**REPORT DOCUMENTATION PAGE**

<table>
<thead>
<tr>
<th><strong>1. REPORT NUMBER</strong></th>
<th><strong>2. GOVT ACCESSION NO.</strong></th>
<th><strong>3. RECIPIENT'S CATALOG NUMBER</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>DINSRDC-81/017</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>4. TITLE (and Subtitle)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>AUTOMATIC VEHICLE SCHEDULING (AVS) PROGRAMMER'S INSTRUCTION MANUAL FOR THE BURLROUGHS 3500 COMPUTERS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>5. TYPE OF REPORT &amp; PERIOD COVERED</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Final</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>6. PERFORMING ORG. REPORT NUMBER</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>7. AUTHOR(S)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>R. Winchell</td>
</tr>
<tr>
<td>R. Melton</td>
</tr>
<tr>
<td>M. Natrella</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>9. PERFORMING ORGANIZATION NAME AND ADDRESS</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>David W. Taylor Naval Ship Research and Developement Center</td>
</tr>
<tr>
<td>Bethesda, Maryland 20084</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>10. PROGRAM ELEMENT, PROJECT, TASK AREA &amp; WORK UNIT NUMBERS</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Work Unit 1800-007</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>12. REPORT DATE</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>February 1981</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>13. NUMBER OF PAGES</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>197</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>15. SECURITY CLASS. (of this report)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>UNCLASSIFIED</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>16. DISTRIBUTION STATEMENT (of this Report)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>APPROVED FOR PUBLIC RELEASE: DISTRIBUTION UNLIMITED</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>19. KEY WORDS (Continue on reverse side if necessary and identify by block number)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>AVS, warehousing, scheduling, straddle trucks, transporters, industrial tractors, NSC</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>20. ABSTRACT (Continue on reverse side if necessary and identify by block number)</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated Vehicle Scheduling (AVS) is a software package designed to assist in scheduling palletized cargo delivery to warehouses in a Navy Supply Center. The package consists of two scheduling programs, which schedule regular and emergency orders, respectively, and a transaction history file update/report program.</td>
</tr>
</tbody>
</table>

**DD FORM 1473 EDITION OF 1 NOV 65 IS OBSOLETE**

**UNCLASSIFIED SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)**
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF FIGURES</td>
<td>iv</td>
</tr>
<tr>
<td>TABLE</td>
<td>iv</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>1</td>
</tr>
<tr>
<td>ADMINISTRATIVE INFORMATION</td>
<td>1</td>
</tr>
<tr>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>BACKGROUND</td>
<td>3</td>
</tr>
<tr>
<td>AVS1 - REGULAR ORDER SCHEDULING</td>
<td>8</td>
</tr>
<tr>
<td>INPUT</td>
<td>8</td>
</tr>
<tr>
<td>METHOD AND ALGORITHM</td>
<td>10</td>
</tr>
<tr>
<td>PROGRAM OUTPUT</td>
<td>11</td>
</tr>
<tr>
<td>AVS2 - EMERGENCY ORDER SCHEDULING</td>
<td>14</td>
</tr>
<tr>
<td>INPUT</td>
<td>14</td>
</tr>
<tr>
<td>METHOD AND ALGORITHM</td>
<td>15</td>
</tr>
<tr>
<td>PROGRAM OUTPUT</td>
<td>16</td>
</tr>
<tr>
<td>SPECIAL TECHNIQUES</td>
<td>17</td>
</tr>
<tr>
<td>TRAVEL TIME TECHNIQUES</td>
<td>17</td>
</tr>
<tr>
<td>LINKED LIST TECHNIQUES</td>
<td>18</td>
</tr>
<tr>
<td>DATA PACKING</td>
<td>23</td>
</tr>
<tr>
<td>AVS3 - HISTORY/UPDATE PROGRAM</td>
<td>24</td>
</tr>
<tr>
<td>INPUT</td>
<td>24</td>
</tr>
<tr>
<td>PROGRAM OUTPUT</td>
<td>29</td>
</tr>
<tr>
<td>ACKNOWLEDGMENT</td>
<td>31</td>
</tr>
<tr>
<td>APPENDIX A - USER'S MANUAL</td>
<td>33</td>
</tr>
<tr>
<td>APPENDIX B - PROGRAMMER'S GUIDE - SUBROUTINE FLOWCHARTS/LISTINGS</td>
<td>49</td>
</tr>
<tr>
<td>APPENDIX C - AVSIN1, A SINR COBOL DRIVER ROUTINE, LISTINGS</td>
<td>123</td>
</tr>
<tr>
<td>APPENDIX D - AVS3, HISTORY/UPDATE, FLOWCHARTS AND LISTINGS</td>
<td>139</td>
</tr>
<tr>
<td>APPENDIX E - SAMPLE RUNS</td>
<td>177</td>
</tr>
</tbody>
</table>
### LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Map of Charleston Navy Base - AVS Serviced Warehouses</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>AVS System Flow Chart</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>All Vehicle Types - Order Allocations</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>Straddle Order Allocations</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>Transporter Order Allocations</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>Tractor Trailers/Industrial Tractors Order Allocation</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>Methods of Data Storage, Duplicate Arrays</td>
<td>20</td>
</tr>
<tr>
<td>8</td>
<td>Methods of Data Storage, Sequential Sort</td>
<td>21</td>
</tr>
<tr>
<td>9</td>
<td>Methods of Data Storage, Linked Lists</td>
<td>22</td>
</tr>
<tr>
<td>10</td>
<td>Individual Entry Update</td>
<td>27</td>
</tr>
<tr>
<td>11</td>
<td>Record Consolidation</td>
<td>27</td>
</tr>
<tr>
<td>12</td>
<td>Update From AVS Schedule</td>
<td>28</td>
</tr>
<tr>
<td>13</td>
<td>AVS3 Frame Description</td>
<td>30</td>
</tr>
<tr>
<td>14</td>
<td>Frame A01</td>
<td>38</td>
</tr>
<tr>
<td>15</td>
<td>Frame A02</td>
<td>39</td>
</tr>
<tr>
<td>16</td>
<td>Frame A03</td>
<td>40</td>
</tr>
<tr>
<td>17</td>
<td>Frame A04</td>
<td>41</td>
</tr>
<tr>
<td>18</td>
<td>Frame A05</td>
<td>44</td>
</tr>
<tr>
<td>19</td>
<td>Frame A07</td>
<td>47</td>
</tr>
</tbody>
</table>

Table 1 - NSC Charleston Warehouses Serviced by AVS Listed by Group and Number Within Group.
ABSTRACT

Automated Vehicle Scheduling (AVS) is a software package designed to assist in scheduling palletized cargo delivery to warehouses in a Navy Supply Center. The package consists of two scheduling programs, which schedule regular and emergency orders, respectively, and a transaction history file update/report program.

ADMINISTRATIVE INFORMATION

This study was authorized by the Naval Supply Systems Command with funding under Task Area 53/531/091.

INTRODUCTION

Automated Vehicle Scheduling (AVS) is a software package designed to assist in scheduling palletized cargo delivery among warehouses in a Navy Supply Center (NSC). The package consists of two scheduling programs, AVS1 and AVS2, which schedule, respectively, regular and emergency orders, and a transaction history file update/report program, AVS3. They are written in FORTRAN and are designed to run on Burroughs B3500 computers at NSC, Charleston, S.C. and at the Fleet Material Support Office, Mechanicsburg, PA. An earlier version was designed for the CDC 6600 computer at DTNSRDC.

AVS can schedule up to 99 orders totalling about 2000 pallets among as many as 99 warehouses. Deliveries and/or pickups are made by as many as 50 vehicles of four general types: straddle trucks, transporter vehicles, tractor trailers, and industrial tractors. Routes are built to "maximum" efficiency within the limitations of the algorithm used.

AVS2 uses the routes prepared by AVS1 to schedule servicing of emergency orders placed during the regular daily routine. An emergency order can include from 1 to 99 pallets; it can preempt regular orders if the dispatcher desires; it can be handled by a single vehicle type or by a mix of vehicles; finally, the vehicles selected to service it may be those used for regular orders, a subset of these, vehicles previously
unused, or any combination of these vehicles. As many as 99 emergency orders may be considered in the same AVS2 run.

For AVS to be successful, the programs must be easily usable by dispatch personnel who have had minimal computer training. In addition, the scheduling programs must execute rapidly to assure fast response to orders. For these reasons the AVS programs are interactive, tutorial, and corrective, using cathode ray tube (CRT) terminals connected to the B3500. Program procedures, execution instructions, and output file storage are simple. Data are requested from the user by the Information Retrieval System (SINR) and inputs are checked for validity by a COBOL driver program, AVSINI. Instructions are available to the user in frame form displayed at the CRT terminal. Schedules are generated only after the user has checked the correctness of the data.
BACKGROUND

Since the impetus for undertaking the AVS project came from NSC Charleston, the AVS programs described here address operations at that installation. A brief description of Charleston's local delivery procedures is given in this section.

NSC Charleston includes 78 pick-up/delivery sites and eight piers, plus six off-base sites (Table 1). Twenty or more of these are used in a typical half-day's schedule. The dispatch operation is run from building 1078. (A map of the Charleston complex is given in Figure 1; the warehouses are listed in Table 1.)

Orders for palletized cargo movement fall into three priority classes. Group 3 orders are telephoned to the dispatcher twice a day: at 1:00 and 1500 hours. Group 2 and Group 1 are priority orders requiring service within 8 hours and 4 hours, respectively. They may be called in at any time, but in practice are usually phoned in at the same time as Group 3 orders. Orders are ready for shipment at the warehouses when the dispatcher is called to request transportation. At present the dispatch supervisor prepares the vehicle schedules from the order list using his knowledge of the base layout; there is no documented formal procedure. The vehicles are radio dispatched to service these requests.

However, there is additional cargo movement which is not handled in this way. At certain warehouses the high volume of cargo that is routinely shipped/received is moved by vehicles assigned exclusively to those locations. These movements will not be scheduled by AVS initially.

Orders are serviced by four types of vehicles: straddle trucks, transporter vehicles, conventional tractor trailers, and industrial tractors. These vehicles will be designated in the remainder of this report by the abbreviations ST, TR, TT, and IT, respectively. These vehicles are distinguished by their operational characteristics, such as highway speed, load time, manner of loading, and the skills and ratings of the drivers who operate them.

TT's carry from four to fourteen pallets, are capable of highway speeds, but are relatively slow at loading and unloading. They must be
backed up to a loading platform and loaded by forklifts. TT's are the only vehicles which service the off-base sites.

TR's carry either ten or twelve pallets, are somewhat slower than TT's, and are more efficient at loading. They drive up to a loading platform, the operator's cab swings out of the way, the height of the truck bed is adjusted, and pallets are loaded onto a gravity conveyor which delivers them to a roller bed in the truck. When gravity conveyors are not available at a site, forklifts must be used instead.

ST's carry five or seven pallets, are slower than TT's or TR's on the road, but are the most efficient at loading. Pallets are aligned at the pick-up site; the ST lowers a set of lifting rails which fit into channels on the sides of the pallets; the ST then lifts the pallets and drives off. The procedure is reversed for unloading. ST's can service up to three warehouse origins per route segment, i.e., loading/unloading cycle.

IT's carry up to fourteen pallets, are slower than TT's, TR's, or ST's on the road. Loading and unloading are the same as for TT's. IT's are for use within the complex only.

The algorithm places minimum load requirements on each vehicle type for route assignment. These requirements are: ST, 3 pallets; TR, 8 pallets; TT, 14 pallets; and IT, 8 pallets.
### Table 1 - NSC Charleston Warehouses Serviced by AVS

Listed by Group and Number Within Group

<table>
<thead>
<tr>
<th>Group</th>
<th>Num</th>
<th>Name(s)</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>NORTH</td>
<td>1</td>
<td>191</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1601a</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1601b</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1602</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1603</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>1604</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>1605</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>1606</td>
<td>B279</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>1621</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>1622</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>1628</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>A</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>13</td>
<td>1620</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>14</td>
<td>1157</td>
<td>&quot;</td>
</tr>
<tr>
<td>CENTR</td>
<td>15</td>
<td>SM, 45</td>
<td>Serve Mart</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>46</td>
<td>6th Nav Dist</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>53C</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>18</td>
<td>64E</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>19</td>
<td>64W</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>66E</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>21</td>
<td>66W</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>67E</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>23</td>
<td>67W</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>198</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>1078</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>26</td>
<td>1127</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>27</td>
<td>1138</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>28</td>
<td>56</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>29</td>
<td>49</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>SF</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>31</td>
<td>SFR</td>
<td>&quot;</td>
</tr>
<tr>
<td>NSYN</td>
<td>32</td>
<td>2</td>
<td>USNSY</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>3</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>34</td>
<td>5</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>8</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>36</td>
<td>35</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>37</td>
<td>43C</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>38</td>
<td>44</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>39</td>
<td>59</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>223</td>
<td>&quot;</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Group</th>
<th>Num</th>
<th>Name(s)</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>WEST</td>
<td>41</td>
<td>1502</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>42</td>
<td>1503</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>43</td>
<td>1507</td>
<td>&quot;</td>
</tr>
<tr>
<td>NSWT</td>
<td>44</td>
<td>80</td>
<td>USNSY</td>
</tr>
<tr>
<td></td>
<td>45</td>
<td>177</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>46</td>
<td>1143</td>
<td>Spec Serv</td>
</tr>
<tr>
<td></td>
<td>47</td>
<td>1199</td>
<td>USNSY</td>
</tr>
<tr>
<td></td>
<td>48</td>
<td>98</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>49</td>
<td>187</td>
<td>USNSY</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>216</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>51</td>
<td>1175</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>1169</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>53</td>
<td>1171</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>54</td>
<td>1172</td>
<td>S.O.A.P.</td>
</tr>
<tr>
<td></td>
<td>55</td>
<td>1173</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>56</td>
<td>1174</td>
<td>USNSY</td>
</tr>
<tr>
<td></td>
<td>57</td>
<td>218</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>58</td>
<td>X10</td>
<td>AS17</td>
</tr>
<tr>
<td></td>
<td>59</td>
<td>193</td>
<td>NSC</td>
</tr>
<tr>
<td></td>
<td>60</td>
<td>224</td>
<td>USNSY</td>
</tr>
<tr>
<td></td>
<td>61</td>
<td>L</td>
<td>PIER</td>
</tr>
<tr>
<td></td>
<td>62</td>
<td>M</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>63</td>
<td>N</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>P</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>65</td>
<td>Q</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>R</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>67</td>
<td>S</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>68</td>
<td>T</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>69</td>
<td>X20</td>
<td>&quot;</td>
</tr>
<tr>
<td>SOUTH</td>
<td>70</td>
<td>30</td>
<td>RTC1</td>
</tr>
<tr>
<td></td>
<td>71</td>
<td>43S</td>
<td>not used</td>
</tr>
<tr>
<td></td>
<td>72</td>
<td>61</td>
<td>FBMTC</td>
</tr>
<tr>
<td></td>
<td>73</td>
<td>84</td>
<td>Comm.Ctr.</td>
</tr>
<tr>
<td></td>
<td>74</td>
<td>202</td>
<td>RTC1</td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>646</td>
<td>USNS</td>
</tr>
<tr>
<td></td>
<td>76</td>
<td>647</td>
<td>&quot;</td>
</tr>
<tr>
<td></td>
<td>77</td>
<td>655</td>
<td>Comm.Store</td>
</tr>
<tr>
<td></td>
<td>78</td>
<td>656</td>
<td>Navy Ex.</td>
</tr>
<tr>
<td></td>
<td>79</td>
<td>52</td>
<td>&quot;</td>
</tr>
</tbody>
</table>

5
<table>
<thead>
<tr>
<th>Group</th>
<th>Num</th>
<th>Name(s)</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCRFT</td>
<td>80</td>
<td>1</td>
<td>Mine Craft</td>
</tr>
<tr>
<td></td>
<td>81</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>82</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>83</td>
<td>23</td>
<td></td>
</tr>
<tr>
<td></td>
<td>84</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td></td>
<td>85</td>
<td>53S</td>
<td></td>
</tr>
<tr>
<td>X54</td>
<td>86</td>
<td>X54</td>
<td>Comm.Ctr.</td>
</tr>
<tr>
<td>OFF BASE</td>
<td>87</td>
<td>ABASE</td>
<td>Air base</td>
</tr>
<tr>
<td></td>
<td>88</td>
<td>NWS</td>
<td>Nav Weap Sta</td>
</tr>
<tr>
<td></td>
<td>89</td>
<td>DEYTN</td>
<td>Deytens SY</td>
</tr>
<tr>
<td></td>
<td>90</td>
<td>BRASW</td>
<td>Braswell SY</td>
</tr>
<tr>
<td></td>
<td>91</td>
<td>CSNWS</td>
<td>ComStoreNWS</td>
</tr>
<tr>
<td></td>
<td>92</td>
<td>NMEDC</td>
<td>Nav Hosp</td>
</tr>
</tbody>
</table>
AVSI - REGULAR ORDER SCHEDULING

Regular order scheduling takes place in four phases. The first phase is interactive data entry from a remote CRT terminal (handled by the SINR Information Retrieval System available on the B3500 computer). In the second phase (subroutine AVSN2) the program examines the input orders individually and sorts them to reduce vehicle order selection time. In the third phase (subroutine ROUTE) the four vehicle-type order lists are assembled into vehicle routes. The last phase converts the vehicle route arrays into usable printout (subroutine TCARP). Figure 2 gives the AVS System Flowchart. The combined program, AVS, which consists of AVSI and AVS2, uses 84 kilo digits (KD) of core locations.

INPUT

When procedures specified in the User's Manual (Appendix A) are followed, the data input to AVSI is accomplished interactively from a remote CRT terminal. The user may enter the following data:

Orders. Orders are entered by listing the order sizes and originating and destination warehouses. Entries are made by "filling in the blanks". Data correctness messages are displayed on the CRT screen.

Vehicles. Vehicles are entered by listing the vehicle type, capacity, and maximum route duration. Omission of either capacity or maximum route duration for any vehicle will cause the algorithm to substitute default (built-in) values.

Begin Time. The beginning time for the schedules must be entered. The program uses 24-hour clock time.

Route Length. A maximum value for route duration in minutes is entered. This entry will replace a default value of 480 minutes. This route length does not supersede the value which is specified in the vehicle entry.

SINR's COBOL Driver Routine, AVSINI, checks the input data for validity. An order's origin and destination warehouses must match a built-in list of warehouse names. Input numeric data (e.g., order size, times, vehicle capacity, and route duration) must be within specified ranges. The corrected data are made available to AVS through the VS2IN file. After all input has been entered satisfactorily, AVSINI will execute AVS.
Figure 2 - AVS System Flow Chart
METHOD and ALGORITHM

Route building is accomplished by subroutines ROUTE, BLDTR, and BUILDST. The schedules for ST's are built first, then for TR's, TT's, and finally for IT's. The order of schedule building for the four vehicle types may be changed to fit the needs of the user. The algorithm operates on the sorted lists of orders. In the following discussion the ST routes are built first, then the variations used in TR, TT, and IT route building.

The list of orders is scanned to determine the combination of orders which will, if serviced by a single ST, provide the greatest time savings (or least time cost) over the situation in which each order is serviced by a separate truck. All time estimates pertain to warehouse loads, rather than to individual warehouses. There is almost always a time savings involved in joining two or more orders in this manner.

However, to prevent excessive order joining and over-utilization of individual vehicles, a least time savings restriction was added to the algorithm. Since joined order routes are assigned to vehicles first, the least time savings restriction reduces the number of joined orders and allows the assignment of single order routes to available vehicles. If the time savings requirement for the vehicle is not met, order segments are not joined.

Having selected the best set of orders to start an ST's route, the algorithm examines the remaining orders in the list for that single order which, if joined to the route, results in the least time cost over the remaining order separately. As in the starting case the limit on the number of orders is applied. The new order is placed at the end the existing route, since examining intermediate positions along the route would be too time consuming and the coding would be too complex.

The algorithm continues in this manner, adding orders to the end of the previous route, until the route time limit for the vehicle prescribes further additions, or until the pool of unassigned orders is exhausted. In the latter case the algorithm proceeds directly to consideration of the TR vehicles. In the former case the next ST route is
begun, using the same method. ST routes are built using a "first on, first off" strategy.

TR, TT, and IT building is exactly like ST route building and requires no additional elaboration. Leftover orders from TR route building are passed to the TT's and leftover TT orders are passed to IT's in the same manner that leftover ST orders are passed to TR's. Figure 3 shows beginning pairs that may be serviced by all vehicles. Figures 4 through 6 illustrate order assignment for each vehicle type. Routes for TR's, TT's, and IT's are built using a "first on, last off" scheme.

Since TT's are the only vehicles equipped for highway travel, they alone service the six off-base activities.

Route building ceases when all IT routes have been built. If any orders are still unserviced, they are printed out so that the dispatcher can schedule them at a later time. They may be scheduled later as "emergency" orders using program AVS2, they may be postponed to the next shift, or they may require special scheduling without the use of AVS.

PROGRAM OUTPUT

Schedule output from AVS1 consists of a summary of the input data and the schedules for the individual vehicles. Each schedule gives the vehicle name, capacity, route starting time, and dates in a header; a list of scheduled stops specifying site, time, pallets picked up or delivered, reference order number, and approximate stay time at the site; and a trailer of finishing time and location, time still available, and number of pallets moved. AVS1 also creates a system schedule file, SCHED1, to be used by AVS2.

The coding for schedule printout is quite complex (see Special Techniques section). Changes to this coding should be made only after a thorough study of the programming details covered in Appendixes A and R.
Figure 3 - All Vehicle Types, Order Allocations

Figure 4 - Straddle Order Allocations

Figure 5 - Transporter Order Allocations
Figure 6 - Tractor Trailers/Industrial Tractors Order Allocation
AVS2 - EMERGENCY ORDERS SCHEDULING

The emergency order program AVS2 comprises three phases; data entry (AVSIN1-Appendix C), order scheduling (Subroutine Route), and schedule printout (Subroutine TCARP). These phases are described in the following paragraphs. Figure 2 gives the AVS2 system flowchart as a subset of AVS.

INPUT

AVS2 also uses SINR, and data items are entered by "filling in the blanks" as specified by the selected frames. AVSIN1, a SINR COBOL driver routine, consolidates the data and makes the results available to AVS2. The corrected data are passed to AVS2 on file, VS21N. When data entry is complete, AVSIN1 executes AVS. Route data from the previous set of schedules, whether generated by AVSI or by an earlier run of AVS2, are used by the program along with the current emergency order data as entered at the terminal. The following data are entered interactively:

Orders. The order origin, size, and destination are entered. Origin and destination are checked against a list of warehouse designations. Order size must be a numerical entry in the range 1-99 pallets. Up to 99 orders may be entered for emergency scheduling; if more than 99 are input, only the first 99 are retained.

Vehicles. A considerable choice of vehicles to service the emergency order is given to the user. All vehicles made available when the schedules were created by AVSI (or augmented by earlier AVS2 runs) may be used; any subset of these may be chosen; or new vehicles may be selected. For new vehicles, capacity and maximum route duration may be specified; otherwise, default (built-in) values for capacity and maximum route duration are used. New vehicles chosen by the algorithm to service the order are added to the available vehicles list for the next use of AVS2, but if a vehicle just added by the user during data entry is not selected by the algorithm, it does not join the list.

Time. The time of the emergency order is input. This time is the basis on which the existing schedules are examined to determine vehicle availability. Consequently, sufficient lead time should be allowed to permit the program to execute and the dispatcher to notify the selected driver of his change in route.
**Date.** It is possible during AVS2 to change the date stored in the AVS1 schedules.

**Bump Option.** A "bump" option can be exercised to allow the servicing of emergency orders before regular orders.

**METHOD AND ALGORITHM**

AVS2 examines the existing vehicle schedules and determines which vehicle or vehicles should service an emergency order. There is a fundamental assumption that the vehicles are, in fact, following the computer generated schedules fairly closely. This assumption allows AVS2 to work with schedule data rather than with real time data.

The criterion for determining which vehicle(s) service an emergency order is quite simple, but complexities in the coding arise from a number of options designed to make the algorithm more flexible, and from the rather complicated method of storing the schedule data (linked list technique). According to the criterion, the emergency vehicle selected is the one which can pick up the emergency order the soonest, subject to restrictions imposed by the algorithm options.

The "bump" option affects this criterion. Under the bump option the user may allow vehicles to exceed their allotted maximum route duration. The default case for this option is not to permit this. Therefore, a vehicle which could service the emergency soonest would not be chosen if such an action meant that its regular route would not be finished on time. This action is altered by specifying the bump option during interactive data entry (see above and Appendix A, User's Manual). The term "bump" signifies that delivery of regular orders would be interrupted to handle the emergency. When time is clearly critical, specifying the bump option will enable the emergency order to receive the fastest possible service. If the bump option is not specified, and if the vehicles under consideration all have relatively full schedules, the program may inform the user that no vehicles are available to service the order.

An important feature of AVS2 is that the user may specify the vehicles to be considered, regardless of which vehicles were made available to the AVS1 algorithm. That is, the same vehicles may be used
as were used for the previous schedules; or a subset of those vehicles may be used; or additional vehicles may be specified. This gives the dispatcher considerable control over the manner in which the program schedules the emergency order. As an extreme example, the user could specify a single truck which was already in use, together with the bump option, to force the algorithm to fit the emergency into that vehicle's schedule.

Several points about the AVS2 algorithm need to be mentioned:

First, the emergency order algorithm may be used any number of times during the processing of the AVS schedules. At the conclusion of each emergency run, the schedules are updated for use by the next run.

Second, but related to the first point, when the schedules are being searched for the placing of a new emergency order, no vehicle servicing a previous emergency order is available for the new order until the previous order is delivered. Emergency orders have a single priority, and are filled on a first come, first served basis.

This leads to a third point: emergency orders should be run in the order in which they are placed. Failure to do this may give erroneous results. Also the program will take a few minutes to run and print out, and this should be considered in specifying the start time for an emergency order.

Fourth, the actual updating of the schedules is not done automatically within the AVS2 program; consequently, if the schedules printed at the remote terminal do not satisfy the user, he may change options, vehicles, or even order data and rerun the program. Previous schedules may be saved or discarded by simple file handling commands given in Appendix A. The schedule may be modified using AVS3, Update Program. Detailed flowcharts and coding descriptions are found in the Appendixes.

PROGRAM OUTPUT

Schedule output for AVS2 is straightforward. A summary of the input data is provided, then a vehicle availability table. Any vehicles which cannot be used are listed and the actual vehicle(s) chosen is (are) given. Finally the new schedules for the chosen vehicle(s) are printed, following the same format as in AVS1.
SPECIAL TECHNIQUES

Several techniques used in the AVS programs will be described here to help in understanding the program coding. All the techniques were used to reduce execution time and core requirements so that the programs could run on the rather limited Burroughs B3500 computers in use by Navy Supply Centers. The penalty for the gain in efficiency of the programs is increased program complexity. Three techniques have to do with the calculation of travel times between warehouses; two are general data storage techniques used to reduce sort times in the AVS algorithms.

TRAVEL TIME TECHNIQUES

The AVS programs were set up to service up to 99 warehouses, and the test facility (NSC Charleston) has, in fact, 92 sites. The algorithms make frequent use of the travel times between sites. The times differ for the four vehicle types; giving more than 15,000 intra-activity time measurements. The prohibitive cost of storing such a collection of data demands that this figure be reduced to a more manageable level; it is this problem that the three techniques mentioned address.

The major reduction in the time array sizes is achieved by grouping the warehouse sites; each group of warehouses in close proximity is considered a single site (area). Figure 1 shows the groupings of the Charleston sites. (These groupings reflect some functional as well as geographic differentiation). The travel times between warehouses within an area are taken to be constant (two minutes).

A further reduction in the time array sizes is gained by considering the six off-base sites separately. These sites are serviced only by TT's and all movements take place between the main base and the sites; i.e., there are no movements between off-base locations. The number of measurements necessary to represent travel to the off-base sites is thus reduced to six.

The final reduction in array size is based on an assumption of symmetry in the travel time matrices; i.e., the time to travel from site A to site B is the same as the time to travel from site B to site A. This assumption is justified by actual travel time data collected at Charleston.
Applying these three techniques reduced the arrays from more than 15,000 to 150 storage locations, with only a slight increase in the procedural code generated and with little or no decrease in the accuracy of the schedules.

LINKED LIST TECHNIQUES

The linked list method of data storage is one of several which were tried in various versions of the AVS programs; it is demonstrably faster than the sorting method and uses considerably less core than the duplicate arrays method, both of which are discussed here.

Two sets of data arrays are used in the programs: one set contains information about the orders and the routes to which they belong; the other set contains information about the vehicles used. In both cases the information contained in the arrays is initially stored in a particular sequence. Later the same information is used in a different sequence. For example, the orders generated by AVS1 are stored in the sequence in which they are input at the remote terminal. They are then scanned repeatedly and assembled into the final vehicle schedules.

The problem is how to re-organize the data in these arrays from their initial sequence to their final sequence. The first and most natural method is actual physical re-organization of the data. The advantage of this method is that the final arrays are easy to process, either by computer or the human mind. For example, the orders processed by vehicle #1 would appear first in the final arrays, and would appear in the order in which the vehicle would service them. There are two ways in which this physical re-organization can take place: through the use of duplicate arrays or by sorting the original arrays.

