University of Leeds
SCHOOL OF COMPUTING
RESEARCH REPORT SERIES
Report 2001.03

TRACS to Better Schedules

by

Sarah Fores, Les Proll & Anthony Wren

January 2001

We discuss the driver scheduling problem in public transport and describe a combined integer linear programming/heuristic approach to its solution. The approach has been applied successfully in many operational and planning scenarios. Recent developments in the algorithms used allow the solution of very large bus and rail problems.

Keywords: scheduling, integer programming, heuristics, transport

---

Introduction

Public transport is a major contributor to the economy of every advanced society, and OR can provide significant contributions to its well-being.[1] The United Kingdom has been at the forefront of such contributions since the 1960s.[2–9]

We briefly run through the stages involved in planning the operations of buses. Subsequently we shall highlight some special considerations that apply to rail driver scheduling. The stages indicated here do not all apply in all circumstances. They are not independent, and often the process involves much backtracking.

- Plan the network (often evolves over time)
- Design the routes, including travel times
- Decide the departure times
- Allocate buses to the journeys
- Integrate schedules over larger network, including odd special journeys
- Compile crew schedules to cover all bus working
- Construct roster to include all crew duties over a period of weeks
- Produce documentation

Planning the network

We concentrate here on urban networks. Bus networks are seldom planned afresh. Transport networks have evolved, and demand patterns have grown to follow the existing network. A major revision is likely to upset many established journeys to work. Thus changes in land use (e.g. new housing developments) are often catered for by piecemeal adjustment of existing bus routes.

Network revision, when done, is often carried out on a sector of a city, based on a main corridor, or in conjunction with a major new development, either in land use or in transport infrastructure.

There have, however, been instances of completely new networks being introduced overnight. One of the earliest of these, in Wallasey, was based on the work of Lampkin and Saalmans.[3] Heuristics were designed to construct a set of possible bus routes based on origin-destination data. This set was supplemented by manually designed routes, and further heuristics selected a subset, with associated service frequencies, which matched the demand as closely as possible within financial constraints. Similar projects were carried out in other British cities during the 1970s, notably in Coventry.[5]

Deregulated commercial environments have made network planning difficult; often the public authority simply has to arrange to plug gaps in a set of commercial routes.

Designing the route

This is more than simply drawing a line on a map. A bus route may be a compromise between the provision of fast direct links and meeting local needs, such as diversions to visit shopping centres or to cover housing development. Consideration should be given to the number of vehicles required to operate the route at different times; this will depend on demand, but also on route length and travel conditions. Such considerations were implicit in the above projects.[3,5]

Deciding the departure times

At certain times of day departure times may be governed (or at least influenced) by specific needs, for example to get workers to a factory before the start of a shift. On routes where only an infrequent service is justified, there will be pressure to ensure that a journey arrives in a city centre shortly before shops or offices open. However, such demand is often present on all routes simultaneously, and it may not be possible to satisfy it entirely within the resources available.

Most urban bus scheduling in the UK is concerned with journeys at regular intervals. Departure times for both directions of travel on a route have to be planned together such that buses are not waiting for wasteful periods at termini. Often several routes run along a common corridor, and it is necessary to plan carefully for an integrated service pattern in which regular intervals between the different services are achieved if possible.[7]

Allocating buses to the journeys

Often the allocation of buses to journeys will follow automatically from the above decisions. A bus will be allocated to the first journey of the day from one of the termini. When it arrives at the opposite terminus it will be assigned to the next available departure, as it will be when it returns to the first terminus. Under constant regular operation, the number of departures from the first terminus until the first allocated bus returns there will determine the number of buses required. Allocation of vehicles to the early journeys from the second terminus is done by extrapolation back from the first terminus. However, where more than one route serves the same terminus, economies may often be achieved by interworking buses between routes. These relatively simple processes were addressed in the TASC system,[10] subsequently incorporated in BUSMAN.[11]

Integrating schedules over larger network, including odd special journeys

Where there are irregular services, or special journeys (e.g., to schools, hospitals or work places), significant savings can often be achieved by careful integration of these journeys with other services. Indeed, a series of projects throughout the 1970s using the VAMPIRES heuristic[4,10] resulted in the saving of over 100 buses across fifteen bus companies.

