Developing Efficient Option Pricing Algorithms by Combinatorial Techniques

Tian-Shyr Dai* Yuh-Dauh Lyuu† Li-Min Liu‡

Abstract

How to price options efficiently and accurately is an important research problem. Options can be priced by the lattice model. Although the pricing results converge to the theoretical option value, the prices do not converge monotonically. Worse, for some options like barrier-options, the prices can oscillate significantly. Thus, large computational time may be required to achieve acceptable accuracy. The combinatorial techniques can be used to improve the performance for pricing a wide variety of options. This paper uses vanilla options, single-barrier options, double-barrier options, and lookback options as examples to show how to derive linear-time pricing algorithms by combinatorial techniques. These algorithms are shown to compare favorably against many other lattice algorithms, which takes at least quadratic time in computation.

Keywords: option pricing, lattice, combinatorial approach, exotic option.

1 Introduction

Options are financial derivatives that give their buyers the right but not the obligation to buy or sell the elementary financial instrument for the exercise price. The elementary financial instrument in this paper is assumed to be stock for convenience. With the rapid growth and deregulation of economies, many complex options have been structured to meet specific financial goals. Although financial innovations make the market efficient, they give rise to new problems in pricing these options efficiently and accurately.

How to assign a fair price to an option given a continuous-time stochastic process for the stock price has been investigated since 1900. Black and Scholes settle the pricing problem for the vanilla option [1]. Although an option must have a unique theoretical value, calculating that price may be intractable. Most of the options can not be evaluated analytically and must be priced by the numerical methods. Finding efficient and accurate numerical pricing methods is thus important.

The lattice method is a popular numerical method for pricing options. A lattice divides a certain time interval from time 0 to T into n discrete time steps and simulates the stock price discretely at each time step. Take a 4-time-step CRR lattice [4] in Fig. 1 as an example. The time interval between the option initial date to the maturity date is evenly divided into 4 time steps. The stock price at time step 0 is S_0. For any node at an arbitrary time step i represents a possible stock price at the i-th time step. From an arbitrary node with stock price S, the CRR lattice says that the stock price after one time step equals Su (the up move) with probability p and Sd (the down move) with probability $1-p$, where $d < u$ and $ud = 1$. Let σ denotes the volatility of the stock price, r denotes the risk-free rate. Then u and d are set to $e^{\sigma\sqrt{T/n}}$ and $e^{-\sigma\sqrt{T/n}}$, respectively. Note that $ud = 1$ in the CRR lattice model. The probability p is set to $(e^{rT/n} - d)/(u - d)$. Note that the stock price S resulting from j down moves and $i - j$ up moves from time step 0 equals $S_0 u^{i-j} d^j$ with probability $\binom{i}{j} p^{i-j}(1-p)^j$. This node located at time step i is denoted as $N(i, j)$ for simplicity.

The pricing results generated by the lattice method converge to the theoretical option value as $n \to \infty$ [6]. To calculate the pricing results, a naive lattice algorithm needs to calculate the option price for each node of the lattice. In other words, the time complexity of such a lattice algorithm is at least $O(n^2)$ since there are $O(n^2)$ nodes in a lattice. Note that pricing some complex options, like lookback options, needs higher time complexity. Even more, Chalasani et al. show that an option can be so defined that its pricing problem is #P-hard [3]. Besides, the lattice method also has some problems in convergence. For some options like lookback options, the pricing results converge very slowly. Worse, for some options like barrier-options, the pricing results can oscillate significantly [2]. Thus, a large n is required to achieve acceptable accuracy.

The combinatorial approach seems to be first suggested by Lyuu. He develops a linear time combinatorial algorithm for pricing single-barrier options on the CRR lattice [11] by taking advantage of the reflection principle. In fact, we find that some combinatorial tools, like the recurrence relation, the reflection principle, and the inclusion-exclusion principle, can be applied to improve the performance for pric-

*Department of Information and Finance Management, National Chiao-Tung University, Taiwan 300, ROC. E-mail:d88006@csie.ntu.edu.tw. Tel: 886-3-2653131. The author was supported by NSC grant 94-2213-E-033-024.

†Department of Finance and Department of Computer Science & Information Engineering, National Taiwan University. The author was supported in part by NSC grant 94-2213-E-002-088.

‡Department of Applied Mathematics, Chung Yuan Christian University. The author was supported in part by NSC grant 94-2213-E-033-031.

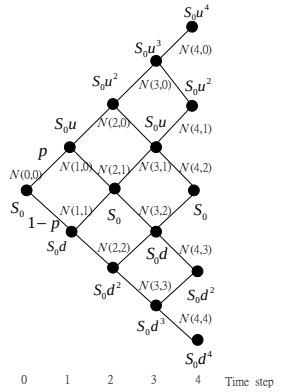

Figure 1: **The CRR Lattice.**

ing a wide variety of options on the CRR lattice. In this paper, we will use vanilla options, single-barrier options, double-barrier options, and lookback options as examples to demonstrate how these tools can improve the lattice pricing algorithms. We will show that all the resulting pricing algorithms runs in linear time. These proposed algorithms are at least an order faster than currently existing algorithms.

