MANAGEMENT SCIENCE
Vol. 50, No. 3, March 2004, pp. 405–418
ISSN 0025-1909 | EISSN 1526-5501 | 04 | 5003 | 0405

informs

DOI 10.1287/mnsc.1030.0138

Multistage Monte Carlo Method for Solving Influence Diagrams Using Local Computation

John M. Charnes, Prakash P. Shenoy
School of Business, University of Kansas, 1300 Sunnyside Avenue, Summerfield Hall, Lawrence, Kansas 66045-7585
{jcharnes@ku.edu, pshenoy@ku.edu}

The main goal of this paper is to describe a new multistage Monte Carlo (MMC) simulation method for solving influence diagrams using local computation. Global methods have been proposed by others that sample from the joint probability distribution of all the variables in the influence diagram. However, for influence diagrams having many variables, the state space of all variables grows exponentially, and the sample sizes required for good estimates may be too large to be practical. In this paper, we develop a MMC method, which samples only a small set of chance variables for each decision node in the influence diagram. MMC is akin to methods developed for exact solution of influence diagrams in that we limit the number of chance variables sampled at any time. Because influence diagrams model each chance variable with a conditional probability distribution, the MMC method lends itself well to influence diagram representations.

Key words: decision analysis; approximations; sequential; simulation; applications; Monte Carlo methods; local computation
History: Accepted by James E. Smith; received August 15, 1999. This paper was with the authors 2 years and 9 months for 6 revisions.

1. Introduction

The main goal of this paper is to propose a new multistage Monte Carlo (MMC) method for solving influence diagrams using *local computation*. Influence diagrams are a compact representation of Bayesian decision problems that were initially proposed as a front end for decision trees (Howard and Matheson 1984). Later, Olmsted (1983) and Shachter (1986) devised a method for solving influence diagrams directly, i.e., without first transforming them to decision trees. The direct influence diagram solution technique uses local computation to obtain the conditionals and an optimal strategy. Smith et al. (1993) have proposed modifications to the influence diagram technique for representing and solving asymmetric decision problems.

Most of the research on representing and solving decision problems assume that all chance and decision variables have discrete state spaces. For problems in which some of the decision and/or chance variables are continuous, several approximation methods have been proposed. The traditional approach is to partition the state space of each continuous variable into a few discrete states (e.g., Miller and Rice 1983, Keefer 1994). A related approach is to summarize continuous distributions by their first few moments, summarize continuous utility functions by their first few derivatives, and then use either the moments, the derivatives, or both to obtain a solution (Howard 1971, Smith 1993). Another approach is to deal directly with continuous variables without discretizations. For example, Shachter and Kenley (1989) have studied influence diagram methodology for decision problems in which the probability model is multivariate Gaussian, and Poland (1994) has developed influence diagrams that use Gaussian mixtures to approximate arbitrary continuous distributions.

For the general case of a decision problem consisting of a mixture of discrete and continuous variables, Jenzarli (1995) has investigated the use of Gibbs sampling for solving such decision problems using local computation. Gibbs sampling is one member of a class of techniques called Markov chain Monte Carlo that draw *dependent* samples from a distribution that in the long run approaches the target joint distribution (e.g., see Gilks et al. 1996). Bielza et al. (1999) explore the problem of using Markov chain Monte Carlo methods to solve a *single-stage* decision problem with continuous decision and chance nodes.

The MMC sampling technique proposed here draws *independent* and identically distributed observations to solve multiple-stage decision problems. Monte Carlo methods that have been proposed in this spirit sample from the entire distribution (e.g., see Hertz 1964, and Shachter and Peot 1990 for Bayesian networks, which are influence diagrams without decision and value nodes). However, when the number of variables is large, the combined state space of all variables is exponentially large, and the sample size required for precise estimates is too large to

be practical. The MMC method generates samples at each stage from a small set of variables in the influence diagram. We use methods developed for exact solution of influence diagrams to limit the number of variables sampled at any time. Because influence diagrams model each chance variable with a conditional probability distribution, the MMC solution method lends itself well to influence diagram representations.

In §2, we provide a statement of the oil wildcatter with secondary recovery (OWSR) problem as an example. In §3, we describe an influence diagram representation of the OWSR problem using the Smith et al. (1993) distribution tree technique. In §4, we describe our MMC method for solving influence diagrams using local computation and illustrate it using the OWSR problem. In §5, we describe an application of our method to pricing Bermudan put options. Finally, in §6, we conclude with a summary and some issues for further research. The online appendix (mansci.pubs.informs.org/ecompanion.html) elaborates on the statistical properties of the MMC method.

2. Oil Wildcatter with Secondary Recovery (OWSR) Problem

The OWSR problem is an adaptation of a problem described by Raiffa (1968). An oil wildcatter must decide whether or not to drill a well at a particular location. He is uncertain whether the well will be dry, wet (some oil), or soaking (lots of oil). The cost of drilling is \$70,000. The expected net revenue (exclusive of drilling cost) resulting from primary recovery of oil from a dry well is \$0, from a wet well is \$120,000 and from a soaking well is \$270,000. The wildcatter's subjective probabilities for the state of the well are 0.5 for dry, 0.3 for wet, and 0.2 for soaking.

Before making the decision whether or not to drill, the wildcatter can have a seismic test done to investigate the underground structure of the drill site. The cost of the seismic test is \$10,000 to classify the site as having either no structure (*ns*), open structure (*os*), or closed structure (*cs*). The underground structure is related to the amount of oil as follows. If the well is dry, the conditional probabilities of *ns*, *os*, and *cs* are 0.6, 0.3, and 0.1, respectively. If the well is wet, the corresponding probabilities are 0.3, 0.4, and 0.3. Finally, if the well is soaking, the corresponding probabilities are 0.1, 0.4, and 0.5.

After drilling, striking oil, and extracting an optimal amount using primary recovery techniques, the wildcatter has the option of extracting more oil using secondary recovery techniques at an additional cost of \$20,000. Secondary recovery will result in no recovery (*nr*) with associated revenues of \$0, low recovery (*lr*) with associated revenues of \$30,000 or high recovery (*hr*) with associated revenue of \$50,000. The amount of secondary recovery depends on the state of the well. If the well is wet, the conditional probabilities of *nr*, *lr*, and *hr* are 0.5, 0.4, and 0.1, respectively. If the well is soaking, the corresponding probabilities are 0.3, 0.5, and 0.2.

3. Influence Diagram Representation

A mathematical representation of a Bayesian decision problem can be broken into four parts: (1) alternatives—the set of alternatives available to the decision maker, (2) uncertainty model—a probability model of the uncertainties faced by the decision maker, (3) preferences for outcomes—a utility function model of the preferences of the decision maker for all possible outcomes, and (4) information constraints—a set of restrictions on the information available to the decision maker each time he or she must make a decision.

An influence diagram representation of a decision problem is specified at two levels: graphical and numerical. At the graphical level, we have an acyclic directed graph with three different types of nodes—chance, decision, and utility—and two different types of directed edges—domain and pure information. At the numerical level, we have a distribution tree for each node in the graph. Figure 1 shows an influence diagram representation of the OWSR problem at the graphical level. Figures 2 and 3 show the influence diagram representation at the numerical level.

In Figure 1, T (seismic test), D (drill), and S (secondary recovery) are rectangular decision nodes representing decision variables; R (seismic test results), O (amount of oil), and SR (secondary oil recovered), are circular chance nodes representing chance variables; and v_1, v_2, and v_3 are diamond-shaped value nodes representing additive factors of the joint utility function.

The solid arrows pointing to a decision variable indicate the *domain* of the conditional for the decision variable, which is the subset of variables included in the conditional. In the OWSR problem, the domain for the conditional for T is $\{T\}$, which we write as $\mathrm{Dom}(\chi_T) = \{T\}$, where χ_T denotes the conditional for T. Also, $\mathrm{Dom}(\chi_D) = \{D\}$ and $\mathrm{Dom}(\chi_S) = \{D, O, S\}$.

