ARTIGO

VEHICLE SCHEDULING WITH NETWORK FLOW MODELS

Gustavo P. Silva
Departamento de Engenharia de Transportes
Escola Politécnica da Universidade de São Paulo

Raymond S. K. Kwan
School of Computer Studies
University of Leeds – United Kingdom

Nicolau D. Fares Gualda
Departamento de Engenharia de Transportes
Escola Politécnica da Universidade de São Paulo

RESUMO

Este trabalho retrata a primeira fase de uma pesquisa de doutorado voltada para a utilização de modelos de fluxo em redes para programação de veículos (de ônibus, em particular). A utilização de modelos deste tipo ainda é muito pouco explorada na literatura, principalmente pela dificuldade imposta pelo grande número de variáveis resultante. Neste trabalho são apresentadas formulações para tratamento do problema de programação de veículos associados a um único depósito (ou garagem) como problema de fluxo em redes, incluindo duas técnicas para reduzir o número de arcos na rede criada e, conseqüentemente, o número de variáveis a tratar. Uma destas técnicas de redução de arcos foi implementada e o problema de fluxo resultante foi direcionado para ser resolvido, nesta fase da pesquisa, por uma versão disponível do algoritmo Simplex para redes.

Problemas teste baseados em dados reais da cidade de Reading, UK, foram resolvidos com a utilização da formulação de fluxo em redes adotada, e os resultados comparados com aqueles obtidos pelo método heurístico BOOST, o qual tem sido largamente testado e comercializado pela School of Computer Studies da Universidade de Leeds, UK. Os resultados alcançados demonstram a possibilidade de tratamento de problemas reais com a técnica de redução de arcos.

ABSTRACT

This paper presents the successful results of a first phase of a doctoral research addressed to solving vehicle (bus, in particular) scheduling problems through network flow formulations. Network flow modeling for this kind of problem is a promising, but not a well explored approach, mainly because of the large number of variables related to number of arcs of real case networks. The paper presents and discusses some network flow formulations for the single depot bus vehicle scheduling problem, along with two techniques of arc reduction. One of these arc reduction techniques has been implemented and the underlying network flow problem addressed to be solved, in this phase of the research, using a code of the network Simplex algorithm.

Test problems based on real cases from the city of Reading, UK, were solved by the network flow approach and the results were compared with those provided by the BOOST system, which has been well tested and commercialised by the School of Computer Studies, University of Leeds, UK. The results show the feasibility of real case treatment with the adopted approach .

1. INTRODUCTION

The bus vehicle schedule problem consists of determining the minimum number of vehicles and sequencing each one, such that a predetermined set of trips can be covered with minimum total cost. Each trip has fixed starting and ending points, and also fixed starting and ending times. The vehicle must leave and return to the same point, i.e., to the same depot.

This problem has several variations depending on the number of depots, the number of vehicle types and the length of the trips. Each of these problems has been extensively studied and several different algorithms and heuristics exist. Daduna and Paixao (1995) describe the relationship among these problems and list a set of algorithms and their complexity. Carraresi and Gallo's (1984) is another relevant work, establishing correspondences between vehicle and crew scheduling problems and network flow models.

If there is more than one depot, the problem can be heuristically decomposed into single depot independent problems. Two alternative heuristic approaches are mentioned in Carraresi and Gallo (1984).

i) Cluster first – Schedule second. The set of trips is partitioned into subsets, one for each depot. Then for each subset a single depot problem is solved.

ii) Schedule first – Cluster second. In this case, a single depot problem on the whole set of trips is solved, yielding a set of vehicle duties. Then each vehicle duty is assigned to one of the depots in such way that the total cost is minimum.

In the same way, if there are different vehicle types and trips relying on a specific type of vehicle, each vehicle type and its scheduled trips can be formulated as an independent and single depot vehicle scheduling problem.

In the integrated vehicle and crew scheduling problem (Freling and Wagelmans, 1997 - Wren and Gualda,1997), the single depot vehicle scheduling problem has to be solved up to thousands of times. Thus, the vehicle scheduling with one depot and one vehicle type is the basic problem, which appears as a sub-problem in most of the other cases. This problem is denominated the *single depot vehicle scheduling problem* (SDVSP), which is the subject of interest in this paper.

