Finding Counterfeit Coin by Pairwise Comparisons

May 16, 2015

1 Introduction

There is a classic puzzle about counterfeit coin problem: a man has 12 coins among which there has a counterfeit coin, which can only be told apart by its weight. How can one tell in not more than three weighings and determine which one is a counterfeit coin [1].

The problem was so popular that have many other variants[2], for example, the weight of counterfeit coin is heavy or light are known; given an extra coin known to be real; or even answer the question by using a spring balance i.e. a weighing device that will return the exact weight. Halbeisen[3] generize this problem when we are allowed to use more than 1 balances and consider more than one counterfeit coin.

Here we extend this question by a new direction: Now we have `a` real coins and `b` counterfeit coins ($\mathtt{a}, \mathtt{b} > 0$). The real coins are all the same weight and so are the counterfeit ones, but weight of a fake coin is less than weight of a real coin. Each time we compare only two coins have the same weight or not. In this case, assume `a`, `b` are known. We want to study the smallest number of comparisons that we can guarantee to find

1. a fake coin
2. a real coin
3. a fake coin and a real coin(need to know which one is real)
4. a fake coin and a real coin(don't need to know which one is real)

Note that the solution to Problem 3 could solve Problem 4,(solution for Problem 4)+(comparison of this pair) could solve Problem 3. That is, $ans(Problem4) \leq ans(Problem3) \leq ans(Problem4) + 1$

We shall discuss Problem 4 first, then solve Problem 1,2,3 similarly.

Problem 4 can transfer to a new circumstances: there is an engine broken because of burned wire, and we need to find a pair of new electric wires to fix it, but all the wire are mass up, we only know that we have `a` positive wires and `b` negative wires in the beginning, each time we can choose 2 of them to connect to the engine and check if they are the same Electrical polarity (not work) or not (work). The object is minimize the worst case of testing times that we can fix the engine.

2 An Equivalence Graph Problem

We can present a set of comparisons as a graph. Vertice are the coins, edges are the comparisons.

Definition 1. *If comparing the edges of a graph results in at least one unbalanced edge in any condition. We say this graph is a testing solution, feasible. Otherwise it's infeasible. An optimal testing solution(denoted by OTS) is a solution with minimum number of edges(comparisons).*

An OTS can presented by an integer partition as well, which is covered in Section 3.

Idea

Consider a graph $\mathcal{G}(\mathtt{a},\mathtt{b})$ contains two disjoint cliques, one is $K_{\mathtt{a}}$, another is $K_{\mathtt{b}}$. The edges in $\mathcal{G}(\mathtt{a},\mathtt{b})$ means comparing the pair would results in balanced. Then we want to find a graph g (number of vertices $\leq \mathtt{a}+\mathtt{b}$) which is not a subgraph of $\mathcal{G}(\mathtt{a},\mathtt{b})$.

Lemma 2.1. *We can guarantee a graph g (number of vertices $\leq \mathtt{a}+\mathtt{b}$) has a unbalanced edge iff g is not a subgraph of $\mathcal{G}(\mathtt{a},\mathtt{b})$.*

Proof. Denote the real coins as '+' and the fake ones as '$-$'.

We compare edges of g. Vertices of g are subset of vertices of $\mathcal{G}(\mathtt{a},\mathtt{b})$. If g is a subgraph of $\mathcal{G}(\mathtt{a},\mathtt{b})$, assign '+'s and '$-$'s from $\mathcal{G}$ to g , g results in all balance. If it's not a subgraph of $\mathcal{G}(\mathtt{a},\mathtt{b})$, then at least one edge results in unbalance. We can prove this by contradiction. If g results in all balance, edges in g are either ('+','+') or ('$-$','$-$'), subset of edges of $\mathcal{G}(\mathtt{a},\mathtt{b})$, thus g is a subgraph of $\mathcal{G}(\mathtt{a},\mathtt{b})$. □

