Efficient Monte Carlo Decision Tree Solution in Dynamic Purchasing Environments

Buffett, Scott; Spencer, Bruce

Publisher's version / Version de l'éditeur:

The International Conference on Electronic Commerce (ICEC'03) [Proceedings], 2003

National Research Council Canada
Institute for Information Technology

Conseil national de recherches Canada
Institut de technologie de l'information

NRC · CNRC

*Efficient Monte Carlo Decision Tree Solution in Dynamic Purchasing Environments **

Buffett, S., and Spencer, B.
October 2003

* published in The International Conference on Electronic Commerce (ICEC'03). Pittsburgh, Pennsylvania, USA. October 1, 2003. NRC 46489.

Efficient Monte Carlo Decision Tree Solution in Dynamic Purchasing Environments

Scott Buffett
National Research Council Canada
Institute for Information Technology - e-Business
46 Dineen Drive
Fredericton, New Brunswick, Canada
E3B 9W4
scott.buffett@nrc.gc.ca

Bruce Spencer
National Research Council Canada
Institute for Information Technology - e-Business
46 Dineen Drive
Fredericton, New Brunswick, Canada
E3B 9W4
bruce.spencer@nrc.gc.ca

ABSTRACT

This paper considers the problem of making decisions in a dynamic environment where one of possibly many bundles of items must be purchased and quotes for items open and close over time. Probability measures on item prices are used when exact prices are not yet known. We show that expected utility estimation can be improved by considering how future information can affect the purchasing agent's behaviour. An efficient Monte Carlo simulation method is presented that determines the expected utility of an option in our decision tree, referred to as a QR-tree, where the number of simulations needed is linear in the size of the tree. In our experiments simulating a purchase agent in a specific market, the expected utility was estimated more than 50 times more accurately than a greedy method that always pursues the bundle with the current highest expected utility.

Keywords

Bundle purchasing, decision analysis, decision tree, expected utility theory, Monte Carlo

1. INTRODUCTION

Strategic purchasing tools that can cleverly ascertain the true value of many different options are becoming extremely important in today's market. More and more businesses are turning to Web-based pricing tools that sift through large volumes of data on product revenues, inventory levels and consumer activity to determine how much to charge for certain items during certain periods of time [14]. This "perfect pricing" translates into higher profits for business, mostly at the expense of the consumer. To combat this trend, buyers need the decision analysis technology that can properly assess not only the current purchasing options, but also the positive or negative potential of future opportunities. This technology is the focus of our work.

When making a decision on which of possibly many items should be purchased at any given time in an electronic marketplace, a purchasing agent will judge each option according to its decision-making criteria, and choose the one that is ranked highest. A common preference ranking approach is to make use of expected utility theory. Based on the possible outcomes, the decision-maker's preference for these outcomes, and the likelihood of these outcomes occurring, the decision-maker's expected utility is assessed for each option. The option with the highest expected utility is believed to be the best.

For the purchase of an item at some known price where there are no uncertain consequences, the expected utility is simply the certain utility of the purchase. However, if there are uncertain consequences of buying an item (e.g. if I buy A then I need to buy B, and B's price is currently unknown), then the utility and likelihood of the outcomes must be considered. The situation becomes more complicated when consequences of a choice in a decision include more decisions (e.g. if I buy A then I need to buy either B or C, and the prices for B and C are currently unknown). A conventional aid for assessing options and making decisions of this type is the decision tree. A decision tree models the system of subsequent decisions and their uncertain consequences that will result from some action. If the chance points in the decision tree give discrete, finite sets of consequences, then expected utility can be calculated by considering each possible outcome. Decision trees are not easily constructed and solved, however, if there are infinite sets of consequences. Such is the case, for example, if only a mean and a variance for the outcome of an item price is known. A discrete approximation method, or alternatively, a simulation method such as Monte Carlo, may be needed in this case. As computing power increases and the number of simulations that can be performed per second rises, Monte Carlo methods are becoming more and more common, as they allow for a much higher degree of accuracy than most approximation methods.

This paper examines the difficulties of using Monte Carlo simulation to determine the expected utility of choices in a decision tree where consequences include continuous outcomes for item prices, as well as subsequent decisions with uncertain information, and presents an algorithm designed to overcome these difficulties. We consider the setting where at any given time, the purchaser has price quotes for some items that are available for some fixed period of time, as well as the knowledge of incoming quotes, availabilities, sales, price fixing, etc., for other items available during some definite future period. The purchaser may have only a probability measure on these future prices. Purchasing decisions therefore have to be made online with incomplete information. The paper is organized as follows: After some background on decision analysis is provided in section 2, section 3 formalizes the problem and describes a greedy method for making purchasing decisions. Section 4 introduces efficient usage of Monte Carlo simulation in decision tree solution by developing a bottom-up solution method that incorporates *Pearson-Tukey* discrete approximation in certain parts of the tree to solve the problem fully. A few results in section 5 show experimentally that this Monte Carlo method works significantly better than greedy decision making in a specific example, based on the utility achieved when using each method over hundreds of thousands of test runs. Finally, section 6 discusses related work while section 7 gives conclusions and outlines plans for future work.

2. BACKGROUND

The main concepts utilized in this paper come from the area of decision analysis. In particular, decision trees, expected utility theory, and Monte Carlo simulation are most important. A decision tree [17], as it pertains to decision analysis, is a schematic presentation of a sequence of decisions and their possible consequences. It is used to model the flow of the entire decision process in order to assist the decision-maker in determining which course of action will most likely optimize some measure (e.g. monetary income, utility). The process of determining the expected outcome of the particular measure for each option in a decision tree is referred to as *rollback solution*. Refer to Raiffa [23] and Goodwin and Wright [9] for more introductory material.

