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

Kenichiro Ohta[1,2], Kunihiko Sadakane, Akiyoshi Shioura, Takeshi Tokuyama[1]

[1] GSIS Tohoku University. (ken,sada,shioura,tokuyama)@dais.is.tohoku.ac.jp
[2] DC Card Co., Ltd., Tokyo, Japan

Abstract. We present efficient and accurate approximation algorithms for computing the premium price of Asian options. First, we modify an algorithm developed by Aingworth et al. in SODA 2000 for pricing the Europian-Asian option and improve its accuracy (both theoretically and practically) by transforming it into a randomized algorithm. Then, we present a new option named Saving-Asian option, whose merit is in the middle of European-Asian and American-Asian options, and show that our method works for its pricing.

1 Introduction

Background
Options are popular financial derivatives. Options give the right, but not the obligation, to buy or sell something (we consider a stock in this paper) at some point in the future for a specified price (called *strike price*).

A simple option permits buying a stock at the end of the year for a predetermined price. If the stock is worth more than that price, then you can use the option to buy the stock for less than you otherwise could. The price of the option (called *premium* of the option) is usually much less than the underlying price of the stock. Use of options hedges risk more cheaply than using only stocks, and cheaply provides a chance to get large profit if one's speculation is good.

For example, if you are interested in a stock of a current price \$200, and forecast that it will possibly go up beyond \$300 in the year-end. You may buy 1000 units of the stock (probably falling in debt), and if your forecast comes true, you will gain \$100,000; however, if the stock price goes down to \$100, you will unfortunately lose \$100,000, which you will not be able to afford. Instead, suppose that you can buy at the premium \$8 an option that 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 *payoff*) for each unit by exercising the option and selling the stock at the market price. Thus, if you buy 1250 units of this option, you have a chance to gain total payoff of \$100,000 (without considering the debt for the premium) reducing the maximum loss to be \$10,000 (just the total premium). You may buy 2000 units of another option that has the strike price \$250 and the premium \$2, and dream to gain \$100,000 with the maximum loss \$4,000. But you will see that the latter option is not always better than the former one. Here, you must ask the question whether the option premiums

\$8 and \$2 are fair or not. Therefore, pricing the options is a central topic in financial engineering.

A standard method (Black-Scholes model) is to model the movement of the underlying financial asset as Brownian motion with drift and then to construct an arbitrage portfolio. This yields a stochastic differential equation, and its solution gives a premium of the option. However, it is often difficult to solve such a differential equation, and indeed no closed-form solution is known for the Asian option discussed in this paper.

Therefore, it is widely practiced to simulate the Brownian motion by using a combinatorial model, and to obtain a solution on the model, which we call the *combinatorial exact premium price* or the *exact premium price*, as an approximation of the premium price obtained from the differential equation. The binomial (or trinomial) model is a combinatorial model, in which the time period is decomposed into n time steps, and the Brownian motion is modeled by using a biased random walk on a directed acyclic graph named *recombinant binary (or trinary) tree* of depth n with $n(n+1)/2$ (or n^2) nodes. Although our algorithms and analysis can be easily adjusted to work on the trinomial model, we focus on the binomial model for simplicity. The binomial model is a very popular model since the combinatorial exact premium price converges to the premium price given by the differential equation if we enlarge the size of the model.

In the binomial model, the process of price-movement of a stock (or any financial asset on which the option is based) is represented by a path in the binary recombinant tree. An option is called *path-dependent* if its value at the time of *exercise* depends not only the current price but also the path representing the process. An option is called American type if it permits early exercise. Path-dependency is necessary for designing an option that is secure against the risk caused by sudden change of the market, and also right of early exercise is convenient for users. However, an option with both functions is often difficult to analyze.

Our Problems and Results

The Asian option is a kind of path-dependent options. If we simulate the Black-Scholes model accurately by using a binomial model, the size often becomes large. Unfortunately, it is known to be #P-hard to compute the exact premium price on the binomial model for a path-dependent option in general [5]. Therefore, we need to design an efficient approximation algorithm with a provable high accuracy.

