PREPRINT

Route optimization applied to school transports – A method combining column generation with greedy heuristics

Mikael Andersson
Peter Lindroth

Department of Mathematics
CHALMERS UNIVERSITY OF TECHNOLOGY
GÖTEBORG UNIVERSITY
Göteborg Sweden 2005

Preprint Mathematics

Route optimization applied to school transports – A method combining column generation with greedy heuristics

Mikael Andersson
Peter Lindroth

CHALMERS | GÖTEBORG UNIVERSITY

Department of Mathematics
Chalmers University of Technology and Göteborg University
SE-412 96 Göteborg, Sweden
Göteborg, March 2005

ISSN 0347-2809

Matematiskt centrum
Göteborg 2005

Abstract

This thesis investigates the possibilities of using mathematical optimization techniques when planning school transports. It includes a mathematical formulation of the problem, proposes methods to solve the problem and by testing and evaluation it concludes on the pros and cons of using the developed techniques for a practical planning case.

The school transport problem, where the objective in our case is a minimization of costs, is viewed upon in an area with many schools and the vehicles performing the transports are allowed to transport pupils going to different schools at the same time. The problem for each school is identified as the well known *Vehicle Routing Problem with Time Windows*.

For solving the problem a column generation method has been implemented where routes for the individual schools are generated and by different greedy heuristics these are weaved together with other ones so that feasible routes visiting more than one school are constructed. A final problem is solved where the cheapest set of generated routes are chosen so that every pupil is served.

A comparison with an existing plan, made with a present manual tool for planning school transports, shows that by using our problem it is possible to create plans with fewer vehicles as well as shorter driving distances than in the existing one. Though, the comparison is suffering from many assumptions and most considerably the problem of valuating a plan. There is also a big flexibility in the model, where it is possible to take a limited vehicle fleet into consideration, where the different vehicles have different capacities, costs and locations.

The conclusion from the project is that it would be beneficial to implement a core in a planning tool taking advantage of optimization techniques. Since it is hard to valuate a plan and thus hard to formulate an appropriate objective and since there are many unspoken rules for an acceptable plan the manual step in the planning must still be there. The use of methods proposed in the thesis should be a complement to the manual planning, where the planner can get a good solution to start from or to see if an existing plan can be improved.

Contents

Acknowledgments

We would like to thank Michael Patriksson, supervisor and examiner and Ann-Brith Strömberg, assisting supervisor for discussions regarding the mathematical models and the methods for solving the problems.

We would also like to thank all people at Tekis AB and Kartena AB, especially Henrik Karlsson for constructing the applications in `C#` creating input data and displaying the solutions on real maps and Jonas Buud for demonstration of the original `Skolskjuts` program.

1 Introduction

Optimization is about formulating a mathematical model of a real physical problem and with the help from this model try to find the best solution to the problem. The valuation of a solution is given by an objective function representing the aim of the problem and the feasibility is given by the regulations and limitations in the problem. Important for a good solution is that the model is representative for the real problem and that the value of the solution measured by the objective is close to the optimal value for the model.

This thesis investigates the possibility, the pros and cons, of utilizing mathematical optimization techniques when planning school transports. It discusses general route optimization problems and how the school bus problem can be formulated as such. A method for solving the problem is proposed and algorithms are implemented and evaluated.

1.1 Background

Due to the Swedish legislation all children must attend school and if the distances are far enough, the municipalities are responsible for the transports of the children between home and school. The costs implied constitute a large share of the municipality budgets. In 2003, the costs for school transports in Sweden totaled more than 2 billion SEK [9].

Tekis AB [25] is a Swedish company manufacturing and selling IT systems mostly to Swedish municipalities. One of their products is the program `Skolskjuts`, a planning tool for school transports. The program is an extension to `ArcView 3` which is a geographic information system (GIS) software. The advantages of the `Skolskjuts` software are the graphical user interface and the clear and thorough reports on the routes which are automatically written. By choosing a set of bus stops to be visited by a certain route, the only optimization made is finding the shortest or fastest (by default) path visiting these.

Kartena AB [22] is a company in the same group as Tekis. Kartena develops and sells map-based products and solutions. For example, they provide the service of letting other companies show a map on their websites with a tool that finds the fastest route for customers to get to their store, and shows this route on the map together with a road description.

At Tekis there is an interest in trying to evaluate what could be accomplished by using mathematical optimization techniques to find a good set of routes serving the pupils. With `Skolskjuts` the individual routes created are optimal in the sense that they are the fastest ones visiting some given set of stops. But using the individual optimal routes together is probably not optimal since the choice of bus stops visited by a route is made manually. In an optimal solution to a vehicle routing problem, also the assignment of the bus stops to the different routes is the best one, taking advantage of geographic clustering of the bus stops and of the capacities of the vehicles as well.

There is also an interest in studying if products from Kartena could be used together with methods from `Skolskjuts` and new methods in a new school transports planning software, perhaps detached from `ArcView`.

1.2 Purpose of the thesis

The purpose of this project is to look at a development of the core in the application, the actual construction of the routes. More intelligence should be built into the system so that it can handle properties like a limited vehicle fleet composed of different vehicle types with different specifications. The project will include a mathematical formulation of the school transport problem, the construction of algorithms to solve the problem, and, by testing and evaluating, to conclude on the possibilities of using mathematical optimization techniques in this type of commercial planning systems. The solution that will be the output from the programs has to be compared, by the practitioners, to the manual plans to conclude on whether the models should be built into a program. The project will also include a study on how different key figures, such as number of vehicles, distance driven and mean and maximum waiting times, change when modifying the parameters and the rules in the program.

1.3 Method

Both authors are studying the program of Engineering Physics with specialization in applied mathematics and optimization in particular. Their main previous knowledge lies in optimization theory and algorithms and not in school bus routing in the real world. This has the implication that some parts of the real planning problem may be foreseen but also that the evaluations will not be based on tradition.

The program `Skolskjuts` has been tested on data from the municipalities of Ljungby and Växjö. This created an understanding of the problem and the present planning tools. A literature study has been performed to see what has been done earlier in solving the actual problem and similar ones. The implementation and testing of our algorithms has been made in `AMPL` (A Mathematical Programming Language) [4], `C++` and `Matlab` [24]. Coding the programs has been the major time consuming part of the work. `AMPL` is a tool with syntax close to `C` and is streamlined for modeling of mathematical optimization problems. The big advantage of this program is that it is directly connected to a couple of solvers. One of these are `CPlex` [21] which is used to solve some parts of the mathematical optimization problems the school transport problem is decomposed into. The computers used when testing our programs are Sun UltraSPARC-III+ (900 MHz, 1Gb RAM).

An application written in `C#` has been used to build the distance and time matrices which are used as inputs to our programs. The application uses the geographic database TeleAtlas and the program `RW NetServer` to calculate the fastest stretches between every bus stop in the problem and the corresponding distances. The reason why time is used instead of distance in the calculation (even though the cost of a route could be more correlated with the distance) is that the fastest stretch is typically preferred against the shortest in a practical driving situation.

1.4 Restrictions

The project is about creating models and algorithms that could be built into a product, it is not about making it ready to use. With this in mind, it is not surprising that an expensive program specialized for optimization programming, `AMPL` with `CPlex`, has been used. During the project an academic license has been used. If deciding to build this into an actual product, it may be better to use another solver that is cheaper and more compatible with PC's and standard formats.

When developing the mathematical model an assumption was made that the structure of the school transport problem is similar to that of the case for Ljungby and Växjö, that is such as the rural areas of the Swedish municipalities. In larger cities the pupils most often travel to school with public transports and there is no specific school transport planning problem. Generally, it is cheaper to let the pupils travel with the public transports where available and therefore these pupils have not been included in the model. Assumptions are also made on the availability and capacity of different types of vehicles and their parameters such as driving costs per kilometer and fixed costs for using a vehicle of a certain type.

We have concentrated on the problem of transporting the pupils from home to the schools in the morning; the problem in the afternoon is analogous, although opposite directed, why this assumption is no limitation.

An assumption that every pupil only has one address is made, the case that the pupil lives alternately at his mother and father at two different addresses is being neglected. It is unreasonable to assign the pupil to two different routes every day since this may lead to a very expensive solution with superfluous capacity. The case that the pupil has a scheme where he or she is staying every week is easily taken care of by just solving different problems for each week.

More restrictions and assumptions will be presented where appropriate later in the thesis.

2 Skolskjuts – the original program

`ArcView 3` is a GIS software, a program that makes it possible to integrate data from different parts of an organization and let the user work with the data on the basis of geographical aspects.

`ArcView` is manufactured by ESRI [19] which also manufactures the software `Network Analyst`, an extension to `ArcView` that is used to conduct different routing analysis on the map, such as finding shortest paths and performing drive-time analysis.

Pupil data systems are programs that keep track of the pupils in a municipality, where they live, how old they are, which grade they are in and other such information.

Tekis has developed and licenses `Skolskjuts` [15]. This is an extension to `ArcView 3` and `Network Analyst`. `Skolskjuts` can be integrated with the standard pupil data systems on the market and depending on the type of pupil data there are functions to locate the children geographically on the map in `ArcView`. The program analyzes which pupils are entitled to transports to the schools based on the geographical positions of their residences, their age and the regulations in the municipality. The system is flexible for exceptions for certain pupils. In a manual way it is then possible to construct routes and bus stops. The construction of routes can be made in several ways, for example by marking a set of pupils or bus stops and then letting the program calculate the fastest path visiting all these. Specifications of the routes constructed, such as which weekdays and times the routes should be driven and the maximum numbers of passengers can be registered and then the routes are stored. It is always possible to change the routes manually at any time. When the user is satisfied a function in the program assigns the pupils to the set of routes stored. If there are many routes fulfilling the constraints given by the rules and the route specifications, the best route in terms of minimum waiting time is chosen.

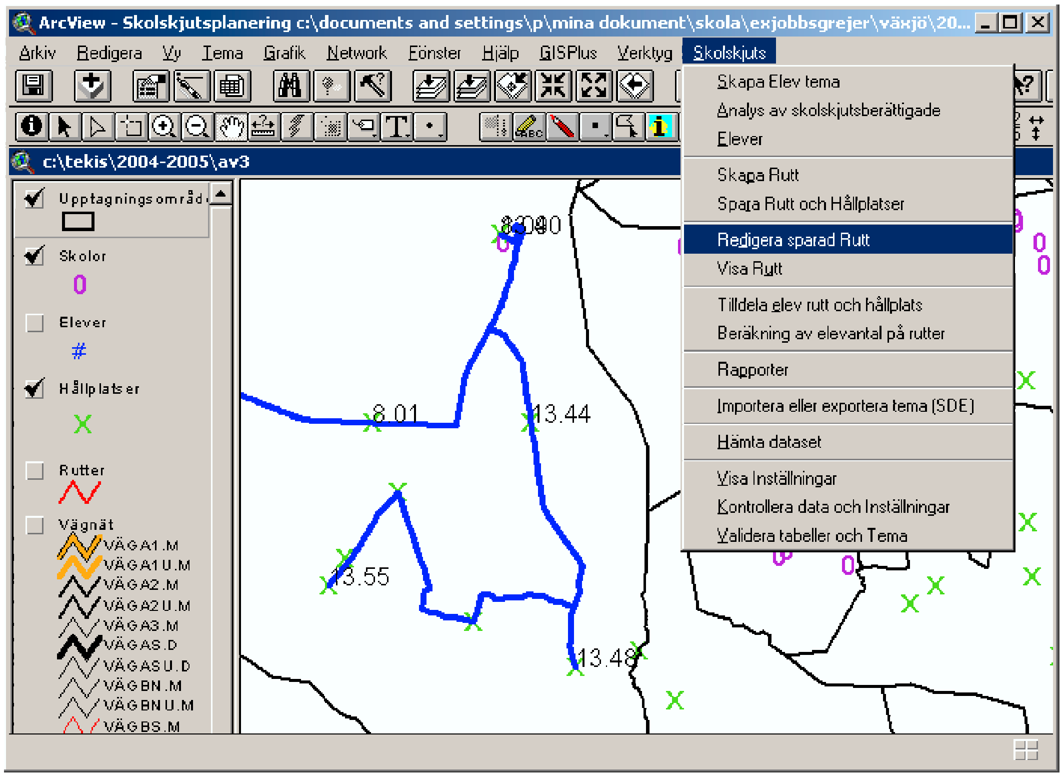


Figure 1: A screenshot from `Skolskjuts`.

A big advantage of `Skolskjuts` is the easiness to create all kinds of reports of the plan made. The general view of the planning is also very good with the graphic interface and the flexibility to change routes at any time. The problem with the program is the lack of intelligence built in; every single route constructed is in some sense optimal. The fastest route visiting certain stops is calculated, but the whole set of routes is not optimal together, where the cheapest set of routes serving all pupils is wanted. Even if this is not the objective of the program, it should be pointed out since it is definitely the objective of the planning.

3 The problem

The problem to solve is a minimization of costs while certain conditions are fulfilled. The conditions can be grouped into constraints regarding the vehicles (the capacity of the fleet and individual load capacities of the vehicles) and the bus stops (which have to be visited in certain time intervals, the time windows). A planning area typically consists of the pickup area of one school for older children attending the senior level of the nine-years compulsory school (a Swedish “högstadieskola”, senior school from now on), which can usually be divided into a few pickup areas for schools for younger children, (“låg-” and “mellanstadieskola”, junior and intermediate school). The pickup areas for the different senior schools in the municipalities are supposed to be distinct and not intersect, thus a plan for the whole municipality can be made by planning each senior school area as a separate problem. A problem here is that there may be a limited vehicle fleet available for the whole municipality. To get a problem reasonable in size the assumption that the senior school areas can be solved one by one still has been made and if there is a limited vehicle fleet, it is assumed to be fixed for each senior school area. In practice, if there is a limited fleet for the whole municipality, trial and error can probably be used to assign the vehicles to the different areas.

The costs associated with the transports are supposed to be a combination of fixed costs for using a certain vehicle and variable costs depending on the distances driven by the vehicles. The waiting times for the pupils are considered in the constraints that define a feasible solution. Note that with this formulation the size of the *waiting time*, which from now on will be defined as the time difference between pickup and start of the school-day, is not a good measurement when valuating a solution. In fact, a good solution to an optimization problem typically uses the size of the time windows well, leading to worse waiting times for the pupils but still a feasible solution. An interesting point that should be emphasized is that using a model like this is not all about economy. Minimization of transport costs have a clear correlation with minimization of influence on the environment which makes the project even more exciting.

The pupils that are entitled to transports are input to the model. `Skolskjuts` has a routine for choosing these from a whole set of pupils (i.e. all pupils in the area) given regulations about minimum distances between home and school. The choosing process could be incorporated into our model by connecting it to a database and perform a number of shortest path calculations.

The vehicles are supposed to start in the morning from certain depots or parking places to which they later return. It is also assumed that the vehicle fleet is limited.

Rules for the time windows has been chosen rather arbitrarily by own rules in the test runs. In section 8.6 we will see how changing these rules will influence the results.

4 Route optimization problems and previous work

One of the classical optimization problems is the *Traveling Salesman Problem* (TSP) which can be described as the problem for a salesman to find the shortest or cheapest way to visit all cities in his district. A generalization to TSP is the m-TSP where there are m salesmen to cover the cities and the objective is to minimize the sum of the distances of the routes. The *Vehicle Routing Problem* (VRP) is a natural extension to the m-TSP where a demand is associated with each city and each salesman/vehicle has a certain capacity, not necessarily identical. The salesmen/vehicles transport goods to the customers from a central depot. If constraints are added on the allowed points in time for the visits of each city, the *time windows*, a problem called the *Vehicle Routing Problem with Time Windows* (VRPTW) appear. In this problem the vehicle visiting a certain city or customer must arrive within a given time interval. It is allowed to arrive before the time window opens but the vehicle then has to wait there until the opening of the window. The VRPTW is the basic model for a large number of practical problems, for example a warehouse sending deliveries to its retailers or distribution of medicine from a central pharmacy to hospitals. The arrivals at each customer must be when the goods departments are held open. An extension to the VRPTW is the m-VRPTW where there are many depots. Here every customer can be served by any of the m depots.

The school transport problem are more complex than the m-VRPTW. There are many schools in the same district which all can be seen, with the same denotion, as a depot. But contrary to the m-VRPTW the customers, which can be seen as bus stops where the demands are the number of pupils, cannot be served from any of the depots, the pupils must arrive at a specific school. A pupil going to a certain school could however be allowed to use the same vehicle as a pupil going to another school (think of a pair of siblings living far off) and then this vehicle has to visit both schools or one of the pupils has to make a bus change. Another possible extension is letting the vehicles be parked in certain depots, geographically separated, which they start from in the morning and return to after the service is done.

4.1 School bus routing and scheduling

The previous work made on school bus routing and scheduling are very different in terms of objectives, constraints and assumptions, and thus also in terms of solution techniques.

Spada et al. [11] use an untraditional objective function, where the level of service provided by the bus operator is being explicitly optimized. Either the mean or maximum time loss over all children is minimized. Normally, and also in this project, the service is captured in the constraints and the objective is focused on operating costs. Spada et al. use a simple heuristic method to get a feasible solution and then use simulated annealing and tabu search to improve on this solution.

Both Spasovic et al. [12] and Braca et al. [2] have developed models for school bus routing and scheduling with focus on urban areas (New Jersey and Manhattan) and they use different kinds of heuristics to solve the problems. They both assume identical vehicles, all big buses, and they use, besides the obvious constraints for time windows, constraints regarding equity among the pupils, the former with maximum time on bus and the latter with maximum traveling distance. This is clearly not suitable for planning of rural areas, since the vehicle fleet consists of different vehicle types and the distances between school and home varies substantially between the pupils why equity probably cannot be that prioritized.

Corberán et al. [3] address the problem of routing school buses in a rural area. They use both the objective of minimizing the costs and and that of maximizing the service level. By a combination of heuristics they produce a set of solutions to choose among. The practitioner can in this way choose a solution that he or she thinks is the best compromise between cost and quality of service.

4.2 Different approaches to solve the Vehicle Routing Problem with Time Windows

The school transportation problem is similar to the traditional VRPTW if the objective is to minimize the cost. Different approaches for solving such problems are present in the literature.

A common method for solving the VRPTW is to use a column generation approach. When talking about route optimization problems the columns represents routes with an associated vehicle. In this case a column is a vector containing ones and zeros, representing if a certain stop is visited by the route given by the column or not. Information of the actual order of the bus stops in the route is not contained and has to be stored somewhere else. In the literature there are some variants of the column generation methods used. Very briefly explained, column generation chooses an optimal set of columns from a subset of all feasible columns. The subset is then extended with more columns and the optimal set is chosen once again. A thorough explanation of the theory behind the column generation method is presented in section 5.1.3.

Both Larsen [6] and Liu et al. [7] use column generation methods for solving VRPTW. They both focus on the operating cost in the objective function and have the service quality in the constraints defining feasible solutions. Thus the setup is very similar to ours. Both authors reach the same mathematical formulations of the problem with the same master- and subproblems. One of the differences between them is the way they solve the subproblems, that is when they are generating new columns. The latter demands that a new route is a real path, that is, no node is visited more than once, whereas the former do not have this requirement. Clearly, the method used by the former is faster because the feasibility check can be discarded but the quality of the generated routes is probably worsened by the subcycles.

Kallehauge et al. [5] solve the problem with the same mathematical formulation as Larsen and Liu et al. They consider the Lagrangian relaxation of the VRPTW where the constraints saying that each customer must be visited exactly once are relaxed. Due to an assumption of identical vehicles the Lagrangian problem can be split up into $|V|$ identical and simpler problems, where V is the set of vehicles. A cutting-plane algorithm combined with a Dantzig-Wolfe algorithm has then been used to solve the problem.

Another approach widely used for solving routing problem is the use of heuristics. Many of these are some sort of genetic algorithms which are characterized by the use of randomness to modify a given solution over and over again. Bent and Van Hentenryck [1] uses a hybrid algorithm with simulated annealing in combination with a large neighborhood search to solve a VRPTW. In this problem they first minimize the number of vehicle needed, all identical, and then minimize the travel cost given this fleet of vehicles. Thanghiah [13] uses a method based on something he calls genetic sectoring which by genetic algorithms adaptively searches for sector rays that partitions the customers into sectors served by each vehicle.

The methods used differ strongly from each other and many of the authors conclude that their methods are the best ones for traditional VRPTW's which they motivate with results on a standard set of test problems, the Solomon Benchmark problems [10].

5 Theory

5.1 Mathematical background

Problems with decision issues of yes/no type such as route optimization problems are often mathematically modeled as integer programs with binary variables which can be either 1 or 0. Most practical problems give rise to models whose sizes make them very hard to solve. They often belong to the set of so called $\mathcal{NP}$–hard problems. There are no known algorithms for such problems which can solve them to optimality within a polynomial time, that is, the time needed for the optimal solution to be found grows faster than any power of the size of the problem. The beautiful theory of their corresponding linear formulations is not applicable and the methods for solving these cannot be used directly. One often has to settle with a near-optimal solution and to be able to find a good one in reasonable time demands the use of some suitable heuristics.

In this section it is assumed that there is a minimization problem to solve. The results are in principle the same for maximization problems, with the inequalities turned the other way since such can be transformed to minimization problems by multiplying with -1.

5.1.1 Linear Programming

The methods used for solving integer problems most often take advantage of linear programming theory. Thus will the most important theory be explained in brief.

A *linear program*, LP, can be written, on the so called *canonical form*, as

$$\begin{aligned} \text{minimize} \quad & z = c^T x, && (1) \\ \text{subject to} \quad & Ax \geq b, && (2) \\ & x \geq 0, && (3) \end{aligned}$$

where $c, x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$. LP's are convex problems, meaning that the objective function and the feasible regions both are convex. Convex problems have a certain property which is utilized in many algorithms; any local minimizer is also a global minimizer.

By adding *slack variables*, $s \in \mathbb{R}^m$, $s \geq 0$ (with zero cost), the constraint (2) can be rewritten as

$$Ax \geq b \Leftrightarrow Ax - s = b,$$

and by renaming according to

$$\begin{aligned} \begin{pmatrix} A & -I \end{pmatrix} &\rightarrow A \\ \begin{pmatrix} c \\ 0 \end{pmatrix} &\rightarrow c \\ \begin{pmatrix} x \\ s \end{pmatrix} &\rightarrow x \end{aligned}$$

the problem can be written on the so called *standard form*:

$$\begin{aligned} \text{minimize} \quad & z = c^T x, && (4) \\ \text{subject to} \quad & Ax = b, && (5) \\ & x \geq 0. && (6) \end{aligned}$$

The reverse transformation is made by noting that

$$Ax = b \Leftrightarrow \begin{pmatrix} Ax \\ -Ax \end{pmatrix} \geq \begin{pmatrix} b \\ -b \end{pmatrix}.$$

A standard technique to find the optimal solution to an LP is the so called *simplex method* [8]. From now on it is assumed that the problem is given on standard form as in (4)–(6), that is, any slack variables needed are already included in x and if so, the matrix A includes entries -1 in the appropriate places. Further A is assumed to have linearly independent rows, which is not a limitation since else the problem either has no feasible solution or has redundant constraints, all depending on b.

In brief, the major idea is to start at a *basic feasible solution*, check if it is optimal and if not, move to a neighboring basic feasible solution. If this new solution is optimal the algorithm stops, else the next neighbor is explored and so on. A basic feasible solution is a solution to the constraint

$$Ax = b,$$

where in addition all components of x satisfy $x_i \geq 0$ and the columns of the constraint matrix corresponding to nonzero variables are linearly independent. Thus at most m variables can be nonzero. A basic feasible solution is said to be *degenerate* if any of the basis variables equals 0. It can be shown, see for example [8], that if an optimal solution to an LP exists, there exists an optimal basic feasible solution.