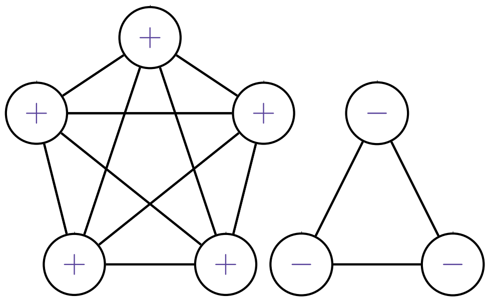

Figure 1: $\mathcal{G}(3,5)$

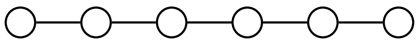
Figure 2: a trivial solution

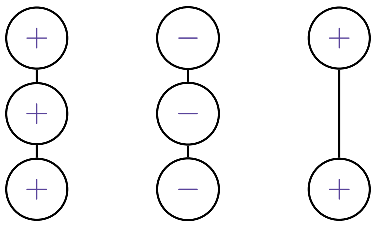

Figure 3: not a solution since it's a subgraph of $\mathcal{G}(3,5)$, we can assign '+'s and '-'s as in the figure such that all edges are balanced.

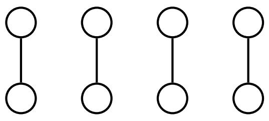
Figure 4: an optimal solution.

Intuitively, if finding unbalance in a comparison, the game ends. But sometimes we can point a pair is unbalance after k comparisons(k is the answer) before the unbalance appears. The following shows $k_{min} = t_{min} - 1$, which means the answer for Problem 4 is $t_{min} - 1$

Lemma 2.2. $t_{min} = k_{min} + 1$

Proof. We divide the proof of the equation into two inequations

If we could point out a pair e is unbalanced through the optimal graph F(with k_{min} comparisons). Consider a graph with these k_{min} edges + e which is not a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$

$$t_{min} \leq k_{min} + 1$$

For a optimal graph E(with t_{min} edges) which is not a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$. Let $E' = (E$ remove an arbitrary edge $e)$, then through E'. If E' has unbalance comparison, we conclude the one is unbalanced, otherwise we could conclude e is unbalanced. Either case shows E' is enough to end the game.

$$k_{min} \leq t_{min} - 1$$

From the above inequations, the lemma holds.

□

Goal

We want to find a graph $g(\mathtt{a}, \mathtt{b})$ that has minimum edges and $g(\mathtt{a}, \mathtt{b})$ is not a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$. Through testing $g(\mathtt{a}, \mathtt{b})-$an arbitrary edge e, we can point out a pair is unbalanced and it's optimal.

Definition 2. *$\tau(\mathtt{a}, \mathtt{b})$ is the number of edges of $OTS(\mathtt{a}, \mathtt{b})$*

For example, $\tau(1, 2) = 2$, since testing a pair may result in balanced. On the other hand, testing two non-duplicate pair, one of them must results in unbalanced.

In this case, testing a pair is enough to point out a real coin and a fake one.

Denote the coins we test as $coin_a$ $coin_b$, the coin we didn't test as $coin_c$ If $coin_a$ and $coin_b$ is not balanced, then point out $coin_a$ and $coin_b$. Otherwise, point out $coin_a$ and $coin_c$

Theorem 2.3. *Any graph with $n \leq \mathtt{a} + \mathtt{b}$ vertices and $m \leq \lfloor(\mathtt{a} + \mathtt{b})/2\rfloor - 1$ edges $(0 < \mathtt{a} \leq \mathtt{b})$ is always a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$ (or equivalently, a subgraph of $\mathcal{G}(\mathtt{b}, \mathtt{a})$).*

Proof. We shall prove this theorem by induction.

(Basis Case:) If $\mathtt{a} = \mathtt{b} = 1$, then $\lfloor(\mathtt{a} + \mathtt{b})/2\rfloor - 1 = 0$. Any graph with $n \leq 2$ vertices and $m \leq 0$ edges (i.e., no edges) is always a subgraph of $\mathcal{G}(1, 1)$.

(Inductive Case:) Suppose that the theorem holds for all $\mathtt{a} + \mathtt{b} \leq k$. Our target is to show that the theorem also holds for the case $\mathtt{a} + \mathtt{b} = k + 1$ with $\mathtt{a} \leq \mathtt{b}$. Consider a graph G with $n \leq k + 1$ vertices and with $m \leq \lfloor(k + 1)/2\rfloor - 1$ edges.

1. If G is connected, then $n \leq m + 1 \leq (k + 1)/2 \leq b$, which is a subgraph of K_b, and thus a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$.

