**cs577\_Project1 Random Walks**

Goal: To use empirical results to understand random walks on 1 and 2 dimensional arrays.

**Part 1: Random walks on 1-dimensional arrays**

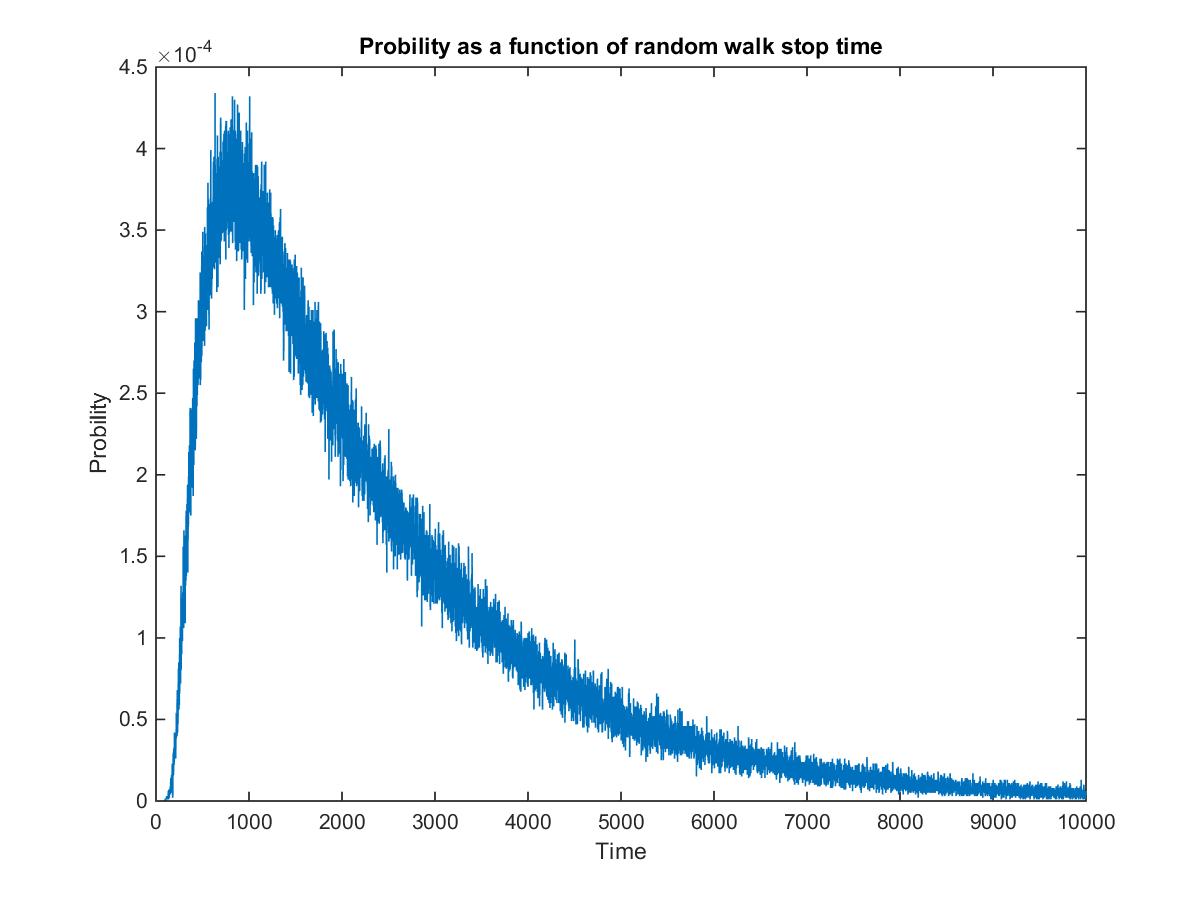
Implementation:

Random walk on a 1-dimensional array.

Initially assume that the array has 100 entries and begin the walk in position 50. Randomly flip a coin and walk toward the beginning of the array for tails and toward the end for heads. Stops if either end is reaches. Keep tracking the number of coin flips.

1. Compute the probability the walk reaches an end of the array. p(t).

Figure 1.1



x: t is the time when the walk reaches an end of the array

y: P(t) = the probability that the walk reaches an end of the array

Way:

Randomly flip coins 1000000 times.

Make the position before the first flip at pos = 50, flip coins.

If the new pos == 0 or 100 (head or tail), terminate the program.

flip coin:

If the coin is head, then go forward to the right,

If the new pos > 50, then go backward to the head.

t = count the number of times flip coins

Return the probability of t.

