Report: Winner Picking

Shawn O'Neil <soneil@cse.nd.edu>

1 Introduction

For this report, we intend to take a close look at a couple of results seen in "Making Decisions in the Face of Uncertainty: How to Pick a Winner Almost Every Time" by Awerbuch et al[1]. In this paper, the authors consider a class of problems where a decision maker must decide to choose amongst a number of resources at some point in time as information about the resources is revealed incrementally.

In particular, the first result demonstrated in the paper deals with job scheduling on multiple machines. In this setting, a number of computing workstations are available publicly to run jobs. Time is discretized into steps, and at each step each workstation is revealed to be either "open" or available to accept jobs, or not. There are n such workstations, and our heroic decision maker wishes to complete a job which will take d steps. The decision maker, then, wishes to initiate his job on a workstation when it is open, and he wants that workstation to be open for at least d subsequent (not necessarily consecutive) steps.

The difficulty with the decision of when and where to start the job lies in the fact that very little is known about the future availability of the workstations. Indeed, it is possible that the vast majority of the workstations are only available for less than d slots in the future.

If our decision maker is only allowed to decide once when and where to start his job, and he either succeeds or fails at getting his d-step job done, what can we do for him? Clearly, we must make some assumptions about the total availability of the workstations. (That is, if it is possible that all workstations are always busy, it is impossible for us to make any real guarantees.) What such assumptions must we make, and what chance can we give our decision maker of success under those assumptions?

As it turns out, the only assumption we make relates to the availability of the most available workstation. If there is at least one workstation which will be available for $D \geq 3dln(n)$ steps, then we can supply a randomized algorithm which succeeds in completing a d step job with high probability; namely with probability $1 - O(\frac{dln(n)}{D} + \frac{1}{n})$.

We'll begin by describing the decision making algorithm, then look at an analysis to show that it does in fact work. While the original paper paper by Awerbuch et al describes the probability bounds in terms of order notation, here we'll go into further detail to get a (slightly) better idea of the actual probability for success.

2 The Algorithm

The algorithm itself is straightforward. We label our n workstations $W_1, W_2, ..., W_n$. At each step, the decision maker checks to see whether each workstation W_i is available. If it is, he flips a coin which comes up heads with probability $n^{3x/(D-2)}/d$, where x is the number of steps that workstation has been available so far. As soon as he sees a heads, he commits his job to that workstation. If the chosen workstation is available for d future steps, his job finishes and he succeeds. If not, his cause is a failure.

We define a sample space S of coin flips, where each element is the result of all the coin flips that could be made by flipping coins at each open slot for each workstation. (We suppose that the algorithm continues to flip coins, even after committing the job to a workstation.) Figure 1 shows a graphical representation for a possible element of S where there are five workstations. The circles depict open slots, with the letter contained indicating the result of the coin flip at that point. Empty spaces indicate non-open slots.

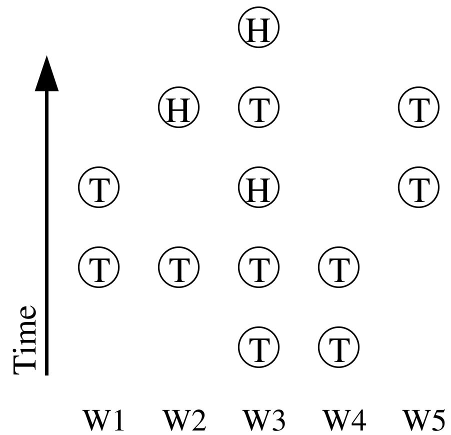

Figure 1: Depiction of a sample space element. Each circle indicates an "open" slot for a workstation, with the contained letter indicating the result of a coin flip at that point. In this example, the job would be scheduled on W_3 in the third time slot.

If an element s of set S is the result of the flips, then the job will be started on the workstation which sees the first heads as we scan from bottom to top, left to right.

Define S_{win} to be a subspace of S wherein each element results in the job being completed, that is, when the workstation selected is available for at least d subsequent steps (and thus will experience at least d more coin flips.) The probability that the job is completed successfully is then $Pr[S_{win}]$.