2. Otherwise, G is not connected. If G has no edges, then G is obviously a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b}, 0)$ since G has at most $\mathtt{a} + \mathtt{b}$ vertices. Else, let C be the connected component of G with the largest number n' of vertices (so that $n' \geq 2$. Then, the number of edges in C is at least $n' - 1$. To complete the proof, it is sufficient to show that $G - C$ is a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b} - n')$, as we can map C as a subgraph in K_b.

The number of vertices in $G - C$ is $k + 1 - n' = \mathtt{a} + (\mathtt{b} - n')$, and the number of edges of $G - C$ is at most

$$\begin{aligned} m - (n' - 1) \leq \lfloor (k+1)/2 \rfloor - n' &= \lfloor (k+1)/2 - n' \rfloor \\ &= \lfloor (k+1-n')/2 - n'/2 \rfloor \\ &\leq \lfloor (k+1-n')/2 - 1 \rfloor \\ &= \lfloor (k+1-n')/2 \rfloor - 1 = \lfloor (\mathtt{a} + (\mathtt{b} + n'))/2 \rfloor - 1. \end{aligned}$$

By induction hypothesis, $G - C$ is a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b} - n')$, and consequently G is a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$.

In all cases, G is a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$. This completes the proof of the induction case, so that by the principle of mathematical induction, the theorem follows. □

Corollary 2.3.1. *for all odd number* $\mathtt{a}, \mathtt{b}$*,* $\tau(a, b) = \frac{\mathtt{a}+\mathtt{b}}{2}$

Consider a graph $g(a, b)$ that have $\frac{a+b}{2}$ components, each components have only two nodes. then $g(\mathtt{a}, \mathtt{b})$ is not a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$, and by Thm2.2, it is the optimal solution.

Theorem 2.4. *For every* a, b*, we have a 2-approximation solution of* $\tau(\mathtt{a}, \mathtt{b})$*.*

Consider a line with $max(\mathtt{a}, \mathtt{b})$ nodes as in Figure 2. The graph is not a subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$, and by Thm2.2, it is a 2-approximation solution.

3 Integer Partition

To find $g(\mathtt{a}, \mathtt{b})$ with minimal number of edges, it's obvious that $g(\mathtt{a}, \mathtt{b})$ should not form a cycle

Proof. if $g(\mathtt{a}, \mathtt{b})$ form a cycle, remove an edge, the number of vertices each component remains same, whether $g'(\mathtt{a}, \mathtt{b})$ is the subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$ is equivalent to whether $g(\mathtt{a}, \mathtt{b})$ is the subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$ □

Definition 3. *We define a multiset avoids x iff its subset-sums has no x, a partition avoids x iff its parts avoids x*

Previously, we didn't consider the coins not compared for $g(\mathtt{a}, \mathtt{b})$. Now we add a single vertex for each coin not compared into $g(\mathtt{a}, \mathtt{b})$. Then the number of vertices of $g(\mathtt{a}, \mathtt{b})$ is $\mathtt{a} + \mathtt{b}$.

Theorem 3.1. *the problem would be transformed to find a partition of* $\mathtt{a} + \mathtt{b}$ *with maximal number of parts that avoids* $\mathtt{a}$

Proof. Let $t =$ number of edge of $g(\mathtt{a}, \mathtt{b})$, n as number of componets in $g(\mathtt{a}, \mathtt{b})$, M as the set of components

since $g(a, b)$ doesn't contain any cycle, $t = \mathtt{a} + \mathtt{b} - n$, to find minimal t is equivalent to find maximal n such that $g(\mathtt{a}, \mathtt{b})$ is not the subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$

if $g(\mathtt{a}, \mathtt{b})$ is the subgraph of $\mathcal{G}(\mathtt{a}, \mathtt{b})$, since each component is conected, each one is either in K_a or K_b, there exist a subset of M in K_a the number of vertices of M = a ,e.g there exist a subset of m is equal to a

if $g(\mathtt{a}, \mathtt{b})$ is not the subgraph $\mathcal{G}(\mathtt{a}, \mathtt{b})$, there must not a subset of m is equal to a, since if there subset of m is equal to a let these componets in K_a and others in K_b, then $g(\mathtt{a}, \mathtt{b})$ is the subgraph $\mathcal{G}(\mathtt{a}, \mathtt{b})$ thus raise contradiction

the sum of m is $\mathtt{a} + \mathtt{b}$, so the theorem holds

□

Definition 4. *We can present an optimal solution as a partition of* $\mathtt{a} + \mathtt{b}$*.*

$\rho(\mathtt{a}, \mathtt{b})$ *is the maximal number of parts of* $\mathtt{a} + \mathtt{b}$ *avoids* $\mathtt{a}$

3.1 Graph Integer partition double-way Transform

Transforming acyclic graph to a integer partition is trivial, just make each connected component a integer(the number of vertices). Inverse way transformation is also easy, make a p_i-vertices tree for each integer part p_i.

