CSCI3610: Special Exercise Set 3

Problem 1. If we run the activity-selection algorithm taught in the class on the following input:
\[ S = \{ [1, 10], [2, 22], [3, 23], [20, 30], [25, 45], [40, 50], [47, 62], [48, 63], [60, 70] \} \]
what is the set of intervals returned?

Problem 2. The following is another greedy algorithm for the activity selection problem. Initialize an empty \( T \), and then repeat the following steps until \( S \) is empty:

- (Step 1) Add to \( T \) the interval \( I \) with the shortest length.
- (Step 2) Remove from \( S \) the interval \( I \), and all the intervals overlapping with \( I \).

Finally, return \( T \) as the answer.

Prove: the above algorithm does not always return an optimal solution.

Problem 3 (Fractional Knapsack). Let \((w_1, v_1), (w_2, v_2), \ldots, (w_n, v_n)\) be \( n \) pairs of positive real values. Given a real value \( W \leq \sum_{i=1}^{n} w_i \), we want to find \( x_1, x_2, \ldots, x_n \) to maximize the objective function

\[
\sum_{i=1}^{n} \frac{x_i}{w_i} v_i
\]

subject to

- \( 0 \leq x_i \leq w_i \) for every \( i \in [1, n] \);
- \( \sum_{i=1}^{n} x_i \leq W \).

W.l.o.g., assume that \( v_1 \geq v_2 \geq \ldots \geq v_n \). Consider the algorithm that works as follows.

1. \textbf{for} \( i \leftarrow 1 \) \textbf{to} \( n \) \textbf{do}
2. \hspace{1em} \( x_i \leftarrow \min\{W, w_i\} \)
3. \hspace{1em} \( W \leftarrow W - x_i \)

Prove: the above algorithm does not always return an optimal solution.

Problem 4 (0-1 Knapsack). Suppose that there are \( n \) gold bricks, where the \( i \)-th piece weighs \( p_i \) bounds and is worth \( d_i \) dollars. Given a positive integer \( W \), our goal is to find a set \( S \) of gold bricks such that

- the total weight of the bricks in \( S \) is at most \( W \), and
- the total value of the bricks in \( S \) is maximized (among all the sets \( S \) satisfying the first condition).

Assuming \( d_1 \geq d_2 \geq \ldots \geq d_n \), let us consider the following greedy algorithm:

1. \( S = \emptyset \)
2. \textbf{for} \( i = 1 \) \textbf{to} \( n \)
3. \hspace{1em} \textbf{if} \( p_i \leq W \) \textbf{then}
4. \hspace{2em} add \( p_i \) to \( S \); \( W \leftarrow W - p_i \)

Prove: the above algorithm does not guarantee finding the desired set \( S \).