COLUMN GENERATION FOR SCHEDULING SHIPMENTS WITHIN A SUPPLY CHAIN NETWORK WITH THE MINIMUM NUMBER OF VEHICLES

George Kozanidis
Systems Optimization Laboratory
Department of Mechanical Engineering
University of Thessaly
Volos 38334, Greece
gkoz@mie.uth.gr

Keywords: Supply Chain Network, Vehicle Routing, Integer Programming, Column Generation, Branch & Price.

Abstract. We consider the problem of scheduling a set of shipments between different nodes of a supply chain network. Each shipment has a fixed departure time, as well as an origin and a destination node, which, combined, determine the duration of the associated trip. The aim is to schedule as many shipments as possible, while also minimizing the number of vehicles utilized for this purpose.

We develop an integer programming model and an associated branch and price solution algorithm for this problem. The optimal solution to the LP relaxation of the problem is obtained through column generation, a methodology for solving linear programs with a huge number of variables, without explicitly considering all of them. In the context of our application, the proposed methodology utilizes a master problem that schedules the maximum possible number of shipments using only a small set of vehicle-routes, and a column generation (colgen) sub-problem that generates cost-effective vehicle-routes which are fed into the master problem. The optimal solution to the colgen sub-problem is obtained with an efficient network optimization solution algorithm, which outperforms existing commercial optimization software packages that can be used alternatively for that purpose. After finding the optimal solution to the LP relaxation of the problem, the algorithm branches on the fractional decision variables (vehicle-routes), in order to reach the optimal integer solution. Special branching rules that expedite the algorithm’s performance are utilized for this purpose.

We describe in detail the steps of the proposed solution algorithm, focusing on several problem aspects that have a strong influence on its performance. We conclude with limited computational results that demonstrate the algorithm’s performance on a particular case study, and a discussion of how various computational difficulties that can possibly arise can be handled.
1 INTRODUCTION

We consider the problem of scheduling a set of shipments between different nodes of a supply chain network. Each shipment has a fixed departure time, as well as an origin and a destination node, which, combined, determine the duration of the associated trip. The aim is to schedule as many shipments as possible, while also minimizing the number of vehicles utilized for this purpose. The optimization is lexicographic, in the sense that the model minimizes the number of unscheduled shipments first, and then tries to determine how this can be attained with the minimum number of vehicles. This implies that the cost of leaving an extra shipment unscheduled is considered much larger than the cost of utilizing an extra vehicle. Motivated from theory that has been developed for addressing a similar problem under a slightly different context [1], we develop an integer programming formulation and an associated branch and price solution algorithm for this problem.

The proposed solution methodology utilizes a master problem that tries to schedule the maximum possible number of shipments using a set of vehicle-routes, and a column generation (colgen) sub-problem that generates cost-effective vehicle-routes which are fed into the master problem. Due to the huge number of alternative vehicle-routes, only a small subset of them is considered by the master problem (which is therefore often called restricted). The column generation sub-problem uses dual information obtained from the optimal solution to the master LP relaxation, in order to generate the most cost-effective (the one with the minimum reduced-cost) vehicle-route, out of those that have not been generated yet. This route is then added to the master problem, resulting in an update of the dual information.

The procedure continues similarly, reaching eventually the optimal solution to the master LP relaxation. This happens when no other vehicle-route with negative reduced-cost can be identified. Next, the algorithm branches on the fractional decision variables (vehicle-routes), in order to reach the optimal integer solution. Special branching rules that enable the simultaneous elimination of multiple decision variables, expediting this way the performance of the algorithm, are utilized for this purpose. Additional decision variables representing vehicle-routes are generated during this phase, which are rendered cost-effective due to the gradual incorporation of the integrality restrictions.

The remainder of this work is structured as follows. In Section 2, we summarize the related literature, while in Section 3 we develop the two optimization models that the algorithm utilizes. In Section 4, we develop the proposed solution algorithm, focusing on several problem aspects that have a strong influence on its performance. We also discuss how various computational difficulties that can possibly arise can be handled. In Section 5, we present limited computational results regarding the application of the algorithm on a case study. Finally, in Section 6, we point to promising directions for future research.

