An Exact Subexponential-Time Lattice Algorithm for Asian Options

Tian-Shyr Dai[*]   Yuh-Dauh Lyuu[†]

Abstract

Asian options are path-dependent derivatives. How to price them efficiently and accurately has been a long-standing research and practical problem. Asian options can be priced on the lattice. But only exponential-time algorithms are currently known if such options are to be priced on a lattice without approximation. Although efficient approximation methods are available, most of them lack accuracy guarantees. This paper proposes a novel lattice for pricing Asian options. The resulting exact pricing algorithm runs in subexponential time. This is the first exact lattice algorithm to break the exponential-time barrier. Because this lattice converges to the continuous-time stock price process, the proposed algorithm is guaranteed to converge to the desired continuous-time option value.

1 Introduction

Path-dependent derivatives are derivatives whose payoff depends nontrivially on the price history of a stock. Path-dependent derivatives play an important role in financial markets. Unfortunately, some of them are known to be difficult to price in terms of speed and/or accuracy, the Asian option being the most prominent example. An Asian option is an option whose payoff depends on the arithmetic average price of the stock. It is useful for hedging transactions whose cost is related to the average price of the stock (such as crude oil). Its price is also less subject to price manipulation; hence the average-price feature is popular in many thinly-traded markets.

There are no exact simple closed-form solutions for Asian option prices. Approximate closed-form solutions are suggested in [17, 21, 24]. Geman and Yor derive an analytical expression for the Laplace transform of the Asian call in [13]. Numerical inversion of this transform is considered in [12] and [23]. Some inversion algorithms based on the Euler and Post-Widder methods can be found in [1]. Most closed-form formulas lack accuracy guarantees. Indeed, some produce large pricing errors in extreme cases [11].

Since no simple closed-form solutions exist yet for the Asian option, the development of efficient numerical algorithms becomes critical. First, there are the popular Monte Carlo and quasi-Monte Carlo methods [3, 4, 5, 16]. Both the Monte Carlo approach and the analytical approach suffer from the inability to price American-style Asian options without bias. (European-style and American-style options will be defined later. Basically, European-style options do not allow early exercise, whereas American-style options do.) Recently, a least-squares Monte Carlo approach to address this problem is proposed [18]. Another problem of Monte Carlo method is that it is inefficient and the result is probabilistic. An alternative approach is the lattice and the related discretized partial-differential-equation method. This approach can handle early exercise. A lattice consists of nodes and edges connecting them. The lattice divides a time interval into n equal time steps. If the stock price process simulated by the lattice converges to the continuous-time stock price process as $n \to \infty$, then the option value priced on this lattice by the so-called backward induction (to be introduced later) also converges to desired continuous-time option value [9].

The difficulty in pricing Asian options is evidenced by the fact that only exponential-time lattice pricing algorithms are known if approximation is not used in backward induction. We call a lattice algorithm exact if it does not adopt approximation in backward induction. (In this paper, a polynomial-time algorithm means that the running time is polynomial in n. The same convention is adopted for exponential-time and subexponential-time algorithms.) To reduce the complexity, Hull and White propose an efficient approximation algorithm in [14]. Their influential paradigm is followed by many such as [15, 22, 25]. The major prob-

lem with the Hull-White paradigm is the convergence guarantee. Specifically, assume that the pricing error between the approximate and the exact algorithms is denoted by e. An approximation lattice algorithm converges to the desired option value if $e \to 0$ as $n \to \infty$. Unfortunately, most Hull-White-type algorithms lack such convergence guarantees. In fact, improper approximations result in divergence or convergence to a wrong value [10]. Aingworth *et al.* and Dai *et al.* provide efficient approximation algorithms that have convergence guarantees for the European-style Asian options [2, 7]. Unfortunately, neither has convergence guarantees for American-style Asian options.[1]

Another paradigm due to Dai and Lyuu [8] constructs a lattice called the multiresolution lattice. It converges to the desired option value as it is exact. Numerical results show that the method works for n up to 160 under some cases. Note that an exponential-time algorithm cannot work with an n that high. There are two drawbacks with the multiresolution lattice. First, no proof is offered to show that this algorithm runs in subexponential time. Second, the multiresolution lattice is constructed by an ad hoc local search, and no proof exists to guarantee that the lattice can always be constructed successfully.

Our paper will propose the first exact pricing algorithm that provably breaks the exponential-time barrier. Here is the rough overview. We construct a new lattice composed of integral stock prices. The stock price sum of a price path on the lattice is therefore an integer. This critical property is used to ensure that no approximations are needed in backward induction, making the algorithm exact and convergent. We also prove that the time complexity is subexponential in n.

The homogeneous property of the option value is crucial to the construction of the said lattice. Suppose we multiply the stock prices by a constant K before pricing the option. The homogeneous property says that this option value divided by K gives the originally desired option value [20]. In our paper, a K is found that guarantees that an integral lattice can be constructed. To ensure that the pricing results based on our lattice converge to the desired continuous-time option value, we have to make sure that our lattice converges to the continuous-time stock price process. The conditions are that the lattice should match the mean and the variance of the stock price process at each time step [9]. It will be proved that an integral price that satisfies the above conditions exists for every node on the lattice.

The paper is organized as follows. The mathemat-

ical model is described in section 2. We will review how a lattice is constructed and how the Asian option is priced on the lattice in section 3. In section 4, we will show how to construct the proposed lattice. Rigorous proofs given in section 5 shows that our pricing algorithm is the first exact pricing algorithm that breaks the exponential-time barrier. Section 6 concludes that paper.