Let x be the current basic feasible solution and order the variables according to

$$x = \begin{pmatrix} x_B \\ x_N \end{pmatrix},$$

where x_B is the vector of basis variables and x_N is the vector of nonbasic variables (i.e. currently having value 0). The objective function can be written as

$$z = c_B^T x_B + c_N^T x_N.$$

Similarly the constraints can be written as

$$Bx_B + Nx_N = b.$$

Since the basis matrix B has full rank the constraints can be rewritten as

$$x_B = B^{-1}b - B^{-1}Nx_N,$$

and substituting for x_B in the objective function gives us

$$z = c_B^T B^{-1} b + (c_N^T - c_B^T B^{-1} N) x_N.$$

With $y = (c_B^T B^{-1})^T$, the so called *simplex multipliers*, we have

$$z = y^T b + (c_N^T - y^T N) x_N.$$

With $\hat{c}_N^T = c_N^T - y^T N$, its j^{th} entry $\hat{c}_j$ is called the *reduced cost* of variable x_j since an increase ϵ of x_j will change the objective value by $\epsilon \hat{c}_j$. If $\hat{c}_j < 0$ the objective can be improved by increasing x_j from zero. Thus, at an optimal solution $\hat{c}_j \geq 0$ for all j.

Suppose $\hat{c}_j^* < 0$ for some j^*. Then (fixing $x_N = 0$ except for entry j^*)

$$x_B = B^{-1} b - B^{-1} N x_N = B^{-1} b - B^{-1} A_{j^*} x_{j^*},$$

where A_{j^*} is the j^*th column of A. The variable x_{j^*} can be increased until any entry of x_B reaches zero, that is, until it reaches the value

$$x_{j^*} = \min_{1 \leq i \leq m} \left\{ \frac{(B^{-1} b)_i}{(B^{-1} A_{j^*})_i} : (B^{-1} A_{j^*})_i > 0 \right\}. \tag{7}$$

The variable x_{j^*} enters the basis at the expense of variable x_{i^*} where i^* corresponds to any minimum in (7).

When no variables with negative reduced cost exist, the current solution is a global optimal solution to the LP problem, since any local optimum is also a global optimum due to the convexity property.

To every LP problem there is a so called *dual* problem with interesting properties. To an LP in *canonical* form, as in (1)–(3), the corresponding dual problem is

$$\begin{array}{ll} \text{maximize} & w = b^T y, \\ \text{subject to} & A^T y \leq c, \\ & y \geq 0, \end{array}$$

which accordingly has m variables and n constraints. The dual variables are sometimes called "shadow prices" since they measure the implicit "costs" of the constraints.

There are two important statements relating the primal and the dual problem in LP theory. The first, *weak duality*, states that the objective value of the primal problem is an upper bound of the objective value of the dual problem and vice versa. This result applies also in various other types of problems. The second, *strong duality*, applies to all LP problems but may not do so to other problems. It states that if one of the problems has an optimal solution then so does the other and the optimal objective values are equal.

Not only the optimal solution to the primal LP, z^*, is found when using the simplex method but also the optimal solution to the dual problem. Actually, they are the same as the simplex multipliers in the last iteration of the algorithm, $y^* = (c_B^T B^{-1})^T$. $y^* \geq 0$ must hold since if $y_j < 0$ for any j the corresponding slack variable s_j has negative reduced cost $\hat{c}_j = 0 - y^T(-e_j) = y_j < 0$ (e_j is the vector with all zeros except for entry j with value 1) contradicting the assumption that the current solution was optimal. It is feasible in the dual problem since

$$\left(A^T y^*\right)^T = y^{*T} A = c_B^T B^{-1} \begin{pmatrix} B & N \end{pmatrix} = \begin{pmatrix} c_B^T & c_B^T B^{-1} N \end{pmatrix} \leq \begin{pmatrix} c_B^T & c_N^T \end{pmatrix} = c^T,$$

where the inequality stems from the fact that the reduced costs are nonnegative in the optimal basic feasible solution. It is optimal since weak duality implies that z^* is an upper bound of w^* and we have

$$b^T y^* = y^{*T} b = c_B^T B^{-1} b = c_B^T x_B^* = c^T x^* = z^*.$$

Thus $w^* = z^*$.

5.1.2 Integer programming

Integer programs, IP's, look very much like linear programs, the "only" difference is that some variables are required to be integer valued. It is not surprising that LP theory is important also when solving these integer counterparts. However, the naive thought that a good solution to an IP is given by rounding the solution to the corresponding LP is often false, which many easy-to-construct problems show. Most often, the solution to the corresponding LP (the IP with the integer constraints relaxed) is used to find bounds to the IP.

A commonly used concept when talking about IP's is the *integrality gap*. This is the gap between the solution value of the IP, z^*_{IP}, and the solution value of the corresponding LP, z^*_{LP}. The following statement can trivially be found, where z_{IP} is any feasible integer solution:

$$z^*_{LP} \leq z^*_{IP} \leq z_{IP}.$$

Even if the value of z^*_{IP} can be hard to find, one still have a clue about how good a feasible solution z_{IP} is if the size of the integrality gap somehow can be estimated and by looking at the optimal value to the linear relaxed problem, z^*_{LP}, which often is relatively easy to find. For some IP problems, finding feasible solutions is easy, and the question is how to find *good* feasible solutions. For example in the TSP problem, if the salesman is allowed to travel between any pair of cities, any permutation of the cities is a feasible solution. For other IP problems it is very hard to find a feasible solution, sometimes as hard as to find an optimal feasible solution.

Another way to use linear relaxations is to prove optimality or infeasibility. If the linearly relaxed problem is infeasible this implies that the original IP is infeasible as well. And if an optimal solution x^* to the relaxed problem is integer it is also optimal in the IP.

5.1.3 Column generation

Column generation is a method used when solving an LP where the number of columns in the constraint matrix is very large, perhaps too large to store, or when there are problems in finding all columns in an easy way. Both these factors are present in some formulations of vehicle routing problems to be seen later on in the thesis. It is often used for solving IP's as well, hoping that columns that are good in the linearly relaxed version of the problem are good also in the original one. The column generation approach utilizes the fact that in the simplex algorithm the matrix A is used only to check whether there are any columns for which $\hat{c}_j = c_j - y^T A_j < 0$, and if so let that column enter the basis matrix. When A has some known specific structure it is possible to find columns with negative reduced cost without the explicit knowledge of all columns. Only the columns needed are generated and most columns of A remain unknown.

The problem not including the whole matrix A is called the *restricted master problem*, RMP, and the columns found so far are collected in the matrix A_R. At an optimal solution to the restricted problem, the simplex multipliers coincide with the optimal solution to the corresponding dual problem so there is a strong connection between column generation and the dual problem. The dual variables are then used to find whether the current solution to the RMP is optimal also in the complete master problem.

The standard example used to explain the column generation technique is the *one-dimensional cutting stock problem*, see [8], which share some features with the vehicle routing problems. This problem arises naturally in some manufacturing facilities producing sheets of some material (e.g. wood, steel or paper) with certain widths and lengths. The customers however demand sheets of the same width but different and smaller lengths, leading to the sheets being cut into smaller pieces. A 5 m wooden board could for example be cut in two pieces of 2 m each, one piece of 75 cm and the remaining 25 cm of waste. In this process, the objective is to minimize the total number of sheets cut in supplying the demand of every customer.

Each alternative of cutting one piece is called a pattern and if there are many different lengths demanded, the number of patterns is huge. Denote the standard length of each sheet by L. Suppose m different lengths $l_1, l_2, \ldots, l_m$ are demanded, b_i pieces of each. A pattern could then be described by a vector $a = (a_1, a_2, \dots, a_m),\ a_i \in \mathbb{Z}_+$ and it is feasible if and only if

$$a_1 l_1 + a_2 l_2 + \dots + a_m l_m \le L.$$

Let x_i be the number of sheets to be cut of pattern i, where $i = 1 \dots n$, and n denotes the total number of feasible patterns. All patterns are conceptually collected in the huge matrix A where the columns are the pattern vectors. The minimization problem becomes

$$\begin{array}{ll} \text{minimize} & z = \sum_{i=1}^{n} x_i, \\ \text{subject to} & Ax \ge b, \\ & x \in \mathbb{Z}_+. \end{array}$$

Note that A in addition to all feasible patterns have to include the slack constraints. The slack variables can be assumed to have zero cost, it is free to throw away unneeded pieces.

The x_i should be integer but in this specific problem the numbers are quite large, in real life, making it possible to round the possibly fractional values in an appropriate way without getting the gap between the solution found and the optimal one too large.

The starting basis matrix can be the most simple one:

$$B = \begin{pmatrix} \left\lfloor \frac{L}{a_1} \right\rfloor & 0 & \cdots & 0 \\ 0 & \left\lfloor \frac{L}{a_2} \right\rfloor & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \left\lfloor \frac{L}{a_m} \right\rfloor \end{pmatrix}.$$

Suppose that we have a basic feasible solution with basis matrix B at some iteration. At first the vector of simplex multipliers $y^T = c_B^T B^{-1}$ is calculated. To see whether this is an optimal solution or not one has to check if there is some j for which

$$\hat{c}_j = 1 - y^T A_j < 0.$$

There might be a column already present in A_R for which this happens and if so we let the corresponding variable enter the basis. There might also be a slack variable with negative reduced cost, which is the case when any $y_j < 0$. For slack variable s_j we then have $\hat{c}_j = 0 - y^T(-e_j) = y_j < 0$, and this variable is added to the basis.

If there is no column with negative reduced cost neither in A_R nor among the slack variables we have to search in the whole of A. Here is where the strength in column generation lies. The column with the most negative reduced cost can be found without explicitly calculating the reduced costs for every column but by solving another optimization problem, the sub- or pricing problem. In the cutting stock problem, the subproblem becomes the well known *knapsack problem*. We are supposed to find the column, or pattern, with the least value of $1 - y^T A_j$, that is the maximum value of $y^T A_j$. The optimization problem thus reads

$$\begin{array}{ll} \text{maximize} & y_1 a_1 + y_2 a_2 + \dots + y_m a_m, \qquad (8) \\ \text{subject to} & l_1 a_1 + l_2 a_2 + \dots + l_m a_m \le L, \qquad (9) \\ & a_i \in \mathbb{Z}_+. \qquad (10) \end{array}$$

This knapsack problem, though $\mathcal{NP}$-hard, can be solved by special algorithms in a quite efficient way and if there is a column with negative reduced cost it can thus be found without the explicit knowledge of each column in A. If the optimal solution to (8)–(10) satisfies $y^T A_j > 1$, then the corresponding variable enters the basis, else the current basic feasible solution is optimal.

The column generation method gives the optimal solution to the LP-relaxed version of the original problem, at least if the problem is non-degenerate. In the cutting stock problem, as said before, this solution can probably be rounded to a good solution to the IP as well. In other problems the columns generated for the LP-relaxed problem might not even give a near-optimal solution to the original IP.

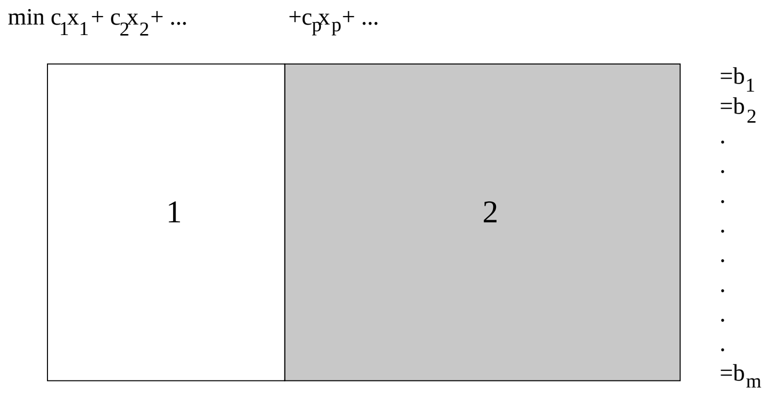


Figure 2: An illustration of the column generation approach.

Figure 2 shows a schematic view of the column generation process. The columns in sector 1 are known whereas the columns in sector 2 are represented by the subproblem. The two sectors together represent the complete master problem. When generating a new column it is moved from sector 2 to sector 1. Normally the sector represented by the pricing algorithm is much bigger than the sector of known columns.

5.2 Algorithms

5.2.1 Branch-and-bound

One of the most common ways to solve IP's is by using a technique called *branch-and-bound*. This method implicitly enumerates all possible solutions and by branching, which corresponds to adding constraints on the variables, and by bounding, provided by linear relaxations, explores the solution tree in a clever way. A simple example taken from [14] will be used to explain the technique. The problem to solve is

$$\begin{aligned} \text{maximize} \quad & 4x_1 - x_2, && (11)\\ \text{subject to} \quad & 7x_1 - 2x_2 \le 14, && (12)\\ & x_2 \le 3, && (13)\\ & 2x_1 - 2x_2 \le 3, && (14)\\ & x \in \mathbb{Z}^2_+. && (15) \end{aligned}$$

To obtain an upper bound to the solution value slack variables s_1, s_2, s_3 are added and the linear relaxation of the problem is solved, for example with the simplex method. The resulting optimal basis representation is

$$\begin{array}{rcllllllcl} \overline{z} & = & \max \frac{59}{7} & & & -\frac{4}{7}\,s_1 & -\frac{1}{7}\,s_2, & & & \\ & & \text{s. t.} & x_1 & & +\frac{1}{7}\,s_1 & +\frac{2}{7}\,s_2 & & = & \frac{20}{7}, \\ & & & & x_2 & & +s_2 & & = & 3, \\ & & & & & -\frac{2}{7}\,s_1 & +\frac{10}{7}\,s_2 & +s_3 & = & \frac{23}{7}, \\ & & & x_1, & x_2, & s_1, & s_2, & s_3 & \ge & 0. \end{array}$$

Thus an upper bound $\overline{z} = \frac{59}{7}$ to problem (11)–(15) with the non-integral solution $(\overline{x}_1, \overline{x}_2) = \left(\frac{20}{7}, 3\right)$ is found. Since no feasible solution is at hand, the lower bound is set $\underline{z} = -\infty$.

The next step in the procedure is to branch the solution space. This is done by limiting some of the integer variables that are fractional in the LP solution. If $\mathcal{S}$ is the feasible region for the LP problem and $x_j = \overline{x}_j \notin \mathbb{Z}_+$ a branching is

$$\begin{aligned} \mathcal{S}_1 &= \mathcal{S} \cap \{x : x_j \leq \lfloor \overline{x}_j \rfloor\}, \\ \mathcal{S}_2 &= \mathcal{S} \cap \{x : x_j \geq \lceil \overline{x}_j \rceil\}. \end{aligned}$$

The optimal solution of LP($\mathcal{S}$) is not feasible in either LP($\mathcal{S}_1$) or LP($\mathcal{S}_2$) and no integer points, possible solutions to the problem, have been cut off. With this problem the only possible branching is $\mathcal{S}_1 = \mathcal{S} \cap \{x : x_1 \leq 2\}$ and $\mathcal{S}_2 = \mathcal{S} \cap \{x : x_1 \geq 3\}$. The process can be represented by the tree in Figure 3.

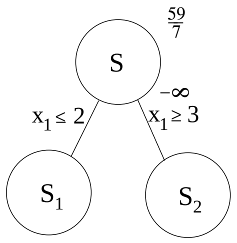


Figure 3: The branch-and-bound tree after one branching.

Either of the nodes $\mathcal{S}_1$ or $\mathcal{S}_2$ is chosen and LP($\mathcal{S}_i$) is solved in the same way as LP($\mathcal{S}$) was. Starting with LP($\mathcal{S}_1$) leads to $\overline{z}_1 = \frac{15}{2}$ with $\left(\overline{x}_1^1, \overline{x}_2^1\right) = \left(2, \frac{1}{2}\right)$ and this is a new upper bound in that part of the tree. Another branching is done and the new nodes $\mathcal{S}_{11} = \mathcal{S}_1 \cap \{x : x_2 \leq 0\}$ and $\mathcal{S}_{12} = \mathcal{S}_1 \cap \{x : x_2 \geq 1\}$ are created.

Trying to solve LP($\mathcal{S}_2$) shows that this node is infeasible, $\overline{z}_2 = -\infty$, and hence node $\mathcal{S}_2$ can be pruned by infeasibility. Left to examine are now node $\mathcal{S}_{11}$ and $\mathcal{S}_{12}$. Solving LP($\mathcal{S}_{12}$) leads to $\overline{x}^{12} = (2, 1)$, an integer solution, with value 7. Since the linearly relaxed problem lead to an integer solution, no better feasible solution further down that branch can be found, and $\mathcal{S}_{12}$ is pruned by optimality. The global lower bound previously used, $\underline{z} = -\infty$, can now be updated to $\underline{z} = 7$.

Examining the last node, solving LP($\mathcal{S}_{11}$), gives a fractional solution, $\overline{x}^{11} = \left(\frac{3}{2}, 0\right)$, with value 6, which is the upper bound further down that branch. Since $\underline{z} = 7 > \overline{z}_{11} = 6$, a conclusion can be drawn that no better solution than already found is present in the branch and thus it is pruned by bound. The whole tree is implicitly examined and the global optimal value is 7.

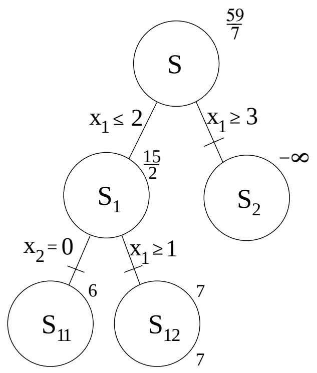


Figure 4: Complete branch-and-bound tree.

The above scheme is implemented in solvers like `CPlex`. By starting from the basis used in the parent node the simplex method, or actually a variant called *dual simplex*, typically does not need more than a few iterations to find the optimal solution with the new constraint from the branching added. The number of possible optimal points grows exponentially with the number of variables, for every binary variable the size of the solution space is multiplied by two. In worst case, the solver needs exponential time to find the solution.

5.2.2 Heuristic algorithms

Many practical problems belong to the class of $\mathcal{NP}$–hard problems and they cannot be solved effectively within polynomial time. For such problems it is common to use some sort of heuristic approach to find a "good" feasible solution in reasonable time. The different variants of heuristics are many; the main features of the ones most applicable to our problem will briefly be explain.

Common heuristics are the *greedy* algorithms. These are algorithms that are naive in the sense that they lack the overview of the problem and in every step chooses what seems to be the best choice at that particular point. The main advantage of the greedy algorithms is that they work fast and the drawback is that they sometimes end up in situations where the "smart" choices in the beginning lead to that only "bad", perhaps even infeasible, alternatives are left in the end. A combinatorial problem to choose the cheapest set of elements satisfying some constraints can be formulated as

$$\min_{S \subseteq N} \{c(S) : v(S) \geq k\}.$$

The steps in a greedy heuristic to solve the problem can be stated as in [14]:

1. Set $S^0 = \emptyset$ (start with the empty set, assumed infeasible). Set $t = 1$.
2. Set $j_t = \text{argmin}\frac{c(S^{t-1} \cup \{j_t\}) - c(S^{t-1})}{v(S^{t-1} \cup \{j_t\}) - v(S^{t-1})}$ (choose the element whose additional cost per unit of resource is minimum).
3. If the previous solution S^{t-1} is feasible, and the cost has not decreased, stop with $S^G = S^{t-1}$.
4. Otherwise set $S^t = S^{t-1} \cap \{j_t\}$. If the solution is now feasible, and the cost function is non-decreasing or $t = |\text{all elements}|$, stop with $S^G = S^t$.
5. Otherwise if $t = |\text{all elements}|$, no feasible solution has been found. Stop.
6. Otherwise set $t \leftarrow t + 1$ and return to 2.

A problem with greedy algorithms is the risk of getting stuck in local minima instead of the global ones searched for. An improved algorithm, a so called *meta-heuristic*, that avoids ending up in local minima is for example *simulated annealing*. When at a local minimum a natural idea is to move to a neighboring point even if its value is worse. In simulated annealing the basic idea is to choose a neighbor randomly. The neighbor then replaces the incumbent (i.e. the current solution) with probability 1 if it has better value and with a probability between 0 and 1 if its value is worse. The probability of replacing the incumbent with a worse neighbor is a function of the difference of the values as well as of the time consumed so far by the algorithm. Simulated annealing can be described as follows [14]:

```
1. Get an initial solution $S$.
2. Get an initial temperature $T$ and a reduction factor $r \in (0, 1)$.
3. While not frozen, do the following:
       - Perform the following loop $L$ times:
           - Pick a random neighbor $S'$ to $S$.
           - Let $\Delta = f(S') - f(S)$.
```

```
        - If $\Delta \leq 0$, set $S = S'$.
        - If $\Delta > 0$, set $S = S'$ with probability $e^{-\Delta/T}$.
    - Set $T \leftarrow rT$. (Reduce temperature.)
4. Return the best solution found.
```

Note that the larger Δ is and the lower the temperature is, the less chance of saving a worse solution. The methods of finding neighboring solutions are specific to the type of problem.

Other variants of meta-heuristics are *genetic algorithms* and *tabu search*.

6 Mathematical formulation

As previously seen, there has been substantial work done on route optimization problems. However, none of the works we have seen consider our specific problem. In this project a model that is as close to a practical model as possible is wanted whereas most of the models used in the scientific studies of the vehicle routing problems are a bit oversimplified. Our problem consists of many depots (which from now on represents the garages) and schools, and vehicles that are not identical. The quality of service has been completely included in the constraints (with a minor modification which will be presented later) and the objective is only focusing on costs. This is the traditional way to plan school transports and also what Tekis wanted us to do. We get a model with big similarities to the traditional VRPTW if just looking at one school at a time and the pupils who are going there. The methods used on those problems are often based on column generation [6]. More complex problems are most often solved by heuristic algorithms.

The big similarities to the traditional VRPTW lead us to use column generation which, since the problem is more complex, is combined with heuristic methods. Hopefully it is, due to the nature of the problem, advantageous to use both types of algorithms.

6.1 The complete model

In an ideal problem the pupils are allowed to change vehicle on the way to school and this can take place at any bus stop. One can allow a fine grid of bus stops whereon only a small subset of these someone is climbing aboard. The available vehicles are parked in a number of garages, which they first leave and then return to. To each pupil a set of allowed home bus stops can be connected, these are bus stops within a reasonable distance from the home. Clearly, the problem grows in this case too big for the computer to handle but for the sake of clarity a mathematical formulation of this ideal model has been made and it is presented below.

