SCHOOL-BUS ROUTING FOR PROGRAM SCHEDULING

JAMES H. BOOKINDER1,* and STEVEN H. EDWARDS2†

1 Department of Management Sciences, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1 and
2 The Durham Board of Education, Oshawa, Ontario, Canada L1J 3H3

(Received May 1988; revised February 1989)

Scope and Purpose—Most school-bus transport involves pick-up of students each morning, delivery always to their same respective schools, and trips back home following the school day. "Program scheduling" concerns transport of groups of students, each with a common origin (their own school) and a common destination (another school, with a special program facility (e.g. a swimming pool) unavailable at their own location). Following an interval for instruction using this facility, those same students are transported back to their own school for other regular classes. In this paper, we first improve the manual procedures in use at the Durham Board of Education (in Ontario, Canada). The organization did not have a computerized transportation information system, nor any priority to develop one. Improved manual methods were thus important to build our credibility and stimulate interest in the transitions to semi-automated and automated systems. We carefully outline these transitions and then formulate an optimization model for the program scheduling problem. Manual and computer solution of some test cases suggests good savings are possible in the form of reduced deadhead kilometers and/or decreased numbers of buses.

Abstract—The program scheduling problem involves a set of student pick-ups and drop-offs for which school-to-school routes must be developed. Each pick-up or drop-off has a scheduled time of occurrence, and these activities must be connected so as to minimize the number of deadhead or empty bus kilometers. We review the literature on routing with time windows and school-bus routing in general. We then formulate a mathematical programming model which has the structure of an assignment problem plus side constraints for time-feasibility and number of buses. Additional 0-1 variables are introduced which indicate those particular tasks that are connected to the depot, and which permit use of a pure assignment formulation and solution. Several actual problems from the Durham Board of Education are solved. Results, improvements and sensitivity questions are discussed, and suggestions are made for further research.

1. INTRODUCTION

The Durham Board of Education, one of the largest in the Province of Ontario, operates approx. 90 elementary schools and 20 secondary schools whose total enrollment is about 50,000 students. The board serves an area of 2400 km² to the east and north of metropolitan Toronto. Although the region of Durham is mostly a rural one, the area does include the urbanized towns of Ajax, Pickering and Whitby. It also includes the city of Oshawa, site of the major car assembly facilities of General Motors of Canada. Annual expenses for school transportation in Durham are $9 million (Cad.), with approx. 1/3 of the pupils receiving some transportation from the board. The most familiar examples are transport from home to school, and back again at the end of the day, in ordinary school buses or in special vans for the disabled. However, the present paper is concerned with transport for "program scheduling", an application which does not appear to have been treated in the literature.

Due to constraints on capital budgets, not all schools are constructed with special facilities for programs such as industrial arts or family and consumer studies. For this reason, pupils must be delivered from their home school to a site where such facilities are available. In general, these programs are compulsory and must be offered to the student. The majority of schools are involved

*James H. Bookinder is a Past-President of the Canadian Operational Research Society (CORS) and a Past-Chairman of the ORSA Transportation Science Section. Before joining the Management Sciences Department of the University of Waterloo in 1982, he had a decade of experience in industry and consulting, most recently (1978-82) as Director of Operations Research at the Toronto Transit Commission. Dr Bookinder holds an MBA from the University of Toronto and a PhD from the University of California, San Diego. His research concerns transportation, production/inventory management and physical distribution, with special interest in direct collaboration with industrial practitioners.

†Steven H. Edwards received a Bachelor's degree in environmental studies and a Master's degree in management sciences, both from the University of Waterloo. Since 1981, he has been a Planner with the Regional Municipality of Durham, where his responsibilities have concerned a number of issues in urban development, in addition to those of transport.
at sometime throughout the year in transportation for program scheduling, as origin or destination, for one program or another.

The system works as follows. School A, which has no industrial arts room, must send a class of students to school B where both facility time and space are available. Children are picked up at A and delivered to B for a fixed period, say 70 min. Once the class is completed, students are returned to their home school to continue their regular classes. The delivery from A to B described above will be defined as a task. Similarly, the return trip from B to A is thus another task which must be carried out.

The two principals must jointly decide upon their particular school-day schedules. These must align, taking account of the inter-school travel time, to permit utilization of the available facilities, and yet allow feasible schedules for the pupils’ remaining subjects. Principals may also determine school-day/class-start times, allowable bus-arrival times and permissible waiting periods. In general, some flexibility (time window) exists for class times, but judgment is required. A class must be supervised, in an appropriate area, while waiting for the bus or for the period to begin. Furthermore, idle time for the students removes a portion of the day’s available instruction time.

At each school, the schedule of activities will vary from one day to the next because students do not necessarily participate daily in programs requiring transport to other schools. Certain groups of schools are on 6-day cycles while others are on 5-day cycles. In any case, these cycles do not remain synchronized since the teachers’ “professional development” days (when school is closed) occur on different days in different locations. Furthermore, special programs such as swimming are implemented on a short-term basis, for say 3 weeks at a time.

The schedule of program runs could thus in principle differ for each of the 183 school days. This is in sharp contrast to regular school transportation: pick-up of students each morning, delivery always to their same respective schools, and return trips home every afternoon. Program runs are similar to the subscription portion of a dial-a-ride service, but even there, most work-related trip requests do not change from one week to the next.

