Assignment 2

The following algorithm is implemented-

```
PARTIALDIGEST(L)
      width ← Maximum element in L
      DELETE(width, L)
      X ← {0, width}
      PLACE(L, X)

PLACE(L, X)
     if L is empty
          output X
          return
     y ← Maximum element in L
     if Δ(y,X) ⊆ L
          Add y to X and remove lengths Δ(y,X) from L
          PLACE(L, X)
          Remove y from X and add lengths Δ(y,X) to L
     if Δ(width – y,X) ⊆ L
          Add width – y to X and remove lengths Δ(width –
          y,X) from L
          PLACE(L, X)
          Remove width – y from X and add lengths Δ(width
          – y,X) to L
          return
```

When Log t vs Cut Points graph is generated for this algorithm it shows a linear behavior. The algorithm shows a sharp increase in the run time once the number of cut points increase ~30.

Here is a reading from the restriction mapping pdf Page 91 line 2 -

It was not clear for a number of years whether or not this algorithm is polynomial in the worst case—sometimes both alternatives are viable. If both "left" and "right" alternatives hold and if this continues to happen in future steps of the algorithm, then the performance of the algorithm starts growing as 2k where k is the number of such"ambiguous" steps.

Thus when the algorithm chooses either left or right alternative then algorithm reduces the size of the problem by one i.e. $T(n) = T(n-1) + O(n)$ and the run time is quadratic. But when both the left and right answers are correct then $T(n)=2T(n-1) + O(n)$ making

algorithm exponential. Therefore this algorithm has a mixed complexity exponential and polynomial.

Following is the data and graph for the algorithm-

Cut points	Log t	time(ms)
5	1.5849625	3
10	2.32192809	5
15	4.45943162	22
30	5.72792045	53
50	10.731319	1700
75	14.3852558	21399
100	15.6335913	50837