Nonhomogeneous Markov Decision Processes Linked by Side Constraints

Christopher L. Morse
James C. Bean
Department of Industrial & Operations Engineering
University of Michigan Ann Arbor, MI 48109

Nejat Karabakal
Production Modeling Corporation
Three Parklane Boulevard, Suite 910 West, Dearborn, MI 48126

Technical Report 97-04
April 1997

Nonhomogeneous Markov Decision Processes Linked by Side Constraints

CHRISTOPHER L. MORSE

JAMES C. BEAN

Department of Industrial and Operations Engineering

University of Michigan, Ann Arbor, Michigan 48109

NEJAT KARABAKAL

Production Modeling Corporation

Three Parklane Boulevard, Suite 910 West, Dearborn, Michigan 48126

April 28, 1997

Abstract

We present a mixed-integer nonlinear formulation of nonhomogeneous Markov decision processes linked by side constraints. A nonlinear penalty function is used to relax the side constraints. The concepts of pressure and volume are introduced in a variant of simulated annealing which is used to heuristically solve the relaxed problem. A multiplier adjustment method is presented to find bounds. Computational results are reported.

Introduction

A Markov decision process is a staged, stochastic optimization model that appropriately models many real problems such as equipment replacement. During each stage of the problem, the system is in one of a finite number of states. These states describe some condition of the system. For each stage and state of the problem, one of a finite number of actions may be chosen for the system. The action induces a reward. The system then proceeds to the next stage where the state is known with a probability dependent upon the stage, current action, and the current state. We will consider finite horizon problems where the horizon is the number of stages in consideration.

In many problems, the data may not be static through the stages of the problem. In this case the model is a nonhomogeneous Markov decision process (NMDP). Research in NMDPs has primarily focused upon solving infinite horizon problems. The literature includes Park, Bean, and Smith [19], Bean, Hopp, and Duenyas [4], Bean, Smith, and Laserre [5], and White [21].

Finite horizon NMDPs are solved by dynamic programs (see Derman [9] and Denardo [8]) that determine an optimal policy. For equipment replacement problems, the optimal policy would be an optimal replacement schedule for the particular asset in question.

Often in equipment replacement problems, a portfolio of assets must be analyzed. In this case a replacement action may have an associated capital expenditure in addition to the reward value. These capital expenditures may be limited in each stage (or period) by budget constraints. To formulate this problem, each asset's replacement is modeled as a NMDP and the processes for all the assets are linked during each period by side constraints representing the budgets.

The equipment replacement problem above motivates the study of the general problem of non-homogeneous Markov decision processes with side constraints. A literature search for this topic revealed no published papers regarding this topic. Related research includes Derman [10], which considers a single constrained homogeneous Markov decision process. Past research in groups of Markov decision processes focuses upon large scale homogeneous Markov decision processes. This includes White and Schlussel [22] and Varaiya [20].

Because this type of problem has not been addressed in the literature, we will introduce a

mixed-integer disjunctive programming formulation. As a method of solution, a simulated annealing technique expanded to include both temperature and pressure will be presented. This algorithm may have further applications in both artificial intelligence and operations research for other problems with constraints which cannot be incorporated into variable definitions. Because the solution from simulated annealing is not necessarily optimal, a bound is presented based on a multiplier adjustment method.

The mixed integer programming formulation and a relaxation using a nonlinear penalty function are in Section 1. Section 2 presents a simulated annealing approach to solve the relaxed problem. Section 3 develops a multiplier adjustment method to find a tight upper bound. Computational experiments are reported in Section 4. Section 5 presents conclusions.

1 Problem Formulation

To facilitate the formulation of nonhomogeneous Markov decision processes with side constraints, we present a dynamic program for solving a single unconstrained problem (see Derman [9] and Denardo [8]). Let p index the process being modeled (initially, we only consider a single process). Let t index the stage and i index the state for process p. Define a *scenario* to be the states of all the processes over the time horizon for a given outcome.

Define:

H = Number of stages.

$v(p,t,i)$ = Maximum expected return for process p of actions in stages t through H if process p is in state i in stage t.

$\pi(p,a,t,i)$ = Resource consumption if action a is chosen during stage t for process p in state i.

$P(p,a,t,i,j)$ = Probability that if action a of process p is chosen in stage t at state i then the result will be state j in stage $t+1$.

$R(p,a,t,i)$ = Reward if action a is chosen during stage t for process p in state i.

$S(p)$ = Set of possible states for process p.

$A(p)$ = Set of possible actions for process p.

α = Discount factor per stage.

$Z(p,i)$ = Salvage value of process p in state i at the end of stage H.

$q(p,1,i)$ = Probability process p is in state i at the beginning of stage 1.

$B(t)$ = Resource limit in stage t.

$s^k_{p,t}$ = The state of process p during period t in scenario k.

K = Number of scenarios.

$N(p,a,t,i)$ = Expected return if action a is chosen for p,t,i.

$= R(p,a,t,i) + \alpha \sum_j P(p,a,t,i,j) v(p,t+1,j)$

The following dynamic program solves for the maximum discounted expected return for process p:

$$v(p,t,i) = \max_a R(p,a,t,i) + \alpha \sum_{j \in S(p)} P(p,a,t,i,j) v(p,t+1,j) \quad (1)$$

$$for\ all\ t, i \in S(p)$$

$$v(p,H+1,i) = Z(p,i), for\ all\ i \in S(p) \quad (2)$$

For a group of processes, each problem can be solved separately using the above dynamic program. However, the resulting actions may violate the side constraints. As a step toward the inclusion of side constraints in the model formulation, a linear program is given below as (P1) (see Denardo [8]). This linear program solves the above dynamic program for process p.

$$\min \sum_{i \in S(p)} q(p,1,i)v(p,1,i) \tag{3}$$

(P1) subject to

$$v(p,t,i) \geq R(p,a,t,i) + \alpha \textstyle\sum_{j \in S(p)} P(p,a,t,i,j)v(p,t+1,j) \tag{4}$$

$$for\ all\ a,i \in S(p); m = 1,\ldots,H-1$$

$$v(p,H,i) \geq R(p,a,H,i) + \alpha \textstyle\sum_{j \in S(p)} P(p,a,H,i,j)Z(p,j) \tag{5}$$

$$for\ all\ a,i \in S(p)$$

The optimal values for $v(p,t,i)$ in (P1) are equal to the maximum discounted expected returns of the set of action choices. The lack of variables explicitly representing these decision choices in the linear program (P1) creates a difficulty in determining resource consumption. The selection of decisions can be represented by the binary variable $y(p,a,t,i)$, which is 1 if action a for process p is chosen in period t at state i, 0 otherwise. With the addition of side constraints, the optimal value for (P1) may be infeasible. The value for $v(p,t,i)$ must take on a value equal to a discounted expected return corresponding to a decision in the set of possible decisions. Thus, there is a set of equality constraints for each $v(p,t,i)$ where each constraint represents the value of a decision choice where only one of these constraints needs to hold for each set. The constraints in each set are *disjunctive.* Disjunctive constraints are connected by the logical operator *or* as opposed to conjunctive constraints which are connected by *and.* Disjunctive constraints are naturally nonlinear. However, it is possible to construct equivalent linear constraints. A linear mixed integer formulation (P2) using the linear equivalent of disjunctive constraints is given below with side constraints (12).

