Konrad-Zuse-Zentrum für Informationstechnik Berlin
Takustraße 7, D-14195 Berlin-Dahlem, Germany

Andreas Kokott Andreas Löbel

Lagrangean Relaxations and Subgradient Methods for Multiple-Depot Vehicle Scheduling Problems

Preprint SC 96–22 (December 1996)

Lagrangean Relaxations and Subgradient Methods for Multiple-Depot Vehicle Scheduling Problems*

Andreas Kokott** Andreas Löbel**

Abstract

This paper presents two Lagrangean relaxation approaches for the $\mathcal{NP}$-hard multiple-depot vehicle scheduling problem in public mass transit and reports on computational investigations. Our Lagrangean relaxation approaches can be applied to generate very tight lower bounds and to compute feasible solutions efficiently. A further application is to use the Lagrangean relaxations as new pricing strategies for a delayed column generation of a branch-and-cut approach.

The computational investigations are based on real-world test sets from the cities of Berlin and Hamburg having up to 25 thousand timetabled trips and 70 million dead-head trips.

Mathematics Subject Classification (1991): 90B06, 90B10, 90C06, 49M29.

1 Introduction

Vehicle scheduling is an important part of the planning process in public transportation. In a current project, see [8], we investigate the Multiple-Depot Vehicle Scheduling Problem (MDVSP) arising at large German public transportation companies. It is known that the MDVSP is $\mathcal{NP}$-hard, see [14, 2].

Besides requiring integrality of the solution, the MDVSP is described by two types of constraints: The **flow conditions** ensuring that each timetabled trip has to be serviced exactly once, and the **flow conservations** ensuring that each vehicle continues its duty until it reaches its destination. Hence two Lagrangean relaxation approaches are self-suggesting. First, a relaxation of the flow conditions resulting in a Lagrangean function that decomposes into independently solvable minimum-cost flow subproblems, e. g., see [8, 2]. Second, a relaxation of the flow conservations resulting in a Lagrangean function that is simply a polynomially solvable single-depot instance, e. g., see [8, 3, 15, 11].

From our point of view, these two Lagrangean relaxation approaches have the following interesting features: First, combined with a subgradient method, we can compute tight lower bounds for real-world problems of any size efficiently. This is the topic of our article.

*This work has been supported by the German Federal Ministry of Education, Science, Research, and Technology grant no. 03-GR7ZIB -7. Responsibility for the contents lies with the authors.

**Konrad-Zuse-Zentrum für Informationstechnik Berlin (ZIB), Takustraße 7, D-14195 Berlin, Germany, URL: `http://www.zib.de/`

Second, a popular approach to generate feasible solutions of large-scale MDVSPs are schedule first – cluster second heuristics, e. g., see [8, 6, 5]. Sometimes, a reschedule step follows the clustering part where for each cluster (i. e., single depot) the timetabled trips are optimally rearranged, see [8, 6]. The Lagrangean relaxation with respect to the flow conservations together with some primal (interchange and/or greedy) heuristic can be viewed as a schedule first – cluster second approach. For instance, [8, 3, 15, 11, 2] discuss such approaches for the MDVSP. Similarly, the Lagrangean relaxation with respect to the flow conditions together with some heuristic can be viewed as a cluster first – schedule second approach, e. g., see [2].

Third, the Lagrangean relaxations described in this paper are utilized in [14, 13, 8] as new column generation techniques for a delayed column generation to solve the linear programming (LP) relaxations of the MDVSP. With these new techniques, LP relaxations of truely large-scale MDVSP instances can be solved to optimality. Combined with some LP-based heuristics within a branch-and-cut algorithm, all our realistic problem instances can be solved to optimality or, at least, can be approximated with a relatively small gap.

Our partners are the IVU GmbH and Berliner Verkehrsbetriebe (BVG), Berlin, and the HanseCom GmbH and Hamburger Hochbahn AG, Hamburg. They provided us with real-world data from the cities of Berlin and Hamburg.

In the following, we assume the reader to be familiar with network flow, integer linear programming, Lagrangean relaxations, and subgradient methods, see [1, 16, 9].

2 The Multiple-Depot Vehicle Scheduling Problem

2.1 Problem Description

Details of the techniques to model vehicle scheduling problems in public transportation can be found, e. g., in [14] or [8]. Here is a brief summary of our version of the MDVSP. We assume that the lines and their frequencies of service are given, i. e., a timetable has been already determined. According to its operational interests, the vehicles of a public transportation company are divided into so-called **depots**, denoted by $\mathcal{D}$, each representing a set of vehicles that do not need to be distinguished. With each depot $d \in \mathcal{D}$ we associate a start point d^+ and an end point d^- where its vehicles start and terminate their daily duty. Let $\mathcal{D}^+ := \{d^+ | \, d \in \mathcal{D}\}$ and $\mathcal{D}^- := \{d^- | \, d \in \mathcal{D}\}$. The number of available vehicles, called **depot capacity**, of each depot d is denoted by κ_d.

A given timetable defines a set of so-called **timetabled trips**, denoted by $\mathcal{T}$, that are used to carry passengers. We associate with each $t \in \mathcal{T}$ a first stop t^-, a last stop t^+, a departure time s_t, an arrival time e_t, and a so-called **depot-group** $G(t) \subseteq \mathcal{D}$. Each $G(t)$ includes those depots whose vehicles are allowed and able to service trip t. Let $\mathcal{T}^- := \{t^- | \, t \in \mathcal{T}\}$, $\mathcal{T}^+ := \{t^+ | \, t \in \mathcal{T}\}$, and $\mathcal{T}_d := \{t \in \mathcal{T} | \, d \in G(t)\}$.

There are further types of trips, all running without passengers: A **pull-out trip** connecting some start point d^+ with some first stop t^-, a **pull-in trip** connecting some last stop t^+ with some end point d^-, and a **dead-head trip** connecting some last stop

t^+ with some succeeding first stop t'^-. For notational simplicity, we call these trips all **unloaded trips**. Let $\Delta_{t,t'}$ denote the duration of the dead-head trip from t^+ to t'^- plus some layover time. Whenever $e_t + \Delta_{t,t'} \leqslant s_{t'}$, we call the corresponding dead-head trip to be **compatible**.

Many MDVSP models use a definition of "compatible" restricting the possible dead-head trips, e.g., [5] consider a dead-head trip to be compatible if its duration, called **turning time**, does not exceed a maximal predefined value (of about 90 to 120 minutes). However, such a model can provide only suboptimal solutions especially concerning the minimal number of vehicles ([14] gives a small counterexample with two depots and four timetabled trips). We do not use such a restriction, the duration of our dead-head trips can be as large as possible. Actually, our partners provide us only with dead-head trips having a maximal predefined turning time. However, we create depot-wise "long-time" dead-head trips whenever it is possible to service two timetabled trips t and t' in sequence by the same vehicle. This is always the case if it is possible to use the pull-in trip from the last stop of t back to the depot, staying in the depot for a certain time, and using later the pull-out trip to the first stop of t'.

A **vehicle schedule** is a chain of trips such that the first trip is a pull-out trip, the last trip is a pull-in trip, and the timetabled trips and unloaded trips occur alternately; a vehicle schedule is called valid if all its trips belong to the same depot. If no dead-head trip stands for both a pull-in and pull-out trip, the vehicle schedule is also called **block**.

