Very Fast Algorithms for Barrier Option Pricing and the Ballot Problem

Yuh-Dauh Lyuu
Professor
Dept. Computer Science and Information Engineering
National Taiwan University
Taipei, Taiwan
Tel: 886-2-362-5336 ext 429
Fax: 886-2-362-8167
lyuu@csie.ntu.edu.tw
http://www.csie.ntu.edu.tw/~lyuu

May 26, 2000

Abstract

Combinatorial methods prove extremely useful towards designing blazingly fast yet simple algorithms for pricing European-style barrier options. Closed-form formulae to standard European-style barrier options can then be easily derived. Combinatorial formulae under the trinomial model are also presented. The common practice in the literature compares algorithms based on their respective numbers of time steps towards convergence. We illustrate the pitfalls of this custom by evaluating the performance of our binomial model-based algorithm and the trinomial tree algorithm, whose superiority over the binomial model is widely accepted. Contrary to common beliefs, however, our algorithm emerges as a clear winner. In fact, the performance gap is two orders of magnitude. Also shattered is the myth that the binomial model converges extremely slowly when the current stock price is very close to the barrier.

1 Introduction

In Feller's [1968] masterpiece, *An Introduction to Probability Theory and Its Applications*, Vol. 1, and, subsequently, Takács's [1967] *Combinatorial Methods in the Theory of Stochastic Processes*, combinatorial methods are found to be useful in the study of stochastic processes. This paper follows their lead in applying these methods to derivative pricing, specifically, European-style barrier option pricing.

The particular branch of combinatorics relevant to our purpose is the solution to Bertrand's ballot problem (Lint and Wilson [1994]). The original problem is concerned with the number of ways a candidate can be ahead of his opponent throughout the vote counting process given the final vote counts. The problem was solved by, among others, André's reflection principle. See Takács [1962] for survey and history. We shall see that pricing barrier options is intimately related to the ballot problem.

Options whose payoff depends on whether the underlying asset's price reaches a certain level are called barrier options (Hull [1997]). Such options are clearly path-dependent. A knock-out option is like an ordinary European option except that it ceases to exist if a certain barrier, H, is reached by the price of its underlying asset. A call knock-out option is sometimes called a down-and-out option if $H < X$, where X denotes the strike price. Similarly, a put knock-out option is sometimes called an up-and-out option when $H > X$. A knock-in option, in contrast, comes into existence if a certain barrier is reached. A down-and-in option is a call knock-in option that comes into existence only when the barrier, $H < X$, is reached. An up-and-in option is a put knock-in option that comes into existence only when the barrier, $H > X$, is reached.

The value of a European down-and-in call is

$$S\, e^{-q\tau} (H/S)^{2\lambda} N(x) - X\, e^{-r\tau} (H/S)^{2\lambda-2} N\left(x - \sigma\sqrt{\tau}\right) \tag{1}$$

where

$$\begin{aligned} x &= \frac{\ln\left(H^2/(SX)\right) + \left(r - q + \sigma^2/2\right)\tau}{\sigma\sqrt{\tau}} \\ \lambda &= \frac{r - q + \sigma^2/2}{\sigma^2} \end{aligned}$$

where $S \geq H$, q is the stock's dividend yield, and τ is the time to maturity. The above formula assumes that the underlying asset price follows geometric Brownian motion and is due to Merton [1994]. A European down-and-out call can be priced via the in-out parity. The value of a European up-and-in put is

$$X\, e^{-r\tau} (H/S)^{2\lambda-2} N\left(-x + \sigma\sqrt{\tau}\right) - S\, e^{-q\tau} (H/S)^{2\lambda} N(-x)$$

for $S \leq H$. A European up-and-out call can be priced via the in-out parity.

Although closed-form solutions exist, the study of numerical methods based on binomial/trinomial models is still useful for the new insights it brings. It also have applications to exotic options where the terminal payoff function is non-standard and closed-form solutions are hard to come by. We will illustrate this point with power options. Finally, in cases where continuous trading is not an appropriate model, the discrete-time model might offer more realistic prices (Levy and Mantion [1997]).

In this paper, we derive a combinatorial formula for the price of down-and-in calls under the binomial model. The technique, as we mentioned before, is based on the solution to the ballot problem. This formula is shown in Lyuu [1997] to converge to the closed-form solution via elementary means instead of the more advanced Fourier transform method. More interestingly, the formula leads directly to a highly efficient algorithm that runs in time proportional to the number of periods the time is partitioned into, denoted throughout the paper by n. In addition, the memory requirement is merely a few variables. This is in sharp contrast to the naive binomial tree algorithm that has a time complexity proportional to n^2 and memory requirement proportional to at least n. Linear-time performance is the key that leads to the re-thinking on the binomial model. Computer experiments show that the time for the algorithm to converge to the analytical value is a mere tens or hundreds of milliseconds on a typical personal computer. Such speed advantage will surely be important for large trading desks with thousands of options to be priced on an hourly basis.

It is a common belief that the binomial model is next to impossible to converge when the current stock price is very near the barrier. The work reported here casts doubt on that belief as our algorithm proves to be robust even under this supposedly tough case. Although the running time does climb, it remains in the vicinity of hundreds of milliseconds.

