MODELS AND ALGORITHMS FOR INTEGRATION OF VEHICLE AND CREW SCHEDULING

RICHARD FRELING, DENNIS HUISMAN AND

ALBERT P.M. WAGELMANS

ERIM REPORT SERIES *RESEARCH IN MANAGEMENT*	
ERIM Report Series reference number	ERS-2000-14-LIS
Publication status / version	draft / version March 2000
Number of pages	33
Email address of corresponding author	Huisman@few.eur.nl
Address	Erasmus Research Institute of Management (ERIM) Rotterdam School of Management / Faculteit Bedrijfskunde Erasmus Universiteit Rotterdam PoBox 1738 3000 DR Rotterdam, The Netherlands Phone: # 31-(0) 10-408 1182 Fax: # 31-(0) 10-408 9020 Email: info@erim.eur.nl Internet: www.erim.eur.nl

Bibliographic data and classifications of all the ERIM reports are also available on the ERIM website:
www.erim.eur.nl

ERASMUS **R**ESEARCH **I**NSTITUTE OF **M**ANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

BIBLIOGRAPHIC DATA AND CLASSIFICATIONS		
Abstract	This paper deals with models, relaxations and algorithms for an integrated approach to vehicle and crew scheduling. We discuss potential benefits of integration and provide an overview of the literature, which considers mainly partial integration. Our approach is new in the sense that we can tackle integrated vehicle and crew scheduling problems of practical size. We propose new mathematical formulations for integrated vehicle and crew scheduling problems and we discuss corresponding Langrangian relaxations and Lagrangian heuristics. To solve the Lagrangian relaxations, we use column generation applied to set partitioning type of models. The paper is concluded with a computational study using real life data, which shows the applicability of the proposed techniques to practical problems. Furthermore, we also address the effectiveness of integration in different situations.	
Library of Congress Classification (LCC)	5001-6182	Business
	5201-5982	Business Science
	HD 69.T54	Scheduling
Journal of Economic Literature (JEL)	M	Business Administration and Business Economics
	M 11 R 4	Production Management Transportation Systems
	C 6	Mathematical Methods and Programing
European Business Schools Library Group (EBSLG)	85 A	Business General
	260 K 240 B	Logistics Information Systems Management
	250 A	Mathematics
Gemeenschappelijke Onderwerpsontsluiting (GOO)		
Classification GOO	85.00	Bedrijfskunde, Organisatiekunde: algemeen
	85.34 85.20	Logistiek management Bestuurlijke informatie, informatieverzorging
	31.80	Toepassingen van de wiskunde
Keywords GOO	Bedrijfskunde / Bedrijfseconomie	
	Bedrijfsprocessen, logistiek, management informatiesystemen	
	Algoritmen, Scheduling, Vervoersmodellen, Roosters	
Free keywords	Vehicle scheduling, crew scheduling, integrated planning, column generation, Lagrangian relaxation	
Other information		

MODELS AND ALGORITHMS FOR INTEGRATION OF VEHICLE AND CREW SCHEDULING

RICHARD FRELING, DENNIS HUISMAN*AND ALBERT P.M. WAGELMANS

Econometric Institute, Erasmus University Rotterdam, The Netherlands

Econometric Institute Report EI2000-10/A

SUMMARY

This paper deals with models, relaxations and algorithms for an integrated approach to vehicle and crew scheduling. We discuss potential benefits of integration and provide an overview of the literature which considers mainly partial integration. Our approach is new in the sense that we can tackle integrated vehicle and crew scheduling problems of practical size.

We propose new mathematical formulations for integrated vehicle and crew scheduling problems and we discuss corresponding Lagrangian relaxations and Lagrangian heuristics. To solve the Lagrangian relaxations, we use column generation applied to set partitioning type of models. The paper is concluded with a computational study using real life data, which shows the applicability of the proposed techniques to practical problems. Furthermore, we also address the effectiveness of integration in different situations.

KEY WORDS: vehicle scheduling, crew scheduling, integrated planning, column generation, Lagrangian relaxation

*Correspondence to: Dennis Huisman, Econometric Institute, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, The Netherlands.
E-mail: huisman@few.eur.nl

1 INTRODUCTION

Although in the early eighties several researchers recognized the need to integrate vehicle and crew scheduling, most of the algorithms published in the literature still follow the sequential approach where vehicles are scheduled before, and independently of, crews. Algorithms incorporated in commercially successful computer packages use this sequential approach as well, while sometimes integration is dealt with at the user level (see e.g. [1]). In the operations research literature, only a few publications address a simultaneous approach to vehicle and crew scheduling. None of those publications makes a comparison between simultaneous and sequential scheduling. Hence, they do not provide any indication of the benefit of a simultaneous approach.

In this paper, we consider a complete integration of vehicle and crew scheduling. To evaluate the effectiveness of this approach, we also consider the traditional sequential approach and the opposite sequential approach of scheduling crews before, and independently of, vehicles.

The paper is organized as follows. Section 2 deals with problem definitions and discusses the need for integration of vehicle and crew scheduling. An overview of the literature on simultaneous vehicle and crew scheduling approaches is provided in section 3. In section 4, we propose mathematical formulations for complete integration of vehicle and crew scheduling and for independent crew scheduling. For the integration, separate approaches are developed for the situation where a crew is allowed to work on more than one vehicle during a duty, and for the situation where this is not allowed. The independent crew scheduling approach can serve as a stand alone algorithm, but it can also be used to measure the potential effectiveness of integrating vehicle and crew scheduling. This measure consists of comparing solution values of the traditional sequential approach with solution values of independent vehicle and crew scheduling.

All algorithms proposed in this paper rely on column generation applied to set partitioning type of models (see e.g. Freling [2] and Desrochers and Soumis [3]). In section 5, we discuss algorithms for solving the models. We use Lagrangian relaxations and Lagrangian heuristics with column generation and we discuss quality guarantees for both the integration and independent crew scheduling. The column generation (pricing) problem, topic of section 6, allows us to exploit the network flow structure of the crew scheduling problem, and is decomposed in two or three phases, each consisting of (constrained) shortest path type of problems. The paper is concluded with a computational study using real life data from a mass transit company in the Netherlands (section 7).

2 INTEGRATED VEHICLE AND CREW SCHEDULING

In this section we define the vehicle and crew scheduling problem and we discuss the potential benefits of an integrated approach to this problem.

2.1 Problem Definition

The *vehicle and crew scheduling problem* (VCSP) is the following: given a set of service requirements or *trips* within a fixed planning horizon, find a minimum cost schedule for the vehicles and the crews, such that both the vehicle and the crew schedule are feasible and mutually compatible. Each trip has fixed starting and ending times, and the traveling times between all pairs of locations are known. A vehicle schedule is feasible if (1) each trip is assigned to a vehicle, and (2) each vehicle performs a feasible sequence of trips, where a sequence of trips is feasible if it is feasible for a vehicle to execute each pair of consecutive trips in the sequence. From a vehicle schedule it follows which trips have to be performed by the same vehicle and this defines so-called vehicle *blocks*. The blocks are subdivided at *relief points*, defined by location and time, where and when a change of driver may occur. A *task* is defined by two consecutive relief points and represents the minimum portion of work that can be assigned to a crew. These tasks have to be assigned to crew members. The tasks that are assigned to the same crew member define a crew *duty*. Together the duties constitute a crew schedule. Such a schedule is feasible if (1) each task is assigned to one duty, and (2) each duty is a sequence of tasks that can performed by a single crew, both from a physical and a legal point of view. In particular, each duty must satisfy several complicating constraints corresponding to work load regulations for crews. Typical examples of such constraints are maximum working time without a break, minimum break duration, maximum total working time, and maximum duration. The cost of a duty is usually a combination of fixed costs such as wages, and variable costs such as overtime payment.

We make the following assumptions:

1. The vehicle scheduling characteristics correspond to one depot, identical vehicles, and no time constraints. It is well known that the *single depot vehicle scheduling problem* with these characteristics is polynomially solvable.

2. The cost function for the VCSP is the summation of the vehicle and crew scheduling cost functions, where the primary vehicle scheduling objective is to minimize the number of vehicles, while the primary crew scheduling objective is to minimize the number of crews. An additional term can be added to the cost function which corresponds

to variable vehicle costs (e.g. distance travelled) and variable crew costs (e.g. overtime payment).

3. The fact whether or not a sequence of tasks on one vehicle block can be performed by a single crew member without interruption only depends on the duration of this sequence. Such a feasible sequence is called a *piece (of work),* and the duration is limited by a minimum and maximum piece length.

The three assumptions make our approach in principle applicable to bus and driver scheduling.

We distinguish between two types of tasks, viz., *trip tasks* corresponding to (parts of) trips, and *dh-tasks* corresponding to deadheading. A *deadhead* is a period that a vehicle is moving to or from the depot, or a period between two trips that a vehicle is outside of the depot (possibly moving without passengers). All trip tasks need to be covered by a crew, while the covering of dh-tasks depends on the vehicle schedules and determines the compatibility between vehicle and crew schedules. In particular, each dh-task needs to be assigned to a crew if and only if its corresponding deadhead is assigned to a vehicle. Note that more than one trip task may correspond to a single trip, depending on the relief points along that trip. Similarly, more than one dh-task may correspond to a single deadhead. For example, a deadhead between the end location of trip i and the starting location of trip j which passes by the depot corresponds to two dh-tasks, one from the end location of trip i to the depot and the other from the depot to the strating location of trip j. Here we assume that waiting at the depot is not a task because vehicle attendance at the depot is not necessary.

