A LAGRANGIAN ALGORITHM FOR THE PARALLEL REPLACEMENT PROBLEM WITH CAPITAL RATIONING CONSTRAINTS

Nejat Karabakal

Jack R. Lohmann

James C. Bean

Technical Report 91-22

August 1991
Revised May 1992

A LAGRANGIAN ALGORITHM FOR THE PARALLEL REPLACEMENT PROBLEM WITH CAPITAL RATIONING CONSTRAINTS *

NEJAT KARABAKAL, JACK R. LOHMANN AND JAMES C. BEAN

Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109
School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, Georgia 30332
Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, Michigan 48109

August 1991
(Revised May 1992)

Abstract

Contrary to serial replacement, parallel replacement problems require a decision maker to evaluate a portfolio of replacement decisions in each time period because of economic interdependencies among assets. In this paper, we describe a parallel replacement problem in which the economic interdependence among assets is caused by capital rationing. The research was motivated by the experience gained from a vehicle fleet replacement study where solutions to serial replacement problems could not be implemented since they violated management's budget plan. When firms use budgets to control their expenditures, competition for the limited funds creates interdependent problems. In this paper, we formulate the problem as a zero-one integer program and develop a branch-and-bound algorithm based on Lagrangian relaxation methodology. A multiplier adjustment method is developed to solve one Lagrangian dual.

Keywords: EQUIPMENT REPLACEMENT; CAPITAL RATIONING; INTEGER PROGRAMMING; LAGRANGIAN RELAXATION; MULTIPLIER ADJUSTMENT METHOD.

*This research was supported in part by National Science Foundation Grants PYI-G-DMC-8352346 and DMM-9018515 to The University of Michigan and by the Great Lakes Center for Truck Transportation Research.

1 Introduction

The replacement problem typically involves a required service provided by one or more assets over either a finite or infinite time horizon. The decision is to determine replacement schedules for individual assets so that a particular measure of economy—usually the net present value (NPV)—is optimized. A replacement schedule specifies a) whether to keep an existing asset (the defender), if one exists, or to replace it immediately with one of the new assets (current challengers), b) a sequence of future challengers to be installed after the current decision, and c) how long each asset in the sequence is to be kept in service.

There is a rich literature on replacement problems. The vast majority of it studies the serial replacement problem. In serial replacement, it is assumed that there is no economic interdependence among the assets that provide the service together so that their replacement decisions can be made separately. The commonly-used traditional treatment considers serial replacement with infinite horizon time and deterministic cash flows. It also assumes that any current challenger would be replaced repeatedly with identical assets in the future. Although it has a simple closed-form solution, its assumptions are usually too restrictive to justify in most problems. Major works that progressively relaxed the restrictive assumptions of the traditional treatment include a) early infinite horizon models that incorporate the effects of inflation and technological improvements but require constant replacement intervals for the current and future challengers (Terborgh 1949; Oakford 1970), b) finite horizon dynamic programming (DP) formulations that completely relax the repeatibility assumption (Wagner 1975; Oakford, Lohmann, and Salazar 1984), c) simulation modeling for uncertain cash flows (Lohmann 1986), and d) planning horizon approaches for the infinite horizon case without the repeatibility assumption (Sethi and Chand 1979; Chand and Sethi 1982; Bean, Lohmann, and Smith 1985, 1990; Hopp and Nair 1991).

Contrary to the case of serial replacement, parallel replacement problems require a decision maker to evaluate a portfolio of replacement decisions in each time period because of economic interdependencies among assets. The economic interdependence

can be caused by various factors. Vander Veen (1985) studied a parallel replacement problem in which the economic interdependence resulted from the requirement of satisfying a prespecified demand and by keeping one or more assets in service at all times. Jones, Zydiak, and Hopp (1991) discussed another economic interdependence caused by a fixed cost that is incurred whenever one or more assets are replaced. A typical use of the fixed replacement cost is to model the economies of scale effect associated with some capital investment problems.

In this paper, we describe another parallel replacement problem in which the economic interdependence among the assets is caused by capital rationing. In serial replacement, it is common to assume that the firm has sufficient capital so that, for all individual assets, indicated capital replacement expenditures can be financed in any time period over the planning horizon. In practice, however, firms frequently use *budgets* to control their expenditures. In this case, it is necessary to consider all replacement decisions in each time period together since competition for the limited funds creates interdependent problems. We call this problem the "capital rationing replacement problem" (CRRP). The research was motivated by a vehicle fleet replacement study with a major utility company in Michigan. Recommended replacement actions were obtained by solving individual serial replacement problems. However, they could not be implemented since they violated management's budget plan.

Parallel replacement problems are more difficult to solve than serial replacement problems because evaluating all replacement decisions together creates a difficult combinatorial problem. Even for reasonably small problems, the set of all combinations is typically so large that the cost of computing an optimal solution using total enumeration is prohibitive. In this research, we formulate the finite horizon, deterministic version of the CRRP as a zero-one integer program and develop a branch-and-bound algorithm based on the Lagrangian relaxation methodology. The major contributions of this work are the multiplier adjustment method to solve one Lagrangian dual and an implementation capable of solving moderately-sized problems.

The remainder of this paper is organized as follows. In the following section, we discuss the nature of the underlying capital rationing environment and its role in making

replacement decisions. In §3, we present an integer programming formulation of the CRRP. Two Lagrangian relaxations are described in §4 along with efficient algorithms to solve them. Next, §5 suggests a simple heuristic procedure for quick lower bounding. In §6, we discuss procedures to determine nonoptimal variables at the outset in an attempt to reduce problem size. Procedures of §4 through §6 are synthesized into a branch-and-bound algorithm in §7 and computational experience with this algorithm is reported. Finally, §8 provides a summary and outlines future research directions.

2 Capital Rationing Environment

Until the well-known article of Lorie and Savage (1955), limited availability of capital was not a concern in most work on capital investment modeling. Their work was very influential in attracting attention to a firm confronted with a variety of possible investment projects and a fixed capital budget. Next, Weingartner (1963, 1966) formulated the Lorie-Savage problem as an integer program which maximized the NPV of project cash flows subject to capital rationing constraints.

In general, capital rationing constraints can be imposed externally or internally. If they are externally imposed, they imply market imperfections in the sense that the firm cannot raise more money from capital markets. As Weingartner (1966) explained, this type of rationing is rare and usually temporary. On the other hand, internally-imposed capital rationing constraints are provisional limits commonly adopted by management as an aid to financial control. For example, some ambitious divisional managers may overstate their investment opportunities. Rather than trying to distinguish which projects really are worthwhile, their corporate headquarters may find it simpler to impose an upper limit on divisional expenditures and thereby force the divisions to set their own priorities. In other cases, management may believe that very rapid corporate growth could be harmful to management and the organization. Since it is difficult to quantify such constraints explicitly, the budget limit may be used as a proxy (Brealey and Myers 1988). Other common reasons for internal rationing were summarized by Gurnani (1984). They included: a) debt limits imposed by internal

management or by an outside agreement, b) inadequate cash in-flows from operations, c) maintenance of certain price/earning ratios, and d) dividend payout policies. In this research, we assume that capital rationing constraints are imposed internally. They do not represent "hard" bounds in the sense of an absolute limit on finance. Rather, they are provisional limitations imposed for the purpose of controlling replacement expenditures. The expenditures to be controlled by capital rationing constraints are capital costs to purchase new assets.

We also assume that a) salvage values of discarded assets, if any, are not added to subsequent budgets, b) remaining (unspent) budget is not carried forward, and c) borrowing is not allowed. Each of these assumptions is consistent with the economic principle of seperating financing decisions from investment decisions. This issue was also raised by Weingartner (1966), "...as a matter of good business practice, the cash in-flows resulting from project adoption should not be made available for reinvestment without first passing through those control channels which set the expenditure ceilings in the first place." From this viewpoint, we believe many managers responsible for replacement planning do not consider those funds obtained from salvage values readily available for financing replacement expenditures. Further, in many replacement problems, salvage and trade-in values are so insignificant that they are considered zero. The latter was the case with the vehicle fleet replacement study that motivated this research. However, there are other replacement problems for which above assumptions may not be appropriate. The following development is not intended to address those problems.

3 The Model

The CRRP can be stated as an integer program that provides a basis for further algorithmic developments. We assume that all replacements and cash transactions occur at the end of the time periods, and that the end of period zero refers to the current time. Let

H = time horizon

n = number of assets in the group

α = discount rate per period

m_a = number of challengers for asset a

N_{ac} = maximum service life of challenger type c of asset a,
$a = 1$ to n, $c = 0$ to m_a ($c = 0$ denotes the defender, if any)

P_{aci} = capital costs to purchase challenger type c of asset a in period i

B_i = budget in period i, $i = 0$ to $H - 1$

π_{acij} = NPV of acquiring challenger type c of asset a in period i
and using it until period j,
$i = 0$ to $H - 1$ and $j = i + 1$ to $\min\{H, i + N_{ac}\}$

To avoid the added computational complexity from consideration of taxes, we assume all quantities to be before taxes. However, the impact of taxation can easily be incorporated into the model by using after-tax cash flows in NPV computations and adjusting the other parameters accordingly. In determining π_{acij} values, we follow an approach similar to that given in Oakford, Lohmann, and Salazar (1984). Therefore

$$\pi_{acij} = F(a, c, i)\,\mathrm{NPV}(a, c, j - i) / (1 + \alpha)^i$$

where, for asset a, $F(a, c, i)$ is a functional relationship to relate future challengers acquired at $i > 0$ to the current challenger c and $\mathrm{NPV}(a, c, p)$ is the NPV of p-periods of usage for the current challenger c which can be acquired now. We note here that $F(a, c, i)$ represents changes over time and is typically used for modeling the effects of inflation and technological improvements on future assets. It does *not* associate a future challenger's cash flow series from period i onward with the condition of the asset in service in period i and/or with the replacement decisions made before period i. Such associations cannot be represented trivially using functional relationships and are discussed in §8 as extensions of the current model.