In this paper, we use multi-attribute expected utility theory [1, 2, 24, 25, 26] for the decision-making criterion. Basing purchasing decisions on expected utility maximization (as opposed to expected cost minimization, for example) seems appropriate since one may wish to be sensitive to the purchaser's attitude toward risk, as well as preferences for attributes such as item quality, compatibility with other items, and supplier reliability. Refer to [7, 8, 15, 16, 18] for more introductory material on expected utility theory. Keeney and Raiffa [15] give general information on the assessment of utility functions, while Meyer and Pratt [20] give a more intensive treatment of quantitatively assessable utility functions, such as those for money, by simultaneously satisfying the quantitative results and the decision-maker's qualitative attitude toward risk. Also see [15] for work on the construction of multi-attribute utility functions.

A common approach for performing risk analysis on a system of decisions and predicting the best course of action is to use a Monte Carlo method [10, 19]. Monte Carlo methods use simulation to approximately solve a mathematical problem. These are often used when variables in the problem are too complex or have too many outcomes to model exactly in a decision tree. The expected value of a course of action can be estimated by simulating the outcomes several times and obtaining the average result.

3. PROBLEM DESCRIPTION

In this paper, we consider the situation where there may be several sets of items, referred to as *bundles*, that are deemed satisfactory by the purchaser. The purchaser's goal is to procure exactly one of these bundles. The market is dynamic, and therefore at any given time some items may be currently available at some given price for some fixed period of time, while others may be available during some future period of time at some currently unknown price. However, based on experience, market history, market conditions or information from some other sources, we assume that the buyer has a rough idea of what the price will be, and a measure such as a probability distribution function on the price can be obtained or estimated.

The problem in such a setting is deciding whether or not to buy an item that is currently available, before it expires. One must examine the bundles to which the item belongs and decide whether, given current costs and the future predicted costs and variability, the item should be purchased or allowed to expire. Item preference is also an issue, since even though the prospect of achieving a low price on a bundle might be good, the items in the bundle may lack quality, may not come from preferred suppliers, may not combine well, etc. All of these factors must be taken into account when making decisions.

Let I be a set of items and $\mathcal{B}$ a set of bundles. Each $i \in I$ has a cost $c(i)$, and each $b \in \mathcal{B}$ has a cost $c(b)$ equal to the sum of its item costs. Note that while $c(b)$ is defined in this way for the sake of simplicity, the model could still handle supplier incentives (e.g. discounts if multiple items are purchased together). At any given time, if an item i has become available, then assume the buyer knows $c(i)$. Otherwise, the buyer may have a probability measure $p : Z \rightarrow \Re$ on the outcome of the cost of i, where Z is the set of monetary units. The goal is to make decisions in hope of ultimately procuring the bundle b such that the two-attribute utility $u(b, c(b))$ of buying b at $c(b)$ is maximized.

3.1 Bundle Utility

Let $u : \mathcal{B} \times Z \rightarrow \Re$ be the bundle purchase utility function, where $\mathcal{B}$ contains the bundles and Z is the set of the monetary units. This function is determined by first obtaining the buyer's utilities for bundles $u_b : \mathcal{B} \rightarrow \Re$, and for spending $u_z : Z \rightarrow \Re$, which are predetermined by the buyer based on his preferences for items in the bundles, the item suppliers, attitude toward risk, and any other influential factors (see Keeney and Raiffa [15]). The two-attribute utility u is then determined as a function of these two. For example, one could choose the additive two-attribute utility function:

$$u(b, z) = k_b u_b(b) + k_z u_z(z) \tag{1}$$

where k_b and k_z are scaling constants that sum to 1.

Table 1: **Summary of time periods during which the buyer will have certain information about an item i**

Interval	Information
$[t_0, t_p(i)]$	nothing is known about i
$[t_p(i), t_n]$	$t_q(i)$ is known; $t_r(i)$ is known; a probability measure on $c(i)$ (and perhaps $c(i)$ itself) is known
$[t_q(i), t_r(i)]$	i is available for purchase
$[t_q(i), t_n]$	the actual price of i is known
$[t_r(i), t_n]$	i is subject to unavailability or price change

3.2 The PQR Protocol

The Prequote-Quote-Rescind (PQR) protocol is a message-passing protocol for information exchange between a supplier and a purchaser for probabilistic and temporal information. It defines when information will become known by the purchaser about items such as cost, the distribution of possible outcomes on cost, the time a quote will be offered, and the time a quote will be terminated. This information can then be used when planning purchases. Let $[t_0, t_n] \subseteq \Re$ be the period of time during which a buyer needs to purchase some bundle b of items I, and let $t_p : I \to \Re$, $t_q : I \to \Re$ and $t_r : I \to \Re$ assign time points to items $i \in I$, where $t_p(i)$ is the *prequote* time, $t_q(i)$ is the *quote* time and $t_r(i)$ is the *rescind* time for i, and $t_p(i) \leq t_q(i) < t_r(i)$. The intervals $[t_p(i), t_q(i)]$ and $[t_q(i), t_r(i)]$ are known as the *prequote interval* and the *quote interval* for i, respectively. The quote time is the time at which the quote will be offered, the rescind time is the time at which the quote expires, and the prequote time is the time at which the buyer learns the quote and rescind times. It is assumed that at the prequote time the buyer also learns or determines the probability measure on the cost outcome of the item. Table 1 summarizes the time periods during which the buyer will have information on the cost, potential cost, and availability of an item.

3.3 Decision Points

Each time an item is about to expire, a decision must be made on whether or not the item should be purchased. In this paper, we consider the decision points to be the set $\{t_r(i) \mid i \in I\}$, although it is likely that decisions really need to be made some short time before $t_r(i)$ to allow for a small period of time to complete the transaction before the actual expiry time. At each decision point, the expected utility of each option (buy the item, not buy the item) needs to be calculated, and the option that maximizes expected utility is chosen.