2.2 Potential Benefits of Integration

It is an obvious observation that the specification of vehicle schedules will put certain constraints on the crew schedules (and vice versa). Because vehicles are often much more flexible to schedule than crews, it may be inefficient to schedule vehicles without considering crew scheduling. An overview of the potential benefits of integration is provided in Freling, Wagelmans and Paixão [4].

We can measure the potential benefit of integration with respect to cost efficiency, without solving the VCSP. To this end, we define the *independent crew scheduling problem* (ICSP) as follows: given a set of trip tasks corresponding to a set of trips, and given the traveling times between each pair of locations, find a minimum cost crew schedule such that all trip tasks are covered in exactly one duty and all duties satisfy crew feasibility constraints. Note that vehicles are completely ignored in this problem. Now consider the solution values of the following three approaches:

1. Traditional sequential approach (solution value v_1): first solve the SD-VSP and then the CSP.

2. Independent approach (solution value v_2): independently solve the SD-VSP and the ICSP.

3. Integrated approach (solution value v_3): solve the VCSP.

The solutions of the independent approach are of no practical use since the resulting vehicle and crew schedules are usually not compatible. However, it is easier to obtain solutions for the first two approaches than for the third approach, and we know that $v_2 \leq v_3 \leq v_1$. Thus, if $v_2 \ll v_1$, it may be that the crew scheduling solution will improve significantly when considering integration as compared to the sequential approach. On the other hand, we know that there is no need to integrate if v_2 and v_1 do not differ much. Because the vehicle schedules are identical for the sequential and independent approach, the potential benefit can be measured by comparing the the cost of the crew schedules only.

3 LITERATURE REVIEW

The traditional sequential strategy is strongly criticized by Bodin et al. [5]. This is motivated by the fact that in North American mass transit organizations the crew costs dominate vehicle operating costs, and in some cases reach as high as 80% of total operating costs. Although simultaneous vehicle and crew scheduling is of significant practical interest, only a few approaches of this kind have been proposed in the literature. They mainly deal with bus and driver scheduling and fall into one of the following three categories:

1. Scheduling of vehicles during a heuristic approach to crew scheduling.

2. Inclusion of crew considerations in the vehicle scheduling process; the actual crew scheduling is only carried out afterwards.

3. Complete integration of vehicle and crew scheduling.

Most of the approaches are of the first category and are based on a heuristic procedure proposed by Ball et al. [6]. This procedure involves the definition of a scheduling network, which consists of vertices characterized by parts of trips called *d-trips* that have to be executed by one vehicle and crew, and two vertices s and t representing the depot. Several types of arcs can be grouped into two categories, those which indicate that a crew and vehicle proceed from one d-trip to another and those which indicate that only the crew proceeds from one d-trip to another (*crew-only* arcs). The solution procedure is decomposed into three components, emphasizing the crew

scheduling problem: a piece construction component, a piece improvement component and a duty generation component. All three components are solved using matching algorithms. The piece construction routine generates a set of pieces whose time duration is less than some constant T. (Note that this corresponds to vehicle scheduling with time constraint.) In the second step pairs of short pieces are combined into partial duties, while in a third step pairs of these duties and longer pieces are combined into 2 and 3-piece duties. Vehicle schedules are generated simultaneously by deleting the crew-only arcs and fixing arcs used by pieces in the solution. This procedure is applied to large size VCSP instances which correspond to the entire physical network, i.e., all lines are considered at once, while no restrictions are placed on interlining, i.e., a crew may work on an arbitrary number of lines.

Other heuristic approaches of the first category are proposed by Tosini and Vercellis [7], Falkner and Ryan [8], and Patrikalakis and Xerocostas [9]. All these approaches use a similar crew scheduling network as in [6].

Approaches of the second category are proposed by Darby-Dowman et al. [1] as an interactive part of a decision support system, and by Scott [10] who heuristically determines vehicle schedules which take crew costs into account. An initial vehicle schedule is heuristically modified according to estimated marginal costs associated with a small change in the current vehicle schedule. The estimated marginal costs are obtained by solving the linear programming dual of the HASTUS crew scheduling model (see [11]). Results obtained with public transport scheduling problems in Montréal, show a slight decrease in estimated crew costs.

Only very recently, approaches of the third category, that is, a complete integration of vehicle and crew scheduling, have been proposed. The first mathematical formulation is proposed by Freling, Boender and Paixão [12]. This formulation is similar to the one considered in this paper (see also Freling, Wagelmans and Paixão [4]). Haase and Friberg [13] propose an approach of an exact algorithm for the VCSP. Both the vehicle and crew scheduling aspects are modelled by using set partitioning type of constraints. A *branch-and-cut-and-price* algorithm is proposed, that is, column generation and cut generation are combined in a branch-and-bound algorithm. The column generation master problem corresponds to the LP relaxation, while the pricing problem corresponds to a shortest path problem for generating vehicle schedules and a constrained shortest path problem for generating duties. The constrained shortest path problem is solved using the algorithm proposed by Desrochers [14]. Preliminary computational results indicate that problems with 10 trips can be solved within one minute, problems with 20 trips can be solved in more than an hour, and problems with 30 trips could not be solved.

Haase, Desaulniers and Desrosiers [15] propose an approach which solves a crew scheduling problem that incorporates side constraints for the buses.

This is done in such a way that the solution of this problem guarantees that an overall optimal solution is found after constructing a compatible vehicle schedule. The solution approach is based on a multi-commodity network flow formulation for the crew scheduling problem with side constraints, which is solved by a branch-and-price algorithm. For larger problem instances a heuristic version of the algorithm is used. Computational experiments with random data instances, simulating a urban mass transit environment, show that instances with up to 200 trips can be solved within 40 minutes of CPU time (on a SUN ULTRA-2/2300 workstation) and with an optimality gap of less than 3.1%. Furthermore, out of 10 instances with 120 trips, 5 instances could be solved to optimality within 1 hour of CPU time. We note, however, that Haase, Desaulniers and Desrosiers make the simplifying assumption that every duty has to start and end at the depot. Because of this and several other reasons (see section 7) it is not straightforward to compare their results with ours.

Finally, Gaffi and Nonato propose a heuristic algorithm based on Lagrangian relaxation with column generation for an integrated approach to multiple-depot vehicle scheduling and crew scheduling. They also propose a mathematical formulation similar to the formulation presented in this paper, with some modifications for the multiple-depot case. Their approach is developed for the ex–urban mass transit setting, where crews are tightly dependent on the vehicle activities or dead-heading of crew is highly constrained. Computational results are provided for Italian public transit operators, which show some improvements over the results of a sequential approach. CPU times (on a Power PC 604, 180 Mh) are over 24 hours for cases with up to 257 trips, and on average 2 to 6 hours.

4 MATHEMATICAL FORMULATIONS

In this section, we propose and review mathematical formulations for the VSP, CSP, VCSP and ICSP. For the VCSP we deal separately with the situation where changeovers are not allowed. A *changeover* is the change of vehicle of a bus driver during his break. An important consideration when formulating the VCSP or ICSP is the way crew tasks are defined without knowing the vehicle schedules in advance. Recall that we consider two types of tasks, namely a set of trip tasks of which we can be sure that they have to be serviced by a crew, and a set of dh-tasks (task between two trips or to/from the depot) that need to be covered by a crew if and only if a vehicle is traversing this deadhead.

4.1 Vehicle scheduling

Let $N = \{1, 2, ..., n\}$ be the set of trips, numbered according to increasing starting time, and let $E = \{(i, j) \mid i < j,\ i, j \text{ compatible},\ i \in N, j \in N\}$ be the set of deadheads. Let s and t both represent the depot at location d. We define the vehicle scheduling network $G = (V, A)$, which is an acyclic directed network with nodes $V = N \cup \{s, t\}$, and arcs $A = E \cup (s \times N) \cup (N \times t)$. A path from s to t in the network represents a feasible schedule for one vehicle, and a complete feasible vehicle schedule is a set of disjoint paths from s to t such that each node in N is covered. Let c_{ij} be the vehicle cost of arc $(i, j) \in A$, which is usually some function of travel and idle time. Furthermore, a fixed cost K for using a vehicle can be added to the cost of arcs (s, i) or (j, t) for all $i, j \in N$. For the remainder of this chapter, we assume that the primary objective is to minimize the number of vehicles. This means that K is large enough to guarantee that this minimum number will be achieved.

Using decision variables y_{ij}, with $y_{ij} = 1$ if a vehicle covers trip j directly after trip i, $y_{ij} = 0$ otherwise, the SDVSP can be formulated as follows as an quasi-assignment problem:

$$\min \sum_{(i,j) \in A} c_{ij} y_{ij} \tag{1}$$

$$\sum_{\{j:(i,j) \in A\}} y_{ij} = 1 \qquad \forall i \in N, \tag{2}$$

$$\sum_{\{i:(i,j) \in A\}} y_{ij} = 1 \qquad \forall j \in N, \tag{3}$$

$$y_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A. \tag{4}$$

In terms of the underlying network, $y_{ij} = 1$ means that the corresponding arc $(i, j) \in A$ is used by a vehicle. Constraints (2) and (3) assure that each trip is assigned to exactly one predecessor and one successor, that is, these constraints guarantee that the network is partitioned into a set of disjoint paths from s to t. The binary condition on the variables can be relaxed to $y_{ij} \geq 0$ because it is well-known that a simplex algorithm produces optimal 0/1-solutions for the LP-relaxation.