The SDVSP can be formulated as a network flow problem in several different ways. The most important formulations presented in the literature are:

1. Transportation model
2. Assignment model
3. Matching model
4. Minimum cost network flow model

Paixão and Branco (1987) have shown the equivalence among the Transportation, Assignment and Matching models. In these approaches, the main problem is related to the number of variables. In graph theory the decision variables represent arcs in the network and have to be kept to a relatively small number, otherwise the problem becomes too huge to be solved. When the number of arcs is small, the problem can be solved to optimality in a relatively short time.

There are different techniques to reduce the number of arcs without loss of optimality. Bokinge and Hasselström (1980) proposed to discard trip links with a long waiting time, represented by *long arcs*. Long arcs are replaced by returning trips to depot through an arc previously created on the network.

A more general approach is the arc generation method, which is similar to the column generation for linear programming described in Freling et al. (1995). This technique makes use of the shadow price given by the dual variables, to decide whether an arc should be in the problem's domain. Ribeiro and Soumis (1994) applied this approach to solve the linear programme related to the multiple depot vehicle scheduling problem, providing a much tighter lower bound than those previously applied to the problem.

This paper aims to discuss different network representations for the Single Depot Vehicle Scheduling Problems, and possible techniques to reduce the number of arcs in these networks.

It also presents the results obtained by solving test problems adapted from real cases in England using:

a) the heuristic method BOOST proposed by Kwan and Rahin (1997).

b) the network reduction technique to eliminate the long arcs. The minimum cost flow problem is then solved for the reduced network using the network simplex algorithm, providing a vehicle scheduling solution.

The arc reduction technique based on the arc generation method has been studied and implemented. Its results are going to be presented in a forthcoming paper.

This paper is divided as follows: the second section presents the mathematical modelling of SDVSP and a brief overview of the algorithms found in the literature to solve it. The third section describes in more details the network representation of the problem, showing that it can be formulated and solved as a minimum cost flow problem. Section four presents two different techniques to reduce the number of arcs in the network representation. A brief description of the heuristic in the BOOST system is found in section five. The results from computational tests are presented in section six. Finally some conclusions are drawn in the last section.

2. THE VEHICLE SCHEDULING PROBLEM

The single depot vehicle scheduling problem (SDVSP) is defined as follow. Given a set of trips $T = \{1, 2, \ldots, n\}$, then each trip i is represented by:

i) sl_i the starting location,

ii) el_i the ending location,

iii) st_i the starting time and

iv) et_i the ending time.

For any pair of trips i and j, $dr(i, j)$ represents the travelling time between el_i and sl_j, which corresponds to a trip without passengers. These trips are called *deadrun trips*. Consider also a depot represented by its two

nodes, *s* and *t*, where *s* is the *departure depot* and *t* is the *arrival depot*; *dr(s, i) is* the travelling time from the depot to sl_i and *dr(i, t)* the travelling time from el_i to the depot.

A sequence (i, j) is a *feasible link* if and only if $et_i + dr(i, j) + \varepsilon \leq st_j$, where e is a prefixed tolerance parameter, introduced to take possible delays into account. The cost c_{ij} for the sequence (i, j) is given by

$$cij = \begin{cases} K_1.dr(i,j) + K_2.(\text{waiting time}) \text{ to start trip j, if (i,j) is a feasibly and} \\ \infty \text{ otherwise} \end{cases} \quad (01)$$

Where K_1 and K_2 are weights to the deadrun trips and the waiting time respectively, for which it is assumed the value 1. The capital cost is assigned to the first trip of each vehicle leaving the depot and it has the effect of minimising the number of vehicles.

Now consider the directed network $G = (N, A)$, with nodes $N = T \cup \{s, t\}$ and arcs $A = \{(i, j) \mid (i, j)$ is a feasibly link $\forall i, j \in N\}$. To illustrate this graph construction, let us take an example.

Table 1 contains 6 trips with its starting time, starting location, ending time and ending locations respectively. In order to simplify the understanding, consider the start location and the ending location as the same point. Figure 1 represents network *G* considering the set of trips shown in table 1.

Table 1
Set of 6 trips with starting and ending time and same starting and ending point.

