Available online at www.sciencedirect.com
SCIENCE DIRECT®
Operations Research Letters 34 (2006) 307–315
Operations Research Letters
www.elsevier.com/locate/orl

SPAR: stochastic programming with adversarial recourse

Matthew D. Bailey*, Steven M. Shechter, Andrew J. Schaefer

Department of Industrial Engineering, University of Pittsburgh, Pittsburgh, PA 15261, USA

Received 15 July 2004; accepted 10 May 2005
Available online 20 June 2005

Abstract

We consider a general adversarial stochastic optimization model. Our model involves the design of a system that an adversary may subsequently attempt to destroy or degrade. We introduce SPAR, which utilizes mixed-integer programming for the design decision and a Markov decision process (MDP) for the modeling of our adversarial phase.

Keywords: Stochastic programming; Markov decision processes; Network interdiction; Game theory

1. Introduction and background

We introduce a general modeling technique for optimally designing a system that an adversary may subsequently attempt to destroy or degrade. Such decisions arise in various situations including the design of computer networks, energy grids, military supply networks, and biological agent sensor placement. In our framework, there are two types of uncertainty: *design uncertainty* and *adversarial uncertainty*. The decision maker must first make a design decision. Typically, there will be a large number of such designs, such as where to locate detectors, or how to allocate protective resources. Since many design decisions have long implementation cycles, these decisions must be made under design uncertainty that is not revealed until after a design has been selected. This uncertainty only occurs once and impacts the parameters of the subsequent adversarial uncertainty. Once the design has been implemented, the decision maker faces threats from an adversary who searches for optimal decisions that negatively impact the system performance. Therefore, the adversary selects his alternatives based on those that create the most significant disruption of service or system damage under the current design. These adversarial decisions will typically occur over multiple time periods, and are subject to the randomness that is inherent in the adversary's system environment, called adversarial uncertainty. The objective of the designer is to select a design that minimizes the installation cost plus the discounted expected expense of the adversary's optimal reaction. In this paper we assume that system redesign is expensive and incurs equally long implementation cycles. As a result, we are modeling the one-time design decision and assume the

* Corresponding author.
E-mail address: mdbailey@pitt.edu (M.D. Bailey).

0167-6377/$ - see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.orl.2005.05.002

objective is to minimize the cost of installation and the impact of a long-run attack strategy. We propose a technique for solving such adversarial, stochastic, and dynamic decision models that combines the best aspects of *stochastic programming* (SP) and *Markov decision processes* (MDPs). This new methodology has many applications and extends the modeling and solution techniques for adversarial multistage decisions under uncertainty. We call this modeling methodology *SPAR: stochastic programming with adversarial recourse*.

SPAR generalizes the stochastic network interdiction problem [6,10,15]. In these models the *interdictor* is trying to maximize one of a variety of objectives, including the expected minimum length between an origin s and destination t; the probability of detecting an adversary on a link in the network; or the probability of causing the minimum length path to be above a specified threshold. In Pan et al. [15], the stochasticity arises from an uncertain origin and destination of the evader. In Hemmecke et al. [10] and Cormicon et al. [6], the characteristics of the network are not known in advance. However, in each case, after the uncertainty is resolved the evader faces a deterministic decision problem. In the case of maximizing the shortest path in a network, this is equivalent to the evader solving a deterministic dynamic programming problem. Unlike previous work, SPAR is able to incorporate a first-stage design cost and focus on minimizing the negative impact of an adversary. In addition to uncertainty between the design and adversary phases, our adversary must make decisions in a stochastic environment. While stochastic network interdiction deals solely with an adversary facing evasion decisions, SPAR models the adversary's possible decisions as a broader stochastic control or allocation problem.

The problem of designing a system in anticipation of adversarial attacks can also be modeled as a special case of a discounted zero-sum stochastic game [9], where the adversary's only control choice during the design stage is to wait, and the system designer is inactive once the design has been selected. At this point, the adversary can dynamically choose from their action sets to decide how to negatively impact the system. However, modeling this as a stochastic game would require a state-space description for every feasible design where the combinatorially explosive number of design choices would cause significant if not insurmountable computational burdens.

