Submission to Algorithmica

A Fast, Accurate and Simple Method for Pricing European-Asian and Saving-Asian Options

Ken'ichiro Ohta, [1] Kunihiko Sadakane, [2] Akiyoshi Shioura, [3 4] Takeshi Tokuyama[3]

March, 2003; revised April, 2004

Abstract

We propose an efficient and accurate randomized approximation algorithm for computing the price of European-Asian options. Our algorithm can be seen as a modification of the approximation algorithm developed by Aingworth et al. (2000) into a randomized algorithm, which improves the accuracy theoretically as well as practically. We also propose a new option named Saving-Asian option which enjoys advantage of both of European-Asian and American-Asian options. It is shown that our approximation algorithm also works for pricing Saving-Asian options.

[1]DC Card Co., Ltd., Tokyo, Japan
[2]Graduate School of Information Science and Electrical Engineering, Kyushu University, Fukuoka 812-8581, Japan
[3]Graduate School of Information Sciences, Tohoku University, Sendai 980-8579, Japan
[4]Corresponding Author, Phone/Fax: +81-22-217-4753, e-mail: shioura@dais.is.tohoku.ac.jp

1 Introduction

1.1 Background

Options are popular and important financial instruments in world financial markets. A (call) option gives the right, but not the obligation, to buy something (usually a stock) at some point in the future for a specified price called the *strike price*. If we have an option on a stock and the stock is worth more than the strike price, then we can *exercise* the option to buy the stock for less than you otherwise could. The price of the option, often called the premium, is usually much less than the actual price of the underlying stock. Options hedge risk more cheaply than stocks only, and provide a chance to get large profit with a small amount of money if one's speculation is good.

For example, suppose that you buy $1,000$ units of a stock with the current price \$200, and forecast that the stock price will possibly go up to \$300 at the end of the year. If your forecast comes true, then you will gain \$100,000; however, if the stock price goes down to \$100, you will unfortunately lose \$100,000 which you cannot afford. Instead, suppose that you can buy at the premium \$8 an option which gives you the right to buy the stock at the strike price \$220. If the stock price goes up to \$300, you will obtain \$80 extra, called the *payoff*, for each unit of the option by exercising it and selling the stock at the market price. Thus, if you buy 1,250 units of this option, you have a chance to gain the total payoff of \$100,000, reducing the maximum loss to be \$10,000 which is just the total premium. You may buy 2000 units of another option with the strike price \$250 and the premium \$2, and dream to gain \$100,000 with the maximum loss \$4,000.

Here, it is natural to ask whether the option premiums \$8 and \$2 are fair or not. As shown in this example, pricing of options is a central topic in financial engineering.

To compute the price of an option, it is needed to model the price movement of each individual stock. Typically, it is modeled as geometric Brownian motion with drift. This yields a stochastic differential equation, and its solution gives the option price. However, it is often difficult to solve this differential equation, and indeed no simple closed-form solution is known for Asian options discussed in this paper. Therefore, it is widely practiced to simulate Brownian motion by using a combinatorial model, and obtain a solution on the model, which we call the *combinatorial exact price* or the *exact price*, as an approximation of the solution obtained from the differential equation. One such combinatorial model is the *binomial tree model* [8], where the time period is decomposed into n time steps, and Brownian motion is modeled by using a biased random walk on a directed acyclic graph called a *binomial tree* of depth n. The combinatorial exact price obtained from the binomial tree model converges to the option price given by the differential equation if n goes to infinity.

In the binomial tree model, the process of the movement of a stock price is represented by a path in a binomial tree. An option is said to be *path-dependent* if the option's payoff depends on the path representing the process as well as the current stock price. Although path-dependency is often useful in designing a secure option against risk caused by sudden change of the market, it makes the analysis of the price of options quite difficult.

1.2 Our Problems and Results

In this paper, we consider the pricing of *Asian* option, which is a kind of path-dependent options. It is known to be #P-hard in general to compute the exact price of path-dependent options on the binomial tree model [6]. Therefore, it is desired to design an efficient approximation algorithm with provable high accuracy, and various pricing techniques have been developed so far (see, e.g, [1, 2, 6, 7, 9]).

European-Asian option is the simplest Asian option which allows exercise only on a expiration date. A most naive method for computing the exact price of European-Asian options, called the *full-path method*, is to enumerate all paths in the binomial tree model. Unfortunately, there are exponential number of paths in the binomial tree, and therefore the full-path method requires exponential time. Hence, the Monte Carlo method that samples paths in the binomial tree is popularly used to compute an approximate value of the exact price. When a polynomial number of samples are taken by naive sampling, however, the error bound depends on the volatility of the stock price, and the bound only holds with high probability. See [10] for more accounts on the Monte Carlo method for option pricing.

Aingworth–Motwani–Oldham [1] proposed the first polynomial-time algorithm with guaranteed worst-case error bound, which enables us to avoid the influence of volatility to the theoretical error bound. The idea is to aggregate exponential number of high-payoff paths by using mathematical formulae during the run of an approximate aggregation algorithm based on dynamic programming and bucketing. They proposed an $O(n^2k)$-time algorithm (referred to the AMO algorithm), and proved that its error is bounded by nX/k, where X is the strike price of the option and k is a parameter giving the time-accuracy tradeoff.

Later, Akcoglu–Kao–Raghavan [2] presented various pricing techniques applicable to approximation algorithms such as the Monte Carlo method and the AMO algorithm. In particular, they use a recursive version of the AMO algorithm and reduce the error bound to $O(n^{\frac{1+\varepsilon}{2}}X/k)$ by spending almost the same time complexity under the condition that the volatility of the stock is small. Quite recently, Dai–Huang–Lyuu [9] developed an improved version of the AMO algorithm. Their algorithm runs in the same time complexity as the AMO algorithm and has the error bound $O(\sqrt{n}X/k)$, which is the best error bound so far.

In this paper, we propose a randomized algorithm with an $O(n^2k)$ time complexity and an $O(\sqrt{n}X/k)$ error bound which does not require a volatility condition. Our algorithm can be regarded as a slight modification of the AMO algorithm as well as a variant of the Monte Carlo method. Thus, our algorithm enjoys advantages of both methods simultaneously. While algorithms on the uniform model has been mainly considered in the literature [1, 2, 6, 7, 9], our algorithm and analysis also work on the non-uniform model where the transition probabilities of the stock price may differ at each node. We note that the analysis of the algorithm by Dai et al. [9], which has the same error bound as ours, works only on the uniform model. Moreover, the error bound of our algorithm can be improved to $O(n^{1/4}X/k)$ for the uniform model. Although we only consider the pricing of call options on the binomial tree model, our algorithm can be easily modified to put options and to the trinomial tree model as in [1, 2, 9].

The idea of our algorithm is as follows. By using novel random variables, we regard the aggregation

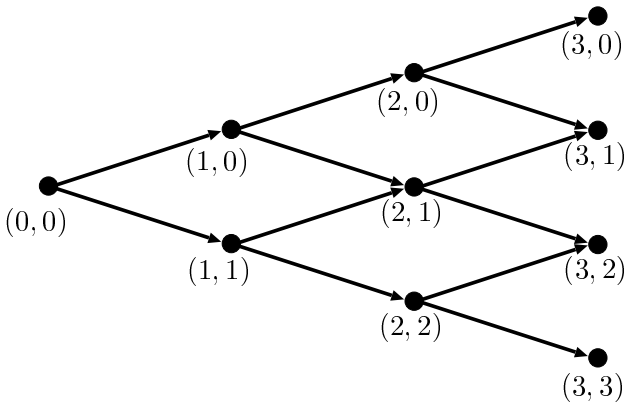

Figure 1: A binomial tree of depth 3

process of the algorithm as a Martingale process with $n+1$ random steps. It can be shown that the expected value of the output by our algorithm equals the exact price, and that the error of a single step is bounded by X/k. Thus, we can apply Azuma's inequality [3] to the Martingale process to obtain the error bound $\mathrm{O}(\sqrt{n}X/k)$. We also show the practical quality of the approximate value computed by our algorithm by some numerical experiments; indeed, its accuracy is better by a factor of nearly 100 than that of the AMO algorithm when $n = 35$.