Compiling crew schedules to cover all bus working

Clearly, every bus must have a driver allocated to it at all times it is in motion (and usually while it is stationary at termini). Drivers work planned shifts which are subject to certain rules, e.g.,

- No-one will work more than five hours without a meal break;
- Meal breaks must be at least 30 minutes;
- No shift will contain more than nine hours work;
- No shift will spread over more than 12 hours.

Driver (or crew) scheduling is the task of drawing up a set of shifts which obey the rules and together cover all the bus work. Each shift will have a cost depending on its work content; this may include special payments for unpopular features. Other unpopular features may be undesirable, while not attracting a specific cost.

A driver schedule is normally presented as a list of shifts, each specifying the work to be done by one person in a day; usually this list shows for each shift the buses involved, and the starting and finishing times on each bus.

Constructing roster to include all crew duties over a period of weeks

In any week a driver will work a certain number of days, and there may be limits on the total number of hours worked in a week. The work done by a single driver in a week is sometimes called a rota. It is necessary to plan the rotas so that every shift on all days of the week is included in exactly one rota.

In most countries a rota will contain two rest days. In some cases, a rota will contain the same shift on every worked weekday, and similar shifts on Saturday and Sunday, if worked; in others, shifts of different lengths are grouped together to achieve a target work total. A roster consists of a list of rotas. Drivers may rotate through the whole list, or work the same rota every week. An early example of the combination of heuristics with a variant of the assignment problem to produce a roster for bus drivers is provided by Bennett and Potts,[12] while similar approaches have been developed for use in Germany[13] and the UK.

Producing documentation

It is important that the documentation used by a transport company is consistent. The driver schedule must refer to vehicles which are in suitable places for driver change-overs according to the vehicle schedule; the roster must contain shifts present in the driver schedule; documents must show for each vehicle the times it is scheduled to leave and return to the depot, and the daily scheduled distance; inspectors at certain points on the network require lists of the times vehicles are scheduled to pass them, perhaps with indications of the drivers involved.

Other documents must enable supervisors to record variations from planned operations. These should be linked to payment systems so that drivers are properly recompensed for the time they have actually worked. Once a system has been developed to accomplish the scheduling tasks automatically, the production of relevant documents is straightforward.

Currently most of the stages are treated independently. As both vehicle and driver scheduling problems are NP-hard, this is hardly surprising, although there have been attempts[14] to integrate the two. In this paper we concentrate on the driver scheduling problem. Annual driver costs in the UK bus and rail industries amount to several hundred million pounds so that even small savings in percentage terms are economically significant. We describe an integer linear programming (IP) approach to the driver scheduling problem and discuss some of its successful applications.

Bus and Train Driver Scheduling

A vehicle schedule may typically be depicted by several horizontal lines, each representing the work of one vehicle throughout the day. Marked along these lines are times at which it is possible to change the driver. These times, along with the location of change, are known as *relief opportunities*. The indivisible period between each pair of relief opportunities is known as a *piece of work*. Figure 1 shows a section of vehicle work and an example of a valid shift that might be formed.

