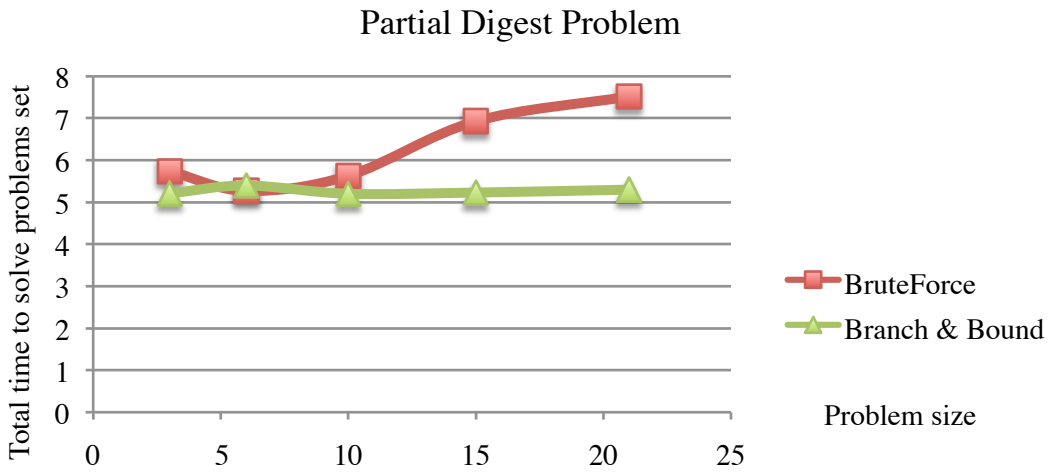
Assignment 5

Laleh Rostami Hosoori A01772483

Program #1:

C- Measure the run time of each algorithm (in A, and B) solving that set of problems for each problem size and plot a log/linear graph of "problem size" (x axis linear) vs. "total time to solve problems set" (y axis log). Both these algorithms will be slow! Problem sizes will be small.

log/linear Plot



D- What is the run time as a function of problem size? Use (f(n) to express the function complexity where n is the problem size)

ANOTHERBRUTEFORCEPDP:

This algorithm examines $\binom{|L|}{n-2}$ different sets of integers, but $|L| = \frac{n(n-1)}{2}$, so AnotherBruteForcePDP takes roughly $O\left(n^{2n-4}\right)$ time.

PARTIALDIGEST:

Let T(n) be the maximum time the algorithm takes to find the solution for an *n*-point instance of the PDP. If there's only one viable alternative

$$T(n) = T(n-1) + O(n)$$

O(n) is the time spend adjusting the sets X and L. This case is quadratic.

$$T(n) = T(n-1) + O(n)$$
$$T(n) = T(n-1) + T(n-2) + O(n-1) + O(n)$$
$$\vdots$$
$$T(n) = 1 + O(1) + \cdots + O(n-2) + O(n-1) + O(n)$$
$$T(n) = O(n^2)$$

But when there are two alternatives:

$$T(n) = 2T(n-1) + O(n)$$

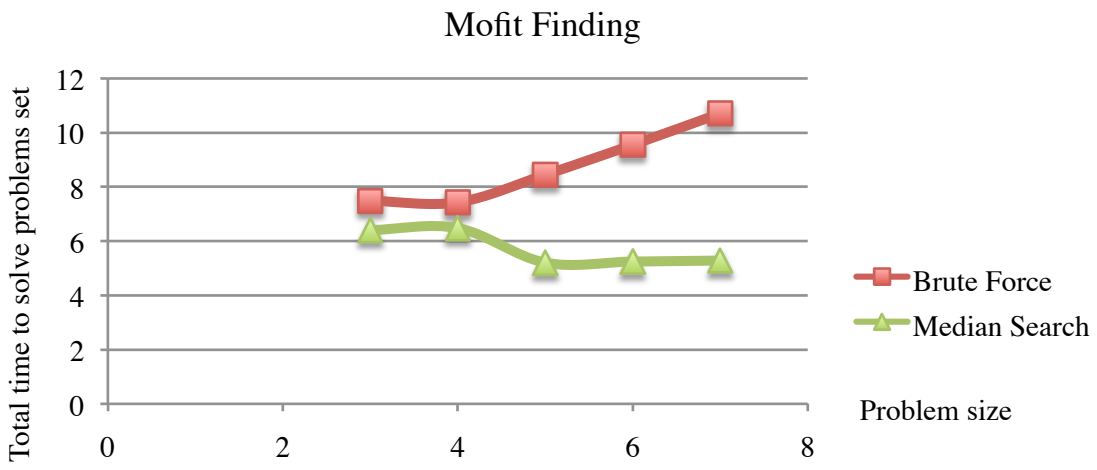
The runtime will be exponential.

$$T(n) = 2T(n-1) + O(n)$$
$$T(n) = 2(2T(n-2) + O(n-1)) + O(n)$$
$$T(n) = 2^2 T(n-2) + 2O(n-1) + O(n)$$
$$\vdots$$
$$T(n) = 2^n T(0) + 2^{n-1} O(1) + \cdots + 2O(n-1) + O(n)$$
$$T(n) = O(2^n)$$

Program#2:

C- Measure the run time of each algorithm (in A, and B) solving that set of problems for each problem size and plot a log/linear graph of "problem size" (x axis linear) vs. "total time to solve problems set" (y axis log). Both these algorithms will be slow! Problem sizes will be small.

log/linear Plot



D- What is the run time as a function of problem size? Use (f(n) to express the function complexity where n is the problem size)

BRUTEFORCEMOFITSEARCH:

The number of sequences is $(n-l+1)^t$ which is exponential in *t*. For each sequence, the algorithm calculates the score, which requires *O(l)* operations. So the overall complexity of the algorithm is evaluated as $O(l\,n^t)$

BRUTEFORCEMEDIANSEARCH:

It considers each of 4^l strings of length *l* and computes TotalDistance at every step which spends *O(nt)*. Therefore, it has running time $O(4^l nt)$.

Output - Exhaustive Algorithms (run)

```
run:
Brute Force: 0 1 3
Partial Digest: 0 1 3
Brute Force: 0 1 4 6
Partial Digest: 0 1 4 6
Partial Digest: 0 10
Partial Digest: 0 15
Partial Digest: 0 21

Partial Digest
Problem Sizes: 3        6       10      15      21
Run Times BF:
5.570542939881897       5.093421685162235       5.550228353055094       6.9018940319738205      7.464280845176612
Run Times BB:
5.02530586526477        5.1003705451175625      5.086359830674748       6.029789470831855       6.144885418287142
Mofit Finding -> Best Mofit: taa
Median search -> Best Word: aaa
Mofit Finding -> Best Mofit: caa
Median search -> Best Word: aaa
Mofit Finding -> Best Mofit: aca
Median search -> Best Word: aaa
Mofit Finding -> Best Mofit: caa
Median search -> Best Word: aaa
Mofit Finding -> Best Mofit: aaa
Median search -> Best Word: aaa

Mofit Finding
Problem Sizes:
3       4       5       6       7
Run Times Mofit Search:
7.283549972002684       7.631585804726977       8.40713586219156        9.565693545925948       10.722060069223364
Run Times Median Search:
6.378942698613438       6.487138375477186       5.2455126678141495      5.285557309007774       5.320146286111054       BUILD SUCCESSFUL (total time: 57
```