$$\max \sum_{p} \sum_{i \in S(p)} q(p,1,i)v(p,1,i) \tag{6}$$

(P2) subject to

$$v(p,t,i) \geq R(p,a,t,i) + \alpha \textstyle\sum_{j \in S(p)} P(p,a,t,i,j)v(p,t+1,j) - M[1 - y(p,a,t,i)] \tag{7}$$

$$for\ all\ p,a,i \in S(p); m = 1,\ldots,H-1$$

$$v(p,t,i) \leq R(p,a,t,i) + \alpha \textstyle\sum_{j \in S(p)} P(p,a,t,i,j)v(p,t+1,j) + M[1 - y(p,a,t,i)] \tag{8}$$

$$for\ all\ p,a,i \in S(p); m = 1,\ldots,H-1$$

$$v(p,H,i) \geq R(p,a,H,i) + \alpha \textstyle\sum_{j \in S(p)} P(p,a,H,i,j)Z(p,j) - M[1 - y(p,a,H,i)] \tag{9}$$

$$for\ all\ p,a,i \in S(p);$$

$$v(p,H,i) \leq R(p,a,H,i) + \alpha \textstyle\sum_{j \in S(p)} P(p,a,H,i,j)Z(p,j) + M[1 - y(p,a,H,i)] \tag{10}$$

$$for\ all\ p,a,i \in S(p);$$

$$\textstyle\sum_{a} y(p,a,t,i) = 1 \tag{11}$$

$$for\ all\ p,t,i \in S(p);$$

$$\textstyle\sum_{p} \sum_{c} \pi(p,a,t,s_p^k) y(p,a,t,s_p^k) \leq B(t) \tag{12}$$

$$for\ all\ t,k$$

$$y(p,a,t,i) \in \{0,1\}$$

where M is a large positive number.

Problem (P2) is based upon a formulation for nonhomogeneous Markov decision processes suggested in Bean, Hopp, and Duenyas [4]. Constraints (7)-(10) are linear equivalents of disjunctive constraints. Previous approaches to disjunctive programs include cutting planes in Balas [1, 2] and Jeroslow [15]. Also, a Bender's decomposition approach was developed in Bean [3] and Bean, Hopp, and Duenyas [4]. However, the combinatorially large number of disjunctive constraints and side constraints impedes a direct solution method for (P2).

The above formulation (P2) of the side constraints (12) is considered a *fat solution* (see Kall and Wallace [16]). Given an action vector y, the probability of a particular scenario occurring in stage t may be very small (or even zero). If this scenario violates the side constraint for stage t the solution would be infeasible. Therefore by including constraints for every possible scenario, good

solutions may be eliminated even if the probability of violating a constraint is very small. This type of constraint heavily effects the objective function. For this reason we now present an alternative formulation of the side constraints using expected values.

Define $q(p,t,i)$ to be the probability of process p in stage t being in state i given the values of all action variables, $y(p,a,t,i)$. We can use the following recursive equation to determine these probabilities:

$$q(p,t,i) = \sum_{j \in S(p)} \sum_{c} q(p,t-1,j) y(p,a,t-1,j) P(p,a,t-1,j,i). \tag{13}$$

Define Y to be the matrix of decisions $y(p,a,t,i)$. Define $E(t,Y)$ to be the expected value of expenditures during stage t. Then the new side constraint for the model is

$$\begin{aligned} E(t,Y) = \textstyle\sum_p \sum_{i \in S(p)} \sum_c q(p,t,i)\pi(p,a,t,i)y(p,a,t,i) \quad & for\ all\ t \\ B(t) - E(t,Y) \geq 0 \quad & for\ all\ t \end{aligned} \tag{14}$$

By substituting constraint (14) for constraint (12) we now have the expected value version (E1) of the original model (P2). Let V(E1) be the value of the objection function of (E1) for the optimal solution. The major drawback in using side constraints based upon the probability equations (13) is the loss of linearity. However, the combinatorially large number of side constraints has been reduced to the expected value side constraints (14).

1.1 Nonlinear Penalty Function

The number of constraints can be further reduced by relaxing the side constraints and introducing a penalty function. A nonlinear penalty function for integer programs was developed in Hadj-Alouane and Bean [12]. This penalty function has the form:

$$p_\lambda(y) = \sum_{t=1}^{H} \lambda_t [\min(0, E(t,Y) - B(t))]^2$$

By using this nonlinear penalty function, strong duality directly follows from Hadj-Alouane and Bean [12]. Define V(E2) to be the optimal objective function value for problem (E2) as given below. There exist multipliers, $\overline{\lambda} \geq 0$ such that for all $\lambda \geq \overline{\lambda}$, V(E1) = V(E2).

Now, with the reduction in constraints, feasible solutions in the relaxed problem can be found easily. Replace the action variable $y(p,a,t,i)$ with the new action variable $\overline{y}(p,t,i)$. Let $\overline{y}(p,t,i)$ equal the index of the action choice, $a \in A(p)$. Now, the multiple-choice constraints are embedded in the solution representation and the linearized disjunctive constraints are reduced to network flow constraints.

The relaxed model, (E2), is:

$$\max \sum_{p} \sum_{i \in S(p)} q(p,1,i)v(p,1,i) - \sum_{t=1}^{H} \lambda_t \min(0, E(t,Y) - B(t))^2 \tag{15}$$

(E2) subject to

$$v(p,t,i) = R(p,\overline{y}(p,t,i),t,i) + \alpha \sum_{j \in S(p)} P(p,\overline{y}(p,t,i),t,i,j)v(p,t+1,j) \ \textit{for all } a,t,i, \tag{16}$$

$$\overline{y}(p,t,i) \in A(p) \tag{17}$$

Now, the objective function (15) and constraints (16) are nonlinear. However, the number of constraints has been greatly reduced from (P1). Additionally, all solutions which satisfy the multiple-choice criteria (17) are feasible to (E2). This allows for simple construction and representation of feasible solutions for (E2).