Variables:

$$
\begin{aligned}
w_{ijke} &= \begin{cases} 1, & \text{if pupil } e \text{ travels with vehicle } k \text{ from bus stop } i \text{ to bus stop } j \\ 0, & \text{otherwise} \end{cases} \\
v_{ek} &= \begin{cases} 1, & \text{if pupil } e \text{ travels with vehicle } k \text{ anytime} \\ 0, & \text{otherwise} \end{cases} \\
x_{ijk} &= \begin{cases} 1, & \text{if vehicle } k \text{ travels from bus stop } i \text{ to bus stop } j \\ 0, & \text{otherwise} \end{cases} \\
y_k &= \begin{cases} 1, & \text{if vehicle } k \text{ is used} \\ 0, & \text{otherwise} \end{cases} \\
s_{ik} &= \text{point in time when vehicle } k \text{ stops at bus stop } i
\end{aligned}
$$

Sets:

$$
\begin{aligned}
\mathcal{N} &= \text{the set of bus stops incl. garages} \\
\mathcal{G}^+ &= \text{the set of garages to leave from, } \mathcal{G}^+ \subset \mathcal{N} \\
\mathcal{G}^- &= \text{the set of garages to return to, } \mathcal{G}^- \subset \mathcal{N} \\
\mathcal{C} &= \text{the set of bus stops excl. garage, } \mathcal{C} \subset \mathcal{N} \\
\mathcal{S} &= \text{the set of schools, } \mathcal{S} \subset \mathcal{C} \\
\mathcal{K} &= \text{the set of vehicles} \\
\mathcal{E} &= \text{the set of pupils} \\
\mathcal{I}_e &= \text{the set of allowed home stops for pupil } e
\end{aligned}
$$

Parameters:

$$
\begin{aligned}
c_{ijk} &= \text{cost of driving with vehicle } k \text{ from bus stop } i \text{ to bus stop } j \\
g_k &= \text{fixed cost of using vehicle } k \\
t_{ijk} &= \text{time needed for traveling from bus stop } i \text{ to bus stop } j \text{ with vehicle } k \\
q_k &= \text{capacity of vehicle } k \\
gb_k &= \begin{cases} 1, & \text{if vehicle } k \text{ belongs to garage } g \\ 0, & \text{otherwise} \end{cases} \\
h_{ei} &= \begin{cases} 1, & \text{if pupil } e \text{ attends school } i \\ 0, & \text{otherwise} \end{cases} \\
a_e &= \text{earliest pickup time for pupil } e \\
b_e &= \text{latest drop time at school for pupil } e \\
p_e &= \text{maximum number of vehicle changes for pupil } e \\
M &= \text{big scalar}
\end{aligned}
$$

The objective function to minimize is

$$
z = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} \sum_{k \in \mathcal{K}} c_{ijk} x_{ijk} + \sum_{k \in \mathcal{K}} g_k y_k, \tag{16}
$$

and the constraints to be fulfilled are

$$\begin{aligned}
\sum_{e\in\mathcal{E}} w_{ijke} &\leq Mx_{ijk},\ \forall i,j \in \mathcal{C}, \forall k \in \mathcal{K}, && (17)\\
\sum_{j\in\mathcal{N}} x_{ijk} &= gb_k,\ \forall i \in \mathcal{G}^+, \forall k \in \mathcal{K}, && (18)\\
\sum_{i\in\mathcal{N}} x_{ijk} &= gb_k,\ \forall i \in \mathcal{G}^-, \forall k \in \mathcal{K}, && (19)\\
\sum_{i\in\mathcal{C}}\sum_{k\in\mathcal{K}} w_{ijke} &= h_{ej},\ \forall e \in \mathcal{E}, \forall j \in \mathcal{S}, && (20)\\
\sum_{i\in\mathcal{I}_e}\sum_{j\in\mathcal{C}}\sum_{k\in\mathcal{K}} w_{ijke} &\geq 1,\ \forall e \in \mathcal{E}, && (21)\\
\sum_{i\in\mathcal{N}} x_{ihk} - \sum_{j\in\mathcal{N}} x_{hjk} &= 0,\ \forall h \in \mathcal{C}, \forall k \in \mathcal{K}, && (22)\\
\sum_{i\in\mathcal{C}}\sum_{k\in\mathcal{K}} w_{ihke} - \sum_{j\in\mathcal{C}}\sum_{k\in\mathcal{K}} w_{hjke} &= 0,\ \forall h \in \mathcal{C}\setminus\mathcal{S}, \forall e \in \mathcal{E}, && (23)\\
\sum_{k\in\mathcal{K}} x_{iik} &= 0,\ \forall i \in \mathcal{N}, && (24)\\
\sum_{e\in\mathcal{E}} w_{ijke} &\leq q_k,\ \forall i,j \in \mathcal{C}, \forall k \in \mathcal{K}, && (25)\\
\sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} w_{ijke} &\leq Mv_{ek},\ \forall e \in \mathcal{E}, \forall k \in \mathcal{K}, && (26)\\
\sum_{i\in\mathcal{C}}\sum_{j\in\mathcal{C}} x_{ijk} &\leq My_k,\ \forall k \in \mathcal{K}, && (27)\\
\sum_{k\in\mathcal{K}} v_{ek} &\leq p_e + 1,\ \forall e \in \mathcal{E}, && (28)\\
s_{ik} + t_{ijk} - M(1 - x_{ijk}) &\leq s_{jk},\ \forall i,j \in \mathcal{N}, \forall k \in \mathcal{K}, && (29)\\
s_{hl} + M(1 - w_{ihke}) + M(1 - w_{hjle}) &\geq s_{hk},\ \forall i,j,h \in \mathcal{C}, \forall k,l \in \mathcal{K}, \forall e \in \mathcal{E}, && (30)\\
s_{ik} + M(1 - \sum_{j\in\mathcal{C}} w_{ijke}) &\geq a_e,\ \forall i \in \mathcal{I}_e, \forall k \in \mathcal{K}, \forall e \in \mathcal{E}, && (31)\\
s_{jk} - M(1 - h_{ej}\sum_{i\in\mathcal{C}} w_{ijke}) &\leq b_e,\ \forall j \in \mathcal{S}, \forall k \in \mathcal{K}, \forall e \in \mathcal{E}, && (32)\\
w_{ijke} &\in \{0,1\},\ \forall i,j \in \mathcal{C}, \forall k \in \mathcal{K}, \forall e \in \mathcal{E}, && (33)\\
v_{ek} &\in \{0,1\},\ \forall k \in \mathcal{K}, \forall e \in \mathcal{E}, && (34)\\
x_{ijk} &\in \{0,1\},\ \forall i,j \in \mathcal{N}, \forall k \in \mathcal{K}, && (35)\\
y_k &\in \{0,1\},\ \forall k \in \mathcal{K}. && (36)
\end{aligned}$$

Constraint (17) forces the vehicle variable to 1 if any pupil travels the stretch with that very vehicle. Constraints (18) and (19) tell that every vehicle leaves its garage and then returns. Note that if a vehicle is not used in the solution, this correspond to the vehicle going directly from its "out-garage" to its "in-garage" (i.e. it is parked all day if these are the same). Constraint (20) insures that all pupils come to the right schools and constraint (21) that they are picked up at some allowed stop. The constraints (22) and (23) balance the vehicles and the pupils at every bus stop. Constraint (24) is just a technical constraint saying that a vehicle cannot travel from a stop to the same stop. The capacity limitations of the vehicles are insured by constraint (25). Constraint (26) controls the variable saying if a pupil uses a certain vehicle and constraint (27) the variable saying if a certain vehicle is being used. The maximum number of bus changes for every pupil is controlled by constraint (28). Constraint (29) controls the time when a vehicle stops at a certain bus stop, (30) the time when the pupils are at the bus stops, important when changing vehicles, (31) and (32) defines the time windows, the allowed times for a vehicle to visit a stop. Constraints (33)–(36) guarantee that the variables attend binary values.

6.2 The set partitioning formulation

It is possible to formulate the routing problem as a so called *set partitioning* problem (SP), where most of the conditions are implicitly taken care of in the definition of a huge complex set $\mathcal{R}$ containing *all feasible routes*. In this formulation the pupils are supposed to have just one home stop which is a simplification compared to the previous model. The master problem becomes easy to understand despite that the constraints defining a feasible route can be very complex and hard to model mathematically in an efficient way. In the uttermost case there could be only a black box telling that a given route is feasible or not. From now on, by a feasible route is meant a route where the pupils are picked up at their home bus stop inside their time window and left at their school before start of school and that the number of pupils on a vehicle at a single time never exceeds the capacity of the vehicle. That is, bus changes will not be allowed. Note that the problem is not exactly a SP, since there are constraints regarding the number of vehicles used. In the future this formulation will still be called the SP-formulation to separate it from the *set covering* (SC) formulation where the equality in (38) is substituted for an inequality ($\geq$).

6.2.1 The master problem

The following notations, not to be mixed up with those in section 6.1, are used:

Variables:

$$x_r = \begin{cases} 1, & \text{if route } r \text{ is in the solution} \\ 0, & \text{otherwise} \end{cases}$$

Sets:

$$\begin{aligned} \mathcal{R} &= \text{the set of } \textit{all} \text{ feasible routes} \\ \mathcal{V} &= \text{the set of vehicle types} \\ \mathcal{B} &= \text{the set of bus stops with pupils} \end{aligned}$$

Parameters:

$$\begin{aligned} c_r &= \text{cost of route } r \\ \delta_{br} &= \begin{cases} 1, & \text{if bus stop } b \text{ is served by route } r \\ 0, & \text{otherwise} \end{cases} \\ f_v &= \text{number of available vehicles of type } v \\ u_{vr} &= \begin{cases} 1, & \text{if vehicle type } v \text{ is used in route } r \\ 0, & \text{otherwise} \end{cases} \end{aligned}$$

In this formulation, the objective function to be minimized reads

$$z = \sum_{r \in \mathcal{R}} c_r x_r, \tag{37}$$

and the constraints are

$$\sum_{r \in \mathcal{R}} \delta_{br} x_r = 1, \quad \forall b \in \mathcal{B}, \tag{38}$$

$$\sum_{r \in \mathcal{R}} u_{vr} x_r \leq f_v, \quad \forall v \in \mathcal{V}, \tag{39}$$

$$x_r \in \{0,1\}, \quad \forall r \in \mathcal{R}. \tag{40}$$

The objective is to minimize the sum of the costs of the chosen routes. The cost of a route, c_r, is defined as a fixed cost of the vehicle used plus the driving cost. The driving cost in turn is defined as a vehicle specific *vehiclefactor* times the stretch driven, where the stretches without pupils loaded are multiplied with a factor $weight \in [0, 1]$. It is assumed that the vehicle leaves a garage before visiting the first bus stop and that it also returns to this garage after visiting the last school in the route.

Also, it is assumed that the pupils are assigned to only one bus stop. Constraint (38) makes sure that all bus stops with pupils are being visited. Constraint (39) controls that all vehicles used are available. The constraint (40) says that all routes are either chosen or not. When a subset of $\mathcal{R}$, $\widetilde{\mathcal{R}} \subset \mathcal{R}$ is used, the corresponding problem is called, as introduced in the theory section, the *restricted master problem* (RMP).

A requirement in the problem formulation is that no bus stop can have more pupils than the capacity of the largest vehicle. Since there are binary variables and equality constraints in the RMP it is not possible to send two vehicles to the same bus stop, thus it would be impossible to find a feasible solution. If this is the case the bus stop has to be multiplied and the pupils put into groups. In addition, pupils going to different schools can use the same bus stop. Also here the pupils are divided into groups and the bus stop is multiplied.

Dividing the pupils on a bus stop with many pupils into groups could also be beneficial even if they fit in one bus, at the expense of running time of the algorithm. One could think of a bus stop passed by two vehicles with 10 seats each, half of them occupied. If there are 10 pupils at the bus stop 5 of them could go with one of the buses and 5 with the other which is not allowed in the SP formulation above. Ideally constraint (38) refers to the pupils rather than the bus stops. Referring to individual pupils makes it possible to modify the model so that a set of home stops for each pupil can be allowed as in the complete model. Also, if a column can correspond to more than one vehicle bus changes can be incorporated in this formulation.

6.2.2 The subproblem

Consider routes serving pupils assigned to one bus stop and going to one school only using a vehicle from a certain garage or depot, and that time windows and vehicle capacity are the only constraints that apply. For each school s, each depot d and each vehicle $v \in \mathcal{V}$ the following notations are used:

Variables:

$$x_{ij} = \begin{cases} 1, & \text{if arc } (i,j)\ i,j \in \mathcal{N}_s \text{ is used in the route} \\ 0, & \text{otherwise} \end{cases}$$

s_i = service time of node $i \in \mathcal{N}_s$

Sets:

$\mathcal{B}_s \subset \mathcal{B}$	=	the set of bus stops with pupils going to school s
$\mathcal{N}_s$	=	$\mathcal{B}_s \cup \{d\} \cup \{s\}$ the set of bus stops + depot d + school s

Parameters:

c_{ij}	=	cost of arc (i,j) $i,j \in \mathcal{N}_s$
t_{ij}	=	time needed for traveling the arc (i,j) $i,j \in \mathcal{N}_s$
f_v	=	number of available vehicles of type v
g_v	=	fixed cost of vehicle type v
q_v	=	capacity of vehicle type v
a_i, b_i	=	start and end time for time window at node $i \in \mathcal{N}_s$
w	=	*weight* parameter for arcs without pupils on bus
d_i	=	number of pupils on bus stop $i \in \mathcal{B}_s$
M	=	big scalar
π_i	=	dual variable corresponding to *bus stop visitation* constraint $i \in \mathcal{B}_s$ in (38)
μ_v	=	dual variable corresponding to *vehicle availability* constraint v in (39)

The objective function to be minimized is

$$y = (g_v - \mu_v) + w \sum_{j \in \mathcal{N}_s} c_{dj} x_{dj} + \sum_{i \in \mathcal{B}_s} \sum_{j \in \mathcal{N}_s} (c_{ij} - \pi_i) x_{ij} + w c_{sd}, \tag{41}$$

and the constraints are

$$\begin{aligned}
\sum_{i \in \mathcal{B}_s} d_i \sum_{j \in \mathcal{N}_s} x_{ij} &\leq q_v, && (42)\\
\sum_{j \in \mathcal{N}_s} x_{dj} &= 1, && (43)\\
\sum_{i \in \mathcal{N}_s} x_{ih} - \sum_{j \in \mathcal{N}_s} x_{hj} &= 0, \ \forall h \in \mathcal{B}_s, && (44)\\
\sum_{i \in \mathcal{N}_s} x_{is} &= 1, && (45)\\
s_i + t_{ij} - M(1 - x_{ij}) &\leq s_j, \ \forall i,j \in \mathcal{N}_s, && (46)\\
a_i \leq s_i &\leq b_i, \ \forall i \in \mathcal{N}_s, && (47)\\
x_{ij} &\in \{0,1\}, \ \forall i,j \in \mathcal{N}_s. && (48)
\end{aligned}$$

The terms in the objective function are in order of appearance: the fixed cost of the vehicle used minus the dual variable, the driving cost from the garage to the first stop, the driving cost from the first stop to the school minus the dual variables for the bus stops visited and the driving cost from the school back to the garage. The dual variable μ_v can be interpreted as the negative cost of using an additional vehicle of a certain type. The minimization formulation together with the $\leq$ condition leads to that $\mu_v \leq 0$. If there is slack in the constraint the dual variable will be 0, if not, that is all vehicles of that type are already in use, it will attend a negative value. A negative value increases the subproblem objective for that specific vehicle type and finding columns with negative reduced cost using that vehicle type becomes harder. The $=$ condition in the visitation constraint (38) results in that π_i can attend any real value. It can be interpreted as how much is marginally earned by visiting a certain bus stop some more. The linear relaxation of the variables makes the physical interpretation less straightforward.

In the literature this problem is known as an Elementary Shortest Path Problem with Time Windows and Capacity Constraints (ESPPTWCC). (42) is the capacity constraint, (43)–(45) are the balance constraints and (46) and (47) are the time constraints. There is no known efficient algorithm for solving this problem exactly [5]. If there are feasible solutions to this problem with costs less than zero the corresponding columns have a negative reduced cost in the master problem and should be added.

6.3 Modeling with transfer buses

One way to handle the problem of letting pupils go to different schools using the same vehicles is to allow changing of buses for the pupils and introduce the concept of transfer buses. These are vehicles driving between the schools, most often from a junior or intermediate school to the senior school, let us now for clarity assume that we allow transfer from a subset of junior and intermediate schools, the transfer schools, to the senior school. The transfer buses are either any of the vehicles which are at the school, that is they have already been used to take pupils to the school or they are designated transfer buses, coming from the garage with their only purpose to do the transfer stretch. It is now possible to put pupils who are going to the senior school in the problem to find routes to the junior or intermediate school. For the routes carrying pupils needing transfer it must be made sure that these arrive early enough for the transfer vehicles to reach the senior school in time. With extra conditions to the SP problem transfer buses are forced to be used to transport the senior school pupils from the junior or intermediate school to the senior school. The SP with the extra conditions is given below.

Variables:

$$
\begin{aligned}
x_r &= \begin{cases} 1, & \text{if route } r \text{ is in the solution} \\ 0, & \text{otherwise} \end{cases} \\
tr_{svt} &= \text{number of transfer buses from school } s \text{ of vehicle type } v \text{ and transfer type } t
\end{aligned}
$$

Sets:

$$
\begin{aligned}
\mathcal{R} &= \text{the set of } \textit{all} \text{ feasible routes} \\
\mathcal{B} &= \text{the set of bus stops with pupils} \\
\mathcal{S} &= \text{the set of transfer schools} \\
\mathcal{V} &= \text{the set of vehicle types} \\
\mathcal{T} &= \text{transfer type}
\end{aligned}
$$

Parameters:

$$
\begin{aligned}
c_r &= \text{cost of route } r \\
ctr_{svt} &= \text{cost of transfer bus } (s,v,t) \\
\delta_{br} &= \begin{cases} 1, & \text{if bus stop } b \text{ is served by route } r \\ 0, & \text{otherwise} \end{cases} \\
f_v &= \text{number of available vehicles of type } v \\
u_{vr} &= \begin{cases} 1, & \text{if vehicle type } v \text{ is used in route } r \\ 0, & \text{otherwise} \end{cases} \\
tcs_{sr} &= \text{the number of pupils at school } s \text{ needed to be transferred caused by route } r \\
q_v &= \text{capacity of vehicle type } v \\
tas_{srv} &= \begin{cases} 1, & \text{if vehicle type } v \text{ is used in route } r \text{ and drives to transfer school } s \\ 0, & \text{otherwise} \end{cases} \\
tab_t &= \begin{cases} -1, & \text{if type } t = \text{'school'} \\ 0, & \text{if type } t = \text{'designated'} \end{cases}
\end{aligned}
$$

To the objective function the costs of the transfer buses are added and it now reads

$$z = \sum_{r \in \mathcal{R}} c_r x_r + \sum_{s \in \mathcal{S}} \sum_{v \in \mathcal{V}} \sum_{t \in \mathcal{T}} ctr_{svt} tr_{svt}. \tag{49}$$

The constraints are

$$
\begin{aligned}
\sum_{r \in \mathcal{R}} \delta_{br} x_r &= 1, \ \forall b \in \mathcal{B}, && (50) \\
\sum_{r \in \mathcal{R}} u_{vr} x_r + \sum_{s \in \mathcal{S}} tr_{sv,t='designated'} &\leq f_v, \ \forall v \in \mathcal{V}, && (51) \\
\sum_{r \in \mathcal{R}} tcs_{sr} x_r - \sum_{v \in \mathcal{V}} \sum_{t \in \mathcal{T}} q_v tr_{svt} &\leq 0, \ \forall s \in \mathcal{S}, && (52) \\
\sum_{r \in \mathcal{R}} tas_{srv} x_r + \sum_{t \in \mathcal{T}} tab_t tr_{svt} &\geq 0, \ \forall s \in \mathcal{S}, \forall v \in \mathcal{V}, && (53) \\
x_r &\in \{0,1\}, \ \forall r \in \mathcal{R}, && (54) \\
tr_{svt} &\in \mathbb{Z}_+, \ \forall s \in \mathcal{S}, \forall v \in \mathcal{V}, \forall t \in \mathcal{T}. && (55)
\end{aligned}
$$

The new constraint (52), *transfer capacity*, makes sure that all pupils transported to a particular school in the set of transfer schools which is not their own are being transferred. The new constraint (53), *transfer availability*, forces the system to use transfer buses of some type if there are pupils that have to be transferred. Constraint (51) has been adjusted so that designated transfer buses are subtracted from the available fleet of vehicles.

7 Solving the problem

The model used is the SP formulation from section 6.2.1. In a column generation approach, where a small subset of $\mathcal{R}$ is used as a starting solution, the set of collected columns $\widetilde{\mathcal{R}}$ is extended by solving the subproblem described in section 6.2.2. Note that the RMP has integer constraints but these are linearly relaxed in this process. Contrary to in the cutting stock problem, rounding is not a good candidate for getting a integer solution to the final problem. When many columns have been generated the problem is solved once again but with integrality constraints. Also important to notice is that the subproblem only finds the column with the least reduced cost in the subset of $\mathcal{R}$ containing routes with pupils to one school only. The search for good routes in the rest of $\mathcal{R}$ is made by heuristic methods explained in this section.

Assumptions made are that there is only one garage, and that there are 4 types of vehicles with capacities of 50, 16, 8 and 4 people respectively.

7.1 Starting solution

The column generation method needs a feasible starting solution at hand. One way to construct a starting solution is simply to send a car to transport every pupil, given that there are cars available. A similar way has been used, marginally more advanced, where a vehicle is sent to every bus stop where there are pupils waiting. Depending on how many pupils there are at every stop a vehicle with an allowed capacity is sent.

If this naive approach fails, for example the number of available vehicles can be too small, one could use some heuristic approach to find a better starting solution or Lagrangian relax the vehicle availability constraints until a feasible solution is found.

There has also been experiments with using a better starting solution (a final solution from an earlier run). It could be a wise choice if a manually made feasible plan is available to use this as an input to the model. Experiments shows that it, naturally, reduces the time needed for solving the problem. Since some of the methods used in the solving process are time limited the extra time set free could be used to further improve the solution.

The naive way of constructing a starting solution explained above has been used throughout the experimenting and evaluation.

7.2 Solving the restricted master problem

To solve the LP relaxation of the RMP, or from here on just the LP, the simplex method explained in section 5.1.1 has been used. By just defining which variables there are, which constraints should be fulfilled and which objective function there is the problem is solved by calling `CPlex` with a command in the `AMPL` code. Linear problems with some ten thousand columns are no problem for `CPlex` to solve in just a few seconds.

7.3 Generating new columns

7.3.1 Routes to one school only

The ESPPTWCC for one school and one vehicle at a time is solved by a fast but inexact algorithm, that is there could exist columns with negative reduced cost not found by it. All routes found by this algorithm start at the depot (the first arc length is multiplied by *weight*) and end at a school. Thereafter the weighted cost between the school and the depot node and the fixed cost of the vehicle are added to the costs of the arcs in the route.

The algorithm used for generating new columns in each subproblem is a mix of the algorithms found in [6] and [7]. As in the latter but unlike the former only paths without subcycles is allowed. The concept of imaginary load found in [7] is not used.

It is a state-space search procedure where it is assumed that time is discretized. A *label* is a state (i, t, l) containing the current node i, current time t and accumulated load l together with an associated vector of already visited nodes at that route.

The cost $c(i, t, l)$ of all labels except for $(0, 0, 0)$ is originally set to ∞. Then a set of non processed labels NPL is created, from the start containing only the depot label $(0, 0, 0)$, with cost $g_v - \mu_v$. The labels containing the school node are not processed and kept in a special set, empty from the start, of solution labels.

The neighbors of the non processed labels are then explored and if the new label is feasible and $c(i, t, d) + c_{ij} - \pi_i < c(j, t + \max(t + t_{ij}, a_j), l + d_j)$ the cost of label $(j, t + \max(t + t_{ij}, a_j), l + d_j)$ is updated and the new label is put in NPL. The school label however, is put in another set of finished labels if the total reduced cost is less than zero. When all neighbors, including the school, are explored the original label is removed from NPL and a new one is selected, the one with the least accumulated time as suggested in [6].

In order to decrease the running time of the algorithm so called *dominated* labels are deleted from NPL. This is what makes the method inexact in addition to discretizing the time. A label (i_1, t_1, l_1) dominates another label (i_2, t_2, l_2) if all of the following conditions are fulfilled:

$$\begin{aligned} c(i_1, t_1, l_1) &< c(i_2, t_2, l_2), \\ t_1 &\leq t_2, \\ l_1 &\leq l_2. \end{aligned}$$

In pseudo-code, the algorithm can be described as follows:

```
Initialization
NPL = {(0,0,0)}
c(0,0,0) = startcost

while (size(NPS) > 0) {
    (i,t,l) = getNextLabel(NPS)
    for (j = neighbours of i) {
        if ( i != j & i != n+1 & t+t_ij <= b_j & d+d_j <= q & route feas ) {
            //Label is feasible
            if ( c(i,t,d)+c_ij-\pi_i < c(j,t+ max(t+t_ij,a_j),l+d_j) ) {
                //New label is better
                TryToInsert(NPL,c(j,t+ max(t+t_ij,a_j),l+d_j))
                c(j,t+ max(t+t_ij,a_j),l+d_j) = c(i,t,d)+c_ij-\pi_i
            }
        }

        if ( t+t_ij <= b_j ) {
            //Label is feasible
            if ( i = n+1 ) {
                if ( c(i,t,d)+c_ij-\pi_i < 0 ) {
                    //New column found with negative reduced cost
                    Insert(solutionlabels,c(j,t+ max(t+t_ij,a_j),l+d_j))
                    c(j,t+ max(t+t_ij,a_j),l+d_j) = c(i,t,d)+c_ij-\pi_i
                }
            }
        }
    }
    removeLabel(NPL,(i,t,l))
}
```

7.3.2 Routes to many schools

This algorithm is mainly the same as in 7.3.1 but searches a larger part of $\mathcal{R}$, including the set of routes visiting many schools, for columns with negative reduced costs. When exploring the neighbors of a label all bus stops, with pupils to other schools as well, and all schools are checked. If there are any pupils on that particular route going to a certain school also that school label is added to NPL and the route could thereafter be extended to other schools.