In using duplicate arrays the first set of arrays is examined and the appropriate element is selected, stored in the second set of arrays, and deleted from (or marked as processed in) the original arrays. The obvious disadvantage to this method, particularly when large amounts of data are being processed, is that the memory requirements of the program are doubled.
The second method of physically re-organizing data is by sorting. The initial arrays are examined; the chosen element is selected and physically moved to the first position in the arrays; the remaining items in the arrays are shifted to make room for it. This process eliminates the need for duplicate arrays and their large memory requirements; however, sorting is a time consuming process when the arrays involved are large.

Both these methods for physical re-organization of data were used in early versions of AVS software, but the constraints of time and space made them unacceptable.

A common method of processing large amounts of data is that of embedded links; this technique is used in several of the large data base management systems now commercially available. In this technique a sequence of data items in a large set of data arrays is linked together by providing an array of pointers or links. The pointer associated with a data element gives the address of the next datum in the sequence. A pointer external to the arrays gives the address of the first element in the sequence, the link variable associated with that element gives the address of the second element in the sequence, etc. The time constraints of sorting, where n^2 movements are required to sort n elements, are not encountered. The introduction of an additional array of pointers does not usually involve a significant increase in storage, since the data elements being sorted are usually made up of corresponding components of many parallel arrays (or the addresses of indices in the pointer array are much smaller than the items which they label).

When the initial duplicate array versions of AVS were reprogrammed using sorting techniques, the size of the program decreased by 50 percent and when further reprogramming introduced the linked list techniques, the time of execution for a relatively large test case (99 orders) was decreased to about 25 per cent of its previous value.

The three methods of data restructuring are illustrated in Figures 7, 8, and 9.
Figure 7 - Methods of Data Storage, Duplicate Arrays
### Figure 8 - Methods of Data Storage, Sequential Sort

<table>
<thead>
<tr>
<th>Before Selection</th>
<th>First Select</th>
<th>Second Select</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A^{(1)}$</td>
<td>$A^{(1)}$</td>
<td>$A^{(1)}$</td>
</tr>
<tr>
<td>$A^{(2)}$</td>
<td>$A^{(2)}$</td>
<td>$A^{(2)}$</td>
</tr>
<tr>
<td>$A^{(3)}$</td>
<td>$A^{(3)}$</td>
<td>$A^{(3)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>First Step in Sort</th>
<th>Second Step in Sort</th>
</tr>
</thead>
<tbody>
<tr>
<td>$B^{(k)}$</td>
<td>$B^{(k)}$</td>
</tr>
<tr>
<td>$B^{(1)}$</td>
<td>$B^{(1)}$</td>
</tr>
<tr>
<td>$B^{(2)}$</td>
<td>$B^{(2)}$</td>
</tr>
<tr>
<td>$B^{(3)}$</td>
<td>$B^{(3)}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Array</th>
<th>Array</th>
<th>Array</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C^{(1)}$</td>
<td>$C^{(1)}$</td>
<td>$C^{(1)}$</td>
</tr>
<tr>
<td>$C^{(2)}$</td>
<td>$C^{(2)}$</td>
<td>$C^{(2)}$</td>
</tr>
<tr>
<td>$C^{(3)}$</td>
<td>$C^{(3)}$</td>
<td>$C^{(3)}$</td>
</tr>
</tbody>
</table>
Figure 9 - Methods of Data Storage, Linked Lists
A major advantage to the linked list method of data organization is that it speeds access to the data. For example, rather than searching an entire array for an item which is in a specific vehicle route, only the items in the route need be examined. The data examination process is made more efficient in the AVS case because there is a separate linked list for unprocessed orders, i.e., those orders not yet assigned to a vehicle route. As routes are built, orders pass from the unprocessed linked list to a specific vehicle's linked list. Thus each successive search of the unprocessed orders takes less time.

The savings in space and time of the linked list system must be paid for by increased complexity of the program code.

A separate linked list must be maintained if the arrays are to be searched in reverse order, or if items are to be inserted in a list. Thus two link arrays must usually be specified to determine a linear chain of items. Examination of the coding for AVS, particularly subroutine TCARP, shows how complicated the coding can become using the linked list technique.

DATA PACKING

Because of the limited magnitude of order size and the number of warehouses considered, it was felt that all order information could be placed, "packed", in one data location rather than in three. The array INFO represents all order information, and each entry has the following format:

INFO Word Configuration

<table>
<thead>
<tr>
<th>Order allocation</th>
<th>Origin</th>
<th>Destination</th>
<th>Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>indicator</td>
<td>warehouse</td>
<td>warehouse</td>
<td>size</td>
</tr>
<tr>
<td>+, unassigned</td>
<td>number</td>
<td>number</td>
<td></td>
</tr>
<tr>
<td>- assigned</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1 digit 2 digits 2 digits 2 digits

Data packing also reduces the number of internal sorts by listing one data element rather than three.
The AVS history file is created from the schedules produced by AVS1 and AVS2. The quantities of orders scheduled for delivery are accumulated by originating warehouse, destination warehouse, and truck name. This file is maintained by AVS3.

AVS3 is the set of subroutines used for creating, maintaining, and reporting records in the AVS history file.

File Description

The history file is sequentially organized, with records sorted by START DATE with the earliest date first.

Record Structure

The record for the history file is 1424 bytes long. Each record is uniquely identified by two of the following three fields:

- START DATE
- END DATE
- SHIFT

The Start Date is the earliest date covered by the data in the record. Similarly, the End Date indicates the end of the period of time covered by the dates. When the two dates are unequal, the shift is set to zero and not used. For equal dates, the shift may or may not be zero. A non-zero shift indicates that the data in the record are for a single shift; a zero shift indicates a record with data accumulated over many shifts. Records for a single shift are created from the SCHED1 file produced by AVS1 or AVS2. Before the new record is inserted in the file, a copy of the mark in the SCHED1 file is made for the record, since AVS2 increments the mark by one when it updates the schedule with an emergency order. This enables AVS3 to determine whether the SCHED1 file used is a newer generation than the file which originally created the record.

Record Creation

AVS3 creates new records in the history file using the schedules generated by either AVS1 or AVS2. These schedules are contained in the SCHED1 file. AVS3 first reads the SCHED1 file. It then reads the history file for the same date and shift as the SCHED1 file. If a record
is found with the specified date and shift, the trailer mark on the history file record is compared with the mark in the SCHEDI file. If the mark on the history file record is the same as the mark on the SCHEDI file, the SCHEDI file created the history file schedule record. If the history file trailer mark is less than that in the SCHEDI file, the incoming SCHEDI file is of a more recent generation than the one that created the record. In this case, as well as when a record is found with the specified date and shift, a new record with the date, shift, and trailer mark of the SCHEDI file is created; otherwise, processing is terminated.

Assuming the record can be added to, or replaced by an existing record on the history file, AVS3 then reads through the SCHEDI schedule, accumulating the number of pallets in each scheduled order by originating warehouse, destination warehouse, truck name, and type of order (regular or special, i.e., emergency). When all scheduled orders are exhausted, AVS3 accumulates the number of pallets of each unscheduled order by originating and destination warehouse. The sum of unscheduled pallets is designated as backlog. The record is placed in its proper position in the history file, and a report of the new record is generated.

**Update**

When an update is made with input from the terminal, AVS3 searches the history file for the record with the specified date and shift. If a record is found which either has the specified date and shift, or spans a period of time which includes the specified date, the record is updated. If the specified record is not in the file, a message is printed and the next update record is read. If several update records are read which update the same record, the record is left in memory. The file is then searched for the new record. When the program is terminated, the file is updated and the last record updated is printed.

**Record Merge**

The merge function of AVS3 is used to reduce the number of records in the file by consolidating a number of records into one. Merges are performed on the file by defining the period of time from which the individual records are to be taken. AVS3 reads through the date span. It then continues to read the file, accumulating entries from the subsequent
records in the first record read. When the end of the specified date span is reached, AVS3 updates the Date fields in the original record to indicate the period of time to which the data apply. The file is then copied, with the new record inserted in its proper place. At the conclusion of the run, a report of the new records is generated. Figures 10, 11, and 12 show record entry, consolidation, and update.
Figure 10 - Individual Entry Update

Figure 11 - Record Consolidation
Figure 12 - Update From AVS Schedule
AVS3 uses SINR, and data entry is accomplished by "filling in the blanks" as specified by the selected frame. AVSIN1, a SINR COBOL driver routine, consolidates the data and makes the resulting file, VS3IN, available to AVS3. When data entry is complete, AVSIN1 executes AVS3.

Route data from the previous set of schedules may be modified or updated and placed on a master file. The function number is specified as data and entered interactively: Figure 13 shows one AVS3 frame.

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Update the history file using the schedule produced by AVS1 or AVS2</td>
</tr>
<tr>
<td>2</td>
<td>Update or add specific entries to the history file</td>
</tr>
<tr>
<td>3</td>
<td>Merge two or more records to create an aggregate of the records in the history file, replacing the old records with the new one</td>
</tr>
<tr>
<td>4</td>
<td>Produce a report of either an individual record or an aggregate of several records</td>
</tr>
</tbody>
</table>

AVS2 produces a Summary Report of the history file for the date and shift specified. This report gives cargo transfer statistics by warehouse, indicating the number of pallets delivered, shipped, and backlogged.

For most functions of AVS3, the program will produce two outputs. One will be the history file with the new or updated record. The other output is a report of the record as created or updated. The report includes:

- the date of the updated or created record
- the shift, if applicable
- the warehouse from which the pallets were sent
- the receiving warehouse
**FRAME A07**

(FRM #A07) AUTOMATED VEHICLE SCHEDULING - HISTORY FILE INPUT

**DOCID** | 1 |  
**FRAME** | 2 | ENTER 7 | FUNCTION | 3 | UPDATE FROM AVS SCHEDULE | FOR FUNCTIONS 3 & 4  
**FOR FUNCTION 1** | UPDATE FROM KEYBOARD | MMDDYY  
**DO YOU WANT BACKLOG INCLUDED** | YES/NO  
**IN NEXT AVS RUN - Y OR N** | YES/NO  
**MMDDYY** | START DATE | END DATE | SHIFT 
**FOR FUNCTION 2** | ENTER DATE FOR ALL UPDATES | 5 | (SHIFT) | 6 |  
**FOR FUNCTION 2** | ENTER DATE FOR ALL UPDATES | 5 | (SHIFT) | 6 |  
**FIELD** | (DESCRIPTION) | (ERROR) | (DESCRIPTION)  
**ERROR** |  |  |  
**ERROR** |  |  |  
**ERROR** |  |  |  

Figure 13 - AVS3 Frame Description
- the name of the truck which delivered the pallets
- the number of pallets delivered as regular orders
- the number of pallets delivered as emergency orders

Report of records that have data from only one shift will also include the number of pallets that were not scheduled as deliveries by AVS1 or AVS2. These statistics are arranged in the same way as the others and are organized by sending and receiving warehouses.

Appendix D gives the AVS3 program listing and flowcharts.

Appendix E provides illustrative examples of AVS1, AVS2, and history file entries and output. The printouts are designed to be used directly as dispatch schedules.

ACKNOWLEDGMENT

The authors wish to acknowledge the cooperation and B3500 System expertise of Robert E. Lee and Robert Owens, Code 61 - NSC, Charleston, S.C. Without their assistance AVS could not have been modified for the B3500 computer system.
APPENDIX A - USER'S MANUAL
INFORMATION RETRIEVAL SYSTEM (SINR)

For ease of user data entry, AVS employs the SINR Routines provided by the Fleet Material Support Office as a uniform automatic data processing system for Naval Supply Centers. User instructions and data formatting are displayed on a CRT screen. This display is referred to by SINR as a "frame". AVS frames are given as follows:

<table>
<thead>
<tr>
<th>FRAME #</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>A00</td>
<td>General description of all frames available to AVS</td>
</tr>
<tr>
<td>A01</td>
<td>Order description input</td>
</tr>
<tr>
<td>A02</td>
<td>Vehicle description input</td>
</tr>
<tr>
<td>A03</td>
<td>Input options for regular and emergency orders</td>
</tr>
<tr>
<td>A04</td>
<td>Run execution</td>
</tr>
<tr>
<td>A05</td>
<td>Input options for emergency orders</td>
</tr>
<tr>
<td>A06</td>
<td>Clear and restart, given as an option of A04</td>
</tr>
<tr>
<td>A07</td>
<td>History file update options</td>
</tr>
</tbody>
</table>

Enter 6 digit password and transmit by pressing XMIT key. When "PASSWORD" has been cleared from the screen, enter the following commands:

**ENTER**

*DIH MODE FRAME (transmit)*

System Reply

FRAME MODE ENABLED

**ENTER**

FRM #Name, #A01, #A02, #A03, #A04, #A05, or #A07, of frame desired (transmit)

* System commands are specified by upper case letters
SYSTEM REPLY
The frame specified will appear on the screen.

ENTER
Key in data between displayed [ ]. The skip tab key may be used to position data. When the frame is complete, (transmit).

SYSTEM REPLY
The system will check the correctness of the data entered and will display error or acceptance messages. If errors appear in the data, position the cursor at the beginning of the frame and re-enter the frame. To clear the screen before the next frame request, key "HOME" and "SHIFT" at the same time. When the screen is clear, the user may request the next frame. When all necessary frames have been completed and AVS has been executed, the user may exit the system by:

ENTER
DCH BYE (transmit)

SYSTEM REPLY
PASSWORD

REGULAR ORDER SCHEDULING INSTRUCTIONS
To perform a regular order scheduling run, complete frames A01, A02, A03, and A04.

FRAME A01
Figure 14 shows a typical Frame A01 as it appears on the CRT screen. Enter the following data on A01:

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;1&quot; for this data frame</td>
</tr>
</tbody>
</table>
FIELD DESCRIPTION
3 Order size or number of pallets to be moved from the origin site to the destination site. If the number of pallets is less than 10, enter leading zero
4 Origin site name, up to 6 characters, for a site as given in Table 1
5 Destination site, up to 6 characters, for a site as given in Table 1

Twenty orders may be entered on each A01 frame. If more than 20 orders are needed, fill and transmit A01 as many times as required. The maximum number of orders per run is 99. If the number of orders needed is less than 20, skip forward and position cursor at "LAST ORDER PROCESSED" space and transmit. Wait for the system's reply. If errors appear, re-enter frame and transmit.

FRAME A02

Figure 15 shows Frame A02 as it appears on the CRT screen. Enter the following data on A02:

FIELD DESCRIPTION
1 "AVS" System Doc ID
2 "2" for this frame
3 "X" if this vehicle is to be used in AVS
   "*" if specified industrial tractor is to be an IT. Otherwise, the vehicle will be considered a tractor trailer, TT.
4 Capacity, maximum number of pallets vehicle can carry. Skip field if default value is desired.
   Defaults:
   ST = 7, TR = 12, TT = 14, IT = 10
5 Maximum route time in minutes for this vehicle. Skip if default is desired
   Default = 480 min
Follow instruction given on frame to transmit data.

FRAME A03

Figure 16 shows Frame A03 as it appears on the CRT screen. Enter the following data on A03:

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;3&quot; for the frame</td>
</tr>
<tr>
<td>3</td>
<td>Date of schedule</td>
</tr>
<tr>
<td>4</td>
<td>Start time of shift using 24 hr clock</td>
</tr>
</tbody>
</table>
| 5     | Maximum length of shift  
  
  Default = 480 min |

Position cursor at "CURSOR" and transmit. Wait until system replies before proceeding.

FRAME A04

Figure 17 shows Frame A04 as it appears on the CRT screen. Enter the following data on A04:

1. "AVS" for system Doc ID

2. "4" to produce schedules for data  
  "6" to clear registers and re-enter data for frames A01, A02, A03, A05

Move cursor to "CURSOR" and transmit. No reply will be made by system.

EMERGENCY ORDERS SCHEDULING INSTRUCTIONS

To perform an emergency order scheduling run, complete frames A01, A02, A03, A04, and A05. The emergency order program (AVS2) uses the same information as the regular orders. Delivery preference options are given in Frame A05, Figure 18. The items for frames A01, A02, and A03 which apply only to the emergency orders are described as follows:
### FRAME A01

**AUTOMATED VEHICLE SCHEDULING - AVS - ORDER INPUT**

<table>
<thead>
<tr>
<th>DOCID</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRAME</td>
<td>2</td>
</tr>
<tr>
<td>ENTER ORDERS</td>
<td>SIZE</td>
</tr>
<tr>
<td>ORDERS</td>
<td>1</td>
</tr>
<tr>
<td>ORDERS</td>
<td>3</td>
</tr>
<tr>
<td>ORDERS</td>
<td>5</td>
</tr>
<tr>
<td>ORDERS</td>
<td>7</td>
</tr>
<tr>
<td>ORDERS</td>
<td>9</td>
</tr>
<tr>
<td>ORDERS</td>
<td>11</td>
</tr>
<tr>
<td>ORDERS</td>
<td>13</td>
</tr>
<tr>
<td>ORDERS</td>
<td>15</td>
</tr>
<tr>
<td>ORDERS</td>
<td>17</td>
</tr>
<tr>
<td>ORDERS</td>
<td>19</td>
</tr>
</tbody>
</table>

**LAST ORDER PROCESSED**

<table>
<thead>
<tr>
<th>ERRORS</th>
<th>ORDER</th>
<th>EXPLANATION</th>
<th>ORDER</th>
<th>EXPLANATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERRORS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERRORS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERRORS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERRORS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERRORS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 14 - Frame A01*
FRAME A02

(STRADDLES

<table>
<thead>
<tr>
<th>TYPE</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>12</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
</tr>
</tbody>
</table>

(TRANSPTRS

<table>
<thead>
<tr>
<th>TYPE</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>8</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td>11</td>
<td></td>
<td></td>
<td>12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(INDCTRTRS

<table>
<thead>
<tr>
<th>TYPE</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
<th>USE</th>
<th>CAPAC</th>
<th>TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>2</td>
<td></td>
<td>2</td>
<td></td>
<td></td>
<td>3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>8</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td>11</td>
<td></td>
<td></td>
<td>12</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

POSITION CURSOR HERE] ![BEFORE XMITTING DATA

WAIT FOR OUTPUT BELOW BEFORE PROCEEDING

|     |     |     |     |     |     |     |     |     |
|     |     |     |     |     |     |     |     |     |
|     |     |     |     |     |     |     |     |     |
|     |     |     |     |     |     |     |     |     |

LAST VEHICLE PROCESSED] ![ }

Figure 15 - Frame A02
Figure 14 - Frame A03

FRAME A03

AUTOMATED VEHICLE SCHEDULING - AVS -

| FRAME | 0 | DOCID | 1 | ENTER FRAME | 2 | ENTER DATE | 3 | MMDDYY | 4 | SCHEDULING START TIME | 4 | 24 HR CLOCK | 5 | MINUTES | [CURSOR] |

| ERRORS | ] | NO ERRORS | ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |

Figure 14 - Frame A03
FRAME A04

(FRM =A04 AUTOMATED VEHICLE SCHEDULING - AVS - (START / CLEAR AND RESTART)
DOCID [ 1 ]

(TO GENERATE SCHEDULES ENTER 4)
(TO CLEAR ALL DATA AND RESTART ENTER 6)) [ 2 ]
(CURSOR)[ ]

Figure 17 - Frame A04
**FRAME A01**

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;1&quot; for this frame</td>
</tr>
<tr>
<td>3</td>
<td>Order size, number of pallets</td>
</tr>
<tr>
<td>4</td>
<td>Origin site name</td>
</tr>
<tr>
<td>5</td>
<td>Destination site name</td>
</tr>
</tbody>
</table>

A maximum of 99 emergency orders may be entered per emergency order run.

**FRAME A02**

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;2&quot; for this frame</td>
</tr>
<tr>
<td>3</td>
<td>&quot;S&quot; to use the same vehicles as the previous regular order run. Otherwise, enter &quot;X&quot; or &quot;*&quot; for each vehicle to be used and complete Field 5 for each vehicle to be modified.</td>
</tr>
</tbody>
</table>
| 5     | Time in minutes of maximum route  
        Default = 480 min |

**FRAME A03**

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;3&quot; for this frame</td>
</tr>
<tr>
<td>3</td>
<td>Date</td>
</tr>
<tr>
<td>4</td>
<td>Time of emergency orders using 24-hr clock</td>
</tr>
</tbody>
</table>
| 5     | Maximum length of shift (same as regular orders)  
        Default = 480 min |
FRAME A04

Frame A04 is used as before to execute the emergency orders program.

FRAME A05

Figure 18 shows Frame A05 as it appears on CRT screen. Data to be entered on A05 are as follows:

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;5&quot; for this frame</td>
</tr>
<tr>
<td>3/4</td>
<td>Answer questions on frame</td>
</tr>
</tbody>
</table>

Position cursor at "CURSOR" and transmit. Do not clear screen until MIX and PROPT values are returned.
FRAME A05

EMERGENCY VEHICLE SCHEDULING - EVS - OPTION INPUT

DOCID(1) ENTER AVS
FRAME(2) ENTER 5 FOR THIS FRAME
MAY THE USE OF MORE THAN ONE VEHICLE TYPE
BE ALLOWED TO FILL THIS ORDER [3] ENTER Y FOR YES - N FOR NO

MAY NON-EMERGENCY ORDERS BE BUMPED TO FILL
THIS ORDER [4] ENTER Y FOR YES - N FOR NO
CURSOR [ ]

RESULTS
MIX [ ]
PROPT [ ]

Figure 18 - Frame A05
History File Instructions

To perform updates and report generation of the AVS history file, frame A07 must be completed (See Figure 19). The following four functions may be performed using this frame:

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Update the history file using the schedule produced by AVS1 or AVS2</td>
</tr>
<tr>
<td>2</td>
<td>Update or add specific entries to the history file</td>
</tr>
<tr>
<td>3</td>
<td>Merge two or more records to create an aggregate of the records in the history file, replacing the old records with the new one.</td>
</tr>
<tr>
<td>4</td>
<td>Produce a report of either an individual record or an aggregate of several records</td>
</tr>
</tbody>
</table>

Frame A07 as it is filled out for each of the functions is described as follows:

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>&quot;AVS&quot; System Doc ID</td>
</tr>
<tr>
<td>2</td>
<td>&quot;7&quot; for this frame</td>
</tr>
<tr>
<td>3</td>
<td>Desired function &quot;1&quot;, &quot;2&quot;, &quot;3&quot;, or &quot;4&quot;</td>
</tr>
</tbody>
</table>

Function 1 Instructions

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>&quot;Y&quot; or &quot;N&quot;. If &quot;Y&quot;, a file containing any orders that were not scheduled by AVS1 or AVS2 will be included in the next AVS run (UNIMPLEMENTED)</td>
</tr>
</tbody>
</table>
### Function 2 Instructions

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Date (MMDYY) of the AVS History File record to be updated</td>
</tr>
<tr>
<td>6</td>
<td>Time (24-hour clock) of AVS history file record to be updated</td>
</tr>
<tr>
<td>7</td>
<td>Warehouse from which shipment was sent</td>
</tr>
<tr>
<td>8</td>
<td>Warehouse which received shipment</td>
</tr>
<tr>
<td>9</td>
<td>Number of pallets</td>
</tr>
<tr>
<td>10</td>
<td>Name of vehicle on which the shipment was made</td>
</tr>
<tr>
<td>11</td>
<td>&quot;Y&quot; emergancy order, &quot;N&quot; otherwise</td>
</tr>
<tr>
<td>12</td>
<td>&quot;Y&quot; replace any data in the History File, &quot;N&quot; corresponding entry in the history file is to be incremented by the number in Field 9</td>
</tr>
</tbody>
</table>

### Function 3 and 4 Instructions

<table>
<thead>
<tr>
<th>FIELD</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>Start Date (MMDYY) for the range of history file records to be included in the report or in the merge record</td>
</tr>
<tr>
<td>14</td>
<td>End Date (MMDYY) to indicate the date of the last record to be included in the report or in the merge record. A value equal to the start date (Field 13) indicates all records for that date are to be included.</td>
</tr>
<tr>
<td>15</td>
<td>Shift of record to be in the report. This field is used only for function 4 (report generator) and only when fields 13 and 14 are the same. It is ignored in all other instances.</td>
</tr>
</tbody>
</table>
### Frame A07

**FORM A07**  
**AUTOMATED VEHICLE SCHEDULING  HISTORY FILE INPUT**

**DOCID** | 1 |  
**FRAME** | 2 | (ENTER 7) | **FUNCTION** | 3 | (1. UPDATE FROM AVS SCHEDULE) | **FOR FUNCTIONS 3&4**  
**FOR FUNCTION 1** | (2. UPDATE FROM KEYBOARD) | **MMDDYY**  
**DO YOU WANT BACKLOG INCLUDED** | (3. MERGE RECORDS) | **START DATE** | 13 |  
**IN NEXT AVS RUN-Y OR N** | 4 | **END DATE** | 14 |  
**MMDDYY** | (SHIFT) | 15 |  

**FOR FUNCTION 2** | (ENTER DATE FOR ALL UPDATES) | 5 | **(SHIFT)** | 6 |  
**FIELD** | **DESCRIPTION** | **ERROR** | **DESCRIPTION**  
**ERROR** | | |  
**ERROR** | | |  
**ERROR** | | |  

*Figure 19 - Frame A07*
ROUTINE: AVS2

Description:

AVS2 is an executive routine which executes each segment of the scheduling algorithm.
ENTER AVS2

READ SCHEDULING PROGRAM OPTION AVS1 OR AVS2

IS AVS1 DESIRED? Y 

CALL AVSN2 TO INPUT ORDERS

N

CALL AVSIN TO INPUT SPECIAL ORDER

CALL TCARP TO DETERMINE AVAILABLE VEHICLES Y

BUMP OPTION DESIRED

N

CALL ROUTE TO COMPILATE SCHEDULES

CALL TCARP TO PRINT SCHEDULES

CALL AVSOUT TO CREATE SCHEDULE FILES

STOP
IDENT AVS2
SIZE INTEGER
SEGMENT AVS2, AVSN2, AVSOUT, ROUTE, ALTR, HULOS, STRDST, TCARP, CONV, NAM
FILE 1=CDLD1, UNIT=DISK, BLOCKING=I, RECORD=80, LOCK
FILE 2=CDLD2, UNIT=DISK, BLOCKING=I, RECORD=80, LOCK
FILE 9=VSZN, UNIT=DISK, BLOCKING=I, RECORD=80, LOCK

*******

(XFORIN COMPILER)

AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2
AVS2

INTEGER PTSUK, CAPAC, TRUCK, DATE, PTK, ONUMB
* TRKSAV
ALPHA, WHEAE
REAL LTIME
COMMON
*/GENRPC, RTIMI,MILDO(4)
*/WANTS/ WARE, WAREA(10)
*/SPDRO/ CLEARL, SAVTIN, ISORO
*/SAVPP/ NSAV(4), TRKSAV(4)
*/TRUCKS/ PTSUK(50), TRUCK(50), CAPAC(50), TIME(50), LTIME(50)
*/RFLIM(50), TLCFT(50), HT
*/WHAUSE/ WMAE(62)
*/TIMATW/ TIME(3, 35), TTIME(18)
*/HSCLWS/ NSW(1), TRNRFS(41), PTK(4)
*/OUTF/ SHIFT, LEA
*/LDS/ LASTA, LASTA, NEXTA, ITTRK, IATYPE, IPASS, NNALT, TIME
*/LOAD/ NSAV(5)
IENP=100
10 102, I=1, NEXT=
T=0, KSAT(1)=1

1000 NSAV(N)=0

UNL=1

READ1(5, 300), IAWS
IF (IAWS, 20, 1) CALL AVS2
IF (IAWS, 20, 2) CALL AVSIN
IF (IAWS, 20, 3) CALL TCARP
CALL ROUTE(IAWS)

50 CALL TCARP

CALL AVSOUT(IAWS)

STOP

END
ROUTINE: AVSIN

Description:

AVSIN inputs the special orders to be considered, the vehicles to handle them, and the schedule run options.
SUBROUTINE AVSIN
C******************************************************************************
C AVSIN INPUTS ALL PARAMETERS NECESSARY FOR THE SCHEDULE
C IT ALSO SORTS ORDERS AND VEHICLES TO
C TO REDUCE EXECUTION TIME.
C******************************************************************************
C
INTEGER OSTART(16), ONESTP(16)
INTEGER PTOR, TRUCK, ONUM
INTEGER PTO, DATE, NWARE, NAREA(16)

ALPHA BUMP(2), WNAME, TYPE(4), AST, IORIG, ITERP, ILPHA

INTEGER PROPT, SORCR, TRKSL(4)

REAL LTIME

COMMON

*/GEN/ RTIME, MINLO(4)
*/SAVEP/ NSAEP(4), TRKSAE(4)
*/WNAME/ NWARE, NAREA(16)
*/STOP/ IORIG, SAVTIM, SORCR
*/TRUCKS/ PTO, TRUCK(50), CAPAC(50), STIME(50), LTIME(50)
*/RTLIM(50), LLEFT(50, 1)
*/SHIFT/ CATE
DATA TYPE/(2HST, 2HTT, 2HST)
DATA SAV/(2HNO, 3HTES)
C MSTAT=D
READ 1.1001 (CNUME(I), LINFO(I), IDESTP(I), IENDP(I), LFRWDP(I))