2 Solution Technique

Because the problem (E2) is nonlinear with many local optima, a direct solution method cannot be employed for realistically sized problems. Therefore, a new variant of simulated annealing is used for the problem.

2.1 Simulated Annealing

Simulated annealing is a general heuristic procedure which follows a physical analogy of annealing. The algorithm generates a sequence of solutions in a search for near optima. To begin each iteration, a neighbor of the current solution is randomly selected. If the objective function value of the neighbor is better than the current solution it is accepted as the new current solution. If the value of the neighbor is worse, it is accepted with a probability dependent upon current temperature and the

objective function value difference between the current solution and the neighbor solution. The temperature during the simulated annealing procedure changes according to a cooling schedule. At the beginning of the procedure, the temperature is high so that worse solutions are likely to be accepted. Towards the end, the temperature is low and only improving solutions are accepted. For a more detailed description of simulated annealing, see Dowsland [11].

The general simulated annealing outline in Dowsland [11] was used. For our problem, a neighbor of solution y has all action choices the same except for one variable. This variable may take any action choice in the set of possible actions. The objective function, $f(y)$, is the function given in equation (15). Define c to be the current temperature. The future temperature is a function of c, $\eta(c)$. The cooling schedule was determined to be a linear function where $\eta(c) = \gamma c$ such that $\gamma < 1$. (Note: this is a geometric cooling function in terms of the iteration count.)

The inclusion of the penalty function and dual multipliers in the objective function (15) provides a unique problem. Previous applications using dual multipliers, Jellet [14] and Mittenthal [18], have assumed that these dual multipliers are given before the simulated annealing procedure begins. By using the nonlinear penalty function it has been shown in Hadj-Alouane and Bean [12] that there exist optimal multipliers, $\overline{\lambda} \geq 0$ such that for all $\lambda \geq \overline{\lambda}$, strong duality exists. Therefore, if the multipliers used are such that $\lambda(t) \geq \overline{\lambda}(t)$ and the solution obtained from simulated annealing is optimal for these multipliers and the solution is feasible in terms of the relaxed constraints then the solution is optimal. It would seem to make sense to set these multipliers to extremely large values such that they are greater than $\overline{\lambda}(t)$ for all t. However, by using such large multipliers the objective function becomes excessively penalized so that only solutions that are feasible to the resource limit constraints will be accepted even when the temperature is high. From empirical experience, the convergence is often slow for these large multiplier values. Since determining the optimal multiplier values is extremely difficult for this problem, a different approach is taken.

2.2 Pressure

Taking from the physical analogy of simulated annealing in which temperature refers to the acceptance probability of a non-improving solution, we develop the concept of a thermodynamic system. Based upon the thermodynamic equation, $PV = nRT$, we effect the system by altering pressure and temperature to induce a change in volume. In our analogy, volume refers to a fuzzy set representing

solutions that are acceptable with a 'reasonable probability.' Thus as temperature decreases, the volume will decrease as less solutions are included in the acceptable set. Pressure relates to the values of the multipliers for the penalty function. When the system is under high pressure the penalty function multipliers are large. Therefore the probability of selecting a nonfeasible solution is small and the volume is low. In low pressure situations (small multipliers) the algorithm is more likely to accept solutions that are infeasible in terms of the side constraints and the volume of the system increases. If the multipliers are in an increasing phase, the system is being compressed. If the multipliers are decreasing the system is being decompressed.

In this application, both cooling and compression follow a linear process. Thus, the volume is decreasing and the set of acceptable solutions is shrinking. The goal is to achieve a feasible solution near the optimal value by the end of the procedure. At the beginning of the algorithm the values for the multipliers start small such that $\lambda(t) \ll \overline{\lambda}(t)$. During the cooling step of the simulated annealing procedure, compression also occurs. The values for the multipliers change such that $\lambda(t)_{new} = \beta(\lambda(t)_{old})$. For our application, $\beta(\lambda(t)_{old}) = \delta(t)\lambda(t)_{old}$ where $\delta(t) > 1$ so that at the end of the algorithm $\lambda(t) \gg \overline{\lambda}(t)$. At the beginning of the procedure this method allows exploration of solutions infeasible in terms of the resource limit constraints. Towards the end of the algorithm, if the current solution is feasible, only improving feasible solutions are accepted. Otherwise, solutions with increased feasibility are preferred.

3 Bound Development

Due to the heuristic nature of simulated annealing, bounds on the optimal objective function value are needed to measure the quality of the solution produced. For large problems, calculating reasonable bounds for the expected value problem is extremely difficult.

We now return to the problem (E1) and introduce a linear relaxation of the side constraints in problem (E1) to form a Lagrangian relaxation. The resulting problem is used as a bound for the value of problem (E1). Lagrangian relaxation is used when the problem resulting from relaxing a set of constraints into the objective function is solvable. Solving this relaxation for a given set of multipliers will produce an upper bound for the optimal solution value to the original problem. Let $v_\Delta(p, t, i)$ be the maximum expected return value penalized with multiplier Δ. Define the penalized

reward value, $R_\Delta(p,a,t,i)$ to be:

$$R_\Delta(p,a,t,i) = R(p,a,t,i) - \Delta_t \pi(p,a,t,i) \tag{18}$$

and the penalized expected return to be:

$$N_\Delta(p,a,t,i) = R_\Delta(p,a,t,i) + \alpha \sum_j P(p,a,t,i,j) v_\Delta(p,t+1,j). \tag{19}$$

Then the following model (LR_Δ) is a Lagrangian relaxation of (E1):

$$V(LR_\Delta) = \sum_p \sum_i q(p,1,i) v_\Delta(p,1,i) + \sum_t \Delta_t \alpha^t B(t) \tag{20}$$

(LR_Δ) subject to

$$v_\Delta(p,t,i) = \max_a R_\Delta(p,a,t,i) + \alpha \sum_j P(p,a,t,i,j) v_\Delta(p,t+1,j \ \forall p,t,i \tag{21}$$

Lemma 1 *The optimal solution value to* (LR_Δ) *for any* $\Delta_t \geq 0$ *is an upper bound for (E2).*

Proof: See Appendix A.

To find a tight upper bound we need to find $\min_\Delta V(LR_\Delta)$. The most common approach to this type of problem is subgradient optimization (see Beasley [6]). This algorithm is an iterative procedure that uses the subgradients of the objective function to alter multiplier values. Subgradient optimization is a general method whereas multiplier adjustment methods (MAMs) rely upon specific problem structures. MAMs adjust the multipliers at each iteration so that the dual value is monotonically decreasing in upper bound problems.