2 LITERATURE REVIEW

The dependency of the effectiveness of modern supply chain distribution networks on fleets of transportation vehicles is steadily growing in today’s globalized economy markets. In the related literature, there is a vast collection of works that consider vehicle routing problems within a supply chain context. Typical decisions that these works address include the load, route and schedule of each vehicle. A very recent work that considers various routing problems in the context of supply chain management is the one by Schmid et al. [2], in which the authors introduce several model extensions to the classical vehicle routing problem, accommodating crucial decisions that are typically addressed in supply chain coordination, such as lotsizing, scheduling, packing, batching, inventory and intermodality.
Column generation is a very efficient methodology for solving linear programs with a huge number of decision variables. It works by explicitly considering only a small subset of them. Successful implementations of column generation methodologies have been used in various real world applications, such as crew scheduling, vehicle routing, location planning, etc. Chang et al. [3] report a recent application of column generation for scheduling in a supply chain that consists of a production and a distribution stage with associated costs.

Bard and Nananukul [4] address a problem that includes a production facility, a set of customers with time varying demand, and a fleet of homogeneous vehicles using a tabu search solution methodology. The aim is to choose production, inventory levels, and vehicle routes, so as to minimize the total production, holding and distribution cost. This work was later extended by the same authors [5], through the development of a solution algorithm that combines exact and heuristic techniques within a column generation framework.

Finally, Archetti et al. [6] study an inventory-routing problem in which shipments are carried out from a central supplier to a group of retailers with a vehicle of given capacity, and the aim is to determine the shipment quantities and the vehicle routes that minimize the total cost. The authors develop a mixed integer linear programming model for formulating this problem, and a branch-and-cut algorithm that utilizes valid inequalities for solving it.

The problem that we address in the current work is similar to the tail assignment problem, i.e., the problem of finding the optimal assignment of flight legs to commercial aircraft. Gabteni and Grönkvist [1] have also studied this problem within a column generation framework. Thus, the present work and the work of Gabteni and Grönkvist exhibit a few similarities. On the other hand, the current work differs from the work of Gabteni and Grönkvist not only in the application context, but also in several design issues which are mainly aimed to improve the computational performance of the proposed solution algorithm.

3 PROBLEM FORMULATION

In this section, we present the model formulation of the master problem and that of its column counterpart. Both these models utilize the following two common sets:

$I$: set of vehicles,
$S$: set of shipments.

Additional notation specific to each of the two formulations is defined in each corresponding sub-section.

3.1 Master Problem

For the formulation of the master problem, we introduce the following mathematical notation:

Sets:
$R_i$: set of routes of vehicle $i$,

Parameters:
$f$: cost for each vehicle utilized,
$h$: cost for each shipment that remains unscheduled,
$a_{ij}s$: binary parameter that takes the value 1 if route $j$ of vehicle $i$ covers shipment $s$, and 0 otherwise, $i \in I, j \in R_i, s \in S$,

Decision Variables:
$z_i$: binary decision variable that takes the value 1 if vehicle $i$ is utilized, and 0 otherwise, $i \in I$,
$x_{ij}$: binary decision variable that takes the value 1 if route $j$ of vehicle $i$ is scheduled, and 0 otherwise, $i \in I, j \in R_i$,
\( y_s \) : binary decision variable that takes the value 1 if shipment \( s \) remains unscheduled, and 0 otherwise, \( s \in S \).

Utilizing this notation, the master problem is formulated as follows:

\[
\begin{align*}
\text{Min} & \quad \sum_{i \in I} f_i x_i + \sum_{s \in S} h y_s \\
\text{s.t.} & \quad \sum_{j \in R_i} x_{ij} \leq z_i, \quad \forall i \in I \\
& \quad y_s + \sum_{i \in I} \sum_{j \in R_s} a_{ij} x_{ij} = 1, \quad \forall s \in S \\
& \quad z_i, x_{ij}, y_s \text{ binary, } \forall i, j, s
\end{align*}
\]  

