Proceedings of the 8th International Conference on Applied Informatics
Eger, Hungary, January 27–30, 2010. Vol. 2. pp. 181–190.

A Flexible System for Optimizing Public Transportation*

Viktor Árgilán, János Balogh, József Békési, Balázs Dávid, Miklós Krész, Attila Tóth

Institute of Applied Sciences, Gyula Juhász Faculty of Education, University of Szeged

Abstract

In this paper we introduce an information system for optimizing public transportation. We integrated state-of-the-art results of the literature and new methods developed by our team. The main results of the research presented in this paper are not just the successful applications of these methods, but a new approach for system design in decision support of public transportation optimization processes.

Keywords: vehicle scheduling, vehicle assignment, driver scheduling, driver rostering, optimization, public transportation

MSC: 90B06, 90C08, 90C27, 90C90

1. Introduction

Scheduling problems arising in public transportation are quite complex tasks, thus a proper information system is needed to ensure effective logistic management. When planning such a system, every stage has to be analyzed to find the ones, where operational costs can be reduced. The general layout of such a system has the following elements: bus routing, timetabling, scheduling of the vehicles, and scheduling of the staff. All of these tasks are quite complex. Considering Hungarian aspects, bus routing is usually defined by the local city, but the same situation also applies to international cases. Timetabling is also defined by the local government, thus bus companies have minor influence in planning these stages. However, vehicle and stuff scheduling is completely managed by the bus company. Articles have been dealing with the optimization of both the vehicle and stuff scheduling problems for decades, and a number of models and methods have been published in both areas.

*This work was partially supported by Szeged City Bus Company (Tisza Volán ZRt., Urban Transport Division).

The complexity of the problem comes from the fact that all sub-problems are NP-hard. Because of this, exact solution of an instance taken from a middle-sized city (e.g. Szeged, 160,000 inhabitants, 2763 trips on an average weekday) is not possible.

This article presents an optimization system for public transportation, with the aim of applying the methods and results of scientific projects to real-life instances. To realize this, new achievements have been reached in the areas of system design, algorithm design and testing algorithm parameters on real-life data. These achievements are presented in Sections 2, 3 and 4.

2. The structure of our system

The problem mentioned in the introduction is usually split into two separate stages by previous studies. The first stage gives a feasible schedule of vehicles, which satisfies all trips of the fixed timetable. The second stage assigns suitable drivers to these schedules.

The models applied to the first stage can group vehicles into different sets, called depots. These depots can be defined by their geographical locations, or by certain vehicle-types that are needed to perform a trip (e.g. articulated, solo, etc.). The models can handle different costs for every depot, as well as the standard events like trips, deadheads (an empty trip connecting two different geographical locations). During our research project, other practical needs have arisen, which cannot be applied to these existing models. Aside from pull-in and pull-out trips, vehicle maintenance tasks and parking, refueling vehicles with different fuel types also became a major aspect. To handle all constraints given by these events, a second stage, vehicle assignment according to vehicle-types, was introduced, based on the results of the theoretical vehicle scheduling (first stage). Splitting the vehicle scheduling into two stages has certain practical advantages. The first stage gives an optimal theoretical solution for the vehicle schedule, which can be used as a good benchmark, but the resulting set of trips may not be executable. Thus it is solved by the separate vehicle assignment stage, the main advantage of which is that it is executed over a period of time (instead of giving a daily schedule), so rules applied to time periods can also be taken into consideration. The change in the resulting costs can also be compared to the theoretical schedule (and impact assessment of parameter changes can be calculated).

Driver scheduling has also been split into two stages, based on practical and project research reasons. In the third stage, a daily schedule of drivers is created, which takes daily rules and constraints into consideration. The fourth stage assigns the drivers to the previously constructed schedules over a planning period (practically two months), according to rules that are applied to drivers with respect to longer "time units" (consecutive days or weeks).

According to the previously mentioned aspects, the system was divided into four separate modules, to guarantee effectiveness: vehicle scheduling, vehicle assignment, driver scheduling, driver rostering. These modules give a well-structured

system, which can be integrated into a production-control subsystem.

The vehicle assignment and driver rostering modules are usually applied to a longer period of time (several weeks or months), given by special rules. The vehicle and driver scheduling modules have to be applied repeatedly to all days of the given time period.