Before adding any school label to the set of finished labels it is made sure that all pupils on the bus arrive at their school.

7.4 Fixing variables

If some of the variables when solving the linear relaxation of the RMP attend the value 1 or a value close to 1 this variable can be thought to represent a good column. The possibility to fix such variable to 1 in the further iterative process of solving the relaxed RMP and the subproblem has been introduced. A motivation of using fixing is that quickly finding the variables present in the solution makes it possible to exclude the bus stops visited by these routes when generating new ones. This will speed up the algorithm and hopefully generate better columns for the rest of the bus stops. When solving the final IP the fixed variables are unfixed again.

7.5 Merging of routes to different schools

Another way, instead of using the transfer model, to approach the problem of letting pupils going to different schools use the same vehicles is to use, what we call, *merging* of routes. This is based on combining routes generated for the different schools with the standard column generation method in a heuristic way. An implementation has been done where routes are generated for the different school individually as in section 7.3.1 and then a greedy heuristic tries to combine feasible routes where in addition to the time windows and the capacity constraints it is made sure that pupils going to a particular school is picked up before the arrival at that school.

First a pair of the routes generated are chosen in some way. A problem specific matrix is constructed where, based on the geography of the area, the compatible schools are given. It is unnecessary to try merging routes generated for two schools far from each other. One of the routes, route 1, are tried to be enclosed in the other route, route 2, by first choosing the cheapest position to incorporate school 1 into route 2 and then every bus stop before school 1 in the same way one by one. If the whole route can be enclosed with the feasibility constraints fulfilled the merged route is saved. A simple sketch of two routes being merged into one is shown in Figure 5.

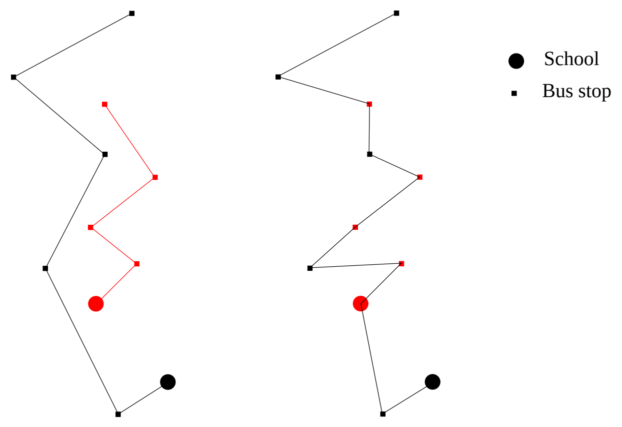


Figure 5: Two routes being merged.

Note that capacity constraints can be fulfilled even though the two routes individually might be crowded since some of the pupils leave the bus at the first school. If two routes cannot be merged because of capacity constraints, an attempt with a larger bus is made.

7.6 Generating improved columns

Some heuristics to improve on the columns generated have been implemented. The use of them could be described as simulated annealing (see section 5.2.2) where the temperature T is kept at zero degrees, which implies that worse solutions are not accepted. Minor experiments have been performed where also worse solutions were allowed but the results from these did not seem promising.

The operators implemented to find neighboring solutions are some of the traditional move operators used in genetic algorithms applied to problems similar to ours, see for instance [1]. They are explained one by one below.

7.6.1 2-Exchange

The *2-Exchange* operator works on one single route. The move operator exchanges the positions of two nodes i and j on the route, that is the arcs (i, i^+) and (j, j^+) are removed, the arcs (i, j) and (i^+, j^+) are added and the orientation of the arcs between i^+ and j are reversed.

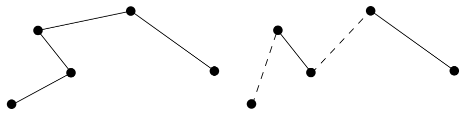


Figure 6: The 2-Exchange operator.

7.6.2 Move sequence

The *Move sequence* operator moves a sequence of n nodes from a route and inserts the sequence elsewhere on the same route. $n = 1$ and $n = 2$ have been used.

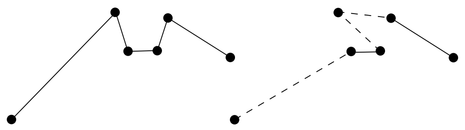


Figure 7: The Move sequence 1 operator.

7.6.3 Crossover

The *Crossover* operator takes two separate routes and exchanges the successors of node i in the first route and node j in the other route. That is, arcs (i, i^+), (j, j^+) are removed and arcs (i, j^+), (j, i^+) are added.

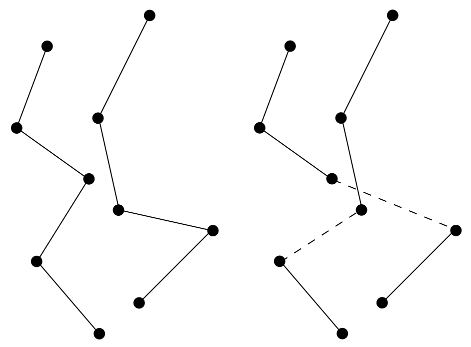


Figure 8: The Crossover operator.

7.6.4 Switch

The *Switch* operator exchanges the positions of nodes i and j in two different routes. It removes the arcs (i^-,i), (i,i^+), (j^-,j) and (j,j^+) and adds the arcs (i^-,j), (j,i^+), (j^-,i) and (i,j^+).

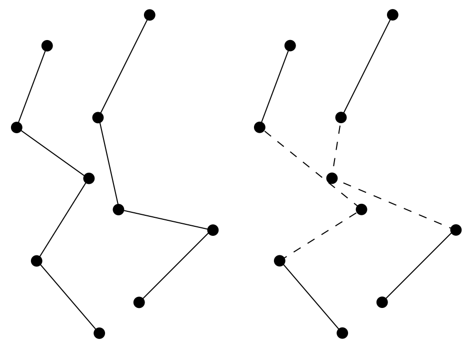


Figure 9: The Switch operator.

7.6.5 Relocation

The *Relocation* operator relocates node i into another route. This can be stated as removing of the arcs (i^-,i), (i,i^+), (j,j^+) and adding of the arcs (i^-,i^+), (j,i), (i,j^+), where j is a node in the other route.

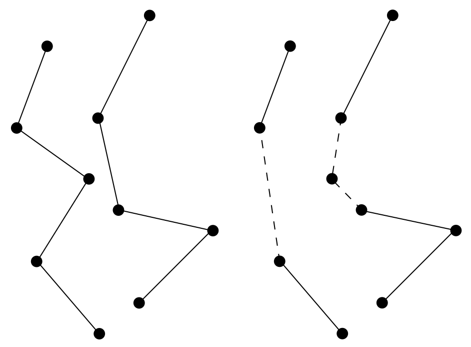


Figure 10: The Relocation operator.

For the move operators working on a pair of routes the same compatibility matrix as for the merging of routes is used.

The order of use of the operators in the program has been determined by trial and error, though different orders did not lead to dramatical differences in the final results.

Note that if the cost of a route is lowered by the 1-route operators, 2-Exchange and Move Sequence, the reduced cost of that route is lowered with the same amount since the dual variables are connected to the nodes rather than the arcs. For the 2-route operators, Crossover, Switch and Relocation, the reduced cost for the sum of the columns is lowered in the same way, when operated on.

7.7 Removing duplicated columns

There is no function in the route generator checking if a generated column is already present in the column pool, only that in the same iteration no identical columns are returned. A more time effective method than checking every time a column is generated that it is not already present in the column pool is to use a function that removes all duplicated routes after all route generation is done. The function only uses a few CPU seconds. In the runnings about 10% of the generated routes have been duplicates. The *removing duplicates* function is also used later in the process after the merging and the improving with the move operators where more duplicated columns are generated.

7.8 Solving the final integer problem

To solve the final integer problem the built-in IP solver in `CPlex` has been used. Here the branch-and-bound technique introduced in section 5.2.1 is implemented. As said before, this is a fast growing algorithm, and a time restriction on the CPU time available for the solver to search the tree has been used. It is possible that better results could be achieved using some other technique, such as *branch-and-price*, which is similar to branch-and-bound but where new columns are generated in the nodes. However this algorithm is not built into `CPlex` and the time restrictions on this project made it impossible to construct such a solver. There are similarities between branch-and-price and the fixing of variables described in section 7.4. Other branching schemes have been suggested, see [6], where branching is not made directly on the variables.

Instead of using the SP formulation when solving the final integer problem the corresponding SC formulation has been used. By the relaxation theorem it is known that the optimal value of the SC problem is at least as good as the optimal value of the SP problem, since the optimal solution to the SP problem is feasible in the SC problem as well. As said before, with the solution method used the whole set of routes $\mathcal{R}$ is not available. If there are two columns which both are very good but share one bus stop they will not be feasible together in the SP formulation. With columns from all of $\mathcal{R}$ also the same columns but without that particular bus stop visited would be present and one of them could be chosen instead and the SP constraints would be satisfied.

Removing a bus stop from a route will not make the route infeasible and the route will not be more expensive since the triangle inequality holds, thus it will still be in $\mathcal{R}$. Therefore, vehicle routing problems have the property that an optimal solution to the SP formulation has the same objective value as one to the SC formulation, since the latter can be modified to satisfy the constraints with equality with a cost that is at least as low.

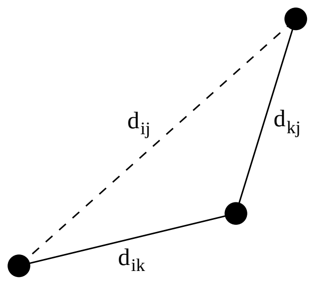


Figure 11: The distance as well as the time matrix fulfills the triangle inequality, $d_{ik} + d_{kj} \geq d_{ij}$.

7.9 Uncovering and improving the solution

The solution to a SC formulation of the final integer problem can, and probably will, include bus stops that are visited by more than one bus. By a simple greedy heuristic this solution can be improved by keeping the overcovered stops only on the routes where they are cheapest. This will be called to *uncover* the solution. Because of the triangle inequality in the distance matrix and thus for the costs an uncovered solution where every bus stop only are visited by one vehicle is at least as cheap as an overcovered solution. After uncovering the solution the improvement and merging of routes heuristics is run once more to improve the final solution. The step of uncovering, merging and improving the solution will be called to *post-improve* the solution.

A serious problem is that it is hard to predict how much improvement there will be in post-improving a solution to the final integer problem. In fact a worse solution to the SC than another could be better off in the end giving rise to unstable performance of the overall algorithm. More on this in section 8.10.

7.10 The programs

There are various ways to combine the solving of the LP's and IP's and the subproblems with the use of heuristics to improve the columns. The procedure used in default mode is:

Overview

1. Construct feasible starting solution.
2. Solve LP.
3. Generate routes for all schools and all vehicle types.
4. Improve and merge generated routes. Return the *returncols* best routes to column pool.
5. `if` (improvement $< limit$) || (elapsed time for routegenerator $> timelimit1$) `then`
   `goto` 6.
   `else goto` 2.

6. `goto` 7 (or fix $x_r \geq fixlimit$ to 1. `goto` 2).

7. Merge routes in column pool until (elapsed time for merging $> timelimit2$).

8. Improve routes in column pool until (elapsed time for improving $> timelimit3$).

9. Solve IP with branch-and-bound. Explore tree max $timelimit4$ s.

10. Post-improve solution.

In order to improve performance and decrease instability some other variations have been tried, not differing very much from the above scheme, see section 7.11.

When improving the columns returned by the route generator, item number 4 in the list above, the order of the 1-route move operators was chosen as *Move Sequence 2*, *Move Sequence 1* and then *2-Exchange*. If a trial to improve a route succeeds it is changed against the new one. All pair of routes, (*returncols*×*returncols*) pairs, are tried to being merged. A limit on this step has been imposed in some tests but that did not seem necessary. The 2-route move operators have not been used in this step since early tests suggested that the time needed was not worth it.

The merging on the whole set of routes, item number 7, is brought to an end by a time limit. To decide on which pair of routes to merge random variables are used together with the compatibility matrix. Other methods have been tried but the problem is to valuate which routes are good merging candidates. One could think that routes with a high LP variable value are good, but such a route normally is quite loaded with not much free space, why they perhaps only can be merged with routes which are almost empty depending on where in the route the schools are. Choosing a pair of routes so that the sum of pupils will fill one of the buses is not such a good idea either since some of the pupils are stepping off the bus at some place, typically before every pupil using the route are on, and there is no obvious pattern where in the routes the schools are located. Merging only the columns at least once used in a solution to the LP problem has been tried but that did not make the overall solution better but worse.

In item number 8, only the 2-route operators are used since the single routes already are improved when returned from the route generator. Random variables are used to choose which pair of routes to improve on as well as which move operator to use. Also here the compatibility matrix is used to define allowed route pairs.

When post-improving the solution in item number 10 the given solution set of routes is first uncovered and then the 1-route improvement heuristics are used on the outcome of the uncovering process. Then the merging and the 2-route improvement heuristics are used and finally the 1-route heuristics are used once again in hope of further improvement.

The setup can be sketched as Figure 12 on the following page where it can be seen which parts of the programs are written in `AMPL` and `C++` respectively. The sizes of the boxes hints about the sizes of the different parts of the program, though they are in no way scaled. The main program is in `AMPL` and the `C++` parts are called with the `shell` command from the `AMPL` program. In total, there are about 450 lines of `AMPL` code and 3800 lines of `C++` code, including the code written to display solution information and calculating key figures. There is some code written in `Matlab` as well, used to convert input data, display routes graphically and create the time diagrams.

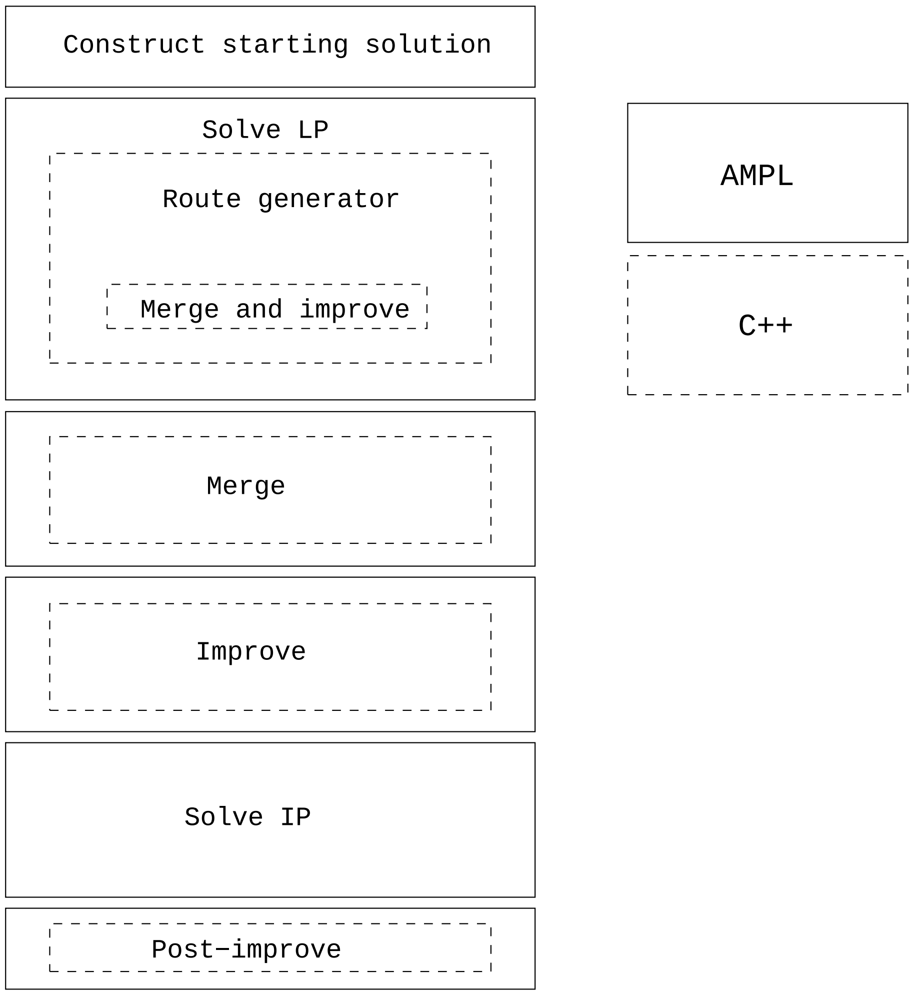


Figure 12: A box sketch over the parts of the constructed program with the used languages for the different parts given.

7.11 Altering the overall algorithm

The post-improving on the final solution gives rise to instabilities, as can be seen in section 8.

Trying to decrease the instabilities and improve overall performance some changes to the scheme in 7.10 was made. The idea here was to reuse the columns post-improved by adding them to the column pool and start over from the LP phase.

Overview

1. Construct feasible starting solution.
2. Solve LP.
3. Generate routes for all schools and all vehicle types.
4. Improve and merge generated routes. Return the $returncols$ best routes to column pool.

5. `if` (improvement $< limit$) || (elapsed time for routegenerator $> timelimit1$) `then` `goto` 6. `else goto` 2.
6. Merge routes in column pool until (elapsed time for merging $> timelimit2$).
7. Improve routes in column pool until (elapsed time for improving $> timelimit3$).
8. Solve IP with branch-and-bound. Explore tree max $timelimit4$ s.
9. Post-improve.
10. Add solution routes to column pool and `goto` 2 until (iterations $> maxiterations$)
11. Solve IP with branch-and-bound. Explore tree max $timelimit5$ s.
12. Post-improve.

7.12 Output from the programs

The output from the main program is a text file containing information about the routes in the solution, the vehicle types used and the bus stops visited. A `C++` program takes this file as input and gives statistics of the routes chosen with number of vehicles used, objective cost, driving distances, waiting and lag times as well as a kind of driving schedule, wherein at which time every stop is visited is given. An example of such a schedule is given in appendix B.1.

Originally a `Matlab` program takes the same file as input and outputs a map on which the routes are drawn. There are straight lines between the stops in the routes, the actual roads are not followed. This makes the map somewhat indistinct. With use of Kartena's products and databases an application in `C#` has been used to display the solution on a map. The solution statistics from the text file are given when the mouse marker is held over the routes, stops and schools. The user can choose between showing all routes at the same time, with the result that they lie on each other or showing them on by one. Two screenshots of a map with routes can be found in appendix B.3.

The `Matlab` program also outputs a bar diagram over the distribution of the waiting times for the actual plan (see appendix A) and a figure where it is possible to see when every bus stop is visited and their time windows. Such diagram can be found in appendix B.2. Different colors corresponds to pupils attending different schools. In these diagrams it is easy to see how often merged routes, where pupils going to different schools ride in the same vehicles, are used in the solution. The thicker horizontal lines represents the time windows of the schools. They really should be infinite to the left, it is allowed to arrive to the school at any time before it starts (or 5 minutes before), the feasibility lies in when the bus stops are visited. The black lines represents the vehicles and the time when they arrive at a certain bus stop is given by the intersection of the black line with the corresponding horizontal line.

8 Experiments

Throughout the experiments a standard setup has been used and then the parameters have been changed one by one to isolate the parameters as much as possible but still solve the problem to be solved in reality. If nothing else is said the parameter settings given in Tables 1 and 2 are used in the analysis.

Partly due to the extended rules but mostly due to the great increase in number of bus stops handled at once the time consumed by the route generation to many schools is very large. A small amount of testing suggests that the heuristic merging of routes to single schools gives final results of the same quality but very much faster than also including this type of route generation. Therefore, route generation has been made to single schools only.

Time limit on route generator	∞
Break route generator if improvement less than	0.001%
Use of fixing variables	No
Time limit on merging	$300s$
Time limit on improving	$300s$
Time limit on branch-and-bound	$300s$
Time limit on post-improving	$30s$
returncols	20

Table 1: Default algorithm settings.

weight	0.5
vehiclefactor	$1.5, 1.3, 1.0, 0.75$
fixed	$10000, 10000, 10000, 10000$
fleet	$\infty, \infty, \infty, \infty$

Table 2: Default problem settings.

8.1 The test problem

Real data from the municipality of Växjö has been used for trying out our ideas and to trim the algorithms. Data with information on all pupils in Växjö was given together with a plan made with `Skolskjuts` (which will be referred to as the existing plan). The significant data could be extracted and translated it into a form suitable for our programs.

As said before an assumption has been made that the problem for the whole municipality can be solved by solving it for each senior school pickup area at a time. The pickup area for Bergundaskolan was chosen as our test problem. This consists of the shaded parts of picture 13 on the next page, showing the municipality of Växjö. The dots are the bus stops with pupils on and the diamonds are the schools. The areas marked with black borders are the pickup areas of each junior school, the pickup areas of the intermediate schools are sometimes more than one junior school area. On the axes we have the standard RT90 coordinates [23].

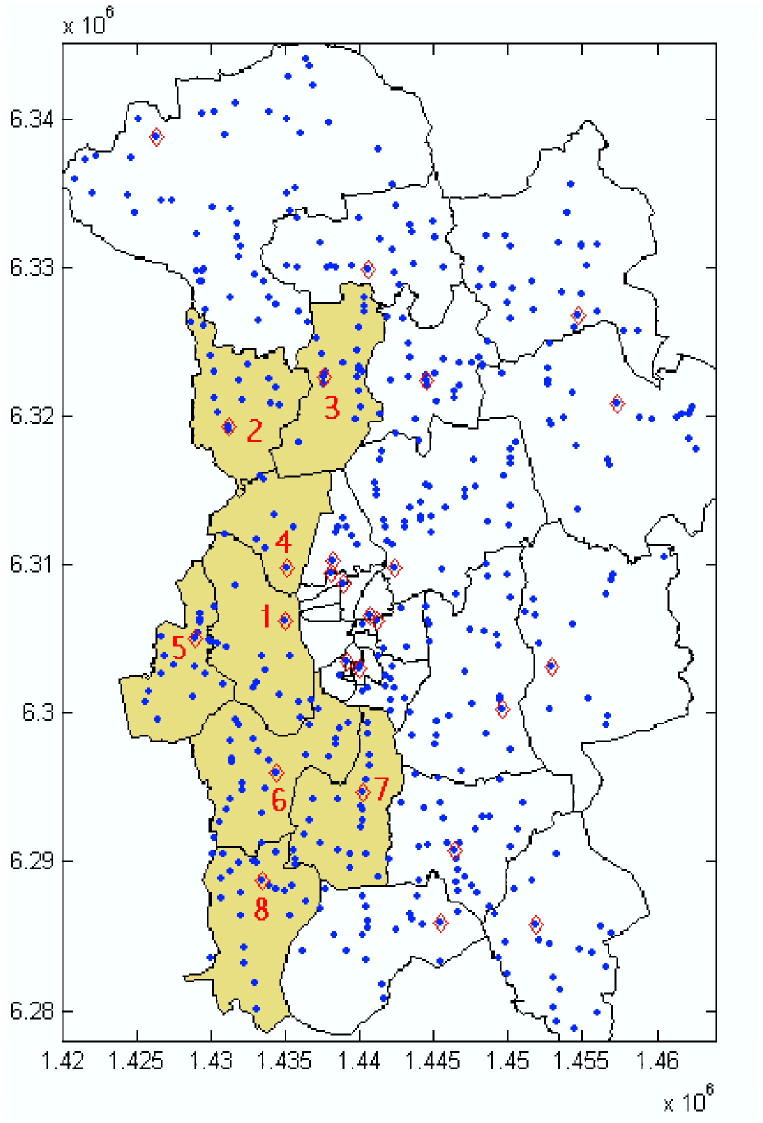


Figure 13: Växjö with pickup areas, bus stops, schools and chosen test problem.

