A Demand-Shifting Feasibility Algorithm For Benders Decomposition

Peiling Wu
Joseph C. Hartman
George R. Wilson
Lehigh University

Report No. 00T-008

A Demand-Shifting Feasibility Algorithm for Benders Decomposition

Peiling Wu, Joseph C. Hartman*and George R. Wilson

Industrial and Manufacturing Systems Engineering Department, Lehigh University, Mohler Lab, 200 W. Packer Ave., Bethlehem, PA 18015, USA

Abstract

Benders decomposition is a popular method for solving problems by resource-directive decomposition. Often, the resource allocations from the master problem lead to infeasible subproblems, as resources are insufficient to meet demand. This generally requires the use of feasibility cuts to reach a feasible solution, which can be computationally expensive. For problems in which subproblems have limited capacity, we propose an efficient algorithm which shifts excess demand to other sources of capacity. The advantages of the algorithm are that it is quick, requires only one solution of each subproblem in each Benders iteration, and does not add any feasibility cuts into the master problem. A computational study is performed on a fleet sizing problem to illustrate the algorithm's efficiency when compared to the traditional feasibility cut method.

Keywords: Large-scale optimization; Benders decomposition; Subproblem feasibility; Fleet sizing

1 Introduction

Benders decomposition [Benders, 1962], also known as Benders' partitioning or outer linearization, is a popular method for solving problems by resource-directive decomposition. It has been applied, with success, to both linear and mixed integer programming problems in a variety of applications. These include solving two-stage recourse problems [Van Slyke and Wets, 1969], the multicommodity distribution system design problem [Geoffrion and Graves, 1974], the quadratic assignment problem [Bazaraa and Sherali, 1980], hierarchical production planning problems [Aardal and Larsson, 1990] and the parallel replacement problem [Chen, 1998], among others.

In general, Benders decomposition works in an iterative fashion. First, the configurative variables (usually the complicating variables) are determined in a master problem and then the remaining, lower-level decision variables are found in subproblems. At each iteration, optimality cuts are generated from the subproblem dual variables and added into the master problem. It is well known that solving the Benders master problem can be computationally intensive. Magnanti et al. [1978, 1981] proposed finding "pareto-optimal" cuts to

*Corresponding Author: E-mail: jch6@lehigh.edu, Phone: 610-758-4430, Fax: 610-758-4886

accelerate Benders decomposition, with multiple optimal solutions from the subproblems. Minoux [1984] suggested solving the master problem only to obtain a feasible solution and a lower bound, due to the disadvantage of degeneracy and round-off errors in the simplex method. Aardal and Larsson [1990] priced out the benders primal cuts such that the remaining Lagrangean relaxation problem could be solved easily by dynamic programming. However, Holmberg [1994] showed that relaxing the Benders primal cuts does not yield a lower bound that is as good as the bound from Lagrangean relaxation of the original problem.

In this paper, we are concerned with the feasibility of the subproblems once the master problem has been solved and allocations to the subproblems have been made. This is not always a problem. In Geoffrion and Graves [1974], it is assumed that the transportation subproblem always has capacity that is greater than the total demand, so as to preclude the possibility of an infeasible solution. The Benders' subproblems of the hierarchical production planning model are always feasible [Aardal and Larsson, 1990], as unlimited overtime production is allowed. Similarly, as the assumption of constant demand implies that the total number of machines is constant over time in [Chen, 1998], the subproblem for each asset group does not encounter infeasibility. The power plant expansion problem has complete recourse, provided that at least one zero-leadtime technology is available [Birge and Louveaux, 1997]. In this paper, we are concerned with problems in which the subproblems have fixed, limited capacity. Specifically, we illustrate this problem with a rental fleet sizing model which decomposes the fleet by truck type and age. As it is assumed that only new trucks can be purchased, the subproblems with older trucks have an upper bound on capacity. While it is possible to penalize the acquisition of additional capacity with arbitrarily high prices, it has been our experience that the convergence of the Benders procedure may be slow with this assumption, thus further motivating the development of our algorithm.

Classical Benders decomposition deals with the infeasibility of subproblems by means of generating feasibility cuts [Van Slyke and Wets, 1969]. Obviously, solution of the master problem is more computationally expensive with the addition of both optimality and feasibility cuts. We are therefore motivated to propose a feasibility algorithm which shifts excess demand to other sources of capacities (from the infeasible subproblem to a different subproblem) such that a feasible solution can be found quickly, and no additional computational burden is placed on solving the master problem. The algorithm, by construction, also has the benefit of only requiring one solution of each subproblem in each iteration of the Benders procedure. Thus, the master problem need not be reformulated and solved repeatedly until all the subproblems are feasible.

We review the traditional feasibility cut approach of Benders decomposition and present the details of the demand-shifting feasibility algorithm in Section 2. In Section 3, a case study on a fleet sizing problem illustrates how the demand-shifting algorithm works effectively on dynamic problems of multiple dimensions (space and time). The computational effectiveness of the algorithm is analyzed, along with a comparison to a traditional, feasibility-cut generating scheme. Conclusions and directions for future research are given in the final section.

2 Demand-Shifting Feasibility Algorithm

In this paper, we consider a large scale linear program (P) of the following form:

$$\min cX + hY$$

subject to:

$$DX + BY = d \tag{1}$$

$$AX \leq r \tag{2}$$

$$X \geq 0, Y \geq 0$$

where B and D have a block separable structure,

$$B = \begin{bmatrix} B_1 & & & \\ & B_2 & & \\ & & . & \\ & & & B_K \end{bmatrix}, \quad D = \begin{bmatrix} D_1 & & & \\ & D_2 & & \\ & & . & \\ & & & D_K \end{bmatrix},$$

and $h = [h_1, h_2, ...h_K]$ and $d = [d_1, d_2, ...d_K]$.

As solving large problems of type P can be demanding, decomposition procedures are generally employed. We observe that the problem (P) can be decomposed into k independent subproblems (SP_k) by fixing the X variables. Without loss of generality, Constraints (1) and (2) can be regarded as demand and resource allocation constraints. The Benders subproblem (SP_k) is defined as follows:

$$(SP_k) \min \sum_{k=1}^{K} h_k Y_k$$

subject to:

$$B_k Y_k = d_k - D_k X \qquad k = 1..K \tag{3}$$

$$Y_k \geq 0$$

The Benders master problem (MP) is defined as follows:

$$(MP) \min cX + z$$

subject to:

$$AX \leq r$$

$$z \geq \sum_{k=1}^{K} \pi_{k(n)}(d_k - D_k X) \quad n = 1, 2, ...N \tag{4}$$

$$X \geq 0$$

The solution of SP_k in the n^{th} iteration produces $\pi_{k(n)}$, the dual variables of Constraints (3), which are used to construct the optimality cut in Constraints (4). Basically, each subproblem (SP_k) finds a myopic optima of Y_k, given the fixed proportion of allocation on X. Acting as a coordinator, the master problem (MP) then adjusts its allocations through the subproblem price information ($\pi_{k(n)}$), embedded in the Benders cuts. Iterations between the master problem and subproblems proceed until a prespecified stopping criteria is met (or optimality is reached).