Figure 1 **Influence Diagram Representation of the OWSR Problem at the Graphical Level**

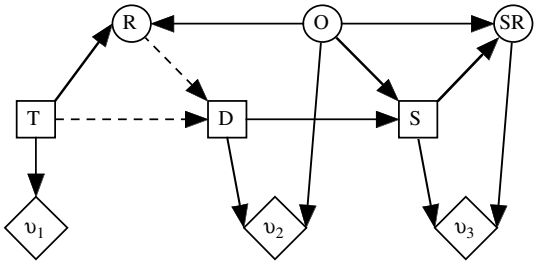
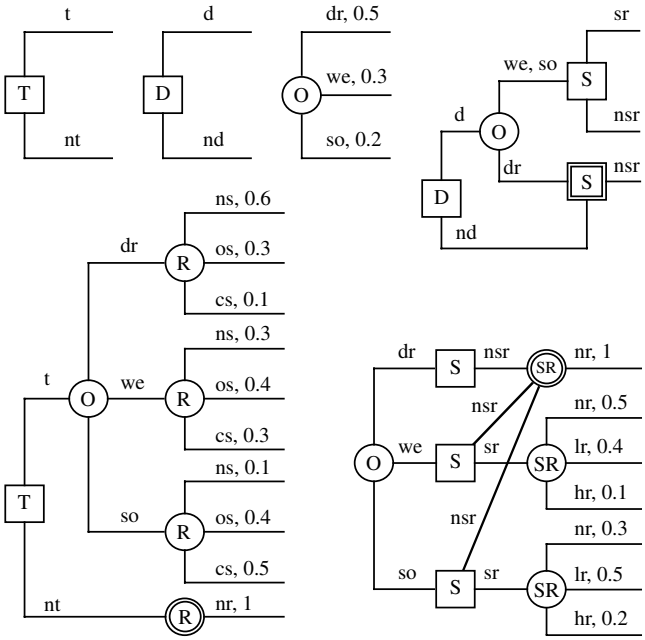
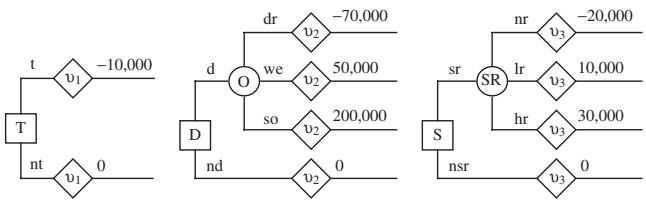


Figure 2 Conditionals for the Decision and Chance Nodes in the OWSR Problem



A conditional for a decision variable constrains the set of possible alternatives available to the decision maker at that variable. Because there are no constraints on the choices at the decision variables T and D, $\mathrm{Dom}(\chi_T) = \{T\}$ and $\mathrm{Dom}(\chi_D) = \{D\}$. Because the choices available at S depend on the choice made at D and the realized value of O, $\mathrm{Dom}(\chi_S) = \{D, O, S\}$. In general, the domain of the variable X consists of the set of variables at the tails of the solid arrows pointing to X and X itself. Solid arrows pointing to decision nodes are also interpreted as information constraints in the sense that if there is a solid arrow from node X (either chance or decision) to decision node A, then the true state of X will be known to the decision maker at the time he or she has to choose an alternative at A. Dashed arrows pointing to decision nodes indicate information constraints only. A dashed arrow from a node X to a decision node A means that the decision maker knows the true value of X at the time he or she has to make a choice at A. This is in contrast to a solid arrow, which indicates both a conditional and an information constraint.

Figure 3 Additive Factors of the Utility Function in the OWSR Problem



The influence diagram literature usually assumes a "no-forgetting" condition, which specifies that if there is a directed path from a node X (chance or decision) to decision node A via other decision nodes only, then this implies that the decision maker knows the value of X at the time he or she has to make a choice at A. Optionally, an implied constraint can be represented in the influence diagram as a dashed arrow from X to A, but because such constraints are easily deduced, we omit them from our graphical representations. In Figure 1, we can deduce dashed arrows from R to S, and from T to S. Thus, at T, the wildcatter has observed nothing; at D, the wildcatter has observed the true state of R but does not yet know either the true state of O or the true state of SR; while at S, the wildcatter has observed the true states of O and R, but not SR.

The solid arrows pointing to chance nodes in Figure 1 specify the domains of the conditionals for the chance variables in the same sense as for decision variables. In the OWSR problem, we have $\mathrm{Dom}(\chi_O) = \{O\}$, $\mathrm{Dom}(\chi_R) = \{T, O, R\}$, and $\mathrm{Dom}(\chi_{SR}) = \{O, S, SR\}$. These come from the following conditionals: $P(O)$, $P(R \mid T, O)$, and $P(SR \mid O, S)$.

The solid arrows pointing to value nodes in Figure 1 indicate the domain of the corresponding utility function in the sense that the domain of the utility function at a value node is the set of variables at the tails of the arrows pointing to it. In the OWSR problem, $\mathrm{Dom}(\upsilon_1) = \{T\}$, $\mathrm{Dom}(\upsilon_2) = \{D, O\}$, and $\mathrm{Dom}(\upsilon_3) = \{S, SR\}$.

At the numerical level, we specify the details of the conditional for each decision node and each chance node and the details of the utility functions. In the OWSR problem, the state spaces for the variables are as follows: $\Omega_{T_1} = \{t, nt\}$ (representing seismic test, and no seismic test, respectively), $\Omega_R = \{ns, os, cs, nr\}$ (no structure, open structure, closed structure, no results), $\Omega_D = \{d, nd\}$ (drill, not drill), $\Omega_O = \{dr, we, so\}$ (dry, wet, soaking), $\Omega_S = \{sr, nsr\}$ (attempt secondary recovery, no secondary recovery), and $\Omega_{SR} = \{nr, lr, hr\}$ (no recovery, low recovery, high recovery).

The conditionals for the decision and chance nodes shown in Figure 2 use Smith et al. (1993) distribution trees notation. A distribution tree consists of paths called conditioning scenarios that lead to atomic distributions. For decision nodes, the atomic distributions are the sets of alternatives available to the decision maker under each conditioning scenario. For chance nodes, the atomic distributions are probability distributions conditioned on the paths leading to the atomic distributions.

In Figure 2, the distribution trees for T, D, and O have no conditioning scenarios and one atomic distribution, Ω_T, Ω_D, and $P(O)$, respectively. The distribution tree for S has two atomic distributions $\{sr, nsr\}$

and {*nsr*} depending on the conditioning scenarios. An atomic distribution with only one alternative is called *degenerate* and is shown using double borders. The distribution tree for R has four atomic distributions, $P(R \mid T, O)$, one of which is degenerate. The distribution tree for SR has three atomic distributions, $P(SR \mid O, S)$, one of which is degenerate. Finally, Figure 3 shows the numerical details of the utility functions.

Unlike a decision tree representation, which requires calculation of the preposterior and posterior probabilities, no preprocessing is required for an influence diagram representation. Thus, an influence diagram may be more compact than a decision tree because it does not attempt to depict all possible scenarios explicitly.

4. Multistage Monte Carlo (MMC) Method

The exact method for solving influence diagrams is similar to the method for solving decision trees. In both methods, we delete decision and chance nodes in a sequence determined by the information constraints. In the OWSR problem, the only deletion sequence allowed by the information constraints is $SR \hookrightarrow S \hookrightarrow O \hookrightarrow D \hookrightarrow R \hookrightarrow T$. The major difference between the two solution methods is the computation of the posterior and preposterior probabilities. With decision trees, these probabilities are computed from the joint probability distribution as a preliminary step in constructing the representation. With influence diagrams, these probabilities are computed using local computation by reversing arcs during the solution phase.

Our method is similar to the exact method for solving influence diagrams for the case of an additive factorization of the utility function (Tatmar and Shachter 1990, Shenoy 1992). This includes solving the influence diagram representation by reduction of decision and chance variables in the reverse order of the sequence determined by the information constraints. In each stage, the domain of the decision function is also the same as in the exact method.

The novelty of our method is the computation of the requisite conditionals for chance variables. In the exact methods, the conditionals are determined by arc reversals (Olmsted 1983, Shachter 1986) or by fusion with respect to a chance variable (Shenoy 1992). In the MMC method, we achieve the same computation by sampling from a partial joint distribution and then estimating the conditional expectation.

In our MMC method, we do not compute the posterior and preposterior distributions explicitly. Instead, we work with an appropriate partial joint distribution and then compute a conditional expectation. In this sense, our method is similar to the game tree solution of decision problems (Shenoy 1998). In game trees, one can represent any decision problem without preprocessing of probabilities. Information constraints are represented using information sets as in von Neumann and Morgenstern's (1944) extensive form games. The only constraints on the ordering of variables in a game tree is that if variable X (chance or decision) is known when decision A is to be made, then X must precede A.