Inspired by the analysis, we then propose an option which is a composition of European-Asian and American-Asian options, and show that our algorithm can be generalized to its pricing. *American-Asian options* can be exercised at any date before the expiration date, and therefore is difficult to anticipate the action of the option holder. This makes accurate pricing of American-Asian options much harder than that of European-Asian options (see, e.g., [4, 5, 12]). Instead of American-Asian option, we propose *Saving-Asian option* which permits early exercise but has a different payoff system so that the action of the option holder can be anticipated more easily and the expected payoff can be computed accurately. The payoff of Saving-Asian options depends on the average stock price, and hence secures against the sudden change of the market. Furthermore, the option reduces risk for the seller of the option when compared with American-Asian option, and therefore the price should be cheaper. Hence, we believe that our new option and its analysis technique will be useful in theory as well as in practice.

2 Preliminaries

2.1 The Binomial Tree Model

A *binomial tree*, also called a *recombinant binary tree*, of depth n is a leveled directed acyclic graph defined as follows (see Figure 1). Each node is labeled as (i, j), where i $(0 \leq i \leq n)$ denotes the level and j $(0 \leq j \leq i)$ denotes the numbering of the nodes in the i-th level. The node $(0, 0)$ in the 0-th level is called the *root*, and each node (n, j) in the n-th level is called a *leaf*. Each non-leaf node (i, j) has two children $(i+1, j)$ and $(i+1, j+1)$. Therefore, each non-root node (i, j) has two parents

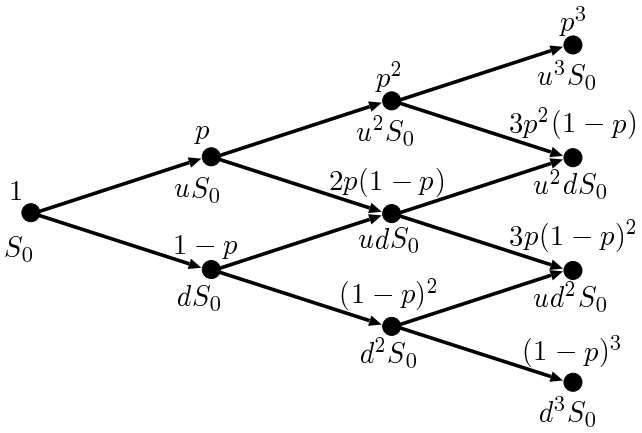

Figure 2: The uniform binomial tree model. The probability of reaching each node (resp., the stock price at each node) is shown above the node (resp., below the node).

$(i-1, j-1)$ and $(i-1, j)$ if $1 \leq j \leq i-1$, and each of $(i, 0)$ and (i, i) has only one parent.

Let us consider a discrete random process simulating the movement of a stock price. We divide the time from the purchase date to the expiration date of an option into n time periods, and the i-th time step means the end of the i-th time period. In particular, 0-th (resp., n-th) time step is the purchase (resp., expiration) date of the option. For $i = 0, 1, \ldots, n$, let S_i be a random variable representing the stock price at the i-th time step, where S_0 is the initial stock price known in advance. The fundamental assumption in the binomial tree model is that in each time step the stock price S either rises to uS or falls to dS, where u and d are predetermined constants with $u > d$. Thus, we can model the stock price movement by using a binomial tree (see Figure 2).

Suppose that we are at a non-leaf node (i, j) in the binomial tree model and the current stock price is S. With probability p_{ij}, we move to the node $(i+1, j)$ and the stock price rises to uS; with probability $1 - p_{ij}$, we move to the node $(i+1, j+1)$ and the stock price falls to dS. Thus, the stock price at the node (i, j) is $S_i(j) = u^{i-j}d^j S_0$.

The binomial tree model is said to be *uniform* if $p_{ij} = p$ for each node (i, j); otherwise it is *non-uniform*. The uniform model has been widely considered [1, 2, 6, 7, 9] since p is uniquely determined under the non-arbitrage condition of the underlying stock. The non-uniform model, however, is often useful to deal with various stochastic models. In the uniform model, the probability that the random walk reaches to (i, j) is $\binom{i}{j} p^{i-j}(1-p)^j$, where $\binom{i}{j} = i!/(i-j)!j!$. We define $r = up + d(1-p) - 1$, which corresponds to the risk-free interest rate for one time period in the risk-neutral probability model.

2.2 Options

Let X be the strike price of an option. The value of the option, which is a random variable, is called *payoff*. In Black–Scholes' theory, the price of an option is given by the discounted expected value of the payoff over the n time periods. Hence, it suffices for the option pricing to compute (or approximate) the expected value of the payoff.

We adopt a convention to write F^+ for $\max\{F, 0\}$. *European option* is one of basic options, and its payoff is given by $(S_n - X)^+$ which is determined by the stock price S_n at the expiration date. It is quite easy to compute the expected value of the payoff of European options under the binomial tree model. A drawback of European options is that the payoff may be affected drastically by the movement of the stock price just before the expiration date; even if the stock price goes very high during most of time periods, it may happen that the option does not make money at the end.

European-Asian option is more reliable for the holder than European option, and its payoff is given by $(A_n - X)^+$, where $A_n = (\sum_{i=0}^{n} S_i)/(n+1)$ is the average of the stock prices during n time periods. Let $T_j = \sum_{i=0}^{j} S_i$ be the *running total* of the stock price up to the j-th time step. If T_j is above the threshold $(n+1)X$, then the holder of the option will surely exercise it at the expiration date and obtain the payoff of at least $T_j/(n+1) - X$.

Our aim is to compute the expected value of the payoff of European-Asian options. A simple method is to compute the running total $T_n(\mathcal{P})$ of the stock price for each path $\mathcal{P}$ in the binomial tree together with the probability $\Pr(\mathcal{P})$ that the path occurs, and exactly compute

$$E((A_n - X)^+) = \sum \left\{ \Pr(\mathcal{P}) \cdot \left(\frac{T_n(\mathcal{P})}{n+1} - X \right)^+ \;\middle|\; \mathcal{P} : \text{ a path from the root to a leaf} \right\}.$$

We call the expected value of the payoff computed as above the *exact value* of the expected payoff, and denote $U = E((A_n - X)^+)$. This simple method, however, needs exponential time since there are 2^n paths in a binomial tree. The Monte Carlo method is a popular method to reduce computation time, although we need a huge number of paths to assure a small provable error bound if we use naive random sampling of paths.

2.3 Saving-Asian Options

American-Asian option allows the holder to exercise it at any time step and to receive $A_i - X$ if the option is exercised at the i-th time step, where $A_i = T_i/(i+1)$. Apparently, American-Asian options are much more advantageous to the holder than European-Asian options, and hence its price should be more expensive. The difficulty of American-Asian options is that the holder's action is highly path-dependent. Even after the current running total exceeds $(n+1)X$, the holder must decide whether to exercise the option immediately. The holder's decision depends both on the running total T_i and on the stock price S_i. Thus, its pricing with provable accuracy seems quite difficult.