The decision variables are

$$x_{acij} = \begin{cases} 1 & \text{if the challenger type } c \text{ of asset } a \text{ is acquired in period } i \\ & \text{and used until period } j \\ 0 & \text{otherwise} \end{cases}$$

The defender can be modeled as a special challenger available only at the current time. Further, we suppose $\pi_{acij} = -\infty$ whenever $c = 0$ and $i > 0$. We also define two index sets:

$$\begin{aligned} \mathcal{F}_{aci} &= \{j \mid i < j \leq \min\{H, i + N_{ac}\}\} \\ \mathcal{P}_{aci} &= \{j \mid \max\{0, i - N_{ac}\} \leq j < i\} \end{aligned}$$

Then, the CRRP can be formulated as an integer program as follows:

$$\text{Maximize NPV} = \sum_{a=1}^{n} \sum_{c=0}^{m_a} \sum_{i=0}^{H-1} \sum_{j \in \mathcal{F}_{aci}} \pi_{acij}\, x_{acij}$$

subject to:

1) Asset sequencing constraints: for each $a = 1$ to n

$$\sum_{c=0}^{m_a} \sum_{j \in \mathcal{F}_{ac0}} x_{ac0j} = 1 \tag{1}$$

$$\sum_{c=0}^{m_a} \sum_{j \in \mathcal{F}_{aci}} x_{acij} - \sum_{c=0}^{m_a} \sum_{j \in \mathcal{P}_{aci}} x_{acji} = 0, \qquad \text{for } i = 1 \text{ to } H-1 \tag{2}$$

$$-\sum_{c=0}^{m_a} \sum_{j \in P_{acH}} x_{acjH} = -1 \tag{3}$$

2) Capital rationing (budget) constraints: for each $i = 0$ to $H - 1$

$$\sum_{a=1}^{n} \sum_{c=0}^{m_a} P_{aci} \sum_{j \in \mathcal{F}_{aci}} x_{acij} \leq B_i \tag{4}$$

3) Multiple choice constraints: for each $a = 1$ to n

$$\sum_{c=0}^{m_a} \sum_{j \in \mathcal{F}_{ac0}} x_{ac0j} = 1 \tag{5}$$

$$\sum_{c=0}^{m_a} \sum_{j \in \mathcal{F}_{aci}} x_{acij} \leq 1 \qquad \text{for } i = 1 \text{ to } H-1 \tag{6}$$

4) Integrality constraints: for all a, c, i, j

$$x_{acij} \in \{0, 1\} \tag{7}$$

The asset sequencing constraints (or flow conservation constraints in network terminology) can be interpreted as project interdependencies; i.e. constraint sets (1) and (3) define mutually exclusive projects whereas constraint set (2) defines contingency relationships. Capital rationing constraint set (4) limits purchasing new assets in each time period. Multiple choice constraints (5) and (6) prevent replacing assets with more than one challenger and are redundant given the asset sequencing constraints. However, they will prove useful in a subsequent Lagrangian relaxation approach in which the asset sequencing constraints are relaxed.

The above formulation has a network characterization. Consider a directed graph where nodes represent the end of periods and arcs represent replacement decisions. Associated with each arc are two parameters, length (net present value benefit of replacement, π_{acij}) and resource consumption (purchase cost, P_{aci}). This network structure is illustrated in Figure 1 for a particular asset with a three-period planning horizon. Here, asset a has two challengers: challenger 1 can be used until the horizon time, whereas challenger 0 (defender) has a remaining life of two periods. We have as many such graphs as the number of assets. Suppose that the last node of each asset's graph (node H) is connected to the first node of the next asset's graph (node 0) with a dummy arc of length zero, except for the last asset. The problem then is to find the longest path from the first asset's first node to the last asset's last node in such a way that no resource (budget) constraint is violated.

4 Lagrangian Relaxation Approaches

Lagrangian relaxation methods are based on the observation that many hard integer programming problems can be viewed as easy problems with a set of "complicating" side constraints. Dualizing such constraints produces a Lagrangian relaxation of the original problem. The Lagrangian problem yields an upper bound (for maximization

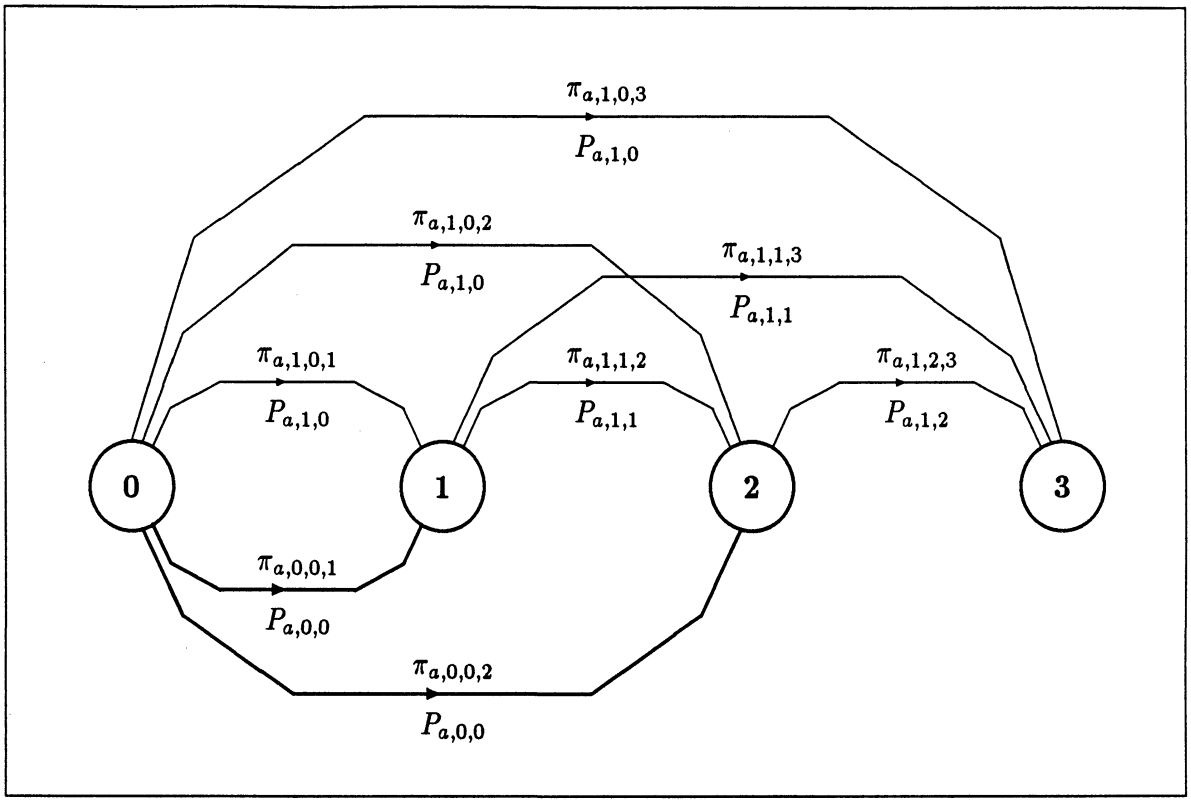

Figure 1: Network structure of a particular asset's replacement problem

problems) on the optimal value of the original problem. Foundations of the Lagrangian relaxation theory and some early successful applications are summarized in expository papers, such as Fisher (1985 and 1981), Shapiro (1978), and Geoffrion (1974).

We use two Lagrangian relaxations for the CRRP based on its integer programming formulation. First, when we relax the capital rationing constraints, we obtain a relaxation with the integrality property which is useful to solve the linear programming (LP) relaxation of the original problem approximately (Geoffrion 1974). Obtaining a quick dual feasible solution for the LP relaxation is important in a variable reduction algorithm to be described later. Additionally, given the assumption that the capital rationing constraints are imposed primarily for expenditure control purposes, and hence soft, the Lagrangian problem may produce acceptable solutions. Second, when we relax the asset sequencing constraints, we obtain a relaxation without the integrality property, and thus, useful to obtain tighter upper bounds for the original problem.

4.1 Relaxing Budget Constraints

The first relaxation is obtained by dualizing the set of budget constraints (4) with $\mu \geq 0$, where μ_i is the multiplier associated with the budget constraint of period i.

$$\underline{\text{PROBLEM } (LR_\mu)}: \quad L(\mu) = \sum_i B_i \mu_i + \max_{x \in S_\mu} \left\{ \sum_a \sum_c \sum_i \sum_j (\pi_{acij} - P_{aci}\mu_i)\, x_{acij} \right\}$$

where $S_\mu = \{x \mid x \text{ satisfies (1), (2), (3), and (7)}\}$. Having constructed the Lagrangian function, we can formulate the Lagrangian dual problem to find the best upper bound:

$$\underline{\text{PROBLEM } (LD_\mu)}: \qquad \min_{\mu \geq 0} L(\mu)$$

Solving PROBLEM (LR_μ) For A Given μ

Relaxing capital rationing constraints eliminates the interdependency of replacement decisions among assets. Therefore, the problem is reduced to n separate replacement problems, each of which is that of finding a longest path on an acyclic graph. We use a DP to solve efficiently each longest path problem. For a given $\mu \geq 0$, let $\bar{\pi}_{acij} = \pi_{acij} - P_{aci}\mu_i$ for all a, c, i, j. Set $\text{NPV}(a, 0) = 0$ for all a. For each a, the following recursive equations find a longest path from node 0 to H:

$$\text{NPV}(a, j) = \max_c \left\{ \max_{i=j-1,\ldots,\underline{i}} \{ \bar{\pi}_{acij} + \text{NPV}(a, i) \} \right\} \qquad \text{for } j = 1,\ldots,H$$

where $\underline{i} = \max\{0, j - N_{ac}\}$. The Lagrangian function value is the sum of individual longest paths plus a constant term:

$$L(\mu) = \sum_a \text{NPV}(a, H) + \sum_i B_i \mu_i$$

Solving PROBLEM (LD_μ)

Finding the best multiplier vector for the solution of a Lagrangian dual is a nondifferentiable optimization problem. Subgradient algorithms have been used on many practical problems successfully. Previous applications include the traveling salesman problem (Held and Karp 1971), the set covering problem (Etcheberry 1977), and resource-constrained assignment scheduling (Mazzola and Neebe 1986). The basic step of a

subgradient algorithm requires solving the Lagrangian relaxation problem to compute a subgradient direction for the multipliers. The multipliers are then changed in the computed direction. Details of subgradient algorithms including convergence properties can be found in Held, Wolfe and Crowder (1974), and Goffin (1977).