These days, it is so routine for an organization to purchase software that this is sometimes done before a proper problem definition. The Board obviously could have acquired software for vehicle routing, perhaps specifically school-bus routing. However, any computerized algorithm needs the support of an information system. Our client had no such system (for transportation), nor any interest in developing one unless there were a clear indication of potential pay-off. To build credibility, we had to retain manual methods long enough to improve them. It was necessary to push to the limit, relationships we found between the program scheduling problem and an assignment problem. These allowed us to go further than we otherwise might have with manual, or quasi-manual, solution procedures and pre- and post-processors.

The remainder of this paper is as follows. In our third section, we present further details of the problem. Sections 4 and 5 contain our formulation of the objective function and constraints, and our suggestions for improving the Board’s manual procedures. The encouraging numerical results here then lead in Sections 6 and 7 to our mathematical programming approach, how this formulation may be related to an assignment problem by inclusion of additional 0-1 variables, and the computer solutions obtained from this model. The final section presents our summary, conclusions and suggestions for further research.

We begin with a review of the literature on general school-bus and dial-a-ride scheduling. Although the program scheduling problem differs from these situations, the basic principles of route construction and schedule improvement which apply there may be adaptable to the problem of program runs.

2. REVIEW OF LITERATURE

A general discussion of school-bus routing and scheduling is contained in Bodin et al. [1]. Bodin and Berman [2] show how one can connect school-bus runs to form bus schedules, in order to minimize total travel time. Beginning with routes and fixed school times, a sequence of transportation problems is solved. Evaluation of potential changes to routes is carried out with the aid of a useful construct called the “mini-stop”, an array of all potential bus stops.

Swersey and Ballard [3] study the scheduling of buses, given predetermined routes, so as to
minimize the total number of vehicles required. Variations in start- and end-times of bus runs were included in the form of $K$ permissible time windows. The formulation minimized the number of buses originating from the depot, employing one decision variable for each combination of run and discrete time interval $K$. The problem was solved as a linear programming model. (If necessary, when the solution was noninteger, a constraint was added to force the number of buses to be the smallest integer greater than that obtained.) Integer solutions were yielded by the LP in the majority of cases, and substantial saving in the number of buses were reported.

Desrosiers et al. [4] describe the capabilities and algorithmic details of TRANSCOL, a system for school-bus transportation developed at the University of Montreal. (What seems to be a predecessor system is described in [5].) The procedure [4] first solves a 0-1 programming model to select the start- and end-times of each school, within permissible limits, so as to minimize the maximum number of routes required in any given interval. Secondly, routing of the vehicles through bus stops begins with a traveling salesman grand tour which is then partitioned into routes which do not exceed bus capacity and which terminate at a school. Finally one decides how individual routes should be connected into daily work schedules for buses. The algorithm for joining partial routes and partial vehicle schedules is similar to that of [2]. TRANSCOL thus proposes an overall interactive approach to the school-bus problem, in which one determines bus stops and routes, school start-times and schedules.

Microcomputers have of course attracted a great deal of attention. Freeman [6] reviews several applications which employed mini-stops [2] to create the network of pick-up points. Mileage savings of 10–20% were reported, but no mention was made of the (proprietary) algorithms used. Other proprietary software is discussed by Spitzer [7].

The latter articles [6, 7] were naturally not directed to an OR audience. However, in a special issue devoted entirely to microcomputers and OR, Golden et al. [8] discuss commercially-available micro-based software for vehicle routing and scheduling. Nine such systems are compared in terms of versatility, hardware requirements, graphics, etc. Most of these could be adapted to the school-bus problem since they have the capability to handle multiple routes per vehicle, start- and end-times for buses, and time windows.

2.1. Time windows

The time components of school-bus transportation concern the scheduling of school start-times, student pick-up times and allowable ride-times. As noted above, time windows are sometimes permitted for these tasks [9]. Desrosiers et al. consider time windows in several separate routing and scheduling applications [10, 11, 12]. In [10], Lagrangian relaxation is studied for the single-depot multiple-vehicle problem in which school buses are required to visit specific task nodes. The results indicate improved computing time over column-generation methods [11], as well as a capability to handle larger network problems. In [12], the authors consider a multi-vehicle many-to-many problem, pertaining for example to transportation for the disabled.

Jaw et al. [13] develop a heuristic algorithm for the multi-vehicle dial-a-ride system with time windows and advanced reservations. The heuristic first searches for feasible insertions of passengers to buses and then selects the most desirable assignment for each passenger. Solomon [14] also found an insertion heuristic performed well for a variety of time-window problems. (Insertion ideas are considered in Section 5 in our discussion of manual techniques.) Solomon and Desrosiers [15] study a number of approaches to time-window constraints. Other aspects of routing and scheduling with desired delivery times are discussed by Sexton and Bodin [16]. Golden and Assad [17] summarize these plus additional recent developments in vehicle routing.

In sum, the literature references on school busing do not seem directly applicable to the program scheduling problem. Any of several of these could surely be adapted, or commercial software purchased, but neither course was initially advisable. Succeeding sections of this paper will furnish details of the Board's environment, its manual scheduling procedures and pencil-and-paper data base. For credibility of our project, it was first necessary to suggest improvements to those manual methods and to demonstrate superiority.