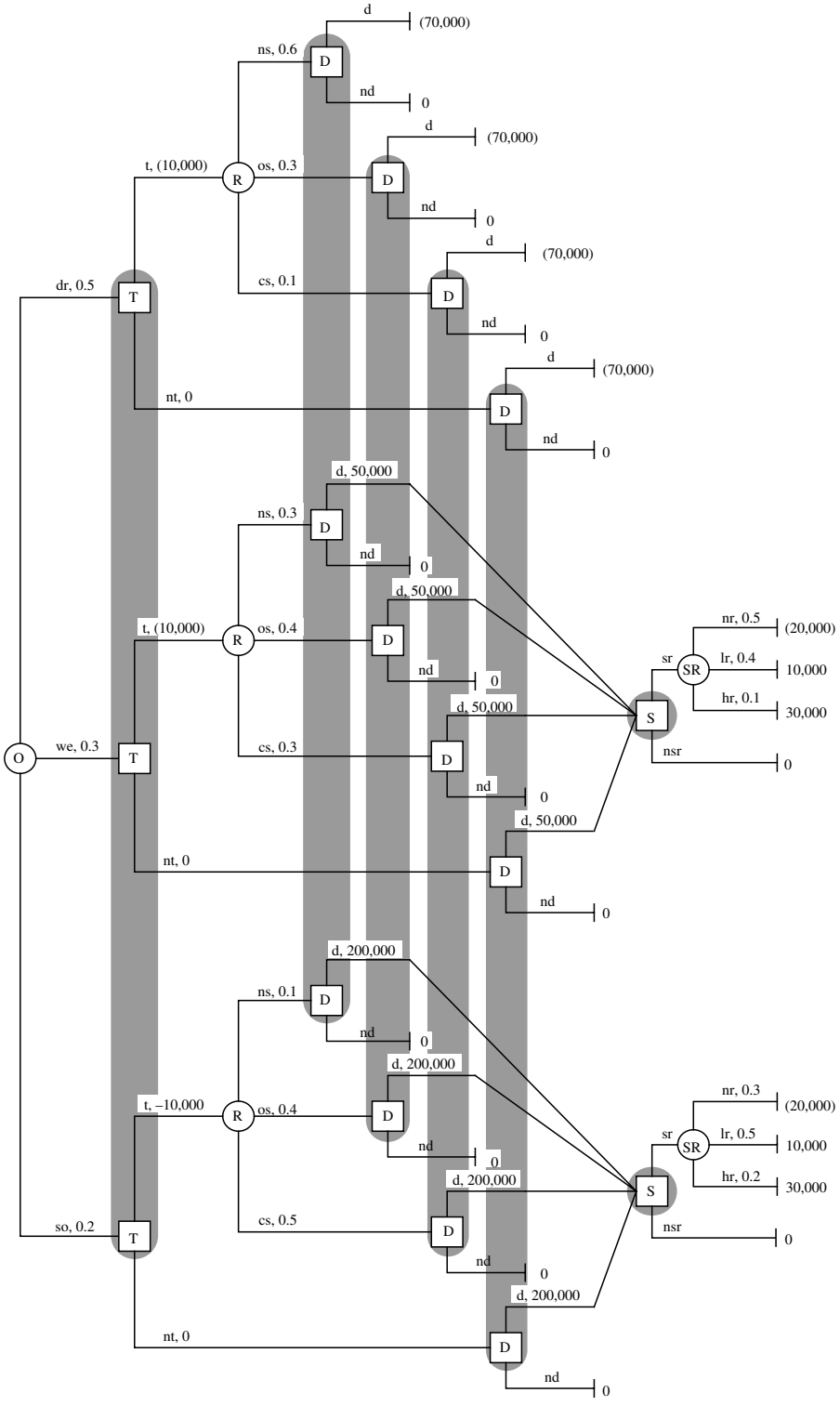
Figure 4 depicts a game tree representation of the OWSR problem. This depiction uses the same probability model specified in the statement of the problem and, thus, does not require any preprocessing of probabilities. In the OWSR problem, we have seven information sets (designated by shaded regions in Figure 4) that partition the set of all decision nodes in the game tree. For example, the three T nodes are in the same information set indicating that the oil wildcatter does not know the true amount of oil when deciding whether to do a seismic test.

Solving a game tree is similar to solving a decision tree by pruning nodes using the rollback procedure. Game tree chance nodes are pruned by averaging the utilities. Information sets are pruned by maximizing conditional expectation, i.e., by first computing the conditional probability distribution for each node in an information set conditioned on the fact that we are in the information set, and then choosing an alternative with the maximum conditional expected utility. The conditional probability distribution for a decision node is computed by multiplying all probabilities on the path from the root to the node. See Shenoy (1998) for a detailed description of this procedure.

In the game tree representation of the OWSR problem (Figure 4), consider the last decision S, whether to attempt secondary recovery. Notice that the subtrees for this decision do not depend on either R or T, but do depend on O and D. The independence of these subtrees of R and T is reflected in the game tree representation in Figure 4 using "coalescence" (Olmsted 1983), where repeating subtrees are shown only once. Reducing the top S node (on the path $O = we$) results in a utility of 0, and reducing the bottom S node (on the path $O = so$) results in a utility of 5,000.

Now consider the three D nodes on the paths $R = ns$ that are in the same information set. This set can only be reached if we decide to perform the test. The probability of reaching this information set is $(0.5 \times 0.6) + (0.3 \times 0.3) + (0.2 \times 0.1) = 0.41$. The conditional probability of being at the top D node in this information set (on the path $O = dr$) given that one is in this information set is $(0.5 \times 0.6)/0.41 = 0.732$. Similarly, the conditional probabilities of being in the middle D node (on the path $O = we$) and bottom D node (on the path $O = so$) are 0.220 and 0.049, respectively. Thus, to

Figure 4 Game Tree Representation of the OWSR Problem



reduce this information set, we compute a conditional expectation: $D = d$ results in a conditional expectation of $0.732(-70,000) + 0.220(50,000) + 0.049(205,000) = -30,195$, and $D = nd$ results in 0. Thus, the optimal choice is $D = nd$ once we know that $R = ns$.

The MMC method works in a similar manner to the game tree solution. However, the MMC method relies on sampling-based estimates of the conditional expectations to specify decision functions in stages. Sampling lets us obtain approximate solutions to

problems that cannot be solved exactly using the influence diagram method.

4.1. Overview

We begin each stage of the MMC method by selecting what we call the *current maximal decision variable*, denoted by M and defined formally in the next section. Then, we determine a decision function for M by iteratively sampling the state of each chance variable in a subset of relevant variables, in an order we call a *sampling sequence*, to estimate the conditional expected utility. After we have completed enough iterations to achieve the desired precision in our estimates, we determine a decision function for M based on the estimated maximum conditional expected utility.

To prepare for the next stage, we modify the influence diagram. We do this by converting M to a chance variable whose conditional is the decision function we have just computed, and replacing a set of utility functions by a single utility function composed from the estimates of conditional expected utilities just obtained. The next stage begins with the selection of a new current maximal decision variable. The main idea behind the algorithm is to determine a decision function for each decision variable in turn by sampling a small subset of relevant chance variables using the conditionals specified in the influence diagram representation. The formal algorithm is presented in Table 1.

4.2. Definitions

In this section, we define the notation used in the algorithm presented in Table 1.

M**—Current Maximal Decision Variable.** The information constraints specified in the ID representation impose a partial order, $<$, on the set of all decision and chance variables. We designate as *current* a decision variable that is maximal with respect to the partial order $<$. If there is more than one such decision variable, we pick one using a heuristic such as "one step look ahead" (Olmsted 1983). At the outset of the solution of the OWSR problem, the partial order obtained is $T < R < D < O < S < SR$ (which, in this case, is a complete order). The current maximal decision variable for stage 1 is $M = S$.