To show that $Pr[S_{win}]$ is large, we'll first consider another subspace S' and show that $Pr[S']$ is large. Then we'll construct and injection f from S' to $f(S')$, where all elements of $f(S')$ are elements of S_{win}. At that point, we'll be able to compute $Pr[f(S')]/Pr[S']$, which must be a lower bound on $Pr[S_{win}]/Pr[S']$, because $|S_{win}| \geq |f(S')|$. Combining that with the known $Pr[S']$, we can finally give a lower bound for $Pr[S_{win}]$.

2.1 S' and a lower bound for $Pr[S']$

We define S' to be the sample points for which there is at least one head and for which the first d flips for each workstation resulted in tails. We are interesed in computing a lower bound for $Pr[S']$.

The probability of a head in one of the first d flips of a particular workstation is at most $n^{3d/(D-2)}/d$. This is because the probability of getting a heads on the first opportunity is $n^{3 \cdot 1/(D-2)}/d$, on the second $n^{3 \cdot 2/(D-2)}/d$, etc.

Thus, the probability of getting a heads among the (at most, for S') dn flips for which $x \leq d$ (which must satisfy the condition that the first d flips for each workstation resulted in tails) is at most

$$\sum_{i=1}^{dn} n^{\frac{3d}{D-2}}/d \;=\; dn\left(n^{\frac{3d}{D-2}}/d\right) \;\leq\; \frac{2}{n}$$

Thus, the probability that the first d flips result in tails is large, namely $\geq 1 - \frac{2}{n}$.

Here, we'll switch gears a bit, and argue that the probability that there are no heads in the last d flips of the workstation which is available for $D \geq 3dln(n)$ steps is small. This implies that the probability that there is at least one head overall (and after the first d open slots) is large, and is the remaining condition for showing that the probability of S' is large.

It is important to note that the probability of a flip at one point is independent of other flips, because the probability depends only on the number of times the workstation has been available. Further, in the sample points considered we are flipping regardless of whether we have previously seen a heads or not.

Now, again, we are interested in the probability that there are no heads among the last d flips for the workstation which is available for D steps. The probability of a heads on the first of the last d flips is given by $n^{3(D-d)/(D-2)}/d$.

This is also a lower bound on the probability of getting a heads on subsequent flips (the probability of heads increases for later flips on this workstation.) So, $1-n^{(D-d)/(D-2)}/d$ is an *upper* bound on the probability of *not* getting a heads for any of those flips, and

$$\left(1-n^{\frac{D-d}{D-2}}/d\right)^d \leq \left(1-\frac{n}{2d}\right)^d \leq e^{\frac{-n}{2}}$$

is an upper bound on the probability of not getting a heads in all of the last d flips. Thus, the probability of getting a heads overall is large, namely at least $1-e^{-n/2}$. Now, we can compute $Pr[S']$:

$$Pr[S'] \quad \geq \quad \left(1-\frac{2}{n}\right)\left(1-e^{-n/2}\right)$$

2.2 The Injection f

Now we need to create and injection from S' to S_{win}, which will show that $|S_{win}| \geq |S'|$.

We consider a sample point s' in S'. We let x be the number of steps the first machine with a heads was open, up to and including the slot where the first head was seen. By the definition of S', we know that $x > d$. We define z' as follows:

$$z' = n^{\frac{3x}{D-2}}/d$$

That is, on the x^{th} flip for that workstation, z' is the a-priori probability that that flip would come up heads (which it did, a-posteriorly.) We also define z as follows:

$$z = n^{\frac{3(x-d)}{D-2}}/d = n^{\frac{3d}{D}}z'$$

So, z is the a-priori probability of heads on the $x-d^{th}$ flip for that workstation. We also let J be the a-priori probability of everything else which occurred in s'. Figure 2 shows a representation of s' along with the probabilities described. We can then compute the probability of s' as:

$$Pr[s'] = z'(1-z)J$$

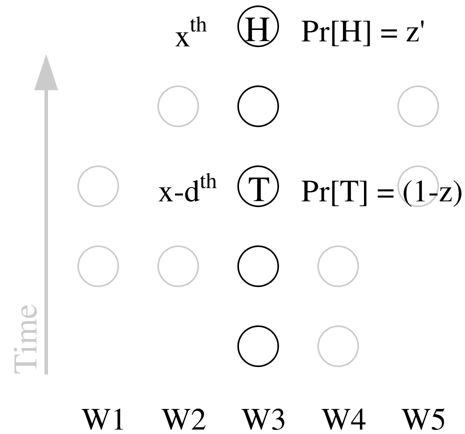

Figure 2: s'. Graphical representation of s', a single element of S'.