For these improvements, references on the general subject of vehicle routing will be helpful later in this paper. We turn now, however, to further details of the problem. These concern preparation
Table 1. The program scheduling tasks for a particular day (Monday 21 October, 1985)

<table>
<thead>
<tr>
<th>Task</th>
<th>Home to site</th>
<th>Site to home</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample code</td>
<td>Time location</td>
<td>Time location</td>
</tr>
<tr>
<td>A3</td>
<td>12:40 Claremont</td>
<td>13:00 Westney</td>
</tr>
<tr>
<td>A4</td>
<td>8:50 Lynde</td>
<td>8:55 Farewell</td>
</tr>
<tr>
<td>A5</td>
<td>10:10 Thornton</td>
<td>10:20 Farewell</td>
</tr>
<tr>
<td>A6</td>
<td>12:50 Fairman</td>
<td>12:55 Farewell</td>
</tr>
<tr>
<td>A7</td>
<td>12:40 Meadowcrest</td>
<td>12:50 Whitey</td>
</tr>
<tr>
<td>A8</td>
<td>10:05 Massey</td>
<td>10:07 Harmony</td>
</tr>
<tr>
<td>A9</td>
<td>10:10 Sunset</td>
<td>10:15 Phillips</td>
</tr>
<tr>
<td>A10</td>
<td>14:05 Sunset</td>
<td>14:10 Phillips</td>
</tr>
<tr>
<td>A11</td>
<td>9:00 Grandview</td>
<td>9:06 Lovell</td>
</tr>
</tbody>
</table>

The code for each task refers to transfer between those two schools at particular times. These codes are used in Figs 2-4.

of the relevant data and definitions of the nodes and arcs, which are necessary to discuss the mathematical programming formulation as well as the manual scheduling methods.

3. FURTHER DETAILS OF THE PROBLEM

For any given day, there is a schedule describing the tasks which must be performed. Table 1 indicates the 22 tasks and associated times for 21 October, 1985. (Later we will discuss small time windows for these tasks.)

Naturally, all tasks will be completed. Those of Table 1 represent 140 km of fixed distance which must be covered in delivering pupils from various schools A to other schools B. Thus, the quality of a solution is measured by the “deadhead kilometers”, the total distance traveled in excess of 140 km.

3.1. Data preparation

All data for this problem, apart from the known schedules, had to be generated for our specific purposes. Not only isn’t this department of the Board computerized, its pencil-and-paper files did not directly contain the data we required. These data concern the nodes, arcs and arc distances.

Distances between pairs of schools were calculated using a 1:10,000 metric street map. In each case, the shortest route was selected based on the existing grid system. No adjustments were made to routes to account for traffic or for intersections at which turns are difficult, etc. Distances were converted to minutes of travel time, based on an average speed of 50 km/hr. The latter is reasonable because few tasks are carried out in the rush hour.

3.2. Node representation

After some experimentation, we chose as network nodes the performance of the required tasks. Each node will have a start time, start location, end time and end location. (Samples nodes are depicted in Fig. 1.) Longer routes can thus be formed by connecting various nodes to each other.
The number of deadhead kilometers will of course differ as various combinations of tasks are joined to develop possible routes.

3.3. Arc set

As mentioned, the nodes themselves take into account all vehicle trips with pupils on board. Arc distances are traversed by empty buses.

There are three possible types of arcs, the most common of which connect tasks nodes (see Fig. 1). The other two types emanate from the depot to the start of a task time and location, or from the end of a task time/location to the depot.

There are $n(n-1)$ arcs which connect $n$ distinct nodes, but many will be infeasible due to time constraints. There are also $n$ arcs from the depot to the start of some task, and another $n$ arcs which join the end of a task and the depot.

4. OBJECTIVE FUNCTION AND CONSTRAINTS

4.1. Introduction

To discuss the objective function, we first note that the Durham Board of Education currently operates a fleet of eight full-sized school buses and employs unionized drivers for them. Board-owned buses are stored at a single location, and start and return there every day.

All remaining transportation is furnished by private contractors who provide regularly scheduled runs and emergency service. Annual negotiations with these outside contractors determine cost increases for the vehicle/driver base rate and gas price escalation.

The resulting agreement with outside suppliers of school-bus service means that the Board is not limited to its particular fleet of vehicles. Thus, one reasonable objective is minimization of the number of deadhead kilometers, for various given numbers of buses.

On the other hand, for the Board itself, almost 90% of its expenses for school transport are fixed costs for vehicles and drivers which cannot be altered. An alternative objective is therefore to minimize the number of buses, without increasing the deadhead kilometers traveled. (We remark that here, as in fixed-route transit, the expense for drivers is the largest component of bus operating costs.)

4.2. Notation

This second objective will be discussed below, but we begin with the former. Measurement of the objective function will thus be in kilometers; the arc cost $C_{ij}$ is the distance between the end point of task $i$ and the starting point of task $j$. A solution will connect tasks in a sequence to form a bus route; the binary variable $X_{ij}$ will be 1 if task $j$ follows task $i$, and 0 otherwise.