The main reason for choosing this test area was that in this part the existing plan from Växjö was the most complete.

Name of school	Type	Location	Pickup areas
Bergundaskolan	Junior, Intermediate, Senior	1	1–8
Öhr skola	Junior	2	2
Åby skola	Junior, Intermediate	3	2–3
Öjaby skola	Junior, Intermediate	4	4
Gemla skola	Junior, Intermediate	5	5
Vederslöv skola	Junior, Intermediate	6	6–8
Kyrkskolan Tävelsås	Junior	7	7
Kalvsvik skola	Junior	8	8

Table 3: Schools in the test problem.

The existing plan of the area did not include the pupils going to Kalvsvikskolan, therefore these were excluded from the test problem.

The road network is taken care of in the `C#` application building the input matrices. The application calculates the fastest paths between every pair of bus stops, the time needed to drive these stretches and their distances. It is assumed that all vehicle types have the same mean velocity. The calculations are not exact since they depend on the fineness of the coordinate grid which they are made over. The euclidic distances between the actual coordinates and the coordinates of the points used in the fastest path calculations were calculated, showing that the mean weighted error (where the weighted error is defined as the quotient between the euclidic distance between the real and the used point divided by the calculated stretch) for the test problem was about 0.93% and the maximum error was as much as 330%. This huge error appeared for two bus stops very near to eachother but which used two different points in the calculation.

Our problem consists of 169 bus stops. 161 of these are stops where a total of 528 pupils are climbing aboard. Remember that stops where there are pupils going to different schools are duplicated. 7 of the stops are stops at the schools and the last stop represents the depot or garage where the vehicles start from. This has been assumed to be located at Bergundaskolan.

8.1.1 Assumptions on the times

To be able to solve the problem time has been discretized. The fineness of the discretization is in conflict to computer time needed for the calculations. A reasonable size of the time blocks was thought to be 30 seconds. When solving the test problem time windows given by rules we think are reasonable have been used, namely that a pupil cannot be picked up earlier than the starting time of the school less the fastest driving time to school plus 20 minutes or 35 minutes depending on if the fastest time to school is less or more than 15 minutes. It is reasonable that the allowed waiting time is longer for children living far away from school. It has been assumed that the buses must reach the school at least 5 minutes before the classes start and that the time per person climbing aboard is 3 seconds. 30 seconds has also been added at every bus stop to compensate for the deceleration and acceleration of the vehicle. The input matrices are given in units of seconds. After adding the aboard climbing time and the stop/start compensation time the time matrix was rounded to the nearest 30 seconds. As said before, waiting time will be defined as the difference in time between pickup and beginning of school and thus is minimum 5 minutes.

In some comparisons against the existing plan from Växjö their rules have been used where all pupils can be picked up 60 minutes or less before the school starts, not depending on where they live.

8.2 Using the transfer model

The lion part of the pupils needing transports are pupils living in a pickup area of a junior school going to the senior school. For the transfer model to be successful it is necessary that the transport buses are more effectively used, by for example picking up further students on the way between the schools, than just going straight between them. Early experiments showed that the transfer model did not give good results to the real problem. The merging technique was developed in parallel to the transfer model and early experiments with this technique was promising. Therefore it was decided to lay down the transfer model and use the merging one. One can think that the transfer model could be improved with more flexible transfer buses which maybe by some heuristic could pick up pupils during their journey, this is true but the merging technique was still thought to be better.

8.3 Using SC or SP formulation in the integer program

The purpose of this section is to show what was said about SC and SP formulation when solving the final IP in section 7.9. Three test problems, the standard problem, the standard problem with all fixed costs multiplied by 10 and the standard problem with all vehicle factors set to 1, have been solved 5 times each with the two different formulations. In Figure 14 the results are presented. The rings represents each run and the bars the mean values. This will be the standard way to display the results in diagrams.

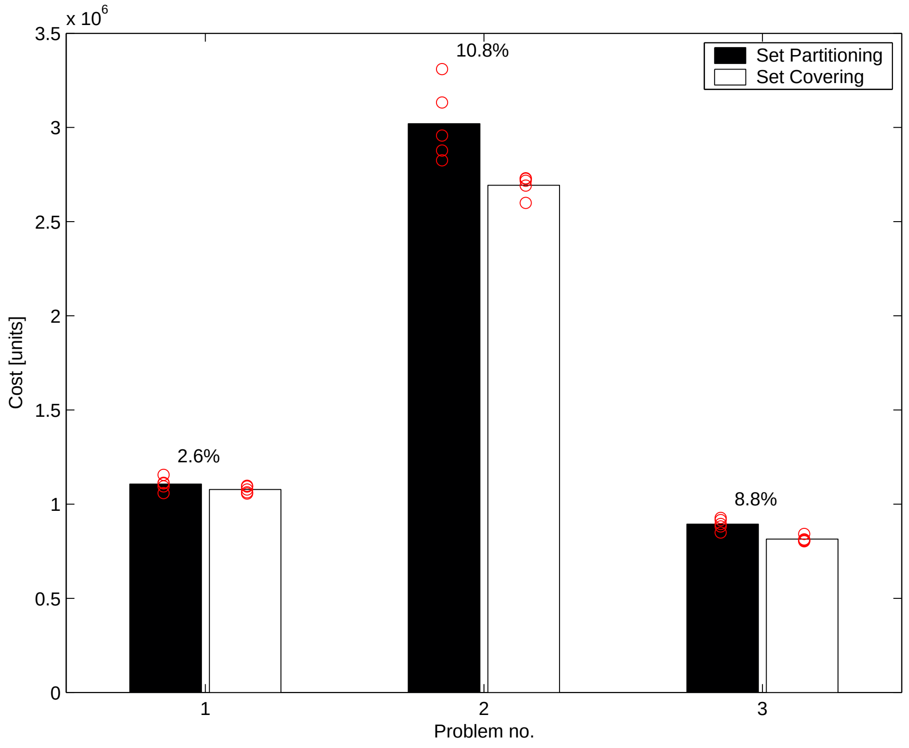


Figure 14: Results with SC and with SP formulation in the final IP. The differences are given as percentages.

The experiment clearly gives support to the theoretical discussion in section 7.9. As expected, SC formulation is better, which gives a hint of that only a very small subset of $\mathcal{R}$ is present. The values also vary more with the SP formulation. The reason for this is that it is hard to find a combination of columns that fulfills every constraint with equality. The set of possible solutions is smaller, as has been said earlier, why the effect of randomization is bigger.

8.4 Varying the *weight* parameter

Varying the weight parameter between 0 and 1 corresponds to varying how the stretches with no pupils in the buses affect the objective function and is a way to include the quality of service in the objective function to some extent. Using $weight = 0$ corresponds to that they do not count at all and $weight = 1$ that they count just as much as the loaded stretches, the stretches where some pupil rides with the vehicle. Strictly economically the total distance ($weight = 1$) would be the reasonable choice since the driving cost changes only marginally with the load, but in a more human sense it would be reasonable to focus on the stretches with pupils in the buses so that the pupils do not have to be in the buses more than necessary. The choice of *weight* somewhere in between would maybe be the best to fulfill both aims, one should definitely use $weight < 1$ to avoid routes picking up pupils at the first time but the second if a bus stop is passed twice on a certain route. Figure 15 shows how the total distance driven and the waiting times are affected by the *weight* parameter.

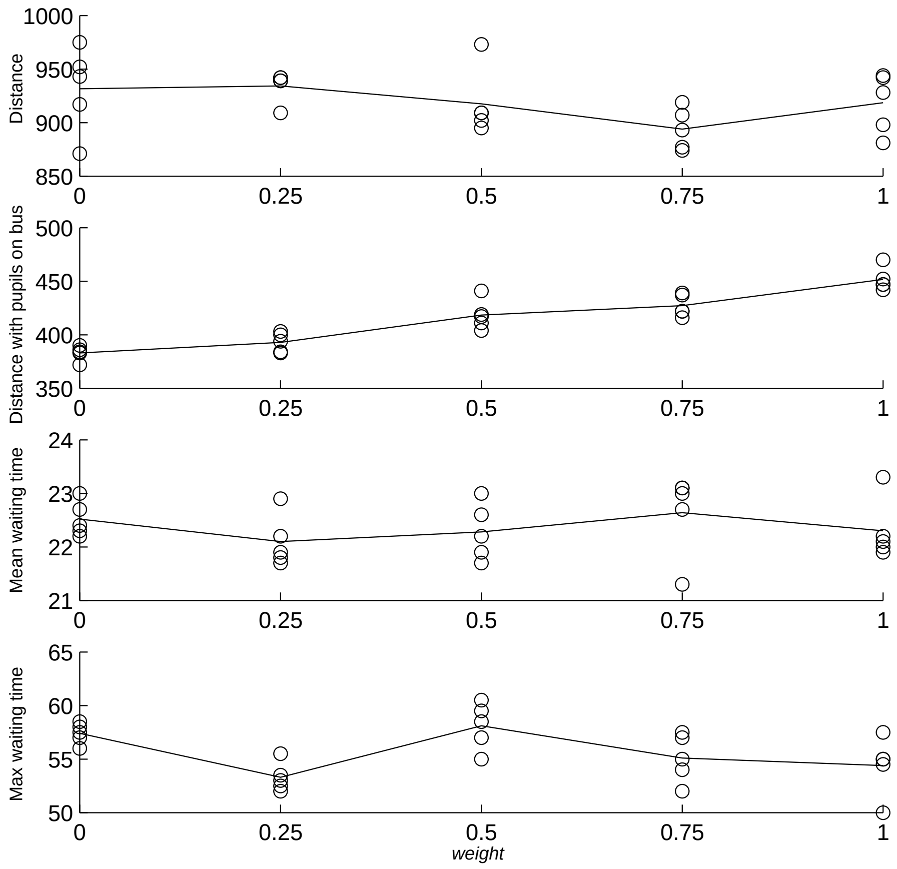


Figure 15: Varying the *weight* parameter. Implication on distances [km] and times [min].

The results are not very convincing in any way. The cost depend on a whole set of variables, for example it depends heavily on the number of vehicles used in the solution and when trying to minimize traveling distance without pupils using fewer, but larger, vehicles is preferred. Since these are more expensive the effect of varying the weight parameter is not obvious.

Some experiments with equal cost for the different vehicles and no fixed charge were also made. Here the results are a bit more as expected. With increasing weight parameter the vehicles tend to be fewer and the total distance is lowered at the expense of waiting times.

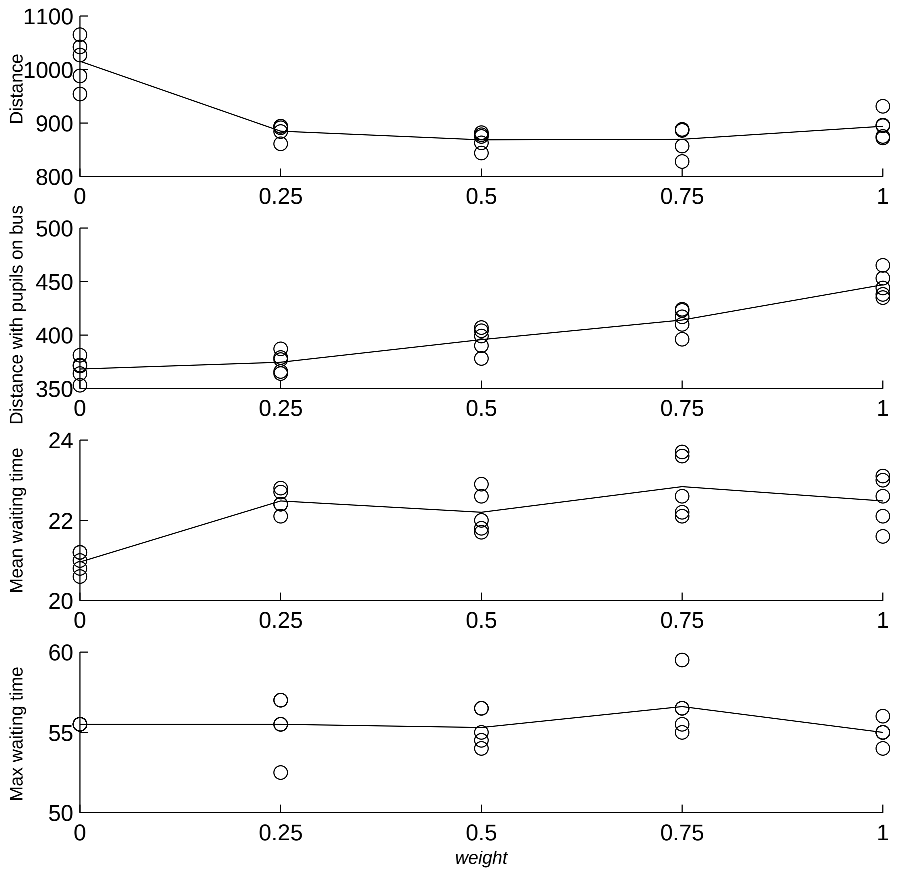


Figure 16: Varying the *weight* parameter with equal driving cost and no fixed charge. Implication on distances [km] and times [min].

If the waiting times do not change dramatically the use of a high parameter value is motivated since that would decrease the real operating cost (not to be mixed up with the objective cost).

Note that the reports outputted from the `Skolskjuts` program do only contain the distances of the stretches from the first pupil pickup to the last school. Minimizing these corresponds to the parameter value $weight = 0$ in our model apart from any empty stretches between a school and a bus stop with pupils going to another school.

8.5 Varying the fixed vehicle costs

In this section experiments with variation of the fixed vehicle costs have been made to see how this influence the results. Below is a figure showing how contributions to the objective cost from the number of vehicles and the driving costs changes when varying the fixed costs. The fixed costs are assumed to be the same for all vehicle types.

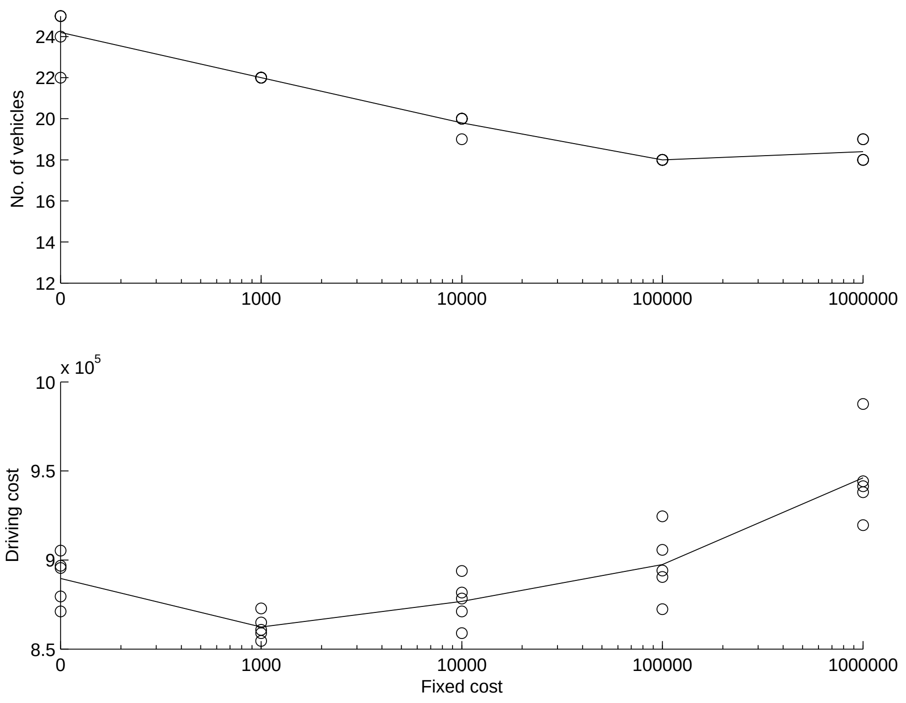


Figure 17: Varying the fixed vehicle costs [units]. Number of vehicles in the solution and the driving costs [units] contribution to the objective.

The results are not very surprising. When increasing the fixed vehicle costs the number of vehicles in the solution tend to be less. The driving costs are increasing with the fixed costs which is expected, since the focus moves from the driving costs toward the fixed costs, the driving costs become less and less important in the objective cost tried to be minimized.

More interesting is that we can see that with no fixed charge the driving costs are higher than with a small fixed charge, and the number of vehicles used is bigger. This is definitely not wanted in a practical optimization and the reasons for this behavior are not obvious. A possible cause is that no fixed charge on the routes decreases the importance of them being well crowded in the LP. When solving the IP there are only columns present with too few stops and thus many vehicles are needed to cover the constraints increasing the distance as well. The conclusion from this is that even if there is no fixed charge it is still better to set a small value in the model.

Another interesting point is that the results for the biggest fixed charge lead to equal or more vehicles and significantly bigger driving costs, another solution never wanted in a real plan. The explanation should be that the number of vehicles already is minimized with the smaller fixed charge. The driving costs for the biggest charge are now only about 5% of the total costs why the algorithm fails to keep them down as they are as small as the normal error margin. The conclusion is that if it is wanted to really minimize the number of vehicles it is better to set a reasonable fixed charge then, what in theory seem like a suitable choice, a fixed charge tending toward $+\infty$.

8.6 Varying the time windows

From early test runs it was concluded that with time windows as assumed in section 8.1.1 the waiting time for the pupils were substantially shorter than in the existing plan which is of course nice for the pupils but probably worse economically.

Wider time windows make the problem more difficult to solve since the number of feasible routes increases dramatically and the generation of new routes needs much more time.

To see how the time windows affect the key figures total distance driven, number of vehicles and waiting times a widening of the time windows was tried with 10 and 20 minutes for every pupil. This widening can be compared to the average time between start of school and earliest pickup time allowed (according to our rules) which is about 34 minutes. 5 runs on the, besides the time modification, standard problem gave the results in Table 4. All data in tables is measured in minutes, kilometers and kilounits for data regarding times, distances and costs respectively. With "Cost" the objective value is meant.

Extension	No. of vehicles	Total distance	Mean w. t.	Max w. t.
0	20.0	905	22.1	55.2
+10	14.6	731	25.5	62.1
+20	12.6	672	28.4	70.5

Table 4: Extending the time windows. Mean values from 5 runs with the standard setup.

Some very interesting results can be seen. By increasing the time windows we dramatically reduce the number of vehicles needed and the total distance driven but yet the mean waiting times are not affected that much. Some pupils are though considerably affected by the extended time windows since the maximum waiting time grows significantly. This result should be really interesting for the practitioners.

8.7 The size of the *returncols* parameter

According to LP theory every column with negative reduced cost found by the route generator will improve the solution value if it is entered into the linearly relaxed RMP, at least if the solution is non-degenerate. Traditionally when using a column generation strategy the one column with the least reduced cost is entered in every iteration. However, the route generator typically generates more than one column with negative reduced cost and since they all, in some sense, are good columns, it could be wise to include more than one of them in every iteration, as suggested in [6]. The fact that the variables corresponding to an optimal LP solution need not be optimal in the final IP is also reason for including more than one column at a time.

To include many columns each time reduces the time needed per column generated but it also worsen the average quality of the columns. The set of columns grows bigger with columns that may not be that likely to be in the final solution and managing all these increase the total time needed. The optimal choice of the *returncols* parameter, the upper bound on the columns returned from the route generator, is hard to find and it also varies for different problems and in different stages of the solution process. Since the optimal value varies so much experiments with changing its size have not been made, but a fixed size has been used for almost all problems with some exceptions.

An upper limit on the number of negative columns found in the route generator has also been imposed. When it has found *maxcols* columns with negative reduced cost it returns the *returncols* best ones, ordered by the values of the reduced costs. The size of *maxcols* has been chosen as ∞ in the default setup and no reason to change this has been seen.

8.8 Fixing of the variables

If using a strategy of fixing variables in the LP problem a crucial choice is the size of the *fixlimit* parameter. If the value of the linear variable, $x_r \in [0, 1]$, is higher than *fixlimit* for the actual LP the variable is fixed to 1 from then on in the iterative process of solving the relaxed RMP and the subproblem. A study has been made on solving the standard problem with different choices of the *fixlimit* parameter. The settings for the problems has been modified so that instead of breaking the route generator when the improvement is smaller than a certain value, a time limit is used. This because the CPU time needed for the different choices becomes more stable. After 300 seconds variables are fixed and the route generator is used for another 300 seconds. By using a value higher than 1 the case of not using fixing has been included also. The result, cost varying with parameter value, is shown in Figure 18.

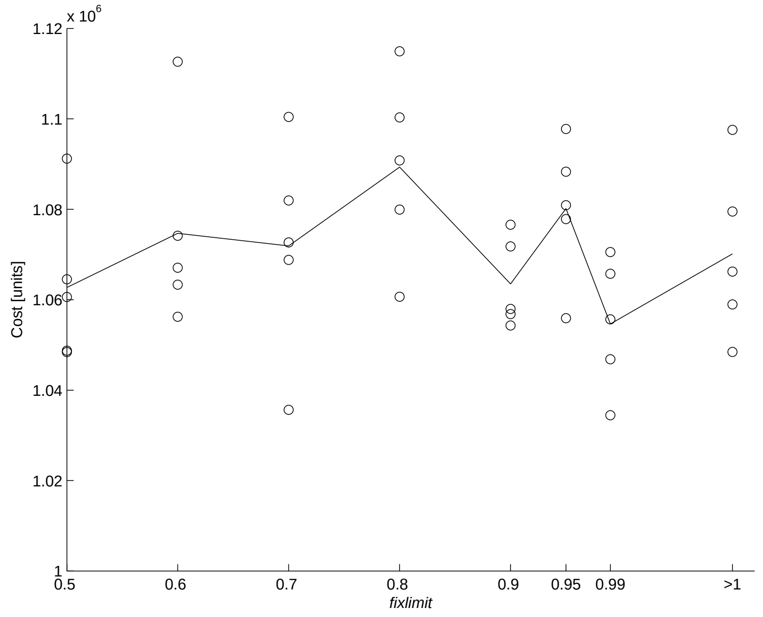


Figure 18: Cost varying with the size of the *fixlimit* parameter.

The conclusions drawn from the study are that the size of the *fixlimit* do not change the final solution value significantly. The fact that the value when not fixing variables (the values at the right border in the figure) do not differ significantly from the other suggest that fixing variables is not beneficial in this case. This is also confirmed by some other test runs on a couple of test problems. The reason for this is that the LP phase is just one part of the total solution process and the methods used thereafter are very important for the final set of routes available for the IP solver. A "bad" set of routes given by the LP phase is somewhat corrected when more improved and merged columns will be found. For a problem where the LP phase has bigger importance for the final solution the use of fixing variables still could be a good choice. One should probably in this case see that a high, but not too high choice of the parameter value, would be the best. A low value of the parameter fixes columns that logically are not very likely to be in the solution and a too high value will lead to that no columns or fewer columns than perhaps should be will be fixed. For our problem, the extra time needed for fixing the variables is better to use for merging or improving the set of columns.

8.9 Improvement with the different operators

To see what improvements are made by the using the 2-route operators the standard test problem were run 5 times counting the number of times, in 300 seconds, each operator manages to find a better pair of routes. The operator and the two routes are chosen at random and every operator is chosen about 45000 times during one run. The results are shown in Figure 19 on the facing page.

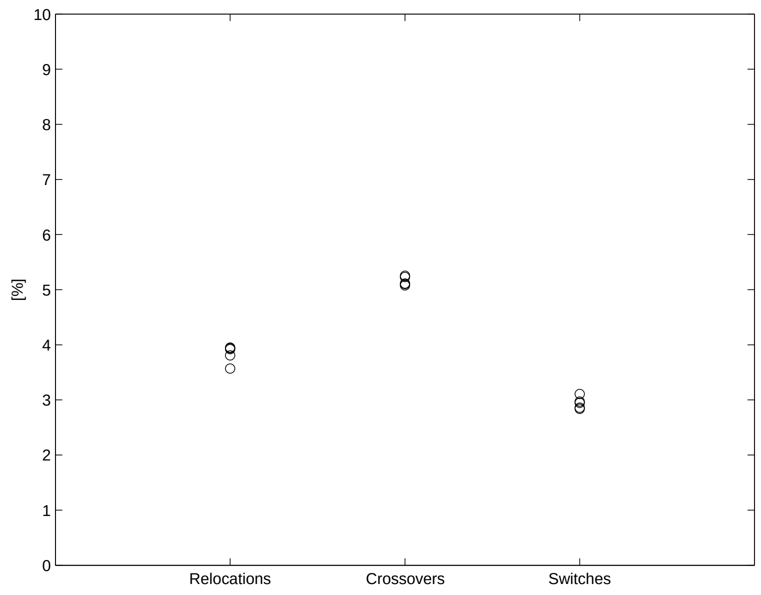


Figure 19: Percentage of all attempts where operators find a better pair of routes.