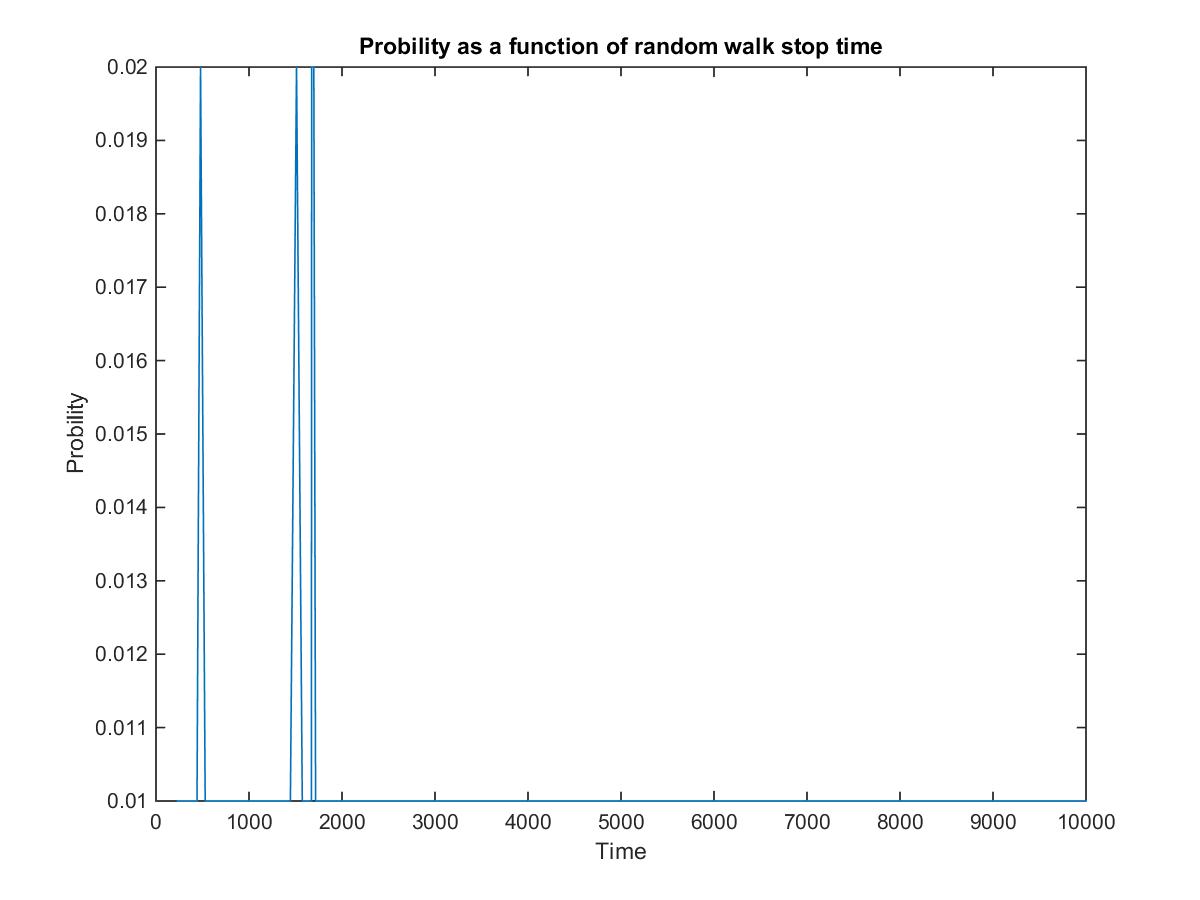
Conclude: the probability increases fast at the beginning and reaches the biggest probability (about 4.4 \* 104) and then decreases slow to a very small positive number from t≈10000 to t≈1000000

**Questions:**

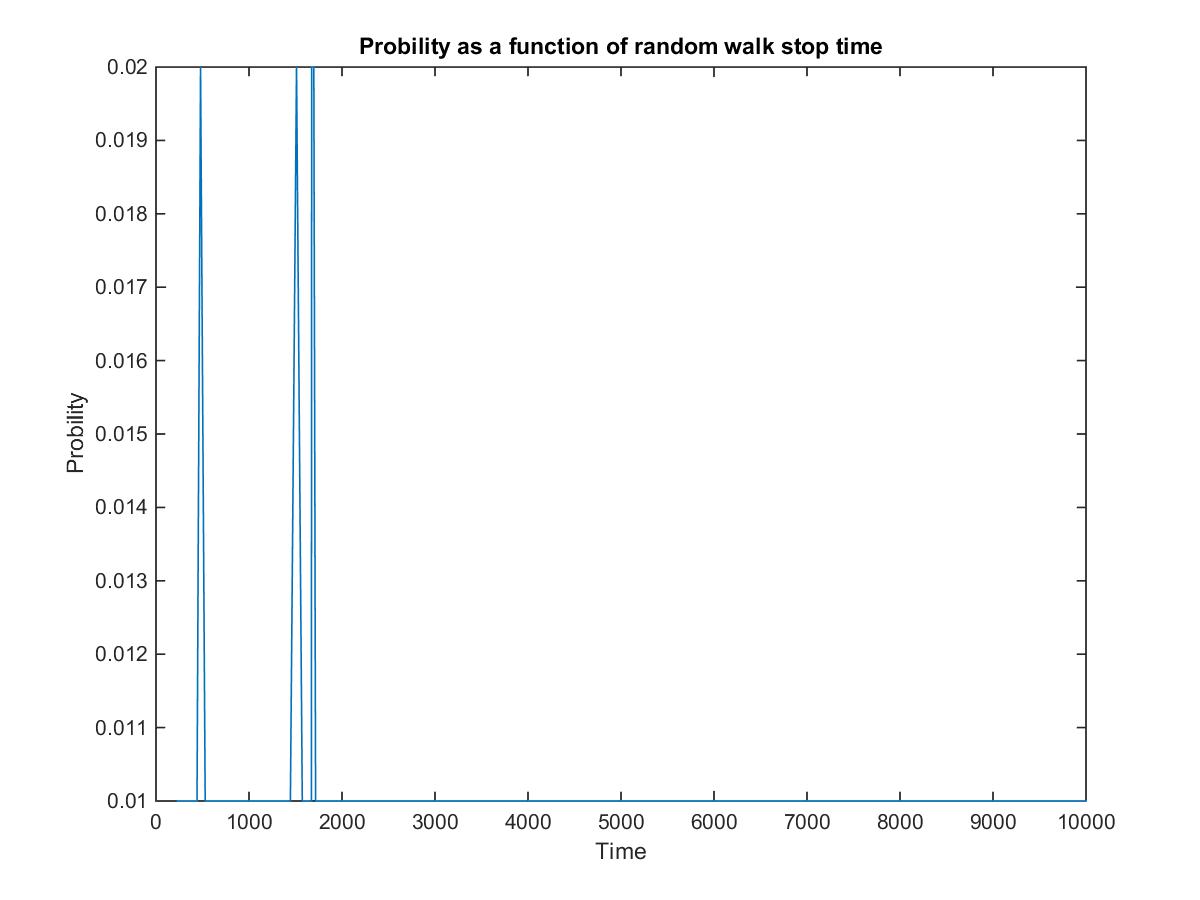
1. # of simulations need to run to make a reliable prediction:

Try different simulations: TOTAL\_RANS

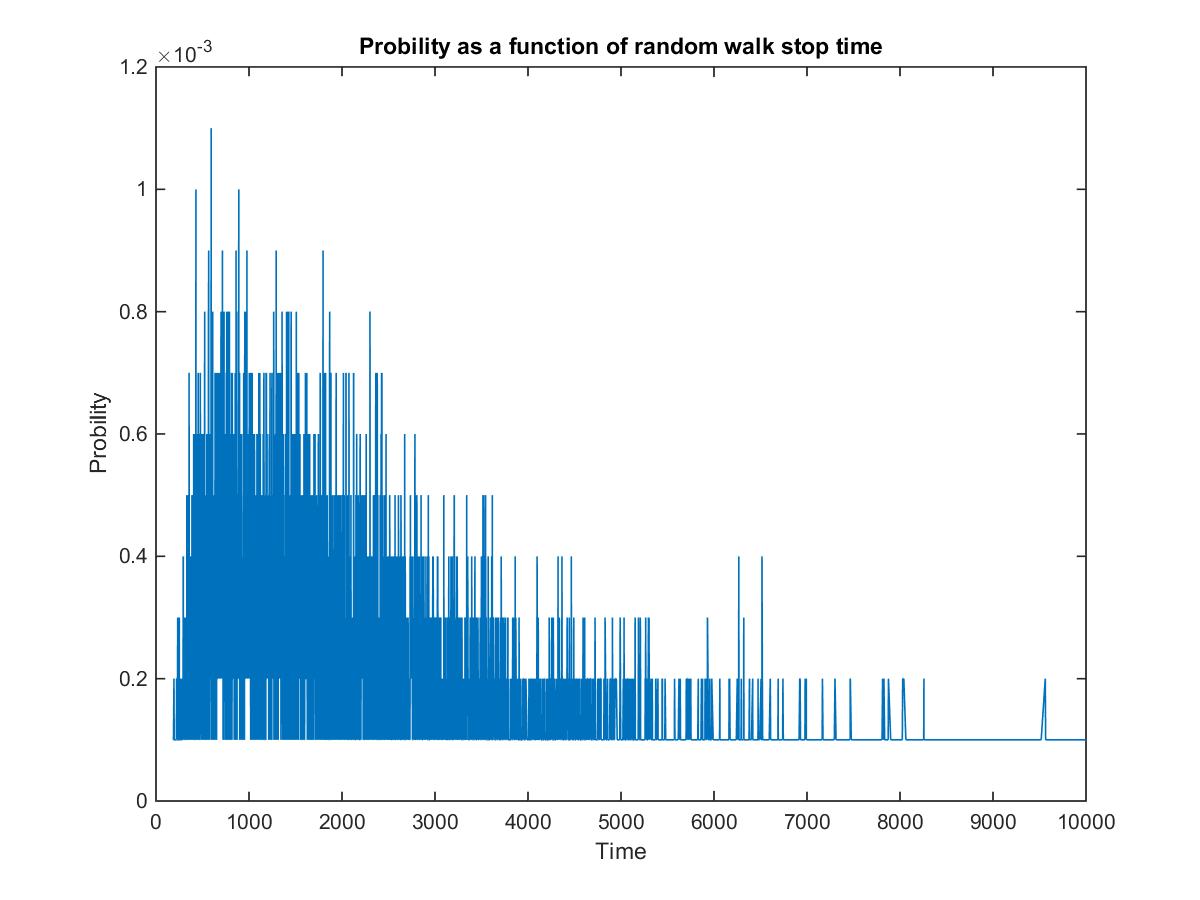
TOTAL\_RUNS = 100



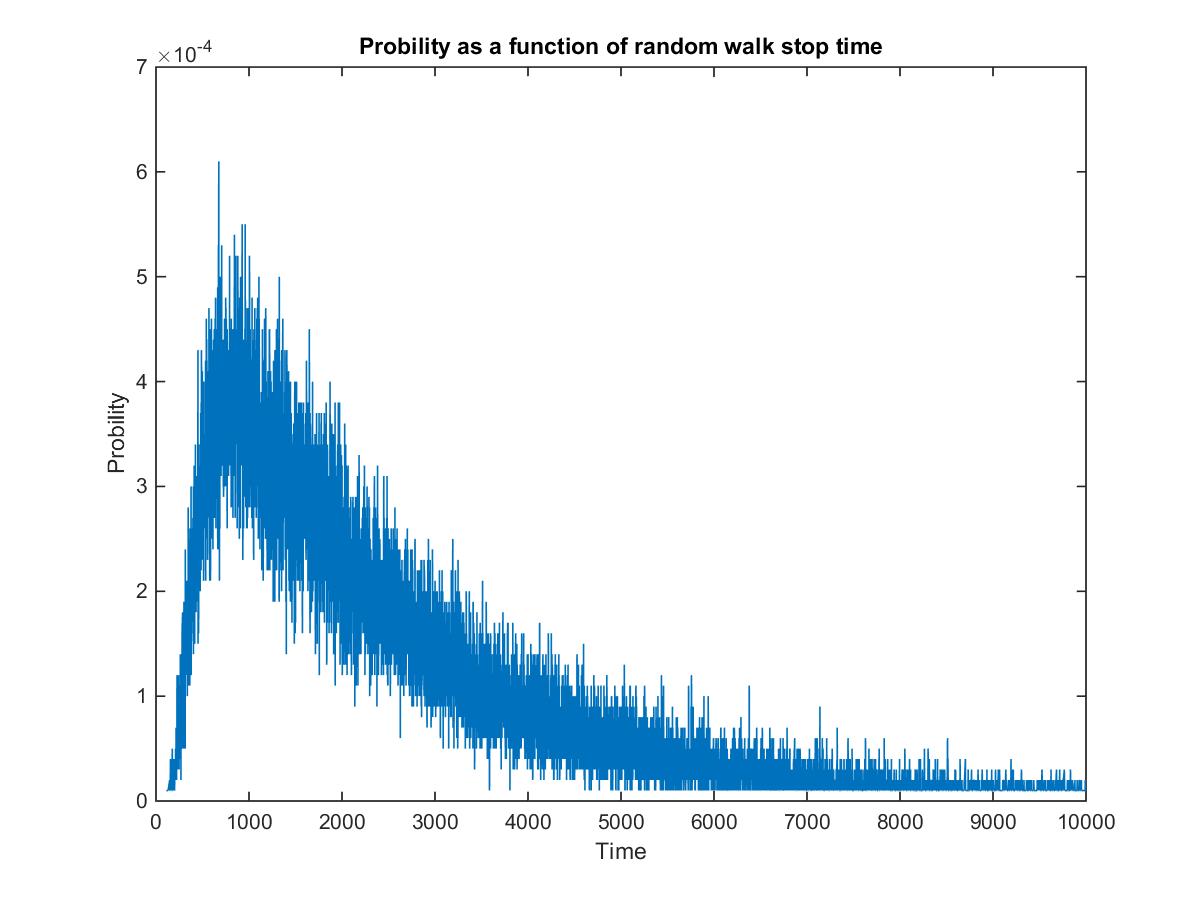
TOTAL\_RUNS = 1000



TOTAL\_RUNS = 10000



TOTAL\_RUNS = 100000

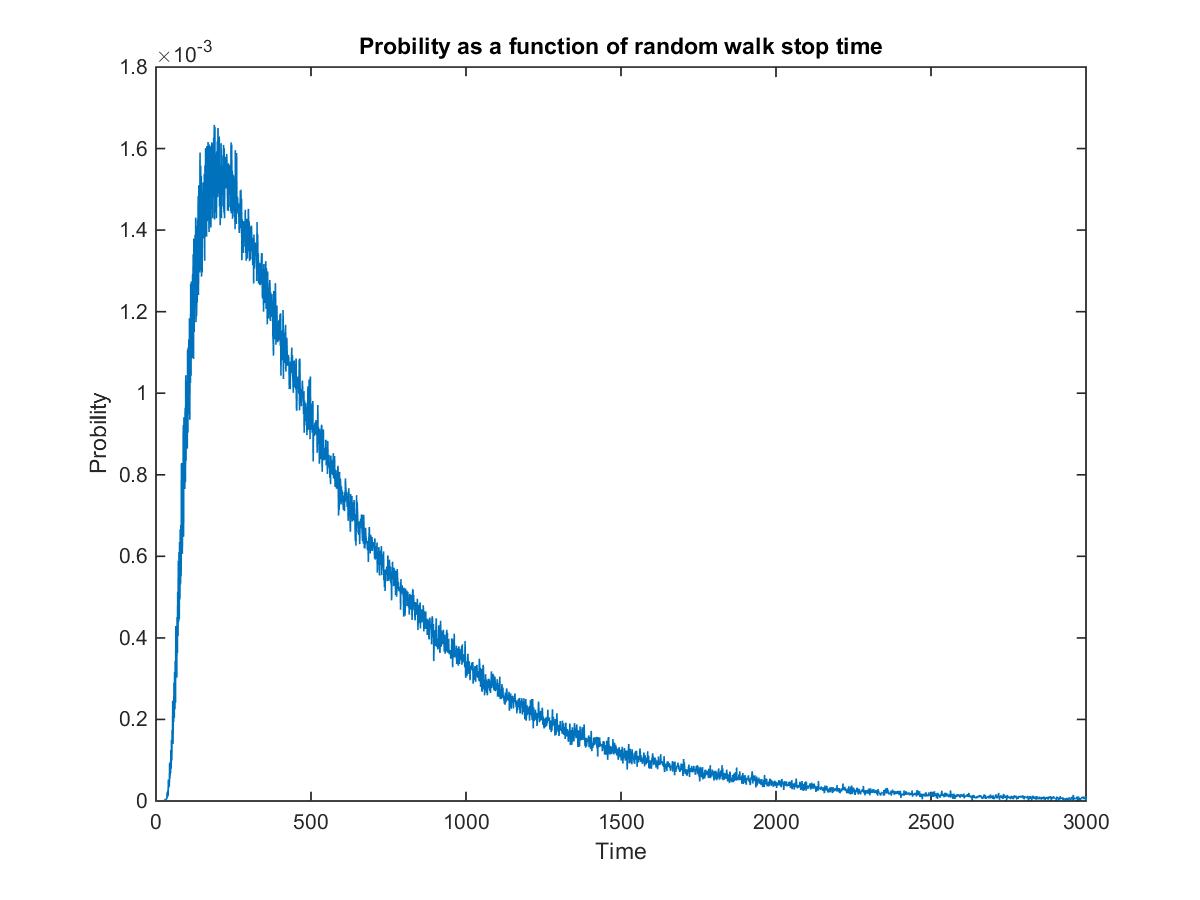


TOTAL\_RUNS = 1000000 as Figure1.1