Now, for the injection. First consider the case where $z' \geq 1/2$. If this is the case, we "flip" the result of the $x-d^{th}$ outcome from tails to heads, and leave everything else alone. Clearly, this new outcome $f(s')$ is in S_{win}. Also, the probability of $f(s')$ in this case is $zz'J$. See figure 3.

If $z' < 1/2$, we use a different transformation for $f(s')$. Here, we flip the $x-d^{th}$ outcome from tails to heads as before, *and* we flip the x^{th} outcome from heads to tails, leaving the rest alone. Again, $f(s')$ in this case is clearly in S_{win}, and the probability of $f(s')$ is $(1-z')zJ$. Figure 4 shows the transformation for this case.

2.3 Proof that f is an Injection

Let's show that f is indeed an injection, meaning that for every element of s', it is mapped to a unique point in $f(s')$.

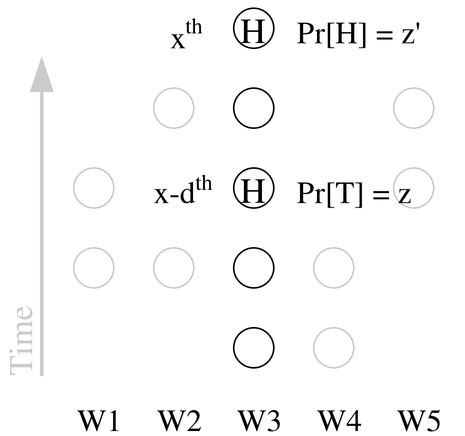

Figure 3: $f(s'), z' \geq 1/2$. Graphical representation of $f(s')$, when $z' \geq 1/2$. The probability of $f(s')$ in this case is $zz'J$.

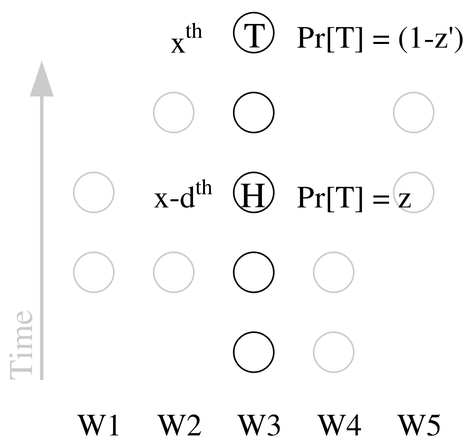

Figure 4: $f(s'), z' < 1/2$. Graphical representation of $f(s')$, when $z' < 1/2$. The probability of $f(s')$ in this case is $(1 - z')zJ$.

We'll do this by showing that for every element of S_{win}, it either "came from" the domain of f where $z' \leq 1/2$, or the domain of f where $z' > 1/2$ (but not both) or from outside the domain of f. (If an element of S_{win} is such that it couldn't have been a result of f on an element of S', we don't need to consider it in determining if f is a valid injection.)

Consider an element e of S_{win}, and let q be the first place where a heads occurs in that series of flips. We can determine if e is a result of the injection f on an element of S' by looking at the flips between the q^{th} flip and the $q + d^{th}$ flip. If any of these are heads, then this element of S_{win} is not the result of the injection on an element of S', because if it were q would not be first place a heads occurs. Figure 5 shows such an element.

