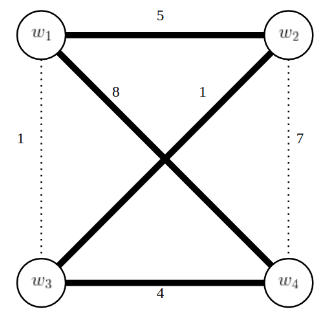
ADA HW #4 - Hand-Written

Student Name: 林楷恩
Student ID: b07902075

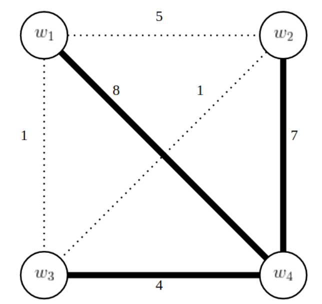
Problem E - Magic Wands Linkings

(1) * The thick black line denotes a link, while the dotted line means it is not a link.

(1-1) **enhanced power**$=2 \times 18 = 36$, **light**$=\{w_1, w_3\}$, **dark**$=\{w_2, w_4\}$,
links$=\{(w_1, w_2), (w_1, w_4), (w_3, w_4), (w_3, w_4)\}$



(1-2) **enhanced power**$=2 \times 19 = 38$, **light**$=\{w_1, w_2, w_3\}$, **dark**$=\{w_4\}$,
links$=\{(w_1, w_4), (w_2, w_4), (w_3, w_4)\}$



(2)

(2-1) Because the original problem is optimization problem, I first define the decision problem version of the two problems:

- **MaxCut:** $\{\langle G, k\rangle \mid G$ has a cut of size $\geq k$ $\}$,
  where $G = \langle V, E\rangle$ denotes the set of vertices and edges.

- **P(2-1):**$\{\langle W, fit, k\rangle \mid \exists$ a linking method satisfying RULE1 s.t. the enhanced value $\geq k\}$,
  where W is the set of wands, fit is the fitness value function defined on $W \times W \to \mathbb{R}$

I will show that **MaxCut** $\leq_p$ **P(2-1)** to prove that **P(2-1)** is NP-hard.

I. **Polynomial Time Reduction**:

$F =$ "On a instance $G = \langle V, E\rangle$ of Maximum Cut Problem:
1. Initialize W be an empty set of wands, and fit be the fitness value function
2. for all vertices $v_i \in V$, add a "wand" w_i to W......$O(N)$
3. for all edges $(v_i, v_j) \in E$, $fit(w_i, w_j) = 1$......$O(N^2)$
4. for all edges $(v_i, v_j) \notin E$, $fit(w_i, w_j) = 0$......$O(N^2)$
5. output $\langle W, fit, 2k\rangle$"
Clearly the reduction runs in polynomial time.

II. **Correctness Proof**:

(a) $G \in$ **MaxCut** $\implies F(G) \in$ **P(2-1)**

$$
\begin{aligned}
& G \text{ has a cut } C = (V_1, V_2) \text{ of size } \geq k \\
\implies & \text{Let the corresponding partition of } W \text{ be } (W_1, W_2) \\
\implies & \text{if } w \in W_1, \text{ assign it } \textbf{light} \text{ attribute, if } w \in W_2, \text{ assign it } \textbf{dark} \text{ attribute} \\
\implies & \text{for all edges } (v_i, v_j) \in C,\ fit(w_i, w_j) = 1 \text{ in } F(G) \text{ by reduction, and } w_i, w_j \\
& \text{should have different attributes because } v_i, v_j \text{ are in different partitions} \\
\implies & (w_i, w_j) \text{ is a valid link, so } 2fit(w_i, w_j) = 2 \times 1 \text{ can be added to the enhanced power} \\
\implies & \text{Since } |C| \geq k, \text{ the total enhanced power} \geq 2k \\
\implies & F(G) \in \textbf{P(2-1)}
\end{aligned}
$$