A bottleneck for solving VSP when using a network such as G is the large size of those networks due to a large number of arcs in E. When a vehicle has an idle time between two consecutive trips which is long enough to let it to the depot, we assume that is does so. In that case the arc between the trips is called a *long arc;* the other arcs between trips are called *short arcs.*

4.2 Crew Scheduling

In this subsection we give a mathematical formulation for the crew scheduling problem. Let d_k be the cost of duty $k \in K$, where K is the set of all

feasible duties, and define $K(i) \in K$ as the set of duties covering task $i \in I$. Consider binary decision variable x_k indicating whether duty k is selected in the solution or not. In the *set partitioning formulation* of the CSP, the objective is to select a minimum cost set of feasible duties such that each task is included in exactly one of these duties. This is the following 0-1 linear program:

$$\min \sum_{k \in K} d_k x_k \tag{5}$$

$$\sum_{k \in K(i)} x_k = 1 \qquad \forall i \in I, \tag{6}$$

$$x_k \in \{0,1\} \qquad \forall k \in K, \tag{7}$$

where constraints (6) assure that each task will be covered by exactly one duty. When solving the CSP, one often replaces constraints (6) by

$$\sum_{k \in K(i)} x_k \geq 1 \qquad \forall i \in I. \tag{8}$$

This yields the *set covering formulation* of the problem. The advantage of working with this formulation is that it is easier to solve than the set partitioning formulation (a similar remark can be made for relaxations of the respective problems). After solving the set covering formulation, we can always change the solution into a set partitioning solution by deleting over-covers of trips. In fact, this boils down to changing some of the selected duties. Instead of being the driver, the person who is assigned to such a duty will make the trip as a passenger. Note that such changes affect neither the feasibility nor the cost of the duties involved.

4.3 Complete Integration: General Case

Patrikalakis and Xerocostas presented in [9] the first mathematical formulation for the VCSP. This is a simplified version of the model presented here. Our formulation is valid under the assumptions that short arcs correspond to only one dh-task and *continuous attendance* is required, i.e., there is always a driver present if the bus is outside the depot. If these assumptions are not met, we can use a slightly different formulation; for details we refer to [16].

The mathematical formulation we propose for the VCSP is a combination of the quasi-assignment formulation for vehicle scheduling based on network $G = (V, A)$ defined in subsection 4.1, and the set partitioning formulation for crew scheduling discussed in subsection 4.2. The quasi-assignment part assures the feasibility of vehicle schedules, while the set partitioning part assures that each trip task is assigned to a duty and each dh-task is assigned to a duty if and only if its corresponding deadhead is part of the vehicle

schedule. Before providing the mathematical formulation, we need to recall and to introduce some notation. As before, N denotes the set of trips, K denotes the set of duties, and $A^s \subset A$ and $A^l \subset A$ denote the sets of short and long arcs, respectively. We assume that deadheads to and from the depot correspond to one dh-task each. Suppose $(i,j) \in A^l$ and let el_i and bl_j denote the ending and starting location of trips i and j, respectively. Then we let $K(i,t)$ and $K(s,j)$ denote the dh-task from el_i to the depot and from the depot to bl_j, respectively. Furthermore, I_1 denotes the set of trip tasks and $K(p)$ is the set of duties covering trip task $p \in I_1$, where we assume that a trip corresponds to one task. $K(i,j)$ denotes the set of dh-tasks corresponding to deadhead $(i,j) \in A^s$. Decision variables y_{ij} and x_k are defined as before, that is, y_{ij} indicates whether a vehicle covers trip j directly after trip i or not, while x_k indicates whether duty k is selected in the solution or not. The VCSP can be formulated as follows.

(VCSP1):

$$\min \sum_{(i,j)\in A} c_{ij} y_{ij} + \sum_{k\in K} d_k x_k \tag{9}$$

$$\sum_{\{j:(i,j)\in A\}} y_{ij} = 1 \qquad \forall i \in N, \tag{10}$$

$$\sum_{\{i:(i,j)\in A\}} y_{ij} = 1 \qquad \forall j \in N, \tag{11}$$

$$\sum_{k\in K(p)} x_k = 1 \qquad \forall p \in I_1, \tag{12}$$

$$\sum_{k\in K(i,j)} x_k - y_{ij} = 0 \qquad \forall (i,j) \in A^s, \tag{13}$$

$$\sum_{k\in K(i,t)} x_k - y_{it} - \sum_{\{j:(i,j)\in A^l\}} y_{ij} = 0 \qquad \forall i \in N, \tag{14}$$

$$\sum_{k\in K(s,j)} x_k - y_{sj} - \sum_{\{i:(i,j)\in A^l\}} y_{ij} = 0 \qquad \forall j \in N, \tag{15}$$

$$x_k, y_{ij} \in \{0,1\} \qquad \forall k \in K, \forall (i,j) \in A. \tag{16}$$

As before, the objective coefficients c_{ij} and d_k denote the vehicle cost of arc $(i,j) \in A$, and the crew cost of duty $k \in K$, respectively. The objective is to minimize the sum of total vehicle and crew costs. The first two sets of constraints, (10) and (11), correspond to the quasi-assignment formulation for the SDVSP discussed in subsection 4.1. Constraints (12) assure that each trip task p will be covered by one duty in the set $K(p)$. Furthermore, constraints (13), (14) and (15) guarantee the link between dh-tasks and deadheads in the solution, where deadheads corresponding to short and long arcs in A are considered separately. In particular, constraints (13) guarantee that each deadhead from i to j is covered by a duty in the set $K(i,j)$

if and only if the corresponding short arc is in the vehicle solution. The other two constraint sets, (14) and (15), ensure that the dh-tasks from el_i to t and from s to bl_j, possibly corresponding to long arc $(i,j) \in A$, are both covered by one duty if and only if the corresponding deadheads are in the solution. Note that the structure of these last three sets of constraints is such that each constraint corresponds to the possible selection of one duty from a large set of duties, where the fact whether or not a duty has to be selected depends on the values of the corresponding y variables. We will refer to these constraints as *partitioning type of constraints.* The model contains $|A| + |K|$ variables and $4|N| + |A^s| + |I_1|$ constraints, which is already a huge number of variables and a very large number of constraints for instances with a small number of trips. This is probably the main reason why complete integration has previously received very little attention.

4.4 Complete Integration: No Changeovers

The formulation presented in the preceding subsection is valid irrespective of the fact whether or not changeovers are allowed. It only differs in the set of feasible duties K. However, if changeovers are not allowed, we can also use a different formulation. We define a *combined duty* as a feasible vehicle duty *and* one or more corresponding feasible crew duties. The compatibility of vehicle and crew duties is guaranteed by the definition of a combined duty. Therefore, the VCSP without changeovers can be modelled as a set partitioning problem, where each trip must be covered by a combined duty. The cost of a combined duty k, denoted by c_k, is defined as the sum of the cost of the vehicle duty and the costs of the corresponding crew duties. Let N be the set of trips, K the set of all combined duties, and $K(i)$ the set of combined duties covering trip i. Furthermore, binary variable x_k indicates whether combined duty k is selected in the solution or not. The VCSP without changeovers can be formulated as follows.

(VCSP2):

$$\min \sum_{k \in K} c_k x_k \tag{17}$$

$$\sum_{k \in K(i)} x_k \geq 1 \qquad \forall i \in N, \tag{18}$$

$$x_k \in \{0,1\} \qquad \forall k \in K, \tag{19}$$

Note that, for similar reasons as mentioned in subsection 4.2, we use a set covering formulation, i.e., constraints (18) require that each trip must be covered by a combined duty, but possibly more than once.

4.5 Independent Crew Scheduling

For the ICSP we propose the same formulation as for the CSP, but the difference is that the set of possible duties is much larger, because the vehicle schedule is not given in advance. Furthermore, we know that some duties will never appear in the optimal solution. For example, consider two duties i and j which are equivalent except that j contains an extra dh-task from the depot to the first relief point. Then, duty j will never appear in an optimal solution if $d_i < d_j$. In general, if the cost of a duty increases with the duration, in an optimal solution no duty begins or ends at the depot. So vehicle movements to and from the depot are not covered. In particular, only duties beginning and ending with a trip task need to be considered.

5 ALGORITHMS

We propose Lagrangian heuristics with column generation for obtaining feasible solutions with quality guarantee for models CSP, VCSP1, VCSP2 and ICSP. We assume that the reader is already familiar with the techniques of Lagrangian relaxation and subgradient optimization (survey on these topics are [17], [18] and [19]). In subsection 5.1 we give a general introduction to column generation and we give a motivation for using a combination of Lagrangian relaxation and column generation. In the subsequent subsections we discuss the Lagrangian relaxation and heuristic for each problem. Details on the solution of the column generation problems are postponed to section 6.

5.1 Column generation in combination with Lagrangian relaxation

Column generation is a technique that is used for problems with a huge number of variables (see e.g. Barnhart et al. [20]). The general idea is to solve a sequence of reduced problems, where each reduced problem contains only a small portion of the set of variables (columns). After solving a reduced problem a new set of columns is obtained by using dual information of the solution. The column generation algorithm converges once it is established that the optimal solution based in the current set of columns can not be improved upon by adding more columns. Then the optimal solution of the reduced problem is the optimal solution of the overall problem. We will refer to the reduced problem as the *master problem* and to the problem of generating a new set of columns as the *pricing problem.*

Usually column generation is used in the context of linear programming, but here we will use it in combination with Lagrangian relaxation (see also

Freling [2] and Carraresi et al. [21]) to compute lower bounds on the optimal solution. Furthermore, we will use the columns which are generated to compute the lower bound to construct a feasible solution. This approach has been chosen to solve CSP, VCSP1, VSCP2 and ICSP for the following reasons:

1. in general, a set of columns generated to compute the lower bound turns out to be a set from which we can select a good feasible solution;
2. since we compute a lower bound on the optimal solution, we obtain an indication about the quality of the constructed feasible solution;
3. Lagrangian relaxation has shown to provide tight bounds for set partitioning type of problems (see e.g. Beasley [19]);
4. because of the number of constraints, using a linear programming relaxation is not a realistic option for the VCSP1 model.

