Tommy Huynh
- tommyh@csu.fullerton.edu
Juhen Mo
- henrymo@csu.fullerton.edu
CPSC 335
March 12, 2020

Project 2

End to beginning:

Pseudocode:

```
Set n to size of vector n
Populate the array H with n size and 0 values
For i = n - 2 to 0 do
        For j = i + 1 to n do
                If A[i] >= A[j] and H[i] <= H[j] do
                        H[i] = H[j] + 1
Calculate in max the length of the longest non-increasing subsequence
Allocate space for the subsequence R
Index = max - 1
J = 0
For i = 0 to n do
        if  H[i] == index do
                R[j] = A[i]
                Decrement index
                Increment j
Return
```

Time complexity:

1 t.u.

1 t.u.

$$\sum_{i=n-2}^{0}\sum_{j=i+1}^{n} 3 + max(2,0) = \sum_{i=n-2}^{0}\sum_{j=i+1}^{n} 5 = \sum_{u=n-2}^{0} 5(n-(i+1)+1) = \sum_{i=n-2}^{0} 5(n-i) = -\sum_{i=0}^{-n+2} 5n + \sum_{i=0}^{-n+2} 5i$$

$$= -5n(-n+2-0-1) + 5(\tfrac{(-n+2)(-n+2+0)}{2})$$

$$= -5n(-n+1) + \tfrac{5}{2}(n^2-4n+4)$$

$$= 5n^2 - 5n + \tfrac{5}{2}n^2 - 10n + 10$$

$$= 7\tfrac{1}{2}n^2 - 15n + 10 \; t.u.$$

1 t.u.

1 t.u.

2 t.u.

1 t.u.

$$\tfrac{n-0+1}{1} + 1 = n + 1 + 1 = n + 2 \; t.u.$$

$$1 + max(5,0) = 6 \; t.u.$$

1 t.u.

$$s.c. = 1 + 1 + 7\tfrac{1}{2}n^2 - 15n + 10 + 1 + 1 + 2 + 1 + n + 2 + 6 + 1$$

$$= 7\tfrac{1}{2}n^2 - 14n + 26$$

$$= O(n^2)$$

Powerset:

Pseudocode:

Set n to size of vector A
Populate the array stack with n+1 size and 0 values
Set k equal to 0
While true
    If stack[k] < n do
        Stack[k+1] = stack[k] + 1
        Increment k
    Else
        Increment stack[k-1]
        Decrement k
    If k = 0
        Break the loop
    For i = 1 to k do
        Push A[stack[i] - 1] to candidate vector
    If best is empty or candidate is nonincreasing and size of candidate is bigger than size of best do
        Set best to candidate
Return

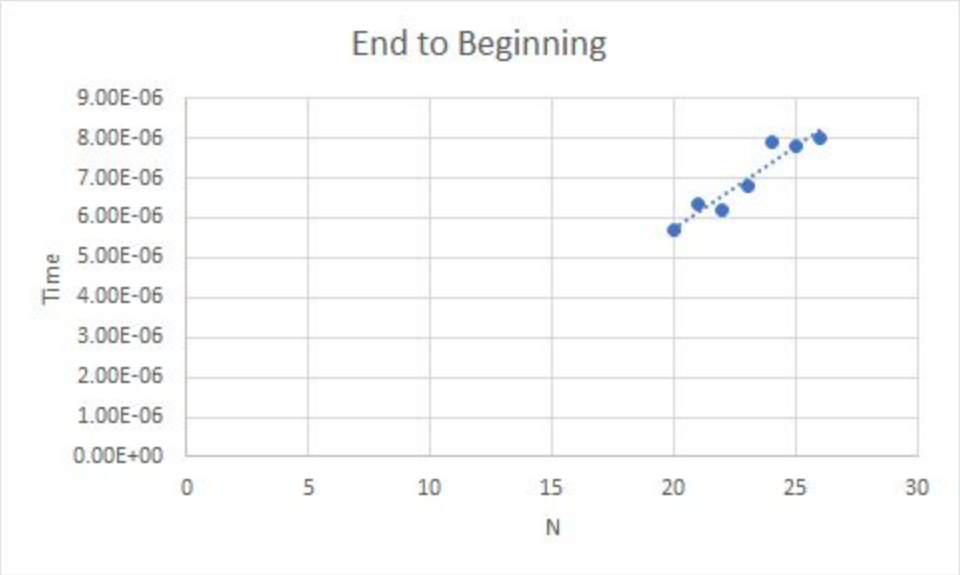
Time complexity:

1 t.u.
2 t.u.
1 t.u.
$2^n t.u.$
$1 + max(5, 5) = 6\ t.u.$
$1 + max(1, 0) = 2$ t.u.
$\frac{n-1}{1} + 1 = n - 1 + 1 = n\ t.u.$
1 t.u.
$3 + max(1, 0) = 4\ t.u.$

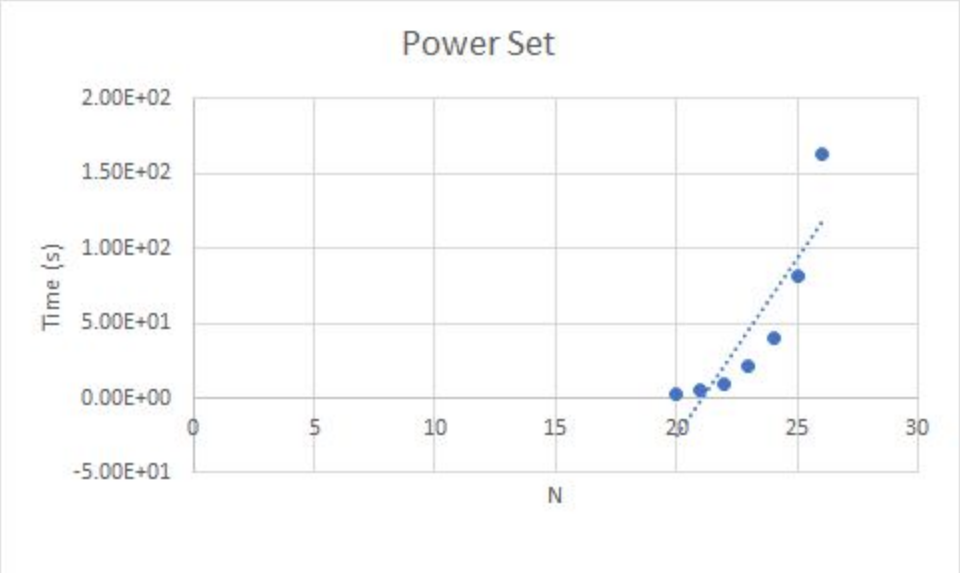
$$
\begin{aligned}
s.c. &= 1 + 2 + 1 + 1 + 2^n * (6 + 2 + n + 4) \\
&= 5 + 2^n * (12 + n) \\
&= 5 + 2^n * n + 12 * 2^n \\
&= O(2^n * n)
\end{aligned}
$$

Scatter plot:

End to beginning:
(20, 5.678e-06), (21, 6.365e-06), (22, 6.257e-06), (23, 6.813e-06), (24, 7.915e-06), (25, 7.824e-06), (26, 8.022e-06)



Powerset:
(20, 2.32883), (21, 4.92625), (22, 9.82563), (23, 21.568), (24, 39.7444), (25, 81.2059), (26, 162.285)



- **Is there a noticable difference between the two algorithms?**
    - There is quite a difference between the two algorithms since the amount of time it takes for the power set is significantly higher and increases exponentially

- **Are the best fit lines consistent or inconsistent with the efficiency classes?**
    - The best fit lines are consistent with their time complexity, after finding out their time complexity, it is easy to see why they have their points are laid out as they are.

- **Is this consistent or inconsistent with the hypothesis previously stated in the prompt?**
    - This is consistent with the hypothesis, the hypothesis stated that the power set would be a lot slower in comparison to the other algorithm, and with the data it proves that it is indeed true. Exhaustive is much easier to implement and saves run time overall, since it consistently takes slightly more time the more N is involved. Where the exponential algorithms had fast starting times but as more N's were added the time would spike up.