
Proceedings Paper:
Fores, S, Proll, L and Wren, A (1999) An improved ILP system for driver scheduling. In: Wilson, NMH, (ed.) Computer-Aided Transit Scheduling. 7th International Workshop on Computer-Aided Scheduling of Public Transport, 05-08 Aug 1997, Cambridge, MA, USA. Lecture Notes in Economics and Mathematical Systems (471). Springer-Verlag , Berlin , pp. 43-61. ISBN 978-3-540-65775-0

https://doi.org/10.1007/978-3-642-85970-0_3

An Improved ILP System For Driver Scheduling

Sarah Fores, Les Proll and Anthony Wren
Scheduling and Constraint Management Group
School of Computer Studies
University of Leeds
Leeds LS2 9JT
UK

Abstract: Mathematical programming approaches to driver scheduling have been reported at many previous workshops and have become the dominant approach to the problem. However the problem frequently is too large for mathematical programming to be able to guarantee an optimal schedule. TRACS II, developed at the University of Leeds, is one such mathematical programming-based scheduling system. Several improvements and alternative solution methods have now been incorporated into the mathematical programming component of the TRACS II system, including a column generation technique which implicitly considers many more valid shifts than standard linear programming approaches. All improvements and alternative strategies have been implemented into the mathematical programming component of TRACS II to allow different solution methods to be used where necessary, and to solve larger problems in a single pass, as well as to produce better solutions. Comparative results on real-world problems are presented.

1 Introduction

The problem of scheduling public transport vehicles and their drivers has been the subject of six international workshops (see, e.g., Desrochers/Rousseau (1992), Daduna/Branco/Paixáo (1995)). With improvements in technology, mathematical programming solution methods are increasingly successful. One of the driver scheduling systems using a mathematical programming approach is TRACS II which was originally developed from the IMPACS system (see Smith/Wren (1988), Wren/Smith (1988)) and which uses a set covering formulation to ensure that all vehicle work is covered. This paper outlines several improvements incorporated into the mathematical programming component of the TRACS II system. Results on a selection of bus and train problems are reported.

2 The Driver Scheduling Problem

Typically a vehicle schedule is produced for which drivers need to be allocated. The vehicle schedule is represented graphically by a series of lines each of which depict the movements of a vehicle during the day, and at various stages the vehicle will pass a location which is convenient for driver changeovers. These location/time pairs are known as *relief opportunities*, and an indivisible period between any two relief opportunities is known as a *piece of work*. Each piece of work then needs to be allocated to a driver so that the minimum number of drivers is required and costs are minimised. The work of a driver in a day is known as a *shift*, which usually consists of two to four spells of work each of which covers several consecutive pieces of work on the same vehicle. The formation of shifts is governed by a set of labour agreement rules to ensure that there is adequate provision for mealbreaks and acceptable working hours etc.

It is possible to formulate the driver scheduling problem as a set covering or set partitioning problem which ensures that all of the vehicle work is covered. A suitable objective function can then be devised which ensures shift and cost minimisation over a set of previously generated valid shifts. The difficulty with this approach is that the number of valid shifts is too large for this approach to be able to guarantee an optimal schedule in most cases. Thus mathematical programming is frequently combined with heuristic approaches to provide a viable solution method.

3 The TRACS II System

TRACS II is the driver scheduling system which was developed at the University of Leeds, and which originated from the commercially available IMPACS system (see Smith/Wren (1988), Wren/Smith (1988)). TRACS II has since been altered to incorporate features required in order to schedule train crews, (see Wren/Kwan/Parker (1994), Kwan/Kwan/Parker/Wren (1996)) and many of the algorithms contained in the individual components have been improved. In principle though, the TRACS II system exhibits a similar overall solution method to the IMPACS system:

- **Stage 1** *Generate a set of shifts which are valid according to labour agreement rules.*

  It should be noted that in a practical problem there are generally many million of such potential shifts. 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. These heuristics have proved effective in a wide range of practical applications.

- **Stage 2** *If necessary, reduce the size of the generated shift set.*

  Where a standard Integer Linear Programming (ILP) approach is used it optimises over the whole shift set, and this is limited in terms of data storage and is time-consuming, even with improvements in technology and algorithms. Heuristic methods have been developed which attempt to reduce the size of the set, while retaining the best possible subset of shifts.

- **Stage 3** *Select from the shift set a subset which covers the vehicle work.*

The problem is one of minimising the overall schedule cost, which includes a wage cost and sometimes a subjective cost reflecting penalties for shifts containing undesirable features. As the introduction of each driver incurs a large cost the primary objective is to minimise the number of shifts. Since the size of shift set originally generated is limited, it may not be possible to use a set partitioning approach which would imply that each piece of work should be covered by exactly one driver. A set covering model is used which guarantees that each piece of work is covered by at least one driver and the details of this method are discussed in the following sections.