READ (1, 101) (FTSOR(I), TRUCK(I), CAPAC(I), STIME(I), LTIME(I),
*/RTLIM(I), LLEFT(I), I=1,50)
READ (1, 102) (TRUCK(I), NWARE(I), I=1,16)
READ (1, 103) (IAREA(I), I=1,200)
READ (1, 104) (ITIME(I), J=1,50), TTIM2
READ (1, 105) (PTOR, NIKS, NOROR, SOROP
READ (1, 106) SHIF'T, DATE
READ (1, 107) TRK
REWIND 1
DO 2000 I=1, NIKS
INFO(I) = IAMS(I, INFOI)
IF (IESTP(I, 16, 0) .GT. 9999) GO TO 9999
INFO(I) = INFO(I)
9999 CONTINUE
WRITE (6, 600)

IMN = DATE / 10000
IDAY = (DATE - 19000) / 100
ITY = DATE - 19000 * IMN - 100 * IDAY
WRITE (6, 602) INFO, IDAY, ITY, SHIFT
WRITE (6, 600) SAVTIM, SAVTIM
SUBROUTINE AVSIN  7/8/74  OPT=O  ROUND=*  TRACE  FTN 4, 6/68  10/17/80  10:04.17

NORD=0  NORD=1  IF(1STSTART.EQ.0) NSTART=NORD

60  NORD=0  SORD=1  RORD=ABS(SORD)=100
WRITE(*,463) 1, SIZE, IORIG, ITERM
IF(1.E-8.E-9)EQ.01 WRITE(4,A00)

65  INFO(1,NORD)=1 concede MATCH(100), IAI10000+MATCH(ITERM, IAI) * IAI
ONUM=1  NORD=NORD+SCDR
GO TO 2600

C  SORT ORDERS BY AREA

2600  LIMIT=1+31  LIMIT
ISTART=1+1  IF(NORD,EQ.0) GO TO 2600

70  DO 2700 1=1+16  IF(1.E-7.GT.NORD) GO TO 2600

75  IF(IAREA(I,10L,1.GE.IAREA(I)) GO TO 2700

ISAVE(I,AAREA(I))
ISAVE(I,INFO(I))
ISAVE(I,ONUMB(I))
ONUMB(I)=ONUMB(I)

INFO(I)=INFO(I)

INFO(I)=ISAVE(I)
ONUMB(I)=ONUMB(I)

2700 CONTINUE

2600 CONTINUE

C

MARK START OF COMMONS, ORIGIN, DESTINATIONS

90  DO 2400 1=1+16  GEND(I)=0

2400  ISTART=1+1  DO 2300 1=1+16  JARE=IARE(I)

95  IF(JSTART(I) IARE(I)) ISTART(I) IARE(I)

IF(INFO(I),TRN(I) EQ.0) GEND(I)=1

IF(INFO(I),TRN(I) GT.0) GEND(I)=1

2300 CONTINUE

C

SORT BY ORIGIN AND DESTINATION IN SAME AREA

100  DO 2200 1=1+16  IF(1.E-7.GT.ISTART(I)) GO TO 2200

ISTART=1+1
END=0

105  IF(IEND(EQ.1,EQ.1)) GEND(I)=1

IF(1.E-7.GT.ISTART(I)) GEND(I)=1

100  DO 2200 =1+16  JEND=IEND(I)

105  IF(IEND(EQ.1,EQ.1)) GEND(I)=1

110  DO 2275 K=1+16  JEND=IEND(I)

INFO(I)=INFO(I)

ISAVE(I,AAREA(I))
ISAVE(I,FOR(I))
ISAVE(I,ONUMB(I))

IAREA(I)=IAREA(I)

2200 CONTINUE

2275 CONTINUE

56
SUBROUTINE AVSI4

115     INFO(JJ)=INFO(K)
ONUMB(JJ)=ONUMB(K)
J&REK(K)=J&REK(A)
INFO(K)=ISAVE1
ONUMB(K)=ISAVEC

120     2275 CONTINUE
2290 CONTINUE
2200 CONTINUE
WRITE(*,608) SAVTIM
READ(5,501) PIX,PROP
WRITE(*,607) BUMP(PROP+1)
IGATKL=0
IF(IPOPT.LE.0) IGATKL=2
NTSAV=1
NTRKS(I)+NTRKS(2)+NTRKS(1)

130     C
C TRUCK INPUT
READ(5,602) ILFHA
IF(ILFHA.EQ.1.0A1) GO TO 4000
GO TO 2225
ISAV(JJ)=NTRKS(JJ)
TRKSAV(JJ)=PTRK(JJ)
PTRK(JJ)=PTRK(JJ)+NTRKS(JJ)

2225 NTRKS(JJ)=0
ISS=SHIFT
KSHFT=1001*1001*1001
ISS=AVTIM
SORTIM=ISS/1001*1001*1001
SORTIM=SORTIM-KSHFT

225     IF(SORTIM.LE.0) RTMAX=SORTIM
IF(INTP.K.LE.36) INTYPE=2
IF(INTP.K.LE.20) INTYPE=1
NTRKS(INTYPE)=NTRKS(INTYPE)+1
NTSAV=NTSAV+1
INTCFT=INTCFT+1
TRUCK(IJJ)=INTK
IF(INTK.LE.3) TYPE=3
IF(INTK.LE.3) TYPE=2
IF(INTK.LE.2) TYPE=1
IGATKL=1
NTRKS(INTYPE)=NTRKS(INTYPE)+1
END

3000 IF(INTK.LE.0) Capac(INTRK)= FLOAT(INTRK)
IF(INTK.LE.0) RTRK(IJJ)= RTRK(IJJ)+1
IF(INTRK.LE.0) RTRK(IJJ)= RTRK(IJJ)+1
END

105     GO TO 3000
170     3100 NIXE
SUBROUTINE AVSIN  74/74  OPT=0  ROUND=°/  TRACE  FTN 4,4460  10/17/80  10.04.17
  
  C
  
  4400 WRITE(6,6041)
  4410 FORMAT(1X,80,1X)
  4415 IX=0
  4420 DO 4450 ITP=1,4
  4425 IF(ITY=ITYPE(1),1,2) GO TO 4400
  4430 ND=FRK(ITYYPE1)
  4435 NO=4450 II=1,AC
  4440 IF(ITY=ITYPE(1),1,2)
  4445 IX=1
  4450 IF(ITY=ITYPE(1),1,2) TRKSAV(ITYYPE)=TRK(ITYYPE)
  4455 IVE=1-TRKSAV(ITYYPE)
  4460 WRITE(6,6051) I,TYPE(ITY),I,VEH,CAPAC(I),ATLIM(I)
  4465 IF((IX-E0.1*I.E0.0) WRITE(5,6061)
  4470 185 4450 FORMAT(A16.2,F11.11)
  4480 100 FORMAT(A16.2,F11.11)
  4490 150 FORMAT(A16.2,F11.11)
  4495 200 FORMAT(A16.2,F11.11)
  205 CONTINUE
  
  4450 CONTINUE
  
  100 FORMAT(A16.2,F11.11)
  101 FORMAT(A16.2,F11.11)
  102 FORMAT(A16.2,F11.11)
  103 FORMAT(A16.2,F11.11)
  104 FORMAT(A16.2,F11.11)
  105 FORMAT(A16.2,F11.11)
  106 FORMAT(A16.2,F11.11)
  107 FORMAT(A16.2,F11.11)
  108 FORMAT(A16.2,F11.11)
  109 FORMAT(A16.2,F11.11)
  110 FORMAT(A16.2,F11.11)
  111 FORMAT(A16.2,F11.11)
  112 FORMAT(A16.2,F11.11)
  113 FORMAT(A16.2,F11.11)
  114 FORMAT(A16.2,F11.11)
  115 FORMAT(A16.2,F11.11)
  116 FORMAT(A16.2,F11.11)
  117 FORMAT(A16.2,F11.11)
  118 FORMAT(A16.2,F11.11)
  119 FORMAT(A16.2,F11.11)
  120 FORMAT(A16.2,F11.11)
  121 FORMAT(A16.2,F11.11)
  122 FORMAT(A16.2,F11.11)
  123 FORMAT(A16.2,F11.11)
  124 FORMAT(A16.2,F11.11)
  125 FORMAT(A16.2,F11.11)
  126 FORMAT(A16.2,F11.11)
  127 FORMAT(A16.2,F11.11)
  128 FORMAT(A16.2,F11.11)
  129 FORMAT(A16.2,F11.11)
  130 FORMAT(A16.2,F11.11)
  131 FORMAT(A16.2,F11.11)
  132 FORMAT(A16.2,F11.11)
  133 FORMAT(A16.2,F11.11)
  134 FORMAT(A16.2,F11.11)
  135 FORMAT(A16.2,F11.11)
  136 FORMAT(A16.2,F11.11)
  137 FORMAT(A16.2,F11.11)
  138 FORMAT(A16.2,F11.11)
  139 FORMAT(A16.2,F11.11)
  140 FORMAT(A16.2,F11.11)
  141 FORMAT(A16.2,F11.11)
  142 FORMAT(A16.2,F11.11)
  143 FORMAT(A16.2,F11.11)
  144 FORMAT(A16.2,F11.11)
  145 FORMAT(A16.2,F11.11)
  146 FORMAT(A16.2,F11.11)
  147 FORMAT(A16.2,F11.11)
  148 FORMAT(A16.2,F11.11)
  149 FORMAT(A16.2,F11.11)
  150 FORMAT(A16.2,F11.11)
  151 FORMAT(A16.2,F11.11)
  152 FORMAT(A16.2,F11.11)
  153 FORMAT(A16.2,F11.11)
  154 FORMAT(A16.2,F11.11)
  155 FORMAT(A16.2,F11.11)
  156 FORMAT(A16.2,F11.11)
  157 FORMAT(A16.2,F11.11)
  158 FORMAT(A16.2,F11.11)
  159 FORMAT(A16.2,F11.11)
  160 FORMAT(A16.2,F11.11)
  161 FORMAT(A16.2,F11.11)
  162 FORMAT(A16.2,F11.11)
  163 FORMAT(A16.2,F11.11)
  164 FORMAT(A16.2,F11.11)
  165 FORMAT(A16.2,F11.11)
  166 FORMAT(A16.2,F11.11)
  167 FORMAT(A16.2,F11.11)
  168 FORMAT(A16.2,F11.11)
  169 FORMAT(A16.2,F11.11)
  170 FORMAT(A16.2,F11.11)
  171 FORMAT(A16.2,F11.11)
  172 FORMAT(A16.2,F11.11)
  173 FORMAT(A16.2,F11.11)
  174 FORMAT(A16.2,F11.11)
  175 FORMAT(A16.2,F11.11)
  176 FORMAT(A16.2,F11.11)
  177 FORMAT(A16.2,F11.11)
  178 FORMAT(A16.2,F11.11)
  179 FORMAT(A16.2,F11.11)
  180 FORMAT(A16.2,F11.11)
  181 FORMAT(A16.2,F11.11)
  182 FORMAT(A16.2,F11.11)
  183 FORMAT(A16.2,F11.11)
  184 FORMAT(A16.2,F11.11)
  185 FORMAT(A16.2,F11.11)
  186 FORMAT(A16.2,F11.11)
  187 FORMAT(A16.2,F11.11)
  188 FORMAT(A16.2,F11.11)
  189 FORMAT(A16.2,F11.11)
  190 FORMAT(A16.2,F11.11)
  191 FORMAT(A16.2,F11.11)
  192 FORMAT(A16.2,F11.11)
  193 FORMAT(A16.2,F11.11)
  194 FORMAT(A16.2,F11.11)
  195 FORMAT(A16.2,F11.11)
  196 FORMAT(A16.2,F11.11)
  197 FORMAT(A16.2,F11.11)
  198 FORMAT(A16.2,F11.11)
  199 FORMAT(A16.2,F11.11)
  200 FORMAT(A16.2,F11.11)
  201 FORMAT(A16.2,F11.11)
  202 FORMAT(A16.2,F11.11)
  203 FORMAT(A16.2,F11.11)
  204 FORMAT(A16.2,F11.11)
  205 FORMAT(A16.2,F11.11)
  206 FORMAT(A16.2,F11.11)
  207 FORMAT(A16.2,F11.11)
  208 FORMAT(A16.2,F11.11)
  209 FORMAT(A16.2,F11.11)
  
  210 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  211 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  212 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  213 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  214 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  215 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  216 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  217 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  218 FORMAT(1X,14HMENU OPTION = ,A6,1/)  
  
  END
ROUTINE: AVSN2

Description:

AVSN2 inputs the regular orders to be considered, the vehicles to handle them, and the date and shift of the schedules to be produced. AVSN2 also sorts orders by areas and origin and destination warehouses to reduce execution time. Order information is packed to reduced storage.
SUBROUTINE AVSN2

C*************************************************************************
C AVSN2: INPUTS ALL PARAMETERS NECESSARY FOR THE SCHEDULE
C COMPUTATIONS. IT ALSO SORTS ORDERS AND VEHICLES TO
C TO REDUCE EXECUTION TIME.
C*************************************************************************

INTEGER OSTART(16), OEND(16)
INTEGER CAPAC, TRUCK, ONUM, PTFK, PTOR

* DATE
AVSN2 1

ALPHA WSNAM, AST, ALPHA, IDIG, ITEM
REAL TIME

*/GEN
AVSN2 2

*/RTTIM, MINDL(4)
*/MIN/LN MARE(16)
*/SCHOOL, OMNUM(1200), INFO(200), IOSTP(1200), IAREA(200), LFRAME(200),
*/LON(200), LYEM(200), LTME(200), STIME(200), STOP(200)
AVSN2 3

*/TRUCKS/ PTSON(50), TRUCK(150), CAPAC(50), STIME(50), LTIME(50),
*/RTIME(50), ILEFTED, IN
*/MNS/ WNAME(4)
*/TIMAB, TIME(S, S), TIMB(6)
*/MSCLS/ NOKOR, NTRK(4), PTRK(4)
*/INOPT/ SHIFT, DATE
*/SPRND/ IGMAI, SAMT, ISORD
DATA TYPE/ 2HTR, 2HTT, 214T/
WRITE(6, 600)
READ(5, 5C21

AVSN2 6

IMO = DATE / 1000
IMY = (DATE - 1000 * IMO) / 100
IK = DATE - 1000 * IMO - 100 * IDAY
READ(5, 5C21
WRITE(6, 600)

AVSN2 9

ISTART = IMO
WRITE(6, 600)

AVSN2 11

2000 READ(5, 5C21
WRITE(6, 600)

AVSN2 12

NORD = 0
2000 READ(5, 5C21
WRITE(6, 600)

AVSN2 13

IF(SIZE(I) .EQ. 0) GO TO 2000
NORD = NORD + 1
I = NORD
IF(1 + (I/1000) .LT. 0) WRITE(I, 1000)

AVSN2 14

IF(IFB(I) .EQ. 1) WRITE(I, 1000)

AVSN2 15

INFORMORD = SIZE(MATCH(I, IAI) + 1000) * MATCH(I, IAI) * 100
AVSN2 16

IF(INFORMORD = 1) GO TO 2000
AVSN2 17

AVSN2 18

AVSN2 19

AVSN2 20

AVSN2 21

AVSN2 22

AVSN2 23

AVSN2 24

AVSN2 25

AVSN2 26

AVSN2 27

AVSN2 28

AVSN2 29

AVSN2 30

AVSN2 31

AVSN2 32

AVSN2 33

AVSN2 34

AVSN2 35

AVSN2 36

AVSN2 37

AVSN2 38

AVSN2 39

AVSN2 40

AVSN2 41

AVSN2 42

AVSN2 43

AVSN2 44

AVSN2 45

AVSN2 46
SUBROUTINE AVSN2

IASAVE=ONUMB(IJ)
ONUMB(IJ)=ONUMB(IJ)
IAREA(IJ)=IAREA(IJ)
INFO(IJ)=INFO(IJ)
ONUMB(IJ)=IASAVE
IAREA(IJ)=IASAVE
INFO(IJ)=IASAVE

2780 CONTINUE
2680 CONTINUE

C MARK START OF COMMONS, ORIGIN, DESTINATIONS
C
DO 2400 K=1,NCR
J=START(J)
J=J+1
IF(Mat(J).LE.0) GO TO 2300
J=J-1
GO TO 2300
IF(Mat(J).LE.0) GO TO 2200
J=J+1
GO TO 2200
IF(Mat(J).LE.0) GO TO 2100
J=J+1
GO TO 2100

2780 CONTINUE
2680 CONTINUE

C SORT BY ORIGIN AND DESTINATION IN SAME AREA
DO 2200 J=1,NCR
IF(Mat(J,LE.0)) GO TO 2200
START=Mat(J)
I=J
INFO(J)=INFO(J-1)
ONUMB(I)=ONUMB(I)
IAREA(I)=IAREA(I)
INFO(I)=INFO(I)
ONUMB(J)=ONUMB(J)
IAREA(J)=IAREA(J)
INFO(J)=INFO(J)
ONUMB(K)=ONUMB(K)
IAREA(K)=IAREA(K)
INFO(K)=INFO(K)
ONUMB(KI)=ONUMB(KI)
2275 CONTINUE
2256 CONTINUE
2200 CONTINUE
C TRUCK INPUT
READ(5,504) ILFMA,(INTR(J),I=1,3),TRMK
TRMK=RTMK
IF(KP(N),NE.0) RUPM=FLOAT(2*TRMK)
I=0
ITCM=0
3000 READ(5,505) AST,INTRM,INTIM,INCAP
IF(INCAP) GO TO 3005
ITCM=ITCM+1
INTRM=ITRM+1
AVSN2 59
AVSN2 60
AVSN2 61
AVSN2 62
AVSN2 63
AVSN2 64
AVSN2 65
AVSN2 66
AVSN2 67
AVSN2 68
AVSN2 69
AVSN2 70
AVSN2 71
AVSN2 72
AVSN2 73
AVSN2 74
AVSN2 75
AVSN2 76
AVSN2 77
AVSN2 78
AVSN2 79
AVSN2 80
AVSN2 81
AVSN2 82
AVSN2 83
AVSN2 84
AVSN2 85
AVSN2 86
AVSN2 87
AVSN2 88
AVSN2 89
AVSN2 90
AVSN2 91
AVSN2 92
AVSN2 93
AVSN2 94
AVSN2 95
AVSN2 96
AVSN2 97
AVSN2 98
AVSN2 99
AVSN2 100
AVSN2 101
AVSN2 102
AVSN2 103
AVSN2 104
AVSN2 105
AVSN2 106
AVSN2 107
AVSN2 108
AVSN2 109
AVSN2 110
AVSN2 111
AVSN2 112
AVSN2 113
AVSN2 114
AVSN2 115

62
SUBROUTINE AVSN2  74/76  CPT=8 ROUND=+1 TRACE  FN 4, 6, 640 10/17/60 10.04.12

140 3005 IF(INTK(1).EQ.0.CH.INTK,GT.50) GO TO 3101
         1=1
          TRUCK(1) = INTK
          IF(INTK(1).NE.0) RTLIM(INTK) = FLOAT(INTK)
          IF(INTK(1).EQ.0) RTLIM(INTK) = NMAX

120    LEFT(1) = INTK(INTK)
          IF (INCAP(NE.Q) CAPAC(INTK) = INCAP
           GO TO 3100

130    3100 NT5=1

125    C SORT TRUCK AND CALCULATE NRK5S

140    3200 NT5(J) = 1.+J
          DU 3200 J = 1.+NT
          IF(TRUCK(J).LE.3200 GO TO 3300

130    3300 NRK5S(J) = NRK5S(J) + 1
           GO TO 3500

140    3400 NRK5S(J) = NRK5S(J) + 1
           GO TO 3500

130    3500 NRK5S(J3) = NRK5S(J3) + 1

145    3600 CONTINUE

C

150    WRITE(6,605)
          DU = DU , J = 1.+NT
          WRITE(JX)
          IYPE=1
          IF(LEFT(J).GT.PTRK(J)) IYPE=2
          IF(LEFT(J).LE.PTRK(J)) IYPE=3
          IF(LEFT(J).LE.PTRK(J)) IYPE=4
          IF(LEFT(J).GE.PTRK(J)) IYPE=5

155    WRITE(6,605) JX,IYPE,IYPE,TRUCK(J),CAPAC(J),RTLIM(J)

400    CONTINUE

C

500    FORMAT(16)

160    501    FORMAT(F5,1)

155    502    FORMAT(18,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)

145    504    FORMAT(18,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)

165    505    FORMAT(18,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)

600    FORMAT(18,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)

600    FORMAT(18,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1)

600    END
**ROUTINE: AVSOUT**

**ARGUMENT:**

L - Disk unit number of SCHED file

1, SCHED1
2, SCHED2

**Description:**

AVSOUT creates SCHED1 or SCHED2 from the order lists and other variable values at the completion of each run.
SUBROUTINE AVSOUT

C

C*************************************************************************
C AVSOUT CREATES UPDATE FILES TO BE USED BY NEXT AVS? RUN
C*************************************************************************

C

INTEGER PSOR,CAPAC,TRK5,SAVE,PTTRK,DATA

ALPHA WHAM

DISPLAY

RETURN

END

10  FORMAT(18/18/S/18/A)

20  FORMAT(18/18/S/18/A)

30  FORMAT(18/18/S/18/A)

40  FORMAT(18/18/S/18/A)

50  FORMAT(18/18/S/18/A)
ROUTINE: BLDTR

ARGUMENTS:

IFILL - Number of order parts assigned to the schedule segment
       (Sign of IFILL indicates direction of delivery)
IRCAP - Remaining vehicle capacity given in pallets
LSTORD - List number of last order added to route segment

Description:
BLDTR builds schedule segments considering space available on the
vehicle and the sequence of order delivery for specified origin and
destination areas. Orders selected are set negative and linked to the
vehicle list. IFILL is decremented and IRCAP is reduced by the size of
the order loaded. BLDTR assigns orders to the vehicle in a "first on,
last off" manner.
ENTER BLOCK

DETERMINE PRESENT WAREHOUSE

DETERMINE ORIGIN & DESTINATION AREAS

DO UNTIL ORDER

IS ORDER ASSIGNED?

Y

SAVE ORDER IN SEG ARRAY NSEG: NSG: GT

CONTINUE

ARE THERE ANY ORDERS TO BE CONSIDERED?

N

RETURN

A
DO 20 I = 1, NSEG

DETERMINE ORIG & DEST & SIZE OF ORDER

IS ORIG PRESENT WAREHOUSE ?

Y

ADD ORDER TO VEHICLE SCHEDULE

COMPUTE REMAIN TIME & SPACE AVAILABLE

IS SPACE OR TIME ELAPSED ?

Y

UPDATE WAREHOUSE & AREA DATA

N

CONTINUE

RETURN

H
DO 30 I = 1, NSEG

HAS ORDER BEEN ASSIGNED?

Y

ADD ORDER TO VEHICLE SCHEDULE

COMPUTE SPACE & TIME LEFT

IS THERE SPACE & TIME LEFT?

Y

CONTINUE

N

UPDATE WAREHOUSE & AREA

RETURN
SUBROUTINE RLOT

ORDER=SKIP(1)
ITSUM=MOD(INFC(IORD)+101)
G) TO 195
101 DJ 175 mg
END=SKIP(1)
IT=MOD(INFC(IORD)+100,101)
DJ 175 IF=1,NORD
IF(IORD+2,13) GO TO 191
IT=MOD(INFC(IORD)+2,100,101)
IF(INFC(IORD)+100,14,11) GO TO 185
IF(MOD(IARE(1),125)+N6,4,EXT) GO TO 180
IF(IITSAVE.EQ.1) GO TO 195

70 IF(IITSAVE.EQ.11) GO TO 175
185 ISUM=0
DJ 175 N=N+1
IF(INC(IORD)+100,12,1) TO 185
IF(JARE(1).NE.JARE(1)) TO 181
ISUM=(SUM+MOD(INFC(1),103))
181 CONTINUE
IF(ISUMLT.MIN=1(IITYPE)) GO TO 180
IF(EXIT)
IF(MOD(INFC(2,23,3)) IPASS+PASS-1
NIGHT
186 IPASS=IPASS-1
IF(MOD,100,13) GO TO 175
100 MOD=MOD(100,13)
DJ 175 MOD=MOD(100,13)
IF(MOD(IORD)+100,13,13,11) GO TO 185
IF(ISUM=ITSUM-MOD(INFC(IORD),103))
SOAP=EWGC1=0
GO TO 195
190 CONTINUE

195 CONTINUE
102 IF(SUMGE.MINI1(IITYPE)) GO TO 123
IF(N-XTIA=LAST1 RETURN
IF(INC(1,14,4)) N=14
103 CONTINUE
110 DO ESERV, LAST1 WHERE ORIGIN LAST1
J=4-4
LT=2
IF(M*1.E22,11) 117,150
1*10 +51
100 40,111,M16
111 IF(INC(IORD)+1,11) GO TO 330
104=100,1
114=111
103=102,1
114=114
103=102,1
J=10,12,21,1
IF(MOD(INFC(IORD)+1,100,12)+MOD(INFC(IORD)+1,100,12)) GO TO 857
103=102,1
114=114
103=102,1
114=114
100 CONTINUE

100 CONTINUE
110 DO 124011=1,3
1000 SPLIT(l=10)
110 1100 TIC+=M62
ROUTINE: BUILDS

ARGUMENTS:

IFILL - Number of order parts assigned to schedule segment
IRCRP - Remaining vehicle capacity given in pallets
LSTORD - List number of last order added to route segment

Description:

BUILDS builds schedule segments considering space available on the vehicle and the sequence of order delivery for specified origin and destination areas. Selected orders are set negative and linked to the vehicle list. IFILL is incremented and IRCAP is reduced by the size of the order loaded. BUILDS assigns orders to a vehicle in a "first on, first off" manner.
ENTER BUILDS

DETERMINE PRESENT WAREHOUSE

DETERMINE ORIGIN & DESTINATION AREAS

DO 10 I = 1, NORDR

IS ORDER ASSIGNED?

Y

SAVE ORDER IN SEG ARRAY NSEG NSEG1

N

CONTINUE

ARE THERE ANY ORDERS TO BE CONSIDERED?

Y

A

N

RETURN
DO 20 I=1,NSEG
A

DETERMINE ORIG & DEST & SIZE OF ORDER

IS ORIG PRESENT WAREHOUSE?

Y

ADD ORDER TO VEHICLE SCHEDULE

COMPUTE REMAINING TIME & SPACE AVAILABLE

IS SPACE OR TIME ELAPSED?

Y

UPDATE WAREHOUSE & AREA DATA

RETURN

CONTINUE

B
DO 30 I=1, NSEG

HAS ORDER BEEN ASSIGNED?

Y

ADD ORDER TO VEHICLE SCHEDULE

COMPUTE SPACE & TIME LEFT

IS THERE SPACE & TIME LEFT?