(b) $F(G) \in$ **P(2-1)** $\implies G \in$ **MaxCut**

$$
\begin{aligned}
& \text{If } w_i \text{ is } \textbf{light}, \text{ make } v_i \in V_1, \text{ if } w_i \text{ is } \textbf{dark}, \text{ make } v_i \in V_2. \\
& \text{The total enhanced value} \geq 2k, \\
\implies & \text{there are at least k links whose fitness value} = 1 \\
& \text{For all links } (w_i, w_j), fit(w_i, w_j) = 1, \text{ they should have different attributes,} \\
& \text{and their corresponding edges } (v_i, v_j) \in E \\
\implies & \text{There are at least k edges } (v_i, v_j) \text{ cross } V_1 \text{ and } V_2 \\
\implies & \text{The size of cut } (V_1, V_2) \geq k \\
\implies & G \in \textbf{MaxCut}
\end{aligned}
$$

(2-2) **Define:**

- $W_{1,t}, W_{2,t}$ be the two set after adding w_t, so $W_{1,1} = \{w_1\}, W_{2,1} = \emptyset$
- $power_t$ be the total enhanced power of the algorithm after adding w_t
- OPT_t be the optimal value of this problem considering only $w_1 \dots w_t$

Then I prove by induction on t to show that $2\,power_N \geq OPT_N$, which indicates the algorithm is a 2-approximation:

- **Base Case:** $t = 1$, so $power_1 = OPT_1 = 0 \implies 2 \times power_1 \geq OPT_1$
- **Inductive Step:** Assume $2 \times power_n \geq OPT_n$, I show that $2 \times power_{n+1} \geq OPT_{n+1}$

$$
\begin{aligned}
power_{n+1} &= power_n + max\left(2 \sum_{w \in W_{1,n}} fit(w_{n+1}, w), 2 \sum_{w \in W_{2,n}} fit(w_{n+1}, w)\right) \\
2 \times power_{n+1} &= 2 \times power_n + 2\,max\left(2 \sum_{w \in W_{1,n}} fit(w_{n+1}, w), 2 \sum_{w \in W_{2,n}} fit(w_{n+1}, w)\right) \\
&\geq 2 \times power_n + (2 \sum_{w \in W_{1,n}} fit(w_{n+1}, w) + 2 \sum_{w \in W_{2,n}} fit(w_{n+1}, w)) \\
&= 2 \times power_n + 2 \sum_{i=1}^{n} fit(w_{n+1}, w_i)
\end{aligned}
$$

When w_{n+1} is added to OPT_n, it increases OPT_n by at most 2 times the sum of all fitness values between w_{n+1} and $\{w_1, \dots, w_n\}$. That is, $OPT_{n+1} \leq OPT_n + 2\sum_{i=1}^{n} fit(w_{n+1}, w_i)$. Thus, we have:

$$
\begin{aligned}
2 \times power_{n+1} &\geq 2 \times power_n + 2 \sum_{i=1}^{n} fit(w_{n+1}, w_i) \\
&\geq OPT_n + 2 \sum_{i=1}^{n} fit(w_{n+1}, w_i) \dots \text{by induction hypothesis} \\
&\geq OPT_{n+1} \ \#
\end{aligned}
$$

(3) Because the original problem is optimization problem, I first define the decision problem version of this problem **P(3)**, then I show that **P(2-1)** $\leq_p$ **P(3)**, which implies **P(3)** is NP-hard.

$$\mathbf{P(3\text{-}1)} := \{< W, fit, k > \mid \text{exists a linking method that satisfies RULE1 \& RULE2 s.t. the total enhanced power} \geq k\}$$

I. **Polynomial Time reduction**:

$F =$ "On a instance $\langle W, fit, k\rangle$ of **P(2-1)**:

1. make W' a set of size $2N$ by adding N wands $W^* = \{w_1^*, \ldots, w_N^*\}$
2. for all $w^* \in W^*, w \in W'$, let $fit'(w^*, w) = 0$
3. for all $w_i, w_j \in W, i \neq j$, let $fit'(w_i, w_j) = fit(w_i, w_j)$
4. Output $\langle W', fit', k\rangle$"

Step 1. is $O(N)$, **step 2. and 3.** is $O(M) = O(N^2)$. Clearly the reduction runs in polynomial time.