4 Original ILP Solution Method (ZIP)

The solution method is referred to as ZIP (Zero-one Integer Programming). Given a problem with M pieces of work in the vehicle schedule, and a previously generated set of N shifts, we can define :

For $\quad j = 1, .., N$

$$x_j = \begin{cases} 1 & \text{if shift } j \text{ is used in the solution} \\ 0 & \text{otherwise} \end{cases}$$

For $\quad i = 1, .., M$

$$u_i = \begin{cases} 1 & \text{if workpiece } i \text{ is uncovered} \\ 0 & \text{otherwise} \end{cases}$$

o_i = number of times that workpiece i is overcovered

(4.1)

The existence of the variables u_i and o_i allows workpieces to remain uncovered and have more than one shift covering them respectively. The desired situation would be where a piece of work is covered by exactly one shift so that the variables u_i and o_i would both have the value zero. Where overcover remains in the final schedule, the relevant shifts can be edited to form shorter shifts which were previously excluded at Stage 2.

4.1 The Objective Function

The complexity of requirements to produce an efficient driver schedule leads to several, sometimes conflicting, objectives being necessary. These arise out of the need to assign a driver to every piece of work and the desire to produce schedules that are satisfactory both from a management and a driver viewpoint. Normally these are addressed in the following decreasing order of importance.

- To minimise the number of uncovered pieces of work; the minimum will normally be zero.
- To minimise the number of shifts used in the schedule. This is given a high priority due to the large overheads associated with employing staff.
- To avoid shifts which contain undesirable features. It is possible to form a schedule consisting of many undesirable shifts with a relatively low wage cost but it is preferable to encourage more acceptable working conditions.

- To minimise wage costs. Wage costs can be affected by overall and spell durations and the wage cost of the shift combination should be reduced.
- To minimise the total duration of overcovered pieces of work.

This can be represented by an objective function of the form :

$$\text{Minimise} \quad \sum_{j=1}^{N} C_j x_j + \sum_{i=1}^{M} D_i u_i + \sum_{i=1}^{M} E_i o_i. \tag{4.2}$$

Setting the D_i coefficient to a large value will address the most important objective of reducing the number of uncovered pieces of work.

The E_i coefficient on the other hand should be lower to reflect a less important objective on the duration of overcovered pieces of work. E_i is proportional to the duration of workpiece i.

The remaining objectives require the C_j coefficient to reflect shift costs, penalty costs and wage costs. However, penalty costs would normally be added in later so as not to detract from the more important objective of minimising the number of drivers. To prioritise the objectives a large constant is added to each shift cost so that minimisation favours a schedule with fewer shifts.

Constants used in weighting the objectives are defined by the scheduler.

4.2 Constraints

As mentioned earlier the model is based on set covering, because the number of shifts generated is limited to those which are deemed to be 'efficient' and many shorter shifts are discarded. Although a final schedule cannot contain pieces of work which are covered by more than one driver the set covering approach potentially allows this. The following constraint ensures that every piece of work is covered by at least one driver:

$$\sum_{j=1}^{N} A_{ij} x_j \geq 1 \text{ for } i = 1, .., M$$

Given that each piece of work has an associated undercover and overcover variable, we can rewrite this as :

$$\sum_{j=1}^{N} A_{ij} x_j + u_i - o_i = 1 \text{ for } i = 1, .., M \tag{4.3}$$

where the A_{ij} identify which pieces of work are covered by which shifts:

$$A_{ij} = \begin{cases} 1 & \text{if shift j covers workpiece i} \\ 0 & \text{otherwise.} \end{cases} \tag{4.4}$$

In practice, the introduction of overcovered pieces of work in the schedule is deterred by the minimisation of overcover *and* wage costs. Although overcover is allowed, it can only be acceptable as an overlap in the middle of the day. Thus it is sensible not to allow overcover early or late in the day. Thus workpiece constraints corresponding to first pieces of work on early departing vehicles and to last pieces

of work on late arriving vehicles are defined as equations. The inclusion of such constraints is, in fact, a requirement for the validity of the constraint branching strategy used within the branch and bound algorithm. The algorithm incorporated into the mathematical programming component automatically identifies the workpieces which shall be formulated as equality constraints, ensuring that the sets of shifts covering each of these workpieces are mutually exclusive.

Hence, constraint (4.3) can now be split into the following:

$$\begin{aligned} &\textstyle\sum_{j=1}^{N} A_{ij}x_j + u_i = 1 && \text{for } i = 1,..,L \\ &\textstyle\sum_{j=1}^{N} A_{ij}x_j + u_i - o_i = 1 && \text{for } i = L+1,..,M \end{aligned} \tag{4.5}$$

where L denotes the number of workpieces which can be identified as equality constraints.