Trip *i*	st_i	sl_i	et_i	el_i
1	07:00	Term A	08:00	Term A
2	07:00	Term A	08:30	Term A
3	08:00	Term A	09:00	Term A
4	08:00	Term A	10:00	Term A
5	09:00	Term A	11:00	Term A
6	09:30	Term A	10:00	Term A

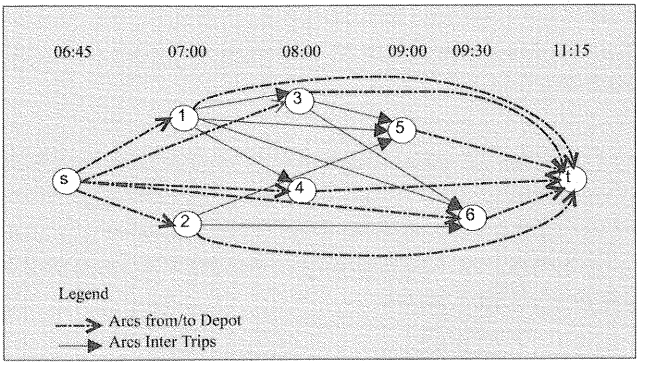

Figure 1
Network representation for the SDVSP presented in table 1.

It can be seen that after performing trip 1, the same bus can perform trips 3, 4, 5 or 6, because trip 1 finishes at 8:00 at terminus A while trips 3, 4 start at 8:00 and trips 5 and 6 start later on, at the same terminus. For this reason, there are arcs linking node 1 to nodes 3, 4, 5 and 6. The arcs linking node 1 to nodes 3, 4, 5 and 6 have cost 0, 0, 60 and 90 minutes respectively, representing the waiting time between trips. The arc from / to depot has the cost of travelling time from / to depot to / from Term A, in this case 15 minutes.

The vehicle scheduling problem consists of minimizing the capital cost plus the operational cost, with the constraint that each trip is carried out by only one vehicle. A vehicle scheduling is composed of vehicle *blocks*, where each block consists of a vehicle departing from the depot, performing a sequence of trips and returning to the depot.

This problem was formulated as a *transportation problem* by Gavish et al. (1978) and solved by applying the primal simplex algorithm. This formulation presents some difficulties related to the sparse cost matrix associated with the trip links.

Bokinge and Hasselström (1980) proposed the use of a *minimal cost flow formulation* to solve the problem as well as the long arc reduction method described in section 4.1.

Hoffstadt (1981) formulated the problem as an *assignment model*, which can be solved through the Hungarian algorithm. This algorithm requires updating the cost matrix and so it becomes necessary to store the full matrix, which may lead to a memory and computational problem when solving large scale problems.

The difficulties mentioned above are overcome in the *quasi-assignment model* presented by Paixão and Branco (1987). The authors adapted the Hungarian method to the problem so that only feasible links are considered. The *matching approach* is an extension from the assignment model, where each trip has its own start and end depots. This approach was used by Bertossi et al. (1987).

3. MINIMUM COST FLOW FORMULATION

The minimum cost flow formulation can be achieved once one trip i in G is represented by two nodes, i' for the beginning and i'' for the ending of the trip. An arc connecting both trip nodes (i', i'') is added to the network with cost null, and the lower and upper bounds equal to one. So it is ensured that each trip will be covered. There is an arc linking the ending node of trip i to the beginning node of trip j, (i'', j') with its cost c_{ij} as described in (1).

The departure depot is linked to each beginning of the trip node, and each end trip node is linked to the arrival depot. There is a return arc from the arrival depot node to the departure depot node, which ensures the circulation process. The return arc has the capital cost previously defined.

In this formulation the nodes do not have supply or demand, like in quasi assignment and transportation cases. Then, the SDVSP can be

solved by any minimum cost flow algorithm. In fact this is a flow circulation problem.

The network corresponding to this case is $G^{mc} = \{ N^{mc}, A^{mc}\}$, where $N^{mc} = \{ i', i'', s, t\}$ and $A^{mc} = \{ (i', i''), (s, i'), (i'', t), (i'', j'), \forall\, i, j \in N\}$. The cost c_{ij} is defined like above. Thus, the resulting model (M1) is:

$$\text{Min} \sum_{(i,j)\in A^{mc}} c_{ij} x_{ij} \qquad \text{subject to} \qquad (2)$$

$$\sum_{i\in N^{mc}} x_{ij} - \sum_{i\in N^{mc}} x_{ji} = 0 \qquad \forall\, j \in N^{mc} \qquad (3)$$

$$x_{ij} \in \{0,1\} \qquad \forall\, (i, j) \in A^{mc} \qquad (4)$$

Where $x_{ij} = 1$ if trip j is performed just after trip i and $x_{ij} = 0$ otherwise.