The union operator $\dot\cup$ is considered as a disjoint union. We introduce the following sets of trips for each depot $d \in \mathcal{D}$: $A_d^{\text{t-trip}} := \{(t^-, t^+) \mid t \in \mathcal{T}_d\}$ and

$$A_d^{\text{u-trip}} := \{(d^+, t^-), (t^+, d^-) \mid t \in \mathcal{T}_d\} \cup \dot{\bigcup_{\substack{\text{compatible dead-head} \\ \text{trips of depot } d \\ \text{connecting } p \text{ with } q}}} \{(p^+, q^-)\}.$$

With each unloaded trip $a \in A_d^{\text{u-trip}}$, we associate a weight $c_a^d \in \mathbb{Q}$ representing its operational or capital costs or a mixture of both. Generally, the weight of each unloaded trip stands always for its operational costs. If the main objective is a fleet size minimization, however, the weight of each pull-out trip is increased by a sufficiently large **big M** standing for the capital costs and being larger than the operational costs of any set of vehicle schedules covering each timetabled trip exactly once. The minimization of this "two-stage" objective function would first minimize the fleet size and secondary minimize the operational costs (among all fleet minimal solutions). With this terminology, the MDVSP is to find a weight minimal set of feasible vehicle schedules such that each timetabled trip is covered by exactly one vehicle schedule.

2.2 Mathematical Model

We will now state an integer programming model for the MDVSP that is based on an integer multicommodity flow formulation. For each depot $d \in \mathcal{D}$, let (d^-, d^+) denote an additional **backward arc** (on which depot capacities can be controlled) and let $A_d := A_d^{\text{t-trip}} \cup A_d^{\text{u-trip}} \cup \{(d^-, d^+)\}$. We define a digraph $D = (V, A)$ with node set $V := \mathcal{D}^+ \cup$

$\mathcal{D}^- \cup \mathcal{T}^- \cup \mathcal{T}^+$ and arc set $A := \dot{\bigcup}_{d\in\mathcal{D}} A_d$. Figure 1 gives an illustration of D for a small example with $\mathcal{D} = \{\mathrm{r,g}\}$, $\mathcal{T} = \{\mathrm{a,b,c,d,e}\}$, $\mathcal{T}_\mathrm{r} = \{\mathrm{a,c,d}\}$, and $\mathcal{T}_\mathrm{g} = \{\mathrm{a,b,c,e}\}$.

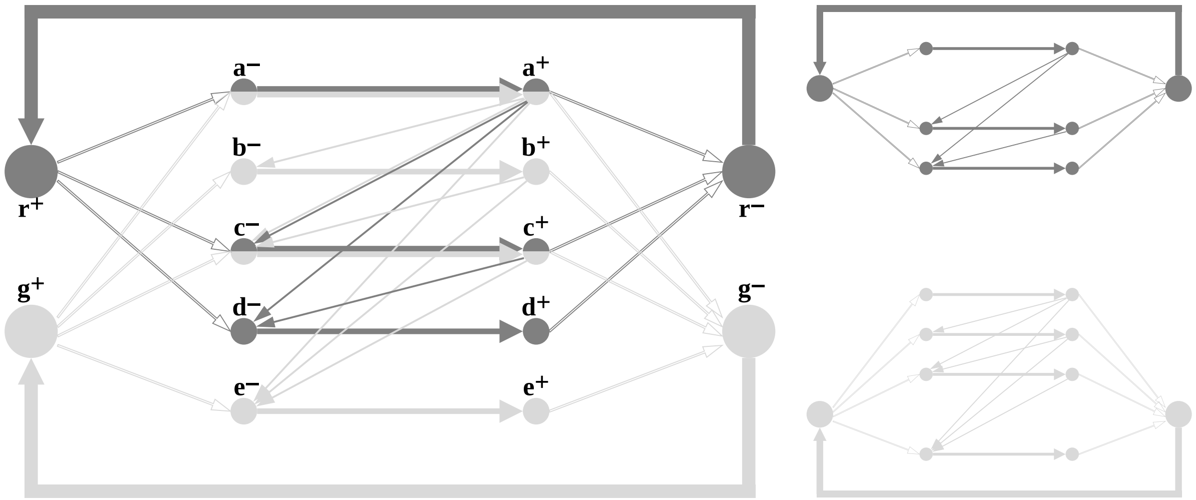

Figure 1: Digraphs (V, A), $(V_\mathrm{r}, A_\mathrm{r})$, and $(V_\mathrm{g}, A_\mathrm{g})$.

Since A – as a disjoint union – includes parallel arcs, we have to handle them carefully. For instance, addressing an arc (t^-, t^+) without knowing its depot $d \in G(t)$ may lead to confusion. If necessary, we distinguish explicitly such an arc by its corresponding depot.

We introduce an integer variable x^d_a for each arc $a \in A$ and each depot $d \in \mathcal{D}$. x^d_a denotes a decision variable indicating whether a vehicle of the depot d runs the trip a or not, except a denotes the backward arc of some depot d. In this case, x^d_a counts all employed vehicles of the depot d. The variables x^d_a are combined to vectors $x^d := (x^d_a)_{a\in A_d} \in \mathbb{R}^{A_d}$, $d \in \mathcal{D}$, and $x := (x^d)_{d\in\mathcal{D}} \in \mathbb{R}^A$. Given nodes u and $v \in V$, let

$$\delta^-(v) := \{a \in A \mid \exists\, d \in \mathcal{D} \text{ and } \exists\, (u,v) \in A_d \text{ such that } a = (u,v) \in A_d\}$$

denote all arcs having its head in v and

$$\delta^+(u) := \{a \in A \mid \exists\, d \in \mathcal{D} \text{ and } \exists\, (u,v) \in A_d \text{ such that } a = (u,v) \in A_d\}$$

denote all arcs having its tail in u. For a given set $\tilde{A} \subset A$ we define

$$x^d(\tilde{A}) := \textstyle\sum_{a\in\tilde{A}\cap A_d} x^d_a \qquad \text{and} \qquad x(\tilde{A}) := \textstyle\sum_{d\in\mathcal{D}} x^d(\tilde{A}).$$

The integer linear programming (ILP) formulation reads:

$$\min \sum_{d\in\mathcal{D}} \sum_{a\in A_d^{\text{u-trip}}} c^d_a\, x^d_a \tag{2.1a}$$

subject to

$$
\begin{array}{lrcll}
(2.1\text{b}) & \sum_{d \in G(t)} x^d_{(t^-,t^+)} & = & 1, & \forall\, t \in \mathcal{T}, \\
(2.1\text{c}) & x^d\big(\delta^+(v)\big) - x^d\big(\delta^-(v)\big) & = & 0, & \forall\, v \in \mathcal{T}_d \cup \{d^+, d^-\} \;\; \forall\, d \in \mathcal{D}, \\
(2.1\text{d}) & x^d_{(d^-,d^+)} & \leqslant & \kappa_d, & \forall\, d \in \mathcal{D}, \\
(2.1\text{e}) & x^d_a & \geqslant & 0, & \forall\, a \in A_d \;\; \forall\, d \in \mathcal{D}, \\
(2.1\text{f}) & & & x \text{ integral.} &
\end{array}
$$

Constraints (2.1b), the flow conditions, ensure that each timetabled trip is serviced exactly once. Constraints (2.1c), the flow conservations, guarantee that the total flow value of each depot d entering some node $v \in V$ also leaves it, in particular, any vehicle leaving the start point d^+ must reach the end point d^- and returns via the backward arc (d^-, d^+). Note that $x\big(\delta^+(t^+)\big) - x\big(\delta^-(t^-)\big) = 0$.