The European Asian option is the simplest Asian option. A naive method (*full-path* method) for computing the exact premium price of an European Asian option is to enumerate all the paths in the model; unfortunately, there are exponential number of paths. Thus, a random sampling method is a popular way to obtain an approximate solution; however, taking a polynomial number of samples naively is not enough to assure a theoretically probable accuracy. There are several polynomial-time approximation algorithms for pricing European-Asian options [5, 6], based on sampling method. For path-dependent call options, howver, the approximation error of a sampling method with a polynomial number of samples has a lower bound that depends on the volatility of the random process represented by the binomial model; moreover, $O(n^4)$ time is necessary to attain the accuracy matching the lower bound.

Recently, Aingworth-Motwani-Oldham [1] gave a breakthrough idea which avoids 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 running an approximate aggregation algorithm based on dynamic programming. They proposed an $O(n^2k)$ time algorithm (referred to 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. Akcoglu et al.[2] presented efficient methods for the pricing of European Asian option, and by using a recursive version of AMO algorithm they reduce the error bound to $n^{\frac{1+\epsilon}{2}}X/k$ spending the same time complexity under the condition that the volatility of the stock is small.

In this paper, we first give a randomized algorithm with an $O(n^2k)$ time complexity and an $O(\sqrt{n}X/k)$ error bound for which we do not need a volatility condition. The algorithm is indeed a variation of AMO algorithm. The modification itself is quite small, and looks almost trivial at a glance. However, by this modification, the algorithm can be regarded as a variant of the sampling method (without limit of the above mentioned lower bound), as well as that of AMO algorithm. Thus, the algorithm can enjoy advantages of both methods simultaneously. Although algorithms on a uniform model has been mainly considered in the literature [1, 2] in algorithm theory, our algorithm and analysis work on a non-uniform model where the transition probabilities of the stock price may depend on the state of the graph modeling the process, and also work on a trinomial model. Moreover, the error bound can be improved to $O(n^{1/4}X/k)$ for the uniform case unless the transition probability p is extremely close to 1 or 0.

Our idea is the following: By considering a novel random variable, the aggregation process of the algorithm can be regarded as a Martingale process with n random steps. The expected value of its output equals the combinatorial exact price, and the error of its single step is bounded by X/k. Thus, we can apply Azuma's inequality [4] on the Martingale process to obtain the error bound. We show practical quality of our algorithm by an experiment: Indeed, its accuracy (for $n = 30$) is better by a factor nearly 100 than that of Aingworth-Motwani-Oldham's original algorithm.

Inspired from the analysis, we propose an intermediate option between American-Asian and European-Asian options, and show that our method also works for this option. Our option, which we name *Saving-Asian option*, permits early exercise, but the payoff system is different from the American option, so that we can anticipate the action of users and compute the expected payoff accurately. The payoff depends on the average stock price, and hence secures against sudden change of the market. Moreover, compared to the American-Asian option, the Saving-Asian option reduces the risk for the seller; thus the premium is cheaper. Therefore, we believe that our new option and its analysis will be useful in both theory and practice.

2 Preliminaries

We divide the period from the purchase date to the expiration date of an option into n time periods, and the t-th time step is the end of the t-th time period. Let S_t $(t = 0, 1, 2, \ldots, n)$ be a random variable representing a stock price at the t-th time step, where S_0 is a constant known as the initial price.

Let X be the strike price of the option. *Payoff* is the value of the option, which is a random variable. In Black-Scholes' theory, the option premium is computed from the expected value of the payoff by subtracting the interest on the premium during the period, and hence it suffices to compute (or approximate) the expected value of the payoff.

2.1 Options

We only consider *call options* in this paper, although pricing of corresponding *put options* can be similarly (and more easily) done. We adopt a convention to write F^+ for $\max\{F, 0\}$.

The **European call option** is the most basic option, and its payoff $(S_n - X)^+ = \max\{S_n - X, 0\}$ is determined by the stock price of the expiration date (i.e., at the n-th time step). Note that S_n above is the real stock value that is revealed on the expiration date.

It is quite easy to compute the expected value of the payoff of the European call option if we use the binomial model. A drawback of the European option is that the payoff may be changed drastically by the movement of the stock price just before the expiration date; thus, even if the stock price goes very high during most of the period, it may happen that the option does not make money at the end.

