Chapter 3

A Generalised Algorithm for Solving *n* Coins Problem

3.1 Overview

In this chapter, we are going to study the new solutions for eight coins problem and algorithms for solving *n* coins problem. This chapter is organized into eight sections. In Section 3.2, we have briefly discussed the eight coins problem and its two new solutions. In Section 3.3, we have discussed algorithms for solving *n* coins problem. In Section 3.4, we have discussed the extension of the algorithm to solve $n \geq 6$ (even) coins problem. In Section 3.5, we have discussed the extension of the algorithm to solve $n \geq 7$ (odd) coins problem. In Section 3.6, we have discussed the applications of the problem, and in Section 3.7, we have shown the experimental results.

3.2 Introduction to Eight Coins Problem

Eight coins problem [1, 3, 25] is a well-known problem in Mathematics as well as in Computer Science. In this problem, eight coins are given. Say A, B, C, D, E, F, G, and H, and we are told that only one is counterfeit (or false), as it has a different weight than each of the others. We want to determine which coin it is, making use of an equal arm balance. At the same time, we want to know using a minimum number of comparisons and determine whether the false coin is heavier or lighter than each of the remaining. The tree in this figure represents a set of decisions by which we can get the solution(s) of our problem. Therefore, it is called a decision tree. We use lower-case *h* or *l* as a suffix to represent the counterfeit (or false) coin as *heavier* or *lighter*, respectively. In the solution of the eight coins problem in the form of a decision tree in Figure 3.1, each internal vertex (other than leaf vertices) represents a comparison between a pair of sets of coins using an equal arm balance. Needless to mention that in this comparison, both the sets contain an equal number of coins. The tree starts with a vertex, where it considers three coins be kept on either side of the equal arm balance. Surely, if we consider all the eight coins at a time to distribute them into two sets of four coins each to be compared, then it is a useless comparison as one coin out of eight coins is given as counterfeit (or false). Thus, we cannot start with all the coins at a time to be compared to finding out the false coin; rather it leads a redundant comparison. In the decision tree in Figure 3.1, we consider three coins on one side of the equal-arm balance, which is the root of the tree. If the weight-sums are equal, then surely each of these coins is a true coin, and the false coin is either G or H with their possibilities either heavier or lighter. In this situation, as A is a true coin, which we use in comparing separately with G and H after a comparison between G and H themselves. If the weight-sum containing coin A is less than the weight-sum containing coin D (i.e. A+B+C < D+E+F), then we can say that the false coin is either of these six coins only, where either A is lighter, or B is lighter, or C is lighter, or D is heavier, or E is heavier, or F is heavier. At the same time in this situation, both G and H are true coins. The next comparison is highly important in order to make the height of the tree as small as possible; we do three things as follows: (*i*) keep a pair of coins A and E on their own sides, (*ii*) another pair of coins B and D interchange their sides, and (*iii*) the remaining pair of coins C and F are removed from the comparison. Therefore, subsequently, the weight-sum of A and D (i.e. A+D) compare with the weight-sum of B and E (i.e. B+E). Three cases may arise.

Case 1: If weight-sums are equal, then either C or F is a false coin (as these coins are removed from this comparison), where either C is lighter, or F is heavier (following the root of the tree). Therefore, we compare C with A (a true coin). If the weight of A is more than the weight of C, then C is lighter; otherwise, these two coins must have the same weight resulting F as heavier.

Case 2: If A+D < B+E, then certainly either A or E is a false coin, as these coins are kept in their own sides, and the logical relation (following the root of the tree) is unchanged. Here either A is lighter, or E is heavier. Therefore, we compare B (a true coin) with A. If the weight of B is more than the weight of A, then A is lighter; otherwise, these two coins must have the same weight resulting E as heavier.

Case 3: In a similar way, if A+D > B+E, then definitely either B or D is a false coin, as these coins have interchanged their sides and the logical relation (following the root of the tree) has also changed. Here, either D is heavier, or B is lighter. Therefore, we compare B with A (a true coin). If the weight of A is more than the weight of B, then B is lighter; otherwise, these two coins must have the same weight resulting D as heavier.

Similarly, we may consider the case of weight-sums following the remaining branch of the root of the tree, where A+B+C > D+E+F. Here either A is heavier, or B is heavier, or C is heavier, or D is lighter, or E is lighter, or F is lighter. The remaining part of the subsequent comparisons is done in a similar way as it is explained above and shown in Figure 3.1.