Vehicle scheduling: this module solves the standard vehicle scheduling problem for each day of the given time period. The number of available vehicles gives constraints for all depots. Vehicles assigned to the same depot are considered identical regarding their standard and operational costs. Each trip has a defined set of depots, and only vehicles from those depots can execute that trip. Considering these data, the algorithm gives an optimal solution regarding all costs, given that all trips are executed. This module results in a set of vehicle schedules, where each schedule has an assigned depot. However, real vehicles are not assigned to the schedules yet, and events regarding vehicles (refueling, parking, servicing) are not included either.

Vehicle assignment: using the schedules given by the previous module, vehicle assignment gives us a real daily result for the input, which accomplishes all vehicle-dependent rules and constrains. This approach uses an assignment model, which matches the vehicles with the vehicle schedules. Vehicle-dependent events are inserted in the schedule during this process. The feasibility of the problem depends on the allowed vehicle-schedule assignment (both the schedule and the vehicle have to share the same depot), and the vehicle-specific events (e.g. the number of vehicles refueling parallel at a station cannot exceed the capacity of the station). The complexity of the problem comes from the simultaneous approach, but solving the two problems one after the other might result in an infeasible solution.

Driver scheduling: this module uses the vehicle schedules provided by the previous modules as an input. The aim is to partition these tasks into sets, each set representing a shift performed by a driver. These shifts have to fulfill all rules regarding drivers. The planning period is usually a day, so only rules applying for a single day have to be considered. The objective function usually minimizes the number of drivers, but costs of the assignment and other factors may apply as well.

Driver rostering: this module gets the shifts from the driver scheduling for each day of a longer planning period. The aim is to assign drivers to the shifts on the given days, taking into consideration rest days also. Rules regarding longer time-periods (weeks, months, etc.) have to be taken into consideration. The optimization not only minimizes the number of drivers, but overtime and short time as well.

3. Applied methods and algorithms

All the sub-problems mentioned above are NP-hard. Usually all are considered as separate problems in the literature. The rest of this chapter deals with the modules introduced earlier, presenting all methods have been used during their development, including both algorithms found in literature and the results of the

research projects.

3.1. Vehicle Scheduling

The aim of the vehicle scheduling problem is to execute all trips of a timetable with a given number of vehicles, while minimizing operational cost. The vehicles are assigned into depots, and the depots are given for each trip that can execute them. Geographical locations can also be connected by deadhead trips. The connections from depots to geographical locations are called pull in trips, and connections from geographical locations to depots are called pull out trips. The operational costs of the problem include the costs of all the applied vehicles, and costs depending on the distance covered by the vehicles.

Generally city bus companies have several depots (several vehicle-types), by which we need to apply a multiple depot vehicle scheduling problem (MDVSP). MDVSP problems were examined first by Bodin et al.[4], and proven to be NP-hard by Bertossi et al. [3].

A widely known and applied representation of this problem is the connection based network model [5]. This model represents all the connections of the problem described above as edges of the network, whereas the nodes are the fixed points of time. All constraints regarding the MDVSP can be applied to the connection based model, but the large amount of possible connections (mainly the deadhead edges) result in a problem which might not give a feasible solution in reasonable time. This issue is solved by the time-space network model.

The time-space network was introduced to public transportation by Kliewer et al. [7], and it is capable of solving larger MDVSP instances in practice. The model uses two dimensions, time and space. Space represents the geographical locations, while time gives so called timelines, which are introduced to all the geographical locations. The departing and arriving times of all the events are represented on the respective timelines, and give the edges of the network. The number of the connection edges (especially deadhead edges) is significantly reduced, as these can be aggregated into waiting edges, introduced on every timeline (see the method in [7]).

Our system applies this time space network to solve the vehicle scheduling problem. The result of the IP based on the model can be acquired using a solver, but still gives us an inadequate running time (finding the first feasible solution in some instances can take up to 2 hours). When planning for a longer period of time, companies can have several (like dozens) different day-types. Getting exact solutions for all of these problems would take a very long time, so this issue had to be dealt with in the system.

A number of heuristics have been introduced to decrease running time, from which the user can select the one best fitting his needs. Using the rounding heuristic based on Suhl et al. [10] can decrease running time by 40-50%, while the maximum gap from the optimal solution is only around 0,2% by our experiences. The variable fixing heuristic of Kliewer et al. [8], and an own approach to variable fixing, using

a greedy method can decrease running time by 85% and 95% respectively, while the maximum gap from the optimum remains around 1-1.5%.

