COMP 360 Homework 2: Bins and Balls

Matt Adelman

October 21, 2012

All code is attached to this assignment.

It must be stated that for all of the tests I was using the following parameters. Mac OS X Lion 10.7.4, a 3.06 GHz Intel Core 2 Duo processor, and 4 GB 1067 MHz DDR3 memory with a Darwin x86 architecture on a 64 bit system. In addition, I was using Java version "1.6.0_33" from Apple Inc. (found from `$ java -version`). The javac compiler also is the same version. Unfortunately, due to time constraints I was unable to do any tests with larger array sizes and still get accurate data.

The testing facility I used is as follows. The basic outline is that there are two `for` loops where the outermost loops through the array sizes 2^{10} to 2^{20} by powers of 2. Then, for our m values we test $m = n, 2n, 4n, 8n$. We stop at $8n$ because for all of our trials it is greater than $n\lg(n)$ and we can use that to see the limiting behavior we are looking for. For each n, m combination, we create an array `array` that is an instance of the bins and balls problem. We then move the array appropriately, and find the maximum and average values in the array. We do each combination `trials` number of times. After all of our trials, we average the average move distances and the max move distances, and fill our array accordingly. At the end of all the combinations, we print the data array. There are 44 lines to our data array, because each n value has 4 associated m values. We then use this data to see the graphs as follows. Our figures are the averages and maximums associated with $m = n, 2n, 4n, 8n$ for a total of 8 graphs.

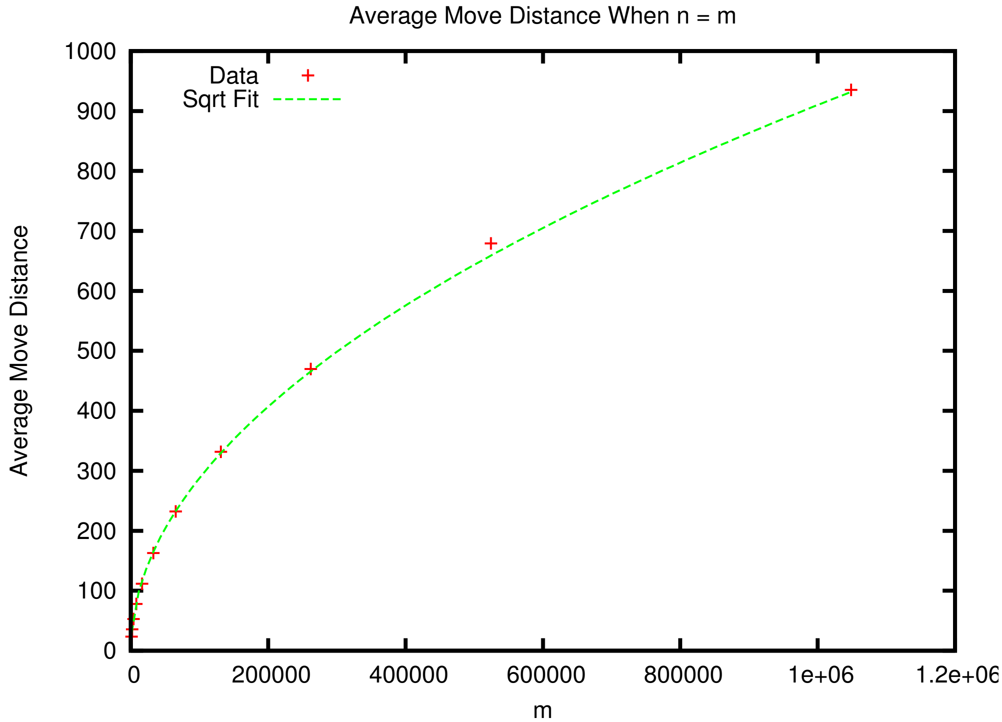
The conclusions I came up with for this experiment are follows. Fortunately, the case when $n = m$ is very easy to interpret, because our graphs line up quite nicely. We see that the average move is proportional to the square root of n, and the maximum move distance is linear in n. These plots can be seen in figures 1 and 2 respectively. Although I am not quite sure why the average would be proportional to the square root of n, it makes sense that the max is linear, because when $n = m$ it is not that unlikely that the closest bin with more than one ball could be very far away.

