAM SIZE: PROCEDURE - 000257 LINKAGE - 000075 STACK - 000046
0000 ERRORS [<SUHS >FTN-REV16.6]
**Source Code**

```
(1099) COMMON/12/I FILE(100)
(1100) COMMON/13/IFLAG(10)
(1101) COMMON/R4/STATE(20,30)
(1102) COMMON/R5/RFILE(100)
(1103) C
(1104) C DIMENSION RPATH(750)
(1105) C///////////
(1106) C SET CONTROL INDICES.
(1107) C///////////
(1108) J = I + 1
(1109) N2 = NPASS + 1
(1110) N3 = NPASS + 3
(1111) C CALCULATE TOTAL AND BAD POPULATION CHANGES.
(1112) C///////////
(1113) DP = RPATH(I) * DBLE(FLOAT(ISGN))
(1114) DB = DP * RPATH(J)
(1115) STATE(N2,IRCOL) = STATE(N2,IRCOL) + DP
(1116) STATE(N3,IRCOL) = STATE(N3,IRCOL) + DB
(1117) IF(IFLAG(1) .EQ. IFLAG(4)) CALL TRACE(1122,
(1118) * I, J, IRCOL, IRCOL, IRROW, IRCOL,
(1119) * DB, RPATH(I), RPATH(J),, DG, DP,
(1120) * DT, STATE(2,IRCOL), 0.00 00, 0.00 00, 0.00 00)
(1121) C
(1122) RETURN
(1123) END
```
PROGRAM SIZE:  PROCEDURE - 000175
0000 ERRORS [<PDCT >FTN-REV16.6] LINKAGE - 000061 STACK - 000050
C******************************************************************************
C******************************************************************************
C SUBROUTINE QDOT(I,IRCOL,RPATH,IRDM,DT,ISGN,NPASS)
C******************************************************************************
C IMPLICIT REAL*B(A-H,O-Z)
C******************************************************************************
C QDOT WILL CALCULATE THE CHANGE IN THE SYSTEM DEFECTIVE POPULATION* 
C FOR PATHS THAT ENTAIL A CHANGE IN QUALITY. THE VALUE OBTAINED IS * 
C THEN USED TO UPDATE THE CHANGE IN BAD IN THE STATE ARRAY FOR LA- * 
C USE.
C******************************************************************************

ARGUMENT LIST
(SEE IFILE AND RFILE FOR COMPLETE DESCRIPTIONS)

1 I = REFERENCE COLUMN INDEX OF REAL VALUES
2 IRCOL = COLUMN INDEX OF REAL NODE REFERENCE POSITION
3 RPATH = VECTOR CONTAINING PATH VALUES
4 IRDM = DIMENSION OF PATH REAL NUMBERS
5 DT = TIME STEP
6 ISGN = CHANGE SIGN
7 NPASS = STATE ARRAY REFERENCE ROW

COMMON/I2/IFILE(100)
COMMON/I3/IFLAG(10)

COMMON/R4/STATE(20,30)
COMMON/R5/RFILE(100)

DIMENSION RPATH(750)

C SET CONTROL INDICES.
J = I + 1
N2 = NPASS + 1
N3 = NPASS + 3

C CALCULATE BAD POPULATION CHANGES.
DB = RPATH(I)*RPATH(J)
N3 = NPASS + 3
STATE(N3,IRCOL) = STATE(N3,IRCOL) - DB

IF(IFLAG(1) .EQ. IFLAG(4))CALL TRACEX(114,
* I, J, IRCOL, IRDM, DT, ISGN, NPASS)

IF(IFLAG(1) .EQ. IFLAG(4))CALL TRACEX(114,
* DB, DBG, DB, RPATH(I), RPATH(J),
* 0.00 00, 0.00 00)

RETURN
END
PROGRAM SIZE: 000145
0000 ERRORS [QDOT] FTN-REV16.6
LINKAGE: 000065
STACK: 000044
SUBROUTINE RCOLUMN (ICL, LSTCL, ICP, LSTCP, IRP, IRCOL, IBDM)

IMPLICIT REAL*8(A-H,O-Z)

C SETS COLUMN(IRP) INDEX VALUE FOR REAL PATH VECTORS

C ARGUMENT

1 ICL = PRESENT CLASS INDEX
2 LSTCL = LAST CLASS INDEX
3 ICP = PRESENT COMPONENT INDEX
4 LSTCP = LAST COMPONENT INDEX
5 IRP = REFERENCE COLUMN OF RPATH
6 IRCOL = REFERENCE COLUMN OF RNODE AND STATE
7 IBDM = DIMENSION OF A SINGLE COST-COMPONENT

IRP = IBDM*(LSTCP*(ICL-1)+(ICP-1))+1
IRCOL = (LSTCP*(ICL-1)+(ICP-1))+1
RETURN

END

PROGRAM SIZE: PROCEDURE - 000030 LINKAGE - 000020 STACK - 000040
0000 ERRORS [RCOLUMN.FTN-REV16.6]
C*****************************************************************************
C*****************************************************************************
SUBROUTINE PCOSTS(I,J,K,PATH,PCOST,RPATH,IRDH)
IMPLICIT REAL*8(A-H,O-Z)
C*****************************************************************************
C THIS SUBROUTINE WILL COMPUTE PATH COSTS BY COMPONENT AND SAVE*
C THEM IN THE RPATH ARRAY
C*****************************************************************************
C*****************************************************************************
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION RPATH(750), PATH(750)
C
L = I+J
II = I+1
I3 = I+3
K1 = L*K
USEQ = PATH(I3)
FLOW = PATH(I)
QUAL = PATH(I1)

DO 100 M = L,K1
   COST = FLOW*QUAL*PATH(M)*USEQ
   +FLOW*PATH(M)*(1.0 - USEQ)
   RPATH(M)=COST
   PCOST = PCOST+COST
100 CONTINUE
RETURN
END

PROGRAM SIZE:  PROCEDURE - 000127    LINKAGE - 000047    STACK - 000044
0000 ERRORS (PCOSTS) FTN-REV16.6
**DOCUMENT:**

```fortran
C**********************************************************************
C**********************************************************************
C SUBROUTINE ACCUME(I,J,LSTCT,RPATH,IRDM)
C**********************************************************************
C ACCUM SUMS UP THE COSTS OF ALL THE COMPONENTS FOR EACH COST CODE *
C AND STORES THE COMPUTED VALUES IN THE ACCUM ARRAY FOR LATER USE. *
*(SEE IFIX AND RFIX FOR COMPLETE DESCRIPTION)*
C**********************************************************************
IMPLICIT REAL*(A-H,O-Z)
C COMMON/R1/ACCUM(750)
C COMMON/I3/IFLAG(10)
C DIMENSION RPATH(750)
C SET CONTROL INDICES
C
K = I+J
L = K+LSTCT
IF( IFLAG(1) .EQ. IFLAG(4) ) CALL TRACE(1560,
(274) I,J,K,L,IRDM,
* ACCUM(1),ACCUM(2),ACCUM(3),ACCUM(4),ACCUM(5),
* ACCUM(6),ACCUM(7),ACCUM(8),ACCUM(9),ACCUM(10))
C BEGIN SUMMATION LOOP.
DO 100 M=K,L
ACCUM(M) = ACCUM(M) + RPATH(M)
100 CONTINUE
C
C IF( IFLAG(1) .EQ. IFLAG(4) ) CALL TRACE(1580,
C I,J,K,L,IRDM,
* RPATH(1),RPATH(2),RPATH(3),RPATH(4),RPATH(5),
* RPATH(6),RPATH(7),RPATH(8),RPATH(9),RPATH(10))
C RETURN
END
```

**PROGRAM SIZE:** PROCEDURE - 00232  LINKAGE - 00073  STACK - 00054

0000 ERRORS [(ACCUME)FTH-REV16.6]
SUBROUTINE FUNCTS(IRP, ISTCT, LSTCT, RPATH, IRDM, IP, RFC, IBDM, 
  MPASS, IUNIT, IVAL, ICP, ICL)

IMPLICIT REAL*8(A-H,0-Z)

FUNCTIONS WILL FETCH INFORMATION STORED IN THE FUNCT ARRAY BY USING ADDRESSED STORED IN RPATH. THIS INFORMATION IS USED TO INDICATE WHICH COST FUNCTIONS ARE NEEDED TO CALCULATE THE VALUES FOR THE COST CODE IN THE CURRENT PATH. THE VALUES ARE THEN PUT IN THE CORRESPONDING RPATH COST COLUMNS. THE ADDRESSES IN THE COST COLUMNS ARE STORED BY ADDRCH BEFORE BEING OVERWRITTEN.

ARGUMENT LIST
(SEE RFILE AND IFILE FOR DESCRIPTIONS)

1   IRP = REFERENCE COLUMN INDEX IN RPATH
2   ISTCT = COLUMN OF FIRST COST ITEM IN RPATH
3   LSTCT = LAST COST INDEX COLUMN VALUE
4   RPATH = PATH VALUES
5   IRDM = DIMENSION OF RPATH VECTOR
6   IP = PATH NUMBER PRESENTLY COMPUTING ON
7   RFC = ARRAY FOR STORING RPATH FUNCT ADDRESSES
8   IBDM = CLASS/COMPONENT BLOCK SIZE
9   MPASS = STATE REFERENCE ROW
10  IUNIT = OUTPUT DEVICE
11  IVAL = OUTPUT INDICATOR
12  ICP = PRESENT COMPONENT
13  ICL = PRESENT CLASS OF OUTPUT

COMMON/I2/IFILE(100)
COMMON/I3/IFLAG(10)
COMMON/R2/FUNCT(10,10)
COMMON/R4/STATE(20,30)
COMMON/R5/RFILE(100)

DIMENSION RPATH(750),RFC(750)

C SET CTRL INDICES.
C STORE FUNCTION ADDRESSES TO BE USED.
C FUNCTION 5:
1050 21 = (DEXP(X-B) - DEXP(-B))/DEXP(X-B) + DEXP(B-K))
1056 22 = (DEXP(X-K0) - DEXP(X0-K))/DEXP(X-K0) + DEXP(X0-K))
1057 23 = (DEXP(-B) - DEXP(B))/DEXP(-B) + DEXP(B))
1058 24 = (DEXP(X-K0) - DEXP(X0))/DEXP(X-K0) + DEXP(X0))
1059 COST = (A*(1+21) + C*(1-22) + Y0)
1060 COST = COST/(A*(1+23) + C*(1-24) + Y0)
1061 GO TO 2000
C FUNCTION 6:
1065 IRC = (IRP + IBDM - 1)/IBDM
1066 QP = STATE(H2, IRC)
1067 COST = A * QP/B + C
1068 GO TO 2000
C REMAINING AVAILABLE FOR OTHER FUNCTIONS.
C
1080 COST = E*8.0
1081 GO TO 2000
C
1090 COST = F*9.0
1091 GO TO 2000
C
C TAKE PRODUCT OF ALL COST FUNCTIONS ASSOCIATED WITH CURRENT COST CODE.
C
2000 FCOST = COST*FCOST
2001 IF (IVAL .EQ. 1) WRITE(IUNIT,2100)IP,ICL,ICP,ILINE,ICODE,
2002 IA,K,FCOST
2003 FORMAT(2X,13.4X,12.5X,12.4X,12.5X,12.5X,12.5X,12.4X,PD10.3,
2004 2X,PD10.3)
2005 IF (FUNCT(IRW,4) .EQ. 0.0) GO TO 4000
2006 IRW = IDINT(FUNCT(IRW,4))
2007 GO TO 200
2008 IF(IFLAG(1) .EQ. IFLAG(4)) CALL TRACE(1540,
2009 I,IRW,K,ILINE,IP,
2010 * FUNCT(IRW,1), FCOST, A, B, C,
2011 D, E, F,RPATH(K), COST)
2012 RPATH(K) = FCOST
2013 CONTINUE
C
**Subroutine CDFUNC**

**Argument List**

- IRP - The index of the first column of flow
- RPATH1 - The incoming node path value
- IRDM - The array dimension for the Rpaths
- CD - The node split fraction to be calculated

**Dimension**

- RPATH1(1:750), RFC1(1:750)

### Program

```
C** Subroutine CDFUNC(IRP, RPATH1, IRDM, CD, IDM, NPASS, NRPTH, IPH, *  
     ADDR)**
C** IMPLICIT REAL*(A-H, O-Z)**
C** This subroutine will calculate the node split fraction  
** C when called.**

C** ARGUMENT LIST**
C IRP - The index of the first column of flow  
C RPATH1 - The incoming node path value  
C IRDM - The array dimension for the Rpaths  
C CD - The node split fraction to be calculated

C** DIMENSION RPATH1(1:750), RFC1(1:750)**
I2 = IRP + 1
I3 = IRP + 2
RFC1(I1) = 0.0
IVAL = 0
ADTEMP = RPATH1(I3)
RPATH1(I3) = ADDR

C** CHECK IF FUNCTION IS APPLICABLE**
IF(RPATH1(IRP) .EQ. 0.00) GO TO 250
C** COMPUTE TEMPORARY CD VALUE**
CALL FUNCTS(IRP, 2, 0, RPATH1, IRDM, 1, RFC1, IDM, NPASS, IUNIT,  
   * IVAL, ICP, ICL)
FLOOR = RPATH1(IRP) * RPATH1(I3) * CD
CD = FLOOR / RPATH1(IRP)
RPATH1(I3) = ADTEMP

C** CHECK FOR CD VALUES GREATER THAN ONE AND LESS THAN ZERO**
IF(IPH .EQ. NRPTH) GO TO 250
IF(CD .GT. 1.00) CD = 1.00
IF(CD .LT. 0.00) CD = 0.00
RETURN
```

Program Size: Procedure - 000171  Linkage - 005734  Stack - 000050

0000 Errors [CDFUNC.FTN-REV16.6]
SUBROUTINE ADDRCH (RPATH, RFC, IRP, ISTCT, LSTCT, IRDM)

C THIS SUBROUTINE WILL RECORD THE POSITIONS OF ADDRESSES IN RPATH AND WHEN DESIRED RESET THOSE COLUMN POSITIONS BACK TO ADDRESSES. RFC(1) MONITORS THE RECORD AND RESET OPERATIONS. IF EQUAL TO 1 IT WILL RESET, IF 0, RECORD.

ARGUMENT VALUES

RPATH - PATH VALUES
IFC - ARRAY TO HOLD COLUMN POSITIONS AND MONITOR
ISTCT - FIRST CLOT COLUMN
IRP - POSITION OF FIRST COLUMN
LSTCT - NUMBER OF FLOW COLUMNS FOLLOWING ISTCT

C IN D I C E S AND D E T E R M I N E IF R E C O R D O R R E S E T

IND1 = IRP + ISTCT
INDL = IND1 + LSTCT
IF(RFC(1) .EQ. 0) GO TO 250

C RE SET ADDRESSES IN RPATH

DO 100 K=IND1,INDL
IF(RFC(K) .EQ. 0) GO TO 100
RPATH(K) = RFC(K)
CONTINUE
G O TO 300

C M ON I T OR RECORD ADDRESSES FROM RPATH.

C R E C O M EN T A RY V A L U E S

C //V///////////////////////////////////////////////////S
IN D 1  =  IRP  +  IS T C T
IN D L  =  I H D 1  +  L S T C T
IF<RFC(1 )  .EQ. 0. > G O T O 250
C///\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\n
END

PROGRAM SIZE: PROCEDURE - 000130  LINKAGE - 000030  STACK - 000036
0000 ERRORS [<ADDCRH>FTN-REV16.6]
D-47

GO TO (1000, 50, 1000), IOSRCE

C //INTERMEDIATE PRINT OUT

C //INTERMEDIATE PRINT OUT

50 IF(IFNT.EQ.0) GO TO 300
CSSTT = 5H COST
SUMHED = 1H
NREFP = IFILE(13)
IGRPH = IFILE(22)
WRITE(IUNIT, 100) CSSTT, SUMHED, (RUNHH(I).I = 1, 2), (DATE(I).I = 1, 2),
PLACE(K).K = 1, 3), STIME

100 FORMAT(5//, 1H, 21X, 18H TRACK MAINTENANCE, 2H0, 7H, 1X, 64(1H-),
* / 19H RUN NUMBER, / 1A0, //, 19H DATE, //, 1A8, //, 19H PLACE, //, 3A0, //, 19H SIMULATION TIME,
* / F8.2, 1X, 64(1H-)
WRITE(IUNIT, 145)

145 FORMAT(1X, 64(1H-), /, 8X, 10H POPULATION, 9X, 5H CLASS, 9X,
* / 8X, 8H IN USE, 26X, 4HNENT, //, 1X, 64(1H-))
DO 146 ICL = 1, LSTCL
DO 147 ICP = 1, LSTCP
CALL RCOLUMN(ICL, LSTCL, ICP, LSTCP, IRP, IRCOL, IBDM)
WRITE(IUNIT, 148) STATE(5, IRCOL), ICL, ICP,
STATE(7, IRCOL)
148 FORMAT(8x, F10.2, 11X, I1, 1, I3, 12, I1, 10H SIMULATION TIME)
CONTINUE

146 CONTINUE

WRITE(IUNIT, 150)

150 FORMAT(1X, 64(1H-), /, 2X, 4H NODE, 2X, 5H PATH, 2X, 5H CLASS, 2X,
* / 6HCMPD - , 9X, 7H QUALITY, 4X, 10H PATH FLOWS, 2X,
* / 11H MAINTENANCE, //, 22X, 5H NENT, 15X, 10H MILES/ YEAR, 4X,
* / 7HDOLLARS, /, 1X, 64(1H-))
IFNT = D

160 CALL RCOLUMN(I CL2, LSTCL, ICP2, LSTCP, IRP, IRCOL, IBDM)
IRP2 = IRP + 1
IUSEQ = IRP + 5
USEQ = RPATH(IUSEQ)
I1 = IRP + LSTCT
I2 = I1 + LSTCT
DO 325 I = I1, I2
PCOST = RPATH(IRP)*RPATH(I)*(1.0 - USEQ) +
RPATH(IRP)*RPATH(IRP2)*RPATH(I)*USEQ + PCOST
325 CONTINUE
PCOST1 = PCOST + PCOST1
IF(RPATH(IRP).EQ.0.0) GO TO 2000
WRITE(IUNIT, 350) NODE, IP, ICL2, ICP2, RPATH(IRP2), RPATH(IRP),
PCOST
IF(RPATH(IRP).EQ.0.0) PCOST
350 FORMAT(2X, I3, 4X, I3, 5X, I1, 6X, I2, 5X, F8.2, 2X, F11.2, 3X,
GO TO 2000

C ABHML AND FINAL STATE PRINT FORMATS
C

1000 SUMHED = 1H

IF(IDSRCE.EQ.3)GO TO 1100
1000 SUMHED = 7H SUMMARY

1100 IF (IFMAT.EQ.2) GO TO 2500

C FINAL COST TABLE FORMAT
C

CSTSTT = 5H COST
WRITE(IUNIT,100)CSTSTT,SUMHED,(RUNN(I),I=1,2),(DATE(I),I=1,2),
*PLACE(I),I=1,3),STIME

WRITE(IUNIT,1244)
I2 = ISTCT + LSTCT
DO 1250 K = ISTCT+I2
   DO 1200 ICL = ISTCL,LSTCL
   DO 1150 ICP = ISTCP,LSTCP
      CALL RCOLUMN(ICL,LSTCL,ICP,LSTCP,IRP,IRCOL,IBDM)
      J = IRP+K
      ACCMED = ACCMED + ACCUM(J)

1150 CONTINUE
1200 CONTINUE
   JCODE = K+25
   WRITE(IUNIT,1204)ICODE,ACCMED

1204 FORMAT(?X,I2,10X,F12.2)
   CTOTAL = CTOTAL + ACCMED

1245 CONTINUE
1244 FORMAT(I4,32(1H-),/6X,5HCOST,10X,11HMAINTENANCE,/, 
*5X,4HCOST,14X,4HCOST,1X,32(1H-))
   1254 FORMAT(4X,14HANNUAL TOTAL 1,1X,F12.2,///)
   WRITE(IUNIT,1254)CTOTAL
   CTOTAL = 0.0
   WRITE(IUNIT,1304)

1304 FORMAT(I4,32(1H-),/6X,5HCOST,10X,11HMAINTENANCE,/, 
*24X,4HCOST,1X,32(1H-))
   DO 1450 ICL = ISTCL,LSTCL
   1450 ICP = ISTCP,LSTCP
      CALL RCOLUMN(ICL,LSTCL,ICP,LSTCP,IRP,IRCOL,IBDM)
      I1 = IRP + ISTCT
      I2 = I1 + LSTCT
      DO 1400 K = I1,I2
         ACCMED = ACCMED + ACCUM(K)

1400 CONTINUE
1425 WRITE(IUNIT,1204)ICL,ACCMED
   CTOTAL = CTOTAL + ACCMED
   ACCMED = 0.0
CSTSTT = 5HSTATE
WRITE(IUNIT,100)CSTSTT,SUMHED,(RUNH(I),I=1,2),(DATE(I),I=1,2),
*(PLACE(I),I=1,3),STIME
WRITE(IUNIT,2550)
2550 FORMAT(11X,)
*53HSYSTEM DEFECT SYSTEM DEFECT SYSTEM DEFECT,/
*1X,36H TRK CPT QUANTITY FRACTION QUANTITY ,
*28H FRACTION RATE OF RATE OF,/
*1X,32H CLS # AT START AT START NOW,6X,3HNOW,6X,
*16HGROWTH CHANGE,/,1X,64(1H-))
DO 2700 ICL = ISTCL,LSTCL
   DO 2600 ICP = ISTCP,LSTCP
      CALL RCOLUM( ICL,LSTCL,ICP,LSTCP,IRP,IRCOL,IBDM)
   DO 1123 NTMP = 5,17,4
      NTMP2 = NTMP + 1
      NTMP3 = NTMP + 2
      NTMP4 = NTMP + 3
      WRITE(IUNIT,2654)ICL,ICP,STATE(1,IRCOL),STATE(3,IRCOL),
      * STATE(NTMP,IRCOL),STATE(NTMP3,IRCOL),STATE(NTMP2,IRCOL),
      * STATE(NTMP4,IRCOL)
   1123 CONTINUE
2654 FORMAT(3X,11.2X,11.2X,PD8.2,1X,PD8.2,1X,PD8.2,1X)
   2600 CONTINUE
   2000 SUMHED = 8H
RETURN
END

PROGRAM SIZE:     PROCEDURE - 003616     LINKAGE - 000552     STACK - 000130
0000 ERRORS [<OUTPT>FTN-REV16.6]
SUBROUTINE HEADING(IU1,DT,TIMEL,TIME,STIME,LSTCP,LSTCL,  
*LSTPH,LSTND,IBDM)
* IMPLICIT REAL*8(A-H,O-Z) 
C**************************************************************************
C THIS SUBROUTINE WILL PRINT OUT A HEADING AT THE TOP OF EACH RUN. 
C IT WILL BE CALLED BY MAIN ONLY, AND IS CALLED ONCE PER RUN.
C**************************************************************************

ARGUMENT LIST

(SEE RFILE AND IFILE FOR DESCRIPTIONS)

IU1 - FORTRAN OUTPUT DEVICE NUMBER
* DT - TIME INCREMENT
* TIMEL - FINAL SIMULATION TIME
* TIME - START SIMULATION TIME
* LSTCP - NUMBER OF COMPONENTS
* LSTCL - NUMBER OF CLASSES
* LSTPH - NUMBER OF PATHS
* LSTND - NUMBER OF NODES
* IBDM - CLASS/COMPONENT BLOCK SIZE

COMMON/I2/IFILE(100)
COMMON/R2/FUNCT(10,10)
COMMON/R4/STATE(20,30)
COMMON/R6/WEIBL(3,30)
COMMON/R7/REVISN(8),SCHMDG(8),DATAFL(8)
LSTPH = IFILE(4)

C**************************************************************************
C PRINT HEADINGS
C**************************************************************************