To solve the Lagrangian dual problem (LD_μ), we implemented the subgradient algorithm as in Fisher (1981). At iteration k, a subgradient $\xi^k = (\xi_i^k)$ is found from

$$\xi_i^k = B_i - \sum_a \sum_c P_{aci} \sum_j x_{acij}^k \qquad \text{for each } i = 0, \ldots, H-1$$

where x^k is optimal for $L(\mu^k)$.

4.2 Relaxing Asset Sequencing Constraints

The second relaxation is obtained by dualizing the asset sequencing constraints (1)–(3) with λ, where λ_{ai} is the multiplier associated with the asset sequencing constraint of period i for asset a.

PROBLEM (LR_λ):

$$L(\lambda) = \sum_a (\lambda_{a0} - \lambda_{aH}) + \max_{x \in S_\lambda} \left\{ \sum_a \sum_c \sum_i \sum_j (\pi_{acij} - \lambda_{ai} + \lambda_{aj})\, x_{acij} \right\}$$

where $S_\lambda = \{x \mid x \text{ satisfies (4), (5), (6), and (7)}\}$. The Lagrangian dual problem corresponding to this relaxation is given by

PROBLEM (LD_λ): $\qquad \min_\lambda L(\lambda)$

Solving PROBLEM (LR_λ) For A Given λ

Relaxing the asset sequencing constraints yields a pure capital rationing problem with a set of mutually exclusive replacement projects. Note that the contingency constraints are not enforced. We observe two special characteristics of these replacement projects that facilitate the solution of the Lagrangian problem significantly:

1. The replacement projects included in a particular period's budget constraint do not appear in any other period's budget constraint. Hence, we can solve each period's project selection problem independently from other periods. This property

decomposes the computation of $L(\lambda)$ into H separate, single-period capital rationing problems with mutually exclusive projects, also known as multiple choice knapsack problems (MCKPs).

2. At a particular period i, the knapsack weights (purchase costs) of the variables x_{acij}, for a fixed a and c, are the same. Thus, we can replace every occurrence of the partial sum $\sum_j x_{acij}$ with $x_{acij_{\max}}$ where $j_{\max} = \text{argmax}_j\{\pi_{acij} - \lambda_{ai} + \lambda_{aj}\}$.

To evaluate $L(\lambda)$ for a given λ, first let $\bar{\pi}_{acij} = \pi_{acij} - \lambda_{ai} + \lambda_{aj}$ for all a, c, i, j. Then, for each i, solve the following MCKP:

$$\text{maximize} \quad \sum_a \sum_c \bar{\pi}_{acij_{max}} x_{acij_{max}}$$

subject to

$$\sum_a \sum_c P_{aci}\, x_{acij_{max}} \leq B_i$$

$$\sum_c x_{acij_{max}} \leq 1 \qquad \text{for } a = 1, \ldots, n$$

$$x_{acij_{max}} \in \{0, 1\} \qquad \text{for all } a, c$$

The above formulation is valid for all periods except the first one. At $i = 0$, the multiple choice constraints are equalities to enforce replacement (recall that keeping the defender is a special replacement in this context). We give an algorithm to solve the above problem efficiently in the next section. Let the optimal objective be MCKP(i). The Lagrangian function value is the sum of individual knapsack solutions plus a constant term:

$$L(\lambda) = \sum_i \text{MCKP}(i) + \sum_a (\lambda_{a0} - \lambda_{aH})$$

An Algorithm To Solve The MCKP

Consider the following generic MCKP:

$$\text{maximize} \quad \sum_{k=1}^{m} \sum_{j=1}^{n_k} c_{kj}\, x_{kj}$$

$$\text{subject to}$$

$$\sum_{k=1}^{m} \sum_{j=1}^{n_k} a_{kj}\, x_{kj} \leq b$$

$$\sum_{j=1}^{n_k} x_{kj} = 1 \qquad \text{for } k = 1, \ldots, m$$

$$x_{kj} \in \{0,1\} \qquad \text{for all } k, j$$

If any multiple choice constraint is of a less-than-or-equal-to type, we can add a slack with zero objective value and zero knapsack weight to bring it into the above canonical form.

The MCKP is a well studied problem in the literature. A common way of solving it has been the branch-and-bound with upper bounds computed by LP relaxations, e.g. Sinha and Zoltners (1979), Dyer, Kayal and Walker (1984). Bean and Syverson (1990) described a DP recursion for the MCKP by extending the knapsack function definition in Gilmore and Gomory (1966). Here, we pursue an alternate DP approach that provides additional information necessary to the multiplier adjustment method that follows.

Let the state space consist of the set of ordered points (k, β), where k enumerates multiple choice sets ($1 \leq k \leq m$) and β enumerates the knapsack capacity ($0 \leq \beta \leq b$). The knapsack function, $f_k(\beta)$ is defined to be the optimal value of a subproblem which uses multiple choice sets $k+1$ through m with a knapsack capacity of $b - \beta$. The MCKP is solved by finding the longest path between states $(0,0)$ and (m, b) in the DP network. In finding the longest path, we will say that an algorithm that starts from state $(0,0)$ heading for state (m, b) makes a *forward* pass (forward algorithm) whereas an algorithm that starts from state (m, b) heading for state $(0,0)$ makes a *backward* pass (backward algorithm).

We formulate a backward pass using the reaching strategy discussed in Denardo

(1982). For efficient computation of $f_k(\beta)$ values, we require that the knapsack with capacity $b-\beta$ be tight for $k > 1$. We will say a solution is "tight" whenever there is no slack in the knapsack, or equivalently, whenever the knapsack constraint is satisfied as an equality. Computation is saved with this requirement because it limits consideration of slack space in the knapsack to the first stage ($k = 1$). If there is no tight solution for stage $k > 1$ and for a particular value of $b - \beta$, the state (k, β) cannot be reached, then its $f_k(\beta)$ is defined to be $-\infty$. Let $M_k = \{\beta \mid f_k(\beta) \neq -\infty\} \subseteq \{0, 1, 2, \ldots, b\}$, $k = 1, \ldots, m-1$. Then, the following recursive equations solve the MCKP optimally:

$$f_{m-1}(\beta) = \max \left\{ -\infty, \max_j \{c_{mj} \mid \beta + a_{mj} = b\} \right\}$$

for $\beta = 0, 1, \ldots, b$,

$$f_k(\beta) = \max \left\{ -\infty, \max_j \{f_{k+1}(\beta + a_{k+1,j}) + c_{k+1,j} \mid \beta + a_{k+1,j} \in M_{k+1}\} \right\}$$

for $\beta = 0, 1, \ldots, b$, and $k = m-2, m-1, \ldots, 2$, and finally

$$f_1(\beta) = \max \left\{ -\infty, f_1(\beta + 1), \max_j \{f_2(\beta + a_{2j}) + c_{2j} \mid \beta + a_{2j} \in M_2\} \right\}$$

for $\beta = 0, 1, \ldots, b-1$. The separate computation of $f_1(\beta)$ is for any possible slack space necessary to set aside in the knapsack. Finally, the optimal objective value is given by

$$z = \max \left\{ -\infty, \max_j \{f_1(a_{1j}) + c_{1j} \mid a_{1j} \leq b \text{ and } a_{1j} \in M_1\} \right\}$$

If $z = -\infty$, the problem is infeasible.

This particular recursion has special properties that allow efficient implementation of the following multiplier adjustment method. An illustrative numeric example is provided in the appendix.

Solving PROBLEM (LD_λ)

In order to solve the Lagrangian dual problem (LD_λ), we could use a subgradient algorithm in which the subgradient vector at iteration k, $\xi^k = (\xi^k_{ai})$ is calculated as

$$\xi_{a0} = 1 - \sum_c \sum_j x^k_{ac0j}$$

$$\xi_{ai} = \sum_c \sum_j x^k_{acji} - \sum_c \sum_j x^k_{acij} \quad \text{for } i = 1 \text{ to } H-1$$

$$\xi_{aH} = -1 + \sum_c \sum_j x^k_{acjH}$$

for all $a = 1$ to n.

In general, given an arbitrary initial multiplier vector, the convergence of a subgradient algorithm is usually poor. To avoid its slow convergence, many researchers have developed dual ascent or multiplier adjustment methods (MAMs), heuristic algorithms for solving Lagrangian dual problems exploiting the special structure of a particular application, as alternatives to the subgradient-based optimization. Previous applications of MAMs include the uncapacitated facility location problem (Erlenkotter 1978), the set partitioning problem (Chan 1987), and the generalized assignment problem (Fisher, Jaikumar and Van Wassenhove 1986; Guignard and Rosenwein 1989). The advantage of a MAM is that it usually guarantees monotonic improvement of the bound. The disadvantages are a) it depends on a specific problem structure, and b) it cannot guarantee bounds better than those obtained by solving the Lagrangian dual with a subgradient procedure. This suggests that a MAM may serve to initialize multipliers before a subgradient procedure.

Although this initialization scheme is used at every node of the branch-and-bound tree, it is particularly useful at the root node because any node except for the root is given the final multipliers of the parent node to have an advanced start. Below, we describe a MAM for solving the Lagrangian dual (LD_λ) of the CRRP.