3.4 Decision Trees

A greedy approach to the problem of making these purchasing decisions is to simply pursue the bundle with the highest expected utility. That is, whenever an item i is about to expire, the decision of whether or not to buy i is based on whether or not the bundle with the highest expected utility includes i. If it does, then i is purchased. This is not the best approach however, since the consequences of the decision are not just the item prices, but also future decisions. These future decisions have to be analyzed in order to obtain an accurate measure of the expected utility of the choices. Traditionally in decision analysis, the *decision tree* is used to model this decision process. However, decision trees that are used for problems such as this can get unreasonably large, since typically there are too many possible outcomes for item prices. Discrete approximations can be used to make the tree size more manageable, but this results in a loss of accuracy. If there are many items involved, trees can still become too large even if only two or three discrete outcomes are chosen for each item. We therefore present a modified version of a decision tree which models the entire decision process and acts as a vehicle for the use of Monte Carlo simulation for determining expected utilities, called the *Quote-Rescind-tree* (QR-tree). No chance nodes are used, but rather the point at which the buyer learns a price is simply indicated in the tree. The entire probability measure on the price outcome can still be specified. In this paper, we assume that prices are drawn from a normal distribution for which only a mean and standard deviation need to be specified. However, any type of probability measure can be used.

The QR-tree is built in a two-step process. Initially, a *purchase procedure tree* is built. This tree models the system of decisions and purchases that, given the current information, the buyer will make to ultimately procure a bundle. The purchase procedure tree is then modified to facilitate the expected utility computation method described later in this paper, resulting in a QR-tree. The trees are defined as follows: A purchase procedure tree is a tree $T = (V, E)$ where V is partitioned into two types of nodes: a set P of purchase nodes and a set D of decision nodes. The purchase nodes are labelled by the items they represent. There are also three functions on the nodes, $t_q : P \to \Re$, $t_r : P \to \Re$, and $t : D \to \Re$. At time t, let I be the set of items not yet procured for which the prequote or quote interval includes t (i.e. $i \in I$ iff $t_p(i) \leq t \leq t_r(i)$), and let $\mathcal{B} \subseteq 2^I$ be a set of bundles. T is a purchase procedure tree at time t on $\mathcal{B}$ iff the following are true:

- Each purchase node in T has at most one child node.
- Each decision node in T has two child nodes.
- Each purchase node p in T represents the purchase of an item i and $t_q(p) = t_q(i)$ and $t_r(p) = t_r(i)$.
- For any two purchase nodes p_1 and p_2 in T, if p_1 is an ancestor of p_2 then $t_r(p_1) \leq t_r(p_2)$.
- For any two sibling nodes v_1 and v_2 in T, if v_1 is to the left of v_2 then $t_r(v_1) \leq t_r(v_2)$.
- For any decision node d with left child $\ell(d)$, $t(d) = t_r(\ell(d))$.
- For any root-to-leaf path in T, the set of items represented by the purchase nodes on the path is a bundle in $\mathcal{B}$, and all elements of $\mathcal{B}$ are represented by some path.

The purchase procedure tree is constructed as described in Algorithm 1.

ALGORITHM 1. Let I and $\mathcal{B}$ be defined as above. For any node n, let I_n be the set of items labelling ancestors of n, let $\mathcal{B}_n = \{b \in \mathcal{B} \mid I_n \subseteq b\}$ be the set of bundles that can be procured below n, and let $I_{\mathcal{B}_n} = \{i \in b \mid b \in \mathcal{B}_n\} \setminus I_n$ be the set of items that can potentially label proper descendents of n.

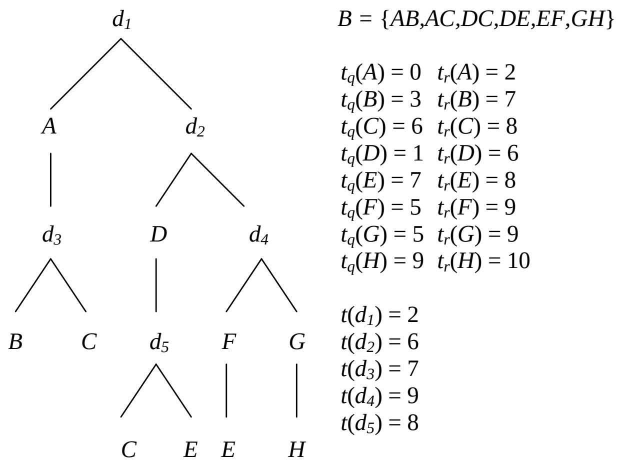

Figure 1: Example tree that is both a purchase procedure tree and a QR-tree

1. Let r be the root;
2. $construct(r)$;
3. While there is a non-terminal leaf node n, $constructChildren(n)$;

$construct(n)$: If $I_{\mathcal{B}_n} = \phi$, then let n be a terminal node. Else if there exists an i such that $t_r(i)$ is a minimum in $I_{\mathcal{B}_n}$ and $i \in b$ for all $b \in \mathcal{B}_n$, then let n be a purchase node labelled by i. Else, let n be a decision node.

$constructChildren(n)$: If n is a purchase node, then let $c(n)$ be the child of n and $construct(c(n))$. Else n is a decision node. Let $\ell(n)$ and $r(n)$ be the left and right children of n respectively, and i be an item in $I_{\mathcal{B}_n}$ such that $t_r(i)$ is minimal. Let $\ell(n)$ be a purchase node labelled by i, let $t(n) = t_r(\ell(n))$, and $construct(r(n))$.