3 Two Lagrangean Relaxations of the MDVSP

We describe in this section the two basic ideas how the Lagrangean relaxation approach can be applied to the MDVSP such that the resulting Lagrangean functions become efficiently solvable minimum-cost flow problems.

3.1 Relaxation of the Flow Conservation Constraints

We deliberately do not include the individual depot capacities in the ILP formulation for this Lagrangean relaxation nor bring them into the objective function. The reasons are as follows: For many of our test instances, no depot capacities are given, or depot capacities are only weak constraints in the sense that it is allowed to shift vehicles from one garage to the other. Hence the variables $x^d_{(d^-,d^+)}$ and flow conservations (2.1c) for each node $v \in \mathcal{D}^+ \cup \mathcal{D}^-$ can be eliminated for this approach.

For the first Lagrangean relaxation, we use a slightly different ILP formulation:

$$
(3.1\text{a}) \qquad \min \sum_{d \in \mathcal{D}} \sum_{a \in A_d^{\text{u-trip}}} c^d_a \, x^d_a
$$

subject to

$$
\begin{array}{lrcll}
(3.1\text{b}) & x\big(\delta^+(t^+)\big) & = & 1, & \forall\, t \in \mathcal{T}, \\
(3.1\text{c}) & -x\big(\delta^-(t^-)\big) & = & -1, & \forall\, t \in \mathcal{T}, \\
(3.1\text{d}) & x^d\big(\delta^+(t^+)\big) - x^d\big(\delta^-(t^-)\big) & = & 0, & \forall\, t \in \mathcal{T}_d \;\; \forall\, d \in \mathcal{D}, \\
(3.1\text{e}) & x^d_a & \geqslant & 0, & \forall\, a \in A_d^{\text{u-trip}} \;\; \forall\, d \in \mathcal{D}, \\
(3.1\text{f}) & & & x \text{ integral.} &
\end{array}
$$

Obviously, the ILP formulations (2.1) and (3.1) are equivalent. It can be shown that (3.1b) and (3.1c) are linear combinations of (2.1b) and (2.1c), and (3.1d) is a linear combination of (2.1c). If we add (3.1b)–(3.1d) to the ILP (2.1), the variables for the timetabled trips, the original flow conditions (2.1b), and the original flow conservations (2.1c) become redundant and can be eliminated.

Let $\pi := (\pi^d \in \mathbb{R}^{\mathcal{T}_d})_{d\in\mathcal{D}}$ denote the Lagrangean multipliers according to the flow conservations (3.1d). With respect to (3.1d), the Lagrangean function of (3.1) is the ILP

$$L_{\text{fcs}}(\pi) := \min_{\substack{x \geqslant 0 \text{ integral} \\ \text{satisfying} \\ (3.1\text{b}) \text{ and } (3.1\text{c})}} \sum_{d\in\mathcal{D}} \Big\{ \sum_{a\in A_d^{\text{u-trip}}} c_a^d\, x_a^d - \sum_{t\in\mathcal{T}_d} \pi_t^d \Big(x^d\big(\delta^+(t^+)\big) - x^d\big(\delta^-(t^-)\big)\Big)\Big\} \tag{3.2}$$

and the Lagrangean relaxation LR_{fcs} of (3.1) reads

$$\max_{\pi} L_{\text{fcs}}(\pi). \tag{3.3}$$

The subscript "fcs" of L_{fcs} and LR_{fcs} stands for **F**low-**C**on**S**ervation.

$L_{\text{fcs}}(\pi)$ corresponds to a large minimum-cost flow problem, and it is well known that for such problems there always exists an integer optimal solution for each objective function provided there exists any optimum. It is easy to show that $L_{\text{fcs}}(\pi)$ is equivalent to a large single-depot instance whereby all depots are considered as one large depot.

Any optimal solution $x := x(\pi)$ attaining the value of $L_{\text{fcs}}(\pi)$ describes a set of vehicle schedules covering each timetabled trip exactly once. Some vehicle schedule, however, violates some flow conservations (3.1d) if its trips belong to different depots.

3.2 Relaxation of the Flow Condition Constraints

For the second Lagrangean relaxation, we use the original ILP formulation (2.1a)–(2.1f) plus additional redundant constraints $x^d_{(t^-,t^+)} \leqslant 1$, for all $(t^-,t^+) \in A^{\text{t-trip}}$ and all $d \in \mathcal{D}$. The formulation reads

$$\min \sum_{d\in\mathcal{D}} \sum_{a\in A_d^{\text{u-trip}}} c_a^d\, x_a^d \tag{3.4a}$$

subject to

$$\begin{array}{llcll}
(3.4\text{b}) & \sum_{d\in G(t)} x^d_{(t^-,t^+)} & = & 1, & \forall\, t \in \mathcal{T}, \\
(2.1\text{c}) & x^d\big(\delta^+(v)\big) - x^d\big(\delta^-(v)\big) & = & 0, & \forall\, v \in \mathcal{T}_d \cup \{d^+, d^-\}\ \ \forall\, d \in \mathcal{D}, \\
(3.4\text{d}) & x^d_{(d^-,d^+)} & \leqslant & \kappa_d, & \forall\, d \in \mathcal{D}, \\
(3.4\text{e}) & x^d_{(t^-,t^+)} & \leqslant & 1, & \forall\, (t^-,t^+) \in A_d^{\text{t-trip}}\ \ \forall\, d \in \mathcal{D}, \\
(3.4\text{f}) & x^d_a & \geqslant & 0, & \forall\, a \in A_d\ \ \forall\, d \in \mathcal{D}. \\
(3.4\text{g}) & & & x \text{ integral.} &
\end{array}$$

Let $\nu := (\nu_t)_{t\in\mathcal{T}} \in \mathbb{R}^{\mathcal{T}}$ denote the Lagrangean multipliers according to the flow conditions (3.4b). With respect to (3.4b), the Lagrangean function of (3.4) is the ILP

$$L_{\mathrm{fcd}}(\nu) := \min_{\substack{x \text{ satisfies} \\ (2.1\,c)-(3.4\,g)}} \Big\{ \sum_{d\in\mathcal{D}} \sum_{a\in A_d^{\text{u-trip}}} c_a^d\, x_a^d \;-\; \sum_{t\in\mathcal{T}} \nu_t \Big(\sum_{d\in G(t)} x^d_{(t^-,t^+)} \;-\; 1 \Big) \Big\} \tag{3.5}$$

and the Lagrangean relaxation $\mathrm{LR}_{\mathrm{fcd}}$ of (3.4) reads

$$\max_{\nu} L_{\mathrm{fcd}}(\nu). \tag{3.6}$$

The subscript "fcd" of L_{fcd} and $\mathrm{LR}_{\mathrm{fcd}}$ stands for **F**low-**C**on**D**ition. Note that the Lagrangean function L_{fcd} can be decomposed into a constant objective shift $\nu^{\mathrm{T}}\mathbb{1}$ and into $|\mathcal{D}|$ independently solvable minimum-cost flow (circulation) problems. It is easy to see that a feasible solution of each subproblem d corresponds to a set of vehicle schedules satisfying the depot specific capacity.