A hybrid SP/MDP method is also presented by Cooper and Homem-de-Mello [5] for the problem of revenue management. Their problem can be modeled as a single large-scale finite-horizon MDP or a multistage stochastic program. They develop a heuristic method that can be viewed as a two-stage procedure where the second stage is derived from the optimal value function of the finite-horizon MDP. They present a technique for solving this stochastic integer and nonlinear optimization problem using Monte Carlo simulation. This model differs from SPAR in that it is not adversarial in nature and presents an approximation method for the solution of a problem that can be represented using conventional techniques. SPAR is an exact algorithm and modeling technique for problems for which there is no conventional modeling representation.

In Section 2 we provide an overview of SPAR and MDPs, illustrative applications, and a detailed mathematical formulation of SPAR. A computational illustration and comparative solution methods are discussed in Section 3.

2. Stochastic programming with adversarial recourse (SPAR)

In SPAR, the design decisions are modeled as mixed-integer programs (MIPs), which select the initial design allocation denoted by the vector x with n_1 elements from a mixed-integer set with an associated cost of $c^{\mathrm{T}}x$ and must satisfy the linear constraints $Gx = g$. After the system design decision x is chosen, the design uncertainty, denoted by a finite discrete random variable $\tilde{\xi}$, is resolved, resulting in a scenario ξ^k with probability q^k, $k = 1, \ldots, K$. The resulting adversarial decision-making process is modeled as an MDP defined by its state space $S(\xi^k)$, action space $A(\xi^k)$, state-action rewards $r(s, a, x, \xi^k)$ for $(s, a) \in (S(\xi^k), A(\xi^k))$, and probability transitions $p(j|s, a, \xi^k)$ for $(s, a) \in (S(\xi^k), A(\xi^k))$ and $j \in S(\xi^k)$. We include the parameter ξ^k to emphasize the variability and dependency of the adversary's model parameters on the design uncertainty. Similarly, the reward is linearly linked to the first-stage decision as will be illustrated in Section 2.4. The decision

sequence and dependencies are represented as:

Design decision x $\rightarrow$ Design uncertainty

ξ^k realized $\rightarrow$ Adversarial MDP (x, ξ^k).

By divorcing the design decisions from the adversary's MDP, SPAR is able to model detailed and complex instances of the adversarial phase while simultaneously considering a large number of initial design decisions.

Tying the design and adversary models together is a two-stage stochastic program with an objective of minimizing the first-stage costs plus the expected damage from an adversary over all possible scenarios. Although stochastic programming is computationally efficient for two-stage problems, it is ill suited for problems with a large number of stages in the adversary phase. Conversely, MDP is an effective tool for multi-stage sequential decisions with a relatively small number of states, but an inefficient technique for large-scale design decisions. Therefore, the principle advantage of SPAR is that it is a hybrid technique that utilizes mixed-integer programming for the design decision, an MDP for the modeling of our adversarial phase, and binds the two phases together with the computational effectiveness of two-stage stochastic programming.

The stochastic program has a mixed-integer first stage, and the subproblems are MDPs. The feasible actions of the subproblems will be linearly linked to the first stage decisions through a penalty parameter assigned to the adversary's rewards. SPAR can be extended to create varying state space, reward, and probability transition structures for the underlying MDP. SPAR exploits the fact that the expected recourse based on the adversary's MDP is piecewise linear and convex in the first-stage decision. As a result, we show that SPAR satisfies the two-stage stochastic programming framework. Additionally, the computational efficiencies of specialized MDP solution techniques can be utilized to accelerate the solution of the two-stage stochastic program.

2.1. Modeling illustrations

To illustrate the capabilities of SPAR we present two applications where SPAR is appropriate. Consider the problem of stochastic shortest path network interdiction [10]. In this model, the system designer (interdictor) with a constrained budget and resources must interdict links within a transportation network to maximize the expected minimum path of an adversary. The interdiction of an arc (i, j) results in an increase of the arc length by a distance of $d_{ij} > 0$. The expectation results from the uncertainty in the possible network configurations. However, for a given scenario the adversary is facing a deterministic shortest path decision. Consider a case where the adversary is traveling in a network with uncertainty. This can occur when an adversary can enter one of several queues whose occupants are subsequently routed to one of several security checkpoints. A network with uncertainty also results if a vehicle's route in an adversary's supply network is dynamic and stochastic due to hourly pickup/delivery uncertainty. By selecting the route or queue the adversary has some control over their travel, however the realization of their path is uncertain. These and similar extensions result in the adversary facing a stochastic shortest path problem [3] that can be modeled as an MDP.

