ADA2021 Homework4

Problem 5

```
/* 討論對象:
  b06303131 沈家睿
  b08501098 柯晨緯
  b09902040 洪郁凱
  b05504066 李旻翰 */
```

1. 我們先定義reduction function f:$f(a_1, a_2, \ldots, a_n;\ T) = (a_1+K, a_2+K, \ldots, a_n+K, \underbrace{K, K, \ldots, K}_{n\ K's};\ T+nK)$

   ，

   其中，$K > |T| + \sum_{i=1}^{n}|a_i|$。如此一來，因為$K > \sum_{i=1}^{n}|a_i| + |T|$，故任意subset of $\{a_1, \cdots, a_n\}$之總和必小於K，因此不存在任何subset of $\{a_1, \cdots, a_n\}$使得該subset的總和加上某個正的倍數的K能夠構成T；且因為$K > \sum_{i=1}^{n}|a_i|$，故也並不存在任何subset of $\{a_1, \cdots, a_n\}$使得該subset的總和可構成nK 。

   Claim: $JJBAP(a_1, \cdots, a_n; T) = 1 \iff JJBAP_+(f(a_1, a_2, \ldots, a_n;\ T)) = 1$

   Proof:

   - $\Rightarrow$:

     當$JJBAP(a_1, \cdots, a_n; T) = 1$，則有某個subset of $\{a_1, \cdots, a_n\}$之總和為T。假設該subset有m個element ($m \le n$)，那我們可以在$(a_1+K, a_2+K, \ldots, a_n+K, \underbrace{K, K, \ldots, K}_{n\ K's};\ T+nK)$取該subset中每個數對應到的$a_i+K$，再額外取$n-m$個$K$即可湊成$T+nK$，可得
     $JJBAP_+(a_1+K, a_2+K, \ldots, a_n+K, \underbrace{K, K, \ldots, K}_{n\ K's};\ T+nK) = 1$

   - $\Leftarrow$:

     當$JJBAP_+(a_1+K, a_2+K, \ldots, a_n+K, \underbrace{K, K, \ldots, K}_{n\ K's};\ T+nK) = 1$， 由於不存在任何subset of $\{a_1, \cdots, a_n\}$使得該subset的總和加上某個正的倍數的K能夠構成T ，因此nK的部分必由K組合而成，而T的部分則由某個subset of $\{a_1, \cdots, a_n\}$組成，代表$JJBAP(a_1, \cdots, a_n) = 1$成立。

   此外，由於我們只需花$\Theta(n)$的時間去計算K，也只要花$\Theta(n)$的時間將$(a_1, a_2, \ldots, a_n;\ T)$轉成$(a_1+K, a_2+K, \ldots, a_n+K)$並在其後再append n個K，因此可知這樣的reduction是polynomial-time的。綜合上述可知，$JJBAP \le_p JJBAP_+$。

3. 定義$DDBP(b_1, \cdots, b_n; k)$為「給一組數字$\{b_1, \cdots, b_n\}$，每個數字皆介於$[0,1]$之間，是否可以將它們分成$k$堆使得每堆的總和介於$[0,1]$之間」 。

   定義reduction function f 為: $f(a_1, a_2, \cdots, a_n) = (\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U};\ 2)$ if $U \neq 0$

   $$f(a_1, a_2, \cdots, a_n) = \begin{cases} (\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U};\ 2), & \text{if } U \neq 0 \\ (\underbrace{0, 0, \cdots, 0}_{n}; 1), & \text{if } U = 0 \end{cases}$$

Claim: $QQP(a_1, a_2, \cdots, a_n) = 1 \iff DDBP(f(a_1, a_2, \cdots, a_n)) = 1$

(1) 先證明$U \neq 0$的case，證明如下：

- $\Rightarrow$:

  當$QQP(a_1, a_2, \cdots, a_n) = 1$ ，我們可以將$\{a_1, a_2, \ldots, a_n\}$分出兩成兩個subset使各自的總和為$\frac{U}{2}$。將這兩個subset的每個element同乘$\frac{2}{U}$，可以看出兩個subset內的數都會介於$[0, 1]$之間且他們各自的總和都為1，故$DDBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U};\ 2) = 1$。

- $\Leftarrow$:

  當$DDBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U};\ 2) = 1$，代表我們可以將$(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U})$分成兩個subset使得他們各自的總和為1。，我將這兩個subset的每個element同乘$\frac{2}{U}$，這兩個subset的總和皆變為$\frac{U}{2}$，而他們剛好就是將$\{a_1, \cdots, a_n\}$ partition的解，因此可知$QQP(a_1, a_2, \cdots, a_n) = 1$。

(2) 再證明$U = 0$的case，證明如下：

當$U = 0$時，代表$\{a_1, a_2, \cdots, a_n\} = \{0, 0, \cdots, 0\}$，所以一定$QQP(a_1, a_2, \cdots, a_n) = 1$。此時$f(a_1, a_2, \cdots, a_n) = (0, 0, \cdots, 0; 1)$，且因所有0放在一堆總和也不會超過1，故$DDBP(0, 0, \cdots, 0; 1) = 1$。