4 Implementation Details

The subgradient algorithm is almost the same for both Lagrangean relaxations $\mathrm{LR}_{\mathrm{fcs}}$ and $\mathrm{LR}_{\mathrm{fcd}}$. Here is our basic method, the details of each step are outlined below:

(4.1) Algorithm. (Subgradient Method for the MDVSP)
Input: Lagrangean function (3.2) L_{fcs} or (3.5) L_{fcd}, denoted by L.
Output: Lower bound for ILP (2.1).
1. Choose initial Lagrangean multiplier $u^{(0)}$ and set $k := 0$.
2. Evaluate $L(u^{(k)})$.
3. Compute subgradient $g^{(k)} := g^{(k)}(u^{(k)})$.
4. If the iteration limit N_1 is reached or $g^{(k)} = 0$ then STOP; otherwise continue.
5. Compute new step length $\sigma^{(k)}$.
6. Compute new step direction $\tilde{g}^{(k)} \in \mathrm{conv}\{g^{(0)}, \ldots, g^{(k)}\}$.
7. Set $u^{(k+1)} := u^{(k)} + \sigma^{(k)}\tilde{g}^{(k)}$.
8. Increment k and goto 2.

Step 1. Choose initial Lagrangean multiplier.
The initial $\pi^{(0)}$ for $\mathrm{LR}_{\mathrm{fcs}}$ is set to zero. Initializing $\nu^{(0)}$ for $\mathrm{LR}_{\mathrm{fcd}}$: It turned out from computational investigations that the following initialization of the Lagrangean multipliers $\nu^{(0)}$ performs best concerning the maximal achieved objective value L_{fcd} and the number of iterations of the subgradient method: Solve the Lagrangean function $L_{\mathrm{fcs}}(0)$ of the Lagrangean relaxation (3.3) $\mathrm{LR}_{\mathrm{fcs}}$. Let ν^+ and ν^- denote the optimal dual multipliers of the constraints (3.1b) and (3.1c), respectively. Then $\nu^{(0)} := \nu^+ - \nu^-$ defines a good starting point since it can be shown that $L_{\mathrm{fcd}}(\nu^{(0)})$ and $L_{\mathrm{fcs}}(0)$ yield the same value.

Step 2. Evaluate Lagrangean function.
The core of our Lagrangean relaxation codes is the network simplex implementation MCF

of [12]. This paper reports about the implementation details for the Lagrangean relaxation LR_{fcs} (3.3). Important ingredients of MCF are the multiple partial pricing rule and a delayed column generation for large-scale test instances. We also use a modified version of MCF to solve the minimum-cost flow problems occurring within the Lagrangean function L_{fcd} (3.6). The minimum-cost flow problem arising in the evaluation of $L(u^{(k)})$ is solved to optimality with delayed column generation for both relaxations. All unloaded trips provided by our partners (and all timetabled trips in the case of LR_{fcd}) define the initial restricted arc set.

Step 3. Compute subgradient.
Let $\pi^{(k)}$ denote the k^{th} Lagrangean multiplier for LR_{fcs}, and let $x^{(k)}$ denote some optimal solution attaining the value of $L_{fcs}(\pi^{(k)})$. Then

$$g^{(k)} = g^{(k)}(\pi^{(k)}) := \left(\left((x^d)^{(k)}\big(\delta^-(t^-)\big) \;-\; (x^d)^{(k)}\big(\delta^+(t^+)\big) \right)_{t \in \mathcal{T}_d} \right)_{d \in \mathcal{D}}$$

is a subgradient for L_{fcs} at $\pi^{(k)}$.

Let $\nu^{(k)}$ denote the k^{th} Lagrangean multiplier for LR_{fcd}, and let $x^{(k)}$ denote now some optimal solution attaining the value of $L_{fcd}(\nu^{(k)})$. Then

$$g^{(k)} = g^{(k)}(\nu^{(k)}) := \mathbb{1} - \left(x^{(k)}\big(\delta^+(t^-)\big) \right)_{t \in \mathcal{T}}$$

is a subgradient for L_{fcd} at $\nu^{(k)}$. It is also easy to see that $g_t^{(k)} \in \{\ldots, -2, -1, 0, 1\}$.

Step 4. Stopping criteria.
First, we check whether $g^{(k)} = 0$. If this condition is satisfied, the corresponding $x^{(k)}$ is feasible as well as optimal, and we can STOP. Second, we check our iteration limit, denoted by N_1. If $k > N_1$ we STOP. Our standard value for N_1 is 100.

Step 5. Compute new step length.
Our goal was to compute good lower bounds quickly. Therefore, we focused our efforts on performance improvements within our given iteration limit rather than satisfying some convergence criteria. The step length and the step direction play a key role here. Based on the parameter setting, we use one of the following step length rules:

A)
$$\sigma^{(k)} := \begin{cases} \frac{\sigma^{(k-1)}}{2}, & \text{if } L \text{ declines for } N_2 \text{ consecutive iteration(s)}, \\ \sigma^{(k-1)}, & \text{else} \end{cases}$$

with $\sigma^{(0)} := 10$ and a maximum failure parameter $N_2 := 2$.

B)
$$\sigma^{(k)} := \frac{\alpha^{(k)}\left(\overline{L} - L(u^{(k)})\right)}{\|g^{(k)}\|^2} \quad \text{with} \quad \alpha^{(k)} := \begin{cases} \frac{\alpha^{(k-1)}}{2}, & \begin{array}{l}\text{if } L \text{ declines for } N_2 \\ \text{consecutive iteration(s)},\end{array} \\ \alpha^{(k-1)}, & \text{else,} \end{cases}$$

$\alpha^{(0)}$ such that $\sigma^{(0)}$ becomes 10, a maximum failure parameter $N_2 := 2$, and an upper bound $\overline{L}$ for L provided by [13].

The objective for all our test data is to minimize the fleet size and, secondary, to minimize the operational costs among all fleet minimal solutions. For almost all of our test instances, the initial Lagrangean function $L_{\mathrm{fcs}}(0)$ and $L_{\mathrm{fcd}}(\nu^{(0)})$ provide already the exact value for the fleet size. Therefore, to preserve this part of the lower bound, we choose a value for $\sigma^{(0)}$ that is significantly smaller than our big M value.

The initial step length is the only sensitive parameter of our subgradient methods. For each of our test instances, we have made several test runnings to find out good starting values for $\sigma^{(0)}$. We are, however, not able to provide any reasonable rule for a good starting configuration. Thus, we have decided to use $\sigma^{(0)} = 10$ as default for our presentation. For our complete test set, this value was one of the best among all that we have tried out. We refer the reader to [10] for a detailed description of these tests.

Step 6. Compute new step direction $\tilde{g}^{(k)}$.
Following the key idea of bundle methods, e. g., see [4], we use the following step direction:

$$\tilde{g}^{(k)} := 0.6g^{(k)} + 0.2g^{(k-1)} + 0.1g^{(k-2)} + 0.1g^{(k-3)}$$

(we set $g^{(-1)} := g^{(-2)} := g^{(-3)} := g^{(0)}$). This turned out to be a robust choice.