In addition to the workpiece constraints, shifts can be categorised by an initially specified type and the user can impose constraints on any of these to ensure that the final schedule does not contain too many or too few shifts of any particular type. Constraints of this form can also be used to limit the total number of shifts in the final schedule and also eliminate undercover. The constraints can be imposed either initially or when reoptimising an existing solution.

The constraints can be expressed as follows:

$$\textstyle\sum_{j=1}^{N} \delta_{kj}x_j \leq U_k, \quad \sum_{j=1}^{N} \delta_{kj}x_j \geq L_k \tag{4.6}$$

where U_k is an upper limit on the number of shifts of type k, and L_k is a lower limit on the number of shifts of type k.

The δ_{kj} are defined as :

$$\delta_{kj} = \begin{cases} 1 & \text{if } x_j \text{ is a shift of type k} \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

4.3 Original (ZIP) Model

In summary, the model can be formulated as follows :

$$\begin{aligned} \text{Minimise} \quad & \textstyle\sum_{j=1}^{N} C_jX_j + \sum_{i=1}^{M} D_iu_i + \sum_{i=1}^{M} E_io_i \\ \text{Subject to} \quad & \textstyle\sum_{j=1}^{N} A_{ij}x_j + u_i = 1 && \text{for } i = 1,..,L \\ & \textstyle\sum_{j=1}^{N} A_{ij}x_j + u_i - o_i = 1 && \text{for } i = L+1,..,M \\ & \text{Plus any user-defined side constraints} \\ & x_j = 0 \text{ or } 1, && \text{for } j = 1,..,N \\ & u_i \geq 0 \end{aligned}$$

$$o_i \geq 0 \tag{4.8}$$

4.4 Method of Solution of the Model

It is possible to execute the model solver in a number of different ways, such as starting or stopping at different points in the solution strategy; however, the method will be described in full as if there were no user intervention in the process. The following is a summary of the stages involved in solving model (4.8).

Stage Z1

Shifts are ranked in groups in order of 'desirability' during the generation phase. The ranking is a crude measure of the ratio of work content to shift content. In the first stage of the solution all three-part shifts and shifts with a relatively low work content are temporarily excluded from the shift set. At this stage penalty costs are not included in shift costs. The integrality constraints on the shifts are relaxed so that a piece of work may be covered by fractions of several different shifts. The relaxed model is solved over the remaining shifts using a primal steepest edge (see Goldfarb/Reid (1977)) variant of the Revised Simplex Method, starting with an initial solution formed by a heuristic which considers all of the previously generated shifts.

The quality of the initial solution affects the number of subsequent iterations required to find the optimal continuous solution. It is developed heuristically and has to be transformed into a basic solution to (4.8) in order to provide an advanced start for the revised simplex method. The heuristic considers each piece of currently uncovered work in ascending order of the number of shifts available to cover it, and selects a shift to be included in the initial solution which minimises the following cost function:

$$\text{Minimise} \quad (\tfrac{C_j}{NU_j}) + OC_j \tag{4.9}$$

where
C_j = cost of shift j
NU_j = number of currently uncovered workpieces covered by shift j
OC_j = increase in overcover costs caused by selecting shift j.

provided its inclusion does not violate a side constraint or an equality workpiece constraint. In order to satisfy a realistic upper limit on the number of shifts of certain types or on the total number of shifts it is possible that some workpieces will remain uncovered. For this reason the 'no uncovered work' constraint is not applied initially.

Stage Z2

The previously excluded shifts are restored and the LP relaxation is reoptimised with the complete set of shifts, again using the primal steepest edge algorithm.

Stage Z3

Having found the optimal Stage Z2 solution, the penalty costs are added to all appropriate shifts. These are not normally added initially because shifts displaying several undesirable features incur a very high cost, and although it is hoped that these shifts do not appear in a schedule it is actually preferable to include some of them rather than to exceed the minimum number of shifts. For this reason the solution strategy has historically been to devise two pre-emptively ordered objectives which are solved in stages. The first is to minimise a function combining the costs of the individual shifts with a large fixed cost per shift, so as to give a significant bias towards minimising the number of shifts; the second is to minimise a cost function, incorporating penalty costs, with the added constraint that the total number of shifts does not exceed the number already ascertained at the end of the first stage. These objectives can be formulated as follows:

$$\begin{aligned} &1) \quad \text{Minimise} \quad \textstyle\sum_{j=1}^{N} C_{1j}x_j + \sum_{i=1}^{M} D_i u_i + \sum_{i=1}^{M} E_i o_i \\ &2) \quad \text{Minimise} \quad \textstyle\sum_{j=1}^{N} (C_{1j} + C_{2j})x_j + \sum_{i=1}^{M} D_i u_i + \sum_{i=1}^{M} E_i o_i \end{aligned} \tag{4.10}$$

where
C_{1j} includes wage costs and the large constant shift cost
C_{2j} is the appropriate penalty cost.