For specific problems, valid inequalities can be derived and included in the master problem to avoid infeasible subproblems. These are often referred to as induced constraints [Birge and Louveaux, 1997]. In Geoffrion and Graves [1974], feasibility is guaranteed as an induced constraint forces total demand to not be greater than the available supply in a given subproblem. Unfortunately, induced constraints are problem dependent. Without induced constraints, feasibility cuts may need to be generated in the form of Constraints (5) [Birge and Louveaux, 1997] to guarantee feasibility, as in (MP′) below:

$$(MP') \min cX + z$$

subject to:

$$\begin{aligned} & AX \leq r \\ & z \geq \sum_{k=1}^{K} \pi_{k(n)}(d_k - D_k X) \qquad n = 1, \ldots N \\ & 0 \geq \sum_{k=1}^{K} \sigma_{k(m)}(d_k - D_k X) \qquad m = 1, \ldots M \\ & X \geq 0 \end{aligned} \tag{5}$$

Here, $\sigma_{k(m)}$ are the dual variables of Constraints (3) when (SP_k) has no feasible solution at the mth iteration. The general Benders procedure, with the addition of feasibility cuts, works as follows:

0. Initialization: Set $m = 0, n = 0$; Select initial allocation X.

1. Solve the Benders subproblems SP_k, $k = 1, \ldots K$ sequentially. If SP_k is infeasible for any k, stop and set $m = m + 1$. Construct feasibility cuts as in (5) and add it into the master problem (MP'). Else, go to 3.

2. Solve the reformulated master problem (MP') and pass the new allocation decision X to the subproblems SP_k. Go to 1.

3. Check optimality. If optimality condition holds or a stopping criteria is reached, stop. Else, set $n = n + 1$, generate optimality cuts as in (4) and continue to Step 4.

4. Solve (MP'). Given new X, go to 1.

As seen by the procedure, feasibility cuts are generated for each infeasible subproblem until all the subproblems are feasible, before any optimality cuts can be added. If many feasibility cuts are added, the process can become computationally expensive in addition to the fact that the master problem becomes increasingly more difficult to solve with a greater number of constraints. This can make the traditional feasibility cut procedure very unappealing.

We propose a demand-shifting feasibility algorithm to alleviate this problem in that it does not require the addition of feasibility cuts, reformulation, and solution of the master problem in each iteration. Without loss of generality, suppose SP_{k0} is infeasible at one iteration, i.e., $B_{k0}Y_{k0} \leq d_{k0} - D_{k0}X$. Note that $(d_{k0} - D_{k0}X - B_{k0}Y_{k0})$ can be regarded as excess demand, where $D_{k0}X$ is the first-stage capacity allocation from the master problem and $B_{k0}Y_{k0}$ is the second-stage capacity allocation. Satisfying the constraint $B_{k0}Y_{k0} = d_{k0} - D_{k0}X$ is equivalent to shifting excess demand to other sources of capacities. This idea of demand-shifting can be applied to a variety of infeasible subproblem instances, although problem-dependent structures may need to be exploited individually. In the next section, we illustrate how the demand-shifting algorithm works in the context of a rental fleet sizing problem.

The Benders decomposition with the demand-shifting feasibility algorithm can be described as follows:

0. Initialization: Set $n = 0$; Order the subproblems and select initial allocation X.

1. Solve the Benders subproblems SP_k, $k = 1, ...K$ sequentially with excess (infeasible) capacity arbitrarily high. If any subproblems SP_k is infeasible (excess capacity is acquired), shift excess demand $d_{k0} - D_{k0}X - B_{k0}Y_{k0}$ to the next subproblem, thereby constructing a feasible solution to SP_k.

2. Check optimality. If optimality condition holds or a stopping criteria is reached, stop. Else, set $n = n + 1$, generate optimality cuts as in (4) and continue to Step 4.

3. Solve (MP'). Given new X, go to 1.

The ordering of the subproblems is problem dependent. As demand is shifted from one problem to the next, the shift should occur according to increased cost for capacity. The benefits of this method, as compared to the traditional feasibility cut procedure are numerous: (1) each subproblem is solved only once in each Benders iteration as feasibility is assured with the demand-shifting algorithm; (2) no feasibility cuts are added into the master problem; (3) the master problem need not be reformulated and solved repeatedly to find feasible solutions of the subproblems.

3 Application to a Rental Fleet Sizing (RFS) Problem

Determining the number of vehicles to be maintained in a transportation system at each location over a certain time period is traditionally known as the fleet sizing problem [Turnquist and Jordan, 1986, Klincewicz et al., 1990, Beaujon and Turnquist, 1991]. In the context of the truck rental industry, the fleet sizing problem is concerned with determining how many trucks of various types (generally different sizes) are needed to service customer demand while minimizing total costs, which include capital and operating costs. The size of the fleet can be altered through the purchase or lease of additional vehicles or the sale of excess vehicles. Typically, a customer reserves a truck of specified size at a given location for a given departure time to be taken to another location within a given time frame.

There are generally two types of truck movements: (1) loaded, where the customer moves the truck and (2) empty, where the company moves the truck in order to service demand at an alternate location. The loaded truck travel time is uncertain, as travel routes and speed are customer dependent. Average travel

time between cities can be estimated from historical information. However, empty truck travel time tends to be deterministic as it is mainly dependent on the travel distance (and is performed by professional drivers).

As fleets are diverse, in that trucks vary in capacity and age, the assignment of a truck to a customer carries a cost. Thus, a fleet sizing model must account for operational details, such as how to allocate trucks to meet customer demands and how to move empty trucks, as well as tactical decisions such as the procurement and disposal of trucks over time. The allocation of trucks to meet customer demands involves considerations of a truck's age, as older trucks tend to be more expensive to operate and maintain, as well as the capacity, as larger capacity trucks can be substituted for smaller trucks originally requested by the customer. This is defined as an "upgrade" policy, as customers are upgraded to a larger truck.

Define the following variables and parameters:

L = number of locations, $l = 1, 2, ...L$;

H = number of time periods in the planning horizon with $t = 1, 2, ...H$;

K = number of truck types with respect to truck capacity, $k = 1, 2, ...K$;

N = maximum age for trucks to be kept in the fleet, $a = 0, 1, ...N$;

$\lambda_{\max}$ =maximum time periods to move loaded trucks from one location to another, $\lambda = 1, 2, ...\lambda_{\max}$;

$\beta_{ll'}(\lambda)$ =percentage of loaded trucks taking λ time periods from location l to l';

$\omega_{ll'}$ =time periods to move empty trucks from location l to l';

$I_l^{k,a}(0)$ =initial fleet of type k trucks of age a at location l;

$d_{ll'}^k(t)$ =demand for type k trucks to move from location l to l' leaving at time period t.