We mention also that Carraresi et al. [21] proposed a similar algorithm for the CSP. The only difference between their and our approach is that in subsequent reduced problems they replace the set of columns by a completely new one, while we only add columns.

5.2 CSP

Before we can solve the CSP, we have to solve the VSP and to generate all feasible pieces and (optional) all feasible duties first. To solve the VSP, we use a very efficient auction algorithm, which is described in detail in Freling, Wagelmans and Paixão [22].

As mentioned before, we solve the CSP as a set covering problem. A global description of the Lagrangian heuristic is given in figure 1.

Suppose that, at some point, K is the set of columns (duties) that we are considering in the master problem. We compute the lower bound with respect to these columns as follows. Associate non-negative Lagrangian multipliers $\lambda_1, \lambda_2, ..., \lambda_{|I|}$ with the constraints (8). This yields the following Lagrangian subproblem:

$$\begin{aligned} \Phi(\lambda) &= \min \sum_{k \in K} d_k x_k - \sum_{i \in I} \lambda_i \left(\sum_{k \in K(i)} x_k - 1 \right) \\ &= \min \sum_{k \in K} \overline{d}_k x_k + \sum_{i \in I} \lambda_i \qquad (20) \\ & \qquad x_k \in \{0,1\}, \qquad \forall k \in K, \qquad (21) \end{aligned}$$

where $\overline{d}_k = d_k - \sum_{i \in I(k)} \lambda_i$ is the *Lagrangian cost* or *reduced cost* with respect to the Lagrangian multipliers $\lambda = (\lambda_1, \lambda_2, ..., \lambda_{|I|})$, and $I(k)$ is the set of tasks

Step 0: Initialization
Generate a set of pieces such that each task can be covered by at least one piece. The initial set of columns consists of these pieces.
Step 1: Computation of lower bound
Solve a Lagrangian dual problem with the current set of columns.
Step 2: Generation of columns
Generate columns with negative reduced cost. If no such columns exist (or another termination criterion is satisfied), go to Step 3;
otherwise, return to Step 1.
Step 3: Construction of feasible solution
Use all the columns generated in Step 0 and Step 2 to construct a feasible solution.

Figure 1: Solution method for CSP

in duty $k \in K$. Let $\overline{x}(\lambda)$ be an optimal solution to this problem for a given λ. Then the problem amounts to *pricing out* the x variables to yield $\overline{x}(\lambda)$. That is, for each $k \in K$, $\overline{x}_k = 1$ if $\overline{d}_k \leq 0$ and $\overline{x}_k = 0$ otherwise.

We use subgradient optimization to solve the Lagrangian dual problem $\max_{\lambda \geq 0} \Phi(\lambda)$ approximately. Besides a good lower bound, we also obtain a final set of Lagrange multipliers. These are used to compute the reduced cost $\overline{d}_k$ of columns $k \notin K$ in the column generation step. We add a number of duties with negative reduced cost to K and repeat the subgradient optimization. Actually, to assure that we do not generate columns twice, we modify the Lagrange multipliers before we generate new columns. The modifications are such that all columns already in K get non-negative reduced cost. For details on the greedy heuristic that we use, we refer to [2].

We stop if there are no duties left with negative reduced cost or if the lower bound obtained by the subgradient optimization does not significantly change for a number of iterations. In the first case we obtain a true lower bound; in the second case we do not have this guarantee.

Finally we compute a feasible solution by solving a set covering problem in which we consider all the collumns which have been generated along the way. We can either do this exactly, using a general or specialized integer programming solver, or heuristically. Most often we use the set covering heuristic of Caprara et al. [23] with a few changes proposed in Freling [2].

5.3 VCSP1

The algorithm for VCSP1 is shown in figure 2.

We use the relaxation of model VCSP1, where all set partitioning type of constraints (12)-(15) are first replaced by set covering constraints, which are subsequently relaxed in a Lagrangian way. That is, we associate non-negative

Step 0: Initialization
Solve VSP and CSP (using the algorithm of figure 1) and
take as initial set of columns the duties in the CSP-solution.
Step 1: Computation of lower bound
Solve a Lagrangian dual problem with the current set of columns.
Step 2: Generation of columns
Generate columns with negative reduced cost.
Compute an estimate of a lower bound for the overall problem.
If the gap between this estimate and the lower bound found in Step 1 is small enough (or another termination criterion is satisfied), go to Step 3; otherwise, return to Step 1.
Step 3: Construction of feasible solution
Based on feasible vehicle solution(s) from Step 1, construct
corresponding feasible crew solution(s) using the algorithm of figure 1

Figure 2: Solution method for VCSP1

Lagrangian multipliers λ_p, μ_q, ν_i and ξ_j with constraints (12), (13), (14) and (15), respectively. Furthermore, we define the corresponding $|I_1|$-vector λ, $|A^s|$-vector μ, $|N|$-vector ν, and $|N|$-vector ξ. The Lagrangian subproblem becomes:

$$\Phi(\lambda, \mu, \nu, \xi) = \Phi_x(\lambda, \mu, \nu, \xi) + \Phi_y(\mu, \nu, \xi) + \sum_{p \in I_1} \lambda_p$$

with

$$\Phi_x(\lambda, \mu, \nu, \xi) = \min \sum_{k \in K} \overline{d}_k x_k \tag{22}$$

$$x_k \in \{0, 1\} \qquad \forall k \in K, \tag{23}$$

and

$$\Phi_y(\mu, \nu, \xi) = \min \sum_{(i,j) \in A} \overline{c}_{ij} y_{ij} \tag{24}$$

$$\sum_{\{j:(i,j) \in A\}} y_{ij} = 1 \qquad \forall i \in N, \tag{25}$$

$$\sum_{\{i:(i,j) \in A\}} y_{ij} = 1 \qquad \forall j \in N, \tag{26}$$

$$y_{ij} \in \{0, 1\} \qquad \forall (i, j) \in A, \tag{27}$$

where

$$\overline{c}_{ij} = \begin{cases} c_{ij} + \mu_{ij} & \forall (i,j) \in A^s, \\ c_{ij} + \nu_i + \xi_j & \forall (i,j) \in A^l, \\ c_{ij} + \nu_i & j = t, \forall i \in N, \\ c_{ij} + \xi_j & i = s, \forall j \in N, \end{cases}$$

is the reduced cost of arc $(i,j) \in A$, and

$$\overline{d}_k = d_k - \sum_{p \in I_1(k)} \lambda_p - \sum_{(i,j) \in A^s(k)} \mu_{ij} - \sum_{i \in N^t(k)} \nu_i - \sum_{j \in N^s(k)} \xi_j,$$

is the reduced cost of duty $k \in K$, where the sets are defined as follows:

- $I_1(k)$ denotes the set of trip tasks I_1 in duty k.
- $A^s(k)$ denotes the set of short arcs A^s in duty k.
- $N^t(k)$ denotes the set of trips that have a corresponding task in duty k, with the end of this trip as starting location and the depot as ending location.
- $N^s(k)$ denotes the set of trips that have a corresponding task in duty k, with the depot as starting location and the start of this trip as ending location.

Let $\overline{x}(\lambda, \mu, \nu, \xi)$ and $\overline{y}(\mu, \nu, \xi)$ denote optimal solutions corresponding to $\Phi_x(\lambda, \mu, \nu, \xi)$ and $\Phi_y(\mu, \nu, \xi)$, respectively, for given λ, μ, ν and ξ. Then $\overline{y}(\mu, \nu, \xi)$ is obtained by solving an instance of the SDVSP, and $\overline{x}(\lambda, \mu, \nu, \xi)$ is obtained by pricing out each variable, that is, for each $k \in K$, $\overline{x}_k = 1$ if $\overline{d}_k \leq 0$ and $\overline{x}_k = 0$ otherwise. We can again use the auction algorithm proposed in Freling, Wagelmans and Paixão [22] for solving the SDVSP. Notice that we get a feasible vehicle solution every time we solve the SDVSP.

As before, we need an additional procedure to update the Lagrangian multipliers after solving the Lagrangian relaxation. This is necessary to assure that all duties in the current master problem have non-negative reduced cost so that these duties will not be generated again in the pricing problem.

In contrast to our approach to the CSP, we cannot generate the complete set of duties before we run the actual optimization algorithm, because we do not have a vehicle solution in advance. Therefore, we have to generate the duties during the process. Details on how this is done will be given in section 6.

In every iteration i we compute an estimate LBT_i of the lower bound for the overall problem. Let LBS_i denote the value of the Lagrangian lower bound in iteration i, then the estimate is computed as

$$LBT_i = LBS_i + \sum_{k \in K_i} \overline{d}_k \tag{28}$$

where K_i is the set of duties added in iteration i.

LBT_i is a lower bound for the overall problem if all duties with negative reduced cost are added to the master problem. Of course, we can stop if LBT_i is equal to LBS_i, but in practice we stop earlier, namely if the relative

difference is small or if there is during a number of iterations no significant improvement in LBS_i.