In order to address the objective prioritisation the following two side constraints are now added :

$$\begin{aligned} &\textstyle\sum_{i=1}^{M} u_i \leq 0 \\ &\textstyle\sum_{j=1}^{N} x_j \leq T. \end{aligned} \tag{4.11}$$

The first ensures that there can be no uncovered work in the solution. The second constraint uses a target number of shifts to ensure that the minimisation of the number of shifts in the solution is still addressed. If the number of shifts at the end of Stage Z2 is integral then T will take this value. However, it is more likely that the number of shifts at the end of Stage Z2 is fractional and, since an integer solution will be the final requirement, T will take this number rounded up to the next integer. This new model is now reoptimised by primal steepest edge.

Stage Z4

If the total number of shifts in the optimal Stage Z3 solution is non-integral and of the form $I.f(0 < f < 1)$, the side constraint

$$\textstyle\sum_{j=1}^{N} x_j \geq I + 1 \tag{4.12}$$

is added. This is because the current solution is minimal for a relaxed LP model, and so any integer solution must require at least the next highest integer number of shifts. The current solution will be infeasible for this new constraint and the new

model must be re-solved using a Dual Simplex algorithm.

If the total number of shifts in the current solution is already integral then no side constraint is added and no reoptimisation performed.

Stage Z5

If the current solution is fractional then the shift set is first reduced and a branch and bound method is used to determine whether a particular shift should be included in the schedule or not.

Smith/Wren (1988) developed a technique to reduce the size of the shift set entering the branch and bound phase, which assumes that an acceptable integer solution can be found by restricting the choice of relief opportunities to the subset used in the LP optimum. In this way a shift which does not use any of these times can be excluded from the search. The *reduce* procedure uses the principle that the LP optimum gives a good indication of how the vehicle work would be covered in an integer solution, and this has been confirmed by extensive practical application. Application of *reduce* typically leads to a reduction in the number of shifts considered at the branch and bound stage of 50-80%, leading to a much reduced solution time. It is possible that the constraint limiting the search to find a schedule with an exact number of shifts may not be satisfied with the limited set of relief opportunities. However, in general there are many different integer solutions, some of which should be contained within the shift set available. The other possibility is that the reduced shift set produces a higher cost schedule than one which would be produced with the whole set, but the process is normally designed to terminate with the first good solution and its quality is largely dependent on the choice of path through the tree, rather than on the number of shifts which are considered.

The branch and bound method has been developed with an emphasis on finding a good integer solution quickly. The integer solution is found by developing a branch and bound tree, in which the lower bound on the objective cost is given by the optimal continuous solution. Once an integer solution has been found the nodes of the tree are fathomed if their cost is greater than or equal to a scheduler specified percentage of the current best integer cost. In most cases the first integer solution found fathoms all of the remaining active nodes and hence the branch and bound process normally terminates with a possibly non-optimal integer solution. A maximum of 500 nodes can be created. The branch and bound process uses a specialised relief time branching strategy developed by Smith/Wren (1988).

5 Current ILP Solution Method (SCHEDULE)

5.1 Sherali Weighted Objective Function

Sherali (1982) noted that there are several disadvantages in using a sequential approach to find a solution which satisfies two objectives. Apart from the fact that two separate linear programs have to be solved which are essentially doing the same task but with a different cost coefficient, a high degree of degeneracy is likely to occur, with typically large numbers of iterations. Also the introduction of side constraints to maintain the subjective ordering of the objectives results in a more complex model to be solved which may increase execution times over a simple model for which efficient solution codes exist.

An alternative approach to solving the driver scheduling model has been explored in Willers/Proll/Wren (1993), Willers (1995). The approach combines the

objectives of minimising the number of shifts and the total shift cost, whilst retaining their preference ordering.

The two objective functions (4.11) can be simplified. The solution strategy adopted by ZIP involves adding a side constraint to prohibit any undercover and therefore it is unnecessary for the u_i variables to appear in the model. Overcover is unproductive and is discouraged by the minimisation of wage costs and so zero costs can be attached to the o_i variables. Also since the more important objective is to minimise the number of shifts, all of the shift costs can be attached in the second phase. This removes the need to include a large prioritising constant to every shift. The two objective functions can now be expressed as :

$$\begin{array}{lll} 1) & \text{Minimise} & \sum_{j=1}^{N} x_j \\ 2) & \text{Minimise} & \sum_{j=1}^{N} C_j x_j \end{array} \tag{5.1}$$

where
C_j combines wage and penalty costs for shift j.