A simple survey for above mentioned options is given as follows. Black and Scholes derive an analytical solution for vanilla options [1]. A barrier option is an option whose payoff depends on whether the stock's price path ever touches certain price levels called the barriers. A single-barrier option is a barrier option with only one barrier, while a double-barrier option is a barrier option with two barriers. When the payoff functions of single-barrier options follow some simple forms, analytical pricing formulas can be obtained from [13]. The valuation of double-barrier options has been studied by [10]. However, no simple exact closed-form pricing formula for double-barrier options is available in the literature. The pricing formula can be expressed as an infinite series of cumulative normal distributions. Although truncation of this infinite series is necessary numerically, it can lead to large pricing errors according to [10]. The payoff of a lookback option depends on the extreme stock price achieved during a certain period of time. The analytical formulas for the lookback options are derived in [8].

The lattice model is a flexible way to price options since it can price an option with only nominal changes even when the option's payoff function is nonstandard, such as the power payoff function used in power options. In contrast, there may not be closed-form formulas for these options. However, the pricing results may converge slowly or even oscillate significantly. The oscillation phenomenon for pricing vanilla options by the lattice model is studied in [12]. Boyle and Lau study the oscillation problem for pricing single-barrier options [2]. To alleviate price oscillation problem, Ritchken provides a trinomial lattice model for pricing both single- and double-barrier options [14]. Figlewski and Gao provides the adaptive mesh model for pricing vanilla options and single-barrier options [7]. Although their algorithms successfully alleviate the price oscillation problem, their algorithms are not efficient since they all run in $O(n^2)$ time. In this paper, we will derive a $O(n)$ time combinatorial algorithm for pricing double-barrier options on the CRR lattice. This algorithm can be applied to improve the performance for pricing double-barrier options on the Bino-Trinomial lattice (see [5]), which is mainly composed of a CRR lattice. Numerical results show that the resulting approach can achieve the same level of accuracy with much less computational time than other lattice approaches.

An efficient lattice pricing algorithm for the lookback option is useful since the pricing results for lookback options converge very slowly. An efficient lattice algorithm for lookback options is thus important. The most efficient algorithm currently known runs in $O(n^2)$ time [9]. In this paper, we propose an $O(n)$ time combinatorial algorithm for pricing lookback options.

Our paper is organized as follows. Required mathematical and financial background knowledge are introduced in section 2. An $O(n)$ time algorithm for pricing double-barrier options is derived in section 3. In section 4, we develop $O(n)$ time pricing algorithms for lookback options. Numerical results are provided in section 5 to show how the combinatorial techniques improve the efficiency of the lattice pricing algorithms. Section 6 concludes the paper.

2 Preliminary

Payoff Functions for Options

A vanilla option gives the holder the right to buy or sell the stock for the exercise price X at the maturity date. A call option allows the option owner to buy the stock with price X at time T, while a put option allows the option owner to sell the stock with the same price at time T. Thus the payoff of a vanilla option at time T can be expressed as follows:

$$\max(\theta S(T) - \theta X, 0), \tag{1}$$

where $S(T)$ denotes the stock price at time T, θ equals to 1 and -1 for call and put options, respectively.

A barrier option is an option whose payoff depends on whether the stock's price path ever touches certain price levels called the barriers. A single-barrier option is a barrier option with only one barrier, says H. Assume that $H > S(0)$ for convenience. Define $S_{\sup} \equiv \sup_{0\le t\le T} S(t)$. The payoff of a single-barrier option at maturity date T is expressed as follows:

$$\text{Payoff} = \begin{cases} \max(\theta S(T) - \theta X, 0), & \text{if } S_{\sup} \ge H, \\ 0, & \text{otherwise.} \end{cases}$$

A double-barrier option is a barrier option with two barriers, says L and H. Assume that $L < S_0 < H$ for convenience. Define $S_{\inf} \equiv \inf_{0\le t\le T} S(t)$. The payoff

for a double barrier option at maturity date can be expressed as follows:

$$\text{Payoff} = \begin{cases} \max(\theta S(T) - \theta X, 0) & \text{if } S_{\sup} \geq H \text{ or } S_{\inf} \leq L, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

The payoff of a lookback option depends on the extreme stock's price during a certain time interval. The payoff function for a lookback option at the maturity date can be described as follows.

$$\text{Payoff} = \begin{cases} S(T) - S_{\inf} & \text{for a call option,} \\ S_{\sup} - S(T) & \text{for a put option.} \end{cases} \tag{3}$$