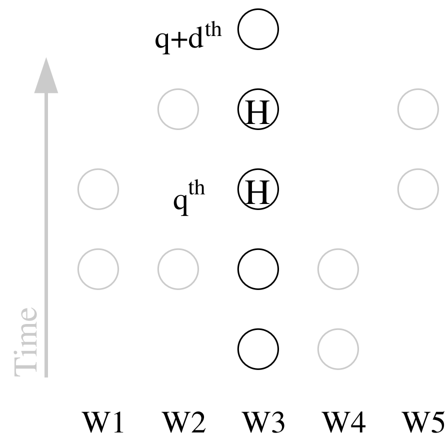

Figure 5: An element of S_{win}. If q is the first place a heads occurs, and there are flips which resulted in heads between the q^{th} and $q + d^{th}$ flips, then this element cannot be the result of the injection.

On the other hand, if there are no heads between the q^{th} and $q + d^{th}$ flip in e, then it is possible that this element is a result of the injection. For f to actually be an injection, we need to ensure that it is not possible for multiple elements of S' to be mapped to this element e. Clearly, all of the different elements of S' for which $z' < 1/2$ map to unique elements in $f(S')$. This is also true for elements of S' for which $z' \geq 1/2$.

Now, given any element of e of S_{win}, if it is possibly the result of the injection, we just need to ensure that it couldn't have come from the domain S' where z' is both less then or greater than or equal to $1/2$. Clearly this is impossible. Further, given such an element e, we can uniquely determine which subset of S' it came from, by inspecting the $q + d^{th}$ result. If it is a heads, then z' was $\geq 1/2$. If it is a tails, then z' was $< 1/2$. This is by the definition of f; glancing back at Figures 3 and 4 will give some intuition.

2.4 Computing $Pr[S_{win}]$

While the original paper used $D \geq 3dln(n)$, we'll instead look at $D = 3drln(n)$, where $r \geq 1$, to get a better look at the actual probability of success.

Using the injection f, we need to find $Pr[f(s')]/Pr[s']$ for all s'. We start by computing z exactly, in terms of z' and our modified D.

$$\begin{aligned} z &= n^{\frac{-3d}{D}} z' \\ &= n^{\frac{-3d}{3drln(n)}} z' \\ &= n^{\frac{-1}{rln(n)}} z' \\ &= \frac{1}{e^{1/r}} z' \end{aligned}$$

As we've seen, if $z' > 1/2$, then

$$\begin{aligned} \frac{Pr[f(s')]}{Pr[s']} &= \frac{z}{1-z} \\ &= \frac{\frac{1}{e^{1/r}} z'}{1 - \frac{1}{e^{1/r}} z'}. \end{aligned}$$

This is decreasing with decreasing z', so we use $z' = 1/2$ to come up with a lower bound:

$$\begin{aligned} \frac{Pr[f(s')]}{Pr[s']} &\geq \frac{\frac{1}{2e^{1/r}}}{1 - \frac{1}{2e^{1/r}}} \\ &= \frac{1}{2e^{1/r} - 1}. \end{aligned}$$

Similarly, if $z' \leq 1/2$, our injection specifies a different formula:

$$\begin{aligned} \frac{Pr[f(s')]}{Pr[s']} &= \frac{z}{1-z} \frac{1-z'}{z'} \\ &= \frac{\frac{1}{e^{1/r}} z'}{1 - \frac{1}{e^{1/r}} z'} \frac{1-z'}{z'} \\ &= \frac{\frac{1}{e^{1/r}} z' - \frac{1}{e^{1/r}} z'^2}{z' - \frac{1}{e^{1/r}} z'^2} \\ &= \frac{\frac{1}{e^{1/r}} - \frac{1}{e^{1/r}} z'}{1 - \frac{1}{e^{1/r}} z'} \\ &= \frac{1-z'}{e^{1/r} - z'} \end{aligned}$$

This is decreasing with increasing z', so we use $z' = 1/2$ to come up with a lower bound:

$$\begin{aligned} \frac{Pr[f(s')]}{Pr[s']} &\geq \frac{\frac{1}{2}}{e^{1/r} - \frac{1}{2}} \\ &= \frac{1}{2e^{1/r} - 1}. \end{aligned}$$

Thus, in either case, $Pr[f(s')]/Pr[s'] \geq 1/(2e^{1/r} - 1)$. Since this is true for all elements of S', $Pr[S_{win}]/Pr[S']$ must also have a lower bound of $1/(2e^{1/r} - 1)$, as $|S_{win}|$ is at least as large as $|f(S')|$.