With few exceptions such as Broadie and Detemple [1996], the usual practice in the literature compares two algorithms based on the n at which they converge. This practice can be grossly misleading. The only objective method of comparing algorithms is their total running times to achieve comparable results. This position has been argued forcefully in Patterson and Hennessy [1994]. We illustrate their point by comparing our binomial model-based algorithm and the trinomial tree algorithm, which is widely accepted to be superior. The evidence overwhelmingly favors the former as it is faster by two orders of magnitude to achieve the same analytical result even if it requires a higher n. This conclusion, independent of the superiority of trinomial algorithms in such aspects as generality, should serve as a caution to efforts in algorithm evaluation.

Finally, we show the wide applicability of the combinatorial method by deriving a combinatorial formula for the price of down-and-in calls under the trinomial model.

2 The Binomial Model

We quickly review the binomial approximation to the geometric Brownian motion, $dS/S = \mu\, dt + \sigma\, dW$. Consider the stock price $\Delta t \equiv \tau/n$ time from now (time zero). From the geometric binomial random walk model, in a period of Δt, the stock price either increases to $S\,u$ with probability p or decreases to $S\,d$ with probability $1-p$.

It is easy to verify that

$$E[\,S(\Delta t)\,] = S\,e^{\mu\,\Delta t} \;\text{ and }\; \mathrm{Var}[\,S(\Delta t)\,] = S^2\left(e^{\Delta t\,\sigma^2}-1\right)e^{2\,\Delta t\,\mu} \to S^2\sigma^2\Delta t.$$

Matching the above two moments and imposing $ud = 1$ leads to

$$u = e^{\sigma\sqrt{\Delta t}}, \quad d = e^{-\sigma\sqrt{\Delta t}}, \quad \text{and} \quad p = \frac{e^{\mu\,\Delta t}-d}{u-d}. \tag{2}$$

Note that $\mu = r$ in a risk-neutral economy.

To derive the combinatorial formula under the binomial model for the down-and-in call, we must count the number of ways the stock price can reach any given terminal price while hitting the barrier on the way. The reflection principle provides the needed tool.

3 The Reflection Principle

Imagine a particle starts at position $(0, -a)$ on the integral lattice and wishes to reach $(n, -b)$. Without loss of generality, assume $a, b \geq 0$. The particle is constrained to move to $(i+1, j+1)$ or $(i+1, j-1)$ from (i, j), the very fashion the price under the binomial model is supposed to evolve in,

$$\begin{array}{lll} & & (i+1, j+1) \quad \text{associated with the up move, } S \to S\,u \\ & \nearrow & \\ (i, j) & & \\ & \searrow & \\ & & (i+1, j-1) \quad \text{associated with the down move, } S \to S\,d \end{array}$$

How many such paths the particle can take that touch the x-axis? This question can be rephrased as the following variant of the ballot problem. Given that a candidate starts with a fewer votes than his opponent (which is not uncommon in many parts of the world) and ends up with b fewer votes, how many ways can the votes be counted in which his vote count equals his opponent's at least once?

Consider any legitimate path from $(0, -a)$ to $(n, -b)$ that touches the x-axis. Let J denote the first position this happens. By reflecting the portion of the path from $(0, -a)$ to J, a path from $(0, a)$ to $(n, -b)$ is thus constructed. Note that this new path hits the x-axis at J. See Figure 1 for illustration. This one-to-one mapping shows the number of paths from $(0, -a)$ to $(n, -b)$ that touch the x-axis is exactly the number of paths from $(0, a)$ to $(n, -b)$. This is the celebrated reflection principle of André's published in 1887 (Lint and Wilson [1994]). Since any such path consisting of n moves must have $b + a$ more down moves ("-1"s) than up moves ("$+1$"), the desired number equals the number of ways to permute $(n - a - b)/2$ "$+1$"s and $(n + a + b)/2$ "-1"s, which is equal to

$$\binom{n}{\frac{n+a+b}{2}} \quad \text{for even } n+a+b \tag{3}$$

with the convention that $\binom{n}{k} = 0$ for $k < 0$ or $k > n$.

4 Combinatorial Formulae for Barrier Options

Consider the down-and-in call option with barrier $H < X$ as a concrete example. Assume $H < S$ without loss of generality for, otherwise, the option is identical to the standard call. Let

$$\begin{aligned} a &\equiv \left\lceil \frac{\ln\left(X/\left(Sd^n\right)\right)}{\ln(u/d)} \right\rceil = \left\lceil \frac{\ln(X/S)}{2\sigma\sqrt{\Delta t}} + \frac{n}{2} \right\rceil \\ h &\equiv \left\lfloor \frac{\ln\left(H/\left(Sd^n\right)\right)}{\ln(u/d)} \right\rfloor = \left\lfloor \frac{\ln(H/S)}{2\sigma\sqrt{\Delta t}} + \frac{n}{2} \right\rfloor \end{aligned}$$

It is easy to see that $\tilde{H} \equiv Su^h d^{n-h}$ is the price among $Su^j d^{n-j}$ $(0 \le j \le n)$ closest to, but not exceeding, H. The role of the barrier will be played by the effective barrier, $\tilde{H}$, for the binomial model. Similarly, $\tilde{X} \equiv Su^a d^{n-a}$ is the price among $Su^j d^{n-j}$ $(0 \le j \le n)$ closest to, but not exceeded by, X. A process with n moves ends up at a price at or above X if and only if the number of up moves is at least a.