Expression (2) minimises the total cost, constraint (3) assures the flow conservation principle while constraint (4) assures the zero / one value to the flow in the arcs on G^{mc}.

It can be taken advantage of this formulation introducing intermediate depot nodes (see figure 2). It is possible to generate of a vehicle block in which the vehicle performs a chain of trips, has a long break at the depot, then perform another chain of trips.

The network built to represent this idea begins with a departure depot node s_i and an arrival depot node t_i for each trip i. The depot nodes are sorted according to the increasing time without considering if the node is a departure or an arrival node.

The depot nodes are linked by arcs from each depot node to the next one with cost zero, minimum flow of zero and maximum flow of n. The return arc links the last depot node to the first depot node with capital cost K, minimum flow of zero and maximum flow of n. The return arc ensures the circulation flow and counts the number of vehicle necessary to cover all the trips.

It is possible to reduce this network by grouping the depot nodes. This can be done by ordering the departure and arrival depot nodes in an increasing order. A new group starts when an arrival depot node is encountered. Each group of nodes is replaced by one node, and arcs incident to and from the nodes in the group are now from and to the new node. This process ensures that a vehicle leaving a depot node has already arrived at a previous depot node.

When this formulation is solved it can be extracted both, the operational and the capital costs directly from the optimal solution. The operational cost is given in terms of deadrun trip time plus waiting time, while the capital cost represents the number of vehicles scheduled. Besides, the solution provides the whole block for each bus.

This feature does not appear in the assignment formulation. In that case the optimal solution for the formulation gives the minimum operational cost and the blocks of duty. These blocks must be combined to obtain the minimum number of vehicles. It means that the problem is solved in two phases.

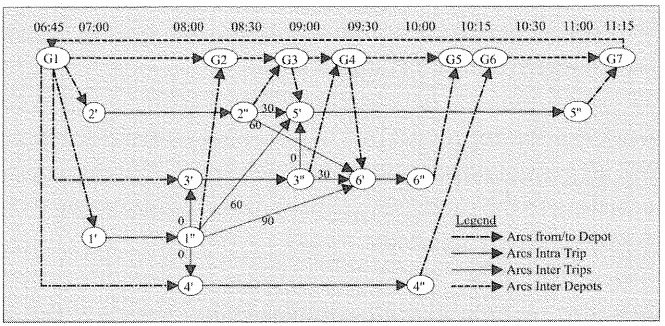

Figure 2
Network G^{mc} for the trips in table 1.

The network in figure 2 is an extension of the one in figure 1, where each node i is now represented by two nodes i' and i'' and the depots are distributed along the timescale.

4. ARC REDUCTION TECHNIQUES

The practical difficulty of the approach described above is that the network built to represent the SDVSP may have a large number of arcs. This kind of problem belongs to the class of NP-hard problems, whose number of arcs increases in an exponential rate, in accordance with the number of nodes becoming impossible to be solved for large scale problems. So one has to discard arcs without losing optimality for the problem. A way to reduce the number of arcs is to eliminate those which will certainly not be in the optimal solution.

This can be done by replacing arcs with higher cost through a pass to the depot before going to perform the next trip. It is easy to see that once there are already arcs from the depot to each trip and from each trip to the depot, the number of arcs will decrease.

Another strategy is to solve a sequence of intermediate problems, where the dual price for the arcs not in the previous domain are found out. If there are arcs that can improve the value of the objective function, these arcs are included in the next domain problem. This process is repeated until no arc with that feature can be found. Although this technique has a sequence of optimisation problems to be solved, usually it is not necessary to solve too many of them.

4.1 Long arcs elimination

Considering the network G defined in section 2, let us define the *long arcs set* A^l as a subset of A whose arcs cost satisfies the following.

$$(i, j) \in A^l <=> c_{ij} \geq dr(i, t) + minimum\ depot\ time + dr(s, j), \qquad (5)$$

where the *minimum depot time* is a parameter to be defined by the operator. In practice, such arcs are less used than the short arcs represented by the set A^s. We can define $A^s = A\text{-}A^l$.

Instead of waiting at the terminus for the next trip, operators prefer to send the vehicle to the depot, even with higher operation cost. This return allows a vehicle tanking, a driver meal break and other little time consuming activities concerning the vehicle and / or drivers. Hence, the long arcs can be deleted and considered as a return to the depot.