In addition to these heuristics, the system is capable of solving multiple instances in parallel, thus further decreasing the running time needed to calculate results for all day-types. The number of parallel threads can also be set by the user. More information about the model and the applied method can be found in [2].

3.2. Vehicle assignment

Vehicle assignment matches vehicles with the vehicle schedules, and inserts all vehicle specific events into the schedules (like refueling and parking). The standard MDVSP models are not able to handle the constraints of the above problems, but solving these sub-problems is needed to acquire an optimal solution. For this, the module needs the vehicle schedules, data regarding the vehicles and the refueling stations (with all their attributes), and all parameters for other important vehicle-specific events (e.g. parking lots, parking rules, etc.)

The vehicle assignment problem is identical to the multi-dimensional assignment problem, so it is NP-hard. The problem uses the following model:

Every refueling station is open at a given time-period, and the time needed for refueling may vary depending on the vehicle. The problem uses the x_{ijkt} variable, which has a value of *1* if and only if schedule i, executed by vehicle j is being refueled at station k at time t (the value is *0* otherwise). However, a variable is generated if and only if it may be considered during the feasible solution. Such a condition is e.g. that the depots of schedule i and vehicle j are the same.

The IP based on the model contains only those variables, where x_{ijkt} exists. After solving the IP, we insert the parking events to the schedules. If the waiting between two events of a schedule exceeds the maximum amount of time that a vehicle can spend at any location, a parking event is inserted along with the appropriate deadhead trips. Inserting vehicle servicing and maintenance events can be inserted using a similar model as described. Further description of the method can be found in [1].

3.3. Driver scheduling

The complexity of the driver scheduling is arisen by the numerous human resource rules that are hard to formalize. There are rules specified by the law (like the maximum driving and resting time, etc.), and specific local rules, which may vary according to company and region (like handling breaks and resting times, special tasks, etc.). A good example for the complexity of a rule is the local rule for assigning breaks, which states that the length of a break must be between 15 and 30 minutes, and it can only be spent at a specified location. Moreover, if a shift has at least 6 working hours, a break has to be assigned before reaching 6 hours of work. If it has at least 9 working hours, then a second break has to be assigned

between the 6^{th} and 9^{th} hour, and if it has at least 10 working hours, the a third break has to be assigned between the 9^{th} and the 10^{th} hour.

Our method developed for the system is able to take all rules into consideration, and the parameters can be easily modified. To decrease the running time, the algorithm uses the result of the vehicle assignment as an input, and considers only the driving events, and travelling events between two geographical locations. All other events are inserted to the shifts after the optimization. The algorithm has the following main steps:

1. Generating work-pieces, which are ordered sequences of events. These events are assigned to the shifts, reducing the size of the problem significantly.
2. Generating parts of the schedules by using different strategies to create larger shifts. These shifts are effective on their own, and fulfill every rule.
3. Fitting the parts of the schedules together, using different methods. These ensure that the resulting full shifts do not violate any rules.
4. Finalization, where all complementary events are inserted in the schedule (passenger getting on and off, administration work, etc.), thus resulting in the final shifts.

During the optimization steps (2 and 3) estimated costs based on statistics are used, which take the structure of the shift into consideration. After Step 4, the real, payment-based costs can also be calculated for the shifts, as each event has a defined payment. The method is described in [9].

3.4. Driver rostering

The problem assigns drivers employed by the company to the set of shifts given by the driver scheduling module. Rules applying to a period of time have to be considered here (e.g. the amount of resting time between two consecutive shifts of a driver, the assignment of rest days, etc.). As the main problem is similar to set partitioning, the driver rostering is also NP-hard.

The boundary conditions of the driver rostering problem are not special ones for public transportation. The regulations for working multiple shifts are set by the law, and are very similar to e.g. call centers, nurses or officials. Because of this, methods applied in the area of staff rostering [6] give an appropriate result. As the optimization of our main problem is done by the previous modules, the system only needs to produce an executive solution in this phase. Considering the above aspects, an assignment is given based on the currently employed workers of the company, where parameters for the overtime and short time are considered.

In our case, a greedy method gives an adequate solution for the staff of the company. The optimization process not only minimizes the number of applied drivers, but also minimizes their overtime and short time. The drivers also have different types of contracts, which give different average daily working hours over a period of time.