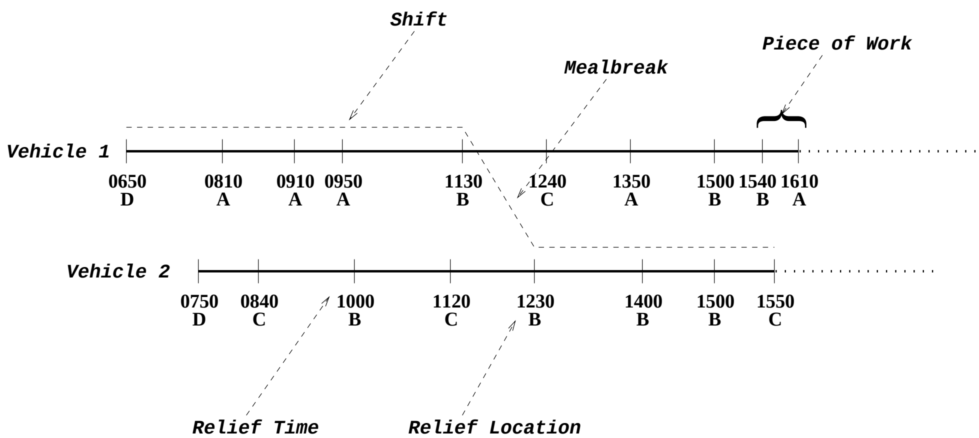

Figure 1: A section of vehicle work

The rail driver scheduling problem is more complex than bus driver scheduling, for example,

- Bus driver scheduling is normally concerned with a single depot, while rail driver scheduling involves several depots;
- Rail Drivers are restricted to routes which they know, as significant training is necessary to drive on a new route. In practice this means that drivers from a particular depot may only drive on certain routes, although there is significant overlap between the routes covered by different depots;
- Drivers from a particular train driver depot may only drive certain types of traction, although there again is overlap;
- Train schedules are more fragmented than bus schedules, often with considerable periods during which trains can be left unattended, so that logically, the resumption of a train after such a period is equivalent to the start of a new train. However, if a train is left unattended, extra time has to be given for the driver to close down the train, and for another driver to restart it;
- Train drivers often travel as passengers on scheduled trains between portions of their shifts, whereas bus drivers usually walk between portions, or are given a fixed time allowance to ride between points;
- There are often very wide differences between the lengths of different train driver shifts. This means that the composition of driver shifts can be very varied.

Although labour agreements and working practices may vary significantly between operators of different vehicle types, operators of the same vehicle type, or even between different groups of drivers within the same operation, the problem of scheduling drivers remains one of minimising a cost function whilst ensuring that all vehicle work is covered and shifts conform to a set of rules. The nature of the rules impacts heavily on the difficulty of the scheduling problem.

Driver scheduling problems arising in bus networks and heavy commuter rail lines are generally more difficult than those arising in inter-urban rail and airline operations because:

- there are very frequent relief opportunities;
- there are many activities at individual relief points;
- different relief points are close together in city centres (bus);
- drivers are often relieved en-route.

These features tend to make very many more options available to a driver after he/she has been relieved, and hence make the scheduling problem more difficult. This may explain why methods allowing the solution of large air and long-distance rail problems, for example see Vance et al.[15] and Kroon and Fischetti[16] , may not transfer to the more intensive problems discussed above.

Mathematical Programming Formulations

Several authors have reported driver scheduling systems. The most successful of these[9,17,18] are based on mathematical programming, usually combined with heuristics. Optimising the driver scheduling problem using a pure IP approach would require all valid shifts to be available to the problem formulation. Since this can often represent many millions of shifts, heuristics may be applied to generate and further retain a smaller set of *good* shifts. This compromises the optimality of the solution. Also it might not be possible to find a feasible schedule on this shift subset using a set partitioning approach, and a set covering formulation is preferable. Suppose we have N feasible shifts and M pieces of work, the set covering formulation can be expressed as :

$$
\begin{array}{ll}
\text{Minimise} & \sum_{j=1}^{N} D_j x_j \\
\\
\text{Subject to} & \sum_{j=1}^{N} A_{ij} x_j \geq 1 \text{ for } i = 1, .., M \\
\\
& \text{Plus any side constraints} \\
\\
& x_j = 1 \text{ if shift } j \text{ is used in the solution}, 0 \text{ otherwise} \\
& A_{ij} = 1 \text{ if shift } j \text{ covers piece of work } i, 0 \text{ otherwise} \\
& D_j \text{ is determined by the objective function used}
\end{array}
\tag{1}
$$

The side constraints which might be added to the model limit, for example, the total number of shifts, the number of shifts at a depot, or the number of shifts of a particular type (e.g., according to the time period in which they operate).