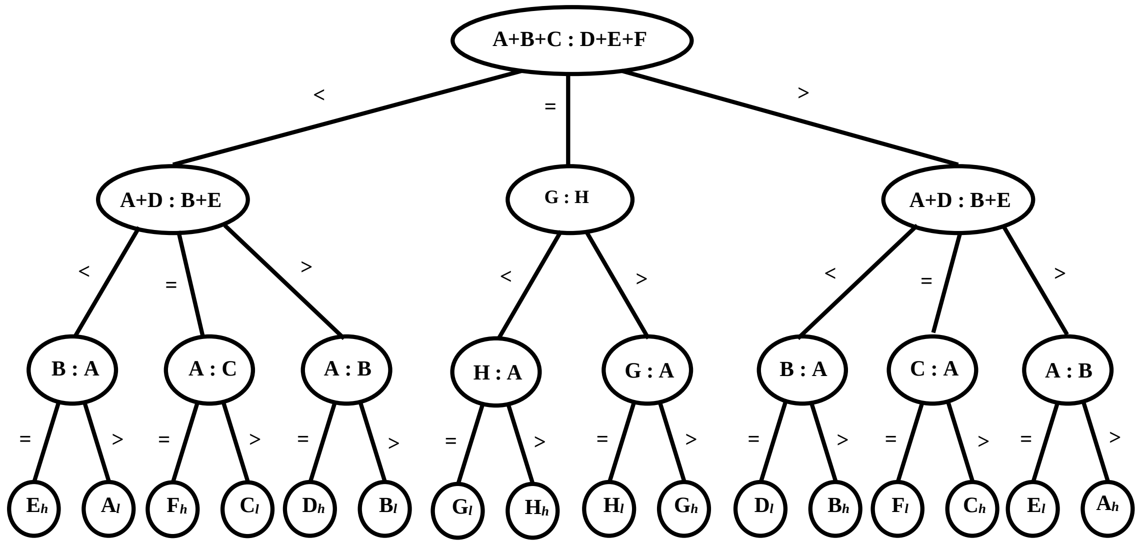


Figure 3.1: The existing solution of the eight coins problem in the form of a decision tree [71].

3.2.1 Two New Solutions of the Eight Coins Problem

In this section, we introduce two new solutions of the eight coins problem and discuss how they are evolving. In the next section, we compare all these solutions based on their relative merits and demerits.

Before developing the newer solutions of the problem under consideration in two subsections of this section, let us show how a false coin, which is either heavier or lighter, can be differentiated in a naive way; this solution is shown in Figure 3.2. In this solution, we compare only a pair of coins to find out the false one. Anyway, this is also a valid solution with sixteen terminal vertices as eight lighter and eight heavier possibilities of eight different coins of the problem. In contrast to the existing solution in Figure 3.1, this solution takes five (levels of) comparisons in the worst-case, where the solution in Figure 3.1 requires at most three (levels of) comparisons. Therefore, Figure 3.2 is not a desired decision tree in finding out all the possibilities of a coin, either heavier or lighter, of the eight coins problem. In a desired solution of the eight coins problem, we can involve at most three comparisons only (as this is the best-known result and the lower bound on the number of comparisons is also three) [24, 25]. Two such solutions are developed and introduced in the following two subsections.

3.2.1.1 A Solution to the Eight Coins Problem

In the eight coins problem, we are having at most eight coins out of which only one coin in counterfeit (or false), though we do not know whether the false coin is heavier or lighter. Now we like to state two basic observations as follows.

Observation 1: It is meaningless to involve all the coins simultaneously in comparison to finding out a false coin. In that case four of them forms a group and the remaining four forms another group. Finally, one side is always heavier (or lighter) than the other, which is known a priority. This is a redundant comparison and results in giving a very few information; coins are subsequently compared further that increases the height of the tree.

Observation 2: On the other hand, if we consider only pairs of coins to be compared, as done in the case of Figure 3.2, the tree height also increases. It might be a naive way of finding out the false coin, either heavier or lighter, but as the height of the tree increases, it is not the desired solution as it is already mentioned earlier.

Based on the above observations what we conclude that the root of the decision tree must start with comparing coins, keeping either two or three of them in a group. In our first (new) solution we start with consisting only four coins, keeping two of them in a group and comparisons. Suppose A, B, C, and D are these coins where A and B are in a group and C and D in the other, as shown in Figure 3.3. Two cases may arise, either equal or not equal. The case of equality tells that all the four coins A through D are true coins; otherwise one of them must be a false coin, with possibility heavier or lighter. In the next step following the case of equality, we introduce three coins E, F, and G in a group and compare them with A, B, and C in another group, where each of the coins A, B, and C is a true coin. For A, B, and C (and D) are true coins, so from this comparison, we may find the false coin among E, F, and G (and H) with possibility heavier or lighter. Three cases may arise.