The **European-Asian Option** is an option that can resolve the above-mentioned drawback of European option. The payoff of the European-Asian option is $(A_n - X)^+$, where $A_n = (\sum_{i=1}^{n} S_i)/n$ is the average of the stock prices during the period. Let $T_j = \sum_{i=1}^{j} S_i$ be the running total of the stock price up to the j-th time step. If $T_j > nX$, we will surely exercise the option at the expiration date, and the payoff is at least $T_j/n - X$. We call that the option is *in-the-money* if this happens. Thus, the European-Asian option is more reliable than the European option for buyers.

The **American-Asian option** permits a buyer to exercise the option in any time period. A buyer receives $A_i - X$ if the option is exercised at the i-th time period, where $A_i = T_i/i$. Apparently, the option is much advantageous for buyers, and hence its premium should be more expensive. One difficulty of this option is that the action of a buyer is highly path-dependent. Even after the status of the option becomes in-the-money, a buyer must decide whether he/she exercises the option immediately; it should depend on both T_i and the current stock price. Thus, its pricing with provable accuracy seems quite difficult (see Section 4).

We propose a new option, named **Saving-Asian option**[1]. In the Saving-Asian option, a buyer can exercise the option at any time period, and receive $e^{-(n-i)r_0/n}(T_i - iX)/n$ if the option is exercised at the i-th time period, where e^{r_0} is the risk-free interest rate for the whole period. Thus, it is an American type option, but different from a standard American-Asian option since it restricts the payoff for early exercise.

For a buyer, this option is clearly advantageous to the European Asian option, since he/she has a choice to keep the option until the expiration date in which case the payoff is $(A_n - X)^+$ that is exactly same as that of European Asian option. On the other hand, if a buyer exercises at the i-th period and re-invest the money, he/she will have $(T_i - iX)/n$ at the n-th step, which might be larger than $A_n - X = (T_n - nX)/n$. Therefore, if the stock price will drastically go down after enjoying some high-price period, a buyer can exercise early to avoid reduction of his/her profit. Moreover, early exercise has an advantage that a buyer can get money for urgent need.

Intuitively, this option simulates accumulative investment permitting discontinuation, in which a buyer has right to buy $1/n$ unit of the stock by X/n dollars for selling it by the market price every time period, and can stop at the i-th step after investing iX/n 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 option given in [5].

Similarly to the American-Asian option, the action of a buyer seems to be path-dependent. However, it is easier to analyze the best action assuming that a buyer has the same model of the stock price movement as the seller. In particular, in the uniform model (defined in the next subsection), once the status of the option becomes in-the-money, a buyer should sell the option in the i-th step if the expectation of the running total after the $(i+1)$-st step is less than $(n-i)X$; thus, the decision depends on the current stock price and the model, but is independent of the history of the movement of the stock. We remark that in our convention in this paper, in-the-money always means $T_i > nX$, although it is common that in-the-money means a buyer can get profit if he/she exercises immediately. We note that a buyer may exercise the option before it becomes in-the-money, and for that case the decision also depends on the current running total; however, this kind of path-dependency can be treated efficiently. We remark that the payoff function can be customized and our analysis still applies (Section 5).

2.2 Binomial Model

Let us consider a discrete random process simulating the movement of the price of a stock. The fundamental assumption in the binomial model (and the Black-Scholes model) is that in each time step the stock price S either rises to uS or falls to dS, where $u > d$ are predetermined constants.

[1] This option might be proposed before, although the authors do not know.

Thus, we can model stock price movement by a recombinant binary tree. A *recombinant binary tree*[2] G is a leveled directed acyclic graph whose vertices have at most two parents and two sons. We label the nodes (i, j) where i denotes the level and j denotes the numbering of the nodes in the i-th level $(0 \leq j \leq i)$. The node (i, j) has two sons $(i+1, j)$ and $(i+1, j+1)$ if $i \leq n-1$. Therefore, the node (i, j) has parents $(i-1, j)$ and $(i-1, j-1)$ if $i \neq 0$ and $1 \leq j \leq i-1$. Each of the nodes (i, i) and $(i, 0)$ has one parent. Intuitively, the graph looks like the structure of Pascal's triangle.