WRITE(IU1,1000)*(REVISN(I),I=1,8),(SCHMDG(I),I=1,8),
   *(DATAFL(I),I=1,8),LSTCL,LSTCP,LSTPH,
   *(LSTND,TIME,TIMEL,DT)
1000 FORMAT(3(1X,8A8,/),6X,25HNUMBER OF TRACK CLASSES :,
   3(1X,8A8,/),6X,25HNUMBER OF TRACK COMPONENTS :,
   3(1X,8A8,/),6X,25HNUMBER OF TRACK PATHS IN SCHEMATIC DIAGRAM :,
   3(1X,8A8,/),6X,25HNUMBER OF TRACK NODES IN SCHEMATIC DIAGRAM :,
   3(1X,8A8,/),6X,25HTIME SIMULATION BEGINS AT YEAR :,
   3(1X,8A8,/),6X,25HTIME SIMULATION ENDS ON YEAR :,
   3(1X,8A8,/),6X,25HTIME INCREMENT SIZE IN YEARS :
   18(1X,8A8,/),6X,25HUEIBULL DISTRIBUTION PARAMETERS :,
   1X,6(1H-)),
   *(LSTND,CLAS,COMPONENT),*(LSTND,CLASS,COMPONENT)
   4(9D9.2,3(1X,8A8,/),3(1X,8A8,/),1X,64(1H-))
DO 1100 ICL = 1,LSTCL
1100 CONTINUE
DO 1200 ICP = 1, LSTCP
CALL RCOLUMN( ICL, LSTCL, ICP, LSTCP, IRP, IRCOL, 18DM)
WRITE(IUNIT,2000) ICL, ICP, (WEIBL(I, IRCOL), I=1, 3)
2000 FORMAT(8X,I1,11X,I1,11X,F6.2,5X,F5.2,6X,F6.2)
CONTINUE
1200 WRITE(IUNIT,3000)
3000 FORMAT(/,15X,37HFUNCTIONS AND SHAPE PARAMETERS ACTIVE,/, *
1X,64(1H-),/2X,64(1H-),/2X,64(1H-))
CONTINUE 1
WRITE(IUNIT,3000)
FUNCTIONS AND SHAPE PARAMETERS ACTIVE,/
* 1X,64(1H-),/2X,64(1H-)
1 = 1
IF(FUNCT(I,1), EQ. 0.0) GO TO 6000
4000 FORMAT(3X,13,3X,12,4X,5(PD9.2,IX))
I = I + 1
4001 FORMAT(12)
GO TO 1150
6000 RETURN
END

PROGRAM SIZE: PROCEDURE - 001137 LINKAGE - 000066 STACK - 000050
0000 ERRORS [<HEADNG>FTN-REV16.6]
C******************************************************************************
C******************************************************************************
SUBROUTINE NDVAL (IRDM, IDM, ISTND, LSTND, ISTCL, LSTCL, ISTCP,
* LSTCP, IBDM, HREFF)
IMPLICIT REAL*8(A-H,O-Z)
C******************************************************************************
C PROGRAM NDVAL CALCULATES NODE SPLIT FRACTIONS FOR SIMPLE *
C AND DEFECTIVE BASED BRANCH DECISIONS AND STORES THESE VA-*
C LUES IN RNODE DATA FILE. *
C
C ARGUMENT LIST
SEE IFILE AND RFILE FOR DETAIL
C
C IRDM - RPATH DIMENSION *
C IDM - IPATH DIMENSION *
C ISTND - FIRST NODE *
C LSTND - LAST NODE *
C ISTCL - FIRST CLASS *
C LSTCL - LAST CLASS *
C ISTCP - FIRST COMPONENT *
C LSTCP - LAST COMPONENT *
C IBDM - CLASS/COMPONENT BLOCK SIZE *
C
COMMON/I2/IFILE(100)
COMMON/I3/IFLAG(10)
COMMON/I7/IODE(6,300)
COMMON/R1/ACCUM(750)
COMMON/R2/FUNCT(10,10)
COMMON/R3/RNODE(30,300)
COMMON/R4/STATE(9,20)
COMMON/R5/RFILE(100)
DIMENSION RPATH(1750),RPATH2(750),RPATH3(750)
DIMENSION IPATH(15),IPATH2(15),IPATH3(15)
C FILL IN EMPTY PATHS BELOW SUM NODES
C
DO 1000 J = ISTND,LSTND
1000 IF1 = IODE(1,J)
IF2 = IODE(2,J)
IF3 = IODE(3,J)
IF(IODE(4,J) EQ.1) GO TO 1000
CALL PATH1(IF1,IPATH1,IDM,RPATH1,IRDM)
CALL PATH2(IF2,IPATH2,IDM,RPATH2,IRDM)
CALL PATH3(IF3,IPATH3,IDM,RPATH3,IRDM)
DO 850 ICL = ISTCL,LSTCL.
DO 900  ICP = ISTCP,LSTCP
    CALL RCOLUMN(ICL,LSTCL,ICP,LSTCP,IRF,IRCOL,IBDM)
    IF(RPATH3(IRP) .GT. 0.0) GO TO 900
    K = IRP + 1
C
    RPATH3(IRP) = RPATH1(IRP)+RPATH2(IRP)
    IF(RPATH3(IRP) .EQ. 0.00) GO TO 900
    QUAL1 = RPATH1(IRP)*RPATH1(K)
    RPATH3(K) = (QUAL1+QUAL2)/RPATH3(IRP)
    CALL POUT(IF3,IPATH3,IIDM,RPATH3,IRDM)
  900  CONTINUE
  850  CONTINUE
  1000 CONTINUE
C DETERMINE SPLIT FRACTIONS FOR BRANCH NODES.
C
DO 1200 J = ISTND,LSTND
  IF(INODE(4,J) .EQ. 0) GO TO 1200
  IF1 = INODE(1,J)
  IF2 = INODE(2,J)
  IF3 = INODE(3,J)
  CALL PATHIN(IF1,IPATH1,IIDM,RPATH1,IRDM)
  CALL PATHIN(IF2,IPATH2,IIDM,RPATH2,IRDM)
  CALL PATHIN(IF3,IPATH3,IIDM,RPATH3,IRDM)
  DO 1150 ICL = ISTCL,LSTCL
    DO 1150 ICP = ISTCP,LSTCP
      CALL RCOLUMN(ICL,LSTCL,ICP,LSTCP,IRF,IRCOL,IBDM)
      K = IRP+1
      IF(INODE(5,J) .EQ. 1) GO TO 800
C
      IF(RPATH1(IRP) .EQ. 0.0) GO TO 700
      RNODE(IRCOL,J) = RPATH2(IRP)/RPATH1(IRP)
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/RPATH1(IRP)
      GO TO 1148
  1148  RNODE(IRCOL,J) = 1.0
      IF(RNODE(IRCOL,J) .LT. 0.0) RNODE(IRCOL,J) = 0.0
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      1148  * IF(RNODE(IRCOL,J) .LT. 0.0) RNODE(IRCOL,J) = 0.0
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      1148  * IF(RNODE(IRCOL,J) .LT. 0.0) RNODE(IRCOL,J) = 0.0
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      1148  * IF(RNODE(IRCOL,J) .LT. 0.0) RNODE(IRCOL,J) = 0.0
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      1148  * IF(RNODE(IRCOL,J) .LT. 0.0) RNODE(IRCOL,J) = 0.0
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      IF(INODE(6,J) .EQ. 1) RNODE(IRCOL,J) = RPATH3(IRP)/BAD1
      1148  * IF(RNODE(IRCOL,J) .LT. 0.0) RNODE(IRCOL,J) = 0.0
  1150 CONTINUE
  1100 CONTINUE
  1200 CONTINUE
**SUBROUTINE INTDAT(LSTPH,IIDM,IRDM)**

**IMPLICIT REAL*(A-H,O-Z)**

**C INTDAT WILL CLEAR THE DIRECT ACCESS FILES PRIOR TO READING IN**

**C THE DATA.**

**ARGUMENT LIST**

**C** (SEE RFILE AND IFILE FOR DETAIL)

**C**

**C**

**C** LSTPH - NUMBER OF PATH IN MODEL

**C** IIDM - DIMENSION OF IPATH

**C** IRDM - DIMENSION OF RPATH

**C**

**C**

**DIMENSION IPATH(15),RPATH(750)**

**DO 100 I = 1,IIDM**

**IPATH(I) = 0**

**RPATH(I) = 0.00**

**100 CONTINUE**

**DO 200 I = IIDM,IRDM**

**RPATH(I) = 0.00**

**200 CONTINUE**

**DO 250 I = 1,LSTPH**

**IPATH(I) = I**

**CALL PATOUT(I,IPATH,IIDM,RPATH,IRDM)**

**250 CONTINUE**

**RETURN**

**END**

**PROGRAM SIZE: PROCEDURE - 000100**

**LINKAGE - 005732**

**STACK - 000024**

**0000 ERRORS ([INTDAT]FTN-REV16.6)**
C*****************************************************************************************
C****************************************************************************
C****************************************************************************
C****************************************************************************
SUBROUTINE RAWDAT(LSTCP,LSTCL,ICODE,IIDM,IRDM,IBDM,ISTCT,
                   *LSTCT)
C*******************************************************************************
C IMPLICIT REAL*(A-H,O-Z)
COMMON/I2/IFILE(100)
C COMMON/R2/FUNCT(10,10)
COMMON/R4/STATE(20,30)
COMMON/R5/FILE(100)
COMMON/R7/REVISION(3),SCHMD(8),DATAFL(8)
DIMENSION REALA(12),REALB(9),RELC(6),RELD(3),REAL2(12),
*FTMP(10,10)
DIMENSION IPATH(15),IFLHK(20)
DIMENSION RPATH(750),VALUE(50),RL(14)
C NPATH = IFILE(13)
NCOSTS = IFILE(17)
IRUNIT = IFILE(22)
ITUNIT = IFILE(23)
DO 7 I=1,20
   IFLHK(I) = I+1
   RPATH(I) = 0.0
7 CONTINUE
DO 9 I = 1,10
   DO 8 K = 1,10
      Ftmp(I,K) = 0.00001
8 CONTINUE
9 CONTINUE
NXTAV = 1
ISTLHK = 1
ITMP = 0
ICOUNT = 0
C ITRACE = 0
WRITE(ITUNIT,1000)ITRACE
CALL ASSIGN(RL,NCOSTS,VALUE,REAL2,IFLAG)
WRITE(ITUNIT,565)VALUE(1),VALUE(9)
565 FORMAT(2(1P10.3,1X)'VALUES IN RAWDAT')
ITRACE = 1
WRITE(ITUNIT,1000)ITRACE
1000 FORMAT(II)
C READ(IRUNIT,83)(DATAFL(K),K=1,8)
READ(IRUNIT,83)IPH,ICD,RL(1),RL(2),RL(3),RL(4),RL(5),RL(6),
*RL(7),RL(8),RL(9),RL(10)
CALL PATHINFO(PH,IPATH,II,D,RPATH,IRDM)
CALL RESET(RL,ICP,ICL,IND,ITYPE,ICST,IPDT,IODT,IMFG,IRPB,
*       IRP = 1
*       IP = 1
ITRACE = 2
WRITE(ITUNIT,1000)ITRACE
CALL RCLEAR(RL)
ITRACE = 3
WRITE(ITUNIT,1000)ITRACE
READ(IRUNIT,*)PH,ICD,RL(1),RL(2),RL(3),RL(4),RL(5),
   RL(6),RL(7),RL(8),RL(9),RL(10),RL(11),
   RL(12),RL(13),RL(14)
CALL RCOLUMN(ICAL,STCL,ICP,STSCP,IRP,IRECOL,IBDM)
IF(ICAL .EQ. 1)GO TO 20
IF(ICAL .EQ. 2)GO TO 30
IF(ICAL .EQ. 3)GO TO 40
WRITE(ITUNIT,565)VALUE(1),VALUE(9)
CALL DATA1(RL,VALUE,HCOSTS,ISTCT,IPH)
ITRACE = 4
WRITE(ITUNIT,1000)ITRACE
GO TO 50
IFLAG = 1
CALL REALCL(REALA,REALB,REALC,REALT,REAL2)
ITRACE = 5
WRITE(ITUNIT,1000)ITRACE
READ(IRUNIT,*)PH,ICD,ICODE,IFHUMB,REALA(1),REALA(2),REALA(3),
   REALA(4),REALA(5),REALA(6),REALA(7),REALA(8),
   REALA(9),REALA(10),REALA(11),REALA(12)
IF(ICAL .EQ. 9)READ(IRUNIT,*)PH,ICD,ICODE,IFHUMB,REALB(1),
   REALB(2),REALB(3),REALB(4),REALB(5),
   REALB(6),REALB(7),REALB(8),REALB(9)
IF(ICAL .EQ. 9)READ(IRUNIT,*)PH,ICD,ICODE,IFHUMB,REALC(1),
   REALC(2),REALC(3),REALC(4),REALC(5),REALC(6)
IF(ICAL .EQ. 9)READ(IRUNIT,*)PH,ICD,ICODE,IFHUMB,REALD(1),
   REALD(2),REALD(3)
CALL REFILL(REALA,REALB,REALC,REALD,REAL2)
ITRACE = 6
WRITE(ITUNIT,1000)ITRACE
WRITE(ITUNIT,37)(REAL2(I2),I2=1,12)
FORMAT(1X,2(6(PB0.3,1X),'/'))
CALL DATA2(RL,IFCODE,NCOSTS,REAL2,ITMP,ICOUNT,
* IDCODE,YCODE,IPH,IFNUMB,ISTCT,FTMP,IFTLNK,NXTAY,
* ISTALLK,RPATH,IRP)
ITRACE = 7
WRITE(ITUNIT,1000)ITRACE
GO TO 50
CALL DATA3(IPH,RL,LSTCP,LSTCL,IDCODE,VALUE,NCOSTS,
* IPATH,ICDM,RPATH,IRDH,IDCODE,ICL,LSTCL,ICP,LSTCP,ISTCT)
ITRACE = 8
WRITE(ITUNIT,1000)ITRACE
CALL RCLEAR(RL)
ITRACE = 9
WRITE(ITUNIT,1000)ITRACE
READ(IRUNIT,*)ICK,ICD,RL(1),RL(2),RL(3),RL(4),RL(5),
* RL(6),RL(7),RL(8),RL(9),RL(10),RL(11),
* RL(12),RL(13),RL(14)
WRITE(ITUNIT,1000)ITRACE
IF((ICK.EQ.999) OR (ICD.EQ.0))GO TO 60
WRITE(ITUNIT,1000)ITRACE
GO TO 15
WRITE(ITUNIT,2000)ITRACE
FORMAT(12)
CALL DUMMY(IRP,IPH,ITYPE,IFCODE,IDCODE,ISTCT,VALUE,
* NCOSTS,IPATH,ICDM,RPATH,IRDH,ICST,IPDT,
* IQDT,INFG,IRPB,LSTCT,FTMP,NPATHR,IRCDL,
* IPRT,ITMP)
ITRACE = 11
WRITE(ITUNIT,2000)ITRACE
WRITE(ITUNIT,57)(IPATH(I),I = 1,15)
FORMAT('POST DUMMY ' ,13)
WRITE(ITUNIT,58)IPH,(RPATH(I),I = 1,20)
FORMAT(13,2X,5(2X,5(1PD10.3,/))
FORMAT('POST N1/KVALUE CHANGE ' ,13)
(2114) N = ISTLNK
(2115) ICHT = ICOUNT + ITMP
(2116) IF(ITMP .EQ. 0) GO TO 850
(2117) DO 875 N = 1, ITMP
(2118) N1 = IDINT(FTMP(N,3))
(2119) WRITE(ITUNIT,2000)N1
(2120) WRITE(ITUNIT,2000)ICOUNT
(2121) VALUE(N1) = 0.0
(2122) WRITE(ITUNIT,200)(FTMP(N,K), K=1,10)
(2123) 200 FORMAT(2(5(IPD10.3,1X),/),/)
(2124) 875 CONTINUE
(2125) N = ISTLNK
(2126) 806 ICOUNT = ICOUNT + 1
(2127) DO 808 I = 1,10
(2128) FUNCT(ICOUNT,I) = FTMP(N,I)
(2129) 808 CONTINUE
(2130) N = IFTLNK(N)
(2131) IF(ICOUNT .LT. ICHT) GO TO 806
(2132) 850 ITMP = 0
(2133) CALL PATOUT(IPH,IPATH,IIDM,RPATH,IRDM)
(2134) ITRACE = 12
(2135) DO 857 I = 1,12
(2136) IF(IFTLNK(I)) = I+1
(2137) RPATH(I) = 0.0
(2138) 857 CONTINUE
(2139) NXTAV = 1
(2140) ISTLNK = I
(2141) WRITE(ITUNIT,2000)ITRACE
(2142) IF(ICK .EQ. 999) GO TO 100
(2143) ICT = ICOUNT
(2144) IPH = ICK
(2145) CALL RESET(RL,ICL,IND,ITYPE,ICST,IPDT,IQDT,IMFG,IRPB,
(2146) * IPRT)
(2147) CALL ASSIGN(RL,NCOSTS,VALUE,REAL2,FLAG)
(2148) CALL PATHIN(IPH,IPATH,IIDM,RPATH,IRDM)
(2149) GO TO 10
(2150) C
(2151) C
(2152) C
(2153) 100 RETURN
(2154) END

PROGRAM SIZE: PROCEDURE - 002232 LINKAGE - 007601 STACK - 000050
0000 ERRORS [<RAUDAT>FTN-REV!6.6]
C***+  ****+*#* +  +  #*****•*** + #**&:* + *## ****=«<*••<<**##:** ■ • { > * * : * $ #
C  #  *  *  * • #  * * ♦ * #  +  *#******## #**#*** +  iM (sJ n |<  * # #  * * * #  + *!♦ **••* #*:)<!)<**#*>»<###* *
S U B R O U T IN E ASSIGN(RL,NCOSTS,VALUE,REAL2,IFLAG)

C** SUBROUTINE ASSIGN(RL,NCOSTS,VALUE,REAL2,IFLAG)

C

C IMPLICIT REAL*8(A-H,O-Z)

DIMENSION REAL2(12)
DIMENSION VALUE(NCOSTS),RL(14)

C DO 100 I = 1,14
100     RL(I) = 0.0

C DO 200 I = 1,NCOSTS
200     VALUE(I) = 0.00D 00

C CONTINUE

C DO 300 I = 1,12
300     REAL2(I) = 0.0

C RETURN

END

PROGRAM SIZE:  PROCEDURE - 000066    LINKAGE - 000024    STACK - 000032
0000 ERRORS [(ASSIGN)FTN-REV16.61]
C  SUBROUTINE RESET(RL, ICP, ICL, IND, ITYPE, ICST, IPDT, IQDT, IMFG, IRPB, IPRT)
C
C  REAL*8 RL(14)
C  ICP = IFIX(RL(1))
C  ICL = IFIX(RL(2))
C  IND = IFIX(RL(3))
C  ITYPE = IFIX(RL(4))
C  ICST = IFIX(RL(5))
C  IPDT = IFIX(RL(6))
C  IQDT = IFIX(RL(7))
C  IMFG = IFIX(RL(8))
C  IRPB = IFIX(RL(9))
C  IPRT = IFIX(RL(10))
C
C  RETURN
C
C  END

PROGRAM SIZE: PROCEDURE - 000071
0000 ERRORS [<RESET >FTH-REV16.6] LINKAGE - 000027
STACK - 000056
SUBROUTINE RCLEAR(RL)
IMPLICIT REAL*8(A-H,O-Z)

DIMENSION RL(14)

DO 100 I = 1,13
   RL(I) = 0.0
100 CONTINUE

RETURN
END
C*************************************************************
C*************************************************************
C*************************************************************
C*************************************************************
SUBROUTINE REFILL (REALA, REALB, REALC, REALE, REALF)
IMPLICIT REAL*(A-H, O-Z)
C*************************************************************
C THIS SUBROUTINE WILL TAKE DATA STORED IN REALA, REALB,
C REALE AND REALF AND PACTH IT INTO REAL2 TO BE USED IN
C DATA2.
C*************************************************************
DIMENSION REALA(12), REALB(9), REALC(6), REALE(3)
DIMENSION REAL2(12)
C
I = 1
REAL2(I) = REALA(I)
I = I + 1
IF(I .EQ. 13) GO TO 200
IF(REALA(I) .NE. 0.0) GO TO 50
C
IF(REALB(I) .EQ. 0.0) GO TO 200
IB = 1
75 REAL2(I) = REALB(IB)
I = I + 1
IF(I .EQ. 13) GO TO 200
IB = IB + 1
IF(IB .EQ. 10) GO TO 80
IF(REALB(IB) .NE. 0.0) GO TO 75
C
80 IF(REALC(I) .EQ. 0.0) GO TO 200
IC = 1
100 REAL2(I) = REALC(IC)
I = I + 1
IF(I .EQ. 13) GO TO 200
IC = IC + 1
IF(IC .EQ. 7) GO TO 120
IF(REALC(IC) .NE. 0.0) GO TO 200
C
120 IF(REALD(I) .EQ. 0.0) GO TO 200
ID = 1
125 REAL2(I) = REALD(ID)
I = I + 1
IF(I .EQ. 13) GO TO 200
ID = ID + 1
IF(ID .EQ. 4) GO TO 200
IF(REALD(I) .NE. 0.0) GO TO 125
200 RETURN
END

PROGRAM SIZE: 000214
PROCEDURE - 0000214
LINKAGE - 000031
STACK - 000032
0000 ERRORS [REFILL]FTN-REV16.6]
C*******************************************************************************
C*******************************************************************************
SUBROUTINE DATA (RL, VALUE, NCOSTS, ISTCT, IPH)
IMPLICIT REAL*8(A-H, O-Z)
DIMENSION RL(14), RVRL(24), VALUE(50)

C
CWRITE(ITUNIT,1)RL(1)
FORMAT('RL',1X,1PD10.3)
ITRACE = 55
CWRITE(ITUNIT,777)ITRACE
FORMAT('ITRACE', 1X, 13)
ICOST = IDINT(RL(1))
CWRITE(IUNIT,2)ICOST
FORMAT('ICOST', 1X, 12)
ITRACE = 56
CWRITE(IUNIT,777)ITRACE
IF((ICOST .GE. 30) .AND. (ICOST .LE. 49))GO TO 200
VALUE(ICOST) = VALUE(ICOST)+RL(2)
ITRACE = 57
CWRITE(IUNIT,777)ITRACE
GO TO 500
C
VALUE(ICOST) = VALUE(ICOST)+(RL(2)*RL(3)*RL(4))
WASH = RL(2)*RL(3)*RL(4)
IF(VALUE(20) .LE. 0.0000)GO TO 500
WASH = WASH * VALUE(20)
ITRACE = 58
CWRITE(IUNIT,777)ITRACE
GO TO 500
RETURN
END

PROGRAM SIZE: PROCEDURE - 000143 LINKAGE - 000201 STACK - 000040
0000 ERRORS [<DATA1>FTN-REV16.6]
**SUBROUTINE DATA**

**REAL**

**RL, IFCODE, NCOSTS, REAL2, ITMP, ITCOUNT, IFTLNK, NXTAV, ISTLNK, RPATH, IRP**

**IMPLICIT REAL*8(A-H, O-Z)**

**DIMENSION REAL2(12), FTMP(10, 10), RL(14), RPATH(750), IFTLNK(20)**

**C**

**THIS SUBROUTINE WILL READ IN VALUES TO BE STORED IN THE FUNCT ARRAY, STORE THEM IN A TEMPORARY ARRAY AND STORE THE FUTURE FUNCT ADDRESS BACK IN RPATH.**

**ARGUMENT LIST**

1. **RL** = ARRAY
2. **IFCODE** = ARRAY HOLDING FUNCT DATA
3. **REAL2** = ARRAY HOLDING FUNCT DATA
4. **ITMP** = # OF LINES IN TEMPORARY ARRAY
5. **ITCOUNT** = # OF LINES CURRENTLY STORED IN FUNCT
6. **IDCODE** = LAST COST CODE
7. **IPH** = PATH PRESENTLY ON
8. **IFNUMB** = FUNCTION #
9. **VCODE** = FUNCTION #
10. **ISTCT** = # OF FLOW COLUMNS
11. **FTMP** = TEMPORARY ARRAY FOR FUNCT VALUES
12. **IFTNK** = FORWARD LINK ARRAY FOR FTMP
13. **NXTAV** = NEXT AVAILABLE SPACE IN FTMP
14. **ISTLNK** = FIRST LINE OF LINKED ARRAY
15. **RPATH** = ARRAY HOLDING FLOW AND COST VALUES AND ADDRESSES
16. **IRP** = REFERENCE COLUMN OF RPATH

