Assignment 1

Student Num. 929395

April 7, 2018

Task 2 - Complexity

The complexity of my program seems to be O(1). The main algorithm can be reduced to:

```
for each read:
    for each kmer:
        if kmer in index:
            for each position of kmer in reference:
                hamming(read, reference)
    repeat for reverse kmers
```

The 'for each read' part of the algorithm is constant as, for this measure of complexity, a constant number of reads is being used. Additionally, the 'for each kmer' part will also be constant as the read length is basically constant (100 or 101). Therefore, the only part of the algorithm that depends on the length of the reference is 'for each position of the kmer in the reference'. This means that the only aspect of the program that depends on the length of the reference is the number of positions in the reference that any particular kmer can be found at. Furthermore, this is unlikely to increase dramatically with the size of the reference, especially at high values of k. In fact, by analyzing the index of the different length of references with k=13 it can be seen that the majority of kmers can only be found at one position in the reference, regardless of reference length. Therefore, most of the time it is likely that only one position in the reference will have to be investigated for each kmer. As a result of this it can be said that this step is also O(1).
The run times for the various reference lengths can be seen in Figure 1. From this diagram the complexity seems to be closer to O(logn) than O(1). This may be because there is an initial increase in the number of positions in

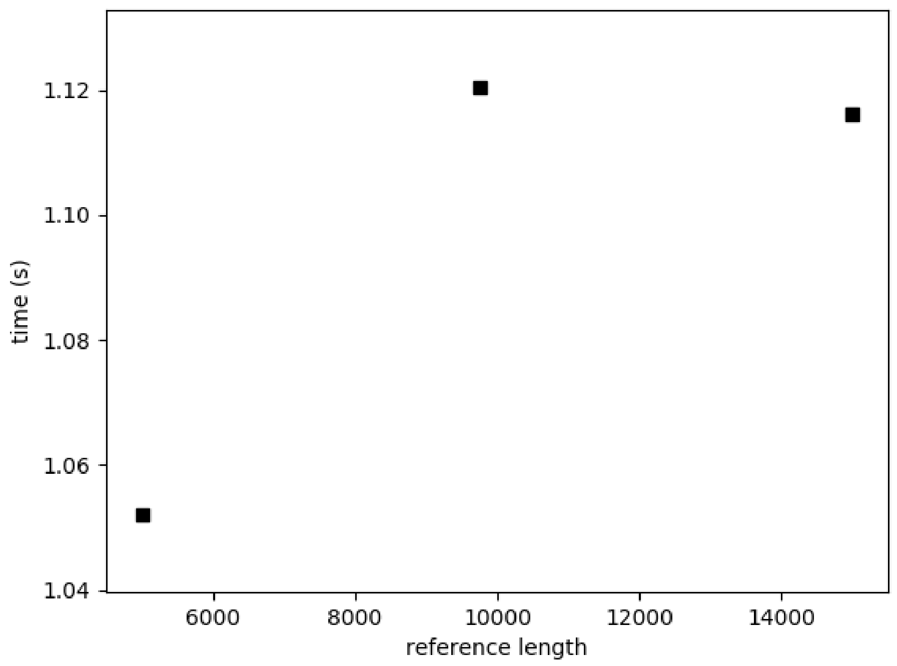

Figure 1: Run time of program with index

the reference each kmer maps to as the reference length jumps from 5000 to 10000, which then levels off. However, as values are so small (1.052-1.1205) it may just be due to variance between the references. For example, one reference may have lots of repeats. This would result in one kmer mapping to many positions in the reference, meaning that many positions will have to be investigated. The fact that the third reference actually runs faster than the second seems to back this up.
It is evident from Figure 2 that the program which utilizes the index performs much better than the original program. The original program has O(n) complexity its run time increases linearly with n (reference length).