In the model, if we are at a node $v = (i, j)$ and the current stock price is S, we move to $(i+1, j)$ with probability p_v and the stock price rises to uS. With probability $1 - p_v$, we move to $(i+1, j+1)$ and the stock price falls to dS. Thus, if we are at the node (i, j), the stock price must be $u^{i-j} d^j S_0$.

The model is called *uniform* if $p_v = p$ for every node v; otherwise it is non-uniform. The uniform model is widely considered [1, 2, 5, 6] since p is uniquely determined under the non-arbitrage condition of the underlying financial object; however, non-uniform model is often useful to customize an option. We consider the uniform model first, and will show later how to deal with the non-uniform model. Our method also works for the trinomial model where each node (except those in the n-th level) has three sons, and stock price moves to one of uS, S, and $u^{-1}S$, although we omit details in this paper. In the uniform model, the probability that the random walk reaches to (i, j) is $\binom{i}{j} p^{i-j}(1-p)^j$. We define $r = up + d(1-p) - 1$, which corresponds to the risk-neutral interest rate for one time period in the risk-neutral model.

Our task is to compute the expected value $E((A_n - X)^+)$ of the payoff. A simple method is to compute the running total $T_n(\mathbf{p})$ of the stock value for each path $\mathbf{p}$ in the graph G together with the probability $prob(\mathbf{p})$ that the path occurs, and exactly compute $E((A_n - X)^+) = \sum_{\mathbf{p}} (prob(\mathbf{p})(T_n(\mathbf{p})/n - X)^+)$. The expected value U of the payoff computed as above is called the *exact value* of the expected pay-off.

However, this needs exponential time complexity with respect to n, since there are 2^n different paths. Random sampling of paths is a popular method to reduce the computation time, although we need to have huge number of paths in order to have a small provable error bound if we naively sample paths.

3 Our Algorithm for Pricing European Asian Option

3.1 AMO Algorithm

We give a brief overview of AMO algorithm (see [1] for details). AMO algorithm is based on dynamic programming and has an $O(n^2 k)$ time complexity with a provable error bound of nX/k, where k is a parameter to give the time-error tradeoff.

For a path $\mathbf{p}$ from the root to a node of level t, its *stamp* is the pair of its current stock price and the running total. Note that the current stock price

[2] Often called binomial lattice

corresponds to the node. The basic idea of AMO algorithm is to approximate the running totals appropriately so that the number of different stamps is at most $(t+1)k$, and to store the approximate stamps at the t-th time step into a table with $(t+1)$ rows and k columns. Moreover, if the running total T_t exceeds nX for a path $\mathbf{p}$ (i.e., the option is in-the-money), the expectation of the payoff of paths containing $\mathbf{p}$ as a prefix is analytically computed (see Section 3.3), and the stamp corresponding to the path $\mathbf{p}$ is pruned away from the table.

The row index corresponds to the stock prices. The stock price takes one of $(t+1)$ values $u^i d^{t-i} S_0$ for $i = 0, 1, .., t$ in the binomial model, and hence naturally we assign paths with the stock price $S_t(i) = u^i d^{t-i} S_0$ to the $(i+1)$-st row. The column index corresponds to the running total T_t of paths. Running totals of unpruned paths are assorted into k buckets $B(s)$ for $s = 1, 2, \ldots, k$, where $B(s)$ represents the interval $[b_{s-1}, b_s) = [(s-1)nX/k, snX/k)$.

A cell in the table is indicated by a pair of a stock value and a bucket. Suppose that many stamps are assorted into the cell $C(S_t(i), B(s))$. Then, the algorithm approximates them as a stamp with the current stock price $S_t(i)$ and running total b_{s-1}. The stamp has a weight $w_t(s, i)$, where $w_t(s, i)$ is the summation of the probability that each of the paths occurs.

Since the error caused in one step of the process is bounded by nX/k for $(T_n - nX)^+$, the error contribution to $(A_n - X)^+$ is at most X/k for one step. Thus, the accumulated errors in the final running total will be n^2X/k, and the error in the estimation of the average stock value is bounded by nX/k. More precisely, if $U = E((A_n - X)^+)$ is the exact value of the expected payoff in the binomial model and Φ is the payoff computed by the algorithm, we have $U \geq \Phi \geq U - nX/k$.