2 Modeling and Definitions

Assume that an Asian option initiates at 0 (in years) and matures at T (in years). Define $S(t)$ as the price of the stock at year t. The stock price is assumed to follow the continuous-time log-normal diffusion process:

$$(2.1) \qquad S(t + dt) = S(t)\exp[(r - 0.5\sigma^2)dt + \sigma dW_t],$$

where W_t is the standard Wiener process, r is the risk-free interest rate per annum, and σ is the annual volatility. It is useful to think of dW_t as normally distributed with mean 0 and variance dt. In the discrete-time approximation, the time between year 0 and year T is partitioned into n time steps. The length of each time step Δt is equal to T/n. For convenience, all the time notations in this paper are expressed in terms of the number of time steps unless stated otherwise. Let S_i denote the stock price at (discrete) time i, which corresponds to $S(i\Delta t)$ in the continuous-time model. Define the sum of a partial price path started at time 0 and ended at time j, $S_0 \to S_1 \to \cdots \to S_j$, as $\sum_{i=0}^{j} S_i$. We call this sum prefix sum. The payoff of an Asian option depends on the average of the stock prices. The price average is defined as

$$A_{\mathrm{avg}}(i) \equiv \frac{S_0 + S_1 + \cdots + S_i}{i + 1}.$$

The payoff for a European-style Asian option at maturity date is

$$(2.2) \quad \text{exercise value} = \begin{cases} A_{\mathrm{avg}}(n) - X, & \text{for a call,} \\ X - A_{\mathrm{avg}}(n), & \text{for a put,} \end{cases}$$

where X is called the exercise price. Our task is to compute $e^{-nr\Delta t}E(\max(A_{\mathrm{avg}}(n) - X, 0))$ and $e^{-nr\Delta t}E(\max(X - A_{\mathrm{avg}}(n), 0))$ in such a way that the value converges to the continuous-time limits $e^{-rT}E[\max(\frac{1}{T}\int_0^T S(t)\,dt - X, 0)]$ and $e^{-rT}E[\max(X - \frac{1}{T}\int_0^T S(t)\,dt, 0)]$, respectively, as n increases.

An American-style option gives the owner the right to exercise the option before maturity, thus an earlier payoff. The exercise value for an American-style option at time i is

$$(2.3) \quad \text{exercise value} = \begin{cases} A_{\mathrm{avg}}(i) - X, & \text{for a call,} \\ X - A_{\mathrm{avg}}(i), & \text{for a put.} \end{cases}$$

---

[1]Aingworth *et al.* give an incorrect convergence proof for their American-style Asian option pricing algorithm [7].

An option will be exercised early by the owner if the option's continuation value (the value to hold the option) is smaller than its exercise value.

3 Preliminaries

Before introducing our new lattice, we first review how a lattice is constructed so that it converges to the log-normal price process. Next we will show how to price an Asian option on a lattice. This part will demonstrate why the typical exact pricing algorithm explodes exponentially and why the approximation approach of Hull and White [14] is problematic. Finally, we will review the ideas of the integral lattice that will be useful later.

3.1 How To Construct a Lattice We use the well-known Cox-Ross-Rubinstein (CRR) binomial lattice [6] to illustrate how a lattice is constructed in principle. A 3-time-step CRR binomial lattice is illustrated in Fig. 1. At each time step, the stock price S can either become Su—the up move—with probability P_u or Sd—the down move—with probability $P_d \equiv 1 - P_u$. The relation

$$(3.4) \qquad ud = 1$$

is enforced by the CRR binomial lattice. The logarithmic stock price mean (μ) and variance (Var) one time step from now are derived from Eq. (2.1) as

$$(3.5) \qquad \mu \equiv (r - 0.5\sigma^2)\Delta t,$$
$$(3.6) \qquad \text{Var} \equiv \sigma^2 \Delta t.$$

To make sure that the lattice converges to the continuous-time stock price process, the mean and the variance of the logarithmic price process should be calibrated by the matching those of the lattice and those of the continuous-time model:

$$(3.7) \qquad P_u \ln u + P_d \ln d = \mu,$$
$$(3.8) \quad P_u(\ln u - \mu)^2 + P_d(\ln d - \mu)^2 = \text{Var}.$$

Note that

$$(3.9) \qquad P_u + P_d = 1.$$

The 4 parameters (P_u, P_d, u, and d) are uniquely obtained by solving Eqs. (3.4), (3.7)–(3.9). The branching probabilities P_u and P_d should be between 0 and 1 to meet the no-arbitrage requirements. In the CRR lattice, this demand can always be met by suitably increasing n [19].

If each node in a lattice can branch to ℓ nodes at the next time step, we call it an ℓ-nomial lattice. The above idea can be applied to construct an ℓ-nomial lattice.
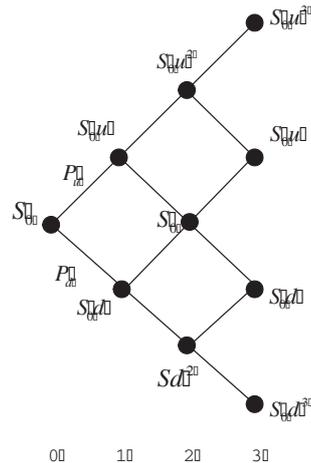

Figure 1: **A 3-time-Step CRR Binomial Lattice.**

Note that 2ℓ degrees of freedoms are provided by an ℓ-nomial lattice. They include ℓ price multiplicative factors (like u and d in the CRR binomial lattice) and ℓ branching probabilities (like P_u and P_d in the CRR binomial lattice). These branching probabilities must be between 0 and 1 to meet the no-arbitrage requirements. We need 2ℓ independent equations to determine these 2ℓ variables uniquely. The calibration of mean and variance gives 2 equations. The branching probabilities sum to 1, giving another one. Additional $2\ell - 3$ equations must be added. For example, Eq. (3.4) is the extra equation used in the CRR binomial model. This paper will construct a trinomial lattice ($\ell = 3$).