Willers implemented a method of converting multi-objective models into single objective models. This involved weighting the objectives and combining them as follows :

$$\text{Minimise} \quad W_1 \sum_{j=1}^{N} x_j + W_2 \sum_{j=1}^{N} C_j x_j. \tag{5.2}$$

The weights W_1 and W_2 must be selected so as to ensure that the objectives are correctly ordered. This produces:

$$W_1 = 1 + UB[\sum_{j=1}^{N} C_j x_j] - LB[\sum_{j=1}^{N} C_j x_j]$$

$$W_2 = 1 \tag{5.3}$$

where
$UB[\sum_{j=1}^{N} C_j x_j]$ is an upper bound value on the shift costs
$LB[\sum_{j=1}^{N} C_j x_j]$ is a lower bound value on the shift costs.

(5.4)

An upper bound can be calculated by setting all shift variables to 1. This upper bound however corresponds to summing all shift costs for the N shifts generated, which would produce a very large weight, potentially leading to numerical difficulties in the solution method. A smaller weight can be calculated by adapting the upper bound so that :

$W_1 = 1+$ sum of X largest C_j values

(5.5)

where X must be an upper bound on the number of shifts in the schedule. An appropriate weight can be ascertained by defining X to be the number of shifts in the initial solution plus the number of uncovered pieces since the sum of the highest X cost values in this case must be an upper bound to any further schedule costs calculated. A potentially smaller weight would result from careful calculation of the lower bound. Willers (1995) investigated several possibilities and concluded that the simple lower bound of 0 was satisfactory.

5.2 The New Model

By incorporating the Sherali weight (5.5) into the objective function and defining :

$$D_j = W_1 + C_j \qquad \text{for } j = 1, .., N \tag{5.6}$$

as the new cost coefficient for every shift variable, the new model can be defined as follows :

$$\begin{aligned}
&\text{Minimise} && \textstyle\sum_{j=1}^{N} D_j x_j && \\
&\text{Subject to} && \textstyle\sum_{j=1}^{N} A_{ij} x_j = 1 && \text{for } i = 1, .., L \\
& && \textstyle\sum_{j=1}^{N} A_{ij} x_j \geq 1 && \text{for } i = L+1, .., M \\
& && \text{Plus any user-defined side constraints} && \\
& && x_j = 0 \text{ or } 1, && \text{for } j = 1, .., N
\end{aligned} \tag{5.7}$$

It may be noted that this model, unlike (4.8), guarantees that the minimum number of shifts is obtained.

5.3 Method of Solution of the (SCHEDULE) Model

It was noted that the process of banning shifts in the first stage of the solution method did not contribute any improvement in solution time and is no longer used. The actual solution stages which remain depend upon the user's choice of solution strategy but a broad outline follows :

- Solve the LP relaxation over the whole shift set using either a conventional linear programming method or column generation. The single objective model is optimised using a primal steepest edge approach.
- Add a constraint which increases the shift total to the next highest integer and re-solve using a dual steepest edge approach due to Forrest/Goldfarb (1992).
- Reduce the number of constraints and variables and find an integer solution using the branch and bound technique used in the original model, but with reoptimisations performed by dual steepest edge.

5.3.1 Initial solution

Recognising that the overcover variables are no longer costed, a new cost function can be defined for the purpose of creating an initial solution:

$$\text{Minimise} \quad \frac{C_j}{NU_j} \tag{5.8}$$

where
C_j = combined wage and penalty costs for shift j
NU_j = number of currently uncovered workpieces covered by shift j.

The above method for constructing an initial solution is an adaptation of that used in ZIP in which the objective is cost minimisation. In SCHEDULE, the primary objective is shift minimisation and the following method addresses that more directly. If all pieces of work are to be covered by the minimum number of shifts, each newly selected shift must cover a lot of work not covered by shifts selected earlier. This suggests selecting shifts by:

$$\text{Maximise} \quad \sum_{i=1}^{M} \Delta_{ij} L_i \tag{5.9}$$

where
$\Delta_{ij} = \begin{cases} 1 \text{ if shift j covers the currently uncovered piece of work i} \\ 0 \text{ otherwise,} \end{cases}$
L_i = duration of workpiece i.

Willers' experiments (see Willers (1995)) show that while this method does not always outperform the earlier method in terms of number of shifts in the initial solution, it does so on average.

5.3.2 Column Generation

In order for a conventional mathematical programming approach to be used to solve the driver scheduling problem, heuristics are necessary to reduce the problem size. This restricts optimality to only those shifts remaining. The heuristics have been developed so that apparently inefficient shifts are removed and solutions have proved better than manual solutions. However it may be the case that some inefficient shifts link well with other shifts to produce the optimal solution. In order to guarantee optimality it must be possible to consider every valid shift.