Observing the network in Figure 2 and considering all depot arcs with cost of 12 minutes and the minimum depot time as 30 minutes, the long arcs are: (1'', 5'), (1'', 6') and (2'', 6'), because they satisfy (5). Then, these arcs can be deleted from network without loss of optimality.

4.2 Arc generation approach

The arc generation approach considers only arcs between trips. Bearing in mind the minimum cost flow formulation for the problem presented in section 3, let us define the node sets N' which corresponds to the beginning trip nodes and the set N'' which corresponds to the ending trip nodes. More over, define $P = \{s\} \cup N''$ and $Q = N' \cup \{t\}$. The reason for this way of grouping nodes is that the source depot node and the ending trip nodes provide vehicles, while the beginning trip nodes and the sink depot node consume vehicles.

Now it can defined the bipartite network $G^{qa} = (P, Q, A^{qa})$ where $A^{qa} = \{(i'', j') \forall i, j \in N\} \cup (s \times N') \cup (N'' \times t) \cup (s, t)$. The cost $c_{i''j'}$ are the same as c_{ij} following the definition in (1), and the arc (s, t) has cost zero. The figure 3 is a sample of network G^{qa} for the trips in table 1.

Each node form network in figure 3 has its amount of supply or demand between braces. For example, node s supplies six buses {-6} and node 1' has a consumption of one bus {1}.

When the problem represented in figure 3 is solved, it must be considered a dummy link (i', i'') for each trip i. Thus, the paths from s to t are the blocks of work, which must be combined to produce the scheduling of the vehicles.

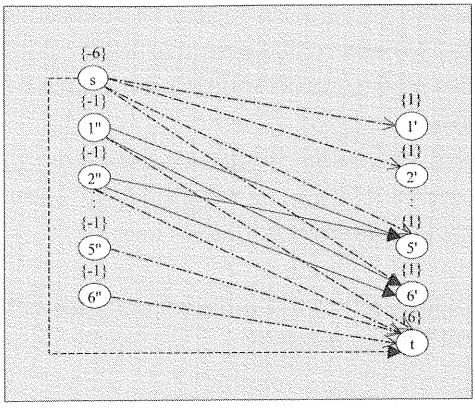

Figure 3
Assignment network representation for the SDVSP in table 1, through a bipartite graph.

In this network, the cost for the arcs from / to the depot are in fact, the travelling time between the depot and the trip and the capital cost is put on the arc from *s* to *t*.

The solution from this problem generates another problem of linking the block, which can be solved again in the same way, but considering the cost for the arcs from / to depot to each block as the capital cost.

The quasi assignment formulation for network in figure 3 (M2) used in the arc generation approach is given by:

$$\text{Min} \sum_{(i,j)\in A^{qa}} c_{ij}x_{ij} \qquad \text{subject to} \tag{6}$$

$$\sum_{j\in Q} x_{ij} = -1 \qquad \forall\, i \in N'' \tag{7}$$

$$\sum_{i\in P} x_{ij} = 1 \qquad \forall\, j \in N' \tag{8}$$

$$\sum_{j\in Q} x_{sj} = -n \tag{9}$$

$$\sum_{i\in P} x_{it} = n \tag{10}$$

$$x_{ij} \in \{0,1\} \qquad \forall\, (i,j) \in A^{mc} \tag{11}$$

The variable x_{ij} and expression (6) has the same meaning as in (M1). Constraints (7) and (8) assure that each trip will be performed by only one vehicle, while constraints (9) and (10) are related to the minimum number of vehicles.

The arcs in network G^{qa} can be divided into two sets, i. e., $A = A^b \cup A^n$, where A^b and A^n can be A^{short} and A^{long} respectively. Now we can outline the steps of the arc generation approach.

1. Solve the problem (M2) to the reduced network $G^{qa} = (N, A^b)$. The resulting dual profit-price vector π.

2. Calculate the reduced costs for each arc $(i, j) \in A^n$ and a given parameter ε:

$$\bar{c}_{ij} = c_{ij} - \pi(i) + \pi(j) + \varepsilon$$

3. For each node $i \in N$, if $\bar{c}_{ij} < 0$ then
   Add arc (i, j) to A^b and
   Delete arc (i, j) from A^n.
   If some arc was generated, return to 1.
   Else stop.

This approach has been used to solve large-scale and special structured linear programming problems, such as vehicle and crew scheduling problems. More details can be found in Desrochers and Soumis (1989), Ribeiro and Soumis (1994) and Fores et al. (1997).