Pricing Options on the CRR Lattice

The theoretical value of an option is expressed as follows:

$$e^{-rT}\,\mathrm{E}\,(\text{Payoff})\,. \tag{4}$$

The dynamic programming technique can be applied when we price an option on the CRR lattice. The option value for each node is evaluated backwardly from time step n to time step 0. Take a vanilla call option as an example. Define the option value for node $N(i,j)$ as $V(i,j)$. Thus $V(n,j)$ is equal to $\max(S_0u^{n-j}d^j - X, 0)$. $V(i,j)$ can be calculated by the following formula:

$$V(i,j) \equiv e^{-rT/n} \times (pV(i+1,j) + (1-p)V(i+1,j+1))\,,$$

where $0 \leq i < n$. The pricing result is $V(0,0)$. Note that this naive pricing method takes $O(n^2)$ time since there are $O(n^2)$ nodes in an n-time-step CRR lattice.

The Recurrence Relation

The recurrence relation can help us to speed up computation. Take the vanilla call option mentioned above as an example. Since the probability to reach node $N(n,j)$ is $\binom{n}{j}p^{n-j}(1-p)^j$, the pricing result of the n-time-step CRR lattice can be derived by applying Eq. (4) as follows:

$$e^{-rT}\sum_{j=0}^{n}\binom{n}{j}p^{n-j}(1-p)^j \max(S_0u^{n-j}d^j - X, 0). \tag{5}$$

This formula can be evaluated in $O(n)$ time by the two following recurrence relations. Define C_j and D_j as $\binom{n}{j}p^{n-j}(1-p)^j$ and $S_0u^{n-j}d^j$, respectively. C_j and D_j can be expressed by the two following recurrence relations: $C_j = C_{j-1} \times \frac{(1-p)\times(n-j+1)}{p\times j}$, and $D_j = D_{j-1} \times \frac{d}{u}$. Note that both C_j and D_j can be evaluated in constant time when C_{j-1} and D_{j-1} are known. To obtain the pricing result, we may first calculate C_0 and D_0 and then evaluate each summation term in Eq. (5) in constant time by the recurrence relations mentioned above. Eq. (5) can be evaluated in $O(n)$ time since there are $n+1$ summation terms in this equation.

The Reflection Principle

The reflection principle can help us to efficiently count the number of paths that hit a specific price level before reaching a certain node at the n-th time step of the CRR lattice. This property is useful for pricing single-barrier options. We will derive a useful combinatorial formula with the help of the grid illustrated in Fig. 2. This grid reflects the structure of a CRR lattice. The x-coordinate denotes the time step of the CRR lattice, and the y-coordinate denotes the stock price level. To fit the stock price movement on the CRR lattice, each path in the grid can move from vertex (i,j) to vertex $(i+1,j+1)$ (the up move) or vertex $(i+1,j-1)$ (the down move). Now consider the following problem that is useful for pricing single-barrier options: How many paths starting form $(0,-a)$ (node A) will reach barrier H before arriving at $(n,-b)$ (node B)? Without loss of generality, we further assume that $a, b \geq 0$.

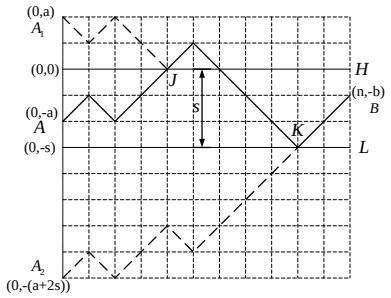

Figure 2: **Count the Number of Paths that Hit Barrier H (or L) by the Reflection Principle and the Inclusion-Exclusion Principle.**

Consider one such path, $\widehat{AJB}$, that hits barrier H at node J. We can reflect the path $\widehat{AJ}$ (marked by the solid curve) with respect to the H-axis to get $\widehat{A_1J}$ (marked by the dash curve). Each path from node A to node J maps to a unique path from node A_1 to node J. For example, the path $\widehat{AJ}$ maps to $\widehat{A_1J}$. Thus the number of paths from node A to node J equals to the number of paths from node A_1 to node J. By this observation, the reflection principle says that the number of paths staring from node A and hitting barrier H before reaching node B equals to the number of paths moving from node A_1 to node B. Assume that x up moves and y down moves are required to move from node A_1 to node B. Thus we have $x + y = n$ and $x - y = -a - b$. By solving these two equations, we have $x = \frac{n-a-b}{2}$. Thus the number of paths that reach H before arriving B is

$$\binom{n}{\frac{n-a-b}{2}} \quad \text{for even, non-negative } n-a-b \tag{6}$$

and zero otherwise.