$RD(M)$**—Relevant Domain of the Decision Function for** M**.** The relevant domain of the decision function for M is a small set of decision and chance variables sufficient to make the decision at M. See Nielsen and Jensen (1999), who describe a complicated procedure for specifying $RD(M)$ based on several rules. Notice that inclusion of irrelevant variables in the relevant domain does not affect the computation of an optimal strategy, but may reduce the efficiency of finding one. One easy rule for determining $RD(M)$ is as follows. Let $U(M)$ denote the set of *unobserved* chance variables with respect to decision M and the partial order, $<$. First, we find a smallest subset of decision and chance variables that d-separates $M \cup U(M)$ from the rest of the variables. Then, $RD(M)$ is equal to the union of this smallest subset and $\{M\}$. This rule may not always result in the smallest subset. In stage 1 of the OWSR problem, for the current maximal decision variable $M = S$, the relevant domain $RD(S) = \{D, O, S\}$.

$RU(M)$**—Relevant Factors of the Joint Utility Function.** Let Ψ denote the set of all utility factors in the influence diagram. Let Γ denote the set of all chance variables in the influence diagram, Δ denote the set of all decision variables, and $\prec$ denote a partial order on $\Gamma \cup \Delta$ defined as follows. Suppose $X_1, X_2 \in \Gamma \cup \Delta$. Then, $X_1 \prec X_2$ if, and only if, there is a directed path from X_1 to X_2 using solid arrows in the influence diagram representation. Because the influence diagram is acyclic, it follows that the binary relation $\prec$ is a partial order. If $X_1 \prec X_2$, we say X_1 is a *predecessor* of X_2 and X_2 is a *successor* of X_1. Let $S(X)$ denote the set of all successors of variable X. In the OWSR problem, the partial order is $\{T, O\} \prec R, \{D, O\} \prec S \prec SR$. Notice that the partial order $\prec$ is different from the partial order $<$ defined earlier.

Consider all utility factors that have M or its successors in their domains and let $H(M)$ denote the union of their domains, i.e., $H(M) = \bigcup\{\text{Dom}(v) \mid v \in \Psi$ such

Table 1 MMC Algorithm for Solving Influence Diagrams Using Local Computation

Step 1. Choose a current maximal decision variable M for the sampling stage. If there are no remaining decision variables, then stop.

Step 2. For each possible state of the variables in $F(M)$, sample for the chance variables in $SV(M)$ using their conditional distributions in a sequence determined by the partial order $\prec$. After each iteration, store the vector of realizations of $RD(M) \cup DRU(M)$ and the associated sum of the utility values of functions in $RU(M)$. After each 50 iterations, for each state of the variables in $RD(M)$, find the mean utility averaged across realizations of all other variables in $DRU(M) \backslash RD(M)$, and compute the standard error of the mean. Stop sampling when the standard error is less than some target, or the number of iterations has reached a prespecified limit, N.

Step 3. Specify an optimal decision function whose domain is $RD(M)$ based on maximizing average utility computed in Step 2 for each state of the variables in $RD(M)$.

Step 4. Make M a chance variable whose conditional distribution is given by the decision function specified in Step 3.

Step 5. Replace the set of utility functions $RU(M)$ by the utility function whose domain is $RD(M) \backslash \{M\}$, and whose values are the corresponding maximum utility values found in Step 3. Go to Step 1.

Note. Refer to §4.2 for definitions and further explanation.

that $\text{Dom}(v) \cap (\{M\} \cup S(M)) \neq \varnothing\}$. It follows from the method proposed by Tatman and Shachter (1990) that only utility factors whose domains include variables in $\{M\} \cup [U(M) \cap H(M)]$ are relevant for the determination of the decision function for M, i.e., $RU(M) = \{v \in \Psi \mid \text{Dom}(v) \cap [\{M\} \cup (U(M) \cap H(M))] \neq \varnothing\}$. In stage 1 of the OWSR problem, $U(S) = \{SR\}$, $H(S) = \{S, SR\}$, and $RU(S) = \{v_3\}$.

$DRU(M)$**—Domain of the Relevant Utility Factors.** Let $DRU(M)$ denote the union of the domains of the relevant utility functions, i.e., $DRU(M) = \bigcup\{\text{Dom}(v) \mid v \in RU(M)\}$. In stage 1 of the OWSR problem, because $RU(S) = v_3$, $DRU(S) = \{S, SR\}$.

$SV(M)$**—The Set of Sampled Chance Variables.** Isolation of this set is unique to our method because we do not compute the conditionals for unobserved chance variables (as is done using arc reversals in the exact methods). Our goal is to sample a subset of unobserved chance variables using the conditionals specified in the ID representation. Clearly, we need to sample all the chance variables in $DRU(M)$. We must also sample all unobserved chance variables that are predecessors of the unobserved chance variables in $DRU(M)$, because predecessor distributions will influence the distributions of the chance variables in $DRU(M)$. Let $P(M)$ denote the set of unobserved chance variables that are predecessors of the chance variables in $DRU(M)$, i.e., $P(M) = \{C \in U(M) \mid C \prec W$ for some $W \in \Gamma \cap DRU(M)\} \backslash DRU(M)$. We can ignore the unobserved chance variables that are successors of the chance variables in $DRU(M)$ because these have no influence on the chance variables in $DRU(M)$.

Also, because we sample directly from the conditionals specified in the influence diagram representation, we also need to sample all chance variables that lie "between" the chance variables in $(\Gamma \cap DRU(M)) \cup P(M)$. Let $B(M)$ denote the variables that are *between* the variables in $(\Gamma \cap DRU(M)) \cup P(M)$, i.e.,

$$\begin{aligned} B(M) = \{&V \in \Gamma \mid V \prec W \text{ for some} \\ &W \in [\Gamma \cap DRU(M)] \cup P(M)\} \\ \backslash \{&V \in \Gamma \mid V \prec W \text{ for all} \\ &W \in [\Gamma \cap DRU(M)] \cup P(M)\} \\ \backslash \{&[\Gamma \cap DRU(M)] \cup P(M)\}. \end{aligned}$$

In stage 1 of the OWSR problem, $P(M) = \varnothing$ and $B(M) = \varnothing$.

The set of chance variables that we sample to determine a decision function for M is given by $SV(M) = (\Gamma \cap DRU(M)) \cup P(M) \cup B(M)$. We avoid doing arc reversals by sampling from the conditional joint distribution to estimate the conditional expectation (as in game trees). In stage 1 of the OWSR problem, $SV(M) = \{SR\}$.

$F(M)$**—The Variables Whose States Are Fixed.** To enable us to sample the variables in $SV(M)$ using the conditionals specified in the ID representation, we must fix the states of the variables that are in the domains of the conditionals of the variables in $SV(M)$ if they are not already included. Let $F(M)$ denote this set of variables. Formally, $F(M) = \bigcup\{\text{Dom}(\chi) \mid \chi$ is a conditional for a variable in $SV(M)\} \cup \{M\} \backslash SV(M)$. Notice that M is always included in $F(M)$. In stage 1 of the OWSR problem, $SV(S) = \{SR\}$ and $F(S) = \{D, O, S\}$. Thus, in stage 1, we solve the subproblem shown in Figure 5. The chance variables in $F(M)$ whose states are fixed are shown using thickly bordered circles.

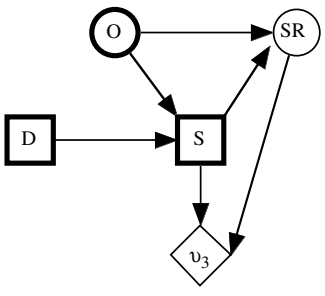
Sampling Sequence. In our algorithm, we sample the variables in $SV(M)$ such that if $V_1 \prec V_2$, then V_1 precedes V_2 in the sampling sequence. The sampling sequence is motivated by the fact that in an influence diagram there is a conditional for each chance and decision variable. In a given sampling sequence, all direct predecessors of V_2 will have been sampled (or fixed) when it is time to sample V_2 and, thus, the conditional distribution for V_2 can be used at that point in the sequence.

4.3. Number of Iterations

For a given stage, working our way through the sampling sequence for the variables in $SV(M)$ constitutes one *iteration*. The required number of iterations at each stage is determined by the desired precision and level of confidence of an approximately optimal strategy. In general, the greater precision and higher level of confidence we desire in the result, the larger the number of iterations required. With MMC, we control the precision of the estimates by specifying the maximum standard error of the estimates at each stage.