To overcome this disadvantage of American-Asian options, we propose a new option named *Saving-Asian option*. The holder of a Saving-Asian option can exercise it at any time step, and receive $e^{-(n-i)r_0/n}\{T_i - (i+1)X\}/(n+1)$ if the option is exercised in the i-th time step, where e^{r_0} is the risk-free interest rate for the whole period. Thus, it is an American-type option, but different from the standard American-Asian option since it restricts the payoff for early exercise.

Saving-Asian options are clearly more advantageous for the holder than European-Asian options; the holder has a choice to keep the option until the expiration date and obtain the payoff $(A_n - X)^+$ which is exactly the same as that of European-Asian options. On the other hand, if the holder exercises the option in the i-th time step and re-invest the money, the holder will have $\{T_i - (i+1)X\}/(n+1)$

in the n-th time step, which might be larger than $A_n - X = \{T_n - (n+1)X\}/(n+1)$. Therefore, even if the stock price will drastically go down after high-price periods, the holder can exercise early to avoid the reduction of the payoff. Moreover, early exercise has advantage that the holder can get money for urgent need.

Intuitively, Saving-Asian options simulate accumulative investment permitting discontinuation, where we can buy $1/(n+1)$ unit of the stock by $X/(n+1)$ dollars for selling it by the market price every time step, and stop at the i-th time step after investing $(i+1)X/(n+1)$ dollars to receive the profit obtained so far. Apparently, the payoff is path-dependent, and thus the option is not in the category of Markovian American options (see [6] for the definition of Markovian options).

Similarly to American-Asian options, the the holder's action seems to be path-dependent. However, it is easier to analyze the best action assuming that the holder has the same model of the stock price movement as the seller. In particular, once the running total of the option exceeds the threshold $(n+1)X$, the conditional expectation of the payoff can be computed analytically as in the case of European-Asian options. Thus, the holder's decision is dependent on the current stock price, but independent of the history of the movement of the stock. We note that the holder may exercise before the running total is above $(n+1)X$, and in such a case the decision depends also on the current running total; however, this kind of path-dependency can be treated efficiently.

3 A New Algorithm for Pricing European-Asian Options

3.1 The AMO Algorithm

We give a brief overview of the AMO algorithm [1] (see Figure 3). The AMO algorithm is based on dynamic programming and computes an approximate value of the option's expected payoff in the range $[U - nX/k, U]$ in $\mathrm{O}(n^2k)$ time, where U is the exact value of the expected payoff and k is any positive integer representing the time-error tradeoff.

For a path $\mathcal{P}$ from the root to a node (i,j) in the i-th level, the *state* of $\mathcal{P}$ is defined as a pair $(S_i(j), T_i)$ of the stock price $S_i(j) = u^{i-j}d^jS_0$ and the running total T_i. We define the *weight* of the state $(S_i(j), T_i)$ as the probability the path $\mathcal{P}$ occurs. The AMO algorithm is based on a simple observation that if the running total of a current state is above the threshold $(n+1)X$, then the conditional expectation of the payoff at this state can be analytically computed (see Section 3.2), and such a state can be pruned away.

Hence, we need to consider only the states with running total less than $(n+1)X$, which may be exponential many. Rather than dealing with each unpruned state individually, we instead aggregate the states by using buckets that divide the interval $[0, (n+1)X)$. The algorithm creates k buckets $B_i(j,h)$ $(h = 0, 1, \ldots, k-1)$, each of which corresponds to the interval $[b_h, b_{h+1}) = [h(n+1)X/k, (h+1)(n+1)X/k)$. Each unpruned state of a path terminating at the node (i,j) is stored in one of k buckets according to its running total. Then, the algorithm approximates all states in the bucket $B_i(j,h)$ by a single state $(S_i(j), b_h)$, where its weight $w_i(j,h)$ is given by the sum of the weights of all states in $B_i(j,h)$.

l.1: Let $\Phi := 0$. Initialize all buckets $B_i(j,k)$.
l.2: Insert the pair of the initial state (S_0, S_0) and its weight 1 in the bucket $B_0(0, \left\lfloor \frac{S_0 k}{(n+1)X} \right\rfloor)$.
l.3: **for all** levels $i = 0, 1, \ldots, n-1$ **do**
l.4: **for all** nodes (i,j) $(j = 0, 1, \ldots, i)$ **do**
l.5: **for all** buckets $B_i(j,h)$ $(h = 0, 1, \ldots, k-1)$ at the node (i,j) **do**
l.6: Compute the sum $w_i(j,h)$ of the weights of all states in $B_i(j,h)$.
l.7: Let $T_i(j,h) = b_h$.
l.8: **if** $T_i(j,h) + S_{i+1}(j) < (n+1)X$ **then**
l.9: Insert the pair of the state $(S_{i+1}(j), T_i(j,h) + S_{i+1}(j))$ and its weight $p_{ij}\, w_i(j,h)$ in the bucket $B_{i+1}(j, \left\lfloor \frac{(T_i(j,h)+S_{i+1}(j))k}{(n+1)X} \right\rfloor)$.
l.10: **else**
l.11: Compute the payoff's conditional expectation Φ' at the state $(S_{i+1}(j), T_i(j,h) + S_{i+1}(j))$, and let $\Phi := \Phi + p_{ij} w_i(j,h)\, \Phi'$.
l.12: **if** $T_i(j,h) + S_{i+1}(j+1) < (n+1)X$ **then**
l.13: Insert the pair of the state $(S_{i+1}(j+1), T_i(j,h) + S_{i+1}(j+1))$ and its weight $(1 - p_{ij})\, w_i(j,h)$ in the bucket $B_{i+1}(j+1, \left\lfloor \frac{(T_i(j,h)+S_{i+1}(j+1))k}{(n+1)X} \right\rfloor)$.
l.14: **else**
l.15: Compute the payoff's conditional expectation Φ'' at the state $(S_{i+1}(j+1), T_i(j,h) + S_{i+1}(j+1))$, and let $\Phi := \Phi + (1 - p_{ij}) w_i(j,h)\, \Phi''$.
l.16: Output Φ.

Figure 3: The AMO algorithm

Each step during the computation of the value T_n yields the error of at most $(n+1)X/k$. Therefore, the contribution in one step to the error of the average stock value A_n is at most X/k, and the error in the estimation of A_n is bounded by nX/k. More precisely, the payoff Φ computed by the AMO algorithm satisfies $U - nX/k \leq \Phi \leq U$.

3.2 Extension to the Non-Uniform Model

While the AMO algorithm deals only with the uniform model in [1], we show that the AMO algorithm can be extended to the non-uniform model. Indeed, the extension is quite straightforward except for the computation of the payoff's conditional expectation when the current running total is above $(n+1)X$.

Suppose that we are at a node (i,j) in the i-th level, with the stock price $S = S_i(j)$ and the running total T $(\geq (n+1)X)$. Then, the payoff's conditional expectation is $\{T + \sum_{l=1}^{n-i}(1+r)^l S\}/(n+1) - X$ on the uniform model [1]. This shows that on the uniform model the conditional expectation of extra running total after the i-th level is path-independent and given by $\sum_{l=1}^{n-i}(1+r)^l S$.

On the non-uniform model, the conditional expectation of extra running total $h(i,j)$ at a node

(i,j) is still path-independent and can be computed by using the following recursive formula:

$$h(i,j) = \begin{cases} 0 & \text{if } i = n, \\ p_{ij}\{h(i+1,j) + S_{i+1}(j)\} + (1-p_{ij})\{h(i+1,j+1) + S_{i+1}(j+1)\} & \text{if } i < n. \end{cases}$$

It takes $\mathrm{O}(n^2)$ time to compute $h(i,j)$ for all nodes (i,j), which does not affect the time complexity for the remaining part of the algorithm.