Additionally, $m^{k,a}(t)$, $v^{k,a}(t), c^{k,a}(t), p^{k,a}(t)$ and $s^{k,a}(t)$ are inventory costs for idle trucks, maintenance costs for loaded trucks, operating & maintenance costs for moving empty trucks, purchase prices and salvage values, respectively, for each truck of type k and age a in period t. Note that these costs are differentiated as the customer pays the operating costs of moving a loaded truck.

Decision variables $X_{ll'}^{k,a}(t)$ and $\overline{d}_{ll'}^{k,k',a}(t)$ define empty movement and demand allocation, or loaded flow with down-grade k', of type k trucks of age a, from location l to l' in period t. Variables $B_l^{k,a}(t)$, $S_l^{k,a}(t)$ and $I_l^{k,a}(t)$ are the number of type k trucks of age a purchased, sold and held in inventory, respectively, at location l in period t.

We present the following fleet sizing model (RFS). The objective function minimizes empty truck operating and maintenance costs, loaded truck maintenance costs, idle truck maintenance costs, purchase costs less salvage revenues for all the trucks in the fleet within the overall time-space network, as follows:

$$\min \sum_{k=1}^{K}\sum_{t=1}^{H}\sum_{k=1}^{K}\sum_{a=0}^{N}\sum_{(l,l')\in L^2}(c^{k,a}(t)\omega_{ll'}X_{ll'}^{k,a}(t) + v^{k,a}(t)\sum_{\lambda=1}^{\lambda_{\max}}\lambda\beta_{ll'}(\lambda)\sum_{k'=1}^{k}\overline{d}_{ll'}^{k',k,a}(t))$$
$$+\sum_{t=1}^{H}\sum_{k=1}^{K}\sum_{a=0}^{N}\sum_{l=1}^{L}(m^{k,a}(t)I_l^{k,a}(t) + p^{k,a}(t)B_l^{k,a}(t) - s^{k,a}(t)S_l^{k,a}(t))$$

subject to:

Demand allocation with the downgrade policy:

$$\sum_{k'=k}^{K}\sum_{a=0}^{N}\overline{d}_{ll'}^{k,k',a}(t) = d_{ll'}^{k}(t) \quad \forall k = 1,2,...K,\ l,l' = 1,2,...L,\ t = 1,2,...H \tag{6}$$

Flow balance at each location node in time:

$$\begin{aligned}&\sum_{l'=1}^{L}\sum_{\lambda=1}^{\lambda_{\max}}\beta_{l'l}(\lambda)\ \sum_{k'=1}^{k}\overline{d}_{l'l}^{k',k,a}(t-\lambda) + \sum_{l'=1}^{L}X_{l'l}^{k,a}(t-\omega_{l'l}) + I_l^{k,a}(t-1) + B_l^{k,a}(t) - S_l^{k,a}(t)\\ &= \sum_{l'=1}^{L}\sum_{k'=1}^{k}\overline{d}_{ll'}^{k',k,a}(t) + \sum_{l'=1}^{L}X_{ll'}^{k,a}(t) + I_l^{k,a}(t) \qquad \forall k = 1,2,...K,\ a = 0,1,...N,\ l = 1,2,...L,\ t = 1,2,...H\end{aligned} \tag{7}$$

Restricting purchases to only new trucks:

$$B_l^{k,a}(t) = 0 \qquad \forall k = 1,2,...K,\ l = 1,2,...,L, t = 1,2,...H, a = 1,2,..N \tag{8}$$

Restricting the sale to only used trucks:

$$S_l^{k,0}(t) = 0 \qquad \forall k = 1,2,...K,\ \ l = 1,2,...,L,\ t = 1,2,...H \tag{9}$$

And restricting all variables to be non-negative:

$$\begin{gathered}\overline{d}_{ll'}^{k,k',a}(t) \geq 0,\ X_{ll'}^{k,a}(t) \geq 0\ (l \neq l'),\ B_l^{k,a}(t) \geq 0,\ S_l^{k,a}(t) \geq 0,\ I_l^{k,a}(t) \geq 0\\ \forall (l,l') \in L^2,\ t = 1,2,...H,\ k,k' = 1,2,...K,\ a = 0,1,...,N\end{gathered}$$

Note that the flow balance constraints (7) in the RFS model present a network flow structure as shown in Figure 1. The figure illustrates three different types of truck movements: empty (dotted line), local and one-way loaded movements (solid line). Loaded movements are associated with travel time probabilities, which are differentiated by arc boldness. Idle trucks can be treated as inventory (double line) from one time period to the next time period. Additionally, for each time-space node, a purchase node and a salvage node are included to model asset acquisition and disposal decisions. Flow is balanced over time and space, in that all the inflow plus truck inventory (including purchase and salvage) must equal the outflow of trucks and ending inventory in every time period at each location.

Demand allocation, referred to Constraints (6) are the complicating constraints in the RFS model. Given customer demand on a certain type of truck, the loaded truck flows are allocated based on vehicle age and the down-grade policy. We observe that if the complicating demand allocation variables, $\overline{d}_{ll'}^{k',k,a}(t)$, are fixed, the RFS problem can be decomposed into a set of minimum cost network flow subproblems, one for each truck type k and age a. In each of these subproblems, time periods and locations are intertwined within the entire transportation network, hence there is no direct way to decompose by time and location. As a result, the demand allocation constraints are separated and solved in the master problem.