At the end we compute a feasible crew schedule given the (feasible) vehicle schedule which resulted from solving the last Lagrangian subproblem. We do this by solving the CSP (using the algorithm described in in figure 1). Of course, it is also possible to compute more feasible solutions by solving the CSP not only for the vehicle solution from the last iteration, but also for vehicle solutions which were encountered earlier on. A reason to actually do this could be that the gap between the lower and upper bound is quite large, which is an indication that the upper bound could be improved upon.

5.4 VCSP2, ICSP

A global description of the solution method for solving the models VCSP2 and ICSP is given in figure 3.

Step 0: Initialization
Solve VSP and CSP without changeovers and take as initial set of columns the duties in the CSP-solution.
Step 1: Computation of lower bound
Solve a Lagrangian dual problem with the current set of columns.
Step 2: Generation of columns
Generate columns with negative reduced cost.
Compute an estimate of a lower bound for the overall problem.
If the gap between this estimate and the lower bound found in Step 1 is small enough (or another termination criterion is satisfied), go to Step 3; otherwise, return to Step 1.
Step 3: Construction of feasible solution
Use all the columns generated in Steps 0 and 2 to construct a feasible solution.

Figure 3: Solution method for VCSP2 and ICSP

For the model VCSP2 we have to take in account that no changeovers are allowed in the initial solution. We do this by solving the CSP where the set of duties consists only of duties without changeovers. Since models VCSP2 and ICSP are pure set covering models, Lagrangian relaxations can be obtained in a similar way as discussed in subsection 5.2. For the model VCSP2 we must generate combined duties instead of duties. How this can be done, will be discussed in detail in section 6. We obtain a feasible solution in a similar way as for CSP by solving a set covering problem.

6 THE COLUMN GENERATION PROBLEM

For the CSP all feasible pieces of work can simply be enumerated on each vehicle block. In subsection 6.2, we will discuss how one can generate duties given a set of pieces.

For the VCSP and ICSP, vehicle blocks are not known and a huge number of feasible pieces of work may exist. Although Ball et al. [6] have proposed a duty generation network that does not consider pieces of work explicitly, this approach is not interesting for column generation purposes due to the size of the network. Therefore, for model VCSP1 and ICSP, we propose a two phase procedure for the column generation pricing problem: in the first phase, a *piece generation network* is used to generate a set of pieces of work which serve as input for the second phase where duties are generated. The second phase is equivalent to generating duties for the CSP.

The column generation pricing problem for model VCSP2 needs an additional procedure in order to generate combined duties using previously generated crew duties as input. This results in a three phase procedure. While the piece and duty generation networks are created only once, the combined duty generation network needs to be built every time combined duties are generated, using the duties generated previously.

Figure 4 demonstrates the procedures.

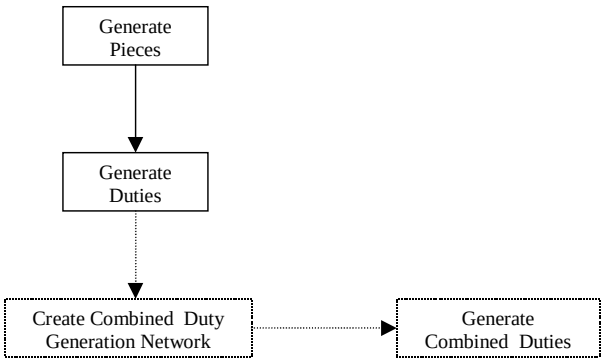

Figure 4: Column generation problem

A disadvantage of the two- or three-phase procedure is that it is possible to generate a duty twice in the second phase, and to generate a combined duty twice in the third phase. For example, a duty containing the same tasks can be constructed from two different pieces of work. It is very time consuming to check for each new duty generated if it has been generated before. However, we have implemented an efficient procedure to check for double duties after a fixed number of iterations of our duty generation algorithm. The current

list of generated duties is then updated so that it contains no double duties, and the algorithm for generating duties is continued.

Next, we discuss networks and algorithms for piece, duty and combined duty generation, respectively.

6.1 Generation of Pieces of Work

Recall that we have defined a piece of work as a continuous sequence of trip tasks and dh-tasks corresponding to (a part of) one vehicle block, and that this sequence of tasks is only restricted by its duration.

6.1.1 Network Structure

The network for piece generation is an extension of the network G for vehicle scheduling (see subsection 4.1). Let a *start point* (*end point*) be defined as the relief point corresponding to the start (end) of a vehicle trip. We define the acyclic network $G^p = (N^p, A^p)$, where nodes correspond to the relief points on each trip, and the source s and the sink t represent the depot. Arcs in A^p correspond to dh-tasks and trip tasks. Figure 5 illustrates network G^p with four trips and two trip tasks per trip, and one dh-task for each deadhead.

Let b_{ij} be the cost associated with each arc $(i,j) \in A^p$. Recall from subsection 5.3 that we associate Lagrangian multipliers λ_p, μ_q, ν_i and ξ_j with constraints (12), (13), (14) and (15), respectively. Then, for model VCSP1, the reduced cost is defined as

$$\overline{b}_{ij} = \begin{cases} b_{ij} - \lambda_p, \text{ for each arc } (i,j) \text{ corresponding to a trip task } p. \\ b_{ij} - \mu_{ij}, \text{ for each arc } (i,j) \text{ corresponding to a short arc } (i,j). \\ b_{ij} - \xi_r, \text{ for each arc } (i,j) \text{ with } i = s \text{ and } j \text{ the start point of trip } r. \\ b_{ij} - \nu_r, \text{ for each arc } (i,j) \text{ with } j = t \text{ and } i \text{ the end point of trip } r. \end{cases}$$

Let λ_p be the Lagrangian multiplier corresponding to a trip or trip task, and associated with the set covering constraints in model VCSP2 or ICSP. Then, for model VCSP2 or ICSP, the reduced cost is defined as

$$\overline{b}_{ij} = \begin{cases} b_{ij} - \lambda_p, \text{ for each arc } (i,j) \text{ corresponding to a trip or trip task } p. \\ b_{ij}, \text{ otherwise.} \end{cases}$$

Thus, the costs on the arcs are defined such that the cost of a path is equal to the reduced cost of the corresponding piece of work. Let tp_i be the point in time associated with node $i \in N^p \setminus \{s,t\}$. Each path $P(u,v)$ between two nodes u and v in network G^p corresponds to a feasible piece of work if its duration satisfies the time constraint

$$dur_{min} \leq tp_v - tp_u \leq dur_{max},$$

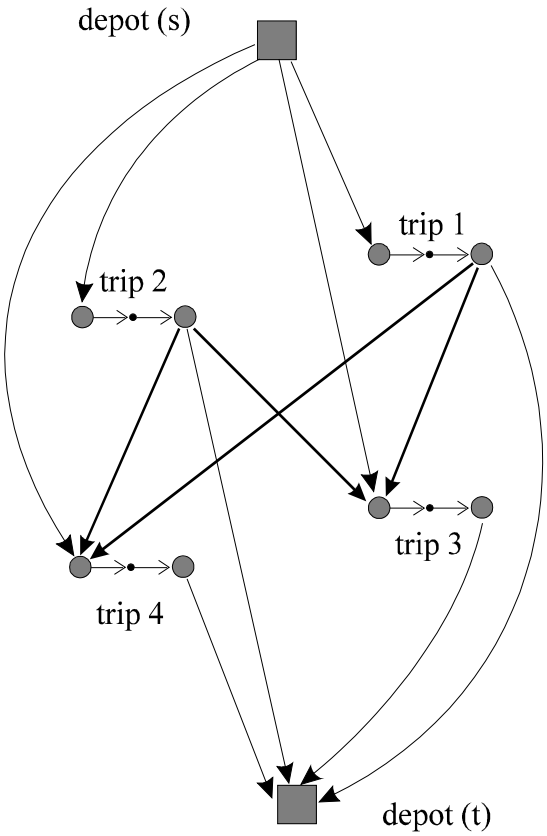

Figure 5: Example - network G^p

where dur_{min} and dur_{max} denote the minimum and maximum allowed duration of a piece of work, respectively. The duration of a piece of work starting at s and/or ending at t is determined by incorporating travel times. For example, a path $s, u, ..., v, t$ corresponds to a feasible piece of work if the inequalities

$$dur_{min} \leq tp_v - tp_u + trav(d, bl_u) + trav(el_v, d) \leq dur_{max}$$

hold, where $trav(d, bl_u)$ and $trav(el_v, d)$ denote the travel times(possibly including sign-on and sign-off times) from d to bl_u and from el_v to d, respectively.