Theorem 3.1. *Let k be any positive integer. For a European-Asian option on a non-uniform binomial tree model, the AMO algorithm computes a value Φ satisfying $U - nX/k \le \Phi \le U$ in $\mathrm{O}(n^2 k)$ time.*

3.3 Our Modified Algorithm

The difference between our algorithm and the AMO algorithm is only in how to approximate the states in a bucket. To approximate the states in a bucket $B_i(j,h)$, we randomly choose a representative state among the states in the bucket, where a state with weight w is chosen with the probability $w/w_i(j,h)$. That is, our algorithm is obtained by replacing *l*.7 of the AMO algorithm in Figure 3 with the following:

> *l*.7: Choose a state $(S_i(j), T)$ in the bucket $B_i(j,h)$ randomly, where a state with weight w is chosen with probability $w/w_i(j,h)$. Let $T_i(j,h) = T$.

At a glance, this modification seems merely a heuristic, and does not lead to the improvement in the theoretical bound. Let Ψ be the payoff value computed by our modified algorithm. Indeed, the error caused in one step is X/k in the worst case, and hence we can only prove that the worst case error bound $|U - \Psi|$ is nX/k as in the AMO algorithm. The modified algorithm, however, can also be viewed as sampling of paths, since each state stored in the buckets is equal to that of some existing path. In our algorithm, the selection of paths is smartly done during the runtime of the algorithm; in each step paths are clustered by using buckets, and a single path is selected from each bucket.

Since our algorithm is randomized, Ψ is a random variable depending on the coin-flips to choose representatives of running totals in the buckets. Let Y_i be a random variable giving the future value of the payoff after running our algorithm up to the i-th level, i.e., after the choice of representatives in all buckets has been determined up to the i-th level. By definition, $Y_0 = U$ and $Y_n = \Psi$.

The following lemma shows that random variables $Y_0, Y_1, \ldots, Y_n$ constitute a *Martingale sequence*.

Lemma 3.2. $E(Y_i \mid Y_0, Y_1, Y_2, .., Y_{i-1}) = Y_{i-1}$ *for* $i = 1, 2, \ldots, n$.

Proof. Consider the set $\{a_1, a_2, \ldots, a_q\}$ of states in a bucket at a node of the i-th level before selecting a representative. For $l = 1, 2, \ldots, q$, let $Y(a_l)$ be the expected payoff (exactly computed from the model) for a path with the state a_l, and $w(a_l)$ be the weight of a_l. If the state a_l is selected, it contributes $Y(a_l)W$ to the payoff, where $W = \sum_{l=1}^{q} w(a_l)$. Thus, the expected contribution of the states after the selection is $\sum_{l=1}^{q}(w(a_l)/W)Y(a_l)W = \sum_{l=1}^{q} w(a_l)Y(a_l)$, where the right-hand side is the expected contribution before the selection. □

Lemma 3.2 also shows that the expected value of the payoff Ψ equals the exact value U of the expected payoff, i.e., $E(Y_n) = E(\Psi) = U$. From the argument in Section 3.1, we have $|Y_i - Y_{i-1}| < X/k$. Thus, famous Azuma's inequality [3] applies (see [11, Theorem 4.16] for the present form).

Theorem 3.3 (Azuma's inequality). *Let $Z_0, Z_1, \ldots$ be a Martingale sequence such that $|Z_k - Z_{k-1}| < c_k$ for each k, where c_k is a constant depending on k. Then, we have*

$$\Pr[|Z_t - Z_0| \geq \lambda] \leq 2\exp\left(-\frac{\lambda^2}{2\sum_{k=1}^{t} c_k^2}\right) \qquad (\forall t = 1, 2, \ldots,\ \forall \lambda > 0).$$

In our case, we have $Y_0 = U$, and $c_k = X/k$. Hence, Azuma's inequality implies

$$\Pr[|Y_n - U| \geq \lambda] \leq 2\exp\left(-\frac{\lambda^2 k^2}{2nX^2}\right) \qquad (\forall \lambda > 0).$$

Thus, we can estimate the error bound of $\Psi = Y_n$ as an approximation of U as follows:

Theorem 3.4. *Let k be any positive integer and c be any positive real number. For a European-Asian option on a non-uniform binomial tree model, our algorithm computes in $O(n^2 k)$ time a value Ψ satisfying $|\Psi - U| \leq c\sqrt{n}X/k$ with probability at least $1 - 2e^{-c^2/2}$.*

It should be noted that the original AMO algorithm and its variants [2, 9] have an important advantage that they give a lower bound Φ of U, whereas Ψ computed by our algorithm is not necessarily a lower bound of U.

3.4 Experimental Results

We show some experimental results to illustrate the performance of our randomized approximation algorithm. In particular, we compare the quality of the option price computed by our algorithm with those by various other algorithms. We implemented the full path method to compute the exact price, and approximation algorithms such as the naive Monte Carlo method (MC), the AMO algorithm (AMO-LB), and the "basic algorithm" in Dai–Huang–Lyuu [9] (DHL-LB), and also variants of AMO-LB and DHL-LB which compute upper bounds of the option's exact price (AMO-UB, DHL-UB). The experiment is done by a Pentium IV 2.60CGHz PC and all programs are implemented in C++.

In the experiment, we consider a uniform model with $S_0 = X = 100$, $u = 1.1$, $d = 1/u$, $pu + (1 - p)d = (1.06)^{1/n}$. The parameter k is set to 1,000 in each of AMO-LB, AMO-UB, DHL-LB, DHL-UB, and our algorithm. While exactly k buckets are used at each node in the AMO algorithms and ours, k is the average number of buckets used at each node in the DHL algorithms. In the experiment, only one trial is made for each of the Monte Carlo method and our randomized algorithm.

Figure 4 gives the result of the experiment when $10 \leq n \leq 35$, showing the ratio of the approximate prices computed by approximation algorithms to the exact price computed by the full-path method. The running time of AMO-LB, DHL-LB, and ours are almost the same, and the Monte Carlo method takes $1500n$ sample paths so that it runs in almost the same time as other three algorithms. Note that full-path method takes more than 9 hours when $n = 35$. The relative error of our algorithm is always less than 0.0004, and smaller than those of the other algorithms with factor up to about 100 in the range $25 \leq n \leq 35$. This result shows that the error bound of our algorithm is much better than the theoretical bound $c\sqrt{n}X/k$. When $n = 35$, the exact price is 14.639494 and the relative error of the payoff obtained by our algorithm is less than 0.00005.