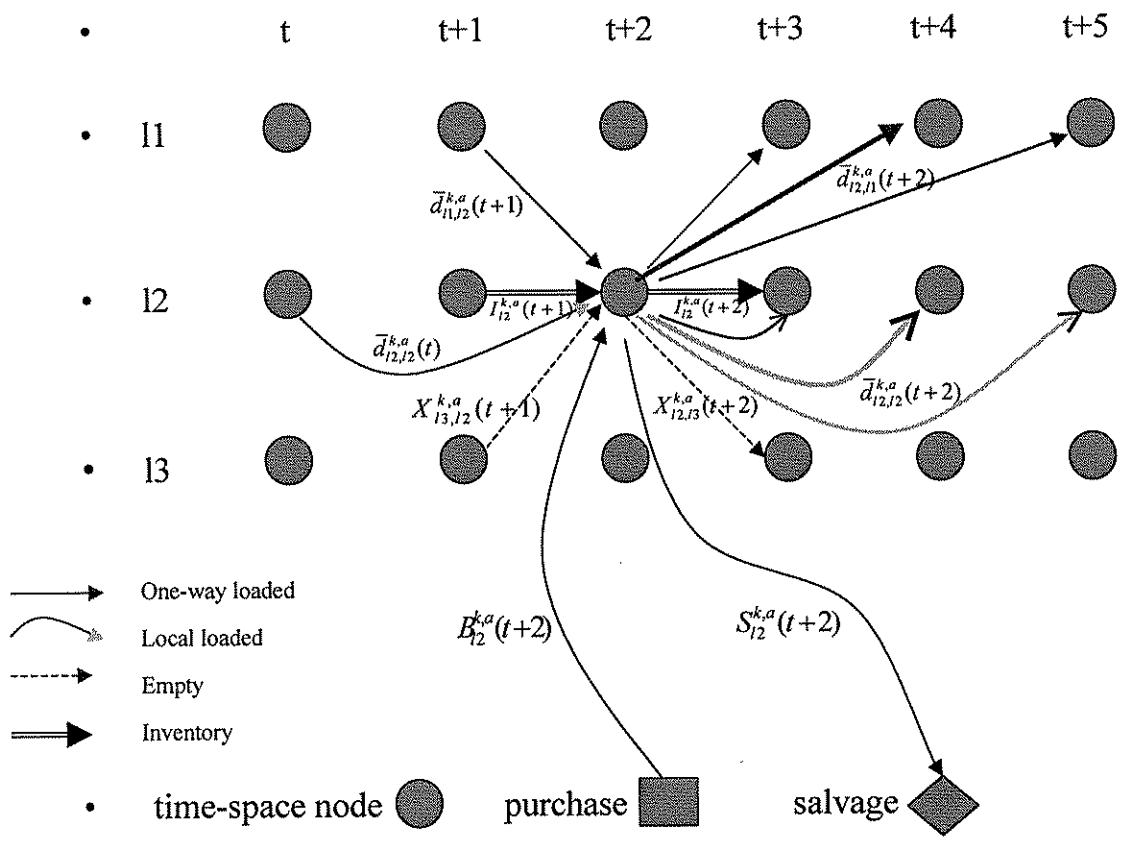

Figure 1: Illustration of flow balance constraint, for truck type k and age a, at time-space node (t+2, $l2$)

The Benders primal subproblems, PS(k,a), are defined as follows:

$$\min \sum_{t=1}^{H} \sum_{(l,l')\in L^2} c^{k,a}(t)\omega_{ll'} X_{ll'}^{k,a}(t) + \sum_{t=1}^{H}\sum_{l=1}^{L}(m^{k,a}(t)I_l^{k,a}(t) + p^{k,a}(t)B_l^{k,a}(t) - s^{k,a}(t)S_l^{k,a}(t))$$

subject to:

$$\sum_{l'=1}^{L} X_{ll'}^{k,a}(1) + I_l^{k,a}(1) - B_l^{k,a}(1) + S_l^{k,a}(1)$$
$$= I_l^{k,a}(0) - \sum_{l'=1}^{L}\sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(1) \qquad \forall l = 1,2,...L \tag{10}$$

$$\sum_{l'=1}^{L} X_{ll'}^{k,a}(t) + I_l^{k,a}(t) - \sum_{l'=1}^{L} X_{l'l}^{k,a}(t-\omega_{l'l}) - I_l^{k,a}(t-1) - B_l^{k,a}(t) + S_l^{k,a}(t) =$$
$$\sum_{l'=1}^{L}\sum_{\lambda=1}^{\lambda_{\max}} \beta_{l'l}(\lambda) \sum_{k'=1}^{k} \overline{d}_{l'l}^{k',k,a}(t-\lambda) - \sum_{l'=1}^{L}\sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(t) \qquad \forall l = 1,2,...L,\ t = 2,3,...H \tag{11}$$

$$X_{ll'}^{k,a}(t) \geq 0,\quad B_l^{k,a}(t) \geq 0,\quad S_l^{k,a}(t) \geq 0,\quad I_l^{k,a}(t) \geq 0 \qquad \forall (l,l') \in L^2,\ t = 1,2,...H$$

Previously defined Constraints (8) and (9) are also included in the subproblems. The limitation on purchasing or leasing used trucks in this problem (Constraint (8)) can lead to infeasible subproblems. In other words, at a certain iteration, the demand allocation determined by the master problem, as in the right-hand side of Constraints (10) and (11), may be too great to be handled by some subproblem defined by truck type and age.

To alleviate this concern, valid inequalities are derived from the flow balance of the time-space network. They are based on the facts that (1) given an initial fleet of trucks, the demand allocated to each location in the first time period cannot exceed its initial inventory; and (2) in any given time period, the feasible demand allocation cannot be more than the sum of incoming empty trucks, idle trucks from the previous period and returning loaded trucks, up to the current period. Now, we prove that the following inequalities are valid.

The first inequality states that the demand allocated in the first period cannot exceed the initial fleet:

Theorem 1 *Given $k \leq K$ and $a \geq 1$, at each location l,*

$$I_l^{k,a}(0) \geq \sum_{l'=1}^{L}\sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(1) \quad \forall l = 1,2,...L \tag{12}$$

are valid inequalities to the RFS model.

Proof. Recall Constraints (10) and note that for all $a \geq 1$, $B_l^{k,a}(t) = 0$, for all l and t, by Constraint (8) and for all l, l' and t, $X_{ll'}^{k,a}(t) \geq 0, S_l^{k,a}(t) \geq 0$, and $I_l^{k,a}(t) \geq 0$. Therefore, at $t = 1$, Constraints (10) imply that

$$I_l^{k,a}(0) - \sum_{l'=1}^{L} \sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(1) \geq 0 \quad \forall l = 1, 2, ...L$$

and (12) are valid inequalities. ■

The second inequality says that the allocated demand in any time period cannot exceed the initial fleet, as excess capacity (used trucks) cannot be acquired. This in turn means that the total loaded flow across any time period cannot exceed the initial fleet.

Theorem 2 *Given $k \leq K$ and $a \geq 1$, for all $t > 1$,*

$$\sum_{l=1}^{L} I_l^{k,a}(0) \geq \sum_{(l,l')} \sum_{\bar{t}=1}^{t} \sum_{\lambda=t+1-\bar{t}}^{\lambda_{\max}} \beta_{l'l}(\lambda) (\sum_{k'=1}^{k} \overline{d}_{l'l}^{k',k,a}(\bar{t})) \quad \forall t = 2, 3, ...H \tag{13}$$

are valid inequalities to the RFS model.

Proof. Consider a cut in the network between any time period t and $t+1$, $t < H$. In traditional network flow theory, the maximum flow in a network is equal to the minimum cut [Ahuja et al., 1993]. In this problem, the maximum *loaded* flow in the network is restricted by the truck capacity, which has an upper bound equivalent to the initial fleet. Thus, the maximum capacity of the system at any time period is the total of initial fleet of trucks at all the locations $\sum_{l=1}^{L} I_l^{k,a}(0)$.