Remark. $\tau(\mathtt{a},\mathtt{b}) = \mathtt{a}+\mathtt{b}-\rho(\mathtt{a},\mathtt{b})$

Theorem 3.2. *Optimal partition for* $\mathtt{a} = 2$

$$\rho(2,\mathtt{b}) = \begin{cases} 1+\frac{\mathtt{b}}{3} & \textit{if}\ \mathtt{b} \equiv 0 \\ 1+\frac{\mathtt{b}-1}{3} & \textit{if}\ \mathtt{b} \equiv 1 \\ 1+\frac{\mathtt{b}+1}{3} & \textit{if}\ \mathtt{b} \equiv 2 \end{cases} \pmod 3$$

Proof. since there is no a in subset sums of the partition P, there is at most one '1' and no '2' in P, $|P| \le \lfloor\frac{\mathtt{b}+2}{3}\rfloor$

for $\mathtt{b} \equiv 0 \pmod 3$, there exist an optimal partition $P = \{1, 4, 3, ...\}$('...' are '3's) that reaches optimal bound and avoids $\mathtt{a}$

for $\mathtt{b} \equiv 1 \pmod 3$, there exist an optimal partition $P = \{1, 5, 3, ...\}$('...' are '3's) that reaches optimal bound and avoids $\mathtt{a}$

for $\mathtt{b} \equiv 2 \pmod 3$, there exist an optimal partition $P = \{1, 3, 3, ...\}$('...' are '3's) that reaches optimal bound and avoids $\mathtt{a}$

□

3.2 Exact Algorithm

```
$R = \{\mathtt{a}+\mathtt{b}\}$
for each partition $P$ of the number $\mathtt{a}+\mathtt{b}$ which avoids $\mathtt{a}$ do
  if $|P| > |R|$ then
    $R = P$
  end if
end for
output $R$
```

check P's avoidence by Dynamic-Programming as solving discrete knapsack problem, keep avoidence vector when enumerating.

Denote the number of partitions of n as $p(n)$

Denote the number of partitions avoids a of n as $p_{aoivds\ a}(n)$

Time Complexity:

$$O(a \times p_{aoivds\ \mathtt{a}}(\mathtt{a}+\mathtt{b})) \le O(\mathtt{a} \times p(\mathtt{a}+\mathtt{b})) \le O(\mathtt{a}\frac{exp(\sqrt{\frac{2(\mathtt{a}+\mathtt{b}))}{3}})}{\mathtt{a}+\mathtt{b}})$$

3.3 Results

$\rho(\mathtt{a},\mathtt{b})$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	1	2	2	3	3	4	4	5	5	6	6	7	7	8	8	9	9	10	10
2	1	2	2	2	3	3	3	4	4	4	5	5	5	6	6	6	7	7	7	8
3	2	2	3	3	4	3	5	4	6	5	7	6	8	7	9	8	10	9	11	10
4	2	2	3	4	4	4	4	4	5	5	5	5	6	6	6	7	7	7	8	8
5	3	3	4	4	5	5	6	5	7	5	8	6	9	7	10	8	11	9	12	10
6	3	3	3	4	5	6	6	6	6	6	6	6	7	7	7	7	7	7	7	8
7	4	3	5	4	6	6	7	7	8	7	9	7	10	7	11	8	12	9	13	10
8	4	4	4	4	5	6	7	8	8	8	8	8	8	8	8	8	9	9	9	10
9	5	4	6	5	7	6	8	8	9	9	10	9	11	9	12	9	13	9	14	10
10	5	4	5	5	5	6	7	8	9	10	10	10	10	10	10	10	10	10	10	10
11	6	5	7	5	8	6	9	8	10	10	11	11	12	11	13	11	14	11	15	11
12	6	5	6	5	6	6	7	8	9	10	11	12	12	12	12	12	12	12	12	12
13	7	5	8	6	9	7	10	8	11	10	12	12	13	13	14	13	15	13	16	13
14	7	6	7	6	7	7	7	8	9	10	11	12	13	14	14	14	14	14	14	14
15	8	6	9	6	10	7	11	8	12	10	13	12	14	14	15	15	16	15	17	15
16	8	6	8	7	8	7	8	8	9	10	11	12	13	14	15	16	16	16	16	16
17	9	7	10	7	11	7	12	9	13	10	14	12	15	14	16	16	17	17	18	17
18	9	7	9	7	9	7	9	9	9	10	11	12	13	14	15	16	17	18	18	18
19	10	7	11	8	12	7	13	9	14	10	15	12	16	14	17	16	18	18	19	19
20	10	8	10	8	10	8	10	10	10	10	11	12	13	14	15	16	17	18	19	20

