Vaibhav Patel

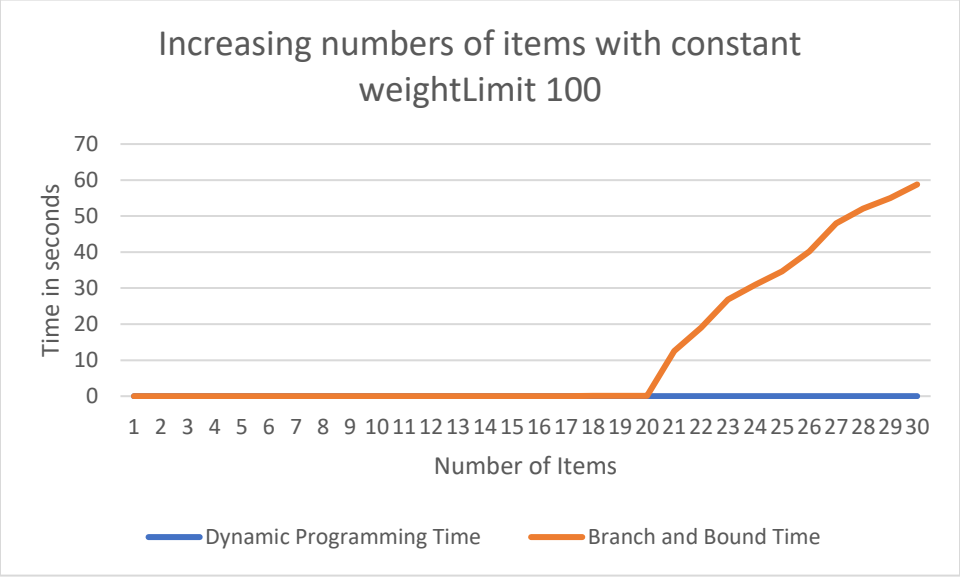
CS-340 Project7B

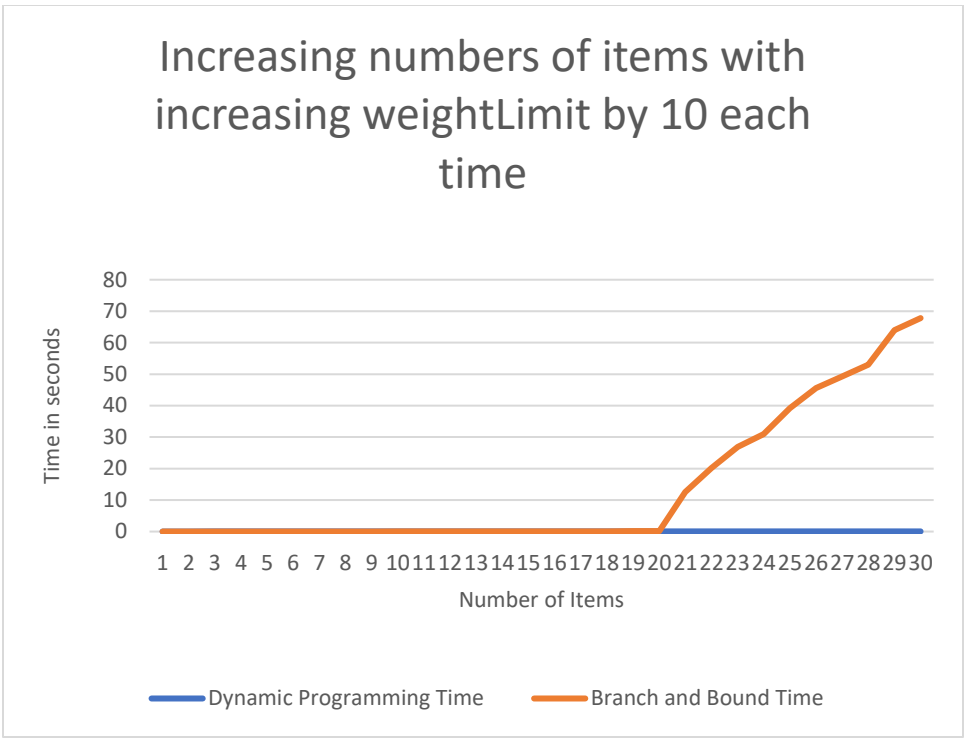
Report and Plots

Comparing two different approaches to find solution for knapsack was very interesting. Knapsack using dynamic programming, strictly follow O(nW) complexity. It is actually really fast when number of items increases compare to branch and bound approach. Between 25 to 30 items Brach and bound algorithm took way too long time. I did not wait to see results. For Branch and Bound approach, complexity depends on how many rules you have applied to come up with solution early, also it depends on what type of numbers are in the value and weight vector. Sometimes there are lots of ties and sometimes not. Dynamic programing guarantees O(nW) complexity where in branch and bound best we can do is O(2n + 1), but worst we can do is O(2^(n+1)) which is even worse. None of these methods, guarantees polynomial time complexity.

Knapsack is very hard to analyze and it is np-complete problem.

Here are some graphs but they will not make more sense because run time for dynamic programing for 20-30 items were very low than branch and bound approach.




Increasing numbers of items with increasing weightLimit by 10 each time
Time in seconds
80
70
60
50
40
30
20
10
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
Number of Items
Dynamic Programming Time
Branch and Bound Time