Problem 1: Oxen pairing

Consider the following problem: We have n oxen, $O_1, \ldots, O_n$, each with a strength rating S_i. We need to pair the oxen up into teams of two to pull a plow of weight P; if O_i and O_j are in a team, we must have $S_i + S_j \geq P$. Each ox can only be in at most one team. Each team has exactly two oxen. We want to maximize the number of teams.

Design an efficient algorithm for maximizing the number of oxen teams. Prove its correctness and state its complexity.

Key Idea:

Match the strongest unselected ox with the weakest unselected ox s.t.

$S_{strongest} + S_{weakest} \geq P$

Algorithm:

Sort the oxen by their strength rating st. for each ox in the sorted sequence Z, $S_{Z[1]} \leq S_{Z[2]} \leq \cdots \leq S_{Z[n]}$. From Z match the strongest ox (i.e. the right most element) with the weakest ox (i.e. the left most element) if $S_{Z[\text{length}(Z)]} + S_{Z[1]} \geq P$, and remove both ox from Z (no ox can be selected more than once). If $S_{Z[\text{length}(Z)]} + S_{Z[1]} < P$, remove $Z[1]$ from Z. Repeat this process until Z has 1 or 0 valid oxen remaining. At this point the paired/matched oxen returned by the previous iterations are the maximum number of teams possible.

Time Complexity:

Sorting takes $O(n \log n)$ time and matching valid pairs is linear in the number of oxen, n. Thus conclude the time complexity is $O(n \log n)$.

Proof of Correctness:

Let GS be the output of the above greedy algorithm, $GS = \{(g_1, g_2), (g_3, g_4), \ldots, (g_{k-1}, g_k)\}$ with each pairing placed according to its strongest ox (consider that to be the one at the odd indexes.) Note $0 \le k \le n$ (i.e. there are at most n oxen in GS or worst no matchings),

Let OS be the output of the optimal solution $OS = \{(O_1, O_2), (O_3, O_4), \ldots, (O_{k'-1}, O_{k'})\}$ with each pairing placed according to strongest ox (same strength metric ordering as GS) with $0 \le k \le k' \le n$.

Goal formulate an equivalent or better solution OS' by exchanging/swapping elements in OS with GS.

Case 1: $g_1 = O_1$

Subcase 1: $g_2 = O_2$

In this case the first pairs match:

$OS' = \{(g_1, g_2), (O_3, O_4), \ldots, (O_{k'-1}, O_{k'})\}$

Subcase 2: $O_2 \neq g_2$

Sub-subcase 1: $g_2 \notin OS$

Exchange g_2 for O_2 as $g_2 \notin OS$ and know since $O_1 = g_1$, then $O_1 = g_1$ & g_2 is a valid pairing by our greedy heuristic.

Sub-subcase 2: $g_2 \in OS$

If g_2 exists in OS then there must exist some other pair in OS s.t. have $(O_1 = g_1, O_2)$ and $(O_{m-1}, g_2 = O_m)$. Note that $g_2 = O_m$ cannot be larger than O_{m-1}, as we know g_2 is the weakest oxen allowable in any pairing as specified by the greedy heuristic (and sorting). Thus if $O_{m-1} \leq g_2$ then $O_{m-1}, O_m = g_2$ is an invalid pairing as the weakest element allowed in a pairing g_2 is paired with an even weaker ox and thus $S_{O_{m-1}} + S_{O_m} < P$. Knowing this we can exchange O_2 with $O_m = g_2$ and create 2 valid pairings since $O_2 \geq g_2$ (as $O_1 = g_1$ & thus $S_{O_2} + S_{g_1} \geq S_{g_2} + S_{g_1}$ since g_2 is the weakest allowable element). Thus pairing 1 is converted to (g_1, g_2) a valid pairing and (O_{m-1}, O_2). Note (O_{m-1}, O_2) has a combined strength $\geq (O_{m-1}, g_2)$

by the same above reasoning $S_{o_2} \geq S_{g_2}$.

<u>Case 2:</u> $g_1 \neq o_1$

<u>Subcase 1:</u> $g_1 \notin OS$

In this case we can simply exchange g_1 w/ o_1 as we know $g_1 \geq o_1$ as defined by greedy metric (g_1 is the strongest ox). Knowing this we can create an equivalent (g_1, g_2) pairing in OS' using the above reasoning in case 1 for all possibilities of o_2.

<u>Subcase 2:</u> $g_1 \in OS$

For this to occur that would mean their is a stronger ox o_1 then g_1 (as we sorted by strength). However we know that is not possible as our greedy algorithm picks the strongest ox, and thus for $S_{o_1} > S_{g_1}$, o_1 is an invalid option not in the oxen provided to us (contradiction). Note in the case when $S_{o_1} == S_{g_1}$ we can swap o_1 & g_1 as the net strength pairs are unchanged. We can create an equivalent pair (g_1, g_2) in OS' using the logic provided for Case 1 above.

Thus by repeating this exchange we can create an equivalent pairing $OS' = \{(g_1, g_2), (g_3, g_4), \ldots, (g_{k-1}, g_k), (o_{k+1}, o_{k+2}) \ldots (o_{k'-1}, o_{k'})\}$

$= GS + \{(O_{K+1}, O_{K+2}), \ldots, (O_{K'-1}, O_{K'})\}$

Note that for $K' > K$ there would need to be more valid pairing then returned by our GS. In this case oxen would need to be selected from invalid weakest oxen not capable of being paired w/ the strongest ox at any time point. Thus the sum of the strength of these pairings must necessarily be $< P$ and thus a contradiction. Thus $K' == K$ and have created an equivalent solution $OS' = GS$