Another SPAR application is as follows: consider a variant of the dynamic weapon target assignment problem under stochastic demand with N possible targets [14]. This problem deals with the allocation of weapons to targets that are revealed dynamically. At time $t = 1, \ldots, T$ an adversary has only discovered $n(t) \leqslant N$ of the targets with varying values, $n(t)$ is nondecreasing with time. The adversary can allocate some of their $M < \infty$ weapons to the known $n(t)$ targets or reserve them for undiscovered and possibly more desirable targets. However, waiting incurs a cost due to various factors, including the finite time from deployment for the weapon to detect and engage the targets. Now consider the military planning decision of locating these N facilities within a finite geographic region with the knowledge that the adversary will seek out and attack the facilities. For a given facility assignment, the adversary's attack strategies will dictate the number and order of the targets which are dynamically discovered. The military planner must allocate these facilities so as to minimize the expected impact of an eventual attack subject to the inherent installation constraints. This is an instance of SPAR where the attack strategies are the scenarios and the planning is forecasting the adversary's dynamic resource allocation decision using an MDP framework.

2.2. Markov decision processes

MDPs are a general technique to formulate problems involving a sequence of decisions made under uncertainty to optimize a given performance criteria. Aside from implicit separability assumptions about these decisions, the framework is very general. The formulation couples the decisions through state variables that succinctly summarize the impact on subsequent decisions of decisions made thus far.

The sequential nature of the problem allows for the defining of stages or decision epochs. We focus on infinite-horizon problems, since finite-horizon problems can be recast as infinite-horizon problems through the standard augmentation of the state with the stage. Following the notation of Puterman [16], we define S as the finite state space of the MDP. For every state $s \in S$, let the finite set of feasible decisions or actions be A_s, where for every action $a \in A_s$ the decision maker receives reward $r(s,a)$, where $|r(s,a)| \leqslant M < \infty$. A transition from state s to state j when action $a \in A_s$ is chosen occurs with probability $p(j \mid s,a)$. A policy $\mu = \{d_1, d_2, \ldots\}$ is a sequence of decision rules, where a decision rule d_i is a function mapping states into actions such that $d_i(s) \in A_s$.

We seek a policy that will maximize the total expected discounted rewards with discount parameter $0 \leqslant \lambda < 1$. It is well known that there exists an optimal policy which is stationary, i.e. $\mu = \{d, d, \ldots\}$ for some deterministic decision rule d. The optimal total expected discounted reward for each state $s \in S$ can be found by solving the standard set of Bellman equations [1]:

$$v(s) = \max_{a \in A_s} \left\{ r(s,a) + \lambda \sum_{j \in S} p(j \mid s,a) v(j) \right\}.$$

We can compute the total expected discounted reward vector, v_d, for a given stationary policy defined by decision rule d as the solution of the following equation:

$$v_d = r_d + \lambda P_d v_d, \tag{1}$$

where r_d is the vector of rewards under decision rule d, i.e. $r_d(s) = r(s, d(s))$, and P_d is the probability transition matrix under decision rule d, i.e. $(P_d)_{ij} = p(j \mid i, d(i))$. From (1), the value vector of a policy can be found by

$$v_d = (I - \lambda P_d)^{-1} r_d, \tag{2}$$

where the inverse of $(I - \lambda P_d)$ exists since P_d is stochastic and $0 \leqslant \lambda < 1$.

The separability of the MDP decisions allows for the above decomposition of the problem into smaller related subproblems. As a result, such decision problems can be solved by using a variety of techniques including value iteration [17], policy iteration [11], modified policy iteration [13], or linear programming [8]; however Koehler [12] has shown that simplex-based linear programming is typically not the most efficient method for solving a single MDP.