可知$QQP(a_1, a_2, \cdots, a_n) = 1 \iff DDBP(f(a_1, a_2, \cdots, a_n)) = 1$得證。

我們將U計算而得只花了$\Theta(n)$的時間，將$\{a_1, \cdots, a_n\}$每個數乘上$\frac{2}{U}$也是花$\Theta(n)$的時間，可知這樣的reduction是polynomial-time。綜合上述，可知$QQP \leq_p DDBP$。

Claim: $DDBP(a_1, \cdots, a_n; k) = 1 \iff DBP(a_1, \cdots, a_n) \leq k$

Proof: 因為當$DDBP(a_1, \cdots, a_n; k) = 1$，代表$\{a_1, \cdots, a_n\}$可切成$k$堆或更少份，因此$DBP(a_1, \cdots, a_n) \leq k$。且當$DBP(a_1, \cdots, a_n) \leq k$時，代表分割成$k$堆是可行的，故$DDBP(a_1, \cdots, a_n; k) = 1$。

因為我們只要把constraint從$\{a_1, \cdots, a_n; k\}$中移除，即可得到對應於DBP的problem instance，因此是polynomial-time reduction，可知$DDBP \leq_p DBP$，從而推知$QQP \leq_p DBP$。

4. **Proof:** $QQP \in NP$

   - 給定一個QQP的解，我們可以用$\Theta(n)$的時間去計算，該解中給的兩個subset是否總和相同。因此我們可以在polynomial time內驗證該解的正確性，因此QQP屬於NP。

   Proof: $DDBP \in NP$

   - 給定一個$DDBP$的解，我們可以用$\Theta(n)$的時間去計算，該解中給的k個subset是否總和不超過1。因此我們可以在polynomial time內驗證該解的正確性，因此$DDBP$屬於NP。

   由subproblem(2)(3)，可知$JJBAP \leq_p QQP$且$QQP \leq_p DDBP$。因此，$DDBP$和QQP都至少比 $JJBAP$難；又因已知$JJBAP$是NP-Complete，故QQP與$DDBP$屬於NP-Complete。

5. 我們先假設存在一個polynomial-time的$(\frac{3}{2} - \epsilon)$-approximation algorithm，並且分析在這樣的情況下計算$QQP(a_1, a_2, \cdots, a_n)$所需要花的時間：

   (1) 如果$\Sigma_{i=1}^{n} a_i = U = 0$，我們可以直接花$\Theta(1)$時間output $QQP(a_1, a_2, \cdots a_n) = 1$。

   (2) 如果$U \neq 0$，我們可以用以下方式解：

   因為存在一個polynomial-time的$(\frac{3}{2} - \epsilon)$-approximation algorithm，我們定義該演算法對$DBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U})$的解為$ans$，且$m$為$DBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U})$的最佳解。由approximation algorithm的性質可知：$m \leq ans \leq (\frac{3}{2} - \epsilon) \cdot m$，且$0 < \epsilon < \frac{1}{2}$。又因$\{\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U}\}$中每個數的總和為2，故$m \geq 2$。

   - 當$m < 3$，則$m = 2$且$m \leq ans \leq (3 - 2\epsilon)$，故知$ans < 3$。可得：$ans \geq 3 \Rightarrow m \geq 2$。
   - 因approximation algorithm得出的結果必定大於最佳解，可知$ans \geq m$ 。可得：$ans < 3 \Rightarrow m < 3$。

故如果該演算法存在，我們可以用該演算法解出ans，再以ans的結果於$\Theta(1)$時間內得出$DBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U})$的範圍，如果得出的範圍是小於3，那我們就可以知道$DDBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U}; 2) = 1$；反之若得出的範圍是大於等於3，那就代表$DDBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U}; 2) = 0$。接著再以$DDBP(\frac{2a_1}{U}, \frac{2a_2}{U}, \ldots, \frac{2a_n}{U}; 2)$的結果，花上$\Theta(1)$的時間在推測出$QQP(a_1, \cdots, a_n)$的結果，整段求出$QQP$的過程是花了polymomial time。

由上述討論可知，當存在一個polynomial-time的$(\frac{3}{2} - \epsilon)$-approximation algorithm，我們必可以用polynomial time解出QQP。因為我們由上題結果得知QQP屬於NP-complete，這樣的結果表示P=NP，然而題目設定P≠NP，因此得到矛盾。可知不存在一個polynomial-time的$(\frac{3}{2} - \epsilon)$-approximation algorithm，$0 < \epsilon$。

6. 因為此題中每顆球中量至少有c，可知一個bin中最多可有$\lfloor \frac{1}{c} \rfloor$顆球。

   我們定義x_i為一個bin中有幾個第i^{th} type的球，可以列出$x_1 + x_n + \cdots + x_m \leq \lfloor \frac{1}{c} \rfloor$，我們可以額外設定一個變數$0 \leq N \leq \lfloor \frac{1}{c} \rfloor$，使得$x_1 + x_2 + \cdots + x_m + N = \lfloor \frac{1}{c} \rfloor$。我們可以得到$\{x_1, x_2, \cdots, x_m\}$可能的解有$H^{m+1}_{\lfloor \frac{1}{c} \rfloor}$，代表一個bin可以有的valid組合數之upper bound。