It is possible that a resulting schedule may assign more than one driver to a piece of work, although the minimisation of a suitable cost function should limit these occurrences. Any *overcover* that does appear in the schedule can often be eliminated by manually adjusting the shifts, or by allowing one of the drivers to travel as a passenger for the overcovered time period.

Column Generation Approaches

It is possible that operating conditions are specified in such a way that the pure IP approach can solve the scheduling problem over all feasible shifts. However, in many cases management wishes to relax operating conditions to explore 'what-if' scenarios. This may lead to a large number of relief opportunities, which increases the combinatorial complexity. It is widely accepted[19] that column generation techniques can successfully be applied to problems with a very large number of columns in the coefficient matrix, such as the class of scheduling problems. The technique works by solving the linear relaxation of the problem with only a subset of the shifts available. The dual values obtained, which are attached to pieces of work, are used to identify the reduced cost of any new shift. A selection of shifts with negative reduced costs is added to the subset and the problem is re-solved. This process continues until no shifts with negative reduced costs can be found and therefore the solution is optimal over the full set of shifts.

Different techniques can be applied to identify new shifts which have negative reduced costs. Desrochers et al.[17] use a constrained shortest path (CSP) technique to generate shifts by constructing a network so that a path from source to sink represents a shift. Some of the labour agreement conditions can be applied to the network, for instance, a path cannot be created which would include a mealbreak in excess of the maximum allowed. Other rules can be catered for in the construction of a shift, for example those related to resource consumption such as time or the number of vehicles worked on. The shortest path equates to finding the shift with the most negative reduced cost, but it may be necessary to further check the validity of the shift or to include penalty costs which are determined by particular characteristics of the shift which are only known once the shift is constructed. Limits on the numbers of shifts of certain types require evaluation simultaneously over the whole schedule. It is beneficial to generate a number of shifts with negative reduced costs, rather than a single one, during each column generation subproblem in order to reduce the number of column generation iterations and achieve a better performance in the eventual search for an integer solution. Yunes et al.[20] exploit the features of a constraint programming (CP) approach which can be used to generate shifts which are valid using the labour agreement rules. They note that using a pure CP approach results in a poor performance because of the absence of lower bounds on schedule cost. The CP approach can however be used to generate shifts with negative reduced costs, and the validity of the shifts is ensured if the constraint programming model incorporates all of the labour agreement rules. Caprara et al.[21] generate shifts using a dynamic programming-based heuristic embedded in a Lagrangian relaxation approach.

We have adopted an approach in which a very large number of potential shifts (the *superset*) is generated in a pre-processing stage. Columns are then generated as necessary to correspond to shifts from this superset which have been identified as beneficial. This approach in embedded in our TRACS II system, which is now outlined.

TRACS II

TRACS II is a software system for bus and train driver scheduling developed at the University of Leeds. It was first installed for Reading Buses[22] in 1998, and is currently being installed for all the UK bus companies of FirstGroup, the UK's largest bus company.[23] In outline, the functions of TRACS II are:

1. **Generate a large number of potential shifts**;
    - Generates all possible travel opportunities for drivers travelling as passengers.
    - Generates potential shifts which would be valid according to the parameters presented to the system. The present version allows shifts driving up to four different trains or buses. The shift generation process only produces a large subset of shifts, chosen heuristically in such a

way that the most likely shifts are formed and that a good choice of shifts is available for each piece of work.

2. **Reduce these to a suitable smaller number by some heuristics**; This optional process applies filtering heuristics to each set of potential shifts generated above in order to eliminate the least efficient shifts which do not contribute significantly to the covering of any individual piece of work.

3. **Merge sets of generated shifts**; If more than one set of shifts has been generated for a problem, these sets can be merged and duplicates removed.

4. **Select from the remaining shifts a suitable schedule using integer linear programming**; This solves a set covering problem, using a combination of sophisticated processes which may be selected by the user from a range available. Details of this method are discussed in the following sections.

