CS170–Fall 2014 — Writeup to Homework 9

Ted Xiao, SID 24452530, `cs170-vg`

November 6, 2014

DNA Shotgun Sequencing

Main idea. We use a greedy algorithm to implement shotgun sequencing. First, we use a substring sieve to remove any strings in the input that are a substring of another string in the input. Then, we sort the input strings by descending length in order to prepare for our iterations. We maintain a cumulative superstring of string's we've already merged in previous iterations. Then, at each iteration, we take the string with the highest overlap with our current superstring, and merge that string with our superstring, and remove the string from the list of remaining strings to merge. We stop when we have no more strings to merge, and return the superstring.

Pseudocode.

```
DNA(strings[] input):
    fragments = []
    input.sortByDescendingLength()
    for i in input:
        if i not substring of any element in fragments:
            fragments.append(i)
    while len(fragments) > 1:
        ovdict = {}
        for j in range(1, len(fragments):
            ovdict[j] = overlap(fragments[0], fragments[j])
        maxOVindex = ovdict.getKey(maxValue)
        fragments[0] = makeShortestSuperstring(fragments[0], fragments[maxOVindex])
        fragments.remove(fragments[maxOVindex])
    return fragments[0]
```

Running time.
$\Theta(n^2k)$

Justification of running time.
The substring sieve, the first operation we do, will be in $\Theta(n^2k)$ because worst case we'd add in one string from the original list to the new list each iteration, and this would take however many 'in' operations as there are elements already in the new list. Thus, there will be $1+2+...+n-1 = O(n^2)$ 'in' operations, which each take $\Theta(k)$ time on average. Even though the overlap function will run in $O(k^2)$ worst case, it's expected runtime is $\Theta(k)$ because we won't have many worst-case scenarios, as many strings will be close to constant runtime.So the substring sieve will be in $\Theta(n^2k)$ running time.

Next, we examine the main loop. The main loop will iterate through the length of the list, which is n iterations. The loop inside that constructs the dictionary will loop through the items in the list and do work $O(k)$, the overlap function. Thus, this portion of the main loop paired with the inner loop will be have $1 + 2 + ... + n - 1 = O(n^2)$ operations of 'overlap', which will be a total of $\Theta(n^2k)$ running time. Additionally, inside the main loop, the other operations are finding the maximum value in a dictionary, constructing a shortest common superstring, and removing an item from a list. However, the dictionary and list operations are linear or constant, and constructing the shortest common superstring of two strings is in $O(k)$ time, but that's less than $O(nk)$. Thus the total running time of this section is dominated by the main loop paired with the inner loop: $\Theta(n^2k)$. Our substring sieve also took this long, so our overall running time is $\Theta(n^2k)$.