An attempt at using subgradient optimization for our problem resulted in slow convergence. Therefore, a multiplier adjustment method is developed below.

3.1 Multiplier Adjustment Method

A successful multiplier adjustment method was implemented in the deterministic case of parallel equipment replacement in Karabakal, Bean, and Lohman [17]. However, because the structure of the problems are significantly different in this paper, a new multiplier adjustment method for our problem is presented.

The subgradients for the side constraints are given by

$$G_t = E(t,Y) - B(t)$$

For stage h with $G_h > 0$, the side constraint is infeasible. In this case we should increase Δ_h by $\theta_h > 0$ and thus encourage a decrease in infeasibility for stage h. For all other stages $t \neq h$, $\theta_t = 0$. Define $V_{\Delta+\theta}(p,t,i)$, $R_{\Delta+\theta}(p,a,t,i)$, and $N_{\Delta+\theta}(p,a,t,i)$ to be their respective values penalized by the multipliers $\Delta + \theta$. Below is a procedure for determining the value of θ_h so that $V(LR_\Delta) > V(LR_{\Delta+\theta})$. This is accomplished by showing, for certain values of θ_h, that the current optimal solution to (LR_Δ) defined as a^*_{phi} remains an optimal solution in $(LR_{\Delta+\theta})$ and the value of the solution is monotonically decreasing. To calculate possible values for θ_h, we need the following:

First, define $\delta(p,h,i)$ to be the difference between the optimal expected return for p,h,i and the next best expected return value or

$$\delta(p,h,i) = \min_{a \neq a^*_{phi}} (N_\Delta(p,a^*_{phi},h,i) - N_\Delta(p,a,h,i)).$$

Let $\zeta(h)$ be the minimum of all the $\delta(p,t,i)$ scaled by the distance from stage h or

$$\zeta(h) = \min_{p,i,t \leq h} (\delta(p,t,i)/\alpha^{h-t}).$$

Define $\pi_{max}(h)$ as the maximum resource consumption for optimal action choices in stage h or

$$\pi_{max}(h) = \max_{p,i} \pi(p,a^*_{phi},h,i)$$

Define the transition vector from stage $t < h-1$ to stage h to be $T_h(p,A_t,t,i,j)$ where A_t is the matrix of actions in periods t through h and

$$T_h(p,A_t,t,i,j) = \sum_k P(p,a,t,i,k) T_h(p,A_{t+1},t+1,k,j) \tag{22}$$

and for stage $h-1$

$$T_h(p,A_{h-1},h-1,i,j) = P(p,a,h-1,i,j) \tag{23}$$

Note that the expected budget in stage h is: $E_\Delta(h) = \sum_p \sum_i \sum_j q(p,1,i) T_h(p,A_1,1,i,j) \pi(p,a^*_{phi},h,i)$.

We now show that for a change of θ_h in the multiplier for period h, the values for the maximum expected returns do not change in the stages following stage h.

Lemma 2 *For stages $t > h$, $v_{\Delta+\theta}(p,t,i) = v_\Delta(p,t,i)$.*

Proof: Since $\theta_t = 0$ and $N_{\Delta+\theta}(p,t,a,i) = N_\Delta(p,t,a,i)$ for all $p,a,i,t > h$, the maximum expected return values do not change. ■

Below is a theorem which states the changes in the maximum expected return values for stage h and the stages prior to h. The theorem is limited to a range of values for θ_h.

Theorem 1 *For $\pi(p,a,t,i) \geq 0$ for all p,a,t,i and for $0 \leq \theta_h \leq (\zeta(h)/\pi_{max}(h))$ the maximum expected return values in $(LR_{\Delta+\theta})$, are:*

a) for stage h : $v_{\Delta+\theta}(p,h,i) = v_\Delta(p,h,i) - \theta_h \pi(p, a^*_{phi}, h, i)$

b) for stages $t < h$: $v_{\Delta+\theta}(p,t,i) = v_\Delta(p,t,i) - \alpha^{h-m}\theta_h \sum_j T_h(p, A_t, t, i, j)\pi(p, a^*_{phi}, h, j)$.

Proof : See Appendix B.

Now, given the changes in the maximum expected return values, the change in the objective function value for problem $(LR_{\Delta+\theta})$ from problem (LR_Δ) can be found.

Corollary 1 *For the conditions given in Theorem 1,*

$$V(LR_\Delta) - V(LR_{\Delta+\theta}) = \alpha^{h-1}\theta_h((E(h) - \alpha B(h))$$

Proof: The objective function of $(LR_{\Delta+\theta})$ is:

$$\begin{aligned}
V(LR_{\Delta+\theta}) &= \sum_p \sum_i q(p,1,i) v_{\Delta+\theta}(p,1,i) + \sum_t (\Delta_t + \theta_t)\alpha^t B(t) \\
&= \sum_p \sum_i q(p,1,i)(v_\Delta(p,1,i) - \alpha^{h-1}\theta_h \sum_j T_h(p, A_1, 1, i, j)\pi(p, a^*_{phi}, h, j)) \\
&\quad + \sum_t \Delta_t \alpha^t B(t) + \theta_h \alpha^h B(h) \\
&= V(LR_\Delta) - \theta_h(\alpha^{h-1} \sum_p \sum_i \sum_j q(p,1,i) T_h(p, A_1, 1, i, j)\pi(p, a^*_{phi}, h, j) - \alpha^h B(h)) \\
&= V(LR_\Delta) - \theta_h \alpha^{h-1}(E_\Delta(h) - \alpha B(h))
\end{aligned}$$

Therefore, since $E_\Delta(h) > B(h), \alpha > 0$, and $\theta_h > 0$ the value of the objective function decreases by $\theta_h \alpha^{h-1}(E(h) - \alpha B(h))$. ■

Theorem 1 and Corollary 1 combine to give us a multiplier adjustment method for finding a tight upper bound. Solving (LR_Δ) for a given value of Δ will give an upper bound for problem (E1). By

increasing the multiplier for an infeasible side constraint in stage h by θ_h as defined in Theorem 1, the resulting solution value will decrease by $\theta_h \alpha^{h-1}(E(h) - \alpha B(h))$. Repeating this for resulting solutions gives a multiplier adjustment method for tightening the upper bound value. Corollary 1 shows that at each iteration, the value for the upper bound is decreasing. The procedure ends when either a feasible solution is

found or no improvement exists after a predefined number of iterations. Therefore the multiplier adjustment method is not guaranteed to give an optimal solution for the dual problem $\min_{\Delta} V(LR_{\Delta})$, but gives a relatively tight bound.