Figure 1 depicts an example purchase procedure tree. Note that a purchase procedure tree built at time t will not depict purchases or decisions that occurred before t. The root always represents the first action starting at t. The QR-tree is constructed by a direct transformation of a purchase procedure tree, as described in Algorithm 2. This process simply reorders the purchase nodes so that they are sorted by t_q time rather than t_r time. Decision nodes and their corresponding decision times are unchanged. Steps 1 and 2 move all purchase nodes to the appropriate segments of the tree (i.e. in between the appropriate pairs of decision nodes). Once all of these reside in the proper segments, step 3 sorts the purchase nodes within each segment by t_q time in ascending order.

ALGORITHM 2. Let T be a purchase procedure tree with a decision node root. T is transformed to a QR-tree as follows:

1. For any decision node d in T, let $t_q(d) = \min\{t_q(v) | \; v \in des(d)\}$

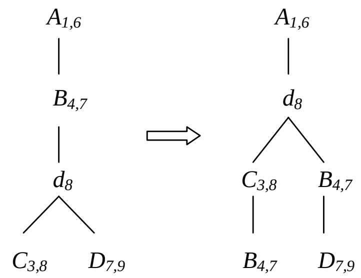

Figure 2: Example of a QR-tree transformation as described in Algorithm 2

2. While there exists a purchase node p with first descendent decision node d such that $t_q(d) < t_q(p)$, remove p from its position in T and insert it below d as $\ell(d)$. Make a copy of $\ell(d)$ and insert the copy as $r(d)$.
3. For every pair of decision nodes d and d' in T such that d is an ancestor of d' and there are no intervening decision nodes on the path from d to d', sort the purchase nodes on this path by their t_q values in ascending order from d to d'.
4. Give each leaf in this tree a child node. These new nodes are called the *endpoints*.

Figure 2 gives an example of such a transformation. For each purchase node in the example, the subscripts denote the t_q and t_r times respectively, while the lone subscript for each decision node denotes the decision time. Notice that in the QR-tree, decisions are made at the same time as in the purchase procedure tree, and the same bundles are pursued for each choice. The only difference is that some purchases are pushed closer to the end. Note that this reordering is done only for the sake of look-ahead to predict expected utility. In reality, all purchases will occur at (or just before) their t_r times. Since endpoints serve no purpose other than for convenience in computation (as is shown later), they typically are omitted in QR-tree drawings. A QR-tree has all of the same properties as a purchase procedure tree except:

- For any two nodes v_1 and v_2 in the tree, if v_1 is an ancestor of v_2 then $t_q(v_1) \leq t_q(v_2)$.
- A decision node d with decision time $t(d)$ may have a descendent purchase node p such that $t_r(p) < t(d)$. If this is the case, however, the purchase represented by p will be part of any bundle procured below d, and is thus not relevant to the decision.

Figure 1 gives an example of a QR-tree. For clarity, this tree is both a purchase procedure tree and a QR-tree (no transformation is necessary).

4. SOLVING THE DECISION TREE

4.1 Solution Strategy

Solving the QR-tree to determine the expected utilities should be done in a bottom-up manner. Top down solution is infeasible since too much simulation is required. Consider attempting top-down solution of the example in Figure 1. To determine the expected utility of proceeding to decision node d_2, for example, the information that will be known at that decision time (namely the prices of D, F, G) must be simulated. For each simulation, the expected utility of each of the left choice (D) and right choice (d_4) must be computed and the higher noted, since this is the choice the decision-maker would make if this simulation represented actual outcomes. To determine the expected utility of the right choice d_4, several simulations of the extra information known at d_4 (namely E) need to be run for the given values of F and G. If d_4 had any descendent decisions, then for each of these simulations, several further simulations would be needed, and so on. For any decision node d, let x be the required number of simulations of item costs that will be known at decision time $t(d)$ in order to compute the expected utility of d. Then if h is the decision node height of d (the maximum number of decision nodes on a path from d to a leaf), then the number of simulations required to compute the expected utility of d is $O(x^h)$. Since x can be quite large (say 10,000 - 100,000 for reasonable accuracy), then x^h can get unmanageably large for even small h.

We propose a bottom-up approach for computing these utilities that ensures that the number of simulations required grows linearly with the number of nodes in the tree. The goal is, for any node n in the tree, to be able to estimate the expected utility of n for any outcome for the nodes above n without doing any further simulation on n's subtree. To explain how this is done, two important concepts must be introduced: the *q-horizon* and the *q-subhorizon*. Let d be a decision node in a QR-tree and $des(d)$ be the set of purchase node descendents of d. The q-horizon of d is the subset of $des(d)$ representing items for which the prices will be known when d must be decided. The q-subhorizon of d is the subset of its q-horizon that are in the q-horizon of an ancestor decision node of d. More formally:

Definition 1. The *q-horizon* of d, denoted by $qh(d) = \{n \in des(d) \mid t_q(n) < t(d)\}$, is the subset of $des(d)$ for which item prices will be known when d must be resolved. The set $qs(d)$ of items that are represented by the nodes in $qh(d)$ is the *q-set* of d.

Definition 2. Let d and d' be decision nodes such that d' is an ancestor of d and the path from d' to d has no decision nodes. The *q-subhorizon* of d, denoted by $qsh(d) = qh(d) \cap qh(d')$, is the subset of $qh(d)$ consisting of elements that are also in $qh(d')$. The set $qss(d)$ of items that are represented by the nodes in $qsh(d)$ is the *q-subset* of d.