Additional notation is as follows: $n =$ number of tasks, $s =$ source (depot), $r =$ sink (depot), $m =$ number of buses originating from or returning to the depot, $a_i =$ start time of task $i$, $b_i =$ end time of task $i$, and $T_{ij} =$ travel time from end location of task $i$ to start location of task $j$. The node set includes tasks 1 through $n$, plus the source $s$ and sink $r$. 
4.3. Formulation

The formulation is then

\[
\text{minimize } \sum_i \sum_j C_{ij} X_{ij}
\]

subject to

\[
\sum_i X_{ij} + X_{sj} = 1, \forall j \tag{2}
\]

\[
\sum_j X_{ij} + X_{ri} = 1, \forall i \tag{3}
\]

\[
\sum_j X_{sj} = m \tag{4}
\]

\[
\sum_i X_{ri} = m \tag{5}
\]

\[
(b_i + T_{ij})X_{ij} \leq a_j \tag{6}
\]

\[
X_{ij} = 0, 1. \tag{7}
\]

Constraints (2) and (3) indicate that all task nodes are connected from a predecessor task or the depot \( s \), and to a subsequent task or the depot \( r \). (The notation \( r \) and \( s \) is introduced to distinguish between a right-side and a left-side depot connection. Naturally, a bus would terminate only at the same depot from which it originated.) Equations (4) and (5) specify the number of buses originating from or returning to the depot. For an arc to be feasible, constraint (6) requires that the end time of the first task plus travel time to the next task be less than or equal to the start time of the second task.

Later in this paper, we will devote considerable attention to solving this mathematical programming model. However, we first discuss some practical matters. These concern the environment at the Board, and the program scheduling problem itself. As will now be explained, we had little choice but to begin by improving the existing manual procedures.

5. IMPROVED MANUAL PROCEDURES

5.1. Introduction

In a "production" mode, as opposed to a research activity, no computer-aided methods are used by the Board for this problem. Routing, scheduling and even data storage and retrieval are all manual. An entire summer is required for the preparation of bus routes and schedules. Transportation staff rely heavily upon historical information in an attempt to exploit former or established routes and schedules. This is extremely difficult, however, since pupil numbers and school timetables will often change from one year to the next.

In developing routes by hand, the possible connections between tasks are evaluated in terms of feasibility and distance. Based upon intuition, scheduling staff suggest whether a particular connection will ultimately lead to a better route than another. It may surprise some academics, but should surprise no practitioner, that until we began the work reported in this article, the Board had never tested the efficiency of its bus routes or schedules.

Naturally, a computer-based solution involving an OR algorithm was (and is) the ultimate objective of this research. However, our previous experience in the transit industry, in scheduling both fixed-route and dial-a-ride services, suggested as an initial goal the development of improved manual procedures. These would be easier to implement and would establish our credibility. Since the Board obviously did not have an information system which could support a computerized algorithm, improvement of the manual procedure seemed quite reasonable to us.

Alternative methods of hand solution were considered and applied to the data shown in Table 1. These approaches established a systematic solution, based upon certain rules and criteria that are commonly included in computerized algorithms [1, 17]. Two of these methods will now be explained.
5.2. Interchange method

In this approach, one begins with a chosen number of buses, say \( m \), and examines the next \( m \) tasks in time order. Every such task is initially covered by a separate route. Succeeding tasks are examined, and appended if possible to existing routes. One then interchanges pairs of tasks, each on a different route, if a decrease in deadhead kilometers can be realized. Figure 2(a) illustrates this possibility.

It is also permissible in this method, in the course of building routes, to add additional vehicles. These are available from an outside contractor, and will be needed when a node cannot be feasibly connected to an already-existing route, as for example in Fig. 2(b). Extra buses are typically added as tasks later in time are examined.

Figure 2(c) shows the result of applying the interchange method to the tasks of Table 1. Compared to the Board’s solution, this method saved 3% on deadhead kilometers and yet could be executed with two less vehicles.

5.3. Insertion method

Here one does not add extra buses, since none will be necessary. One rather begins with an unreasonably large number \( n \) equal to the number of tasks, and thus subtracts buses, by removing a task from one route and switching it to another. (Note the difference between this approach where tasks are examined singly, and the interchange method where one examines pairs of tasks.) An insertion is carried out if it yields a saving in deadhead kilometers.

Good insertions will eliminate route segments, and hence eventually routes and vehicles. Figure 3(a) indicates two routes and several possible insertions between two particular nodes. The insertion solution of Fig. 3(b) represents a savings of 16% in deadhead kilometers, relative to current Board practice, and again with a reduction of two buses.

5.4. Variations and results

Two variations in the above approaches were considered. In the interchange method, one could vary the number of buses \( m \) used at the first step. In the insertion method, rather than seek the largest absolute savings in deadhead kilometers, one might consider inserting tasks which lead to
a tighter schedule. One would thus select an insertion which decreases idle driver time between the inserted node and its predecessor.

Solutions for 21 October, 1985, with these variations, are shown in Fig. 4, which also contains the actual solution used by the Board of Education. Those cases, plus results from the basic interchange and insertion methods, are summarized in Table 2.

5.5. Difficulties

In every case, the four proposed techniques generated solutions better than that currently in use at the Board. Nevertheless, these methods have inherent difficulties. One would in principle have to enumerate every feasible interchange or insertion to attain optimality. Table 1 contains only 22 tasks, and without a full enumeration, we still evaluated 35 interchanges and 97 insertions respectively. An exhaustive manual solution is thus highly impractical for the complicated days which have about 60 tasks.