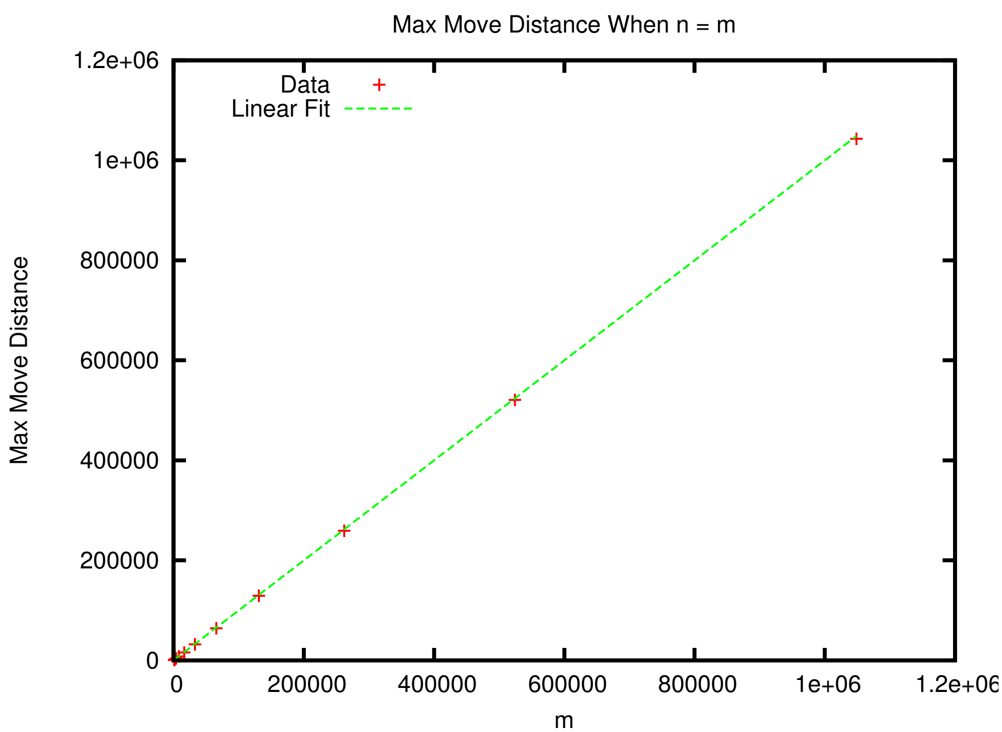
For when $m = 2n$, we see that average very quickly stabilizes to a constant value between .1692 and .1694. This is a very quick drop from the square root scale of our last average. Then for the maximum, it now follows a log like pattern. These can be seen in figures two and three.