2.3. MDP extensive formulation

If there are a finite number of design decisions, the model as we have described, could in its entirety be formulated as a large-scale MDP. However, this would effectively require enumerating all the feasible designs to create the action space in the design phase and then computing the maximum-expected cost over each of the scenarios for the adversarial phase. To solve a problem instance using an MDP formulation for the first- and second-stage problems, where there are n_1 binary decisions in the design phase and K adversarial scenarios, could require the solution of $2^{n_1}K$ adversary subproblems. Typically, even relatively small design problems could have exponentially many possible solutions. Enumerating all possible complete designs would be burdensome, let alone making optimal decisions over this action space. SPAR decomposes the problem into many smaller problems using a variant of Benders' decomposition [2], thus allowing for the solution of significantly fewer first-stage and adversarial problems than would be required in solving a single large-scale MDP.

2.4. Mathematical formulation of SPAR

We now formally introduce the relationship between the design decision and the adversary's decision framework. We demonstrate that these interlinked problems satisfy the standard two-stage stochastic programming framework by showing that the expected recourse function (the adversary's expected impact) is piecewise linear and convex over $\{x \in X \mid Gx = g\}$.

In addition, for decomposition purposes we illustrate a supporting hyperplane based on the optimal policies of the adversary's MDPs for a given first-stage design vector.

In a conventional two-stage stochastic linear program [4], a first-stage LP must be solved before all of the problem parameters are known with certainty. The uncertainty is realized after this first stage is solved. At this point, the decision-maker solves a second linear program, known as a *recourse* problem, that considers the solution to the first LP as well as the outcome of the random event. The objective is to minimize the first-stage cost plus the expected second-stage cost. Let $m_1(n_1)$ be the number of rows (columns) in stage 1. Let $X = \mathbb{R}^l \times \mathbb{Z}^{n_1-l}$ for some $0 \leqslant l \leqslant n_1$, so that $\{x \in X \mid Gx = g\}$ is the set of feasible designs for the first stage, where G is a known matrix of size $m_1 \times n_1$ and g is a known vector in $\mathbb{R}^{m_1}$. The cost of design x is given by $c^{\mathrm{T}}x$, where c is a known vector in $\mathbb{R}^{n_1}$. Let $\tilde{\xi}$ be a discretely distributed random variable describing the design uncertainty and let Ξ be the finite support of $\tilde{\xi}$. For $k = 1, \ldots, K = |\Xi|$, let ξ^k describe the kth element in Ξ, called the *design scenario*, and let q^k be the probability that design scenario ξ^k is realized.

Suppose that the adversary's rewards r, transition probabilities p, and/or the discount factor λ are unknown before the design phase. The interpretation of the rewards is dependent on the decision maker: for a system designer these are costs to be minimized over the long run and for the adversary they are rewards for system disruption to be maximized over the long run. After the design decision x has been made and ξ^k is realized, the parameters defining the adversary's MDP are known.

For any design scenario ξ^k, $k = 1, \ldots, K$, let $p(j|s, a, \xi^k)$ be the adversary's transition probability from state $s \in S(\xi^k)$ to $j \in S(\xi^k)$ when action $a \in A_s(\xi^k)$ is chosen under scenario ξ^k. Similarly, let $r(s, a, \xi^k)$ be the reward for state-action pair (s, a) under scenario ξ^k. Let $\lambda(\xi^k)$ be the discount rate under scenario ξ^k. In addition, we define bounded adversary impediment values (benefiting the system designer), $t_i(s, a, \xi^k) \geqslant 0$ for $i = 1, \ldots, n_1$, $s \in S(\xi^k)$, $a \in A_s(\xi^k)$, that link the ith element of the design decision vector x to the reward of an action choice from the states in the adversary's realized MDP. The adversary's reward $r(s, a, \xi^k)$ is reduced by $t_i(s, a, \xi^k)x_i$ for $i = 1, \ldots, n_1$. As an example of modeling flexibility, if the given impediment value $t_i(s, a, \xi^k)$ is very large this may represent the ability of the designer to eliminate a decision possibility at this state if the first-stage decision variable is nonzero. This is analogous to removing an arc in a classic deterministic network. Typically, it will reflect an added "cost" the adversary must incur to choose this action as a result of the system designer's decisions.