N

UPDATE WAREHOUSE & AREA

Y

CONTINUE

RETURN
SUBROUTINE NULG15 (P, M, IORD, IAC, IS, IE, N, L)

IF (PASS .LT. 2) NELC = MOD (INFO (IORD) + 100) / 100

61 LIMIT = NSEG - 1
NELC = 0
PS = LIMIT
IF (PS .EQ. LIMIT) RETURN
IS = 1
NO = 1
J = JSTART + NSIG
TOWN = CPU
JORD = SORDER (J)
IF (MOD (INFO (IORD) / 100, 111) .LE. MOD (INFO (IORD) / 100, 111) + 49) RETURN
SORDER (J) = JORD
RETURN

72 NSEG = 40
CONTINUE
78 CONTINUE
79 CONTINUE
81 LIMIT = NSIG + MOD (INFO (IORD) / 100, 111)
IF (MOD (INFO (IORD) / 100, 111) + 49 .EQ. LIMIT) RETURN
IF (ISEM .LT. 8) RETURN
85 ISEG = 65
CONTINUE
90 IS = 0
L = 0
J = 0
JORD = 0
GO TO 50
95 JORD = JORD + MOD (INFO (IORD) / 100, 111)
IF (ISEM .EQ. 8) RETURN
CONTINUE
100 CONTINUE
101 IF (ISSAVE .LT. 5) GO TO 60
CONTINUE
110 IF (ISSAVE .EQ. 1) GO TO 150
CONTINUE

83
SUBROUTINE BULITS 7/74  OPT=0  NOJMN */ TRACE  FID 4,64441  1Z17/80  09.07.85

330 INFO(IOORD)=INFO(IOORD)
IDENT=IOORD=1
LST=0=IOORD
IF (IOORD.EQ.ITRUCK).LE.0) IY504(ITRUCK)*IOORD
LST=FS04(ITRUCK)
IF (LST.EQ.LSTART) 1
335
LST=LSTART(LSTART)
410 IF (LINK.EQ.0) GO TO 400
ISAM=LINK
LST=LSTART(LSTART)
GO TO 195
400 IF (LST(EQ.LORAD) GO TO 423
IF (LST+3(EQ.LORAD) GO TO 423
IF (LST+5(EQ.LORAD) GO TO 421
GO TO 200
421 LASTF=LST
LST=FIRST
200 CONTINUE
250 CONTINUE:
NOPE=0
DD 500 ISPLIT=1,MSPLIT
IF (ISPLIT=1) LINK=1
IF (LINK.LE.1) GO TO 530
LST=0=IOORD
GO 500 111,MIGG
IY504(IIORD)=I
IF (IOORD.EQ.ITRUCK) GO TO 500
IF (IOORD.EQ.ITRUCK) GO TO 500
GO 530
5 ENPINE PALLTS
463 IF (NEXTAN=LASTA) GO TO 505
IF (WIDTH.EQ.0) WARE=0
IF (WIDTH.EQ.0) WARE=0
IF (WIDTH.EQ.0) WARE=0
505 IY504(WAVE=INFO(IOORD),IOORD)
GO TO 4201,500,522,531,532
501 IF (ISPLTE.I=INCAP) GO TO 502
GO TO 502
502 IF (ISPLT=*X=6) INCAP GO TO 503
GO TO 502
503 IF (ISPLT=*X=0) INCAP GO TO 502
IF (NEXTAN=LASTA) GO TO 503
IF (NEXTAN=LASTA) GO TO 503
IF (NEXTAN=LASTA) GO TO 503
505 ISAVE=ISPLT=INCAP
600 FORM NEW INFO ORDER ELEMENTS
INFO(INFO(IOORD)+100)*100=ISAVE
INFO(INFO(IOORD)+100)*100=ISAVE
INFO(INFO(IOORD)+100)*100=ISAVE
INFO(INFO(IOORD)+100)*100=ISAVE
INFO(INFO(IOORD)+100)*100=ISAVE
0UER(IOORD)=UER(INORD)
0UER(IOORD)=UER(INORD)
0UER(IOORD)=UER(INORD)
0UER(IOORD)=UER(INORD)
0UER(IOORD)=UER(INORD)
NPALTS=NPALTS+INCAP
5 COMPUTE TIME REMAINING FOR VEHICLE
TLEFT(ITRUCK)=TLEFT(ITRUCK)+TIME+2.*ELTIME(ITRUCK)*FLOAT(INORD)
L=STIM(ITRUCK))
INCAP=0
285
C  LINK SEGMENT TO SCHEDULE
LSTEND=ORD
IF (PTSCV(ITRUCK)=1.E0) PTSCV(ITRUCK)=NORM
LSTART=PTSCV(ITRUCK)
ISEAVE=LAST
LINK=FRM0D(LAST)
625 IF (LINK.LE.0) GO TO 650
I=IF=LINK
LINK=FRM0D(LINK)
625 GO TO 675
650 IF (ISAVE.EQ.NORM) GO TO 660
IF (ISAVE.EQ.NORM) ISAVE=LAST
IF (ISAVE.EQ.NORM) ISAVE=LAST
IF (ISAVE.EQ.NORM) ISAVE=LAST
660 IF (IFILL.EQ.1)
I=IF=IFILL
IF (IFILL.EQ.1) GO TO 671
TLFF=TLFF(I TRUCK)=LEFT(ITRUCK)*TIME
IF (ISAVE.EQ.0) ISAVE=IORD
IF (ISAVE.EQ.0) ISAVE=IORD
IF (IFILL.EQ.0) GO TO 500
RETURN
670 IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
675 IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
IF (INERTIA.NE.LASTA) GO TO 676
680 IF (INERTIA.NE.LASTA) GO TO 676
LSTEND=ORD
IF (PTSCV(ITRUCK)=1.E0) PTSCV(ITRUCK)=NORM
LSTART=PTSCV(ITRUCK)
ISEAVE=LAST
LINK=FRM0D(LAST)
710 IF (LINK.LE.0) GO TO 700
I=IF=LINK
LINK=FRM0D(LINK)
GO TO 710
700 IF (SAVE) LSTEN(D) GO TO 720
LINK=IF=LAST
GO TO 720
720 IF (IFILL.EQ.0) GO TO 721
GO TO 720
721 LAST=NEXTA
LAST=IORD
RETURN
500 CONTINUE
5000 CONTINUE
IF (LAST=IORD) RETURN
LAST=NEXTA
LAST=IORD
452  CONTINUE
ROUTINE: MATCH

ARGUMENTS:

NAME - Alphanumeric warehouse name
IAREA - Area indicator of "NAME" warehouse

Description:

MATCH searches the warehouse name list for a match for NAME and returns its warehouse and area numbers.
FUNCTION MATCH(NAME,IAREA)
C FUNCTION MATCH FINDS THE POSITION OF NAME IN THE ARRAY
C WMRA
C
C *** COMMONS ***
C ALPHA WMRA,NAPE
C COMMON/WMRA(6)/ WMRA,NAPE(6)
C COMMON/WMRA(6)/ WMRA(4)
C AREA=0
C MATCH = 0
C I = 1000 I = 1, WMRA
C IF(NAME.EQ.WMRA(I)) GO TO 2000
C 1000 CONTINUE
C RETURN
C 2000 MATCH=I
C I=I
C ICH$=0
C DO 500 I=1,16
C ICH$=ICH$+WMRA(I)
C IF(I,LL,IC$) GO TO 4000
C 500 CONTINUE
C RETURN
C 4000 IAREA=J
C RETURN
C END
ROUTINE: ROUTE

ARGUMENT: IAVS - Run option indicator
1, AVS1
2, AVS2

Description:

ROUTE applies the algorithm used in assigning orders to vehicles. It determines the next area of delivery for each available vehicle by searching the available orders, noting transfer and cargo movement times. The next area to be encountered by a vehicle is determined by least travel time for which a minimum quantity of cargo is to be moved. Other restrictions, such as the accessibility of the area to the vehicle and the order of cargo delivery, are also imposed on the selection of the next area. Each area selected may be an order origin area, destination area, or both. Once origin and destination areas are known, ROUTE calls BLDTR (first on, last off) or BUILDS (first on, last off) to assign orders to the vehicle. The current area is updated and the next area selection process is repeated until the vehicle is out of time or the quantity of unassigned cargo does not meet delivery requirements.
DO J NT

Determine next vehicle to be considered

Determine last warehouse & area

DO I-1 NORDR

NOW THIS ORDER BEEN ASSIGNED

IS ORIGIN AREA LAST AREA

Determine delivery area

IS THIS DEL AREA THE NEAREST SO FAR?

A

B
AUTOMATIC VEHICLE SCHEDULING (AVS) PROGRAMMER'S INSTRUCTION MAN--ETC(U)
A

DOES AREA HAVE SUFFICIENT CARGO?

SAVE DELIVERY AREA & TRANSFER TIME

B

WAS A DELIVERY AREA FOUND?

DO i-1, NORDR

HAS ORDER BEEN ASSIGNED?

DETERMINE ORIGIN & DESTINATION

IS ORIGIN AREA NEAREST SO FAR?

IS DEST AREA NEAREST SO FAR?

C

E

D

93
D

SUFFICIENT CARGO

Y
SAVE ORIGIN AREA
SAVE DEST AREA

E

HAS ORIGIN AREA BEEN FOUND

N
I

Y
COMPUTE TIME TO GET TO ORIGIN AREA

G

SUFFICIENT TIME LEFT TO HANDLE AREA

N
I

Y
LASTA=ORIGIN
NEXTA=DESTINATION

H
H

IS THIS A ST?
Y N

CALL BLOTR

IS VOLUME ALL USED IRCAP=0
Y N

RESET LASTA AREA

SUFFICIENT TIME LEFT
Y N

I

RETURN
SUBROUTINE OUTF(7,4/7,4/0, ROUTE, TRACE)

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(10,10)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

# GEN/ RTTIM,MNLO(4)
# SOHOLY/ NWHNAR, NAREA(16)
# SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
# TFKR/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
# TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

*/GEN/

INTEGER LSTSVE(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES

INTEGER LSTSV(16,16)
INTEGER PTKR,TRUCK, DATE
INTEGER TRUCK(4)
ALPHA WHNAM
REAL LTIME
COMMON

*/GEN/ RTTIM,MNLO(4)
*/SCHOOL/ NWHNAR, NAREA(16)
*/SCHOOL/ ONUMA(2), INF(2), DEST(2), ARS(2), TRUCK(4)
*/TRUCK/ TRUCK(16), TRUCK(16), LTRK(12), ATIME(2), STOPF(2)
*/TRUCK/ TRUCK(16), TRUCK(16), CAPA(5), STIME(5), LTIME(5)

* RTEI(M3), TLF(M3), DT

C ROUTE ASSIGNS ORDERS TO AVAILABLE VEHICLES
SUBROUTINE ROUTE
7/2/74 OPT=0 ROUND=/* TRACEx FTE 4.646f 12/12/80 99 G7.53

ITYPE=ITYPE4(IIGN)
IF(ITYPE.GT.2.AND.IDEPT.GE.NE.10) GO TO 100
50

IF(ITYPE.EQ.1) GO TO 100
DO AC JHC=1,16
DO AC MM=1,16
50

LSTMJ(IB,MM)=C
GO TO 200
LSTMJ(IB,MM+5)=C
100

ITYPE=ITYPE4(IIGN)
ITPUCK=ITRKS(IIGN)
LOCFF=C
INH)=1.16
00

INH)=1.16
D00

LINK=3
IF(NTRK.GT.0) GO TO 70
70

ITCAP=ICAP4(IIGN)
C DETERMINE LAST WAREHOUSE OF VEHICLE
LAST=21
LAST=2
IF(IGNTH=NE.10) GO TO 100
70

IF(MOPLT.EQ.1) GO TO 100
100

IF(IGNTH=ITRKS(IIGN) .GT. 100)
LINK=3
LAST=MOD(IFLAG(INFJ(LINK),11))
LAST=MOD(IFLAG(INFJ(LINK)/128,10))
LAST=MOD(IFLAG(INFJ(LINK)/128,10))
125

LINK=LSK3(LINK)
IF(LINKS.LE.0) GO TO 150
50

IF(IGNTH=NE.10) GO TO 150
100

IF(MOPLT.EQ.1) GO TO 150
100

LINK=LSK3(LINK)
LAST=MOD(IFLAG(INFJ(LINK),100))
LAST=MOD(IFLAG(INFJ(LINK)/100,100))
150

LINK=LSK3(LINK)
GO TO 165
155

IF(LEFT4(IIGN)-T,LASTA,LP4,ITYPE,LE реакци GO TO 150
TLEFT(IIGN)=THE
90

LAST=21
LAST=21
GO TO 165
95

C DETERMINE NEAREST WAREHOUSE WITH UNFILLED ORDER
C ORIGIN AREA = LASTA
C CHECK IF ANY ORDER ORIGENATES IN LAST AREA
95

IPASS=1 BUILD NON-DEST END ORDER
C IPASS = ? BUILD DEADEY AREAS
C
100

IPASS=IPASS
NOFLC=G
IF(IOLOAD.LE.0) GO TO 300
300

DO 169 KK=1,16
DO 169 MM=1,16
169

LSTMJ(KK,MM)=C
105

IPASS=IPASS+1
300

IF(IPASS.GT.1) GO TO 200
307

MYTAG
TIME=1.000.
ISTART=1
EJECT=NORD
C SEARCH TO FIND NEAREST DESTINATION AREA TO LASTA
362

OJ 250 I=ISTART,1000
IF(IPASS.GT.1) GO TO 300
IF(OMUNH(I).GE.0) GO TO 250

97
SUBROUTINE ROUTE 7474 OPT=0 ROUND+= TRADE 84.6=41 12/17/88 10.07.88

113  IF(INUM==11111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111111
SUBROUTINE ROUTE

I=174  OPT=4  ROUND** TRICE  74.6463  12/12/88  15:07:53

230 IF(INFO(III) .EQ. 1) GO TO 273
231 IF( (PASS.GT.6) ) GO TO 276
232 IF(INVNXAM(I,III).LE.0) GO TO 276
233 IF(INDEK(III).GE.(1+10)) GO TO 275
234 G3 TO 270
235 CONTINUE
236 GO TO 240
237 IF (TEMP(LIII).EQ.IDEST(I)) GO TO 265
238 C JES W.RF AR.HAVE MIN LOAD FOR THIS TYPE
239 IF (TYPE(N,3) .GT. 204)
240 IF (IDAMPFF/>.LE. .LT.,MINLD(I)) GO TO 265
241 IF (IDAMPFF/>.LE. .LT.,MINLD(I)) GO TO 265
242 IF (IDAMPFF/>.LE. .LT.,MINLD(I)) GO TO 265
243 CONTINUE
244 TIME=TEMP+TEMP
245 IF (HE,GF.ACT(I)) GO TO 250
246 TIME=TIME
247 TIM=LSAVE
248 NEXT=SAVEA
249 LIST=I
250 CONTINUE
251 IF (TEMP(LIII).EQ.IDEST(I)) GO TO 265
252 IF (ITYPE .GT. 100) GO TO 265
253 NSAM=NSAM+1
254 NPALTA=NPALTA
255 IF (NEXT .LT. 100) GO TO 265
256 LSAM=LSAM+1
257 IF (ITYPE .EQ. 100) GO TO 265
258 IF (LSAM .LT. 100) GO TO 265
259 CONTINUE
260 IF (NEXT .LT. 100) GO TO 265
261 IF (NEXT .LT. 100) GO TO 265
262 IF (NEXT .LT. 100) GO TO 265
263 IF (NEXT .LT. 100) GO TO 265
264 IF (NEXT .LT. 100) GO TO 265
265 IF (NEXT .LT. 100) GO TO 265
266 IF (NEXT .LT. 100) GO TO 265
267 IF (NEXT .LT. 100) GO TO 265
268 IF (NEXT .LT. 100) GO TO 265
269 IF (NEXT .LT. 100) GO TO 265
270 IF (NEXT .LT. 100) GO TO 265
271 IF (NEXT .LT. 100) GO TO 265
272 IF (NEXT .LT. 100) GO TO 265
273 IF (NEXT .LT. 100) GO TO 265
274 IF (NEXT .LT. 100) GO TO 265
275 IF (NEXT .LT. 100) GO TO 265
276 CONTINUE
277 IF (NEXT .LT. 100) GO TO 265
278 IF (NEXT .LT. 100) GO TO 265
279 IF (NEXT .LT. 100) GO TO 265
280 IF (NEXT .LT. 100) GO TO 265
281 IF (NEXT .LT. 100) GO TO 265
282 IF (NEXT .LT. 100) GO TO 265
283 CONTINUE
284 IF (NEXT .LT. 100) GO TO 265
285 CONTINUE
286 CONTINUE
287 CONTINUE
288 CONTINUE
289 CONTINUE
290 CONTINUE
291 CONTINUE
292 CONTINUE
293 CONTINUE
294 CONTINUE
295 CONTINUE
296 CONTINUE
297 CONTINUE
298 CONTINUE
299 CONTINUE
300 CONTINUE
301 CONTINUE
302 CONTINUE
303 CONTINUE
304 CONTINUE
305 CONTINUE
306 CONTINUE
307 CONTINUE
308 CONTINUE
309 CONTINUE
310 CONTINUE
311 CONTINUE
312 CONTINUE
313 CONTINUE
314 CONTINUE
315 CONTINUE
316 CONTINUE
317 CONTINUE
318 CONTINUE
319 CONTINUE
320 CONTINUE
321 CONTINUE
322 CONTINUE
323 CONTINUE
324 CONTINUE
325 CONTINUE
326 CONTINUE
327 CONTINUE
328 CONTINUE
329 CONTINUE
330 CONTINUE
331 CONTINUE
332 CONTINUE
333 CONTINUE
334 CONTINUE
335 CONTINUE
336 CONTINUE
337 CONTINUE
338 CONTINUE
339 CONTINUE
340 CONTINUE
341 CONTINUE
342 CONTINUE
343 CONTINUE
344 CONTINUE
345 CONTINUE
346 CONTINUE
347 CONTINUE
348 CONTINUE
349 CONTINUE
350 CONTINUE
351 CONTINUE
352 CONTINUE
353 CONTINUE
354 CONTINUE
355 CONTINUE
356 CONTINUE
357 CONTINUE
358 CONTINUE
359 CONTINUE
360 CONTINUE
361 CONTINUE
362 CONTINUE
363 CONTINUE
364 CONTINUE
365 CONTINUE
366 CONTINUE
367 CONTINUE
368 CONTINUE
369 CONTINUE
370 CONTINUE
371 CONTINUE
372 CONTINUE
373 CONTINUE
374 CONTINUE
375 CONTINUE
376 CONTINUE
377 CONTINUE
378 CONTINUE
379 CONTINUE
380 CONTINUE
381 CONTINUE
382 CONTINUE
383 CONTINUE
384 CONTINUE
385 CONTINUE
386 CONTINUE
387 CONTINUE
388 CONTINUE
389 CONTINUE
390 CONTINUE
391 CONTINUE
392 CONTINUE
393 CONTINUE
394 CONTINUE
395 CONTINUE
396 CONTINUE
397 CONTINUE
398 CONTINUE
399 CONTINUE
400 CONTINUE
401 CONTINUE
402 CONTINUE
403 CONTINUE
404 CONTINUE
405 CONTINUE
406 CONTINUE
407 CONTINUE
408 CONTINUE
409 CONTINUE
410 CONTINUE
411 CONTINUE
412 CONTINUE
413 CONTINUE
414 CONTINUE
415 CONTINUE
416 CONTINUE
417 CONTINUE
418 CONTINUE
419 CONTINUE
420 CONTINUE
421 CONTINUE
422 CONTINUE
423 CONTINUE
424 CONTINUE
425 CONTINUE
426 CONTINUE
427 CONTINUE
428 CONTINUE
429 CONTINUE
430 CONTINUE
431 CONTINUE
432 CONTINUE
433 CONTINUE
SUBROUTINE ROUTE  7/W:4  0PTE  ROUNT:4* TRACE  FTN 4.64441  12/12/80  56.07.56

GC  TO  203

227 IF(NOFF+LDOFF+GE.MINLOC(ITYPE)) GO TO 203
IF(NOFF+LDOFF+GE.MINLOC(ITYPE)) GO TO 284
IF(INPALTS+GE.MINLOC(ITYPE)) GO TO 203

290 3 TRY TO FIND NEXT AREA WITH MIN LOAD TO BE MOVED

262 LISTS(VLSTA)+NEXTA=LSTA
GO TO 212

3 BUILD NEXT PORTION OF SCHEDULE BETWEEN AREAS
LSTA AND NEXTA - TRAVEL TIME = TIME

7 CONSIDER ORDERS WITH SAME ORIGIN AND DESTINATION
SAME DESTINATION - DIFFERENT ORIGINS

9 SELECT REST SCHEDULE SECTIONS NO OF PALLET# / TIME
ADD SEGMENT TO CURRENT VEHICLE SCHEDULE

502 GO TO 212

280 TIME=TIME+MINLOC(ITYPE)
IF(IFILL+NC+AND+CS+30.1F(L=MINLOC(ITYPE)) GO TO 39)
TIMES=TIME+TIMEL(L=TPA.ITYPE)+TIME(ITYPE)

303 IF(ITRUCK+TIMES)+3.01 GO TO 203
TIMES=TIME(ITYPE)+TIME(NEXTA+TPA.ITYPE)
IF(IFILL+NC+AND+NSAVE+ED+NEXTA) GO TO 395
IF(IFILL+NC+AND+NSAVE+ED+NEXTA) GO TO 203

390 IFFITRUCK+TIME=TIME(ITYPE)

310 UPDATED THE TRUCKS

500 GO TO 212

3000 IFFILL=1AS IFILL) GO TO 334
IF(ITYPE.EQ.1) GO TO 326
GO TO 3000

326 BUILD SCHEDULES FOR TRACTOR TRAILERS

300 CALL BLDTRIFILL(IFILL+CAP+LSTOR+WEI)
IF(IFILL+CAP+LSTOR+WEI) GO TO 4050
GO TO 3000

323 BUILD SCHEDULES FOR TRANSPORTERS

2800 CALL BLDTRIFILL(IFILL+CAP+LSTOR+WEI)
IF(IFILL+CAP+LSTOR+WEI) GO TO 4050
GO TO 3000

395 BUILD SCHEDULES FOR TRACTOR TRAILERS

300 CALL BLDTRIFILL(IFILL+CAP+LSTOR+WEI)
IF(IFILL+CAP+LSTOR+WEI) GO TO 4050

315 CONTINUE

GO TO (1000.7065.3000.3000) + ITYPE
LSAVE=LSTA
NSAVE=NEXTA
GO TO 315

G BUILD SCHEDULES FORHexString

1000 CALL BLDSTIFILL(IFILL+CAP+LSTOR+WEI)
IF(IFILL+CAP+LSTOR+WEI) GO TO 4050
GO TO 315

330 NO 362 IF(I=1NORDR1) GO TO 334
IF(IFILL+ED+IFILL=2) GO TO 332
IF(MOD(IPASS+2)+NC+1) GO TO 4050

332 NO 362 IF(I=1NORDR1) GO TO 334
IF(IFILL+ED+IFILL=2) GO TO 332
IF(IFILL+ED+IFILL=2) GO TO 332
GO TO 4050

335 CONTINUE

GO TO 335

340 CALL BLDTRIFILL(IFILL)
IF(ITYPE.EQ.1) GO TO 4050
SUBROUTINE ROUTE 7/14 OPT.8 ROUND=%// TRACE 7/14 4.64R1 12/12/88 20.07.55

IF(JFILL.LT.3) GO TO 4060
CALL SRDSTILSTORO,TYPE,JFILL1

4060 ICAP=CAPAC(TTRUCK)
IFILL=0
LOAD=0
IF(TLEFT(TTRUCK).LE.0.00 50 TO 200

350 IF(NPALTS.GT.0) GO TO 150
LISTVLSAVE+MSAVE+=1
GO TO 150

4800 LAST=LSAVE
NPALT=NPALT-1
IF(NPALTS.LE.O) LISTVLSAVE+MSAVE+=1
IF(TLEFT(TTRUCK).LE.0.00 50 TO 200
IF(IPASS.LT.1) GO TO 300
IF(LSTSVLSAVE+MSAVE+=1.11) GO TO 193
IPASS=IPASS
GO TO 300

50 CONTINUE
IF(ICNTRL.NE.1) GO TO 9005
IF(NSPLIT.EQ.2) GO TO 9010
IF(NSPLIT.EQ.3) GO TO 9010
NSPLIT=NSPLIT+1
GO TO 50
GO TO 9010
IF(IRESET.NE.0) RETURN

9010 IRESET=1
ICNTRL=0
IPASS=0
NSPLIT=1
DO 4920 I=1,NORDR

375 IF(INFO(I).LE.0) GO TO 9020
GO TO 50
9020 CONTINUE
RETURN
END
ROUTINE: SRTDST

ARGUMENTS:

LSTORD - List number of last order added to route segment
ITYPE - Vehicle type number
JFILL - Number of orders in route segment

Description:

SRTDST re-orders the last route segment for a given vehicle. Order destinations are re-arranged to allow maximum vehicle utilization with a minimum travel time.
ENTER SRIDST

DETERMINE BEGINNING OF ROUTE SEGMENT

SAVE LAST WAREHOUSE TO BE VISITED BY VEHICLE

SORT ORDERS IN SEGMENT BY DELIVERY WAREHOUSES

PLACE ORDERS WITH LAST WAREHOUSE VISITED AT END OR BEGINNING

LINK SORTED ORDERS TO ROUTE

RETURN
SUBROUTINE SRTST (LSTORD, ITYPE, JFILL)

C-----------------------------------------------------------------------
C SRTST - LAST LIST ENTRY OF ROUTE SEGMENT
C ITYPE - VEHICLE TYPE
C JFILL - NUMBER OF ORDERS IN ROUTE SEGMENT
C C
C INTEGER ONUM, WARE(14), PTR(14)

COMMON

C SFTRSR, ONUM, INFO, IDEST, IAREA, DUM(400), DUM(400),
C NZE(400), LNAME(400), IDUM(400), DUM(400)
C ON = 0, I = 1, 14
C WAPF(J) = P
C PTR(J) = 0
C
C FIND START OF SEGMENT
C ISTART = LSTORD
C
C 100 WAPF(JFILL-JFILL) = MAX(IARS(INFO(ISTART))/100,100)
C
C 1. ICNT = ICNT + 1
C 2. IF (JFILL-MCNT) = ISTART
C 3. IF (JFILL-MCNT) = ISTART
C 4. IF (JFILL-MCNT) GO TO 200
C 5. ISTART = IDUM(ISTART)
C 6. GO TO 100
C 7. 200 ISTART = LNAME(ISTART)
C 8. GO TO 100
C
C GROUP LIKE DESTINATIONS
C C SPRT GROUP SEGMENT
C 100 LIMIT = ICNT - 1
C I = 1
C 150 JSTART = I
C MNT = 0
C 200 J = JSTART, ICNT
C IF (WAPF(J) = WAPF(J)) GO TO 400
C MNT = MNT + 1
C IF (J = JSTART) GO TO 400
C 3. ISAVF = WAPF(I) + MNT
C 4. JSAVE = PTR(J) + MNT
C 5. WAPF(J) = ISAVF
C 6. PTR(J) = JSAVE
C 7. CONTINUE
C 8. IF (MCNT = LIMIT) I = I + 1
C 9. IF (MCNT = LIMIT) GO TO 150
C 400 CONTINUE
C IF (MCNT = 0) I = I + 1
C I = I + 1
C 5. IF (MCNT = LIMIT) GO TO 350
C
C ADJUST LINKED LIST'S POINTERS WITH DELIVERY SEQUENCE
C ICNT = ICNT - 1
SUBROUTINE SRTOST  73/74  OPT=7 ROUND=*/ TRACE  

DO 550 I=1,JCTN
    LINK=PTR(I)
    FRNDO(LINK)=PTR(I+1)
    IF(I.GT.1) LBKNO(LINK)=PTR(I-1)
    IDESTP(LINK)=-I
550 CONTINUE

60 LINK=PTR(ICNT)
    IDFSTP(LINK)=ICNT
    LFRND0(LINK)=0
    LBKNO(LINK)=PTR(ICNT-1)
    RETURN
END
ROUTINE: T

ARGUMENTS:
   IAREAO – Origin area number
   IAREAT – Destination area number
   K      – Vehicle type number

Description:
T computes the travel time between two areas, IAREAO and IAREAT, for a vehicle of type K. If IAREAO equals IAREAT, T is set equal to 2 minutes. If IAREAO and IAREAT are both off base sites, T is set equal to 1000 minutes to prevent travel between the two areas.
ENTER T

IS IAREAO OR IAREAT OFF-BASE?

Y

T=2

RETURN

N

IS IAREAO = IAREAT?

Y

N

USE TTIME TABLE TO FIND T

RETURN

ARE IAREAO & IAREAT OFF-BASE?

Y

T=1000

N

USE TTIM2 TABLE TO FIND T

RETURN
FUNCTION T(AREA,AAREAT,K)

1  THIS FUNCTION GIVES THE TRAVEL TIME BETWEEN POINTS I AND IT
FOR TRUCK TYPE K.

5  TIMES ARE STORED IN TRIANGULAR ARRAYS WITH ZERO ON THE DIAGONAL.

10  ALPHA VAR

15  COMMUN/TIMEM/TIME(E,451,TTIMZ61)

20  COMMUN/MAKES/MAKEM}

25  NAGUH MUST BE CHANGED IF WAREHOUSE GROUPING CHANGES.

30  NAGUH = 10

35  IF(NAGUH=U) GO TO 450C.

40  IF(NAGUH=U) GO TO 5500.

45  NAGUH = NAGU + NAGU - 10.

50  FIND TIME FOR SITES IN DIFFERENT AREAS.

55  L = 1

60  SUM = 150 + NAGU + 10.

65  IF(L=SUM) GO TO 3000.

70  N = C - 1

75  SUM = SUM + L

80  IF(L=SUM) GO TO 3000.

85  GO TO 4500.

90  GO TO 4500.

95  FIND TIME FOR SITES IN SAME AREA.

100  GO TO 4500.

105  FIND TIME FOR OFFBASE SITES.

110  GO TO 4500.

115  T = 1003.

120  PREVENTS TRAVEL BETWEEN OFFBASE SITES.

125  T = 1003.

130  RETURN.

135  ENC.
ROUTINE: TCARP

Description:

The primary purpose of TCARP is to translate the linked lists notation for each vehicle in service into readable schedules.