The example tree in Figure 3 shows the tree from Figure 1 with q-horizons indicated by dotted lines. Note that Algorithm 2 constructs QR-trees in such a way that, for any decision node d and purchase node n, if $n \in qh(d)$ then $n' \in qh(d)$ for all purchase nodes n' on the path between d and n. So all elements of a q-horizon are connected. To solve the tree bottom up, the expected utility of each node must be able to be ascertained (or estimated) for any occurrence above it. Such occurrences include not only outcomes for money spent before the node is encountered, but also

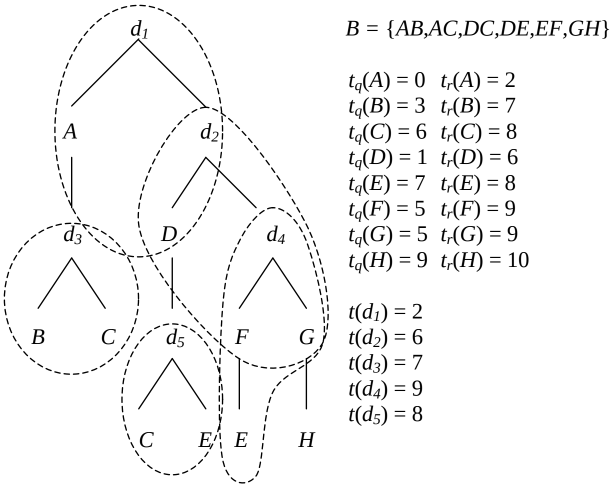

Figure 3: Example of a QR-tree with q-horizons indicated by dotted lines. For example, for d_4 the q-horizon consists of the nodes representing purchases of F, G and E, and the q-subhorizon consists of the nodes representing purchases of F and G.

outcomes of the items represented in the q-subhorizon of decision nodes, since they are part of ancestor q-horizons and therefore are simulated when these ancestor nodes are evaluated. Solution is done by computing 1) an *above-function* for each purchase node, and 2) a *q-subset-mapping* for each decision node. Each of these two types of functions takes a state at a node, consisting of the amount already spent at that point and, for decision nodes, the costs of items in the q-subset. Since the actual state is known for each child node at the root of the QR-tree, the function for each child is a constant representing the expected utility of that choice.

4.2 Computing an Above-function

Let n be a purchase node in a QR-tree. The set A_n of *above* values for n is the set of all possible outcomes for the sum of the cost of items represented by proper ancestor purchase nodes of n plus the amount already spent on items procured before the QR-tree was built. An above-function $a_n : A_n \rightarrow \Re$ is a function that maps a value $a \in A_n$ to the expected utility of buying n, given that a is the amount spent before n is encountered. If A_n is infinite, then a_n likely cannot be computed exactly over the entire domain. However, since a_n is a monotone strictly decreasing function (higher cost leads to lower expected utility), if we have actual values for some points then we know that the values at other points are tightly constrained and therefore can be accurately estimated. So a rather simple solution to the problem of computing an above-function for n is to choose a few above values $A'_n \subset A_n$, compute the expected utility a'_n for each chosen value, and then fit a curve, thus specifying an estimated a_n for all of A_n. For this paper, the set of points chosen is $A'_n = \{x \mid P(X < x)$ is a multiple of .05 and $.05 \leq P(X < x) \leq .95\}$.

Consider the following notation used to represent the set of joint price outcomes. Let $I' \subseteq I$ be a subset of the items represented in a QR-tree, let $N_{I'}$ be the set of purchase nodes that represent an $i \in I'$, and let $K(I')$ be the set of joint price outcomes for items in I' where each $k : N_{I'} \rightarrow \Re$ in $K(I')$ is a function that assigns a price to each purchase node in $N_{I'}$. Let $MC(I', \alpha(k), \varepsilon)$ be a function that takes a set I' of items and a function $\alpha : K(I') \rightarrow \Re$, and returns the average result of $\alpha(k)$ for several independently generated random outcomes $k \in K(I')$, within a standard error of ε. Above-mappings are computed as follows: If n is an endpoint, then the above-mapping is simply the two-attribute utility function for the bundle procured above n. That is, if b is the particular bundle procured, then for each $a \in A_n$,

$$a_n(a) = u(b, a) \tag{2}$$

If n is a purchase node with child n', then for each $a \in A'_n$, the expected utility $a'_n(a)$ is computed by

$$a'_n(a) = MC(\{i_n\}, a_{n'}(a + k(n)), \varepsilon) \tag{3}$$

where i_n is the item represented by n. Testing shows that using regression to find a degree-three polynomial representing a_n works quite well.

The expected utility of n can now be predicted for any above value on the continuous scale. Note that if n is a purchase node that resides in a q-horizon for a decision d, then it is not considered separately but rather as part of the information available at d. Thus its above-function is irrelevant and not computed.

4.3 Computing a q-subset-mapping

Computing expected utilities for decision nodes is much more complicated, since future information must be considered. At the time a decision node d must be resolved, the prices of all items in the q-set of d will be known. Therefore, in order to compute the expected utility of d for some above value a, all possible joint outcomes of the prices of items in the q-set should be considered. For each outcome, the expected utility of d is taken as the choice with the higher expected utility, since we assume that the buyer will always make choices that maximize expected utility, given the available information. Making the computation even more complicated is the fact that the q-horizon of d may contain other decision nodes. So, at decision time, the purchaser will know part of the information that will be known at these future decisions. This problem can be overcome, however, by carefully considering what information known at a decision node will also be known at ancestor decisions, and properly passing that information up during solution. This is the purpose of the q-subhorizon.