6.1.2 An All-Pairs Shortest Path Algorithm

In this subsection, we propose a polynomial all-pairs shortest path algorithm for generating a set of pieces. As before, we assume that the nodes are numbered according increasing time. Hence, G^p is acyclic. A set of pieces is generated by solving a shortest path problem between each pair of nodes in

network G^p that satisfy the constraints on the piece duration. For all feasible paths $P(u,v) = u, ..., v$ between nodes $u \neq s$ and $v \neq t$ three additional paths are considered, namely path $s, u, ..., v$, path $u, ..., v, t$ and path $s, u, ..., v, t$. In some applications, all crew breaks need to be assigned to the depot and all crew starts and ends their duty at the depot. In this case, only paths of the form $s, u, ..., v, t$ are considered. The algorithm is shown in figure 6.

```
for i ∈ N^p\{s,t} do
begin
   Determine k = arg max{j ∈ N^p\{s,t} | tp_j − tp_i ≤ dur_max}.
   SHORTESTPATHS(i,k).
   RETRIEVEPIECES(i,k).
end.
```

Figure 6: Piece Generation Algorithm

The procedure SHORTESTPATHS(i,k) returns the shortest paths $P^*(i,j)$ from i to $j = i+1, ..., k$ in $O(m)$ time because G^p is acyclic, where m is the number of arcs in the network G^p. The procedure RETRIEVEPIECES(i,k) is shown in figure 7.

```
for j = i+1, ..., k do
begin
   if tp_j − tp_i ≥ dur_min then accept P*(i,j).
   if dur_min ≤ tp_j − tp_i + trav(d,bl_i) ≤ dur_max then accept s + P*(i,j).
   if dur_min ≤ tp_j − tp_i + trav(el_j,d) ≤ dur_max then accept P*(i,j) + t.
   if dur_min ≤ tp_j − tp_i + trav(d,bl_i) + trav(el_j,d) ≤ dur_max
      then accept s + P*(i,j) + t.
end.
```

Figure 7: Procedure RETRIEVEPIECES(i,k)

The term $s + P^*(i,j)$ denotes the extension of path $P^*(i,j)$ by adding s as the initial node, and the term $P^*(i,j) + t$ denotes the extension of path $P^*(i,j)$ by adding t as the terminal node. Accepted pieces are stored in a set of potential pieces to be used for the duty generation phase. The running time of procedure RETRIEVEPIECES(i,k) is $O(n)$, where n equals the number of trips. Thus, the overall running time of the piece generation algorithm is $O(nm)$.

When generating sets of pieces we have to assure that the column generation optimality condition is satisfied, that is, no negative reduced cost duties exist at convergence of the column generation procedure. To facilitate the discussion, we do not consider s and t as relief points. Hence, the first

relief point of a path $s, i, ..., j$ is the point corresponding to node i. Let S denote the set of pieces containing all pairs of shortest paths generated by the algorithm in figure 6. Furthermore, we assume that the duty generation algorithm satisfies the column generation optimality condition, that is, negative reduced cost duties are detected as long as corresponding paths exist in the duty generation network. Recall that we have assumed that the resource consumption feasibility of a piece of work depends on the starting and ending relief point only. We now have the following proposition.

Proposition 1 *If the duty generation algorithm with set S as the set of pieces does not return a negative reduced cost duty under the assumptions stated above, then the relaxation has been solved to optimality.*

Proof. Consider an arbitrary feasible duty with pieces not in S, and suppose that this duty has negative reduced cost. Any piece in this duty that is not in S can be replaced by a piece in S with the same starting and ending relief point and with lower or equal reduced cost, while the duty remains feasible due to the assumptions stated above. Thus, if at least one duty with negative reduced costs exist, then at least one such duty exists with pieces in S only. Hence, if no negative reduced cost duties are found with pieces in S only, then no negative reduced cost duties exist when considering all pieces, and this means that the relaxation has been solved to optimality. ■

6.2 Generation of duties

Duties consists of a number of pieces with a given maximum number of pieces. In practice this maximum is very often equal to 2 or 3. This is the reason why we simply enumerate all possible combinations of pieces and check if such a combination is feasible. (To generate duties consisting of more than 3 pieces, one may use a duty generation network, as proposed by Freling [2].) The reduced cost of a duty can be easily computed if the reduced cost of a piece is already known: the reduced cost of a duty is equal to the sum of the reduced cost of the pieces it is built from, plus the reduced cost of the breaks between the pieces.

6.3 Generation of Combined Duties

An additional procedure is necessary to generate combined duties for model VCSP2. The input for this procedure is a set of duties generated by the two phase procedure, that is, pieces of work are generated as described in subsection 6.1, while duties are generated as described in the preceding subsection . The purpose of the third phase is to find combined duties with negative reduced cost. This is done by constructing a combined duty generation network

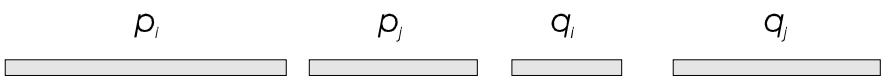

Figure 8: Example - combined duty with two interrupted crew duties

based on the given duties, and solving a sequence of (unconstrained) shortest path problems in this network. The negative reduced cost combined duties generated in this way are added to the master problem, while convergence of the column generation algorithm is attained once no negative reduced cost combined duties are obtained.

Combined duty generation network G is constructed as follows. Nodes correspond to the set of duties previously generated plus a source and a sink. An arc from the source to a duty node exists if the starting relief point of the duty is located at the depot, while an arc from a duty to the sink exists if the ending relief point of the duty is located at the depot. Furthermore, an arc between two duty nodes i and j exist if duty i and j form a compatible pair of duties on one vehicle, that is, if $el_i = bl_j$ and $et_i \leq bt_j$. In addition, an arc exists if two duties can serve together on one vehicle while interrupting each other. Suppose for example that two duties i and j each contain two pieces p_i, q_i and p_j, q_j, respectively. Then, arc (i, j) exists if pairs $(p_i, p_j), (p_j, q_i)$ and (q_i, q_j) form compatible pairs of pieces. This example is illustrated in figure 8.

Reduced vehicle trip costs and vehicle deadhead costs are incorporated in the reduced cost of pieces and duties. This is achieved by adjusting arc costs in the piece generation network. The costs of arc (i, j), $i \neq s$ in network G equals the reduced cost of duty j plus the vehicle cost of the corresponding deadhead. If $i = s$ then the fixed vehicle cost is added as well. A path starting at s and ending at t is denoted by an (s, t)-path. We can determine the combined duty corresponding to the shortest (s, t)-path in network G in $O(m)$ time, where m is the number of arcs. This shortest path is added to the master problem if its cost is negative, while the column generation procedure is terminated otherwise. In the first case, more shortest (s, t)-paths are determined for multiple pricing purposes. The nodes and arcs corresponding to the current shortest path are deleted from the network and the shortest path problem is resolved. This process is repeated until no (s, t)-path exists in the current network. The advantage of this strategy (in comparison to, for instance, deleting only one arc in the path) is that we generate really different combined duties in every iteration and that the generation process terminates very fast.

We next show that optimality can be guaranteed at convergence of the column generation procedure. Assume that the set S of pieces is generated

as described previously, that is, all-pairs shortest paths are included. Furthermore, assume that all feasible duties which can be constructed using the pieces in S are input for the combined duty generation.

Proposition 2 *If the shortest path in the combined duty generation network has non-negative reduced cost under the assumptions stated above, then the relaxation has been solved to optimality.*

Proof. The replacement argument in the proof of Proposition 1 remains valid here due to the assumptions. ■

7 COMPUTATIONAL TESTS FOR BUS AND DRIVER SCHEDULING

In this section we present a computational study of integrated bus and driver scheduling. The aim of this study is to investigate

1. the effectiveness of integration as compared to the traditional sequential approach;
2. the applicability of the proposed techniques in practice.

7.1 Data and parameter settings

We have used data from the RET, the public transport company in Rotterdam, the Netherlands, that provides passenger service by bus, metro and tram. The data corresponds to bus and driver scheduling of individual bus lines. Instead of using the actual restrictions that the driver duties at the RET have to satisfy, we have changed some RET specific restrictions into more general restrictions. In a second paper [24], we will consider the real RET restrictions, which will make the problem much more difficult.

The restrictions that we have taken into account, are as follows. A driver can only be relieved by another driver at the start or end of a trip or at the depot and continuous attendance is required. This implies that a trip always corresponds to exactly one trip task. If a driver starts/ends his duty at the depot, there is a sign-on/sign-off time of 11 and 12 minutes, respectively. Another restriction is that every bus waits at least 3 minutes after a trip at the end location before starting the next trip. There are four different types of duties and their properties are given in figure 9.

The following points are relevant for all the different approaches proposed in this section:

	Type 1 (Tripper)		Type 2 (Early Duty)		Type 3 (Normal Duty)		Type 4 (Late Duty)	
	minimum	maximum	minimum	maximum	minimum	maximum	minimum	maximum
number of pieces	1	1	2	2	2	2	2	2
start time (min.)				645	646	914	915	
end time (min.)				1177		1334		
piece length (min.)	30	300	30	300	30	240	30	300
break length (min.)			15	90	15	90	15	90
duty length (min.)	30	300		532		420		570
work time (min.)	30	300		532		420		570

Figure 9: Properties of all different types of duties

1. The primary objective is to minimize the sum of buses and drivers used in the schedule. For bus scheduling we have a secondary objective of minimizing a combination of travel and idle time as defined in subsection 4.1. For crew scheduling we do not have a secondary objective. We use fixed costs in order to assure that the minimum number of buses and drivers is the primary objective. For the computational results presented in this section, we only report the number of buses and drivers since these are relevant for our study.

2. The pricing problems are solved separately for each type of duty. We work with reduced networks for each type of duty, that is, only nodes corresponding to pieces of work are considered which could be part of a duty of a certain type.

3. The Lagrangian multipliers are initialized to zero. The maximum number of iterations $k_{\max}$ is intially set to 100 for problem instances with a small number of trips; for instances with a larger number of trips, we take for the independent approach $k_{\max} = 1000$. In every iteration of the column generation algorithm, after solving the master problem, $k_{\max}$ is increased by 3. A greedy heuristic is used to get only non-negative Lagrangian cost variables in the master problem. The initial values of the Lagrangian multipliers for a new master problem are set to the values resulting from the previous master problem, before the greedy heuristic was applied.