A Multiplier Adjustment Method

The basic idea behind the MAM is to reduce infeasibility resulting from the violation of dualized constraints. Since these constraints are equalities, any constraint with a nonzero subgradient vector element indicates infeasibility. Therefore, the multiplier corresponding to any infeasible constraint should be changed to adjust its current penalty. Specifically, given a multiplier vector $\lambda = (\lambda_{ai})$ and a subgradient vector

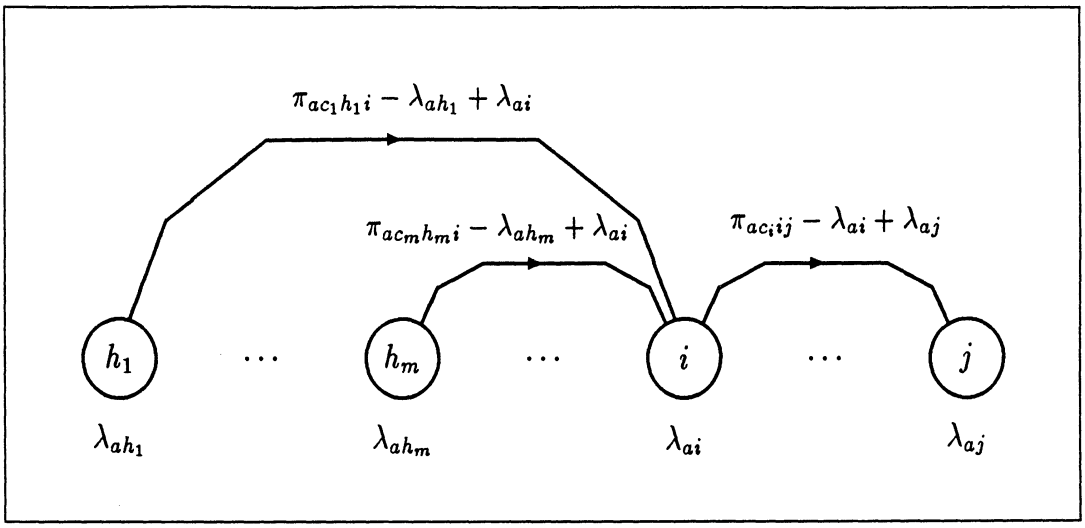

Figure 2: Multiplier λ_{ai} is decreased to reduce infeasibility and to improve $L(\lambda)$.

$\xi = (\xi_{ai})$, we should

$$\text{decrease} \quad \lambda_{ai} \quad \text{if} \quad \xi_{ai} > 0$$

$$\text{increase} \quad \lambda_{ai} \quad \text{if} \quad \xi_{ai} < 0$$

For any asset a, decreasing λ_{ai} has the effect of discouraging too many assets from being retired at i whereas increasing λ_{ai} has the effect of discouraging too many purchases at i. Systematic procedures of adjusting multipliers for a monotonic reduction of such infeasibilities, and hence improving the Lagrangian function $L(\lambda)$ are discussed below.

First, consider the situation shown in Figure 2 which calls for decreasing λ_{ai}. We are looking for a Δ such that decreasing λ_{ai} by Δ will lead to a strict decrease in $L(\lambda)$. Find the set of m nodes $h_1 < h_2 < \ldots < h_m < i$ such that $x_{ac_1h_1i} = x_{ac_2h_2i} = \ldots = x_{ac_mh_mi} = 1$, where c_k is the selected challenger of asset a at h_k, in the current solution. For $x_{acpq} = 0$ in the current optimal solution, let δ_{acpq} be the least amount of increase in $\bar{\pi}_{acpq}$ so that x_{acpq} could be selected as well, creating an alternative optimal solution. Note that $\delta_{acpq} = 0$ for $x_{acpq} = 1$ in the current solution. Leaving the actual computation of δ_{acpq} to a later discussion for the moment, we proceed with defining

$$\Delta_p = \min_{k,c,q} \{ \delta_{a,c,h_k,q} \}$$

where $h_k < q < \min\{h_k + N_{ac}, H\}$ and $c \neq c_k$ for $q = i$. The quantity Δ_p, an upper bound on Δ, shows how much λ_{ai} can be decreased without changing the current MCKP solutions in periods $h_1, h_2, \ldots, h_m$. If there is no replacement of asset a in

period i, let

$$\Delta_f = \min_{c,j} \{ \delta_{a,c,i,j} \}$$

where $i < j < \min\{i+N_{ac}, H\}$. Here, Δ_f, another upper bound on Δ, shows how much λ_{ai} can be decreased to prevent any challengers of asset a from entering the current MCKP solution in period i. If there is a replacement at i, we set $\Delta_f = +\infty$. Now, we can define Δ as follows:

$$\Delta = \min \{\Delta_p, \Delta_f\}$$

After decreasing λ_{ai} by $\Delta > 0$, $L(\lambda)$ will be strictly reduced by

$$\begin{cases} (m-1)\Delta & \text{if } i = H \text{ or } \Delta_f = +\infty \\ m\Delta & \text{otherwise} \end{cases}$$

On the other hand, empirical results indicate that increasing a multiplier to reduce the Lagrangian function is not computationally justifiable (see Karabakal 1991). Hence, we chose not to increase any multiplier in designing our MAM.

The MAM starts with finding a pair of indices (a, i) for which $\xi_{ai} > 0$, that is, finding a dualized asset sequencing constraint violated by retiring too many challengers of asset a in period i. Then, we determine the value of Δ by which we decrease λ_{ai}. If Δ turns out to be zero, indicative of a situation that the current MCKP solution in period i has an alternative optimum, decreasing λ_{ai} does not decrease $L(\lambda)$. If $\Delta > 0$, we update λ_{ai} by subtracting Δ from its current value. With a new λ, we solve $L(\lambda)$ again and compute a new ξ vector. The method stops either when all attempts to strictly decrease a multiplier fail or when all subgradient vector elements become zero. The latter situation implies feasibility, and supposing a branch-and-bound environment, it yields an optimal solution to the CRRP at the root node and an optimal completion at any other node.

A crucial point in the implementation of the MAM is an efficient computation of δ quantities in determining Δ. Next, we describe a method which is based on a post-optimality analysis of the objective coefficients over the current optimal MCKP solutions.

A Post-Optimality Analysis For The MCKP

Essentially, we seek to answer the following question after solving a MCKP optimally: if x_{kj} is not chosen in the current optimum, what is the minimum Δ such that $c_{kj} + \Delta$ would cause x_{kj} to be included in an alternative optimum assuming that all other objective coefficients remain unchanged?

Recall that the backward pass computes the length of the longest path from state (m, b) to (k, β) for each feasible state (k, β) in the DP network. These longest path lengths were given by $f_k(\beta)$. In particular, the longest path length from (m, b) to $(0, 0)$ was z. Similarly, we can make a forward pass to compute $g_k(\beta)$ defined as z minus the length of the longest path from state $(0, 0)$ to (k, β) for the same feasible state (k, β) encountered in the backward pass. Specifically,

$$g_1(a_{1j}) = z - c_{1j}$$

for $j = 1, \ldots, n_1$ and $a_{1j} \in M_1$

$$g_1(\beta) = \min\{\, g_1(\beta), g_1(\beta - 1)\,\}$$

for all $\beta \in M_1$ and $\beta \neq \min_j\{a_{1j}\}$

$$g_{k+1}(\beta) = \min_j\{g_k(\beta - a_{k+1,j}) - c_{k+1,j} \mid (\beta - a_{k+1,j}) \in M_k\}$$

for $k = 1, \ldots, m-1$ and all $\beta \in M_{k+1}$.

Proposition 1: *Suppose x_{kj} is not in the current optimum. Assuming that all other objective coefficients remain unchanged, the minimum Δ such that $c_{kj} + \Delta$ would cause x_{kj} to be included in the current optimum is given by Δ_{kj}, where*

$$\Delta_{mj} = g_{m-1}(b - a_{mj}) - c_{mj} \qquad \text{for } j = 1, \ldots, n_m \tag{8}$$

$$\Delta_{kj} = \min_{\beta \in M_k}\{g_{k-1}(\beta - a_{kj}) - f_k(\beta) - c_{kj} \mid \beta - a_{kj} \in M_{k-1}\} \tag{9}$$

for $k = m-1, m-2, \ldots, 2$ and $j = 1, \ldots, n_k$, and

$$\Delta_{1j} = z - f_1(a_{1j}) - c_{1j} \qquad \text{for } j = 1, \ldots, n_1 \tag{10}$$

