Project One
Algorithms

Jennifer Felton
jfelton@csu.fullerton.edu

Telegraph Style String Algorithm

telegraph_string(s):

String outgoing = s
outgoing.ToUpperCase
Replace all punctuation with “.”
Remove all symbols from outgoing
Replace all multiple spaces with single spaces from outgoing
Remove all “\n\t\r” from outgoing
If (length < 5):
append “STOP.”
Else:
Substring length -5 to length “STOP.”
Return outgoing

Mathematical analyses

Dip Search:

```
dip_search(V):
    last_dip = None
    for i from 0 through n-3:
        if V[i] == V[i+2] and V[i+1] < V[i]:
            last_dip = i
    return last_dip
```

O(1) + O(n) + O(1) +O(1) + O(1) = O(n)
The time complexity is O(n)

Longest Balanced Span:

```
longest_balanced_span(V):
    best = None
    for s from 0 through n:
        for e from s+1 through n:
            if the sum of elements in V[s] up to but not including V[e] is zero:
                if best is None or [s,e) contains more elements than best:
                    best = [s, e)
    return best
```

O(1) + O(n) * O(n) +O(1) + O(1) + O(1) + O(1)= O(n^2)
The time complexity is O(n)

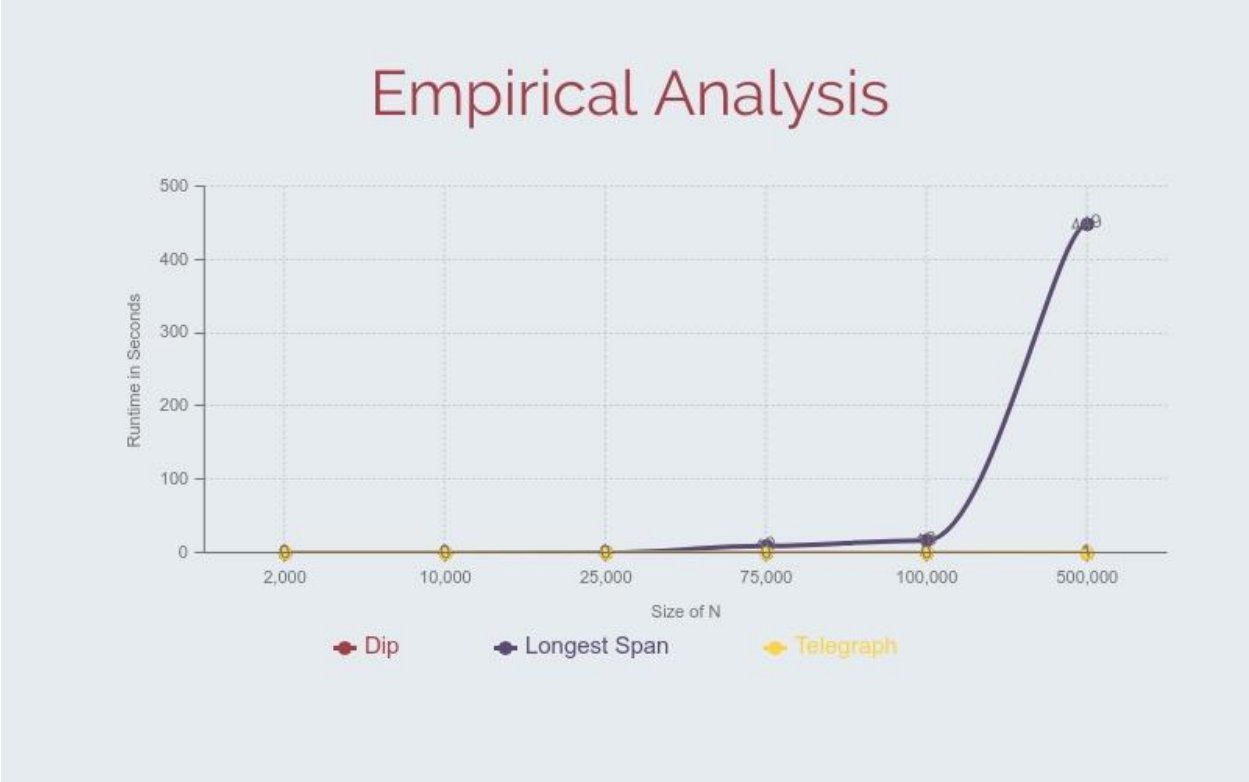
Telegraph Style String:

```
telegraph_string(s):
    String outgoing = s
    outgoing.ToUpperCase
    Replace all punctuation with "."
    Remove all symbols from outgoing
    Replace all multiple spaces with single spaces from outgoing
    Remove all "\n\t\r" from outgoing
    If (length < 5):
     append "STOP."
    Else:
    Substring length -5 to length "STOP."
    Return outgoing
```

O(1) + O(n) + O(n)+ O(n)+ O(n) +O(n)+ O(1) = O(n)
The time complexity is O(n)

Scatter Plot

A comparison of the running time of the three algorithms



Questions:

1) According to my analysis, the efficiency classes of the algorithms are as follows:

    Dip Search: O(n) by the trivial theorem
    Longest Balanced Span: O(n^2) by the trivial theorem
    Telegram Style: O(n) by the trivial theorem

2) Between the dip search and the longest balanced span algorithms, the longest balance span has the longest run times. I was trying to test an n of 1,000,000 and it was taking over five minutes. I wasn't sure if my VM was running out of space, or if it was just taking an extremely long time. The longest balance is exponentially slower than the dip search.
3) The fit lines on my scatter plot are consistent with what I predicted by my math analysis. If you look at the lines, the longest balance span algorithm grows exponentially. The other two have a linear growth pattern.
4) All of the evidence is consistent with the hypothesis on the front page. The runtimes all correlate to the scatterplots and the mathematical efficiency of the algorithms.