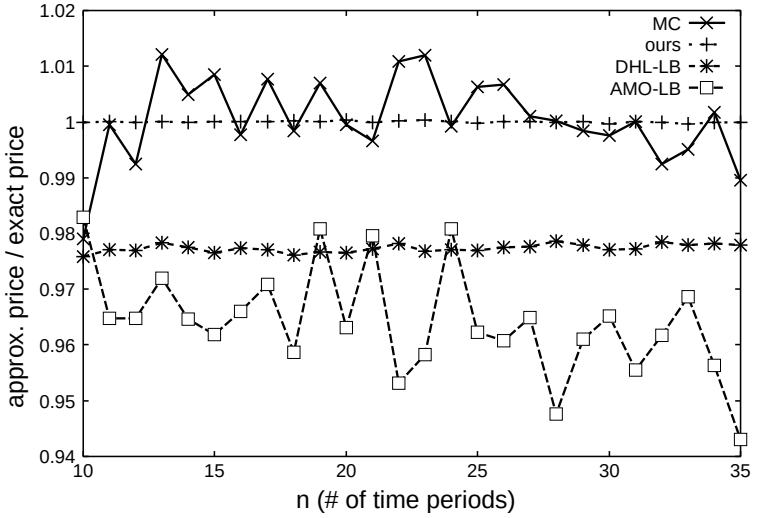

Figure 4: Errors of option prices computed by four approximation algorithms

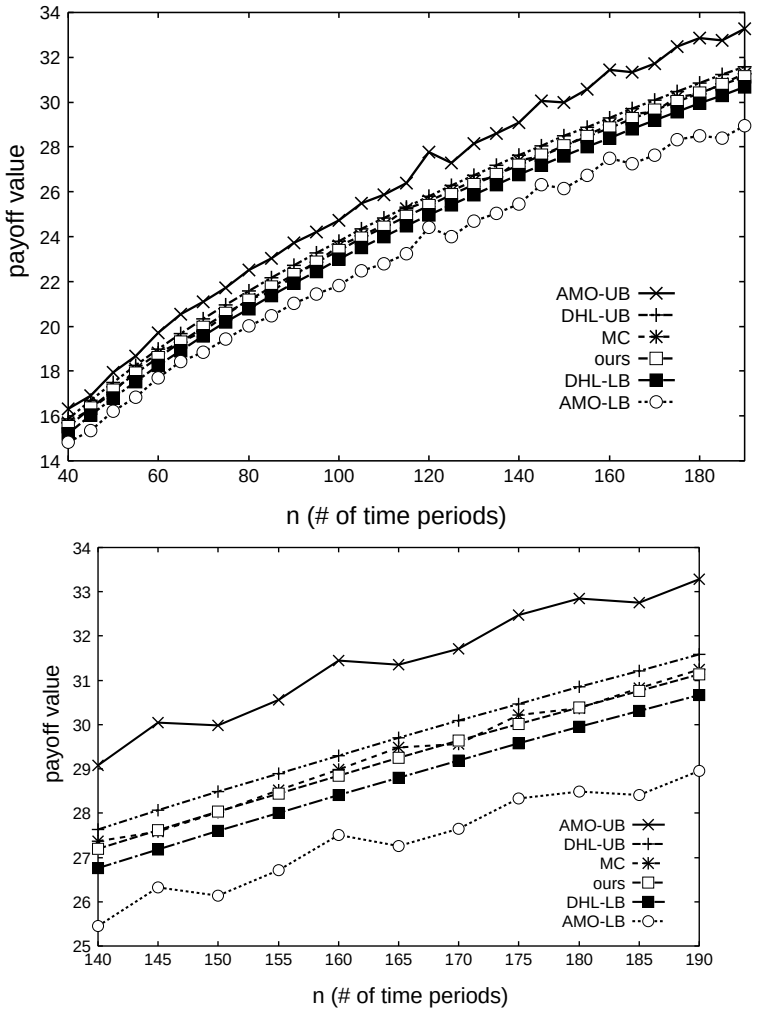

Figure 5: Option prices computed by approximation algorithms. Top: original graph, Bottom: enlarged graph in the range $[140, 190]$.

We also run the naive Monte Carlo method spending computation time 100 times more, but its accuracy is still worse than that of our algorithm. Note that we do not implement the AMO algorithm with flexible bucket size, which is a heuristic reported to be better than the original AMO algorithm [1], since the performance depends on tuning of parameters and the heuristic can be also combined with our algorithm.

Figure 5 shows the result when $40 \leq n \leq 190$. In this case, we do not compute the exact price of the option since the full-path method takes at least 10 days even when $n = 40$. Instead, we apply AMO-UB and DHL-UB to compute upper bounds of the exact price. It can be observed from Figure 5 that the option price computed by our algorithm is quite stable.

3.5 More Precise Analysis for the Uniform Model

Experimental results shown above indicate that the analysis in Section 3.3 overestimates the error. In this section, we restrict our attention to the uniform model, and analyze the error bound of our algorithm more precisely. In the following, we assume $0 < p < 1$ and put $\alpha = \max\{p, 1-p\}$. We also assume, w.l.o.g., that $p \geq 0.5$ since the case $p < 0.5$ can be dealt with in the same way.

To refine the analysis of the random process, we consider that the algorithm processes nodes of the i-th level one by one. We group the coin flips for the buckets of a node into one random process.

Let $Y_{i,j}$ be the random variable giving the future value of the payoff just after the algorithm processes the node (i, j) in the i-th level. Thus, we have a random process with $\sum_{i=0}^{n}(i+1) = (n+1)(n+2)/2$ steps. When the algorithm processes a node (i, j), the running totals of the paths terminating at (i, j) are approximated with the error less than X/k, and the running totals of other paths remain the same. Hence, the sequence $Y_{0,0}, Y_{1,0}, Y_{1,1}, \ldots, Y_{n,n-1}, Y_{n,n}$ is a Martingale process satisfying

$$\begin{aligned} |Y_{i,j+1} - Y_{i,j}| &< \omega(i, j+1)X/k \qquad \text{if } 0 \leq j < i \leq n, \\ |Y_{i+1,0} - Y_{i,i}| &< \omega(i+1, 0)X/k \qquad \text{if } 0 \leq i < n, \end{aligned} \tag{1}$$

where $\omega(i, j) = \binom{i}{j} p^{i-j}(1-p)^j$ denotes the total weight of the paths terminating at (i, j).

In order to apply Azuma's inequality to the analysis of the error bound, we estimate the value

$$\Gamma_p(n) \equiv \sum_{i=1}^{n} \sum_{j=0}^{i} \omega(i, j)^2$$

as a function in n. As shown below, we have $\Gamma_p(n) = \mathrm{O}(\sqrt{n})$ for any fixed p with $0 < p < 1$. Hence, we can show the following error bound.

Theorem 3.5. *Let k be any positive integer and c be any positive real number. For a European-Asian option on a uniform binomial tree model with $0 < p < 1$, our algorithm computes in $\mathrm{O}(n^2 k)$ time a value Ψ satisfying $|\Psi - U| = \mathrm{O}(cn^{1/4}X/k)$ with probability at least $1 - 2e^{-c^2/2}$.*

Proof. Since (1) holds, Theorem 3.3 yields the following inequality:

$$\Pr[|Y_{n,n} - U| \geq \lambda] \leq 2\exp\left(-\frac{\lambda^2}{2\sum_{i=1}^{n}\sum_{j=0}^{i}(\omega(i, j)X/k)^2}\right) = 2\exp\left(-\frac{\lambda^2 k^2}{2X^2\Gamma_p(n)}\right),$$

which, together with $\Gamma_p(n) = \mathrm{O}(\sqrt{n})$, implies the statement of the theorem. □

We now start the analysis of the value $\Gamma_p(n)$, where we use well-known Stirling's formula:

Proposition 3.6 (Stirling's formula). $m! \sim \sqrt{2\pi m}(m/e)^m$.

We can prove $\Gamma_p(n) = \mathrm{O}(\sqrt{n})$ easily by using Stirling's formula when $p = 1/2$:

$$\Gamma_{1/2}(n) = \frac{1}{2^{2n}} \sum_{i=1}^{n} \sum_{j=0}^{i} \left[\binom{i}{j} \right]^2 = \frac{1}{2^{2n}} \sum_{i=1}^{n} \binom{2i}{i} \sim \sum_{i=1}^{n} \frac{1}{\sqrt{\pi i}} = \mathrm{O}(\sqrt{n}).$$

In general, we have $\Gamma_{1/2}(n) \le \Gamma_p(n) \le n = \Gamma_0(n) = \Gamma_1(n)$, and $\Gamma_p(n)$ converges to n when p converges 0 or 1. Hence, if p is close to 0 or 1 then the analysis of $\Gamma_p(n)$ becomes more complicated. In the following, we give a rigorous proof for $\Gamma_p(n) = \mathrm{O}(\sqrt{n})$.