II. **Correctness Proof**:

(a) $\langle W, fit, k\rangle \in \mathbf{P(2\text{-}1)} \implies \langle W', fit', k\rangle \in \mathbf{P(3)}$

There is a linking method s.t. the total enhanced power $\geq k$
$\implies$ By reduction, all links in G are also in $F(G)$.
$\implies$ Let W_1 be **light**, W_2 be **dark** in G, assume $|W_1| = x,\ |W_2| = N - x$
$\implies$ Let $W^* = \{w_1^*, \ldots, w_N^*\}$, add $N - x$ wands in W^* to W_1, add x wands in W^* to W_2
$\implies$ Now $|W_1| = |W_2| = N$, which satisfy **RULE2**,
and by F, the fitness value between any $w^* \in W^*$ and all other wands is 0
$\implies$ The total enhanced value should remain the same ($\geq k$)
$\implies \langle W', fit', k\rangle \in \mathbf{P(3)}$

(b) $\langle W', fit', k\rangle \in \mathbf{P(3)} \implies \langle W, fit, k\rangle \in \mathbf{P(2\text{-}1)}$

There is a linking method between W_1'(**light**), W_2'(**dark**) of power $\geq k$
$\implies$ Take out all $w^* \in W^*$ from W_1' and W_2'
$\implies$ The remaining wands and links are a valid linking method of power $\geq k$ in $\langle W, fit, k\rangle$
$\implies \langle W, fit, k\rangle \in \mathbf{P(2\text{-}1)}$

(4) Let this problem be **P(4)**, I show that **P(3)** $\leq_p$ **P(4)**, which implies **P(4)** is NP-hard.

I. **Polynomial Time Reduction**:

F = "On input $\langle W, fit, k\rangle$:

1. Computes the maximum FIT_{max} of the M fitness values between each pair of wands.
2. Let $W' = W$, $fit'(w_i, w_j) = FIT_{max} - fit(w_i, w_j)$ for all pairs of wands (w_i, w_j)
3. Let $k' = 2(\frac{N}{2})^2 \times FIT_{max} - k$
4. Output $\langle W', fit', k'\rangle$

Step 1. takes $O(M) = O(N^2)$, while **step 2.** takes $O(N + M) = O(N^2)$, and **step 3.** is $O(1)$. Clearly the reduction run in polynomial time.

II. **Correctness Proof**:

(a) $x \in \mathbf{P(3)} \implies F(x) \in \mathbf{P(4)}$

Since all the fitness values are non-negative, we can assume that each wand is assigned **light** or **dark** and linked to all wands that have the opposite attribute. Thus, W is partitioned into exactly 2 sets W_{light} and W_{dark}. By RULE2, $|W_{light}| = |W_{dark}| = \frac{N}{2}$. So the number of links is $\frac{N}{2} \times \frac{N}{2}$. Let the set of links in x be L, which is also a valid linking method in $F(x)$ by my reduction:

$$x \in \mathbf{P(3)} \implies 2 \sum_{(w_i,w_j)\in L} fit(w_i, w_j) \geq k \;\ldots\ldots ①$$

$$\begin{aligned}
\text{total enhanced power of } F(x) &= 2 \sum_{(w_i,w_j)\in L} fit'(w_i, w_j) \\
&= 2\Big(\sum_{(w_i,w_j)\in L} FIT_{max} - fit(w_i, w_j)\Big) \\
&= 2\Big(\frac{N}{2}\Big)^2 \times FIT_{max} - 2 \sum_{(w_i,w_j)\in L} fit(w_i, w_j) \\
&\leq 2\Big(\frac{N}{2}\Big)^2 \times FIT_{max} - k \text{ (by ①) } \implies F(x) \in \mathbf{P(4)} \ \#
\end{aligned}$$

(b) $F(x) \in \mathbf{P(4)} \implies x \in \mathbf{P(3)}$

By specified condition: **for each wand w, there exists at least one link to w**, this implies each wand w should be assigned a attributes **light** or **dark**. Combined with another condition: **all light wands are linked to all dark wands**, we can infer that a valid linking method in $\mathbf{P(4)}$ must have the following property:

A. the wands are partitioned into exactly 2 sets W_{light} and W_{dark} by their attributes
B. $|W_{light}| = |W_{dark}| = \frac{N}{2}$ by RULE2
C. If w_i and w_j belongs to different sets, there is a link between them