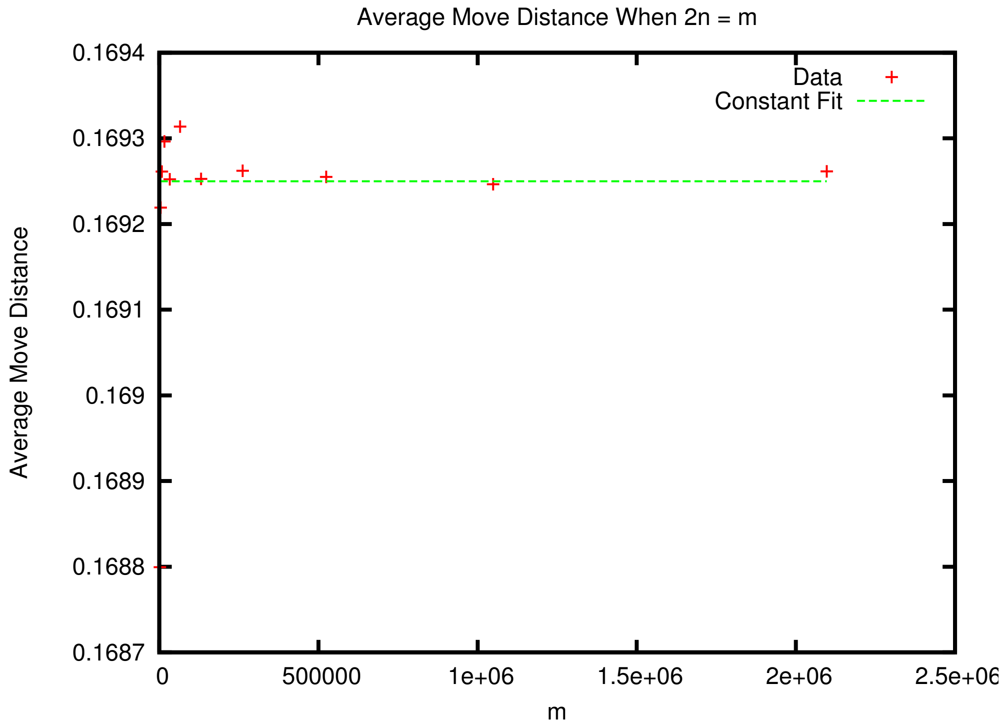
Now, for when $m = 4n$, we get that our best fit is a constant value around .01847. This is another big drop from our last plot, and we see that the best fit for our max data is something like a log(log)) plot, but I couldn't get something exactly right. These results can be seen in figures four and five.

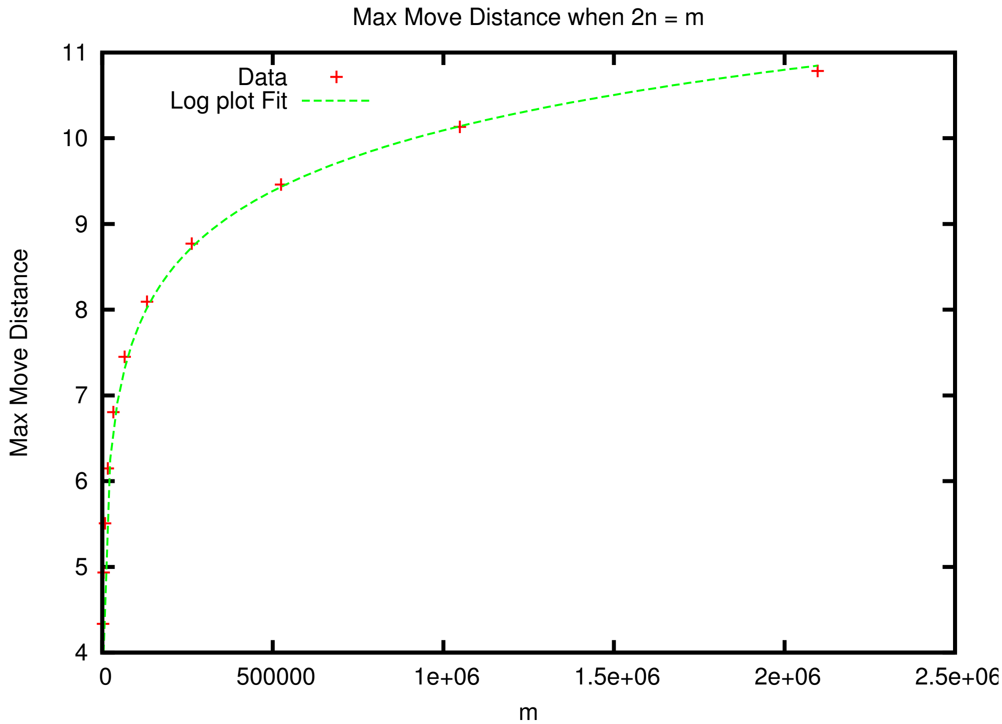
Finally, when $m = 8n > n\lg(n)$, we see that our average move distance is effectively zero. This makes sense, because more often than not, our bins will have more than two balls in them. Also, if a bin somehow does not have a ball, we should be able to find one with more than one ball quite close to it. For tha max distance, we approach one rather rapidly, which makes sense, because if a bin doesn't have a ball, which is already unlikely, it is even more unlikely that the two bins on either side also have less than or equal to one balls. These results can be seen in figures seven and eight.