Lyuu derives an $O(n)$ time combinatorial algorithm for pricing a single-barrier option with barrier H by taking advantage of Eq. (6) [11]. His work is sketched as follows. A CRR lattice is placed on a grid as illustrated in Fig. 3. Assume that barrier H is equal to S_0u^{n-2h} for convenience. The CRR lattice is placed so barrier H coincides with the x-axis. The single-barrier call option can be priced by taking the discounted expected payoff of the option at maturity date as in Eq. (4). We calculate the values contributed by theses nodes at the n-th time step. For convenience, we use "terminal nodes" to refer to these nodes. Two different kinds of terminal nodes are de-

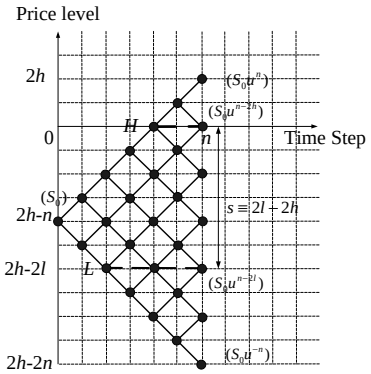

Figure 3: **Placing a CRR Lattice on a Grid (for Pricing both Single- and Double-Barrier Option).** The coordinate of the root node of the CRR lattice is $(0, 2h - n)$. s denotes the distance between H and L. The values in parentheses denote the stock prices.

scribed as follows. For a terminal node that is above barrier H (inclusive) like node A, all the paths that reach this node must hit barrier H. The option value contributed by these terminal nodes can be calculated as follows:

$$V \equiv \sum_{i=0}^{h} \binom{n}{i} p^{n-i}(1-p)^i \max(S_0 u^{n-i} d^i - X, 0).$$

For a terminal node that is below barrier H (exclusive) like node B, all the paths that reach this node before hitting barrier H can be efficiently computed by Eq. (6). The option value contributed by these terminal nodes can be calculated as follows:

$$V' \equiv \sum_{i=h+1}^{n} \binom{n}{2h-i} p^{n-i}(1-p)^i \max(S_0 u^{n-i} d^i - X, 0).$$

The value of the single barrier option priced by the CRR lattice is simply $V + V'$.

3 Double-Barrier Options

3.1 A Combinatorial Formula

We derive a combinatorial formula that can efficiently count the number of paths that hit one of some predetermined price levels before reaching a certain node at the n-th time step in a CRR lattice. This formula is helpful to derive an $O(n)$ algorithm for pricing double-barrier options. Consider the gird illustrated in Fig. 2. How many price paths starting form $(0, -a)$ (node A) will reach either barrier H or barrier L before arriving at $(n, -b)$ (node B)? Obviously, the reflection principle is not directly applicable. This problem can be solved by applying the reflection principle repeatedly and by the inclusion-exclusion principle.

Before solving this problem, a simplified problem is considered first: How many price paths moving form node A to node B will hit barrier H before one hit of barrier L? One such path may hit barrier H at J and barrier L at K. We can first reflect the path $\widehat{AJ}$ with respect to the H-axis to get $\widehat{A_1J}$. The reflection principle says that the number of paths staring from node A and hitting barrier H before reaching node B equals to the number of paths moving from node A_1 to node B. The reflection principle can be applied more than once. The curve $\widehat{A_1K}$ can be reflected with respect to the L-axis to obtain $\widehat{A_2K}$. By the reflection principle, the number of paths staring from node A_1 and hitting barrier L before reaching node B equals to the number of paths moving from node A_2 to node B. Thus the number of paths moving from node A to node B and reaching barrier H at least once before hitting the barrier L is equal to the number of paths moving from A_2 to B. Assume that x up moves and y down moves are required to move from node A_2 (with coordinate $(0, -(a+2s))$) to node B (with coordinate $(n, -b)$). Thus we have $x + y = n$ and $x - y = a - b + 2s$. We get $x = \frac{n+a-b+2s}{2}$ by solving the above two equations. So the answer to this simplified problem mentioned above is

$$\binom{n}{\frac{n+a-b+2s}{2}} \quad \text{for even, non-negative } n + a - b. \tag{7}$$

Note that a path counted by Eq. (7) may hit L first before hitting H. The point is that among the hits, one hit of barrier H must appear before one hit of barrier L.

The problem of counting the number of paths that will hit either barrier H or barrier L before arriving at node B is now within reach. First, a function f is constructed to map each path to a string. This string contains the information about the barrier hitting sequence. For example, $f(\widehat{AB}) = HHL$ since the path $\widehat{AB}$ hits the barrier H twice before hitting the barrier L. Next, we define α_i as the set of paths whose f value *contains* $\overbrace{H^+L^+H^+\cdots}^{i}$ with $i \geq 1$. L^+ and H^+ denote a sequence of Ls and Hs, respectively. Obviously, the path $\widehat{AB}$ belongs to both set α_1 and set α_2. Similarly, define β_i as the set of paths whose f value contains $\overbrace{L^+H^+L^+\cdots}^{i}$ with $i \geq 1$. Thus the path $\widehat{AB}$ belongs to set β_1. The number of elements in set α_i and β_i can be calculated by repeatedly using the reflection principle mentioned in last paragraph. The number of elements in each set is listed as follows:

$$|\alpha_i| = \begin{cases} \binom{n}{\frac{n+a+b+(i-1)s}{2}} & \text{for odd } i \\ \binom{n}{\frac{n+a-b+is}{2}} & \text{for even } i \end{cases} \tag{8}$$

$$|\beta_i| = \begin{cases} \binom{n}{\frac{n-a-b+(i+1)s}{2}} & \text{for odd } i \\ \binom{n}{\frac{n-a+b+is}{2}} & \text{for even } i \end{cases}$$

