CS 170 Homework 11

Due **Monday 11/13/2023, at 10:00 pm (grace period until 11:59pm)**

1 Study Group

List the names and SIDs of the members in your study group. If you have no collaborators, you must explicitly write "none".

Solution: Got some help on Q2 and Q4 of coding from Bill during homework party, the writeup and everything else is my own.

2 Some Sums

Given an array $A = [a_1, a_2, \dots, a_n]$ of nonnegative integers, consider the following problems:

1 **Partition**: Determine whether there is a subset $S \subseteq [n]$ ($[n] := \{1, 2, \cdots, n\}$) such that $\sum_{i \in S} a_i = \sum_{j \in ([n] \setminus S)} a_j$. In other words, determine whether there is a way to partition A into two disjoint subsets such that the sum of the elements in each subset equal.

2 **Subset Sum**: Given some integer k, determine whether there is a subset $S \subseteq [n]$ such that $\sum_{i \in S} a_i = k$. In other words, determine whether there is a subset of A such that the sum of its elements is k.

3 **Knapsack**: Given some set of items each with weight w_i and value v_i, and fixed numbers W and V, determine whether there is some subset $S \subseteq [n]$ such that $\sum_{i \in S} w_i \leq W$ and $\sum_{i \in S} v_i \geq V$.

For each of the following clearly describe your reduction and justify its correctness.

(a) Find a linear time reduction from Subset Sum to Partition.

Solution: Let a_T denote the total sum of A. That is,

$$a_T = \sum_{i=1}^{n} a_i$$

Then, for the reduction from Subset Sum to Partition, we can insert the element $|a_T - 2k|$ and run Partition on it. If Partition returns yes, then Subset Sum also returns yes.

Proof of Correctness: We show that with this added term, if Partition returns yes, then so does Subset Sum. To do this, consider the set $A \cup \{|a_T - 2k|\}$, and we consider two cases depending on the sign of $a_T - 2k$.

Case 1: $a_T - 2k > 0$. If this is the case, then $|a_T - 2k| = a_T - 2k$. Then, the total sum of this new set will be $2a_T + 2k$, meaning that if Partition found a proper partition, then each of the two subsets (call them A'_L and A'_R) will sum to $a_T + k$.

Then, we focus on the set that contains the element $a_T - 2k$, and WLOG say we find it in A'_R. Then, this means that the sum of the set $A'_R \setminus \{a_T - 2k\}$ sums to k, so we've found a subset in A that sums to k. This completes case 1.

Case 2: $a_T - 2k < 0$. Here, $|a_T - 2k| = 2k - a_T$. If we consider a similar analysis as in case 1, then we find that the total sum of the elements in $A \cup \{|a_T - 2k|\}$ will be $2k$[1]. Then, if we partition this new

[1] $a_T + 2k - a_T = 2k$

set, we'll find that the sets A'_L and A'_R sum to k. Now consider the set that doesn't contain $a_T - 2k$, WLOG say that it's not in A'_R. Then, the elements in A'_R will sum to k, hence we've found a subset in A that sums to k. This completes case 2, and completes the proof.

(b) Find a linear time reduction from SUBSET SUM to KNAPSACK.

Solution: We set $W = V = k$, and set each $w_i = v_i = a_i$. Then, if knapsack returns a yes, then SUBSET SUM also returns yes.

Proof of Correctness: Consider what knapsack finds: it finds a subset of the elements in A such that $\sum_i w_i \leq W$, and $\sum_i v_i \geq V$. However, if $W = V = k$ and $w_i = v_i$ for all i, then this condition is the same as finding a subset of A such that $\sum_i a_i \leq k$ and simultaneously $\sum_i a_i \geq k$ for the same subset. Similar to linear programming, when we have the condition that some value $x \leq k$ and simultaneously $x \geq k$, then the only feasible solution is $x = k$. This is the same case here: so the only feasible solution would be a subset of A whose sum *equals* k, which is exactly what SUBSET SUM aims to find. Hence, with this configuration if KNAPSACK returns yes then SUBSET SUM also does.

3 Coding Questions: Reduction to Integer LP

For this week's coding questions, we'll walk through reducing the **Set Cover** problem to an **Integer Linear Program** and see how reductions can be used in practice. There are two ways that you can access the notebook and complete the problems:

1. **On Local Machine**: `git clone` (or if you already cloned it, `git pull`) from the coding homework repo,

   `https://github.com/Berkeley-CS170/cs170-fa23-coding`

   and navigate to the `hw011` folder. Refer to the `README.md` for local setup instructions.

2. **On Datahub**: Click here and navigate to the `hw11` folder if you prefer to complete this question on Berkeley DataHub.

Notes:

- *Submission Instructions:* Please download your completed submission `.zip` file and submit it to the Gradescope assignment titled "Homework 11 Coding Portion".
- *OH/HWP Instructions:* Designated coding course staff will provide conceptual and debugging help during office hours and homework parties.
- *Edstem Instructions:* Conceptual questions are always welcome on the public thread. If you need debugging help first try asking on the public threads. To ensure others can help you, make sure to:
  1. Describe the steps you've taken to debug the issue prior to posting on Ed.
  2. Describe the specific error you're running into.
  3. Include a few small test cases, alongside both the output you expected to receive and your function's actual output.

  If staff tells you to make a private Ed post, make sure to include *all of the above items* plus your full function implementation. If you don't provide them, we will ask you to provide them.
- *Academic Honesty Guideline:* We realize that code for some of the algorithms we ask you to implement may be readily available online, but we strongly encourage you to not directly copy code from these sources. Instead, try to refer to the resources mentioned in the notebook and come up with code yourself. That being said, we **do acknowledge** that there may not be many different ways to code up particular algorithms and that your solution may be similar to other solutions available online.