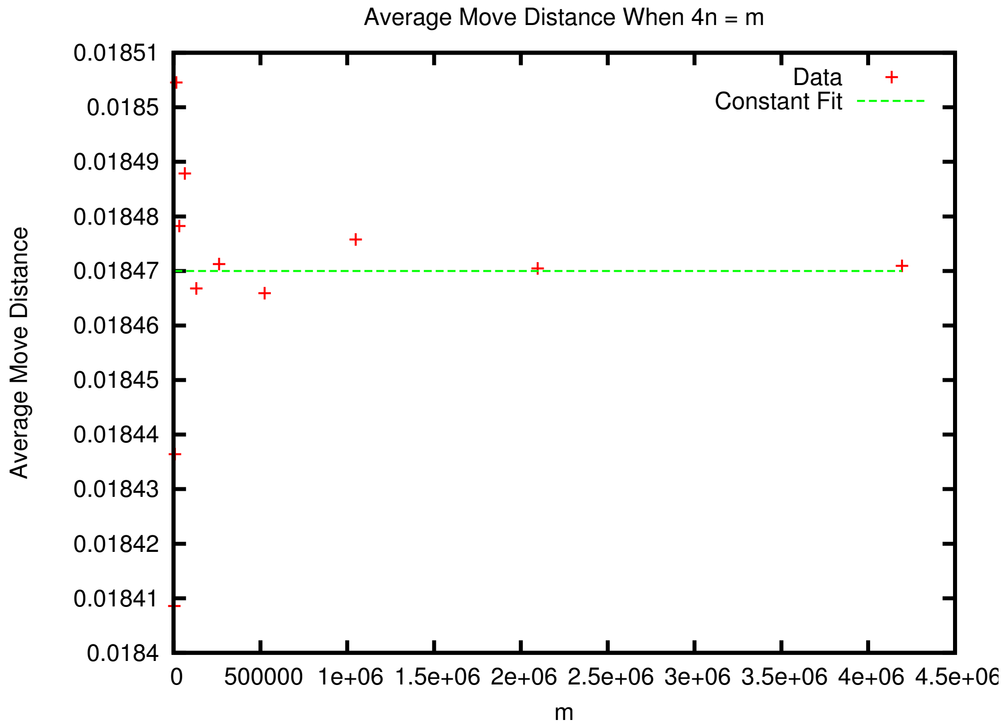
The figures:

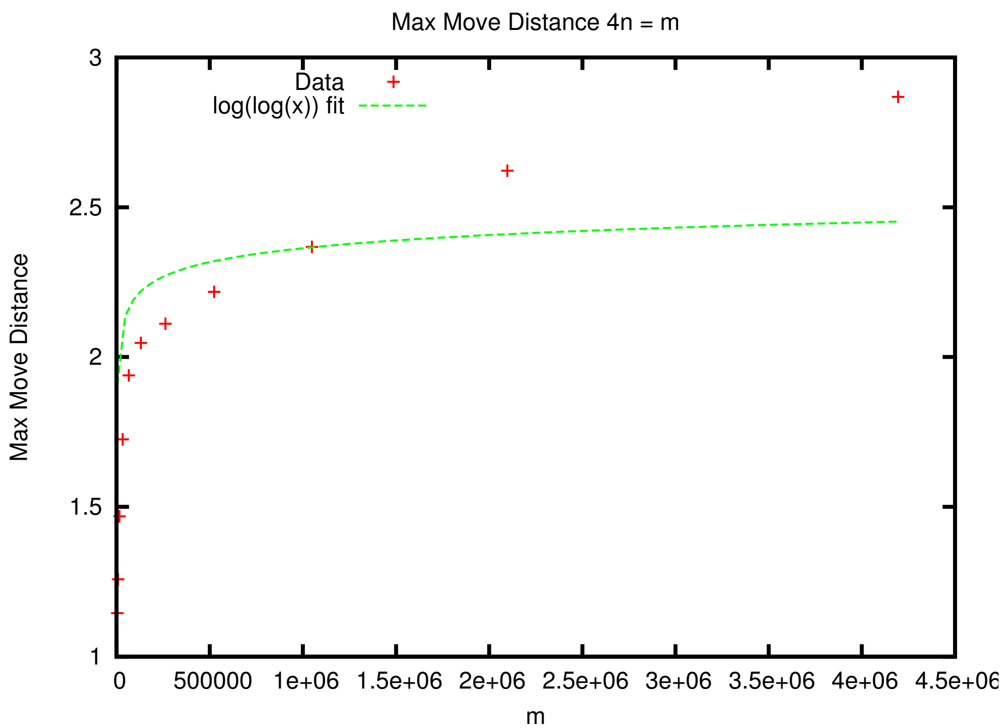

Average Move Distance When n = m
Data
Sqrt Fit
Average Move Distance
m
0
100
200
300
400
500
600
700
800
900
1000
200000
400000
600000
800000
1e+06
1.2e+06

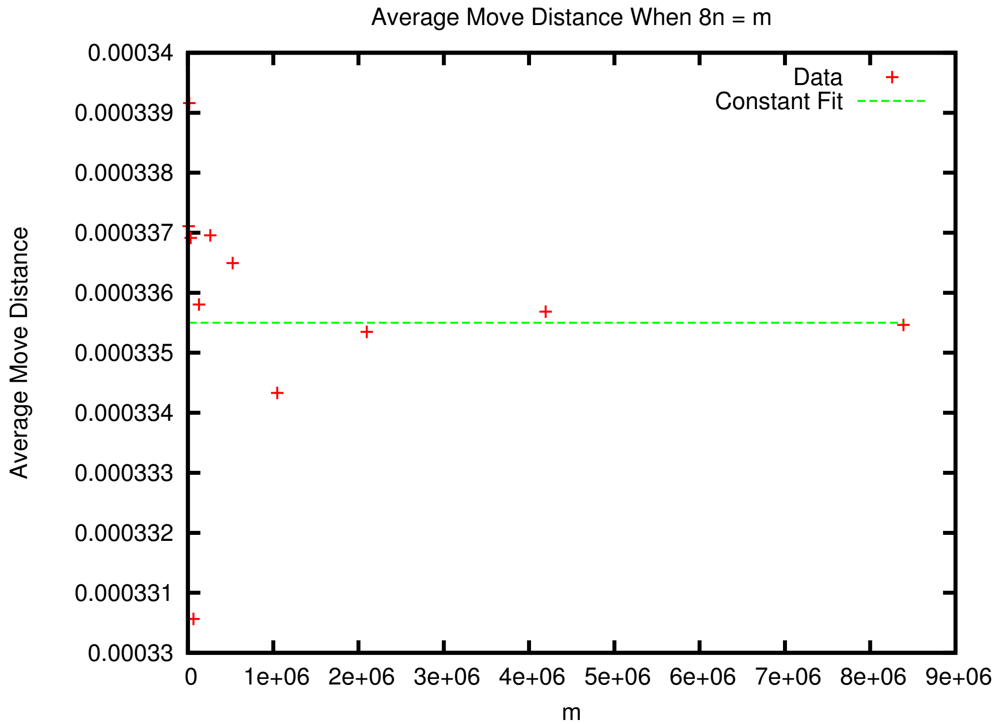

Max Move Distance When n = m
Data
Linear Fit
Max Move Distance
m
0
200000
400000
600000
800000
1e+06
1.2e+06