3.2 Pricing Asian Options on a Trinomial Lattice We now show how to price an Asian call with a trinomial lattice. In a trinomial lattice, each node can branch to three successor nodes in next time step. Let $S_{i,j}$ denotes $(j + 1)$th largest stock price of the nodes at time i. A 2-time-step trinomial tree is illustrated in Fig. 2. Take the root node as an example. Its stock price is $S_{0,0}$. The stock price can move upward to $S_{1,0}$ with probability P_u, move flatly to $S_{1,1}$ with probability P_m, and move downward to $S_{1,2}$ with probability P_d. The branching probabilities will vary for different nodes. There are $2\ell + 1$ nodes at time ℓ.

Note that the option value of a partial price path is influenced by the prefix sum of this partial price path. To price an Asian option exactly, enough number of states are required at each node to keep the option values corresponding to different prefix sums. The option value at time n can be calculated by Eq. (2.2). Define $V(S, C)$ as the option value whose corresponding prefix sum is C and whose stock price is S. Again, let
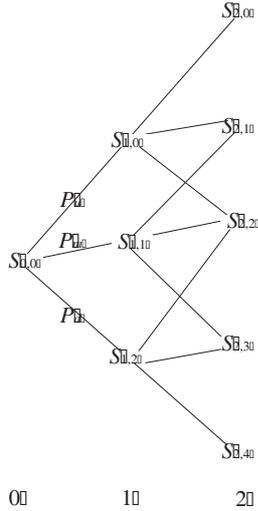
Figure 2: **A 2-Time-Step Trinomial Lattice.**
The initial stock price is $S_{0,0}$. $S_{i,j}$ denotes the $(j+1)$th largest stock price at time i. P_u, P_m, and P_d denote the branching probabilities, respectively.

P_u, P_m, and P_d denote the branching probabilities of the node with stock price $S_{i,j}$. The backward induction formulae for the European-style and American-style options are

$$
\begin{aligned}
\mathrm{V}(S_{i,j}, C) &= e^{-r\Delta t}\left[P_u \mathrm{V}(S_{i+1,j}, C + S_{i+1,j}) + \right.\\
&\qquad P_m \mathrm{V}(S_{i+1,j+1}, C + S_{i+1,j+1}) + \\
&\qquad \left. P_d \mathrm{V}(S_{i+1,j+2}, C + S_{i+1,j+2})\right]
\end{aligned}
$$
(3.10)

and

$$
\begin{aligned}
\mathrm{V}(S_{i,j}, C) &= \max\left(e^{-r\Delta t}\left[P_u \mathrm{V}(S_{i+1,j}, C + S_{i+1,j}) + \right.\right.\\
&\qquad P_m \mathrm{V}(S_{i+1,j+1}, C + S_{i+1,j+1}) + \\
&\qquad \left.\left. P_d \mathrm{V}(S_{i+1,j+2}, C + S_{i+1,j+2})\right], E\right),
\end{aligned}
$$

respectively. The exercise value E in the latter formula is defined in Eq. (2.3). The above formulae can be applied in a backward fashion from time n to time 0. The root node thus gives the desired option value. The backward-induction procedure implies that the overall running time is proportional to the total number of prefix sums on the lattice. Unfortunately, the number of price paths grows exponentially in n, giving as many prefix sums. This makes an exact pricing algorithm explode exponentially.

Hull and White suggest an efficient algorithm by limiting the number of prefix sums at each node to some manageable magnitude k [14]. It then resorts to interpolation as an approximation scheme in backward induction. This approach is efficient, with a running time of $O(kn^2)$. However, it is problematic since interpolation errors are introduced and accumulated at each time step. The Hull-White approach therefore may not converge to the desired option value as $n \to \infty$.

Aingworth *et al.* [2] and Dai *et al.* [7] also use interpolation in their approximation algorithms. Their algorithms provide convergence guarantees for European-style Asian options. This relies on the observation that $V(S, C)$ can be evaluated by a simple formula if $C \geq nX$. They can then focus on evaluating $V(N, C)$ for $C < nX$ by using Eq. (3.10) and interpolation. However, no efficient approximation algorithms have convergence guarantees for pricing American-style Asian options.

3.3 An Integral Lattice Reducing the number of prefix sums by restricting the possible stock price of each node in the lattice is first suggested by Dai and Lyuu [8]. In our paper, we will construct a trinomial lattice composed of integral stock prices. Note that if the stock price of each node in the lattice is an integer, all possible prefix sums must be integers since integers are closed under additions. A proof to show that the time complexity of the algorithm is subexponential in n will be given later.

Note that the stock price at the root node S is not required to be an integer. S is a rational number when storing in the computer. Thus S can be represented as $S' + a$ for some integer $S' \geq 0$ and some rational number a where $0 \leq a < 1$. Assume that the maximum prefix sum in the lattice is F. Any possible prefix sum must belong to the set $\{X : X \leq F, X = I + a, I$ is a nonnegative integer.$\}$. The key to show that our algorithm breaks exponential time barrier is to prove that the number of elements in this set is bounded by a subexponential function in n.