Let L be the set of links in $F(x)$, by my reduction, it is also a valid linking method in x. And by **C.**, $|L| = (\frac{N}{2})^2$.

$$F(x) \in \mathbf{P(4)} \implies 2 \sum_{(w_i,w_j)\in L} fit'(w_i, w_j) \leq 2\Big(\frac{N}{2}\Big)^2 \times FIT_{max} - k$$

$$\implies 2 \sum_{(w_i,w_j)\in L} FIT_{max} - fit(w_i, w_j) \leq 2\Big(\frac{N}{2}\Big)^2 \times FIT_{max} - k$$

$$\implies \cancel{2\Big(\frac{N}{2}\Big)^2 \times FIT_{max}} - 2 \sum_{(w_i,w_j)\in L} fit(w_i, w_j) \leq \cancel{2\Big(\frac{N}{2}\Big)^2 \times FIT_{max}} - k$$

$$\implies -2 \sum_{(w_i,w_j)\in L} fit(w_i, w_j) \leq -k$$

$$\implies 2 \sum_{(w_i,w_j)\in L} fit(w_i, w_j) \geq k \text{ (total enhanced power in x } \geq k)$$

$$\implies x \in \mathbf{P(3)}$$

(5)

I. **Polynomial Time Reduction**:

F = "On input A and W, where $A = (a_1, a_2, \ldots, a_n)$:

1. Compute $S = \sum_{i=1}^{n} a_i$ ......$O(n)$

2. Let $T = (a_1, a_2, \ldots, a_n, S + 2W, 2S)$ ......$O(n)$

3. Output $\langle T \rangle$"

Clearly the reduction run in polynomial time.

II. **Correctness Proof**:

(a) $x \in$ **Subset-Sum** $\implies F(x) \in$ **P(5)**

Exists set $X \subseteq A$ whose sum $= W \implies$ The sum of $A \setminus X = S - W$

$\implies$ Let $Y = X \cup \{2S\}$, then $\displaystyle\sum_{a \in Y} a = W + 2S = \frac{1}{2}\sum_{t \in T} t$

$\implies F(x) \in$ **P(5)** #

(b) $F(x) \in$ **P(5)** $\implies x \in$ **Subset-Sum**

Exists set $X \subseteq T$ whose sum $= 2S + W$, and the sum of $T \setminus X$ also $= 2S + W$

$\implies$ Since $2S + W < (S + 2W) + (2S)$, $(S + 2W)$ and $(2S)$ must in 2 different sets

$\implies$ Take out $(S + 2W)$ and $(2S)$, then the sum of 2 sets become W and $S - W$

$\implies$ Exists subset whose sum equal to $W \implies x \in$ **Subset-Sum** #

Problem F - Band Dream

(1) ***reference**: www.cslog.uni-bremen.de/teaching/summer17/approx-algorithms/resource/lec3.pdf

(1-1)

(a) Just before x_k is added, $|U| = n - (k-1) = n - k + 1$

(b) $\forall\ x \in U$, x must not in any $S \in C$ by the algorithm
$\implies$ OPT must select some sets $S \in F \setminus C$ to cover U, Let these sets be O_k, $O_k \subseteq F \setminus C$

(c) By algorithm, $price(x_k) \le \frac{cost(S)}{|S \cap U|}$, $\forall\ S \in F \setminus C$

$\implies price(x_k) \le \frac{cost(S)}{|S \cap U|}$, $\forall\ S \in O_k$

$\implies price(x_k) \le \min_{S \in O_k} \frac{cost(S)}{|S \cap U|}$

(d) With (a), (b), (c), we can derive:

$$\begin{aligned}
price(x_k) &\le \min_{S \in O_k} \frac{cost(S)}{|S \cap U|} \\
&\le \frac{\sum\limits_{S \in O_k} cost(S)}{\sum\limits_{S \in O_k} |S \cap U|} \text{ .....(\textbf{Inequality 1})} \\
&\le \frac{OPT}{\sum\limits_{S \in O_k} |S \cap U|} \text{ .....(}O_k\text{ is just part of OPT)} \\
&\le \frac{OPT}{|U|} \text{ .....(\textbf{Inequality 2})} \\
&= \frac{OPT}{n-k+1}
\end{aligned}$$

