Problem #5

#1 Show JJBAP $\le_p$ JJBAP$_+$

let $S=(a_1, a_2, \dots a_n)$ and $L=\{S;T\}$ be an instance of JJBAP

let the transformation $f(L)$ define as:

$$f(L)=\{S_+;\ T+nk\} \text{ where } S_+=\{a_i+k \mid a_i\in S\}\cap\{k\}^{*}n個$$

$$=\{a_1+k, a_2+k, \dots, a_n+k, \underbrace{k, k\dots k}_{n個}\}$$

$$\text{and } K=\max\{\textstyle\sum|a_i|+1+|T|\}$$

$\rightarrow f(L)$ is a legal instance for JJBAP$_+$ $\because K\ge -\min\limits_{a_i\in S} a_i$

$\therefore \forall a_i\in S,\ a_i+k\ge 0$

① proof correctness of reduction $L\in$ JJBAP $\Leftrightarrow f(L)\in$ JJBAP$_+$

<pf> "$\rightarrow$" if $L\in$ JJBAP, we can find a subset S' in S s.t. $\text{sum}\{S'\}=T$

在$f(L)$中，如果選擇所有S'中的element a_i 由f轉換後對應到的a_i+k

再選擇 $n-|S'|$ 個值為K的element，就會得到一個set S_+'

其sum為$T+nK \rightarrow f(L)\in$ JJBAP$_+$

"$\leftarrow$" if $f(L)\in$ JJBAP 可以找到一個 subset S_+' in S_+ s.t. $\text{sum}\{S_+'\}=T+nk$

假設 $|S_+'|=m$, let $S_+'=\{b_1+k, b_2+k, \dots, b_j+k, k, \dots k\}$

let S'' 為S_+'中，由S transform 過去的那些原先在S中的 set. $S''=\{b_1, b_2 \dots b_j\}$

$\text{sum}\{S_+'\}=T+nk=\text{sum}\{S''\}+mk$

因為 $K>$ biggest sum of subset of S 且 $-K<$ smallest sum of subset in S

$$\quad\ \ \sum_{a_i\in S,\, a_i\ge 0} a_i \qquad\qquad\qquad\qquad \sum_{a_i\in S,\, a_i<0} a_i$$

考慮以下2種case

case1: $sum\{S''\} \geq 0$

$\because K > \sum_{a_i \in S} |a_i| > sum\{S''\}$，且 $K > |T|$

$\therefore m \cdot K \leq sum\{S''\} + mK = T + n \cdot K < (m+1)K$

所以 $n = m \rightarrow sum\{S''\} = T$

$\therefore$ 在L中選 S'' 可以使得 sum 為 $T \rightarrow L \in JJBAP$

case2: $sum\{S''\} < 0$

$\because K > \sum_{a_i \in S} |a_i| > -sum\{S''\} \rightarrow sum\{S''\} > -K$，且 $|T| < K$

$\therefore (m-1)K < sum\{S''\} + mK = T + nK \leq mK$

$\therefore n = m \rightarrow sum\{S''\} = T$

在L中選 S'' 可使得 sum 為 $T \rightarrow L \in JJBAP$

② polynomial time reduction:

given input size $n = |S|$

要找到 K 可以 $O(n)$ 做絕對值加總

找到 K 之後，要把 S 變成 S' 的 cost 也是 $O(n)$，只需要將所有 S 中的 element 加上 K，再多加 n 個 K 加 S' 就好了。

2 Show JJBAP $\le_p$ QQP
(knapsack) (partition)

<pf> ① correctness
② poly. time reduction

let $L=\{S; T\}$ be an instance of JJBAP, where $S=(a_1, \dots a_n)$ is a sequence of integers and T is the target sum.

define the transformation $f(L)=\{S'\}$

for S', add $a_{n+1} = \left|\sum_{a_i \in S} a_i - 2T\right|$

① proof correctness of the reduction: $L \in JJBAP \leftrightarrow f(L) \in QQP$

"$\rightarrow$" if $L \in JJBAP$, there exists a subset P of S s.t. $\sum_{a_i \in P} a_i = T$
(also of S')