$\tau(\mathtt{a},\mathtt{b})$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1	2	2	3	3	4	4	5	5	6	6	7	7	8	8	9	9	10	10	11
2	2	2	3	4	4	5	6	6	7	8	8	9	10	10	11	12	12	13	14	14
3	2	3	3	4	4	6	5	7	6	8	7	9	8	10	9	11	10	12	11	13
4	3	4	4	4	5	6	7	8	8	9	10	11	11	12	13	13	14	15	15	16
5	3	4	4	5	5	6	6	8	7	10	8	11	9	12	10	13	11	14	12	15
6	4	5	6	6	6	6	7	8	9	10	11	12	12	13	14	15	16	17	18	18
7	4	6	5	7	6	7	7	8	8	10	9	12	10	14	11	15	12	16	13	17
8	5	6	7	8	8	8	8	8	9	10	11	12	13	14	15	16	16	17	18	18
9	5	7	6	8	7	9	8	9	9	10	10	12	11	14	12	16	13	18	14	19
10	6	8	8	9	10	10	10	10	10	10	11	12	13	14	15	16	17	18	19	20
11	6	8	7	10	8	11	9	11	10	11	11	12	12	14	13	16	14	18	15	20
12	7	9	9	11	11	12	12	12	12	12	12	12	13	14	15	16	17	18	19	20
13	7	10	8	11	9	12	10	13	11	13	12	13	13	14	14	16	15	18	16	20
14	8	10	10	12	12	13	14	14	14	14	14	14	14	14	15	16	17	18	19	20
15	8	11	9	13	10	14	11	15	12	15	13	15	14	15	15	16	16	18	17	20
16	9	12	11	13	13	15	15	16	16	16	16	16	16	16	16	16	17	18	19	20
17	9	12	10	14	11	16	12	16	13	17	14	17	15	17	16	17	17	18	18	20
18	10	13	12	15	14	17	16	17	18	18	18	18	18	18	18	18	18	18	19	20
19	10	14	11	15	12	18	13	18	14	19	15	19	16	19	17	19	18	19	19	20
20	11	14	13	16	15	18	17	18	19	20	20	20	20	20	20	20	20	20	20	20

4 Futher Speedup

Previously, to find OTS, we enumerate all partitions of $\mathtt{a} + \mathtt{b}$. It cost much time. Discovering from tables in Results 3.3. We found the cycle property. Take $\mathtt{a} = 5$ as example.

$\tau(\mathtt{a}, \mathtt{b})$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
5	3	4	4	5	5	6	6	8	7	10	8	11	9	12	10	13	11	14	12	15

From $\mathtt{b} \geq 9$, $\tau(\mathtt{a}, \mathtt{b})$ forms cycle of length 2. In general, when $\mathtt{b}$ is large enough $\tau(\mathtt{a}, \mathtt{b})$ would forms into cycles. Using this property to improve our algorithm for the case $\mathtt{b}$ is much larger than $\mathtt{a}$. The improved algorithm runs in time complexity with no $\mathtt{b}$.

Definition 5. *d is the smallest non-factor of* $\mathtt{a}$

Lemma 4.1. *$\rho(a, b)$ has lower-bound $\lceil \frac{b}{d} \rceil$*

Proof. To proof a lower-bound, we show a case whose number of partitions=$\lceil \frac{b}{d} \rceil$

The case is trivial, divide $a + b$ into $\lceil \frac{b}{d} \rceil - 1$ instances of d and a $a + b - d\lceil \frac{b}{d} \rceil - 1$ (denoted by s)

Prove the case avoids $\mathtt{a}$ by contradiction

If the subset contains s since $s > a \implies$ contradiction

If the subset doesn't contains s, it contains only several d, but since $d \not| a \implies$ contradiction

$\square$

Lemma 4.2. *If a multiset S avoids a and composed only by factors of a, $|S| < a$*

Proof. We shall prove this by contradiction

We denote the number of i in S as X_i, that is, we have X_i 'i'(s) in S

, the first k elements of S as S_k. S_0 is the empty set.