**C**

**1000 FORMAT(12)**

**C**

**L = 0**

**C**

IF( (IFCODE .GE. 20) .AND. (IFCODE .LE. 21)) GO TO 110

GO TO 310

J = IRP

M = IFCODE + 1

DO 300 I = 20, 21

IF( IFCODE .EQ. I) GO TO 250

GO TO 275

L = J + 2

J = J + 1

300 CONTINUE

C

GO TO 500

C

J = IRP + ISTCT

C

WRITE(UNIT,22)
(2354) 22 FORMAT('M',2X,I3)
(2355) 2 WRITE(ITUNIT,21)IDCODE,NCOSTS
(2356) 21 FORMAT('IDCODE',1X,I2,1X,'NCOSTS',1X,I2)
(2357) DO 400 I = 31,IDCODE
(2358) IF(IFCODE .NE. I)GO TO 375
(2359) L = J
(2360) 375 J = J + 1
(2361) 400 CONTINUE
(2362) WRITE(ITUNIT,23)L
(2363) 23 FORMAT('L',2X,I2)
(2364) WRITE(ITUNIT,501) VCODE
(2365) 500 VCODE = RL(2)
(2366) WRITE(ITUNIT,501) VCODE
(2367) 501 FORMAT('VCODE:',I3)
(2368) CALL FPLACE(IADDRESS,ITMP,IPH,IFNUMB,IFCODE,REALL2,FTMP,NXTAV,
(2369) *VCODE,IFTLNK,ISTLNK,RPATH,L)
(2370) C
(2371) C
(2372) C
(2373) C
(2374) RETURN
(2375) END

PROGRAM SIZE: PROCEDURE - 000215 LINKAGE - 000032 STACK - 000076
0000 ERRORS [DATA2 ]FTRN-REV16.6]
C****************************************************************************************************
C****************************************************************************************************
C****************************************************************************************************
C****************************************************************************************************
C****************************************************************************************************
    SUBROUTINE FPLACE(ICOUNT, ITMP, IPH, IFNUMB, IFCODE, REAL2, FTMP, NXTAV, *
             YCODE, IFTLHK, ISTLHK, RPATH, L)
      IMPLICIT REAL*8(A-H, O-Z)

C THIS SUBROUTINE WILL INPUT FUNCTION DATA INTO THE LINKED LIST
C FTMP TO BE HELD UNTIL IT IS TO BE READ INTO THE FUNCT FILE
C
C    ICOUNT IS THE NUMBER OF LINES CURRENTLY IN THE FUNCT FILE
C    ITMP - THE NUMBER OF LINES CURRENTLY FILLED IN FTMP
C    IPH - THE CURRENT PATH NUMBER
C    IFNUMB - THE FUNCTION NUMBER OF THIS DATA
C    IFCODE - THE COST CODE
C    REAL2 - ARRAY HOLDING FUNCTION CONSTANTS
C    FTMP - TEMPORARY HOLDING ARRAY FOR FUNCT
C    NXTAV - THE NEXT AVAILABLE SPACE IN FTMP
C
C    VCODE - X FUNCTION #
C    IFTLHK - FORWARD LINK ARRAY
C    FSTLHK - POINTS TO THE FIRST DATA LINE IN FTMP
C    RPATH - ARRAY HOLDING THE FLOW AND COST DATA
C    L - COLUMN IN RPATH CORRESPONDING TO CURRENT COSTCODE

C DIMENSION FTMP(10,10), REAL2(12), IFTLHK(20), RPATH(750)

C WRITE(ITUNIT,999)REAL2(1), REAL2(2), REAL2(3), REAL2(4)
C 999 FORMAT(4(F2.0,1X))

C WRITE(ITUNIT,997)YVAL
C 997 FORMAT('THIS IS YVAL IN FPLACE',F5.0)

C FTMP(NXTAV,1) = DBLE(FLOAT(IPH))
C
C FTMP(NXTAV,2) = DBLE(FLOAT(IFNUMB))
C FTMP(NXTAV,3) = DBLE(FLOAT(IFCODE))
C FTMP(NXTAV,5) = VCODE
C K = 1
C DO 100 I = 6,10
C    FTMP(NXTAV,I) = REAL2(K)
C 100 CONTINUE
C CALL LHKLST(IFTLHK, ICOUNT, ITMP, FTMP, NXTAV, RPATH, L, ISTLHK)
C
C WRITE(ITUNIT,94)(FTMP(ITMP,K),K=1,10)
C 94 FORMAT(' FTMP-',/,2(5(PD10.3,1X),/))
C RETURN
C END

PROGRAM SIZE:  PROCEDURE - 000220 LINKAGE - 000061 STACK - 000066
0000 ERRORS [(FPLACE FTN-REV16.6)
C******************************************************************************
C******************************************************************************
C SUBROUTINE IFLNK (IFLNK, ICOUNT, ITMP, FTMP, HXTAV, RPATH, L, ISTLNK)
C******************************************************************************
DIMENSION IFLNK(20)
DIMENSION FTMP(10, 10), RPATH(750)
C THIS SUBROUTINE IS DESIGNED TO INSERT FUNCTION DATA IN
C A LINKED TEMPORARY ARRAY. WHEN THE DATA IS INSERTED THE FUTURE
C FUNCTION ADDRESS IS PLACED IN THE CORRESPONDING
C RPATH COLUMN, WHEN ALL THE FUNCT DATA FOR A PARTICULAR PATH-
C CLASS-COMPONENT IS INSERTED THE ARRAY IS THEN FED INTO THE
C FUNCT FILE AND CLEARED FOR THE NEXT RUN.
C IFLNK - ARRAY HOLDING THE FORWARD LINKS
C HXTAV - NEXT AVAILABLE SPACE
C FTMP - TEMPORARY ARRAY FOR FUNCT DATA
C RPATH - ARRAY HOLDING COST AND FLOW DATA
C L - COLUMN IN RPATYH WHERE FUNCT ADDRESS IS STORED
C ICOUNT - NUMBER OF DATA LINES IN FUNCT FILE
C******************************************************************************
ITMP = ITMP+1
50 FORMAT( 'L-', 12.1X, 'RPATH(L)-', PD10.3)
C WRITE(IUTHIT,75)ITMP, ICOUNT
75 FORMAT( ITMP-';', 12.1X, 'ICOUNT--', 12)
LINK = ISTLNK
NEWLK = HXTAV
IDONE = 0
100 IF(IDONE .EQ. 1)GO TO 500
IF(Link .EQ. NEWLK) GO TO 400
IF(FTMP(LINK, 3) .LT. FTMP(HEWLK, 3))GO TO 300
IF(FTMP(LINK, 3) .GE. FTMP(HEWLK, 3))GO TO 200
IF(FTMP(LINK, 4) .EQ. 0.0)FTMP(LINK, 4) = DBLE(FLOAT(ICOUNT+ITMP))
200 LLINK = LINK
LINK = IFLNK(LINK)
GO TO 100
300 IDONE = 1
400 NXTAV = IFLNK(NEWLK)
IFLNK(NEWLK) = LINK
IF(Link .EQ. ISTLNK)GO TO 350
IFLNK(LINK) = NEWLK
GO TO 375
350 ISTITLNK = NEWLK
375 IF((RPATH(L) .GT. 10.0**-47) .OR. (RPATH(L) .LT. 10.0**-50))
*RPATH(L) = DBLE(FLOAT(ICOUNT+ITMP))*10.0**-50
400 GO TO 100
(2474) GO TO 100
(2475) C500 WRITE(ITUNIT,550)(IFTLNK(K),K=1,20),NXTAV,ISTLNK
(2476) C550 FORMAT('IFTLNK',2(10(I3,1X),/),'/','NXTAV-',I2,' IST-',I2)
(2477) C WRITE(ITUNIT,50)L,RPATH(L)
(2478) 500 RETURN
(2479) END

PROGRAM SIZE: PROCEDURE - 000364 LINKAGE - 000036 STACK - 000054
0000 ERRORS [<LHKLST>FTN-REV16.6]
C*******************************************************************************
C*******************************************************************************
C subroutine DUMMY(IRP,IPH,ITYPE,ICODE,IDCODE,ISTCT,VALUE,
C NCOSTS,IPATH,IDM,FPATH,IRDM,ICST,IPDT,
C IQDT,IMFG,IRPB,LSTCT,FPATH,NPATH,IRCOL,
C IPRT,ITMP)
C*******************************************************************************
C IMPLICIT REAL*8(A-H,O-Z)
COMMON/R4/STATE(20,30)
DIMENSION IPATHIIDI
DIMENSION FPATH(750),VALUE(50),ITMP(10,10)
C DO 30 I = 3,4
C RPATH(I) = 0.0
C 30 CONTINUE
C IPATH(1) = IPH
C IPATH(2) = IPATH(3) =
C IPATH(4) =
C IPATH(5) = ITYPE
C IPATH(8) = IRPB
C IPATH(9) =
C IPATH(10) = ICST
C IPATH(11) = IPDT
C IPATH(12) = IQDT
C IPATH(13) = IMFG
C IPATH(14) = IPRT
C IPATH(15) =
C IF(ITMP .LE. 0) GO TO 100
C 875 N = 1,ITMP
C N1 = IDINT (ITMP (N,3))
C WRITE (ITUNIT,2000)N1
C WRITE (ITUNIT,2000)ICOUNT
C IF (N1 .EQ. 20) .OR. (N1 .EQ. 21)) GO TO 875
C FPATH(N,6) = VALUE(N1)+ITMP(N,6)
C 875 CONTINUE
C WRITE (ITUNIT,19)IRP
C FORMAT ('IRP IN DUMMY','1X,I1')
C WRITE (ITUNIT,43)
C FORMAT ('VALUE IN DUMMY')
C WRITE (ITUNIT,44)(VALUE(I),I = 20,49)
C FORMAT ('5P10.3,1X')
C WRITE (ITUNIT,42)
C WRITE (ITUNIT,44)(RPATH(I),I = 1,20)
C WRITE (ITUNIT,10)RPATH(3),RPATH(4)
`PROGRAM 10

10 FORMAT(2(1PE10.3,1X), 'RPATH 3, 4')
100 IF (.NOT. IRP) GO TO 21
101 STATE(1, IRCOL) = VALUE(20)
102 STATE(3, IRCOL) = VALUE(21)
103 K = 20
104 J = IRP + 1
105 C WRITE(ITUNIT, 54)J
54 FORMAT(1, ', 13)
58 DO 200 I = IRP, J
59 VALUE(K) = 0.0
60 WRITE(ITUNIT, 64)K
61 FORMAT(112)
62 K = K + 1
63 CONTINUE
64 CONTINUE
65 DO 400 I = IRP, J
66 IF (RPATH(I) .LT. 10.0**-47).AND. (RPATH(I) .GE. 10.0**-
67 50) GO TO 300
68 WRITE(ITUNIT, 64)K
69 VALUE(K) = 0.0
70 K = K + 1
71 CONTINUE
72 CONTINUE
73 WRITE(ITUNIT, 10)RPATH(3), RPATH(4)
74 J = IRP + I - STCT - 1
75 C WRITE(ITUNIT, 54)J
76 K = 30
77 M = J + STCT + 1
78 DO 400 I = J, M
79 IF (RPATH(I) .LT. 10.0**-47).AND. (RPATH(I) .GE. 10.0**-
80 50) GO TO 300
81 WRITE(ITUNIT, 64)K
82 VALUE(K) = 0.0
83 K = K + 1
84 CONTINUE
85 CONTINUE
86 WRITE(ITUNIT, 10)RPATH(3), RPATH(4)
87 C WRITE(ITUNIT, 43)
88 C WRITE(ITUNIT, 44)( VALUE(I), I = 1, 20)
89 C WRITE(ITUNIT, 42)
42 FORMAT(1 'RPATH IN DUMMY')
44 C WRITE(ITUNIT, 44)( RPATH(I), I = 1, 20)
45 RETURN
46 END

PROGRAM SIZE: PROCEDURE - 000436 LINKAGE - 000054 STACK - 000122
0000 ERRORS [DUMMY >FTN-REV16.6]
SUBROUTINE REALCL(REA, REALB, REALC, REALD, REAL2)

DIMENSION REALA(12), REALB(9), REALC(6), REALD(3), REAL2(12)

DO 100 I = 1, 12
   REALA(I) = 0.0
100 CONTINUE

DO 200 I = 1, 9
   REALB(I) = 0.0
200 CONTINUE

DO 300 I = 1, 6
   REALC(I) = 0.0
300 CONTINUE

DO 400 I = 1, 3
   REALD(I) = 0.0
400 CONTINUE

DO 500 I = 1, 12
   REAL2(I) = 0.0
500 CONTINUE

RETURN

END

PROGRAM SIZE:  PROCEDURE - 000117   LINKAGE - 000025   STACK - 000032
0000 ERRORS [(REALCL)FTN-REV16.6]
APPENDIX E

Program Subroutines
## PROGRAM SUBROUTINE INDEX

<table>
<thead>
<tr>
<th>Line</th>
<th>Title</th>
<th>Line</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>78</td>
<td>SWEEP</td>
<td>223</td>
<td>ABNRML</td>
</tr>
<tr>
<td>161</td>
<td>SETARR</td>
<td>1250</td>
<td>ACCUM</td>
</tr>
<tr>
<td>223</td>
<td>ABNRML</td>
<td>1491</td>
<td>ADDRCH</td>
</tr>
<tr>
<td>379</td>
<td>DRATE</td>
<td>2158</td>
<td>ASSIGN</td>
</tr>
<tr>
<td>414</td>
<td>NORMAL</td>
<td>915</td>
<td>BRANCH</td>
</tr>
<tr>
<td>658</td>
<td>WEIBUL</td>
<td>1448</td>
<td>CDFUNC</td>
</tr>
<tr>
<td>712</td>
<td>DATAIN</td>
<td>2272</td>
<td>DATA1</td>
</tr>
<tr>
<td>809</td>
<td>ERRORS</td>
<td>2307</td>
<td>DATA2</td>
</tr>
<tr>
<td>831</td>
<td>TRACE</td>
<td>712</td>
<td>DATAIN</td>
</tr>
<tr>
<td>883</td>
<td>PATHIN</td>
<td>379</td>
<td>DRATE</td>
</tr>
<tr>
<td>899</td>
<td>PATOUT</td>
<td>2483</td>
<td>DUMMY</td>
</tr>
<tr>
<td>915</td>
<td>BRANCH</td>
<td>809</td>
<td>ERRORS</td>
</tr>
<tr>
<td>1029</td>
<td>SUMS</td>
<td>1296</td>
<td>FUNCTS</td>
</tr>
<tr>
<td>1092</td>
<td>PDOT</td>
<td>2379</td>
<td>FPLACE</td>
</tr>
<tr>
<td>1142</td>
<td>QDOT</td>
<td>1766</td>
<td>HEADNG</td>
</tr>
<tr>
<td>1192</td>
<td>RCOLUMN</td>
<td>1937</td>
<td>INTDAT</td>
</tr>
<tr>
<td>1214</td>
<td>PCOSTS</td>
<td>2427</td>
<td>LNKLST</td>
</tr>
<tr>
<td>1250</td>
<td>ACCUME</td>
<td>1835</td>
<td>NDVAL</td>
</tr>
<tr>
<td>1296</td>
<td>FUNCTS</td>
<td>414</td>
<td>NORMAL</td>
</tr>
<tr>
<td>1448</td>
<td>CDFUNC</td>
<td>1538</td>
<td>OUTPUT</td>
</tr>
<tr>
<td>1491</td>
<td>ADDRCH</td>
<td>883</td>
<td>PATHIN</td>
</tr>
<tr>
<td>1538</td>
<td>OUTPT</td>
<td>899</td>
<td>PATOUT</td>
</tr>
<tr>
<td>1766</td>
<td>HEADNG</td>
<td>1214</td>
<td>PCOSTS</td>
</tr>
<tr>
<td>1835</td>
<td>NDVAL</td>
<td>1092</td>
<td>PDOT</td>
</tr>
<tr>
<td>1937</td>
<td>INTDAT</td>
<td>1142</td>
<td>QDOT</td>
</tr>
<tr>
<td>1967</td>
<td>RAWDAT</td>
<td>1967</td>
<td>RAWDAT</td>
</tr>
<tr>
<td>2158</td>
<td>ASSIGN</td>
<td>2210</td>
<td>RCLEAR</td>
</tr>
<tr>
<td>2185</td>
<td>RESET</td>
<td>1192</td>
<td>RCOLUMN</td>
</tr>
<tr>
<td>2210</td>
<td>RCLEAR</td>
<td>2574</td>
<td>REALCL</td>
</tr>
<tr>
<td>2225</td>
<td>REFILL</td>
<td>2225</td>
<td>REFILL</td>
</tr>
<tr>
<td>2272</td>
<td>DATA1</td>
<td>2185</td>
<td>RESET</td>
</tr>
<tr>
<td>2307</td>
<td>DATA2</td>
<td>161</td>
<td>SETARR</td>
</tr>
<tr>
<td>2379</td>
<td>FPLACE</td>
<td>1029</td>
<td>SUMS</td>
</tr>
<tr>
<td>2427</td>
<td>LNKLST</td>
<td>78</td>
<td>SWEEP</td>
</tr>
<tr>
<td>2483</td>
<td>DUMMY</td>
<td>831</td>
<td>TRACE</td>
</tr>
<tr>
<td>2574</td>
<td>REALCL</td>
<td>658</td>
<td>WEIBUL</td>
</tr>
</tbody>
</table>
This appendix consists of a program index of all subroutines by line numbers of the listing in Appendix D and a detailed explanation of selected subroutines. Although each subroutine has explanatory comments in the program listing, as well as in Appendix C (flow charts), some routines require more explanation. In particular, a full description of subroutine FUNCTS is included in this appendix. FUNCTS provides for all user implemented functional relationships needed for modifying cost or flow values. At the present time seven (7) families of functions have been entered to the program and are available to the user (see equations below). These functions are activated by entering appropriate input data to the data file such as that shown in Appendix G.

All functions make use of the operating array FUNCT (10,10) which is assembled from user supplied numbers during the data read in cycle of the program. The FUNCT array has five columns for control operation and five columns for numeric shaping of the functions programmed. The various columns 1 through 10 have the meanings shown in Figure E-1.

Assume that the amount of track from accidents being maintained is to be modified as a linear function of current rail quality. Figure E-2 shows the various steps in implementing this function by using program function number 6 which is a linear operating function using the present or current state quality. Figure E-2 shows the original graphic form of the desired function, its associated equation, the input data for activating it, and two program outputs associated with its use. The two outputs of the program are; the prompting message to the user at the top of the run output, and then the "flow" outputs from each time point of the simulation.

The prompting message lists the path and functions that are operating on those paths so that the user knows when functional "modifiers" are operating on the base data for those paths.

Seven functional families of equations are programmed for use. In order that a specific functional equation is from any of the seven forms programmed, the user must supply input data values for A, B, C, X₀ and Y₀. See example at
end of this appendix on how these values can generate different functions.

Family of equation form by programmed function number in Subroutine FUNCTS:

1. Linear  \[ F(X) = A \cdot X + B \]

2. Power  \[ F(X) = A \cdot (X - X_0)^B + Y_0 \]

3. Exponential  \[ F(X) = A \cdot (\text{Exp}(B \cdot (X - X_0))) + Y_0 \]

4. Weibull Probability Density  \[ F(X) = \frac{(B \cdot X)^{B-1}}{A^B} \cdot (\text{Exp}(-\frac{X}{A}) ) \]

5. Hyperbolic Tangent  \[ F(X) = A (1+Z_1) + C(1-Z_2) + Y_0 \]

   where \( Z_1 \) and \( Z_2 \) are explicitly given in Subroutine "FUNCTS"

6. Linear on Quality  \[ F(Q) = A \cdot Q/B + C \]

7. Explicit on Flow  \[ F(f_1) = A\left(\frac{f_1}{f_1 + A \cdot B}\right) \]

where

\( X \) = a dummy variable, usually simulation time in years during a simulation run. Independent variable selectable, see Figure E-1 column 5.

\( A, B, C, X_0, Y_0 \) = Constants input by user (See example at the end of this appendix).

\( Q \) = Quality of component at time step during simulation.

\( f_1 \) = Path flow of component during simulation on path designated by user.

*This results from Weibull cumulative distribution which is defined as

\[ f(x) = 1 - \text{Exp}[-\left(\frac{X}{A}\right)B] \]

where \( A > 0, B > 0, X > 0 \)

The failure rate \( H(X) \) is given by \[ H(X) = \frac{B}{A^B} (X^{B-1}) \]
(FUNCT ARRAY Rows Presently Limited to 10, Expandable Indefinitely by User, See Program Common Block R2)

<table>
<thead>
<tr>
<th>PATH</th>
<th>FUNCTION CODE</th>
<th>COST</th>
<th>NUMBER OF MULTIPLIERS LEFT</th>
<th>INDEPENDENT VARIABLE</th>
<th>MAINTENANCE ACTION DIAGRAM PATH NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

User Supplied Data for Shaping Functions

- See Figure 3-1, Section 3 for number meanings.
- Handled automatically by program if user enters more than one function modifier for this path item.
- See Figure 3-1, Section 3 for number meanings.
- Program function number itemized in Subroutine "FUNCTS".
- Maintenance action diagram path number.

FIGURE E-1. FUNCT ARRAY SPECIFICATIONS BY COLUMN NUMBER
Graphic Display of Function to be Used in Model to Modify Accident "Flow" on Path 17.

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>FORM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rail Quality, Q</td>
<td>Accident Track Multiplier</td>
</tr>
<tr>
<td>Linear Equation of Above Graphic Display. See Function 6 of &quot;FUNCTS&quot; Subroutine</td>
<td></td>
</tr>
<tr>
<td>Input Data for Activating and Describing Function to be Used in Modifying Track Flow with Track Quality. See data in Appendix G.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PATH</th>
<th>FUNC</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>X0</th>
<th>Y0</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>7</td>
<td>2.000 00</td>
<td>3.900 01</td>
<td>0.000-01</td>
<td>0.000-01</td>
<td>0.000-01</td>
</tr>
<tr>
<td>17</td>
<td>6</td>
<td>2.900-01</td>
<td>1.000-01</td>
<td>7.100-01</td>
<td>0.000-01</td>
<td>0.000-01</td>
</tr>
</tbody>
</table>

Sample Program Output "Flows" Along Path 17 With and Without Function 6 Operating Over the Ten Year Simulation Period

<table>
<thead>
<tr>
<th>Time Yr</th>
<th>Flow* No Function</th>
<th>Flow* With Function Modifier on Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>8.00</td>
<td>8.00</td>
</tr>
<tr>
<td>.5</td>
<td>6.09</td>
<td>5.67</td>
</tr>
<tr>
<td>1.0</td>
<td>5.21</td>
<td>4.68</td>
</tr>
<tr>
<td>1.5</td>
<td>4.78</td>
<td>4.22</td>
</tr>
<tr>
<td>2.0</td>
<td>4.56</td>
<td>3.99</td>
</tr>
<tr>
<td>2.5</td>
<td>4.44</td>
<td>3.87</td>
</tr>
<tr>
<td>3.0</td>
<td>4.38</td>
<td>3.81</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>10</td>
<td>4.31</td>
<td>3.74</td>
</tr>
</tbody>
</table>

*Component 1 of Class 1, 2, 3 Track

FIGURE E-2. SAMPLE APPLICATION OF A SINGLE FUNCTION TO MODIFY THE AMOUNT OF TRACK FROM TRACK ACCIDENTS BEING MAINTAINED DURING SIMULATION MAINTENANCE ACTION DIAGRAM EXAMPLE #1, PATH 17
In a similar manner, any other path may have functional "modifiers" operating on their flows or costs. One must merely generate the form of the functional dependence and activate the "modifier" function on the path component flows and/or costs known to be affected by the relation. If two or more functional "modifiers" are known to affect the path data then each is entered separately. During the simulation, then both functions will "modify" the base information carried by taking the product of the two functions.

Actually "any" number of functions may be used to modify the path data as long as the input shape parameters can be given to prescribe the relationships used in the simulation modeling.

As an example of how the shape parameters can influence the nature of the programmed family of curves consider the displays of Figure E-3. The upper plot of this figure demonstrates the curve shape obtained from programmed Equation #4 by setting parameter B equal to .5, 1, 2 and 4, respectively. The plot is made for all dummy variable values of "X" such that $0 \leq X \leq 3A$ where A is one of the parameters in programmed Equation #4.

This type of programmed formulation can be used to produce scalar outputs which may be known to depend in general on some "state variable" or "time" during the simulation. A collection of mechanical components whose rate of failure sometimes depends upon the age of the components can often be described by a distribution such as the Weibull.
WEIBULL PROBABILITY DENSITY
Programmed Equation #4
for
\[ B = .5, 1, 2, 4 \]
\[ X = 0 \ldots 3A \]

WEIBULL CUMULATIVE DISTRIBUTION
for
\[ B = .5, 1, 2, 4 \]
\[ X = 0 \ldots 3A \]

FAILURE RATE FUNCTION
\[ H(x) \]
for
\[ B = .5, 1, 2, 4 \]
\[ X = 0 \ldots 3A \]

FIGURE E-3. DIFFERENT FUNCTIONAL FORMS OF WEIBULL DISTRIBUTIONS RESULTING FROM NUMERIC ADJUSTMENT OF SHAPE PARAMETERS A AND B
APPENDIX F

Computer Program Control and Operating Data File Structuring
This appendix describes program array structuring and element by element descriptions needed for understanding their operation and use.