3.2 Modified Algorithm

Our modification of AMO algorithm is quite simple. In order to represent the stamps in a cell $C(S_t(i), B(s))$, we apply random sampling so that a stamp with weight w is selected with a probability $w/w_t(s, i)$, and give a weight $w_t(s, i)$ to it. Let Ψ be the payoff value computed by the algorithm. At a glance, it looks merely like a heuristic, and does not improve the theoretical bound.

Indeed, in the worst case, the error caused in one step is only bounded by X/k, and hence we can only prove that the worst case error bound $|U - \Psi|$ is nX/k. However, the algorithm can be also viewed as a sampling method of paths, since stamps stored in the table are real stamps of some suitable paths. We can observe that the selection of paths is smartly done during the runtime of the algorithm: In each step the path-prefixes are clustered by using the table, and a path-prefix is selected from each cluster to continue.

Since the algorithm is randomized, Ψ is a random variable depending on the coin-flips to choose representatives of stamps in the table. Let Y_i be a random variable giving the "exact" payoff value after running the algorithm up to the i-th time step; in other words, after the choice of representatives in all the cells of the table has been determined up to the i-th time step, we consider all the possible suffixes of the representing paths to compute Y_i exactly in the binomial model. Of course, the computation time is exponential, and thus Y_i is only used in the analysis of the performance of our modified AMO algorithm. By definition, $Y_0 = U$ and $Y_n = \Psi$.

Lemma 1. *$E(Y_i|Y_0, Y_1, Y_2, .., Y_{i-1}) = Y_{i-1}$ for $i = 1, 2, \ldots, n$. In particular, $E(Y_n) = U$.*

Proof. Consider the set $\{a_1, a_2, \ldots, a_q\}$ of stamps in a cell at the i-th time step before selecting its representative. Suppose $Y(a_j)$ is the estimated payoff (exactly computed from the model) for a path with the stamp a_j, and $w(a_j)$ is the weight of a_j. Let $W = \sum_{j=1}^{q} w(a_j)$. If a stamp a_j is selected, it contributes $Y(a_j)W$ to the payoff. Thus, the expectation of the contribution of the stamps after the selection is $\sum_{j=1}^{q}(w(a_j)/W)Y(a_j)W = \sum_{j=1}^{q} w(a_j)Y(a_j)$, which equals the expected contribution before the selection. □

Lemma 1 shows that the sequence $Y_0, Y_1, \ldots, Y_n$ is a *Martingale sequence*. From the argument in the previous section, $|Y_i - Y_{i-1}| < X/k$. Thus, we apply famous Azuma's inequality [4] (also see [3, 7].)

Theorem 1 (Azuma's inequality). *Let $Z_0, Z_1, \ldots$ be a Martingale sequence such that for each i, $|Z_i - Z_{i-1}| < c_i$, then for all $t \geq 0$ and any $C \geq 0$,*

$$Pr[|Z_t - Z_0| \geq C(\sum_{i=1}^{t} c_i^2)^{1/2}] \leq 2exp(-C^2/2).$$

In our case, $c_i = X/k$, and hence, we have the following:

Theorem 2. *$|\Psi - U| < C\sqrt{n}X/k$ with probability $1 - 2e^{-C^2/2}$.*

3.3 Aggregation When the Status Is In-The-Money

It is shown in [1] that in the uniform model, if the status is in-the-money at the t-th time step and the current stock price and running total are S and T, the expectation of payoff is $\{T + \sum_{j=1}^{n-t-1}(1+r)^j S\} - X$. Recall that $r = pu + (1-p)d - 1$. Indeed, $\sum_{j=1}^{n-t-1}(1+r)^j S$ is the expectation of extra running total after the $(t+1)$-st step.