Since we need to know the expected utility of each choice at a decision node d given the amount spent so far and the costs of items represented in the q-subhorizon, if there is a descendent decision node d' where $qh(d) \cap qh(d') \neq \phi$, then we need to be able to determine the expected utility of d' given item prices represented by nodes above d' and in the q-subhorizon of d'. For this reason, a q-subset-mapping is computed. The q-subset-mapping $qssm_d$ for a decision node d is a function that maps a joint outcome for prices of items in the q-subset of d to an above-function. The above-function in turn maps the above amount to the expected utility. Consider determining the q-subset-mapping for d_1 in the partial QR-tree given in Figure 4. Given a joint outcome k for the items in $qss(d_1) = \{A, D\}$, the above-function for d_1 is computed as follows: For each $a \in A'_{d_1}$, the items in $qs(d_1) - qss(d_1) = \{B, C\}$ are simulated. For each simulation, the expected utility for each choice at d_1 is computed: For a path below the decision node, if the final node in the q-horizon is a purchase node (as is the case with node B in the left path below d_1 in the example) with child n', then the expected utility of that choice is $a_{n'}(x)$, where x is the sum of the prices of all items above n' given a, k and the simulated prices (if n' is a decision node, since its q-horizon must be empty then $a_{n'} = qssm_{n'}$). Otherwise the path reaches a decision node before the q-horizon is exited, as is the case with d_2. In this case, given a, k and the simulated prices, the outcome for prices of items in $qss(d_2)$ is entered into $qssm_{d_2}$ to determine the appropriate above-function to use for d_2, and the sum of item prices above d_2 are used with the above-function to determine the expected utility.

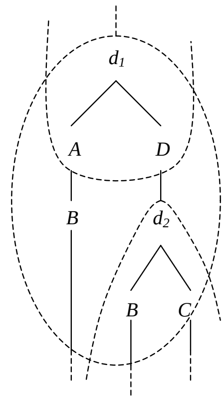

Figure 4: Partial QR-tree

The problem here lies in the difficulty of specifying the expected utility of a decision node for any given outcome of the item prices in its q-subset. If there are only one or two items then techniques such as regression can be used to estimate a function, given the utilities for a few chosen values. However, for more items this technique can be time consuming and produce very inaccurate results. Instead, we incorporate the Pearson-Tukey three-point approximation (PT-approximation) [13, 21] for such item prices. When computing q-subset-mappings, q-subset items are assumed to have only three possible outcomes. An above mapping for a decision node with m q-subset items will thus be computed for each of the 3^m joint outcomes. This is a reasonable number for fairly small m. The three discrete outcomes $\{x_1, x_2, x_3\}$, each with probability $p(x)$ of occurring, associated with a PT approximation of a continuous probability density function for a random variable X are as given in Table 2. If X is a normally distributed random variable with mean μ and standard deviation σ, then the three outcomes in a PT approximation are as given in Table 3. We define the set $K(I')$ of joint outcomes where each $k \in K(I')$ is taken in accordance with the PT three-point approximation to be the set of *PT-outcomes*. Thus, even if the true num-

Table 2: Outcomes and probabilities for the PT three-point approximation

Outcome x	$p(x)$
the x such that $P(X > x) = .95$	.185
the x such that $P(X > x) = .5$	.63
the x such that $P(X > x) = .05$	.185

Table 3: Outcomes and probabilities for the PT approximation of a normal random variable

Outcome x	$p(x)$
$x = \mu - 1.645\sigma$	.185
$x = \mu$	.63
$x = \mu + 1.645\sigma$	.185

ber of outcomes for items in I' is infinite, the number of PT-outcomes for I' is $3^{|I'|}$.

The formal technique for determining the q-subset-mapping is now given. Let d be a decision node. The q-subset-mapping $qssm_d$ for d maps each PT-outcome for $qss(d)$ to an above function, which is computed as follows: Given a PT-outcome, the outcomes for items in the *q-subset-complement* $qss^c(d) = qs(d) - qss(d)$ are simulated, but only those items that do not reside in a descendent decision node's q-subset, since only PT-outcomes are considered for those nodes. Let d be a decision node, d_ℓ and d_r the first left and right descendent decision nodes of d respectively (if they exist), let $qss'(d) = qss(d_\ell) \cup qss(d_r)$ be the union of the q-subsets for d_ℓ and d_r (if either of d_ℓ or d_r does not exist then treat their q-subset to be empty), and let $sim(d) = qss^c(d) - qss'(d)$ be the subset of the q-subset-complement of d to be simulated. The q-subset-mapping $qssm_d$ maps an outcome k' for the q-subset of d to an above-mapping a_d, approximated by a'_d, where, for an $a \in A'_d$, $a'_d(a) =$

$$\sum_{k'' \in K(qss'(d))} p(k'') \cdot MC(sim(d), \max\{\omega(a, k+k'+k'', \ell(d)), \omega(a, k+k'+k'', r(d))\}, \varepsilon) \tag{4}$$

where $k+k'+k''$ is the concatenation of the outcomes given by k, k' and k'' for the set of items $sim(d) \cup qss(d) \cup qss'(d) = qs(d)$, $p(k'')$ is the probability of k'' occurring according to the PT approximation, $\ell(d)$ and $r(d)$ are the left and right children of d, and $\omega(a, k, n)$ for a node n is computed as follows: If n is a purchase node and $k(n)$ exists (i.e. k assigns an outcome to n's item), then $\omega(a, k, n) = \omega(a + k(n), k, n')$ where n' is the child of n. If $k(n)$ does not exist (as is the case with decision nodes and endpoints), $\omega(a, k, n) = a_n(a)$ where, if n is a decision node, $a_n = qssm_n(k')$ where k' is the outcome for items in $qss(n)$ consistent with the item outcomes given by k. Informally, $\omega(a, k, n)$ is the expected utility of n for a given $a \in A_n$ and a given outcome k for some of the descendents of n.

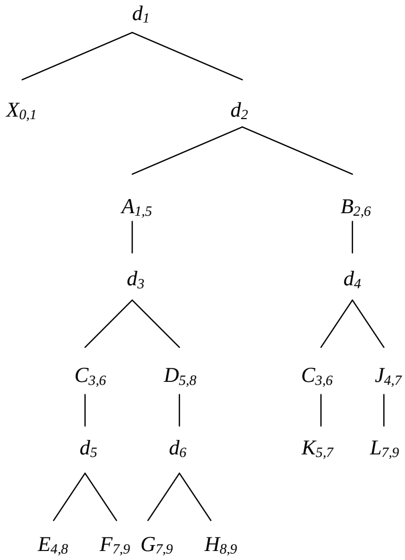

Figure 5: QR-tree for experiments. The subscripts for each purchase node denote the t_q and t_r times, respectively. The q-sets for each decision node are $d_1 : X$, $d_2 : ABCEJ$, $d_3 : CDE$, $d_4 : CJK$, $d_5 : EF$, $d_6 : GH$.