4 Computational Results

The simulated annealing with compression algorithm was coded in C and run on a Sun Ultra-2 workstation. Random test problems were created using a data generator based upon pavement maintenance problems. Pavement maintenance is a case of parallel equipment replacement where the problem is to schedule maintenance or restoration alternatives for street segments. The goal is to maximize service benefits so that the costs of actions are within budget limits. Karabakal, Bean, and Lohmann [17] present an approach for solving deterministic pavement maintenance problems. Since pavement deterioration is often highly variable, the deterministic assumption is not valid in many cases. In this section, we present computational results using pavement maintenance problems with stochastic deterioration.

4.1 Problem Generator

The data generator was based upon the scheme for deterministic problems in Karabakal, Bean, and Lohmann [17]. Stochastic deterioration follows from Carr and Kim [7]. For pavement maintenance, each street segment is considered an asset. The street segment areas $\rho(p)$ were sampled from the uniform distribution: U(1000,7000). Each asset could be in seven possible service states and has four possible actions. The service level of a street segment is given by its pavement condition index (PCI). PCI is a number between 0 and 100, 0 being worst, 100 being best. The cost for each action option was determined by the following equations where $i \in \{1, 2, \ldots, 7\}$ is the service state number, $PCI(i)$ is the corresponding pavement condition index for state i, t is the current stage,

and $r(p)$ is the area of street segment p:

$$\begin{aligned}
\pi(p,1,t,i) &= 15.016r(p)(0.948)^{PCI(i)} \\
\pi(p,2,t,i) &= r(p)(8.5+0.02m+0.1(100-PCI(i))) \\
\pi(p,3,t,i) &= r(p)(10+.02m+.12(100-PCI(i))) \\
\pi(p,4,t,i) &= r(p)(16.42+.124m)
\end{aligned}$$

Action 1 represents the do-nothing option, actions 2 and 3 are two types of overlays and action 4 is reconstruction. The cost equations follow from Karabakal, Bean, and Lohmann [17]. The benefit for each action is determined by weighted value of the service state and the cost of the action. The following equation is the benefit value:

$$R(p,a,t,i) = 1000i - \pi(p,a,t,i) \tag{24}$$

Salvage values, $Z(p,i)$, were zero for all assets regardless of service state. Transition probabilities were assumed to follow an exponential deterioration as in the deterministic approach in Carr and Kim [7]. The following scheme was used to determine transition probabilities:

Let i^* be the maximum possible service state level for an asset during the next stage. This level was determined by the current state level and action choice using the following equations:

action 1: $i^* = i - 1$

action 2: $i^* = i$

action 3: $i^* = i + 1$

action 4: $i^* = 7$ (the maximum service state)

For the do-nothing option (action 1) the service level degrades by one degree. For the first overlay option (action 2) the service level remains constant while for the second overlay option (action 3) the service level improves one degree. When reconstruction (action 4) is chosen, the pavement section is at the maximum service level. For each segment p, define $d(p) = \rho(p)/7000 + .9$, where $(.9 < d(p) \leq 1.1)$. The deterioration rate is $\phi(p,t,i^*) = (d(p)i^*)/(.2m + i^* - 1)$. The transition probabilities are then determined by the following equations:

$$P(p,a,t,i)(i^*) = (1 - e^{-\phi(p,t,i^*)})/(1 - e^{-i^*\phi(p,t,i^*)}) \tag{25}$$

$$P(p,a,t,i,j) = (e^{(j-i^*)\phi(p,t,i^*)} - e^{(j-i^*-1)\phi(p,t,i^*)})/(1 - e^{-i^*\phi(p,t,i^*)}) \tag{26}$$

$$j = 1, \ldots, i^* - 1$$

Budget values are proportional to the area for each segment: $B(t) = \epsilon \sum_p \rho(p)$, where ϵ can be set to vary the tightness of the budget constraints.

4.2 Computational Experiments

Computational experiments were conducted on problems of varying problem sizes to measure computational times for varying values for ϵ. The values for ϵ were 6, 7, and 8 for 50 asset problems. For 100 asset problems the values of ϵ were 5, 6, and 7. Note that larger values of ϵ represent larger budget values. The values of ϵ were varied to measure the effect of tighter constraints on CPU time. Time horizons of 5 and 10 periods were used for the 50 asset problems. Four problem instances were tested for each problem size and value of ϵ. Each problem instance was run using the simulated annealing with compression code using ten different random number seeds. A discount factor of .9 was used for the data sets. The simulated annealing code used the following cooling and compression schedule: $\eta(c) = .98c$ and $\beta(\lambda(t)) = 1.05\lambda(t)$. The simulated annealing procedure was ended when the value reached a given tolerance to the upper bound.

Table I contains the results for solution values for problems with 50 assets over a 5 period time horizon. The first column corresponds to the problem parameters where p is the number of assets, t is the number of periods, and ϵ is the budget level as given above. The tolerance allowed is given in TOL. VAR is the total number of integer variables. The multiplier adjustment method was coded in C and the resulting upper bound is reported in column MAM. These minimum, median, and maximum values from the simulated annealing solution are reported in the columns corresponding to SA SOL. CPU times for reaching the given tolerance are reported in seconds.

Table II contains results for problems with 50 assets and 10 periods. Table III is for problems with 100 assets and 5 periods.

5 Conclusions

The stochastic cases of parallel equipment replacement problems motivated the study of nonhomogeneous Markov decision processes linked by side constraints. No previous research for this complex problem exists in the literature. A mixed-integer nonlinear programming formulation of the problem

was introduced. A nonlinear penalty function was applied to relax the side constraints. A multiplier adjustment method was constructed to produce upper bound values for the problem.

The relaxed problem with the nonlinear penalty function was heuristically solved by a variant of simulated annealing. This variant introduced the concepts of pressure and volume to alter the multiplier values. This variant of simulated annealing has the potential to address the general class of global optimization problems with relaxed constraints where the multiplier values are not easily determined.

Because no previous research was found, a library of test problems was generated. Computational results were completed for problems of 7,000 and 14,000 binary variables. These problems were solved within 5% and 10% tolerances of the upper bound value respectively. Most problems were solved within a reasonable amount of CPU time. Problems of $\epsilon = 6$ for the 50 asset problems did not have a median CPU time of less than 10,000 seconds for all problem instances. As expected, computational times were shown to decrease with decreasing tightness of the budget constraints. The formulation, multiplier adjustment method and the simulated annealing variant with compression presented in this paper combine to give an effective approach for solving the complex problem of nonhomogeneous Markov decision processes linked by side constraints. Computational results show that these methods can solve a 14,000 binary variable nonlinear mixed-integer program in a reasonable amount of CPU time.