So, given

$$\frac{Pr[S_{win}]}{Pr[S']} \geq \frac{1}{2e^{1/r} - 1}.$$

and

$$Pr[S'] \geq \left(1 - \frac{2}{n}\right)\left(1 - e^{-n/2}\right),$$

we know that

$$\begin{aligned} Pr[S_{win}] &\geq \frac{Pr[S']}{2e^{1/r} - 1} \\ &\geq \frac{\left(1 - \frac{2}{n}\right)\left(1 - e^{-n/2}\right)}{2e^{1/r} - 1}. \end{aligned}$$

2.5 Comments on $Pr[S_{win}]$

Now we have a firm bound on the probability of success in terms of the number of machines and r, our factor for how much bigger D actually is than $3dln(n)$, and we can closely look at our chances. As is expected, the more the most available machine is open is (the larger r), the larger our probability bound as the expression given in the last section is increasing in r. It is also increasing in n, indicating that having more machines to choose from also improves our chances (in terms of the bound given by the analysis anyway). Figure 6 shows a graph indicating the probability of success for a given run, in terms of r and n.

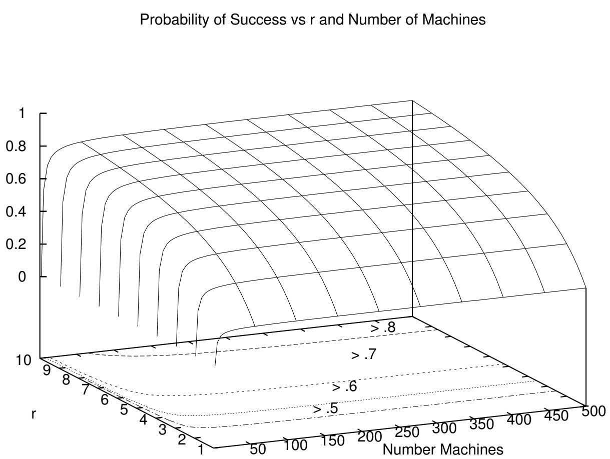

Figure 6: Probability of success given in terms of r and n.

Note that while the original paper only assumed $D \geq 3dln(n)$, this could be as bad as having $r = 1$. If we set $r = 1$ in the bound used above, we see than no amount of machines will allow us a probability better than $1/(2e-1) \approx .225$.

3 An Interesting Idea

We can attempt to apply the algorithm shown to the problem of stock picking. Specifically, if we are given a number of stocks, and told that one of the bunch will rise very high over some time interval, we can use the algorithm to get some of that increase with high probability.

We look at each stock as a machine. Given a prediction P for the percentage increase of the best stock, we break the increase up into steps of a geometrically increasing nature. That is, given a stock price S_0 at time 0 and a multiplicative factor α (which could be, for example, 1.05), we have steps at $S_0\alpha, S_0\alpha^2, S_0\alpha^3$, and so on. We consider a stock to be "open" for selection the first time we see one of our steps on that stock. Figure 7 shows a graphical representation.

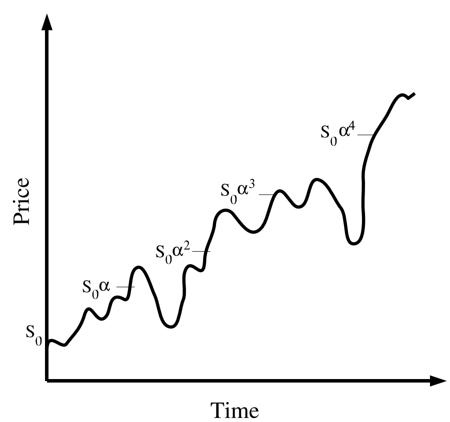

Figure 7: View of a stock process, and how it is discretized into steps. The stock is said to be "open" for selection when it is at a new high $S_0\alpha^i$ value.