The question of how best to make statistical inferences via computer sampling has received much attention in the simulation output analysis literature. Variance reduction techniques (e.g., see Law and Kelton 2000) may be more extensively employed in future refinements of the MMC method to reduce the required number of iterations. In §5, we use an antithetic variates variance reduction procedure to

Figure 5 OWSR Sub-ID Solved in Stage 1



obtain the results presented there. The ranking and selection, or multiple comparisons methods described by Goldsman and Nelson (1998) might also be applied to further reduce the required number of iterations. Quasi-Monte Carlo sampling (Niederrreiter 1992) could provide a third approach to reducing the computation required for solving influence diagrams using simulation. However, in this paper, we focus on a straightforward sampling technique to convey the logic of our method.

We begin by defining what we mean by an *approximately optimal strategy*. Each well-defined decision problem has a maximum expected utility associated with at least one optimal strategy. Suppose our goal is to find a strategy whose estimated expected utility is within ϵ of the maximum expected utility. For example, in the OWSR problem, we wish to find a strategy whose expected profit is within $\epsilon = \$1{,}000$ of the maximum expected profit. With Monte Carlo methods, there is no guarantee that we will be able to find such a strategy, so we settle for doing so with a $100(1-\alpha)\%$ level of confidence. For example, we may choose to run enough iterations to become 90% confident that the strategy identified by the MMC method has an expected utility that is within $\epsilon = \$1{,}000$ of the maximum. We call such a strategy an (ϵ, α) *approximate strategy*.

We also wish to find an (ϵ, α) approximate strategy in a reasonable amount of time, which depends upon the number of iterations run. In practice, we prefer to do just enough sampling at each stage to achieve the desired precision, but we also specify an upper limit, N, on the number of iterations to bound the execution time.

The number of estimates required for each stage is determined after we reduce, in two ways, the amount of sampling required. The first sampling reduction strategy that is already built into our algorithm is not to sample for variables in $F(M)$. Instead, we sample only the variables in $SV(M)$ for all possible states of the variables in $F(M)$. The second sampling reduction strategy is to exploit some of the asymmetric features of the influence diagram representation. First, if a particular combination of states of variables in $F(M)$ is ruled out by the conditionals, then no sampling of the variables in $SV(M)$ is required for this combination. Second, for a given set of states of variables in $F(M)$, if all chance variables in $SV(M)$ are degenerate, then no sampling is necessary. Third, if the utility factors in $RU(M)$ are independent of the chance variables in $SV(M)$ for a particular combination of states of variables in $F(M)$, then again no sampling is necessary.

Let W denote the estimated mean utility at the final stage of sampling, which is a function of the estimated mean utilities at all stages. We wish to place a bound on $\mathrm{Var}(W)$, the variance of W. At each stage, we can reduce the conditional variance of the estimates (conditioned on the estimates from previous stages) to an arbitrarily small value by running the simulation for a sufficiently large number of iterations. Let σ_j^2 denote the upper bound on the conditional variances (squared standard errors) of utility estimates associated with chance variables (and conditioned on estimates from previous stages) obtained in stage j, which we control through specification of the number of iterations.

Using a property of conditional variance, one can show that the unconditional variance of the average utilities associated with chance variables in stage j is less than or equal to $\sum_{i=1}^{j} \sigma_i^2$. In the process of finding a decision function for the decision variable in stage j, we choose the maxima of the average utilities associated with the chance variables. Using a result from order statistics, one can show that the variances of the maximum utilities associated with the decision variable in stage j is less than or equal to $\sum_{i=1}^{j} \sigma_i^2$. Using induction, one can show that $\mathrm{Var}(W) \leq \sum_{i=1}^{k} \sigma_i^2$. A proof of this assertion for a canonical example is given in the online appendix.

After all sampling is complete, we use the standard normal cumulative distribution function $\Phi(\cdot)$ to form the confidence interval $W \pm \Phi^{-1}(1-\alpha/2)\sqrt{\mathrm{Var}(W)}$. If the half-width of this interval is smaller than ϵ, we are confident that we have obtained an approximate (ϵ, α) estimate of mean utility. In testing, we find our method to be conservative in the sense that after many replications of the method, more than $100(1-\alpha)\%$ of the estimates are within ϵ of the true mean utilities for test problems where the true mean utilities are known.

One way to partition $\mathrm{Var}(W)$ into k components is to make a fixed, small number of pilot iterations (e.g., $n = 100$) to estimate the sampling error of each estimate at each stage for the fixed number of iterations. Then, $\mathrm{Var}(W)$ can be allocated in direct proportion to each stage's maximum sampling error variance obtained with the pilot iterations. This is not necessarily the best way to partition $\mathrm{Var}(W)$, but is a method that can be automated and used with minimal user intervention.

The estimated utilities obtained with the MMC method are slightly biased in a statistical sense because we are using sampling rather than exact calculations to obtain conditional expectations. Although our estimators for conditional expected utility in each stage are unbiased, we introduce bias when choosing the superior alternative having maximum estimated expected utility. This bias is most severe when the expected utilities of the superior and inferior alternatives are close to each other, so the bias will be lower in many practical problems where we wish to discern

differences in utilities that are relatively large. Our estimates in each stage are consistent, so performing more iterations in each stage will mitigate this bias. It is important to emphasize that at this point, we are interested in identifying an approximately optimal strategy, not necessarily the expected utility of the identified strategy. Further, if the bias from a previous stage affects all alternatives in a given stage equally, the bias will have no effect on the selection of the maximum.

We can obtain an estimate of overall expected utility that is biased slightly downward by running a global simulation to obtain a final estimate of the expected utility of any policy we identify as optimal with the MMC method. By definition, the expected utility of any policy must be less than or equal to the expected utility of the optimal policy, so this final estimate of maximum expected utility will be unbiased if we have, indeed, identified the optimal policy, and biased slightly downward if not. To obtain a better estimate of the expected utility of the optimal policy, we can average the biased high estimate obtained in the final stage of the MMC algorithm and the biased low estimate obtained in a final global simulation. Broadie and Glasserman (1997), who argue that under some restrictions there can be no general unbiased estimator of the price of American financial options, use a similar averaging technique. However, even though such bias exists, our experiments (some of which are not reported here) lead us to conclude that it is small. Moreover, as the number of iterations is increased in each stage of the solution, the bias diminishes so that the estimator of overall expected utility obtained from the MMC method is consistent (i.e., asymptotically unbiased).

In practice, it appears that our estimate of the required number of iterations is overly conservative. Ortiz and Kaelbling (2000) describe an alternative method for determining the number of iterations in a single-stage decision problem based on results from the field of multiple comparisons. Future research will focus on extending the multiple-comparison approach to multiple-stage problems.

4.4. Solving the OWSR Problem

Suppose our goal is to compute an (ϵ, α) approximate strategy for the OWSR problem, where $\epsilon = 1{,}000$ and $\alpha = 0.10$. The solution is found in three stages. Based on pilot runs of $n = 100$ observations, for each stage the estimated maximum standard errors at Stages 1, 2, and 3 were 12,929, 189,712, and 40,621, respectively. With $\epsilon = 1{,}000$ and $\alpha = 0.10$, we require a final sampling error variance of 369,612. Partitioning this in proportion to the maximum variances obtained with the pilot runs, we set $\sigma_1^2 = 3{,}130$, $\sigma_2^2 = 350{,}416$, and $\sigma_3^2 = 16{,}066$ (see the online appendix for further explanation of these bounds on variance), and $N = 300{,}000$ as the maximum number of interactions for any estimate.

In the first stage of the simulation (see Figure 5), the maximal decision variable is S. We sample chance variable SR for some combinations of the 12 possible states of $\{D, O, S\}$. Of these, four states are ruled out by the conditional for S (see Figure 2). Also, for six of the remaining eight states (when $S = nsr$), v_3 does not depend on SR (see Figure 3). Therefore, sampling is required only for the two remaining cases. Because only one variable is sampled, sequence is not an issue. Using the estimated expected utility values, the decision function for S is determined from Table 2 as follows: If $D = d$ and $O = we$, then $S = nsr$, and if $D = d$ and $O = so$, then $S = sr$. In all other cases, $S = nsr$ (because the conditional for S allows no other choices).

In stage 2, we replace S by a (degenerate) chance variable whose conditional distribution is the decision function just determined. Also, we replace v_3 by v_4 whose domain is $\{D, O\}$ and whose values are the maximum values (displayed in bold type) in Table 2. Specifically, $v_4(d, so) = 5{,}035.51$, $v_4(d, dr) = v_4(d, we) = v_4(nd, dr) = v_4(nd, we) = v_4(nd, so) = 0$. The resulting ID is shown in Figure 6.