Column generation is an approach which can be used to solve mathematical programming problems involving many variables and it is shown that column generation methods improve upon the solution and/or speed of the process. TRACS II uses a conventional mathematical programming method which requires all shifts to be available. The Revised Simplex Method is then used iteratively to improve a current solution by swapping a basic variable for a non-basic variable with favourable reduced cost, until no further improvement can be made. In the case of column generation only a subset of the columns is available at the outset. The Revised Simplex Method is used to find the solution which is optimal over the subset. One then generates new columns or searches through columns not previously considered and adds some or all shifts with favourable reduced costs as non-basic variables. The new subset is then reoptimised. For any LP relaxation which is optimal over

its available subset the overall optimal solution is attained when no more columns which could improve the objective can be added to the set.

The HASTUS scheduling system (see Rousseau/Blais (1985)) contains a module, Crew-Opt (see Desrochers/Gilbert/Sauvé/Soumis (1992)), which uses column generation techniques to form bus driver schedules and has produced encouraging results (see Rousseau (1995)). A subset of valid shifts is identified and further shifts are added to the subset based upon utilising a shortest path algorithm to generate and identify shifts which will improve the current solution. Once no new shift can be found which will reduce the schedule cost the current solution is optimal over the relaxed model. A branch and bound technique which also incorporates column generation is then applied to find a good driver schedule. Crew-Opt allows all shifts to be available by generating them as constrained shortest paths through a network. In order to ensure optimality over *all* possible valid shifts it must be possible to generate any valid shift by using a method such as that used by HASTUS, or else all shifts must be generated initially and therefore available to the process. Since TRACS II already has a good shift generation process, and the shift costs include penalty costs which are more complicated to model using a shortest path technique, the option of generating complicated shift structures through potentially large networks was not considered. The technique which was implemented considers a previously generated shift superset which allows many more shifts to be available to the mathematical programming solver. Although the continuous solution is still not guaranteed to be optimal, better solutions lead to a branch and bound search for schedules with fewer shifts. A network formulation is not used, and so shifts from this larger set are selected to enter a working subset by means of a less sophisticated enumeration method.

Fores (1996) details the column generation implementation strategies. The solution method is outlined as follows :

- **Step 0 Generate a shift superset**
  This uses the TRACS II technique of generating shifts, possibly allowing more shifts to be produced by relaxing some of the conditions set previously to restrict the shift generation. No further heuristic reduction is then necessary.

- **Step 1 Create an initial solution and form an initial shift subset**
  An initial solution is generated from the shift superset, using one of the methods described. As shifts are being considered in forming the initial solution a shift subset is also being selected. This subset needs to be sufficiently large and varied to reduce the number of shift additions required.

- **Step 2 Solve the LP over the current shift subset**
  Use the Revised Simplex Method to solve the LP over the current shift subset.

- **Step 3 Add a set of shifts to the current subset which will improve the solution**
  Simplex multipliers produced at the end of Step 2 are used to calculate the reduced costs of shifts currently not selected from the superset. The reduced cost of a shift k is defined to be :

  Minimise $(c_k - \sum_{i=1}^{M} \pi_i a_{ik})$.

(5.10)

where :
M is the number of constraints
c_k is the cost of shift k
π_i is the simplex multiplier for constraint i
a_{ik} is the coefficient of shift k in constraint i

Since the shift costs include the large Sherali weight they do not vary as significantly as the simplex multipliers. The simplex multipliers are therefore considered in decreasing order and shifts covering their corresponding pieces of work are added to the subset if they have a favourable reduced cost and if they are allowed to be added based upon parameters which control the shift addition.

- **Step 4 If no favourable shifts can be found then the LP solution is optimal, otherwise go to Step 2**

- **Step 5 Find an integer solution using branch and bound**
  Since we still cannot guarantee a relaxed solution optimal over *all* possible shifts, and the existence of subjective weightings allows ambiguity over the best cost solution, no alteration is made to the branch and bound strategy.

The column generation implementation was tested against a Sherali version of ZIP. This allowed the most accurate comparison of two single objective models. A summary of column generation results will be given in section 7.

5.3.3 A Dual Approach

It has long been known that LPs of the form (5.7) are inherently degenerate (see Marsten (1974)) and that primal simplex approaches to their solution spend many iterations making no reduction in the value of the objective function. A number of degeneracy resolving procedures (e.g. see Wolfe (1963)) have been tried in SCHEDULE without success. An alternative is to solve the LP relaxation by a dual simplex approach to lessen the effect of degeneracy. In order to use a dual approach, the initial solution has to be transformed into a basic dual feasible solution, i.e. one in which all the reduced costs are non-favourable. Beale (1968) shows how this can be done. We then find the optimal solution to the LP relaxation using a dual steepest edge algorithm (see Forrest/Goldfarb (1992)) and use an improved *reduce* procedure and branch and bound to find a good schedule.