The percentages seem to be quite constant during the different runs of the programs. Since they all seem to be contributing about the same it is motivated to use all of them.

8.10 Robustness

Previous results show that the solution values between different runs with identical setup are not very stable. See for example Figures 14, 15 and 18. Obviously, the objective value varies heavily. It is also interesting to see how other key figures, such as the distances, the number of vehicles and the waiting times, vary. The standard test problem as well as two modifications of it were run 5 times each and the results are presented in Tables 5–7.

Cost	Distance	Distance w. p.	No. of vehicles	Mean w. t.	Max w. t.
1106	943	413	21	23.0	55.5
1104	948	427	20	22.9	51.5
1072	903	418	19	23.4	54.5
1082	894	438	19	22.7	54.5
1059	896	410	19	22.2	55.5
1085	917	421	19.6	22.8	54.3

Table 5: Results from 5 identical setups on the standard problem. Mean values at the bottom.

Cost	Distance	Distance w. p.	No. of vehicles	Mean w. t.	Max w. t.
2700	873	439	18	23.0	53.5
2746	929	415	18	23.5	59.0
2602	856	399	17	23.2	55.0
2728	881	418	18	23.2	60.0
2590	868	404	17	23.5	56.0
2673	881	415	17.6	23.3	56.7

Table 6: Results from 5 identical setups on the standard problem but with all fixed costs multiplied by 10. Mean values at the bottom.

Cost	Distance	Distance w. p.	No. of vehicles	Mean w. t.	Max w. t.
798	841	395	18	22.3	55.0
810	870	390	18	23.4	57.5
822	874	405	18	23.0	57.5
836	893	395	19	23.4	59.0
831	878	404	19	22.3	55.0
819	871	398	18.4	22.9	56.8

Table 7: Results from 5 identical setups on the standard problem but with all vehicle factors set to 1. Mean values at the bottom.

The difference between the best and the worst solution in terms of the objective costs are for the three test problems 4.4%, 6.0% and 4.8%. The difference between the other key figures varies between 2.2% (mean waiting time in the second test problem) and 11.5% (maximum waiting time in the second test problem). The mean improvement between the IP solution and the final solution was in this case 6.2%. The conclusions from this experiment is that the objective cost typically varies within around 5% and that the other figures vary about as much.

When solving a problem many times with identical settings all values are the same until the LP phase is done and the set of routes generated is the same when moving into the merging process. In the merging process and the improvement process thereafter random variables and time limits are used. The set of routes outputted from these will differ from each other. Remember that with a column generation approach columns are generated, which are the best if the linear relaxation of the RMP is to solve and when satisfied with the set of routes generated the hope is that the set also is good for the original IP. With different sets of routes generated to solve the problem linearly relaxed the solution values probably do not differ that much but with the IP at hand a small difference in the matrix A_R can lead to a great change in objective value. In section 8.3 there is evidence that the SC formulation, where it is allowed for the bus stops to be visited by more than one bus, is much better than the SP formulation in the IP. But this also leads to that the solution given by the IP solver may be overcovered, and a better solution will be found if the overcovered bus stops are uncovered, that is the solution routes is modified in some clever way so that only one vehicle will visit every bus stop. The uncovering of the solution set of routes may lead to that seats are released in the vehicles and it can be possible to merge some of the routes given by the IP solver. By merging routes in the solution one or more vehicles that the IP solver thought would be needed can be left unused which in case of large fixed vehicle costs have great influence on the total cost. The solver does not have any information about which columns put into the solution which can be improved and merged thereafter. The solver is in some way being tricked, it optimizes in blindness.

The standard problem has been run 25 times to see how often merging of the solution routes succeeds, that is how often one or more vehicle can be left out from the solution given by the solver and after uncovering.

Mergings succeeded	Number of runs
0	12
1	8
2	4
3	1

Table 8: Number of succeeded mergings of the solution routes, 25 runs on the standard problem.

In 20% of the runs 2 or more of the routes in the solution could be merged after the solution had been uncovered. In the standard problem the number of vehicles needed in the solution is around 20, why in 20% of the runs 10% or more of the vehicle fleet given by the IP solver could be left unused. The fixed costs contribution to the objective function are significantly decreased and it is not hard to understand that since this happens now and then, the solution value will be unstable.

Because of the instability it could be motivated to run the program for a couple of times and pick the best solution found. For problems that naturally are as unstable as ours this should be seen as a standard approach.

8.11 Running times

The application of a planning tool such as this determines the maximum allowed run time of the program. For practical use perhaps around one hour of run time may be suitable. This have affected the design of the program. If shorter run times are demanded, different algorithms can be used, perhaps taking advantage of present plans which only are adjusted step by step with some iterative heuristic approach.

It should be pointed out that a substantial part of the time needed is used when reading from and writing to files. This is slow in `AMPL` and the time needed for reading 30000 columns, in sparse form, is more than 160 seconds.

To control the total run time a number of different parameters can be adjusted. The sizes of these parameters have in the default mode been set by trial and error during the development and early testing of the model. Naturally the chances of finding better solution grows bigger if the program is allowed to run for a longer time. To see what happen, even if it is not allowed in an actual product, experiments have been made where the standard problem is run for substantially longer time.

Running the standard test problem with the standard setup (given in Table 1 and 2) demands about 30 minutes of time on our computer. About half of the time is needed before the LP phase is brought to an end and half of the time for the merging, improving, branch-and-bound and post-improving on the solution. When changing the running time affecting parameters according to Table 9 the total run time was about 6 hours on our computer. Still the LP phase needed about 15 minutes and the remaining time was used by the merging, improving and the branch-and-bound phases.

Time limit on route generator	∞
Break route generator if improvement less than	0.00001%
Time limit on merging	15000 s
Time limit on improving	15000 s
Time limit on branch-and-bound	15000 s
Time limit on post-improving	15 s

Table 9: Parameters used when experimenting with longer run times.

Running this setup 5 times showed that the mean cost decreased by about 1.1% compared to the standard setup.

Running with substantially smaller time limits has also been tested. Changing the parameters according to Table 10 lead to total run times of about 3 minutes, where the LP phase used about two thirds of the time.

Time limit on route generator	∞
Break route generator if improvement less than	10%
Time limit on merging	15 s
Time limit on improving	15 s
Time limit on branch-and-bound	15 s
Time limit on post-improving	5 s

Table 10: Parameters used when experimenting with short run times.

Running the standard problem 5 times gave a mean cost about 5.3% worse than with the default setup. The results of the smaller run times, together with the standard and the longer run times are presented in Figure 20.

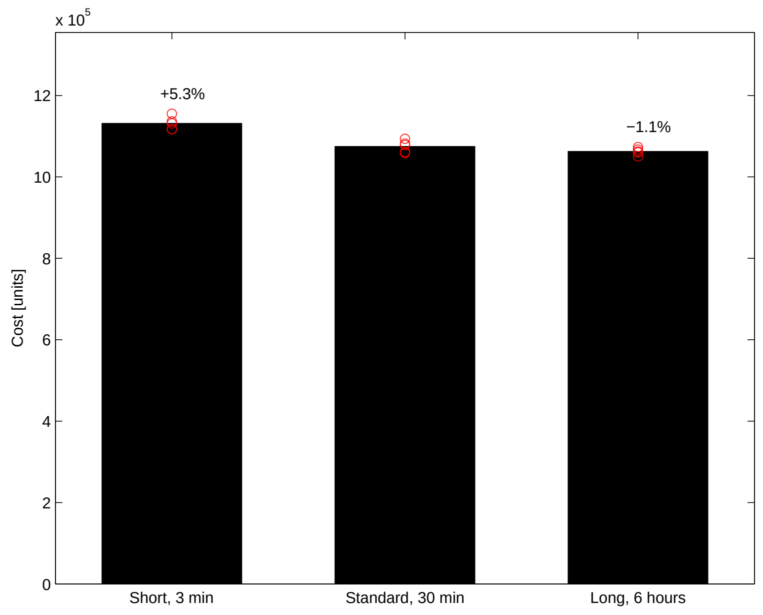


Figure 20: Results on the standard problem with different running times. Mean values, individual values and differences compared to the standard setup.

The results shows that running the program for just a few minutes gives a result that is not that far from the result with the default setup. 5.3% should though be seen as a significant difference and the extra time needed is well motivated. The diffence between the standard setup and the long setup is less clear, the best result from the 5 runs on the standard setup is better than the mean value for the longer setup. Since about 12 runs on the standard setup can be made in the same time as one run on the long setup this motivates that if the program is allowed to work for 6 full hours, it is better to do a lot of shorter runs and pick the best solution found than to do just one long run.

8.12 Altering the overall algorithm

In trying to decrease instabilities the algorithm in section 7.11 was tried on the standard problem from Table 2 on page 32. The parameters used can be seen in Table 11.

Time limit on route generator	∞
Break route generator if improvement less than	0.1%
Use of fixing variables	No
Time limit on merging	30 s
Time limit on improving	30 s
Time limit on branch-and-bound	60 s / 300 s
Time limit on post-improving	5 s
$returncols$	20
$maxiterations$	5

Table 11: Algorithm settings when reusing solution columns.

The performance of this algorithm seems to be a little better, with approximately the same running time. The results are shown in Table 12 and should be compared with those in Table 5 on page 41. The value achieved using this algorithm is on average 3.6% better than using the original one. The instabilities are still there though, with a difference of about 5% between the best and the worst result.

Cost	Distance	Distance w. p.	No. of vehicles	Mean w. t.	Max w. t.
1074	921	421	20	22.9	60.0
1037	871	405	19	22.8	52.0
1020	843	410	18	23.1	57.5
1041	886	407	19	22.3	54.0
1056	901	417	19	22.6	56.5
1046	884	412	19	22.7	56.0

Table 12: Results from 5 identical setups with alternative algorithm on the standard problem. Mean values at the bottom.

To see how the objective value varies with the number of iterations a plot was made from the most time consuming run in section 9.2.4, see Figure 21 on the following page. Using this algorithm, the objective value of the IP in the last iteration coincided with the value after post-improving the solution in every of the 10 runnings since the post-improved columns are returned to the set of columns. The reason that the objective value increases between the iterations in some of the tests is the time limit for finding the integer solution. Since the columns in the former solution are still present, a solution at least as good should be found without that limit.

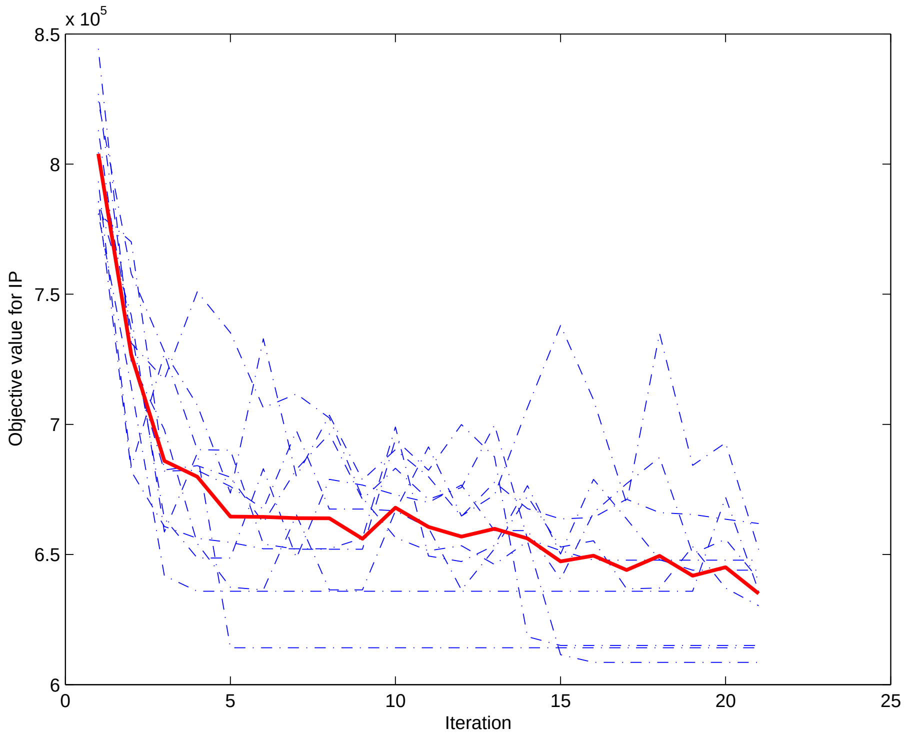


Figure 21: The objective value given by the IP solver as a function of the iteration number for 10 tests. Average result in bold style.

There is a trend in the figure suggesting that more iterations would lower the average result even more.

9 Evaluation

The distance calculations in the `Skolskjuts` program, made with `Network Analyst`, for the existing plan and the distance calculations made in the `C#` application with `RW NetServer` for our program are different. To investigate on the differences 10 routes were taken at random and the distances used in our program were compared to the distances given by constructing these routes in the `Skolskjuts` program. The mean value of the absolute values of the differences, which varied between 0.2% and 11.5%, was 2.9%, 4 of the routes were shorter in `Skolskjuts` than in our program, 6 were longer. The sum of the 10 routes was 0.5% shorter in our program than in `Skolskjuts`. The conclusion is that the individual differences between the routes can be big, but from our simple study no pattern can be seen and the size of the differences seem to be about the same in both directions.

9.1 Quantitative data from the existing plan

With the Växjö data came a plan made with the original program. Reports were built and out of these it was possible to extract data for the pupils in our test problem area. Unfortunately there were a couple of pupils in the project which had not been assigned to any route.

The actual regulations for the time windows in Växjö allow the pupils to be picked up at their bus stops not earlier than 60 minutes before the school starts [16]. For some pupils living far away from school the time windows are somewhat wider, but these pupils do not belong to the areas investigated in this project.

The planners at the municipality of Växjö have little concern in choosing vehicle types. They design the routes and leave for the public transports to assign vehicles to the routes. In their turn, the public transports purchase the individual stretches from local transport companies. In some cases the planners are asked to change their routes in order to make them fit better to the available vehicles.

In Växjö as well as in four other municipalities, Länstrafiken Kronoberg, is the company responsible for the transports being performed [17]. Länstrafiken are also responsible for the transportation service of the disabled. Totally their activities demands $375 - 400$ vehicles of all possible sizes. They use about 50 entrepreneurs performing the actual transports, each and every one of them with their own price tag. The actual prices are not available for us due to secrecy.

The most considerable problem for a comparison between the existing plan and a plan made with our program is that a valuation of a school transportation plan is not easy to do. The objective value in the optimization has been defined as a cost closely correlated to the real operating cost. But in the real world is it not that simple since the execution of the complete plan often is purchased from a transport company planning which vehicles to use. It is not obvious what the cost for each part in the plan is for the transport company. If, for example, the transport company uses a certain vehicle for public transport between point A and point B it may be able to fit in a school transport between point B and point C and then another public transport from point C. In this case it could be discussed what the fixed cost of the school transport should be. Such co-planning is also hard to include in the model but could be by adding more imaginary garages assuming that the other transports are fixed.

The quantitative outputs from the existing plan which can be used for comparisons are the total distance driven with pupils loaded, the number of routes and the waiting times for the pupils. The number of routes is probably not equivalent to the number of vehicles used. It is hard to know for a quantitative study what parameter setting that should be used to have the same objective as the one used for the existing plan.

Distance driven	362 km
Number of routes	20
Mean waiting time	35.4 min
Max waiting time	75.0 min

Table 13: Quantitative data from the existing plan.

It is likely that some vehicles are used for more than one route in this plan. In addition, probably more than one vehicle is needed for at least one very crowded route. The distribution of waiting times is shown in Figure 22 in appendix A. Here it can be seen that the 60 min rule is disobeyed several times.

There are several difficulties in comparing our results with the real data. There are pupils missing in the existing plan, the senior school areas intersect some and which vehicles used are unknown. Some efforts have however been made in making the prerequisites as similar as possible and thereby get a quantitative comparison as valid as possible.

One of the routes in the existing plan has a maximum load of 79 people. Since the largest vehicle in our model has been assumed to have room for 50 people not all of them fit in one bus. We find it likely that also in the existing plan more than one bus is needed on this route and hence the distance of this route has been counted twice. Some of the pupils in the problem area are not assigned to any route on Monday morning (491 out of 528 pupils are assigned in the existing plan). The reasons for the missing pupils are different. The majority of the missing pupils are assigned to a route that does not reach the school before almost 10 minutes after the start of school if it should pick up the first pupil according to the rules. Thus, this route does not fulfill the regulations and has not been used in the solution. Some other pupils have not been picked up since if the nearest route would do the extra stretch to visit their bus stops the route would not stay feasible. The plan has been modified and the infeasible route and the extra stretches have been added to the solution in which now all pupils will be picked up, but it will disobey the time rules even more. The problem with the intersection between the senior school areas has been taken care of by removing the part of the stretches without any of our pupils on. By looking at the schedules of the routes in the solution we have tried working out how many vehicles are needed to accomplish the transports. The conclusion drawn is that at least 16 vehicles are needed. To be able to compare the total distance driven an assumption has been made that all vehicles in the existing plan start from a garage at Bergundaskolan where they later return to. Note though that this assumption probably not is very accurate. With the adjustments to the existing plan and with the assumptions made, the quantitative data to compare against is given by Table 14.

Total distance	755 km
Distance with pupils	339 km
Number of vehicles	16
Mean waiting time	$\geq$ 35.4 min
Max waiting time	$\geq$ 75.0 min

Table 14: Comparison data from the existing plan.

Remember that the plan was not feasible before our adjustments, see Figure 22 on page 57 where 24 of the 491 pupils have waiting times longer than the allowed 60 minutes, and the rules are even more violated now. Since our model guarantee feasible solutions we are generous to the manual planner in the comparisons. When modifying the solution the waiting times grows bigger. It seems unfair to use the modified waiting times why the original times still are used.

9.2 Solving the real problem with different aims

The problem has been solved with a number of different parameter settings to show the flexibility of the program. Settings that are thought to be reasonable are used as well as settings that give better outputs to compare with the ones in the actual problem, even if we think that the solutions would be more expensive. The time windows used are the most restrictive of the Växjö rules and our own rules (the standard rules from section 8.1.1 + 10 minutes). From now own, the Växjö rules will be denoted *TW1*, the own rules *TW2* and the restrictive rules given by the lower bound of the Växjö rules and own rules will be denoted *TW3*. We think that latter rules are more reasonable, especially for the pupils living near the school. Probably, the manual planner implicitly uses rules more like ours when planning, if not there would probably be loud protests.

TW1	60 minutes
TW2	Standard rules + 10 minutes
TW3	minimum (Standard rules + 10 minutes, 60 minutes)

Table 15: Maximum waiting time for the different time windows.

The algorithm described in section 7.11 has been applied with settings as in Table 12 on page 45 to three different problems, two with time rules from *TW1* and one with *TW3*. With *TW3* a value of 10 instead of 5 for the *maxiterations* parameter has been used. The total running time was anyway shorter compared with the two other cases due to the narrower time windows.

Important to notice is that in the total distance the distance to and from a garage located at Bergundaskolan is included.

9.2.1 Optimizing with focus on distance using rules from Växjö

In the objective function $weight = 0$ was used, all vehicle factors were set to 1 and the fixed costs were set to 1000 units which according to section 8.5 seem to be good choices when minimizing the distance with pupils on bus. The *lag time* is defined as the difference between the waiting time and the fastest time to school. Compared to previous tables columns for mean and maximum lag times are now added. Table 16 contain the mean values from 5 runs. In appendix C.1 a table with all individual runs can be found.

Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
860	368	13.8	28.1	59.1	12.4	47.9

Table 16: Quantitative data from the plan with focus on distance with pupils on bus. Mean values from 5 runs.

The running times were between 2.5 and 3 hours, the high values mainly because the wide time windows make the running times of the route generator larger than normal. Although the distance driven becomes quite small these solutions suffer from very poor waiting times for the pupils, though allowed by the rules. Some pupils have to leave home more than 45 minutes before they have had to if they could travel directly to the school.

There are some major drawbacks with this objective function. At first, some buses carry pupils to one school first, then travels to another area *for free*, since $weight = 0$, and carry other pupils to another school. Secondly, some pupils living near a school are picked up very early and left at the school a very long time prior to start of school in order for the bus to be able to pick up pupils going to some other school.

9.2.2 Optimizing with focus on the number of vehicles using rules from Växjö

In the objective function $weight = 0$ was used and all vehicle factors were set to 1. The fixed costs were set to 100000 units which according to section 8.5 is suitable for minimizing the number of vehicles. The running times here were approximately 2 hours and 15 minutes. The complete results are presented in appendix C.2.

Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
766	402	12	29.6	58.7	13.9	46.6

Table 17: Quantitative data from the plan with focus on the number of vehicles. Mean values from 5 runs.

In 1 of the 5 runs a solution was found where 11 vehicles with 550 seats in total were used. This can be compared to the total number of pupils which is 528 though it is possible for a bus to transport more pupils than its capacity since pupils getting off the bus at a school leave seats for new ones. Since the vehicles are expensive and it is free to travel without pupils on the bus there will be even more traveling between school areas for the buses, compared to the previous section, forcing some pupils to wait at school for a really long time.

9.2.3 Optimizing with focus on both the number of vehicles and the total distance with our rules

Here $weight = 0.5$ was used since this would prevent the buses from traveling long distances for free between different school areas. The fixed costs were set to 10000 units in order to keep the number of vehicles down and all vehicle factors were set to 1. In addition, the time windows were tightened for the pupils living near the schools according to *TW3*. The running times for the algorithm with this setup were approximately 1 hour and 15 minutes. The complete results are presented in appendix C.3.

Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
704	373	14	24.4	57.3	8.8	31.5

Table 18: Quantitative data from the plan with focus on both total distance and number of vehicles. Mean values from 5 runs.

The focus on both vehicles and total distance leads to an increase of the distance with pupils on bus of about 1.4% compared to the case focusing only on distance with pupils. The total distance driven has decreased with 18% with approximately the same number of vehicles which lowers the real operating cost dramatically. The tightening of the time windows also benefits the pupils a lot. The worst lag time has decreased, on average between runnings, from 47.9 min to 31.5 min, or about 34%, and the mean lag time has decreased with about 29%.

A comparison between the combined objective function and the one focusing mostly on the number of vehicles shows that the number of vehicles in the former case is larger by two. The overall distance driven is less though, with a difference of about 8%. Also here the pupils benefit a lot from the tighter time windows given by our rules with a decrease in mean and max lag time of about 32% and 37%, respectively.

As expected, the focus on both number of vehicles and total distance increases both the number of vehicles and the distance with pupils on bus. However, the real operating cost could be substantially lower anyway since the total distance is much shorter. This depends heavily on the fixed costs of the vehicles which could be hard to model in the real world as well. In addition, the pupils will be a lot more satisfied, at least on average, with the narrower time windows.

Of course a nonzero *weight* could have been included in the two first cases; these examples served mostly to consider different extremities of the model.

9.2.4 Increasing running time

Some more time consuming runs of the problem with our time window rules and the combined objective function have been made. Three setups were used, one that needed on about 2 hours, one that needed about 3 hours and one that needed about 5 hours. The extension was the allowed time for merging and improving in the two former cases and both allowed time and the number of iterations in the latter. The following parameters for the algorithm described in section 7.11 were used:

Total time (approx.)	2 h	3 h	5 h
Time limit on route generator	∞	∞	∞
Break route generator if improvement less than	0.1%	0.1%	0.1%
Use of fixing variables	No	No	No
Time limit on merging	60 s	120 s	60 s
Time limit on improving	60 s	120 s	60 s
Time limit on branch-and-bound	180 s / 900 s	180 s / 600 s	180 s / 900 s
Time limit on post-improving	15 s	30 s	15 s
$returncols$	20	20	20
$maxiterations$	10	10	20

Table 19: Algorithm settings for the longer runs.