In stage 2, the maximal decision variable is D. We sample chance variables in $\{O, R\}$ for all possible states of variables in $\{T, D\}$. The sub-ID that is solved in stage 2 is shown in Figure 7. Notice that variables S and SR are barren and play no role in this stage. The variables O and R are sampled using the sequence O, R for each possible state of $\{T, D\}$. Notice that for the state $T = nt$, R is degenerate so sampling for it is not necessary in this state. Also, when $D = nd$, neither v_2 nor v_4 depends on O or R and, thus, no sampling is necessary when $D = nd$.

The decision function for D is determined from Table 3 as follows: If $T = t$ and $R = ns$, then $D =$

Table 2 Results from Stage 1 of OWSR Simulation

State	$O = dr$	$O = we$	$O = so$
$D = d$ $S = sr$	*	(2,891.07) $se = 46.94$ $n = 145{,}600$	**5,035.51** $se = 46.94$ $n = 147{,}550$
$D = d$ $S = nsr$	**0** $se = 0$ $n = 0$	**0** $se = 0$ $n = 0$	0 $se = 0$ $n = 0$
$D = nd$ $S = sr$	*	*	*
$D = nd$ $S = nsr$	**0** $se = 0$ $n = 0$	**0** $se = 0$ $n = 0$	**0** $se = 0$ $n = 0$

Note. Cell entries are estimated expected utility, standard error, and number of iterations, respectively (* denotes there were no observations in these cells).

Figure 6 **OWSR ID After Stage 1**

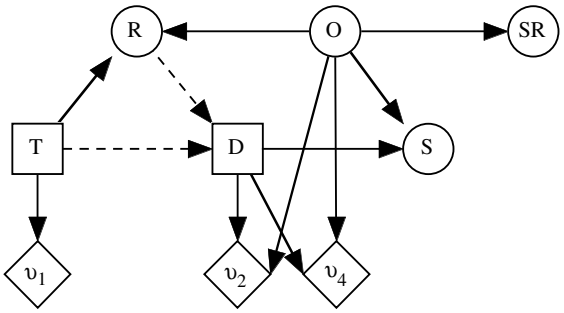


Figure 7 **OWSR Sub-ID Solved in Stage 2**

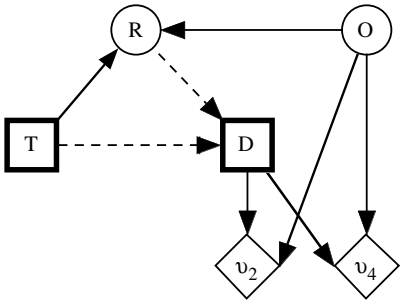


Figure 8 **OWSR ID After Stage 2**

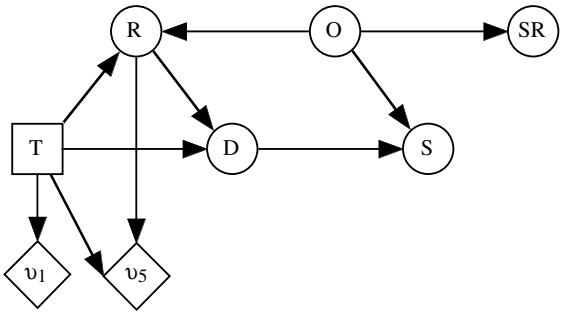
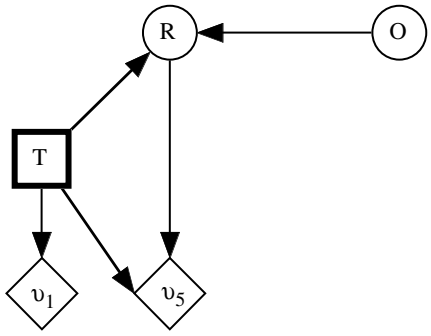


Figure 9 **OWSR Sub-ID Solved in Stage 3**



nd, and if $T = t$ and $R = os$ or $R = cs$, then $D = d$. Also, if $T = nt$, then $D = d$. In stage 3, we replace D by a degenerate variable whose conditional distribution is the decision function identified above. Also, we replace υ_2 and υ_4 by υ_5 whose domain is $\{T, R\}$ and whose values are the maximum values in Table 3. Specifically, $\upsilon_5(t, ns) = 0$, $\upsilon_5(t, os) = 34{,}302.41$, $\upsilon_5(t, cs) = 90{,}063.42$, and $\upsilon_5(nt, nr) = 21{,}145.49$. The resulting ID is shown in Figure 8.

The sub-ID that is solved in stage 3 is shown in Figure 9. Notice that variables D, S, and SR are barren and play no role in this stage. For each possible state of the maximal decision variable T, we sample for O and R using their conditionals in the sequence $O \hookrightarrow R$. Because the conditional for R is degenerate for the case $T = nt$ and O is not in the domain of the utility factors, no sampling is necessary for this case. The decision function for T is determined from Table 4 as follows: Do seismic test, which completes the solution of the OWSR problem.

To investigate the behavior of our method, we used the procedure described above to solve the OWSR problem several times using independent runs. To test the conservatism of our method, we set the limit of the number of iterations per stage, N, at various levels, then counted the number of times out of 100 independent replications the MMC method identified a strategy whose expected value is within $\epsilon = \$1{,}000$ of the optimal expected profit of \$23,400. The results are shown in Table 5, which indicates that our conservative method leads to more iterations than necessary, but yields a near-optimal policy with a high degree of

Table 3 **Results from Stage 2 of OWSR Simulation**

State	$R = nr$	$R = ns$	$R = os$	$R = cs$
$T = t$ $D = d$	$*$	(30,229.97) $se = 257.50$ $n = 79{,}286$	**34,302.41** $se = 409.69$ $n = 67{,}553$	**90,063.42** $se = 496.82$ $n = 46{,}211$
$T = t$ $D = nd$	$*$	**0** $se = 0$ $n = 0$	0 $se = 0$ $n = 0$	0 $se = 0$ $n = 0$
$T = nt$ $D = d$	**21,145.49** $se = 496.79$ $n = 45{,}200$	$*$	$*$	$*$
$T = nt$ $D = nd$	0 $se = 0$ $n = 0$	$*$	$*$	$*$

Note. Cell entries are average utility, standard error, and number of iterations, respectively ($*$ denotes there were no observations in these cells).

Table 4 **Results from Stage 3 of OWSR Simulation**

$T = t$	$T = nt$
23,649.05 $se = 106.10$ $n = 109{,}300$	21,145.49 $se = 0$ $n = 0$

Note. Cell entries are average utility, standard error, and number of iterations, respectively.

Table 5 Results from Running 100 Independent Replications of the OWSR Simulation when the Number of Iterations per Stage was Limited to Each Value Shown in the Top Row

Limit on iterations per stage, N	100	200	400	800	1,000	5,000	10,000
Number of approximate optimal identifications	56	62	63	67	76	90	96

Note. Bottom row cell entries are the number of times the MMC method identified a strategy whose true expected profit is within $\epsilon = \$1{,}000$ of the optimal expected profit of \$23,400.

confidence. Once a near-optimal policy is identified, we can obtain an unbiased estimate of the utility associated with that policy by running a final set of iterations in which the near-optimal decision functions are embedded.

Note that future research could focus on refining the estimates obtained by the MMC method. Much like discrete event simulation, where independent replications of a simulation model are frequently used to construct estimates of mean measures of performance, one could set the maximum number of iterations, N, relatively low, but repeatedly run the estimation procedure through all stages to obtain independent, identically distributed (IID) observations from which a confidence interval on the mean utility could be constructed using the IID observations. With some additional effort, one could also calculate estimates of the variances and correlations between estimates of the means at each stage. Although the bias will remain, this information could be useful for gauging the conservatism of the method by estimating the magnitude of the bias of the estimates obtained with the MMC method.

5. Bermudan Put Option Valuation

A *put* option grants its owner the right, but not the obligation, to sell stock shares in the future. Investors are interested in determining fair values for puts and other stock options for speculation and hedging.

Black and Scholes (1973) give an exact expression for the value of a *European* put option, which grants its holder the right, but not the obligation, to sell shares of a common stock for the exercise price, X, *at expiration* time T. An *American* put option grants its holder the right, but not the obligation, to sell shares of a common stock for the exercise price, X, *at or before expiration* time T. The Black-Scholes (1973) solution for a European put yields an approximation for the value of an American put option with the same exercise price, but in practice numerical techniques are used to obtain closer approximations of American options values.