5.3.4 Improved REDUCE

The majority of the constraints in (4.8) are associated with pieces of work, each of which starts and ends at a relief opportunity. The vehicle block

+——+————+————+——+

where + denotes a relief opportunity and - denotes a vehicle movement, would generate four workpiece constraints. As any LP solution covers all the pieces of work, the first and last relief opportunity on any block must be used by at least one of the shifts in the LP solution. Thus any unused relief opportunity separates two pieces of work. Suppose the third relief opportunity in the above block is unused and that

we eliminate it from the problem. The block would change to

and would generate only three workpiece constraints. Thus eliminating an unused relief opportunity allows one of the two workpiece constraints surrounding it to be removed. For technical reasons, it is not usually possible to remove one workpiece constraint for every unused relief opportunity and get a good starting solution to the reduced problem. Willers (1995) provides an algorithm for constructing an optimal LP basis to the reduced problem from the optimal LP basis to the original problem. This algorithm removed between 13% and 50% of the workpiece constraints over a set of twenty practical problems and also allowed the removal of additional shift variables beyond those removed by the earlier *reduce* procedure. The reduction in the number of constraints leads to reductions in the solution time of the branch and bound phase.

5.4 Other Improvements to SCHEDULE

Apart from the fact that the name of the mathematical programming component has been altered so that its function is more apparent to the users, the algorithms and code have been updated and the following two features have been added.

Since the earlier components of TRACS II have been modified to consider train crews as well as bus crews, SCHEDULE has also been altered to accept the train data generated. In particular it is now possible to define duty types rather than having specific names which differ depending on whether bus or train crews are being scheduled. Also it is necessary for SCHEDULE to handle four part shifts in train driver scheduling. In fact this has also been generalised so that shifts can be formed with any number of spells.

Since there are various different solution strategies which have been tested, those which have proved successful have been implemented in SCHEDULE. It is therefore possible for the user to select a solution strategy which would be more appropriate to a given problem. A default solution strategy can be adopted, along with various parameters which control the sensitivity of the algorithms.

6 Computational Results

One of the objectives in developing the SCHEDULE system was to allow several solution techniques to be available to the user. The viability of each method has been tested independently and it is infeasible to compare the quality and solution speed of each combination. Also, developments have taken place in parallel on different platforms and compared with versions in various stages of improvement. The justifications for including the dual approach and the column generation technique are given in the following two sections. Although these two lines of development have been pursued independently, there is some potential for integrating them which may lead to further improvement.

6.1 Dual Approach Results

Willers (1995) tested various implementations of the dual approach, including different methods of calculating the Sherali weights, different initial solution methods and different methods of determining an initial basic dual feasible solution. The best of these have been described above. On a sample of 20 bus crew scheduling problems, the dual approach reduced total solution time on a 33MHz 486 PC from 2525 minutes to 1243 minutes, an average reduction of 51% over the ZIP model.

6.2 Column Generation Results

Willers (1995) has shown that the Sherali weighted objective function gives improved results over the objectives used in the older version. The column generation implementation also uses the Sherali weighted objective function and was therefore tested against a version of ZIP using that objective in order to gauge any potential improvement made by this technique. Various improvements were made to both systems to allow the most accurate comparison.

For seven problems, two sizes of data set were used in order to test any improvements in solution or speed of solution for a problem which consisted of a larger set of previously generated shifts. The smaller set is that which would normally be run through the mathematical programming component of TRACS II. As the non column generation produces an LP solution, using the column generation method on the same data set will not improve the cost, as both will be optimal.

The following table compares the results obtained using the modified ZIP on a heuristically reduced shift set against those of a column generation system with a larger set of previously generated shifts. Timings are reported for a Silicon Graphics Iris Indigo Workstation with 33MHz R3000 MIPS Processor and are in minutes :

Data Set	Modified ZIP			Column Generation		
	Shifts	Run Time		Shifts	Run Time	
		LP	Total		LP	Total
AUC	87	13	36	87	21	91
CTJ	88	10	16	87	9	19
CTR	–	–	–	88	18	54
GMB	34	0.6	0.9	33	0.7	0.9
RI2	45	0.9	2.2	44	1.0	1.2
STK	61	5.2	16	59	21	24
SYD	56	5.4	5.8	56	15	18

Table 1: Comparison of Solutions and Timings

It can be seen from Table 6.1 that the number of shifts required was reduced in 5 out of the 7 sets. This includes the one problem where no solution could previously be found. It is difficult to compare timings through a branch and bound tree because the route through it will be different. However, it is useful to note that where there is an increase in the overall execution times, the times themselves would still be acceptable to users. Where the same size of data set were compared, results showed an average reduction in execution time of 41% using column generation (see Fores (1996)).