Case 1: $\sum_{a_i \in S} a_i - 2T \ge 0 \rightarrow \sum_{a_i \in S'} a_i = \sum_{a_i \in S} a_i + \left(\sum_{a_i \in S} a_i - 2T\right)$

the target of QQP is to find $\frac{\sum_{a_i \in S'} a_i}{2} = \sum_{a_i \in S} a_i - T$

因為 S' 中有 subset P, where $\sum_{a_i \in P} a_i = T$, let $P' = P \cup \{a_{i+1}\}$

$\rightarrow P'$ meets the requirement

則 $P' = T + \sum_{a_i \in S} a_i - 2T$
$= \sum_{a_i \in S} a_i - T$

Case 2: $\sum_{a_i \in S} a_i - 2T < 0 \rightarrow \sum_{a_i \in S'} a_i = \sum_{a_i \in S} a_i + 2T - \sum_{a_i \in S} a_i = 2T$

the target of QQP is to find $\frac{\sum_{a_i \in S'} a_i}{2} = \frac{2T}{2} = T$.

chosing the subset P meet the requirement.

$\therefore f(L) \in QQP$

"$\leftarrow$" if $f(L) \in QQP$, the target is to find a subset in S, s.t. $\sum_{a_i \in S} a_i = T$

Case 1: $\sum_{a_i \in S} a_i - 2T \ge 0$, S' 中可找到 subset P_1 和 P_2, 且

$\sum_{a_i \in P_1} = \frac{\sum_{a_i \in S} a_i + \sum_{a_i \in S} a_i - 2T}{2} = \sum_{a_i \in S} a_i - T$. $\supset \sum_{a_i \in P_2} a_i = \sum_{a_i \in P_1} a_i$

WLOG, let P_1 contains $a_{n+1} = \sum_{a_i \in S} a_i - 2T$

let $P_1' = P_1 \setminus \{a_{n+1}\}$, 則 $\sum_{a_i \in P_1'} = \sum_{a_i \in P_1} - \left(\sum_{a_i \in S} a_i - 2T\right) = T$

$\therefore$ for L, we can choose P_1' that meets the requirement

Case 2: $\sum_{a_i \in S} a_i - 2T < 0$，$S'$ 中可找到 subset P_1 和 P_2，且

$$\sum_{a_i \in P_1} = \frac{\sum_{a_i \in S} a_i + \left(2T - \sum_{a_i \in S} a_i\right)}{2} = T, \sum_{a_i \in P_2} a_i = \sum_{a_i \in P_1} a_i$$

WLOG, let P_1 contains a_{n+1}.

則 P_2 is a subset of S and meets the requirement

$\therefore L \in$ JJBAP

② polynomial time reduction

找到 $a_{i+1} = \left|\sum_{a_i \in S} a_i - 2T\right|$ 只要 $O(n)$ sum S 就好。

#3

DDBP: (decision version of DBP)

Given a collection of n balls $\mathcal{B}$ with weights $a_1, \cdots, a_n$ kilograms respectively, with constraint that $a_i \in [0,1]$, i.e., a single ball's weight is at most 1 kilogram and at least 0 kilogram (yes, a_i may be 0, to make the following problem simpler). The decision *Dragon Ball Problem* seeks whether there is a way to partition balls into a k number of bins such that all the bins weigh at most 1 kilogram.

show QQP $\le_p$ DDBP

let $S = (a_1, a_2, \dots a_n)$, $L = S$ be an instance of QQP,

define the transformation $f(L)$ as:

let all $a_i \in S$ become $a_i' = a_i \cdot \frac{2}{\sum_{a_i \in S} a_i}$ and now we have S'

let $k = 2$

$f(L) = \{S'; k\}$

① proof the correctness $L \in QQP \leftrightarrow f(L) \in DDBP$

"→" if $L \in QQP$, there $\exists$ 2 different subset P_1、P_2, whose sum are equal and are $\frac{\sum_{a_i \in S} a_i}{2}$.