If we know that our best stock will increase P percent, then we are guaranteed that that stock will be open D steps, where $\alpha^D = P$. First recall that for the algorithm given previously, D must be $3drln(n)$, where $r \geq 1$. By selecting α appropriately, we can make D as large as we like. We also have control over r, which will come into play later.

By the winner picking algorithm, we can with high probability guarantee d open slots, meaning that we can purchase a stock at some point and see d α-increases before we sell it again. The return is then α^d. By solving $\alpha^D = P$ for α and substitution we have

$$\begin{aligned} HighProbabilityReturn &= \alpha^d = \left(P^{\frac{1}{D}}\right) \\ &= \left(P^{\frac{1}{3drln(n)}}\right)^d \\ &= P^{\frac{1}{3rln(n)}} \end{aligned}$$

Assuming the stock process is continual and we can trade continuously, α can be as small as necessary and d is immaterial. A bound on the expected return is then the former high probability return times the actual probability of success:

$$E[return] \geq \left(P^{\frac{1}{3rln(n)}}\right) \frac{\left(1 - \frac{2}{n}\right)\left(1 - e^{-n/2}\right)}{2e^{1/r} - 1}$$

Figure 8 plots this curve with $P = 100$, r from 1 to 100, and n from 10 to 1000. (Very small values of n do guarantee very high returns, but obviously it is difficult to say that one of only a few stocks will do very well. The figure shown is even very optimistic, assuming a constant 100-fold increase in the best stock.)

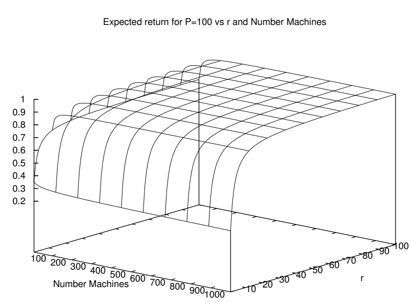

Figure 8: Expected return of running the basic winner picking algorithm on a group of n stocks, where the best stock always experiences a 100-fold increase.

From Figure 8, it seems that our exercise is doomed to failure. It might be interesting, though, to take it to its logical conclusion. Given a number of homogeneous stocks modeled as geometric Brownian motions in the standard way, how well can we expect the best one to do? How many should we select, and what is our expected return given the information we've seen thus far?

We know that if a stock with drift μ and variance σ has a price at time 0 of S_0, then the probability that its price at time T, S_T, is above some value k is given by:

$$Pr[S_T \geq k] = N\left(\frac{ln\left(\frac{S_0}{k}\right) + \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}\right)$$

Where $N()$ is the cumulative normal distribution function. The probability that n such stocks don't end up above k at time T is is then

$$\left(1 - N\left(\frac{ln\left(\frac{S_0}{k}\right) + \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}\right)\right)^n$$

We would like this probability to be small with respect to n (namely $1/n$), so we set the above equal to $1/n$ and solve for k, which gets us the highest value we can expect from any stock at time T with probability $1 - 1/n$.

$$k = S_0 e^{\mu T - \sigma^2 T - \sigma\sqrt{2T} Erf^{-1}\left(0, 1 - 2\left(\frac{1}{n}\right)^{1/n}\right)}$$

Where Erf^{-1} is the inverse of the generalized error function. Without loss of generality, we assume $S_0 = 1$ for all the stocks, and using $P = K$, we have that the expected return in time T for n homogeneous stocks modeled as geometric Brownian motions with drift μ and variance σ is

$$\begin{aligned} E[return] \quad &\geq \quad \frac{\left(1 - \frac{2}{n}\right)\left(1 - e^{-n/2}\right)}{2e^{1/r} - 1}\left(1 - \frac{1}{n}\right) \\ &\cdot \quad e^{\frac{\mu T - \sigma^2 T - \sigma\sqrt{2T} Erf^{-1}\left(0, 1 - 2\left(\frac{1}{n}\right)^{1/n}\right)}{3rln(n)}}. \end{aligned}$$

The $(1 - 1/n)$ term accounts for the fact that one of the stocks will actually reach $P = K$ with probability $1 - 1/n$. While we were unable to get Mathematica to plot the graph in 3 dimensions (for some unexplained reason), Figures 9, 10, and 11 plot the expected return for various values of n holding r constant at 1, 10, and 100 respectively. For all graphs, $\mu = .08$, $\sigma = .25$, and $T = 2$ years.

