Assignment 3

COMP2711 Computer Programming 2

Tim Norton  
FAN NORT0144

# Question 1

Being able to quickly find a value in a dataset is a frequent requirement of a computer program, and the efficiency of a program to do this is calculated in terms of how many times the program needs to:

* compare two values to determine which is larger
* swap two values to obtain a specific order
* copy a value from one location to another

An example of a dataset could be 9 randomly selected cards from a pack that are individually numbered of 1-13.

Example Dataset: 2 - 11 - 5 - 1 - 13 - 4 - 9 - 12 - 7

Two selection methods are **Quick Select** and **counting select**, which can locate a specific value as described below.

## Quick Select to Find Median value

To explain the Quick Select algorithm, think of the list as individual playing cards in line on the table. You have 3 position markers coloured red, green and blue. We will call the blue market the pivot.

The target card is that at the median (middle value of the pile). If the pile has an odd number of cards, the median will be the exact middle but for an even number, it is the higher card. In this case there are 9 cards, so the median is the 5th smallest value (4 above and 4 below).

**Step 1**

Place the blue and red markers on the right- hand card.  
Place the green marker on the left-hand card.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [11] | [5] | [1] | [13] | [4] | [9] | [12] | [7] |
| o |  |  |  |  |  |  |  | o o |

**Step 2**

Compare the card at the green marker with the card at blue. If green is lower, move the marker one position right. Keep doing this until you reach either of the following points:

* If the green is greater or equal to blue go to step 3
* If the red and green are on the same card go to step 5

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [11] | [5] | [1] | [13] | [4] | [9] | [12] | [7] |
| o |  |  |  |  |  |  |  | o o |
|  | o |  |  |  |  |  |  | o o |

**Step 3**

Leave green in that position and move red one position left.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [11] | [5] | [1] | [13] | [4] | [9] | [12] | [7] |
|  | o |  |  |  |  |  | o | o |

If the red card is greater than the blue card, move red one position to the left. Keep doing this until you reach either of the following points:

Keep doing this until you reach a point where red is less than blue or red and green are on the same card.

* If the red is less than or equal to blue go to step 4
* If the red and green are on the same card go to step 5

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [11] | [5] | [1] | [13] | [4] | [9] | [12] | [7] |
|  | o |  |  |  |  | o |  | o |
|  | o |  |  |  | o |  |  | o |

**Step 4**

Swap the card at green with the card at red. Do not move the markers.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [13] | [11] | [9] | [12] | [7] |
|  | o |  |  |  | o |  |  | o |

Repeat steps 2 & 3 until the green and red pointers are together, then go to step 5.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [13] | [11] | [9] | [12] | [7] |
|  |  | o |  |  | o |  |  | o |
|  |  |  | o |  | o |  |  | o |
|  |  |  |  | o | o |  |  | o |
|  |  |  |  | o o |  |  |  | o |

**Step 5**

Now swap the card at the blue marker with the one at the red/green marker.   
NOTE: If green, red and blue are on the same card, you have found the target card and have finished.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [7] | [11] | [9] | [12] | [13] |
|  |  |  |  | o o |  |  |  | o |

The cards are now sorted into two subsets such that everything above the pivot (blue) is higher and below is lower. Go to step 6.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [7] | [11] | [9] | [12] | [13] |
| Low Subset | | | | Pivot | High Subset | | | |

**Step 6**

The position of the pivot is compared with the target.

If the pivot is higher, working with only with the lower subset, repeat steps 1 - 5.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [7] | [11] | [9] | [12] | [13] |
| o |  |  | o o |  |  |  |  |  |

If the pivot is lower, working with only the higher subset, repeat steps 1 – 5.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [7] | [11] | [9] | [12] | [13] |
|  |  |  |  |  | o |  |  | o o |

If the target is equal to the pivot, all markers should be on the same card and you have found the target card, which is the closest to the median value.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [2] | [4] | [5] | [1] | [7] | [11] | [9] | [12] | [13] |
|  |  |  |  | o o o |  |  |  |  |

In this case, the pivot is 5 and we are looking for the 5th smallest card so we have located it and the card value is [7] which is the next highest card closest to the true median of 6.5.

### Example 2

This example requires more steps to locate the median. With 10 items, the target is at position 6