7 Conclusion

The original TRACS II system consistently achieves better results than those obtained by conventional methods and users are happy with the quality of the solution obtained. Improvements to the system, including the mathematical component, will therefore benefit the user in terms of ease of use and of the potential improvements in solutions. The expansion to a more generalised system allows different types and sizes of problems to be addressed. Also, the improvements in algorithms and the introduction of the Sherali weighted objective function and the column generation technique allow better solutions to be found more quickly. A parameter

driven approach gives the user more flexibility in solving problems using different solution techniques. Since the original draft of this paper, the column generation approach has been used successfully in many scheduling exercises for clients. This has given us the confidence to tackle the solution of considerably larger problems than previously.

References

Beale, E.M.L. (1968): Mathematical Programming in Practice. (Pitman) London.

Daduna, J.R., Branco, I., Paixáo, J.M.P. (eds), (1995): Proceedings of the Sixth International Workshop on Computer-Aided Scheduling of Public Transport, Computer-Aided Transit Scheduling. (Springer-Verlag) Berlin, Heidelberg, New York.

Desrochers, M., Gilbert, J., Sauvé, M., Soumis, F. (1992): CREW-OPT: subproblem modelling in a column generation approach to urban crew scheduling. in: Desrochers, M., Rousseau, J.-M. (eds): Computer-Aided Transit Scheduling. (Springer-Verlag) Berlin, Heidelberg, New York.

Desrochers, M., Rousseau, J.-M. (eds), (1992): Proceedings of the Fifth International Workshop on Computer-Aided Scheduling of Public Transport, Computer-Aided Transit Scheduling. (Springer-Verlag) Berlin, Heidelberg, New York.

Fores, S. (1996): Column Generation Approaches to Bus Driver Scheduling. PhD Thesis, University of Leeds.

Forrest, J.J., Goldfarb, D. (1992): Steepest edge simplex algorithms for linear programming. in: Mathematical Programming, 57, pp 341 - 374.

Goldfarb, D., Reid, J.K. (1977): A practicable steepest-edge simplex algorithm. in: Mathematical Programming 12, 774 - 787.

Kwan, A.S.K., Kwan, R.S.K., Parker, M.E., Wren, A. (1996): Producing train driver shifts by computer. in: Allan, J./Brebbia, C.A./Hill, R.J./Sciutto, G./Sone, S. (eds): Computers in Railways V, Volume 1: Railway Systems and Management, (Computational Mechanics Publications) Southampton, Boston.

Marsten, R.E. (1974): An algorithm for large set partitioning problems. in: Management Science 20, 774 - 787.

Rousseau, J.-M. (1995): Results obtained with Crew-Opt, a column generation method for transit crew scheduling. in: Daduna, J.R./Branco, I./Paixáo, J.M.P. (eds): Computer-Aided Transit Scheduling. (Springer-Verlag) Berlin, Heidelberg, New York.

Rousseau, J.-M., Blais, J.-Y. (1985): HASTUS: an interactive system for buses and crew scheduling. in: Rousseau, J.-M. (ed): Computer Scheduling of Public Transport 2. (North-Holland) Amsterdam, New York, Oxford.

Sherali, H.D. (1982): Equivalent weights for lexicographic multi-objective programs: characterizations and computations. in: European Journal of Operations Research, 18, 57 - 61.

Smith, B.M., Wren, A. (1988): A bus crew scheduling system using a set covering formulation. in: Transportation Research 22A, 97 - 108.

Willers, W.P. (1995): Improved Algorithms for Bus Crew Scheduling, PhD Thesis, University of Leeds.

Willers, W.P., Proll, L.G., Wren, A. (1993): A dual strategy for solving the linear programming relaxation of a driver scheduling system. in: Annals of Operations Research, 58, 519 - 531.

Wolfe, P. (1963): A technique for resolving degeneracy in linear programming. in: SIAM Journal 11, 205 - 211.

Wren, A., Kwan, R.S.K., Parker, M.E. (1994): Scheduling of rail driver duties. in: Murthy, T.K.S./Mellitt, B./Brebbia, C.A./Sciutto, G./Sone, S. (eds): Computers in Railways IV, Volume 2: Railway Operations. (Computational Mechanics Publications) Southampton, Boston.

Wren, A., Smith, B.M. (1988): Experiences with a crew scheduling system based on set covering. in: Daduna, J.R./Wren, A. (eds): Computer-Aided Transit Scheduling. (Springer-Verlag) Berlin, Heidelberg, New York.