Note that each path that hits the barrier may belong to more than one set. For example, $\widehat{AB}$ in Figure 2 belongs to set α_1, α_2, and β_1. The inclusion-exclusion principle is then applied to calculate the exact number of paths that moves from A (with coordinate $(0, -a)$) to B (with coordinate $(n, -b)$) and that hit either barrier H or L at least once as follows:

$$\mathbf{N}(a, b, s) = \sum_{i=1}^{\lceil \frac{n}{s} \rceil} (-1)^{i+1} (|\alpha_i| + |\beta_i|). \tag{9}$$

3.2 Linear Time Algorithm

We first put the CRR lattice on a grid as displayed in Fig. 3. Assume that the barriers H and L equal to $S_0u^{n-h}d^h(= S_0u^{n-2h})$ and $L = S_0u^{n-l}d^l(= S_0u^{n-2l})$, respectively. The exercise price X satisfies the following equality $S_0u^{n-a}d^a \leq X < S_0u^{n-a+1}d^{a-1}$ for some integer a.

Next we analyze the option value contributed by a price path that reaches node $N(n,j)$. The probability for this price path is $p^{n-j}(1-p)^j$. The payoff at node $N(n,j)$ is $\max(S_0u^{n-j}d^j - X, 0)$. Thus the value contributed by this price path is

$$p(j) \equiv e^{-rT}p^{n-j}(1-p)^j \max(S_0u^{n-j}d^j - X, 0), \qquad (10)$$

if this price path hits either barrier H or barrier L. Furthermore, the number of price paths that reach node $N(n,j)$ is $\binom{n}{j}$. If node $N(n,j)$ is above the barrier H (inclusive) or below the barrier L (inclusive), the value contributed by this node $\binom{n}{j}p(j)$. This is because all the price paths that reach this node must also hit barrier H or L.

The combinatorial algorithm for pricing the double-barrier call option is now within reach. First, a "combination table" is built for storing the value of $\binom{n}{k}$, where $0 \leq k \leq n$. Thus $\binom{n}{k}$ can be evaluated in constant time by looking up this table. Next, two non-degenerate cases are considered as follows.

Case 1. $L < X < H$:

The option value can be decomposed into two following parts: (1) the value contributed by the terminal nodes between X and H (exclusive), and (2) the value contributed by the terminal nodes above H (inclusive).

The first part of the option value can be computed by accumulating the values contributed by the terminal nodes, says $N(n,j)$ $(a > j > h)$, between X and H. The number of paths that reach one of the barriers before reaching $N(n,j)$ is $\mathbf{N}(n-2h, 2j-2h, 2l-2h)$. The value contributed by such a path is $p(j)$ (see Eq. (10)). Therefore, the value contributed by node $N(n,j)$ is $\mathbf{N}(n-2h, 2j-2h, 2l-2h)p(j)$. The sum of the values contributed by the terminal nodes between X and H is

$$P_0 \equiv \sum_{j=h+1}^{a-1} \mathbf{N}(n-2h, 2j-2h, 2l-2h)p(j). \qquad (11)$$

The second part of the option value is computed by accumulating the values contributed by the terminal nodes, says $N(n,i)$ $(0 \leq i \leq h)$, above the barrier H (inclusive). The value contributed by $N(n,i)$ is $\binom{n}{i}p(i)$. Therefore, the second part of the option value is

$$P_1 \equiv \sum_{i=0}^{h} \binom{n}{i} p(i) = e^{-rT} \sum_{i=0}^{h} \binom{n}{i} p^{n-i}(1-p)^i \max(S_0u^{n-i}d^i - X, 0). \qquad (12)$$

We conclude that the value of a double-barrier call option is $P_0 + P_1$ under the condition $L < X < H$.

Case 2. $X \leq L$:

The option value can be decomposed into three following parts: (1) the option value contributed by the terminal nodes between L (exclusive) and H (exclusive), (2) the option value contributed by the terminal nodes above H (inclusive), and (3) the option value contributed by the terminal nodes between L (inclusive) and X.