When the bump option is set for a special order run (AVS2), TCARP traces through each vehicle's schedule to determine whether the vehicle is suitable to move special orders. Each suitable vehicle is made available for the special order run.
112
SUBROUTINE TCARP

C*****************************************************************************
C
C TCARP - DECODES SCHEDULES FROM THE LINKED LISTS

C*****************************************************************************

C

INTEGER ONUMB, TRUCK, PTSOR, PTTRK, NODEST(15)
INTEGER TRKSAV, CAPAC
REAL BTIM, TYPE(4), PR1S(2), IPR
REAL LTIM
COMMON
* MSLNS/ NORDR, NTTRK(4), PTTRK(4)
* NVSAV(4), TRKSAV(4)
* SHSAV/ JSAV(3)
* HOUSE/ HMMAP(2)
* NSCHDLY/ ONUMB(200), LINFO(200), IDESTP(200), IAREA(200), LFMAP(200),
* LCFNMA(200), LCCMDI(200), ATIME(200), STOPT(200)
* TRUCKV/ PTSOR(50), TRUCK(50), CAPAC(50), TIME(50), LTIM(50),
* RTIM(50), LLEFT(50), NT
*C
"FINOPT/ SHIFT, IODATE
DATA TYPE/2HST, 34E,F, 2HTT.
DATA PRTS/6M, NPSCL, / 
C
LIST ALL UNMOVED ORDERS 
PRTS=SHIFT 
SAVS=SHIFT
"IPCOL=23
JAI=2
IF (TRKSAV(IJ), LE, 10) GO TO 1400
MSTART=TRKSAV(IJ)
LJ=PTTRK(IJ)
IF (PTTRK(IJ), LE, 10) GO TO 2000
NPALT=0
LSTOP=0
"T=0
LSTOP=IPCOL
SA=JAI
"MPL=1
LINK=PTSOR(IJ)
LINKSAV=LINK
"C

J, TRK, VEHICLE, TYPE
"TYPE=1
IF (TRK, GT, PTTRK(IJ), TYPE=2
IF (TRK, GT, PTTRK(IJ), TYPE=3
IF (TRK, LT, PTTRK(IJ), TYPE=4
J=0
TIME=SHIFT
60 IF (TRKSAV(IJ), LE, 10, IVEH=TRK-PTTRK(IJ), IVEH)
IF (TRKSAV(IJ), LE, 10, IVEH=TRK-PTTRK(IJ), IVEH)
55 IF (TRKSAV(IJ), LE, 10, IVEH=TRK-PTTRK(IJ), IVEH)

SUBROUTINE TCA9P
    7x74    OPF*0    ROUND**/ TRACK

    10/17/00    10.04.07

2010 INFO(4) = IABS(INFO(4))
    IF (INFO(4).LT.0) GO TO 2014

    IF (INFO(4).EQ.0) GO TO 2015

70    CLEAR VEHICLE SCHEDULE

2020 INFO(4) = IABS(INFO(4))
    ISTATE(1) = 0

    LAST(LINK) = 0

    LKWH(LAST) = 0

    STOP(LINK) = 0

    AIM(I(I)) = 0

    LNK(LINK(LINK)) = 0

    LNK(LINK).GT.0 GO TO 2025

    UPDATE TIME REMAINING FOR VEHICLE

    ISS*SHIFT

    KSHIFT = (ISS/I30)*60*MOD(ISS,100)

    TLEFT(I) = TIME(LINK(I)) + (ISS/1000)*60*MOD(ISS,100)

    TLEFT(I) = KSHIFT

    TLEFT(I) = TLEFT(I) + (ISS/1000)*60*MOD(ISS,100)

    TLEFT(I) = TLEFT(I) + MOD(TIME,100)

    GO TO 2005

2030 INFO(4) = IABS(INFO(4))

    ISTATE(1) = 0

    LAST(LINK) = 0

    LKWH(LAST) = 0

    STOP(LINK) = 0

    AIM(I(I)) = 0

    LNK(LINK(LINK)) = 0

    LNK(LINK).GT.0 GO TO 2025

    UPDATE TIME REMAINING FOR VEHICLE

    ISS*SHIFT

    KSHIFT = (ISS/I30)*60*MOD(ISS,100)

    TLEFT(I) = TIME(LINK(I)) + (ISS/1000)*60*MOD(ISS,100)

    TLEFT(I) = TLEFT(I) + (ISS/1000)*60*MOD(ISS,100)

    TLEFT(I) = TLEFT(I) + MOD(TIME,100)

    GO TO 2005

2014 LNK(LINK).GT.0 GO TO 2014

    ISS*SHIFT

    KSHIFT = (ISS/I30)*60*MOD(ISS,100)

    TLEFT(I) = TIME(LINK(I)) + (ISS/1000)*60*MOD(ISS,100)

    TLEFT(I) = TLEFT(I) + (ISS/1000)*60*MOD(ISS,100)

    TLEFT(I) = TLEFT(I) + MOD(TIME,100)

    GO TO 2016

2015 CONTINUE

100 INFO(4) = INFO(4)/10000

    ISTATE = MOD(INFO(4),100,100)

    ISIZE = MOD(INFO(4),100)

    NPALS = NPALS + ISIZE

    NOESEL = NPALS

105 NOESEL(NPELS) = ISTATE

    IA2 = IAREALINK(I)

    IF (IA2.EQ.0) GO TO 4065

    ATIME(LINK(I)) = TIME(I) + I2.4
tYPE)

    TIME = ATTIME(LINK(I)) + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)

    TIME = TIME + (ISS/1000)*60*MOD(ISS,100)

    TIME = TIME + MOD(TIME,100)

    TIME = TIME + TLEFT(I)
SUBROUTINE TCARP 7x74 OPT=0 ROUND=* TRACE  

115 IF(FACTBL(GJ.GT.0)) GO TO 2000  
NUM=MOD(NUM,NUMLINK))  
IF(NUMLINK.GE.0) GO TO 101  
IPRT=PTSS(2)  
IF(NUM.LE.100) IPRT=6T*SPLIT  
NUM=MOD(NUM,100)  

101 WRITE(6,603) ISTOP,WHNM3(TIRG1),ITIME,ISIZE,NUM,ISTAY,IPRT  
IPRT=PTSS(1)  

8000 JORIG=JORIG  
GO TO 4010  

125 4005 STAY=TIME(ITR)+FLOAT(ISIZE)  
STAY=STAY*4  
TIME=TIME*STAY  
IF(FACTBL(GJ.GT.0)) GO TO 2010  
NUM=MOD(NUM,NUMLINK))  
IF(NUMLINK.GE.0) GO TO 112  
IPRT=PTSS(2)  
IF(NUM.LE.100) IPRT=6T*SPLIT  
NUM=MOD(NUM,100)  

135 12U WRITE(6,605) ISIZE,NUM,ISTAY,IPRT  
IPRT=PTSS(1)  

4010 JDEST=3  
NEXT=LFNM3(LINK)  
NEXT=NEXT  

144 IF(IESTP(GJ).EQ.0) GO TO 8200  
IF(IJESTP1(LINK).GE.IESTP(NEXT)) GO TO 3000  
IF(IJESTP1(LINK).LT.IESTP(NEXT)) GO TO 3200  
GO TO 5000  

145 4000 IF(IJESTP1(LINK).GT.IESTP(NEXT)) GO TO 3100  
GO TO 5000  

5000 LINK=NEXT  
LIST=JORIG  
IA1=IA2  
GO TO 2010  

150 5 COMPUTE TIME TO DESTINATION  
3000 IA2=MOD(EQAALLINKJ.IOO)  
TIME=TIME+TIA1,IA2,IYPE  
STOP(LINK)=TIME  
IS0P=JSTOP1  
TIME=JONVTIME,PRSH1  
STAY=TIME(ITR)+FLOAT(ISIZE)+TIME(ITR)  
TIME=TIME*STAY  
ISTAY=ISTAY+4  
IF(IESTP(GJ).EQ.0) GO TO 8001  
NUM=MOD(NUM,NUMLINK))  
IF(NUMLINK.GE.IO) GO TO 103  
IPRT=PTSS(2)  
IF(NUM.LE.100) IPRT=6T*SPLIT  
NUM=MOD(NUM,100)  

165 103 WRITE(6,6041) ISTOP,WHNM3(IEST),ITIME,ISIZE,NUM,ISTAY,IPRT  
IPRT=PTSS(1)  
8001 JEST=0  
J0RIG=IEST  
LINK=NEXT  
LINKSA=NEXT  
NOEL=0  

TCARP1 16  
TCARP1 17  
TCARP1 18  
TCARP1 19  
TCARP1 20  
TCARP1 21  
TCARP1 22  
TCARP1 23  
TCARP1 24  
TCARP1 25  
TCARP1 26  
TCARP1 27  
TCARP1 28  
TCARP1 29  
TCARP1 30  
TCARP1 31  
TCARP1 32  
TCARP1 33  
TCARP1 34  
TCARP1 35  
TCARP1 36  
TCARP1 37  
TCARP1 38  
TCARP1 39  
TCARP1 40  
TCARP1 41  
TCARP1 42  
TCARP1 43  
TCARP1 44  
TCARP1 45  
TCARP1 46  
TCARP1 47  
TCARP1 48  
TCARP1 49  
TCARP1 50  
TCARP1 51  
TCARP1 52  
TCARP1 53  
TCARP1 54  
TCARP1 55  
TCARP1 56  
TCARP1 57  
TCARP1 58  
TCARP1 59  
TCARP1 60  
TCARP1 61  
TCARP1 62  
TCARP1 63  
TCARP1 64  
TCARP1 65  
TCARP1 66  
TCARP1 67  
TCARP1 68  
TCARP1 69  
TCARP1 70  
TCARP1 71  
TCARP1 72  

115
SUBROUTINE TCARP  7/4/74  CPT=0 ROUND=+/ TRACE FTN 4.6*60  10/17/80  10.04.12

IA1=IA2  
LSTOP=INEST  
GJ TO 3100  
TCARP173

175  C  FIRST ON - FIRST OFF  
2000 IF(ITYPE.EQ.1) GO TO 3100  
IA1=IA2  
LNSAV=LINK  
GO TO 3210  
TCARP176

180  3100 IF(NOEL.EQ.3) GO TO 7000  
03  6000 II=1.NOEL  
LDEST=INDEST(II)  
IA2=L(II)*L(II)*L(II)*L(II)  
ISIZE=L(II)*L(II)*L(II)*L(II)  
STOP(LNSAV)=TIME  
STAY=TIME(ITRK)*FLOAT(ISIZE)  
TIME=TIME+STAY  
ISTAY=STAY+.9  
TCARP184

190  IF(INTRL.EQ.0) GO TO 3110  
NUM=INS(0,0)  
STOP=NUM(SAVE),GE.0) GO TO 104  
IPRT=IPRT(2)  
IF(NUM.LE.100) IPRT=NUM+SPLIT  
TCAQP195

195  104 WRITE(6,607) ISIZE,NUM,ISTAY,IPRT  
IPRT=IPRT(1)  
GO TO 3110  
TCARP199

200  3105 STOP=ISTOP=1  
JOEST=INEST  
TIME=TIME+TI120,IA2,ITYPE  
STOP(LNSAV)=TIME  
STAY=TIME(ITRK)*FLOAT(ISIZE)+TIME(ITRK)  
TIME=TIME+STAY  
ISTAY=STAY+.9  
TCARP206

205  IF(MTOL=STOP(LNSAV),IPRTSHF)  
IF(MTOL.EQ.0) GO TO 3110  
NUM=NUM(SAVE),GE.0) GO TO 105  
IPRT=IPRT(3)  
TCARPP209

210  IF(NUM.LE.100) IPRT=NUM+SPLIT  
NUM=MAX(NUM,100)  
TIME,ISIZE,NUM,ISTAY,IPRT  
IPRT=IPRT(3)  
TCARPP213

215  3110 IA1=IA2  
LNSAV=LAREA(LNSAV)  
STOP=INEST  
GO TO 3100  
TCARPP216

220  6000 CONTINUE  
NOEL=0  
IF(LNSAV.LT.0) GO TO 7000  
TCARPP218

225  7000 LNSAV=LNSAV(INEXT)  
GO TO 20  
TCARPP222

C  FIRST ON - LAST OFF  
3200 LNSAV=LNSAV(INEXT)  
TCARPP225

3210 IA2=L(D(II)*L(II)*L(II)*L(II)  
LOCST=MTOL(SAVE),L(II),100)  
ISIZE=L(D(II)*L(II)*L(II)*L(II)  
IF(JOEST.ME.INEST) GO TO 3205  
TCARPP229

116
SUBROUTINE TCAQP  7/17/74  CPT=0  ROUNDS=7  TRACE  FINT 4.6+60  12/17/80  10:54:17

STOP(LNKSAV)=TIME
STAY=TIME*(ITR)*FLOAT(ISIZE)
ISTAY=STAY+.9
TIME=TIME+STAY
IF(INCTRL.EQ.0) GC TO 3220
NUM=ABS(NUMB(LNKSAV))
IF(IRONUM(LNKSAV).EQ.0) GC TO 100
IPRT=PRTS(2)
IF(INCTRL.LE.10) IPRT=H*N*SPLIT
NUM=MOD(NUM,100)
106 WRITE(*,607) ISIZE(NUM),ISTAY,IPRT
IPRT=PRTS(1)
GO TO 3220
3205 IOEST=IOEST
ISTOP=ISTOP+1
TIME=TIME*(IA1,IA2,ITYPE)
STOP(LNKSAV)=TIME
STAY=TIME*(ITR)*FLOAT(ISIZE)+STAY*(ITR)
ISTAY=STAY+.9
TIME=TIME+STAY
IF(INCTRL.LE.6) GC TO 3220
NUM=ABS(NUMB(LNKSAV))
IF(INCTRL.GT.6) GC TO 107
IPRT=PRTS(1)
IF(INCTRL.LE.10) IPRT=H*N*SPLIT
NUM=MOD(NUM,100)
107 WRITE(*,604) ISTOP,WHAM(IOEST),TIME,ISIZE(NUM),ISTAY,IPRT
IPRT=PRTS(1)
3220 LA1=IA2
LSTOP=IOEST
NOEL=NOEL-1
LINKS=LINKS
IF(INCTRL.LE.3) GC TO 3220
IF(INXSAV.LE.0) GC TO 7000
LINKS=LINKS
LINKS=VANXSAV
JORG=IOLST
GO TO 2010
C END ROUTE FOR VEHICLE
7000 TIME=TIME*(IA1,JA1,ITYPE)
TIME=CONVTIME(FAT,SHF)
IF(INCTRL.LE.6) GC TO 2005
WRITE(6,606) WHAM,EPCEL,TIME,NPALS
2001 WRITE(6,601) LHNUM(LHNUM,LCM),TIME,NPALS
SHF=SAVSHF
SAVSH=0.0
275 WRITE(*,600) SAVSH
2000 CONTINUE
1000 CONTINUE
DO 9000 I=1,NCROR
INFO(I)=INFO(I)
9000 CONTINUE
WRITE(6,600) RETURN
280 DO 6000 I=1,NCROR
END
SUBROUTINE TCARP

TRAN = 0 ROUND**/ TRACE

FTN 4.6W40
10/17/80 10.24.17

IF(INF(I),LL,L) GC TO 3001
I=IABS(INF(I)/10000)
IT=MOD(IARS(INFO(I))/100,100)
ISIZE=MOD(IARS(INFO(I)),100)

WRITE(6,601) CUNHET(I),IT,SIZE,MPNAME(I0),HNAME(IIT)

9001 CONTINUE

RETURN

600 FORMAT(1X,50(1H*),/)

298 CONTINUE

RETURN

601 FORMAT(1X,BORDER,14,2H,17,6FROM,A6,4X TO,A6)

299 CONTINUE

*F5.0/20X,UHOR,9X,10//,1M0,76/1N-1/6X,WHSTOP,2X,WHSITE,1X,WHTIME

*5X,2DELIVER,7X,7FUCKUP,5X,5HORDEF,5X,155TIME(MIN) /

300 CONTINUE

RETURN

602 FORMAT(1X,14,1X,14,1X,15,17,12,AM PALLETS,4X,14,15,1X,A)

301 CONTINUE

RETURN

603 FORMAT(1X,14,1X,14,1X,15,17,12,AM PALLETS,10X,14,15,1X,A)

302 CONTINUE

RETURN

604 FORMAT(1X,14,1X,14,1X,15,17,12,AM PALLETS,10X,14,15,1X,A)

303 CONTINUE

RETURN

605 FORMAT(4X,12,4M PALLETS,1X,14,15,1X,A)

304 CONTINUE

RETURN

606 FORMAT(12,4M PALLETS,1X,14,15,1X,A)

305 CONTINUE

RETURN

607 FORMAT(12,4M PALLETS,1X,14,15,1X,A)

306 CONTINUE

END
ROUTINE: TCONV

ARGUMENTS:

A - Relative time in minutes
S - Start of work shift (24-hour clock)

Description

TCONV adds the relative time to the start time of the shift. The sum is converted to 24-hour clock time. TCONV is then set equal to the result.
TCONV

CONVERT SHIFT
TIME TO HR
& MINS

CONVERT
ATIME TO
HR & MINS

ADD ATIME
TO SHIFT
TIME

IS
MINS \geq 60
?

YES

CONVERT
TO HRS & MINS

NO

SET TCONV
TO MODIFIED
SUM

RETURN
FUNCTION TCONV(A,S)
C THIS FUNCTION TAKES REAL INPUTS A = REL. TIME IN MINUTES
C AND S = START OF SHIFT (24 HR. CLOCK)
C ADDS A TO S AND OUTPUTS THE RESULT, TCONV, AS 24 HR. CLOCK TIME
SHR = FLOAT(IFIX(S/100.))
SMIN = S - SHR*120.
XHR = FLOAT(IFIX(A/60.))
XMIN = A - XHR*60.
YMIN = SMIN + XMIN
YHR = SHR + XHF
TCONV = 100.*YHR + YMIN
IF(YMIN.LT.50.) RETURN
TCONV = TCONV + 48.
RETURN
END

121
APPENDIX C - AVSIN1, A SINR COBOL DRIVER ROUTINE, LISTINGS
00130 IDENTIFICATION DIVISION.
00135 PROGRAM-ID.
00139 AVS1.
00140 AUTHOR.
00145  R. McElton.
00150 INSTALLATION.
00155 OTNSRDG.
00160 DATE-WRITTEN.
00165 14 DEC 74.
00170 DATE-EXECUTED.
00175 14 DEC 74.
00180 SECURITY.
00185 UNCLASSIFIED.
00190 REMARKS.
00195 COMBINED INPUT-FRAME PROGRAM FOR AVS1 AND AVS2.
00200 ENVIRONMENT DIVISION.
00205 CONFIGURATION SECTION.
00210 SOURCE-PROGRAM.
00215 OBJECT-PROGRAM.
00220 INPUT-OUTPUT SECTION.
00225 FILE-CONTROL.
00230 SELECT SOFILE ASSIGN TO DISK.
00235 RESERVE 1 ALTERNATE AREAS.
00240 SELECT VSIN ASSIGN TO DISK.
00245 RESERVE 1 ALTERNATE AREA.
00250 SELECT VSIN ASSIGN TO DISK.
00255 RESERVE 1 ALTERNATE AREA.
00260 DATA DIVISION.
00265 FILE SECTION.
00270 FD SOFILE COPY SOFILE.
00275 01 AVS-RECORD.
00280 03 AVS-PREFIX PIC X(24).
00285 03 AVS-DATA PIC X(284).
00290 INPUT-DATA.
00295 03 FILLER PIC X(2).
00300 03 COMMON-PREFIX.
00305 05 OCCID PIC X.
00310 05 FRAME-NR PIC X.
00315 03 FILLER PIC X(2).
00320 01 FRAME-1-INPUT.
00325 03 FILLER PIC X(2).
00330 03 FILLER PIC X(1).
00335 03 ORDER OCCURS 2 TIMES.
00340 05 O-SIZE PIC X.
00345 05 O-CRIG PIC XXX.
00350 05 O-NEST PIC X(1).
00355 01 FRAME-2-INPUT.
00360 03 FILLER PIC X(2).
00365 03 FILLER PIC X(1).
00370 03 TRICK OCCURS 30 TIMES.

124
00215C 05 V-US PIC X.
002170 05 CAFAC PIC XX.
002190 05 TIME-LIM PIC XXX.
002210 03 FILLER PIC X(110).
002230 01 FRAME-1-INPUT.
002250 03 FILLER PIC X(24).
002270 03 FILLER PIC X(4).
002290 03 I-DATE PIC X(4).
002300 03 START-TIME PIC X(4).
002320 03 MAX-ROUTE PIC XXX.
002340 03 FILLER PIC X(256).
002370 01 FRAME-2-INPUT.
002390 03 FILLER PIC X(24).
002410 03 FILLER PIC X(4).
002430 03 M-OPTION PIC X.
002450 03 P-OPTION PIC X.
002470 03 FILLER PIC X(256).
002490 01 FRAME-3-INPUT.
002510 03 FILLER PIC X(24).
002530 03 FILLER PIC X(4).
002550 03 FLN-TYPE PIC X.
002570 03 START-DATE PIC X(4).
002590 03 M-OPTION PIC X.
002610 03 FND-CATE PIC X(4).
002630 03 M-SHIFT PIC X(-3).
002650 03 TODAY-DATE PIC X(4).
002670 03 TODAY-SHIFT PIC X(-1).
002690 03 UPDATE-RECS OCCURS 6 TIMES.
002710 05 ORIG PIC X(4).
002730 05 DEST PIC X(-1).
002750 05 SIZE PIC XX.
002770 05 VIEW PIC X(-1).
002790 05 EMER PIC X.
002810 05 PLEH PIC X.
002830 03 FILLER PIC X(122).
002850 FD VS2IN FILE CONTAINS 20 BY 1000 RECORDS
002870 10 RECORD CONTAINS 80 CHARACTERS
002890 10 DATA RECORD IS CARD-IMAGE.
002910 01 CARD-IMAGE PIC X(40).
002930 FD VS2IN FILE CONTAINS 2 BY 1000 RECORDS
002950 10 RECORD CONTAINS 80 CHARACTERS
002970 01 DATA RECORD IS CARD-IMAGE.
002990 01 CARD-IMAGE PIC X(40).
003010 00 WORKING-STORAGE SECTION.
003030 10 PROGRAM-NAME PIC X(4) VALUE "AVSINI".
003050 77 ERR-CNTR PIC 9999 COMP.
003070 77 LLAST PIC 9999 COMP VALUE ZEROES.
003090 77 JJ PIC 9999 COMP VALUE ZEROES.
003110 77 HIST-CHECK PIC X VALUE SPACE.
003130 77 TRAILER-CAFAC PIC XX VALUE "05".
003150 77 TRANS-CAFAC PIC XX VALUE "11".
003170 77 TRAILER-CAFAC PIC XX VALUE "14".
003190 77 MAX-CAPAC PIC XX.
003210 77 SAME-REC PIC X VALUE SPACE.
003230 77 SAME-REC PIC X VALUE SPACE.
003330 77 J-DSN PIC 4y.
003350 77 J PIC 9999 COMP.
003370 77 M PIC 99 COMP.
003390 77 X PIC 99 COMP.
003410 77 JLAST PIC 99 COMP.
003430 77 JLAST PIC 99 COMP.
003450 77 J-FLAG PIC 9 COMP.
003470 77 K-FLAG PIC 9 COMP.
003490 77 GRO0FkS-FLLL FIC
003510 77 JLAST PIC 99 COMP.
003530 77 J-FLAG PIC 9 COMP.
003550 77 J-FLAG PIC 9 COMP.
003570 77 TYPE CT PIC 9.
003590 77 TEST-NAME PIC X(6).
003610 77 TRUCK-TYPE PIC X.
003630 77 OPT-NAME PIC X(6).
003650 77 I-WORK-AREA COPY ISTOKE.
003670 01 TYPE-NAME PIC 9.
003690 01 TEST-NAME PIC 9.
003710 01 FIRST-NAME PIC 9.
003730 01 TYP-NAME PIC 9.
003750 01 TYPE-NAME PIC 9.
003770 01 VALUE "EX AVS1 CG 162404 ."
003790 01 VALUE "EX AVS1 CG 162404 ."
003810 01 TYPE-CT-SP.
003830 01 ORDER-OUTPUT OCCURS 99 TIMES.
003850 01 ORDER-NR PIC 9.
003870 01 ORDER-STORE.
003890 01 ORDER-STORE.
003910 01 ORDER-STORE.
003930 01 ORDER-STORE.
003950 01 ORDER-STORE.
003970 01 ORDER-STORE.
003990 01 ORDER-STORE.
004010 01 ORDER-STORE.
004030 01 ORDER-STORE.
004050 01 ORDER-STORE.
004070 01 ORDER-STORE.
004090 01 ORDER-STORE.
004110 01 ORDER-STORE.
004130 01 ORDER-STORE.
004150 01 ORDER-STORE.
004170 01 ORDER-STORE.
004190 01 ORDER-STORE.
004210 01 ORDER-STORE.
004230 01 ORDER-STORE.
004250 01 ORDER-STORE.
004270 01 ORDER-STORE.
004290 01 ORDER-STORE.
004310 01 ORDER-STORE.
004330 01 ORDER-STORE.
004350 01 ORDER-STORE.
004370 01 ORDER-STORE.
004390 01 ORDER-STORE.