COMMON INTEGER ARRAYS:

<table>
<thead>
<tr>
<th>PROGRAM LOCATION AND ARRAY NAME</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>COMMON/I1/ERROR(10)</td>
<td>Used for program diagnostics. The ten elements of this vector are used in subroutine ERRORS for printing and tracing typical errors resulting from possible clerical mistakes.</td>
</tr>
<tr>
<td>COMMON/I2/IFILE(100)</td>
<td>A user file whose various values define various other program array lengths, as well as input/output controls. Each vector element of this file used by the program has a variable &quot;name&quot; used in the program. The full length of this data file with variable names and their meanings is listed at the end of this Appendix.</td>
</tr>
<tr>
<td>COMMON/I3/IFLAG(10)</td>
<td>Used for program diagnostics. These ten values control the printed error messages selected by subroutine TRACE. Setting IFLAG(1) = 0 will normally disable or shut this option off.</td>
</tr>
<tr>
<td>COMMON/I7/INODE(6,300)</td>
<td>Up to a maximum of 300 nodes. This array carries six numeric values which correspond to each of the various path branching or sum points (nodes) of the maintenance action diagram. The six columns contain the following information. Column Numberic Value is:</td>
</tr>
<tr>
<td>1.</td>
<td>Incoming path number to node.</td>
</tr>
<tr>
<td>2.</td>
<td>In or out path number depending upon whether path is a summing node or branch (splitting) node, respectively.</td>
</tr>
</tbody>
</table>
3. Outgoing path number of corresponding node.

4. Zero (0) if a sum node, One (1) if a branch node.

5. Zero (0) if split fraction of path flow based on total incoming flow. One (1) if split fraction of path flow is based on defective portion of incoming flow to the node.

6. This column only has meaning for branching nodes (one path in-two out). Action diagrams branch nodes are always referred to with the following convention.

```
1
```
```
Incoming path
```
```
Node point
```
```
Path 3
(Next clockwise path on node from Path 2)
```
```
Path 2
(First clockwise path from incoming)
```

If the numeric value in column six is zero (0) then path 2 is the reference path. If it is one (1) then path 3 is the reference path.

The reference path flow given in the input data is used at the beginning of the program execution to compute the node split fraction automatically by the program. See RNODE array values.
Up to fifteen (15) integer codes are allowed by this array to be attached to each path (up to 600) of the maintenance action diagram of the cost model. Not all of the fifteen have been designated. Those that have are to be associated with the meanings given in the data entry and formsetting procedure of the Appendix G data example.

This array holds the cost summary from all paths. The dimension of this array is to be at least as large as the product of three variables entered by the user in the IFILE ARRAY, i.e., IBDM*LSTCP*LSTCL. See notation in IFILE at the end of this Appendix.

Used by the FUNCTS subroutine in all computations requiring formula type relationships involving costs or system path flows known to have explicit dependencies which can be expressed by a formula. Although this array presently has only 10 rows for demonstration purposes it should be dimensioned to have at least as many rows as there would be functional (shape parameter formula related) data lines in the user generated input cost data (See Appendix G for example). The functional shape parameters and numeric control items for this array for each column (for up to ten columns) are explained in detail in Appendix E.
This array is cleared and filled at the beginning of the program execution without user control from the flow information supplied by the user as shown in the example of Appendix G. The numeric elements of this array are the fractional path flow split values associated with each node. Thus, dimension 300 must be at least as large as the number of nodes (sum and branch) in maintenance action diagram being used in the simulation. The dimension 30 corresponds to the product of the number of track structural components (typically 5) and FRA track classes (typically 6) used in the simulation.

This array is used during the simulation to store computations on the system populations (amount of each component), their rates of change, the system qualities (one for each component and class) and their rates of change. There are five (5) sets of each of these four (4) items stored for the separate Runge-Kutta computations. Thus the dimension 20. The thirty (30) dimension corresponds to a separate set of 20 for each track structural component and FRA track class (5 times 6 respectively).

This array has two conceptual blocks of information associated with it and only a small portion of this array has been made use of. The first 50 elements were set up to carry certain real time values which could be available in common. The first three elements are simulation starting time (in years),
simulation time step in years, and a final reference time point in years which stops the simulation cycle. The final time entered by the user to array element three is usually one time step less than the last time point to be simulated. Array element 4 is not used. Elements five and six give the number of time steps that must pass before an output cost table is printed. Although a full set of costs and/or flows are generated at each time step the user can set the printed output cycle separately with these variables. This condenses the amount of output that must be printed if small time steps are used in the simulation.

The last 50 elements (50 through 99) are intended as simple real value numbers which are defined to be related through the simulation state variable "time" and the fifty variable parameters listed in Figure 3-1 of Section 3.

This array holds the user set of Weibull constants used for depicting the amount of track component degradation which can occur naturally. There are three parameters; average age, distribution shape parameter $\beta$, and characteristic life of the Weibull distribution. There is allowed a separate distribution for each track structural component and each FRA track class ($5 \times 6$).

The Weibull distribution is general enough to describe component systems which have either
an increasing or decreasing failure rate depending upon the value of \( \beta \). If \( \beta = 1 \) then the exponential distribution results, and the failure rate is constant. The Weibull distribution is used as a time rate of failure descriptive technique and has been found to be useful for describing wear out rates for large collections of components such as bearings. (See References 9 and 11.)

These arrays are used in passing the program revision number, the maintenance action "schematic diagram" used, and the cost data file number used in the simulation.

The program revision number REVISN(8) is set at the very top of the program and should be renumbered if altered in the future.

Revision '6' was the final Prime Computer version and revision '8' was the final Dec 10 version. The slight differences in the two were alterations for machine compatibility.

This array is used to hold all cost and flow information associated with each path of the maintenance action diagram. For convenience the data by row number (up to 600 rows in this example) in this array corresponds to the information attached to the path of the same number in the maintenance action diagram used in the simulation. If paths in the action diagram are not numbered sequentially then some space will be wasted. Redimensioning the 600 value of this array for the minimum
number of paths used in the simulation will conserve computer space.

The 750 dimension of this array can also be decreased in size to fit the simulation run if space in the computer needs to be conserved. This dimension must be at least as large as the product of program variables IBDM, LSTEL, and LSTCP. This corresponds to the minimum block of numbers (IBDM) needed for each FRA track class (LSTCL), and number of track structural components (LSTCP) used in the simulation. The IBDM minimum data block size must be at least 8 if only one cost code is required to cover costs occurring on any path. For example purposes IBDM was set at 20 and 25 for the data in Appendices G and H, respectively. This allowed for several types of costs (such as labor, materials, scrap, etc.) to be kept separately in the simulation.

**Dimension Arrays**

**Dimension RPATH(750), IPATH(15)**

RPATH along with RPATH1, RPATH2, and RPATH3 are vectors used in passing individual real number path costs and flow information between the RARRAY and the various program subroutines. The 750 dimension must be at least as large as the product of program variables IBDM*LSTCL*LSTCP. The structure of this vector and its elements are displayed in Figure F-1.

IPATH, IPATH1, IPATH2, and IPATH3 are vectors used in passing integers (typically zero and one type of control indices associated with
**RPATH\#(N)**

Disk stored path data where

\[ n = 1, 2, 3 \text{ or } (\text{IBM} + \text{LSTCL}) \]

All numbers are double precision real.

LSTCP = number of track structural components

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>COMPONENT 2</th>
<th>COMPONENT LSTCP</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD</td>
<td>RECORD</td>
<td>RECORD</td>
</tr>
</tbody>
</table>

**FIGURE F-1.** Array storage structure for path data by array subscript number
the paths) between the IARAY and the various subroutines. The 15 dimension was selected arbitrarily before the actually needed numbers of this kind of integer was fully established. Actually only eight are used during the program execution, (for all indicators I=Yes, 0=No)i.e.,

\[ IPATH(1) = \text{The path number of the maintenance action diagram path to which the rest of the array elements are to be associated. Variable IPH.}\]

\[ IPATH(5) = \text{Path type. Variable ITYPE.}\]

\[ IPATH(8) = \text{Maintenance action Repair Block number in which this path resides. This provides cost outputs to be summarized by Repair Block or type. Variable IRPB.}\]

\[ IPATH(10) = \text{An indicator as to whether this path contains costs. This indicator allows the program to skip the cost computations if the path has no costs associated with it. Variable ICST.}\]

\[ IPATH(11) = \text{An indicator as to whether there is a net change in the simulated system population on this path. Source or sink flows such as from manufacturers or scrap yards require this information. Variable IPDT.}\]

\[ IPATH(12) = \text{An indicator showing that the quality of the components on this path are altered from the labor or work applied to them. Variable IQDT.}\]
IPATH(13) = Indicator of manufacturing path.
Variable IMFG.

IPATH(14) = Indicator allowing this path to be skipped at printout time during the simulation. This allows printouts of small portions of the maintenance action diagram to be obtained. Variable IPRT.

Dimension Place(10), RUNN(10) Handle heading information for tabular outputs.

Copies of IFILE, RFILE, IFLAG, and WEIBL are listed below. These files along with the input cost data file of Appendix G are the only non-zero files needed to obtain the sample run output file shown in Appendix I.
FILE Listing
(For Example #1 Run, Appendix I)

BASE MOD 2
SHAKER RESEARCH CORP.

1 1 ISTDND FIRST NODE OF SCHEMATIC TO BEGIN COMPUTATIONS
9 2 LSTDND LAST NODE OF SCHEMATIC TO STOP
1 3 ISTPH FIRST PATH & PATH DATA FILE NUMBER ON DISK
18 4 LSTPH LAST PATH & PATH DATA FILE NUMBER ON DISK
1 5 ISTCIP FIRST COMPONENT NUMBER USED IN COMPUTATIONS
2 6 LSTCP LAST COMPONENT NUMBER USED IN COMPUTATIONS
1 7 ISTCL FIRST TRACK CLASS TO USE IN COMPUTATIONS
2 8 LSTCL LAST TRACK CLASS TO USE IN COMPUTATIONS
6 9 LSTCT COLUMN CONTAINING FIRST COST MINUS ONE
13 10 LSTCT TOTAL COST COLUMN BLOCK SIZE MINUS ONE
15 11 IIDM DIMENSION OF IPATH ARRAY
30 12 IRDM DIMENSION OF RPATH ARRAYS
13 13 IREFP SCHEMATIC DIAGRAM IN USE REFERENCE PATH NUMBER
6 14 LSTFL TOTAL FLOW COLUMN BLOCK SIZE - EQUIVALENT TO ISTCT
1 15 IOUT PATH DATA WRITE TO DISK CONTROL INDICATOR
20 16 IBDM NUMBER OF FLOW PLUS COST COLUMNS (BLOCK SIZE)
50 17 MCST Maximum number of data codes
44 18 LCTCD NUMERIC VALUE OF LAST USER DEFINED COST CODE
5 19 NPASS STATE BLOCK REFERENCE ROW IN NORMAL
0 20 AVAILABLE
10 21 IUNIT FORTRAN UNIT DEVICE NUMBER FOR OUTPUT PRINTING
11 22 IUNIT FORTRAN UNIT FOR READING IN RAW DATA
1 23 IUNIT FORTRAN UNIT FOR PRINTING DIAGNOSTIC STATEMENTS
4 24 LUNITST NUMBER OF COLUMNS IN THE STATE ARRAY
1 25 IPNMT INDICATOR FOR NEED OF PATH FLOW/COST PRINTOUT
0 26 AVAILABLE
0 27 ISPRT INDICATOR FOR NEED OF COST/STATE PRINTOUT
0 28 IFMTS INDICATES COST/STATE PRINT OPTIONS
1 29 NDOUT INDICATES NEED FOR FINAL COST/STATE PRINTOUT
1 30 IFMT INDICATES FINAL PRINT OPTIONS
1 31 N1 STATE POPULATION REFERENCE ROW
3 32 N2 STATE QUALITY REFERENCE ROW
17 33 N1 STATE POPULATION UPDATE ROW
19 34 N2 STATE QUALITY UPDATE ROW
1 35 ICOST INDICATOR FOR NEED OF COST COMPUTATIONS
0 36 AVAILABLE
0 37 AVAILABLE
1 38 NBLOCK NUMBER OF REPAIR BLOCKS IN SYSTEM
0 39 AVAILABLE
44 40 ICODE LAST COST CODE - EQUIVALENT TO LCTCD
1 41 RAW DATA PROCESSING INDICATOR
1 42 MODEL RUN INDICATOR
0 43 AVAILABLE
0 44 AVAILABLE
0 45 AVAILABLE
0 46 AVAILABLE
0 47 AVAILABLE
0 48 AVAILABLE
0 49 AVAILABLE
0 50 AVAILABLE
0 51 AVAILABLE
<table>
<thead>
<tr>
<th>Line</th>
<th>Value</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>052</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>053</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>054</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>055</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>056</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>057</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>058</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>059</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>060</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>061</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>062</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>063</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>064</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>065</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>066</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>067</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>068</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>069</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>070</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>071</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>072</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>073</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>074</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>075</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>076</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>077</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>078</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>079</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>080</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>081</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>082</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>083</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>084</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>085</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>086</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>087</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>088</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>089</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>090</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>091</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>092</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>093</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>094</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>095</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>096</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>097</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>098</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>099</td>
<td>AVAILABLE</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>AVAILABLE</td>
<td></td>
</tr>
</tbody>
</table>
RFILE Listing
(For Example #1 Run, Appendix I)

0.0 1 STIME  STARTING TIME FOR MAIN/NORMAL LOOP
0.1 2 DTIME  TIME STEP FOR MAIN/NORMAL LOOP
1.9 3 TIMEL  FINISH TIME FOR TIME RUN
0.0 4 AVAILABLE
5.0 5 OUTLMD OUTPUT FREQUENCY COUNTER
5.0 6 OUTLFQ OUTPUT FREQUENCY
0.0 7 AVAILABLE
0.0 8 AVAILABLE
0.0 9 AVAILABLE
0.0 10 AVAILABLE
0.0 11 AVAILABLE
0.0 12 AVAILABLE
0.0 13 AVAILABLE
0.0 14 AVAILABLE
0.0 15 AVAILABLE
0.0 16 AVAILABLE
0.0 17 AVAILABLE
0.0 18 AVAILABLE
0.0 19 AVAILABLE
0.0 20 AVAILABLE
0.0 21 AVAILABLE
0.0 22 AVAILABLE
0.0 23 AVAILABLE
5.0 24 NORMAL OUTPUT FREQ.
5.0 25 NORMAL OUTPUT START
0.0 26 AVAILABLE
0.0 27 AVAILABLE
0.0 28 AVAILABLE
0.0 29 AVAILABLE
0.0 30 AVAILABLE
0.0 31 AVAILABLE
0.0 32 AVAILABLE
0.0 33 AVAILABLE
0.0 34 AVAILABLE
0.0 35 AVAILABLE
0.0 36 AVAILABLE
0.0 37 AVAILABLE
0.0 38 AVAILABLE
0.0 39 AVAILABLE
0.0 40 AVAILABLE
0.0 41 AVAILABLE
0.0 42 AVAILABLE
0.0 43 AVAILABLE
0.0 44 AVAILABLE
0.0 45 AVAILABLE
0.0 46 AVAILABLE
0.0 47 AVAILABLE
0.0 48 AVAILABLE
0.0 49 AVAILABLE
0.0 50 AVAILABLE
51  51 AVAILABLE
0.0 52 AVAILABLE
53  53 AVAILABLE
54  54 AVAILABLE
<table>
<thead>
<tr>
<th>Address</th>
<th>Value</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>55.00</td>
<td>55</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>56</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>57</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>58</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>59</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>60</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>61</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>62</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>63</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>64</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>65</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>66</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>67</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>68</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>69</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>70</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>71</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>72</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>73</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>74</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>75</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>76</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>77</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>78</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>79</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>80</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>81</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>82</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>83</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>84</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>85</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>86</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>87</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>88</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>89</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>90</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>91</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>92</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>93</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>94</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>95</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>96</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>97</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>98</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>99</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td>100</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td></td>
<td>EXTRA LINE AVAILABLE</td>
</tr>
<tr>
<td>0.0</td>
<td></td>
<td>EXTRA LINE AVAILABLE</td>
</tr>
</tbody>
</table>

F-16
INODE Listing
(Example Run #1, Appendix I)

**Schematic Diagram Number:**

<table>
<thead>
<tr>
<th>Data File Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>13 14 1 1 0 1</td>
</tr>
<tr>
<td>1 2 18 1 1 1</td>
</tr>
<tr>
<td>13 14 17 1 1 1</td>
</tr>
<tr>
<td>18 17 3 0 0 0</td>
</tr>
<tr>
<td>3 5 4 1 0 0</td>
</tr>
<tr>
<td>5 6 7 0 0 0</td>
</tr>
<tr>
<td>7 9 8 1 1 1</td>
</tr>
<tr>
<td>9 11 10 1 1 1</td>
</tr>
<tr>
<td>4 11 12 0 0 0</td>
</tr>
</tbody>
</table>

REST OF ARRAY FILLED WITH ZEROS
**WEIBL Listing**

*(For Example Run #1, Appendix I)*

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>20.0</td>
<td>20.0</td>
<td>20.0</td>
<td>20.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>2.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>32.0</td>
<td>32.0</td>
<td>32.0</td>
<td>32.0</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**NOTE:**

δ = 20 - fixed average life 20 yrs.

β = 2 - failure rate shape parameter

α = 32 - characteristic life years

For this example, failure rate is assumed constant for each component and given by

\[
\frac{\beta}{\alpha} \left( \frac{\delta}{\alpha} \right)^{\beta-1} \approx 4\%
\]
APPENDIX G

Input Cost Data Example #1
(For Simplified Track Maintenance Action Diagram #1)
This appendix contains a listing of an example input data file of costs and path flows for the example run output listed in Appendix I. Also included here is an explanation of the file coding structure so the reader may generate a different file for simulation if desired.

An input data coding structure which is compatible with either a computer file or card entry technique has been adopted. Maintenance data in this scheme are coded onto a series of cards or file lines. Each card or file line contains a part of the data required for the cost simulation model. (See listing at end of appendix for a complete data listing of the cost and flow values for example number 1 track maintenance system simulation.)

Ultimately, every data entry to the cost model program is used by the computer to generate a dollar cost for performing every maintenance operation considered. Certain data conventions have been followed in order to place the costing data in a computerized format. Figure G-1 is an overall schematic of the data card entry format assumed. Each card or data entry is read in free field format. Thus, all spaces or commas are read as numeric delimiters separating the numbers entered.

The path number is the first number on each data line. Immediately following the "path number" each data line (card) has a single digit entry (0-9) in column five. This entry refers to the kind of data that follows it on that line. Data beyond column five can be path descriptive numbers, direct path costs and flows, or function shape parameters.

Figure G-1 shows the position of the data entry codes. Codes 3 through 8 are still undefined, but numbers 0-2 and 9 have the following meanings:

0 - Refers to the fact that path information will be found in the numeric data space following this entry.

1 - Indicates that the numeric data that follow on this line comprise a direct cost or flow input set of numbers.

2 - Is used to designate that the numeric data will be function
Schematic Diagram Path Number

Numeric Data Space for up to ten separate real numbers

XXX_ X_XX

/Data Comments

DATA ENTRY CODE

0 = Refers to overall path information in 'numeric data space'
1 = 'Numeric Data' to be associated with cost or flow code shown
2 = 'Numeric Data' are for functional formulas (shape factors, etc.)
3-8 = Available
9 = Indicates more data on next card (line) for this data entry code.

FIGURE G-1. LAYOUT OF DATA CARD OR LINE FILE ENTRY FORMAT
parameters. These shape parameters will be passed during program execution to user defined program equations for function evaluation.

3-8 - Available for definition.

9 - Indicates the numbers for this line (card) exceed the space provided on the card above it; therefore, read the next card and append.

As shown in Figure G-1 each card contains a reference "path number" which is keyed to the maintenance action diagram of the example modeled track maintenance system. In addition, each data card contains a two digit data descriptor located in columns 7 and 8. A list of the data descriptors 30 through 49 are given in Figure G-2. Codes 30 to 49 are intended for the various cost inputs. This set of 20 data subcategories 30-49 can be expanded by user selection if more are needed. This data structure allows each path cost to be entered and/or examined visually. Comments placed after the slash (/) are useful in checking over the data entries and their meanings.

Conventions have been adopted for the data to be entered in the "numeric data space" of each data card. There is a separate entry convention for each data entry code 0 through 2. The following paragraphs discuss the various numbers required for each of these three entry codes.

Entry Code 0 - Path Information
Requires a minimum of two numbers after it. Each number is to be separated by a space or a comma. These are a component number and an FRA track class number.

Entry Code 1 - Cost or Flow Information
Requires between two and five numbers. If the data code descriptor is:
20 - enter starting path component flow after it;
21 - enter starting path component quality after it;
30 through 49 - enter appropriate cost numbers.
<table>
<thead>
<tr>
<th>Data Descriptor Code</th>
<th>Description of Data Under This Code</th>
<th>UNITS OF MEASURE</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>Quantity of Component</td>
<td>First Numeric Entry Beyond Data Descriptor Code</td>
</tr>
<tr>
<td>21</td>
<td>Quality of Component</td>
<td>Second Numeric Entry Beyond Data Descriptor Code</td>
</tr>
<tr>
<td>22</td>
<td>Amount of Bad Component</td>
<td>Third Numeric Entry Beyond Data Descriptor Code</td>
</tr>
<tr>
<td>23</td>
<td>Amount of Good Component</td>
<td>Number of Miles Per Year</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bad Miles/Total Miles</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of Miles Per Year</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of Miles Per Year</td>
</tr>
<tr>
<td>30</td>
<td>Material Costs</td>
<td>Items Per Mile</td>
</tr>
<tr>
<td>31</td>
<td></td>
<td>Dollars Per Item</td>
</tr>
<tr>
<td>32</td>
<td>Equipment Costs</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>33</td>
<td></td>
<td>Dollars Per Hour</td>
</tr>
<tr>
<td>34</td>
<td>Delay Costs</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>35</td>
<td></td>
<td>Dollars Per Hour</td>
</tr>
<tr>
<td>36</td>
<td>Scrap Return Costs</td>
<td>Items Per Mile</td>
</tr>
<tr>
<td>37</td>
<td></td>
<td>Dollars Per Item</td>
</tr>
<tr>
<td>38</td>
<td>Contracted Costs</td>
<td>Contracts Per Mile</td>
</tr>
<tr>
<td>39</td>
<td></td>
<td>Dollars Per Contract</td>
</tr>
<tr>
<td>40</td>
<td>Travel and Living</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>41</td>
<td></td>
<td>Dollars Per Hour</td>
</tr>
<tr>
<td>42</td>
<td>Trackmen Labor</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>43</td>
<td>Mechanic Labor</td>
<td>Dollars Per Hour</td>
</tr>
<tr>
<td>44</td>
<td>Machine Operator Labor 1</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>45</td>
<td></td>
<td>Dollars Per Hour</td>
</tr>
<tr>
<td>46</td>
<td>Foremen Labor</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>47</td>
<td></td>
<td>Dollars Per Hour</td>
</tr>
<tr>
<td>48</td>
<td>Supervisor Labor</td>
<td>Hours Per Mile</td>
</tr>
<tr>
<td>49</td>
<td></td>
<td>Dollars Per Hour</td>
</tr>
</tbody>
</table>

FIGURE G-2. ASSIGNED DATA DESCRIPTORS 20-49 AND THEIR UNITS OF MEASURE FOR CODING
Entry Code 2 - **Function Data**

Requires two data cards. The first card is not used for computation and contains a set of designator codes for reminder commentary only. The second card with an entry code 2 will carry seven numbers beyond the entry code. These seven numbers will include one designator code, one user defined function number, and up to five shape parameters for that function being activated.

Entry Code 3-8 - Available for definition.

Entry Code 9 - Append this data card entry to the one above.

**Descriptive Commentary for Cost Data File for Example Run #1**

As explained in Section 6.0, this run was intended to portray a hypothetical small railroad for showing some aspects of the simulation technique and methodology. The railroad and its physical plant (pertinent to this procedure) are fully described in Figure 6-1 of the main body of the report. The values displayed in that figure were contrived simply for this example run; however, the numeric magnitudes of the data selected could very well be found in railroads of this size.

Out of seven hundred miles of track a typical road of today might have the distribution shown between jointed and welded rails (750 to 250, respectively). Accident rates were chosen in a similar fashion based on a national annual average of about 2000 track incidents for the 200,000 miles of track in use, thus .01/mile. Since different accident rates can exist for each FRA track class or type of rail, separate values of .011, .02, .01 and zero accidents per mile of track were entered respectively.