The amount of loaded flow in a cut between any time periods t and $t+1$ is equivalent to the loaded flow that departed from time t, plus the loaded movements that departed time $t-1$ with more than one travel time period, etc. This is reflected in the multiple travel periods, as given in the right hand side of (13). As this is the maximum loaded flow across the cut, it is bound by the initial fleet and (13) are valid inequalities. ■

Now, the restricted Benders master problem (PM) is defined, with the inclusion of the valid cuts (12) and (13), as follows:

$$\min \sum_{k=1}^{K} \sum_{a=0}^{N} \sum_{t=1}^{H} \sum_{(l,l') \in L^2} v^{k,a}(t) \sum_{\lambda=1}^{\lambda_{\max}} \lambda \beta_{ll'}(\lambda) \sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(t) + \sum_{k=1}^{K} \sum_{a=0}^{N} z^{k,a}$$

subject to:

$$z^{k,a} \geq \sum_{l=1}^{L} \theta_l^{k,a,n}(I_l^{k,a}(0) - \sum_{l'=1}^{L}\sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(1))$$

$$+\sum_{l=1}^{L}\sum_{t=2}^{H} \pi_{l,t}^{k,a,n}(\sum_{l'=1}^{L}\sum_{\lambda=1}^{\lambda_{\max}} \beta_{l'l}(\lambda) \sum_{k'=1}^{k} \overline{d}_{l'l}^{k',k,a}(t-\lambda) - \sum_{l'=1}^{L}\sum_{k'=1}^{k} \overline{d}_{ll'}^{k',k,a}(t))$$

$$\forall k = 1,2,..K, \quad a = 0,1,...N, \; n = 1,2,...\# \text{ of generated optimality cuts}$$

$$\sum_{k'=k}^{K}\sum_{a=0}^{N} \overline{d}_{ll'}^{k,k',a}(t) = d_{ll'}^{k}(t) \quad \forall k = 1,2,...K, \; l,l' = 1,2,...L, \; t = 1,2,...H$$

$$\overline{d}_{ll'}^{k,k',a}(t) \geq 0, \; z^{k,a} \text{ unrestricted}, \forall k,k' = 1,2,...K, \; a = 0,1,...N, \; l,l' = 1,2,...L, \; t = 1,2,...H$$

Again, $\theta_l^{k,a}$ and $\pi_{l,t}^{k,a}$ are the dual variables associated with (10) and (11).

Solving the master problem with the valid inequalities (12) and (13) dramatically decreases the requirement of purchasing used trucks in RFS. In particular, at the first time period $t = 1$, no used trucks need be purchased as the location of the truck inventory is known with certainty at the beginning of the problem. Nevertheless, adding these valid cuts cannot guarantee, in general, feasibility of the subproblems beyond $t = 1$. This is because the valid inequalities (13) restrict total flow between any time periods t and $t+1$, but demand allocations, $\overline{d}_{ll'}^{k,k',a}(t)$, are for each node (l,t). Thus an allocation may be feasible according to (13), but because of the distribution of capacity over space, the allocation may be infeasible. In essence, infeasibilities are due to the dynamics and high dimension of the problem as well as multiple travel time periods .

Further note that subtracting the excess purchases from allocated demand is not sufficient for achieving feasibility. The dynamic time-space structure accumulates excess demand at other nodes in later time periods. Thus, subtracting excess purchases still leave downstream infeasibilities. Although, we could also price the excess (infeasible) capacity arbitrarily high (or in some stepwise fashion), that leads to slow convergence in our experience as the subproblems must be continually re-solved.

In the demand-shifting algorithm, we allow excess (infeasible) capacity to be acquired. Then, the excess capacity, which corresponds to excess allocated demand, is determined at each location in each time period. Once determined, the excess is removed and shifted to the following subproblem, which is then solved and the process is repeated.

In the fleet sizing application, we push excess demand for a used truck group towards newer trucks such that available used trucks are sufficient to cover the allocated demand and the remaining flows are all balanced. Assume that the maximum in-service vehicle age $N = 3$ and total truck types $K = 3$. This leads to 12 subproblems, as shown in Figure 2. We begin with the oldest truck group $N = 3$. At a certain iteration, given the allocated demand from the master problem, if there is no need to buy any trucks of age N, it means that all subproblems PS(k,N) are feasible. If feasible, the demand allocation for the age N

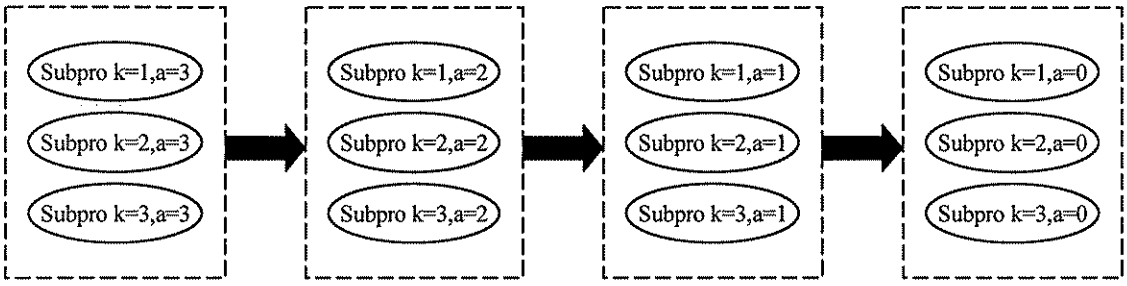

Figure 2: Order of Demand-Shifting in the RFS application with 12 Asset Classes

trucks are fixed and the age $N-1$ group of trucks are then examined. Suppose that there exists an infeasible subproblem PS($k^*, N-1$). In order to avoid purchasing age $N-1$ trucks, the excess demand allocation to the age $N-1$ group of trucks are shifted to age $N-2$ group, while maintaining flow balance constraints (10) and (11). This is done by tracing the path of a used truck purchase and removing the entire path. After examining all the subproblems PS($k, N-1$), solve the subproblems PS($k, N-2$) for $k = 1, 2, ..K$ to obtain the updated demand allocation for age group $N-2$. The above procedure is repeated for the truck group of age $N-2$, until reaching the age 0 (new) group which it is allowed to purchase additional capacity.

As a truck may follow a number of paths from a node, the algorithm uses the following priority structure when removing the path associated with excess capacity purchases. This ordering was determined to achieve the fastest convergence in our problem testing. At any time-space node (l_1, t), there are four types of possible actions for a truck, including salvage at (l_1, t), being held inventory to $(l_1, t+1)$, empty movement to $(l_2, t+\omega_{l_1,l_2})$, and loaded movement to $(l_2, t+\lambda)$ with the probability of $\beta_{l_1 l_2}(\lambda)$, $\lambda = 1, 2, ...\lambda_{\max}$. To avoid purchasing extra used trucks at (l_1, t), the first priority is to avoid selling used ones at (l_1, t), as purchase price is generally greater than salvage value. The next is to decrease the held inventory since idle trucks are not allocated to meet demand at this node. Empty trucks are moved only to meet potential demand, therefore can be reduced if necessary. Lastly, loaded trucks are re-allocated so as to shift excess demand. This priority structure allows for the minimum amount of allocated demand to be shifted while achieving feasibility. In summary, to eliminate $B_{l_1}^{k,a}(t)$ at node (l_1, t) efficiently, outgoing arcs are examined in the above order. This is shown algorithmically in the following steps.