Let $Q(x, \xi^k)$ be the expected reward of the optimal policy of the adversarial MDP resulting from design x under scenario ξ^k and $\alpha(\xi^k)$ the scenario specific probability vector over the initial states in the MDP subproblem. Then,

$$Q(x, \xi^k) = \alpha^{\mathrm{T}}(\xi^k)v^*(x, \xi^k),$$

where the $|S(\xi^k)|$-dimensional vector $v^*(x, \xi^k)$ is the solution of the Bellman equations for the adversarial MDP under design x and scenario ξ^k,

$$v(s) = \max_{a \in A_s(\xi^k)} \left\{ r(s, a, \xi^k) - \sum_{i=1}^{n_1} t_i(s, a, \xi^k)x_i + \lambda(\xi^k) \sum_{j \in S(\xi^k)} p(j \mid s, a, \xi^k)v(j) \right\}$$

for all $s \in S(\xi^k)$. We define $\mathcal{Q}(x)$, the *expected adversarial recourse function*, by

$$\mathcal{Q}(x) = \mathbb{E}_{\tilde{\xi}}[Q(x, \tilde{\xi})].$$

The objective of SPAR is to find a design that minimizes $cx + \mathcal{Q}(x)$ subject to $Gx = g$, $x \in X$. Since a maximization MDP can be formulated as a minimization linear program [8], it is possible to formulate SPAR as a two-stage stochastic linear program. However, to provide insight into the fundamental nature of SPAR in terms of the adversarial policies, we prove the following proposition directly, even though they follow from the two-stage stochastic LP formulation of SPAR. This proposition allows us to exploit the computational efficiencies of algorithms specific to two-stage stochastic programming and MDPs.

Proposition 1.

(a) *The expected adversarial recourse function $\mathcal{Q}(x)$ is a piecewise linear convex function.*

(b) *Let $d^*(\hat{x}, \xi^k)$ be the decision rule defining the optimal stationary policy for a given design $\hat{x}$ and scenario ξ^k, $k = 1, \ldots, K$. Then*

$$\sum_{k=1}^{K} q^k \alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d^*(\hat{x},\xi^k)}(\xi^k)]^{-1} \times [r_{d^*(\hat{x},\xi^k)}(\xi^k) - T_{d^*(\hat{x},\xi^k)}(\xi^k)x] \quad (3)$$

is a supporting hyperplane of $\mathcal{Q}(x)$ at $\hat{x}$.

Proof. For an MDP defined by x and ξ^k, let d be a deterministic decision rule defining a stationary policy with reward vector $r_d(\xi^k)$ and probability transition matrix $P_d(\xi^k)$. Define $T_d(\xi^k)$ as the $|S(\xi^k)| \times n_1$ designer-benefit matrix where element (s, i) of the matrix is the adversary impediment value for state-action pair $(s, d(s))$ linked to the first-stage design decision value x_i under scenario ξ^k, i.e. $[T_d(\xi^k)]_{si} = t_i(s, d(s), \xi^k)$. Let $v_d(x, \xi^k)$ be the value vector of the policy defined by d. Then by (2),

$$v_d(x, \xi^k) = [I - \lambda(\xi^k) P_d(\xi^k)]^{-1}[r_d(\xi^k) - T_d(\xi^k)x],$$

which is linear in x. Now,

$$v^*(x, \xi^k) = \max_{d \in D}\{v_d(x, \xi^k)\},$$

where D is the set of all deterministic decision rules. Therefore, $v^*(x, \xi^k)$ is clearly piecewise linear and convex. As a convex combination of these functions, $\alpha^{\mathrm{T}}(\xi^k)v^*(x, \xi^k)$ is also piecewise linear and convex. Similarly, $\mathcal{Q}(x) = \mathbb{E}_{\tilde{\xi}}[Q(x, \tilde{\xi})]$ is piecewise linear and convex, completing the proof of (a).

For a design x and scenario ξ^k, the expected reward of the resulting MDP under stationary policy $d^*(\hat{x}, \xi^k)$ is

$$\alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d^*(\hat{x},\xi^k)}(\xi^k)]^{-1} \times [r_{d^*(\hat{x},\xi^k)}(\xi^k) - T_{d^*(\hat{x},\xi^k)}(\xi^k)x] \quad (4)$$

by (2). However, since $d^*(\hat{x}, \xi^k)$ is not necessarily optimal for x and ξ^k,

$$\alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d^*(x,\xi^k)}(\xi^k)]^{-1} \times [r_{d^*(x,\xi^k)}(\xi^k) - T_{d^*(x,\xi^k)}(\xi^k)x] \quad (5)$$

lies above (4).

