Sample of the highest instance solved

Weak bound

this is problem: Instance With 50 Items

id: 22 weight: 7 value: 86

id: 47 weight: 18 value: 68

id: 32 weight: 21 value: 76

id: 36 weight: 33 value: 75

id: 40 weight: 28 value: 56

id: 23 weight: 36 value: 68

id: 14 weight: 33 value: 52

id: 18 weight: 47 value: 67

id: 38 weight: 33 value: 46

id: 39 weight: 64 value: 81

id: 13 weight: 68 value: 75

id: 24 weight: 80 value: 86

id: 30 weight: 31 value: 30

id: 19 weight: 101 value: 87

id: 50 weight: 108 value: 89

id: 33 weight: 101 value: 83

id: 1 weight: 106 value: 83

id: 20 weight: 117 value: 91

id: 45 weight: 33 value: 23

Total value: 1322 Total weight: 1065 Max weight: 1106

The number of partial solutions generated: 2084048

Strong Bound:

this is problem: Instance With 50 Items

id: 22 weight: 7 value: 86

id: 47 weight: 18 value: 68

id: 32 weight: 21 value: 76

id: 36 weight: 33 value: 75

id: 40 weight: 28 value: 56

id: 23 weight: 36 value: 68

id: 14 weight: 33 value: 52

id: 18 weight: 47 value: 67

id: 38 weight: 33 value: 46

id: 39 weight: 64 value: 81

id: 13 weight: 68 value: 75

id: 24 weight: 80 value: 86

id: 30 weight: 31 value: 30

id: 19 weight: 101 value: 87

id: 50 weight: 108 value: 89

id: 33 weight: 101 value: 83

id: 1 weight: 106 value: 83

id: 20 weight: 117 value: 91

Total value: 1299 Total weight: 1032 Max weight: 1106

The number of partial solutions generated: 2082726

Conclusion

There isn’t much significant improvement on the number of solutions generated, I suspect that I should have used better tiebreakers for my min heap or possibly the methods I used to generate my bounds were either too similar or just poorly implemented.

Weak Lower bound:

Guaranteed future cost + every item that would weigh too much to fit (GFC)

Weak Upper Bound:

Guaranteed Future cost+ we use a greedy heuristic where the items are sorted in decreasing order for value/weight and we go through the list taking the first item that fits and adding the value of each item not taken to the Upper bound

Possible weak upper bound:

We could have not sorted the list and going down the list taking every item that fits and if it doesn’t add the value to the upper bound but the structure of my program depended on having the items sorted so I didn’t do this. Thus my initial upper bound pretty strong;

Strong Lower bound:

Does same as upper bound however also groups the remaining in a fashion where we must take out at least 1 item from the group to add the group to the items being taken and of this group of items we take the item with the smallest value and add that value to the lower bound.

Strong Upper Bound

Same as upper bound but also calculates taking every other item and adding the value of the items not taken to a temp upper bound we take whichever upper bound is less.