5 Test Data

Our computational investigations are based on real-world test instances from the cities of Berlin and Hamburg. Table 1 gives the characteristics of our test set problems; the average size of the depot-group $G(t)$ for a timetabled trip $t \in \mathcal{T}$ is denoted by $\varnothing|G(t)|$.

The problems from Berlin

Berlin 1: At present, BVG maintains 9 garages and runs 10 different vehicle types (2 double-decker types, 6 single-decker types, and 3 articulated bus types). Combining the garages with their available vehicle types results in 44 different depots. There are no depots containing vehicles of different garages. For a normal weekday, about 28,000 timetabled trips have to be serviced. This number reduces to about 24,900 timetabled trips since some of the timetabled trips are out-sourced to third-party companies. The number of unloaded trips for our largest Berlin instance is about 70 millions from which about 846 thousand are explicitly user-defined.

Berlin 2: Since the LP relaxation for Berlin 1 is extremely large – more than 125 thousand equations – we have generated a smaller problem with more depots, but much fewer unloaded trips. This problem is based on the same set of timetabled trips as the problem Berlin 1, but the depots and the dead-head trips are generated with different rules resulting in fewer degrees of freedom.

Berlin 3: This is a relatively small test instance including 9 lines from the south of Berlin and 3 depots from one single garage.

Berlin-Spandau: All the test sets denoted by Berlin-Spandau are defined on the data of the district of Spandau for different weekdays and different depot generation rules.

Test Sets[a]	$\lvert\mathcal{D}\rvert$	$\lvert\mathcal{T}\rvert$	$\lvert A\rvert$		$\varnothing\lvert G(t)\rvert$
			User-def.	Total	
Berlin 1	44	24,906	846,000	69,700,000	4.03
Berlin 2	49	24,906	304,000	13,200,000	1.56
Berlin 3	3	1,313	77,000	2,300,000	2.33
Berlin-Spandau 1	9	2,424	164,000	3,700,000	4.94
Berlin-Spandau 2	9	3,308	327,000	8,800,000	5.49
Berlin-Spandau 3	13	2,424	39,000	590,000	1.92
Berlin-Spandau 4	13	3,308	72,000	1,530,000	2.25
Berlin-Spandau 5	13	3,331	75,000	1,550,000	2.25
Berlin-Spandau 6	13	1,998	28,000	380,000	1.90
Berlin-Spandau 7	7	2,424	145,000	3,300,000	4.16
Berlin-Spandau 8	7	3,308	283,000	7,800,000	5.02
Hamburg 1	12	8,563	1,322,000	10,900,000	2.23
Hamburg 2	9	1,834	99,000	1,000,000	2.02
Hamburg 3	2	791	30,000	200,000	1.32
Hamburg 4	2	238	2,000	23,000	1.04
Hamburg 5	2	1,461	85,000	580,000	1.31
Hamburg 6	2	2,283	176,000	1,600,000	1.33
Hamburg 7	2	341	6,000	34,000	1.32
Hamburg-Holstein	4	3,413	230,000	4,000,000	1.68

[a]Compared to the problems presented in [8], the number of timetabled trips and unloaded trips and the weights for some unloaded trips have been changed for some instances due to slightly different rules for the depot generation and compatibility of dead-head trips.

Table 1: Real-world test sets.

The problems from Hamburg and Hamburg-Holstein

At the moment, HHA together with other transportation companies maintain 14 garages with 9 different vehicle types. The existing garage-vehicle combinations define 40 depots. In Hamburg, more than 16,000 daily timetabled trips must be scheduled with about 15.1 million unloaded trips. The problem in Hamburg decomposes into a 12-depot problem, a 9-depot problem, five smaller 2-depot problems, and nine small 1-depot problems. We consider only the multiple-depot problems, denoted by Hamburg 1–7. The planning of the region Hamburg-Holstein is based on 18 garages, each defining one depot.

6 Computational Results

We have made many test runnings to find out the right parameter configurations for our subgradient methods. Among them we first present the computational results for our default configuration: a maximal number of iterations $N_1 := 100$; the step length rule A with maximal number of consecutive failures $N_2 := 2$ and $\sigma^{(0)} := 10$; and the step direction $\tilde{g}^{(k)} := 0.6g^{(k)} + 0.2g^{(k-1)} + 0.1g^{(k-2)} + 0.1g^{(k-3)}$. Afterwards, we show some effects if the one or the other parameter is changed.

6.1 Results for the Default Configuration

In Tab. 2 we summarize the objective values (fleet size and operational weight) of the investigated subgradient methods for $\mathrm{LR}_{\mathrm{fcs}}$ and $\mathrm{LR}_{\mathrm{fcd}}$ in comparison with the values of the LP relaxation taken from [13] and the optimum taken from [14]. The values of the column "Best run" in Tab. 2 and 3 are the objectives of the best parameter configurations provided in [10].

Table 3 gives the gaps relative to the optimum. The most surprising result is that the trivial problem relaxation $L_{\mathrm{fcs}}(0)$ neglecting the flow conservations provides almost always a tight lower bound for the fleet size. The largest gap for the fleet size is at most 0.01 %. It is remarkable that this simple problem relaxation gives such tight estimations of the fleet size. Rounding up the fleet sizes given by the LP relaxation to the next integer value, the gap to the optimal fleet size is always zero. Hence, we consider our problems to be well conditioned in some sense, which we see to be confirmed by our computational results. Ignoring the fleet sizes and considering only the operational costs, the gap between $L_{\mathrm{fcs}}(0)$ and the optimum is already not larger than 16.01 %. With a good parameter configuration depending on each single instance, this gap can be reduced with our subgradient methods to less than 10 %.

As we can observe from Tab. 4, none of the two relaxations together with the subgradient methods with standard parameter configuration yield significantly better lower bounds for our test set. From our point of view, we cannot prefer one of them definitely. The advantage of the relaxation with respect to the flow conditions is that the largest instance, Berlin 1, is solved significantly faster. This may be strengthened if the evaluation of the Lagrangean function L_{fcd} is parallelized. On the other side, however, the other Lagrangean relaxation with respect to the flow conservations can be employed within a schedule first – cluster second heuristic, which works quite good in practice.

Theoretically, the two Lagrangean relaxations and the LP relaxation yield the same optimal value. Nevertheless, our subgradient methods do not attain the LP value for three instances, but the resulting gap are very small and can be neglected, see Tabs. 2 and 3. However, the big advantage of the Lagrangean relaxations is that they can be solved by far faster than the LP relaxation, see Tab. 4.

6.2 Variations of Certain Parameters

Considering a fixed parameter configuration, our methods show a similar behaviour for each test instance. Thus, we present the effects of varying a single parameter and use the problem Hamburg 1 as an example.