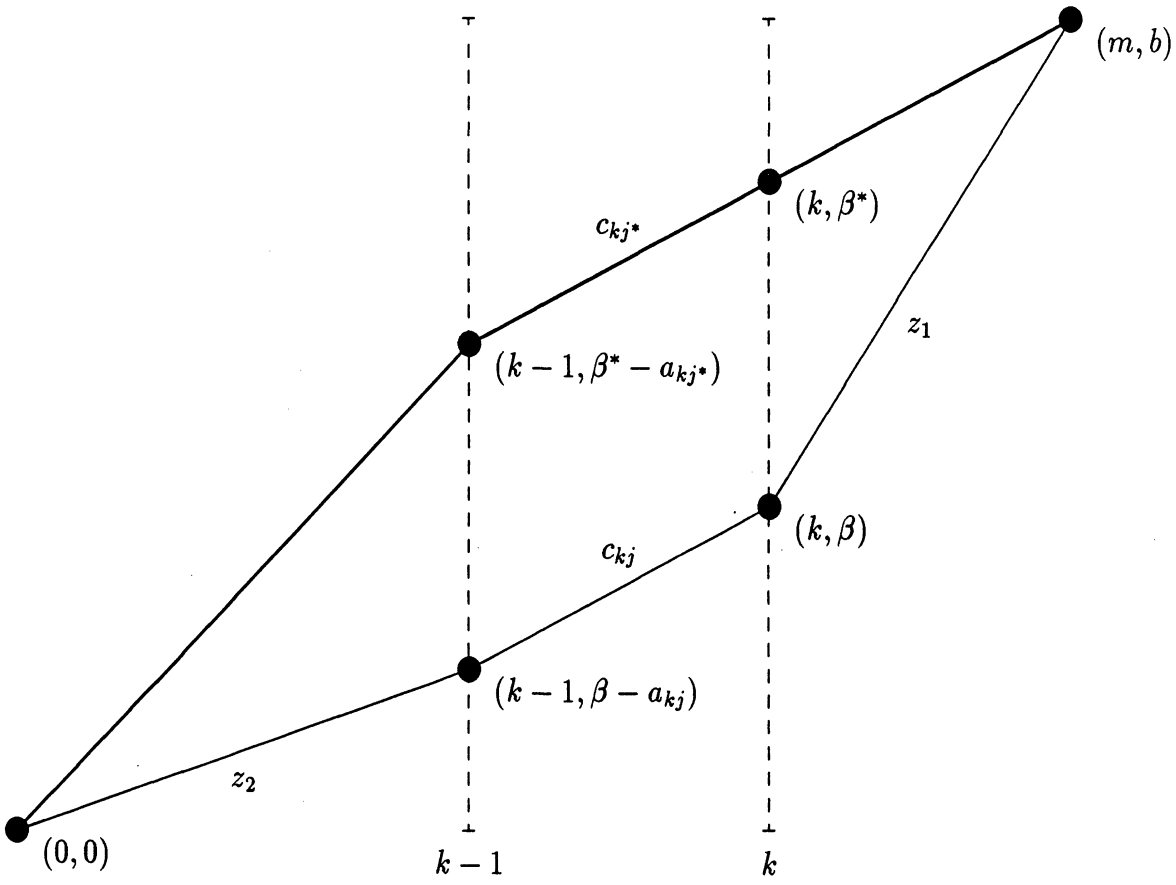

Figure 3: Illustration of the post-optimality analysis

Proof: Assume that x_{kj^*} was selected from the k-th multiple choice set. Consider another variable x_{kj}, $j \neq j^*$, from the same set, and let one of the arcs that represents x_{kj} in the DP network be from (k, β) to $(k-1, \beta - a_{kj})$ (see Figure 3). Define z_1 and z_2 as the longest path lengths from (m, b) to (k, β) and from $(k-1, \beta - a_{kj})$ to $(0,0)$, respectively. Hence, the length of the longest path from (m, b) to $(0,0)$ constrained to include the arc from (k, β) to $(k-1, \beta - a_{kj})$, denoted z^R, would be

$$z^R = z_1 + c_{kj} + z_2 \leq z$$

In order for x_{kj} to be selected from multiple choice set k in an alternative optimal solution, z^R should be at least as large as z. To assure this selection, c_{kj} must

be increased by at least $z - z^R$, where

$$\begin{aligned} z - z^R &= z - (z_1 + c_{kj} + z_2) \\ &= (z - z_2) - z_1 - c_{kj} \\ &= g_{k-1}(\beta - a_{kj}) - f_k(\beta) - c_{kj} \end{aligned}$$

Since there may be more than one arc corresponding to the variable x_{kj}, it is necessary to identify the minimum increment of c_{kj} over all these arcs.

$$\begin{aligned} \min_{\beta \in M_k} \{z - z^R\} &= \min_{\beta \in M_k} \{g_{k-1}(\beta - a_{kj}) - f_k(\beta) - c_{kj} \mid (\beta - a_{kj}) \in M_{k-1}\} \\ &= \Delta_{kj} \end{aligned}$$

as given by equation (9). Consider the boundary conditions. For $k = m$, set $z_1 = 0$ to obtain equation (8). For $k = 1$, set $z_2 = 0$ to obtain equation (10). In both cases, there is a one-to-one match between variables and arcs, therefore the use of the minimum operator over β is not required. ■

The corollary below follows directly from Proposition 1:

Corollary 1: *Suppose x_{kj^*} is selected from the k-th multiple choice set in the current optimum. Note that $\Delta_{kj^*} = 0$. The minimum Δ such that $c_{kj^*} - \Delta$ would cause x_{kj^*} to be removed from the current optimum (or, equivalently, the maximum Δ such that $c_{kj^*} - \Delta$ would keep x_{kj^*} in the current optimum) is given by $\min_{j \neq j^*}\{\Delta_{kj}\}$ assuming that all other objective coefficients remain unchanged. If $\Delta = 0$, there is an alternative optimum to the current solution.*

Remark: This procedure is similar to finding total floats in an activity network by the critical path method. The floats define Δ_{kj} values.

Improving The Lower Bound

Each computation of $L(\lambda)$ returns a primal solution, x, which, although feasible with respect to the capital rationing constraints, is usually infeasible with respect to the asset sequencing equations. Some of these infeasible solutions, however, can be made feasible easily. If the purchase of a particular challenger is feasible at the end of period i

to be used for p periods, it is also feasible to use it any p' periods where p' can range from one period to the challenger's maximum service life. This correction obviously does not change any expenditure pattern and hence the budget-feasibility is retained. There may be many possible ways of correcting a given infeasible replacement schedule by either decreasing or increasing the service lives of assets. If several corrections are possible, the one that results in the maximum NPV can be found by modifying slightly the DP recursion given earlier for solving $L(\mu)$.

Let c_{ai} be the challenger of asset a to be acquired in period i as prescribed by the current infeasible solution x. For each a

$$\text{NPV}(a,j) = \max_{i=j-1,\ldots,0} \{\, \bar{\pi}_{a,c_{ai},i,j} + \text{NPV}(a,i) \,\} \qquad \text{for } j = 1,\ldots,H$$

where $\bar{\pi}_{a,c_{ai},i,j} = -\infty$ if $x_{a,c_{ai},i,j}$ cannot be selected. If $\text{NPV}(a,H) = -\infty$ for any a, the attempt to generate a feasible solution fails. Computational tests indicate that this procedure often improves the current lower bound.

5 Heuristic Approaches

Obtaining a feasible solution to the CRRP is important because a) in branch and bound algorithms a good lower bound prunes many subtrees early in the tree search, b) subgradient algorithms used in solving Lagrangian duals require a lower bound to determine better step sizes, and c) just a "good" feasible solution may be the only hope for some big problems.

The algorithm uses the idea of a state expressed as an n-tuple $(t_1,\ldots,t_a,\ldots,t_n)$, where t_a is a "partial horizon" for asset a. A transition to state $(t_1,\ldots,t_a,\ldots,t_n)$ from a state $(t'_1,\ldots,t'_a,\ldots,t'_n)$ is possible only if it is feasible to purchase a challenger of asset a at t'_a to be used until $t_a > t'_a$ whenever $t'_a \neq H$ for all a. If $t'_a = H$, the next state, if any, must have $t_a = H$ too. The purpose is to find a path from the start state $(0,\ldots,0)$ to the goal state $(H,\ldots,H)$ passing through feasible states in between. Instead of enumerating all the states, however, we use a "heuristic rule" to accomplish the state transitions.

A systematic way of implementing these state transitions is the depth-first tree search technique which is devised to reach a goal state as quickly as possible (Nilsson 1971). Hence, the technique is useful to find a quick feasible, but not necessarily good solution to the CRRP. The general idea is as follows:

1. Put the start state $(0,\ldots,0)$ on a list called OPEN.

2. If OPEN is empty, exit with failure; otherwise continue.

3. Remove the *first* state from OPEN and put it on a list called CLOSED. Let this state be $i = (i_1,\ldots i_n)$.

4. Expand state i, i.e. use heuristic rules to define transitions from state i to successor states. If no transition is possible, go to step 2. Otherwise, if any of the successors is the goal state $(H,\ldots,H)$, exit with solution obtained by tracing back through the pointers. If neither of them is the goal state, put these successor states (in arbitrary order) at the *beginning* of OPEN and provide pointers back to i and go to step 2.

As an illustration, consider a three-asset fleet and 10-year planning horizon. Figure 4 shows a possible path followed by the algorithm to reach the goal state $(10,10,10)$. In this example, only two heuristic rules are used to accomplish the state transitions. The successors to the left are generated first. Note that all attempts to expand state (7,7,6) fail due to the budget constraints at these times so, the algorithm proceeds with expanding its sibling state (7,8,6).

Within the general framework of the depth-first search, many heuristic rules can be devised to expand a state i in step 4. We introduce five rules here. Given a state i and total expenditures from state $(0,\ldots,0)$ to state i, for each a, find a challenger and its period of usage such that:

1. the immediate contribution to the overall NPV is maximized,

2. the ratio of the immediate contribution to the overall NPV by the purchase cost is maximized,