4 Lattice Construction

To reduce the number of prefix sums at each node in our lattice, the stock price of each node (except the root node) is restricted to be an integer. To ensure that the stock price process simulated by our lattice converges to the stock price process mentioned in Eq. (2.1), the mean and the variance of the logarithmic stock price process are matched at each node in the lattice. In this section, we will first show how the lattice is constructed step by step in order to meet the above two requirements. Proof will be given in the next section to show that our lattice provides a subexponential-time algorithm for Asian options.

The homogeneous property says [20]:

$$
E[\max(A - X, 0)] = \frac{1}{K} E\left[\max\left(KA - KX, 0\right)\right].
$$

Thus we can multiply the initial stock price (S_0) and the strike price X by a constant K and price this hypothetical option. The desired option value is then obtained by dividing this hypothetical option price by K. To ensure that a proper integral price can be assigned to each node (except the root node), K is defined as follows:

(4.11)
$$K \equiv (0.25 S_0 \sigma)^{-1} \sqrt{n/T} \exp\left[(0.5\sigma^2 - r) T + 2\sigma\sqrt{Tn} \right].$$

Note that $K \in e^{O(\sqrt{n})}$. We will show later that this K works.

Next a trinomial lattice is constructed to price this hypothetical option. Note that the stock price of the root node ($S_{0,0}$) is equal to KS_0.[2] Our goal is to find integral stock prices $S_{i,j}$ for $0 < i \leq n$, $0 \leq j \leq 2i$. Define the V-log-price of stock price V' as $\ln(V'/V)$ and $c_{i,j} \equiv (r - 0.5\sigma^2)\, i\Delta t + 2(i - j)\,\sigma\sqrt{\Delta t}$. $S_{i,j}$ will be some integer whose KS_0-log-price belongs to the following interval centered around $c_{i,j}$: $(c_{i,j} - 0.25\sigma\sqrt{\Delta t}, c_{i,j} + 0.25\sigma\sqrt{\Delta t})$. We call $c_{i,j}$ the log-price center for $S_{i,j}$. Take the 2-time-step trinomial lattice in Fig. 3 as an example. The x-axis marks the time step in the lattice, and the y-axis denotes KS_0-log-prices. Each log-price center is depicted as a hollow circle. Each dotted line segment begins at $c_{i,j}$ and ends at $c_{i+1,j+1}$. The slopes of these dotted lines represent the expected growth rate of the logarithmic stock price, $(r - 0.5\sigma^2)\Delta t$. The integral stock price for each node is depicted as a solid circle. Take $S_{2,0}$ as an example. Because $c_{2,0} = (r - 0.5\sigma^2)\, 2\Delta t + 4\sigma\sqrt{\Delta t}$, the KS_0-log-price of $S_{2,0}$ should fall within the interval $(c_{2,0} - 0.25\sigma\sqrt{\Delta t}, c_{2,0} + 0.25\sigma\sqrt{\Delta t})$. The proof in section 5.1.1 shows that there is always at least one integer whose KS_0-log-price falls within the said interval.

The branching probabilities for each node are computed as follows. Take a node with price $S_{i,j}$. Recall that the probabilities for the stock price moving to $S_{i+1,j}$, $S_{i+1,j+1}$, and $S_{i+1,j+2}$ are P_u, P_m, and P_d, respectively. Define α, β, and γ as follows:

(4.12) $\quad \alpha \equiv \ln(S_{i+1,j}/S_{i,j}) - \mu,$

(4.13) $\quad \beta \equiv \ln(S_{i+1,j+1}/S_{i,j}) - \mu,$

(4.14) $\quad \gamma \equiv \ln(S_{i+1,j+2}/S_{i,j}) - \mu,$

where μ is defined in Eq. (3.5). The branching probabilities satisfy

(4.15) $\quad P_u \alpha + P_m \beta + P_d \gamma = 0,$

(4.16) $\quad P_u \alpha^2 + P_m \beta^2 + P_d \gamma^2 = \mathrm{Var},$

(4.17) $\quad P_u + P_m + P_d = 1,$

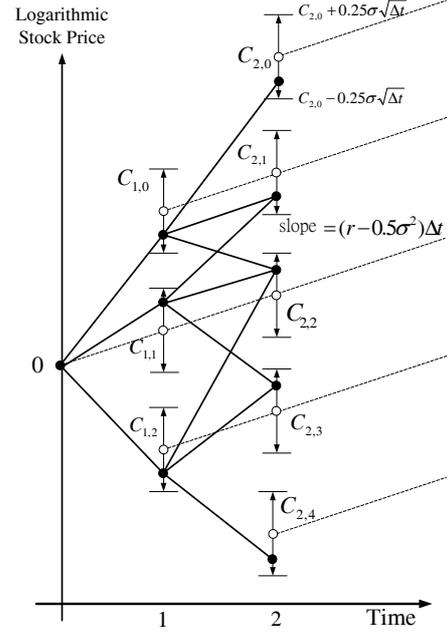[2]Note that KS_0 is not necessary an integer since both K and S_0 are not necessary integers, too.

Figure 3: **A 2-Time-Step Trinomial Lattice over KS_0-log-prices.**

where Var is defined in Eq. (3.6). Eqs. (4.15) and (4.16) match the mean and the variance of the logarithmic stock price, respectively. Hence our lattice converges weakly to the lognormal stock price process. That Eqs. (4.15)–(4.17) give valid branching probabilities will be proved in section 5.1.2.

5 Proof

A proof is given to show that our approach provides a subexponential-time algorithm for pricing Asian options. The proof is broken up into two parts. First, we will show that a valid lattice is constructed. Next we will show that the exact pricing algorithm based on this lattice runs in subexponential time.