Define $adj\_S_{l_1}^{k,a}(t)$, $adj\_I_{l_1}^{k,a}(t)$, $adj\_X_{l_1 l_2}^{k,a}(t)$, and $adj\_\overline{d}_{l_1 l_2}^{k',k,a}(t)$ as the reduction in flows of salvage, held inventory at node (l, t), empty movement and loaded movement from (l_1, t) to (l_2, t), respectively. Also, define $NB(l_1, t)$ as the new balance at node (l_1, t).

Specifically, the algorithm is as follows in the RFS application:

0. Initialization: Set $NB(l_1, t) = 0$, for all (l_1, t). Start from $t = 2$, $l_1 = 1$.

1. Check used truck purchase $B_{l_1}^{k,a}(t)$. If $B_{l_1}^{k,a}(t) = 0$, go to 6; Else, let $NB(l_1, t) = NB(l_1, t) + B_{l_1}^{k,a}(t)$.

2. If $S_{l_1}^{k,a}(t) = 0$, go to 3; If $S_{l_1}^{k,a}(t) > NB(l_1, t)$, all the used truck purchase can be eliminated. Let $adj\_S_{l_1}^{k,a}(t) = NB(l_1, t)$, $S_{l_1}^{k,a}(t) = S_{l_1}^{k,a}(t) - NB(l_1, t)$, and $NB(l_1, t) = 0$; Else, update $NB(l_1, t) =$

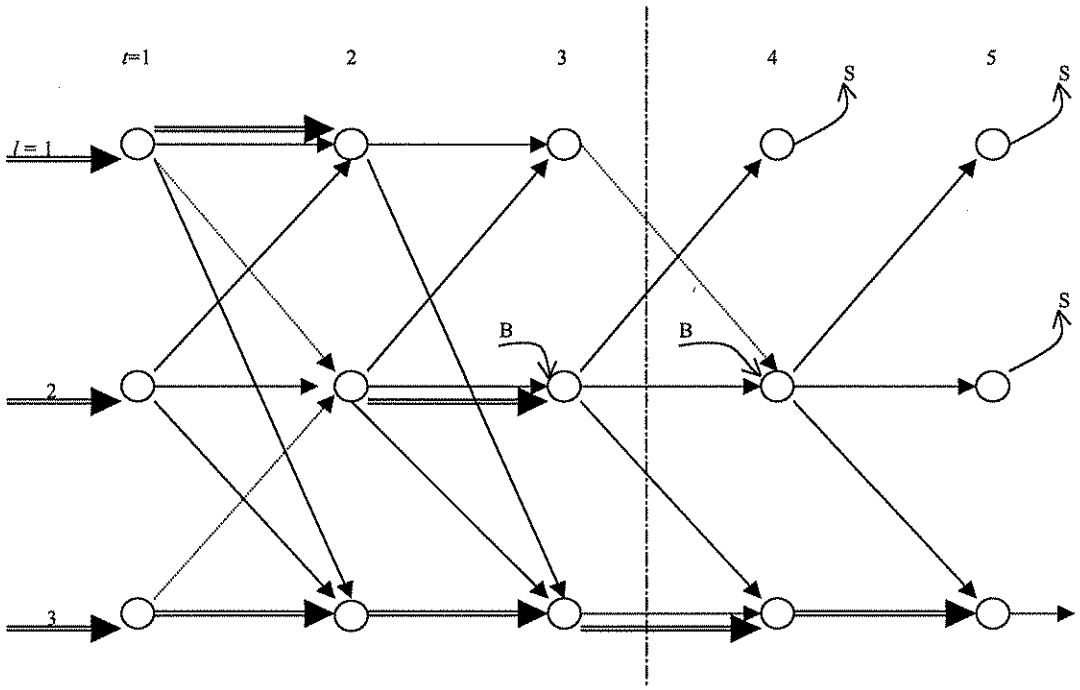

Figure 3: Illustration of Truck Flows in the Time-Space ($H = 5, L = 3$) Network

$NB(l_1,t) - S^{k,a}_{l_1}(t)$, let $adj\_S^{k,a}_{l_1}(t) = S^{k,a}_{l_1}(t)$, and $S^{k,a}_{l_1}(t) = 0$. If updated $NB(l_1,t) > 0$, go to 3; Else go to 6.

3. If $I^{k,a}_{l_1}(t) = 0$, go to 4; If $I^{k,a}_{l_1}(t) > NB(l_1,t)$, all the used truck purchase can be eliminated. Let $adj\_I^{k,a}_{l_1}(t) = NB(l_1,t)$, $I^{k,a}_{l_1}(t) = I^{k,a}_{l_1}(t) - NB(l_1,t)$, and $NB(l_1,t) = 0$; Else, update $NB(l_1,t) = NB(l_1,t) - I^{k,a}_{l_1}(t)$, let $adj\_I^{k,a}_{l_1}(t) = I^{k,a}_{l_1}(t)$, and $I^{k,a}_{l_1}(t) = 0$. If updated $NB(l_1,t) > 0$, go to 4; Else go to 6.

4. For each location $l_2 = 1, ..., L$ ($l_1 \neq l_2$), if $X^{k,a}_{l_1 l_2}(t) = 0$, let $l_2 = l_2 + 1$ and go to 4; Else, if $X^{k,a}_{l_1 l_2}(t) > NB(l_1,t)$, all the used truck purchase can be eliminated. Let $adj\_\ X^{k,a}_{l_1 l_2}(t) = NB(l_1,t)$, $X^{k,a}_{l_1 l_2}(t) = X^{k,a}_{l_1 l_2}(t) - NB(l_1,t)$, and $NB(l_1,t) = 0$; Else, update $NB(l_1,t) = NB(l_1,t) - X^{k,a}_{l_1 l_2}(t)$, let $adj\_\ X^{k,a}_{l_1 l_2}(t) = X^{k,a}_{l_1 l_2}(t)$, and $X^{k,a}_{l_1 l_2}(t) = 0$. If updated $NB(l_1,t) > 0$ and $l_2 + 1 < L$, let $l_2 = l_2 + 1$ and go to 4. If updated $NB(l_1,t) > 0$ and $l_2 + 1 = L$, go to 5; Else go to 6.

5. For each location $l_2 = 1, ..., L$, if $\overline{d}^{k',k,a}_{l_1 l_2}(t) = 0$, let $l_2 = l_2 + 1$ and go to 5; Else, if $\overline{d}^{k',k,a}_{l_1 l_2}(t) > NB(l_1,t)$, all the used truck purchase can be eliminated. Let $adj\_\overline{d}^{k',k,a}_{l_1 l_2}(t) = NB(l_1,t)$, $\overline{d}^{k',k,a}_{l_1 l_2}(t) = \overline{d}^{k',k,a}_{l_1 l_2}(t) - NB(l_1,t)$, and $NB(l_1,t) = 0$; Else, update $NB(l_1,t) = NB(l_1,t) - \overline{d}^{k',k,a}_{l_1 l_2}(t)$, let $adj\_\overline{d}^{k',k,a}_{l_1 l_2}(t) = \overline{d}^{k',k,a}_{l_1 l_2}(t)$ and $\overline{d}^{k',k,a}_{l_1 l_2}(t) = 0$. If updated $NB(l_1,t) > 0$ and $l_2 + 1 < L$, let $l_2 = l_2 + 1$ and go to 5.

