Double hashing technique in closed hashing search process

Robbi Rahim$^{1,2,*,}$, Iskandar Zulkarnain$^3$ and Hendra Jaya$^4$

$^1$Postgraduate Ph.D. Student Universiti Malaysia Perlis, Universiti Malaysia Perlis, Malaysia
$^2$Department of Health Information, Akademi Perekam Medik dan Infokes Imelda, Jl. Bilal Ujung Medan 20116, Indonesia
$^3$Department of Computer System, STMIK Triguna Dharma, Indonesia
$^4$Department of Computer Engineering, STMIK Triguna Dharma, Jl. Jenderal Abdul Haris Nasution No.73, Medan 20219, Indonesia

*usurobbi85@zoho.com

Abstract. The search process is used in various activities performed both online and offline, many algorithms that can be used to perform the search process one of which is a hash search algorithm, search process with hash search algorithm used in this study using double hashing technique where the data will be formed into the table with same length and then search, the results of this study indicate that the search process with double hashing technique allows faster searching than the usual search techniques, this research allows to search the solution by dividing the value into the main table and overflow table so that the search process is expected faster than the data stacked in the form of one table and collision data could avoided.

1. Introduction

Searching is a job that we often do in everyday life. In a text editor, we often do a word search, or search for a word and replace it with another word, or search for a particular word and calculate the frequency of occurrence of the word in the document[1][2].

The search methods are various and each method has advantages and disadvantages, one of the search methods that has better place efficiency is Hash Search[3][4][5]. Hash Search uses a certain formula to perform the process of placement and search data, Hash Search is divided into 2 kinds, namely Closed Hashing and Open Hashing. In Closed Hash there is the possibility of more than one data having the same function result[4][5]. This results in collision occurring at the time of placement.

This study aims to show how the work of placement and search data with hash search algorithm with double hashing techniques and strategies to overcome the collision of data that may occur.

2. Theory

Hash Search method is almost similar to the method of direct search that is by using a certain formula both at the time of placement and search data[8]. Hash Search has better place space efficiency than direct search[9]. The function (I - 1) used by the direct search method has poor place use efficiency, so the Hash Search method improves it by using the modulo operation function [1][8]. Modulo operation function is often referred to as the Hash Function and the data shelter is called the Hash Table. The Hash function is not a one-to-one function such as the sharing function of the direct search method so it is possible that some data have the same function results [4][5]. This results in collisions occurring at the time of placement.
of data into tables so that a strategy is needed to overcome this collision. Strategies to deal with these collisions vary and each has their respective advantages and disadvantages. This strategy will be explained in the next section [6][7].

In general, Open Hashing means the use of table size is limited, but each index of the table utilizes a linked list to place data so that the amount of data is not limited [3], while Close Hash is a function with limited table size so that the amount of data that can be placed is also limited [3][7]. This Hash function causes the data stored in the Hash Table to have 2 types of address (address) that is:

a. Home Address, is the location obtained by using the Hash Function.
b. Real (Physical) address is the location where data is stored in table.

This double hashing strategy overcomes collisions by dividing the table into 2 pieces, the Main Table and the table overflow with the provisions of the size of the table as large[5]. The way of placement of data in the Main Table is done by using the Hash function. In the event of a collision, the data is placed on the overflow table. How to place data in the overflow table can be done in a sequential way or using the new Hash function. If the table is full, then the data entered into the overflow table will be rejected [3][4].

3. Proposed Method

Closed Hash data on the placement is done by using a hash function. Placement of data in Closed Hash is represented by using tables. Because the hash function allows multiple data to produce the same value, then the data placement process on Closed Hash can cause a collision. The problem can be solved by using the Double Hashing strategy.

Suppose the double hash table size is 10 then the hash function \( h(x) = x \mod 10 \) and the overflow function \( g(x) = x \mod 10 \). The table numbering starts from 0, for the numbers of the hashing tables in the contents as follows: 43,12,76,32,11,66,39,28,21,59,5,91,9,33, for the placement process for each table as follows:

![Figure 1. Arrangement of numbers before the hashing process](image)

a. Data '43'
   Main Table -> \( h(x) = 43 \mod 10 = 3 \), location 3 is empty so that '43' data is placed at location 3
b. Data '12'
   Main Table -> \( h(x) = 12 \mod 10 = 2 \), location 2 is empty so data '12' is placed on location 2
c. Data '76'
   Main Table -> \( h(x) = 76 \mod 10 = 6 \), location 6 is empty so that '76' data is placed at location 6
d. Data '32'
   Main Table -> \( h(x) = 32 \mod 10 = 2 \) -> collision occurs with data '12' the search continues to Overflow Table.
   Table Overflow -> \( g(x) = 32 \mod 10 = 2 \), location 2 is empty so '32' data is placed on location 2
e. Data '11'
Main Table \(\rightarrow h(x) = 11 \mod 10 = 1\), location 1 is empty so the data '11' is placed at location 1

f. Data '66'
Main Table \(\rightarrow h(x) = 66 \mod 10 = 6\) -> collision with data '76' the search continues to Overflow Table
Overflow Table \(\rightarrow g(x) = 66 \mod 10 = 6\), location 6 is empty so '66' data is placed on location 6