1. What happens if increase the array size?

SIZE = 100 as figure 1.1

SIZE = 50, initial pos = 25

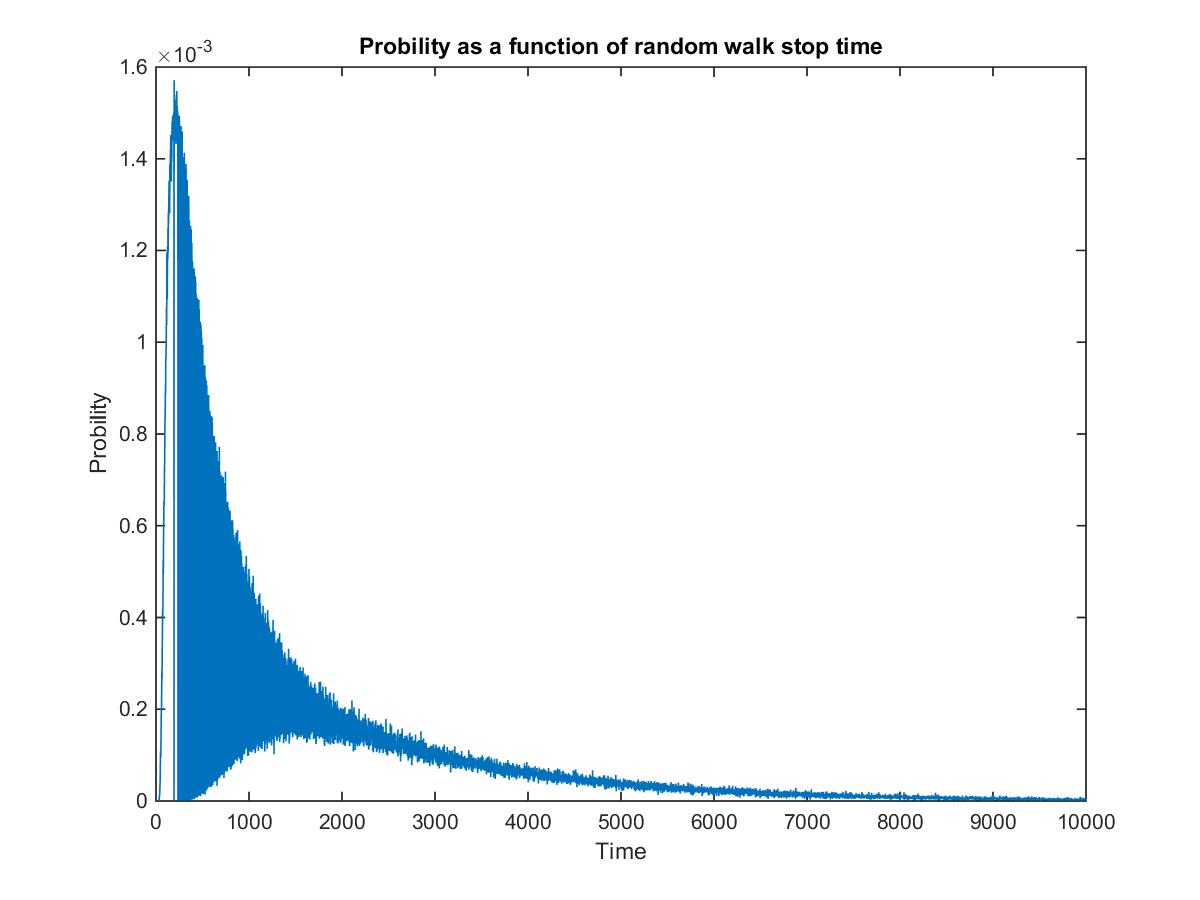


Conclude: The shape of the figures are almost the same with different peaks.c

1. What happens if change the start point? Initial