6. Let $l_1 = l_1 + 1$. If $l_1 \leq L$, go to 1.

7. Let $t = t + 1$. If $t > H$, stop; otherwise, for each location l_1, re-calculate $NB(l_1,t) = NB(l_1,t) + adj\_I^{k,a}_{l_1}(t) + \sum_{l=1}^{L} adj\_\ X^{k,a}_{l,l_1}(t - \omega_{l,l_1}) + \sum_{l=1}^{L} \sum_{\lambda=1}^{\lambda_{\max}} \beta_{ll_1}(\lambda) \sum_{k'=1}^{k} \overline{d}^{k',k,a}_{ll_1}(t - \lambda)$. Go to 1.

Thus, the algorithm terminates after each node is visited and balanced. We show in the following that at most one call of demand-shifting subroutine can produce a feasible solution.

Theorem 3 *The demand-shifting subroutine produces a feasible solution to any subproblem PS(k, a), a > 0, for any demand allocation* $\overline{d}_{ll'}^{k',k,a}(t)$.

Proof. It must be noted, and as shown in Figure 3, that all flow moves from left to right (from one time period to a later time period) in the time-space network. Thus, flow cannot move between two nodes in the same time period. The algorithm sequentially examines each node in each successive time period. If a used asset is purchased, representing an infeasibility, the inflow on the purchase arc is removed and a corresponding outflow is removed to achieve node balance (NB). This removal of an outflow causes an imbalance at another node in a later time period. This imbalance is noted by labeling the node (NB>0). When moving from node to node in successive time periods, the imbalances created in earlier periods are removed. Thus, when all nodes in a given period have been examined, there are no imbalances. Moreover, no further imbalances in that given period can be created as all flow is from left to right. As the algorithm terminates in the last period, there cannot be a node with an imbalance and there are no used asset purchases. Thus, the subproblem cannot be infeasible. ■

The above is a single-step procedure in that the path of the excess demand allocation is determined only by the prespecified order of action at the current node. We can also look ahead more than one period when excess demand allocation has more than one path to reach the next time-space node. This requires a change in Step 5 of the above algorithm, as not only the current node but also all the next-period destination nodes are examined. Referred to as a two-step scheme, each of the possible destination nodes is first ranked by the prespecified order of actions. Choose l_2 if it has the highest rank. Same as in Step 5, if $\overline{d}_{l_1 l_2}^{k',k,a}(t) > NB(l_1, t)$, all the used truck purchase can be eliminated. Let $adj\_\overline{d}_{l_1 l_2}^{k',k,a}(t) = NB(l_1, t)$, $\overline{d}_{l_1 l_2}^{k',k,a}(t) = \overline{d}_{l_1 l_2}^{k',k,a}(t) - NB(l_1, t)$, and $NB(l_1, t) = 0$; Else, update $NB(l_1, t) = NB(l_1, t) - \overline{d}_{l_1 l_2}^{k',k,a}(t)$, let $adj\_\overline{d}_{l_1 l_2}^{k',k,a}(t) = \overline{d}_{l_1 l_2}^{k',k,a}(t)$ and $\overline{d}_{l_1 l_2}^{k',k,a}(t) = 0$. If updated $NB(l_1, t) > 0$, reduce the rank by one, choose another l_2 with the second highest rank and repeat the same procedure.

For example, refer to Figure 3, $B_2^{k,a}(3) > 0$. There are three loaded movement arcs $((2,3),(1,4))$, $((2,3),(2,4))$ and $((2,3),(3,4))$. As $S_1^{k,a}(4) > 0$, $I_3^{k,a}(4) > 0$, and $\overline{d}_{2,l_2}^{k',k,a}(4) > 0$ for $l_2 = 1, 2, 3$, the ranking is $l_2 = 1$ highest, $l_2 = 3$ second highest, and $l_2 = 1$ at last. Flow is removed according to this ranking.

Theorem 4 *The worst-case complexity of the (single or two-step) demand-shifting subroutine is* $O(L^2H)$ *in one subproblem iteration of the RFS problem.*

Proof. (1) Singe-step scheme:

The single-step scheme examines each node in the network at most once. As there are $L * (H - 1)$ nodes in the network, it takes $O(LH)$ steps for each subproblem PS(k, a). At each node, two examinations occur: (1) if a used asset has been purchased; and (2) if the removal of a used asset purchase in a previous period

leads to a node imbalance. Thus, when examining a node, the node balance must be examined and the incoming "used asset purchase arc" are examined. If the node is imbalanced and/or a used asset purchase arc is positive, an existing arc must be eliminated to achieve balance. This occurs in the order specified earlier. In the worst case, all exiting arcs are examined, totaling $2+2L$ arcs, or $O(L)$. Thus, the complexity of the single step scheme is $O(L^2H)$.

(2) Two-step scheme:

The only difference in the two-step scheme is that more exiting arcs are examined at each node. Specifically, $2+9L$ arcs are examined. Thus, the resulting complexity is the same ($O(L^2H)$). ■

Of course, the algorithm may be further generalized such that paths are traced to the end of the time horizon at each instance. The trade-off to be examined is the solution accuracy versus solution time.

4 Computational Results

The Benders procedure with the demand-shifting feasibility algorithm is coded in Visual C/C++ with the CPLEX callable library [ILOG, 1999] on a PC with 300 Mhz Pentium II processor with 64 MB RAM. The network structure of the subproblems is exploited through use of the network simplex method. The master problem was solved with the dual simplex method to take advantage of the basis from the previous iteration.

The two presented algorithmic schemes are compared to the classical feasibility cut approach. Results are summarized for a fleet consisting of trucks which K=3 (type) and N=2 (maximum age) in Tables 1 and 2, where ratio gap = (UB-LB)/UB. In the larger problems, the solutions produced average fleet sizes near 11,000 trucks. As shown in the tables, the demand-shifting algorithm leads to better solution gaps much more quickly than the traditional method. (It should be noted that the negative objective function values achieved in Table 1 are due to a large initial fleet for relatively low demand over a short time horizon, leading to the sale of excess trucks at a profit.)

For H=5 and L=3, as shown in Table 1, the average CPU time for the single-step and two-step algorithms is 10.38 and 11.19 seconds, respectively. The average CPU time for the traditional feasibility cut method is about 316 seconds. Similarly, for H=30 and L=10, as shown in Table 2, the average CPU time for single-step and two-step schemes is 552.7 and 545.9 seconds, respectively, while the average CPU time for the traditional feasibility cut method is about 2,112 seconds.

For another frame of reference, we solved the H=30 and L=10 problems using the approach that excess (infeasible) capacity could be acquired at an arbitrarily high price (100*new asset purchase price). For 5 instances, it took on average over one hour to complete eight iterations with an average ratio gap of 7.03. Note that no feasibility cuts were added in this procedure.