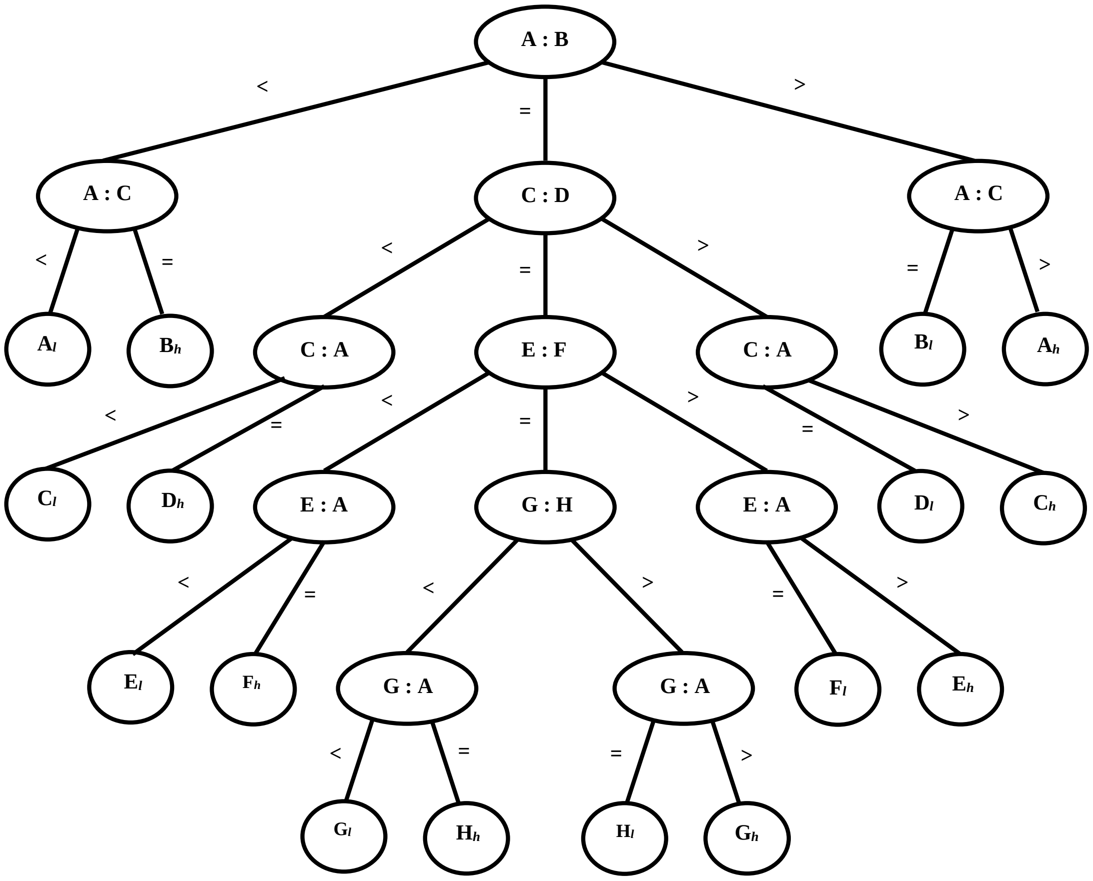


Figure 3.2: A naive solution of the eight coins problem that has been solved with the help of a decision tree.

Case 1: The weight-sum of coins E, F, and G is less than the weight-sum of coins A, B, and C (i.e. E+F+G < A+B+C). This case tells that only either E or F or G is a lighter coin, as A, B, and C are true coins.

Case 2: If the weight-sum of coins E, F, and G is same as the weight-sum of coins A, B, and C (i.e. E+F+G = A+B+C), then surely each of all these coins is a true coin, resulting only H as the false coin, either heavier or lighter.

Case 3: The weight-sum of coins E, F, and G is greater than that of coins A, B, and C (i.e. E+F+G > A+B+C). This case tells that only either E or F or G is a heavier coin, as A, B, and C are all true coins.