Naturally, the preceding methods would be greatly improved if good bounds were available on the potential savings in deadhead kilometers, since then one would quit attempting improvements long before evaluating all possibilities. Even without such bounds, however, the goal of these techniques was achieved. By showing the scheduling staff how to make their manual procedures more systematic without requiring a computer, there was greater interest in this initial phase of our work. That in turn stimulated thoughts on “computerization of the solution procedure”, i.e. development of an OR algorithm. We now turn attention in this direction.

6. MATHEMATICAL PROGRAMMING APPROACH

We previously formulated a model to minimize deadhead distance [eqn (1)], subject to constraints (2) to (7). This model is potentially quite a large one. For example, the simple problem of Table 1 (22 tasks) has 508 constraints, ignoring the 0-1 restrictions (7). For the most complicated day faced by the Board (66 tasks), there are approx. 4400 constraints, again ignoring (7). An organization
possessing sufficient expertise in mathematical programming can attack head-on such a 0-1 optimization model. Our client is not such an organization.

We thus attempted to outsmart the problem. (So, too, should the personnel of the former organization.) Note that constraint (6) specifies which connections between nodes $i$ and $j$ are feasible. For any given set of tasks, this inequality may be used to identify arcs which need not be considered in any solution, and for which the decision variables $X_{ij}$ can be excluded from the problem. After eliminating all such variables, constraint (6) may be removed from the formulation. (We defer for the moment the impact on the problem size of this and other proposed changes.)

Constraints (4) and (5) pertain to the number of buses $m$. One could of course specify $m$ in advance or solve parametrically for various values. On the other hand, the model (1), (2), (3) and (7) is essentially an assignment problem for which very fast algorithms exist [18]. We now show that a manual method for solving the assignment problem can be adapted to our larger model, and in the process identify the number of vehicles necessary. [Naturally, this assumes constraint (6) has been implicitly applied.] As we shall see, proper consideration of the depot is important.

6.1. The assignment problem

Figure 5(a) shows the cost matrix $C_{ij}$ for a simple assignment problem. Each row or column, supply or demand, respectively represents an empty bus available at the end of task $i$ or required
Fig. 5. Assignment matrices $C_{ij}$. Examples which are feasible (a) and infeasible (b) for the program scheduling problem. A slash indicates an arc which violates the time constraint (6). $M$ denotes an essentially infinite cost.

at the start of task $j$. As before, the matrix elements $C_{ij}$ correspond to the distance between the end point of task $i$ and the starting point of task $j$. The diagonal elements are set equal to $M$, essentially an infinite cost. As mentioned, infeasible arcs such as from $c$ to $b$ have been removed from the $C_{ij}$ matrix, and denoted here by a slash symbol but later also set equal to $M$.

The matrix in Fig. 5(a) is feasible, but that in Fig. 5(b) is not. A location is required to receive the empty bus from $a$, while a source is required to supply an empty vehicle to $e$. The interpretation is that after completing task $a$, the bus must return to the depot. Similarly, the vehicle traveling to $e$ must originate from the depot.

It is thus necessary, to render feasible the example of Fig. 5(b), to explicitly consider the depot. In general, such consideration is also necessary for optimality in Fig. 5(a), even if that route could be executed by a single bus. The deadhead costs of moving this vehicle in and out of the depot have not been included, but they must, in order to properly evaluate candidate solutions.

6.2. Consideration of the depot

To consider the depot while retaining the assignment formulation, we introduce an additional task $d_i$ corresponding to each node $i$. The new task $d_i$ may be thought of as a vehicle traveling to or from the depot. [It will not be necessary to distinguish between $r$ and $s$ as in constraints (2) and (3).]

$X_{dif} = 1$ requires that a bus travel from the depot to the start of task $i$. Similarly, $X_{iend}$ = 1 indicates that the bus which performs task $i$ will then proceed to the depot. Naturally, the entries in the cost matrix, $C_{df}$ or $C_{da}$, will correspond to the appropriate distances from or to the depot (except in the following two instances, for the manual solution only).

Suppose there is a task $p$ for which no feasible successor can be found. $X_{dip}$ must thus equal unity. Such a result will be automatic with a full computer solution, but the manual approach greatly benefits if this assignment is forced from the beginning. In the cost matrix for the manual method, one thus sets $C_{dp}$ equal to zero, but in tallying the total solution costs, its actual nonzero distance to the depot must be added back in. Similar remarks pertain to the case for which $p$ has no feasible predecessor. Except for these two instances, however, the cost matrix contains actual task-depot kilometers $C_{dp}$ (or $C_{dp}$) as elements for a given task $p$. Naturally, one takes $C_{dp} = M$ and $C_{dp} = M$ for $i \neq j$, similarly, $C_{dp} = M$ for $i = j$, while for $i = j$, this can be taken as zero.