Acknowledgment

This work was supported in part by the National Science Foundation grants DDM-9202849 and DMI-9634712, the Great Lakes Center for Truck and Transit Research, and the Alfred P. Sloan Foundation through the University of Michigan Trucking Industry Program.

Table I: Computational Experiments for $(p = 50, t = 5)$ problems using a Sun Ultra-2 workstation.

				SA SOL			Seconds		
(p, t, ϵ)	TOL	VAR	MAM	Min	Median	Max	Min	Median	Max
(50,5,6)	5%	7000	10021	9394	9520	9520	68.49	3965.93	>10,000
(50,5,6)	5%	7000	10194	9534	9684	9957	225.72	9792.73	>10,000
(50,5,6)	5%	7000	10628	10005	10098	10100	50.73	1503.68	84.66
(50,5,6)	5%	7000	9789	8887	9184	9276	>10,000	>10,000	>10,000
(50,5,7)	5%	7000	10693	10159	10160	10162	38.07	45.47	180.9
(50,5,7)	5%	7000	10615	10084	10085	10085	38.38	53.42	5392.83
(50,5,7)	5%	7000	10489	9857	9965	9966	42.97	630.81	$> 10,000$
(50,5,7)	5%	7000	10390	9871	9871	9873	40.00	49.30	2783.57
(50,5,8)	5%	7000	10533	10007	10007	10007	43.82	49.43	628.57
(50,5,8)	5%	7000	10758	10221	10221	10224	41.81	46.92	55.49
(50,5,8)	5%	7000	11083	9884	9884	9886	57.98	137.52	137.52
(50,5,8)	5%	7000	10474	9950	9950	9970	42.26	55.00	1150.65

Table II: Computational Experiments for $(p = 50, t = 10)$ problems using a Sun Ultra-2 workstation.

				SA SOL			Seconds		
(p, t, ϵ)	TOL	VAR	MAM	Min	Median	Max	Min	Median	Max
(50,10,6)	10%	14000	17399	15272	15364	15506	>10,000	>10,000	>10,000
(50,10,6)	10%	14000	17317	15114	15198	15449	>10,000	>10,000	>10,000
(50,10,6)	10%	14000	17923	15789	15938	16132	8463.53	>10,000	>10,000
(50,10,6)	10%	14000	17626	15503	15709	15864	6182.16	>10,000	>10,000
(50,10,7)	10%	14000	17530	15531	15775	15779	419.94	8174.62	>10,000
(50,10,7)	10%	14000	18142	16328	16328	16338	150.14	681.57	3696.30
(50,10,7)	10%	14000	18047	16226	16243	16245	144.77	1065.65	>10,000
(50,10,7)	10%	14000	17614	15765	15853	15856	179.99	1865.56	>10,000
(50,10,8)	10%	14000	18888	17000	17002	17010	83.90	93.80	100.78
(50,10,8)	10%	14000	18385	16547	16549	16553	87.12	102.13	125.14
(50,10,8)	10%	14000	18567	16711	16712	16720	86.66	100.03	122.17
(50,10,8)	10%	14000	17907	16117	16117	16121	96.32	105.90	135.50

Table III: Computational Experiments for $(p = 100, t = 5)$ problems using a Sun Ultra-2 workstation.

				SA SOL			Seconds		
(p, t, ϵ)	TOL	VAR	MAM	Min	Median	Max	Min	Median	Max
(100,5,5)	10%	14000	19463	17517	17519	17553	102.13	137.24	648.96
(100,5,5)	10%	14000	19634	17671	17672	17747	103.60	127.25	227.05
(100,5,5)	10%	14000	19843	17859	17861	17988	96.13	122.19	491.97
(100,5,5)	10%	14000	19918	17926	17927	18001	95.13	134.52	318.79
(100,5,6)	10%	14000	20054	18049	18050	18084	92.28	110.49	151.04
(100,5,6)	10%	14000	20894	18805	18807	18999	91.04	104.70	134.06
(100,5,6)	10%	14000	20956	18861	18864	19049	88.50	102.20	168.01
(100,5,6)	10%	14000	20744	18670	18673	18755	89.43	95.50	115.09
(100,5,7)	10%	14000	21335	19202	19208	19526	78.60	95.04	152.3
(100,5,7)	10%	14000	21302	19172	19175	19180	84.78	89.63	101.67
(100,5,7)	10%	14000	20838	18755	18761	19245	94.95	99.66	126.91
(100,5,7)	10%	14000	21704	19353	19542	19549	78.73	83.44	92.43

6 Appendix A

Proof of Lemma 1:

To facilitate the proof of Lemma 1 we use the following notation as described below. Define p, a, t, and i as before and:

X_t^p = Vector of actions where each element corresponds to the action in state i for process p during stage t.

X^p = Matrix of actions for process p.

X_t = Matrix of actions for stage t.

X = Matrix of actions for all processes and stages.

$\overline{X}$ = Set of all possible X.

$P_t^p(X_t^p)$ = Transition matrix from stage t to stage $t+1$ for process p if policy X_t^p is implemented during stage t.

$R_t^p(X_t^p)$ = Vector of benefits for policy X_t^p for process p during stage t.

$T_{1m}^p(X^p)$ = $\prod_{k=1}^{t} P_t^p(X_k^p)$

Q_1^p = Vector of initial state probabilities for process p.

V = $\sum_p \sum_{t=1}^{H} \alpha^t Q_1^p T_{1t}^p(X^p) R_t^p(X_t^p)$

$\Pi_t^p(X_t^p)$ = Vector of costs for policy X_t^p for process p during stage t.

$E_t(X_t)$ = $\sum_p Q_1^p T_{1m}^p(X^p) \Pi_t^p(X_t^p)$

Now, (E1) can be written in this notation as problem (E3).

$$\max_{X \in \overline{X}} V \tag{27}$$

(E3) subject to

$$E_t(X_t) - B_t \leq 0 \ for\ all\ m \tag{28}$$

Now, for $\Delta_t \geq 0$ define the penalized reward to be:

$$R_t'^p(X_t^p) = R_t^p(X_t^p) - \Delta_t \Pi_t^p(X_t^p) \tag{29}$$

and the penalized value to be

$$V' = \sum_p \sum_{t=1}^{H} \alpha^t Q_1^p T_{1m}^p(X^p) R_t'^p(X_t^p) \tag{30}$$

Define problem (UE3) with penalized reward values to be:

$$U(\Delta) = \max_{X \in \overline{X}} V' + \sum_t \Delta_t \alpha^t B_t \tag{31}$$

Note that (E3) and (LR_Δ) are equivalent.