pos (when SIZE = 100, TOTAL\_RUNS = 1000000)

pos = 50

initial pos = 25



Analysis: why the probability oscillate with time?

Consider t=73, 74, 75:

t=73, p(73) =

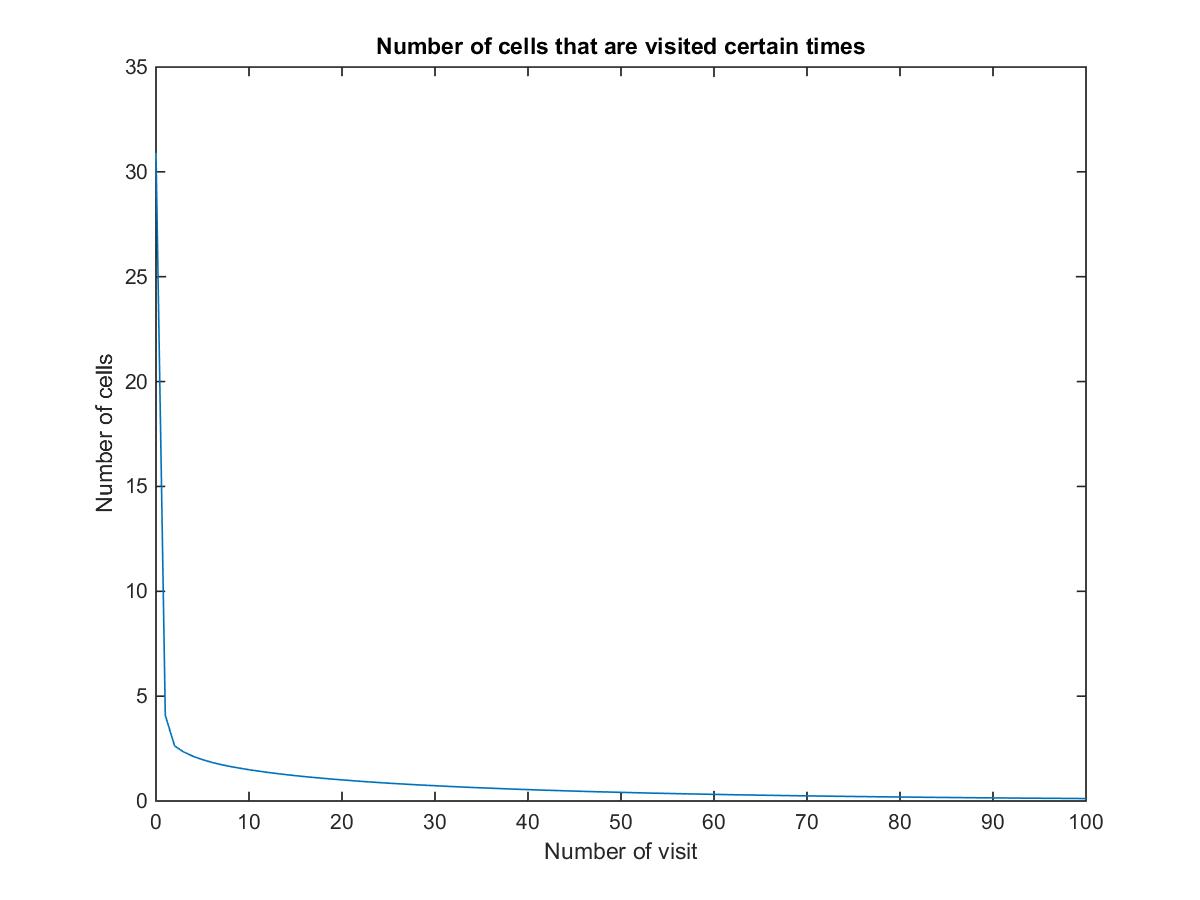
t=74, p(74) =

t=75, p(75) =

p(74)<<p(75) and p(74)<<p(73), so oscillate.