The fair value of an American put option is the discounted expected value of its future cash flows. The cash flows arise because the put can be exercised at the next instant, dt, or the following instant, $2dt$, if not previously exercised, . . ., ad inf. To demonstrate its effectiveness for decision problems having many stages, we use the MMC method to value a put option that can be exercised at one of a large number of times, $k = 30$, including its expiration at time T. This type of financial instrument is called a *Bermudan put option*. By choosing k large enough, the computed value of a Bermudan option will be practically equal to the value of an American option.

Geske and Johnson (1984) develop a numerical approximation for the value of an American option based on extrapolating values for Bermudan options having small numbers (viz., $k = 1$, 2, and 3) of exercise opportunities. They show their results to be exact in the limit as $k \to \infty$. Broadie and Glasserman (1997) use simulation to price American options by generating two estimators, one biased high and one biased low, both asymptotically unbiased and converging to the true price. Our MMC algorithm yields an estimate in a manner similar to Broadie and Glasserman's (1997) high estimator, which uses a dynamic programming algorithm applied to a tree of simulated stock prices. However, Broadie and Glasserman (1997) consider only a small number of exercise opportunities, $k = 3$, because their method is exponential in the number of exercise opportunities. Because we use local computation, our method is linear in the number of exercise opportunities, which makes our technique tractable even when k is large.

5.1. Solving the Bermudan Put Option Problem

An influence diagram representation of the Bermudan option problem is depicted in Figure 10. Following standard practice in the financial literature, the stock prices, $S_1, S_2, \ldots, S_k$ evolve according to the discrete stochastic process

$$S_j = S_{j-1} \exp\left[\left(r - \frac{\sigma^2}{2}\right)\Delta t + \sigma\sqrt{\Delta t} Z_j\right] \quad \text{for } j = 1, \ldots, k, \quad (1)$$

Figure 10 Influence Diagram for the Bermudan Put Option

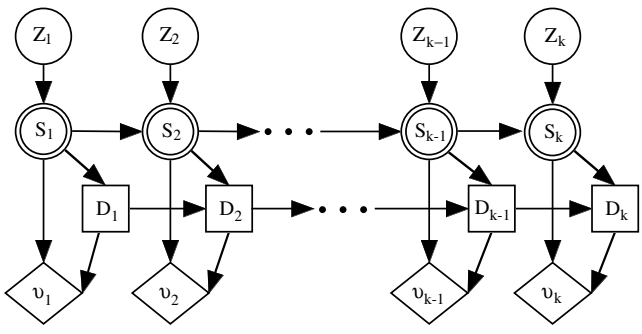


Figure 11 Conditionals for the Bermudan Put Option Decision Nodes

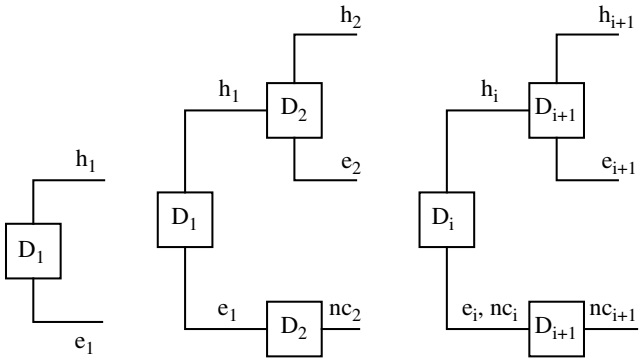
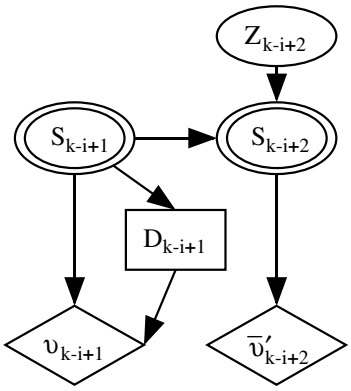


Figure 12 Bermudan Put Option Sub-ID for Stage *i*



where S_j is the stock price at time $j\Delta t$, r is the riskless interest rate, σ is the stock's volatility, $\Delta t = T/k$, and the Z_j are IID standard normal random variables. See Glasserman (2003) for more details about using simulation to value financial options.

The conditionals for the decision nodes in the problem are shown in Figure 11, where e_i, h_i, and nc_i indicate the alternatives exercise, hold, or no choice, respectively, for decision i. Because the option can be exercised only once, if it is exercised at node j, then *no choice* becomes the only alternative available for nodes $j+1, \ldots, k$. As the chance variables $Z_1, Z_2, \ldots, Z_k$ are independent, their conditionals are not shown. The additive factors of the utility function are: $\upsilon_j = \exp(-rj\Delta t)\max(X - S_j, 0)$ if $D_j = e_j$; $\upsilon_j = 0$ if $D_j = h_j$ or nc_j. This function reflects the asymmetric payoff of a put option—if the stock price S_j is greater than the exercise price X, then the option will not be exercised. Thus, the owner's potential loss is limited to the put's purchase price.

Figure 12 shows the sub-ID that is solved in the ith stage. The chance nodes for this problem represent continuous variables. We discretize the stock price variables to obtain a finite number of states for the simulation, and use linear interpolation to obtain values falling between finite states when we require continuity. The algorithm for a (0.10, 0.10) approximate strategy is shown in Table 6. Following the number of iterations discussion in §4.3, we require in each stage only one estimate, $H(S_i)$, the value at time $i+1$ of holding the option when the stock price is S_i at stage i. Thus, in each stage, we simulate long enough to obtain a standard error of $0.10/(1.645\sqrt{30}) = 0.011$ or we reach $N = 100$ iterations. The results of our experiment (using antithetic variates) are shown in Table 7.

For each exercise opportunity, $j = 1, \ldots, k$, the stock price S_j^* below which the expected value of exercising at opportunity j exceeds the discounted expected value of holding, is on the optimal exercise boundary. To use our procedure, we need not identify the optimal exercise boundary unless we wish to estimate the put value with global simulation. For any exercise opportunity j, we determine the boundary price as the stock price S_j^* such that $X - S_j^* \approx \exp(-r\Delta t)H(S_j^*)$. The optimal exercise boundary is the plot of S_j^* versus j, which is sufficiently smooth to allow the use of linear interpolation to obtain good estimates of the boundary prices at times between each $(j-1)\Delta t$ and $j\Delta t$ in the local simulation.

6. Summary and Conclusions

Monte Carlo methods for solving decision problems are not new. One of the first global methods was described by Hertz (1964) for decision trees. However, in the MMC method proposed here, we use information about the domains of probability conditionals and utility functions that is coded explicitly in influence diagrams to obtain a solution using local computation. Thus, like the exact methods, this will sometimes allow us to solve large decision

Table 6 MMC Algorithm for Valuing a Bermudan Put Option Using Local Computation

Step 1. For stage k, set $S_k \in \{0, 0.25, \ldots, 2X\}$. Create a table that lists $\bar{v}'_k(S_k) = \max(X - S_k, 0)$ as a function of S_k.

Step 2. For stage $i = k-1, k-2, \ldots, 1$, set $S_i \in \{0, 0.25, \ldots, 2X\}$. For each S_i, generate a value S_{i+1} according to Expression (1) in the text. Look up the value $\bar{v}'_{i+1}(S_{i+1})$ in the table created in the previous stage. Use linear interpolation to obtain values for prices falling between table elements. Iterate as many times necessary to estimate $H(S_i)$, the average value of holding one more period at price S_i, with a standard error of $\epsilon/(\Phi^{-1}(1-\alpha)\sqrt{k})$, where $\Phi(\cdot)$ is the standard normal cumulative distribution function, or the number of iterations has reached a prespecified limit, N. Compute $\bar{v}'_i(S_i) = \max(X - S_i, \exp(-r\Delta t)\overline{H}(S_i))$. Create a table that lists each $\bar{v}'_i(S_i)$, the estimated expected value of the option at stage i, as a function of observed price S_i.

Step 3. To price the option at time 0, use S_0 to generate a value S_1 according to (1). Look up the value $\bar{v}'_1(S_1)$ in the table created during stage $k-1$. Iterate as many times as needed to estimate $H(S_1)$, the average value of holding until time 1, with a $100(1-\alpha)\%$ confidence interval having half-width less than ϵ/k, or the number of iterations has reached a prespecified limit, N. Compute $v'_0 = \exp(-r\Delta t)H(S_0)$, which is the estimated price of the option.