5.1 Validity of the Lattice

5.1.1 Existence of Integral Stock Prices When constructing the trinomial lattice, an integral stock price is assigned to each node. More specifically, $S_{i,j}$ should be an integer whose KS_0-log-price falls in $(c_{i,j} - 0.25\sigma\sqrt{\Delta t}, c_{i,j} + 0.25\sigma\sqrt{\Delta t})$. To ensure that such an integer exists, it suffices to show that

$$KS_0 \left[\exp\left(c_{i,j} + 0.25\sigma\sqrt{\Delta t} \right) - \exp\left(c_{i,j} - 0.25\sigma\sqrt{\Delta t} \right) \right] > 1.$$

Our goal now is to show that our choice of K in Eq. (4.11) satisfies the above inequality. Without loss of generality, only the case $S_{n,2n}$ is considered as $\exp(c_{i,j} + 0.25\sigma\sqrt{\Delta t}) - \exp(c_{i,j} - 0.25\sigma\sqrt{\Delta t})$ is minimized when

$i = n$ and $j = 2n$. Thus it suffices to show that our K satisfies

$$K > S_0^{-1} e^{-c_{n,2n}} \left(e^{0.25\sigma\sqrt{\Delta t}} - e^{-0.25\sigma\sqrt{\Delta t}} \right)^{-1}.$$

Indeed,

$$S_0^{-1} e^{-c_{n,2n}} \left(e^{0.25\sigma\sqrt{\Delta t}} - e^{-0.25\sigma\sqrt{\Delta t}} \right)^{-1}$$

$$= S_0^{-1} \exp\left[(0.5\sigma^2 - r)\,T + 2\sigma\sqrt{Tn} \right]$$

$$\times \left(e^{0.25\sigma\sqrt{\Delta t}} - e^{-0.25\sigma\sqrt{\Delta t}} \right)^{-1}$$

$$(5.18) \quad < \quad S_0^{-1} \exp\left[(0.5\sigma^2 - r)\,T + 2\sigma\sqrt{Tn} \right] / (0.25\sigma\sqrt{T/n})$$

$$\leq \quad (0.25 S_0\sigma)^{-1} \sqrt{n/T} \exp\left[(0.5\sigma^2 - r)\,T + 2\sigma\sqrt{Tn} \right]$$

$$= \quad e^{O(\sqrt{n})},$$

where Eq. (5.18) holds because
$$e^{0.25\sigma\sqrt{\Delta t}} - e^{-0.25\sigma\sqrt{\Delta t}} > \left(1 + 0.25\sigma\sqrt{\Delta t}\right) - 1 = 0.25\sigma\sqrt{\Delta t}.$$

5.1.2 Validity of Branching Probabilities

In this section, we will prove that Eqs. (4.15)–(4.17) result in valid branching probabilities for the stock price $S_{i,j}$. To be more specific, the branching probabilities P_u, P_m, and P_d, are proved to be strictly larger than 0 to meet the no-arbitrage requirement (note that this suffices to ensure that they are all less than 1). First we will derive constraints on α, β, γ defined in Eqs. (4.12)–(4.14). Then we will show that valid branching probabilities are obtained by solving Eqs. (4.15)–(4.17) given the constraints on α, β, and γ.

We use the plot in Fig. 4 to aid the proof. Note that a KS_0-log-price of x becomes a $S_{i,j}$-log-price of $x + \frac{\ln KS_0}{\ln S_{i,j}}$. The following computations are in $S_{i,j}$-log-prices unless sated otherwise. The log-price center of $S_{i,j}$, c', equals $c_{i,j} + \frac{\ln KS_0}{\ln S_{i,j}}$, whereas c, the log-price center of $S_{i+1,j+1}$, equals $c' + (r - 0.5\sigma^2)\Delta t$. The log-price centers of $S_{i+1,j}$ and $S_{i+1,j+2}$ are $c + 2\sigma\sqrt{\Delta t}$ and $c - 2\sigma\sqrt{\Delta t}$, respectively. The $S_{i,j}$-log-price of $S_{i,j}$ is 0. The conditional mean of the stock price one time step after it reaches $S_{i,j}$ is $\mu = (r - 0.5\sigma^2)\Delta t$. By construction, the distance between the $S_{i,j}$-log-price of $S_{i,j}$, which equals 0, and its log-price center c' is smaller than $0.25\sigma\sqrt{\Delta t}$. This implies that $|c'| < 0.25\sigma\sqrt{\Delta t}$. Hence,

$$|\mu - c| = \left| (r - 0.5\sigma^2)\Delta t - \left[c' + (r - 0.5\sigma^2)\Delta t \right] \right|$$

$$= |c'| < 0.25\sigma\sqrt{\Delta t}.$$

Thus μ falls within interval $(c - 0.25\sigma\sqrt{\Delta t}, c + 0.25\sigma\sqrt{\Delta t})$. Now $\beta = \ln(S_{i+1,j+1}/S_{i,j}) - \mu$ falls within interval $(-0.5\sigma\sqrt{\Delta t}, 0.5\sigma\sqrt{\Delta t})$ as the $S_{i,j}$-log-price of $S_{i+1,j+1}$, $\ln(S_{i+1,j+1}/S_{i,j})$, falls within interval $(c - 0.25\sigma\sqrt{\Delta t}, c + 0.25\sigma\sqrt{\Delta t})$. Figure 4 illustrates a case where $\beta < 0$.

