1. Introduction

This assignment have been completed into the following enviroment and the system specification

1. GCC Compiler,
2. HP – 2000,
3. RAM – 2GB
4. Processor - Intel® Core™ i3-3110M CPU @ 2.40GHz × 4
5. Operating System (OS) – Ubuntu 16.04 (64 bit)

2. Hashing using Chaining Technique

Ans:

Hash Function

h(key) = key%M; M – prime number (should NOT be close to EVEN Number)

Hash Function used for experiment:

a) h(key) = key %1000003

b) h(key) = key % 1000033

Dataset	**Hash Function 1 (M = 1000003)** **Elapsed Time in Seach Operation**		**Hash Function 2 (M = 1000033)** **Elapsed Time in Seach Operation**		**Alpha = N/M**
	Succefull	**Unsuccessful**	**Successful**	**Unsuccessful**	
100000	2	1	3	2	0.1
500000	3	1	2	1	0.5
800000	2	1	2	1	0.8
990003	2	1	3	1	0.99

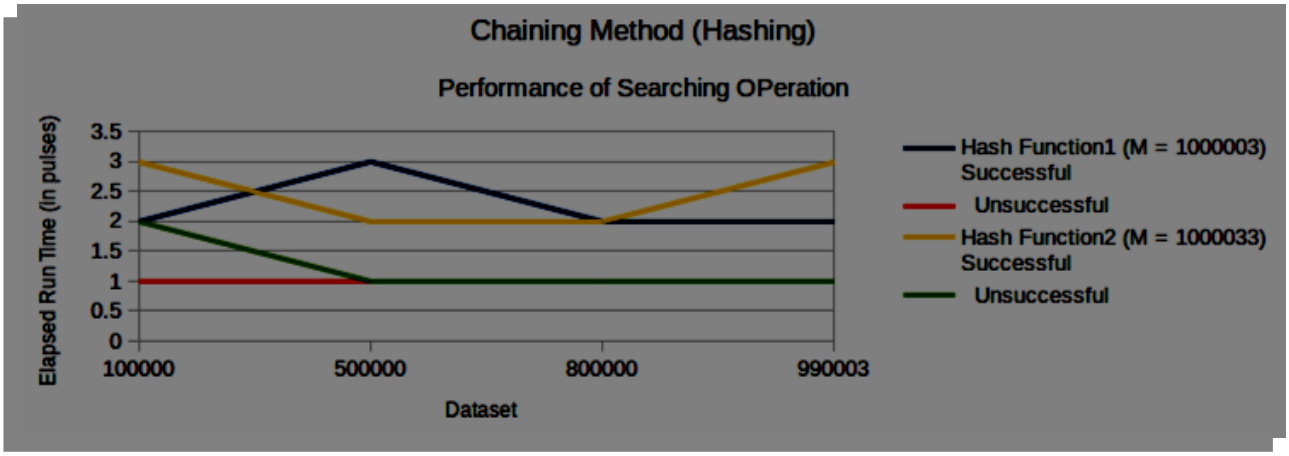

Figure:1. Performance of Search Operation in Chaining Method of Hashing

Observation:

1. From the experimet – it is clear that the seach operation takes a constant amout of time

2. From the theoretical point of view which is:

T(N) = Hash function calculation time + Alpha

- unsuccessful Search: T(N) = 1 + (Alpha) = 1 + (N/M)
- Successful Search : T(N) = 1 + (Alpha) = 1 + (N/2*M) – (N/2*M*N) = 1 + Alpha

Hence, it akes the constant searching time that is **'T(N) = 1 + ALPHA'** irrespective of the unsuccessful or successful seach.

3. Quadratic Method

Ans:

Hash Function
h(key) = key%M; M – prime number (should NOT be close to EVEN Number)

Hash Function used for experiment:
Frist Hash Funciton
a) h1(key) = key % 1000003, and h1(key) = (h1(key) + i*i)%1000003

Second Hash Function
b) h1(key) = key % 1000033, and h1(key) = (h1(key) + i*i)%1000033

Quadratic Hashing

Dataset	**Hash Function 1 (M = 1000003)** **Elapsed Time in Seach Operation**		**Hash Function 2 (M = 1000033)** **Elapsed Time in Seach Operation**		**Alpha = N/M**
	Succefull	**Unsuccessful**	**Successful**	**Unsuccessful**	
100000	3	1	3	1	0.1
500000	1	1	2	2	0.5
800000	2	1	2	1	0.8
990003	2	1	2	1	0.99