We expected that the two-step demand-shifting scheme would perform better than the single-step. However, the overall testing performance shows that single-step is more efficient regarding solution quality and CPU time. We justify this by the facts that: (1) very few used truck purchases are required with the addition of the valid inequalities (12) and (13); and (2) the two-step demand-shifting scheme is only applicable

for the situation when multiple loaded trucks at a given location move towards different destination nodes. Thus, it may prove better in larger problems with more locations. Nevertheless, both schemes significantly outperform the traditional method.

Table 1: Computational Comparison for H=5 and L=3

Instance	Scheme	UB	LB	Ratio Gap	Iterations	CPU (sec.)
1	Single-Step	1067316	1067216	0.000094	32	8.52
	Two-Step	1067228	1067208	0.000019	36	7.58
	Feasibility Cut	1067271	1067036	0.00022	200	~330
2	Single-Step	-140003	-140003	0	39	8.79
	Two-Step	-140003	-140003	0	47	9.5
	Feasibility Cut	-139280	-139998	-0.0051	200	~320
3	Single-Step	72348	72348	0	49	10.99
	Two-Step	72350	72347	0.000042	49	10.98
	Feasibility Cut	75057	71984	0.0409	200	~330
4	Single-Step	68227	68220	0.0001	55	12.79
	Two-Step	68222	68221	0.000015	60	15.43
	Feasibility Cut	505848	63714	0.874	200	~300
5	Single-Step	-238815	-238838	-0.000096	56	10.82
	Two-Step	-238818	-238829	-0.000046	61	12.46
	Feasibility Cut	-53578	-242323	-3.523	200	~300

Table 2: Computational Comparison for H=30 and L=10

Instance	Scheme	UB	LB	Ratio Gap	Iterations	CPU (sec.)
1	Single-Step	256516753	243619438	0.05027	10	508.55
	Two-Step	253716614	243495563	0.04028	10	404.69
	Feasibility Cut	280696963	243591045	0.1322	20	~2760
2	Single-Step	249513384	237681515	0.04742	10	467.64
	Two-Step	251021672	237715689	0.05301	10	627.36
	Feasibility Cut	274426670	235568562	0.1416	20	~1620
3	Single-Step	250591342	238392725	0.04868	10	590.78
	Two-Step	250144989	238342657	0.04718	10	541.9
	Feasibility Cut	277381207	237503074	0.1438	20	~2400
4	Single-Step	258285801	246751859	0.04466	10	457.53
	Two-Step	259161228	246770508	0.04781	10	571.56
	Feasibility Cut	284532730	246727764	0.1329	20	~1800
5	Single-Step	250516852	239234283	0.04504	10	738.91
	Two-Step	251499288	239160919	0.04906	10	583.97
	Feasibility Cut	276369832	237127017	0.1420	20	~1980

5 Conclusions and Future Research

We present a demand-shifting feasibility algorithm to quickly find feasible solutions to infeasible subproblems in the context of a Benders decomposition procedure. Development of the algorithm was motivated by work on a rental fleet sizing problem which was decomposed by truck type and age. As a policy did not allow for the acquisition of used trucks, the subproblem capacities were bound, thus leading to the possibility of infeasibilities in the subproblems. When compared to a traditional feasibility cut scheme, the benefits of demand-shifting feasibility algorithm are numerous in that: (1) each subproblem is solved only once in each Benders iteration as feasibility is assured with the demand-shifting algorithm; (2) no feasibility cuts are added into the master problem; (3) the master problem need not be reformulated and solved repeatedly to find feasible solutions of the subproblems.

Computational experience shows that demand-shifting algorithm leads to efficient solutions within a traditional Benders decomposition framework. This is illustrated through a computational study on a dynamic fleet sizing model with static considerations, such as asset purchases and disposals. It is believed that the algorithm would be applicable to any problem in which the subproblems have limited capacity and assuring feasible master problem allocations is not trivial. In the fleet sizing model, the allocation is complicated due to the dynamics of moving assets through space and time. Furthermore, this algorithm can also be applied as a primal heuristic to find a feasible upper bound solution for the Lagrangean relaxation approach.

As we pointed out earlier, the algorithm may be generalized to more than two-step, such that paths

would be traced to the end of the time horizon at each instance. Further computational results on multi-step schemes are to be tested, especially for larger problems.

References

Aardal, K., Larsson, T., 1990. A Benders Decomposition Based Heuristic for the Hierarchical Production Planning Problem, European Journal of Operational Research 45, 4-14.

Ahuja, R.K., Magnanti, T.L., Orlin, J.B., 1993. Network Flows, Theory, Algorithms, and Applications, Prentice-Hall, Inc.

Bazaraa, M.S., Sherali, H.D., 1980. Benders' Partitioning Scheme Applied to a New Formulation of the Quadratic Assignment Problem, Naval Research Logistics Quarterly 27(1), 29-41.

Beaujon, George J., Turnquist, Mark A., 1991. A Model for Fleet Sizing and Vehicle Allocation, Transportation Science 25(1), 19-45.

Benders, J.F., 1962. Partitioning Procedures for Solving Mixed-Variables Programming Problems, Numerische Mathematik 4, 238-252.

Birge, J. R., Louveaux, F., 1997. Stochastic Programming, Springer-Verlag New York, Inc.

Chen, Z., 1998. Solution Algorithm for the Parallel Replacement Problem under Economy Scale, Naval Research Logistics 45, 279-295.

Geoffrion, A.M., Graves, G.W., 1974. Multicommodity Distribution System Design by Benders Decomposition, Management Science 20(5), 822-844.

Holmberg, K., 1994. On Using Approximations of the Benders Master Problem, European Journal of Operational Research 77, 111-125.

ILOG, 1999. ILOG CPLEX 6.5, Reference Manual.

Klincewicz, John G., Luss, Hanan, Pilcher, Martha G., 1990. Fleet Size Planning When Outside Carrier Services Are Available, Transportation Science 24(3), 169-182.

Magnanti, T.L., Simpson, R.W., 1978. Transportation Network Analysis and Decomposition Methods, U.S. Department of Transportation, DOT-TSC-RSPD-78-6.

Magnanti, T.L., Wong, R.T., 1981. Accelerating Benders Decomposition: Algorithmic Enhancement and Model Selection Criteria, Operations Research 29(3), 464-484.

Minoux, M., 1984. Subgradient Optimization and Benders Decomposition for Large Scale Programming, Mathematical Programming, Cottle, R.W., Kelmanson, M.L. and Korte, B. (Editors), 271-288.

Turnquist, Mark A., Jordan, William C., 1986. Fleet Sizing under Production Cycles and Uncertain Travel Times, Transportation Science 20(4), 227-236.

Van Slyke, R., Wets, R. J-B, 1969. L-Shaped Linear Programs with Application to Optimal Control and Stochastic Programming, SIAM Journal on Applied Mathematics 17, 638-663.