4. The column generation is stopped once the difference between the estimate of the lower bound for the overall problem and the current lower bound is less than one percent (0.1 percent for VCSP1) or if no improvement in the lower bound is obtained for 5 iterations (250 iterations for VCSP1). The latter is the so-called *tailing-off criterion.*

5. For the integrated approach with changeovers allowed, we generate 10 feasible solutions, where 5 follow from the last iterations of the column generation and the other 5 follow from the last iterations of a subgradi-

ent optimization after the convergence of the column generation step. We choose the best feasible solution. If the gap between the lower bound and the best feasible solution is large, we can easily adapt our approach to compute some additional feasible solutions in the last subgradient optimization, i.e., the one after the convergence of the column generation step.

6. The strategies for generating columns in the pricing problem have been obtained after extensive testing and tuning on different types of problems (see Freling [2]). We add for all types of duties the first 200 duties with negative reduced cost that we find. This means that we do not take the duties with the most negative reduced cost. The advantage of this strategy that we do not have to generate all the duties in all iterations. The disadvantage is that the convergence process goes slower, but overall this is faster than the alternative strategy.

7. All tests are executed on a Pentium 400 pc with 128Mb of computer memory.

We cannot compare our test results with the results of others, because — to the best of our knowledge — only Haase, Desaulniers and Desrosiers [15] perform a computational study for simultaneous scheduling of vehicles and crews, but they have different problems. There are four important differences between their random data and our real life data:

- their trip lengths are much longer than ours; on average they have between 2 and 3 trips per duty and about 4 per vehicle, whereas we have, for the largest problem, on average more than 10 trips per duty and 25 per vehicle;

- they consider a network of four lines, while we consider individual bus lines; note that this means that for the same total number of trips, the (average) frequency per line is higher for our problem instances;

- they assume that bus drivers are only allowed to start and end their duty at the depot, whereas in our case drivers can start and end at every relief point (i.e., at the start/end of every trip and at the depot);

- they have only one duty type with 2 pieces and one consisting of a single piece, while we consider three duty types with 2 pieces and one with a single piece.

7.2 Results of different approaches

In this section we show the results of the sequential, independent and integrated approach. For the sequential and the integrated approach we look at the cases with and without changeovers.

7.2.1 Sequential approach

In figure 10 we present the computational results for the sequential method with and without changeovers. The number of trips for prob24 is equal to 24 and so on.

	changeovers				no changeovers			
problem	low	upp	bus	driv	low	upp	bus	driv
prob24	12	12	6	6	12	12	6	6
prob42	19	19	9	10	19	20	9	11
prob72	15	15	5	10	15	15	5	10
prob113	23	24	8	16	26	26	8	18
prob148	34	34	11	23	45	45	11	34
prob238	27	29	9	20	30	33	9	24

Figure 10: Results sequential approach RET data

Because the intial step of the the integrated approach is the sequential approach, the difference in the computation time between the sequential and integrated approach is equal to the computation time of the lower and upper bound phase of the latter approach (which we will present later on). Moreover, the computation times of the sequential approach is negligible. For these reasons, we have not mentioned them here.

7.2.2 Independent approach

In figure 11 we show the computational results for the independent crew scheduling. The top part of the figure focuses on the lower bound phase and the bottom part on the upper bound phase. For the lower bound phase, we report the number of nodes and arcs in the piece generation network G^p, the number of column generation iterations, the average number of pieces of work during the column generation, the number of duties in the final master problem, the total computation times in seconds required for the master problems, the pricing problems, and the lower bound phase, and the resulting lower bound on the number of drivers in the solution, respectively. For the upper bound phase, we report the cost (number of drivers) of the feasible solution, the relative gap between lower and upper bounds and the computation times in seconds for the upper bound phase.

As can be seen from figure 11, the gap between the lower and upper bounds for the independent crew scheduling is zero for smaller problem instances, but it increases for the larger instances prob113 (7.14%), prob148

	prob24	prob42	prob72	prob113	prob148	prob238
nodes	48	84	144	226	296	476
arcs	132	312	460	846	1,376	2,260
iterations	5	6	9	9	15	20
average pieces	463	1,568	4,606	9,891	16,237	52,538
duties	529	1,689	6,026	9,329	28,208	70,307
cpu m.	0	0	3	7	1,261	1,845
cpu p.	1	8	41	136	6,907	14,422
cpu t.	1	8	44	178	8,169	16,274
lower	6	9	9	13	19	16
upper	6	9	9	14	22	18
gap (%)	0.00	0.00	0.00	7.14	13.64	11.11
cpu	0	0	0	50	53	52

Figure 11: Results independent approach RET data

(13.64%) and prob238 (11.11%). As mentioned before, for the larger problem instances we perform more iterations of the subgradient optimization than for the smaller instances. By doing so, we obain improved lower bounds for these problems. For example, for prob148 we improved the lower bound from 17 to 19. This is possible, because only a small part of the computational effort is in the master problem and the major part is in the pricing problem.

Recall from subsection 2.2 that we can measure the potential benefit of integration by comparing solutions obtained by the sequential approach with solutions obtained by the independent approach. The minimum number of drivers resulting from model ICSP is a lower bound on the minimum number of drivers overall. Thus, if we compare the number of drivers resulting from the independent approach with the number of drivers in the CSP solution, taking the gap with the lower bound into account, we also have an indication of the potential benefit. That is, we have an indication of how many drivers could be saved from the sequential approach by changing the bus schedule.

7.2.3 Integrated approach

In figure 12 we show the computational results for the integrated approach with and without changeovers. We report the same information as in figure 11, except that we also report the number of combined duties in the case of "no changeovers" and the number of buses in the solution.

The gap between the lower and upper bounds is at most 3.57% for the approach with changeovers and 7.32% if changeovers are not allowed. However, we should note that we do not have a guaranteed lower bound in all cases, because of the tailing-off criterion. For prob72 we have computed an additional feasible solution, as described in the preceding subsection. If would not have done this, the best solution was 15 and the gap 6.67%. In the case where changeovers are allowed we have computed the feasible solution for prob148 and prob238 with integer programming solver of CPLEX instead of the heuristic of Caprara et al. [23], because for these problems the heuristic

	changeovers						no changeovers					
	prob24	prob42	prob72	prob113	prob148	prob238	prob24	prob42	prob72	prob113	prob148	prob238
nodes	48	84	144	226	296	476	48	84	144	226	296	476
arcs	132	312	460	846	1,376	2,260	132	312	460	846	1,376	2,260
iterations	5	12	19	44	106	254	3	4	12	9	15	14
average pieces	463	1,568	4,606	9,891	16,237	52,538	1,367	4,706	8,716	19,511	33,435	101,187
duties	512	1,711	7,063	12,644	29,667	80,945	2,201	3,780	5,092	3,150	8,984	16,536
combined duties							213	438	628	323	1,816	4,972
cpu m.	0	2	18	150	911	27,644	0	0	7	0	3	12
cpu p.	1	14	34	837	4,188	214,436	1	19	289	41	255	3,927
cpu t.	1	18	56	1,024	5,185	242,080	1	19	296	41	258	3,942
real_lower	12	18	14	23	33	27	12	18	14	26	38	30
upper	12	18	14	23	34	28	12	18	14	26	41	32
buses	6	9	5	8	11	9	6	9	5	8	12	9
drivers	6	9	9	15	23	19	6	9	9	18	29	23
gap (%)	0.00	0.00	0.00	0.00	2.94	3.57	0.00	0.00	0.00	0.00	7.32	6.25
cpu	4	3	37	212	351	1,907	0	0	42	0	29	35

Figure 12: Results integrated approach on RET data

performed poorly. By doing this, we saved one vehicle and three drivers for prob148 and one vehicle and one driver for prob238.

Again, the major computational effort is in the pricing problem. For prob148 (with changeovers), the computation time of the lower bound phase is about 86 minutes. About 80% is spent on the pricing problem, where an average number of 16,237 pieces of work and a total number of 29,667 duties are generated. In the next subsection we give a comparison between the different approaches.

7.3 A Comparison of Different Approaches

In this subsection, we are interested in a comparison of the approaches discussed in this paper. In figure 13 we present the results obtained by the sequential, the independent and the integrated approach for the RET data. For the sequential approach, the upper bounds (lower bounds) are obtained by the summation of the minimum number of vehicles obtained by solving the SDVSP and the (lower bound on the) number of drivers resulting from the Lagrangian heuristic for the CSP.

				changeovers								no changeovers							
	independent			sequential				integration				sequential				integration			
problem	low	upp	driv	low	upp	bus	driv	low	upp	bus	driv	low	upp	bus	driv	low	upp	bus	driv
prob24	12	12	6	12	12	6	6	12	12	6	6	12	12	6	6	12	12	6	6
prob42	18	18	9	19	19	9	10	18	18	9	9	19	20	9	11	18	18	9	9
prob72	14	14	9	15	15	5	10	14	14	5	9	15	15	5	10	14	14	5	9
prob113	21	22	14	23	24	8	16	23	23	8	15	26	26	8	18	26	26	8	18
prob148	30	33	22	34	34	11	23	33	34	11	23	45	45	11	34	38	41	12	29
prob238	25	27	18	27	29	9	20	27	28	9	19	30	33	9	24	30	32	9	23

Figure 13: Comparison between different approaches for RET data

The number of buses is not mentioned for the independent approach

because it is the same as for the sequential approach. The results for the integrated approach correspond to the best lower and upper bounds obtained using different column generation strategies (as mentioned in the previous subsection). For the case with changeovers, comparing the sequential with the integrated approach shows that the savings when integrating bus and driver scheduling are at most one driver. This confirms the expectation based on the potential savings resulting from comparing the sequential with the independent approach. In 4 out of 6 test cases, the integrated solution was better than the sequential solution due to the saving of one driver. However, for the case without changeovers a significant saving is obtained with the integrated approach. This also confirms the expectation based on the potential savings resulting from comparing the sequential with the independent approach. In 4 out of 6 test cases, the integrated solution was better than the sequential solution due to the saving of up to 5 drivers at the cost of only one additional bus. Interestingly, for the small problems the integrated approach gives the same results for the cases with and without changeovers, but for the larger problems allowing changeovers can save many drivers and buses.

