Homework 3

Elliott Pryor

24 October 2021

Problem 2

By hand, build an FM-index for S = gacgaacgac$. You may report the BW[], C[] and occ[,] data structures as simple tables.

For our reference, we give the suffix array so it is easier for us to build the FM-index by hand. We list all the suffixes:

gacgaacgac$, acgaacgac$, cgaacgac$, gaacgac$, aacgac$, acgac$, cgac$, gac$, ac$, c$, $

i	SA	Suffix	BW[i]
1	11	$	c
2	5	aacgac$	g
3	9	ac$	g
4	2	acgaacgac$	g
5	6	acgac$	a
6	10	c$	a
7	3	cgaacgac$	a
8	7	cgac$	a
9	4	gaacgac$	c
10	8	gac$	c
11	1	gacgaacgac$	$

Table 1: The last column is the BW[] for the FM-Index

Letter	Count
$	0
a	1
c	5
g	8

Table 2: C[] for the FM-Index

i	$	a	c	g
1	0	0	1	0
2	0	0	1	1
3	0	0	1	2
4	0	0	1	3
5	0	1	1	3
6	0	2	1	3
7	0	3	1	3
8	0	4	1	3
9	0	4	2	3
1	0	4	3	3
1	1	4	3	3

Table 3: occ table for the FM-Index

Problem 3 Trace the backwards search algorithm to determine if the pattern Q = gac belongs to S = gacgaacgac$.

Iteration 1: st = 1, ed = 11, i = 3
x = c, st' = C[c] + occ(c, 0) + 1 = 6, ed' = C[c] + occ(c, 11) = 8

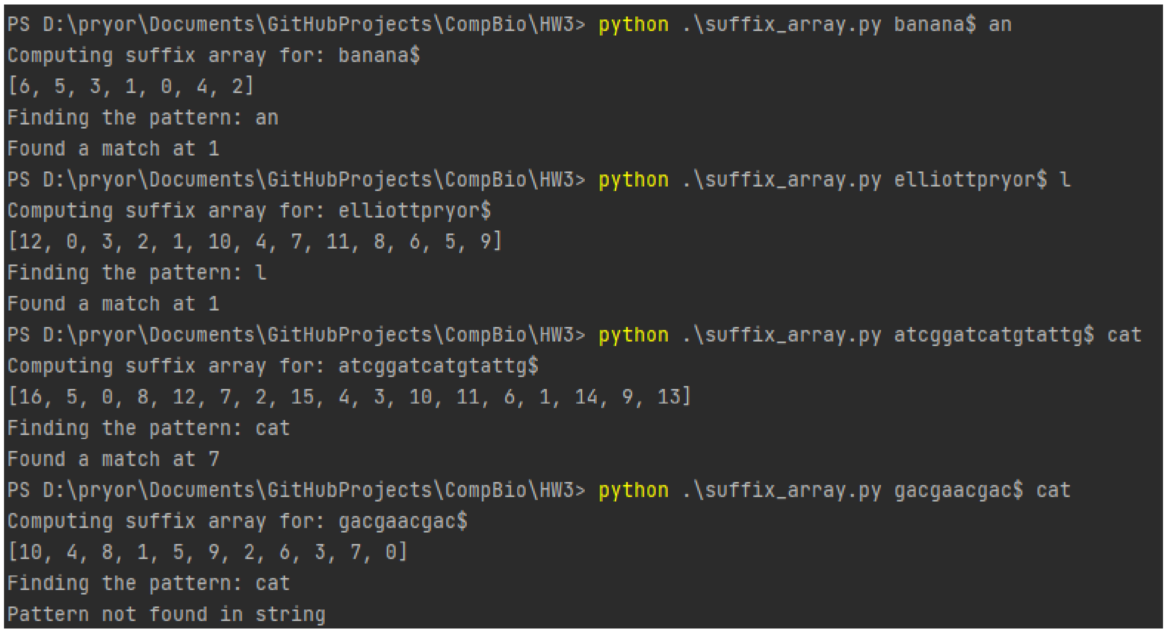
Iteration 2: st = 6, ed = 8, i = 2
x = a, st' = C[a] + occ(a, 5) + 1= 3, ed' = C[a] + occ(a, 8) = 5

Iteration 3: st = 3, ed = 5, i = 1
x = g st' = C[g] + occ(g, 2) + 1 = 10, ed' = C[g] + occ(g, 5) = 11

Iteration 4: Output 2 matches at 10, 11

Problem 1 Write a program to compute the suffix array for a given input string. You can either prompt the user for a string, or use a command-line argument to specify the string. Demonstrate it works on a few strings.

Then, use your program to find the suffix array for the string S = gacgaacgac$.



```
PS D:\pryor\Documents\GitHubProjects\CompBio\HW3> python .\suffix_array.py banana$ an
Computing suffix array for: banana$
[6, 5, 3, 1, 0, 4, 2]
Finding the pattern: an
Found a match at 1
PS D:\pryor\Documents\GitHubProjects\CompBio\HW3> python .\suffix_array.py elliottpryor$ l
Computing suffix array for: elliottpryor$
[12, 0, 3, 2, 1, 10, 4, 7, 11, 8, 6, 5, 9]
Finding the pattern: l
Found a match at 1
PS D:\pryor\Documents\GitHubProjects\CompBio\HW3> python .\suffix_array.py atcggatcatgtattg$ cat
Computing suffix array for: atcggatcatgtattg$
[16, 5, 0, 8, 12, 7, 2, 15, 4, 3, 10, 11, 6, 1, 14, 9, 13]
Finding the pattern: cat
Found a match at 7
PS D:\pryor\Documents\GitHubProjects\CompBio\HW3> python .\suffix_array.py gacgaacgac$ cat
Computing suffix array for: gacgaacgac$
[10, 4, 8, 1, 5, 9, 2, 6, 3, 7, 0]
Finding the pattern: cat
Pattern not found in string
```

Figure 1: Examples showing that my program works. I also implemented the bonus and a query string is provided for all examples. Also note the last example is the one that we were asked to provide. Please note that I use 0-indexing.

```python
import sys


def suffix_array(S):
    suffix = [(i, S[i:]) for i in range(len(S))]  # generate list of tuples
    suffix.sort(key=lambda tup: tup[1])  # sort by suffixes
    return [fix[0] for fix in suffix]


def pattern_search(pattern, S):
    suffix = suffix_array(S)
    l, r = 0, len(S)-1
    found = False
    while l <= r:
        m = int((l + r) / 2)
        st = S[suffix[m]:]  # string version of the suffix
        if pattern == st[:len(pattern)]:
            print(f'Found a match at {suffix[m]}')
            found = True
            break
        elif pattern > st:
            l = m + 1
        else:
            r = m - 1

    if not found:
        print("Pattern not found in string")


if __name__ == '__main__':
    pattern = None

    if len(sys.argv) == 2:
        input_str = sys.argv[1]
    elif len(sys.argv) == 3:
        input_str = sys.argv[1]
        pattern = sys.argv[2]
    else:
        input_str = input("Input the string to compute a suffix array for: ")

    if pattern is None:
        pattern = input("Input the pattern to search for: ")

    print(f"Computing suffix array for: {input_str}")
    print(suffix_array(input_str))
    print(f"Finding the pattern: {pattern}")
    pattern_search(pattern, input_str)
```