- Proof of **Inequality 1**:

$$\text{Let } \min_{S \in O_k} \frac{cost(S)}{|S \cap U|} = \rho \implies \rho \le \frac{cost(S)}{|S \cap U|},\ \forall\ S \in O_k$$

$$\implies \rho\,|S \cap U| \le cost(S),\ \forall\ S \in O_k$$

$$\implies \rho \sum_{S \in O_k} |S \cap U| \le \sum_{S \in O_k} cost(S)$$

$$\implies \rho \le \frac{\sum_{S \in O_k} cost(S)}{\sum_{S \in O_k} |S \cap U|}\ \#$$

- Proof of **Inequality 2**:

By my definition of O_k, O_k should cover U.
Thus, for every element $x_i \in U$, there exists a $S \in O_k$ such that $x_i \in S$

$$\implies |\bigcup_{S \in O_k} (S \cap U)\ | = |U| \implies \sum_{S \in O_k} |S \cap U| \ge |U|$$

$$\implies \frac{1}{\sum_{S \in O_k} |S \cap U|} \le \frac{1}{|U|}\ \#$$

(1-2) **Time Complexity Analysis**:

i. Initialize C and U as boolean array: $O(n)$
ii. In each iteration, linearly search for best S. For each S_i, computing $|S \cap U|$ takes $O(|S_i|) = O(n)$ time. So the time complexity of this procedure is $O(n^2)$
iii. $C \cup S$ takes $O(1)$
iv. $U \setminus S$ takes $O(n)$
v. In each iteration at least one element is taken out from U, so we have at most n iteration
vi. Overall time complexity: $O(n) + n \times O(n^2) = O(n^3) \implies$ polynomial time.

Approximation Factor Proof:
We can observe that the total cost of this approximation algorithm is exactly $\sum_{k=1}^{n} price(x_k)$, so we need to prove that $\frac{\sum_{k=1}^{n} price(x_k)}{OPT} \leq ln(n) + O(1)$.

$$\text{price}(x_k) \leq \frac{\text{OPT}}{n-k+1}$$
$$\implies \sum_{k=1}^{n} \text{price}(x_k) \leq \sum_{k=1}^{n} \frac{\text{OPT}}{n-k+1} = \sum_{i=1}^{n} \frac{OPT}{i} \leq OPT((\int_1^n \frac{dx}{x}) + c) = OPT\,(ln(n) + c)$$
$$\implies \frac{\sum_{k=1}^{n} price(x_k)}{OPT} \leq ln(n) + O(1) \text{ (c is a constant so $c = O(1)$)}$$

(2) **reference:** www.cs.dartmouth.edu/ ac/Teach/CS105-Winter05/Notes/wan-ba-notes.pdf

(2-1)

Polynomial Time Reduction:

1. Initialize emptyset F, cost function $cost()$
2. For all $M_i \in M$, add M_i to F, and $cost(M_i) = |M_i|$......$O(n)$
3. For all $2\,C_2^n$ pairs of strings M_i, M_j where $i \neq j$, we need to generate all possible "merges" of M_i and M_j. This can be done by enumerating all possible length k of the overlapping part, where $1 \leq k \leq min(|M_i, M_j|) - 1 < l$, and each time we should check whether the last k characters of M_i is the same as the first k characters of M_j. If it is a valid merge, let this new string be w_{ij}^k, we define $set(w_{ij}^k)$ as $\{s\ |s \in M$ and s is a substring of $w_{ij}^k\}$, to determine $set(w_{ij}^k)$, we iterate through all string in M and utilize some string matching algorithm(e.g. KMP) to check if a string is a substring of w_{ij}^k. After finishing the construction of $set(w_{ij}^k)$, add $set(w_{ij}^k)$ to F and let $cost(set(w_{ij}^k)) = |w_{ij}^k| = len(M_i) + len(M_j) - k$.
4. Let $X = M$
5. Output $\langle X, F, cost\rangle$

The part **1.** is $O(1)$ and part **2.** is $O(n)$ clearly, while the time complexity of part **3.** is analyzed as follow: There are $2\,C_2^n = n^2 - n$ possible pairs of strings to merge, since for each pair (M_i, M_j), both M_i and M_j can be the prefix of merged string. Then to check all possible merge method of two strings, the algorithm enumerates k, which is bounded by l. And for each k, the string comparision occurs k times, which is also bounded by l. For the construction of $set(w_{ij}^k)$, assume the string matching algorithm runs in linear time(e.g. KMP), then we can do it in $n\,O(l)$ time. Thus, the overall time complexity of part **3.** is:

$$(n^2 - n) \times l \times l \times n\,O(l) = O(n^3 l^3)$$

So the overall complexity is $O(1) + O(n) + O(n^3l^3) = O(n^3l^3)$, which is polynomial time.