8 CONCLUSIONS

The results reported in the previous section show that we can get good solutions within reasonable computation times on a personal computer. We did not expect that it would be useful to integrate under every circumstance. Based on the results we can conclude that the benefit of integration may be significant when changeovers are not allowed. In practice, it often occurs that either changeovers are not possible due to long distances or changeovers are not allowed for legal or technical reasons. When changeovers are allowed in our test problems, one driver may be saved by considering an integrated approach. Of course, the saving of one driver is considerable when taking into account that we investigated instances for one bus line each. The interpretation of the computational results depends on the ratio between the fixed vehicle and crew costs. If fixed vehicle costs are much higher as compared to crew costs it becomes less attractive to apply the integrated approach. On the other hand, if crew costs are higher the integrated approach becomes more attractive.

Furthermore, we have seen that the results of the independent approach give a good indication of the potential benefit of integration.

Finally, we would like to remark the following. Although, from an employment point of view, it may be interpreted negatively when we say that drivers can be saved by using integration, a different point of view is that the cost savings may allow for an increase in service while no driver needs to

loose his or her job.

Acknowledgments The authors are grateful to the RET, the Rotterdam public transport company, for providing the data.

References

[1] K. Darby-Dowman, J.K. Jachnik, R.L. Lewis, and G. Mitra. Integrated decision support systems for urban transport scheduling: Discussion of implementation and experience. In J.R. Daduna and A. Wren, editors, *Computer-Aided Transit Scheduling: Proceedings of the Fourth International Workshop*, pages 226–239. Springer Verlag, Berlin, 1988.

[2] R. Freling. *Models and Techniques for Integrating Vehicle and Crew Scheduling.* PhD thesis, Tinbergen Institute, Erasmus University Rotterdam, 1997.

[3] M. Desrochers and F. Soumis. A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, 23:1–13, 1989.

[4] R. Freling, A.P.M. Wagelmans, and J.M. Pinto Paixão. An overview of models and techniques for integrating vehicle and crew scheduling. In N.H.M. Wilson, editor, *Computer-Aided Transit Scheduling*, pages 441–460. Springer Verlag, Berlin, 1999.

[5] L. Bodin, B. Golden, A. Assad, and M. Ball. Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research*, 10(2):63–211, 1983.

[6] M. Ball, L. Bodin, and R. Dial. A matching based heuristic for scheduling mass transit crews and vehicles. *Transportation Science*, 17:4–31, 1983.

[7] E. Tosini and C. Vercellis. An interactive system for extra-urban vehicle and crew scheduling problems. In J.R. Daduna and A. Wren, editors, *Computer-Aided Transit Scheduling: Proceedings of the Fourth International Workshop*, pages 41–53. Springer Verlag, Berlin, 1988.

[8] J.C. Falkner and D.M. Ryan. Express: Set partitioning for bus crew scheduling in Christchurch. In M. Desrochers and J.M. Rousseau, editors, *Computer-Aided Scheduling: Proceedings of the Fifth International Workshop*, pages 359–378. Springer Verlag, Berlin, 1992.

[9] I. Patrikalakis and D. Xerocostas. A new decomposition scheme of the urban public transport scheduling problem. In M. Desrochers and J.M. Rousseau, editors, *Computer-Aided Transit Scheduling: Proceedings of the Fifth International Workshop*, pages 407–425. Springer Verlag, Berlin, 1992.

[10] D. Scott. A large linear programming approach to the public transport scheduling and cost model. In J.M. Rousseau, editor, *Computer Scheduling of Public Transport 2*, pages 473–491. North Holland, Amsterdam, 1985.

[11] J.M. Rousseau and J.Y. Blais. Hastus: An interactive system for buses and crew scheduling. In J.M. Rousseau, editor, *Computer Scheduling of Public Transport 2*, pages 473–491. North Holland, Amsterdam, 1985.

[12] R. Freling, C.G.E Boender, and J.M. Pinto Paixão. An integrated approach to vehicle and crew scheduling. Technical Report 9503/A, Econometric Institute, Erasmus University Rotterdam, Rotterdam, 1995.

[13] K. Haase and C. Friberg. An exact branch and cut algorithm for the vehicle and crew scheduling problem. In N.H.M. Wilson, editor, *Computer-Aided Transit Scheduling*, pages 63–80. Springer Verlag, Berlin, 1999.

[14] M. Desrochers. Shortest path problems with resource constraints. Technical Report GERAD G-88-27, École des Hautes Études Commerciales, Montréal, 1988.

[15] K. Haase, G. Desaulniers, and J. Desrosiers. Simultaneous vehicle and crew scheduling in urban mass transit systems. Technical Report GERAD G-98-58, École des Hautes Études Commerciales, Montréal, 1999.

[16] D. Huisman. A note on model VCSP1. World Wide Web, http://www.few.eur.nl/few/people/huisman/publications.htm, 2000.

[17] A.M. Geoffrion. Lagrangean relaxation for integer programming. *Mathematical programming study*, 2:82–114, 1974.

[18] M.L. Fisher. The lagrangian relaxation method for solving integer programming problems. *Management Science*, 27:1–18, 1981.

[19] J.E. Beasley. Lagrangean relaxation. In C.R. Reeves, editor, *Modern Heuristic Techniques for Combinatorial Problems*, pages 243–303. McGraw-Hill, London, 1995.

[20] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance. Branch-and-price: Column generation for solving huge integer programs. Technical report, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia, 1996.

[21] P. Carraresi, L. Girardi, and M. Nonato. Network models, Lagrangean relaxation and subgradients bundle approach in crew scheduling problems. In J.R. Daduna, I. Branco, and J.M. Pinto Paixão, editors, *Computer-Aided Transit Scheduling, Proceedings of the Sixth International Workshop*, pages 188–212. Springer Verlag, Berlin, 1995.

[22] R. Freling, J.M. Pinto Paixão, and A.P.M. Wagelmans. Models and algorithms for vehicle scheduling. Technical Report 9562/A, Econometric Institute, Erasmus University Rotterdam, Rotterdam, 1995. To appear in Transportation Science.

[23] A. Caprara, M. Fischetti, and P. Toth. A heuristic algorithm for the set covering problem. *Operations Research*, 47:730–743, 1999.

[24] R. Freling, D. Huisman, and A.P.M. Wagelmans. Applying an integrated approach to vehicle and crew scheduling in practice. Technical Report EI2000-11/A, Econometric Institute, Erasmus University Rotterdam, Rotterdam, 2000.

ERASMUS **R**ESEARCH **I**NSTITUTE OF **M**ANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

Publications in the Report Series Research* in Management

Impact of the Employee Communication and Perceived External Prestige on Organizational Identification
Ale Smidts, Cees B.M. van Riel & Ad Th.H. Pruyn
ERS-2000-01-MKT

Critical Complexities, from marginal paradigms to learning networks
Slawomir Magala
ERS-2000-02-ORG

Forecasting Market Shares from Models for Sales
Dennis Fok & Philip Hans Franses
ERS-2000-03-MKT

A Greedy Heuristic for a Three-Level Multi-Period Single-Sourcing Problem
H. Edwin Romeijn & Dolores Romero Morales
ERS-2000-04-LIS

Integer Constraints for Train Series Connections
Rob A. Zuidwijk & Leo G. Kroon
ERS-2000-05-LIS

Competitive Exception Learning Using Fuzzy Frequency Distribution
W-M. van den Bergh & J. van den Berg
ERS-2000-06-LIS

Start-Up Capital: Differences Between Male and Female Entrepreneurs, 'Does Gender Matter?'
Ingrid Verheul & Roy Thurik
ERS-2000-07-STR

The Effect of Relational Constructs on Relationship Performance: Does Duration Matter?
Peter C. Verhoef, Philip Hans Franses & Janny C. Hoekstra
ERS-2000-08-MKT

Marketing Cooperatives and Financial Structure: a Transaction Costs Economics Analysis
George W.J. Hendrikse & Cees P. Veerman
ERS-2000-09-ORG

A Marketing Co-operative as a System of Attributes: A case study of VTN/The Greenery International BV,
Jos Bijman, George Hendrikse & Cees Veerman
ERS-2000-10-ORG

---

* ERIM Research Programs:
LIS Business Processes, Logistics and Information Systems
ORG Organizing for Performance
MKT Decision Making in Marketing Management
F&A Financial Decision Making and Accounting
STR Strategic Renewal and the Dynamics of Firms, Networks and Industries

ERASMUS **R**ESEARCH **I**NSTITUTE OF **M**ANAGEMENT

REPORT SERIES
RESEARCH IN MANAGEMENT

Evaluating Style Analysis
Frans A. De Roon, Theo E. Nijman & Jenke R. Ter Horst
ERS-2000-11-F&A

From Skews to a Skewed-t: Modelling option-implied returns by a skewed Student-t
Cyriel de Jong & Ronald Huisman
ERS-2000-12-F&A

Marketing Co-operatives: An Incomplete Contracting Perspective
George W.J. Hendrikse & Cees P. Veerman
ERS-2000-13-ORG