5. RESULTS AND ANALYSIS

Figure 5 shows the QR-tree on which tests were run. Fifty random instances were chosen, where the means were chosen from a uniform distribution with range [0.9, 1.1] and standard deviations from a uniform distribution with range [0,0.3] for the costs of items A-L. All bundle utilities were considered to be equal and therefore could be ignored, and the risk neutral utility function

$$u_z(z) = 1 - \frac{z-2}{2} \tag{5}$$

for money was used. Thus, the two-attribute utility function was simply $u(b, z) = 0(u_b(b)) + 1(u_z(z))$. Quote intervals were kept static throughout testing. For each instance, the expected utility of the right subtree was calculated both greedily (as the maximum expected utility of all bundles that could be procured in the right subtree) and using our approach. The mean of item X (the left child of d_1) was then taken as the cost that would make the utility of buying X equal to the average of the two utilities calculated for the right subtree. This was done to ensure that many instances used would be relevant. If the expected utility of the left subtree is less than or greater than both expected utilities calculated for the right subtree, then a decision-maker using either method will make the same choice, making for an irrelevant test case. Setting the left mean utility to be exactly in between the two right utilities with a relatively small standard deviation (.05 was used) ensures that most test runs will be relevant. Monte Carlo simulations in calculating the expected utilities made use of antithetic variate sampling [11], and calculations used a standard error threshold of .001. For each instance, 5000 outcomes for item costs were selected at random according to the means and vari-

Table 4: Summary of results for 250,000 test runs

Measure	Greedy	New
Avg utility achieved	0.574	0.596
Avg expected utility (right subtree)	0.537	0.593
Avg utility achieved (right subtree)	0.593	0.594

ance, and each method of decision making was applied and tested for each case.

Table 4 gives a summary of the results. For each method, the average utility achieved over all 250,000 runs is given. Also, in an effort to show the relative accuracy of our Monte Carlo method's expected utility estimation, each method was tested on the right subtree for every test run. Table 4 shows the average estimated expected utility of the right subtree as well as the average utility achieved. Note that the utility achieved given that the right subtree is chosen is almost the same for each method, since much of the important information becomes known before the next decision (d_2) needs to be made and therefore each method will almost always suggest the same choices after d_1.

The new method clearly outperforms the greedy method, achieving .022 more utility on average. By the utility function used, this results in an average savings of \$0.044. Notice that, in this example, item costs are very low (around \$1). If instead the same problem involved items that were around \$1000 (with standard deviations increased by the same factor), then this would translate into an average savings of \$44.

Tests on the right subtree show that this new method for estimating expected utility is very accurate. While the average highest expected utility of all bundles procured in the right subtree (taken as the expected utility of the right subtreee by the greedy method) is .537, the purchaser actually achieved an average of .593 using this method, for a difference of .056. On the other hand, the new method predicted that the true expected utility of proceeding in that direction is .593 while the purchaser actually achieved .594 using this method, for a difference of only .001 from the estimate. While this is a very specific (but randomly chosen) case, for this particular tree and distributions from which item price means and standard deviations are selected, the new method has 1/56 the error when estimating the expected utility than the greedy method. This increased accuracy provides for better decision-making at d_1, resulting in the overall increase of .022 in achieved utility.

6. RELATED WORK

Recent work has focused on decision procedures for bundle purchasing where there are multiple auctions in which to bid. Boutilier *et al.* [3, 4] consider the model where a bundle of items[1] must be purchased by participating in a subset of several *sequential* auctions. These auctions are first-price sealed-bid, have known start/end times, and do not overlap. At each decision point (auction start time), the optimal bidding strategy is computed and the amount to bid (if any) in the current auction is determined. Our work differs from this in both the auction mechanism used as well as in the timings, as we allow for quotes to be open in parallel. Byde [5] considers multiple simultaneous auctions, but the purchaser's goal in this case is to buy only one single item. The problem where there are multiple simultaneous auctions has been examined by Byde *et al.* [6]. In their model, the purchaser attempts to buy possibly multiple units of only a single good. Finally, Preist *et al.* [22] discuss bundle purchasing[2] in the setting where there are multiple simultaneous auctions. While their problem is more daunting than ours since they consider English, Dutch and sealed-bid auctions, their decision-making method is similar to our greedy method. At each decision point, the optimal set of auctions (in terms of expected utility) in which to bid is computed, and this set is pursued. Since the algorithm does not truly commit to this set, but instead re-evaluates its options at each decision point, this expected utility is not an accurate account of the true expected utility of the choice.[3] The main idea of our paper is to predict how the algorithm will behave in the future in order to estimate the true expected utility of a choice as accurately as possible.

The idea of using simulation to solve decision trees is not entirely new. Hespos and Strassman [12] proposed the use of *stochastic decision trees* for risk analysis in investment decisions. These decision trees allow discrete chance forks to be replaced by continuous probability distributions. However, the trees used are much simpler in that they do not have the "q-horizon overlap" problem seen in the purchasing trees in this paper. That is, information does not become available several decisions before the one to which it is relevant. These non-overlapping q-horizons are more commonly the case in the investment decision-making domain, since investment decisions typically need to be made before information on those investments becomes known. Moreover, when their trees do become too complex, simplification techniques are applied that prune the tree of certain branches. Our method works well on any tree and no pruning is necessary to keep the solution process feasible.