Test Sets	Subgradient methods						LP relaxation		Optimum	
	Fleet	Weight					Fleet	Weight	Fleet	Weight
			Default		Best run[a]					
		$L_{\text{fcs}}(0)$	LR_{fcs}	LR_{fcd}	LR_{fcs}	LR_{fcd}				
Berlin 1	1323	715714	708008	704666	—	—	1323	759162.0[b]	1329[c]	850680
Berlin 2	1350	715623	719952	718463	—	—	1353.7	797918.8	1354	777823[d]
Berlin 3	69	14043	14075	14105	14101	14111	69	14119.0	69	14119
B-Spandau 1	125	65585	64984	65021	65586	65586	125	65610.5	125	65611
B-Spandau 2	184	78947	78204	77342	79038	78963	184.5	79110.0	185	79052[d]
B-Spandau 3	127	90514	92711	92298	93461	93534	127	93745.0	127	93745
B-Spandau 4	191	195844	197786	203178	205752	209023	191	230846.0	191	230846
B-Spandau 5	191	191141	194436	197965	202886	207237	191	227580.0	191	227580
B-Spandau 6	98	91109	92490	91797	98392	92700	98	101075.0	98	101075
B-Spandau 7	125	65585	65135	65427	—	—	125	65610.5	125	65611
B-Spandau 8	184	78947	78657	78072	—	—	184.5	79110.0	185	79160[d]
Hamburg 1	432	66874	69728	68447	70266	68447	432	71068.3	432	71069
Hamburg 2	103	15356	15769	15759	15814	15769	103	16070.0	103	16070
Hamburg 3	39	5557	5796	5733	5796	5860[e]	39	5860.0	39	5860
Hamburg 4	6	1358[f]	—	—	—	—	6	1358.0	6	1358
Hamburg 5	62	12092	12430	12374	12439	12502	62	12502.0	62	12502
Hamburg 6	111	15705	15791	15791[g]	15791[h]	15791	111	15791.0	111	15791
Hamburg 7	15	2832	2961	2959	2961[i]	2961[j]	15	2961.0	15	2961
H-Holstein	201	28697	28860	28916	28972	29018	201	29027.0	201	29027

[a]Best parameter configuration of [10]
[b]Best known LP value, which is certainly not optimal.
[c]Best known solution value.
[d]Not proved to be optimal.
[e]Feasible and optimal after 65 iterations.
[f]Already feasible and optimal.
[g]Feasible and optimal after 24 iterations.
[h]Feasible and optimal after 55 iterations.
[i]Feasible and optimal after 31 iterations.
[j]Feasible and optimal after 50 iterations.

Table 2: Vehicle demand and operational weights.

Test Sets	Gaps relative to optimum in %							
	Fleet[a]		Weight					
			Subgradient methods					LP relax.
				Default		Best run[b]		
	Lagr. relax.	LP relax.	$L_{\mathrm{fcs}}(0)$	$\mathrm{LR}_{\mathrm{fcs}}$	$\mathrm{LR}_{\mathrm{fcd}}$	$\mathrm{LR}_{\mathrm{fcs}}$	$\mathrm{LR}_{\mathrm{fcd}}$	
Berlin 2[c]	0.01	0.00	—	—	—	—	—	—
Berlin 3	0.00	0.00	0.54	0.31	0.10	0.13	0.06	0.00
B-Spandau 1	0.00	0.00	0.03	1.00	0.90	0.04	0.04	0.00
B-Spandau 2[c]	0.01	0.00	—	—	—	—	—	—
B-Spandau 3	0.00	0.00	3.45	1.10	1.54	0.30	0.22	0.00
B-Spandau 4	0.00	0.00	15.16	14.32	11.98	10.87	9.45	0.00
B-Spandau 5	0.00	0.00	16.01	14.56	13.01	10.85	8.94	0.00
B-Spandau 6	0.00	0.00	9.86	8.49	9.18	2.65	8.29	0.00
B-Spandau 7	0.00	0.00	0.03	0.72	0.28	—	—	0.00
B-Spandau 8[c]	0.01	0.00	—	—	—	—	—	—
Hamburg 1	0.00	0.00	5.90	1.89	3.69	1.13	3.69	0.00
Hamburg 2	0.00	0.00	4.44	1.87	1.93	1.59	1.87	0.00
Hamburg 3	0.00	0.00	5.17	1.09	2.16	1.09	0.00	0.00
Hamburg 4	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Hamburg 5	0.00	0.00	3.27	0.57	1.02	0.50	0.00	0.00
Hamburg 6	0.00	0.00	0.54	0.00	0.00	0.00	0.00	0.00
Hamburg 7	0.00	0.00	4.36	0.00	0.07	0.00	0.00	0.00
H-Holstein	0.00	0.00	1.13	0.57	0.38	0.19	0.03	0.00

[a]Berlin 1 is not solved to optimality.

[b]Best parameter configuration of [10]

[c]There is still a fleet size gap between the optimum and all Lagrangean functions. Therefore, we are not able to give weight gaps.

Table 3: Gaps relative to the optimum.

Test Sets	Subgradient methods			LP relax.[a]
	$L_{\mathrm{fcs}}(0)$	$\mathrm{LR}_{\mathrm{fcs}}$	$\mathrm{LR}_{\mathrm{fcd}}$	
Berlin 1	916	49258	31602	—[b]
Berlin 2	229	11051	4819	30985
Berlin 3	17	365	417	311
B-Spandau 1	27	1501	1172	43777
B-Spandau 2	93	4447	3834	170408
B-Spandau 3	9	357	236	739
B-Spandau 4	25	1116	858	4384
B-Spandau 5	31	1037	818	5264
B-Spandau 6	17	189	142	162
B-Spandau 7	23	1367	1063	24717
B-Spandau 8	67	3919	3118	62041
Hamburg 1	185	6838	6643	50246
Hamburg 2	12	386	415	685
Hamburg 3	4	52	89	31
Hamburg 4	2	2	2	3
Hamburg 5	10	221	332	155
Hamburg 6	18	198	267	84
Hamburg 7	2	7	12	8
H-Holstein	40	1371	1919	2087

[a]Fastest method taken from [13].

[b]More than 375 hours cpu time were necessary to reach the best known LP value, and 200 hours to reach a fleet minimal value.

Table 4: Running times in seconds on a SUN Model 170 UltraSPARC with 512 MByte main memory.

Figures 2 and 3 show the influence of different step directions: "1 subgradient" means $\tilde{g}^{(k)} := g^{(k)}$, "2 subgradients" means $\tilde{g}^{(k)} := 0.7g^{(k)} + 0.3g^{(k-1)}$, and "4 subgradients" means (default) $\tilde{g}^{(k)} := 0.6g^{(k)} + 0.2g^{(k-1)} + 0.1g^{(k-2)} + 0.1g^{(k-3)}$. Using more than one subgradient leads to a faster convergence and, in general, yields better lower bounds. For our test set and for our computational test runnings, the default step direction shows the fastest convergence and provides, on the average, the better lower bounds than the other tested directions.

As we can observe from Figs. 4 and 5, the two presented step size rules perform comparable. This is the main reason to use the simpler rule A as default since we do not need to provide an upper bound $\overline{L}$ for the Lagrangean functions.