The Lagrangian relaxation of constraint (28) in (E3) is the problem (LE3):

$$\max_{X \in \overline{X}} V_\lambda = V - \sum_t \lambda_t (E_t(X_t) - B_t) \tag{32}$$

For $\lambda_t = \Delta_t \alpha^t \geq 0$, (LE3) is an upper bound for (E3) and

$$\begin{aligned} V_\lambda &= V - \sum_t \Delta_t \alpha^t (\sum_p Q_1^p T_{1m}^p(X^p) - B_t) \\ &= \sum_p \sum_{t=1}^{H} \alpha^t Q_1^p T_{1m}^p(X^p) R_t^p(X_t^p) - \sum_t \Delta_t \alpha^t \sum_p Q_1^p T_{1m}^p(X^p) \Pi_t^p(X_t^p) + \sum_t \Delta_t \alpha^t B_t \\ &= \sum_p \sum_{t=1}^{H} \alpha^t Q_1^p T_{1m}^p(X^p) R_t'^p(X_t^p) + \sum_t \Delta_t \alpha^t B_t \\ &= V' + \sum_t \Delta_t \alpha^t B_t \end{aligned}$$

Therefore, the optimal solution to (UE3) and is an upper bound for (E3) ■

Thus, in our original notation, penalizing the rewards so that,

$$R_\Delta(p, a, t, i) = R(p, a, t, i) - \Delta_t \pi(p, a, t, i)$$

gives an upper bound for (E1).

7 Appendix B

Proof of Theorem 1: Proof (a): If a^*_{phi} is the optimal choice to (LR_Δ) then

$$\begin{aligned}
N_{\Delta+\theta}(p, a^*_{phi}, h, i) &= N_\Delta(p, a^*_{phi}, h, i) - \theta_h \pi(p, a^*_{phi}, h, i) \\
&\geq N_\Delta(p, a^*_{phi}, h, i) - (\zeta(h)/\pi_{max}(h))\pi(p, a^*_{phi}, h, i) \\
&\geq N_\Delta(p, a^*_{phi}, h, i) - \zeta(h) \\
&\geq N_\Delta(p, a^*_{phi}, h, i) - (\delta(p, h, i)/\alpha^{h-h}) \\
&\geq N_\Delta(p, a^*_{phi}, h, i) - (N_\Delta(p, a^*_{phi}, h, i) - N_\Delta(p, a, h, i))\ \forall c \\
&= N_\Delta(p, a, h, i)\ \forall c \\
&\geq N_{\Delta+\theta}(p, a, h, i)
\end{aligned}$$

Thus, a^*_{phi} remains optimal in $(LR_{\Delta+\theta})$ and value of $v_{\Delta+\theta}(p, h, i) = N_{\Delta+\theta}(p, a^*_{phi}, h, i) = v_\Delta(p, h, i) - \theta_h \pi(p, a^*_{phi}, h, i)$. ■

Proof (b): Proof by induction. Define $z = h - m$.

For $z = 1$:

The expected return values in $(LR_{\Delta+\theta})$ for stage $h-1$ are:

$$\begin{aligned}
N_{\Delta+\theta}(p, a, h-1, i) &= R(p, a, h-1, i) - (\Delta_{h-1} + \theta_{h-1})\pi(p, a, h-1, i) \\
&\quad + \alpha \sum_j P(p, a, h-1, i, j) v_{\Delta+\theta}(p, h, j) \\
&= R(p, a, h-1, i) - \Delta_{h-1}\pi(p, a, h-1, i) \\
&\quad + \alpha \sum_j P(p, a, h-1, i, j)(v_\Delta(p, h, j) - \theta_h \pi(p, a^*_{phi}, h, j)) \\
&= N_\Delta(p, a, h-1, i) - \alpha\theta_h \sum_j P(p, a, h-1, i, j)\pi(p, a^*_{phi}, h, j))
\end{aligned}$$

The expected return values in $(LR_{\Delta+\theta})$ for stage $h-1$ for the action choices optimal to (LR_Δ) are:

$$\begin{aligned}
N_{\Delta+\theta}(p, a^*_{p,h-1,i}, h-1, i) &= N_\Delta(p, a^*_{p,h-1,i}, h-1, i) - \alpha\theta_h \sum_j P(p, a^*_{p,h-1,i}, h-1, i, j)\pi(p, a^*_{phi}, h, j)) \\
&\geq N_\Delta(p, a^*_{p,h-1,i}, h-1, i) - \alpha\zeta(h-1) \\
&\geq N_\Delta(p, a^*_{p,h-1,i}, h-1, i) - \alpha(\delta(p, h-1, i)/\alpha^{h-(h-1)}) \\
&\geq N_\Delta(p, a^*_{p,h-1,i}, h-1, i) - (N_\Delta(p, a^*_{p,h-1,i}, h-1, i) - N_\Delta(p, a, h-1, i))\ \forall a \\
&= N_\Delta(p, a, h-1, i)\ \forall a
\end{aligned}$$

$$\geq N_{\Delta+\theta}p, a, h-1, i \; \forall a$$

Therefore, the same action choices remain optimal in $(LR_{\Delta+\theta})$ and

$$\begin{aligned} v_{\Delta+\theta}(p, h-1, i) &= N_{\Delta+\theta}(p, a^*_{p,h-1,i}, h-1, i) \\ &= N_{\Delta}(p, a^*_{p,h-1,i}, h-1, i) - \alpha\theta_h \sum_j P(p, a^*_{p,h-1,i}, h-1, i, j)\pi(p, a^*_{phi}, h, j)) \\ &= v_{\Delta}(p, h-1, i) - \alpha\theta_h \sum_j T_h(p, A_{h-1}, h-1, i, j)\pi(p, a^*_{phi}, h, j) \end{aligned}$$

Induction step. Assume that for z:

$$v_{\Delta+\theta}(p, h-z, i) = v_{\Delta}(p, h-z, j) - \alpha^z\theta_h \sum_j T_h(p, A_{h-z}, h-z, i, j)\pi(p, a^*_{p,h-z,i}, h-z, j)$$