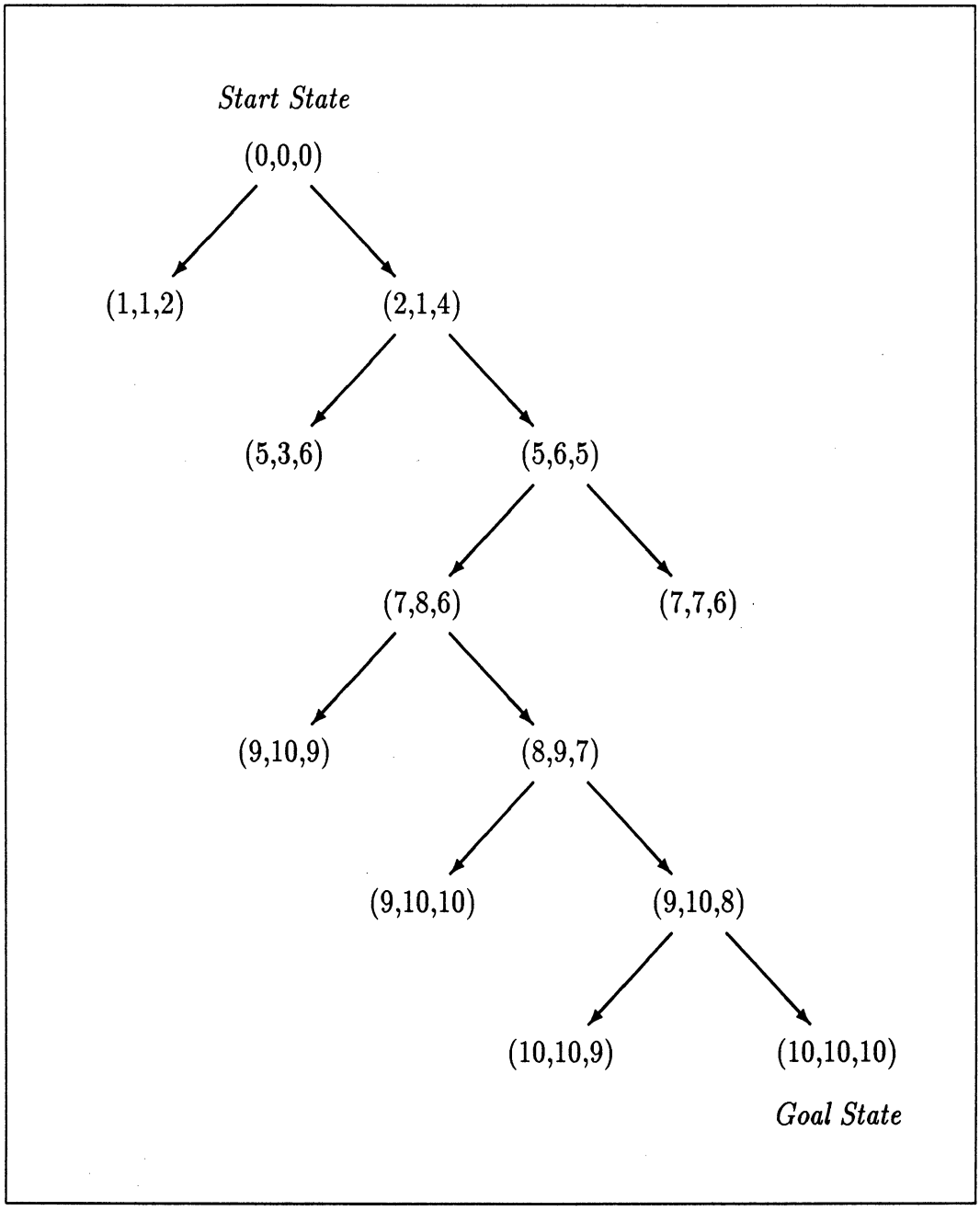

Figure 4: An example to illustrate the depth-first heuristic search algorithm

3. the ratio of the immediate contribution to the overall NPV by the purchase cost weighted by the remaining budget at the time of purchase is maximized,

4. the immediate expense is minimized,

5. the ratio of the immediate expense to the remaining budget at the time of purchase is minimized.

Although many other similar rules could be devised, the above five prove sufficient in locating a quick feasible solution on random test problems.

6 Variable Elimination

Before starting the branch-and-bound, there is usually a computational advantage to invest some time for reducing the problem size. We present two ways of eliminating variables that cannot appear in an optimal solution.

A Dominance Criterion

It is possible to eliminate some variables immediately after reading problem data using a knapsack-type dominance criterion (Sinha and Zoltners 1979). The rule is, for a fixed asset a and a fixed period of service, say from i to j, if a challenger c_1 costs more than challenger c_2 in period i, and if c_1's NPV contribution is less than that of c_2, then a decision to buy c_1 in period i and use it up to j is dominated. Symbolically

$$\text{IF } P_{a,c_1,i} \geq P_{a,c_2,i} \text{ AND } \pi_{a,c_1,i,j} \leq \pi_{a,c_2,i,j} \text{ THEN ELIMINATE } x_{a,c_1,i,j}$$

Variable Elimination By LP Relaxation

The second variable elimination method requires a dual feasible solution for the LP relaxation of the CRRP. Let $(\bar{P})$ denote the LP relaxation of the CRRP excluding the

redundant multiple choice constraints (5) and (6). Let $(\bar{D})$ be its dual.

$$(\bar{D}): \qquad \text{minimize} \quad \sum_a (\lambda_{a0} - \lambda_{aH}) + \sum_i B_i \mu_i$$

subject to

$$\lambda_{ai} - \lambda_{aj} + P_{aci}\,\mu_i \geq \pi_{acij}$$

$$\mu_i \geq 0$$

The constraints of $(\bar{D})$ can also be written as

$$\lambda_{ai} - \lambda_{aj} + P_{aci}\,\mu_i + \bar{\pi}_{acij} = \pi_{acij}$$

$$\mu_i \geq 0,\ \bar{\pi}_{acij} \leq 0$$

or

$$\bar{\pi}_{acij} = \pi_{acij} - \lambda_{ai} + \lambda_{aj} - P_{aci}\,\mu_i \leq 0$$

$$\mu_i \geq 0$$

Note that $\bar{\pi}_{acij}$ values correspond to the reduced profits of $(\bar{P})$ and the dual feasibility condition $\bar{\pi}_{acij} \leq 0$ is an optimality condition for $(\bar{P})$.

We consider a variable elimination method proposed by Sweeney and Murphy (1981). Let UB be an optimum objective for $(\bar{P})$. Suppose we are given a feasible solution and let LB be its objective value. Then, a variable could be eliminated if its absolute value reduced profit exceeds $UB - LB$, since any solution with this variable equal to one would have an objective value less than LB.

The preceding variable elimination rule can be made stronger for the CRRP by following a suggestion by Noon and Bean (1988). Let the *shortest reduced profit path* be defined as the shortest path from one node to another over the reduced arc profits. Let $SP(a,i,j)$ be the shortest reduced profit path from node i to node j in asset a's network.

Proposition 2: *If*

$$|\,\bar{\pi}_{acij} + SP(a,0,i) + SP(a,j,H)\,| > UB - LB \tag{11}$$

then no solution with $x_{acij} = 1$ can be optimal and thus, x_{acij} can be eliminated from the problem.

Proof: Whenever $x_{acij} = 1$, its corresponding arc must be on a path from node 0 to node H. On this path, the length from node 0 to node i is at least as long as $SP(a,0,i)$. Similarly, the length from node j to node H is at least as long as $SP(a,i,H)$. Hence, $SP(a,0,i)$ and $SP(a,i,H)$ can be added to $\bar{\pi}_{acij}$ to strengthen the elimination rule. ■

Determining reduced profits by any direct solution of $(\bar{P})$ or $(\bar{D})$ is costly. Instead, we take advantage of the integrality property that the Lagrangian relaxation (LR_μ) possesses. The integrality property assures that a) $L(\mu)$ is minimized by setting $\mu = \mu^*$, where μ^* is optimal for $(\bar{D})$, and b) $L(\mu^*)$ is equal to the optimum objective of the LP relaxation (Geoffrion 1974). We approximate the solution using the iterative subgradient algorithm. Suppose the solution of (LD_μ) yields μ^* and x^*, where x^*_{acij} is marked as "basic" for DP tree arcs, x^*_{acij} is "nonbasic" otherwise. Set $UB \leftarrow L(\mu^*)$. Consider the dual constraints

$$\lambda_{ai} - \lambda_{aj} + P_{aci}\,\mu^*_i \geq \pi_{acij}$$

From the complementary slackness condition, dual constraints corresponding to the "basic" primal variables are equalities. Hence, set $\lambda^*_{a0} = 0$ for all a and solve the equations

$$\lambda^*_{aj} = \lambda^*_{ai} - \pi_{acij} + P_{aci}\,\mu^*_i \qquad \text{for } j = 1,\ldots,H$$

corresponding to the "basic" x^*_{acij}. Compute reduced profits

$$\bar{\pi}_{acij} = \pi_{acij} - \lambda^*_{ai} + \lambda^*_{aj} - P_{aci}\,\mu^*_i \leq 0$$

Eliminate a variable x_{acij} satisfying condition (11). However, since x^* above is computed by determining a DP tree rooted at node zero of each asset's network, we note that $SP(a,0,i) = 0$ for all a and $i > 0$. Therefore, condition (11) is equivalent to

$$|\,\bar{\pi}_{acij} + SP(a,j,H)\,| > UB - LB \tag{12}$$

If at least one variable is eliminated by (12), we apply heuristic algorithms to the reduced problem in an attempt to improve the LB. If the LB is strictly increased, condition (12) is checked again to eliminate more variables and we repeat; otherwise we stop.

7 Implicit Enumeration

After variable elimination, we solve (LD_λ) with initial multipliers set to λ^*. If an upper bound obtained by either Lagrangian relaxation turns out to be feasible with respect to the relaxed constraints, it is optimum to the CRRP—we stop. Otherwise, we have a "duality gap" and we use a branch-and-bound implicit enumeration to close the gap.

Nonoptimality of a solution obtained by solving (LD_λ) indicates that one or more of the asset sequencing constraints are violated. Every infeasible solution must have an asset a for which there are too many challengers retired in a certain period, say $i > 0$. We branch from such solutions by imposing further restrictions in an attempt to reduce infeasibility. We have not found the selection of the pair of indices (a, i) to be critical; we simply select them such that the current subgradient element ξ_{ai} is maximum.

The set of variables that represent the challengers of asset a retired in period i possesses a "multiple choice" structure because any feasible solution may include at most one variable from this set. This structure can be exploited by incorporating a branching strategy similar to the approach found in Bean (1984) for multiple choice variable sets. Therefore, if a variable is fixed at one, the other variables in its set are automatically fixed at zero.

Let $h_1, \ldots, h_m$ be the periods in which the challengers retired in i were purchased. There may be three possible violations of asset sequencing constraints. We discuss how to branch from each case below.