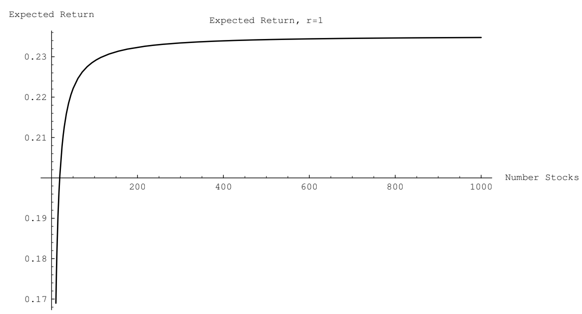

Figure 9: Graph of the expected return as a function of the number of stocks, holding r constant at 1. $\mu = .08$, $\sigma = .25$, $T = 2$ years.

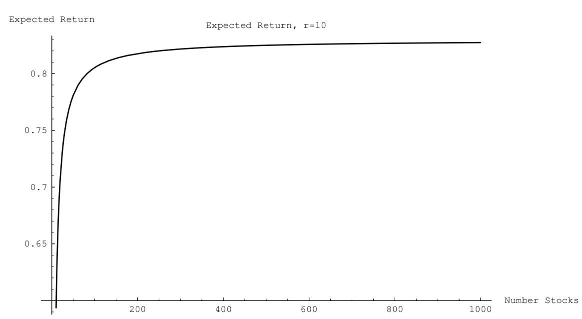

Figure 10: Graph of the expected return as a function of the number of stocks, holding r constant at 10. $\mu = .08$, $\sigma = .25$, $T = 2$ years.

These figures indicate that our solution will not work. Even making plots such as those shown with very high drift rates doesn't have much of an affect. It should also be noted that in our analysis we are looking at the probability that any stock will be above k *at* time T,

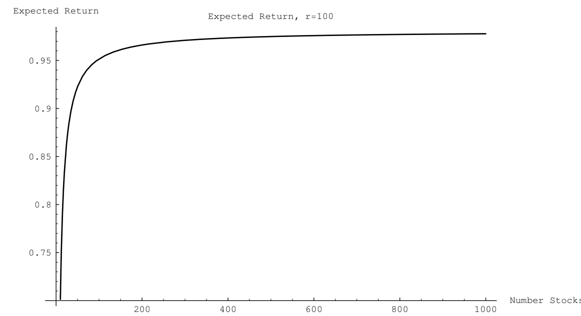

Figure 11: Graph of the expected return as a function of the number of stocks, holding r constant at 100. $\mu = .08$, $\sigma = .25$, $T = 2$ years.

whereas for the algorithm all that is really necessary is for some stock to reach some k *before* T. The relevent expression for determining the maximum of a geometric Brownian motion over a time range is

$$\begin{aligned} Pr\left[\max_{0<t<T} S_t \geq H\right] &= \frac{1}{2}Erfc(d_1) + \frac{1}{2}\left(\frac{H}{S_0}\right)^{\frac{2\mu}{\sigma^2}-1} \\ &\cdot \quad Erfc(d_2) \\ d_1 &= \frac{ln\left(\frac{S_0}{k}\right) - \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \\ d_2 &= \frac{ln\left(\frac{S_0}{k}\right) + \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}} \end{aligned}$$

Where $Erfc()$ is the complimentary error function. So far, we've been unable to use this formula and solve for H giving us the high probability guarantee algebraically. Initial numerical tests indicate that the maximum values seen aren't that much higher, however, and don't lead to any better performance.

References

[1] B. Awerbuch, Y. Azar, A. Fiat, T. Leighton, "Making commitments in the face of uncertainty: how to pick a winner almost every time," Proc. of 28th STOC (1996), 519-530.