Any price of the form $Su^k d^{n-k}$ is at a distance of $2k$ from the lowest possible price, Sd^n, on the binomial tree. This holds because

$$Su^k d^{n-k} = Sd^{-k} d^{n-k} = Sd^{n-2k}. \tag{4}$$

Based on the above observation, Figure 2 plots the relative distances of various prices on the binomial tree.

The number of paths from S leading to a terminal price $Su^j d^{n-j}$ is $\binom{n}{j}$ each with the same probability $p^j(1-p)^{n-j}$. With reference to Figure 2, the reflection principle can be applied with $a = n - 2h$ and $b = 2j - 2h$ in (3) by treating the S-line as the x-axis and the $\tilde{H}$-line as the barrier. Therefore,

$$\binom{n}{\frac{n+(n-2h)+(2j-2h)}{2}} = \binom{n}{n-2h+j}$$

among these paths hit $\tilde{H}$ in the process for $h \le n/2$. We conclude that the terminal price $Su^j d^{n-j}$ is reached by a path that hits the effective barrier with probability $\binom{n}{n-2h+j} p^j(1-p)^{n-j}$. Since each terminal payoff should be weighted by its probability of occurrence in a risk-neutral world, the option value must equal

$$\widehat{R}^{-n} \sum_{j=a}^{2h} \binom{n}{n-2h+j} p^j (1-p)^{n-j} \left(Su^j d^{n-j} - X\right), \tag{5}$$

where $\widehat{R} \equiv e^{r\tau/n}$ is the riskless return per period. Lyuu [1997] has shown that the above formula converges to Merton's formula as $n \to \infty$. We emphasize that (5) is merely an alternative characterization of the binomial tree algorithm for European down-and-in calls.

4.1 Applications to other types of barrier options

Formulae for other types of barrier options can be similarly derived. Even exotic barrier options whose payoff is an arbitrary function of the terminal stock price and the strike price can be priced. For the down-and-in type of barrier options, just replace $Su^j d^{n-j} - X$ in the pricing formula with the payoff function. We illustrate this step with two examples. Recall that a binary call pays off Q if it finishes above the strike price and nothing otherwise (Hull [1997]). The price of a binary down-and-in call would be the same as the pricing equation but with $Su^j d^{n-j} - X$ replaced by Q. Take power options as another example. Their payoff functions are non-standard: $\max\left((S(\tau) - X)^p, 0\right)$ and $\max\left(S(\tau)^p - X, 0\right)$ (Zhang [1997]). To price such exotic options with a knock-in-barrier, just replace $Su^j d^{n-j} - X$ by $(Su^j d^{n-j} - X)^p$ and $(Su^j d^{n-j})^p - X$, respectively.

5 Algorithmic Description and Performance Evaluation

The implementation of the pricing formula is straightforward. Define

$$a_j \equiv \binom{n}{n-2h+j} p^j (1-p)^{n-j} u^j d^{n-j} \quad \text{and} \quad b_j \equiv \binom{n}{n-2h+j} p^j (1-p)^{n-j}.$$

The pricing formula (5) thus becomes $\widehat{R}^{-n} \sum_{j=a}^{2h} (Sa_j - Xb_j)$. Since

$$\begin{aligned} a_j &= a_{j+1} \times \frac{n-2h+j+1}{2h-j} \times \frac{1-p}{p} \times \frac{d}{u} \\ b_j &= b_{j+1} \times \frac{n-2h+j+1}{2h-j} \times \frac{1+p}{p} \end{aligned}$$

all the a_j's and b_j's can be computed in linear time. Consequently, the pricing formula can be computed in linear time as well. In practice, a_j's and b_j's need to be stored in their logarithms to preserve precision.

The running time is actually proportional to $2h - a$, which may be substantially less than n. We can be more precise. Since

$$2h - a \approx \frac{n}{2} + \frac{\ln\left(H^2/(SX)\right)}{2\sigma\sqrt{\tau/n}} = \frac{n}{2} + O\left(\sqrt{n}\right), \tag{6}$$

the total running time should be proportional to n for sufficiently large n, independent of the other parameters (of course, how large n needs to be does depend on these parameters). This observation will prove useful later in accurately predicting the algorithm's performance without actually running it. Furthermore, the memory requirement is minimal; a few number of variables instead of arrays suffices.

5.1 Handling the sawtooth-like convergence

As with the binomial tree algorithm for standard European options, formula (5) leads to sawtooth-like convergence (Figure 3). Worse, as Boyle and Lau [1994] pointed out, unlike the binomial algorithm for standard European options, the swings now are much larger, and the convergence rate is slow.

A solution quickly suggests itself once we understand the reasons for the slowness. The true barrier most likely does not equal the effective barrier. The same holds for strike price and effective strike price. Both introduce specification errors (Derman, *et al.* [1995]). The problem with the strike price is less significant, as is testified by the fast convergence of binomial tree algorithms for standard European options; its influence is limited to the terminal price. The problem with the barrier is not negligible, because the barrier exerts its influence throughout the price process. Hence, this part of the specification error is more pronounced.