1. Too many challengers ($m \geq 2$) are retired in period $i = H$. Since we need exactly one challenger to be retired in H, we create m branches, each fixing a single decision variable to one. Another possibility is to ignore all of these decisions. We do this in the $(m + 1)$-st branch by eliminating all current retirements in H

from the subproblem.

$$\begin{array}{lll} \texttt{Branch } 1: & \text{Set} & x_{ac_1h_1H} = 1 \\ \vdots & & \vdots \\ \texttt{Branch } m: & \text{Set} & x_{ac_mh_mH} = 1 \\ \texttt{Branch } m+1: & \text{Set} & x_{ac_1h_1H} = \cdots = x_{ac_mh_mH} = 0 \end{array}$$

2. Exactly one challenger is retired in period $i \neq H$, but no challenger is purchased in i. There are two ways of correcting this infeasibility. We can either eliminate the current retirement decision from the subproblem or we can fix this retirement decision and, at the same time, make sure that there will be exactly one purchase in i.

$$\begin{array}{lll} \texttt{Branch } 1: & \text{Set} & x_{ac_1h_1i} = 0 \\ \texttt{Branch } 2: & \text{Set} & x_{ac_1h_1i} = 1 \quad \text{and} \quad \sum_c \sum_j x_{acij} = 1 \end{array}$$

3. Too many challengers ($m \geq 2$) are retired in period $i \neq H$. There may or may not be a challenger purchased in i. To correct it, we use a strategy combination of those given for case 1 and case 2.

$$\begin{array}{lll} \texttt{Branch } 1: & \text{Set} & x_{ac_1h_1i} = 1 \quad \text{and} \quad \sum_c \sum_j x_{acij} = 1 \\ \vdots & & \vdots \\ \texttt{Branch } m: & \text{Set} & x_{ac_mh_mi} = 1 \quad \text{and} \quad \sum_c \sum_j x_{acij} = 1 \\ \texttt{Branch } m+1: & \text{Set} & x_{ac_1h_1i} = \cdots = x_{ac_mh_mi} = 0 \end{array}$$

The implicit enumeration algorithm begins by choosing a pair of indices (a, i), and branching. Each resulting subproblem is upper-bounded and checked for fathoming. It is fathomed if it is either infeasible or an optimal completion, in which case it is compared with the current incumbent. For upper-bounding a subproblem, (LD_λ) is first solved by the MAM. If it is not fathomed, final multipliers of the MAM are used as initial multipliers of the subgradient algorithm. The upper bound returned from the subgradient algorithm with this initialization strategy is tighter than the bound that would be returned with arbitrary initialization. Subproblems that cannot be fathomed

are put on a list. The subproblem with the largest upper bound is selected to branch next. The algorithm continues until the subproblem list is empty or the largest upper bound of the list is less than the incumbent. At this point, the optimal solution is the current incumbent.

Computational Experiments

The optimal algorithm was coded in Pascal and tested on a series of randomly generated problems. Test problems were generated for group sizes $n = 2, 4, 6, 8$, and 10 with time horizons $H = 4, 6, 8$, and 10. Following uniform distributions were used to generate challenger parameters.

Number of challengers of an asset	$U(1,3)$
Life of a challenger	$U(2,H)$
Remaining life of the defender	$U(1,3)$
Purchase cost of a challenger	$U(10,30)$
NPV(a,c,p)	$U(50,60) + (p-1)\,U(5,35)$

Budgets were set to half of the total expenditures if the cheapest challenger of every asset were purchased in each time period. For every n-H combination, we solved a set of five problems. The results are shown in Table 1. Column (3) shows the median problem size for each set in terms of the number of constraints times the number of binary variables when the CRRP is formulated as an integer program. Columns (4), (5), and (6) display the median, minimum, and maximum CPU times, respectively, to solve each set of test problems optimally. The times reported are in seconds on an IBM 3090-600E at The University of Michigan, running the MTS operating system. An asterisk next to table element indicates that the corresponding problem could not be solved optimally in 2,000 CPU seconds. The next three columns give the performance of the upper bounding procedures used at the root node expressed as a median percent deviation from the optimal objective (for the problems that could not be solved optimally, percent deviation from the value of the best incumbent at the termination of execution). The upper bound of column (7) is obtained by disregarding all capital rationing constraints and hence, column (7) gives a measure of tightness of

Table 1: Results of the computational experiments

			CPU Times (seconds)			Upper Bounding (%)			Lower	Variable
n	H	Size	Median	Min	Max	$LD_{\mu=0}$	LD_{μ}	LD_{λ}	Bounding (%)	Reduction (%)
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)
	4	14 x 36	0.03	0.02	0.04	33.31	0.00	—	2.66	25
2	6	20 x 51	0.24	0.04	0.32	27.71	4.05	0.00	9.77	22
	8	26 x 93	0.53	0.05	2.29	19.40	2.41	0.00	1.87	35
	10	32 x 191	0.11	0.07	1.16	12.95	0.00	—	0.00	24
	4	24 x 86	1.67	0.04	3.71	18.40	4.07	0.04	0.00	41
4	6	34 x 133	7.79	0.11	9.46	18.37	4.13	0.47	12.10	24
	8	44 x 235	10.81	0.08	19.73	19.08	3.29	0.30	6.94	29
	10	54 x 362	15.50	0.15	17.79	17.37	3.70	0.32	14.04	27
	4	34 x 112	1.61	0.07	8.70	15.99	2.68	0.82	8.73	33
6	6	48 x 209	19.12	0.10	71.92	18.76	2.99	0.23	18.96	26
	8	62 x 309	55.65	41.92	114.84	15.50	3.52	0.42	10.64	24
	10	76 x 542	304.79	0.23	1386.98	10.82	2.71	0.20	16.96	27
	4	44 x 162	0.13	0.10	11.61	27.15	0.00	—	2.85	33
8	6	62 x 303	89.73	0.16	141.66	15.35	1.07	0.17	10.63	26
	8	80 x 434	111.50	0.33	211.62	17.75	0.72	0.16	4.04	22
	10	98 x 724	874.35	183.49	2000.00*	12.99	1.78	0.27*	8.88*	24
	4	54 x 166	32.20	0.20	79.06	14.20	0.77	0.01	14.53	27
10	6	76 x 385	858.13	0.46	1015.27	16.59	0.89	0.46	10.71	33
	8	98 x 565	2000.00*	0.53	2000.00*	34.61*	21.29*	20.03*	20.03*	27
	10	120 x 986	2000.00*	38.79	2000.00*	25.56*	16.97*	11.78*	11.78*	20

* problem could not be solved in 2,000 CPU seconds

the capital rationing constraints. Column (8) shows the performance of the best upper bound obtained by solving (LD_μ), and column (9) shows that obtained by solving (LD_λ). For all problems, (LD_λ) provided strictly tighter bounds than (LD_μ). However, as the problem size increases, the difference between these two bounds becomes less significant. Similarly, column (10) shows the performance of the heuristic approach for lower bounding expressed as a median percent deviation from the optimal objective (for the problems that could not be solved optimally, percent deviation from the value of the best upper bound at the termination of execution). Since the purpose is to find a quick feasible solution, these lower bounds are not as close to the optimum as the upper bounds provided by Lagrangian relaxation. Finally, the last column indicates the median percent variable elimination at the root node to reduce the problem size. Combined performance of the two variable elimination procedures ranged from 20% to 41%.

It should be noted that budgets used in the test problems are relatively small. The DP approach of solving multiple-choice knapsack subproblems does show an increase in computation time with larger budgets. One way to deal with this problem is to scale P_{aci} and B_i values. As a result, there is some loss in precision since they must take integral values. However, in many applications, future budgets and purchase costs are not known to a high precision and/or the budget constraints have some flexibility.

8 Summary and Extensions

A parallel, finite horizon, and deterministic replacement economy problem was studied to determine the replacement schedules for individual assets such that a) the NPV of the cash flows resulting from the schedules is optimized, and b) budget constraints imposed for each time period within the planning horizon are satisfied. It was motivated by the experience gained from a vehicle fleet replacement study. The problem is formulated as a zero-one integer program and an optimum algorithm using Lagrangian relaxation methodology is developed. A new multiplier adjustment method is developed to solve one of the Lagrangian duals. The implementation of the algorithm solved moderately-

sized problems in reasonable times.

This research concentrated on the type of replacement problems in which the cash flows of a future asset do not depend on the the service condition at future asset's time of installation, nor the previous replacement decisions. Many replacement problems satisfy these requirements, including vehicle fleet and machine replacements where salvage and trade-in values are insignificant. In other problems, those dependencies may need to be introduced. If the trade-in values are significant, a future asset's purchase cost may depend on the condition of the asset in use at the contemplated replacement time. For example, in street pavement maintenance problem, the cost of a major maintenance action (corresponding to a challenger) usually depends not only on the pavement segment's current condition, but also on the frequency of previous rehabilitations. One way to generalize the current model to cover condition/history-dependent cash flows is to add more subscripts to the variable and parameter definitions. Although such an approach increases the number of variables and constraints, the structure that allowed us to develop effective Lagrangian techniques remains undisturbed. These extensions will be addressed in a separate paper.

Appendix

A Numerical Example To Illustrate The MCKP Algorithm

Consider the following MCKP with $m = 4$, $n_1 = 3$, $n_2 = n_3 = n_4 = 2$, $b = 6$, and

$$(c_{kj}) = \begin{pmatrix} 1 & 6 & 4 \\ 2 & 4 & \\ 3 & 2 & \\ 1 & 5 & \end{pmatrix} \qquad (a_{kj}) = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & \\ 1 & 2 & \\ 1 & 3 & \end{pmatrix}$$

The DP network created by applying the algorithm is shown in Figure 5. Backward and forward pass computations are shown below. Asterisks indicate DP nodes along the optimal path. An optimum solution is given by $x_{12} = x_{21} = x_{31} = x_{41} = 1$, and all others are zero. The optimum objective is 12.

State (k, β)	Backward Pass $f_k(\beta)$	Forward Pass $g_k(\beta)$
(4, 6) *	0	0
(3, 5) *	1	1
(3, 3)	5	6
(2, 4) *	4	4
(2, 3)	3	4
(2, 2)	8	9
(1, 3) *	6	6
(1, 2) *	6	6
(1, 1)	10	11
(0, 0) *	12	12

Calculations for the post-optimality analysis yield:

$$\begin{aligned} \Delta_{11} &= \min\{1\} = 1 & \Delta_{31} &= \min\{0, 1\} = 0 \\ \Delta_{12} &= \min\{0\} = 0 & \Delta_{32} &= \min\{1\} = 1 \\ \Delta_{13} &= \min\{2\} = 2 & \Delta_{41} &= \min\{0\} = 0 \\ \Delta_{21} &= \min\{0, 1, 1\} = 0 & \Delta_{42} &= \min\{1\} = 1 \\ \Delta_{22} &= \min\{3\} = 3 \end{aligned}$$

Variables selected in the optimum solution have zero Δ values. Proposition 1 tells that to select, for example, x_{22} in an alternative optimum solution, it is necessary to increase c_{22} by at least 3 units.