7. CONCLUSIONS AND FUTURE WORK

This paper gives an efficient and effective technique for using Monte Carlo simulation to solve decision trees for procuring bundles of items in a dynamic purchasing environment. We consider the setting where some items are currently available for a fixed period of time at known prices, while other items may be available in the future during some known period of time and the buyer has some probability measure over the possible price outcomes. We study the problem of deciding whether or not to buy an item that is about to expire. To do so, a QR-tree is constructed, and a Monte Carlo method is used to estimate the expected utility of the two options by estimating the expected utilities of all future decisions that will result as consequences of each choice. Experiments show that this technique gives a much more accurate estimate of expected utility, and helps the purchaser to achieve a significantly higher utility than a greedy method that simply instructs the purchaser to pursue the best bundle.

[1] The authors refer to these as "resources".

[2] The authors refer to bundle purchasing as "service composition"

[3] This value is, however, more difficult to compute than our greedy estimate since the auctions in which the buyer currently holds the winning bid must be taken into consideration.

In future work, we plan to examine even more accurate methods of expected utility estimation. While PT-approximation works well, we feel that there is room for improvement. Our next endeavour will be to test the application of a learning technique to predict under which outcomes for q-subset items does the left choice at a decision provide a higher expected utility, and under which outcomes does the right choice provide a higher utility. Simulation could then be performed top-down, and choices at each decision for the given q-subset outcomes can be made based on what was learned from the observed data.

Another project is to examine the effect of relaxing the constraint in the model that only the items of one particular bundle can be purchased. It could be beneficial to allow the purchaser to buy extraneous items in the case that several limited-time offers arise that are too good to refuse. This may cause the purchaser to travel down more than one branch in the tree simultaneously. The problem here is that the number of options to assess at any point in the tree becomes too large. However, certain not-so-restrictive constraints could be imposed to make this assessment feasible.

We also plan to extend the model to allow the buyer to participate in online auctions. While the addition of various auction mechanisms would greatly magnify the computational burden, it would certainly make the methods described much more useful. New techniques, based on those developed in this paper, would likely be needed to accomplish this goal.

8. REFERENCES

[1] F. J. Anscombe and R. J. Aumann. A definition of subjective probability. *Annals of Mathematical Statistics*, 34:199–205, 1963.

[2] D. Bernoulli. Exposition of a new theory on the measurement of risk. *Econometrica*, 22, 1954.

[3] C. Boutilier, M. Goldszmidt, and B. Sabata. Continuous value function approximation for sequential bidding policies. In *the Fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI-99)*, pages 81–90, Stockholm, 1999.

[4] C. Boutilier, M. Goldszmidt, and B. Sabata. Sequential auctions for the allocation of resources with complementaries. In *the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*, pages 527–534, Stockholm, 1999.

[5] A. Byde. A dynamic programming model for algorithm design in simultaneous auctions. In *WELCOM'01*, Heidelburg, Germany, 2001.

[6] A. Byde, C. Preist, and N. R. Jennings. Decision procedures for multiple auctions. In *Proc. 1st Int Joint Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS'02)*, pages 613–620, Bologna, Italy, 2002.

[7] P. C. Fishburn. Utility theory. *Management Science*, 14:335–378, 1968.

[8] P. C. Fishburn. *Utility Theory for Decision Making.* John Wiley and Sons, Inc., 1970.

[9] P. Goodwin and G. Wright. *Decision Analysis for Management Judgement.* Chichester John Wiley & Sons, Ltd., New York, NY, 1999.

[10] J. M. Hammersley and D. C. Handscombe. *Monte Carlo Methods.* Wiley, New York, 1964.

[11] J. M. Hammersley and K. W. Morton. A new Monte Carlo technique: antithetic variates. In *Cambridge Phil. Soc.*, volume 52, pages 449–475, 1956.

[12] R. F. Hespos and P. A. Strassmann. Stochastic decision trees for the analysis of investment decisions. *Management Science*, 11(10):B244–B259, 1965.

[13] D. L. Keefer and S. E. Bodily. Three point approximations for continuous random variables. *Management Science*, 29(5):595–609, 1983.

[14] F. Keenan. The price is really right. *BusinessWeek Online*, March 31, 2003.

[15] R. L. Keeney and H. Raiffa. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs.* John Wiley and Sons, Inc., 1976.

[16] D. M. Kreps. *Notes on the Theory of Choice.* Westview Press, 1988.

[17] J. F. Magee. Decision trees for decision making. *Harvard Business Review*, 42 (July-August):126–139, 1964.

[18] A. Mas-Colell, M. D. Whinston, and J. R. Green. *Microeconomic Theory.* Oxford University Press, New York, NY, 1995.

[19] N. Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44(247):335–341, 1949.

[20] R. F. Meyer and J. W. Pratt. The consistent assessment and fairing of preference functions. *IEEE Systems Science and Cybernetics*, SSC-4:270–278, 1968.

[21] E. S. Pearson and J. W. Tukey. Approximating means and standard deviations based on distances between percentage points of frequency curves. *Biometrika*, 52(3-4):533–546, 1965.

[22] C. Preist, A. Byde, C. Bartolini, and G. Piccinelli. Towards agent-based service composition through negotiation in multiple auctions. In *AISB'01 Symp. on Inf. Agents for Electronic Commerce*, 2001.

[23] H. Raiffa. *Decision Analysis: Introductory Lectures on Choices under Uncertainty.* Addison-Wesley Publishing Company Inc., Massachusetts, USA, 1968.

[24] L. J. Savage. *The Foundations of Statistics.* Wiley, New York, USA, 1954.

[25] J. von Neumann and O. Morgenstern. *Theory of games and economic behaviour, 2nd ed.* Princeton University Press, Princeton NJ, USA, 1947.

[26] A. Wald. *Statistical Decision Functions.* Wiley, New York, 1950.