The last Fig. 6 shows the behaviour for different maximal allowed consecutive failures N_2. The Lagrangean functions improve quickly within few iterations if we use $N_2 = 1$,

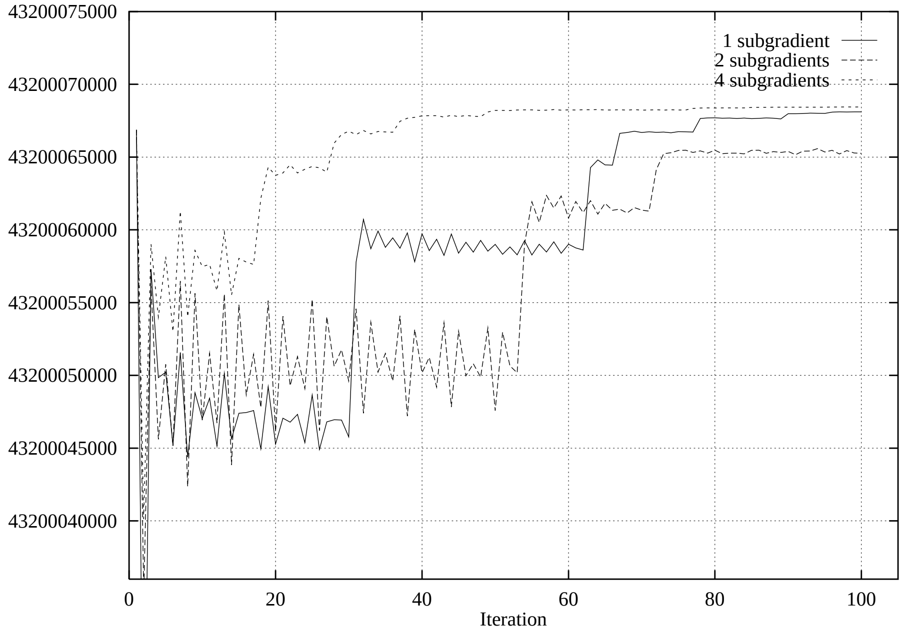

Figure 2: Comparing different step directions for Hamburg 1 and LR_{fcd}.

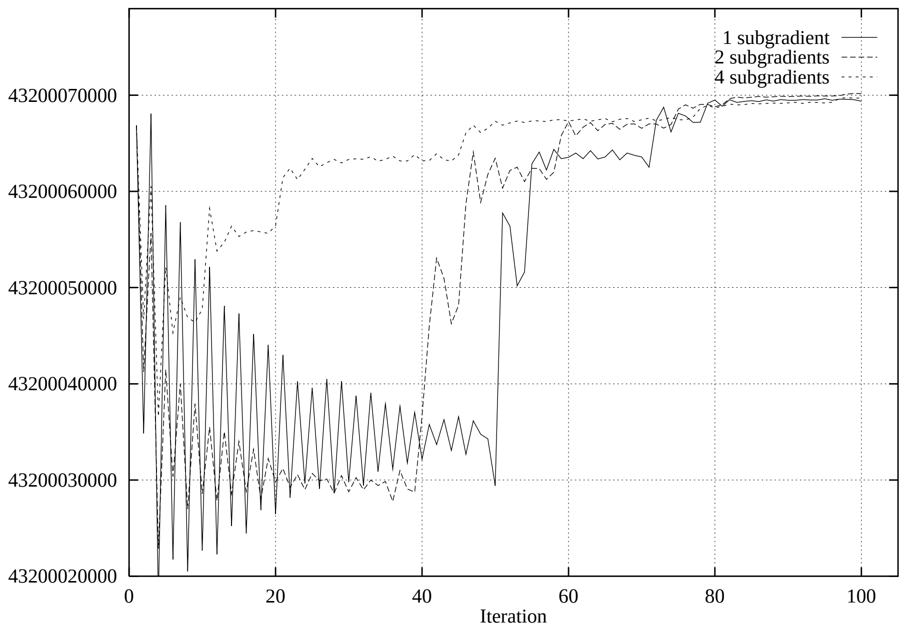

Figure 3: Comparing different step directions for Hamburg 1 and LR_{fcs}.

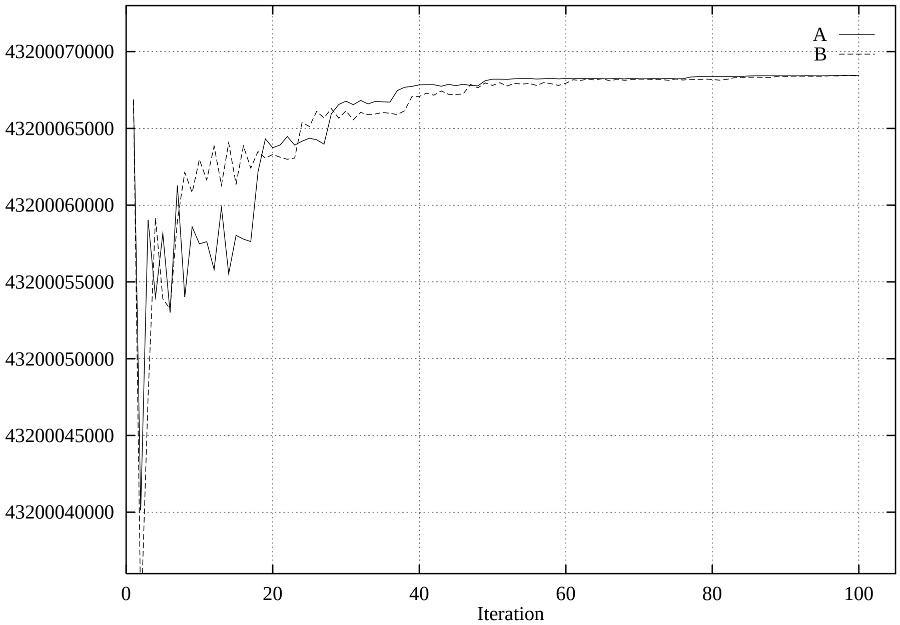

Figure 4: Comparing the two step length rules for Hamburg 1 and $\mathrm{LR_{fcd}}$.

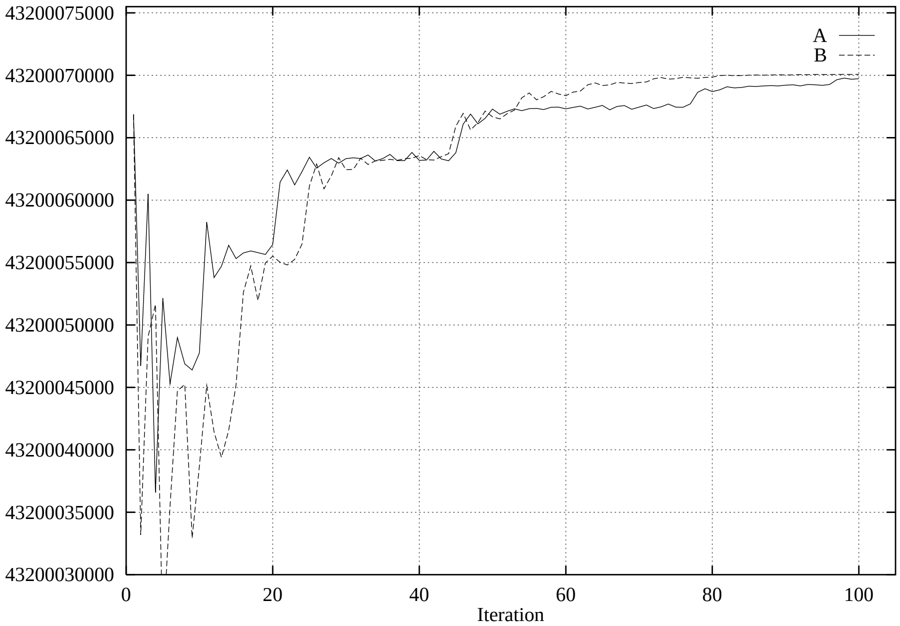

Figure 5: Comparing the two step length rules for Hamburg 1 and $\mathrm{LR_{fcs}}$.

but afterwards, no further significant improvements can be achieved since the step length is reduced far too fast. For $N_2 = 3$, the step length is reduced only once or twice such that the subgradient method cannot converge to an optimal solution, but starts to oscillate until the iteration limit is reached. Empirically based on our observations, we can claim that $N_2 = 2$ is a good compromise.

