CS 312: Algorithm Analysis

Homework Assignment #21 – Local Search

Design a local search algorithm for the 0-1 knapsack problem. Assume there are n items $x_1 \ldots x_n$ each with weight w_i and value v_i. The knapsack can have at most one of each item and the total weight cannot exceed W. You want to maximize the total value in the knapsack.

Question 1: Show the psuedocode/explanation for your algorithm.

Neighbor operators:

Add_i – Adds an arbitrary item x_i if a) x_i is not currently in the sack, and b) the addition of x_i would not cause the total weight to exceed W.

$Swap_{ij}$ – Adds item x_i and takes out item x_j if a) x_i is not currently in the sack, b) x_j is currently in the sack, and c) the swap would not cause the total weight to exceed W.

Initialize knapsack to any set of items whose summed weight is $\leq W$
Repeat
    For any i and j
        Execute any one legal Add_i or $Swap_{ij}$ which increases the current summed value
    If no such Add_i or $Swap_{ij}$ exisits then terminate with the current knapsack

Question 2. Is it guaranteed to find an optimal solution? Justify your answer.

No. It could get stuck in a local maxima, such as where we need to swap in a heavy item to get to an optimum, but we would need to swap out multiple items not exceed W. To help with this we could add neighbor operators which allow to swap in 1 item and swap out multiple items at one time, at the cost of more neighbor options to check.