Demonstration of how the track accident rate could vary with a state variable such as "quality" is implemented on only the jointed rail of FRA classes 1, 2, 3 (see Figures 6-1 and 6-2 of the report).

The following tabular commentary is intended to elucidate some of the numeric entries contained in the listing at the end of this appendix.
# ANNOTATIONS TO LINE ENTRIES IN LISTING OF
## "COST DATA FILE FOR EXAMPLE RUN #1"
### CONTAINED AT END OF THIS APPENDIX

<table>
<thead>
<tr>
<th>Line Counted Top From Entry</th>
<th>Description or Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Commentary line passed to run output</td>
</tr>
<tr>
<td>2</td>
<td>Action Diagram path number (&quot;in-use&quot; path not actually numbered on Fig. 4-1)</td>
</tr>
<tr>
<td>2</td>
<td>The following entries describe this path</td>
</tr>
<tr>
<td>2</td>
<td>These entries are for jointed rail (Comp. 1)</td>
</tr>
<tr>
<td>2</td>
<td>These entries are for track class 1</td>
</tr>
<tr>
<td>2</td>
<td>This path has no cost</td>
</tr>
<tr>
<td>2</td>
<td>This path does not change the system population</td>
</tr>
<tr>
<td>2</td>
<td>This path does not change the system quality</td>
</tr>
<tr>
<td>2</td>
<td>This path is not from a manufacturing source</td>
</tr>
<tr>
<td>2</td>
<td>Path not in a repair block</td>
</tr>
<tr>
<td>2</td>
<td>Skip this path on printout</td>
</tr>
<tr>
<td>2</td>
<td>Comments only</td>
</tr>
<tr>
<td>3</td>
<td>Action Diagram Path number (&quot;in-use&quot; path not actually numbered on Fig. 4-1)</td>
</tr>
<tr>
<td>3</td>
<td>Comment 1 on this</td>
</tr>
<tr>
<td>3</td>
<td>Data descriptor code Fig. 3-1 &quot;Quantity of Track&quot; entered next</td>
</tr>
<tr>
<td>3</td>
<td>Zero miles of track</td>
</tr>
<tr>
<td>4</td>
<td>Action Diagram path number (&quot;in-use&quot; path not actually numbered on Fig. 4-1)</td>
</tr>
<tr>
<td>4</td>
<td>Component 1 on this</td>
</tr>
<tr>
<td>4</td>
<td>Data descriptor code Fig. 3-1 &quot;Quality of Track&quot; to be entered next</td>
</tr>
<tr>
<td>4</td>
<td>Quality = 0</td>
</tr>
<tr>
<td>5</td>
<td>Action Diagram path number (see Fig. 4-1)</td>
</tr>
<tr>
<td>5</td>
<td>Following entries describe this path</td>
</tr>
<tr>
<td>5</td>
<td>These entries are for jointed rail (Comp. 1)</td>
</tr>
<tr>
<td>5</td>
<td>These entries are for track class 1</td>
</tr>
<tr>
<td>5</td>
<td>This path has no cost</td>
</tr>
<tr>
<td>Line # Counted From Top</td>
<td>Entry</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>6</td>
<td>012</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>.0909</td>
</tr>
<tr>
<td>7</td>
<td>013</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>01</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>013</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>700</td>
</tr>
<tr>
<td>9</td>
<td>013</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>21</td>
</tr>
<tr>
<td>9</td>
<td>.1</td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>005</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>01</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>005</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
</tr>
<tr>
<td>20</td>
<td>70</td>
</tr>
<tr>
<td>Line # Counted From Top</td>
<td>Entry</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>21</td>
<td>005</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
</tr>
<tr>
<td>21</td>
<td>20</td>
</tr>
<tr>
<td>21</td>
<td>50</td>
</tr>
</tbody>
</table>

| 22                     | 005   | Path #                      |
| 22                     | 2     | Following is function input data |
| 22                     | 20    | Function "modifies" rail quantity on this path |
| 22                     | 7     | Data to be entered for Equation #7, see Subroutine "FUNCTS" |
| 22                     | 2.0   | "A" parameter, See Appendix E |
| 22                     | 39.0  | "B" parameter, see Appendix E |

More inputs for other paths

| 50                     | 004   | Path #                      |
| 50                     | 1     | Data descriptor next |
| 50                     | 34    | Data descriptor (see Fig. 3-1) costs entered next will be "delay related" |
| 50                     | 1.0   | Hours delay per mile of track on this path (see Fig. G-2) |
| 50                     | 25000 | Dollars per hour train delay cost |
| 50                     | 1.0   | Multiplying factor |

More inputs for other paths

| 73                     | 008   | Path #                      |
| 73                     | 0     | Following data describe path |
| 73                     | 02    | Component #                 |
| 73                     | 1     | Track class                 |
| 73                     | 1     | This path has costs         |

See Fig. 5-1
<table>
<thead>
<tr>
<th>Line # Counted From Top</th>
<th>Entry</th>
<th>Description or Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>73</td>
<td>-1</td>
<td>This path reduces the system population</td>
</tr>
<tr>
<td>73</td>
<td>0</td>
<td>No</td>
</tr>
<tr>
<td>73</td>
<td>0</td>
<td>Not a manufacturing path</td>
</tr>
<tr>
<td>73</td>
<td>1</td>
<td>Associate this path with &quot;repair&quot; block 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>See Fig. 5-1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>More inputs for other paths</td>
</tr>
<tr>
<td>128</td>
<td>010</td>
<td>Path #</td>
</tr>
<tr>
<td>128</td>
<td>1</td>
<td>Data Descriptor code next</td>
</tr>
<tr>
<td>128</td>
<td>35</td>
<td>Code &quot;35&quot; relates cost data to be entered with scrap (see Fig. 3-1)</td>
</tr>
<tr>
<td>128</td>
<td>1.0</td>
<td>Items per mile (see Fig. G-2)</td>
</tr>
<tr>
<td>128</td>
<td>-10000</td>
<td>Scrap (negative) dollars per item removed (see Fig. G-2)</td>
</tr>
<tr>
<td>128</td>
<td>1.0</td>
<td>Multiplier on costs (see Fig. G-2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>More entries</td>
</tr>
<tr>
<td>221</td>
<td>999</td>
<td>Last data input truncates read operation &quot;999&quot;</td>
</tr>
</tbody>
</table>
COST DATA FILE FOR EXAMPLE RUN #1

DATA FILE NUMBER : 0001

014 001 110 000001 / TO USE
014 1200 / TO USE
014 120 0.0 / TO USE
012 001 1.1 / TO USE
012 120 .0909 / TO USE
013 001 1 / IN USE
013 120 700 / IN USE
013 121 .10 / IN USE
013 002 1 / IN USE
013 120 50 / IN USE
013 121 .10 / IN USE
013 002 2 / IN USE
013 120 150 / IN USE
013 121 .10 / IN USE
005 001 1.1 / TO REPAIR
005 120 70 / TO REPAIR
005 220 50 / TO REPAIR
005 220 72.0 39.0 / TO REPAIR
005 121 1.0 / TO REPAIR
005 002 1.1 / TO REPAIR
005 120 5 / TO REPAIR
005 121 1.0 / TO REPAIR
005 001 2.1 / TO REPAIR
005 120 10 / TO REPAIR
005 121 1.0 / TO REPAIR
005 002 2.1 / TO REPAIR
005 120 15 / TO REPAIR
005 121 1.0 / TO REPAIR
002 001 1 / RETURN TO USE
002 120 630 / RETURN TO USE
002 121 0.0 / RETURN TO USE
002 002 1 / RETURN TO USE
002 120 45 / RETURN TO USE
002 121 0.0 / RETURN TO USE
002 001 2 / RETURN TO USE
002 120 90 / RETURN TO USE
002 121 0.0 / RETURN TO USE
002 002 2 / RETURN TO USE
002 120 135 / RETURN TO USE
002 121 0.0 / RETURN TO USE
003 001 1 / PROPOSED MAINT.
003 120 / PROPOSED MAINT.
004 001 1.1 / POSTPONED
004 120 8 / POSTPONED
004 121 1.0 / POSTPONED
004 134 1.0 25000 1.0 / POSTPONED
004 002 1.1 / POSTPONED
004 120 1 / POSTPONED
004 121 1.0 / POSTPONED
004 134 1.0 25000 1.0 / POSTPONED
004 0 01 2 1
004 1 20 1
004 1 21 1.0
004 1 34 1.0 25000 1.0
004 0 02 2 1
004 1 34 1.0 25000 1.0
006 0 01 1 1 1 0 1 1
006 1 20 70
006 1 31 1.0 18000 1.0
006 0 02 1 1 1 0 1 1
006 1 20 5
006 1 31 1.0 86000 1.0
006 0 01 2 1 1 0 1 1
006 1 20 10
006 1 31 1.0 45000 1.0
006 0 02 2 1 1 0 1 1
006 1 20 15
006 1 31 1.0 86000 1.0
008 0 01 1 1 -1 0 0 1
008 1 20 56
008 1 21 1.0
008 0 02 1 1 -1 0 0 1
008 1 20 4
008 1 21 1.0
008 0 01 2 1 1 -1 0 0 1
008 1 20 9
008 1 21 1.0
008 0 02 2 1 -1 0 0 1
008 1 20 14
008 1 21 1.0
009 0 01 1 1 1 0 0 1
009 1 20 84
009 1 21 .1667
009 0 02 1 1 1 0 0 1
009 1 20 6
009 1 21 .1667
009 0 01 2 1 1 0 0 1
009 1 20 12
009 1 21 .1667
009 0 02 2 1 1 0 0 1
009 1 20 16
009 1 21 .1667
011 0 01 1 1 1 0 0 1
011 1 20 70
011 0 02 1 1 1 0 0 1
011 1 20 5
011 0 01 2 1 1 0 0 1
011 1 20 10
011 0 02 2 1 1 0 0 1
011 1 20 15
001 0 01 1 1
001 1 20 700
001 2 20 50
001 2 20 5 2.0 2.0 1.0 6.0 5.0

/ POSTPONED
/ POSTPONED
/ POSTPONED
/ POSTPONED
/ POSTPONED
/ POSTPONED
/ POSTPONED
/ MFG PATH
/ MFG PATH
/ MFG PATH
/ MFG PATH
/ MFG PATH
/ MFG PATH
/ MFG PATH
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ TO RENEW
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ SEPARATE SCRAP
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ GOOD MTRL.
/ TO INSPECT
/ TO INSPECT
/ TO INSPECT
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>001 1 21.10</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 43 20.0 15.0 1.0</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 0 02 1 1</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 20 50</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 21.10</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 43 20.0 15.0 1.0</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 0 01 2 1</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 20 100</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 21.10</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 43 20.0 15.0 1.0</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 0 02 2 1</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 20 150</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 21.10</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>001 1 43 20.0 15.0 1.0</td>
<td>TO INSPECT.</td>
</tr>
<tr>
<td>007 0 01 1 1 0 0 0 1</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 20 140</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 21.5</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 30 1.0 1.0 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 31 1.0 11785 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 32 1.0 785 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 37 1.0 157 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 38 1.0 314 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 40 1.0 314 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 41 1.0 314 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 42 1.0 942 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 43 1.0 942 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 44 1.0 157 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 0 02 1 1 0 0 0 1</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 20 10</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 21.5</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 30 1.0 1.0 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 31 1.0 10500 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 32 1.0 700 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 37 1.0 140 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 38 1.0 280 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 40 1.0 280 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 41 1.0 280 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 42 1.0 840 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 43 1.0 840 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 44 1.0 140 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 0 01 2 1 0 0 0 1</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 20 20</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 21.5</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 30 1.0 1.0 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 31 1.0 20700 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 32 1.0 1380 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 37 1.0 276 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 38 1.0 552 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 40 1.0 552 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 41 1.0 552 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 43 1.0 1656 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 43 1.0 1656 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 1 44 1.0 276 1.0</td>
<td>SORT MTRL.</td>
</tr>
<tr>
<td>007 0 02 2 1 0 0 0 1</td>
<td>SORT MTRL.</td>
</tr>
</tbody>
</table>
007 1 20 30  // SORT MTRL.
007 1 21 .5  // SORT MTRL.
007 1 30 1.0 1.0 1.0  // SORT MTRL.
007 1 31 1.0 1550 1.0  // SORT MTRL.
007 1 32 1.0 103 1.0  // SORT MTRL.
007 1 37 1.0 20.66 1.0  // SORT MTRL.
007 1 38 1.0 41.33 1.0  // SORT MTRL.
007 1 40 1.0 41.33 1.0  // SORT MTRL.
007 1 41 1.0 41.33 1.0  // SORT MTRL.
007 1 42 1.0 124 1.0  // SORT MTRL.
007 1 43 1.0 124 1.0  // SORT MTRL.
007 1 44 1.0 20.66 1.0  // SORT MTRL.
010 0 01 1.1 -1 0 0 1  // TO SCRAP
010 1 20 14  // TO SCRAP
010 1 21 1.0  // TO SCRAP
010 1 35 1.0 -10000 1.0  // TO SCRAP
010 0 02 1.1 -1 0 0 1  // TO SCRAP
010 1 20 1  // TO SCRAP
010 1 21 1.0  // TO SCRAP
010 1 35 1.0 -10000 1.0  // TO SCRAP
010 0 01 2.1 -1 0 0 1  // TO SCRAP
010 1 20 2  // TO SCRAP
010 1 21 1.0  // TO SCRAP
010 1 35 1.0 -10000 1.0  // TO SCRAP
010 0 02 2.1 -1 0 0 1  // TO SCRAP
010 1 20 1  // TO SCRAP
010 1 21 1.0  // TO SCRAP
010 1 35 1.0 -10000 1.0  // TO SCRAP
015 0 01 1  // RETURN
015 1 20  // RETURN
017 0 01 1.1  // ACCIDENT PATH
017 1 20 8  // ACCIDENT PATH
017 2 20 50  // ACCIDENT PATH
017 2 20 6 0.29 0.1 0.71  // ACCIDENT PATH
017 1 21 1.0  // ACCIDENT PATH
017 1 36 1.0 3000 1.0  // ACCIDENT PATH
017 0 02 1.1  // ACCIDENT PATH
017 1 20 1  // ACCIDENT PATH
017 1 21 1.0  // ACCIDENT PATH
017 1 36 1.0 3000 1.0  // ACCIDENT PATH
017 0 01 2.1  // ACCIDENT PATH
017 1 20 1  // ACCIDENT PATH
017 1 21 1.0  // ACCIDENT PATH
017 1 36 1.0 3000 1.0  // ACCIDENT PATH
017 0 02 2.1  // ACCIDENT PATH
017 1 36 1.0 3000 1.0  // ACCIDENT PATH
018 0 01 1  // TO REPAIR EVAL.
018 1 20 70  // TO REPAIR EVAL.
018 1 21 1.0  // TO REPAIR EVAL.
018 0 02 1  // TO REPAIR EVAL.
018 1 20 5  // TO REPAIR EVAL.
018 1 21 1.0  // TO REPAIR EVAL.
018 0 01 2  // TO REPAIR EVAL.
018 1 20 10  // TO REPAIR EVAL.
<table>
<thead>
<tr>
<th>Code</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>018 1 21 1.0</td>
<td>TO REPAIR EVAL.</td>
</tr>
<tr>
<td>018 0 02 2</td>
<td>TO REPAIR EVAL.</td>
</tr>
<tr>
<td>018 1 20 15</td>
<td>TO REPAIR EVAL.</td>
</tr>
<tr>
<td>018 1 21 1.0</td>
<td>TO REPAIR EVAL.</td>
</tr>
<tr>
<td>999 0</td>
<td>END OF DATA</td>
</tr>
</tbody>
</table>
APPENDIX H

Input Data Example #2
(For Comprehensive Track Maintenance Action - Diagram #2)

This Appendix contains file listings

IFILE - For Control Indices of Example Run #2
INODE - Maintenance Action Diagram Path Numbering Linkages
COST DATA - Path Starting Cost and Flow Data
**PROGRAM REVISION NUMBER:**

**FOR RUN EXAMPLE #2**

**SHAKER RESEARCH CORP.**

<table>
<thead>
<tr>
<th>1</th>
<th>1</th>
<th>ISTHD</th>
<th>FIRST NODE OF SCHEMATIC TO BEGIN COMPUTATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
<td>LSTHD</td>
<td>LAST NODE OF SCHEMATIC TO STOP</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>ISTPH</td>
<td>FIRST PATH &amp; PATH DATA FILE NUMBER ON DISK</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>LSTPH</td>
<td>LAST PATH &amp; PATH DATA FILE NUMBER ON DISK</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>ISTCP</td>
<td>FIRST COMPONENT NUMBER USED IN COMPUTATIONS</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>LSTCP</td>
<td>LAST COMPONENT NUMBER USED IN COMPUTATIONS</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>ISTCL</td>
<td>FIRST TRACK CLASS TO USE IN COMPUTATIONS</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>LSTCL</td>
<td>LAST TRACK CLASS TO USE IN COMPUTATIONS</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>ISTCT</td>
<td>COLUMN CONTAINING FIRST COST MINUS ONE</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>LSTCT</td>
<td>TOTAL COST COLUMN BLOCK SIZE MINUS ONE</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>IDM</td>
<td>DIMENSION OF IPATH ARRAY</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>IRDM</td>
<td>DIMENSION OF RPATH ARRAYS</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>HREFP</td>
<td>SCHEMATIC DIAGRAM IN USE REFERENCE PATH NUMBER</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>LSTFL</td>
<td>TOTAL FLOW COLUMN BLOCK SIZE - EQUIVALENT TO ISTCT</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>IOU7</td>
<td>PATH DATA WRITE TO DISK CONTROL INDICATOR</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>IBDM</td>
<td>NUMBER OF FLOW PLUS COST COLUMNS (BLOCK SIZE)</td>
</tr>
<tr>
<td>17</td>
<td>17</td>
<td>NCOSTS</td>
<td>MAXIMUM NUMBER OF DATA CODES</td>
</tr>
<tr>
<td>18</td>
<td>18</td>
<td>LCTCD</td>
<td>NUMERIC VALUE OF LAST USER DEFINED COST CODE</td>
</tr>
<tr>
<td>19</td>
<td>19</td>
<td>NPASS</td>
<td>STATE BLOCK REFERENCE ROW IN NORMAL</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>21</td>
<td>21</td>
<td>IUNIT</td>
<td>FORTRAN UNIT DEVICE NUMBER FOR OUTPUT PRINTING</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>IRUNIT</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>23</td>
<td>23</td>
<td>ITUNIT</td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>24</td>
<td>24</td>
<td>LNTHST</td>
<td>NUMBER OF COLUMNS IN THE STATE ARRAY</td>
</tr>
<tr>
<td>25</td>
<td>25</td>
<td>IPRNT</td>
<td>INDICATOR FOR NEED OF PATH FLOW/COST PRINTOUT</td>
</tr>
<tr>
<td>26</td>
<td>26</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>27</td>
<td>27</td>
<td>ISPRT</td>
<td>INDICATOR FOR NEED OF COST/STATE PRINTOUT</td>
</tr>
<tr>
<td>28</td>
<td>28</td>
<td>INFMTS</td>
<td>INDICATES COST/STATE PRINT OPTIONS</td>
</tr>
<tr>
<td>29</td>
<td>29</td>
<td>NDOUT</td>
<td>INDICATES NEED FOR FINAL COST/STATE PRINTOUT</td>
</tr>
<tr>
<td>30</td>
<td>30</td>
<td>IFMT</td>
<td>INDICATES FINAL PRINT OPTIONS</td>
</tr>
<tr>
<td>31</td>
<td>31</td>
<td>H1</td>
<td>STATE POPULATION REFERENCE ROW</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
<td>H2</td>
<td>STATE QUALITY REFERENCE ROW</td>
</tr>
<tr>
<td>33</td>
<td>33</td>
<td>N1</td>
<td>STATE POPULATION UPDATE ROW</td>
</tr>
<tr>
<td>34</td>
<td>34</td>
<td>N2</td>
<td>STATE QUALITY UPDATE ROW</td>
</tr>
<tr>
<td>35</td>
<td>35</td>
<td>ICOST</td>
<td>INDICATOR FOR NEED OF COST COMPUTATIONS</td>
</tr>
<tr>
<td>36</td>
<td>36</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>37</td>
<td>37</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>38</td>
<td>38</td>
<td>NBLOCK</td>
<td>NUMBER OF REPAIR BLOCKS IN SYSTEM</td>
</tr>
<tr>
<td>39</td>
<td>39</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>40</td>
<td>40</td>
<td>IDCDE</td>
<td>LAST COST CODE - EQUIVALENT TO LCTCD</td>
</tr>
<tr>
<td>41</td>
<td>41</td>
<td>IRDATA</td>
<td>RAW DATA PROCESSING INDICATOR</td>
</tr>
<tr>
<td>42</td>
<td>42</td>
<td>MOUT</td>
<td>MODEL RUN INDICATOR</td>
</tr>
<tr>
<td>43</td>
<td>43</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>44</td>
<td>44</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>45</td>
<td>45</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>46</td>
<td>46</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>47</td>
<td>47</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>48</td>
<td>48</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>49</td>
<td>49</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>50</td>
<td>50</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>51</td>
<td>51</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td></td>
<td>AVAILABLE</td>
</tr>
<tr>
<td>DATA FILE NUMBER</td>
<td>SCHEMATIC DIAGRAM NUMBER:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0002</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 10 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 12 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 14 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 16 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 18 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 20 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 22 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 24 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 26 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 28 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 30 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 32 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 34 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 36 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 38 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 40 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 42 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 44 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 46 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 48 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 50 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 52 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 54 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 56 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 58 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1 2 60 1 0 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DATA FILE</td>
<td>00 02</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>171 0 01 1</td>
<td>1 0 0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>171 1 20 2000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>171 1 21 1.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>201 0 01 1</td>
<td>1 1 0</td>
<td>1</td>
<td>NEW MFGS RAIL</td>
</tr>
<tr>
<td>201 1 20 2000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>201 1 31 473720</td>
<td>.123</td>
<td>1</td>
<td>RAIL MATERIAL RF.3</td>
</tr>
<tr>
<td>201 1 37 17.9</td>
<td>12.5</td>
<td>7</td>
<td>RAIL DELIVERY RF.4</td>
</tr>
<tr>
<td>261 0 01 1</td>
<td>1 0 0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>261 1 20 4000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>261 1 21 0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>261 1 37 1.0</td>
<td>70.71</td>
<td>1</td>
<td>EQUIP DELIVERY RF.4</td>
</tr>
<tr>
<td>261 1 38 4.1</td>
<td>17.5</td>
<td>5.7</td>
<td>T &amp; L RF.4</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.5</td>
<td>1</td>
<td>CRIB AND BROOM RF.1</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>16.27</td>
<td>1</td>
<td>CRIB AND BROOM RF.4</td>
</tr>
<tr>
<td>261 1 43 10.9</td>
<td>15.00</td>
<td>6</td>
<td>PULL SPIKES</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>1.67</td>
<td>6</td>
<td>PULL SPIKES RF.4</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>2</td>
<td>GAUGE THREADER</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>9.31</td>
<td>2</td>
<td>GAUGE THREADER RF.4</td>
</tr>
<tr>
<td>261 1 40 10.9</td>
<td>7.50</td>
<td>1</td>
<td>LIFT ANG &amp; PLT SF.6</td>
</tr>
<tr>
<td>261 1 40 10.9</td>
<td>7.50</td>
<td>1</td>
<td>PLUG &amp; CREOSOTE</td>
</tr>
<tr>
<td>261 1 31 24264</td>
<td>.021</td>
<td>4</td>
<td>PLUG &amp; CREOS 6/TIE RF.4</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>4</td>
<td>CRIBBING MACH</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>16.67</td>
<td>1</td>
<td>CRIBBING MACH</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>2</td>
<td>ADZER FACE TIES</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>16.27</td>
<td>1</td>
<td>ADZER FACE TIES</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>2</td>
<td>TIE PLATE LINER</td>
</tr>
<tr>
<td>261 1 31 6066</td>
<td>4.52</td>
<td>1</td>
<td>TIE PLATE LINER RF.3/4</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>13.72</td>
<td>1</td>
<td>TIE PLATE LINER</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>1</td>
<td>PLACE RAIL</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>9.31</td>
<td>2</td>
<td>PLACE RAIL</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>1</td>
<td>HEAT RAIL</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>7.25</td>
<td>1</td>
<td>HEAT RAIL RF.4</td>
</tr>
<tr>
<td>261 1 31 10.9</td>
<td>.20</td>
<td>1</td>
<td>HEAT RAIL</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>6</td>
<td>LINTER SPIKER</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>16.27</td>
<td>1</td>
<td>LINTER SPIKER</td>
</tr>
<tr>
<td>261 1 31 27297</td>
<td>.25</td>
<td>1</td>
<td>LINTER SPIKER 09/TIE RF.6</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>1</td>
<td>TAMPER</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>13.72</td>
<td>1</td>
<td>TAMPER RF.4</td>
</tr>
<tr>
<td>261 1 43 10.9</td>
<td>15.00</td>
<td>4</td>
<td>ANCHOR MACH</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>13.72</td>
<td>4</td>
<td>ANCHOR MACH RF.4</td>
</tr>
<tr>
<td>261 1 31 1625</td>
<td>3.0</td>
<td>1</td>
<td>ANCHOR MACH RF.5</td>
</tr>
<tr>
<td>261 1 40 10.9</td>
<td>7.50</td>
<td>1</td>
<td>GREASE MACH</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>5.0</td>
<td>1</td>
<td>GREASE MACH</td>
</tr>
<tr>
<td>261 1 41 10.9</td>
<td>10.00</td>
<td>1</td>
<td>RAIL QUEEN</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>7.25</td>
<td>1</td>
<td>RAIL QUEEN</td>
</tr>
<tr>
<td>261 1 41 10.9</td>
<td>10.00</td>
<td>4</td>
<td>POWER DRILL RF.4</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>16.27</td>
<td>1</td>
<td>POWER DRILL</td>
</tr>
<tr>
<td>261 1 43 10.9</td>
<td>15.00</td>
<td>2</td>
<td>WELD MACH</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>9.82</td>
<td>1</td>
<td>WELD MACHINE RF.4</td>
</tr>
<tr>
<td>261 1 42 10.9</td>
<td>12.50</td>
<td>2</td>
<td>HEAVY CRIB</td>
</tr>
<tr>
<td>261 1 32 10.9</td>
<td>16.27</td>
<td>2</td>
<td>HEAVY CRIB RF.4</td>
</tr>
<tr>
<td>261 1 31 2080</td>
<td>1</td>
<td>1</td>
<td>OTHER TRK MTL RF.4</td>
</tr>
<tr>
<td>261 1 45 10.9</td>
<td>20.00</td>
<td>2</td>
<td>SUPERVISORS RF.7</td>
</tr>
</tbody>
</table>
1 ASST SUPER RF.7
1 FORMAN RF.7
1 ASST FORMAN RF.7
1 LABORS RF.7
1 PATH INFO
1 TO RENEW RF.4
1 TO RENEW RF.4
1 TO RENEW RF.1
1 TO SCRAP INFSP
1 TO SCRAP INFSP
1 PATH INFO
1 TO SCRAP RF.4
1 TO SCRAP
2 NEW RAIL
2 NEW RAIL
2 NEW RAIL
2 NEW RAIL
2 RAIL DELIVERY RF.1
2 MFGS RAIL RF.3
2 NEW RAIL
2 NEW RAIL
2 NEW RAIL
2 EQUIPMENT DELIVERY RF.4
2 T & L RF.4
2 PULL SPIKES
2 PULL SPIKES RF.4
2 REMOVE ANCHORS
2 LIFT OUT OLD RAIL
2 LIFT OUT OLD RAIL RF.4
2 LIFT TIE PLATES
2 DRIVE IN BROKEN SPIKES
2 INSERT WOODEN PLUGS
2 INSERT WOODEN PLUGS RF.4
2 FACE OFF TIE
2 FACE OFF TIE
2 BROOM OFF TIES RF.1
2 BROOM OFF TIES RF.4
2 NEW SPIKES/ANCHORS/PLATES RF.6
2 NEW SPIKES/ANCHORS/PLATES
2 NEW ANCHORS RF.5
2 NEW PLATES RF.4
2 BEGIN RAIL SET IN
2 BEGIN RAIL SET IN
2 CONTINUE SET-IN
2 CONTINUE SET-IN
2 GAUGE LINER
2 GAUGE LINER
2 ANCHOR RAILS
2 ANCHOR RAILS RF.4
2 END MATCHING PROCESS
2 END MATCHING PROCESS
2 FOLLOW UP SPIKING
262 1 32 10.9 1.67 4  
262 1 43 10.9 15.00 2  
262 1 32 10.9 9.02 1  
262 1 31 2060 1 1  
262 1 45 10.9 20.00 2  
262 1 45 10.9 20.00 1  
262 1 44 10.9 17.50 1  
262 1 44 10.9 17.50 2  
262 1 43 10.9 7.50 10  
402 0 01 1 1 0 0 0 2  
402 1 20 900  
402 1 21 1.0  
402 1 37 17.9 12.5 7  
372 0 01 1 1 0 0 0 2  
372 1 20 1100  
372 1 21 0.0909  
432 0 01 1 1 0 0 0 2  
432 1 20 100  
432 1 21 1.0  
432 1 37 17.9 12.5 7  
173 0 01 1 1 0 0 0 3  
173 1 20 1000  
173 1 21 1.0  
203 0 01 1 1 1 0 1 3  
203 1 20 1000  
203 1 37 17.9 12.5 7  
203 1 31 478720 .125 1  
263 0 01 1 1 0 0 0 3  
263 1 20 2000  
263 1 21 0.5  
263 1 37 1.0 70.71 1  
263 1 39 4.1 17.5 5.7  
263 1 43 10.9 15.00 6  
263 1 32 10.9 1.67 6  
263 1 42 10.9 12.50 2  
263 1 32 10.9 9.31 2  
263 1 42 10.9 12.5 4  
263 1 42 10.9 12.50 6  
263 1 32 10.9 9.31 2  
263 1 40 10.9 7.50 4  
263 1 32 10.9 9.31 2  
263 1 42 10.9 12.50 6  
263 1 32 10.9 16.27 1  
263 1 31 27297 .25 1  
263 1 43 10.9 15.00 4  
263 1 43 10.9 13.72 4  
263 1 31 1625 3.0 1  
263 1 42 10.9 12.50 2  
263 1 32 10.9 9.31 2  
263 1 43 10.9 15.00 2  
263 1 32 10.9 9.02 1  
263 1 42 10.9 12.50 4  
263 1 32 10.9 1.67 4  
263 1 31 2060 1 1  