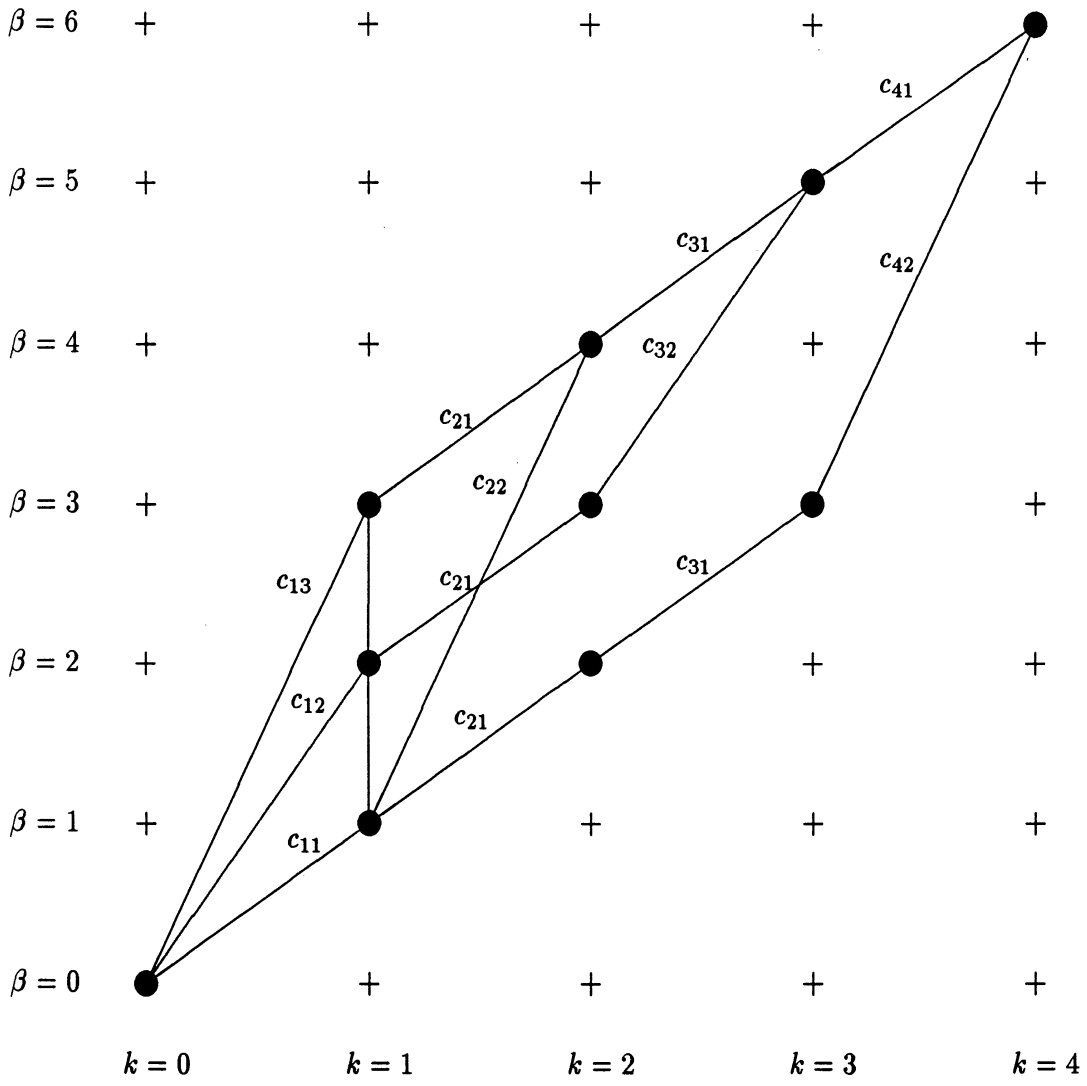

Figure 5: DP network for the example problem

References

BEAN, J. C., "A Lagrangian Algorithm for the Multiple Choice Integer Program," *Operations Research*, 32 (1984), 1185-1193.

———, J. R. LOHMANN AND R. L. SMITH, "A Dynamic Infinite Horizon Replacement Economy Decision Model," *The Engineering Economist*, 30 (1985), 99-120.

———, ——— AND ———, "Equipment Replacement Under Technological Change," Technical Report 90-5, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, 1990.

——— AND K. A. SYVERSON, "A Hybrid Algorithm for the Multiple Choice Knapsack," Technical Report 90-33, Department of Industrial and Operations Engineering, The

University of Michigan, Ann Arbor, 1990.

BREALEY, R. A. AND S. C. MYERS, *Principles of Corporate Finance*, 3rd Edition, McGraw-Hill, 1988.

CHAN, T. J., "A Multiplier-Adjustment-Based Branch-and-Bound Algorithm for Solving the Set Partitioning Problem," Ph.D. Dissertation, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, 1987.

CHAND, S. AND S. SETHI, "Planning Horizon Procedures for Machine Replacement Models with Several Possible Replacement Alternatives," *Naval Research Logistics Quarterly*, 29 (1982), 483-493.

DENARDO, E. V., *Dynamic Programming: Models and Applications*, Prentice Hall Inc., 1982.

DYER, M., N. KAYAL AND J. WALKER, "A Branch and Bound Algorithm For Solving the Multiple-Choice Knapsack Problem," *Journal of Computational and Applied Mathematics*, 11 (1984), 231-249.

ERLENKOTTER, D., "A Dual-Based Procedure for Uncapacitated Facility Location," *Operations Research*, 26 (1978), 992-1009.

ETCHEBERRY, J., "The Set Covering Problem: A New Implicit Enumeration Algorithm," *Operations Research*, 25 (1977), 760-772.

FISHER, M. L., "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science*, 27 (1981), 1-18.

———, "An Applications Oriented Guide To Lagrangian Relaxation," *Interfaces*, 15 (1985), 10-21.

———, R. JAIKUMAR AND L. N. VAN WASSENHOVE, "A Multiplier Adjustment Method for the Generalized Assignment Problem," *Management Science*, 32 (1986), 1095-1103.

GEOFFRION, A. M., "Lagrangian Relaxation for Integer Programming," *Mathematical Programming Study*, 2 (1974), 82-114.

GILMORE, P. AND R. GOMORY, "The Theory and Computation of Knapsack Functions," *Operations Research*, 14 (1966), 1045-1074.

GOFFIN, J. L., "On Convergence Rates of Subgradient Optimization Methods," *Mathematical Programming*, 13 (1977), 329-347.

GUIGNARD, M. AND M. B. ROSENWEIN, "An Improved Dual Based Algorithm for the Generalized Assignment Problem," *Operations Research*, 37 (1989), 658-663.

GURNANI, C., "Capital Budgeting: Theory and Practice," *The Engineering Economist*, 30 (1984), 19-46.

HELD, M. AND R. M. KARP, "The Traveling-Salesman Problem and Minimum Spanning Trees: Part II," *Mathematical Programming*, 1 (1971), 6-25.

———, P. WOLFE AND H. P. CROWDER, "Validation of Subgradient Optimization," *Mathematical Programming*, 6 (1974), 62-88.

HOPP, W. J. AND S. K. NAIR, "Timing Replacement Decisions Under Discontinuous Technological Change," *Naval Research Logistics*, 38 (1991), 203-220.

JONES, P. C., J. L. ZYDIAK AND W. J. HOPP, "Parallel Machine Replacement," *Naval Research Logistics*, 38 (1991), 351-365.

KARABAKAL, N., "The Capital Rationing Replacement Problem," Ph.D. Dissertation, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, 1991.

LOHMANN, J. R., "A Stochastic Replacement Economic Decision Model," *IIE Transactions*, 18 (1986), 182-194.

LORIE, J. AND L. J. SAVAGE, "Three Problems in Capital Rationing," *Journal of Business*, XXVIII (1955), 229-239.

MAZZOLA, J. B. AND A. W. NEEBE, "Resource-Constrained Assignment Scheduling," *Operations Research*, 34 (1986), pp. 560-572.

NILSSON, N. J., *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, 1971.

NOON, C. E. AND J. C. BEAN, "A Lagrangian Based Approach to the Asymmetric Generalized Traveling Salesman Problem," *Operations Research*, 39 (1991), 623-632.

OAKFORD R. V., *Capital Budgeting: A Quantitative Evaluation of Investment Alternatives*, The Ronald Press Co., New York, 1970.

———, J. R. LOHMANN AND A. SALAZAR, "A Dynamic Replacement Economy Decision Model," *IIE Transactions*, 16 (1984), 65-72.

SETHI, S. AND S. CHAND, "Planning Horizon Procedures for Machine Replacement Models," *Management Science*, 25 (1979), 140-151.

SHAPIRO, J. F., "A Survey of Lagrangean Techniques for Discrete Optimization," *Annals of Discrete Mathematics*, 5 (1979), 113-138.

SINHA, P. AND A. A. ZOLTNERS, "The Multiple-Choice Knapsack Problem," *Operations Research*, 27 (1979), 503-515.

SWEENEY, D. AND R. MURPHY, "Branch-and-Bound Methods for Multi-Item Scheduling," *Operations Research*, 29 (1981), 853-864.

TERBORGH, G., *Dynamic Equipment Policy*, Machinery and Allied Products Institute, Washington D.C., 1949.

VANDER VEEN, D. J., "Parallel Replacement Under Nonstationary Deterministic Demand," Ph.D. Dissertation, Department of Industrial and Operations Engineering, The University of Michigan, Ann Arbor, 1985.

WAGNER, H. M., *Principles of Operations Research*, Prentice Hall Inc., 1975.

WEINGARTNER, H. M., *Mathematical Programming and the Analysis of Capital Budgeting Problems*, Prentice Hall, 1963.

———, "Criteria for Programming Investment Project Selection," *Journal of Industrial Economics*, 15 (1966), 65-76.

UNIVERSITY OF MICHIGAN
3 9015 04733 8481