after transformation, we have P_1'、P_2' and the sum are now both $\sum_{a_i' \in P_1'} a_i' = \sum_{a_i \in P_1} \left(a_i \times \frac{2}{\sum_{a_i \in S} a_i}\right) = \left(\sum_{a_i \in P_1} a_i\right) \times \frac{2}{\sum_{a_i \in S} a_i} = 1$

and because $\forall a_i \ge 0$, $\rightarrow \forall a_i' \ge 0$

$$\forall a_i \le \max_{a_i \in S}(a_i) \le \frac{\sum_{a_i \in S} a_i}{2} \rightarrow a_i' \le \max_{a_i' \in S}(a_i') \le \frac{\sum_{a_i \in S} a_i}{2} \cdot \frac{2}{\sum_{a_i \in S} a_i} = 1$$

$\rightarrow \forall a_i', 0 \le a_i' \le 1$

So if we have $S' = P_1' \cap P_2'$ as the input of DDBP, P_1'、P_2' can be exactly 2 partitions, so $f(L) \in DDBP$

"$\Leftarrow$" if $f(L) \in DDBP \rightarrow \forall a_i'\ 0 \le a_i' \le 1$

there are 2 partitions of S', said P_1', P_2' that have sum smaller than 1.

However, $\because \sum_{a_i' \in S'} a_i' = \sum_{a_i \in S} a_i \left(\frac{2}{\sum_{a_i \in S} a_i}\right) = 2$.

$$\begin{cases} 0 \le \sum_{a_i' \in P_1'} a_i' \le 1 \\ 0 \le \sum_{a_i' \in P_2'} a_i' \le 1 \\ \sum_{a_i' \in S'} a_i' = \sum_{a_i' \in P_1'} a_i' + \sum_{a_i' \in P_2'} a_i' = 2 \end{cases} \rightarrow \begin{matrix} \sum_{a_i' \in P_1'} a_i' = 1 \\ \sum_{a_i' \in P_2'} a_i' = 1 \end{matrix}$$

令 P_1 為 P_1' 所有 element a_i' transform 回 transformation 之前的 a_i

i.e. $\forall a_i' \in P_1$, let $a_i = a_i' \times \frac{\sum_{a_i \in S} a_i}{2}$, P_2' 也同理轉換回 P_2

則 $\sum_{a_i \in P_1} a_i = \sum_{a_i' \in P_1'} a_i' \times \frac{\sum_{a_i \in S} a_i}{2} = \left(\sum_{a_i' \in P_1'} a_i'\right) \frac{\sum_{a_i \in S} a_i}{2} = \frac{\sum_{a_i \in S} a_i}{2}$

$\therefore P_1$ is a set that have sum $= \frac{\text{all sum in } S}{2}$.

$\therefore L \in QQP$.

② polynomial time reduction.

轉換 S 為 S' 只須求 sum of S 並 transform 每個 element in S 即可

$\sim O(n)$

因為所有 decision problem 可以在 polynomial-time reduce 為 optimization problem

課堂上已證明.

(可建構一個 algorithm 對 k 在上下界範圍內以 binary search 的方式找到使 DDBP output true 的 upper bound)

所以 $QQP \le_p DDBP \le_p DBP$

$\rightarrow QQP \le_p DDP$

u — 不可

uu — 不可

uuu — 可

uuuu — 可

#4 ① proof $QQP \in NP$:

with input problem instance: S (a sequence)

given a certificate C (a subsequence)

- first, check if C is a subsequence of S by sorting both S and C, and use 2-pointer to check 1 by 1. $\sim O(n\log n)$
- next, sum up S and sum up C, check if sum of $C = \frac{\text{sum of } S}{2}$ $\sim O(n)$

if we pass both "checks", C is in our decision problem QQP and we verify it in polynomial time.

② proof $DPBP \in NP$

with input problem instance k and a sequence of n balls S

given a certificate $C = \{C_1; C_2; \dots C_n\}$ (k different subsequence of S)

- first, check if concating all $C_i \in C$ and sorting it get the result is the same as sorted S $\quad O(n\log n)$
- next, check if $\forall\, C_i \in C,\ \sum_{x \in C_i} x \le 1$ $\quad O(n)$

if we pass both "checks", C is in our decision problem $DPBP$ and we verify it in polynomial time.