Given a small positive constant δ, let M be a sufficiently large positive integer such that

$$1 - \delta < \frac{m!}{\sqrt{2\pi m}(m/e)^m} < 1 + \delta \qquad (\forall m \ge M). \tag{2}$$

The following analysis ignores the small number δ to improve the readability since our final result (Theorem 3.5) only gives an asymptotic error bound.

For a small positive constant ε, we take a constant β satisfying

$$1/2 < \beta < 1, \qquad \beta^\beta (1-\beta)^{1-\beta} > (1+\varepsilon)\alpha, \tag{3}$$

where such β always exists since $\lim_{\beta \to 1} \beta^\beta (1-\beta)^{1-\beta} = 1$. For example, if $\alpha = 2/3$, we can take $\beta = 0.9$ to attain $\beta^\beta (1-\beta)^{1-\beta} \sim 0.72$.

Lemma 3.7. *For any integer i with $i - \lceil \beta i \rceil \ge M$, the following inequalities hold:*

$$\sum_{j=0}^{i-\lceil \beta i \rceil} \omega(i,j)^2 < \frac{1}{2\pi\beta(1-\beta)(1+\varepsilon)^{2i}}, \tag{4}$$

$$\sum_{j=i-\lceil \beta i \rceil + 1}^{\lceil \beta i \rceil - 1} \omega(i,j)^2 < \frac{1}{\sqrt{\pi\beta(1-\beta)i}}, \tag{5}$$

$$\sum_{j=\lceil \beta i \rceil}^{i} \omega(i,j)^2 < \frac{1}{2\pi\beta(1-\beta)(1+\varepsilon)^{2i}}. \tag{6}$$

Proof. We first prove the inequality (6). Since $1 - p \le p = \alpha$, it holds that

$$\sum_{j=\lceil \beta i \rceil}^{i} \omega(i,j)^2 \le \alpha^{2i} \sum_{j=\lceil \beta i \rceil}^{i} \left[\binom{i}{j} \right]^2 \le \alpha^{2i} (i - \lceil \beta i \rceil - 1) \left[\binom{i}{\lceil \beta i \rceil} \right]^2 \le \alpha^{2i} i \left[\binom{i}{\lceil \beta i \rceil} \right]^2,$$

where the second inequality is by $j \ge \lceil \beta i \rceil > i/2$. By using Stirling's formula and (3), we have

$$\begin{aligned}
\binom{i}{\lceil \beta i \rceil} &\sim \frac{1}{\sqrt{2\pi}} \frac{i^{i+\frac{1}{2}}}{(\lceil \beta i \rceil)^{\lceil \beta i \rceil + \frac{1}{2}} \cdot (i - \lceil \beta i \rceil)^{i - \lceil \beta i \rceil + \frac{1}{2}}} \\
&\le \frac{1}{\sqrt{2\pi}} \frac{i^{i+\frac{1}{2}}}{(\beta i)^{\beta i + \frac{1}{2}} \cdot (i - \beta i)^{i - \beta i + \frac{1}{2}}} \\
&= \frac{1}{\sqrt{2\pi\beta(1-\beta)i}} \left(\frac{1}{\beta^\beta (1-\beta)^{1-\beta}} \right)^i < \frac{1}{\sqrt{2\pi\beta(1-\beta)i}\, \alpha^i (1+\varepsilon)^i},
\end{aligned}$$

where the second inequality follows from the fact that $\log(x^{x+\frac{1}{2}})$ is a convex function in x in the interval $[1/2, +\infty)$. Hence (6) follows. The proof of (4) is the same as that for (6) and therefore omitted.

Finally, we prove (5). For any j with $i - \beta i < j < \beta i$, Stirling's formula implies

$$\binom{2i}{2j} \Big/ \left[\binom{i}{j}\right]^2 = \binom{2i}{i} \Big/ \left[\binom{2j}{j}\binom{2i-2j}{i-j}\right] \sim \sqrt{\frac{\pi j(i-j)}{i}} > \sqrt{\pi\beta(1-\beta)i},$$

where the inequality follows from the inequality $i - \beta i < j < \beta i$ and the concavity of the function $\psi(x) = x(i-x)$. Hence, we have

$$\sqrt{\pi\beta(1-\beta)i} \cdot \sum_{j=i-\lceil\beta i\rceil+1}^{\lceil\beta i\rceil-1} \omega(i,j)^2 < \sum_{j=i-\lceil\beta i\rceil+1}^{\lceil\beta i\rceil-1} \binom{2i}{2j}[p^{i-j}(1-p)^j]^2 \;\le\; \sum_{j=0}^{2i}\binom{2i}{j}p^{2i-j}(1-p)^j = 1,$$

from which (5) follows. □

Lemma 3.8. *We have*

$$\Gamma_p(n) < \frac{1}{\pi\beta(1-\beta)\varepsilon} + \frac{2\sqrt{n}}{\sqrt{\pi\beta(1-\beta)}} + C, \tag{7}$$

where C is a constant given by

$$C = \sum_i \left\{ \sum_{j=0}^{i} \omega(i,j)^2 \;\middle|\; i \ge 1,\ i - \lceil\beta i\rceil < M \right\}.$$

Proof. By Lemma 3.7, it holds that

$$\Gamma_p(n) = \sum_{i=1}^{n}\sum_{j=0}^{i}\omega(i,j)^2 < \sum_{i=1}^{n}\left\{\frac{1}{\pi\beta(1-\beta)(1+\varepsilon)^{2i}} + \frac{1}{\sqrt{\pi\beta(1-\beta)i}}\right\} + C,$$

from which follows (7) since $\sum_{i=1}^{n}(1+\varepsilon)^{-2i} \le \sum_{i=1}^{\infty}(1-\varepsilon)^i < 1/\varepsilon$ and $\sum_{i=1}^{n}(1/\sqrt{i}\,) \le 2\sqrt{n}$. □

This lemma shows that $\Gamma_p(n) = \mathrm{O}(\sqrt{n})$ for any fixed p with $0 < p < 1$, which concludes the proof of Theorem 3.5.

4 Pricing Saving-Asian Options

We present an approximation algorithm for the expected payoff of Saving-Asian options proposed in Section 2.3, which can be obtained by slightly modifying our algorithm for European-Asian options.

We first show that if the running total of the current state is above the threshold $(n+1)X$, then the conditional expectation of the payoff can be computed analytically, as in the case of European-Asian options. For each node (i, j) of a binomial tree, we define two values $f(i, j)$ and $g(i, j)$ to decide whether to exercise the option at the node (i, j). For a leaf node (n, j), we define

$$f(n,j) = 0, \qquad g(n,j) = \frac{S_n(j) - X}{n+1}.$$

If $i < n$, then we recursively define

$$f(i,j) = \max\{p_{ij}g(i+1,j) + (1-p_{ij})g(i+1,j+1), 0\}, \qquad g(i,j) = \frac{S_i(j) - X}{n+1} + f(i,j).$$

All the values $f(i,j)$ and $g(i,j)$ can be computed in a bottom-up fashion in $O(n^2)$ time. The value $f(i,j)$ (≥ 0) denotes the "value" of the right of postponing the exercise; $f(i,j) = 0$ means that there is no merit in postponing the exercise, and $f(i,j) > 0$ means that the conditional expectation of the extra payoff obtained by postponing the exercise is given by $p_{ij}g(i+1,j)+(1-p_{ij})g(i+1,j+1)$ (> 0). Hence, the conditional expectation of the payoff can easily be computed when the current running total is above $(n+1)X$.