(3) **Goal:** Show that $OPT_{SetCover} \leq 2OPT_{string}$

Consider an **optimal** string OPT_{string} that covers M, so we can order all the strings in M by their left most occurence in OPT_{string}, let this order be $(m_1, m_2, \ldots, m_n)$. Though the indices of this ordering may not be the same as the original indices in M, in the following proof I will use its indices for convenience and it does not affect the correctness.

Now I want to partition this sequence of strings into groups such that each group corresponds to some element in the F of the **Set Cover** instance after reduction. Let l_i denote the indices of the first string in the i^{th} group, and define r_i to be the highest possible index such that m_{l_i} overlaps m_{r_i}. By the definition, $l_1 = 1$ clearly, and $l_2 = r_1 + 1$, $l_3 = r_2 + 1$......until $r_t = n$. t is the number of groups.

Since m_{l_i} overlaps m_{r_i} by some k_i number of characters, $w^{k_i}_{l_i r_i}$ is a possible "merge" of m_{l_i} and m_{r_i}, and $set(w^{k_i}_{l_i r_i}) = \{m_{l_i}, m_{l_i+1}, \ldots, m_{r_i}\} \in F$, with cost $|w^{k_i}_{l_i r_i}|$ by my reduction. By my definition of l_i and r_i, these groups should cover all strings in M, therefore, $\{set(w^{k_1}_{l_1 r_1}), \ldots, set(w^{k_t}_{l_t r_t})\}$ is a valid set cover in the **Set Cover** instance after reduction, whose cost is $\sum_{i=1}^{t} |w^{k_i}_{l_i r_i}| \geq OPT_{SetCover}$.

By observation, each character in OPT_{string} is "covered" by at most 2 groups. This comes from my definition of group and the given assumption that "there is no M_i is sub-string of M_j, for $i \neq j$". There are 2 key inference to say:

(a) m_{r_i} **ends before** $m_{l_{i+1}}$ **ends**: By definition of group $l_{i+1} = r_i + 1 \implies m_{r_i}$ begins before $m_{l_{i+1}}$. So if m_{r_i} end after $m_{l_{i+1}}$ then $m_{l_{i+1}}$ is a sub-string of m_{r_i}, which contradicts to the assumption.

(b) $m_{l_{i+2}}$ **starts after** $m_{l_{i+1}}$ **ends**: If $m_{l_{i+2}}$ starts before $m_{l_{i+1}}$ ends, then $m_{l_{i+2}}$ overlaps $m_{l_{i+1}}$, then by definition of group, they should be in the same group, which forms a contradiction.

Thus, $m_{l_{i+2}}$ starts after m_{r_i} ends, means the i^{th} group must not overlap the $(i+2)^{th}$ group, which indicates **each character in OPT_{string} is "covered" by at most 2 groups ($w^{k_i}_{l_i r_i}$)**. So we can derive that:

$$OPT_{SetCover} \leq \sum_{i=1}^{t} |w^{k_i}_{l_i r_i}| \leq 2OPT_{string}$$
$$\implies OPT_{SetCover} \leq 2OPT_{string}$$

Combined with the result of (1-2), with *GreedySetCover* algorithm, we have:

$ANS_{GreedySetCover} \leq (ln(n) + O(1))OPT_{SetCover} \leq 2(ln(n) + O(1))OPT_{string}$
Since $2 \times O(1) = O(1)$, $ANS_{GreedySetCover} \leq (2ln(n) + O(1))OPT_{string}$ #

$\rho(n) = 2ln(n) + O(1) \implies (2ln(n) + O(1))$-approximation.