126
004410 01 FRAME-3-OUTPUT.
004430 03 ERR-FIELD-3 OCCURS 6 TIMES.
004450 05 INPUT-ERROR PIC X(4).
004470 05 EXPLANATION-3 PIC X(21).
004490 01 FRAME-6-OUTPUT.
004510 03 OUT-MIX PIC X.
004530 03 OUT-CPTPIC PIC X.
004550 01 FRAME-1-OUTPUT.
004570 03 ERR-FIELD-4 OCCURS 6 TIMES.
004590 05 INPUT-ERROR PIC X(4).
004610 05 EXPLANATION-4 PIC X(19).
004630 01 CPT-CARE.
004650 03 O-OPT PIC X.
004670 03 FILLER PIC X(7) VALUE SPACES.
004690 01 OATE-CARD.
004710 03 O-DATE PIC X(6).
004730 03 FILLER PIC X(7) VALUE SPACES.
004740 01 SHIFT-CARD.
004750 03 O-SHIFT PIC X(1) VALUE SPACES.
004760 03 FILLER PIC X(7) VALUE SPACES.
004800 01 CPT-CARE-E.
004820 03 O-MIX PIC X VALUE "O".
004850 03 FILLER PIC X VALUE SPACE.
004870 03 PROPT PIC X VALUE "O".
004890 03 FILLER PIC X(7) VALUE SPACES.
004910 01 OKOLR-CARD.
004930 03 O-OPT PIC X.
004950 03 FILLER PIC X(7) VALUE SPACES.
004970 03 O-ORIGIN PIC X(6).
004990 03 FILLER PIC X VALUE SPACE.
005010 03 O-DESTIN PIC X(6).
005030 03 FILLER PIC X(7) VALUE SPACES.
005050 01 ORDER-CARD-E.
005070 03 O-TIME PIC X(4).
005100 03 FILLER PIC X(7) VALUE SPACES.
005120 03 O-SIZE PIC X.
005140 03 FILLER PIC X VALUE SPACE.
005160 03 O-ORIGIN PIC X(6).
005180 03 FILLER PIC X VALUE SPACE.
005200 03 O-DEST PIC X(1).
005210 03 FILLER PIC X(7) VALUE SPACES.
005230 01 END-CARD.
005250 03 C-EXIT-SYMBO PIC XX VALUE "-1".
005270 03 FILLER PIC X(7) VALUE SPACES.
005290 01 TRUCK=-EXIT-CARD-E.
005310 03 O-TRUCK-TITLE-1 PIC X(16) VALUE "TRUCKS".
005330 03 FILLER PIC X VALUE SPACE.
005350 03 O-NUMP-1+1+ PIC X.
005370 03 FILLER PIC X VALUE SPACE.
005390 03 O-NUMP-1+1+ PIC X.
005410 03 FILLER PIC X VALUE SPACE.
005430 03 O-NUMP-1+1+ PIC X.
005450 03 FILLER PIC X VALUE SPACE.
005470 03 O-ALL-1+1+ PIC X.
005490 03 FILLER PIC X(3) VALUE SPACES.
006470  "wo 53C  6E  64E  66E  66W  67E  67W  198 ".
006490  03 FILLER PIC X(114) VALUE
006510  "1127  11M  SF  2  3  5  8  35  43C  44
006530  "59  223  190E  1903  1937  80  177  1143  1199 ".
006550  03 FILLER PIC X(114) VALUE
006570  "187  21H  218  1013  1169  1171  1173  1172  1174  X10
006590  "193  224  32  435  61  84  202  046  647 ".
006610  03 FILLER PIC X(102) VALUE
006630  "65b  1  7  16  23  26  53S  X54  ARASE MWS
006650  "DENTY.BRASS.CSNWS.NMEDC 1078  98  655 ".
006670  03 FILLER PIC X(49) VALUE
006690  "L  M  N  P  Q  R  S  T ".
006710  01 WAREHOUSE-NAME  RECEIVES WAREHOUSE-DATA.
006730  02 W-NAME  OCCURS 82 TIMES  PIC X(6).
006750  01 PATCH-AREA  PIC X(200).
006770  PROCEDURE DIVISION.
006790  C12-START-FORMGRAP SECTION.
006810  G11-START.
006830  PERFORM I-START.
006850  C20-UPLN-FILE SECTION 51.
006870  G21-OPEN.
006890  COPY SOCKET.
006910  G30-PROCESS SECTION.
006930  G31-INITIALIZE.
006950  MOVE ZERO TO JLAST.
006970  MOVE ZERO TO KLAST.
006990  MOVE ZERO TO ORDRS-FULL.
007010  MOVE ZERO TO LLAST.
007030  G40-ARE-ON.
007050  MOVE ONE TO K.
007070  MOVE "GC" TO J-OSP.
007090  MOVE SPACES TO FRAME-1-INPUT.
007110  MOVE SPACES TO FRAME-2-INPUT.
007130  MOVE SPACES TO FRAME-3-INPUT.
007150  MOVE SPACES TO FRAME-6-INPUT.
007170  MOVE SPACES TO FRAME-7-INPUT.
007190  PERFORM SONREAD.
007210  IF FRAME-NR EQUALS "1" GO TO 100-FRAME-1.
007230  IF FRAME-NR EQUALS "2" GO TO 200-FRAME-2.
007250  IF FRAME-NR EQUALS "3" GO TO 300-FRAME-3.
007270  IF FRAME-NR EQUALS "4" GO TO 400-OUTPUT.
007290  IF FRAME-NR EQUALS "5" GO TO 500-EMERGENCY-INPUT.
007310  IF FRAME-NR EQUALS "6" GO TO 599-CLEAR-RESTART.
007330  IF FRAME-NR EQUALS "7" GO TO 700-HISTORY.
007350  GO TO 940-READ.
007370  100-FRAME-1.
007390  IF ORDRS-FULL EQUALS ONE MOVE "59" TO J-OSP
007410  GO TO 120-LAST-ORDER.
007430  MOVE ZERO TO K-FLAG.
007450  MOVE "01" TO J-OSP.
007470  PCVE ONE TO K.
007490  110-NEXT-ORDER.
007510  MOVE ZERO TO J-FLAG.
007530  IF O-SIZE T(J) EQUALS SPACES
007550  IF O-ORIG (J) EQUALS SPACES
007570  IF C-TEST (J) EQUALS SPACES
007590  SUBTRACT 1 FROM J-DSP
007610  GO TO 120-LAST-ORDER.
007630  IF 0-SIZE (J) NOT NUMERIC
007650  IF K-FLAG EQUALS CAE GC TO 120-LAST-ORDER
007670  ELSE MOVE J-DSP TO ORDER-NR (K)
007690  MOVE "SIZE NOT NUMERIC " TO EXPLANATION-1 (K)
007710  MOVE ONE TO J-FLAG
007730  ADD ONE TO K
007750  IF K GREATER THAN 10 MOVE ONE TO K-FLAG.
007770  MOVE O-ORIG (J) TO TEST-NAME.
007790  PERFORM 500-NAMES-TEST.
007810  IF TEST-NAME NOT :EQUAL O-ORIG (J)
007830  IF K-FLAG EQUALS ONE GO TO 120-LAST-ORDER
007850  ELSE MOVE J-DSP TO ORDER-NR (K)
007870  MOVE "ORIGIN MISMATCH" TO EXPLANATION-1 (K)
007890  MOVE ONE TO J-FLAG
007910  ADD ONE TO K
007930  IF K GREATER THAN 10 MOVE ONE TO K-FLAG.
007950  MOVE O-DEST (J) TO TEST-NAME.
007970  PERFORM 500-NAMES-TEST.
007990  IF TEST-NAME NOT :EQUAL O-DEST (J)
008010  IF K-FLAG EQUALS ONE GO TO 120-LAST-ORDER
008030  ELSE MOVE J-DSP TO ORDER-NR (K)
008050  MOVE "DESTINATION MISMATCH" TO EXPLANATION-1 (K)
008070  MOVE ONE TO J-FLAG
008090  ADD ONE TO K
008110  IF K GREATER THAN 10 MOVE ONE TO K-FLAG.
008130  IF J-FLAG NOT :EQUAL ONE
008150  ADD ONE TO JLAST
008170  IF JLAST GREATER THAN 99 MOVE ONE TO ORDERS-FULL
008190  GO TO 120-LAST-ORDER
008210  ELSE MOVE 0-SIZE (J) TO OUT-SIZE (JLAST)
008230  MOVE O-ORIG (J) TO OUT-ORIG (JLAST)
008250  MOVE O-DEST (J) TO OUT-DEST (JLAST)
008270  ADD ONE TO J.
008290  IF J GREATER THAN 20 GO TO 120-LAST-ORDER.
008310  ADD 1 TO J-DSP.
008330  GO TO 110-NEXT-ORDER.
008350  120-LAST-ORDER.
008370  MOVE J-DSP TO LAST-ORD-PK02.
008390  MOVE 226 TO SOFILE-RECORD-SIZE.
008410  MOVE 15 TO SOFILE-ROW.
008430  150-WRITE.
008450  COPY SQWRIT REPLACING SQWRIT001 BY SOFILE-PREFIX
008470  SQWRIT002 BY FRAME-1-OUTPUT.
008490  MOVE SPACES TO FRAME-1-OUTPUT.
008510  GO TO 040-READ.
008530  200-FRAME-2.
008550  MOVE ZERO TO K-FLAG.
008570  MOVE ONE TO J.
008590  MOVE "01" TO J-DSP.
008610  MOVE ONE TO K.
008630  MOVE "ST" TO TRUCK-TYPE.
083690  MOVE STRAN-CAPAC TO MAX-CAPAC.
083760  210-NEXT-TRUCK.
083900  MOVE ZERO TO J-FLAG.
084110  IF V-USE (J) EQUAL "S" MOVE "S" TO SAME-REG
084240  MOVE ZERO TO SEGONE-DIG
084300  GO TO 230-LAST-TRUCK.
084470  IF V-USE (J) EQUALS SPACES GO TO 226-INCREMENT-J.
087900  212-USE-TRUCK.
088110  IF CAPAC (J) EQUALS SPACES MOVE MAX-CAPAC
088180  TO CAPAC (J).
088350  IF CAPAC (J) NOT NUMERIC
088470  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
088590  ELSE PERFORM 213-DISP-VEH-NUM
088710  MOVE SEGONE-DIG TO V-NUMB (K)
088830  MOVE TRUCK-TYPE TO V-TYPE (K)
088950  MOVF "CAPAC NOT NUMERIC" TO EXPLANATION-2 (K)
089070  MOVE ONE TO J-FLAG
089240  ADD ONE TO K
089610  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
089680  IF CAPAC (J) NUMERIC
089800  IF CAPAC (J) GREATER THAN MAX-CAPAC
089920  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
090040  ELSE PERFORM 213-DISP-VEH-NUM
090160  MOVE SEGONE-DIG TO V-NUMB (K)
090280  MOVE TRUCK-TYPE TO V-TYPE (K)
090400  MOVE "CAPAC TOO LARGE" TO EXPLANATION-2 (K)
090520  MOVE ONE TO J-FLAG
090690  ADD ONE TO K
090810  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
090930  IF TIME-LIM (J) EQUALS SPACES
090990  MOVE "2x6" TO TIME-LIM (J).
091120  IF TIME-LIM (J) NOT NUMERIC
091240  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
091360  ELSE PERFORM 213-DISP-VEH-NUM
091480  MOVE SEGONE-DIG TO V-NUMB (K)
091600  MOVE TRUCK-TYPE TO V-TYPE (K)
091720  MOVE "TIME NOT NUMERIC" TO EXPLANATION-2 (K)
091840  MOVE ONE TO J-FLAG
091910  ADD ONE TO K
092130  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
092250  IF J-FLAG EQUALS SPACES GO TO 226-INCREMENT-J.
092370  MOVE "2x6" TO TIME-LIM (J).
092490  IF TIME-LIM (J) NOT NUMERIC
092610  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
092730  ELSE PERFORM 213-DISP-VEH-NUM
092850  MOVE SEGONE-DIG TO V-NUMB (K)
092970  MOVE TRUCK-TYPE TO V-TYPE (K)
093090  MOVE "TIME NOT NUMERIC" TO EXPLANATION-2 (K)
093210  MOVE ONE TO J-FLAG
093380  ADD ONE TO K
093500  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
093620  IF J-FLAG EQUALS ONE GO TO 226-INCREMENT-J.
093740  MOVE "2x6" TO TIME-LIM (J).
093860  IF TIME-LIM (J) NOT NUMERIC
093980  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
094100  ELSE PERFORM 213-DISP-VEH-NUM
094220  MOVE SEGONE-DIG TO V-NUMB (K)
094340  MOVE TRUCK-TYPE TO V-TYPE (K)
094460  MOVE "TIME NOT NUMERIC" TO EXPLANATION-2 (K)
094580  MOVE ONE TO J-FLAG
094700  ADD ONE TO K
094920  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
094930  IF J-FLAG EQUALS SPACES GO TO 226-INCREMENT-J.
094990  MOVE "2x6" TO TIME-LIM (J).
095110  IF TIME-LIM (J) NOT NUMERIC
095230  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
095350  ELSE PERFORM 213-DISP-VEH-NUM
095470  MOVE SEGONE-DIG TO V-NUMB (K)
095590  MOVE TRUCK-TYPE TO V-TYPE (K)
095710  MOVE "TIME NOT NUMERIC" TO EXPLANATION-2 (K)
095830  MOVE ONE TO J-FLAG
095950  ADD ONE TO K
096170  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
096290  IF J-FLAG EQUALS SPACES GO TO 226-INCREMENT-J.
096410  MOVE "2x6" TO TIME-LIM (J).
096530  IF TIME-LIM (J) NOT NUMERIC
096650  IF K-FLAG EQUALS ONE GO TO 230-LAST-TRUCK
096770  ELSE PERFORM 213-DISP-VEH-NUM
096890  MOVE SEGONE-DIG TO V-NUMB (K)
097010  MOVE TRUCK-TYPE TO V-TYPE (K)
097130  MOVE "TIME NOT NUMERIC" TO EXPLANATION-2 (K)
097250  MOVE ONE TO J-FLAG
097370  ADD ONE TO K
097590  IF K GREATER THAN 8 MOVE ONE TO K-FLAG.
097710  IF J-FLAG EQUALS SPACES GO TO 226-INCREMENT-J.
009750 MOVE CAPAC (J) TO TRUCK-CAPAC (KCHECK).
009770 MOVE TIME-LIM (J) TO TRUCK-TIME-LIM (KCHECK).
009790 GO TO 220-INCREMENT-J.
009810 218-NEW-TRUCK.
009830 ADD ONE TO KLAST.
009850 MOVE BASE-NR TO TRUCK-NUM (KLAST).
009870 MOVE CAPAC (J) TO TRUCK-CAPAC (KLAST).
009890 MOVE TIME-LIM (J) TO TRUCK-TIME-LIM (KLAST).
009910 IF V-USE (J) EQUALS ***
009930 MOVE "TC" TO TRUCK-CAPAC (KLAST).
009950 MOVE "TC" TO TRUCK-TIME-LIM (KLAST).
009970 IF V-USE (J) EQUALS ***
009990 MOVE "TC" TO TRUCK-CAPAC (KLAST).
009950 MOVE "TC" TO TRUCK-TIME-LIM (KLAST).
009970 ELSE MOVE SPACE TO TRK-USE (KLAST).

100020 220-INCREMENT-J.
010040 ADD ONE TO J.
010060 ADD 1 TO J-1SP.
010080 IF J GREATER THAN 12
010100 MOVE TRANS-CAPAC TO MAX-CAPAC
010120 MOVE "TC" TO TRUCK-TYPE.
010140 IF J GREATER THAN 18
010160 MOVE TRAILER-CAPAC TO MAX-CAPAC
010180 MOVE "TC" TO TRUCK-TYPE.
010200 IF J GREATER THAN 30 GO TO 230-LAST-TRUCK.
010220 GO TO 210-NEXT-TRUCK.
010240 230-LAST-TRUCK.
010260 MOVE TRUCK-TYPE TO LAST-TYPE.
010280 MOVE SECOND-DIG TO LAST-NUM.
010300 MOVE 191 TO SOFILE-RECORD-SIZE.
010320 MOVE 17 TO SOFILE-ROW.
010340 253-WRITE.
010360 COPY SQWRITE REPLACING SQWRITE1 BY SOFILE-PREFIX
010380 SQWRITE2 BY FRAME-2-OUTPUT.
010400 MOVE SPACE'S TO FRAME-2-INPUT.
010420 GO TO 046-READ.
010440 300-FRAME-3.
010460 MOVE ONE TO K.
010480 IF I-DATE NOT NUMERIC
010500 MOVE "DATE" TO INPUT-ERROR (K)
010520 MOVE "NOT NUMERIC" TO EXPLANATION-3 (K)
010540 ADD ONE TO K.
010560 IF START-TIME NOT NUMERIC
010580 MOVE "TIME" TO INPUT-ERROR (K)
010600 MOVE "NOT NUMERIC" TO EXPLANATION-3 (K)
010620 ADD ONE TO K.
010640 IF MAX-ROUTE EQUALS SPACES MOVE "240G" TO MAX-ROUTE.
010660 IF MAX-ROUTF NOT NUMERIC
010680 MOVE "MAXROUT" TO INPUT-ERROR (K)
010700 MOVE "NOT NUMERIC" TO EXPLANATION-3 (K)
010720 ADD ONE TO K.
010740 IF K EQUALS CM MOVE 1-DATE TO 0-DATE
010760 MOVE START-TIME TO 0-SHIFT
010780 MOVE OPTION TO 0-SHIFT
010800 MOVE MAX-ROUTE TO C-ALL-LIM
010820 MOVE "NO ERRORS" TO EXPLANATION-3 (K).
010840 MOVE 96 TO SOFILE-RECORD-SIZE.
010860 MOVE 11 TO SOFILE-ROW.
010770 350-WRITE.
010790 CPY SOWRIT REPLACING SOWRIT001 BY SOFILE-PREFIX.
010810 SOWRIT002 TO FRAMES-OUTPUT.
010830 MOVE SPACE'S TO FRAME-3-OUTPUT.
010850 GO TO C40-READ.
010870 690-EMERGENCY-INPUT.
010890 MOVE "L" TO EMR-CHECK.
010910 MOVE "0" TO M-OPTS.
010930 IF M-OPTION EQUALS "Y" MOVE "1" TO M-OPTS.
010950 MOVE "2" TO F-OPTS.
010970 IF F-OPTION EQUALS "Y" MOVE "1" TO F-OPTS.
011010 MOVE "3" TO F-OPTS.
011030 MOVE M-OPTS TO OUT-OPTION.
011050 MOVE M-OPTS TO OUT-MIX.
011070 MOVE "A" TO SOFILE-RECALC-SIZE.
011090 MOVE "A" TO SOFILE-PREFIX.
011110 650-WRITE.
011130 CPY SOWRIT REPLACING SOWRIT001 BY SOFILE-PREFIX.
011150 SOWRIT002 TO FRAME-6-OUTPUT.
011170 MOVE SPACE'S TO FRAME-6-OUTPUT.
011190 GO TO Q40-READ.
011210 730-HISTORY.
011230 MOVE "H" TO LIST-CHECK.
011250 MOVE FCN-TYPE TO FCN.
011270 MOVE ZEROS TO ERR-CNT.
011290 IF FCN-TYPE EQUALS "2" GO TO 750-FUNCTION-2.
011310 IF FCN-TYPE EQUALS "1"
011330 MOVE ZEROS TO START-DATE.
011350 MOVE ZEROS TO END-DATE.
011370 MOVE ZEROS TO M-SHIFT.
011390 IF START-DATE NOT NUMERIC
011410 ADD ONE TO ERR-CNT.
011430 IF EXPLANATION-. ERR-CNT
011450 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011470 IF END-DATE NOT NUMERIC
011490 ADD ONE TO ERR-CNT.
011510 MOVE "END DATE NOT NUMERIC”.
011530 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011550 IF START-DATE NOT= END-DATE
011570 MOVE ZEROS TO M-SHIFT.
011590 IF M-SHIFT NOT NUMERIC
011610 ADD ONE TO ERR-CNT.
011630 MOVE "SHIFT NOT NUMERIC".
011650 IF EXPLANATION-. ERR-CNT
011670 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011690 GO TO 795-DISPLAY-ERRORS.
011710 MOVE START-DATE TO DATE1.
011730 MOVE END-DATE TO DATE2.
011750 MOVE M-SHIFT TO SHIFT-ALL.
011770 IF M-OPTION EQUALS “Y”
011790 MOVE "1" TO OPTION-M ELSE MOVE "0" TO OPTION-M.
011810 GO TO 795-DISPLAY-ERRORS.
011830 750-FUNCTION-2.
011746 MOVE ERRORS TO JJ
011810 MOVE ERRORS TO ERR-CNT.
011830 IF TODAY-DAT NOT NUMERIC
011850 AND ONE TO ERR-CNT
011870 MOVE "DAT NOT NUMERIC" TO EXPLANATION-4 (ERR-CNT)
011890 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011910 IF TODAY-SHIFT NOT NUMERIC
011930 AND ONE TO ERR-CNT
011950 MOVE "SHIFT NOT NUMERIC" TO EXPLANATION-4 (ERR-CNT)
011970 MOVE ERR-CNT TO INPUT-ERR (ERR-CNT).
011990 IF ERR-CNT GREATER THAN ZERO GO TO 799-DISPLAY-ERRORS.
012010 MOVE TODAY-DAT TO DATE-SAVE
012030 MOVE TODAY-SHIFT TO SHIFT-SAVE.
012050 790-INCREMENT.
012070 IF LLAST EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012090 ADD ONE TO JJ.
012110 IF JJ GREATER THAN 6
012130 GO TO 799-DISPLAY-ERRORS.
012150 IF ORIG-JJ EQUALS SPACES GO TO 799-DISPLAY-ERRORS.
012170 MOVE ORIG (JJ) TO TEST-NAME.
012190 ADD ONE TO ERR-CNT
012210 MOVE "ORIGIN MISMATCH" TO EXPLANATION-4 (ERR-CNT)
012230 MOVE JJ TO INPUT-ERR (ERR-CNT).
012250 IF ERR-CNT EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012270 MOVE DEST (JJ) TO TEST-NAME.
012290 PERFORM 500-NAME-TEST.
012310 IF TEST-NAME NOT EQUAL ORIG (JJ)
012330 ADD ONE TO ERR-CNT
012350 MOVE "DEST MISMATCH" TO EXPLANATION-4 (ERR-CNT)
012370 MOVE JJ TO INPUT-ERR (ERR-CNT).
012390 IF ERR-CNT EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012410 IF SIZEW (JJ) NOT NUMERIC
012430 ADD ONE TO ERR-CNT
012450 MOVE "SIZE NOT NUMERIC" TO EXPLANATION-4 (ERR-CNT)
012470 MOVE JJ TO INPUT-ERR (ERR-CNT).
012490 IF ERR-CNT EQUALS 6 GO TO 799-DISPLAY-ERRORS.
012510 IF ERR-CNT GREATER THAN ZERO
012530 GO TO 760-INCREMENT.
012550 ADD ONE TO LLAST.
012570 MOVE ORIG (JJ) TO ORIGS (LLAST)
012590 MOVE CESE (JJ) TO CESES (LLAST)
012610 MOVE SIZEW (JJ) TO SIZES (LLAST).
012630 MOVE VEWH (JJ) TO VEWS (LLAST).
012650 IF EPER (JJ) EQUALS "Y"
012670 MOVE "Y" TO EMERS (LLAST) ELSE
012690 MOVE "Y" TO EMES (LLAST).
012710 IF PPLEW (JJ) EQUALS "Y"
012730 MOVE "Y" TO PPLES (LLAST) ELSE
012750 MOVE "Y" TO PPLES (LLAST).
012770 GO TO 760-INCREMENT.
012790 799-DISPLAY-ERRORS.
012810 IF ERR-CNT EQUALS ZEROS
012830 MOVE "NO ERRORS" TO EXPLANATION-4 (1)
012850 MOVE ZER0CS TO INPUT-ERR (11).
012870 MOVE 226 TO SOFILE-RELOAD-SIZE.
012890 MOVE 16 TO SOFILE-ROW.
012910 799-WRITE.
012920 CPY SOWIT REPLACING SQWT001 BY SOFILE-PREFIX
012930 SQWRT002 BY FRAME-7-OUTPUT.
012950 MOVE SPACES TO FRAME-7-OUTPUT.
012970 GO TO 040-READ.
013010 400-OUTPUT.
013030 IF HIST-CHECK EQUALS "4" GO TO 400-HISTORY-CARDS.
013050 IF EMER-CHECK NOT EQUAL "E" GO TO 400-OUTPUT-REGULAR.
013070 OPEN OUTPUT VSZIN.
013090 MOVE ONE TO J.
013110 410-ORDER-OUTPUT-E.
013130 MOVE O-SHIFT TO O-TIME.
013150 MOVE OUT-SIZE (J) TO O-SIZ.
013170 MOVE OUT-JRIG (J) TO O-ORIGIN.
013190 MOVE OUT-DEST (J) TO O-DESTIN.
013210 MOVE ORDER-CARD-E TO CARD-IMAGE.
013230 WRITE CARD-IMAGE.
013250 ADD ONE TO J.
013270 IF J GREATER THAN JLAST NEXT SENTENCE
013290 ELSE GO TO 410-ORDER-OUTPUT-E.
013310 MOVE END-CARD TO CARD-IMAGE.
013330 WRITE CARD-IMAGE.
013350 MOVE M-OPTS TO O-MIX.
013370 MOVE P-OPTS TO PROF1.
013390 MOVE OPT-CARD-E TO CARD-IMAGE.
013410 WRITE CARD-IMAGE.
013430 MOVE ZERO TO O-NUMB-STS.
013450 MOVE JERC TO O-NUMB-TRS.
013470 MOVE ZERO TO O-NUMB-TTS.
013490 IF SAME-FEG EQUALS "S"
013510 MOVE "A" TO O-NUMB-STS.
013530 MOVE TRUCK-HEADER-CARD TO CARD-IMAGE.
013550 WRITE CARD-IMAGE.
013570 IF SAME-FEG EQUALS "S"
013590 GO TO 421-END-CARD.
013610 MOVE ONE TO J.
013630 420-TRUCK-OUTPUT-E.
013650 MOVE TRK-USE (J) TO TRK-USE-IT
013670 MOVE TRUCK-NUM (J) TO O-TRUCK-CODE.
013690 MOVE TRUCK-CAPAC (J) TO O-TRUCK-CAPAC.
013710 MOVE TRUCK-CARR TO CARD-IMAGE.
013730 WRITE CARD-IMAGE.
013750 ADD ONE TO J.
013770 IF J GREATER THAN JLAST GO TO 421-END-CARD
013790 ELSE GO TO 420-TRUCK-OUTPUT-E.
013815 421-END-CARD.
013830 CLOSE VSZIN WITH RELEASE.
013850 ZIP AVS2-EXECUTE.
013870 GO TO 040-READ.
013890 400-OUTPUT-REGULAR.
013910 OPEN OUTPUT VSZIN.

135
MOVE DATE-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
MOVE SHIFT-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
MOVE OPT-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
MOVE ONE TO J.
MOVE 410-ORDER-OUTPUT.
MOVE OUT-SIZE (J) TO O-SIZE.
MOVE OUT-ORIG (J) TO O-ORIGIN.
MOVE OUT-DEST (J) TO O-DESTIN.
MOVE ORDER-CARD TO CARD-IMAGE.
WRITE CARD-IMAGE.
ADD ONE TO J.
IF J GREATER THAN LAST NEXT SENTENCE
ELSE GO TO 410-ORDER-OUTPUT.
WRITE CARD-IMAGE.
MOVE ZEROS TO O-NUMB-STS-R
MOVE ZEROS TO O-NUMB-TRN-R
MOVE ZEROS TO O-NUMB-TTS-R
MOVE TRUCK-FAILED-CARD-R TO CARD-IMAGE.
WRITE CARD-IMAGE.
ADD ONE TO J.
IF J GREATER THAN LAST NEXT SENTENCE
ELSE GO TO 420-TRUCK-OUTPUT.
CLOSE VS2IN WITH RELEASE.
ZIP VS2-EXECUTE.
GO TO 040-READ.
OPEN OUTPUT VS3IN.
MOVE FCN-ALL-CARD TO CARD-IMAGE2.
WRITE CARD-IMAGE2.
IF FCN ACT EQUL "2"
GO TO 410-RUN-AVS3.
MOVE ZEROS TO JJ.
05-INCREMENT.
ADD ONE TO JJ.
IF JJ GREATER THAN LAST
GO TO 410-RUN-AVS3.
MOVE ORIGS (JJ) TO ORIG-SAVE
MOVE DESTS (JJ) TO DEST-SAVE
MOVE SIZES (JJ) TO SIZE-SAVE
MOVE ETCS (JJ) TO EMERG-SAVE
MOVE RFLRS (JJ) TO RPE-SAVE.
MOVE VEHs (JJ) TO VEH-NU.
015598 MOV ECA-2-CARD TO CARD-IMAGE2.
015616 WRITL CARD-IMAGE2.
015630 GC TO 405-INCREMENT.
015650 410-RUN-AVS3.
015670 CLOSE WS3IN WITH RELEASE.
015690 ZIP WS3X-EXECUTE.
015710 GO TO 040-READ.
015730 500-NAMF-TEST SECTION.
015750 510-START.
015770 MOVE ONE TO M.
015790 520-LIST-LOOP.
015810 IF TEST-NAME EQUALS W-NAME (M) GC TO 530-EXIT.
015830 ADD ONE TO M.
015850 IF M GREATER THAN 87 GO.
015870 MOVE SPACES TO TEST-NAME.
015890 GO TO 530-EXIT.
015910 GO TO 520-LIST-LOOP.
015930 530-EXIT.
015950 EXIT.
015970 599-CLEAR-RESTART SECTION.
015990 599-START.
015910 MOVE SPACES TO SAME-REG.
015930 MOVE SPACES TO EIER-CHECK.
015950 MOVE SPACES TO KEY-CHECK.
015970 MOVE ONE TO J.
015990 MOVE ONE TO K.
016010 MOVE ONE TO JJ.
016030 599-LOOP.
016050 MOVE SPACES TO ORIGS(JJ).
016070 MOVE SPACES TO DES1S(JJ).
016090 MOVE SPACES TO SPES(JJ).
016110 MOVE SPACES TO EMRS(JJ).
016130 MOVE SPACES TO RPLES(JJ).
016150 MOVE SPACES TO VEMS(JJ).
016170 ADD ONE TO JJ.
016190 IF JJ GREATER THAN 6 GO TO 599-LOOP ELSE GO TO 598-LOOP.
016210 598-LOOP.
016230 IF ORDER-OUTPUT (J) NOT EQUAL SPACES
016250 MOVE SPACES TO ORDER-OUTPUT (J).
016270 ADD ONE TO J.
016290 IF J GREATER THAN 99 GO TO 031-INITIALIZE.
016310 IF K GREATER THAN 50 GO TO 599-LOOP.
016330 IF TRUCK-OUTPUT (K) NOT EQUAL SPACES
016350 MOVE SPACES TO TRUCK-OUTPUT (K).
016370 ADD ONE TO K.
016390 GO TO 599-LOOP.
016410 CLOSE-FILE SECTION 94.
016430 05-CLOSE-IN.
016450 COPY SUCLTH.
016470 06-SQCLIN.
016490 06-CLOSE-OUT.
015910 GO TO I-EDJ.
015930 OPEN-SURK SECTION 51.
015950 07-OPEN-SURK.
015970 COPY IOPWR.

137
0159y6 BASE-SURA SECTION.
016010 BY-READ-SURB.
016630 COPY SURED1 REPLACING SURED133 BY 05-CLOSE-IN.
016650 G9-WRITE-SURB.
016670 COPY SUCUT1 REPLACING SUCUT1032 BY 16-OUTPUT-ERROR.
016690 SNUH-SURF SECTION 52.
016110 10-CHM-INPUT.
016130 COPY SOPEN2.
016150 11-DCP-OUTPUT.
016170 COPY SOOUT2.
016190 DUMMY-PARA SECTION.
016210 TOTAL.
016230 SOOUT3.
016250 CLOSE-SURF SECTION 54.
016270 15-CLOSE-SURF.
016290 COPY ILC65R.
016310 ERR-PROC SECTION 55.
016330 16-OUTPUT-ERKOM.
016350 MOVE "CPUT ERR" TO 1-ABORT-MSG.
016370 TRACZ ZC.
016390 GO TO 1-ABORT-
016410 COMMON-SURF SECTION 34.
016430 17-COMMON-SLE4.
016450 COPY ICCMCN.
016470 ENN-OF-JOB.
APPENDIX D - AVS 3, HISTORY/UPDATE, FLOWCHARTS AND LISTINGS
MAIN PROGRAM

START

READ UNIT 5
FUNC, DATE 1, DATE 2, SHIFT, OPTION

FUNC = 1

OPTION .NE. 0
Y
NEWOR = .F.
N
NEWOR = .T.