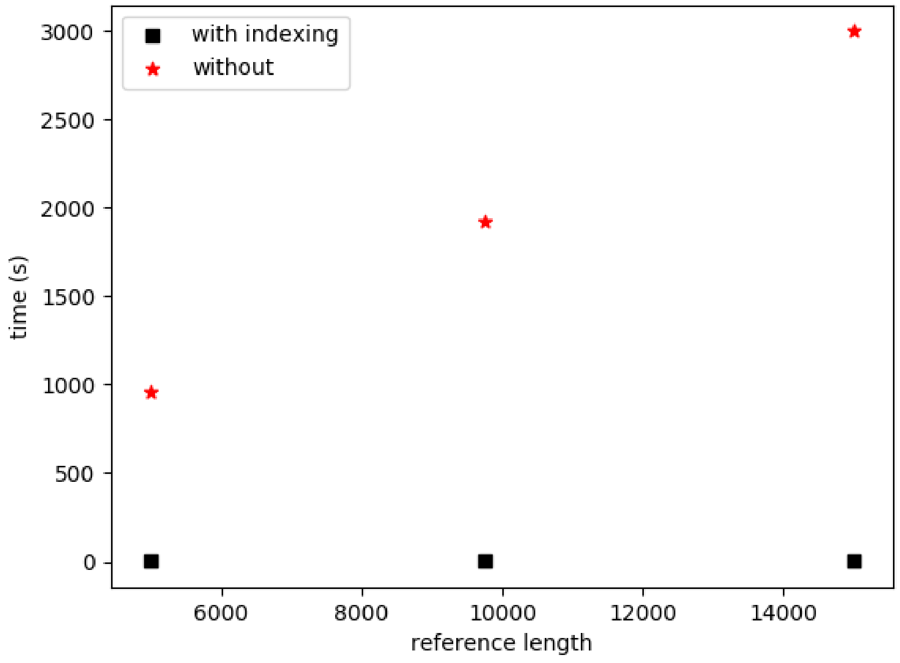

Figure 2: Comparison of both programs

Task 3 - Comparison of Quality Scores

For all the bases of all reads the average quality score decreases towards the end of the read. At the beginning of the read the median changes from around 31 to a high of about 40. The median then decreases slowly to about 28 at the end of the read. Additionally, the spread of the quality scores increases significantly towards the end of the read. In the first half of the read the range is quite small with only some outliers falling much below the median. However, towards the end of the read the range is much larger and spreads from 0 to just above the median.
On average the quality scores of base mismatches are lower than the normal bases. Furthermore, in the first portion of the read the spread of quality scores is much higher. The median varies between 15 and 40 for this section. As would be expected, the quality scores decrease towards the right. However, these scores are even lower than the normal bases in this region. There are only a few outliers with high or average quality values in this area of the read.
One possible reason why the quality scores are lower for the mismatched bases is that these bases are likely the result of a sequencing error. However, there are some bases with high quality reads. For example within the first portion of the read there are approximately 4 bases of quality 40 or above. These are quite high scores even for this portion of the read. Therefore it is likely that these are legitimate differences between the read and the reference and not the result of a sequencing error. The same can be said for the outliers in the last portion of the graph. There are quite a few bases with quality scores of 30 or above which is quite high for that area of the read. Therefore it is highly likely these are actual differences in the read.
The distribution of quality scores of mismatched bases compared to all bases is useful information for detecting the presence of SNPs(Single Nucleotide Polymorphisms). A high quality score of a mismatched base would suggest that that base is not an error and is in fact an SNP. Equally, a low quality score would suggest that the base mismatch is the result of a sequencing error.

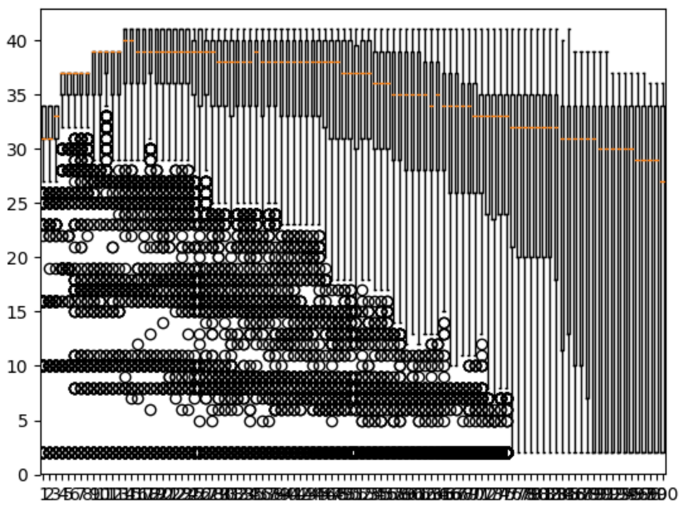
Figure 3: Quality Scores of All Reads

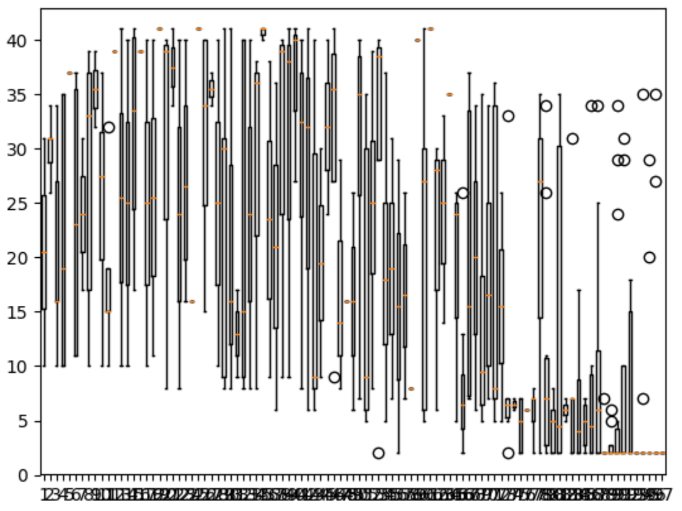
Figure 4: Quality Scores of Mismatched Bases