Proposition 4.1. *Suppose that we are at the node (i,j) and the current running total T is above the threshold $(n+1)X$. Then, the conditional expectation of the payoff is $(T-(i+1)X)/(n+1)+f(i,j)$.*

To approximate the expected payoff of Saving-Asian options, we first run our algorithm for European-Asian options explained in Section 3. We then backtrack the process of the algorithm from leaves to the root of the binomial tree so that we can find early-exercise states and compute (approximate) conditional expectation of the payoff (including the interest obtained by re-investment) at each state.

Suppose that we are at a state $(S_i(j), T_i)$ in the i-th level during the backtracking process. Note that the option should be exercised at this state if the payoff obtained by the early exercise at this state is more than the weighted average of the payoffs of its two "child" states $(S_{i+1}(j), T'_{i+1})$ and $(S_{i+1}(j+1), T''_{i+1})$ in the $(i+1)$-st level. If the current running total T_i is above the threshold $(n+1)X$, the conditional expectation of the payoff can be analytically computed as shown in Proposition 4.1. If $T_i < (n+1)X$, then the conditional expectation of the payoff is given by

$$\max\left\{\frac{T_i - (i+1)X}{n+1},\ p_{ij}\eta' + (1-p_{ij})\eta''\right\},$$

where $(T_i - (i+1)X)/(n+1)$ is the payoff obtained by the early exercise at this state, and η' and η'' are the (approximate) conditional expectation of the payoff at the "child" states $(S_{i+1}(j), T'_{i+1})$ and $(S_{i+1}(j+1), T''_{i+1})$, respectively.

In this way, we compute the (approximate) expected payoff of all states from the n-th level to the 0-th level. Then, the expected payoff at the unique state in the 0-th level is an approximate value of the option's expected payoff. It is easy to see that the backtracking process requires additional $O(n^2k)$ time, which does not affect the total time complexity. Hence, we have the following theorem, where the error bound can be analyzed in the similar way to Theorems 3.4 (the non-uniform model) and 3.5 (the uniform model).

Theorem 4.2. *Let k be any positive integer and c be any positive real number. For a Saving-Asian option on a non-uniform binomial tree model, our algorithm computes in $O(n^2k)$ time a value Ψ satisfying $|\Psi - U| \leq c\sqrt{n}X/k$ with probability at least $1 - 2e^{-c^2/2}$. Moreover, on a uniform model the error bound is refined to $O(cn^{1/4}X/k)$.*

5 Concluding Remarks

In this paper, we proposed a randomized approximation algorithm for pricing European-Asian options, and then generalized the algorithm to Saving-Asian options. Our algorithm can be further generalized to path-dependent options satisfying the following conditions:

- The payoff of early exercise at a node (i, j) in the i-th level after the movement of stock price $\mathcal{P} = \{S_0, S_1, \ldots, S_i\}$ is given by $\gamma(i)\{\sum_{t=0}^{i} f_t(S_t) - G(i, j)\}$, where $0 < \gamma(i) \leq 1$, $G(i, j) \geq 0$, and each f_t is a nondecreasing function in the stock value S_t.
- There is a threshold value B such that once $\sum_{t=0}^{i} f_t(S_t) \geq B$, the option should be exercised in the future.
- Once $\sum_{t=0}^{i} f_t(S_t)$ exceeds the threshold B, the difference between the payoff (including the interest) of immediate exercise and the expected payoff obtained by delaying the exercise is path-independent.

For example, $\gamma(i) = e^{-(n-i)r_0/n}$, $f_t(S_t) = S_t/(n+1)$, $G(i, j) = (i+1)X/(n+1)$, and $B = X$ for Saving-Asian options. Thus, we may add a path-independent function to the payoff of Saving-Asian options without losing the accuracy; for example, we may design a variation of Saving-Asian options which pay back a portion of the price for early exercise.

It is often the case in the real financial market that an upper limit L is set to the payoff value of an option since otherwise the seller may not afford to pay the payoff. For European-Asian options (resp., Saving-Asian options) with this modification, if $T_i > (n+1)L+(n+1)X$ (resp., $T_i > (n+1)L+(i+1)X$) happens, the holder will simply receive the upper limit payoff. Our algorithm can be easily modified to have the error bound $\mathrm{O}(n^{1/4}(L+X)/k)$ and the time complexity $\mathrm{O}(n^2k)$ on the uniform model. If L is small (say, $\mathrm{O}(X)$), our algorithm keeps high theoretical accuracy. However, if L is very large, then the current error bound is not good, and some other techniques are required for better accuracy.

The experimental results in Section 3.4 imply that our theoretical analysis is not tight; indeed, we may further refine the random process by regarding each coin flip (or set of coin flips) at a bucket of each node as one step of a Martingale process. Intuitively, if n is large enough, this will further improve the error analysis by a factor up to $\sqrt{k}$, which explains why the experimental performance is so good; however, the theoretical analysis seems complicated and difficult.

It is also valuable to investigate the applicability of the AMO algorithm and our algorithm to several existing options, and to design new useful options based on the insight obtained in this paper.

Acknowledgement

The authors thank Prof. Syoiti Ninomiya of Tokyo Institute of Technology for his counsel on financial engineering, and anonymous referees for their many helpful comments for improving the paper. This work is supported by Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan.

References

[1] D. Aingworth, R. Motwani, and J. D. Oldham, Accurate approximations for Asian options, *Proc. 11th ACM-SIAM Symposium on Discrete Algorithms* (2000), 891–900.

[2] K. Akcoglu, M.-Y. Kao, and S. V. Raghavan, Fast pricing of European Asian options with provable accuracy: single-stock and basket options, *Proc. Annual European Symposium on Algorithms 2001, Lecture Notes in Computer Science* **2161** (2001), 404–415.

[3] K. Azuma, Weighted sum of certain dependent random variables, *Tohoku Mathematical Journal* **19** (1967) 357–367.

[4] H. Ben-Ameur, M. Breton, and P. L'Ecuyer, A dynamic programming procedure for pricing American-style Asian options, *Management Science* **48** (2002), 625–643.

[5] P. Chalasani, S. Jha, F. Egriboyun, and A. Varikooty, A refined binomial lattice for pricing American Asian options, *Review of Derivatives Research* **3** (1999), 85–105.

[6] P. Chalasani, S. Jha, and I. Saias, Approximate option pricing, *Algorithmica* **25** (1999), 2–21.

[7] P. Chalasani, S. Jha, and A. Varikooty, Accurate approximation for European Asian options, *Journal of Computational Finance* **1** (1998), 11–29.

[8] J. C. Cox, S. A. Ross, and M. Rubenstein, Option pricing: a simplified approach, *Journal of Financial Economics* **7** (1979), 229–263.

[9] T.-S. Dai, G.-S. Huang, and Y.-D. Lyuu, Extremely accurate and efficient tree algorithms for Asian options with range bounds, 2002 NTU International Conference on Finance, National Taiwan University, Taiwan, May 2002.
`http://www.fin.ntu.edu.tw/~conference2002/proceeding/5-2.html`

[10] P. Glasserman, *Monte Carlo Method in Financial Engineering*, Springer, Berlin, 2004.

[11] R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, London, 1995.

[12] R. Wu and M.C. Fu, Optimal exercise policies and simulation-based valuation for American-Asian options, *Operations Research* **51** (2003), 52–66.

Appendix for Referees: Proofs in Details

Detailed Proof of Lemma 3.7: We first prove the inequality (6). Since $1-p \le p = \alpha$, it holds that

$$\sum_{j=\lceil \beta i \rceil}^{i} \omega(i,j)^2 \le \alpha^{2i} \sum_{j=\lceil \beta i \rceil}^{i} \left[\binom{i}{j} \right]^2 \le \alpha^{2i}(i - \lceil \beta i \rceil - 1) \left[\binom{i}{\lceil \beta i \rceil} \right]^2 \le \alpha^{2i}\, i \left[\binom{i}{\lceil \beta i \rceil} \right]^2, \tag{8}$$

where the second inequality is by $j \ge \lceil \beta i \rceil > i/2$.