The first part of the option value is computed by accumulating the values contributed by the terminal nodes between L and H as follows:

$$P_0' \equiv \sum_{j=h+1}^{l-1} \mathbf{N}(n-2h, 2j-2h, 2l-2h)p(j). \qquad (13)$$

The second part of the option value is computed by accumulating the values contributed by the terminal nodes above the barrier H (inclusive) as listed in Eq. (12). The third part of the option value is computed by accumulating the option value contributed by the terminal nodes, says $N(n,k)$ $(l \leq k < a)$, between L (inclusive) and X. This part of the option value is

$$P_1' = \sum_{k=l}^{a} \binom{n}{k} p(k) = e^{-rT} \sum_{k=l}^{a} \binom{n}{k} p^{n-k}(1-p)^k \max(S_0u^{n-k}d^k - X, 0). \qquad (14)$$

Thus, the value of a double-barrier call option is $P_0' + P_1 + P_1'$ under the condition $X \leq L$.

Finally, we prove that our algorithm runs in $O(n)$ time. Our pricing algorithm can be divided into three parts. The first part denotes the construction of the combination table, the second part denotes the evaluation of Eq. (12) and (14), and the last part denotes the evaluation of Eq. (11) (in case 1) or Eq. (13) (in case 2). We show that our algorithm runs in $O(n)$ time by showing that these three parts can be computed in $O(n)$ time.

First, recall that the combination table is a table that stores the value of $\binom{n}{k}$, where $0 \leq k \leq n$. All the fields in the combination table could be filled in $O(n)$ time by the recurrence equation $\binom{n}{k} = \binom{n}{k-1} \times (n-k+1) \div k$. Next, Eq. (12) and (14) can also be calculated in $O(n)$ time by the recurrence relations used to calculate Eq. (5). Finally, we show that both Eq. (11) and Eq. (13) can be calculated in $O(n)$ time. Note that the terms $|\alpha_i|$ and $|\beta_i|$ defined in Eq. (8) can be represented in $\binom{n}{k}$ form, so $|\alpha_i|+|\beta_i|$ can be evaluated in constant time by looking up the combination table. Thus $N(a,b,s)$ defined in Eq. (9) can be solved in $O(\lceil \frac{n}{s} \rceil)$ time. In both Eq. (11) and Eq. (13), $N(n-2h, 2j-2h, 2l-2h)$ would be evaluated less than $l-h$ times.Consequently, it takes about $O(\frac{n}{2(l-h)}(l-h)) \approx O(n)$ time to evaluate Eq. (11) and Eq. (13). Thus the option value $P_0 + P_1$ (in case 1) or $P_0' + P_1 + P_1'$ (in case 2) can be calculated in $O(n)$ time. So we conclude that our pricing algorithm runs in $O(n)$ time.

4 Pricing Lookback Options

An efficient combinatorial algorithm for pricing a lookback option on a CRR lattice model is derived by repeatedly applying the reflection principle (see Eq. (6)) in this section. Recall that the payoff of a lookback option depends on the extreme stock's price

occurred from time 0 to time T (see Eq. (3)). The value contributed by a price path at the maturity date therefore depends only on the terminal node reached by the price path and the extreme stock price of the price path. We can divide all possible price paths into some groups by these two factors. We will first show how to count the number of price paths in each group. This helps us to calculate the value contributed by each group. The value contributed by a terminal node N is calculated by summing the values contributed by the groups consisted of the paths reaching N. Finally, the value of a lookback option is obtained by summing the values contributed by all the terminal nodes.

A simple example for pricing a lookback option on a 4-time-step CRR lattice illustrated in Fig. 1 is given as follows. All possible price paths for this 4-time-step CRR lattice is divided into 9 groups as illustrated in Fig. 4. Take the terminal node $N(4,2)$ as an example. $6\left(\equiv \binom{4}{2}\right)$ price paths reach node $N(4,2)$, and these price paths can be divided into 3 different groups by the minimal stock prices of these price paths. Consider the group with minimal stock price S_0d. The number of paths in this group can be computed by applying Eq. (6) twice. First, we calculate the number of paths that passing through $N(0,0)$ and $N(4,2)$ and that hitting the price level S_0d by Eq. (6). The answer is $4\left(\equiv \binom{4}{1}\right)$. But some of these four paths may have minimal prices lower than S_0d and these price paths must hit the price level S_0d^2. We can apply Eq. (6) again to compute the number of price paths and the answer is $1\left(\equiv \binom{4}{0}\right)$. Thus we can conclude that there are $3\left(\equiv \binom{4}{1}-\binom{4}{0}\right)$ price path in the group with minimal stock price S_0d. Note that the payoff and the probability of each price path in this group is S_0-S_0d and $p^2(1-p)^2$, respectively. The value contributed by this group is $e^{-rT}p^2(1-p)^2\left(\binom{4}{1}-\binom{4}{0}\right)(S_0-S_0d)$.