In the non-uniform model, we can compute the value of extra running total for an in-the-money case by using a bottom-up computation as follows: We compute a real value $h(v)$ representing the extra running total for each node of the recombinant binary tree defined as follows: $h(v) = 0$ if v is a leaf (i.e., a node in level n), and $h(v) = p_v(h(w) + S(w)) + (1 - p_v)(h(w') + S(w'))$ if w and w' are sons of v and $S(w)$ and $S(w')$ are the stock values associated with the nodes w and w', respectively. The expectation of payoff is $T + h(v) - X$ if we are at v, the current running total is T, and the option is in-the-money. The computation time for $h(v)$ for all v is $O(n^2)$, and hence the total asymptotic time complexity does not increase. Therefore, we have the following:

Theorem 3. *Our algorithm approximates the expectation of pay-off in $O(n^2k)$ time, and its error from the exact expectation is at most $c\sqrt{n}X/k$ with probability $1 - 2e^{-c^2/2}$ for any positive value c.*

3.4 Experimental Performance

Figure 1 gives the comparison of three methods: 1. random sampling, 2. original AMO algorithm, and 3. our algorithm. Here, we consider a uniform model where $S_0 = X = 100$, $u = 1.1$, $d = 1/u$, $pu + (1-p)d = (1.06)^{1/n}$, and we set $k = 1000$. The premium is computed by multiplying $(1.06)^{-1}$ to the expected payoff value. In the random sampling method, we take $20n$ sample paths for one trial and take average over 1000 trials[3]. For our randomized algorithm, we only run single trial.

The graphs show the error from the exact premium computed by using the full-path method. The running time is approximately the same for the three methods in the range $10 \leq n \leq 30$, and about 0.08 second for $n = 30$, whereas the full-path method takes 1092 seconds. The error of our algorithm is always less than $0.03 = 0.3X/1000$, and smaller than the other methods with factors up to about 100. Also the error tends to decrease if n is increased, and its average is about 0.005 when $25 \leq n \leq 30$; therefore, it is much better than the theoretical bound $c\sqrt{n}X/1000$. At $n = 30$, the exact premium value is 11.5474 and the error ratio to the premium value is less than 0.0005.

We also run the random sampling algorithm spending 100 times of running time, but the accuracy is still not competitive to our algorithm. Note that we do not implement AMO algorithm with flexible bucket size (a heuristic that is reported to be better than the original AMO algorithm [1]), since its performance depends on tuning of parameters and also the heuristic can be combined with our algorithm, too.

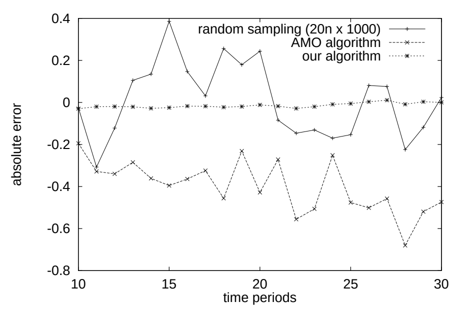

Fig. 1. Errors of the computed premium by three algorithms from the exact value

Figure 2 gives the premium prices computed by the algorithms for the case $50 \leq n \leq 80$, where we use a version of AMO algorithm in which we take the upper value in each bucket in order to give an upper bound of the premium price. The full-path method is not feasible for such a large n. It can be observed that the premium price computed by our algorithm is quite stable.

[3] Taking $20000n$ samples at once is oversampling for a small n.

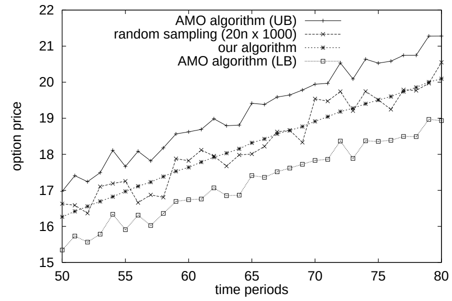

Fig. 2. Premiums computed by the algorithms

3.5 More Precise Analysis for the Uniform Case

The experimental result shows that the analysis in the previous sections overestimates the error. In this subsection, we prove the following theorem:

Theorem 4. *For a uniform model satisfying that* $1-\alpha < p < \alpha$ *for a constant* $\alpha < 1$*, our algorithm approximates the expected pay-off in* $O(n^2k)$ *time, and its error from the exact expectation is* $O(n^{1/4}X/k)$ *with probability* $1-2e^{-c^2}$*, where* c *is any given positive constant.*

We refine the analysis of the random process in the following way: nodes of the t-th level are processed one-by-one, and coin flips for the cells associated with a node is grouped into one random process; in other words, processing of each row of the table corresponds to one step of the random process.