The objective function (1) minimizes the total cost, which comprises of the vehicle utilization cost and the cost of the unscheduled shipments. Cost coefficient \( h \) is always much larger than cost coefficient \( f \), imposing the relative priority between the two objectives. Constraint set (2), in conjunction with the penalty \( f \) imposed on variables \( z_i \) in the objective, states that a vehicle is utilized if and only if one of its routes is scheduled. It also ensures that at most one route is scheduled for each vehicle. We call these constraints the vehicle-rows. Constraint set (3) states that each shipment, \( s \), will either be covered by exactly one vehicle-route, or by variable \( y_s \), in which case the corresponding penalty \( h \) will be imposed in the objective. We call these constraints the shipment-rows. Finally, constraint set (4) restricts the decision variables of the problem to binary values.

Due to technical restrictions, each shipment must be carried out independently, i.e., a vehicle cannot accommodate more than one shipments simultaneously. As a consequence, the delivery of each shipment must be carried out immediately after its pickup. This implies that any two distinct shipments covered by the same vehicle-route must be temporally non-overlapping. At the same time, we also make the key assumption that a vehicle is not allowed to travel empty. This implies that for any pair of consecutive shipments covered by the same vehicle, the arrival node of the preceding one must coincide with the departure node of the succeeding one. These, as well as several other rules that the generated vehicle-routes must abide by, are incorporated into the colgen sub-problem formulation which is presented next.

### 3.2 Column generation sub-problem

The routes that are candidate to enter the master problem are ranked in terms of their reduced-cost with respect to the optimal solution of the current master LP relaxation. The aim of the colgen sub-problem is to identify the vehicle-route with the minimum reduced cost. If this reduced-cost is negative, this is an indication that the associated vehicle-route has the potential to improve this solution; therefore, it is added to the master problem, and the new optimal master LP dual solution is updated. If not, this implies that no other vehicle-route can improve the optimal solution of the current master LP relaxation; therefore, the column generation procedure terminates.

Let \( dl_s, al_s, dt_s, \) and \( at_s \) be the departure location, the arrival location, the departure time, and the arrival time of shipment \( s \), respectively. In practice, the arrival/departure location of any shipment coincides with one of the nodes of the associated supply chain network. For each shipment \( s \in S \), we define two sets, as explained next. \( N_s \) is the set of shipments which are next-compatible with shipment \( s \), while \( P_s \) is the set of shipments which are previous-compatible with shipment \( s \). A shipment \( s' \) is next-compatible with shipment \( s \) if \( al_s = dl_s' \) and \( at_s \leq dt_s' \). A shipment \( s' \) is previous-compatible with shipment \( s \), if shipment \( s \) is next-
compatible with shipment $s'$. Let also $l_i$ be the current location of vehicle $i$, which also coincides with one of the nodes of the associated supply chain network. We also define two additional sets. For each vehicle $i \in I$, $F_i$ is the set of shipments which are \textit{next-compatible} with vehicle $i$, while for each shipment $s \in S$, $V_s$ is the set of vehicles which are \textit{previous-compatible} with shipment $s$. A shipment $s$ is next-compatible with vehicle $i$ if $l_i = d_{ls}$. A vehicle $i$ is previous-compatible with shipment $s$ if $s$ is next-compatible with $i$.

For the formulation of the colgen sub-problem, we consider a network $N = \{A, V\}$. The set of vertices, $V$, includes one node for each vehicle, one node for each shipment, as well as one fictitious node $E$, which acts as the terminal node. This latter node is fictitious, because it is assumed that the trip of each vehicle stops upon delivery of the last shipment. The set of arcs, $A$, comprises of all the edges which connect each vehicle node with every node that corresponds to a next-compatible shipment of the associated vehicle, all the edges which connect pairs of nodes that correspond to compatible shipments, as well as one edge for each shipment that connects the node that corresponds to this shipment with the terminal node. The aim of the colgen sub-problem is to identify the longest path in this network that begins in one of the vehicle nodes, visits at least one shipment node, and ends in the terminal node. The length of any path is equal to $c_i + \sum_{s \in C} d_s$, where $i$ is the index of the vehicle associated with the node this path begins from, $C \subseteq S$ is the set of shipment nodes that this path visits, and $c_i/d_i$ is the dual variable of the corresponding vehicle/shipment row in the current master LP optimal solution. Since the cost of any vehicle route in the master problem is equal to 0, this is equivalent to finding the vehicle-route with the minimum reduced-cost. With these in mind, we introduce the following mathematical notation for the colgen sub-problem:

\textbf{Sets:}

- $F_i$: set of shipments which are next-compatible with vehicle $i$, $i \in I$,
- $V_s$: set of vehicles which are previous-compatible with shipment $s$, $s \in S$,
- $N_s$: set of shipments which are next-compatible with shipment $s$, $s \in S$,
- $P_s$: set of shipments which are previous-compatible with shipment $s$, $s \in S$,

\textbf{Parameters:}

- $c_i$: dual value of vehicle row $i$ in current master LP optimal solution, $i \in I$,
- $d_s$: dual value of shipment row $s$ in current master LP optimal solution, $s \in S$,

\textbf{Decision Variables:}

- $z_i$: binary decision variable that takes the value 1 if the generated route utilizes vehicle $i$, and 0 otherwise, $i \in I$,
- $x_s$: binary decision variable that takes the value 1 if the generated route includes a direct travel from vehicle-node $i$ to shipment-node $s$, and 0 otherwise, $i \in I$, $s \in F_i$,
- $x_{st}$: binary decision variable that takes the value 1 if the generated route includes a direct travel from shipment-node $s$ to node $t$, and 0 otherwise, where $t$ is either a shipment node, or the terminal node, $s \in S$, $t \in N_s \cup \{E\}$,
- $y_s$: binary decision variable that takes the value 1 if the generated route covers shipment $s$, and 0 otherwise, $s \in S$.

Utilizing this notation, the colgen sub-problem is formulated as follows:

\begin{align*}
\text{Min} & \quad \sum_{i \in I} -c_i z_i + \sum_{s \in S} -d_s y_s \\
\text{s.t.} & \quad \sum_{i \in I} z_i = 1 \tag{5}
\end{align*}
The objective function (5) minimizes the reduced-cost of the vehicle-route that will be identified. Constraint (6) ensures that this route will utilize exactly one vehicle. Constraint set (7) states that the selected vehicle should visit a node that corresponds to one of its next-compatible shipments first. Constraint set (8) ensures the flow balance at each shipment-node. Incoming flow can originate either at a vehicle-node or at a shipment-node, while outgoing flow can only be directed to a shipment-node or to the terminal node. Constraint (9) states that a shipment is covered if and only if there is incoming flow in the corresponding shipment-node. Finally, constraint set (10) imposes integrality on the decision variables.

4 SOLUTION METHODOLOGY

The proposed solution procedure is a branch and price algorithm that utilizes an associate search tree. At each node of this tree, the algorithm solves the linear relaxation of the associated master problem with column generation. The following three subsections portray in reasonable detail the proposed methodology.

4.1 Solving the master LP relaxation using column generation

Each node of the branch and price tree is associated with a distinct master problem and its companion colgen sub-problem, which are based upon the two fundamental formulations (1)-(4) and (5)-(10), respectively. Two distinct master or colgen problems differ from each other only with respect to the additional constraints that have been added as a result of branching. The exact nature of these extra constraints is explained in the next sub-section. The optimal solution of each master LP relaxation is obtained with column generation, according to the logic-flow shown in Figure 1.

\[
z_i = \sum_{s \in P_i} x_{is}, \ \forall i \in I \tag{7}
\]

\[
\sum_{s \in \mathcal{I}_r} x_{is} + \sum_{r \in \mathcal{P}_r} x_{rs} = \sum_{r \in \mathcal{N}_{r \cup (E)}} x_{is}, \ \forall s \in S \tag{8}
\]

\[
y_s = \sum_{s \in \mathcal{I}_r} x_{is} + \sum_{r \in \mathcal{P}_r} x_{rs}, \ \forall s \in S \tag{9}
\]

\[
z_i, x_{is}, x_{rs}, x_{it}, y_s \text{ binary, } \forall i, s, r, t \tag{10}
\]
4.2 Solving the colgen sub-problem