/ 2 FOLLOW UP SPIKING  
/ 2 THERMITE WELD  
/ 2 THERMITE WELD  
/ 2 OTHR TRK MATL RF.4  
/ 2 SUPERVISORS RF.7  
/ 2 ASST SUPER RF.7  
/ 2 FORMAN RF.7  
/ 2 ASST FORMAN RF.7  
/ 2 LABORS RF.7  
/ 2 PATH INFO  
/ 2 TO RENEW RF.4  
/ 2 TO RENEW RF.4  
/ 2 TO RENEW RF.1  
/ 2 TO SCRAP  
/ 2 TO SCRAP  
/ 2 PATH INFO  
/ 2 TO SCRAP RF.4  
/ 2 SCRAP  
/ 2 TO SCRAP  
/ 3 PATH INFO  
/ 3 DELAYS  
/ 3 DELAYS  
/ 3 PATH INFO  
/ 3 RAIL TO SITE  
/ 3 RAIL TO SITE  
/ 3 RAIL TO SITE RF.3  
/ 3 PATH INFO  
/ 3 TRAVEL TO SITE  
/ 3 TRAVEL TO SITE  
/ 3 TRAVEL TO SITE RF.4  
/ 3 TRAVEL TO SITE RF.4  
/ 3 PULL SPIKES/ANCHOR RF.1  
/ 3 PULL SPIKES/ANCHOR RF.4  
/ 3 LIFT OUT OLD RAIL  
/ 3 LIFT OUT OLD RAIL RF.4  
/ 3 LAYOUT SPIKES/ANCHORS  
/ 3 SET RAIL  
/ 3 SET RAIL RF.4  
/ 3 CONTINUE SET-IN  
/ 3 CONTINUE SET-IN  
/ 3 SPIKE DOWN RAIL  
/ 3 SPIKE DOWN RAIL  
/ 3 SPIKE DOWN RAIL RF.6  
/ 3 ANCHOR RAIL  
/ 3 ANCHOR RAIL RF.4  
/ 3 ANCHOR RAIL RF.5  
/ 3 MATCH END PROCESS  
/ 3 MATCH END PROCESS  
/ 3 THERMITE WELD  
/ 3 THERMITE WELD  
/ 3 FOLLOW UP SPIKING  
/ 3 FOLLOW UP SPIKING  
/ 3 OTHR TRK MATL
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>263</td>
<td>SUPERVISORS</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>45</td>
<td>10.9 20.00 2</td>
</tr>
<tr>
<td>263</td>
<td>ASST SUPER</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>44</td>
<td>10.9 17.50 1</td>
</tr>
<tr>
<td>263</td>
<td>FORMAN</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>44</td>
<td>10.9 17.50 2</td>
</tr>
<tr>
<td>263</td>
<td>ASST FORMAN</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>43</td>
<td>10.9 7.50 9</td>
</tr>
<tr>
<td>403</td>
<td>LABORS</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>1 1 0 0 0 3</td>
</tr>
<tr>
<td>403</td>
<td>RENEW</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>900</td>
</tr>
<tr>
<td>403</td>
<td>TO SCRAP PATH</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>37</td>
<td>17.9 12.5 7</td>
</tr>
<tr>
<td>373</td>
<td>PATH SCRAP</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>1 1 0 0 0 3</td>
</tr>
<tr>
<td>373</td>
<td>RENEW RF.4</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>1100</td>
</tr>
<tr>
<td>373</td>
<td>PATH SCRAP</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>21</td>
<td>0.0909</td>
</tr>
<tr>
<td>373</td>
<td>RENEW RF.1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>478720 .125 1</td>
</tr>
<tr>
<td>264</td>
<td>RAIL TO SITE</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>2000</td>
</tr>
<tr>
<td>264</td>
<td>JOINTED RAIL</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>21</td>
<td>0.5</td>
</tr>
<tr>
<td>264</td>
<td>PULL ONE SECTION RF.6</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>135 7.50 10</td>
</tr>
<tr>
<td>264</td>
<td>JOINTED MILES</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>32</td>
<td>135 16.27 1</td>
</tr>
<tr>
<td>264</td>
<td>INSERT NEW RAIL</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>42</td>
<td>135 12.50 10</td>
</tr>
<tr>
<td>264</td>
<td>INSERT NEW RAIL</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>32</td>
<td>135 16.27 1</td>
</tr>
<tr>
<td>264</td>
<td>SPIKES RF.6</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>9260 .25 1</td>
</tr>
<tr>
<td>264</td>
<td>ANCHOR</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>400 3.00 1</td>
</tr>
<tr>
<td>264</td>
<td>BARS</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>541 17.00 1</td>
</tr>
<tr>
<td>264</td>
<td>PLATES</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>3033 4.52 1</td>
</tr>
<tr>
<td>264</td>
<td>REPLACE SIGNAL RF.6</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>41</td>
<td>45 10.00 1</td>
</tr>
<tr>
<td>264</td>
<td>OTHER TRACK MATL</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>31</td>
<td>3600 1 1</td>
</tr>
<tr>
<td>264</td>
<td>FORMAN RF.6</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>44</td>
<td>135 17.50 1</td>
</tr>
<tr>
<td>264</td>
<td>TO RENEW</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>001</td>
<td>1 1 0 0 0 4</td>
</tr>
<tr>
<td>404</td>
<td>TO RENEW RF.4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>900</td>
</tr>
<tr>
<td>404</td>
<td>TO RENEW RF.1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>21</td>
<td>1.0</td>
</tr>
<tr>
<td>404</td>
<td>TO RENEW RF.4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>37</td>
<td>17.9 12.5 7</td>
</tr>
<tr>
<td>374</td>
<td>TO SCRAP PATH</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>1 1 0 0 0 4</td>
</tr>
<tr>
<td>374</td>
<td>TO SCRAP</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>1100</td>
</tr>
<tr>
<td>374</td>
<td>TO SCRAP PATH</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>21</td>
<td>0.0909</td>
</tr>
<tr>
<td>434</td>
<td>TO SCRAP</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>1 1 0 0 0 4</td>
</tr>
<tr>
<td>434</td>
<td>TO SCRAPRF.4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>434</td>
<td>TO SCRAP</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>37</td>
<td>4.48 12.50 -7</td>
</tr>
<tr>
<td>175</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>1 1 0 0 0 5</td>
</tr>
<tr>
<td>175</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>1000</td>
</tr>
<tr>
<td>175</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>21</td>
<td>1.0</td>
</tr>
<tr>
<td>205</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>1 1 0 0 0 5</td>
</tr>
<tr>
<td>205</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>0</td>
</tr>
<tr>
<td>265</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>01</td>
<td>1 1 0 0 0 5</td>
</tr>
<tr>
<td>265</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>20</td>
<td>1000</td>
</tr>
<tr>
<td>265</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>135 7.50 5</td>
</tr>
<tr>
<td>265</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>32</td>
<td>135 16.27 1</td>
</tr>
<tr>
<td>265</td>
<td>TO SCRAP</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>42</td>
<td>135 12.50 5</td>
</tr>
<tr>
<td>Line</td>
<td>Value 1</td>
<td>Value 2</td>
</tr>
<tr>
<td>------</td>
<td>---------</td>
<td>---------</td>
</tr>
<tr>
<td>280</td>
<td>32</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>43</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>32</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>42</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>32</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>42</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>32</td>
<td>11.1</td>
</tr>
<tr>
<td>280</td>
<td>0.05</td>
<td>11</td>
</tr>
<tr>
<td>280</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>410</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>410</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>410</td>
<td>0</td>
<td>05</td>
</tr>
<tr>
<td>410</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>380</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>380</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>380</td>
<td>1</td>
<td>21</td>
</tr>
<tr>
<td>380</td>
<td>0</td>
<td>05</td>
</tr>
<tr>
<td>380</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>380</td>
<td>1</td>
<td>21</td>
</tr>
<tr>
<td>440</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>440</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>410</td>
<td>0</td>
<td>05</td>
</tr>
<tr>
<td>410</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>440</td>
<td>0</td>
<td>05</td>
</tr>
<tr>
<td>440</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>181</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>181</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>181</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>211</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>211</td>
<td>1</td>
<td>37</td>
</tr>
<tr>
<td>211</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td>281</td>
<td>0</td>
<td>03</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>20</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>37</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>36</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>40</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>42</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>43</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>42</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>42</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>41</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>31</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>43</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>43</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>32</td>
</tr>
<tr>
<td>281</td>
<td>1</td>
<td>31</td>
</tr>
</tbody>
</table>
281 1 44 11.1 17.50 1
281 1 44 11.1 17.50 2
411 0 03 1 1 0 0 0 11
411 1 20 1000
381 0 03 1 1 0 0 0 11
381 1 21 2000
381 1 21 0.5
441 0 03 1 1 1 0 0 11
441 1 20 1000
441 1 21 1.0
182 0 01 1 1 0 0 0 12
182 1 20 1000.0
212 0 01 1 1 1 0 1 12
212 1 20 0
282 0 01 1 1 0 0 0 12
282 1 20 1000
282 1 36
412 0 01 1 1 0 0 0 12
412 1 20 0
382 0 01 1 1 0 1 0 12
382 1 20 1000.0
442 0 01 1 1 1 0 0 12
442 1 20 0
183 0 01 1 1 0 0 0 13
183 1 20 1000
183 1 21 1.0
213 0 01 1 1 1 0 0 13
213 1 20 0
283 0 01 1 1 0 0 0 13
283 1 20 1000
283 1 20 1000.0
283 1 32 8.0 3.92 1
283 1 31 100 1.1
283 1 42 8.0 12.50 4
413 0 01 1 1 0 0 0 13
413 1 20 0
383 0 01 1 1 0 1 0 13
383 1 20 1000
383 1 21 1.0
443 0 01 1 1 1 0 0 13
443 1 20 0
184 0 01 1 1 0 0 0 14
184 1 20 1000
214 0 01 1 1 1 0 0 14
214 1 20 0
284 0 01 1 1 0 0 0 14
284 1 20 1000
284 1 36 100 1.1
414 0 01 1 1 0 0 0 14
414 1 20 0
384 0 01 1 1 0 1 0 14
384 1 20 1000
384 1 21 1.0
444 0 01 1 1 1 0 0 14
444 1 20 0

/11 FOREMAN
/11 ASST. FOREMAN
/11 PATH INFO
/11 TO RENEW
/11 PATH INFO
/11 TO SCRAP PATH
/11 TO SCRAP PATH
/11 PATH INFO
/11 TO SCRAP
/11 TO SCRAP
/12 PATH INFO
/12 JOINT REPAIRS
/12 PATH INFO
/12 MFG SOURCE
/12 PATH INFO
/12 REPAIRS @ 1/1 MILE
/12 REPAIRS
/12 PATH INFO
/12 TO RENEW
/12 PATH INFO
/12 TO SCRAP PATH
/12 PATH INFO
/12 TO SCRAP
/13 PATH INFO
/13 TURN OUT REPAIRS
/13 TURN OUT REPAIRS
/13 PATH INFO
/13 MFG SOURCE
/13 PATH INFO
/13 REPAIRS @ 1/1 MILE
/13 REPAIRS
/13 REPAIRS
/13 REPAIRS
/13 PATH INFO
/13 TO RENEW
/13 PATH INFO
/13 TO SCRAP PATH
/13 TO SCRAP PATH
/13 PATH INFO
/13 TO SCRAP
/14 PATH INFO
/14 MILES REBUILT
/14 PATH INFO
/14 MFG SOURCE
/14 PATH INFO
/14 REPAIRS
/14 REBUILT RF .5
/14 PATH INFO
/14 TO RENEW
/14 PATH INFO
/14 TO SCRAP PATH
/14 TO SCRAP PATH
/14 PATH INFO
/14 TO SCRAP
185 0 02 1 1 0 0 0 15
185 1 20 3000
215 0 02 1 1 1 0 0 15
215 1 20 0 0
285 0 02 1 1 0 0 0 15
285 1 20 3000
285 1 21 1.0
285 1 36 5 20 1 1
415 0 02 1 1 0 0 0 15
415 1 20 0
385 0 02 1 1 0 1 0 15
385 1 20 3000
385 1 21 1.0
445 0 01 1 1 1 0 0 15
445 1 20 0
495 0 02 1 1 0 0 0 15
485 1 20 3000
485 1 21 0.0
186 0 05 1 1 0 0 0 15
186 1 20 1000
186 1 21 1.0
216 0 05 1 1 1 0 1 15
216 1 20 1000
286 0 05 1 1 0 0 0 15
286 1 20 2000
286 1 35 500 1 1
416 0 05 1 1 0 0 0 16
416 1 20 0
386 0 05 1 1 0 0 0 16
386 1 20 2000
386 1 21 0.5
446 0 05 1 1 1 0 0 16
446 1 20 1000
187 0 01 1 1 0 0 0 17
187 1 20 1000
217 0 01 1 1 1 0 0 17
217 1 20 0
287 0 01 1 1 0 0 0 17
287 1 20 1000
287 1 35 175 1 1
417 0 01 1 1 0 0 0 17
417 1 20 0
387 0 01 1 1 0 1 0 17
387 1 20 1000
387 1 21 1.0
447 0 01 1 1 1 0 0 17
447 1 20 0
188 0 01 1 1 0 0 0 18
188 1 20 0
218 0 01 1 1 0 1 1 18
218 1 20 0
288 0 01 1 1 0 0 0 18
288 1 20 0
418 0 01 1 1 0 0 0 18

✓ 15 PATH INFO
✓ 15 RAIL GRIND
✓ 15 PATH INFO
✓ 15 NULL
✓ 15 PATH INFO
✓ 15 REPAIRS
✓ 15 REPAIRS
✓ 15 REPAIRS RF. 6
✓ 15 PATH INFO
✓ 15 TO RENEW
✓ 15 PATH INFO
✓ 15 TO SCRAP PATH
✓ 15 TO SCRAP PATH
✓ 15 PATH INFO
✓ 15 TO SCRAP
✓ 15 RETURN
✓ 15 RETURN
✓ 15 RETURN
✓ 16 PATH INFO
✓ 16 MILES CLEANED
✓ 16 MILES CLEANED
✓ 16 PATH INFO
✓ 16 MFG SOURCE
✓ 16 PATH INFO
✓ 16 REPAIRS
✓ 16 REPAIRS
✓ 16 REPAIRS
✓ 16 PATH INFO
✓ 16 TO RENEW
✓ 16 PATH INFO
✓ 16 TO SCRAP PATH
✓ 16 TO SCRAP PATH
✓ 16 PATH INFO
✓ 16 PATH INFO
✓ 16 TO SCRAP
✓ 16 TO SCRAP
✓ 17 PATH INFO
✓ 17 BRUSH & WEED MILES
✓ 17 PATH INFO
✓ 17 MFG SOURCE
✓ 17 REPAIR ITEMS
✓ 17 REPAIR ITEMS RF. 5
✓ 17 PATH INFO
✓ 17 TO RENEW
✓ 17 TO SCRAP PATH
✓ 17 TO SCRAP PATH
✓ 17 TO SCRAP PATH
✓ 17 PATH INFO
✓ 17 TO SCRAP
✓ 18 PATH INFO
✓ 18 DITCHING
✓ 18 PATH INFO
✓ 18 MFG SOURCE
✓ 18 PATH INFO
✓ 18 PATH INFO
✓ 18 PATH INFO

H-16
| 418 1 20 0 | /18 TO RENEW |
| 448 0 01 1 1 1 0 0 18 | /18 PATH INFO |
| 448 1 20 0 | /18 TO SCRAP |
| 189 0 01 1 1 0 0 0 19 | /19 PATH INFO |
| 189 1 20 | /19 SHOW & ICE REMOVAL |
| 219 0 01 1 1 1 0 1 19 | /19 PATH INFO |
| 219 1 20 0 | /19 MFG SOURCE |
| 289 0 01 1 1 0 0 0 19 | /19 PATH INFO |
| 289 1 20 0 | /19 REPAIR ITEMS |
| 419 0 01 1 1 0 0 0 19 | /19 PATH INFO |
| 419 1 20 0 | /19 TO RENEW |
| 449 0 01 1 1 1 0 0 19 | /19 PATH INFO |
| 449 1 20 0 | /19 TO SCRAP |
| 190 0 01 1 1 0 0 0 20 | /20 PATH INFO |
| 190 1 20 0 | /20 SLIDES & WASHOUTS |
| 220 0 01 1 1 1 0 1 20 | /20 PATH INFO |
| 220 1 20 0 | /20 BALLAST/TIES/RAIL |
| 290 0 01 1 1 0 0 0 20 | /20 PATH INFO |
| 290 1 20 0 | /20 REPAIR ITEMS |
| 420 0 01 1 1 0 0 0 20 | /20 PATH INFO |
| 420 1 20 0 | /20 TO RENEW |
| 450 0 01 1 1 1 0 0 20 | /20 PATH INFO |
| 450 1 20 0 | /20 TO SCRAP |
| 002 0 01 1 1 0 0 0 0 1 | / DUMPING PATH |
| 002 1 21 0.00 | / DUMPING PATH |
| 001 0 01 1 1 0 0 0 0 0 | / TRACK IN USE |
| 001 1 20 100000 | / TRACK IN USE RF.2 |
| 001 1 21 .15 | / TRACK IN USE |
| 001 0 02 1 1 0 0 0 0 0 | / TRACK IN USE |
| 001 1 20 100000 | / TRACK IN USE RF.2 |
| 001 1 21 .04 | / TRACK IN USE |
| 001 0 03 1 1 0 0 0 0 0 | / TRACK IN USE |
| 001 1 20 200000 | / TRACK IN USE RF.2 |
| 001 1 21 .05 | / TRACK IN USE |
| 001 0 04 1 1 0 0 0 0 0 | / TRACK IN USE |
| 001 1 20 0 | / TRACK IN USE |
| 001 0 05 1 1 0 0 0 0 0 | / TRACK IN USE |
| 001 1 20 200000 | / TRACK IN USE RF.2 |
| 001 1 21 .05 | / TRACK IN USE |
| 016 0 01 1 1 0 0 0 0 | / PATH INFO |
| 016 1 20 200000 | / FED CAR INSPT |
| 016 1 21 0.15 | / FED CAR INSPT |
| 017 0 01 1 1 0 0 0 0 | / PATH INFO |
| 017 1 20 1000 | / FED CAR RESULTS |
| 017 1 21 1.0 | / FED CAR RESULTS |
| 018 0 01 1 1 0 0 0 0 | / PATH INFO |
| 018 1 20 200000 | / SPERRY RAIL CAR |
| 018 1 21 0.15 | / SPERRY RAIL CAR |
| 018 1 36 25 1 1 | / SPERRY RAIL CAR |
| 019 0 01 1 1 0 0 0 0 | / PATH INFO |
| 019 1 20 1000 | / SPERRY RAIL CAR |
| 019 1 21 1.0 | / SPERRY RAIL CAR |
| 020 0 01 1 1 0 0 0 0 | / PATH INFO |
| 020 1 20 200000 | / SOUND RAIL CAR |