6.3. Absence of constraints involving $X_{didi}$

The depot-task $d_i$ allows proper treatment of routes which may begin or end with task $i$. However, no assignment-type constraints should be imposed on row $d_i$ or column $d_i$. That is, the variable $X_{didi}$ should not appear in constraints of the form (2) or (3). This can be shown by contradiction. Consider the small route, depot $\rightarrow b \rightarrow a \rightarrow$ depot, involving tasks $a$ and $b$. Since $X_{da} = 1$ and $X_{a} = 0$, one finds an inconsistency between the constraints

\[ X_{dab} + X_{aba} = 1 \]

and

\[ X_{dab} + X_{ada} = 1. \]
Table 3. The “depot-tasks” $d_i$, introduced to retain the assignment formulation

1. With each task $i$, associate a depot-task $d_i$ for connecting $i$ to or from the depot.
2. Append the $d_i$ variables ($i = 1, 2, \ldots, n$) to the rows and columns of the assignment matrix $X$ (now $2n \times 2n$, rather than $n \times n$).
3. The only assignments allowed for $d_i$ are:
   
   $d_i \rightarrow i$ (task $i$ is the first task on a route),
   
   $i \rightarrow d_i$ (task $i$ is the last task on the route), or
   
   $d_i \rightarrow d_i$ (task $i$ has no direct connection to the depot).

No mixing of $d_i$ and $d_j$, or $i$ and $d_i$, for $i \neq j$.

4. Elements of the cost matrix $C$

   $C_{di} = 0$

   $C_{dd} = $ number kilometers to or from the depot, respectively.

   Exceptions (for manual algorithm, only). $C_{di} = 0$ or $C_{dd} = 0$, if task $i$ has no feasible successor or predecessor (respectively forcing $X_{di} = 1$ or $X_{dd} = 1$). After solving, actual distances to or from depot are added back in.

   Elements such as $C_{dd}$ or $C_{di}$, for $i \neq j$, are set equal to a large number $M$.

5. $X_{dd}$ is to be omitted from constraints (2), (3) [see eqns (2'), (3')].

Remarks. The above assumes we have taken $C_{ij}$ as $M(-\infty)$ for tasks that would violate (6). Also, number of buses identified in post-processor phase (see later in this paper).

<table>
<thead>
<tr>
<th>Problem size*</th>
<th>Problem size†</th>
<th>Problem size‡</th>
</tr>
</thead>
<tbody>
<tr>
<td>22 tasks (Table 1)</td>
<td>462 variables</td>
<td>229 variables</td>
</tr>
<tr>
<td>66 tasks</td>
<td>308 constraints</td>
<td>46 constraints</td>
</tr>
<tr>
<td></td>
<td>4290 variables</td>
<td>1020 variables</td>
</tr>
<tr>
<td></td>
<td>4400 constraints</td>
<td>134 constraints</td>
</tr>
</tbody>
</table>

*Original formulation [eqns (1)-(7)].
†After pre-processing using (6).
‡Case † plus introduction of new depot-tasks $d_i$.
§Number possibly nonzero variables. Ones like $X_{di}$ or $X_{dd}$, technically do appear in the assignment matrix, but their corresponding entries of $M$ in the cost matrix make these variables impossible.
‖No row constraints are added by inclusion of the depot-tasks.

Naturally, for each task $i$, one does include the constraints

$$\sum_{k=1, k \neq i}^{n} X_{ki} + X_{dd} = 1 \quad (2')$$

and

$$\sum_{j=1, j \neq i}^{n} X_{ij} + X_{dd} = 1. \quad (3')$$

Simply, task $i$ must be connected from a predecessor [one of the other $(n-1)$ distinct tasks $k$, or the depot], and $i$ must also be succeeded by another task or the depot. As well, the objective function is now

$$\text{minimize } \sum_{i \neq j} C_{ij} X_{ij} + \sum_{i=1}^{n} [C_{di} X_{di} + C_{dd} X_{dd}]. \quad (1')$$

To summarize, Table 3 contains our procedure to retain the assignment formulation [eqns (1'), (2'), (3'), (7)] while still properly accounting for the depot. (Sample problem sizes are presented in Table 4.) It is this procedure and formulation for which the mathematical programming results (both computer-based and manual) will now be discussed.

7. MATHEMATICAL PROGRAMMING RESULTS

We chose for computer solution four of the larger program scheduling problems faced by the Board. To begin, we tested several small sub-problems using the “student-version” of LINDO [19] on the microcomputer network at the University of Waterloo. These tests indicated that our formulation provided output which could be translated into sensible routes.
Table 5. Computer results for the larger program scheduling problems

<table>
<thead>
<tr>
<th>Sample problem</th>
<th>Board solution</th>
<th>New solution</th>
<th>% Savings</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of tasks</td>
<td>No. of buses</td>
<td>Deadhead kilometers</td>
<td>No. of buses</td>
</tr>
<tr>
<td>A</td>
<td>23</td>
<td>7</td>
<td>199</td>
</tr>
<tr>
<td>B</td>
<td>34</td>
<td>8</td>
<td>311</td>
</tr>
<tr>
<td>C</td>
<td>35</td>
<td>7</td>
<td>233</td>
</tr>
<tr>
<td>D</td>
<td>47</td>
<td>10</td>
<td>305</td>
</tr>
</tbody>
</table>

The larger problems were then run on an IBM 4381 mainframe, employing the optimization software MINOS [20]. Its capabilities are far more than needed for our model, but the enhanced version of LINDO was unavailable, and we had difficulties in interfacing with other mathematical programming software on the Waterloo system. The results in Table 5 show reductions of between 17 and 20% in deadhead kilometers, and in some cases a reduced number of vehicles as well.