The total weight $w(t,j)$ of the paths corresponding to the node (t,j) is $\binom{t}{j}p^j(1-p)^{t-j}$. Let $Y_{t,j}$ be the random variable giving the "exact" payoff value just after the algorithm processes the j-th node in the t-th level. Thus, we have a random process with $\sum_{t=0}^{n-1}(t+1) = n(n+1)/2$ total steps. We can easily see that this gives a Martingale process, and $|Y_{t,j} - Y_{t,j+1}| < w(t,j)X/k$ for $j \neq t$, and $|Y_{t,t} - Y_{t+1,0}| < w(t,t)X/k$.

Let $c_{t,j} = w(t,j)X/k$. Then, in order to apply Azuma's inequality, we estimate $\Gamma(p) = \sum_{0\leq j\leq t\leq n-1} c_{t,j}^2 = (X/k)^2 \sum_{t=0}^{n-1} g(t,p)$, where $g(t,p) = \sum_{j=1}^{t}(\binom{t}{j}p^j(1-p)^{t-j})^2$. Because of lack of space, we only give the estimation for $p = 1/2$ in this conference version of the paper, although it is not difficult to generalize it.

By definition, $g(t,1/2) = 2^{-t}\sum_{j=1}^{t}\binom{t}{j}^2$. It is easy to see $\sum_{j=1}^{t}\binom{t}{j}^2 = \sum_{j=1}^{t}\binom{t}{j}\binom{t}{t-j} = \binom{2t}{t}$. Since $\binom{2t}{t} \sim 2^{2t}/\sqrt{\pi t}$, we have $g(t,1/2) \sim 1/sqrt\pi t$. Thus, we have $\Gamma(p) = O((X/k)^2\sqrt{n})$, and we can obtain the theorem easily by applying Azuma's inequality.

4 Computing Payoff of Saving Asian Option

In [1], it is claimed that a variant of AMO algorithm also works for the American-Asian option. However, it is based on a claim (or assumption) that the early exercise always occurs before the status becomes in-the-money. This is not always true, since a buyer of the American-Asian option should hold the option while the stock price continues to go up. Thus, it seems to be difficult to apply AMO algorithm for the American Asian option.

On the other hand, we can modify AMO algorithm to work on the Saving-Asian option. Recall that if a buyer exercises at the i-th time step and re-invest the money, he/she will receive $(T_i - iX)/n$ at the n-th time step. Suppose that the status is in-the-money at the i-th step, and thus the advantage that a user need not exercise the option is no more valid. In the uniform model, the decision merely depends on whether $T_i - iX$ is larger than the expected value of $T_n - nX$ (knowing the current stock price S) or not; In other words, we should exercise early if and only if $E(T_n - T_i | S_i = S) = S \sum_{j=1}^{n-j} (1+r)^j < (n-i)X$, which means that the expectation of the average stock price after the i-th step is less than X. This condition is path-independent except the path-dependent assumption that the status is in-the-money.

In the non-uniform model, the situation is a little more complicated, since even if $T_i - iX$ is larger than the conditional expectation of $T_n - nX$, it may happen that we should wait for a while. For example, we may postpone to exercise during the X'mas week if a particular stock (e.g. a stock of a department store) is expected to go up during the week in the model provided that the current stock price S is in a certain range.

For each node v in the recombinant binary tree, we define real values $f(v)$ and $g(v)$ as follows: If v is a leaf, $f(v) = 0$ and $g(v) = S(v) - X$, where $S(v)$ is the stock price associated with v. If v has sons w and w' such that w is selected with probability p_v, $f(v) = \max\{p_v g(w) + (1-p_v) g(w'), 0\}$ and $g(v) = S(v) - X + f(v)$.