$\because JJBAP \in NPC$

$JJBAP \le_p QQP \rightarrow QQP \in NP\text{-hard}$, $\& \because QQP \in NP \rightarrow QQP \in NPC$

$QQP \le_p DPBP \rightarrow DPBP \in NP\text{-hard}$, $\& \because DPBP \in NP \rightarrow DPBP \in NPC$

#5 Show: there is no polynomial time $(\frac{3}{2}-\varepsilon)$-approximation for DBP

<反證>

如果∃ Alg_{AP} 為 poly-time $(\frac{3}{2}-\varepsilon)$-approximation 演算法 for DBP

我們可以用 Alg_{AP} 來 construct QQP 並在 poly-time 解 QQP，但這起矛盾了.

- 考慮#3 小題用的 f，在 polynomial time 將 $x\in$ instance of QQP 轉成 $f(x)\in$ instance of DPBP，但在 optimization problems 中沒有 k，我們可以進一步將原本的 DPBP for k=2，轉換為：if DBP output value ≤ 2 ?

(DBP output $\leq 2 \leftrightarrow$ DPBP for k=2 is true)

令 $f'(x)$ 為 $f(x)$ 中，除 x 轉換後的 set，$f'(x)\in$ instance of DBP

- proof correctness of reduction.

- 由#3 的說明，$x\in QQP \leftrightarrow f(x)\in DPBP$

在原 $f(x)$ 中，k 恆 $=2$，∴ if $f(x)\in DPBP$，Alg_{AP} input $f(x)'$ 必輸出 2，

因為若 output 任何 ≥ 3 的數字，皆違反 $P(n)=\frac{\text{approx. answer}}{2}=\frac{3}{2}-\varepsilon$

$\rightarrow x\in QQP \leftrightarrow f(x)\in DPBP$

$\leftrightarrow Alg_{AP}(f(x)')$ output 2

則我們可以用上述方法在 poly. time 內解 QQP，但已知 QQP $\in$ NPC，

($\rightarrow\leftarrow$)

#6. 因為最小重量是c，所以一個 single bin 裡最多只會有 $\lfloor \frac{1}{c} \rfloor$ items，

所有球總共m種不同種類，考慮每種最多只會選$\lfloor \frac{1}{c} \rfloor$次，將多餘的去除，原本 n balls 減少為 $m * \lfloor \frac{1}{c} \rfloor$ balls。

考慮 $\lfloor \frac{1}{c} \rfloor$ items、$\lfloor \frac{1}{c} \rfloor - 1$ items, ..., 0 items 的 bin，

所有可能的 bin 排列組合有

m種不同球各 $\lfloor \frac{1}{c} \rfloor$ 個

$\lfloor \frac{1}{c} \rfloor - 1$

$$\binom{\lfloor \frac{1}{c} \rfloor + m-1}{m-1} + \binom{\lfloor \frac{1}{c} \rfloor + m-2}{m-1} + \cdots + \binom{m-1}{m-1} \text{ 種可能}$$

#7. 現在有k個 indistinguishable bin，考慮每種可能的組合可以取 0~k 次，每次取就佔據一個 bin，直到佔滿為止，

則以上列舉方法的列舉數的 upperbound 為 $(k+1)^M$

$$\text{又 } (k+1)^M = k^M + C^M_1 k^{M-1} + C^M_2 k^{M-2} + \cdots + C^M_M k^0$$

$$\leq k^M + C^M_1 k^M + C^M_2 k^M + \cdots + C^M_M k^M = (1 + \underbrace{C^M_1 + C^M_2 + \cdots + C^M_M}_{= C_M}) k^M$$

#8. - 先以#6的方法，用 $O(M)$ ($\sim O(1)$) 時間找到所有可能可以放進 bin 的 sets，

每個 set 以 $O(\lfloor \frac{1}{c} \rfloor)$ ($\sim O(1)$) 驗證是否合法，會檢查 total weight 是否 ≤ 1，以及各類取用數目是否超過 n balls 中該類別的數目 ← 可以先 sort n balls，統計每種球總類 $\sim O(n \log n)$

$\rightarrow O(n)$

- 對k做#7的列舉方法

以 $O(k^M)$ 考慮所有可能放在bin的方法。

每種放法都以 $O(n^2)$ 驗證是否為符合規則的放法

最終cost為 $O(n^2 * k^M) = O(n^2 \cdot n^M) = O(n^{M+2})$

#1. 每個vertices都有偶數degree時，且每個non-zero degree的點G都屬於同一個connected-component.