Claim 1:

$$(\beta i)^{\beta i + \frac{1}{2}} \cdot (i - \beta i)^{i - \beta i + \frac{1}{2}} \le (\lceil \beta i \rceil)^{\lceil \beta i \rceil + \frac{1}{2}} \cdot (i - \lceil \beta i \rceil)^{i - \lceil \beta i \rceil + \frac{1}{2}}.$$

[Proof of Claim] Since the function $\varphi(x) = \log(x^{x+\frac{1}{2}}) = (x + \frac{1}{2}) \log x$ is a convex function in x in the interval $[1/2, \infty)$, we have

$$\log[(x-d)^{x-d+\frac{1}{2}} \cdot (y+d)^{y+d+\frac{1}{2}}] = \varphi(x-d) + \varphi(y+d) \le \varphi(x) + \varphi(y) = \log[x^{x+\frac{1}{2}} \cdot y^{y+\frac{1}{2}}]$$

for any $x, y \in \mathbf{R}$ with $x > y > 0$ and $d \in [0, (x-y)/2]$. Hence, we have

$$(x-d)^{x-d+\frac{1}{2}} \cdot (y+d)^{y+d+\frac{1}{2}} \le x^{x+\frac{1}{2}} \cdot y^{y+\frac{1}{2}}.$$

Then, the statement of the claim follow immediately from this inequality by putting $x = \lceil \beta i \rceil$, $y = i - \lceil \beta i \rceil$, and $d = \lceil \beta i \rceil - \beta i$. [End of Claim]

Then, we have

$$\begin{aligned}
\binom{i}{\lceil \beta i \rceil} &= \frac{i!}{\lceil \beta i \rceil ! (i - \lceil \beta i \rceil)!} \\
&< \frac{1+\delta}{(1-\delta)^2} \frac{\sqrt{2\pi i}(i/e)^i}{\sqrt{2\pi \lceil \beta i \rceil}(\lceil \beta i \rceil / e)^{\lceil \beta i \rceil} \cdot \sqrt{2\pi(i - \lceil \beta i \rceil)}((i - \lceil \beta i \rceil)/e)^{i - \lceil \beta i \rceil}} \\
&= \frac{1+\delta}{(1-\delta)^2} \frac{1}{\sqrt{2\pi}} \frac{i^{i+\frac{1}{2}}}{(\lceil \beta i \rceil)^{\lceil \beta i \rceil + \frac{1}{2}} \cdot (i - \lceil \beta i \rceil)^{i - \lceil \beta i \rceil + \frac{1}{2}}} \\
&\le \frac{1+\delta}{(1-\delta)^2} \frac{1}{\sqrt{2\pi}} \frac{i^{i+\frac{1}{2}}}{(\beta i)^{\beta i + \frac{1}{2}} \cdot (i - \beta i)^{i - \beta i + \frac{1}{2}}} \\
&= \frac{1+\delta}{(1-\delta)^2} \frac{1}{\sqrt{2\pi\beta(1-\beta)i}} \left(\frac{1}{\beta^\beta (1-\beta)^{1-\beta}} \right)^i \\
&< \frac{1+\delta}{(1-\delta)^2} \frac{1}{\sqrt{2\pi\beta(1-\beta)i}\, \alpha^i (1+\varepsilon)^i},
\end{aligned} \tag{9}$$

where the first inequality is by (2), the second by Claim 1, and the third by (3). By the inequalities (8) and (9), we have

$$\sum_{j=\lceil \beta i \rceil}^{i} \omega(i,j)^2 < \alpha^{2i}\, i \cdot \frac{(1+\delta)^2}{(1-\delta)^4} \frac{1}{2\pi\beta(1-\beta)i} \frac{1}{\alpha^{2i}(1+\varepsilon)^{2i}} = \frac{(1+\delta)^2}{(1-\delta)^4} \frac{1}{2\pi\beta(1-\beta)} \frac{1}{(1+\varepsilon)^{2i}}.$$

This implies (6) if we regard δ as 0.

The proof for (4) is the same as that for (6) and therefore omitted.

Finally, we prove (5).

Claim 2: If $i - \beta i < j < \beta i$, then

$$\sqrt{\pi\beta(1-\beta)i}\left[\binom{i}{j}\right]^2 < \binom{2i}{2j}.$$

[Proof of Claim] For $h \geq M$, (2) implies

$$\frac{1-\delta}{(1+\delta)^2}\frac{2^{2h}}{\sqrt{\pi h}} < \binom{2h}{h} < \frac{1+\delta}{(1-\delta)^2}\frac{2^{2h}}{\sqrt{\pi h}}.$$

Hence, it holds that

$$\begin{aligned}
\binom{2i}{2j} \Big/ \left[\binom{i}{j}\right]^2 &= \binom{2i}{i} \Big/ \left[\binom{2j}{j}\binom{2(i-j)}{(i-j)}\right] \\
&> \frac{(1-\delta)^5}{(1+\delta)^4}\frac{2^{2i}/\sqrt{\pi i}}{(2^{2j}/\sqrt{\pi j})(2^{2(i-j)}/\sqrt{\pi(i-j)})} \\
&= \frac{(1-\delta)^5}{(1+\delta)^4}\sqrt{\frac{\pi j(i-j)}{i}} \;>\; \frac{(1-\delta)^5}{(1+\delta)^4}\sqrt{\pi\beta(1-\beta)i},
\end{aligned}$$

where the second inequality is by the inequality $i - \beta i < j < \beta i$ and the concavity of the function $\psi(x) = x(i-x)$. This implies the statement of the claim if we regard δ as 0. [End of Claim]

Claim 2 implies

$$\begin{aligned}
\sqrt{\pi\beta(1-\beta)i}\cdot \sum_{j=i-\lceil\beta i\rceil+1}^{\lceil\beta i\rceil-1} \omega(i,j)^2 &< \sum_{j=i-\lceil\beta i\rceil+1}^{\lceil\beta i\rceil-1} \binom{2i}{2j}[p^j(1-p)^{i-j}]^2 \\
&\leq \sum_{j=0}^{2i}\binom{2i}{j}p^{2i-j}(1-p)^j \;=\; \{p+(1-p)\}^{2i} \;=\; 1,
\end{aligned}$$

from which (5) follows. □

Detailed Proof of Lemma 3.8. By Lemma 3.7, it holds that

$$\Gamma_p(n) < \sum_{i=1}^{n}\left\{\frac{2}{2\pi\beta(1-\beta)(1+\varepsilon)^{2i}} + \frac{1}{\sqrt{\pi\beta(1-\beta)i}}\right\} + C. \tag{10}$$

Since ε is a sufficiently small positive number, we have $(1+\varepsilon)^{-2} \leq 1-\varepsilon$, and therefore

$$\sum_{i=1}^{n}\frac{1}{(1+\varepsilon)^{2i}} \leq \sum_{i=1}^{\infty}(1-\varepsilon)^i = \frac{1-\varepsilon}{\varepsilon} < \frac{1}{\varepsilon}. \tag{11}$$

We also have

$$\sum_{i=1}^{n}\frac{1}{\sqrt{i}} \leq \int_0^n \frac{1}{\sqrt{i}} = 2\sqrt{n}. \tag{12}$$

Combining the inequalities (10), (11), and (12), we have

$$\Gamma < \frac{1}{\pi\beta(1-\beta)\varepsilon} + \frac{2\sqrt{n}}{\sqrt{\pi\beta(1-\beta)}} + C.$$

□