The value $p_v g(w) + (1-p_v) g(w')$ is the expectation of extra (possibly negative) payoff obtained by postponing the exercise of the option at v, and thus $f(v)$ is the value of the right of postponing the exercise. Indeed, if we postpone and the process goes to w, $S(w) - X$ is added to the current payoff (including the interest) and $f(w)$ gives the value of the right of postponing at w.

We call a node v in the i-th level of the recombinant binary tree a *pseudo-exercise node* if $f(v) = 0$. The values $f(v)$ and $g(v)$ can be computed in a bottom-up fashion in $O(n^2)$ time for all nodes v. The following is the key lemma:

Lemma 2. *If the status is in-the-money, one should exercise the option at the first pseudo-exercise node that is encountered.*

Now, we can run a dynamic programming algorithm that is basically the same as the algorithm in the previous section for the European Asian option (we omit details because of space limitation).

Theorem 5. *Our algorithm approximates the expected pay-off of the Saving-Asian option in $O(n^2 k)$ time, and its error from the exact expectation is at most $c\sqrt{n}X/k$ with probability $1 - 2e^{-c^2/2}$. Moreover, in the uniform model with $1 - \alpha \leq p \leq \alpha$ for a constant $\alpha < 1$, the error bound becomes $O(n^{1/4} X/k)$.*

5 Concluding Remarks

Our algorithm works for a path-dependent option with the following three conditions: 1). The payoff of early exercise at a node v in the t-th level after the path $\mathbf{p}$ of movement on the binomial model is written as $(\gamma(t)F(\mathbf{p}) - G(v))^+$, where $0 < \gamma(t) \leq 1$ is a functions on t, $G(v)$ is a nonnegative function, and $F(\mathbf{p})$ is a summation $\sum_{i=0}^{t} f_i(S_i)$ of the values of nondecreasing functions f_i on the path $\mathbf{p} = S_0, S_1, \ldots, S_t$ of stock movement. 2). There is a threshold value L such that once $F(\mathbf{p}) > L$, the payoff will be destined to be positive (in-the-money). 3). Once $F(\mathbf{p}) > L$, the difference between the payoff (including interest) of immediate exercise and the expected payoff obtained by delaying the exercise is path-independent.

For our Saving-Asian option, $\gamma(t) = e^{-(n-t)r_0/n}$, $f_i(S_i) = S_i/n$, $L = X$, and $G(v) = t(v)X/n$, where $t(v)$ is the level of v. Thus, we may add a path-independent function to the payoff of our Saving-Asian option without losing the accuracy. For example, we may design a variation in which we pay-back a portion of premium for an early exercise satisfying some path-independent conditions.

The experimental performance implies that our theoretical analysis is not tight: Indeed, we may further refine the random process so that each coin flip at a cell of the DP table is 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 to be complicated and difficult.

It will have industrial value to investigate the applicability of AMO method (or its extension) to several existing options, and design new useful options based on the insight.

Acknowledgement
The authors thank Prof. Syoichi Ninomiya of Center of Res. in Adv. Financial Tech., Tokyo Institute of Technology for his counsel on financial engineering.

References

1. D. Aingworth, R. Motwani, and J.D. Oldham, Accurate Approximations for Asian Options, *Proc. 11th ACM-SIAM SODA* (2000), pp. 891-901.
2. K. Akcoglu, M.-Y. Kao, and S. V. Raghavan, Fast Pricing of European Asian Options with Provable Accuracy: Single-Stock and Basket Option, *Proc. ESA 2001, Springer LNCS 2162* (2001), pp. 404-415.
3. N. Alon, J. Spencer, P. Erdős, *The Probabilistic Method*, Jhon Wiley & Sons, 1992.
4. K. Azuma, Weighted Sum of Certain Dependent Random Variables, *Tohoku Mathematical Journal* **19** (1967) pp.357-367.
5. P. Chalasani, S. Jha, and I. Saias, Approximate Option Pricing, *Algoirthmica* **25** (1999), pp.2-21.
6. P. Chalasani, S. Jha, and A. Varikooty, Accurate Approximation for Europian Asian Options, *Journal of Computational Finance* **1** (1999), pp.11-29.
7. R. Motwani and P. Raghavan, *Randomized Algorithms*, Cambridge University Press, 1995.