#2. 因為 $\underset{\forall u, deg(u) \text{ is odd}}{sum}\{degree(u)\} + \underbrace{\underset{\forall u, deg(u) \text{ is even}}{sum}\{degree(u)\}}_{even} = \underbrace{2^{*}|E|}_{even}$

$\underset{\forall u, deg(u) \text{ is odd}}{sum}\{deg(u)\}$ must be even, $\therefore |\{u \text{ s.t. } deg(u) \text{ is odd} / \forall u \in T\}|$ must be even.

(偶數個奇數加總為偶數，奇數個奇數加總仍為奇數)

#3. <反證法> if $cost(M) > \frac{OPT}{2}$

let T 為G的 TSP 解形成的環. $(G=(V,E))$

1. 若|G|為even，則在T上可以找到一個minimum cost perfect matching M'，$E-M'$ 形成另一個perfect matching，$\because cost(M') < cost(E-M')$
且 $cost(M') + cost(E-M') = OPT$
$\rightarrow cost(M') \le \frac{OPT}{2}$ $(\rightarrow\leftarrow)$

2. 若|G|為odd，則在T上任意去掉任何一個node u，所能找到的minimum cost perfect matching中，cost最小的為 M''.

$\because cost(M'') < cost$(other matching in G', where G' is G去掉任一個node)
$< cost(E-M'')$

$\rightarrow$ 又 $cost(M'') + cost(E-M'') = OPT$

$\rightarrow cost(M'') \le \frac{OPT}{2}$ $(\rightarrow\leftarrow)$

$\therefore cost(M) \le \frac{OPT}{2}$ #

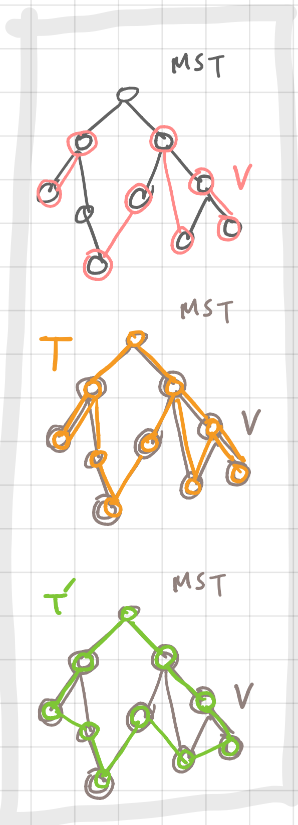
#4.

Step 1: find a MST for G $\sim O(E \lg V)$

Step 2: let G' be a completed graph constructed by all the nodes in MST that have odd degree. (by 2, we know $|G'|$ is even) $\sim O(E)$

Step 3: using Oracle(G') to find V for G ~polynomial time

V: minimum cost perfect tree



We add all the edge in V to MST and get the graph T. $\sim O(E)$

in T, all the nodes have even degree. so $\exists$ eulerian cycle,

T 為 a path, whose $cost(T) = cost(MST) + cost(V)$

用課堂上的抄捷徑方法, 把 T 轉換為 tour T'. $\sim O(E)$

$\because$ G 滿足 Triangle inequality $cost(T') < cost(T)$

let T^* 為 optimal tour. $\because$ T^* 為 Tree plus some edge.

$$cost(T^*) > cost(MST)$$

又由3. $$\frac{cost(T^*)}{2} \geq cost(V)$$

$$\therefore cost(T') < cost(T) < cost(T^*) + \frac{cost(T^*)}{2} = \frac{3}{2} cost(T^*)$$

$$\therefore \rho(n) = \frac{3}{2}$$

all the above step run in polynomial time.