Figure 4 shows the details of Figure 3. It suggests that convergence is actually good if we limit n to certain values—191 in the figure, for example. These values correspond to the cases where the true barrier coincides with, or just above, one of the stock price levels so that

$$H \approx Sd^j = Se^{-j\sigma\sqrt{\tau/n}}$$

for some integer j, as pointed out by Boyle and Lau [1994]. The preferred n's are thus

$$n = \left\lfloor \frac{\tau}{(\ln(S/H)/(j\,\sigma))^2} \right\rfloor, \quad j = 1, 2, 3, \ldots$$

There is only one minor technicality left. We picked the effective barrier, $\tilde{H}$, to be one of the $n+1$ possible terminal stock prices. However, the effective barrier above, Sd^j, corresponds to a terminal stock prices only when $j = n - 2k$ for some k by (4). To close this gap, we decrement n by one, if necessary, to make $n - j$ an even number. The final list of admissible n's is

$$n = \ell - (|\,\ell - j\,| \bmod 2), \quad \text{where } \ell = \left\lfloor \frac{\tau}{(\ln(S/H)/(j\,\sigma))^2} \right\rfloor \text{ and } j = 1, 2, 3, \ldots \quad (7)$$

These observations yield the simple rule: Evaluate the pricing formula (5) only for the n's in (7). The result is shown in Figure 5.

5.2 Performance of the algorithm

The computation is blazingly fast with high precision. For the calculation depicted in Figure 5, for example, it takes about 0.0247 second for $n = 2138$ on a personal computer equipped with a 100 MHz Intel Pentium processor and 32 MB of DRAM running Windows NT 4.0. It is much faster than the quadratic-time trinomial tree algorithm to be introduced in Subsection 6.2. Specifically, the trinomial tree algorithm takes about 15.4 seconds on the same platform to approach the analytical value at a smaller n of 1731. Therefore, we can afford to pick very large n's for the calculation. Even at $n = 53450$, the running time does not exceed 0.7 second. Figure 6 tabulates the running times of these two approaches. Without any doubt, our binomial model-based algorithm is superior to the trinomial tree algorithm *qua* performance. The trinomial tree algorithm has the advantage of being generalizable to cases with time-varying barriers and early exercise features as in Cheuk and Vorst [1996] and Ritchken [1995].

From (6) and the data for $n > 10000$ in Figure 6, the performance of our algorithm can be predicted by the following formula

$$0.012826 \times n \quad \text{(milliseconds)}. \tag{8}$$

This predictor will prove accurate and useful in a moment.

5.3 The supposedly hard case: when the stock price meets the barrier

It has been widely claimed and accepted that the binomial model is "extremely difficult" to achieve convergence when the barrier is close to the current price of the underlying asset, $S \approx H$. Such a claim may be justified by (7), which says n, being proportional to $1/\ln^2(S/H)$, is huge when $S \approx H$. But this pessimism should be mitigated by an efficient algorithm implementation. This is vindicated by the data in Figure 7. The numbers there hardly signify an algorithm that ventures beyond its boundary of applicability. For instance, the maximum running time when convergence is achieved is 368 milliseconds.

5.4 Predicting the performance

The actual running times of our algorithms could have been estimated by (8). Figure 8 tabulates both the estimated and the actual running times based on the Windows/Intel platform. They are close enough to be useful as rough estimates. Similar conclusions hold for the Sun SPARCstation platform: Visually inspecting the parenthesized numbers in Figure 7 yields a coefficient about 0.01 (vs 0.012826 for the Windows/Intel platform).

6 Trinomial Tree Algorithms

An alternative to accelerating the computation uses trinomial tree algorithms such as the one due to Ritchken [1995]. Ritchken also shows that such algorithms can price barrier options with time-varying barriers or even multiple barriers.

6.1 Setting up the trinomial model

We first review Ritchken's trinomial approximation to geometric Brownian motion. The stock price Δt time from now will be

$$
\begin{array}{ccll}
 & & S\, e^{\lambda\sigma\sqrt{\Delta t}} & \text{with probability } p_u \\
 & \nearrow & & \\
S & \rightarrow & S & \text{with probability } p_m \\
 & \searrow & & \\
 & & S\, e^{-\lambda\sigma\sqrt{\Delta t}} & \text{with probability } p_d
\end{array}
$$

where $\lambda \geq 1$ and

$$
\begin{aligned}
p_u &\equiv \frac{1}{2\lambda^2} + \frac{\mu'\sqrt{\Delta t}}{2\lambda\sigma} \\
p_m &\equiv 1 - \frac{1}{\lambda^2} \\
p_d &\equiv \frac{1}{2\lambda^2} - \frac{\mu'\sqrt{\Delta t}}{2\lambda\sigma}
\end{aligned}
$$

with $\mu' \equiv \mu - \sigma^2/2$. See Figure 9. Here, λ is a parameter that can be tuned. Note that the trinomial model reduces to the binomial model when $\lambda = 1$.