We define set $U(s)$ for a multiset s as subset-sums$\leq a$ of s. For example, $\mathtt{a} = 5$, $U(1, 3, 3) = ignore_{>5}(\{0, 1, 3, 4, 6, 7\}) = \{0, 1, 3, 4\}$

Note that for any s, $U(s)$ always include zero.

We divide the proof into two sections.

1. $\forall k, |U(P_{k+1})| > |U(P_k)|$

Denote $U(P_{k+1})$ as u' and $U(P_k)$ as u.

Prove this by contradiction, if $|u'| = |u|$ for some k, let w is the k+1th element of S,then Since $u \subseteq u'$, $u = u'$

$\forall i \geq w, i \in u \implies i + w \in u' \implies i + w \in u$

Since U always include 0, $0 \in u \implies tw \in u \forall t$

w is a factor of $\mathtt{a}$, thus $w \in u$, contradicts S avoids $\mathtt{a}$

2. We discover $U(S_0)$ to $U(S_{|S|})$, since $U(S_0) = \{0\}$, each time U at least a new element. If $|S| \geq \mathtt{a}$, then $U(S)$ must be $\{0, 1, ..., \mathtt{a}\}$, contradicts S avoids $\mathtt{a}$

$\square$

Theorem 4.3. $\forall \mathtt{b} > d(d + 1)(\mathtt{a} - 1) + (\mathtt{a} - (\mathtt{a} \mod d)$*, an optimal partition contains at least $\lfloor \frac{\mathtt{a}}{d} \rfloor$ instances of d*

Proof. Let S is a partition of $\mathtt{a} + \mathtt{b}$ avoids $\mathtt{a}$, X_i is the number of times i appears in S and $X = \sum_i X_i$

Assume $X_d < \lfloor \frac{\mathtt{a}}{d} \rfloor$,

Then

$$X = \sum_{i<d} X_i + X_d + \sum_{i>d} X_i$$

Since $\sum_i X_i \times i = \mathtt{a} + \mathtt{b}$, to maximize X, we use as much small number as possible. Hence

$$\begin{aligned} X &\leq \mathtt{a} - 1 + \lfloor \frac{\mathtt{a}}{d} \rfloor + \lfloor \frac{\mathtt{b} - (\mathtt{a} - (\mathtt{a} \mod d))}{d+1} \rfloor \\ &\leq \mathtt{a} - 1 + \frac{\mathtt{a}}{d} + \frac{\mathtt{b} - (\mathtt{a} - (\mathtt{a} \mod d))}{d+1} \end{aligned}$$

$\forall \mathtt{b} > d(d+1)(\mathtt{a}-1) + (\mathtt{a} - (\mathtt{a} \mod d))$,
$X < \mathtt{a} + \lfloor \frac{\mathtt{a}}{d} \rfloor + \mathtt{a}d - d \leq \lceil \frac{\mathtt{b}}{d} \rceil$. contradicts lemma 4.1

Hence S is not optimal □

Theorem 4.4. *$\forall \mathtt{b} > d(d+1)(\mathtt{a}-1) + (\mathtt{a} - (\mathtt{a} \mod d)$, $\rho(\mathtt{a}, \mathtt{b}+d) = \rho(\mathtt{a}, \mathtt{b}) + 1$*

Proof. S is a optimal partition of a+b avoids a ($b > d(d+1)(a-1) + (a - (a \mod d))$

$S' = S$+'d' is a partition of a+b+d

We divide the proof into avoidance part and optimal part

(Avoidance) Prove this by contradiction

If S' has subset-sum $\mathtt{a}$, S must has subset-sum $\mathtt{a}$ or $\mathtt{a} - d$. The former is not possible from definition.

If the later happens, S has subset-sum $\mathtt{a}$ since it has at least $\lfloor \frac{a}{d} \rfloor$ instances of 'd', and $\mathtt{a} - d$ couldn't has all of them. Also contradict definition.

S' avoids $\mathtt{a}$.