Instead of using commercial optimization software for solving each colgen sub-problem, one can utilize a simple modification (so as to make it suitable for finding the longest path instead) of the shortest path algorithm of Morávek [7] in order to find the vehicle-route with the minimum reduced-cost, taking advantage of the fact that the associated network is acyclic. For each node, this algorithm stores a label denoting the largest path distance of this node from the source, which is updated accordingly each time that an improved path length is discovered. Naturally, since this algorithm runs in linear time, its performance is significantly superior to that of commercial optimization software packages that can be utilized for solving the colgen sub-problem alternatively, especially on large scale problems. Moreover, the additional constraints that are added as a result of branching can be accommodated with only minor modifications into this algorithm.

4.3 Branching

When the optimal solution to the currently explored master LP relaxation is fractional, the algorithm performs branching in order to continue the search for the optimal integer solution. As in the case of a typical branch and bound solution algorithm, new sub-problems are created through the addition of constraints that eliminate fractional solutions. A typical design involves the selection of one such fractional decision variable for branching, and the partition of the solution space by setting this variable equal to 0 and 1. Utilizing the special structure of the master problem in our case, we adopt a more sophisticated branching rule, which improves considerably the efficiency of the algorithm. This rule, whose validity is proved in the next result, is a suitable modification of the rule originally proposed by Dyer and Foster [8] under a slightly different context.

**Proposition 1:** If the optimal solution to a master LP relaxation contains one or more decision variables with fractional value, then there exist one vehicle-row and one shipment-row, such that the sum of the variables that appear in both rows is fractional.

**Proof:** If the optimal solution to a master LP relaxation is fractional, then at least one of the route-variables is clearly fractional, too. Let $x_{ij}$ be one such variable and $s$ be the index of a shipment that the associated route covers (since no vehicle-node is directly connected with the terminal node in the associated network, at least one such shipment will exist). Since the sum of the variables in shipment-row $s$ is equal to 1, at least one of the other variables that appear in this constraint is also positive. If $y_s > 0$ or $x_{ij}$ is the only positive route-variable of vehicle $i$ that covers shipment $s$, then the proposition holds for vehicle-row $i$ and shipment-row $s$. Otherwise, let $x_{ik}$ be another positive route-variable of vehicle $i$ that also covers shipment $s$. Since any two route-variables in the master problem are distinct, there exists at least one shipment with index $t \neq s$, which is covered by either $x_{ij}$ or $x_{ik}$, but not by both. In this case, the proposition holds for vehicle-row $i$ and shipment-row $t$. □

Proposition 1 provides an elegant way of selecting the branching variables. Whenever the optimal solution to a master LP relaxation is fractional, at least one pair of rows satisfying the condition of Proposition 1 can be identified. The set of branching variables comprises of all the variables that appear in both these rows, and the two new tree nodes are created by branching simultaneously on all of them. In the left sub-problem, the sum of these variables is set equal to 0, whereas in the right sub-problem, it is set equal to 1. This partition of the feasible space is merely imposing the restriction that either exactly one of the associated routes will be selected for the corresponding vehicle, or none of them. Rather than adding the associated constraints explicitly to the master problem, we incorporate them directly into the existing formulation instead. The benefit of doing this is that it retains the same number of master
problem constraints, and thus we do not have to deal with extra dual variables. The exact procedure for doing this is described next.

In the left sub-problem, each of the branching variables is deleted from the master problem, and a suitable constraint is added to the associated colgen sub-problem, ensuring that this variable (vehicle-route) will not be generated again. For the sake of illustration of how this can be done, we augment colgen decision variables $z_i$ and $y_s$ with a second index that takes the value 1 or 2. Index 1 refers to the branching variable that we want to exclude, while index 2 refers to any other variable yet to be generated by colgen. This restriction can then be expressed mathematically with the following constraint:

$$\sum_{i \in I} |z_{i,2} - z_{i,1}| + \sum_{s \in S} |y_{s,2} - y_{s,1}| \geq 1,$$