H-17
020 1 21 0.15
020 1 36 25 1 1
021 0 01 1 1 0 0 0 0
021 1 20 1000
021 1 21 1.0
022 0 01 1 1 0 0 0 0
022 1 20 200000
022 1 21 0.15
022 1 36 25 1 1
023 0 01 1 1 0 0 0 0
023 1 20 1000
023 1 21 1.0
024 0 01 1 1 0 0 0 0
024 1 20 200000
024 1 21 0.15
024 1 36 25 1 1
025 0 01 1 1 0 0 0 0
025 1 20 1000
025 1 21 1.0
010 0 01 1 1 0 0 0 0
010 1 20 10000
010 1 21 0.15
010 1 41 .1 10.00 2
011 0 01 1 1 0 0 0 0
011 1 20 1000
011 1 21 1.0
012 0 01 1 1 0 0 0 0
012 1 20 10000
012 1 21 0.15
012 1 41 .1 10.00 2
013 0 01 1 1 0 0 0 0
013 1 20 1000
013 1 21 1.0
014 0 01 1 1 0 0 0 0
014 1 20 26667
014 1 21 0.15
014 0 02 1 1 0 0 0 0
014 1 20 100000.00
014 1 21 0.04
014 0 03 1 1 0 0 0 0
014 1 20 80000
014 1 21 0.05
014 0 04 1 1 0 0 0 0
014 1 20 0
014 1 21 0.15
014 0 05 1 1 0 0 0 0
014 1 20 80000
014 1 21 0.05
026 0 01 1 1 0 0 0 0
026 1 20 1000
026 1 21 0.15
026 1 36 25000 1 1
027 0 01 1 1 0 0 0 0
027 1 20 150

/ SOUND RAIL CAR
/ SOUND RAIL CAR
/ PATH INFO
/ SOUND RAIL CAR
/ SOUND RAIL CAR
/ PATH INFO
/ TRACK GEOM CAR
/ TRACK GEOM CAR
/ PATH INFO
/ TRACK GEOM CAR
/ TRACK GEOM CAR
/ PATH INFO
/ RAIL WEAR CAR
/ RAIL WEAR CAR
/ PATH INFO
/ RAIL WEAR CAR
/ RAIL WEAR CAR
/ PATH INFO
/ FEDERAL VIS INSPI
/ FEDERAL VIS INSPI
/ PATH INFO
/ FEDERAL VIS INSPI
/ FEDERAL VIS INSPI
/ PATH INFO
/ STATE VIS INSPI
/ STATE VIS INSPI
/ PATH INFO
/ STATE VIS INSPI
/ STATE VIS INSPI
/ PATH INFO
/ SCHED TRACK REVIEW
/ SCHED TRACK REVIEW
/ PATH INFO
/ SCHED TRACK REVIEW
/ SCHED TRACK REVIEW
/ PATH INFO
/ SCHED TRACK REVIEW
/ SCHED TRACK REVIEW
/ PATH INFO
/ SCHED TRACK REVIEW
/ SCHED TRACK REVIEW
/ PATH INFO
/ ACCIDENTS
/ ACCIDENTS
/ ACCIDENTS RF . 8
/ PATH INFO
/ ACC RESULTS
027 1 21 1.0 / ACC RESULTS
030 0 01 1 1 0 0 0 0 / PATH INFO
030 1 20 200000 / JOINT INSP CAR
030 1 21 0.15 / JOINT INSP CAR
030 1 36 10 1 1 / JOINT INSP CAR
031 0 01 1 1 0 0 0 0 / PATH INFO
031 1 20 1000 / JOINT RESULTS
031 1 21 1.0 / PATH INFO
015 0 01 1 1 0 0 0 0 / INSPECT RESULTS
015 1 20 9150.0 / INSPECT RESULTS
015 1 21 1.0 / INSPECT RESULTS
015 0 02 1 1 0 0 0 0 / PATH INFO
015 1 20 4.0D+03 / INSPECT RESULTS
015 1 21 1.0 / INSPECT RESULTS
015 0 03 1 1 0 0 0 0 / PATH INFO
015 1 20 4000.0 / INSPECT RESULTS
015 1 21 1.0 / INSPECT RESULTS
015 0 04 1 1 0 0 0 0 / PATH INFO
015 1 20 0. / INSPECT RESULTS
015 0 05 1 1 0 0 0 0 / PATH INFO
015 1 20 4000.0 / INSPECT RESULTS
015 1 21 1.0 / PATH INFO
061 0 01 1 1 0 0 0 0 / TO REPAIR
061 1 20 10000.0 / TO REPAIR
061 1 21 1.0 / TO REPAIR
061 0 02 1 1 0 0 0 0 / PATH INFO
061 1 20 3000.0 / TO REPAIR
061 1 21 1.0 / TO REPAIR
061 0 03 1 1 0 0 0 0 / PATH INFO
061 1 20 2000.0 / TO REPAIR
061 1 21 1.0 / TO REPAIR
061 0 04 1 1 0 0 0 0 / PATH INFO
061 1 20 0. / TO REPAIR
061 0 05 1 1 0 0 0 0 / PATH INFO
061 1 20 4000.0 / TO REPAIR
061 1 21 1.0 / TO REPAIR
062 0 01 1 1 0 0 0 0 / PATH INFO
062 1 20 8000.0 / AFTER MAINT 1
062 0 02 1 1 0 0 0 0 / PATH INFO
062 1 20 3000.0 / AFTER MAINT 1
062 1 21 1.0 / AFTER MAINT 1
062 0 03 1 1 0 0 0 0 / PATH INFO
062 1 20 2000.0 / AFTER MAINT 1
062 1 21 1.0 / AFTER MAINT 1
062 0 04 1 1 0 0 0 0 / PATH INFO
062 1 20 0. / AFTER MAINT 1
062 0 05 1 1 0 0 0 0 / AFTER MAINT 1
062 1 20 4000 / AFTER MAINT 1
062 1 21 1.0 / AFTER MAINT 1
121 0 01 1 1 0 0 0 0 / PATH INFO
121 1 20 2000 / TO MAINT #1
121 1 21 1.0 / TO MAINT #1
063 0 01 1 1 0 0 0 0 / PATH INFO
063 1 20 7000 / AFTER MAINT 2
076 0 05 1 1 0  / PATH INFO
076 1 20 1000  / AFTER MAINT 15
135 0 02 1 1 0  / PATH INFO
135 1 20 3000  / TO MAINT #15
077 0 01 1 1 0  / PATH INFO
077 1 20 1000  / AFTER MAINT 16
077 0 02 1 1 0  / PATH INFO
077 1 20 0  / AFTER MAINT 16
077 0 03 1 1 0  / PATH INFO
077 1 20 0  / AFTER MAINT 16
077 0 04 1 1 0  / PATH INFO
077 1 20 0  / AFTER MAINT 16
077 0 05 1 1 0  / PATH INFO
077 1 20 0  / AFTER MAINT 16
136 0 05 1 1 0  / PATH INFO
136 1 20 1000  / TO MAINT #16
079 0 01 1 1 0  / PATH INFO
079 1 20 0  / AFTER MAINT 17
079 0 02 1 1 0  / PATH INFO
079 1 20 0  / AFTER MAINT 17
079 0 03 1 1 0  / PATH INFO
079 1 20 0  / AFTER MAINT 17
079 0 05 1 1 0  / PATH INFO
079 1 20 0  / AFTER MAINT 17
137 0 01 1 1 0  / PATH INFO
137 1 20 1000  / TO MAINT #17
300 0 01 1 1 0  / NOT REPAIRED
300 1 20 1150.0  / NOT REPAIRED
300 1 21 1.0  / NOT REPAIRED
300 0 02 1 1 0  / NOT REPAIRED
300 1 20 3000.0  / NOT REPAIRED
300 1 21 1.0  / NOT REPAIRED
300 0 03 1 1 0  / NOT REPAIRED
300 1 20 2000.0  / NOT REPAIRED
300 1 21 1.0  / NOT REPAIRED
302 0 01 1 1 0  / PATH INFO
302 1 20 2000  / SLOW ORDER
302 0 02 1 1 0  / PATH INFO
302 1 20 1000  / SLOW ORDER
302 0 03 1 1 0  / PATH INFO
302 1 20 2000  / SLOW ORDER
302 0 04 1 1 0  / PATH INFO
302 1 20 0  / SLOW ORDER
302 0 05 1 1 0  / PATH INFO
302 1 20 0  / SLOW ORDER
306 0 01 1 1 0  / PATH INFO
306 1 20 0  / IDLE TRACK
308 0 01 1 1 0  / PATH INFO
308 1 20 0  / CLOSE TRACK
312 0 01 1 1 0  / PATH INFO
312 1 20 3750  / NOTED BAD RF.4
313 0 01 1 1 0  / PATH INFO
APPENDIX I

Output Costs From Program Run Example #1
WEIBULL DISTRIBUTION PARAMETERS

<table>
<thead>
<tr>
<th>CLASS</th>
<th>COMPONENT</th>
<th>AVG. AGE</th>
<th>BETA</th>
<th>C-LIFE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
</tbody>
</table>

FUNCTIONS AND SHAPE PARAMETERS ACTIVE

<table>
<thead>
<tr>
<th>PATH</th>
<th>FUNC.</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>X0</th>
<th>Y0</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>7</td>
<td>2.000</td>
<td>3.900</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>1</td>
<td>5*</td>
<td>2.000</td>
<td>2.000</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>6</td>
<td>2.900</td>
<td>1.000</td>
<td>7.100</td>
<td>0.000</td>
<td>0.000</td>
</tr>
</tbody>
</table>

*NOTE: This function was also deleted and the full simulation rerun. See Section 6 for comparison discussion with and without the inspection modifying function in operation.
## TRACK MAINTENANCE COST

**RUN NUMBER:** BASE MOD 2  
**DATE:** THU, AUG 14 1980  
**PLACE:** SHAKER RESEARCH CORP.

**SIMULATION TIME:** 0.00

### POPULATION IN USE

<table>
<thead>
<tr>
<th>Node</th>
<th>Path</th>
<th>Class</th>
<th>Component</th>
<th>Quality</th>
<th>Path Flows MILES/YEAR</th>
<th>Maintenance DOLLARS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.100000</td>
<td>7.0000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.100000</td>
<td>5.0000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.100000</td>
<td>100.00</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>630.00</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>90.00</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>10.00</td>
</tr>
<tr>
<td>17</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>135.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>8.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>7.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>6.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>5.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>4.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>3.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>2.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
<tr>
<td>18</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1.00</td>
</tr>
</tbody>
</table>

**TOTAL $ 5312979**
### TRACK MAINTENANCE COST

**RUN NUMBER**: BASE MOD 2  
**DATE**: THU, AUG 14 1980  
**PLACE**: SHAKER RESEARCH CORP.

**SIMULATION TIME**: 0.50

<table>
<thead>
<tr>
<th>POPULATION IN USE</th>
<th>CLASS</th>
<th>COMPONENT</th>
<th>QUALITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>700.00</td>
<td>1</td>
<td>1</td>
<td>0.07587</td>
</tr>
<tr>
<td>50.00</td>
<td>1</td>
<td>2</td>
<td>0.07369</td>
</tr>
<tr>
<td>100.00</td>
<td>2</td>
<td>1</td>
<td>0.07369</td>
</tr>
<tr>
<td>150.28</td>
<td>2</td>
<td>2</td>
<td>0.07506</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NODE PATH</th>
<th>CLASS</th>
<th>COMPONENT</th>
<th>QUALITY</th>
<th>PATH FLOWS</th>
<th>MAINTENANCE MILES/YEAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.07587</td>
<td>711.66</td>
<td>213496</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0.07369</td>
<td>50.00</td>
<td>15000</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.07369</td>
<td>150.00</td>
<td>45000</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.07506</td>
<td>659.66</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1.00000</td>
<td>54.00</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>2</td>
<td>0.00000</td>
<td>46.32</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.00000</td>
<td>3.68</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1.00000</td>
<td>92.63</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>7.27</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>2</td>
<td>0.00000</td>
<td>139.00</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>1.00000</td>
<td>11.28</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>16935</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1.00000</td>
<td>5.65</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>0.74</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0.74</td>
<td>2210</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>59.64</td>
<td>4422</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>8.11</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>2</td>
<td>0.00000</td>
<td>11.28</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>0.00000</td>
<td>46.39</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>331408</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>13.26</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>0.00000</td>
<td>3.68</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
<td>0.074</td>
<td>18423</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>2</td>
<td>7.37</td>
<td>18423</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>11.28</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>2</td>
<td>0.074</td>
<td>834913</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>1</td>
<td>0.074</td>
<td>728694</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>5.58</td>
<td>316882</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>7.37</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>0.00000</td>
<td>313621</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>1</td>
<td>0.00000</td>
<td>14.74</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>203394</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>11.28</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2</td>
<td>0.00000</td>
<td>970016</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>1</td>
<td>0.00000</td>
<td>22036</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>23306</td>
<td>0</td>
</tr>
</tbody>
</table>

**TOTAL**: $4035933
### Track Maintenance Cost

**Run Number:** BASE MOD 2  
**Date:** THU, AUG 14 1980  
**Place:** SHAKER RESEARCH CORP.

**Simulation Time:** 1.00

<table>
<thead>
<tr>
<th>Population in Use</th>
<th>Class</th>
<th>Component</th>
<th>Quality</th>
<th>Path Flows Miles/Year</th>
<th>Maintenance Dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td>700.00</td>
<td>1</td>
<td>1</td>
<td>0.06402</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50.00</td>
<td>1</td>
<td>2</td>
<td>0.05849</td>
<td></td>
<td></td>
</tr>
<tr>
<td>100.00</td>
<td>2</td>
<td>1</td>
<td>0.05849</td>
<td></td>
<td></td>
</tr>
<tr>
<td>150.49</td>
<td>2</td>
<td>2</td>
<td>0.05849</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Node Path</th>
<th>Class</th>
<th>Component</th>
<th>Quality</th>
<th>Path Flows Miles/Year</th>
<th>Maintenance Dollars</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.00000</td>
<td>74.34</td>
<td>22020</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0.058476</td>
<td>50.00</td>
<td>30000</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0.060338</td>
<td>150.49</td>
<td>45147</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.00000</td>
<td>692.89</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>47.38</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>2.92</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0.00000</td>
<td>9.415</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>5.95</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2</td>
<td>0.00000</td>
<td>141.41</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>1</td>
<td>0.00000</td>
<td>13760</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>2</td>
<td>0.00000</td>
<td>1754</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>1</td>
<td>0.00000</td>
<td>1754</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>1</td>
<td>0.00000</td>
<td>51.96</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>2</td>
<td>0.00000</td>
<td>3.51</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>5</td>
<td>2</td>
<td>0.00000</td>
<td>6.43</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>4</td>
<td>2</td>
<td>0.00000</td>
<td>9.08</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>1</td>
<td>0.00000</td>
<td>14.67</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>1</td>
<td>0.00000</td>
<td>2.92</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.00000</td>
<td>0.58</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.00000</td>
<td>5.95</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.00000</td>
<td>9.08</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.00000</td>
<td>14619</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>2</td>
<td>0.00000</td>
<td>366800</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>1170</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>1816</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>18763</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>18763</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>21765</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>44.75</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>44.75</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>671257</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>721257</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>745958</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>815449</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>161350</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>161350</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>161350</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>367257</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>4475</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>671257</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>721257</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>745958</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>815449</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>161350</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>161350</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>2</td>
<td>0.00000</td>
<td>161350</td>
<td>0</td>
</tr>
</tbody>
</table>

**Total:** $3404921
## Track Maintenance Cost

**Run Number:** BASE MD 2  
**Date:** THU, AUG 14 1980  
**Place:** SHAKER RESEARCH CORP.

**Simulation Time:** 1.50

### Population in Use

<table>
<thead>
<tr>
<th>Node</th>
<th>Class</th>
<th>Component</th>
<th>Quality</th>
</tr>
</thead>
<tbody>
<tr>
<td>700.00</td>
<td>1</td>
<td>1</td>
<td>0.056800</td>
</tr>
<tr>
<td>30.00</td>
<td>1</td>
<td>2</td>
<td>0.049670</td>
</tr>
<tr>
<td>100.00</td>
<td>2</td>
<td>1</td>
<td>0.049670</td>
</tr>
<tr>
<td>150.67</td>
<td>2</td>
<td>2</td>
<td>0.051650</td>
</tr>
</tbody>
</table>

### Node Path Class Component Quality Path Flows Miles/Year Maintenance Dollars

<table>
<thead>
<tr>
<th>Node</th>
<th>Path</th>
<th>Class</th>
<th>Component</th>
<th>Quality</th>
<th>Flows</th>
<th>Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.056805</td>
<td>7.39</td>
<td>2398.02</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0.049674</td>
<td>5.00</td>
<td>1500.00</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.051649</td>
<td>150.67</td>
<td>4520.01</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>753.93</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>95.03</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>4.97</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>142.89</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>7.78</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>375.32</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>11925</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>375.32</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>375.32</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>375.32</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>1490</td>
<td>0</td>
</tr>
</tbody>
</table>

**Total:** $3112344
<table>
<thead>
<tr>
<th>POPULATION IN USE</th>
<th>CLASS</th>
<th>COMPONENT</th>
<th>QUALITY</th>
<th>PATH FLOWS MILES/YEAR</th>
<th>MAINTENANCE DOLLARS</th>
</tr>
</thead>
<tbody>
<tr>
<td>700.00</td>
<td>1</td>
<td>1</td>
<td>0.050944</td>
<td>890.78</td>
<td>267.235</td>
</tr>
<tr>
<td>50.00</td>
<td>2</td>
<td>2</td>
<td>0.044582</td>
<td>55.03</td>
<td>150.00</td>
</tr>
<tr>
<td>106.93</td>
<td>2</td>
<td>1</td>
<td>0.044582</td>
<td>50.00</td>
<td>30.00</td>
</tr>
<tr>
<td>200.621</td>
<td>2</td>
<td>2</td>
<td>0.046516</td>
<td>150.83</td>
<td>452.47</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NODE PATH</th>
<th>CLASS</th>
<th>COMPONENT</th>
<th>QUALITY</th>
<th>PATH FLOWS MILES/YEAR</th>
<th>MAINTENANCE DOLLARS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0.050944</td>
<td>890.78</td>
<td>267.235</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0.044582</td>
<td>55.03</td>
<td>150.00</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0.044582</td>
<td>50.00</td>
<td>30.00</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>845.40</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>45.38</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0.000000</td>
<td>47.77</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>94.54</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0.000000</td>
<td>4.46</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>143.81</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>7.02</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>3.50</td>
<td>104.87</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>0.45</td>
<td>133.77</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>0.45</td>
<td>133.77</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>48.88</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>2.67</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>4.90</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>7.02</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>33.79</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>15.09</td>
<td>377.03</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>2.23</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>0.45</td>
<td>111.45</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>4.46</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>0.45</td>
<td>111.45</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>7.02</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>37.79</td>
<td>608.293</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>67.59</td>
<td>530.905</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>2.23</td>
<td>191.704</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>4.46</td>
<td>312.07</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>4.46</td>
<td>206.521</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>8.92</td>
<td>123.047</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>7.02</td>
<td>603.361</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>14.03</td>
<td>144.96</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>14.03</td>
<td>144.96</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>14.03</td>
<td>144.96</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>14.03</td>
<td>144.96</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>14.03</td>
<td>144.96</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1.000000</td>
<td>14.03</td>
<td>144.96</td>
</tr>
</tbody>
</table>

| TOTAL | $2990914 |
# TRACK MAINTENANCE COST SUMMARY

**Run Number**: BASE MOD 2  
**Date**: THU, AUG 14 1980  
**Place**: SHAKER RESEARCH CORP.

**Simulation Time**: 2.00

<table>
<thead>
<tr>
<th>Code</th>
<th>Maintenance Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>2128810.42</td>
</tr>
<tr>
<td>32</td>
<td>34963.76</td>
</tr>
<tr>
<td>33</td>
<td>0.00</td>
</tr>
<tr>
<td>34</td>
<td>399331.18</td>
</tr>
<tr>
<td>35</td>
<td>-82700.33</td>
</tr>
<tr>
<td>36</td>
<td>13162.15</td>
</tr>
<tr>
<td>37</td>
<td>6993.17</td>
</tr>
<tr>
<td>38</td>
<td>13986.42</td>
</tr>
<tr>
<td>39</td>
<td>0.00</td>
</tr>
<tr>
<td>40</td>
<td>13986.42</td>
</tr>
<tr>
<td>41</td>
<td>34576.45</td>
</tr>
<tr>
<td>42</td>
<td>406825.09</td>
</tr>
<tr>
<td>43</td>
<td>6993.17</td>
</tr>
<tr>
<td>44</td>
<td>13936.42</td>
</tr>
</tbody>
</table>

**Annual Total**: 2990914.31

<table>
<thead>
<tr>
<th>Class</th>
<th>Maintenance Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1972324.34</td>
</tr>
<tr>
<td>2</td>
<td>1018589.97</td>
</tr>
</tbody>
</table>

**Annual Total**: 2990914.31

<table>
<thead>
<tr>
<th>Component</th>
<th>Maintenance Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2083623.55</td>
</tr>
<tr>
<td>2</td>
<td>907290.76</td>
</tr>
</tbody>
</table>

**Annual Total**: 2990914.31

<table>
<thead>
<tr>
<th>Repair Block</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2220938.07</td>
</tr>
</tbody>
</table>
APPENDIX J

Output Costs From Program Run, Example #2
### Weibull Distribution Parameters

<table>
<thead>
<tr>
<th>CLASS</th>
<th>COMPONENT</th>
<th>AVG. AGE</th>
<th>BETA</th>
<th>C-LIFE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>20.00</td>
<td>2.00</td>
<td>32.00</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

### Functions and Shape Parameters Active

<table>
<thead>
<tr>
<th>PATH FUNC</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>XO</th>
<th>YO</th>
</tr>
</thead>
</table>

### Track Maintenance Cost

- **Run Number**: BASE MOD 2
- **Date**: TUE, AUG 12 1980
- **Place**: SHAKER RESEARCH CORP.
- **Simulation Time**: 0.00

### Population

<table>
<thead>
<tr>
<th>IN USE</th>
<th>CLASS</th>
<th>COMPONENT</th>
<th>QUALITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>100000.00</td>
<td>1</td>
<td>1</td>
<td>0.15000</td>
</tr>
<tr>
<td>100000.00</td>
<td>1</td>
<td>2</td>
<td>0.04000</td>
</tr>
<tr>
<td>200000.00</td>
<td>1</td>
<td>3</td>
<td>0.05000</td>
</tr>
<tr>
<td>0.00</td>
<td>1</td>
<td>4</td>
<td>0.00000</td>
</tr>
<tr>
<td>200000.00</td>
<td>1</td>
<td>5</td>
<td>0.05000</td>
</tr>
</tbody>
</table>