6.2 Pricing barrier options

We mentioned that the binomial model introduces specification error by replacing the barrier with the effective barrier. The trinomial tree algorithm due to Ritchken solves the problem cleverly by adjusting λ so that the barrier is hit exactly. Here is the idea. Observe that it takes

$$n_h = \frac{\ln(S/H)}{\lambda\,\sigma\sqrt{\Delta t}}$$

consecutive down moves to go from S to H if n_h is an integer, that is. But this is easy to achieve by adjusting λ. Typically, we find the smallest $\lambda \geq 1$ such that n_h is an integer, that is,

$$\lambda = \min_{j=1,2,3,\ldots} \frac{\ln(S/H)}{j\,\sigma\sqrt{\Delta t}}.$$

This done, one of the layers of the trinomial tree must coincide with the barrier.

A quick look at Figures 3 and 10 gives the impression that trinomial model converges faster than binomial model. But this can be misleading. We cautioned before against comparing algorithms based on their convergence towards the analytical value with respect to the number of time steps (n in the current scenario). This metric ignores important details; for instance, the supposedly faster convergence may be overwhelmed by a huge time complexity. The true comparison must be based on the total running time (Patterson and Hennessy [1994]), by which our algorithm has an edge. Figure 6 demonstrates this point clearly.

6.3 Combinatorial formulae

Consider the down-and-in call option with barrier $H < X$. Assume without loss of generality that $H < S$. Under the trinomial model, there are $2n + 1$ stock prices, Su^j for $-n \leq j \leq n$, where $u = e^{\lambda\sigma\sqrt{\Delta t}}$. Let

$$a \equiv \left\lceil \frac{\ln(X/S)}{\lambda\,\sigma\sqrt{\Delta t}} \right\rceil \quad \text{and} \quad h \equiv \frac{\ln(S/H)}{\lambda\,\sigma\sqrt{\Delta t}} > 0.$$

A process with n moves ends up at a price at or above X if and only if the number of up moves exceeds that of down moves by at least a because $Su^a \geq X > Su^{a-1}$. Furthermore, the starting price is separated from the barrier by h down moves because $Su^{-h} = H$. See Figure 11. Note that the meanings of a and h are different from those in the binomial model.

The reflection principle applied to trinomial random walks, the following formula for down-and-in calls can be similarly derived as

$$\widehat{R}^{-n} \sum_{m=0}^{n-2h-a} \sum_{\substack{j \geq \max(a, m-n) \\ n-m+j \text{ is even}}}^{n-m-2h} \frac{n!}{((n-m+j+2h)/2)!\, m!\, ((n-m-j-2h)/2)!}$$
$$\times p_u^{(n-m+j)/2} \times p_m^m \times p_d^{(n-m-j)/2} \times \left(Su^j - X\right).$$

The above formula is just an alternative characterization of the trinomial tree algorithm for down-and-in calls. It implies a simple algorithm that runs in time proportional to $(n - 2h - a)^2$, which, though not linear (hence not competitive with the binomial model), is substantially less than n^2. The bounds on m and j can be easily verified. Formulae for the other three types of barrier options have been similarly derived in Lyuu [1997].

7 Conclusions

Combinatorial methods have found wide applicability in many fields. This paper extends their use to pricing European-style barrier options even with non-standard payoffs. Furthermore, the combinatorial formulae yield highly efficient algorithms in terms of both time and memory requirements. We expect combinatorial methods to be similarly applicable to more sophisticated derivatives such as barrier options with complex barriers. It has been shown in Lyuu [1997], for example, that lookback options and barrier options with double barriers can be tackled with identical techniques.

By comparing our binomial model-based algorithm against the supposedly superior trinomial tree algorithm on European-style barrier options, a picture contrary to the common belief surfaces: The former is a clear winner *qua* performance. This conclusion sheds doubt on the common methodology in the literature regarding algorithm evaluation. The total running time to achieve comparable numerical results, not any other proxies, remains the only objective metric.

Acknowledgments

The author thanks Professors Ren-Raw Chen and Douglas G. Rogers for discussions and Yu-Hong Liu for assistance. The detailed comments from an anonymous referee improved the manuscript immensely.

References

[1] Boyle, P., and S. H. Lau. "Bumping Up against the Barrier with the Binomial Method." *The Journal of Derivatives*, Summer 1994, 6–14.

[2] Broadie, M., and J. Detemple. "American Option Valuation: New Bounds, Approximations, and a Comparison of Existing Methods." *The Review of Financial Studies*, 9, No. 4 (Winter 1996), 1211–1250.

[3] Cheuk, T. H. F., and T. C. F. Vorst. "Complex Barrier Options." *The Journal of Derivatives*, Fall 1996, 8–22.

[4] Derman, E., I. Kani, D. Ergener, and I. Bardhan. "Enhanced Numerical Methods for Options with Barriers." *Financial Analysts Journal*, 51, No. 6 (November–December 1995), 65–74.

[5] Feller, W. *An Introduction to Probability Theory and Its Applications*, Vol. 1. 3rd ed. New York: John Wiley, 1968.

[6] Hull, J. C. *Options, Futures, and Other Derivative Securities.* 3rd ed. Englewood Cliffs, New Jersey: Prentice-Hall, 1997.