in which $z_{i,2}$ and $y_{s,2}$ are the regular decision variables of colgen, whereas $z_{i,1}$ and $y_{s,1}$ are the specific values defining the route-variable that we want to exclude. This constraint ensures that any route that will be generated next by colgen in the associated sub-problem will differ from the one being excluded either in the vehicle selection, or in the inclusion/exclusion of at least one shipment. Note that since $z_{i,1}$, $z_{i,2}$, $y_{s,1}$ and $y_{s,2}$ take binary values, the nonlinearities can be easily eliminated from the above constraint, since each absolute term can be set equal to the enclosed quantity when the corresponding parameter $z_{i,1}/y_{s,1}$ is equal to 0, or to the negative of the enclosed quantity when the corresponding parameter $z_{i,1}/y_{s,1}$ is equal to 1.

In the right sub-problem, all the remaining route-variables associated with the same vehicle as that of the branching variables are deleted from the master problem, and a suitable constraint is added to the associated colgen sub-problem, setting the corresponding variable $z_i$ equal to 0. This prohibits the generation of any additional route for this vehicle in the associated sub-problem, since one of the vehicle-routes represented by the branching variables must be selected eventually.

For even better performance, the above branching rule needs to be refined further by fine-tuning additional factors such as the size of the branching variable set, the exact distribution of the branching variable values, etc. Out of many rules that we have examined, the rule of branching on the variable set that includes the variable with the largest fractional value out of all the sets that satisfy the condition of Proposition 1 seems to be the one with the best performance, at least with our current experience. Of course, in the extreme case, this set may be a singleton. The reason that this rule works well is because it selects quickly specific vehicle-routes for inclusion into the solution that the algorithm builds. This strategy behaves well, in general, because the vast number of alternative vehicle-routes leaves plenty of room to the column generation procedure for correcting the involved decisions that may turn out to be poor.

### 4.4 Further algorithmic enhancements

In this sub-section, we present additional algorithmic enhancements that improve the efficiency of the proposed solution algorithm. The first such enhancement involves the addition of multiple vehicle-routes with negative reduced-cost to the master problem each time the colgen sub-problem is solved. Naturally, this enables the solution of the master LP relaxation considerably faster than when only the minimum reduced-cost vehicle-route is added instead.

CPLEX 12.5.1 [9], the commercial optimization software that our code utilizes, provides an excellent feature for doing this, which is called the solution pool. The solution pool makes it possible to record multiple solutions to a mixed integer program, which are chosen from among the ones which are encountered during the search for the optimum. The user can influence the way that the solution pool is populated in several ways through suitable parameter settings, but in the simplest case he/she doesn’t have to do so, because the solution pool is populated by default with predefined parameter settings. In the case of our problem, it is cru-
cial for the user to ensure that only solutions with negative objective are allowed to enter the solution pool, in order to ensure that the reduced-cost of the associated vehicle-routes will be negative. Special attention should be paid to the fact that the solution pool may sometimes store duplicate solutions upon termination. Although rare, this situation may actually come up; therefore, it cannot be excluded altogether. Thus, a pairwise comparison of the identified solutions is deemed necessary in order to ensure that the vehicle-routes added to the master problem are always distinct.

The addition of multiple vehicle-routes to the master problem at each iteration is also possible when the network optimization algorithm of Section 4.2 is used for the solution of the colgen sub-problem. This algorithm can be easily modified to be able to identify the minimum reduced-cost route of each vehicle separately; then, every such route with negative reduced-cost can be added to the master problem, instead of only the optimal one.

Another crucial issue that affects the computational performance of the proposed solution algorithm is related to the exact definition of the master LP relaxation. In order to avoid having to deal with associated dual variables, the master LP relaxation should not include upper bound constraints ($\leq 1$) on the relaxed binary variables $x_{ij}$ and $y_s$, since the feasible set does not change when these bounds are set equal to infinity instead.

Further computational performance improvements are achieved when the solution algorithm is refrained from performing column generation on a node associated with a 0-branch decision, immediately after this node is created. As it turns out, a small percentage of these nodes actually need to be explored for finding the optimal solution. Therefore, significant time savings are attained when each such node is added to the branch and price tree with LP bound equal to that of its parent node, and column generation for finding its optimal LP solution is postponed for when (and if) this node will be selected for exploration.