Next, we calculate the values contributed by a terminal node by summing the values contributed by the groups consisted of the price paths reaching this terminal node. Take $N(4,2)$ in Fig. 4 as an example. The value contributed by this node is simply

$$e^{-rT}p^2(1-p)^2\left[\binom{4}{0}\left(S_0-S_0d^2\right)+\left(\binom{4}{1}-\binom{4}{0}\right)(S_0-S_0d)+\left(\binom{4}{2}-\binom{4}{1}\right)(S_0-S_0)\right].$$

This above formula can be further rewritten by grouping the positive terms and the negative terms as follows:

$$e^{-rT}p^2(1-p)^2S_0\left[\binom{4}{0}+\left(\binom{4}{1}-\binom{4}{0}\right)+\left(\binom{4}{2}-\binom{4}{1}\right)\right]$$
$$-e^{-rT}p^2(1-p)^2\left[\binom{4}{0}S_0d^2+\left(\binom{4}{1}-\binom{4}{0}\right)S_0d+\left(\binom{4}{2}-\binom{4}{1}\right)S_0\right].$$

For convenience, the former term and the latter term are called the positive part and the negative part of the value contributed by $N(4,2)$, respectively.

Next, we show two recurrence relations on the positive part and the negative part of the values contributed by the terminal nodes. Let the value contributed by node $N(n,i)$ as $E(i)$. $E(i)$ can be divided into two parts: the positive part (denoted as $F(i)$) and the negative part (denoted as $G(i)$). Assume that n is a even number and $p_x=\frac{1-p}{p}$, then $F(i)$ and $G(i)$ can be expressed by the following recurrence relation:

$$F(0)=e^{-rT}p^n\binom{n}{0}S_0u^n,\qquad G(0)=e^{-rT}p^n\binom{n}{0}S_0,$$

$$\mathrm{F(i)}=\begin{cases}F(i-1)p_x/u^2+e^{-rT}p^{n-i}(1-p)^i\left(\binom{n}{i}+\binom{n}{i-1}\right)S_0u^{n-2i},\\ \text{if } 0<i\le n/2,\\ F(i-1)p_x/u^2-e^{-rT}p^{n-i}(1-p)^i\left(\binom{n}{n-i+1}-\binom{n}{n-i}\right)S_0u^{n-2i},\\ \text{if } i>n/2,\end{cases}$$

$$\mathrm{G(i)}=\begin{cases}G(i-1)p_x/u+e^{-rT}p^{n-i}(1-p)^i\left(\binom{n}{i}+\binom{n}{i-1}\right)S_0,\\ \text{if } 0<i\le n/2,\\ G(i-1)p_x/u-e^{-rT}p^{n-i}(1-p)^i\left(\binom{n}{n-i+1}-\binom{n}{n-i}\right)S_0d^{2i-n-1},\\ \text{if } i>n/2.\end{cases}$$

The $O(n)$ time combinatorial pricing algorithm for lookback options is now within reach. The value of a lookback option can be expressed as follows:

$$\sum_{i=0}^{n}E(i)=\sum_{i=0}^{n}\left(F(i)-G(i)\right). \tag{15}$$

We can evaluate it in $O(n)$ time if each term in the summation $(F(i)-G(i))$ can be calculated in constant time. Two tables are constructed to achieve this goal. A combination table is constructed to store the values of $\binom{n}{i}$, where $0\le i\le n$. A probability table is constructed to store the values of $p^i(1-p)^{n-i}$, where $0\le i\le n$. All the fields in both tables can be filled in $O(n)$ time by the recurrence relations $\binom{n}{k}=\binom{n}{k-1}\times(n-k+1)\div k$ and $p^i(1-p)^{n-i}=p^{i-1}(1-p)^{n-i+1}\times p\div(1-p)$, respectively. Note that for any arbitrary i, both $\binom{n}{i}$ and $p^i(1-p)^{n-i}$ can be obtained in constant time by looking up the two tables mentioned above. $F(i)$ and $G(i)$ can now be calculated in constant time by the recurrence relations listed above if $F(i-1)$ and $G(i-1)$ are known. Thus we conclude that Eq. (15) can be calculated in $O(n)$ time.

5 Experimental results

Pricing barrier options on a CRR lattice will result in significant oscillation. To alleviate oscillation, Ritchken proposes a novel trinomial lattice model [14]. His model is not quite efficient since it runs in $O(n^2)$ time. Dai and Lyuu propose another new lattice model, BTT model, which is mainly composed of a CRR lattice [5]. Thus pricing barrier options on the BTT model can be done in $O(n)$ time by taking advantage of the combinatorial algorithms mentioned in this paper. The numerical results for pricing a double-barrier options is illustrated in Fig. 5. The accurate value is about 10.1993. It costs the BTT model about 0.08 seconds to converge to 10.1993, but it costs the Ritchken's trinomial lattice model about 5 seconds to converge to the same value.