We next represent α and γ in terms of β. Define d_1 as the difference between $S_{i+1,j}$'s and $S_{i+1,j+1}$'s $S_{i,j}$-log-prices and d_2 as the difference between $S_{i+1,j+1}$'s and $S_{i+1,j+2}$'s $S_{i,j}$-log-prices. Thus α and γ can be represented as

$$\alpha = \ln(S_{i+1,j}/S_{i,j}) - \ln(S_{i+1,j+1}/S_{i,j})$$
$$+ \ln(S_{i+1,j+1}/S_{i,j}) - \mu$$
$$= d_1 + \beta,$$
$$\gamma = \ln(S_{i+1,j+2}/S_{i,j}) - \ln(S_{i+1,j+1}/S_{i,j})$$
$$+ \ln(S_{i+1,j+1}/S_{i,j}) - \mu$$
$$= -d_2 + \beta.$$

Note that $1.5\sigma\sqrt{\Delta t} < d_1 < 2.5\sigma\sqrt{\Delta t}$ because

$$c + 1.75\sigma\sqrt{\Delta t} < \ln(S_{i+1,j}/S_{i,j}) < c + 2.25\sigma\sqrt{\Delta t},$$
$$c - 0.25\sigma\sqrt{\Delta t} < \ln(S_{i+1,j+1}/S_{i,j}) < c + 0.25\sigma\sqrt{\Delta t},$$
$$d_1 = \ln(S_{i+1,j}/S_{i,j}) - \ln(S_{i+1,j+1}/S_{i,j}).$$

Similarly, $1.5\sigma\sqrt{\Delta t} < d_2 < 2.5\sigma\sqrt{\Delta t}$. Note also that $\alpha = d_1 + \beta > \sigma\sqrt{\Delta t} > 0$ as $\beta \in (-0.5\sigma\sqrt{\Delta t}, 0.5\sigma\sqrt{\Delta t})$ and $d_1 \in (1.5\sigma\sqrt{\Delta t}, 2.5\sigma\sqrt{\Delta t})$. Similarly, $\gamma = -d_2 + \beta < -\sigma\sqrt{\Delta t} < 0$ as $d_2 \in (1.5\sigma\sqrt{\Delta t}, 2.5\sigma\sqrt{\Delta t})$. It is also obvious that $\alpha > \beta > \gamma$ as $d_1, d_2 > 0$.

We now show that positive branching probabilities are obtained given the constraints on α, β, and γ derived above and summarized below:

$$\beta \in (-0.5\sigma\sqrt{\Delta t}, 0.5\sigma\sqrt{\Delta t}).$$
$$\alpha = d_1 + \beta, \text{ where } d_1 \in (1.5\sigma\sqrt{\Delta t}, 2.5\sigma\sqrt{\Delta t}).$$
$$\gamma = -d_2 + \beta, \text{ where } d_2 \in (1.5\sigma\sqrt{\Delta t}, 2.5\sigma\sqrt{\Delta t}).$$

The branching probabilities are solved by applying Cramer's rule to Eqs. (4.15)– (4.17). Define

$$\det = (\beta - \alpha)(\gamma - \alpha)(\gamma - \beta),$$
$$\det_u = (\beta\gamma + \mathrm{Var})(\gamma - \beta),$$
$$\det_m = (\alpha\gamma + \mathrm{Var})(\alpha - \gamma),$$
$$\det_d = (\alpha\beta + \mathrm{Var})(\beta - \alpha).$$

Then $P_u = \det_u/\det$, $P_m = \det_m/\det$, and $P_d = \det_d/\det$. Note that $\det < 0$ as $\alpha > \beta > \gamma$. To show that the branching probabilities are valid, we have to show that $P_u, P_m, P_d > 0$. As $\det < 0$, it is sufficient to show $\det_u, \det_m, \det_d < 0$. Moreover, as $\gamma - \beta < 0$, $\alpha - \gamma > 0$, and $\beta - \alpha < 0$, we only need to show that $\beta\gamma + \mathrm{Var} > 0$, $\alpha\gamma + \mathrm{Var} < 0$, and $\alpha\beta + \mathrm{Var} > 0$ instead.

1. $\beta\gamma + \mathrm{Var} > 0$: Note that

$$\beta\gamma + \mathrm{Var} = \beta(\beta - d_2) + \sigma^2\Delta t$$
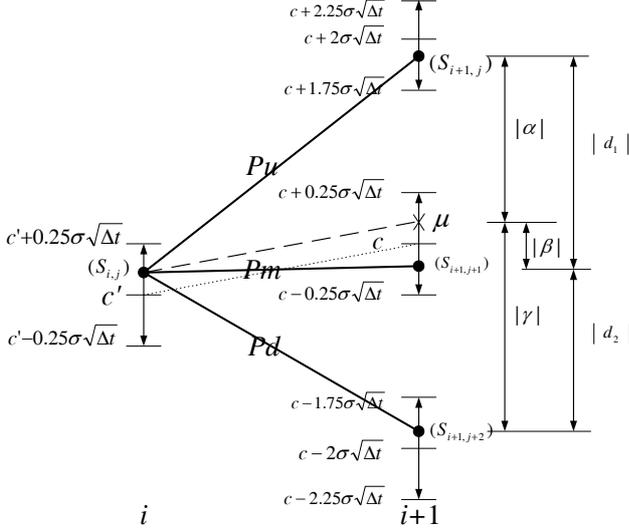$$= (\beta - 0.5 d_2)^2 - 0.25 d_2^2 + \sigma^2\Delta t.$$

Figure 4: **Branching Probabilities for the Node with Price $S_{i,j}$.**
All the values in this figure are $S_{i,j}$-log-prices except the ones that are parenthesized. The nodes with stock prices $S_{i,j}$, $S_{i+1,j}$, $S_{i+1,j+1}$, and $S_{i+1,j+2}$ are represented by solid circles. The branches that connect these nodes are represented by thick lines. The log-price centers of $S_{i,j}$ and $S_{i+1,j+1}$ are c' and c, respectively. P_u, P_m, and P_d denote the branching probabilities for the upper, middle, and lower branches from the node with price $S_{i,j}$. Values α, β, and γ are defined in Eqs. (4.12)–(4.14). Finally, $|d_1|$ denotes the distance between $S_{i+1,j}$'s and $S_{i+1,j+1}$'s $S_{i,j}$-log-prices, and $|d_2|$ denotes the distance between $S_{i+1,j+1}$'s and $S_{i+1,j+2}$'s $S_{i,j}$-log-prices.