4. Computational results

The system we have developed as the combination of the introduced modules was tested on real life data. The database used was a collection of instances from the daily routine of Szeged local bus company, Tisza Volán Zrt. Our results are presented in four different sub-sections, each corresponding to one of the four modules of the system.

Although the four modules are connected to each other, and each module uses the output of the previous ones as an input, their respective outputs can be examined and compared to the present practice, schedules "hand"-compiled by the transportation engineers of the company, using their decades of experience. The results of the four modules are presented and analyzed one after the other.

The scheduled days can be divided into 3 separate groups: workdays, Saturdays, and holidays. We present data of these 3 day-types (the company uses 14 day-types in practice, but all of these are similar to one of the 3 marked types, and this way not all 14 will be analyzed). The biggest challenge which requires the biggest cost is the scheduling of the workdays, as can be seen from the number of used vehicles and resources. A usual workday has 2763 trips to schedule, which the company presently satisfies with 107 vehicles and 162 drivers. Since Tisza Volán uses 4 different bus types as demand for the different trips, our problem instances apply 4 vehicle depots.

4.1. Results for the vehicle scheduling phase

The results of the first phase, the vehicle scheduling, show that an average of 10% decrease can be reached in the theoretical number of vehicles (see Table 1). Our

		Km	*Deadhead time*	*Deadhead km*	*Number of vehicles*
Workday	*Tisza Volán*	19914,3	37150	340,6	107
	Our solution	19914,3	23140	100,8	97
	Savings	0	14010	239,8	10
	Savings %		**37,71%**	**70,41%**	**9,35%**
Saturday	*Tisza Volán*	14594,4	19647	189	67
	Our solution	14594,4	12100	27,9	57
	Savings	0	7547	161,1	10
	Savings %		**38,41%**	**85,24%**	**14,93%**
Holiday	*Tisza Volán*	13270,7	18104	265,5	57
	Our solution	13270,7	12592	41,5	50
	Savings	0	5512	224	7
	Savings %		**30,45%**	**84,37%**	**12,28%**

Table 1: Computational results for the daily vehicle schedule

optimal method also shows a significant decrease in the total length of the deadhead trips, but we also have to consider that deadheads are usually 6-10% of the total daily travelled distance. Our deadhead savings are 30-38% of the original.

Our solution is nearly optimal, as we stopped the solver reaching the first feasible solution, but the gap of this value is at most 0,2% from the optimal. (It is

known from the prescribed stop condition of the executions.) This is also an important data for the company, which they can apply as a comparison of the optimum solution to the present practice.

The running time was not broken down into specific days, as the vehicle schedules are created for all days of the given period of time (usually 1-2 months). Giving all schedules for this time period takes about 1-2 hours, (with test runs on a standard, modern PC).

4.2. Results for the vehicle assignment phase

The most important result of our second phase is the fact that the vehicle assignment can be accomplished with the same number of vehicles as the corresponding vehicle schedule. This means, that the result received after the assignment is not worse, than the result of the theoretical schedule. The total covered distance of deadhead trips is about the same amount, as Tisza Volán has in practice. The running time of this module is around 1-2 hours for a time period of 1-2 months.

Table 2 summarizes the number of vehicles needed for the different day-types. While we showed only the number of vehicles in Table 1 (97 versus 107, 57 versus 67 and 50 versus 57 for the different day-types respectively), Table 2 also analyzes the vehicles according to their different fuel-types (diesel oil, natural gas).

The real cost could be decreased in the daily practice by choosing vehicles according to their costs, but all the vehicles of the company have to be used over a longer period of time. The amount of deadheads could also be decreased, if the number of vehicles were increased.

	Workday		Saturday		Holiday	
	Number of vehicles	Deadhead (km)	Number of vehicles	Deadhead (km)	Number of vehicles	Deadhead (km)
Diesel oil	74	1252,1	49	834,7	45	898,6
Natural gas	23	389,4	8	123,2	5	135,7
In total	**97**	**1641,5**	**57**	**957,9**	**50**	**1034,3**

Table 2: Computational results for the daily vehicle assignment

4.3. Results for the driver scheduling phase

The third phase is the driver scheduling, which can be given based on the results of the vehicle assignment. The schedules given by our vehicle modules result in "dense" shifts. Increasing the amount of used vehicles would give better results for the driver scheduling, and fewer number of vehicles give worse results. Different heuristic methods have been tested and compared to the earlier results of Tisza Volán. We used two different costs as comparison (see the two columns of Table 3). This means two different ways to calculate the costs. The first, so called statistical average, is based on the average cost of total length driven, and travelled distances of a driver. The second simply models the daily wage of the drivers. Both columns show the daily costs of all drivers.