[7] Levy, E., and F. Mantion. "Discrete by Nature." *RISK*, 10, No. 1 (January 1997), 74–75.

[8] Lint, J. H. van, and R. M. Wilson. *A Course in Combinatorics.* Cambridge: Cambridge University Press, 1994.

[9] Lyuu, Y.-D. *Introduction to Financial Computation: Principles, Mathematics, Algorithms.* Manuscripts. February 1995–1997.

[10] Merton, R. C. *Continuous-Time Finance.* Revised ed. Cambridge, Massachusetts: Blackwell, 1994.

[11] Patterson, D. A., and J. H. Hennessy. *Computer Organization & Design: the Hardware/Software Interface.* San Mateo, California: Morgan Kaufmann, 1994.

[12] Ritchken, P. "On Pricing Barrier Options." *The Journal of Derivatives*, Winter 1995, 19–28.

[13] Takács, L. "Ballot Problems." *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, 1 (1962), 154–158.

[14] Takács, L. *Combinatorial Methods in the Theory of Stochastic Processes.* New York: John Wiley, 1967.

[15] Zhang, P. G. *Exotic Options: a Guide to Second Generation Options*. Singapore: World Scientific, 1997.

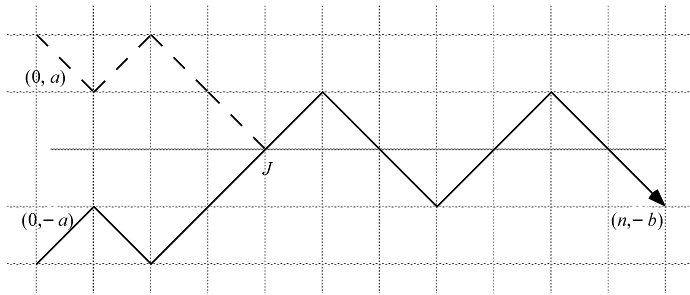

Figure 1: THE REFLECTION PRINCIPLE FOR BINOMIAL RANDOM WALKS.

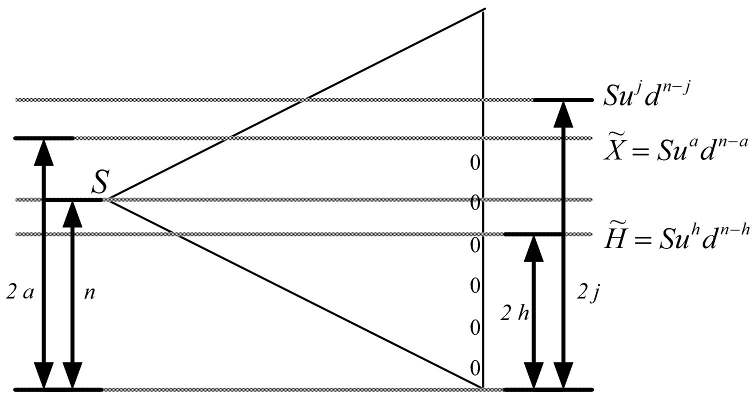

Figure 2: DOWN-AND-IN CALL AND BINOMIAL TREE. The effective barrier is the $\tilde{H}$-line, and the process starts at the S-line.

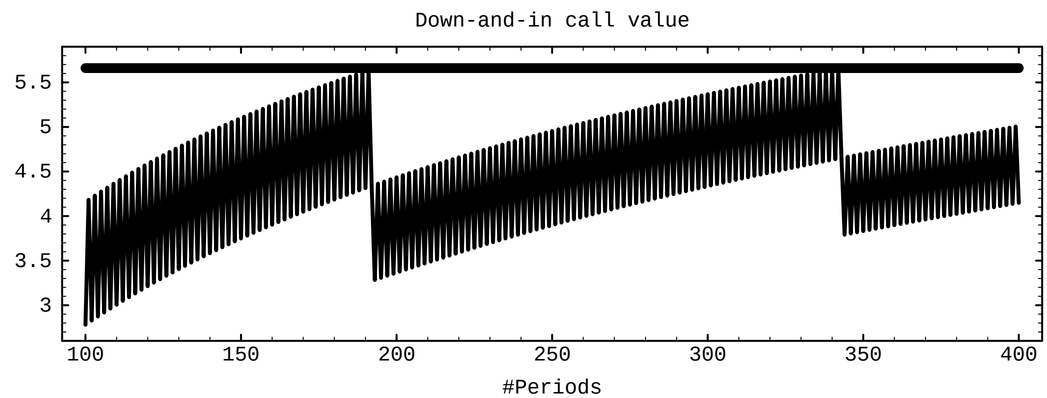

Figure 3: CONVERGENCE OF BINOMIAL MODEL FOR DOWN-AND-IN CALLS. Plotted are the option values as computed by (5) against the number of time periods, n. The option's parameters are $S = 95$, $X = 100$, $H = 90$, $r = 10\%$ (continuously compounded), $\sigma = 0.25$, and $\tau = 1$ (year). The analytical value, 5.6605, is also plotted for reference.