1. Compute how many cells are visited x time.

Figure 1.21: compute number of cells has been visited x time



x: number of visited

y: number of cells

Ways:

With TOTAL\_RUNS = 1000000:

for(index of cells from 0 to SIZE)

Count the number of times each cells has been visited before the walk ends.

For each specific number of times x, add on when a cell has the equal number of visited time.

Return

Conclude:

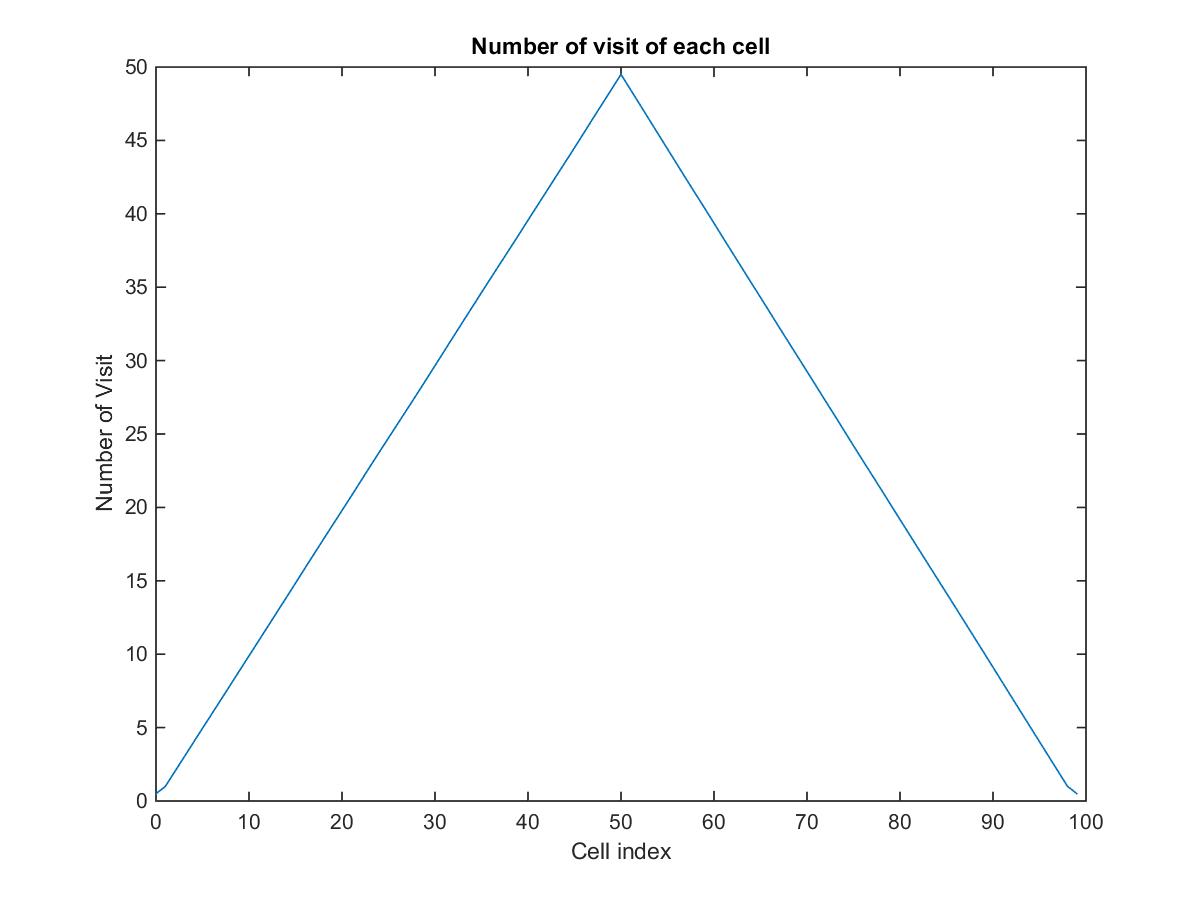
With x increase, the number of cells decreases from a relatively bigger number to about 3 very fast and then decrease very slow.

1. Compute average visited

Figure 1.22

x: number of array location (1 to 100)

y: number of times that cell was visited



Way:

In my TOTAO\_RUNS = 1000000 simulations, count the number of time each cells been visited.

Conclude:

The number of times of the 50th cell has been visited most frequently.

From the first cell to the middle cell, the number of visited increase linearly, and then decrease linearly from the middle cell to the last cell.

1. Figure 2.1

Way:

Randomly flip coins 1000000 times.

Make the position before the first flip at pos = 50, flip coins.

If the new pos = 0 or 100 (head or tail), terminate the program.

flip coin:

If the coin is head, then go forward to the right,

If the coin is tail, then go backward to the left.

t = count the number of times flip coins

Return the probability of t.

Conclude:

The probability as a function of a random walk terminate time in 2D arrays are increase from 0 very fast at the beginning, at , it reaches the biggest probability (10-3. And then decrease slower and slower. At about t = 3000, the probability

CS577\_Project2

1. Scenarios:
   1. Tape/Disk file read and write(random)
   2. Brower(time locality)
   3. Apple App Store(global hot topic)
2. Characteristics of different scenarios:
   1. Tape/Disk file read and write:

Need to physically wind tape between reels to read the specific piece of data, which make seeking time before reading and writing on a specific location of a tape/disk very long.

* 1. Browser: In a limit time period, the hot topic will be static, but as time goes, the hot topic will change very soon.
  2. Apple App Store: Time period of the hot topics is longer, which avoid the disadvantage of this method that will become too flexible when the access patterns change too rapidly.

1. Design data sets
   1. Tape/Disk file search (random)

ListSize= 1000 - 15000

The list contains distinct node with integers, in range [0, listSize-1], visit the list 1000000 times. Each time we randomly visit one node in the list.