Naturally, some savings in kilometers were anticipated here based on the manual methods presented earlier. The savings attained are good, and perhaps as much as can be expected, given the geography of the problem. Firstly, connections between municipalities are rare because of the large distances involved between schools in adjoining municipalities. A computer algorithm and of course a manual scheduler would each avoid connections across municipal boundaries. Secondly, families of schools tend to be clustered around the site offering a special program. The scheduling of classes and the location of services are designed from the beginning to exploit this clustering.

One also would expect to save buses on certain days’ schedules. On days of relative inactivity (not reported here), a Board driver might be given only two or three tasks. These could easily be inserted into another route. However, in order that each driver should have some work to do, smaller routes have been used by the Board.

7.1. Determining the number of buses

Naturally, the solution to an optimization problem is of course feasible. The situation may differ here because we have ignored constraints (4) and (5) in the model (1) to (7). It is not clear what the number of buses should be from either a manual or computer solution of the model (1'), (2'), (3'), (7). We first discuss the remedy for a manual approach.

Solution of the assignment problem is based on the equivalent matrix of “cost penalties” [18]. Values of zero in this reduced matrix are highly desirable and indicate assignments which can be made at zero penalty cost. In solving the problem by hand, one crosses out all the zeroes, trying to use a minimum number of lines (no more than the number of variables).

A simple set of rules enables the assignment algorithm to be carried out this way. In the process one determines the number of buses. A flowchart is as follows [naturally only time-feasible assignments consistent with (6) are considered]:

1. (a) Assign first, those tasks i with only a single possible connection to a successor task j.
   (b) Cross out row i and column j from the matrix of cost penalties.

2. If any tasks remain with more than one possible successor, first select a successor for each task with 2 possibilities, then 3, then 4, etc. each time applying 1(b) above.

3. Continue steps 1 and 2 until all tasks have a predecessor and a successor. (Use the depot as the predecessor or successor only when there is no other possible predecessor or successor.)

An extra bus is indicated whenever an additional task is connected with the depot; the required number of vehicles equals the total number of depot-task (or task-depot) connections. In Fig. 6, a single vehicle would be used for the route, depot → c → b → a → depot. We remark that the depot should not be chosen to succeed any task which has another possible successor. We would therefore chose the connection b → a, rather than b → depot in Fig. 6, to avoid an additional bus. Note that as one continues to make assignments, other tasks could then be left with only the depot as a possible successor or predecessor, and hence must be assigned next.

Remarks similar to those above pertain to the computer solution. The required fleet size was
School-bus routing for program scheduling

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>(d_a)</th>
<th>(d_b)</th>
<th>(d_c)</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>M</td>
<td>/</td>
<td>0</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>b</td>
<td>0</td>
<td>M</td>
<td>M</td>
<td>0</td>
<td>M</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>0</td>
<td>M</td>
<td>M</td>
<td>2</td>
</tr>
<tr>
<td>d</td>
<td>1</td>
<td>M</td>
<td>0</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>d</td>
<td>M</td>
<td>0</td>
<td>M</td>
<td>0</td>
<td>M</td>
</tr>
<tr>
<td>d</td>
<td>M</td>
<td>M</td>
<td>0</td>
<td>M</td>
<td>0</td>
</tr>
</tbody>
</table>

Fig. 6. Manual solution. Obtaining the arcs in the final solution, and the number of buses, from the matrix of cost penalties. Arcs selected in the sequence: \(a\) \(\rightarrow\) depot; \(b\) \(\rightarrow\) \(a\); \(c\) \(\rightarrow\) \(b\); depot \(\rightarrow\) \(c\).

**TWO ROUTES FROM BOARD SCHEDULE FOR PROBLEM D**

ROUTE 1 (last 2 tasks shown)

- **Lincoln** - 19 km - **Depot**
- **Claremont** - 20 km - **Depot**
- **Westney** - 25 km - **Depot**

ROUTE 2

- **Depot** - 20 km - **Westney** - 25 km - **Depot**
- **Claremont** - 20 km - **Depot**

**COMPUTER TREATMENT**

- **Lincoln** - 19 km - **Depot**
- **Claremont** - 20 km - **Depot**
- **Westney** - 25 km - **Depot**

Fig. 7. An example of the savings from Problem D (see Table 5).

determined from the number of connections with the depot. By hand, the basis printout was translated into routes, and checked for feasibility, accuracy and reasonableness.

7.2. Decreasing the number of buses

In solving the preceding problems, the sole objective was minimization of deadhead kilometers. The number of vehicles was not specified, not even via (4) and (5). Nevertheless, minor reductions were often obtained relative to current Board practice.

For example, two buses were eliminated in Problem D (Table 5). One portion of this solution is illustrated in Fig. 7. Connecting these two routes saved 40 km, the majority of the savings realized by the computer solution. It is also this connection which made possible one of the two buses saved. The motivation of the formulation, however, was a decrease in deadhead kilometers and not in the number of vehicles.

That vehicles can often be saved *en passant* is clear from Fig. 7. This change and the reduction in deadhead kilometers both result from eliminating one connection to the depot. Indeed, it is the dead connections to the *depot* that are critical, since we could always attain zero deadhead kilometers between nodes by assigning each task to a separate bus.

