Assignment 2

Qinglin Li, 5110309074

Problem 1, 8.3

pseudo-polynomial

Let $B = \sum_{i=1}^{n} a_i$

Let $f(i,s) = \begin{cases} 1 & \exists S \subseteq [1..n] \sum_{i \in S} a_i = s \\ \\ 0 & otherwise \end{cases}$

Let $c(i,s) = \begin{cases} S & f(i,s) = 1 \\ \emptyset & f(i,s) = 0 \end{cases}$

We can obtain f and c by a dynamic programming algorithm.

```
f(0, 0) = 1
for i = 1 to n
  for s = 0 to B
    if f(i-1, s)=1
      f(i, s)=1
      c(i, s)=c(i-1, s)
    else if s >= a[i] and f(i-1, s-a[i])=1
      f(i, s)=1
      c(i, s)=union(c(i-1, s-a[i]), {i})
for s = B/2 to B
  if f(n, s)=1
    S1=c(n, s)
    break
S2=A/S1
```

Obviously, this algorithm is pseudo-polynomial.

Approximation algorithm

For $m = 2, \cdots, n$ let us denote by I_m the instance of Subset-Sums Ratio which consists of the m smallest numbers $a_1, ..., a_m$. At the top level, the algorithm executes its main procedure on the inputs I_m, for $m = 2, \cdots, n$, and takes as solution the best among the solutions obtained for these instances.
Given any ε in the range $0 < \varepsilon < 1$, we set $k(m) = \varepsilon^2 \cdot a_m/(2m)$.Let $n_0 \leq n$ be the greatest integer such that $k(n_0) < 1$.

We now describe the algorithm on the instance I_m.
If $m \leq n_0$, then we apply the pseudo-polynomial algorithm of the previous subsection to I_m. Since $a_{n_0} \leq 2n/\varepsilon^2$, this will take polynomial time.
If $n_0 < m \leq n$, then we transform the instance I_m into another one that contains only polynomial-size numbers. Set $a'_i = \lfloor a_i/k(m) \rfloor$ for $i = 1, \cdots, m$. Observe that $a'_m = \lfloor 2m/\varepsilon^2 \rfloor$ is indeed of polynomial size. Let us denote by I'_m the instance of Subset-Sums Ratio that contains the numbers a'_i such that $a'_i \geq m/\varepsilon$. Suppose that I'_m contains t numbers, $a'_{m-t+1}, \cdots, a'_m$. Since $\varepsilon \leq 1$,we have $a'_m \geq m/\varepsilon$, and therefore $t > 0$. We will distinguish between two cases according to the value of t.
Case 1: $t = 1$. Let j be the smallest nonnegative integer such that $a_{j+1} + \cdots + a_{m1} < a_m$. If $j = 0$, then the solution will be $S_1 = \{m\}$ and $S_2 = \{1, \cdots, m-1\}$. Otherwise the solution will be $S1 = \{j, j+1, \cdots, m-1\}$ and $S2 = \{m\}$.
Case2: $t > 1$.We solve I'_m,using the pseudo-polynomial algorithm which will take only polynomial time on this instance. Then we distinguish between two cases, depending on the value of the optimum of I'_m.
Case2a: $opt(I'_m) = 1$. The algorithm returns the solution which realizes this optimum for I'_m.
Case2b: $opt(I'_m) = 1$. We consider all possible pairs of disjoint subsets $P(m)$ and $Q(m)$ of $\{m-t+1, \cdots, m\}$ with sum of $P(m)$ greater than $Q(m)$. In the decreasing order, add $m-t, \cdots, 1$ to $Q(m)$ until sum of $Q(m)$ is greater than $P(m)$ or all elements are added into $Q(m)$. Then let $S_1(m)$ be the one with greater sum in $P(m)$ and $Q(m)$, $S_2(m)$ be the other.
Choose S_1 and S_2 among all $S_1(m)$ and $S_2(m)$ to minimize the ratio.

The above algorithm is FPTAS.

Problem 2

Problem: In the 2-approximation algorithm for job schedule, what if we change the algo- rithm by considering jobs according to their processing time. Analyze the approximation ratio of this variant algorithm.

Suppose the jobs are ordered by processing time, $p_1 \geq p_2 \geq \cdots \geq p_n$.
Let the last finished job be p_j with starting time s.
Remove all jobs with starting time greater than s. Let the new instance be I'. $OPT(I') \leq OPT(I)$ and the algorithm should give the same answer in both instances.
If $p_j \leq OPT(I')/3$, since all jobs starting after s are removed, $s \leq \frac{\sum_{i \neq j} q_i}{m} \leq OPT(I')$, $p_j + s \leq \frac{4}{3} OPT(I') \leq OPT(I)$
If $p_j > OPT(I')/3$, there cannot be two jobs before p_j, so the algorithm gives the optimal solution.
Overall, the algorithm is $\frac{4}{3} - approximation$