CALL AUSRD (NEWOR, IDATE, SHIFT)
CALL CRDRD

CALL RPRT (IDATE, IDATE, SHIFT)
STOP

CALL MRGIT (DATE 1, DATE 2)
STOP

CALL RPRT (DATE 1, DATE 2, SHIFT)
STOP

FUNC > 4

PRINT ERROR MSG
STOP

FUNC = 2

FUNC = 3

FUNC = 4

STOP
PROGRAM START. 7X74 CRT=2 ROUNDS/TRACE FIN 4,6,6..,0 10/15/80 12.29.65

1 IDENT AVS3
2 $ # IDENT=7
3 SEGMENT AVS3,=MARK,=GROUP,=DROP,=DROP,=DROP,=DROP
4 FILE 7=FILE,=UNIT=DISK,=BLOCKING=1,RECORD=12
5 FILE 13=FILE,=UNIT=DISK,=BLOCKING=1,RECORD=12
6 FILE 2=FILE,=UNIT=DISK,=BLOCKING=1,RECORD=80

1 2 CL SHIFT
2 INTERFACE G-R2
3 LOGICAL UPD,=UPD3
4 INTEGER DAT=1,DATEZ,FUNC,=OPTION,=DATEY1,=DATEY2,=DATEZ,=INSHF,=OPTION
5 READ (5,1000) FUNC,=DATE1,=DATE2,=INSHF,=OPTION
6 FUNC=(1.0,1.0,1.0,1.0,1.0,1.0)
7 PAINT 1100
8 IF (S,="INVALID OPTION=PROGRAM TERMINATED") STOP
9 IF (S,="FILE NOT OPEN") STOP
10 IF (S,="FILE CLOSED") STOP
11 IF (S,="FILE NOT OPEN") STOP
12 IF (S,="FILE CLOSED") STOP
13 IF (S,="FILE NOT OPEN") STOP
14 IF (S,="FILE CLOSED") STOP
15 IF (S,="FILE NOT OPEN") STOP
16 IF (S,="FILE CLOSED") STOP
17 IF (S,="FILE NOT OPEN") STOP
18 IF (S,="FILE CLOSED") STOP
19 IF (S,="FILE NOT OPEN") STOP
20 IF (S,="FILE CLOSED") STOP
21 IF (S,="FILE NOT OPEN") STOP
22 IF (S,="FILE CLOSED") STOP
23 IF (S,="FILE NOT OPEN") STOP
24 IF (S,="FILE CLOSED") STOP
25 IF (S,="FILE NOT OPEN") STOP
26 IF (S,="FILE CLOSED") STOP
27 IF (S,="FILE NOT OPEN") STOP
28 IF (S,="FILE CLOSED") STOP
29 IF (S,="FILE NOT OPEN") STOP
30 IF (S,="FILE CLOSED") STOP
31 IF (S,="FILE NOT OPEN") STOP
32 IF (S,="FILE CLOSED") STOP
33 IF (S,="FILE NOT OPEN") STOP
34 IF (S,="FILE CLOSED") STOP
35 IF (S,="FILE NOT OPEN") STOP
36 IF (S,="FILE CLOSED") STOP
37 IF (S,="FILE NOT OPEN") STOP
38 IF (S,="FILE CLOSED") STOP
39 IF (S,="FILE NOT OPEN") STOP
40 IF (S,="FILE CLOSED") STOP
41 IF (S,="FILE NOT OPEN") STOP
42 IF (S,="FILE CLOSED") STOP
43 IF (S,="FILE NOT OPEN") STOP
44 IF (S,="FILE CLOSED") STOP
45 IF (S,="FILE NOT OPEN") STOP
46 IF (S,="FILE CLOSED") STOP
47 IF (S,="FILE NOT OPEN") STOP
48 IF (S,="FILE CLOSED") STOP
49 IF (S,="FILE NOT OPEN") STOP
50 IF (S,="FILE CLOSED") STOP
51 IF (S,="FILE NOT OPEN") STOP
52 IF (S,="FILE CLOSED") STOP
53 IF (S,="FILE NOT OPEN") STOP
54 IF (S,="FILE CLOSED") STOP
55 IF (S,="FILE NOT OPEN") STOP
56 IF (S,="FILE CLOSED") STOP
57 IF (S,="FILE NOT OPEN") STOP
58 IF (S,="FILE CLOSED") STOP
59 IF (S,="FILE NOT OPEN") STOP
60 IF (S,="FILE NOT OPEN") STOP
61 IF (S,="FILE NOT OPEN") STOP
62 IF (S,="FILE NOT OPEN") STOP
63 IF (S,="FILE NOT OPEN") STOP
64 IF (S,="FILE NOT OPEN") STOP
65 IF (S,="FILE NOT OPEN") STOP
66 IF (S,="FILE NOT OPEN") STOP
67 IF (S,="FILE NOT OPEN") STOP
68 IF (S,="FILE NOT OPEN") STOP
69 IF (S,="FILE NOT OPEN") STOP
70 IF (S,="FILE NOT OPEN") STOP
71 IF (S,="FILE NOT OPEN") STOP
72 IF (S,="FILE NOT OPEN") STOP
73 IF (S,="FILE NOT OPEN") STOP
74 IF (S,="FILE NOT OPEN") STOP
75 IF (S,="FILE NOT OPEN") STOP
76 IF (S,="FILE NOT OPEN") STOP
77 IF (S,="FILE NOT OPEN") STOP
78 IF (S,="FILE NOT OPEN") STOP
79 IF (S,="FILE NOT OPEN") STOP
80 IF (S,="FILE NOT OPEN") STOP
81 IF (S,="FILE NOT OPEN") STOP
82 IF (S,="FILE NOT OPEN") STOP
83 IF (S,="FILE NOT OPEN") STOP
84 IF (S,="FILE NOT OPEN") STOP
85 IF (S,="FILE NOT OPEN") STOP
86 IF (S,="FILE NOT OPEN") STOP
87 IF (S,="FILE NOT OPEN") STOP
88 IF (S,="FILE NOT OPEN") STOP
89 IF (S,="FILE NOT OPEN") STOP
90 IF (S,="FILE NOT OPEN") STOP
91 IF (S,="FILE NOT OPEN") STOP
92 IF (S,="FILE NOT OPEN") STOP
93 IF (S,="FILE NOT OPEN") STOP
94 IF (S,="FILE NOT OPEN") STOP
95 IF (S,="FILE NOT OPEN") STOP
96 IF (S,="FILE NOT OPEN") STOP
97 IF (S,="FILE NOT OPEN") STOP
98 IF (S,="FILE NOT OPEN") STOP
99 IF (S,="FILE NOT OPEN") STOP
100 IF (S,="FILE NOT OPEN") STOP
101 IF (S,="FILE NOT OPEN") STOP
102 IF (S,="FILE NOT OPEN") STOP
103 IF (S,="FILE NOT OPEN") STOP
104 IF (S,="FILE NOT OPEN") STOP
105 IF (S,="FILE NOT OPEN") STOP
106 IF (S,="FILE NOT OPEN") STOP
107 IF (S,="FILE NOT OPEN") STOP
108 IF (S,="FILE NOT OPEN") STOP
109 IF (S,="FILE NOT OPEN") STOP
110 IF (S,="FILE NOT OPEN") STOP
111 IF (S,="FILE NOT OPEN") STOP
112 IF (S,="FILE NOT OPEN") STOP
113 IF (S,="FILE NOT OPEN") STOP
114 IF (S,="FILE NOT OPEN") STOP
115 IF (S,="FILE NOT OPEN") STOP
116 IF (S,="FILE NOT OPEN") STOP
117 IF (S,="FILE NOT OPEN") STOP
118 IF (S,="FILE NOT OPEN") STOP
119 IF (S,="FILE NOT OPEN") STOP
120 IF (S,="FILE NOT OPEN") STOP
121 IF (S,="FILE NOT OPEN") STOP
122 IF (S,="FILE NOT OPEN") STOP
123 IF (S,="FILE NOT OPEN") STOP
124 IF (S,="FILE NOT OPEN") STOP
125 IF (S,="FILE NOT OPEN") STOP
126 IF (S,="FILE NOT OPEN") STOP
127 IF (S,="FILE NOT OPEN") STOP
128 IF (S,="FILE NOT OPEN") STOP
129 IF (S,="FILE NOT OPEN") STOP
130 IF (S,="FILE NOT OPEN") STOP
131 IF (S,="FILE NOT OPEN") STOP
132 IF (S,="FILE NOT OPEN") STOP
133 IF (S,="FILE NOT OPEN") STOP
134 IF (S,="FILE NOT OPEN") STOP
135 IF (S,="FILE NOT OPEN") STOP
136 IF (S,="FILE NOT OPEN") STOP
137 IF (S,="FILE NOT OPEN") STOP
138 IF (S,="FILE NOT OPEN") STOP
139 IF (S,="FILE NOT OPEN") STOP
140 IF (S,="FILE NOT OPEN") STOP
141 IF (S,="FILE NOT OPEN") STOP

141
AVSRD

DEV1 8
UPDT  F

READ SCHED1
FILE
(DATE, SHFT,
MARK, BS)

REFORMAT
DATE FROM
SCHED1 FILE

YRMODA = DATE

CALL DTFND
(DATE, SHFT,
MARK)

I1 = 0

DO LOOP
I = 1,50

SET NAME TO
1th TRUCK NAME

POINTER
TO 1th TRUCK
ROUTE EQ 0

V
A

142
SET IPTR TO VALUE OF POINTER

ORG = ORIGN (IPTR)
DEST = TERMIN (IPTR)
PAL = LGIZE (IPTR)
EMERG = 0

ORDER = EQ. 999

ORDER # LE. 0

EMERG = 1

CALL UPDATE

SET IPTR TO NEXT ITEM IN LIST

IPTR NE. 0

I EQ. 50

Y

N

Y

N

143
SHIFT OF HISTORY FILE REC EQ D
I1 NE 0
Y
N
I1 = 1
POINTER TO UNASSIGNED ORDERS EQ 0
Y
N
SET IPTR TO VALUE OF POINTER TO UNASSIGNED ORDERS
NAME = "ZZZZ"
1 GT 50
Y
N
NEWORD EQ T.
Y
N
G
H
POINTER TO UNASSIGNED ORDERS EQ 0
Y
F
D
E
PRINT HEADER FOR LIST OF UNDELIVERED ORDERS

SET IPTR TO TOP OF UNASSIGNED ORDERS

WRITE ORDER NUMBER, SIZE, ORIGIN, DESTINATION TO TAPER AND PRINTER

SET IPTR TO NEXT ORDER IN LIST

IF TR HD O

REWIND DEV1

READ RECORD FROM DEVC (DEV1)

END OF FILE?

DATE OF RECORD READ TO EO

I K

M

PRINT HEADER
SUBROUTINE AVS3N 7=74 OPT=0 ROUND=*// TRACE FTM 4.6460 10/15/80 12.23.05

100 CONTINUE
READ(7,1100) (PTSOR(I),I=1,50) AVS3 144
60 READ(7,1750) (FJUNK(I),I=1,14) AVS3 117
1750 FORMAT(3F6.1) AVS3 118
READ(7,1600) (FJUNK(I),I=1,12) AVS3 119
1600 FORMAT(4F8/16/2I6) AVS3 120
65 READ(7,1400) SPP, IDATE AVS3 121
1300 FORMAT(F6.1,1B) AVS3 122
READ(7,1800) PARK AVS3 123
1400 FORMAT(16) AVS3 124
C ORGANIZE DATA IN FORMAT DATE, TRUCKNAME, ORIGIN, DESTINATION, LOGSIZE
76 C EMERGENCY=1, EMERGENCY=0 REGULAR
IF(IPTR.NE.1) CALL UPTOF(IPT1) AVS3 125
N1000 IF(IPTR.EQ.0) GO TO 700 AVS3 126
N1000 IF(IPTR.NE.0) GO TO 200 AVS3 127
C GET FIRST ELEMENT IN CHAIN
IPTR=PTSOR(1) AVS3 128
20 DOJ=1,DIM(IPTR) AVS3 129
DLST=TcH(IMIPTR) AVS3 130
P=LSIZE(IMIPTR) AVS3 131
EMER=0 AVS3 132
IF(ONUM(IPTR,JL).LT.59) GO TO 145 AVS3 133
IF(ONUM(IPTR,JL).EQ.1) EMER=1 AVS3 134
AVS3 135
C CALL UPDATE/Routine-VARIABLES PASSED THROUGH AVS3N IN COMMON
AVS3 136
CALL UPTOF AVS3 137
145 IPTR=LFND(IPTR) AVS3 138
IF(IPTR.NE.1) GO TO 200 AVS3 139
20 CONTINUE AVS3 140
AVS3 141
95 IF(UNOT=NULL) GO TO 300 AVS3 142
AVS3 150
300 IF(UNOT.HTL.30) AVS3 151
AVS3 152
AVS3 153
AVS3 154
AVS3 155
AVS3 156
AVS3 157
AVS3 158
AVS3 159
AVS3 160
AVS3 161
AVS3 162
AVS3 163
AVS3 164
AVS3 165
AVS3 166
AVS3 167
AVS3 168
AVS3 169
AVS3 170
AVS3 171
AVS3 172
AVS3 173
AVS3 174
AVS3 175
AVS3 176
AVS3 177
148
SUBROUTINE AVS3

115 25 WRITE(12,3000)I(NMB(IPTR),LSIZE(IPTR),ORIGIN(IPTR),TERMN(IPTR)) AVS3 171
3000 FORMAT(12,3246) AVS3 172
WRITE(14,3300)I(NMB(IPTR),LSIZE(IPTR),ORIGIN(IPTR),TERMN(IPTR)) AVS3 173
3305 FORMAT(I15.5X,15,5X,48X) AVS3 174
CONTINUE
120 IF(I(FILMK.EQ.0)) GO TO 28 AVS3 175
GOTO 30 AVS3 176
28 WRITE(14,3200) AVS3 177
3200 FORMAT("ALL ORDERS SCHEDULED FOR DELIVERY") AVS3 178
C END AVS3 INPUT

125 K=WIND-1
SHIFT=ISHFT
35 READDEVL=150,D,ENC*100,TOTE,TSHT,T,MARK,TNAME,TRIG,TOSTN,
1 TREGF,TEPAL AVS3 181
1500 FORMAT(15E6.11,5E4.16CA4.1000) AVS3 182
150 IF(I(FIT,DATE,EQ.0)) GOTO 100 AVS3 183
150 IF(I(FIT,DATE,GT.DATE)) GOTO 40 AVS3 184
150 IF(I(FIT,DATE,LT.DATE)) GOTO 40 AVS3 185
30 WRITE(11,1500)DATE,TSHT,TMARK,TNAME,TRIG,TOSTN,TRIG,F,TEPAL AVS3 186
30 IF(TMARK,=.EQ.,DATE) GOTO 78 AVS3 187
30 IF(TMARK,GT.DATE) GOTO 78 AVS3 188
30 IF(TMARK,LT.DATE) GOTO 78 AVS3 189
30 DATE=0 AVS3 190
GOTO 35 AVS3 191
40 IF(I(FIT,DATE,GT.DATE)) GOTO 50 AVS3 192
WRITE(11,1500)DATE,TSHT,TMARK,TNAME,TRIG,TOSTN,TRIG,F,TEPAL AVS3 193
40 IF(I(FIT,DATE,GT.DATE)) GOTO 50 AVS3 194
GOTO 35 AVS3 195
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 196
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 197
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 198
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 199
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 200
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 201
GOTO 35 AVS3 202
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 203
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 204
GOTO 35 AVS3 205
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 206
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 207
GOTO 35 AVS3 208
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 209
50 IF(TMARK,GT.ISHFT) GOTO 30 AVS3 210
GOTO 35 AVS3 211
100 IF(DATE,.LT.,DATE) WRITE(11,1500)DATE,TSHT,TMARK,TNAME,TRIG
1 TOSTN,TRIG,F,TEPAL AVS3 212
100 IF(DATE,.LT.,DATE) WRITE(11,1500)DATE,TSHT,TMARK,TNAME,TRIG
1 TOSTN,TRIG,F,TEPAL AVS3 213
CALL ENOF411) AVS3 214
5000 CALL ERROR(200) AVS3 215
STOP AVS3 216
END AVS3 217

149
READ INPUT FROM UNIT 5

END-OF-FILE?

Y

N

REFORMAT DATE FROM INPUT (YIRMODA)

DATE EQ 0

Y

N

CALL RPLCE

CALL RPRT (DATE1, DATE2, IDATE=YIRMODA, SHIFT=SHIFT, MARK=0)

A
CALL DTFND (DATE, SHIFT, MARK)

Y

DATE EQ 0

CALL UPDATE

DATE1-DATE
DATE2=D2
SHIFT2=SHIFT

N

CALL UPDATE

D

DATE NE 0

CALL RPLCE

DATE 0

DATE1 NE 0

CALL RPRT (DATE1, DATE2, SHIFT2)
E

TRANSFORM FILE FROM DEV1 TO TAPE11

RETURN
SUBROUTINE DHFIT 74/74  OPT=0 ROUND=** TRACE  FTN 4, 6A460  10/15/80  12.23.05

1 SUBROUTINE DHFIT  
C ALPHA  
ALPHA INAME(55), ORG(50), OSM(50)  
C BINARY  
 INTEGER FZ  
 INTEGER DATE, REGP(50), EPAL(50)  
 REAL ISPHT  
 COMMON/HSFCT/DATE, ISPHT, IMARK, REGP, EPAL  
 COMMON/HSFCT2/ISPHT  
 DATE = 0  
 REMIND 11  
 1  F30.11, 10S6, END=6, DATE, ISPHT, IMARK, INAME, ORG(50), REGP, EPAL  
 F30.11  
 1200 FORMAT(*I16, F6.1)  
 1350 DATE = DATE % 7  
 1500 FORMAT(*I16, F6.1)  
 1600 DATE = DATE + 1  
 1700 IF DATE < 2 THEN DATE = 0  
 1800 GO TO 1500  
 2500 PRINT 2400  
 2600 FORMAT(*END OF FILE*)  
 3000 RETURN  
 3200 END  

1 SUBROUTINE DATAN (DATE, IN, IDA, IYR, MONT)  
C ALPHA  
ALPHA IS, DYTELES(121), MONT  
C BINARY  
 INTEGER FZ  
 INTEGER DATE, IN, IDA, IYR, MONT  
 DATA MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG/  
 1 JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP /  
 10 SEP, OCT, NOV, DEC /  
 19 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 19 SEP, OCT, NOV, DEC /  
 28 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 28 SEP, OCT, NOV, DEC /  
 37 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 37 SEP, OCT, NOV, DEC /  
 46 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 46 SEP, OCT, NOV, DEC /  
 55 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 55 SEP, OCT, NOV, DEC /  
 64 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 64 SEP, OCT, NOV, DEC /  
 73 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 73 SEP, OCT, NOV, DEC /  
 82 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 82 SEP, OCT, NOV, DEC /  
 91 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 91 SEP, OCT, NOV, DEC /  
 100 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 100 SEP, OCT, NOV, DEC /  
 109 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 109 SEP, OCT, NOV, DEC /  
 118 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 118 SEP, OCT, NOV, DEC /  
 127 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 127 SEP, OCT, NOV, DEC /  
 136 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 136 SEP, OCT, NOV, DEC /  
 145 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 145 SEP, OCT, NOV, DEC /  
 154 MONTES/1, JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP,  
 154 SEP, OCT, NOV, DEC /  

154
PRINT MSG "ATTEMPT TO UPDATE RECORD TWICE"

STOP

B

UPDT .EQ. .T.

Y

F

SET-UP NEW BLANK RECORD

RETURN

PRINT MSG "REQUESTED RECORD NOT FOUND"

DATE1=0

RETURN
SUBROUTINE DTFND 7474 CPI=8 ROUND=*" TRACE FTN 4.6+60 10/15/80 12.23.05

1  SUBROUTINE DTFND(date,shift,mark)
   ALPHANUMERIC
   ALPHA (name(50), orig(50), dstn(50))
   BINARY
   INTEGER date,carry(10), jsave, regp(50) , epal(50),
   locl, dev2, date1, shifts, ishift
   LOGICAL update, updt
   COMMON/ISRC1/car_02, ishift, regp, epal
   COMMON/ISRC2/shift
   COMMON/isrc3/name, orig, dstn
   COMMON/CTRL/LFD, carry, dev1, dev2, update
   RETURN
   END

10 READ dev1, iol, end=201 date, 02, ishift, ishift, name, orig, dstn, regp.
11 PRI
1200 FORMAT(2'I6, F6.1, I6, 50A1, 100A6, 100I6)
20 IF (date .eq. 0) GOTO 20
   IF (date .gt. 0) GOTO 30
   IF (.not. (date .ge. date, and date, le., date, .eq. 1) GOTO 10
   IF (ishift .eq. 0) RETURN
   IF (ishift .eq. 1) GOTO 10
   IF (update) RETURN
   IF (mark .eq. 1) RETURN
   IF (update) GOTO 20
   PRINT 2000
20 FORMAT(" ATTEMPT TO UPDATE RECORD TWICE WITH SAME DATA",
1." PROGRAM ABORTED")
   RETURN
23 IF (update) GOTO 40
30 DATE=DATE
   02=02
   ISHIFT=ISHIFT
   ISHIFT=ISHIFT
   JMARK=MARK
   Do 30 j=1, 50
   NAME(j)=10M
   ORIG(j)=10M
   DSTN(j)=10M
   P00(j)=0
30 RETURN
40 PRINT 3000, date, shift
3000 FORMAT(" NO RECORD FOUND IN HISTORY FILE FOR ENTRY WITH DATE ",
1." NAME AND SHIFT ", "F6.1)
   NAME=J
   NAME=N
   END
SUBROUTINE ERR

1

SUBROUTINE ERR(CRIPSGNO)
INTEGER MSGNO
PRINT 1000,MSGNO
1000 FORMAT('ERROR NUMBER ','I4
STOP
END

1

SUBROUTINE LNCF(ICEV)
INTEGER DATE,02,1PARK,REGP(50),EPAL(50)
REAL ISHT
COMMON/HISRC/DATE,02,1PARK,REGP,EPAL
COMMON/HISRC2/ISHT
COMMON/HISRC3/NAME,ORIG,OSTN
DATE=0
450.11.15)DATE,02,1PARK,1NAME,ORIG,OSTN,REGP,EPAL
500.1 FORMAT(21A,2E15.E9A4,200A6,1001m)
RETURN
END

158
MERGE

REWIND IDEV

READ RECORD FROM IDEV INTO DATE AREA

DATE TO 0

DATE2 LE D2 AND DATE2 GE DATE

DATE1 GE DATE AND DATE1 LE D2

A

DATE2 GE D2 AND DATE1 LE DATE

DATE1 GT D2

RECORD NOT FOUND ERROR

STOP
B

TDTE LT. DATE

V

DATE = TDTE

N

ACCUMULATE "TDTE" ENTRIES IN "DATE" TABLE

C
<table>
<thead>
<tr>
<th>Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SUBROUTINE MEDE (DATE1, DATE2, DEV1)</td>
</tr>
<tr>
<td>2</td>
<td>G, DATE1=START DATE, DATE2=TERMINAL DATE</td>
</tr>
<tr>
<td>3</td>
<td>END</td>
</tr>
<tr>
<td>4</td>
<td>DATE1=DATE1, DATE2, IPTR2(50), DATE2, IMARK, SEARCH,</td>
</tr>
<tr>
<td>5</td>
<td>INDEX1(50), IPAL(50), TDATE1, TMARK, TREGP(50), TEPAL(50)</td>
</tr>
<tr>
<td>6</td>
<td>LOGICAL RESET</td>
</tr>
<tr>
<td>7</td>
<td>G, DATE1=START, TSHIFT, SHIFTF</td>
</tr>
<tr>
<td>8</td>
<td>C ALPH</td>
</tr>
<tr>
<td>9</td>
<td>APH.N, TMPNF, TPFS, ADATE1, ADATE2, INAME(50), ORIG(50), SSTM(50),</td>
</tr>
<tr>
<td>10</td>
<td>INDEX1(50), TOSMT(50), TORG(50), JNAME(50), JORG(J), JEST</td>
</tr>
<tr>
<td>11</td>
<td>COMMON/IPAL(50), TDATE1, TMARK, TREGP, TEPAL</td>
</tr>
<tr>
<td>12</td>
<td>COMMON/TMPNF2/SHIFTF</td>
</tr>
<tr>
<td>13</td>
<td>COMMON/TMPNF3/TMPNF1/INDEX1, TOSMT, TORG</td>
</tr>
<tr>
<td>14</td>
<td>COMMON/TOSMT(50)/INDEX1, TOSMT, TREGP</td>
</tr>
<tr>
<td>15</td>
<td>COMMON/HMPNF3/INAME, ORIG, SSTM</td>
</tr>
<tr>
<td>16</td>
<td>KEPT1(50), ENCP50, DATE1, DATE2, TSHIFT, TMARK, INAME, ORIG, SSTM, REGP,</td>
</tr>
<tr>
<td>17</td>
<td>TEPAL</td>
</tr>
<tr>
<td>18</td>
<td>1000 IF(DATE1.EQ.0) I1000, F20, DO 11, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>19</td>
<td>IF(DATE2.LE.0) I1000, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>20</td>
<td>GOTO 20</td>
</tr>
<tr>
<td>21</td>
<td>10 IF(DATE1.EQ.0) I10, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>22</td>
<td>IF (DATE2.EQ.0) I10, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>23</td>
<td>GOTO 20</td>
</tr>
<tr>
<td>24</td>
<td>15 IF (DATE1.EQ.0) I5, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>25</td>
<td>GOTO 15</td>
</tr>
<tr>
<td>26</td>
<td>1 IF (DATE2.EQ.0) I1, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>27</td>
<td>GOTO 16</td>
</tr>
<tr>
<td>28</td>
<td>20 DO 25 I1, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>29</td>
<td>IF (TOSMT(I).L.E.31) I1000, GOTO 20</td>
</tr>
<tr>
<td>30</td>
<td>IF (TOSMT(I).L.E.31) I1000, GOTO 20</td>
</tr>
<tr>
<td>31</td>
<td>GOTO 20</td>
</tr>
<tr>
<td>32</td>
<td>10 KEPT1(50), ENCP50, DATE1, DATE2, TSHIFT, TMARK, INAME, ORIG, SSTM, REGP,</td>
</tr>
<tr>
<td>33</td>
<td>TEPAL</td>
</tr>
<tr>
<td>34</td>
<td>20 DATE1=DATE1, DATE2, IPTR2(50), DATE2, IMARK, SEARCH,</td>
</tr>
<tr>
<td>35</td>
<td>INDEX1(50), IPAL(50), TDATE1, TMARK, TREGP(50), TEPAL(50)</td>
</tr>
<tr>
<td>36</td>
<td>LOGICAL RESET</td>
</tr>
<tr>
<td>37</td>
<td>G, DATE1=START, TSHIFT, SHIFTF</td>
</tr>
<tr>
<td>38</td>
<td>C ALPH</td>
</tr>
<tr>
<td>39</td>
<td>APH.N, TMPNF, TPFS, ADATE1, ADATE2, INAME(50), ORIG(50), SSTM(50),</td>
</tr>
<tr>
<td>40</td>
<td>INDEX1(50), TOSMT(50), TORG(50), JNAME(50), JORG(J), JEST</td>
</tr>
<tr>
<td>41</td>
<td>COMMON/IPAL(50), TDATE1, TMARK, TREGP, TEPAL</td>
</tr>
<tr>
<td>42</td>
<td>COMMON/TMPNF2/SHIFTF</td>
</tr>
<tr>
<td>43</td>
<td>COMMON/TMPNF3/TMPNF1/INDEX1, TOSMT, TORG</td>
</tr>
<tr>
<td>44</td>
<td>COMMON/TOSMT(50)/INDEX1, TOSMT, TREGP</td>
</tr>
<tr>
<td>45</td>
<td>COMMON/HMPNF3/INAME, ORIG, SSTM</td>
</tr>
<tr>
<td>46</td>
<td>KEPT1(50), ENCP50, DATE1, DATE2, TSHIFT, TMARK, INAME, ORIG, SSTM, REGP,</td>
</tr>
<tr>
<td>47</td>
<td>TEPAL</td>
</tr>
<tr>
<td>48</td>
<td>1000 IF(DATE1.EQ.0) I1000, F20, DO 11, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>49</td>
<td>IF(DATE2.LE.0) I1000, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>50</td>
<td>GOTO 20</td>
</tr>
<tr>
<td>51</td>
<td>10 IF(DATE1.EQ.0) I10, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>52</td>
<td>IF(DATE2.LE.0) I10, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>53</td>
<td>GOTO 20</td>
</tr>
<tr>
<td>54</td>
<td>15 IF(DATE1.EQ.0) I5, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>55</td>
<td>GOTO 15</td>
</tr>
<tr>
<td>56</td>
<td>1 IF(DATE2.EQ.0) I1, TOSMT(50), TOSMT(50)</td>
</tr>
<tr>
<td>57</td>
<td>GOTO 16</td>
</tr>
<tr>
<td>58</td>
<td>20 DO 25 I1, TOSMT(50), TOSMT(50)</td>
</tr>
</tbody>
</table>
SUBROUTINE MERGE    7/4/74   CPG: 0  ROUNDXF IMAGE   FTN 4.54469   10/15/80  12.74.05

  RSET=.F.
  INUE=SEARCH(JUATE,JNAME,JORG,JDEST,RSET)
  IF(INDEX.NE.0) GO TO 35
  JNAME=.IN.
  JORG=.IN.
  JDEST=.IN.
  INUE=SEARCH(JUATE,JNAME,JORG,JDEST,RSET)
  INDEX=INDEX+1
  JNAME=SEARCH(JUATE,JNAME,JORG,JDEST,RSET)
  IF(INDEX.NE.1) INDEX=INDEX+1
  ORIG(INDEX)=ORG(INDEX)
  DSN(INDEX)=DSN(INDEX)
  TR(INDEX)=TR(INDEX)
  MPGF(INDEX)=MPGF(INDEX)
  REM(IINDEX)=RECM(IINDEX)
  TC(IINDEX)=TC(IINDEX)
  IF(INDEX.EQ.1) INDEX=INDEX+1
  IF(INDEX.EQ.0) GO TO 40
  PRINT 1300,DSN(INDEX),ORIG(INDEX),DSN(INDEX),TR(INDEX),TC(INDEX),MPGF(INDEX),RECM(INDEX)
  PRINT 1500 FORM(14) FOLLOWING ENTRY NOT INCLUDED "",21G,24B,24B,21G
  CONTINUE
  GO TO 27
  CALL ERR(SUC)
  5000 RETURN
  END
RPRT

DATE1 NE DATE2

SHIFT=0.