7.3. Time windows for tasks

As mentioned earlier, school principals do have small amounts, say 10 min of flexibility within which to vary the start- or end-times of instructional periods or even the school day. Constraint
(6) for the time feasibility of connected nodes might then be relaxed to

\[(b_i + T_{ij})X_{ij} - a_j \leq 10. \]  

(6')

Use of (6') for the tasks of Table 1 rendered feasible an additional 28 arcs. Solution by the interchange method yielded slightly different routes and a further 1% reduction in deadhead kilometers.

For each of Problems A–D (Table 5), the relaxed constraint led to about 20 additional feasible pairings between tasks i and j. (The increase in the number of arcs seemed to be influenced more by the particular times \(b_i, a_j\) than by the problem size.) Nevertheless, the final routes were unchanged compared to those obtained using constraint (6). Lengthening the window beyond 10 min would eventually produce some changes, but would be increasingly more difficult to implement because of the impact on entire school schedules. As a result, we pursued this no further.

8. SUMMARY AND CONCLUSIONS

The results of our hand calculations (Table 2 and Figs 2–4) for the schedule of 21 October, 1985 (Table 1) showed that improved manual procedures were possible. Indeed, the Board’s routes for that program scheduling problem were “dominated” in the sense of multiple-objective optimization [21]: the routes could be carried out with fewer deadhead kilometers and with a decreased number of vehicles.

Following this initial research, a mathematical programming model [eqns (1)–(7)] was formulated. This was eventually shown to be related to an assignment problem, whose “tasks” were defined as the delivery of a class of pupils from school A to school B. Naturally all tasks must be done. It is worth emphasizing, therefore, the actual transport of pupils has no bearing on the solution or the technique used to obtain it. The quality of a solution is measured in terms of deadhead kilometers beyond those required for the given tasks (see Fig. 1).

Application of our model to several larger program scheduling problems produced improved routes (Table 5 and Fig. 7), ones which again often dominated those developed by staff at the Board. The results in general support the desirability of implementing our method at the Durham Board of Education.

This would obviously remove the tedium of the present manual approach. Increased speed and accuracy in the generation of routes would of course result as well, but considerable work would be necessary to fully computerize our procedure. For all the sample problems discussed in this paper, many hours were required for preparation of the input data. In addition, the computer solutions were translated (manually and slowly) by us into understandable, readable schedules. For this, a post-processor should be developed. A matrix generator is required for the input function, but of course that would assume the Board has a computerized transportation data base.

Granted this, we outline in Fig. 8 the design of a system to permit the scheduler to enter the tasks for which routes are desired, and from which a readable schedule is output. Development of such a system would naturally be a major undertaking, one for which the Board may possess only the project management skills. However, the data base and input generator portions in Fig. 8 would be part of any commercial package for vehicle routing and scheduling [8]. If this type of software were purchased, it is reasonable that the solution process could be streamlined, and all interfaces could be customized, with the aid of external consultants. A “streamlined” solution procedure of course suggests the availability of mathematical programming software with no substantial restrictions on problem size, etc. One would thus bypass step 7. Step 6 would also not be done as such; constraint (6) would be part of the model [eqns (1)–(7)] solved in step (8).

Apart from the system aspects presented, further research could be done regarding the mathematical model itself. Constraints (4) and (5) presuppose the fleet size as \(n\), and also require the number of vehicles leaving the depot to equal the number returning. An alternative formulation is to replace (4) and (5) by (8)–(10):

\[
\sum_j X_{ij} \leq n \tag{8}
\]

\[
\sum_i X_{iv} \leq n \tag{9}
\]

\[
\sum_j X_{ij} - \sum_i X_{iv} = 0. \tag{10}
\]
DATA BASE
1. School locations and schedules (start- and end-times) : depot location(s).
2. Definition of tasks (see figure 1).
3. Distances between schools and to depot.

INPUT GENERATOR
4. Specify tasks to be scheduled.
5. Get schools' schedules and distance data.

SOLUTION PROCEDURE
6. Apply constraint (6) to eliminate infeasible arcs.
7. Develop formulation as an Assignment problem (Table 3).
8. Create file in required format, e.g. MPS; call program, e.g. MINOS, to solve problem.

REPORT WRITER
9. Translate basis (those tasks connected by arcs) into readable routes and schedules

Fig. 8. Outline of a computer system for the program scheduling problem.

This allows some experimentation with the number of buses \( m \), which no longer appears explicitly but of course is bounded above by the number of tasks \( n \).

Recall that we never actually used constraints (4) and (5), in order that correspondence could be made with the assignment problem (see Table 3). Thus, to apply the alternative formulation with equations (8)–(10), it appears that an algorithm for mixed-integer programming would be necessary.

Finally, the question of labor relations is worth noting. The alternative formulation would be of interest in the unionized environment at the Board, where work must be provided to, or shared between, a given number of drivers. Although our results do dominate the current schedules, the union must clearly be consulted. Indeed, changes to drivers’ existing schedules would often be resisted, when such changes require additional work in the same time interval. These improvements could only be implemented slowly, over a period of perhaps several years. It is interesting that, by attrition, the drivers’ new manual schedules could be brought more in line with what OR methods might suggest, during the same period that an information system to support such an algorithm were being developed.

Acknowledgements—It is a pleasure to thank, for their cooperation in this work, the staff of the Transportation and Data Processing Departments at the Durham Board of Education. This project was also partially supported by the Natural Sciences and Engineering Research Council of Canada through Grant No. A5292.

REFERENCES