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | [2] | [11] | [5] | [1] | [7] | [13] | [9] | [12] | [16] | [4] |
| Step 1 | o |  |  |  |  |  |  |  |  | o o |
| Step 2 |  | o |  |  |  |  |  |  | o | o |
| Step 3 |  | o |  |  |  |  |  | o |  | o |
| Step 3 |  | o |  |  |  |  | o |  |  | o |
| Step 3 |  | o |  |  |  | o |  |  |  | o |
| Step 3 |  | o |  |  | o |  |  |  |  | o |
| Step 3 |  | o |  | o |  |  |  |  |  | o |
| Step 4 | [2] | [1] | [5] | [11] | [7] | [13] | [9] | [12] | [16] | [4] |
| Step 2 |  |  | o | o |  |  |  |  |  | o |
| Step 3 |  |  | o o |  |  |  |  |  |  | o |
| Step 5 | [2] | [1] | [4] | [11] | [7] | [13] | [9] | [12] | [16] | [5] |
| Step 1 |  |  |  | o |  |  |  |  |  | o o |
| Step 3 |  |  |  | o |  |  |  |  | o | o |
| Step 3 |  |  |  | o |  |  |  | o |  | o |
| Step 3 |  |  |  | o |  |  | o |  |  | o |
| Step 3 |  |  |  | o |  | o |  |  |  | o |
| Step 3 |  |  |  | o | o |  |  |  |  | o |
| Step 3 |  |  |  | o o |  |  |  |  |  | o |
| Step 5 | [2] | [1] | [4] | [5] | [7] | [13] | [9] | [12] | [16] | [11] |
| Step 1 |  |  |  |  | o |  |  |  |  | o o |
| Step 2 |  |  |  |  |  | o |  |  |  | o o |
| Step 3 |  |  |  |  |  | o |  |  | o | o |
| Step 3 |  |  |  |  |  | o |  | o |  | o |
| Step 3 |  |  |  |  |  | o | o |  |  | o |
| Step 4 | [2] | [1] | [4] | [5] | [7] | [9] | [13] | [12] | [16] | [11] |
| Step 2 |  |  |  |  |  |  | o o |  |  | o |
| Step 5 | [2] | [1] | [4] | [5] | [7] | [9] | [11] | [12] | [16] | [13] |
| Step 1 | o |  |  |  |  | o o |  |  |  |  |
| Step 2 |  | o |  |  |  | o o |  |  |  |  |
| Step 2 |  |  | o |  |  | o o |  |  |  |  |
| Step 2 |  |  |  | o |  | o o |  |  |  |  |
| Step 2 |  |  |  |  | o | o o |  |  |  |  |
| Step 5 |  |  |  |  |  | o o o |  |  |  |  |

The target at location 6 is [9], which is the next highest card closest to the true median of 8.

## Counting Select to Find Median value

Using the same dataset as above and the concept of cards laid out in a line on the table, the median is found with the Counting Select algorithm as follows.

**Step 1**

Count blue markers equal to the median of the count of cards (9 cards so median is 5).

**Step 2**

Create a list of numbers from 1 to the maximum value of cards you can have. (13 in this case but the list could go up to an infinitely high value).

For every position in the list, if you have a card equal to that value, put a green marker. If a number is duplicated, put an additional marker for each duplicate.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Median | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| o o o o o | o | o |  | o | o |  | o |  | o |  | o | o | o |

**Step 3**

Place a red marker at position 1 and move up the list one position at a time. For every green marker you find, take 1 marker away from the blue pile until there are no more blue markers.

The position will be the median of the values. In this case [7]

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Median | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
| o o o o o | o | o |  | o | o |  | o |  | o |  | o | o | o |
| o o o o | o o | o |  | o | o |  | o |  | o |  | o | o | o |
| o o o | o | o o |  | o | o |  | o |  | o |  | o | o | o |
| o o o | o | o | o | o | o |  | o |  | o |  | o | o | o |
| o o | o | o |  | o o | o |  | o |  | o |  | o | o | o |
| o | o | o |  | o | o o |  | o |  | o |  | o | o | o |
| o | o | o |  | o | o | o | o |  | o |  | o | o | o |
|  | o | o |  | o | o |  | o o |  | o |  | o | o | o |

# Question **2**

To determine the most efficient median finding algorithm, the following needs to be considered.

* The number of items in the list
* The difference of the range of the upper and lower bound values
* If the upper and lower bound values are known
* Memory Usage

A Quick Select algorithm has an average time complexity (how long it will take to run) of n log n, where n is the number of items in the list. In a worst case this becomes n2. This is expressed as O(n log n) or O(n2).

With 20,000 items the average time complexity with be 20000 \* Log(20000), which is 86,020. The worst case would be 200002, which is 400,000,000.

For a Counting Select algorithm, the average time complexity is O(1.5n+k) where n is the number of items in the list and k is the range from the upper to lower bound values. In a worst case this becomes O(2n+k).   
Where the upper and lower are not known, you need to find these values by scanning through all the items in the list. The will change the Counting Select average time complexity to O(2.5n+k).

With 20,000 items and a range from 0 to 40,000 the average time complexity with be  
1.5 \* 20000, which is 70,000. The worst case would be 2 \* 2000 + 40000, which is 80,000.

Figure 1 shows a comparison of Quick Select against Counting Select. From this, as an unofficial guideline, if the formula n > 2k/(Log(k)) is true, then Counting Select becomes a better option than Quick Select.

Example:

100,000 items with a range for 1 – 1,000,000

2\*1000000/(Log(1000000)) = 333,333

As n is 100,000, it is less than 333,333 so Quick Select will be a better option

Example:

50,000 items with a range of 1 – 100,000

2\*100000/Log(100000) = 40,000

As n is 50,000, it is greater than 40,000 so Counting Select would be a good choice.

Figure

If memory use is an issue, Quick Select is the better option as the algorithm does not need to occupy more memory that that used by the list itself. Where as Counting Select requires making a second list to the length of the upper bound value, which may be in the millions. As an example, a possible range of 1 million will occupy around 7MB of memory during the search, which is not significant in most computer systems. However 100 Million will require around 700MB, which may cause issues.