For a given d_2, $\beta\gamma + \text{Var}$ reaches its minimum when $\beta = 0.5d_2$. Recall the constraints that $0.5d_2 \in (0.75\sigma\sqrt{\Delta t}, 1.25\sigma\sqrt{\Delta t})$ and $\beta < 0.5\sigma\sqrt{\Delta t}$. Hence $\beta\gamma + \text{Var}$ reaches its minimum for a given $d_2 \in (1.5\sigma\sqrt{\Delta t}, 2.5\sigma\sqrt{\Delta t})$ when $\beta \to 0.5\sigma\sqrt{\Delta t}$. Thus, we have

$$
\begin{aligned}
\beta\gamma + \text{Var} &= \beta(\beta - d_2) + \sigma^2\Delta t \\
&> 0.5\sigma\sqrt{\Delta t}\,(0.5\sigma\sqrt{\Delta t} - d_2) + \sigma^2\Delta t \\
&= -0.5\sigma\sqrt{\Delta t}\,d_2 + 1.25\sigma^2\Delta t \\
&> -1.25\sigma^2\Delta t + 1.25\sigma^2\Delta t = 0.
\end{aligned}
$$

2. $\alpha\gamma + \text{Var} < 0$: Note that

$$
\alpha\gamma + \text{Var} = (\beta + d_1)(\beta - d_2) + \sigma^2\Delta t.
$$

As $\beta + d_1 > \sigma\sqrt{\Delta t}$ and $\beta - d_2 < -\sigma\sqrt{\Delta t}$, we have

$$(\beta+d_1)(\beta-d_2)+\sigma^2\Delta t < (\sigma\sqrt{\Delta t})(-\sigma\sqrt{\Delta t})+\sigma^2\Delta t = 0.$$

3. $\alpha\beta + \text{Var} > 0$: This is proved similarly as we prove $\beta\gamma + \text{Var} > 0$. Note that

$$
\begin{aligned}
\alpha\beta + \text{Var} &= (\beta + d_1)\beta + \sigma^2\Delta t \\
&= (\beta + 0.5d_1)^2 - 0.25d_1^2 + \sigma^2\Delta t.
\end{aligned}
$$

For a given d_1, $\alpha\beta + \text{Var}$ reaches its minimum when $\beta = -0.5d_1$. Recall the constraints that $-0.5d_1 \in (-1.25\sigma\sqrt{\Delta t}, -0.75\sigma\sqrt{\Delta t})$ and $\beta > -0.5\sigma\sqrt{\Delta t}$. Hence $\alpha\beta + \text{Var}$ reaches its minimum for a given $d_1 \in (1.5\sigma\sqrt{\Delta t}, 2.5\sigma\sqrt{\Delta t})$ when $\beta \to -0.5\sigma\sqrt{\Delta t}$. Thus we have

$$
\begin{aligned}
\alpha\beta + \text{Var} &= (\beta + d_1)\beta + \sigma^2\Delta t \\
&> (-0.5\sigma\sqrt{\Delta t} + d_1)(-0.5\sigma\sqrt{\Delta t}) + \sigma^2\Delta t \\
&= -0.5\sigma\sqrt{\Delta t}\,d_1 + 1.25\sigma^2\Delta t \\
&> -1.25\sigma^2\Delta t + 1.25\sigma^2\Delta t = 0.
\end{aligned}
$$

5.2 Running-Time Analysis Note that the time complexity of an exact pricing algorithm is proportional to the total number of prefix sums on the lattice. Thus the pricing algorithm is subexponential in n if the total number of prefix sums is bounded by a subexponential function. We will first show that the maximum prefix sum in our lattice is bounded by a subexponential function. Then we will show that the total number of prefix sums is, too.

The maximum stock price $S_{n,0}$ is bounded by $KS_0 e^{c_{n,0}+0.25\sigma\sqrt{\Delta t}}$, where

$$
\begin{aligned}
c_{n,0} &= (r - 0.5\sigma^2)n\Delta t + 2(n-0)\sigma\sqrt{\Delta t} \\
&= (r - 0.5\sigma^2)\,T + 2\sigma\sqrt{Tn}.
\end{aligned}
$$

Thus $S_{n,0}$ is bounded by a subexponential function as both K and $e^{c_{n,0}+0.25\sigma\sqrt{\Delta t}}$ are subexponential in n. The maximum prefix sum in the lattice is equal to $\sum_{i=0}^{n} S_{i,0} \le (n+1)S_{n,0}$. Note that $(n+1)S_{n,0}$ is also a subexponential function. We define $F \equiv (n+1)S_{n,0}$ for simplicity.