7. 令x_i為第i個組合被使用的次數，所以$x_1 + x_2 + \cdots + x_M = k$，因此$k$個相同bin可能的放法為$H^k_M = C^{k+M-1}_k$，展開得：

$$
\begin{aligned}
C^{k+M-1}_k &= \frac{(k+M-1)(k+M-2)\cdots(k+1)}{k \cdot (k-1) \cdots 1} \\
&= (1 + \frac{k}{M-1})(1 + \frac{k}{M-2}) \cdots (1+k) \\
&= O(k^{M-1}) \\
&= O(k^M)
\end{aligned}
$$

8. 我們可以知道，切成k個bin可能的組合是$O(k^{M-1})$種，且要計算一個組合是否每堆之總和介於$[0, 1]$之間，我需要花$\Theta(n)$的時間加總，所以看切成k個bin是不是可能的，我只需要花$O(nk^{M-1}) = O(n^M)$。因此我可以從$k = 1$檢查到$k = n$，即可知道最少可以切成幾堆，因此總複雜度為$O(n \cdot n^M) = O(n^{M+1})$。因為要知道切成$k$個bin是否可能，需要花$O(n^M)$，所以$DDBP$是polynomial-time solvable，代表$DDBP$是P問題。

Problem 6

```
/* 討論對象:
  b06303131 沈家睿
  b08501098 柯晨緯
  b09902040 洪郁凱
  b05504066 李旻翰 */
```

1. 構成Eulerian cycle的充要條件為：$\forall v \in G.V,\ degree(v)$為偶數。
2. 由於增加一個edge會使一個graph的總degree多加2，因此對任意graph G而言，$\Sigma_{v\in G.V}\ degree(v) = 2|G.E|$，假設一個tree T存在奇數個odd-degree vertices，那麼odd-degree verices的degree總和會是奇數，又因為even-dgree vertices的degree總和必為偶數，可知T的degree總和是奇數，此與先前的結論矛盾，因此一個tree T必存在偶數個odd-degree vertices。
3. 取$E' \subseteq E$，使得$G' = (V', E')$形成一個complete graph，我們可以假設G'的**TSP** problem之最佳解為OPT'且$OPT' \leq OPT$ (否則根據三角不等式，OPT'就不會是G'的最佳解)。我們首先假設$cost(M) > OPT'/2$，並且定義OPT'的路徑為$v_1 \to v_2 \to v_3 \to \cdots \to v_n \to v_1$，其中$\forall i \in [1, n],\ v_i \in V'$。因為$|V'|$是偶數，所以如果拆解$OPT'$的路徑可以得到兩種perfect matching of G'，分別為$S_1 = \{(v_1, v_2),\ (v_3, v_4) \cdots,\ (v_{n-1}, v_n)\}$以及$S_2 = \{(v_2, v_3),\ (v_4, v_5) \cdots,\ (v_{n-}, v_1)\}$，我們可以不失一般性的假設$cost(S_1) \geq OPT'/2$，而$cost(S_2) \leq OPT'/2$ (因為$cost(S_1) + cost(S_2) = OPT'$)，如果$cost(M) \geq OPT'/2$，代表我可以找到令一個$cost$更低的perfect matching，$M$就不會是minimum cost perfect matching of G'，因此可知$cost(M) \leq OPT'/2 \leq OPT/2$。
4. 給定一個graph $G(V, E)$，我們先用$O(|V|^2)$的時間去用Prim's algorithm找出G的minimum spanning tree T，再花$O(|V|^2)$的時間找出complete graph $G'(V', E')$，其中$E' \subseteq E$。我們對G'以**Oracle**花polynomial time找出其minmum cost perfect matching M，並將G''定義為$G'' = T \bigcup M$，特別注意如果聯集之後有重疊部分，不可以將原本的邊覆蓋掉而是要形成重邊，可知因為所有奇數點都變成even-degree，G''成為一個Eulerian Cycle。接下來我們可以對G''從$T.root$開始往下走，只要走到一個點u，如果與u相連的所有點都已經被走過，那我們就隨便找一個點v與u相連並且移除原本在Eulerian cycle上從$u \to v$的路徑，根據三角不等式可知這樣一來cost會再降低，且這樣所花的時間為$O(|V|^2)$，因為每走到一點就要檢查其他$|V|$個點哪些被走過，如果要移除邊也要花$O(|V|)$的時間去移除。最終再將最後一個走到的點u'連到$T.root$並且把在原本的Eulerian cycle上連接$u' \to T.root$的路徑移除。可以得到最後的G''的總cost小於等於$T \bigcup M$的cost，且因為$cost(T) \leq OPT$ (因為T是MST)，且$cost(M) \leq OPT/2$，所以$cost(G'') \leq cost(T \bigcup M) \leq \frac{3}{2}OPT$，因此這是一個$\frac{3}{2}$-approximation algorithm。此外因為整個演算法的每個步驟都是花polynomial time，因此可知整體也是花polynomial time。