Mathematical Programming Solution Method

TRACS II uses a set covering model to select a schedule from a previously generated set of shifts. Although an optimal solution cannot be guaranteed over all shifts, the mathematical programming algorithms have been adapted to deal with an effectively unlimited problem size.

The user has control over some of the algorithmic parameters, and therefore the solution process is flexible and can handle different user requirements. An outline of the solution process is as follows :

1. **Create an initial solution**

   Initial solution strategies are discussed later.

2. **Solve the LP relaxation**

   - Dual Steepest Edge (For problems with fewer than 30,000 generated shifts)

     This solves the LP over all generated shifts using a dual strategy as described by Willers et al.[24]

   - Column Generation

     A primal steepest edge[25] variant of the Revised Simplex Method is used to solve the LP over a shift subset. Reduced costs of shifts outwith the subset are calculated and a selection of those which will improve the solution are added. The problem is re-solved and shifts added until no further improvement can be made. This is described in more detail later.

3. **Determine the target number of shifts**

   If the total number of shifts in the LP solution is of the form $I.f(0 < f < 1)$, and the user wishes to minimise the number of shifts, the cut

$$\sum_{j=1}^{N} x_j \geq I + 1 \quad \text{or} \quad \sum_{j=1}^{N} x_j = I + 1 \tag{2}$$

   is added and the model re-solved using a dual simplex algorithm.

4. **Reduce the problem size in terms of variables and constraints.** (Optional)

   Smith[6] noted that banning shifts which do not use the relief opportunities identified in the relaxed solution considerably reduces the time taken in branch and bound without any deterioration in the quality of the solution obtained. This procedure has been further enhanced by Willers[26] to eliminate the associated workpiece constraints yielding a substantial further reduction in branch asnd bound time.

5. **Find an integer solution using branch and bound.**

   This is described in more detail later.

Objective functions. There are essentially six objective functions available to the user. The equations given in **(3)** show the three cost functions used in the objective function to find respectively: the minimum overall shift cost, the minimum number of shifts, the minimum cost solution having the minimum no. of shifts.

$$\begin{aligned} D_j &= C_j && (i) \\ D_j &= 1 && (ii) \\ D_j &= W + C_j && (iii) \end{aligned}$$

where C_j are the costs associated with the shifts
and $W = 1 +$ sum of X largest C_j values

(3)

For all three objective functions, the user can choose to include or exclude a subjective penalty cost attached to each shift, reflecting any legal but undesirable features. Objective function **(iii)** uses a Sherali weighted cost function described by Willers et al.[24] Minimisation of the number of shifts is prioritised using a suitable upper bound on the schedule costs as a weighting factor to be added to the shift costs.

The number of shifts in the initial solution (assuming all pieces of work are covered) is a suitable upper bound (X) on the number of shifts in the solution, and the largest X shift costs in the generated set can be summed as an upper bound on the schedule cost.

Initial Solution Strategies. Several methods of constructing an initial schedule are available in TRACS II. In each case a greedy heuristic[27] is used which successively chooses a minimal cost shift to cover a previously uncovered workpiece; the difference between the methods lies in the choice of cost function. It is known that greedy heuristics generally do not find very good solutions of the set covering problem.[28] Their principal purpose here is to provide the LP with an advanced start; thus the greedy solution has to be transformed into a basic solution to the LP. If side constraints are present, the greedy solution may not be feasible. In this case a Phase 1 procedure is performed.

Column Generation. All generated shifts are considered when forming the initial solution, based upon one of the strategies described above. The pieces of work are searched in increasing order of the number of shifts available which cover them. As a currently uncovered piece of work is being considered, all shifts covering it are evaluated and the best, according to the greedy cost function, is included in the initial solution. Some of these shifts are also selected to be included in the initial shift subset, in order to ensure where possible that a certain number of shifts (currently 10) are available to cover each piece of work. Although not every piece of work is explicitly considered when finding the initial solution, the inclusion of all shifts in the initial solution in the initial subset ensures that all work can be covered if possible.