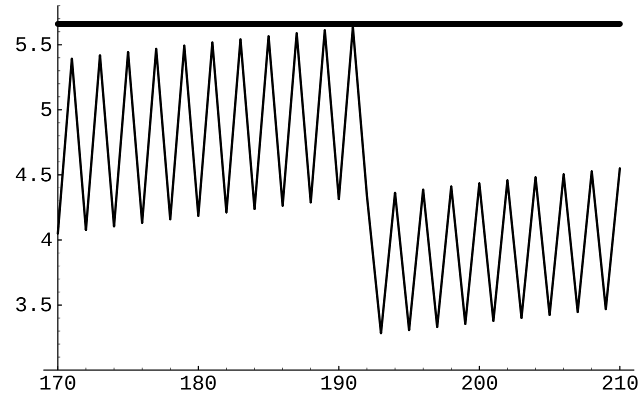

Figure 4: Convergence of binomial model for down-and-in calls (detailed). Note that the approximation is quite close (5.63542 vs 5.6605) at $n = 191$. Also observe that the formula consistently underestimates the analytical value.

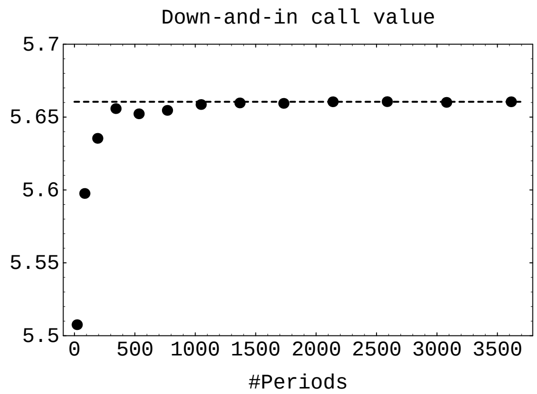

Figure 5: CONVERGENCE OF BINOMIAL MODEL FOR DOWN-AND-IN CALLS AT WELL-CHOSEN n'S. The formula (5) is evaluated at $n = 21\,(1)$, $84\,(2)$, $191\,(3)$, $342\,(4)$, $533\,(5)$, $768\,(6)$, $1047\,(7)$, $1368\,(8)$, $1731\,(9)$, $2138\,(10)$, $2587\,(11)$, $3078\,(12)$, $3613\,(13)$ with the correponding j parenthesized. The analytical value is reached when n equals 2138.

n	Combinatorial method		Trinomial tree algorithm	
	Value	Time	Value	Time
21	5.507548	0.30		
84	5.597597	0.90	5.634936	35.0
191	5.635415	2.00	5.655082	185.0
342	5.655812	3.60	5.658590	590.0
533	5.652253	5.60	5.659692	1440.0
768	5.654609	8.00	5.660137	3080.0
1047	5.658622	11.10	5.660338	5700.0
1368	5.659711	15.00	5.660432	9500.0
1731	5.659416	19.40	5.660474	15400.0
2138	5.660511	24.70	5.660491	23400.0
2587	5.660592	30.20	5.660493	34800.0
3078	5.660099	36.70	5.660488	48800.0
3613	5.660498	43.70	5.660478	67500.0
4190	5.660388	44.10	5.660466	92000.0
4809	5.659955	51.60	5.660454	130000.0
5472	5.660122	68.70		
6177	5.659981	76.70		
6926	5.660263	86.90		
7717	5.660272	97.20		
8552	5.660596	107.90		
9427	5.660215	120.80		
10348	5.660588	132.70		
11309	5.660360	146.90		
12314	5.660389	159.70		
13361	5.660287	173.00		
14452	5.660389	187.30		
15585	5.660367	202.60		
16762	5.660511	219.30		
17979	5.660296	235.20		
19242	5.660479	253.30		
20545	5.660342	270.00		
21892	5.660346	288.70		
23283	5.660346	307.00		
24714	5.660327	325.70		
26189	5.660306	343.10		
27708	5.660385	358.80		
29269	5.660399	376.00		
30872	5.660356	395.20		
32519	5.660402	415.40		
34208	5.660416	436.80		
35939	5.660382	460.90		
37714	5.660422	488.90		
39531	5.660417	518.00		
41392	5.660477	530.60		
43293	5.660391	555.30		
45240	5.660470	573.90		
47227	5.660413	595.00		
49258	5.660416	620.30		
51333	5.660472	649.10		
53450	5.660491	684.30		

Figure 6: THE LINEAR-TIME, BINOMIAL MODEL-BASED ALGORITHM VS THE TRINOMIAL TREE ALGORITHM. All the times are in thousandths of a second (milliseconds). The analytical value is again 5.6605. The data were generated on a personal computer equipped with a 100 MHz Intel Pentium processor and 32 MB of DRAM, running Windows NT 4.0. (The trinomial tree algorithm converges at $n = 1569$, to be precise.) Note that the binomial model-based algorithm takes less time at $n = 2138$ than the trinomial tree algorithm at $n = 84$.