DATE2 GE DATE1

EXCHANGE DATE1 & DATE2

ICOUNT=0 IDEV=DEV1

SHIFT NE 0

REWIND DEV1

CALL MERGE (DATE1, DATE2, IDEV)

READ DEV1

DATE EQ 0

C

A

B

D
DATE EO.DATE1 AND ISHIFT EO. SHIFT

N

D2 GE DATE1 AND DATE LE DATE1 AND SHIFT EO 0

Y

SET INPUT PARAMETERS TO VALUES OF RECORD READ

SORT ENTRIES BY ORIG, DSTN, & TRUCK

FORMAT REPORT BODY

PRINT REPORT

RETURN

PRINT ERROR MSG "RECORD NOT IN FILE"

RETURN
SUBROUTINE RPR1 DATE1=START DATE DATE2=TERMINAL DATE SHIFT=VALID WHEN DATE1 DATE2
INTEGER DEV2
INTEGER DATE1,DATE2,IPTR2(50),DATE,DEV1,MARK,
INTEGER DATE1,DATE2,IPTR2(50),DATE,DEV1,MARK,
INTEGER DATE1,DATE2,IPTR2(50),DATE,DEV1,MARK,
DATE1,DATE2,MARK,TEGPR(50),TEPAL(50),DEVI,
DATE1,DATE2,MARK,TEGPR(50),TEPAL(50),DEVI,
DATE1,DATE2,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
MARK,TEGPR(50),TEPAL(50),DEVI,MARK,TEGPR(50),TEPAL(50),DEVI,
SUBROUTINE RPT 

115 IF(SHIFT.EQ.0) GCTO 140
120 PRINT 2125
PRINT 2150
PRINT 2175
PRINT 2205
PRINT 225
GOTO 126
140 PRINT 2125
PRINT 2150
PRINT 2275
PRINT 2295
145 IF(IPTRZ(I)) IF ORM(I).EQ.13M GCTO 115
PRINT 2225
2225 IF(NET(IM+1)) ILINE=ILINE+1
130 IF(NAME(I),EQ.,WSTRING) GCTO 125
PRINT 2700,0RIG(I),OSTM(I),NAME(I),REG(I),CPAL(I)
2400 FORMAT(IM+17,4511.1.15,16,15,16)
GOTO 120
125 PRINT 2500,0RIG(I),OSTM(I),REG(I)
135 FORMAT(IM+17,4511.1.15,16,15,16)
120 CONTINUE
RETURN
5000 CALL ERROR(JCC)
END
SEARCH

RESET EQ J

NOT (JDATE EQ DATE OR JDATE EQ 0)

CALL DTFND (JDATE)

NOT (UNAME EQ \( \text{NAME} \)) OR (JNAME EQ BLANKS)

NOT (JORG EQ \( \text{ORIG} \)) OR (JORGEQ BLANKS)

NOT (JDEST EQ \( \text{DSTN} \)) OR (JDEST EQ BLANKS)

INAME, JORG, & JDEST ALL BLANKS

INAME \( \text{(ICURS), ORIG (ICURS), \& DSTN (ICURS)} \) EQ BLANKS

ICURS ICURS-1
FUNCTION SEARCH

1. INTER-FUNCTION SEARCH(IODATE, JNAME, JORG, JDEST, RETSET)
2. COMMON/AWSR,C,JNAME, JORG, JDEST, JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
3. GET NAME
4. INTER-FUNCTION SEARCH(IODATE, JNAME, JORG, JDEST, RETSET)
5. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
6. GET NAME
7. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
8. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
9. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
10. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
11. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
12. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
13. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
14. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
15. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
16. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
17. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
18. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
19. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
20. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
21. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
22. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
23. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
24. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
25. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
26. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
27. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
28. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
29. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
30. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
31. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
32. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
33. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
34. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
35. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
36. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
37. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
38. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
39. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
40. COMMON/AWSR,C,JNAME, JORG, JDEST, NAME (SO), ORIG(SO), TRK.
SUBROUTINE UPDATE

C ALPHANUMERIC
ALPHA JORG JEST JNAME ORG DEST NAME MSG ORG DEST NAME MSG.
C T -

C BINARY
INTEGER DATE,DI
INTEGER JNAME,JMERC,REGP(10),FAL,FRMADA,EPAL(S0),DARRY(10).
INTEGER BLK1,DEZ,SEARCH
REAL ISHT
LOGICAL REST,UPDATE,UPDT
COMMON VSREGS/FRMADA,EPAL/REGP/IFTN
COMMON/AVSREG/SHFT
COMMON/3VREGS/NAME,TRK JORG DEST
COMMON/1SRL1/CATE021/IMARK,REGP,EPAL
COMMON/1SRL2/SHFT
COMMON/ISRL3/NAME,ORG,INST
COMMON/CATE021/UPDT,EPAL,EPAL,EPAL,EPAL,EPAL,EPAL,EPAL,EPAL,EPAL
C C -
JOTA = 0
JNAM, JNAME
JORG = ORG
JEST = DEST
JSEARCH, UPDT, JNAME, JORG, JEST, REST
IF (JM, J) GOTO 12
UPDT = 0
JOTA = 1
JORG = JORG
JEST = JEST
JSEARCH, UPDT, JNAME, JORG, JEST, REST
IF (JM, J) GOTO 5
JSEARCH, UPDT, JNAME, JORG, JEST, REST
GOTO 31

1 200
FUNCTION "TABLE SATURATION= FOLL OWING NOT INCLUDED":,
1 10 IF I = 040 "PALLETS FROM " SUB TO " ABV " VIA " ABV"
1 20 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 30 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 40 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 50 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 60 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 70 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 80 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 90 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 100 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 110 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 120 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 130 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 140 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 150 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 160 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 170 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 180 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 190 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 200 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 210 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 220 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 230 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 240 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 250 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 260 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 270 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 280 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 290 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 300 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 310 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 320 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 330 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 340 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 350 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 360 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 370 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 380 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 390 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 400 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 410 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 420 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 430 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 440 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 450 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 460 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 470 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 480 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 490 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 500 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 510 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 520 IF I = 040 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 530 IF I = 000 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
1 540 IF I = 900 "PALLETS FROM " ABV TO " ABV " VIA " ABV"
APPENDIX E - SAMPLE RUNS
SAMPLE - AVS1 RUN, REGULAR ORDERS
### AVS Regular Order Program

**10/20/80**

**B06.0**

(OPT=0)

----------

**ORDERS**

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>11 PALLETS FROM 5M</td>
<td>TU MWS</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0 PALLETS FROM 1a00</td>
<td>TO MWS</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>9 PALLETS FROM 2f</td>
<td>TO MWS</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4 PALLETS FROM 1</td>
<td>TO MWS</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>12 PALLETS FROM 1</td>
<td>TU MWS</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0 PALLETS FROM 1172</td>
<td>TO 198</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0 PALLETS FROM 14a</td>
<td>TO 1172</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>12 PALLETS FROM 0-W</td>
<td>TO 198</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>10 PALLETS FROM 0-W</td>
<td>TO 224</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>11 PALLETS FROM 0-W</td>
<td>TO 110</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>12 PALLETS FROM 0-W</td>
<td>TO 1e00</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>3 PALLETS FROM 67E</td>
<td>TO 224</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>11 PALLETS FROM 67E</td>
<td>TU 5M</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>13 PALLETS FROM 67E</td>
<td>TU 110</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1 PALLETS FROM 67E</td>
<td>TO 1e</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>1 PALLETS FROM 67E</td>
<td>TU 23</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>1 PALLETS FROM 67E</td>
<td>TO 61</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>1 PALLETS FROM 67E</td>
<td>TO 647</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>1 PALLETS FROM 67E</td>
<td>TU 1</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>4 PALLETS FROM 67E</td>
<td>TO 1e21</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>1 PALLETS FROM 67E</td>
<td>TO 49</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>1 PALLETS FROM 67E</td>
<td>TU 84</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>11 PALLETS FROM 1604</td>
<td>TU 224</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>11 PALLETS FROM 1604</td>
<td>TO 647</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>11 PALLETS FROM 1604</td>
<td>TO 63</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1 PALLETS FROM 1604</td>
<td>TO 1</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>11 PALLETS FROM 1604</td>
<td>TO 1172</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>11 PALLETS FROM 1603</td>
<td>TO 1603</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>11 PALLETS FROM 1603</td>
<td>TO 1513</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>11 PALLETS FROM 1603</td>
<td>TO 1138</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>5 PALLETS FROM 1603</td>
<td>TO 1009</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>1 PALLETS FROM 141</td>
<td>TO 224</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>1 PALLETS FROM 141</td>
<td>TO 110</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>2 PALLETS FROM 141</td>
<td>TO 24</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>2 PALLETS FROM 1402</td>
<td>TO 1e21</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>7 PALLETS FROM 1402</td>
<td>TO 224</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>3 PALLETS FROM 1402</td>
<td>TO 224</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>11 PALLETS FROM 1409</td>
<td>TO 5M</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>3 PALLETS FROM 1409</td>
<td>TO 1e00</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>11 PALLETS FROM 1409</td>
<td>TO 1e00</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>1 PALLETS FROM 1409</td>
<td>TO 23</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>1 PALLETS FROM 1409</td>
<td>TO 247</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>1 PALLETS FROM 1409</td>
<td>TO 1513</td>
<td></td>
</tr>
</tbody>
</table>

179
44 2 PALLETS FROM 1002 TO 24
45 7 PALLETS FROM 1002 TO SK
46 1 PALLETS FROM 1002 TO 23
47 1 PALLETS FROM 1002 TO 040
48 2 PALLETS FROM 1002 TO 047
49 30 PALLETS FROM 1002 TO 1507
50 1 PALLETS FROM 00E TO 1503
51 10 PALLETS FROM 00E TO 191
52 1 PALLETS FROM 60E TO 224
53 1 PALLETS FROM 00E TO SH
54 1 PALLETS FROM 00E TO X10
55 19 PALLETS FROM 1004 TO 647
56 2 PALLETS FROM 1004 TO 224
57 1 PALLETS FROM SM TO 1172
58 1 PALLETS FROM 67W TO 5P

-----------------
VEHICLES SELECTED
-----------------
1 VEHICLE ST 1 CAPACITY = 7 PALLETS, ROUTE DURATION = 240.0 MINS.
2 VEHICLE ST 2 CAPACITY = 7 PALLETS, ROUTE DURATION = 240.0 MINS.
3 VEHICLE ST 3 CAPACITY = 7 PALLETS, ROUTE DURATION = 240.0 MINS.
4 VEHICLE IT 1 CAPACITY = 12 PALLETS, ROUTE DURATION = 240.0 MINS.
5 VEHICLE IT 1 CAPACITY = 14 PALLETS, ROUTE DURATION = 240.0 MINS.
6 VEHICLE IT 1 CAPACITY = 10 PALLETS, ROUTE DURATION = 240.0 MINS.
7 VEHICLE IT 2 CAPACITY = 10 PALLETS, ROUTE DURATION = 240.0 MINS.

180
<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE</th>
<th>TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1602</td>
<td>409</td>
<td>7 PALLETS</td>
<td>45</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>SM</td>
<td>014</td>
<td>7 PALLETS</td>
<td>45</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>1602</td>
<td>823</td>
<td>7 PALLETS</td>
<td>36</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>224</td>
<td>838</td>
<td>1 PALLETS</td>
<td>54</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>60E</td>
<td>653</td>
<td>1 PALLETS</td>
<td>52</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>07C</td>
<td>657</td>
<td>1 PALLETS</td>
<td>12</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>7</td>
<td>410</td>
<td>911</td>
<td>1 PALLETS</td>
<td>54</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>224</td>
<td>915</td>
<td>1 PALLETS</td>
<td>52</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>1603</td>
<td>941</td>
<td>3 PALLETS</td>
<td>31</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>10</td>
<td>1006</td>
<td>506</td>
<td>3 PALLETS</td>
<td>31</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>1004</td>
<td>1111</td>
<td>2 PALLETS</td>
<td>50</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>1602</td>
<td>1615</td>
<td>2 PALLETS</td>
<td>44</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>191</td>
<td>514</td>
<td>1 PALLETS</td>
<td>32</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>14</td>
<td>224</td>
<td>1444</td>
<td>2 PALLETS</td>
<td>52</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>191</td>
<td>1110</td>
<td>2 PALLETS</td>
<td>44</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>1602</td>
<td>1114</td>
<td>1 PALLETS</td>
<td>32</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>23</td>
<td>1143</td>
<td>2 PALLETS</td>
<td>30</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1 PALLETS</td>
<td>30</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

ROUTE ENDED
LOCATION = 1074
TIME = 1211
NO OF PALLETS MOVED = 32
VEHICLE - ST 2
START TIME - 4:00.
DATE 102980

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1602</td>
<td>609</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>2</td>
<td>1507</td>
<td>618</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>3</td>
<td>1602</td>
<td>634</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>4</td>
<td>1507</td>
<td>639</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>5</td>
<td>1602</td>
<td>655</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>6</td>
<td>1507</td>
<td>900</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>7</td>
<td>1602</td>
<td>416</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>8</td>
<td>1507</td>
<td>921</td>
<td>7 PALLETS</td>
<td>49</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>9</td>
<td>1404</td>
<td>937</td>
<td>7 PALLETS</td>
<td>23</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>10</td>
<td>2245</td>
<td>1000</td>
<td>7 PALLETS</td>
<td>23</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>11</td>
<td>1404</td>
<td>1026</td>
<td>4 PALLETS</td>
<td>23</td>
<td>3 *SPLIT</td>
</tr>
<tr>
<td>12</td>
<td>2245</td>
<td>1631</td>
<td>4 PALLETS</td>
<td>23</td>
<td>3 *SPLIT</td>
</tr>
</tbody>
</table>

ROUTE ENDED
LOCATION = 1076
TIME = 1047
NO OF PALLETS MOVED = 39
VEHICLE - ST J
START TIME - 800
DATE 102080

*** STOP SITE TIME DELIVER PICK UP ORDER STAY TIME ***

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE</th>
<th>TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1604</td>
<td>064</td>
<td>7 PALLETS</td>
<td>25</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>23</td>
<td>083</td>
<td>7 PALLETS</td>
<td>25</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1604</td>
<td>098</td>
<td>4 PALLETS</td>
<td>25</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>23</td>
<td>012</td>
<td>7 PALLETS</td>
<td>25</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1604</td>
<td>022</td>
<td>7 PALLETS</td>
<td>24</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>647</td>
<td>055</td>
<td>7 PALLETS</td>
<td>24</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1604</td>
<td>102</td>
<td>4 PALLETS</td>
<td>24</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>647</td>
<td>103</td>
<td>4 PALLETS</td>
<td>24</td>
<td>3 * SPLIT</td>
<td></td>
</tr>
</tbody>
</table>

ROUTE ENDED
LOCATION = 1076
TIME = 1053
NO OF PALLETS MOVED = 22
VEHICLE = TR 1  
START TIME = 100.  
DATE = 102080  

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 64W</td>
<td>062</td>
<td>12 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>2 14A</td>
<td>015</td>
<td>16 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>3 91W</td>
<td>028</td>
<td>12 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>4 1608</td>
<td>046</td>
<td>12 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>5 1603</td>
<td>060</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>6 1605</td>
<td>012</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>7 SM</td>
<td>030</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>8 07E</td>
<td>091</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>9 00E</td>
<td>1006</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>10 SM</td>
<td>1008</td>
<td>1 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>11 91W</td>
<td>1021</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>12 X10</td>
<td>1042</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13 1603</td>
<td>1113</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>14 1138</td>
<td>1134</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>15 1603</td>
<td>1151</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>16 1203</td>
<td>1213</td>
<td>11 PALLETS</td>
<td>0</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

ROUTE ENDED  
LOCATION = 1074  
TIME = 1227  
NO OF PALLETS MOVED = 91
<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE</th>
<th>TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 SM</td>
<td>802</td>
<td>10 PALLETS</td>
<td>1</td>
<td>22</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 NMS</td>
<td>909</td>
<td>10 PALLETS</td>
<td>1</td>
<td>22</td>
<td>1</td>
<td>19</td>
</tr>
<tr>
<td>3 1005</td>
<td>1016</td>
<td>6 PALLETS</td>
<td>2</td>
<td>16</td>
<td>4</td>
<td>19</td>
</tr>
<tr>
<td>4 1</td>
<td>1053</td>
<td>8 PALLETS</td>
<td>4</td>
<td>19</td>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>5 NMS</td>
<td>1158</td>
<td>6 PALLETS</td>
<td>2</td>
<td>11</td>
<td>2</td>
<td>11</td>
</tr>
</tbody>
</table>

ROUTE END
LOCATION = 1078
TIME = 1310
NO OF PALLETS MOV'D = 24
**VEHICLE** - IT 1  
**START TIME** - 400.  
**DATE** - 10/20/00

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE</th>
<th>TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>66E</td>
<td>8:02</td>
<td>10 PALLETS</td>
<td>10</td>
<td>51</td>
<td>22</td>
</tr>
<tr>
<td>2</td>
<td>191</td>
<td>8:33</td>
<td>10 PALLETS</td>
<td>10</td>
<td>35</td>
<td>22</td>
</tr>
<tr>
<td>3</td>
<td>1604</td>
<td>8:57</td>
<td>10 PALLETS</td>
<td>10</td>
<td>55</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>447</td>
<td>9:49</td>
<td>10 PALLETS</td>
<td>10</td>
<td>9</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>1631</td>
<td>10:02</td>
<td>10 PALLETS</td>
<td>10</td>
<td>9</td>
<td>22</td>
</tr>
<tr>
<td>6</td>
<td>1137</td>
<td>11:02</td>
<td>10 PALLETS</td>
<td>10</td>
<td>14</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>1210</td>
<td>12:10</td>
<td>10 PALLETS</td>
<td>10</td>
<td>14</td>
<td>22</td>
</tr>
</tbody>
</table>

**ROUTE END**  
**LOCATION** = 1074  
**TIME** = 1640  
**NO OF PALLETS MOVED** = 40
VEHICLE - IT 2
START TIME - 1002
DATE 102080

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE</th>
<th>TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 B7E</td>
<td>802</td>
<td>8 PALLETS</td>
<td>20</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 102</td>
<td>829</td>
<td>6 PALLETS</td>
<td>20</td>
<td>19</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 1605</td>
<td>850</td>
<td>3 PALLETS</td>
<td>37</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 191</td>
<td>902</td>
<td>1 PALLETS</td>
<td>33</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5 1602</td>
<td>910</td>
<td>1 PALLETS</td>
<td>47</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 1805</td>
<td>923</td>
<td>2 PALLETS</td>
<td>49</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 B47</td>
<td>959</td>
<td>1 PALLETS</td>
<td>42</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 646</td>
<td>1611</td>
<td>2 PALLETS</td>
<td>46</td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 X10</td>
<td>1027</td>
<td>1 PALLETS</td>
<td>47</td>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 820</td>
<td>1030</td>
<td>3 PALLETS</td>
<td>37</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 1605</td>
<td>1184</td>
<td>10 PALLETS</td>
<td>40</td>
<td>22 *SPLIT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 X10</td>
<td>1154</td>
<td>16 PALLETS</td>
<td>46</td>
<td>22 *SPLIT</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

ROUTE END:
LOCATION = 107A
TIME = 1229
NO OF PALLETS MOVED = 26
**ORDERS NOT MOVED**

**--------------------------**

<table>
<thead>
<tr>
<th>ORDER</th>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
<td>2F</td>
<td>102</td>
</tr>
<tr>
<td>27</td>
<td>11F</td>
<td>102</td>
</tr>
<tr>
<td>26</td>
<td>1F</td>
<td>104</td>
</tr>
<tr>
<td>39</td>
<td>3F</td>
<td>105</td>
</tr>
<tr>
<td>43</td>
<td>1F</td>
<td>105</td>
</tr>
<tr>
<td>46</td>
<td>1F</td>
<td>105</td>
</tr>
<tr>
<td>41</td>
<td>1F</td>
<td>105</td>
</tr>
<tr>
<td>50</td>
<td>1F</td>
<td>105</td>
</tr>
<tr>
<td>21</td>
<td>1F</td>
<td>105</td>
</tr>
<tr>
<td>17</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>22</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>10</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>19</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>15</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>16</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>20</td>
<td>1F</td>
<td>107</td>
</tr>
<tr>
<td>7</td>
<td>6F</td>
<td>198</td>
</tr>
<tr>
<td>57</td>
<td>1F</td>
<td>198</td>
</tr>
<tr>
<td>35</td>
<td>2F</td>
<td>150</td>
</tr>
<tr>
<td>6</td>
<td>6F</td>
<td>117</td>
</tr>
<tr>
<td>7</td>
<td>12F</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>5F</td>
<td>23</td>
</tr>
</tbody>
</table>

188
SAMPLE - AVS2 RUN, SPECIAL ORDERS
AVS SPECIAL ORDER PROGRAM

10/20/80
800.0

-----
ORDERS
-----
1 14 PALLETS FROM 67E TO NW5
2 10 PALLETS FROM 67E TO SM
3 14 PALLETS FROM 67E TO 1601A

SPECIAL ORDER TIME = 90C.

BUMP OPTION = NO

------------------
VEHICLES SELECTED
------------------
1 VEHICLE ST4 CAPACITY = 7 PALLETS, ROUTE DURATION = 240.0 MINS.
2 VEHICLE ST5 CAPACITY = 7 PALLETS, ROUTE DURATION = 240.0 MINS.
3 VEHICLE TT2 CAPACITY = 14 PALLETS, ROUTE DURATION = 240.0 MINS.
**VEHICLE** - ST 4  
**START TIME** - 800.  
**DATE** - 10/20/80

---

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE TIME</th>
<th>DELIVER TIME</th>
<th>PICK UP TIME</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>176E</td>
<td>902</td>
<td>1 PALLETS</td>
<td></td>
<td>7 PALLETS</td>
<td>2</td>
</tr>
<tr>
<td>2SM</td>
<td>907</td>
<td></td>
<td></td>
<td>3 PALLETS</td>
<td>2</td>
</tr>
<tr>
<td>376E</td>
<td>912</td>
<td>3 PALLETS</td>
<td></td>
<td>3 PALLETS</td>
<td>2</td>
</tr>
<tr>
<td>4SM</td>
<td>916</td>
<td>3 PALLETS</td>
<td></td>
<td>3 PALLETS</td>
<td>2</td>
</tr>
<tr>
<td>576E</td>
<td>920</td>
<td>7 PALLETS</td>
<td></td>
<td>7 PALLETS</td>
<td>3</td>
</tr>
<tr>
<td>61601A</td>
<td>932</td>
<td>1 PALLETS</td>
<td></td>
<td>7 PALLETS</td>
<td>3</td>
</tr>
<tr>
<td>776E</td>
<td>944</td>
<td>7 PALLETS</td>
<td></td>
<td>7 PALLETS</td>
<td>3</td>
</tr>
<tr>
<td>81601A</td>
<td>950</td>
<td>7 PALLETS</td>
<td></td>
<td>7 PALLETS</td>
<td>3</td>
</tr>
</tbody>
</table>

ROUTE ENDED  
LOCATION = 107A  
TIME = 1002  
NO OF PALLETS MOVED = 24
VEHICLE = TT 2  
START TIME = 800.  
DATE = 102080  

<table>
<thead>
<tr>
<th>STOP</th>
<th>SITE</th>
<th>TIME</th>
<th>DELIVER</th>
<th>PICK UP</th>
<th>ORDER</th>
<th>STAY TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>67E</td>
<td>902</td>
<td>14 PALLETS</td>
<td>1</td>
<td>29 <em>SPCL</em></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>NWS</td>
<td>1015</td>
<td>14 PALLETS</td>
<td>1</td>
<td>29 <em>SPCL</em></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1129</td>
<td>12 PALLETS</td>
<td>5</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>NWS</td>
<td>1239</td>
<td>12 PALLETS</td>
<td>5</td>
<td>26</td>
<td></td>
</tr>
</tbody>
</table>

ROUTE ENDED  
LOCATION = 107A  
TIME = 1350  
NO OF PALLETS MOVED = 26  

192
<table>
<thead>
<tr>
<th>ORDER</th>
<th>FROM</th>
<th>TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2</td>
<td>1507</td>
</tr>
<tr>
<td>27</td>
<td>11</td>
<td>1172</td>
</tr>
<tr>
<td>20</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>39</td>
<td>3</td>
<td>1606</td>
</tr>
<tr>
<td>43</td>
<td>1</td>
<td>1621</td>
</tr>
<tr>
<td>44</td>
<td>1</td>
<td>110</td>
</tr>
<tr>
<td>41</td>
<td>1</td>
<td>23</td>
</tr>
<tr>
<td>56</td>
<td>1</td>
<td>1903</td>
</tr>
<tr>
<td>21</td>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>17</td>
<td>1</td>
<td>61</td>
</tr>
<tr>
<td>22</td>
<td>1</td>
<td>87</td>
</tr>
<tr>
<td>18</td>
<td>1</td>
<td>647</td>
</tr>
<tr>
<td>19</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>1</td>
<td>16</td>
</tr>
<tr>
<td>16</td>
<td>1</td>
<td>23</td>
</tr>
<tr>
<td>58</td>
<td>1</td>
<td>5M</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>172</td>
</tr>
<tr>
<td>57</td>
<td>1</td>
<td>5M</td>
</tr>
<tr>
<td>35</td>
<td>2</td>
<td>1621</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>198</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>NWS</td>
</tr>
</tbody>
</table>
AVS HISTORY FILE REPORT
26JUN73
SHIFT- 745.0

<table>
<thead>
<tr>
<th>WAREHOUSE NAME</th>
<th>TRUCK</th>
<th>DEFLIVERIES (PALLETS)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>FROM</td>
<td>TO</td>
</tr>
<tr>
<td>113A</td>
<td>ARA SF</td>
<td>DFYTN</td>
<td>TT2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TT3</td>
</tr>
<tr>
<td>64F</td>
<td></td>
<td>ST1</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>1601A</td>
<td>TT1</td>
<td></td>
</tr>
<tr>
<td>19A</td>
<td>1138</td>
<td>ST1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1601A</td>
<td>ST1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1605</td>
<td>ST2</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1605</td>
<td>TR1</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>DFYTN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1138</td>
<td></td>
<td>ST2</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td></td>
<td>TT1</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td></td>
<td>TT2</td>
<td></td>
</tr>
<tr>
<td>64W</td>
<td></td>
<td>ST1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>ST2</td>
<td></td>
</tr>
</tbody>
</table>

195
### INITIAL DISTRIBUTION

<table>
<thead>
<tr>
<th>Copies</th>
<th>Code</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>NAVMAT</td>
<td>O8T246 (LCDR J. Bland)</td>
</tr>
<tr>
<td>1</td>
<td>NAVMATO</td>
<td>(Code 11)</td>
</tr>
<tr>
<td>2</td>
<td>NPGS</td>
<td>(Library)</td>
</tr>
<tr>
<td>3</td>
<td>FMSO</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Code 953 (J. Blaylock)</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Code 953 (Dave Polm)</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>Code 953 (Bill McDevitt)</td>
</tr>
<tr>
<td>1</td>
<td>NAVSUPSCOL</td>
<td>(Athens, GA Lib)</td>
</tr>
<tr>
<td>10</td>
<td>NAVSUP</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>SUP</td>
<td>04112B (Sandra Jones)</td>
</tr>
<tr>
<td>1</td>
<td>SUP</td>
<td>04141A (Carol Steward)</td>
</tr>
<tr>
<td>6</td>
<td>SUP</td>
<td>043</td>
</tr>
<tr>
<td>2</td>
<td>SUP</td>
<td>053</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Bill McClure)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(CDR Palmatier)</td>
</tr>
<tr>
<td>8</td>
<td>NSC</td>
<td>CHARLESTON</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>Code 400 (CDR Dell)</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Code 61 (Bob Owens)</td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>Code 61 (Bob Lee)</td>
</tr>
<tr>
<td>1</td>
<td>NSC</td>
<td>San Diego (Mr. Osborne)</td>
</tr>
<tr>
<td>1</td>
<td>NSC</td>
<td>Puget Sound (Code 43)</td>
</tr>
<tr>
<td>1</td>
<td>NSC</td>
<td>Norfolk</td>
</tr>
<tr>
<td>1</td>
<td>NSC</td>
<td>Oakland</td>
</tr>
<tr>
<td>1</td>
<td>NSC</td>
<td>Oakland</td>
</tr>
<tr>
<td>1</td>
<td>NSC</td>
<td>Pearl Harbor</td>
</tr>
<tr>
<td>12</td>
<td>DTIC</td>
<td></td>
</tr>
</tbody>
</table>

### CENTER DISTRIBUTION

<table>
<thead>
<tr>
<th>Copies</th>
<th>Code</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>18</td>
<td>G.H. Gleissner</td>
</tr>
<tr>
<td>2</td>
<td>1809.3</td>
<td>D. Harris</td>
</tr>
<tr>
<td>1</td>
<td>185</td>
<td>T. Corin</td>
</tr>
<tr>
<td>20</td>
<td>187</td>
<td>R. Melton</td>
</tr>
</tbody>
</table>