5. THE BOOST HEURISTIC METHOD

The BOOST (which stands for "Basis for Object Oriented Scheduling of Transport") bus scheduling algorithm belongs to the class of heuristics generally known as "2-opt". BOOST has been developed in recent years using a fresh object oriented approach. A much earlier development with the same 2-opt heuristic basis used the traditional procedural approach (see Wren 1972, Smith and Wren 1981), which has shown to be highly unsatisfactory for long-term continuous development. BOOST has achieved significant improvement in terms of schedule quality and

computational speed, and it forms a sound basis for further extensions to cater for complex scheduling requirements. Since October 1997 BOOST has been incorporated into a new bus scheduling package called Openbus, which also includes driver scheduling and rostering modules. BOOST has been tested using data from various bus companies in the past, and used in some recent feasibility studies involving bus companies in the UK and Brazil. One UK bus company, with its main depot in the city centre, has recently been relocated to a new site and they have used BOOST to re-schedule their entire new operation.

BOOST first forms a rough initial schedule using a given target number of buses. The target is either estimated by an automatic process or specified by the user. It is possible that some of the links in the initial schedule are time infeasible, i.e. the bus arrives later than it is due to depart. The schedule is then iteratively refined. In each iteration, a selected pair of links is broken up and re-linked in the alternative way if the overall infeasibility and / or cost of the schedule is reduced. At the end BOOST returns either the lowest cost feasible solution it has found, or if the target number of buses is too low pointers to critical bus trips that might be retimed to make the target bus number feasible.

Turning the above outline heuristic into a good practicable algorithm is non-trivial. Naive designs would lead to endless iterations and not arriving at the best solution. Object orientation has enabled the heuristic model to be articulated at appropriate levels of abstraction and specialisation. As a result, many alternative algorithmic designs could be developed, tested and compared quickly.

6. COMPUTATIONAL TESTS

Some computational tests have been performed to compare the quality of solutions provided by the Network Simplex Algorithm and the Heuristic Method BOOST.

Based on the timetable containing the same type of information found in table 1 and the dearun matrix $dr(i, j)$, the corresponding network

representation was built. Applying the long arc elimination and the depot node grouping, the network had the number of arcs reduced without loss of optimality. Over this network, the circulation problem described in section 3 was solved as a minimum cost flow problem through the specialized strongly feasible simplex method (Ahuja et al., 1993). The optimal bus schedule was built from the solution of this problem. This solution was compared with those obtained by the BOOST system.

6.1 The data characteristics

The tests were run using data adapted from real data from the city of Reading (UK), considering a sequence of routes such that the number of trips and the number of buses and trips increased progressively.

When BOOST is forced to use less than the minimum number of buses, it would return the "best" infeasible solution. As the exact methods do not allow infeasible solutions, we forced BOOST to generate only feasible solutions using a suitable target number of buses.

6.2 Results

In order to simplify the presentation of the results in the table 2, MCF stands for the Minimum Cost Flow algorithm. Notice that in the last test problem the CPU time limit for the MCF available version was exceeded.

Table 2
Results from seven test problems using MCF and BOOST methods.

Number of Routes	Total of Trips	Minimum of Vehicles	Solution	
			MCF	BOOST
1	50	4	529	529
2	120	9	916	916
3	187	12	1094	1094
4	208	14	1304	1304
6	231	19	1593	1593
9	368	27	2462	2462
13	859	49	------	4412

7. CONCLUSIONS

The performed tests have shown the potential of the heuristics methods, which achieved the optimal solution in all analysed cases. Moreover, it was possible to solve bigger problems with the BOOST than those solved by MCF. Unfortunately, the available version of the MCF algorithm cannot solve larger problems, for which probably the heuristic methods would not achieve the optimal solution. In fact, the performance of the MCF algorithm can be reasonably efficient, depending on the algorithm code. Silva (92) has solved the maximum flow problem through the same simplex algorithm for networks with up to 3,000 nodes and 200,000 arcs, in a similar machine.

Instead of devoting efforts to improve the present code of the MCF algorithm, the research is now devoted to replace it by a developed version of the Out-of-Kilter algorithm, which shall eliminate the CPU time limits imposed by the present MCF code, and to develop a heuristic to be embedded in the arc reduction technique based on the arc generation approach.