The fact that (3) is a supporting hyperplane of $\mathcal{Q}(x)$ is seen by observing that (3) and $\mathcal{Q}(x)$ are the expected values of (4) and (5) respectively and

$$\mathcal{Q}(\hat{x}) = \sum_{k=1}^{K} q^k \alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d^*(\hat{x},\xi^k)}(\xi^k)]^{-1} \times [r_{d^*(\hat{x},\xi^k)}(\xi^k) - T_{d^*(\hat{x},\xi^k)}(\xi^k)\hat{x}],$$

completing the proof of (b). □

By Proposition 1, each $\hat{x} \in \{x \in X \mid Gx = g\}$ induces a supporting hyperplane for $\mathcal{Q}(x)$ based on the stationary policies defined by the decision rules $d^*(\hat{x}, \xi^1), d^*(\hat{x}, \xi^2), \ldots, d^*(\hat{x}, \xi^K)$. Since each adversarial MDP has a finite number of states and actions, there are only a finite number of decision rules and hence stationary policies for each ξ^k. Let $D^*(\xi^k)$ be the set of stationary policies which are optimal for a first-stage decision under scenario ξ^k. Therefore, if (x^*, θ^*) is an optimal solution to the following MIP:

$$\min\ c^{\mathrm{T}}x + \theta$$

subject to

$$Gx = g \quad (6)$$

$$\theta \geqslant \sum_{k=1}^{K} q^k \alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d(\xi^k)}(\xi^k)]^{-1} \times [r_{d(\xi^k)}(\xi^k) - T_{d(\xi^k)}(\xi^k)x], \quad (7)$$

for all $d(\xi^k) \in D^*(\xi^k)$, $k = 1, \ldots, K$,

$$x \in X,\ \theta \in \mathbb{R},$$

then x^* is the optimal first-stage decision.

Applying these results, we present the SPAR algorithm based on a simplified version of the L-shaped method of Van Slyke and Wets [18]. It is clear that as long as the first-stage decision does not impact the states or actions then we have *relatively complete recourse*, which means that every solution to $\{x \in X \mid Gx = g\}$ results in feasible subproblems for every scenario. As a result, it is not necessary to consider feasibility cuts in the SPAR algorithm. In addition, since there are a finite number of stationary policies for each scenario, the SPAR algorithm terminates finitely with an optimal solution.

Step 0: Set $s = v = 0$.

Step 1: Set $v = v + 1$. Solve the master problem

$$\min c^{\mathrm{T}}x \ + \ \theta$$

subject to

$$Gx = g$$

$$E_l x + \theta \geqslant e_l, l = 1, \ldots, s, \tag{8}$$

$$x \in X, \theta \in \mathbb{R}.$$

Let x^v, θ^v be an optimal solution. If no constraint (8) is present, set $\theta^v = -\infty$ and it is not considered in the solution of x^v. E_l and e_l are defined in step 2.

Step 2: For all scenarios $k = 1, \ldots, K$, find the optimal stationary policy for the MDP with Bellman equations

$$v(s) = \max_{a \in nA_s(\xi^k)} \left\{ r(s, a, \xi^k) - \sum_{i=1}^{n_1} t_i(s, a, \xi^k) x_i + \lambda(\xi^k) \sum_{j \in S(\xi^k)} p(j \mid s, a, \xi^k)\ v(j) \right\}$$

for all $s \in S(\xi^k)$. Define

$$E_{s+1} = \sum_{k=1}^{K} q^k \alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d^*(x^v, \xi^k)}(\xi^k)]^{-1} \times [T_{d^*(x^v, \xi^k)}(\xi^k)]$$

and

$$e_{s+1} = \sum_{k=1}^{K} q^k \alpha^{\mathrm{T}}(\xi^k)[I - \lambda(\xi^k) P_{d^*(x^v, \xi^k)}(\xi^k)]^{-1} \times [r_{d^*(x^v, \xi^k)}(\xi^k)].$$

If $E_{s+1}x^v + \theta^v \geqslant e_{s+1}$, that is, the $s + 1$st cut of the form (8) does not cut off the current solution (x^v, θ^v), stop; (x^v, θ^v) is the optimal solution. Otherwise, set $s = s + 1$ and add the current cut to the constraint set (8) and return to step 1.