10 runs of each setup were made and the average values as in Table 18 on the preceding page were caluculated. Tables of the individual runs can be found in appendices C.4–C.6.

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
652	666	369	13.4	24.6	56.9	8.9	34.4
644	659	363	13.3	24.3	57.7	8.5	35.3
635	651	361	12.9	24.5	57.3	8.8	34.7

Table 20: Average values from longer runs. Running times are 2 h for the upper, 3 h for the middle and 5 h for the lower row.

As we can see, it is advantageous increasing the running time. Note that this is not in conflict to the conclusions in section 8.11 since we use the alternative algorithm, basically built on those inferences, where the problem is solved over and over again with the extension that information from previous runs is saved and used further on. Compared to the average value from Table 18 the improvements are 4.0%, 5.2% and 6.4% respectively. As always in optimization business there is a tradeoff between the quality of the solution and the time needed to find it.

9.3 Comparison against the existing plan

Comparing with the existing plan we see that even when focusing on the distance we still do not reach shorter distances with our program with reasonable running times if we look at the mean values. The difference in distance is about 8%, but we also see that the number of vehicles are about 14% fewer in our solution. In the existing plan feeding routes, where a short route is used to transport some pupils to a stop where they change to another route, are used sometimes, and bus changes have not been built into our model. Though it should be easy to manually adjust a solution and to include feeding routes. Even when focusing on the distance the design of our model will lead to solutions with fewer vehicles than in the existing plan. More vehicles could be added as feeders, this would reduce the total distance with pupils aboard, but it would at the same time lead to more vehicles and possibly higher costs, depending on the fixed charges compared to the driving costs. Feeders are good to use if the fixed cost of the feeder is small, if the vehicle used for the feeding route is parked near the route or if the vehicle cost of the feeding vehicle is substantially lower. Another reason why it is hard to produce solutions with shorter distance with pupils on bus with our program compared to the existing plan is that feasible solutions are found and the manual planner violates the time rules for about 5% of the pupils and the violations are even worse after the modifications to include all of the pupils in the area.

Focusing on the number of vehicles shows that it is possible to find a feasible solution that uses 4 vehicles fewer, that is, 25% less, than in the existing plan. If the manual planner really has focused on the number of vehicles used, this difference is remarkable.

When using an objective which combines the driving costs and the fixed costs in a reasonable way, the results show that the number of vehicles used in our plans in average are 13% fewer than in the existing plan. The distance with pupils aboard is 10% longer than the existing plan, the comparison still suffering from the fact that our plan is feasible and the existing one is not. The total distance driven, including the stretches to and from the first and last stop in each route, in the existing plan is unknown but with a rough assumption a figure is created. With our reasonable objective the empty stretches (with a garage at Bergundaskolan) are considered and it can be seen that this leads to, not surprisingly, shorter total distances than if they are not, as in the runnings focusing on distances and numbers of vehicles. The conclusion is that a solution marginally longer regarding distance with pupils on, is found, with significantly fewer vehicles, and where the empty stretches also are considered. In addition, the waiting times for the pupils are only about 2/3 of those in the existing plan, with the reservation that many vehicles in this plan arrive very early at the schools.

The best solution found, based on the objective thought to be reasonable, is presented in Table 21. This solution was found in one of the 5 hour runs from section 9.2.4 aiming at both distance and number of vehicles. The outputs from our program, the driving schedule, the arrival times diagram and the map with the routes, are presented in appendix B.

Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
626	351	12	24.1	59	8.4	35

Table 21: Best solution found.

A solution with lower distance with pupils on bus than in the existing plan has also been found on a 5 h run focusing on distance but with the $TW3$ time rules. One vehicle more is used and the total distance is longer compared to the existing plan.

Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
876	334	17	23.0	56.5	7.3	26.5

Table 22: Solution with shortest distance with pupils.

10 General conclusions and discussion

The use of our model for planning schools transport would certainly reduce the time needed for the planning. Probably it will also reduce the costs for the transports. The more information known and taken into account in a planning, such as garage locations and vehicle costs, the harder it is to find a good solution manually and the better it would be to use a model like ours. It would be very interesting, and a good test for the model, to make up a competition case and to compete against a manual planner with vehicle data specified in advance.

To be able to try the model on another area there are some problem-specific inputs needed. First the distance and time matrices are needed. To calculate these the `C#` program needs the coordinates for all bus stops in the problem. Text files must be constructed and supplied with the time windows for all bus stops. Another file should contain the number of pupils at the stops. A compatibility matrix is also needed where the compatible areas when merging and improving pairs of routes are given. Finally there is need for text files containing information about which bus stops belong to which schools.

The work is not a replacement for manual planning, and should not be seen as such, but a complement. It could be used to get a good solution for the manual planner to start from; the possibility to modify the solution given by our program manually is then of course a must, or to see if improving changes can be made to an existing plan. An iterative process between the program and the planner would probably be clever. A strength of the SP formulation is that manually created routes can be given any optional cost and added to the set of columns chosen from. Solving the RMP thereafter will deduce whether this manually created route is a good choice to include in the final solution or not. Other big advantages of the model are the flexibility built in and that the planner can try different parameter settings to examine how they will affect the results. The possibility to consider a limited number of vehicles with different costs which are parked at different locations leads to a model that is impossible to solve manually but will be solved to near-optimality with this model. One always comes back to the problem of valuating a plan though, and perhaps a solution given by the model could be unacceptable in the real world due to possible breakages of unspoken rules unknown to the model.

For the model to be really effective it would be advantageous if the person planning the routes is the one coordinating the vehicles. If the actual fleet and its location is known, it is definitely better taking advantage of that when trying to minimize costs.

An interesting remark from the experimental parts is that widening the time windows to some extent may lead to much cheaper solutions. The conclusion is that the planner could alter the time rules some to see if improvements can be made without worsening it too much for the pupils.

Before using a solution given by the model some routes must be driven to see if the time calculations in the time matrix and the assumptions on the times spent on the bus stops we have made are reasonable. The errors in the distance calculations for the distance matrix must also be studied more in detail to see if they have a practical influence. If so, the margins must be made bigger.

10.1 The algorithms and future work

The solution values vary considerably, which is typical for combinatorial problems like ours. To get a really good result one should run the program a couple of times and pick the best solution found. The alternative algorithm is basically built on this remark and seems to be the best choice. Furthermore, fixing of variables does not improve the results.

A more clever way to choose the pairs of routes in the merging and improving methods would be desirable. This would probably decrease the effect of randomness since if only the interesting routes are tried to being merged more good routes to many schools are found in less time. There is a need for further investigation of how to distribute the running time optimally between the phases route generation, improving and merging and how many overall iterations to do. This is likely to depend on the details of the specific problem to solve.

Further studies need to be done on other pickup areas than ours. It is possible that for areas with greater distances, different shapes, more bus stops or more schools there is a need for adjusted algorithms to solve the problem efficiently. To decrease running time, or make a better distribution of allowed running time, a manual geographic sectoring of the area, splitting the problem in smaller parts, could be tried. The route generation would probably need considerably less time and perhaps route generation to many schools can be used.

Instead of using branch-and-bound in the IP problem other algorithms may be implemented. The quality of the IP solution is not that important for the final solution because of the post-improving used thereafter. Some approximative heuristic could perhaps be used instead of branch-and-bound.

It is possible that the errors in the time and distance matrices influence on the quality of the result. A more accurate calculation of these, with a finer grid, would be desirable, perhaps also using different velocities for different vehicles.

Including feeding routes and bus changes, more flexible than in the transfer bus model, would be a natural step in a further development of the model.

10.2 Building a commercial program

`ArcView` is a powerful but expensive tool. It has many features but they are not all needed for the school transportation problem. With the use of Kartena's products a solution from the program can be displayed on a map with the routes drawn following the roads (see appendix B.3). A commercial program based on the methods in this thesis could look like this. With `ArcView` there is a direct connection between the map and the data. When using Kartena's map to display the solution it is just a graphical display, it is not possible to modify the data via the map. If (and this has to be the case for a commercial planning tool) it should be possible to adjust a solution manually, a connection from the map and back to the data is needed. Such a connection is not present now, but it should be possible to create.

When solving the linearly relaxed RMP as well as the final IP we have used `CPlex` which is a very expensive solver. There are other alternatives, able to perform the branch-and-bound procedure, on the market which are cheaper and the solver `glpk` [20] is even free. We have no knowledge of the quality of these programs and it should be emphasized that we have never tried any of these. The biggest difference between the solvers is probably the time needed for solving the problem. For solving LP's there are a number of alternatives. If the solving of the linear problems can be handled in some way but not the integer ones it is possible that the solving of the IP could be exchanged to some approximative heuristic. As has been seen, the quality of the solution given by the IP solver is not that important for the final result. Another competitor to `CPlex` is `XPress MP` [18]. This is an extension to Excel, it is well compatible with PC's and it is less expensive.

If it is possible to build all parts of the algorithm in a single program the information does not have to be stored in text files but in the memory, and the time now needed for reading and writing the text files would be released. Reading from files is very slow in `AMPL` and since this is done often removing this step would reduce the running times substantially.

References

[1] R. Bent and P. V. Hentenryck, *A two-stage hybrid local search for the vehicle routing problem with time windows*, Tech. Rep. CS-01-06, Department of Computer Science, Brown University, Box 1910, Providence, RI 02912, 2001.

[2] J. Braca, J. Bramel, B. Posner, and D. Simchi-Levi, *A computerized approach to the new york city school bus routing problem*, Working paper, (1994).

[3] A. Corber n, E. Fernandez, M. Laguna, and R. Mart , *Heuristic solutions to the problem of routing school buses with multiple objectives*, Journal of the Operational Research Society, 53 (2002), pp. 427–453.

[4] R. Fourer, D. M. Gay, and B. W. Kerninghan, *AMPL – A Modeling Language for Mathematical Programming*, Brooks/Cole, Canada, 2003. ISBN 0-534-38809-4.

[5] B. Kallehauge, J. Larsen, and O. B. Madsen, *Lagrangian duality applied on vehicle routing with time windows – experimental results*, Tech. Rep. IMM-TR-2001-9, Informatics and Mathematical Modelling, Technical University of Denmark, DK-2800 Kgs. Lyngby, Denmark, 2001.

[6] J. Larsen, *Vehicle routing with time windows–finding optimal solutions efficiently*, DORSnyt (engl.), no. 116, (1999).

[7] Q. Liu, H. C. Lau, D. Seah, and S. Chong, *An efficient near-exact algorithm for large-scale vehicle routing with time windows*, 1998. `citeseer.ist.psu.edu/266394.html`.

[8] S. G. Nash and A. Sofer, *Linear and Nonlinear Programming*, McGraw-Hill, Singapore, 1996. ISBN 0-07-114537-0.

[9] Skolverket, *Kostnader–Riksnivå, Sveriges officiella Statistik om förskoleverksamhet, skolbarnomsorg, skola och vuxenutbildning*, Rapport 247, 2004.

[10] M. M. Solomon, *Algorithms for the vehicle routing and scheduling problems with time windows constraints*, Operations Research, 35 (1998), pp. 254–265.

[11] M. Spada, M. Bierlaire, and T. Liebling, *Decision-aid methodology for the school bus routing and scheduling problem*. 3rd Swiss Transport Research Conference, Monte Verità/Ascona, Switzerland, Mar 19th–21th 2003.

[12] L. Spasovic, S. Chien, C. Kelnhofer-Feeley, Y. Wang, and Q. Hu, *A methodology for evaluating of school bus routing – a case study of riverdale, new jersey*. Transportation Research Board, 80th Annual Meeting, Washington D.C., USA, Jan 7th–11th 2001.

[13] S. R. Thangiah, *Vehicle routing with time windows using genetic algorithms*, Application Handbook of Genetic Algorithms: New Frontiers, Volume II, (1995), pp. 253–277.

[14] L. A. Wolsey, *Integer Programming*, John Wiley & Sons Inc., USA, 1998. ISBN 0-471-28366-5.

[15] *Handledning Tekis Skolskjutsapplikation*, Jan 16th 2004.

[16] R. Bucquoy, Planning Department at Skol- och barnomsorgsförvaltningen, Växjö, Telephone interview 2005-02-02.

[17] A. Johansson, Länstrafiken Kronoberg, Correspondence by e-mail 2005-02-09.

[18] Dash Optimization product overview, `www.dashoptimization.com/home/products/products_overview.html`.

[19] ESRI product overview, `www.esri.com/products.html`.

[20] GNU Linear Programming Kit, `www.gnu.org/software/glpk/glpk.html`.

[21] ILOG CPlex, `www.ilog.com/products/cplex`.

[22] Kartena AB, `www.kartena.se`.

[23] Lantmäteriet, `www.lm.se/geodesi/refsys/eng/refsys-eng.htm`.

[24] Matlab, `www.mathworks.se/products`.

[25] Tekis AB, `www.tekis.se`.

A Waiting time distributions

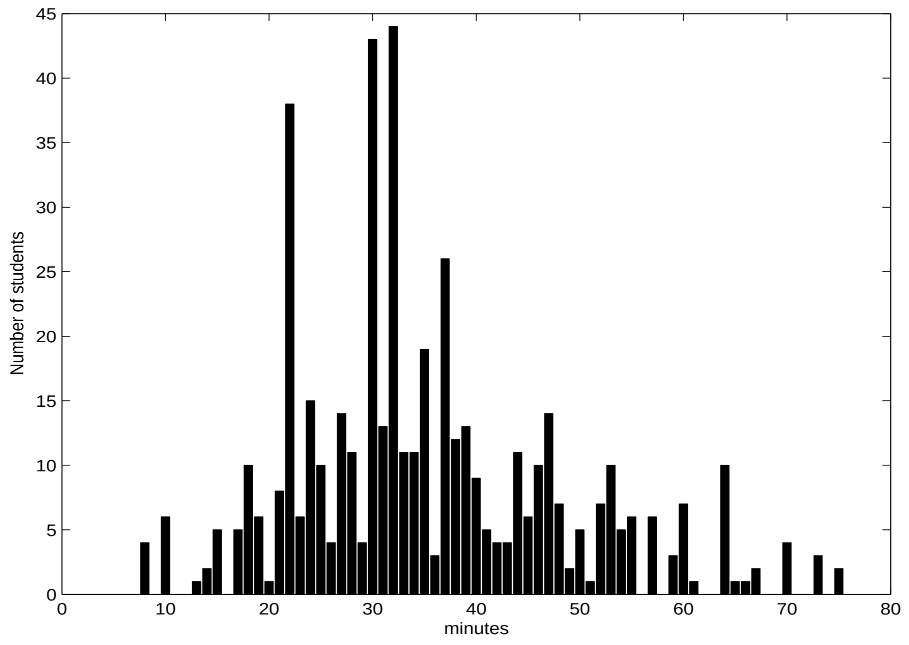


Figure 22: Distribution of waiting times for the existing plan.

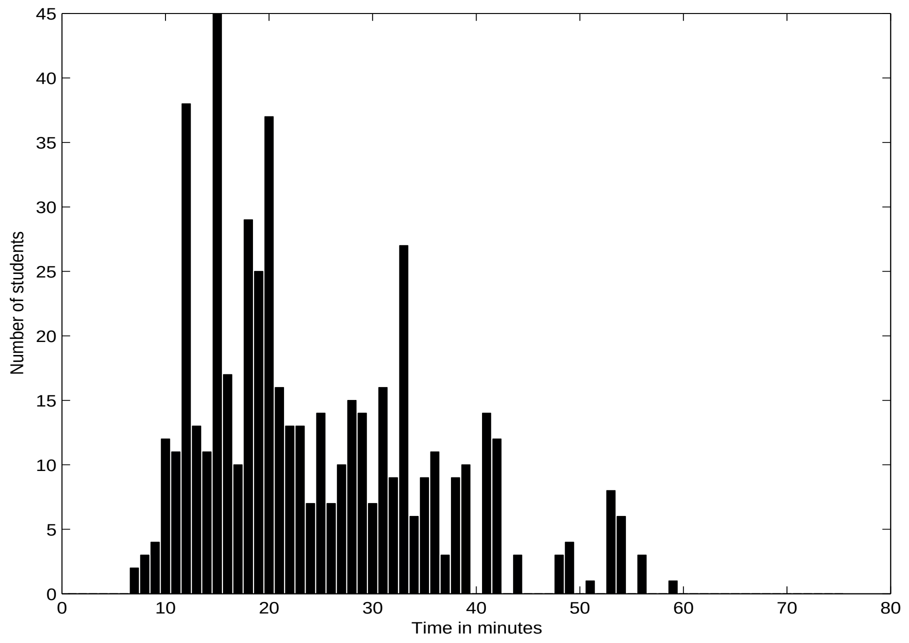


Figure 23: Distribution of waiting times in best solution found.