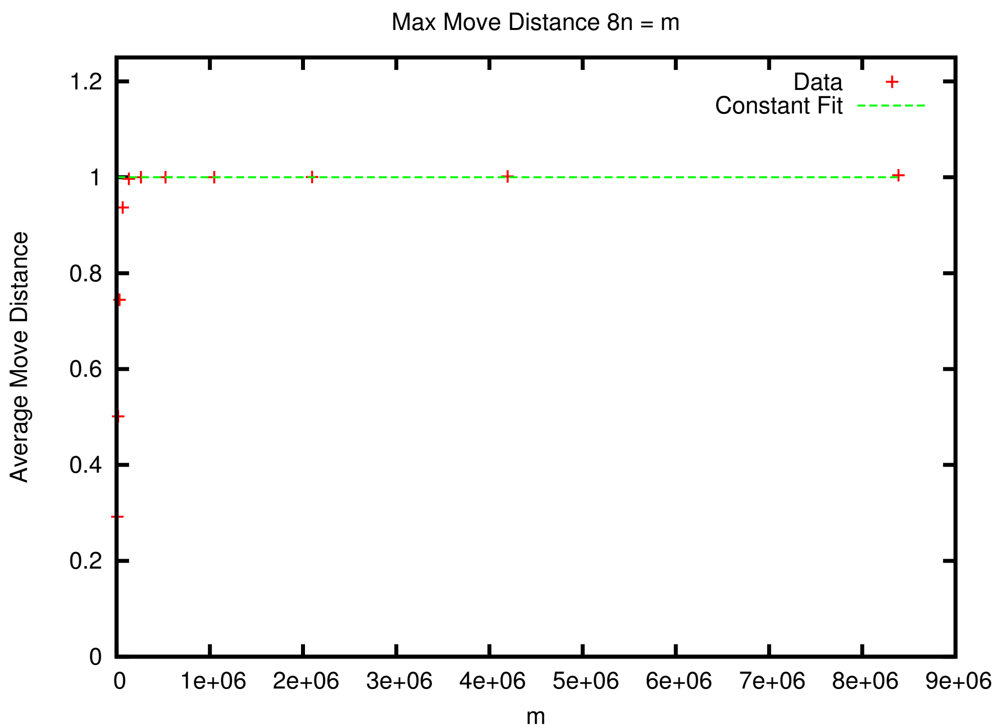

Average Move Distance When 2n = m
Average Move Distance
m
Data
Constant Fit
0.1694
0.1693
0.1692
0.1691
0.169
0.1689
0.1688
0.1687
0
500000
1e+06
1.5e+06
2e+06
2.5e+06


Max Move Distance when 2n = m
Max Move Distance
m
Data
Log plot Fit
11
10
9
8
7
6
5
4
0
500000
1e+06
1.5e+06
2e+06
2.5e+06


Average Move Distance When 4n = m
Data
Constant Fit
Average Move Distance
0.01851
0.0185
0.01849
0.01848
0.01847
0.01846
0.01845
0.01844
0.01843
0.01842
0.01841
0.0184
0
500000
1e+06
1.5e+06
2e+06
2.5e+06
3e+06
3.5e+06
4e+06
4.5e+06
m


Max Move Distance 4n = m
Data
log(log(x)) fit
Max Move Distance
3
2.5
2
1.5
1
0
500000
1e+06
1.5e+06
2e+06
2.5e+06
3e+06
3.5e+06
4e+06
4.5e+06
m


Average Move Distance When 8n = m
Data
Constant Fit
Average Move Distance
m
0.00034
0.000339
0.000338
0.000337
0.000336
0.000335
0.000334
0.000333
0.000332
0.000331
0.00033
0
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
7e+06
8e+06
9e+06


Max Move Distance 8n = m
Data
Constant Fit
Average Move Distance
m
1.2
1
0.8
0.6
0.4
0.2
0
0
1e+06
2e+06
3e+06
4e+06
5e+06
6e+06
7e+06
8e+06
9e+06