3. Computational results

The effectiveness of SPAR lies in its ability to reduce the number of adversarial subproblem MDPs that must be solved. To illustrate SPAR's effectiveness, we present a simplified computational example on randomly generated instances. In these instances, each element of the first-stage decision vector has a negative effect on the adversary's reward in the MDP subproblem. We restrict ourselves to the case where only the adversary impediment values, $t_i(s, a, \xi^k)$, vary for each scenario ξ^k, $k = 1, \ldots, K$ and $X = \mathbb{B}^{n_1}$. As stated previously the simplex method can be used to solve the MDP, however it is typically inefficient for solving single instances of MDPs. In these instances, we are solving several MDPs for which only the rewards are altered. This motivates the use of the dual-simplex method for efficiently finding the solution of one adversary's subproblem when provided with the basis from a previous solution in step 2 of the SPAR algorithm. We will compare the computational efficiencies of this methodology with standard policy iteration. The basic problem parameters are provided in Table 1.

In Table 2, we report the average results for 30 instances of each problem class in Table 1, where each replication involves randomly generated probability transitions, rewards/costs and adversary impediment values. We compare the subproblem and total solution times when the subproblems are solved with CPLEX 7.0 [7] using dual simplex and when the subproblems are solved as MDPs using standard policy iteration. In addition, we report the average number of subproblems solved in the computation of the optimal solution. The difference in the number of iterations between the dual-simplex method and policy iteration is due to alternate optimal dual solutions.

From Table 2, we see that the SPAR algorithm is a significant improvement over the solution of the MDP extensive form, which can be estimated by multiplying $|X|$ by K and the average subproblem solution time. In addition, for these problem instances policy iteration presents a significant improvement over dual simplex. However, in practice the probability transition matrices will come from a process with typically only a proportion of their states accessible from each state, and so they will not be completely dense. As a result, we investigate the effect of varying this density.

Tables 3 and 4 reveal the dependence of the results on the density of the probability transition matrix. In Tables 3 and 4, for each probability transition density we generated 30 instances and computed average solution times using policy iteration and dual simplex for

Table 1
Problem instance data

	First-stage			Adversary's MDP	
SPAR parameter set	Number of variables (n_1)	Number of feasible designs ($\|X\|$)	Number of scenarios ($\|\Xi\|$)	Number of states ($\|S\|$)	Number of actions ($\|A\|$)
1	30	1.07 E9	20	100	20
2	50	1.13 E15	20	150	30

Table 2
Comparative average solution times of SPAR (in seconds) for completely dense transition matrices

	Policy iteration			Dual simplex		
SPAR parameter set	Average number of iterations	Average total time	Average subproblem time	Average number iterations	Average total time	Average subproblem time
1	215	9.62	0.04	213	17.48	0.08
2	607	107.94	0.13	602	366.00	0.59

Table 3
The effect of probability transition density with SPAR parameter set 1

	Average computation time (seconds)					
Probability transition matrix density	0.15	0.175	0.2	0.225	0.25	0.275
SPAR (dual simplex)	8.95	9.63	10.07	10.99	11.68	11.67
SPAR (policy iteration)	10.68	10.52	10.25	10.29	10.05	9.66
Ratio (dual simplex/policy iteration)	0.84	0.92	0.98	1.07	1.16	1.21

Table 4
The effect of probability transition density on SPAR parameter set 2

	Average computation time (seconds)					
Probability transition matrix density	0.025	0.05	0.075	0.1	0.125	0.15
SPAR (dual simplex)	104.94	135.79	148.23	176.56	189.62	196.99
SPAR (policy iteration)	188.50	157.02	130.61	129.42	122.92	120.05
Ratio (dual simplex/policy iteration)	0.56	0.86	1.13	1.36	1.54	1.64

the adversary subproblems. For relatively low probability transition densities and therefore sparse probability transition matrices dual simplex was more efficient. However, for problems with densities at least .225 for SPAR parameter set 1 and at least .075 for SPAR parameter set 2 policy iteration appears to be the preferred technique. In addition, we see that policy iteration is faster as the density increases, not just relatively faster. Again, since only the rewards are varying for each scenario instance, we utilized the dual-simplex method to update the solution of an adversary subproblem efficiently.