Table 7 **Results of Applying the MMC Algorithm with a (0.10, 0.10) Approximate Strategy to Determine the Value P, of a Bermudan Put Option with Selected Values of Strike Price X and Volatility σ**

X	σ	P (Analytic)	P (MMC)	Std. Dev.	Number of iterations/stage
35	0.3	1.2194	1.2236	0.0063	28,000
35	0.4	2.1568	2.1571	0.0090	28,000
40	0.3	3.1733	3.1715	0.0090	32,000
40	0.4	4.3556	4.3525	0.0101	32,000
45	0.3	6.2365	6.2418	0.0080	36,000
45	0.4	7.3831	7.3806	0.0093	36,000

Note. For each option, the initial stock price equals \$40 ($S_0 = \$40$), the time to exercise $T = 7$ months, and the number of possible exercise instants (stages) $k = 30$. The P (Analytic) values are from Table I of Geske and Johnson (1984). The P (MMC) values are the means of estimates from 100 independent runs of the algorithm for each put option. Std. Dev. is the standard deviation of the estimates across runs.

problems that would be intractable using decision trees. It also allows us to obtain, with less computation, the same degree of precision resulting from global simulation (Hertz 1964).

If all variables are discrete and the problem is tractable, we could use exact methods for solving large problems. If either all variables are not discrete or the problem is not tractable, then the method described here may be useful. Because we are using Monte Carlo sampling, variables need not be discrete. For example, in the OWSR problem, the conditional distribution for SR could have been modeled with a continuous density function. This poses no problem as long as we can sample from such density functions. Notice that our method cannot easily handle continuous variables that are information predecessors of decision variables. In such cases, finding the optimal decision function is more complex and requires discretization of the continuous chance variables that are in the relevant domain of the decision variables, as we do in the Bermudan put option. If a problem is intractable by exact methods, then the problem may be tractable using our method if one is willing to settle for a crude approximation, i.e., a high ϵ and low level of confidence $100(1-\alpha)\%$.

Some limitations of our method are as follows. First, our method is not designed to compute an optimal strategy. Instead, it is designed to compute an approximately optimal strategy. The quality of the approximation can be improved but at the computational cost of doing more sampling. Second, although our method is designed to use local computation, it is not as localized as exact methods that compute the requisite conditionals for chance variables. In problems with all discrete variables, such requisite conditionals can be easily computed. This is not true in problems with a mixture of continuous and discrete variables. In our method, we have opted for ease of sampling of the chance variables (whether discrete or continuous) at the expense of computation in the form of more extensive sampling. Thus, in our multistage sampling process, not only do we sample for variables in $DRU(M) \cup P(M)$, we also sample for variables in $B(M)$. We can imagine a Markov chain Monte Carlo method that samples only the variables in $DRU(M) \cup P(M)$, but the distributions from which these variables are sampled will have to be computed from those specified in the influence diagram representation. One of the advantages of our sampling method is that observations are IID. This allows us to devise rules for stopping the simulation as functions of the easily computed standard errors of the estimated mean utility values. Third, the method given here for selecting the number of iterations is quite conservative. Further refinement of the method will lead to a reduction in the number of iterations required for practical applications of the method.

An electronic appendix is available for this paper at mansci.pubs.informs.org/ecompanion.html.

Acknowledgments

The authors are grateful to the Decision Sciences Group at the University of Kansas School of Business for helpful comments. Insightful discussions with Department Editor Jim Smith, Jean-Yves Jaffray, Steffen Lauritzen, and Barry Nelson led to many improvements. This work was partially supported by research grants from Sprint and Nortel Networks to the first author and from Sparta, Inc. to the second author.

References

Bielza, C., P. Müller, D. Rios Insua. 1999. Decision analysis by augmented probability simulation. *Management Sci.* **45**(7) 995–1007.

Black, F., M. Scholes. 1973. The pricing of options and corporate liabilities. *J. Political Econom.* **81** 637–659.

Broadie, M., P. Glasserman. 1997. Pricing American-style securities using simulation. *J. Econom. Dynam. Control* **21** 1323–1352.

Geske, R., H. E. Johnson. 1984. The American put option valued analytically. *J. Finance* **39**(5) 1511–1524.

Gilks, W. R., S. Richardson, D. B. Spiegelhalter, eds. 1996. *Markov Chain Monte Carlo in Practice*. CRC Press, London, U.K.

Glasserman, P. 2003. *Monte Carlo Methods in Financial Engineering*. Springer-Verlag, New York.

Goldsman, D., B. L. Nelson. 1998. Comparing systems via simulation. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*. John Wiley and Sons, New York.

Hertz, D. B. 1964. Risk analysis in capital investment. *Harvard Bus. Rev.* **42**(1) 95–106. Reprinted in *Harvard Bus. Rev.* 1979. **57**(5) 169–181.

Howard, R. A. 1971. Proximal decision analysis. *Management Sci.* **19**(9) 507–541.

Howard, R. A., J. E. Matheson. 1984. Influence diagrams. R. A. Howard, J. E. Matheson, eds. *Readings on the Principles and Applications of Decision Analysis*, Vol. 2. Strategic Decisions Group, Menlo Park, CA, 719–762.

Jenzarli, A. 1995. Modeling dependence in project management. Ph.D. thesis, University of Kansas Business School, Lawrence, KS.

Keefer, D. L. 1994. Certainty equivalents for three-point discrete-distribution approximations. *Management Sci.* **40**(6) 760–773.

Law, A. M., W. D. Kelton. 2000. *Simulation Modeling and Analysis*, 3rd ed. McGraw-Hill, New York.

Miller, A. C. III, T. R. Rice. 1983. Discrete approximations of probability distributions. *Management Sci.* **29** 352–362.

Niederreiter, H. 1992. Random number generation and quasi-Monte Carlo methods. *SIAM*, CBMS-NSF, Philadelphia, PA.

Nielsen, T. D., F. V. Jensen. 1999. Well-defined decision scenarios. K. B. Laskey, H. Prade, eds. *Uncertainty in Artificial Intelligence: Proc. Fifteenth Conf.*, Morgan Kaufmann, San Francisco, CA, 502–511.

Olmsted, S. M. 1983. On representing and solving decision problems. Ph.D. thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

Ortiz, L. E., L. P. Kaelbling. 2000. Sampling methods for action selection in influence diagrams. *Proc. Seventeenth National Conf. Artificial Intelligence* (AAAI-2000). AAAI Press, Menlo Park, CA.

Poland III, W. B., 1994. Decision analysis with continuous and discrete variables: A mixture distribution approach. Ph.D. thesis, Department of Engineering-Economic Systems, Stanford University, Stanford, CA.

Raiffa, H. 1968. *Decision Analysis: Introductory Lectures on Choices Under Uncertainty*. Addison-Wesley, Reading, MA.

Shachter, R. D. 1986. Evaluating influence diagrams. *Oper. Res.* **34**(6) 871–882.

Shachter, R. D., C. R. Kenley. 1989. Gaussian influence diagrams. *Management Sci.* **35**(5) 527–550.

Shachter, R. D., M. A. Peot. 1990. Simulation approaches to general probabilistic inference on belief networks. M. Henrion, R. D. Shachter, L. N. Kanal, J. F. Lemmer, eds. *Uncertainty in Artificial Intelligence*, Vol. 5. North-Holland, Amsterdam, The Netherlands, 221–231.

Shenoy, P. P. 1992. Valuation-based systems for Bayesian decision analysis. *Oper. Res.* **40**(3) 463–484.

Shenoy, P. P. 1998. Game trees for decision analysis. *Theory Decision* **44** 149–171.

Smith, J. E. 1993. Moment methods for decision analysis. *Management Sci.* **39**(3) 340–358.

Smith, J. E., S. Holtzman, J. E. Matheson. 1993. Structuring conditional relationships in influence diagrams. *Oper. Res.* **41**(2) 280–297.

Tatman, J. A., R. D. Shachter. 1990. Dynamic programming and influence diagrams. *IEEE Trans. Systems, Man, Cybernetics* **20**(2) 365–379.

von Neumann, J., O. Morgenstern. 1944. *Theory of Games and Economic Behavior*, 1st ed. Princeton University Press, Princeton, NJ.