* 1. Browser(several local hot topics)

ListSize= 1000 - 15000

The list contains distinct node with integers, in range [0, listSize-1], visit the list 1000000 times. Among every 200 visits, we visit the same node 100 times and the other 100 times among the others.

c

* 1. Apple App Store(one global hot topic)

ListSize= 1000 - 15000

The list contains distinct node with integers, in range [0, listSize-1], visit the list 1000000 times. Amon all visits, we visit the same node half of the times and the other randomly.

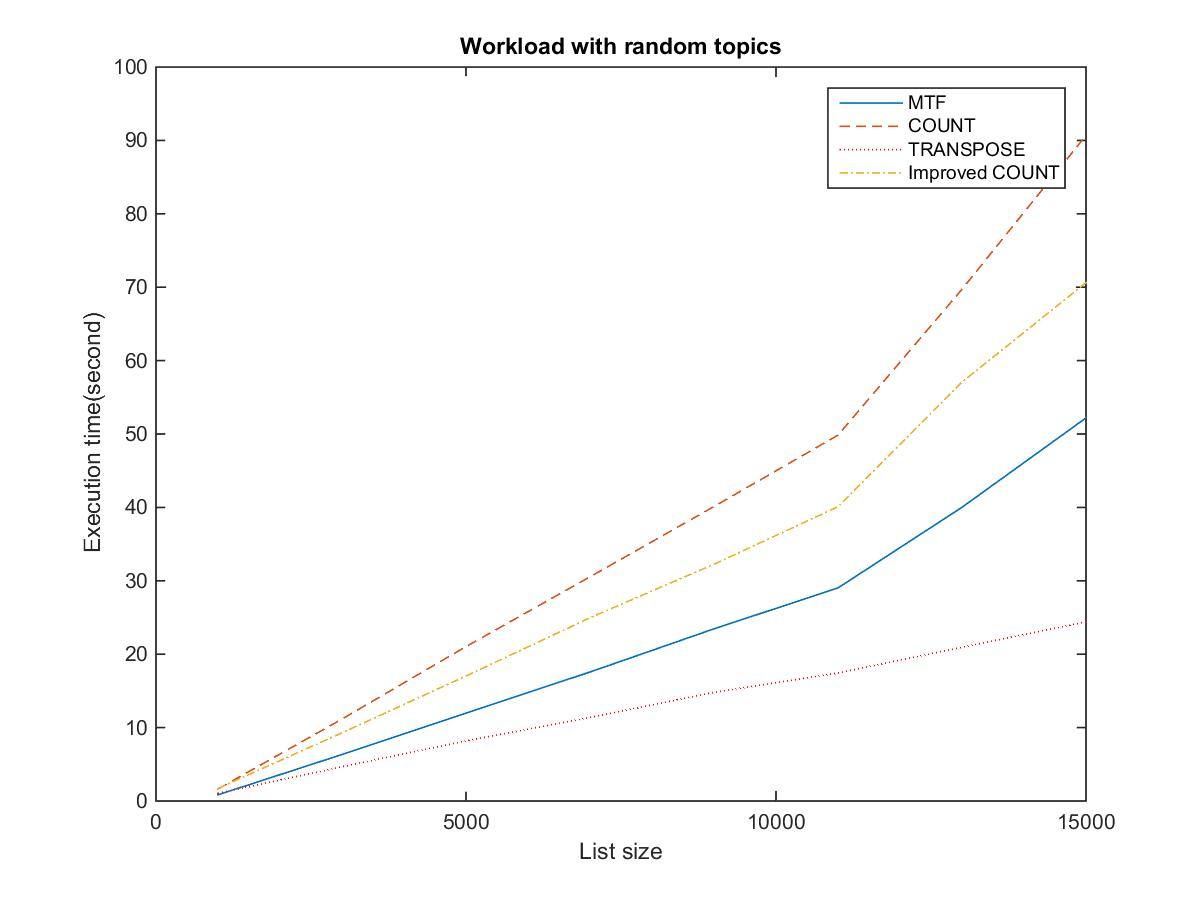
Part 2: comparing strategies for updating a self-organizing list

1. Our strategy:

We improve the count method, we don’t sort the list after every time we visit a node. Instead, we sort the list after 100-visit, and then clear the count for the next round.

2. Results:

1) Random topics:

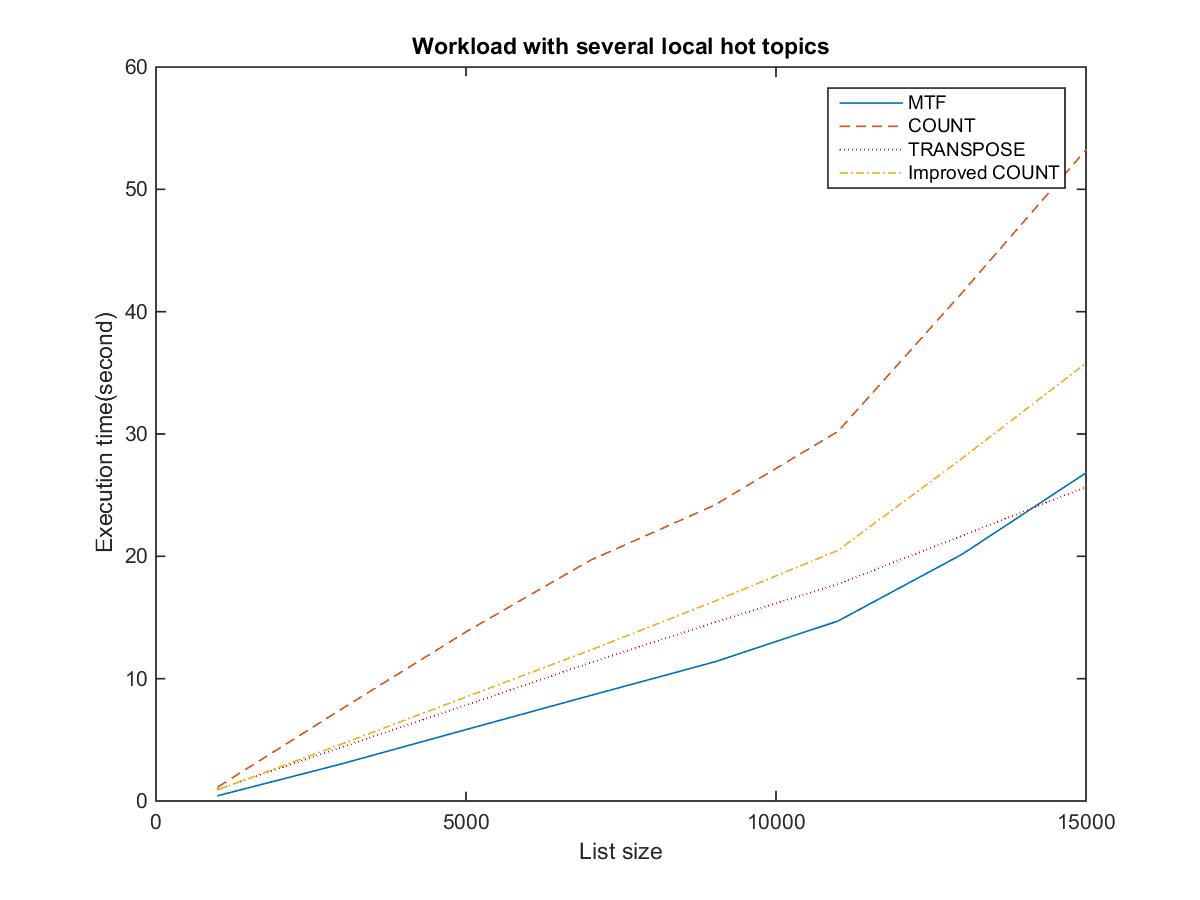


Worst: the count method, because we sort the list every time we visit a node.

Our improved count method is better because we avoid sort the list every time.

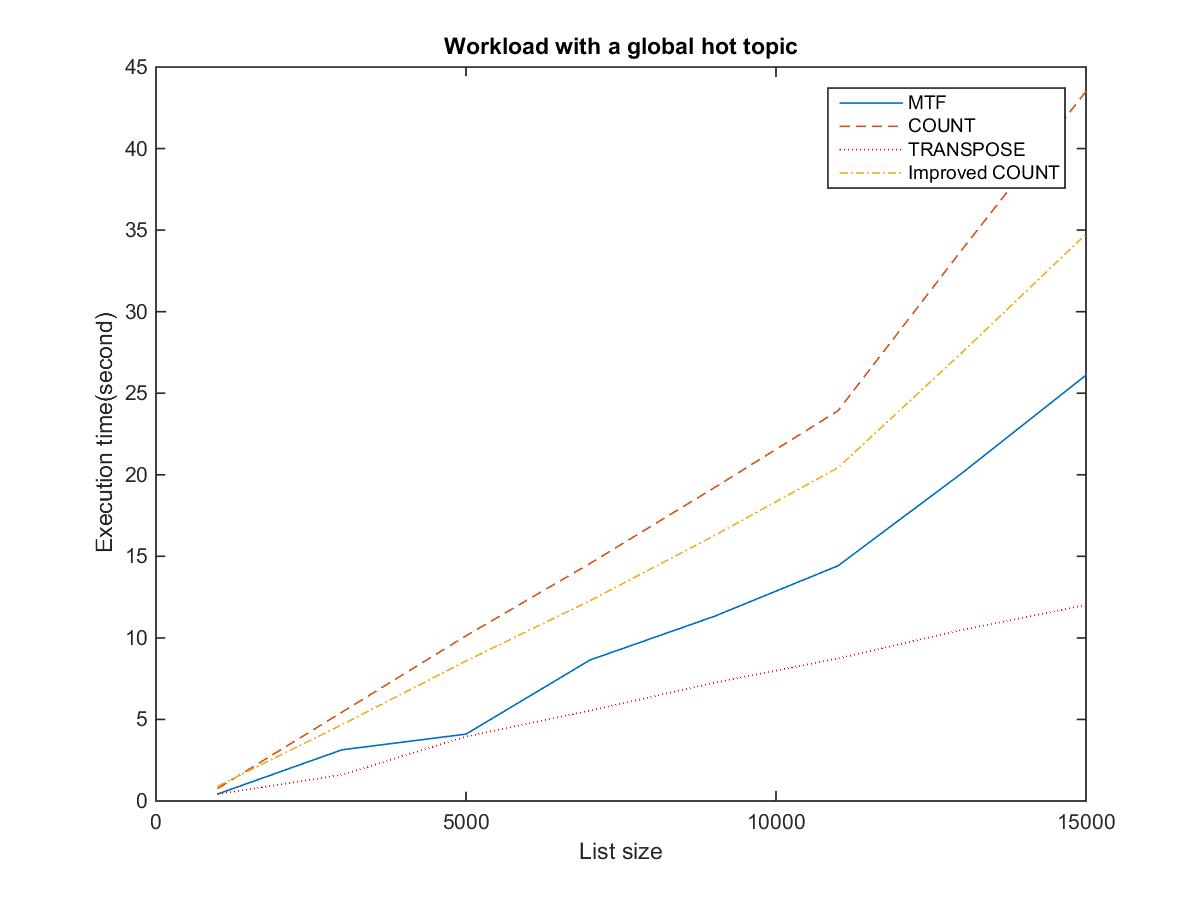
Transpose is better and MTF. Because in MTF, sometimes the nodes we move to the front are not a “hot topic”.

2) Several local hot topics:



Count method is the worst because it orders the list every time we insert a new node.

MTF is the best: in a limit period, we have some hot topic repeatedly visited, MTF moves them to the front, however, Transpose method just swap one pair. C.M

3) global hot topic 

Count method is the worst because it orders the list every time we insert a new node.

Transpose method vs. MTF: In MTF it’s not necessary to put every newly visited node to the front, because it may not be a hot topic. So, transpose is better than MTF.