g. Data '39'
Main Table \(\rightarrow h(x) = 39 \mod 10 = 9\), location 9 is empty so '39' data is placed on location 9

h. Data '28'
Main Table \(\rightarrow h(x) = 28 \mod 10 = 8\), location 8 is empty so that '28' data is placed at location 8

i. Data '21'
Main Table \(\rightarrow h(x) = 21 \mod 10 = 1\) -> a collision with data '11' the search continues to Overflow Table.
Overflow Table \(\rightarrow g(x) = 21 \mod 10 = 1\), location 1 is empty so data '21' is placed at location 1

j. Data '59'
Main Table \(\rightarrow h(x) = 59 \mod 10 = 9\) -> collision with data '39' the search is continued into Overflow Table.
Overflow Table \(\rightarrow g(x) = 59 \mod 10 = 9\), location 9 is empty so '59' data is placed on location 9

k. Data '5'
Main Table \(\rightarrow h(x) = 5 \mod 10 = 5\), location 5 is empty so data '5' is placed at location 5

l. Data '91'
Main Table \(\rightarrow h(x) = 91 \mod 10 = 1\) -> collision with data '11' the search is continued to Overflow Table.
Overflow Table \(\rightarrow g(x) = 91 \mod 10 = 1\) -> collision with data '21'
\(G(x) = (1 + 1) \mod 10 = 2\) -> a collision occurs with data '32'
\(G(x) = (2 + 1) \mod 10 = 3\), location 3 is empty so the '91' data is placed at location 3

m. Data '9'
Main Table \(\rightarrow h(x) = 9 \mod 10 = 9\) -> collision with data '39' the search is continued to Overflow Table.
Overflow Table \(\rightarrow g(x) = 9 \mod 10 = 9\) -> collision with data '59'
\(G(x) = (9 + 1) \mod 10 = 0\), location 0 is empty so the data '9' is placed at location 0

n. Data '33'
Main Table \(\rightarrow h(x) = 33 \mod 10 = 3\) -> collision with data '43' the search is continued to Overflow Table.
Overflow Table \(\rightarrow g(x) = 33 \mod 10 = 3\) -> collision with data '91'
\(G(x) = (3 + 1) \mod 10 = 4\), location 4 is empty so the '33' data is placed at location 4

The following is the order of the numbers in the table hashing:
After determining the position of the numbers in the main table and the overflow table, the next test is to perform the following search process:

a. Search Data '91' using Double Hash technique

Main Table Size = 10, hash Function = h (x) mod 10
Overflow Table Size = 10, hash Function = g (x) mod 10

Figure 2. (a) Main Table and (b) Overflow Table

After determining the position of the numbers in the main table and the overflow table, the next test is to perform the following search process:

Main Table Size = 10, hash Function = h (x) mod 10
Overflow Table Size = 10, hash Function = g (x) mod 10

b. Main Table -> h (x) = 91 mod 10 = 1 -> Data '91' no search found continued to Overflow Table

Overflow Table ->
G (x) = 91 mod 10 = 1 -> Data '91' not found.
G (x) = (1 + 1) mod 10 = 2 -> Data '91' not found.

In the search process the number 91 obtained the number is on the overflow table and hashing process stops after the data is found.
4. Conclusion
In the process of searching by double hashing technique can be concluded that the value will be divided into two tables to place the value in the main table and overflow, and for the search process is done first once in the main table and continued on the table overflow until all the numbers listed in the table are processed.

References

[1] N. Arora, G. Bhasin and N. Sharma, 2014, "Two way Linear Search Algorithm," International Journal of Computer Applications, vol. 107, no. 21, pp. 6-8.

[2] D. Kumar and S. Manish , 2014, "BINARY SEARCH IS FASTER THAN THE LINEAR," INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN TECHNOLOGY, vol. 1, no. 5, pp. 796-799.

[3] D. Jia, W. Xie and Y. Jiang, 2012, "Research of database query technology based on hash function," in IEEE 8th International Conference on Information Science and Digital Content Technology (ICIDT), Jeju, South Korea.

[4] E. Lodi and F. Luccio, 1985, "SPLIT SEQUENCE HASH SEARCH," Information Processing Letters, vol. 20, pp. 131-136.

[5] J. N, V. V, K. M and K. A, 2012, "Frequent Itemset Generation using Double Hashing Technique," in International Conference on Modelling Optimization and Computing.

[6] K. K. Pandey and N. Pradhan, 2014, "A Comparison and Selection on Basic Type of Searching Algorithm in Data Structure," International Journal of Computer Science and Mobile Computing, vol. 3, no. 7, pp. 751-758.

[7] Y. M. K. Omar, H. Osama and A. Badr, 2017, "Double Hashing Sort Algorithm," Computing in Science & Engineering, vol. 19, no. 2, pp. 63-69.

[8] L. J. Guibas and E. SZEMEREDI, 1978, "The Analysis of Double Hashing," JOURNAL OF COMPUTER AND SYSTEM SCIENCES, vol. 16, pp. 226-274.

[9] A. Mehta, A. Saxena, J. Patel and A. Thanna, 2015, "A REVIEW ON COMPARISION OF BINARY SEARCH AND LINEAR SEARCH," International Journal Of Engineering Sciences & Management Research, vol. 2, no. 10, pp. 85-89.