B Output from best solution found

B.1 Driving schedule

```
**********************

Route no 1 with a large bus

Total distance of route: 68.217 km
Total distance with pupils on bus: 19.444 km
Maximum load is: 22 (44%)

Busstop: 92    Time: 7:38   Pupils on bus: 2    Difference: 2
Busstop: 93    Time: 7:41   Pupils on bus: 5    Difference: 3
Busstop: 97    Time: 7:43   Pupils on bus: 8    Difference: 3
Busstop: 94    Time: 7:45   Pupils on bus: 11   Difference: 3
Busstop: 95    Time: 7:46   Pupils on bus: 12   Difference: 1
Busstop: 98    Time: 7:49   Pupils on bus: 13   Difference: 1
Busstop: 101   Time: 7:52   Pupils on bus: 15   Difference: 2
Busstop: 100   Time: 7:55   Pupils on bus: 20   Difference: 5
Busstop: 96    Time: 8:01   Pupils on bus: 21   Difference: 1
Busstop: 99    Time: 8:02   Pupils on bus: 22   Difference: 1
School:  3     Time: 8:05   Pupils on bus: 0    Difference: -22

**********************

**********************

Route no 2 with a large bus

Total distance of route: 34.934 km
Total distance with pupils on bus: 23.633 km
Maximum load is: 18 (36%)

Busstop: 103   Time: 7:40   Pupils on bus: 4    Difference: 4
Busstop: 104   Time: 7:42   Pupils on bus: 5    Difference: 1
Busstop: 106   Time: 7:48   Pupils on bus: 7    Difference: 2
Busstop: 59    Time: 7:48   Pupils on bus: 9    Difference: 2
Busstop: 105   Time: 7:51   Pupils on bus: 12   Difference: 3
Busstop: 58    Time: 7:51   Pupils on bus: 15   Difference: 3
Busstop: 102   Time: 7:58   Pupils on bus: 16   Difference: 1
Busstop: 55    Time: 7:59   Pupils on bus: 18   Difference: 2
School:  4     Time: 8:05   Pupils on bus: 7    Difference: -11
Busstop: 54    Time: 8:08   Pupils on bus: 8    Difference: 1
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -8

**********************

**********************

Route no 3 with a large bus

Total distance of route: 55.288 km
Total distance with pupils on bus: 37.195 km
Maximum load is: 40 (80%)

Busstop: 70    Time: 7:26   Pupils on bus: 6    Difference: 6
Busstop: 69    Time: 7:31   Pupils on bus: 7    Difference: 1
Busstop: 90    Time: 7:31   Pupils on bus: 8    Difference: 1
Busstop: 91    Time: 7:34   Pupils on bus: 10   Difference: 2
Busstop: 89    Time: 7:40   Pupils on bus: 12   Difference: 2
Busstop: 68    Time: 7:41   Pupils on bus: 15   Difference: 3
Busstop: 67    Time: 7:43   Pupils on bus: 16   Difference: 1
Busstop: 66    Time: 7:45   Pupils on bus: 20   Difference: 4
Busstop: 88    Time: 7:46   Pupils on bus: 22   Difference: 2
Busstop: 87    Time: 7:47   Pupils on bus: 24   Difference: 2
Busstop: 65    Time: 7:48   Pupils on bus: 25   Difference: 1
```

```
Busstop: 86    Time: 7:49   Pupils on bus: 26   Difference: 1
Busstop: 63    Time: 7:52   Pupils on bus: 30   Difference: 4
Busstop: 62    Time: 7:54   Pupils on bus: 32   Difference: 2
School:  2     Time: 7:55   Pupils on bus: 22   Difference: -10
Busstop: 78    Time: 8:00   Pupils on bus: 40   Difference: 18
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -40

**********************

**********************

Route no 4 with a large bus

Total distance of route: 60.978 km
Total distance with pupils on bus: 24.367 km
Maximum load is: 42 (84%)

Busstop: 151   Time: 7:31   Pupils on bus: 1    Difference: 1
Busstop: 128   Time: 7:33   Pupils on bus: 2    Difference: 1
Busstop: 154   Time: 7:37   Pupils on bus: 3    Difference: 1
Busstop: 131   Time: 7:40   Pupils on bus: 4    Difference: 1
Busstop: 129   Time: 7:42   Pupils on bus: 9    Difference: 5
Busstop: 130   Time: 7:44   Pupils on bus: 10   Difference: 1
Busstop: 135   Time: 7:48   Pupils on bus: 11   Difference: 1
Busstop: 136   Time: 7:50   Pupils on bus: 21   Difference: 10
Busstop: 152   Time: 7:51   Pupils on bus: 24   Difference: 3
Busstop: 137   Time: 7:54   Pupils on bus: 30   Difference: 6
Busstop: 138   Time: 7:56   Pupils on bus: 32   Difference: 2
Busstop: 143   Time: 7:58   Pupils on bus: 38   Difference: 6
Busstop: 142   Time: 8:00   Pupils on bus: 40   Difference: 2
Busstop: 141   Time: 8:03   Pupils on bus: 42   Difference: 2
School:  6     Time: 8:05   Pupils on bus: 0    Difference: -42

**********************

**********************

Route no 5 with a large bus

Total distance of route: 38.763 km
Total distance with pupils on bus: 25.371 km
Maximum load is: 46 (92%)

Busstop: 50    Time: 7:38   Pupils on bus: 7    Difference: 7
Busstop: 109   Time: 7:39   Pupils on bus: 15   Difference: 8
Busstop: 108   Time: 7:43   Pupils on bus: 18   Difference: 3
Busstop: 48    Time: 7:44   Pupils on bus: 19   Difference: 1
Busstop: 82    Time: 7:47   Pupils on bus: 36   Difference: 17
Busstop: 115   Time: 7:49   Pupils on bus: 39   Difference: 3
Busstop: 116   Time: 7:58   Pupils on bus: 40   Difference: 1
Busstop: 113   Time: 8:00   Pupils on bus: 43   Difference: 3
Busstop: 114   Time: 8:00   Pupils on bus: 46   Difference: 3
School:  5     Time: 8:03   Pupils on bus: 25   Difference: -21
Busstop: 45    Time: 8:05   Pupils on bus: 36   Difference: 11
Busstop: 77    Time: 8:06   Pupils on bus: 39   Difference: 3
Busstop: 46    Time: 8:07   Pupils on bus: 43   Difference: 4
Busstop: 47    Time: 8:08   Pupils on bus: 46   Difference: 3
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -46

**********************

**********************

Route no 6 with a large bus

Total distance of route: 67.609 km
Total distance with pupils on bus: 45.463 km
```

```
Maximum load is: 40 (80%)

Busstop: 71    Time: 7:24   Pupils on bus: 2    Difference: 2
Busstop: 72    Time: 7:27   Pupils on bus: 5    Difference: 3
Busstop: 84    Time: 7:32   Pupils on bus: 8    Difference: 3
Busstop: 75    Time: 7:36   Pupils on bus: 10   Difference: 2
Busstop: 76    Time: 7:39   Pupils on bus: 17   Difference: 7
Busstop: 73    Time: 7:41   Pupils on bus: 20   Difference: 3
Busstop: 74    Time: 7:42   Pupils on bus: 27   Difference: 7
Busstop: 64    Time: 7:47   Pupils on bus: 30   Difference: 3
Busstop: 60    Time: 7:56   Pupils on bus: 32   Difference: 2
Busstop: 56    Time: 7:57   Pupils on bus: 33   Difference: 1
Busstop: 57    Time: 7:59   Pupils on bus: 36   Difference: 3
Busstop: 41    Time: 8:07   Pupils on bus: 40   Difference: 4
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -40

**********************

**********************

Route no 7 with a large bus

Total distance of route: 33.918 km
Total distance with pupils on bus: 23.666 km
Maximum load is: 45 (90%)

Busstop: 107   Time: 7:39   Pupils on bus: 1    Difference: 1
Busstop: 51    Time: 7:47   Pupils on bus: 3    Difference: 2
Busstop: 110   Time: 7:48   Pupils on bus: 5    Difference: 2
Busstop: 52    Time: 7:51   Pupils on bus: 7    Difference: 2
Busstop: 111   Time: 7:52   Pupils on bus: 16   Difference: 9
Busstop: 53    Time: 7:55   Pupils on bus: 20   Difference: 4
Busstop: 112   Time: 7:55   Pupils on bus: 27   Difference: 7
Busstop: 117   Time: 7:58   Pupils on bus: 44   Difference: 17
School:  5     Time: 8:02   Pupils on bus: 8    Difference: -36
Busstop: 43    Time: 8:02   Pupils on bus: 23   Difference: 15
Busstop: 44    Time: 8:05   Pupils on bus: 43   Difference: 20
Busstop: 40    Time: 8:10   Pupils on bus: 45   Difference: 2
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -45

**********************

**********************

Route no 8 with a large bus

Total distance of route: 53.564 km
Total distance with pupils on bus: 25.561 km
Maximum load is: 40 (80%)

Busstop: 159   Time: 7:27   Pupils on bus: 1    Difference: 1
Busstop: 163   Time: 7:29   Pupils on bus: 4    Difference: 3
Busstop: 158   Time: 7:31   Pupils on bus: 6    Difference: 2
Busstop: 157   Time: 7:32   Pupils on bus: 8    Difference: 2
Busstop: 167   Time: 7:38   Pupils on bus: 9    Difference: 1
Busstop: 156   Time: 7:39   Pupils on bus: 12   Difference: 3
Busstop: 155   Time: 7:41   Pupils on bus: 15   Difference: 3
Busstop: 119   Time: 7:41   Pupils on bus: 16   Difference: 1
Busstop: 120   Time: 7:43   Pupils on bus: 19   Difference: 3
Busstop: 150   Time: 7:45   Pupils on bus: 21   Difference: 2
Busstop: 168   Time: 7:45   Pupils on bus: 22   Difference: 1
Busstop: 121   Time: 7:48   Pupils on bus: 24   Difference: 2
Busstop: 160   Time: 7:49   Pupils on bus: 26   Difference: 2
Busstop: 122   Time: 7:50   Pupils on bus: 28   Difference: 2
Busstop: 161   Time: 7:50   Pupils on bus: 29   Difference: 1
Busstop: 123   Time: 7:51   Pupils on bus: 31   Difference: 2
Busstop: 162   Time: 7:52   Pupils on bus: 33   Difference: 2
```

```
Busstop: 148   Time: 7:53   Pupils on bus: 40   Difference: 7
School:  7     Time: 7:55   Pupils on bus: 19   Difference: -21
School:  6     Time: 8:05   Pupils on bus: 0    Difference: -19

**********************

**********************

Route no 9 with a large bus

Total distance of route: 42.715 km
Total distance with pupils on bus: 22.505 km
Maximum load is: 26 (52%)

Busstop: 144   Time: 7:34   Pupils on bus: 3    Difference: 3
Busstop: 149   Time: 7:39   Pupils on bus: 4    Difference: 1
Busstop: 145   Time: 7:44   Pupils on bus: 6    Difference: 2
Busstop: 126   Time: 7:48   Pupils on bus: 7    Difference: 1
Busstop: 125   Time: 7:49   Pupils on bus: 9    Difference: 2
Busstop: 124   Time: 7:52   Pupils on bus: 12   Difference: 3
Busstop: 118   Time: 7:54   Pupils on bus: 20   Difference: 8
Busstop: 146   Time: 7:59   Pupils on bus: 24   Difference: 4
Busstop: 147   Time: 8:01   Pupils on bus: 26   Difference: 2
School:  6     Time: 8:05   Pupils on bus: 0    Difference: -26

**********************

**********************

Route no 10 with a large bus

Total distance of route: 58.294 km
Total distance with pupils on bus: 38.892 km
Maximum load is: 50 (100%)

Busstop: 24    Time: 7:21   Pupils on bus: 1    Difference: 1
Busstop: 22    Time: 7:24   Pupils on bus: 2    Difference: 1
Busstop: 16    Time: 7:27   Pupils on bus: 7    Difference: 5
Busstop: 25    Time: 7:29   Pupils on bus: 8    Difference: 1
Busstop: 23    Time: 7:31   Pupils on bus: 11   Difference: 3
Busstop: 9     Time: 7:38   Pupils on bus: 12   Difference: 1
Busstop: 10    Time: 7:39   Pupils on bus: 18   Difference: 6
Busstop: 12    Time: 7:42   Pupils on bus: 20   Difference: 2
Busstop: 13    Time: 7:43   Pupils on bus: 21   Difference: 1
Busstop: 61    Time: 7:44   Pupils on bus: 28   Difference: 7
Busstop: 8     Time: 7:46   Pupils on bus: 32   Difference: 4
Busstop: 15    Time: 7:48   Pupils on bus: 36   Difference: 4
Busstop: 14    Time: 7:49   Pupils on bus: 37   Difference: 1
Busstop: 34    Time: 7:55   Pupils on bus: 40   Difference: 3
Busstop: 33    Time: 7:57   Pupils on bus: 42   Difference: 2
Busstop: 37    Time: 8:03   Pupils on bus: 45   Difference: 3
Busstop: 38    Time: 8:05   Pupils on bus: 46   Difference: 1
Busstop: 35    Time: 8:07   Pupils on bus: 49   Difference: 3
Busstop: 36    Time: 8:11   Pupils on bus: 50   Difference: 1
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -50

**********************

**********************

Route no 11 with a large bus

Total distance of route: 53.175 km
Total distance with pupils on bus: 33.331 km
Maximum load is: 45 (90%)

Busstop: 166   Time: 7:26   Pupils on bus: 2    Difference: 2
```

```
Busstop: 164   Time: 7:29   Pupils on bus: 3    Difference: 1
Busstop: 165   Time: 7:30   Pupils on bus: 4    Difference: 1
School:  7     Time: 7:32   Pupils on bus: 0    Difference: -4
Busstop: 11    Time: 7:38   Pupils on bus: 4    Difference: 4
Busstop: 133   Time: 7:42   Pupils on bus: 7    Difference: 3
Busstop: 134   Time: 7:45   Pupils on bus: 8    Difference: 1
Busstop: 127   Time: 7:47   Pupils on bus: 13   Difference: 5
Busstop: 132   Time: 7:49   Pupils on bus: 15   Difference: 2
Busstop: 81    Time: 7:50   Pupils on bus: 16   Difference: 1
Busstop: 153   Time: 7:51   Pupils on bus: 19   Difference: 3
Busstop: 80    Time: 7:53   Pupils on bus: 20   Difference: 1
Busstop: 31    Time: 7:56   Pupils on bus: 23   Difference: 3
Busstop: 139   Time: 7:56   Pupils on bus: 26   Difference: 3
Busstop: 32    Time: 7:58   Pupils on bus: 30   Difference: 4
Busstop: 140   Time: 7:59   Pupils on bus: 34   Difference: 4
Busstop: 26    Time: 8:01   Pupils on bus: 45   Difference: 11
School:  6     Time: 8:02   Pupils on bus: 24   Difference: -21
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -24

**********************

**********************

Route no 12 with a large bus

Total distance of route: 58.522 km
Total distance with pupils on bus: 31.663 km
Maximum load is: 42 (84%)

Busstop: 17    Time: 7:36   Pupils on bus: 1    Difference: 1
Busstop: 79    Time: 7:39   Pupils on bus: 2    Difference: 1
Busstop: 18    Time: 7:41   Pupils on bus: 5    Difference: 3
Busstop: 83    Time: 7:45   Pupils on bus: 10   Difference: 5
Busstop: 19    Time: 7:47   Pupils on bus: 13   Difference: 3
Busstop: 21    Time: 7:49   Pupils on bus: 14   Difference: 1
Busstop: 20    Time: 7:51   Pupils on bus: 16   Difference: 2
Busstop: 28    Time: 7:55   Pupils on bus: 17   Difference: 1
Busstop: 27    Time: 7:56   Pupils on bus: 18   Difference: 1
Busstop: 29    Time: 7:57   Pupils on bus: 23   Difference: 5
Busstop: 30    Time: 8:00   Pupils on bus: 28   Difference: 5
Busstop: 85    Time: 8:04   Pupils on bus: 31   Difference: 3
Busstop: 49    Time: 8:06   Pupils on bus: 32   Difference: 1
Busstop: 42    Time: 8:08   Pupils on bus: 41   Difference: 9
Busstop: 39    Time: 8:10   Pupils on bus: 42   Difference: 1
School:  1     Time: 8:15   Pupils on bus: 0    Difference: -42

**********************

Total cost: 608534 units
Total distance driven: 625.977 km
Total distance with pupils on bus: 351.091 km
Number of vehicles : 12
12 large buses
0 medium buses
0 small buses
0 cars
Mean distance travelled: 14.3182 km
Mean waiting time: 24.1354 min
Max waiting time: 59 min
Mean lag time: 8.44886 min
Max lag time: 35 min
```

B.2 Arrival times

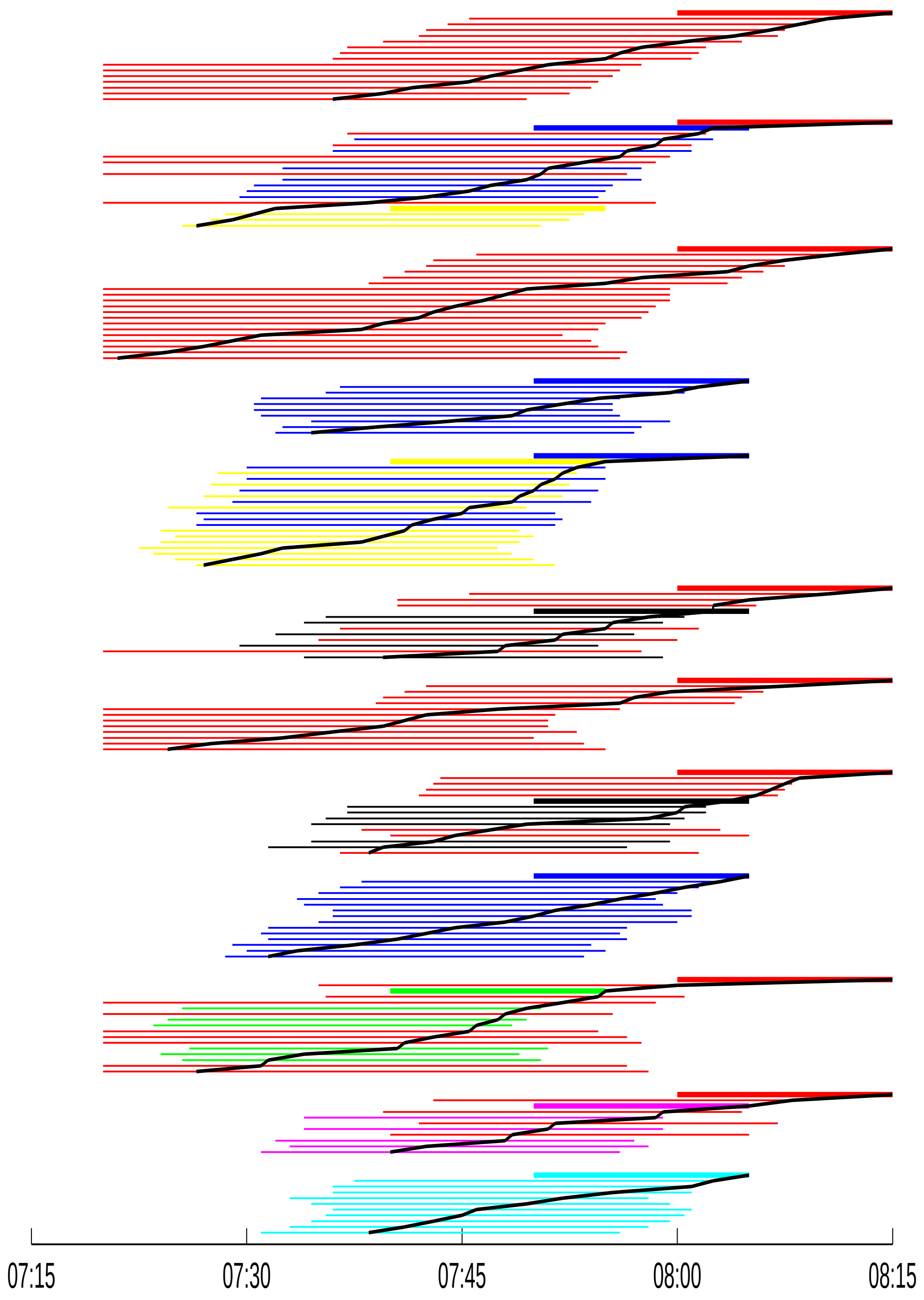
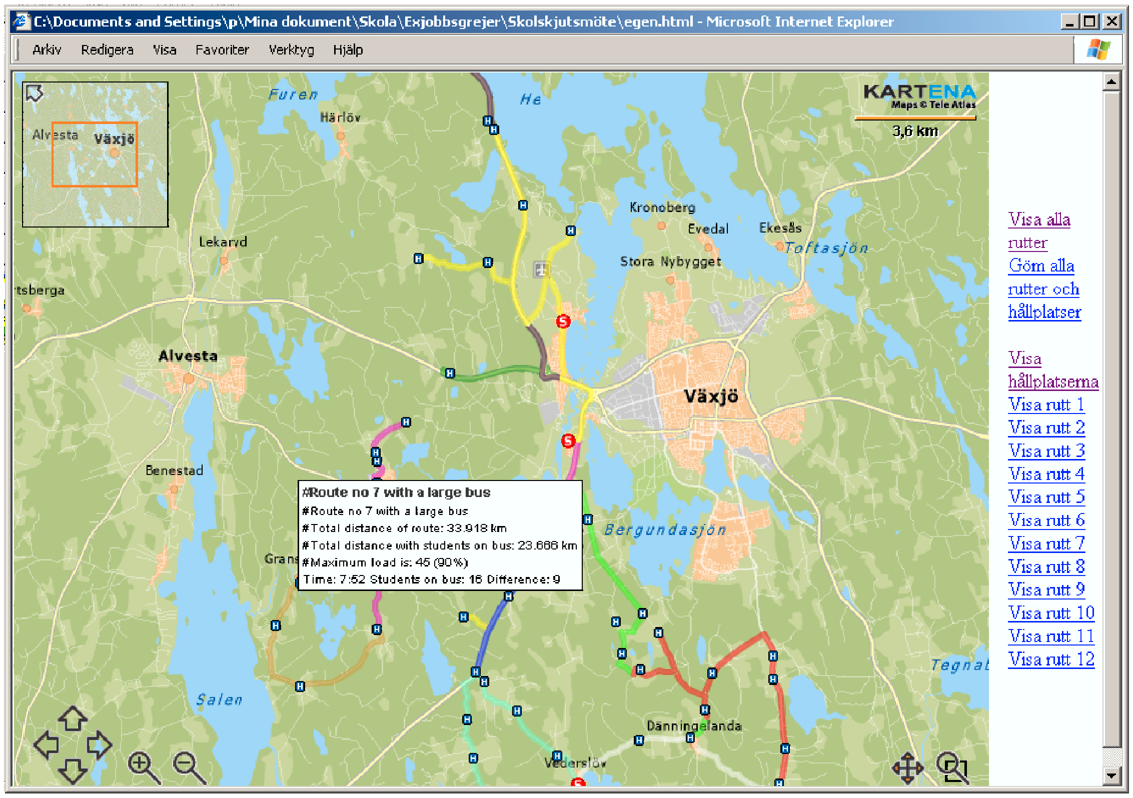


Figure 24: Arrival times in best solution ever found. Horizontal lines represents the time windows, thicker lines for schools. Different colors for pupils going to different schools. Intersection with the black line represents each arrival time.

B.3 Map with routes



```
#Route no 7 with a large bus
#Route no 7 with a large bus
#Total distance of route: 33.918 km
#Total distance with students on bus: 23.666 km
#Maximum load is: 45 (90%)
Time: 7:52 Students on bus: 16 Difference: 9
```

Figure 25: Map with all routes selected for the best solution. Statistics shown for one of the routes.

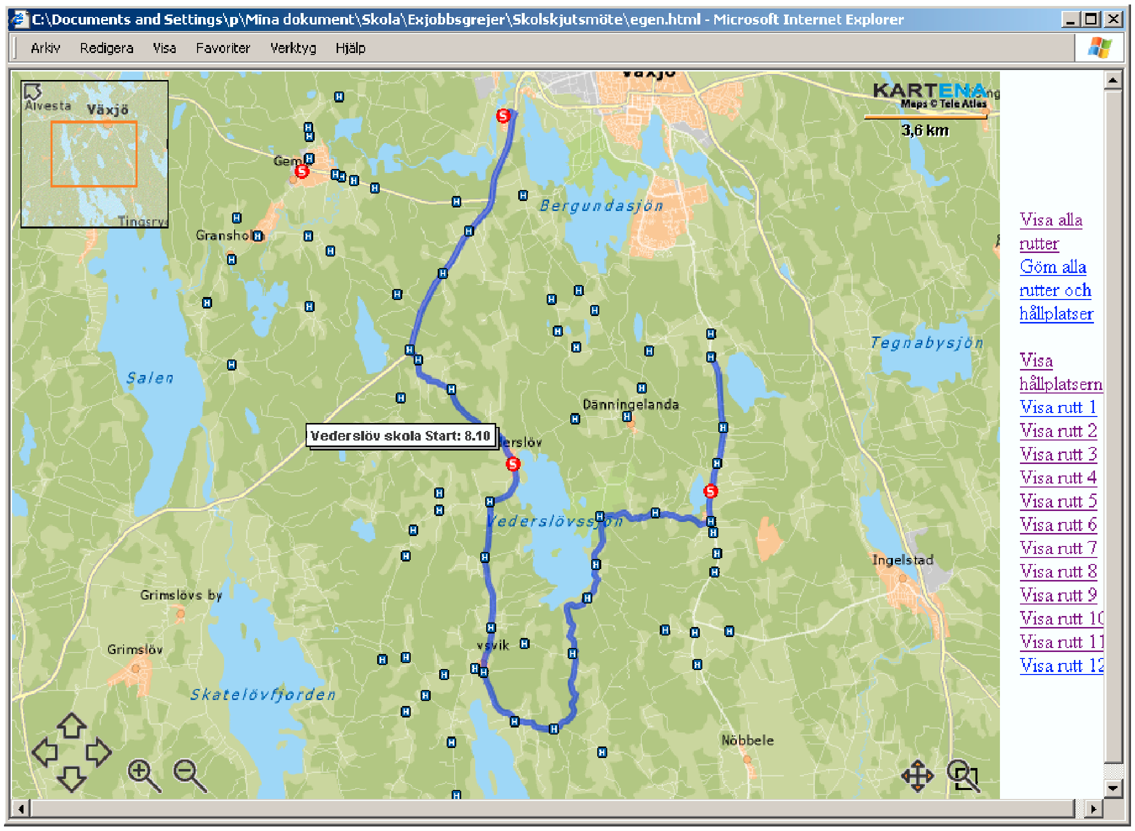


Figure 26: Map with just one route selected. Name and start time shown for one of the schools.

C Complete tables

C.1 Focus on distance with pupils on bus

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
381	858	368	13	28.4	60.0	12.7	48.5
384	883	369	15	29.2	59.5	13.5	46.0
391	884	377	14	28.5	58.0	12.9	46.0
382	830	369	13	28.7	58.0	13.0	48.5
372	844	358	14	25.6	60.0	9.9	50.5
382	860	368	13.8	28.1	59.1	12.4	47.9

Table 23: Quantitative data from the planning with focus on distance with pupils on bus. Mean values at the bottom. Using *TW1* rules.

C.2 Focus on the number of vehicles

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
1473	707	372	11	29.3	58.0	13.6	45.5
1695	761	395	13	28.0	59.5	12.3	45.5
1735	849	435	13	31.0	58.0	15.2	47.5
1511	771	411	11	30.5	59.5	14.8	49.0
1595	743	395	12	29.2	58.5	13.6	45.5
1602	766	402	12	29.6	58.7	13.9	46.6

Table 24: Quantitative data from the planning with focus on the number of vehicles. Mean values at the bottom. Using *TW1* rules.

C.3 Focus on both total distance and number of vehicles

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
649	672	366	13	25.1	56.5	9.5	31.5
666	682	369	14	23.4	56.5	7.7	26.0
692	723	381	14	25.2	59.5	9.5	34.5
691	724	378	14	25.4	57.5	9.8	34.0
694	718	369	15	23.0	56.5	7.4	31.5
678	704	373	14	24.4	57.3	8.8	31.5

Table 25: Quantitative data from the planning with focus on both total distance and number of vehicles. Mean values at the bottom. Using *TW3* rules.

C.4 2 hours running time

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
662	674	370	14	23.7	53.5	8.0	31.5
667	688	365	14	24.5	53.5	8.8	31.5
645	666	363	13	24.2	56.0	8.5	31.5
657	668	366	14	24.1	60.0	8.4	38.5
641	634	388	13	24.7	53.5	9.0	31.5
665	682	368	14	23.9	55.5	8.2	32.0
656	676	376	13	24.6	60.0	8.9	38.5
657	682	372	13	25.8	60.0	10.1	38.5
641	658	363	13	25.5	60.0	9.8	38.5
626	634	358	13	24.7	57.0	9.0	31.5
652	666	369	13.4	24.6	56.9	8.9	34.4

Table 26: 2 hours running time. Quantitative data from the planning with focus on both total distance and number of vehicles. Mean values at the bottom. Using *TW3* rules.

C.5 3 hours running time

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
651	661	360	14	23.7	57.0	8.0	32.0
660	675	366	14	23.8	60.0	8.1	38.5
636	653	360	13	25.1	59.0	9.4	35.0
660	680	360	14	23.6	60.0	7.9	38.5
646	666	367	13	25.0	59.5	9.3	38.0
644	645	364	14	23.0	53.5	7.4	31.5
619	636	362	12	26.2	60.0	10.5	38.5
642	665	359	13	24.3	55.5	8.6	31.5
662	678	367	14	23.5	53.5	7.8	31.5
614	628	360	12	25.2	58.5	9.5	38.0
644	659	363	13.3	24.3	57.7	8.5	35.3

Table 27: 3 hours running time. Quantitative data from the planning with focus on both total distance and number of vehicles. Mean values at the bottom. Using *TW3* rules.

C.6 5 hours running time

Cost	Dist.	Dist. w. p.	No. of veh.	Mean w. t.	Max w. t.	Mean l. t.	Max l. t.
644	661	347	14	23.4	54.0	7.7	32.0
630	641	360	13	24.4	58.5	8.7	38.0
615	639	351	12	25.9	55.5	10.2	31.5
614	630	358	12	24.8	60.0	9.1	38.5
640	654	367	13	25.1	60.0	9.4	38.5
648	663	372	13	25.1	55.0	9.4	31.5
662	681	363	14	23.5	56.0	7.8	31.5
652	672	372	13	24.8	55.5	9.1	32.5
636	642	369	13	24.2	59.0	8.5	37.5
609	626	351	12	24.1	59.0	8.4	35.0
635	651	363	12.9	24.5	57.3	8.8	34.7

Table 28: 5 hours running time. Quantitative data from the planning with focus on both total distance and number of vehicles. Mean values at the bottom. Using *TW3* rules.