	Barrier at 95.0			Barrier at 99.5			Barrier at 99.9	
n	Value	Time	n	Value	Time	n	Value	Time
	⋮		795	7.47761	8.0	19979	8.11304	253.0
2743	2.56095	31.1	3184	7.47626	38.0	79920	8.11297	1013.0
3040	2.56065	35.5	7163	7.47682	88.0	179819	8.11300	2200.0
3351	2.56098	40.1	12736	7.47661	166.0	319680	8.11299	4100.0
3678	2.56055	43.8	19899	7.47676	253.0	499499	8.11299	6300.0
4021	2.56152	48.1	28656	7.47667	368.0	719280	8.11299	8500.0
4378	2.56095	53.0	39003	7.47674	500.0	979019	8.11299	11800.0
4751	2.56160	57.7	50944	7.47669	(510.0)			
			64475	7.47673	(650.0)			
			79600	7.47670	(820.0)			
			96315	7.47673	(980.0)			
			114624	7.47671	(1120.0)			
			134523	7.47673	(1320.0)			
			156016	7.47671	(1530.0)			
			179099	7.47673	(1760.0)			
			203776	7.47671	(1990.0)			
			230043	7.47673	(2250.0)			
			257904	7.47672	(2520.0)			
			287355	7.47672	(2830.0)			
Analytic value	2.5615			7.4767			8.1130	

Figure 7: THE LINEAR-TIME ALGORITHM WHEN THE CURRENT STOCK PRICE IS NEAR THE BARRIER. All the times are in milliseconds. The parameters are taken from Cheuk and Vorst [1996], where $S = 100$, $X = 100$, $r = 10\%$ (continuous compounded), $\sigma = 0.2$, and $\tau = 0.5$. The analytic value of an otherwise identical European call is 9.2778. The analytic values of the down-and-in calls are then calculated from the down-and-out call values in Exhibit 8 of Cheuk and Vorst [1996] via the in-out parity. Numbers in parenthesis are measures based on a faster Sun SPARCstation.

Barrier at 95.0			Barrier at 99.5			Barrier at 99.9		
n	Estimated	Actual	n	Estimated	Actual	n	Estimated	Actual
	⋮		795	10.20	8.0	19979	256.25	253.0
2743	35.18	31.1	3184	40.84	38.0	79920	1025.05	1013.0
3040	38.99	35.5	7163	91.87	88.0	179819	2306.36	2200.0
3351	42.98	40.1	12736	163.35	166.0	319680	4100.22	4100.0
3678	47.17	43.8	19899	255.22	253.0	499499	6406.57	6300.0
4021	51.57	48.1	28656	367.54	368.0	719280	9225.49	8500.0
4378	56.15	53.0	39003	500.25	500.0	979019	12556.90	11800.0

Figure 8: The accuracy of the performance predictor. Data for the actual running times are extracted from Figure 7.

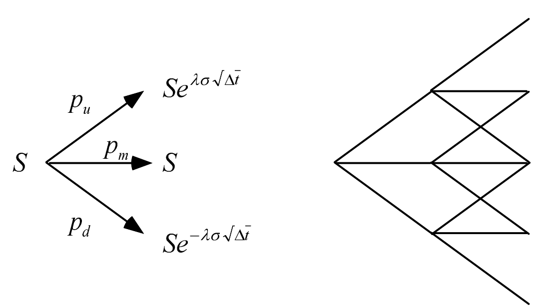

Figure 9: TRINOMIAL MODEL FOR STOCK PRICES.

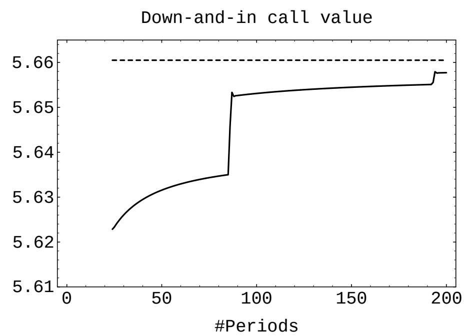

Figure 10: CONVERGENCE OF TRINOMIAL MODEL FOR DOWN-AND-IN CALLS. Plotted are the down-and-in call values as computed by the trinomial tree algorithm against the number of time steps. The parameters are identical to those used in Figure 3. The analytical value, 5.6605, is also plotted for reference.

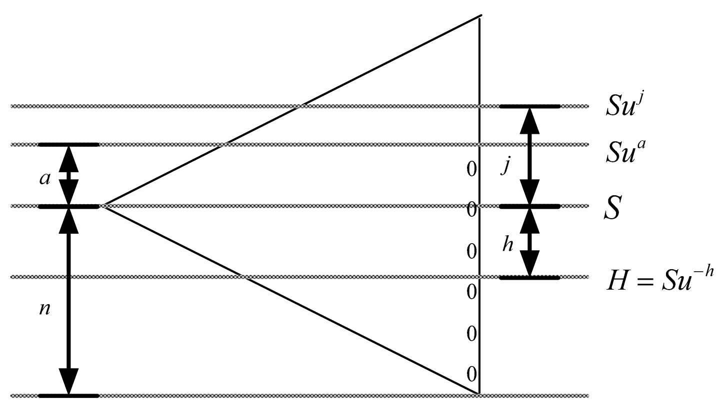

Figure 11: DOWN-AND-IN CALL AND TRINOMIAL TREE. Note that the interpretations of a, j, and h differ from those in Figure 2.