The next goal is to show that the total number of prefix sums is bounded by a subexponential function. Recall that the stock price for each node except the root node in our lattice must be an integer. The stock price at the root node (KS_0) can be represented as $S' + a$ for some integer $S' \ge 0$ and some rational number a where $0 \le a < 1$. Thus all the possible prefix sums must belong to the set $\{X : X \le F, X = I + a, I \text{ is a nonnegative integer.}\}$. The number of elements in this set is at most $\lceil F \rceil$. Thus the maximum number of prefix sums for each node is bounded by $\lceil F \rceil$. Since there are $(n+1)^2$ nodes in an n-time-step trinomial lattice, the total number of prefix sums is bounded above by the subexponential function $(n+1)^2 \lceil F \rceil$. The

time complexity of our algorithm is thus subexponential in n.

6 Conclusions

This paper develops a new trinomial lattice particularly with the Asian option in mind. The lattice uses the notion of integrality of stock prices to reduce the time complexity of an exact pricing algorithm dramatically. Rigorous proof is given to show that the time complexity is subexponential in n. Our lattice is guaranteed to converge to the continuous-time stock price process. The proposed pricing algorithm is guaranteed to converge to the desired option value as it is exact. This is the first exact lattice algorithm to break the exponential-time barrier.

References

[1] ABATE, J., AND W. WHITT. (1995). "Numerical Inversion of Laplace Transforms of Probability Distributions." *ORSA Journal of Computing* 7, 36–43.

[2] AINGWORTH, D., R. MOTWANI, AND J.D. OLDHAM. (2000). "Accurate Approximations for Asian Options." In *Proc. 11th Annual ACM-SIAM Symposium on Discrete Algorithms.*

[3] BOYLE, P.P., M. BROADIE, AND P. GLASSERMAN. (1997). "Monte Carlo Methods for Security Pricing." *Journal of Economic Dynamics & Control* 21, 1267–1321.

[4] BROADIE, M., AND P. GLASSERMAN. (1996). "Estimating Security Price Derivatives Using Simulation." *Management Science* 42, 269–285.

[5] BROADIE, M., P. GLASSERMAN, AND S. KOU. (1999). "Connecting Discrete and Continuous Path-Dependent Options." *Finance and Stochastics* 3, 55–82.

[6] COX, J., S. ROSS, AND M. RUBINSTEIN. (1979). "Option Pricing: A Simplified Approach." *Journal of Financial Economics* 7, 229–264.

[7] DAI, T.-S., G. -S. HUANG, AND Y.-D. LYUU. (2002). "Extremely Accurate and Efficient Algorithms for Asian Options with Error Bounds." Quantitative Methods in Finance 2002 Conference, Cairns, Australia.

[8] DAI, T.-S., AND Y.-D. LYUU. (2002). "Efficient, Exact Algorithms for Asian Options with Multiresolution Lattices." *Review of Derivatives Researches* 5, 181–203.

[9] DUFFIE, D. (1996). *Dynamic Asset Pricing Theory.* 2nd ed. Princeton NJ: Princeton University Press.

[10] FORSYTH, P. A., K. R. VETZAL, AND R. ZVAN. (2002). "Convergence of Numerical Methods for Valuing Path-Dependent options Using Interpolation." *Journal of Computational Finance* 5, 273–314.

[11] FU, M.C., D.B. DILIP, AND T. WANG. (1998/9). "Pricing Continuous Asian Options: A Comparison of Monte Carlo and Laplace Transform Inversion Methods." *Journal of Computational Finance* 2, 49–74.

[12] GEMAN, H., AND A. EYDELAND. (1995). "Domino Effect." *Risk* 8, 65–67.

[13] GEMAN, H., AND M. YOR. (1993). "Bessel Processes, Asian Options, and Perpetuities." *Mathematical Finance* 3, 349–375.

[14] HULL, J., AND A. WHITE. (1993). "Efficient Procedures for Valuing European and American Path-Dependent Options." *Journal of Derivatives* 1, 21–31.

[15] KLASSEN, T.R. (2001). "Simple, Fast and Flexible Pricing of Asian Options." *Journal of Computational Finance* 4, 89–124.

[16] KEMNA, A.G.Z., AND A.C.F. VORST. (1990). "A Pricing Method Based on Average Asset Values." *Journal of Banking and Finance* 14, 113–129.

[17] LEVY, E. (1992). "Pricing European Average Rate Currency Options." *Journal of International Money and Finance* 11, 474–491.

[18] LONGSTAFF, F.A., AND E.S. SCHWARTZ. (2001). "Valuing American Options by Simulation: a Simple Least-Squares Approach." *Review of Financial Studies* 14, 113–147.

[19] LYUU, Y.-D. (2002). *Financial Engineering and Computation: Principles, Mathematics, and Algorithms.* Cambridge, U.K.: Cambridge University Press.

[20] MERTON, R.C. (1994). *Continuous-Time Finance.* Revised ed. Cambridge, MA: Blackwell.

[21] MILEVSKY, M.A., AND S.E. POSNER. (1998). "Asian Options, the Sum of Lognormals, and the Reciprocal Gamma Distribution." *Journal of Financial and Quantitative Analysis* 33, 409–422.

[22] RITCHKEN, P., L. SANKARASUBRAMANIAN, AND A.M. VIJH. (1993). "The Valuation of Path Dependent Contracts on the Average." *Management Science* 39, 1202–1213.

[23] SHAW, W.T. (1998). *Modeling Financial Derivatives with Mathematica.* Cambridge, U.K.: Cambridge University Press.

[24] TURNBULL, S.M., AND L.M. WAKEMAN. (1991). "A Quick Algorithm for Pricing European Average Options." *Financial and Quantitative Analysis* 26, 377–389.

[25] ZVAN, R., AND K. VETZAL. (1999). "Discrete Asian Barrier Options." *Journal of Computational Finance* 3, 41–68.