The table shows that the costs of our methods based on the statistical average are much lower than that of Tisza Volán, while the second cost-type results in around the same cost values. The difference in the two types of costs is that the statistical average contains not only the costs based on wages, but other incidental costs also (statistically). The good results of the first column might come from the good structure of our schedules, while the second column considers wage only as the parameter for the total cost.

Running time can be several hours in this phase, as creating a daily optimization can take up to 5 minutes, depending on the method.

Workday		
Methods	**Statistical daily average**	**Daily wage**
Method #1	185 067	85 474
Method #2	205 091	87 602
Method #3	192 161	85 944
Method #4	189 591	84 890
Method #5	193 902	84 807
Method #6	189 038	85 135
Tisza Volán	258 650	84 588

Table 3: Computational results for the driver scheduling phase

4.4. Results for the driver rostering phase

Driver rostering assigns drivers for the driver schedules for every day of the planning period. While the third phase was creating daily schedules, driver rostering has to be applied for the whole planning period, and thus considers different types of rules (e.g. number and frequency of rest day). As different heuristics can be used to solve the problem with the present staff of the company, it is not detailed further.

5. Conclusions

The paper summarizes our achievements in the field of the optimization of public transportation. Our system divides the usual vehicle and driver scheduling optimization tasks into 2-2 sub-problems. We introduced the system, which consists of these 4 modules, and presented the different solution methods and algorithms applied to solve these modules. Section 4 summarizes our computational experiments for these modules, comparing the results with the present practice earlier used by the company. The advantage of our system is that the calculations in the different stages are based on real life data, which results in a nearly optimal solution. This gives a specific cost as a result for the bus company, which can be measured and compared to their earlier results.

References

[1] Balogh, J., Békési, J., Galambos, G., Krész, M., An Assignment Model for Real-World Vehicle Scheduling Problem with Refueling, *submitted for publication* (2010).

[2] Békési, J., Brodnik, A., Krész, M., Pas, D., An Integrated Framework for Bus Logistics Management: Case Studies. *Logistik Management* (2009), 389–411.

[3] Bertossi, A.A., Carraresi, P., Gallo, G., On Some Matching Problems Arising in Vehicle Scheduling Models. *Networks,* Vol. 17 (1987), 271–281.

[4] Bodin, L., Golden, B., Assad, A., Ball, M., Routing and Scheduling of Vehicles and Crews: The State of the Art, *Computers and Operations Researc*h Vol. 10 (1983), 63–212.

[5] Daduna, J.R., Paixão, J.M.P., Vehicle scheduling for public mass transit—an overview, In: J.R. Daduna et al., Editors, *Computer-aided transit scheduling, LNEMS* Vol. 430, Springer, StateplaceBerlin (1995), 76–90.

[6] Ernst A.T., Jiang, H., Krishnamoorthy,M., Sier, D., Staff scheduling and rostering: A review of applications, methods and models, *European Journal of Operational Research* Vol. 153 (2004), 3–27.

[7] Kliewer, N., Mellouli, T., Suhl, L., A time-space network based exact optimization model for multi-depot bus shceduling,. *European Journal of Operational Research* Vol. 175 (2006), 1616–1627.

[8] Kliewer, N., Mellouli, T., Suhl, L., Solving large multiple-depot multiple-vehicle-type bus scheduling problems in practice, *OR Spectrum*, 27 (2005), 507–523.

[9] Krész, M., Tóth, A., Driver scheduling by a flexible optimization process: a case study, *submitted for publication* (2010).

[10] Suhl, U. H., Friedrich, S., Waue, W., Progress in solving large scale multi-depot multi-vehicle-type bus scheduling problems with integer programming., *Wirtschaft-informatik Proceedings*, Paper 81 (2007).

Viktor Árgilán, János Balogh, József Békési,
Balázs Dávid, Miklós Krész, Attila Tóth
Institute of Applied Sciences
Gyula Juhász Faculty of Education
University of Szeged
6725 Szeged, Boldogasszony sgt. 6.
Hungary

e-mail: {gilan,balogh,bekesi,davidb,kresz,attila}@jgypk.u-szeged.hu