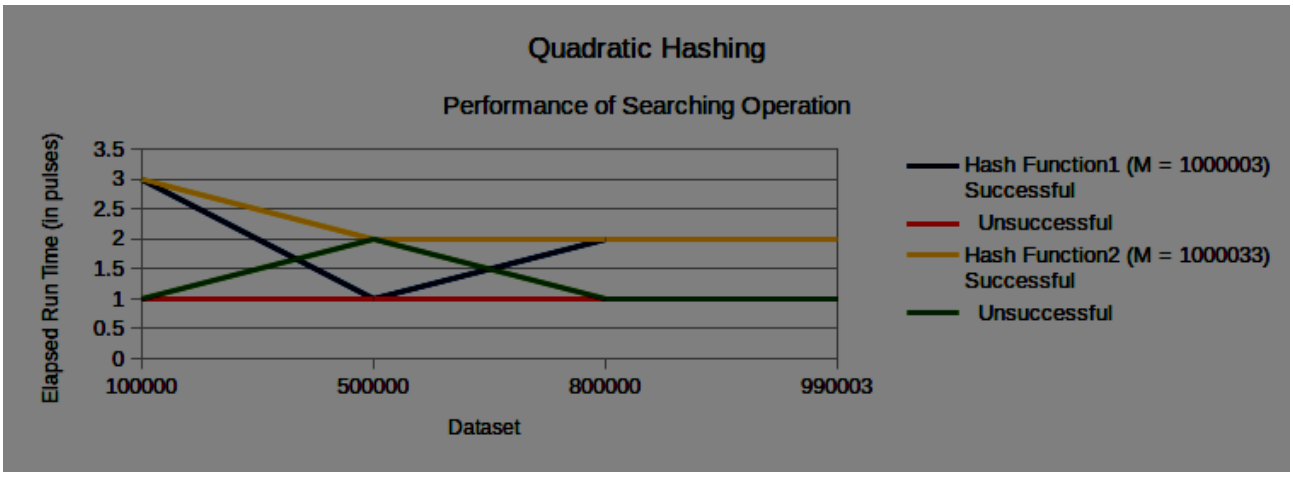

Figure:2. Performance of Search Operation in Quadratic Hashing

Observation:

1. From the experimet – it is clear that the seach operation takes a constant amout of time

2. From the theoretical point of view which is:
T(N) = Hash function calculation time + Alpha
- unsuccessful Search: T(N) = 1 + (Alpha) = 1 + (N/M)
- Successful Search : T(N) = 1 + (Alpha) = 1 + (N/2*M) – (N/2*M*N) = 1 + Alpha

Hence, it akes the constant searching time that is **'T(N) = 1 + ALPHA'** irrespective of the unsuccessful or successful seach.

4. Double Hashing Technique

Ans: Hash Function

h(key) = key%M; M – prime number (should NOT be close to EVEN Number)

Hash Function used for experiment:
a) h1(key) = key % 1000003, h2(key) = h1(key) = key % 1000001
for each iteration
h1(key) = (h1(key) + i*h2(key))%1000003

b) h1(key) = key % 1000033, h2(key) = h1(key) = key % 1000031
for each iteration
h1(key) = (h1(key) + i*h2(key))%10000033

Dataset	**Hash Function 1 (M = 1000003)** **Elapsed Time in Seach Operation**		**Hash Function 2 (M = 1000033)** **Elapsed Time in Seach Operation**		**Alpha = N/M**
	Succefull	**Unsuccessful**	**Successful**	**Unsuccessful**	
100000	5	1	4	1	0.1
500000	5	1	4	1	0.5
800000	6	1	4	1	0.8
990003	6	1	4	1	0.99

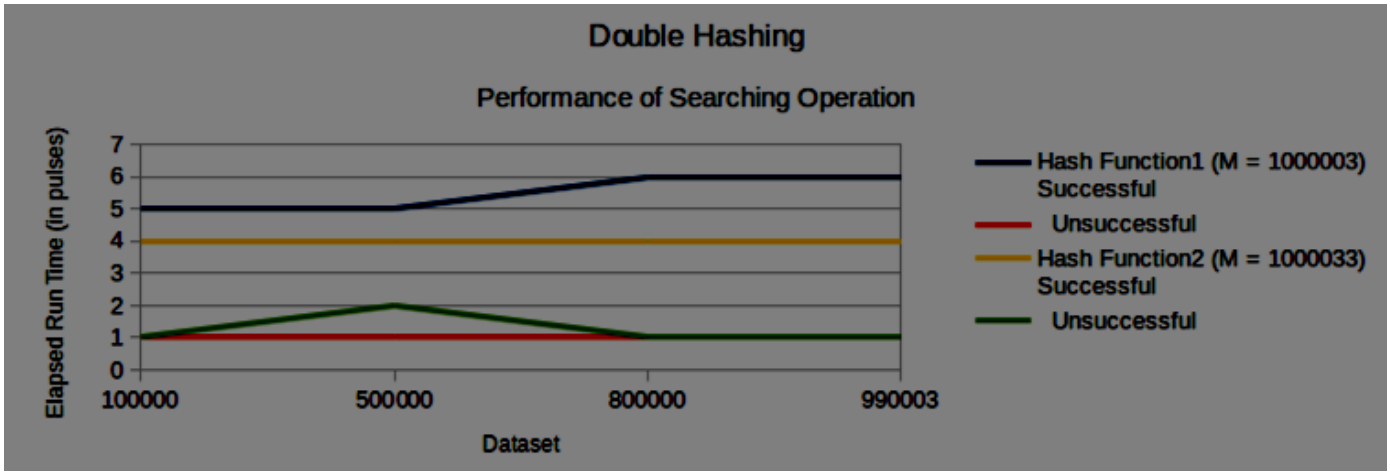

Figure:2. Performance of Search Operation in Double Hashing

Observation:

1. From the experimet – it is clear that the seach operation takes a constant amout of time

2. From the theoretical point of view which is:
T(N) = Hash function calculation time + Alpha
- unsuccessful Search: T(N) = 1 + (Alpha) = 1 + (N/M)
- Successful Search : T(N) = 1 + (Alpha) = 1 + (N/2*M) – (N/2*M*N) = 1 + Alpha

Hence, it akes the constant searching time that is **'T(N) = 1 + ALPHA'** irrespective of the unsuccessful or successful seach.

References:

1. Cormen, Thomas H., et al. *Introduction to algorithms, (30, 37)*. MIT press, 2010.
2. Horowitz, Ellis. "Sartaj Sahni.., Fundamentals of Computer Algorithms, (145, 154)." (1998).
3. https://en.wikipedia.org/wiki/chainhin_sort
4. https://en.wikipedia.org/wiki/double_sort
5. https://en.wikipedia.org/wiki/double_sort