A primal simplex approach is then used to solve the relaxed model on the shift subset in order to obtain an upper bound and generate dual values. The reduced cost of any new shift is :

$$D_k - \sum_{i=1}^{M} \pi_i a_{ik}$$

$D_k =$ cost of shift k,

$\pi_i =$ dual value of piece of work i,

$a_{ik} =$ coefficient of shift k in constraint i.

(4)

Thus for any shift not yet selected from the full shift set, which we shall define to be the *superset*, the reduced cost can be calculated. The greatest rate of improvement in the solution should correspond to introducing the shift with the most negative reduced cost. The relative variation in shift costs will be small, especially with the Sherali weight added, and so it is more likely that shifts covering pieces of work with higher simplex multipliers will be more useful. Ultimately all remaining shifts should be evaluated, but the strategy of making larger improvements to the solution in the earlier stages should reduce the number of potential shifts to be added later. The pieces of work are therefore considered in decreasing order of their associated dual values. For the current piece of work, the reduced costs of the shifts which cover it in the superset are evaluated. A number of parameters control the number of pieces of work examined in an iteration and the number of shifts added for each piece. This strategy is used in order to maintain sufficient variety in the subset.

For problems which contain more than 30,000 shifts, a column generation solution approach is automatically adopted. It is also possible to request a column generation approach for fewer shifts, but experimentation has shown that there is often no great time saving at the risk of having a prohibitively small shift set available to the branch and bound tree. Since we have specified an upper limit of 30,000 shifts which can be stored internally, we have implemented an algorithm to remove shifts from the subset if the column additions reach this limit. This ensures that the optimal LP over all generated shifts is guaranteed. Shifts not in the current basis are 'overwritten' by shifts with negative reduced costs during a column generation iteration, but they can be re-introduced if necessary. A similar mechanism is used to ensure that the initial subset does not exceed 30,000 shifts. Once the shift subset had been reduced according to the relief opportunities used in the LP solution, the user may request the addition of shifts from the superset which use these relief opportunities in order to provide the branch and bound search with a larger set to select from. This is often not necessary but provides the user with additional flexibility.

Branch and Bound. Given the solution of the LP relaxation, a limited branch and bound search is performed to find a schedule. At the simplest level a branch and bound search requires specification of rules for choosing the active node to evaluate and the entity on which to branch. The choice of node is made in a fairly conventional way, i.e. by means a composite function of the integer infeasibility of the node and the value of the objective at the node. The only sufficient condition for the LP solution at a node of the search tree to be a schedule is that all shift variables have integer values. This suggests that branching should take place on the shift variables. However such branching leads to an unbalanced search;[29] setting a shift variable to zero makes little progress as there are likely to be many other shifts which can be used to cover its work, particularly early in the search. We therefore make use of several necessary conditions to provide alternative branching rules which lead to a better balanced and more efficient search. Clearly the number of drivers who are relieved at a particular relief opportunity must be an integer. If this is not so in the LP solution, we can branch to ban, on the one hand, all shifts which use this relief opportunity and, on the other, all shifts which do. This provides a set-based branching rule. Other necessary conditions, viz that the number of shifts operating out of a particular depot must be integer, and the number of the shifts of a particular type must be integer, provide further branching rules. In these instances, appropriate upper and lower bounds on the number of shifts are imposed as local cuts.

The benefits of relief opportunity branching have long been established[6] but, whilst they have shown some promise, the effectiveness of depot and type branching and the precise way in which they are integrated with the other branching rules is still under investigation.

Application

TRACS II and its predecessor IMPACS, which was incorporated into the BUSMAN system,[11] have been applied to driver scheduling problems in the UK, Australia, Hong Kong, South Africa and Zimbabwe and in bus, tram, and surface and underground rail operations. Typically savings of 1-5% of driver costs have been achieved. Within the BUSMAN system, IMPACS was limited to problems of about 350 pieces of work and 5000 generated shifts. Larger problems had to be decomposed; the decomposition into subproblems requires vehicle trips to be allocated to a particular subproblem. If, for example, the problem is decomposed by depot and a trip can be operated by either of two depots, the trip will have to be allocated to one of the depots without knowledge of how well it will integrate with other work allocated to that depot. Efficient decomposition requires skill and experience and may require some iterations. Whilst some improvement could be expected from advances in computer technology, the principal boost to TRACS II's capability arises from the algorithmic developments described earlier and their efficient implementation.