For similar reasons, the algorithm adopts a “branch on 1 first” strategy, according to which the branch to 1 is always explored before its branch to 0 counterpart. As far as the node selection strategy is concerned, the next tree node to be explored is always the most promising, i.e., the one with the minimum LP bound.

5 COMPUTATIONAL IMPLEMENTATION

In this section, we illustrate the performance of the proposed solution algorithm on a case study drawn from the operation of a realistic supply chain network. The problem includes a set of 732 shipments to be delivered by a fleet of 34 vehicles which are dispersed among various nodes of an underlying 53-node network. The duration of each trip varies from approximately 30 minutes, up to a few hours, while all the shipments must be fulfilled within a one week planning horizon. The solution of the master LP relaxation with column generation takes about 13 minutes on an i5-330 @ 3.0 GHz Intel processor with 4 GB system memory when CPLEX is used for the solution of the colgen sub-problem. This time is reduced to less than 7 minutes when the network optimization solution algorithm of Section 4.2 is utilized for that purpose instead.

CPLEX was configured to disallow entry into the solution pool to those solutions whose objective was at least 50% worse than that of the exact optimum. The capacity of the solution pool was set equal to 10, a limit that was never reached, since the number of solutions returned by the software at each iteration was almost always between 3 and 5. On the other hand, every identified vehicle-route with negative reduced cost was added to the master problem when the network optimization solution algorithm of Section 4.2 was utilized for the solution of the colgen sub-problem instead of CPLEX. The algorithm created a total of 196 tree nodes and required a total of approximately 54 minutes in order to find the optimal integer
solution. The actual number of those explored, however, was smaller, due to the fact that branch-to-0 nodes were only solved upon selection for exploration.

The above computational results cannot be considered representative, since they only refer to a single problem instance. They are merely aimed to provide the reader with a slight sense of the algorithm’s computational capabilities and requirements. Besides space limitations, we also defer the presentation of more extensive computational results for a later manuscript version, due to the fact that several of the proposed computational enhancements that can critically improve the algorithm’s performance are currently under development.

6 FUTURE WORK

The size of the case study problem of the previous section is considered medium in the context of our application. For problems of such size, the algorithm can successfully provide the exact optimal solution in very reasonable computational times. In practice, however, problems of much larger size are often encountered, for which a solution of acceptable quality cannot be obtained in reasonable computational time, unless drastic enhancements are incorporated into the design of the proposed solution algorithm. Future research should definitely be directed towards the development of such enhancements, in order to enable the handling of large scale problems.

A crucial development in that direction involves the stabilization of the column generation procedure, which enables the faster solution of the LP relaxations which are encountered. This technique aims to cure the slow convergence and degeneracy difficulties that column generation often exhibits, especially on large scale problems. Several stabilization techniques have been proposed in the related literature, the most popular (and at the same time the easiest to implement) being the ones which do not allow the dual values to oscillate uncontrollably, by imposing bounds on them, or by imposing a penalty when they deviate from a stability center (e.g., see du Merle [10]).

Finally, when the problem is too large and cannot be handled efficiently even with the proposed enhancements, the user must inevitably compromise for a near-optimal solution instead of the exactly optimal one. The most common technique for achieving this is the incorporation of tolerances on the optimal objective. When such tolerances are present, the algorithm does not backtrack to nodes created earlier in the associated search tree, unless these tolerances are violated. In any other case, the algorithm continues its dive in the tree, which makes it easier to obtain faster a near optimal integer solution. Through the choice of the tolerance values and the relaxation bounds on the optimal objective, the user can control how close this solution will be to the truly optimal one, and may select to interrupt the execution of the algorithm before its termination if the quality of the best integer solution that has been found so far is acceptable.

ACKNOWLEDGMENTS

This research has been co-financed by the European Union (European Social Fund - ESF) and Greek national funds through the Operational Program “Education and Lifelong Learning” of the National Strategic Reference Framework (NSRF) - Research Funding Program: Thales. Investing in knowledge society through the European Social Fund.

REFERENCES