Even with the computational benefit of solving the related MDPs as LPs and using the dual simplex method, standard MDP methods are still more computationally effective in the problems with higher

Table 5
Average solution time (seconds) of adversary LPs for SPAR parameter set 2

	Average computation time (seconds)					
Probability transition matrix density	0.025	0.05	0.075	0.1	0.125	0.15
Initial LP	0.466	0.576	0.723	0.873	0.979	1.099
Average subsequent LPs	0.141	0.182	0.224	0.261	0.286	0.303

densities. However, we see in Table 5 that the majority of the LP solution time is spent solving the LP of the first adversary problem investigated and each subsequent solution is quickly updated. Therefore, our results suggest that the most effective solution technique for SPAR may be a hybrid technique which selects the solution method (dual simplex, policy iteration, or a combination) depending on the density (and other parameters) of the resulting subproblem.

Acknowledgments

We would like to thank the area editor and an anonymous referee for their valuable input on improving an earlier version of this paper. We would also like to thank Oguzhan Alagoz for his helpful comments and insight. Andrew Schaefer was supported by NSF grant DMI-0217190.

References

[1] R. Bellman, Dynamic Programming, Princeton University Press, Princeton, NJ, 1957.
[2] J.F. Benders, Partitioning procedures for solving mixed variables programming problems, Numerische Math. 4 (1962) 238–252.
[3] D.P. Bertsekas, J.N. Tsitsiklis, Analysis of stochastic shortest path problems, Math. Oper. Res. 16 (1991) 580–595.
[4] J.R. Birge, F.V. Louveaux, Introduction to Stochastic Programming, Springer, New York, 1997.
[5] W.L. Cooper, T. Homem-de-Mello, A class of hybrid methods for revenue management, submitted for publication. available from http://users.iems.northwestern.edu/~tito/pubs/rmso_submitted.pdf.
[6] K.J. Cormicon, D.P. Morton, K.R. Wood, Stochastic network interdiction, Oper. Res. 46 (2) (1998) 184–197.
[7] ILOG Corporation, CPLEX 7.0 documentation, 2001.
[8] F. D'Epenoux, A probabilistic production and inventory problem, Management Sci. 10 (1) (1963) 98–108.
[9] J. Filar, K. Vrieze, Competitive Markov Decision Processes, Springer, New York, 1997.
[10] R. Hemmecke, R. Schultz, D. Woodruff, Interdicting stochastic networks with binary interdiction effort, in: Woodruff (Ed.), Network Interdiction and Stochastic Integer Programming, Kluwer Academic, 2002, pp. 69–83.
[11] R. Howard, Dynamic Programming and Markov Processes, MIT Press, Cambridge, 1960.
[12] G.J. Koehler, A case for relaxation methods in large scale linear programming, in: G. Guardabassi, A. Locatelli (Eds.), Large Scale Systems Theory and Applications, IFAC, Pittsburgh, 1976, pp. 293–302.
[13] T.E. Morton, On the asymptotic convergence rate of cost differences for Markovian decision processes, Oper. Res. 19 (1971) 244–248.
[14] R.A. Murhpy, An approximate algorithm for a weapon target assignment stochastic program, in: P.M. Pardalos (Ed.), Approximation and Complexity in Numerical Optimization: Continuous and discrete problems, Kluwer, 1999.
[15] F. Pan, W. Charlton, D.P. Morton, A stochastic program for interdicting smuggled nuclear material, in: X. Woodruff (Ed.), Network Interdiction and Stochastic Integer Programming, Kluwer Academic, 2002, pp. 1–19.
[16] M.L. Puterman, Markov Decision Processes, Wiley, New York, NY, 1994.
[17] L.S. Shapley, Stochastic games, Proc. National Acad. Sci. 39 (1953) 1095–1100.
[18] R. Van Slyke, R.J-B. Wets, L-shaped linear programs with applications to optimal control and stochastic programming, SIAM J. Appl. Math. 17 (1969) 638–663.