As an example of the improved capability, we recently revisited rail scenarios involving two different UK train operating companies which we had tackled during 1996-98. In both cases the then limits of TRACS II forced us into decomposition; this is no longer necessary with its enhanced capability. In one case we were only able to reduce the number of shifts used to 169 from 170. However the decomposition into two subproblems had taken several days, and investigating the effect of changes in working practices over the whole network was not viable. In the other case the ability to handle the five subproblems as a single problem lead to a reduction in shifts used from 326 to 317. In this case the deadline on the study did not allow the same attention to be paid to the decomposition. Fuller details of these problems are given in Fores et al.[30]

Recently we have successfully solved a complex bus problem arising in a major British city involving 1500 workpieces and in excess of 1.5 million generated shifts in approximately 4 hours on a 650 MHz

Pentium III. This large problem arose because the user correctly anticipated that a better solution could be found compared with the solution obtained by combining the shifts found in the decomposed problem. The parameters controlling shift generation were relaxed to allow relatively short working time on vehicles to provide as large and varied a set of shifts to the IP as possible.

Conclusions

The commercially sensitive nature of the data for driver scheduling problems makes direct comparison of TRACS II with other systems impossible and consequently we do not make claims about its relative performance. We can, however, claim that TRACS II produces acceptable results in, and is readily portable to, a variety of operational environments.[31] Enhancements to the mathematical programming algorithms, together with TRACS II's inherent flexibility, enable the solution of increasingly large and complex driver scheduling problems.

Acknowledgement

We are grateful to the UK Engineering and Physical Sciences Research Council for financial support under grant GR/M23205.

References

[1] A. Odoni and N. H. M. Wilson. Transportation. In S. M. Pollock, M. H. Rothkopf, and A. Barnett, editors, *Handbook of Operations Research and Management Science 6: Operations Research and the Public Sector*, chapter 7. Elsevier Science, 1994.

[2] K. Wolfenden and A. Wren. Locomotive scheduling by computer. In *British Joint Computer Conference, IEE conference publication 19*, pages 31–37. 1966.

[3] W. Lampkin and P. D. Saalmans. The design of routes, service frequencies and schedules for a municipal bus undertaking: a case study. *Operational Research Quarterly*, 18:375–397, 1967.

[4] A. Wren. Bus scheduling: an interactive computer method. *Transportation Planning and Technology*, 1:115–122, 1972.

[5] A. J. P. Percival. The Coventry bus routing project. In *Proceedings of the Fifth Annual Seminar on Bus Operations Research*, pages 13.1–13.4. University of Leeds, 1973.

[6] B. M. Smith and A. Wren. A bus crew scheduling system using a set covering formulation. *Transportation Research*, 22A:97 – 108, 1988.

[7] R. S. K. Kwan. Co-ordination of joint headways. In J. R. Daduna and A. Wren, editors, *Computer-aided transit scheduling*, pages 304–314. Springer-Verlag, 1988.

[8] A. Wren and B. M. Smith. Experiences with a crew scheduling system based on set covering. In J. R. Daduna and A. Wren, editors, *Computer-Aided Transit Scheduling*, pages 104–118. Springer-Verlag, 1988.

[9] S. Fores, L. G. Proll, and A. Wren. An improved ILP system for driver scheduling. In *Computer-Aided Transit Scheduling*, pages 43–62. Springer-Verlag, 1999.

[10] B. M. Smith and A. Wren. VAMPIRES and TASC: two successfully applied bus scheduling programs. In A. Wren, editor, *Computer Scheduling of Public Transport*, pages 97–124. North Holland, 1981.