Anyhow, this phase of the research has provided a broader understanding of the problem and of the difficulties to solve the SDVSP through network flow algorithms. A computational approach to solve the basic vehicle scheduling problem making use of a network flow algorithm was achieved. The quality of the solution provided by this approach matches that given by BOOST. The success of the adopted approach have shown the feasibility of modeling real world vehicle scheduling problems with network flow models.

ACKNOWLEDGEMENTS

The authors are grateful to the British Council and CAPES for financing the academic exchange, which has been making this work possible. The authors wish to thank Prof. Anthony Wren and Dr. Les Proll from the School of Computer Studies, University of Leeds, UK; and Dr. Claudio Barbieri da Cunha, from the Departamento de Engenharia de Transportes, Escola Politécnica, Universidade de São Paulo, Brasil, for their useful suggestions,

comments and assistance. And to FAPESP, for providing support for the presentation of this work at XII ANPET.

REFERENCES

AHUJA, R. K., MAGNANTI, T. L., ORLIN, J. B. [1993] Network Flows: Theory, Algorithms and Aplications. *Prentice-Hall, Inc.*, Englewood Cliffs, New Jersey, USA.

BERTOSSI, A. A., CARRARESI, P. , GALLO, G. [1987] On some matching problems arising in vehicle scheduling models, *Networks* 19:531-548.

BOKINGE, U., HASSELSTRÖM, D. [1980] Improved vehicle scheduling in public transport through systematic changes in the time-table. *European Journal of Operations Research*, 5:388-395.

CARRARESI, P., GALLO, G. [1984] Network models for vehicle and crew scheduling. *European Journal of Operational Research*, 16:139-151.

DADUNA, J. R., PAIXÃO, J. M. P. [1995] Vehicle scheduling for public mass transport- an overview. In *Daduna, J. R.; I. Branco, and J. M. P. Paixão (eds.) Computer-Aided Transit Scheduling, Lectures Notes in Economics and Mathematical Systems.* Springer Verlag, Berlin, Alemanha.

DESROCHERS, M, SOUMIS, F. [1989] A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23:1-13

FORES, S., PROLL, L., WREN, A. [1997] An improved ILP system for driver scheduling. In *7th International Workshop on Computer-Aided Scheduling of Public Transport*, Boston, August 1997. Pp 507-526.

FRELING, R.; PAIXÃO, J. M. P., WAGELMANS, A. P. M. [1995] Models and Algorithms for Vehicle Scheduling. *Aviable via WWW at http://kaa.cs.few.eur.nl/few/people/freling/.*

FRELING, R., WAGELMANS, A. P. M. [1997] Integration of Vehicle and Crew Scheduling with Application to Bus and Driver Scheduling in Rotterdam. In *7th International Workshop on Computer-Aided Scheduling of Public Transport*, Boston, USA.

GAVISH, B., SCHWEITZER, P., SHLIFER, E. [1978] Assigning buses to schedules in a metropolitan area. *Computers and Operations Research*, 5:129-134.

HOFFSTADT, J. [1981] Computerized vehicle and driver scheduling for the Hamburger Hochbahn Artieingesellschaft. *In:* Wren A. *(ed.) Computer Scheduling of Public Transport.* North-Holland, Amsterdam, Netherlands.

KWAN, R. K., RAHIN, M. A. [1997] Object orientede bus scheduling - the BOOST system. In *7th International Workshop on Computer-Aided Scheduling of Public Transport*, Boston, USA.

PAIXÃO, J. M. P., BRANCO, L. [1987] A quasi-assignment algorithm for bus-scheduling, *Networks*, 17:249-269.

RIBEIRO, C., SOUMIS, F. [1994] A column genaration approach to the multiple depot vehicle scheduling problem, *Operations Research*, 42:41-52.

SILVA, G. P. [1992] Um estudo do algoritmo de Goldberg e Tarjan para o problema do fluxo máximo. *MSc Dissertation*, Universidade Estadual de Campinas, Brazil.

SMITH, B. M., WREN, A. [1981] VAMPIRES and TASC: two successfully applied bus scheduling programs. In: *Wren A. (ed.) Computer Scheduling of Public Transport. North-Holland*, Amsterdam, Netherlands.

WREN, A. [1972] Bus scheduling: and interactive computer method. *Transportation Planning and Technology*, 1: 115-122.

WREN, A., GUALDA, N. D. F. [1997] Integrated scheduling of buses and drivers. In *7th International Workshop on Computer-Aided Scheduling of Public Transport*, 53-75, Boston, USA.