(Optimal) prove by contradiction

If S' is not optimal there exist an optimal partition P' that $|P'| > |S'|$ and has no subset-sum=a

P' has at least one d, let $P = P' - d(sum(P) = sum(S))$ then P has no subset-sum=a and $|P| = |P'| - 1 > |S'| - 1 = |S|$, which contradicts S is optimal.

S' is optimal.

Thus the theorem holds. □

Corollary 4.4.1. *for a constant $\mathtt{a}$, $\rho(\mathtt{a}, \mathtt{b})$ forms cycle of length d if $\mathtt{b}$ is sufficiently large*

4.1 Algorithm using cycle property

let $K = \mathtt{a} + \mathtt{b} - kd > d(d+1)a$ (choose maximal k, e.g. minimal K)

```
$R = \{\mathtt{a} + \mathtt{b}\}$
for each partition $P$ of the number K which avoids $\mathtt{a}$ do
  if $|P| > |R|$ then
    $R = P$
  end if
end for
output $R$+k instances of 'd'
```

check P's avoidence by Dynamic-Programming as solving discrete knapsack problem, keep avoidence vector when enumerating.

Time Complexity:

Denote the number of partitions of n as $p(n)$

Denote the number of partitions avoids a of n as $p_{aoivds\ a}(n)$

$$O(k+a\times p_{aoivds\ \mathtt{a}}(K)) = O(\mathtt{a}\times p(K)) = O(\mathtt{a}\frac{exp(\sqrt{\frac{2(1+d(d+1)(\mathtt{a}-1)+\mathtt{a})}{3}})}{1+d(d+1)(\mathtt{a}-1)+\mathtt{a}}) = O(\frac{exp(\sqrt{\frac{2(1+d(d+1)(\mathtt{a}-1)+\mathtt{a})}{3}})}{d^2})$$

5 extended problem

Let's consider problem 1, 2 mentioned in the introduction. We shall solve Problem 1 first, then similarly solve Problem 2.

Optimal testing solution for finding a fake coin

Similar to Section 2, we could present a solution as a graph.

If after testing a graph, we could solve problem 1. This graph is a testing solution for fake (TS^-), feasible. Otherwise it's infeasible.

Definition 6. *Optimal testing solution for fake (OTS^-) is TS^- with minimum number of comparison.*

deonte the number of edges of OTS^- as $\tau^-(a,b)$.

Note that $OTS(a,b)$ could find a real and a fake coin. Thus $\tau(a,b)$ is an upper bound of $\tau^+(a,b)$ and $\tau^-(a,b)$.

A $TS^-(a,b)$ better than $OTS(a,b)$ must use another strategy, which is, though all comparisons result in balance we could still find a '-'.

Before solving TS^-, we observe how a solution works.

When claimming a solution, after these comparisons, there are two cases

Case 1. there is an unbalanced

Case 2. they are all balanced

If **Case 1.** happens then the solution works. If **Case 2.** happens, we must claim a fake coin, and this coin couldn't be a real one in any condition.

We can still present a solution as a integer partition.

Example for **Case 2.** let (a,b)=(2,6) the optimal solution is [3,3,'1',1] the '1' is the fake one

Denote the part of the fake coin as $i(i \leq \mathtt{a})$. Beyound the remainning $\mathtt{a}+\mathtt{b}-i$ parts,

(1)they must avoid $\mathtt{a}$, since if there's a subset sum $\mathtt{a}$, assign the subset '+', this claim is wrong

(2)they must contain $\mathtt{a}-i$ or equivalently contain $\mathtt{b}$, since there must exist a condition such that the coin is fake.

Definition 7. *For convience, we defone the number of edges of a partition of n as number of edges of the graph transformed in Section 3. $Q(n,a,b)$ as the number of edges of optimal partition avoids a, but contains b*

Note that $Q(n,a,b) \geq \tau(a,n-a)$

There are only **Case 1.** and **Case 2.**, and for **Case 2.**, i=1 a for TS^-, thus we get the following lemma.

Lemma 5.1. $\tau^{-}(\mathtt{a},\mathtt{b}) = min\{\tau(\mathtt{a},\mathtt{b}), \{i-1+Q(\mathtt{a}+\mathtt{b}-i,\mathtt{a},\mathtt{b})|1 \le i \le \mathtt{a}\}\}$

We are ready to solve OTS^{-} now.