[11] R. H. Williamson. BUSMAN: the United Kingdom's integrated approach to transit scheduling. In J. M. Rousseau, editor, *Computer Scheduling of Public Transport 2*, pages 19–44. North-Holland, 1985.

[12] B. T. Bennett and R. B. Potts. Rotating roster for a transit system. *Transportation Science*, 2:14–34, 1968.

[13] J. R. Daduna and M Mojsilovic. Computer-aided vehicle and duty scheduling system using the HOT programme system. In J. R. Daduna and A. Wren, editors, *Computer-Aided Transit Scheduling*, pages 133–146. Springer-Verlag, 1988.

[14] R. Freling, A. P. M. Wagelmans, and J. M. P. Paixáo. An overview of models and techniques for integrating vehicle and crew scheduling. In N. H. M. Wilson, editor, *Computer-Aided Transit Scheduling*, pages 441–460. Springer-Verlag, 1999.

[15] P. H. Vance, C. Barnhart, E. L. Johnson, and G. L. Nemhauser. Airline crew scheduling: a new formulation and decomposition algorithm. *Operations Research*, 45:188–200, 1997.

[16] L. Kroon and M. Fischetti. Crew scheduling for Netherlands Railways "Destination Customer". In *CASPT2000 - the 8th International Conference on Computer Aided Scheduling of Public Transport*, June 2000.

[17] M. Desrochers and F. Soumis. A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23:1–13, 1989.

[18] C. Falkner J and D. M. Ryan. EXPRESS: set partitioning for bus crew scheduling in Christchurch. In M. Desrochers and J. M. Rousseau, editors, *Computer-aided transit scheduling*, pages 359–378. Springer-Verlag, 1992.

[19] L. A. Wolsey. *Integer Programming*. John Wiley & Sons, 1998.

[20] T. H. Yunes, A. V. Moura, and C. C. de Souza. Hybrid column generation approaches for solving real world crew management problems. *Relatório Técnico IC-00-18*, 2000.

[21] A. Caprara, M. Fischetti, P. L. Guida, P. Toth, and D. Vigo. Solution of large-scale railway crew planning problems: the Italian experience. In N. H. M. Wilson, editor, *Computer-Aided Transit Scheduling*, pages 1–18. Springer-Verlag, 1999.

[22] A. Wren and R. S. K. Kwan. Installing an urban transport scheduling system. *Journal of Scheduling*, 2:3–17, 1999.

[23] M. Meilton. Selecting and implementing a computer aided scheduling system for a large bus company. In *CASPT2000 - the 8th International Conference on Computer Aided Scheduling of Public Transport*, June 2000.

[24] W. P. Willers, L. G. Proll, and A. Wren. A dual strategy for solving the linear programming relaxation of a driver scheduling system. *Annals of Operations Research*, 58:519–531, 1995.

[25] J. J. Forrest and D. Goldfarb. Steepest-edge simplex algorithms for linear programming. *Mathematical Programming*, 57:341–3741, 1992.

[26] W. P. Willers. *Improved Algorithms for Bus Crew Scheduling*. PhD thesis, School of Computer Studies, 1995.

[27] E. Silver, R. V. Vidal, and D. de Werra. A tutorial on heuristic methods. *European Journal on Operations Research*, 5:153–162, 1980.

[28] E. Balas and M. C. Carrera. A dynamic subgradient-based branch and bound procedure for set covering. *Operations Research*, 44:875–890, 1996.

[29] E. M. L. Beale and J. A. Tomlin. Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. In *Proceedings of the 5th International Conference of O.R.*, pages 447–454, 1970.

[30] S. Fores, L. G. Proll, and A.Wren. Experiences with a flexible driver schedule. In *CASPT2000 - the 8th International Conference on Computer Aided Scheduling of Public Transport*, June 2000.

[31] A. S. K. Kwan, R. S. K. Kwan, M. E. Parker, and A.Wren. Proving the versatility of automatic driver scheduling on difficult train and bus problems. In *CASPT2000 - the 8th International Conference on Computer Aided Scheduling of Public Transport*, June 2000.