Now, for $z+1$:

$$\begin{aligned} N_{\Delta+\theta}(p, a, h-z-1, i) &= R(p, a, h-z-1, i) - (\Delta_{h-z-1} + \theta_{h-z-1})\pi(p, a, h-z-1, i) \\ &\quad +\alpha \sum_j P(p, a, h-z-1, i, j)v_{\Delta+\theta}(p, h-z, j) \\ &= R(p, a, h-z-1, i) - \Delta_{h-z-1}\pi(p, a, h-z-1, i) \\ &\quad +\alpha \sum_j P(p, a, h-z-1, i, j)(v_{\Delta}(p, h-z, j) \\ &\quad -\alpha^z\theta_h \sum_k T_h(p, A_{h-z}, h-z, j, k)\pi(p, a^*_{p,h-z,i}, h-z, k)) \\ &= N_{\Delta}(p, a, h-z-1, i) \\ &\quad -\alpha^{z+1}\theta_h \sum_k T_h(p, A_{h-z-1}, h-z-1, j, k)\pi(p, a^*_{p,h-z,i}, h-z, k)) \end{aligned}$$

The expected return values in $(LR_{\Delta+\theta})$ for the action choices optimal to (LR_{Δ}) are:

$$\begin{aligned} N_{\Delta+\theta}(p, a^*_{p,h-z-1,i}, h-z-1, i) &= N_{\Delta}(p, a^*_{p,h-z-1,i}, h-z-1, i) \\ &\quad -\alpha^{z+1}\theta_h \sum_k T_h(p, A_{h-z-1}, h-z-1, j, k)\pi(p, a^*_{p,h-z,i}, h-z, k)) \\ &\geq N_{\Delta}(p, a^*_{p,h-z-1,i}, h-z-1, i) - \alpha^{z+1}\zeta(h-z-1) \\ &\geq N_{\Delta}(p, a^*_{p,h-z-1,i}, h-z-1, i) - \alpha^{z+1}(\delta(p, h-z-1, i)/\alpha^{h-(h-z-1)} \\ &\geq N_{\Delta}(p, a^*_{p,h-z-1,i}, h-z-1, i) - (N_{\Delta}(p, a^*_{p,h-z-1,i}, h-z-1, i) \end{aligned}$$

$$
\begin{aligned}
&-N_{\Delta}(p,a,h-z-1,i))\ \forall a \\
&= N_{\Delta}(p,a,h-z-1,i)\ \forall a \\
&\geq N_{\Delta+\theta}(p,a,h-z-1,i)\ \forall a
\end{aligned}
$$

Therefore, the same action choices remain optimal in $(LR_{\Delta+\theta})$ and

$$
\begin{aligned}
v_{\Delta+\theta}(p,h-z-1,i) \quad &= N_{\Delta+\theta}(p,a^{*}_{p,h-z-1,i},h-z-1,i) \\
&= N_{\Delta}(p,a^{*}_{p,h-z-1,i},h-z-1,i) - \alpha\theta_h \sum_{j} T_h(p,A_h,h-z-1,i,j)\pi(p,a^{*}_{phi},h,j)) \\
&= v_{\Delta}(p,h-z-1,i) - \alpha\theta_h \sum_{j} T_h(p,A_h,h-z-1,i,j)\pi(p,a^{*}_{phi},h,j))
\end{aligned}
$$

Thus, by mathematical induction, the statement holds. ■

References

[1] E. Balas, 1975. Disjunctive Programming: Cutting Planes from Logical Conditions. *Nonlinear Programming 2.*

[2] E. Balas, 1979. Disjunctive Programming. *Annals of Discrete Mathematics*, 5:3-51.

[3] James C. Bean, 1992. A Bender's Approach to Disjunctive Programming. Technical Report 92-26, University of Michigan.

[4] James C. Bean, Wallace J. Hopp and Izak Duenyas, 1992. A Stopping Rule for Forecast Horizons in Nonhomogeneous Markov Decision Processes. *Operations Research*, 40:1188-1199.

[5] James C. Bean, Robert L. Smith, and Jean B. Lasserre, 1990. Denumerable State Nonhomogeneous Markov decision processes. *Journal of Mathematical Analysis and Applications*, 153:64-77.

[6] John E. Beasley. 1993 Lagrangian Relaxation in Colin Reeves (ed), *Modern Heuristic Techniques for Combinatorial Problems.*

[7] Robert I. Carr and Chang-Duk Kim, 1993. Construction and Maintenance Actions for Multi-Unit Deteriorating Facilities. *Fifth International Conference on Computing in Civil and Building Engineering*, 1-8.

[8] Eric V. Denardo, 1982. *Dynamic Programming: Models and Applications*, Prentice-Hall.

[9] Cyrus Derman, 1970. *Finite State Markov decision processes*, Academic Press, Orlando.

[10] Cyrus Derman, 1972. Constrained Markov action Chains. *Management Science*, 19:389-390.

[11] Kathryn A. Dowsland, 1993. Simulated Annealing in Colin Reeves(ed), *Modern Heuristic Techniques for Combinatorial Problems.*

[12] Atidel Ben Hadj-Alouane and James C. Bean, 1997. A Genetic Algorithm for the Multiple-Choice Integer Program. *Operations Research*, 45:92-101.

[13] IBM Corporation. *Optimization Subroutine Library Guide and Reference.*

[14] P.M. Jellet, 1990. Simulated Annealing for a Constrained Allocation Problem. *Mathematics and Computers in Simulation*, 32:149-154.

[15] R. Jeroslow, 1979. An Introduction to the Theory of Cutting Planes. *Annals of Discrete Mathematics*, 5:71-95.

[16] Peter Kall and Stein W. Wallace, 1994. *Stochastic Programming*, John Wiley, Chichester.

[17] Nejat Karabakal, James C. Bean and Jack R. Lohmann, 1994. Scheduling Pavement Maintenance with Deterministic Deterioration and Budget Constraints. Technical Report 94-18, University of Michigan.

[18] John Mittenthal, 1997. A Comparison of Global Search Heuristics for the TSSP+1. Working Paper, University of Alabama.

[19] Yunsun Park, James C. Bean and Robert L. Smith, 1993. Optimal Average Value in Nonhomogeneous Markov Decision Processes. *Journal of Mathematical Analysis and Applications*, 179:525-536.

[20] P. Varaiya, 1978. Optimal and Sub-Optimal Stationary Controls for Markov Chains. *IEEE Transactions A.C.*, AC-23:388-394.

[21] D.J. White, 1995. Finite Horizon Markov Decision Processes with Uncertain Terminal Payoffs. *Operations Research*, 43:862-869.

[22] Chelsea C. White and Kent Schlussel, 1981. Suboptimal Design for Large Scale, Multimodule Systems. *Operations Research*, 29:865-875.