Theorem 5.2. *$OTS^{-}(\mathtt{a},\mathtt{b}) = optimal\{OTS(a,b), \{$a tree of i-1 nodes and $OTS(a,b-i)|1 \le i \le a\}\}$*

Proof. We divide the proof into feasible part and optimal part.
(Feasible Part:)
$OTS(a,b)$ could find a fake coin by definition.
For {a tree of i nodes and $OTS(a,b-i)|1 \le i \le a\}$}, the i part is real only if
1. remainning parts contain b-i or
2. there is at least one unbalanced pair.
The first condition contradicts definition of $\tau(\mathtt{a},\mathtt{b}-i)$.
If the second condition happens then we get a fake coins from the unbalanced pair, the graph is feasible.
(Optimal Part:) from lemma 5.2

$$\begin{aligned} tau^{-}(a,b) &= min\{tau(a,b), \{i-1+Q(a+b-i,a,b)|1 \le i \le a\}\} \\ &\ge min\{tau(a,b), \{i-1+\tau(a,b-i)|1 \le i \le a\}\} \end{aligned}$$

is a optimal lower bound.
And $optimal\{OTS(a,b), \{$a tree of i nodes and $OTS(a,b-i)|1 \le i \le a\}\}$ reaches this lower bound.

□

Optimal testing solution for finding a real coin

OTS^{+} is trivial. That is, comparing $\mathtt{a}$ coins in a group.

Theorem 5.3. *$OTS^{+}(\mathtt{a},\mathtt{b}) =$ a tree of $\mathtt{a}+1$ nodes*

Proof. We divide the proof into feasible part and optimal part.
(Feasible Part:)
If it results in an unbalanced pair. The heavier coin of the pair is the real one.
If it results in all balanced. This means this group is real.
(Optimal Part:)

$$\begin{aligned} \tau^{+}(a,b) &= min\{\tau(a,b), \{i-1+Q(a+b-i,b,a)|1 \le i \le b\}\} \\ &\ge min\{\tau(a,b), \{i-1+\tau(a-i,b)|1 \le i \le b\}\} && \text{by Theorem 2.2.} \\ &\ge min\{\lfloor\frac{a+b}{2}\rfloor, i-1+\frac{a-i+b}{2}\} \\ &\ge min\{a,a\} = a \end{aligned}$$

is a optimal lower bound.
a tree of $\mathtt{a}+1$ nodes reaches optimal lower bound. □

6 Open questions

6.1 Cycle happens earlier

As in theorem 4.3 and 4.4, we proved $\rho(\mathtt{a}, \mathtt{b})$ would be in cycle for $\mathtt{b} > d(d+1)(\mathtt{a}-1) + (\mathtt{a} - (\mathtt{a} \bmod d)$. But observing ρtable in Results 3.2, ρ seems to get in cycle earlier. After computer comfirmation, we got results bellow.

a	b: $\rho(\mathtt{a}, \mathtt{b})$ start to cycle by theorem	b: $\rho(\mathtt{a}, \mathtt{b})$ start to cycle by computer comfirmation
1	1	1
2	13	1
3	15	5
4	40	10
5	29	9
6	105	29
7	43	13
8	91	22
9	57	17
10	118	28

Since the algorithm cost exponential time, we could only comfirm this in small case.

But if it get in cycle earlier in general case, we can use a better(smaller) K algorithm 4.1 and run faster.

6.2 Avoidance integer partition

As in algorithm 3.1,4.1, to find an optimal solution require enumerating partitions of $\mathtt{a} + \mathtt{b}$ avoiding $\mathtt{a}$. In the algorithms, we enumerating all partitions of $\mathtt{a} + \mathtt{b}$ and check if it avoids $\mathtt{a}$. So the time complexity is analyzed as the (number of partitions) times checking avoidance. If exist a way to enumerate avoidance and bound it tighter, the time complexity would be better.

7 Refferences

1. Howard D. Grossman, The twelve-coin problem, Scripta Math., 11 (1945) 360-361.
2. Richard K. Guy ; Richard J. Nowakowski, Coin-Weighing Problems, The American Mathematical Monthly, Vol. 102, No. 2. (Feb., 1995), pp. 164-167.
3. Lorenz Halbeisen, Norbert Hungerbfihler, The general counterfeit coin problem, Discrete Mathematics 147 (1995) 139-150.