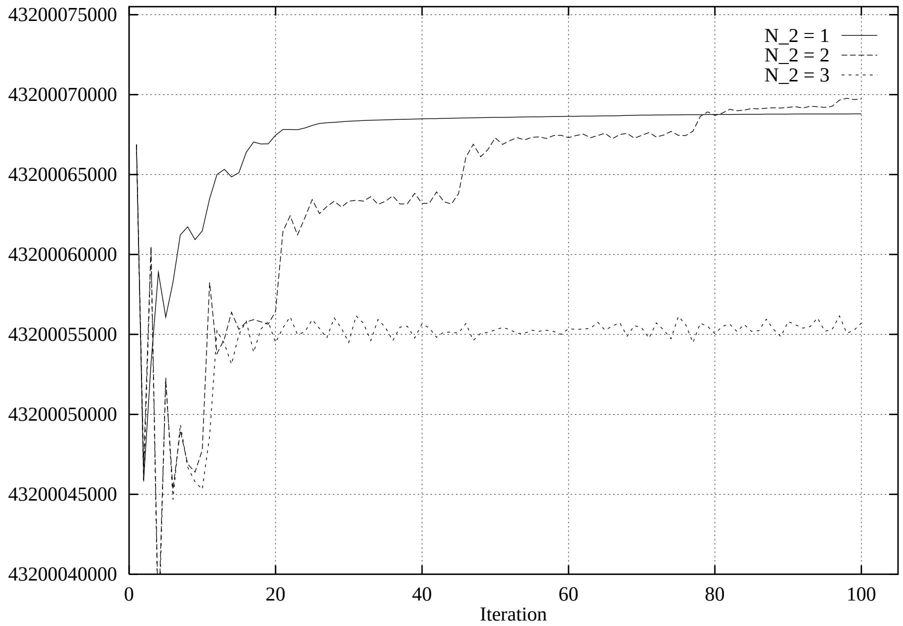

Figure 6: Comparing different maximum failure values N_2 for Hamburg 1 and $\mathrm{LR}_{\mathrm{fcs}}$.

7 Conclusions

We have presented two Lagrangean relaxations for the MDVSP and reported on numerical tests for the subgradient methods on truely large-scale real-world test instances. The main results of this paper are as follows:

Our real-world test instances seem to be fairly well structured. On the one hand, the trivial problem relaxation neglecting the flow conservation constraints gives already a very good approximation for the minimum integral fleet size. On the other hand, this relaxation can be evaluated efficiently for our instances. So, we are able to quickly compute tight lower bounds for the fleet size.

If we ignore the values for the fleet size, there are still gaps (up to 16 %) of the operational costs between our trivial problem relaxation and the LP relaxation. To reduce them, we employed the presented subgradient methods. For these methods, testing many parameter configurations for each instance may lead to a significant gap reduction. However, we are unable to provide a default starting parameter setting improving significantly the lower bound for the operational costs for each of our test instances. The possible improvements obtained by many runs of the subgradient methods testing different parameter settings is entirely disproportionate to the necessary running times.

Since our methods provide tight lower bounds for the fleet size, public transportation companies can use them to simulate different scenarios in vehicle scheduling, e. g., different compatibility rules or different weights of unloaded trips, and so easily find out the best of them. From our point of view, the Lagrangean relaxations are of further importance within a delayed column generation approach to solve the LP relaxation exactly, see [13, 14].

8 Acknowledgements

This work has been supported by the German Federal Ministry of Education, Science, Research, and Technology grant no. 03-GR7ZIB -7. We are grateful to Manfred Völker and Anna Neufeld of HanseCom GmbH and Uwe Strubbe of IVU GmbH for providing us with real-world problems from the cities of Hamburg and Berlin. We are grateful to the Berliner Verkehrsbetriebe and the Hamburger Hochbahn AG for their kind permission to use and publish their data.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1993.

[2] A. A. Bertossi, P. Carraresi, and G. Gallo. On some matching problems arising in vehicle scheduling models. *Networks*, 17:271–181, 1987.

[3] I. Branco, A. Costa, and J. M. P. Paixão. Vehicle scheduling problem with multiple type of vehicles and a single depot. In J. R. Daduna, I. Branco, and J. M. P. Paixão, editors, *Computer-Aided Transit Scheduling*, Lecture Notes in Economics and Mathematical Systems, pages 115–129. Springer Verlag, 1995.

[4] H. Crowder. Computational improvements for subgradient optimization. In *Symposia Mathematica*, volume 19. Istituto Nazionale di Alta Matematica, Academic Press London and New York, 1976.

[5] J. R. Daduna and M. Mojsilovic. Computer-aided vehicle and duty scheduling using the HOT programme system. In J. R. Daduna and A. Wren, editors, *Computer-Aided Transit Scheduling*, Lecture Notes in Economics and Mathematical Systems, pages 133–146, 1988.

[6] M. Dell'Amico, M. Fischetti, and P. Toth. Heuristic algorithms for the multiple depot vehicle scheduling problem. *Management Science*, 39(1):115–125, January 1993.

[7] M. Desrochers and J.-M. Rousseau, editors. *Computer-Aided Transit Scheduling*, Lecture Notes in Economics and Mathematical Systems. Springer Verlag, 1992.

[8] M. Grötschel, A. Löbel, and M. Völker. Optimierung des Fahrzeugumlaufs im öffentlichen Nahverkehr. In K.-H. Hoffmann, W. Jäger, T. Lohmann, and H. Schunck, editors, *Mathematik – Schlüsseltechnologie für die Zukunft*, pages 609–624. Springer Verlag, 1997. In German available as ZIB preprint SC 96-7 via ftp or WWW*.

[9] J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex Analysis and Minimization Algorithms I: Fundamentals*. Springer Verlag, 1993.

[10] A. Kokott. Lagrange-Relaxierungen für das Fahrzeugumlaufplanungsproblem. Master's thesis, Technische Universität Berlin, 1996.

[11] A. Lamatsch. An approach to vehicle scheduling with depot capacity constraints. In Desrochers and Rousseau [7].

[12] A. Löbel. Solving large-scale real-world minimum-cost flow problems by a network simplex method. Preprint SC 96-7, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1996. Available via ftp or WWW*.

[13] A. Löbel. Solving the LP relaxations of large-scale multiple-depot vehicle scheduling problems exactly. Preprint SC 96-26, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1996. Available via ftp or WWW*.

[14] A. Löbel. *Optimal Vehicle Scheduling in Public Transit*. PhD thesis, Technische Universität Berlin. To appear in 1997.

[15] M. Mesquita and J. Paixão. Multiple depot vehicle scheduling problem: A new heuristic based on quasi-assignment algorithms. In Desrochers and Rousseau [7].

[16] A. Schrijver. *Theory of Linear and Integer Programming*. John Wiley & Sons Ltd., Chichester, 1989.

*Available via anonymous-ftp: `ftp ftp.zib.de` ... `cd /pub/zib-publications` or at URL: `http://www.zib.de/ZIBbib/Publications/`.