Pricing lookback options on a lattice would suffer from slow convergence problem. Finding an algorithm that can efficiently handle large n is thus important. Hull provides an $O(n^2)$ time algorithm to price the lookback option on the CRR lattice [9]. This paper provides a combinatorial algorithm that runs in

	S_0d^4	S_0d^3	S_0d^2	S_0d	S_0
$N(4,0)$	0	0	0	0	$\binom{4}{0}\left(S_0u^4 - S_0\right)$
$N(4,1)$	0	0	0	$\binom{4}{0}\left(S_0u^2 - S_0d\right)$	$\left(\binom{4}{1} - \binom{4}{0}\right)\left(S_0u^2 - S_0\right)$
$N(4,2)$	0	0	$\binom{4}{0}\left(S_0 - S_0d^2\right)$	$\left(\binom{4}{1} - \binom{4}{0}\right)(S_0 - S_0d)$	$\left(\binom{4}{2} - \binom{4}{1}\right)(S_0 - S_0)$
$N(4,3)$	0	$\binom{4}{0}\left(S_0d^2 - S_0d^3\right)$	$\left(\binom{4}{1} - \binom{4}{0}\right)\left(S_0d^2 - S_0d^2\right)$	0	0
$N(4,4)$	$\binom{4}{0}\left(S_0d^4 - S_0d^4\right)$	0	0	0	0

Figure 4: **Node Contribution Table for a 4-Time-Step Lattice.** The x-axis denotes the lowest price level reached by the price paths, and the y-axis denotes the terminal node reached by the price paths. Each field in the table denotes the product of the number of price paths and the payoff.

	Ritchken		BTT	
Time(sec)	n	Value	n	Value
0.054	500	10.2020	5244	10.1997
0.083	700	10.2010	10142	10.1993
5.125	5000	10.1993		
Accurate Value 10.1993				

Figure 5: **Pricing a Double-Barrier Option.** The initial stock price is 95, the exercise price is 100, the risk-free rate is 10%, the volatility of the stock price is 25%, the time to maturity is 1 year, and the two barriers are 140 and 90, respectively. The accurate value is computed by the BTT model by setting the number of time steps as 20000.

n	Value	Time (in seconds)	
		Hull	Combinatorics
3000	23.997554	11.313	0.005
5000	24.043836	31.687	0.008
Accurate Value:24.203853			

Figure 6: **Running-Time Comparison for Pricing Lookback Options.** The initial stock price is 100, the risk-free rate is 6%, the volatility is 30%, and the time to maturity is 1 year. "Hull" denotes the $O(n^2)$ time algorithm in [9]. "Combinatorics" denotes the combinatorial algorithm in this paper.

$O(n)$ time. The running times for both algorithm are in Fig. 6. Obviously, our algorithm runs faster than Hull's one.

6 Conclusions

Combinatorial methods have found wide applicability in many fields. This paper extends their use in improving the performance for pricing a wide variety of options. This paper describes how to derive $O(n)$ time combinatorial pricing algorithm for vanilla options, single-barrier options, double-barrier options, and lookback options. These algorithms are shown to compare favorably against many other lattice methods, which takes at least quadratic time in computation.

References

[1] Black, F., Scholes, M. "The pricing of options and corporate liabilities." *J. Political Econom.,* 81 (1973), pp. 637–659.

[2] Boyle P., and Lau, S. "Bumping Against the Barrier with the Binomial Method," *J. of Derivatives*, 1 (1994), pp. 6–14.

[3] P. Chalasani, S. Jha, and I. Saias, "Approximate Option Pricing", *Algorithmica*, 25 (1999), pp. 2–21.

[4] J. Cox, S. Ross, and M. Rubinstein, "Option Pricing: A Simplified Approach", *J. of Financial Econom.* 7 (1979), pp. 229–264.

[5] Dai, T.-S., and Lyuu, Y.-D., "The Binotrinomial Tree Model. " Manuscript, February, 2006.

[6] Duffie, D. *Dynamic Asset Pricing Theory.* 2nd ed. Princeton, NJ: Princeton University Press, 1996.

[7] Figlewski, S., and Gao, B. "The Adaptive Mesh Model: A New Approach to Efficient Option Pricing." *J. Financial Econom.*, 53 (1999), pp. 313–351.

[8] Goldman, B., H. Sosin, and M. A. Gatto. "Path-Dependent Options: Buy at the Low, Sell at the High." *J. Finance*, 34 (1979), pp. 1111-1127.

[9] Hull, J. *Options, Futures, and Other Derivatives.* 5th ed. Englewood Cliffs, New Jersey: Prentice-Hall, 2003.

[10] Luo, L. "Various Types of Double Barrier Options." *J. Comput. Finance*, 4 (2001), pp. 125–138.

[11] Lyuu, Y.-D. "Very Fast Algorithms for Barrier Option Pricing and the Ballot Problem." *J. Derivatives,* 5 (1998), pp. 68–79.

[12] Omberg, E. "A Note on the Convergence of Binomial-Pricing and Compound-Option Models." *J. Finance,* (1987), pp. 463–469.

[13] Reiner, E. and Rubinstein, M. "Breaking Down the Barriers." *Risk,* 4 (1991), pp. 28–35.

[14] Ritchken, P "On Pricing Barrier Options". *J. Derivatives* 3 (1995), pp. 19–28.