### Node Path

<table>
<thead>
<tr>
<th>NODE</th>
<th>PATH</th>
<th>CLASS</th>
<th>COMPONENT</th>
<th>QUALITY</th>
<th>PATH FLOWS</th>
<th>MAINTENANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>10000.00</td>
<td>20000</td>
</tr>
<tr>
<td>2</td>
<td>12</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>10000.00</td>
<td>20000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>26667.00</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>1</td>
<td>2</td>
<td>0.040000</td>
<td>100000.00</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>1</td>
<td>3</td>
<td>0.050000</td>
<td>80000.00</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
<td>1</td>
<td>5</td>
<td>0.050000</td>
<td>80000.00</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>200000.00</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>200000.00</td>
<td>5000000</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>200000.00</td>
<td>5000000</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>200000.00</td>
<td>5000000</td>
</tr>
<tr>
<td>8</td>
<td>24</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>200000.00</td>
<td>5000000</td>
</tr>
<tr>
<td>9</td>
<td>26</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>1000.00</td>
<td>25000000</td>
</tr>
<tr>
<td>11</td>
<td>30</td>
<td>1</td>
<td>1</td>
<td>0.150000</td>
<td>200000.00</td>
<td>2000000</td>
</tr>
<tr>
<td>12</td>
<td>50</td>
<td>1</td>
<td>1</td>
<td>0.055556</td>
<td>9000.00</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>48</td>
<td>1</td>
<td>1</td>
<td>0.055556</td>
<td>9000.00</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>32</td>
<td>1</td>
<td>1</td>
<td>0.145729</td>
<td>199000.00</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>31</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>44</td>
<td>1</td>
<td>1</td>
<td>0.145729</td>
<td>199000.00</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>17</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>42</td>
<td>1</td>
<td>1</td>
<td>0.145729</td>
<td>199000.00</td>
<td>0</td>
</tr>
<tr>
<td>16</td>
<td>19</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>40</td>
<td>1</td>
<td>1</td>
<td>0.145729</td>
<td>199000.00</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>21</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>38</td>
<td>1</td>
<td>1</td>
<td>0.145729</td>
<td>199000.00</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>23</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>36</td>
<td>1</td>
<td>1</td>
<td>0.145729</td>
<td>199000.00</td>
<td>0</td>
</tr>
<tr>
<td>19</td>
<td>25</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>100.00</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>34</td>
<td>1</td>
<td>1</td>
<td>0.000000</td>
<td>950.00</td>
<td>0</td>
</tr>
<tr>
<td>20</td>
<td>27</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>150.00</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>35</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>150.00</td>
<td>0</td>
</tr>
<tr>
<td>23</td>
<td>37</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>1150.00</td>
<td>0</td>
</tr>
<tr>
<td>24</td>
<td>39</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>2150.00</td>
<td>0</td>
</tr>
<tr>
<td>25</td>
<td>41</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>3150.00</td>
<td>0</td>
</tr>
<tr>
<td>26</td>
<td>43</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>4150.00</td>
<td>0</td>
</tr>
<tr>
<td>27</td>
<td>45</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>5150.00</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>46</td>
<td>1</td>
<td>1</td>
<td>0.287584</td>
<td>31817.00</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>46</td>
<td>1</td>
<td>2</td>
<td>0.040000</td>
<td>100000.00</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>46</td>
<td>1</td>
<td>3</td>
<td>0.050000</td>
<td>90000.00</td>
<td>0</td>
</tr>
<tr>
<td>28</td>
<td>46</td>
<td>1</td>
<td>5</td>
<td>0.050000</td>
<td>80000.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>15</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>9150.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>52</td>
<td>1</td>
<td>1</td>
<td>0.000002</td>
<td>22667.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>15</td>
<td>1</td>
<td>2</td>
<td>1.000000</td>
<td>400.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>52</td>
<td>1</td>
<td>2</td>
<td>0.000000</td>
<td>96000.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>15</td>
<td>1</td>
<td>3</td>
<td>1.000000</td>
<td>400.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>52</td>
<td>1</td>
<td>3</td>
<td>0.000000</td>
<td>76000.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>15</td>
<td>1</td>
<td>5</td>
<td>1.000000</td>
<td>400.00</td>
<td>0</td>
</tr>
<tr>
<td>29</td>
<td>52</td>
<td>1</td>
<td>5</td>
<td>0.000000</td>
<td>76000.00</td>
<td>0</td>
</tr>
<tr>
<td>30</td>
<td>47</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>200.00</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>49</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>11150.00</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>49</td>
<td>1</td>
<td>2</td>
<td>1.000000</td>
<td>400.00</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>49</td>
<td>1</td>
<td>3</td>
<td>1.000000</td>
<td>400.00</td>
<td>0</td>
</tr>
<tr>
<td>31</td>
<td>49</td>
<td>1</td>
<td>5</td>
<td>1.000000</td>
<td>400.00</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>61</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>10000.00</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>300</td>
<td>1</td>
<td>1</td>
<td>1.000000</td>
<td>1150.00</td>
<td>0</td>
</tr>
<tr>
<td>Code</td>
<td>Key</td>
<td>Qty</td>
<td>Description</td>
<td>Rate</td>
<td>Total</td>
<td>Taxes</td>
</tr>
<tr>
<td>------</td>
<td>-----</td>
<td>-----</td>
<td>-------------</td>
<td>------</td>
<td>-------</td>
<td>--------</td>
</tr>
<tr>
<td>152</td>
<td>530</td>
<td>1</td>
<td>0.000000</td>
<td>3000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>152</td>
<td>530</td>
<td>1</td>
<td>0.000000</td>
<td>1000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>152</td>
<td>530</td>
<td>1</td>
<td>0.000000</td>
<td>4000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>171</td>
<td>240</td>
<td>1</td>
<td>1.000000</td>
<td>0.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>180</td>
<td>325</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>181</td>
<td>324</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>182</td>
<td>323</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>183</td>
<td>322</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>184</td>
<td>321</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>185</td>
<td>320</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>186</td>
<td>319</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>187</td>
<td>318</td>
<td>1</td>
<td>1.000000</td>
<td>900.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>187</td>
<td>318</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>188</td>
<td>317</td>
<td>1</td>
<td>1.000000</td>
<td>1800.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>188</td>
<td>317</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>189</td>
<td>316</td>
<td>1</td>
<td>1.000000</td>
<td>2700.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>189</td>
<td>316</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>190</td>
<td>315</td>
<td>1</td>
<td>1.000000</td>
<td>4500.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>190</td>
<td>315</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>191</td>
<td>301</td>
<td>1</td>
<td>1.000000</td>
<td>1150.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>191</td>
<td>301</td>
<td>1</td>
<td>1.000000</td>
<td>1000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>191</td>
<td>301</td>
<td>1</td>
<td>1.000000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>192</td>
<td>302</td>
<td>1</td>
<td>1.000000</td>
<td>1150.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>192</td>
<td>302</td>
<td>1</td>
<td>1.000000</td>
<td>1000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>192</td>
<td>302</td>
<td>1</td>
<td>1.000000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>196</td>
<td>310</td>
<td>1</td>
<td>1.000000</td>
<td>4500.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>196</td>
<td>310</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>197</td>
<td>311</td>
<td>1</td>
<td>0.285714</td>
<td>0.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>197</td>
<td>312</td>
<td>1</td>
<td>1.000000</td>
<td>4500.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>197</td>
<td>312</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>198</td>
<td>313</td>
<td>1</td>
<td>1.000000</td>
<td>4500.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>199</td>
<td>315</td>
<td>1</td>
<td>1.000000</td>
<td>4500.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>199</td>
<td>315</td>
<td>1</td>
<td>0.500000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>200</td>
<td>333</td>
<td>1</td>
<td>1.000000</td>
<td>1150.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>200</td>
<td>333</td>
<td>1</td>
<td>1.000000</td>
<td>1000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>200</td>
<td>333</td>
<td>1</td>
<td>1.000000</td>
<td>2000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>201</td>
<td>340</td>
<td>1</td>
<td>0.087453</td>
<td>13150.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>201</td>
<td>340</td>
<td>1</td>
<td>0.250000</td>
<td>4000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>201</td>
<td>340</td>
<td>1</td>
<td>0.666667</td>
<td>3000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
<tr>
<td>201</td>
<td>340</td>
<td>1</td>
<td>0.000000</td>
<td>4000.00</td>
<td>0.000000</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**TOTAL $ 1005067954**
## Track Maintenance Cost Summary

**Run Number:** Example Run #2  
**Date:** Tue, Aug 12 1980  
**Place:** Shaker Research Corp.

**Simulation Time:** 0.00

<table>
<thead>
<tr>
<th>Cost Code</th>
<th>Maintenance Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>797912137.60</td>
</tr>
<tr>
<td>32</td>
<td>39579229.00</td>
</tr>
<tr>
<td>33</td>
<td>0.00</td>
</tr>
<tr>
<td>34</td>
<td>0.00</td>
</tr>
<tr>
<td>35</td>
<td>0.00</td>
</tr>
<tr>
<td>36</td>
<td>48575000.00</td>
</tr>
<tr>
<td>37</td>
<td>17631170.55</td>
</tr>
<tr>
<td>38</td>
<td>3903200.00</td>
</tr>
<tr>
<td>39</td>
<td>0.00</td>
</tr>
<tr>
<td>40</td>
<td>44442000.00</td>
</tr>
<tr>
<td>41</td>
<td>40080000.00</td>
</tr>
<tr>
<td>42</td>
<td>99030834.00</td>
</tr>
<tr>
<td>43</td>
<td>24285884.00</td>
</tr>
<tr>
<td>44</td>
<td>10468500.00</td>
</tr>
<tr>
<td>45</td>
<td>5232000.00</td>
</tr>
<tr>
<td>46</td>
<td>0.00</td>
</tr>
<tr>
<td>47</td>
<td>0.00</td>
</tr>
<tr>
<td>48</td>
<td>0.00</td>
</tr>
<tr>
<td>49</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Annual Total:** $1005067954.

<table>
<thead>
<tr>
<th>Class</th>
<th>Maintenance Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1005067954.00</td>
</tr>
</tbody>
</table>

**Annual Total:** $1005067954.
### Component Maintenance Cost

<table>
<thead>
<tr>
<th>Component</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>893686666.15</td>
</tr>
<tr>
<td></td>
<td>300000.00</td>
</tr>
<tr>
<td></td>
<td>101609408.00</td>
</tr>
<tr>
<td></td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>9471880.00</td>
</tr>
<tr>
<td><strong>Annual Total</strong></td>
<td><strong>$1005067954.00</strong></td>
</tr>
</tbody>
</table>

### Repair Block Cost

<table>
<thead>
<tr>
<th>Repair Block</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$41430583</td>
</tr>
<tr>
<td>2</td>
<td>$62800043</td>
</tr>
<tr>
<td>3</td>
<td>$110000793</td>
</tr>
<tr>
<td>4</td>
<td>$191200793</td>
</tr>
<tr>
<td>5</td>
<td>$180100450</td>
</tr>
<tr>
<td>6</td>
<td>$434000450</td>
</tr>
<tr>
<td>7</td>
<td>0.00</td>
</tr>
<tr>
<td>8</td>
<td>75230056</td>
</tr>
<tr>
<td>9</td>
<td>94800024</td>
</tr>
<tr>
<td>10</td>
<td>580200560</td>
</tr>
<tr>
<td>11</td>
<td>43570048</td>
</tr>
<tr>
<td>12</td>
<td>0.00</td>
</tr>
<tr>
<td>13</td>
<td>53130000</td>
</tr>
<tr>
<td>14</td>
<td>1000000</td>
</tr>
<tr>
<td>15</td>
<td>1300000</td>
</tr>
<tr>
<td>16</td>
<td>0.00</td>
</tr>
<tr>
<td>17</td>
<td>175000</td>
</tr>
<tr>
<td>18</td>
<td>0.00</td>
</tr>
<tr>
<td>19</td>
<td>0.00</td>
</tr>
<tr>
<td>20</td>
<td>0.00</td>
</tr>
</tbody>
</table>

J-10
APPENDIX K

TRACK COST MODEL MAINTENANCE ACTION DIAGRAMS

This appendix contains drawings of all the maintenance action diagrams used for showing examples of the simulation cost methodology. Figure K-1 is a simplified maintenance action diagram for example run number 1. This diagram is to be used along with the input and output data of Appendices G and I, respectively. The costs in Appendices G and I are keyed to the path numbers shown in this Figure K-1.

Figure K-2 is a full maintenance action diagram for example run number 2. This diagram has twenty (20) separate maintenance repair blocks which are individually drawn out, numbered, and shown in Figures K-4 through K-24 of this appendix. Figure K-3 is a blank repair block numbering sheet which can be used to renumber or add paths to the existing diagram as shown in Figure K-2 if needed in the future.
Figure K-1. Track in Use, Schematic Diagram Number 0001

N indicates node by (N) number-in-circle. Small numbers (n) are path numbers.
FIGURE K-3. REPAIR OPERATION
# Repair Operation 1

**Lay New Welded Rail (Method 1)**

<table>
<thead>
<tr>
<th>Component Name</th>
<th>Path</th>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bring Rail to Site and Distribute</td>
<td></td>
<td>171</td>
<td>x</td>
</tr>
<tr>
<td>Bring Special Equipment</td>
<td></td>
<td>201</td>
<td>x</td>
</tr>
<tr>
<td>Crib and Broom</td>
<td></td>
<td>261</td>
<td>x</td>
</tr>
<tr>
<td>Pull Spikes and Stack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wide Gauge Threader</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lift Anchors and Plates</td>
<td></td>
<td>261</td>
<td>x</td>
</tr>
<tr>
<td>Set Wooden Plugs/Creosote</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cribbing Machine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adzer (Dual) Face Off Ties</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tie Plate Liner</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gauge Threader (Place New Rail)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heat Rail</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gauge Liner and Spiker</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tamper</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rail Anchoring Machine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rail Grease Machine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Rail Quenching</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Power Drill</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Weld Processing Machine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heavy Duty Cribbing Machine</td>
<td></td>
<td>261</td>
<td>x</td>
</tr>
</tbody>
</table>

## Unit Specifications
- **171**: Bring Rail to Site and Distribute
- **201**: Bring Special Equipment
- **261**: Crib and Broom, Pull Spikes and Stack, Wide Gauge Threader, Lift Anchors and Plates, Set Wooden Plugs/Creosote, Cribbing Machine, Adzer (Dual) Face Off Ties, Tie Plate Liner, Gauge Threader (Place New Rail), Heat Rail, Gauge Liner and Spiker, Tamper, Rail Anchoring Machine, Rail Grease Machine, Rail Quenching, Power Drill, Weld Processing Machine, Heavy Duty Cribbing Machine

## Notes
- To Renew (Reusable Material Only): 401
- To Scrap (All Defective): 431

---

**Figure K-4. Repair Operation 1**
### REPAIR 2
LAY NEW WELDED RAIL (METHOD 2)

<table>
<thead>
<tr>
<th>COMPONENT NUMBER</th>
<th>PATH</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>172</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>202</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>262</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>372</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>402</td>
<td>x</td>
</tr>
<tr>
<td></td>
<td>432</td>
<td>x</td>
</tr>
</tbody>
</table>

- Bring rail to site and distribute
- Bring in special equipment
- Full spikes
- Remove anchors
- Lift out old rail
- Lift tie plates
- Drive in exposed broken spikes
- Insert wooden plugs
- Face off tie with adzer (single)
- Broom off ties
- Distribute new spikes/anchors/plates
- Begin rail set-in (match end)
- Continue set-in process
- Gauge liner and spike
- Anchor rails
- End matching process
- Follow up spiking
- Themite weld

#### TO RENEW (REUSABLE MATERIAL ONLY)
402 x x

#### TO SCRAP (ALL DETECTIVE)
432 x x

---

**FIGURE K-5. REPAIR OPERATION 2**

K-6
## Repair Operation 3

### Lay New Welded Rail (Same WT. Welded)

<table>
<thead>
<tr>
<th>Path</th>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>173</td>
<td></td>
<td></td>
</tr>
<tr>
<td>203</td>
<td></td>
<td></td>
</tr>
<tr>
<td>263</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Bring Rail to Site and Distribute
- Bring Special Equipment
- Pull Spikes/Anchor (Leave Plates)
- Lift Out Old Rail
- Lay Out Spikes/Anchors/Joint Bars
- Begin Setting Rail (Match End)
- Continue Set-In Process
- Special Crossing Considerations
- Spike Down Rail (Initial)
- Anchor Rail
- Match End Process
- Thermite Weld

### Follow Up Spiking

<table>
<thead>
<tr>
<th>Path</th>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>263</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Follow Up Spiking

### Special Crossing

- Spike Down Rail (Initial)
- Anchor Rail

### Return Path

- To Renew (Reusable Material Only)
- To Scrap (All Defective)

---

**Figure K-6. Repair Operation 3**
# Repair Operation 4: Rail Change Out (Jointed)

**Procedure:**
- Bring rail to repair site
- Distribute in advance where needed
- Work around train flow
- Pull one section of rail
- Insert new section of rail
- Spike/anchor
- Replace signal wires

**Component List:**

<table>
<thead>
<tr>
<th>Path</th>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>174</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>204</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>264</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>264</td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>264</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

**Disposition Options:**
- To renew (reusable material only) 404 ✓
- To scrap (all defective) 434 ✓

---

*FIGURE K-7. REPAIR OPERATION 4*
**Figure K-8. Repair Operation 5**

### Repair 5

<table>
<thead>
<tr>
<th>Activity</th>
<th>Path</th>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Travel to Repair Site</td>
<td>175</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pull Spikes</td>
<td>265</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lift Out Rail Sections</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drive Down Broken/Exposed Spikes</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Face Off Ties With Adzer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plug Spike Holes &amp; Creosote</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset Rail Sections</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spike &amp; Anchor Rail</td>
<td>265</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regulate Ballast as Needed</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**No Renew**

<table>
<thead>
<tr>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>405</td>
<td></td>
</tr>
</tbody>
</table>

**No Scrap**

<table>
<thead>
<tr>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>435</td>
<td></td>
</tr>
</tbody>
</table>
FIGURE K-9. REPAIR OPERATION 6
**REPAIR 7**

**TRACK PANEL INSTALL**

<table>
<thead>
<tr>
<th>COMPONENT NUMBER</th>
<th>PATH</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>PREDROP PANELS AT REPAIR SITE</td>
<td>177</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PULL SPIKES</td>
<td>207</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PULL RAIL/ANCHORS/PLATES</td>
<td>277</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>LIFT TIES</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PREPARE BALLAST</td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>SET PANEL</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>SECURE/LINE ENDS</td>
<td>277</td>
<td>x</td>
<td>x</td>
<td></td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

TO RENEW (ALL REUSABLE MATERIAL) 407 x
TO SCRAP (ALL DEFECTIVES) 437 x

**FIGURE K-10. REPAIR OPERATION 7**
FIGURE K-11. REPAIR OPERATION 8

<table>
<thead>
<tr>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
</tr>
</tbody>
</table>

- DUMP BALLAST AT REWORK SITE: Component #178
- TRAVEL TO REPAIR SITE: Component #208
- LIFT AND LEVEL TRACK: Component #278
- TAMPER: Component #278
- ALIGN: Component #278
- REGULATE BALLAST (MACH.): Component #278
- TIGHTEN BOLTS (IF NEEDED): Component #278
- SPECIAL CROSSING FEATURES: Component #278
- SPECIAL SWITCH FEATURES: Component #278
- TRACK BROOM: Component #278

NO RENEW: Component #408
NO SCRAP: Component #438
SCRAP: Component #438
REPAIR 9

SMOOTHING

<table>
<thead>
<tr>
<th>COMPONENT</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUMP BALLAST AT REWORK SITE</td>
<td>x x</td>
</tr>
<tr>
<td>TRAVEL TO SITE</td>
<td>x x</td>
</tr>
<tr>
<td>TAM²</td>
<td></td>
</tr>
<tr>
<td>LINER</td>
<td>x x</td>
</tr>
<tr>
<td>REGULATE BALLAST</td>
<td>209</td>
</tr>
</tbody>
</table>

FIGURE K-12. REPAIR OPERATION 9
**REPAIR 10**

**TIMBER & SURFACING**

<table>
<thead>
<tr>
<th>COMPONENT NUMBER</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>12345</td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>x</td>
</tr>
<tr>
<td>210</td>
<td>x</td>
</tr>
<tr>
<td>280</td>
<td>x</td>
</tr>
</tbody>
</table>

- **Distribute New Ties Along Site**
- **Travel to Site**
- **Tie Broom and Crib**
- **Pull Spikes**
- **Remove Anchors**
- **Remove Ties**
- **Reset New Ties**
- **Spike and Anchor**
- **Add Ballast As Needed**
- **Regulate Ballast**
- **Tie Plate Broom**
- **Tamp Ballast**
- **Align Track**
- **Track Broom & Sweep**
- **Special Features at Crossings**
- **Special Features at Switches**

**PATH**

<table>
<thead>
<tr>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>180</td>
<td></td>
</tr>
<tr>
<td>210</td>
<td></td>
</tr>
<tr>
<td>280</td>
<td></td>
</tr>
</tbody>
</table>

**TO RENEW (ALL REUSABLE UNITS)**

<table>
<thead>
<tr>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>410</td>
<td></td>
</tr>
</tbody>
</table>

**TO SCRAP (ALL DEFECTIVES)**

<table>
<thead>
<tr>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>440</td>
<td></td>
</tr>
</tbody>
</table>

**FIGURE K-13. REPAIR OPERATION 10**
<table>
<thead>
<tr>
<th>Component Number</th>
<th>PATH</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>181</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>211</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>281</td>
<td></td>
<td>x</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>411</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>441</td>
<td></td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Distribute new ties along site
- Mark ties for replacement
- Travel to site
- Pull spikes
- Cut up ties
- Push out remains
- Set new ties
- Inject
- Lift rail/insert plates
- Spike
- tamp
- Regulate ballast
- Special features at crossings
- Special features at switches

**Figure K-14. Repair Operation 11**
FIGURE K-15. REPAIR OPERATION 12
FIGURE K-16. REPAIR OPERATION 13
FIGURE K-17. REPAIR OPERATION 14
REPAIR [15]

RAIL GRIND

<table>
<thead>
<tr>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>185</td>
<td>X X</td>
<td>X</td>
</tr>
<tr>
<td>285</td>
<td>X X</td>
<td>X</td>
</tr>
<tr>
<td>285</td>
<td>X X</td>
<td>X</td>
</tr>
<tr>
<td>285</td>
<td>X X</td>
<td>X</td>
</tr>
</tbody>
</table>

MOVE HEAVY EQUIPMENT TO SITE
RAILROAD OVERVIEW OF CONTRACTOR
GRIND RAIL AS NEEDED
INSPECT AS COMPLETED

FIGURE K-18. REPAIR OPERATION 15
<table>
<thead>
<tr>
<th>COMPONENT NUMBER</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>DUMP NEW BALLAST ALONG SITE</td>
<td>x</td>
</tr>
<tr>
<td>BRING HEAVY EQUIPMENT TO SITE</td>
<td>x</td>
</tr>
<tr>
<td>OPERATE CLEANING MACHINE</td>
<td></td>
</tr>
</tbody>
</table>

FIGURE K-19. REPAIR OPERATION 16
### Repair Operation 17

**Brush & Weed Control**

<table>
<thead>
<tr>
<th>Path</th>
<th>Component Number</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>187</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>287</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>287</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>287</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

**Activities:**
- Travel to Site
- Manual Brush Removal
- Large Equipment Brush Removal
- Automatic Spray & Weed Dispense

---

**Return Path**

- No Renew
- No Scrap

---

**FIGURE K-20. REPAIR OPERATION 17**
FIGURE K-21. REPAIR OPERATION 18
## REPAIR OPERATION 19

**SNOW/ICE REMOVAL**

<table>
<thead>
<tr>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>170</td>
<td></td>
<td></td>
</tr>
<tr>
<td>139</td>
<td></td>
<td></td>
</tr>
<tr>
<td>110</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td></td>
<td></td>
</tr>
<tr>
<td>258</td>
<td></td>
<td></td>
</tr>
<tr>
<td>113</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- TRAVEL TO SITES
- HAND SHOVEL (CLEAR SWITCHES)
- PLOWS (WORK TRAIN)
- BLOWER (WORK TRAIN)
- NO RENEW
- NO SCRAP

**Figure K-22.** REPAIR OPERATION 19
REPAIR [20]

<table>
<thead>
<tr>
<th>SLIDES/WASHOUTS</th>
<th>PATH</th>
<th>COMPONENT NUMBER</th>
<th>COST</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEND BALLAST/TIES/RAIL</td>
<td>190</td>
<td>x x x x</td>
<td>x x</td>
</tr>
<tr>
<td>MOVE MEN/EQUIPMENT TO SITE</td>
<td>220</td>
<td>x x x x</td>
<td>x x</td>
</tr>
<tr>
<td>FILL SITE</td>
<td>290</td>
<td>x x x x</td>
<td>x x</td>
</tr>
<tr>
<td>RIP-RAP</td>
<td>290</td>
<td>x x x x</td>
<td>x x</td>
</tr>
<tr>
<td>REINSTALL TRACK</td>
<td>290</td>
<td>x x x x</td>
<td>x x</td>
</tr>
</tbody>
</table>

FIGURE K-23. REPAIR OPERATION 20
APPENDIX L

REPORT OF NEW TECHNOLOGY

The work described in this report concerns the application of a methodology, the simulation cost model (SCM), to the economic aspects of maintaining and/or laying of rail, ties, and ballast. Because the work was not concerned with devices, testing, or construction, no inventions were developed. However, the work did result in a methodology which can be applied to economic systems beyond those associated with a single track maintenance system. Given the proper input data most any track maintenance operation can be modeled and simulated by the computer program developed. Alternate systems most appropriately treated by the SCM consist mainly of large sets of individual components. The SCM methodology requires the user to draw a maintenance action diagram and provide an initial set of input data for its paths before running the computer program to obtain a cost simulation output. The output includes a quantitative description of current (present time) annual system operation costs and indicates quantitatively the most costly portions of the system as well as provides a projection of future system operating costs.

55 copies