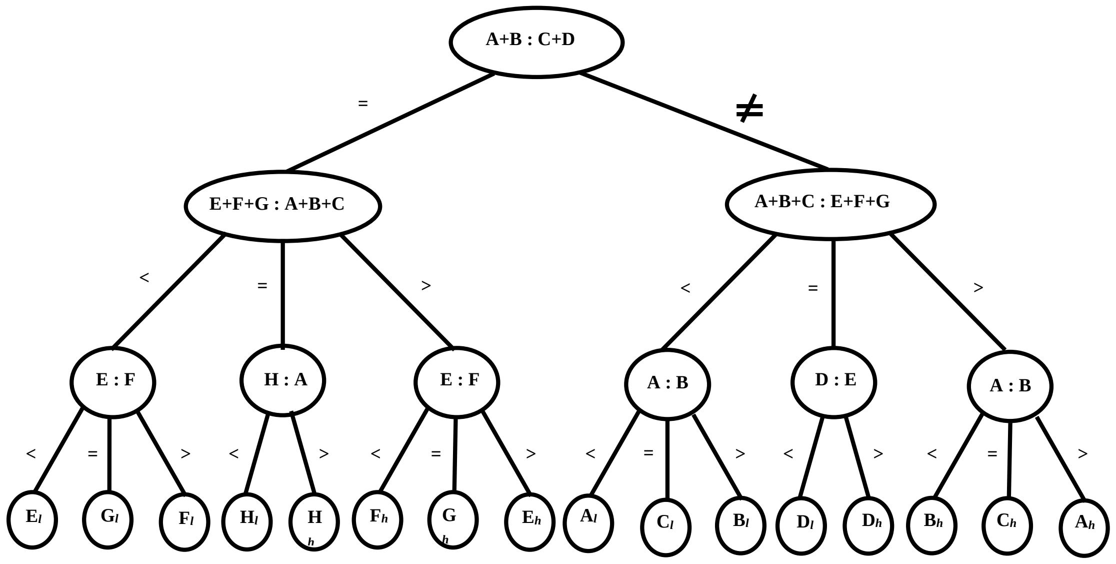


Figure 3.3: The first new solution of the eight coins problem in the form of a decision tree.

Following Case 1 above, to know which of the coins E, F, and G is lighter, we compare any two of them, as shown in Figure 3.3. We compare E and F and conclude the following. If the weight of E is less than the weight of F, then coin E is lighter (as none of them is heavier). If the weight of the E is greater than that of F, then coin F is lighter. Otherwise, if both the coins are having the same weight, then coin G is the false coin with possibility lighter.

Following Case 2 above, we naturally conclude that the coin H is the false coin with possibility either heavier or lighter. This can easily be determined by comparing the coin H with a true coin A, as shown in Figure 3.3. If the weight of H is less than that of A, then coin H is lighter; otherwise, H is heavier.

Following Case 3 above, to identify the false coin among the coins E, F, and G as heavier, we compare any two of them, as shown in Figure 3.3. We compare E and F and conclude the following. If the weight of E is less than the weight of F, then coin F is heavier (as none of these coins is a lighter coin). If the weight of the E is greater than that of F, then coin E is a heavier coin. Otherwise, if both the coins are of the same weight, then coin G is the false coin with possibility heavier.

Now we consider the other part of the root vertex of the decision tree. Here the weight-sum of A and B is not equal to the weight-sum of C and D (i.e. $A+B \neq C+D$); then we conclude that the false coin belongs to the set of these four coins only. Following this case of inequality, we introduce three coins E, F, and G in a group and compare them with the group of coins A, B, and C (as we did in the case of equality above). Note that in this case E, F, and G (and H) are all true coins.

Since the false coin belongs to the set of coins A, B, C, and D, we want to find out them with their possibilities of either heavier or lighter through the following cases; three cases may arise.

Case 1: The weight-sum of coins A, B, and C is less than the weight-sum of coins E, F, and G (i.e. $A+B+C < E+F+G$). This case tells that only either A or B or C is a lighter coin, as E, F, and G are all true coins.

Case 2: If the weight-sum of coins A, B, and C is same as the weight-sum of coins E, F, and G (i.e. $A+B+C = E+F+G$), then certainly each of all these coins is a true coin, resulting only D as the false coin, either heavier or lighter.

Case 3: The weight-sum of coins A, B, and C is greater than that of coins E, F, and G (i.e. $A+B+C > E+F+G$). This case tells that only either A or B or C is a heavier coin, as E, F, and G are all true coins.

Following Case 1 above (in the case of inequality), to know which of the coins A, B, and C is lighter, we compare any two of them, as shown in Figure 3.3. We compare A and B and conclude the following. If the weight of A is less than the weight of B, then coin A is lighter (as none of them is a heavier coin). If the weight of A is greater than that of B, then coin B is lighter. Otherwise, if both the coins are of the same weight, then surely coin C is the false coin with possibility lighter.

Following Case 2 above, we eventually conclude that the coin D is the false coin with possibility either heavier or lighter. This can easily be determined by comparing the weight of coin D with that of a true coin E, as shown in Figure 3.3. If the weight of D is less than that of E, then coin D is lighter; otherwise, D is heavier.

Following Case 3 above, to know which of the coins A, B, and C is heavier, we compare any two of them, as shown in Figure 3.3. We compare A and B and conclude the following. If the weight of A is less than that of B, then coin B is heavier (as none of these coins is a lighter coin). If the weight of A is greater than that of B, then coin A is a heavier coin. Otherwise, if both the coins are of the same weight, then coin C is the false coin with its only possibility heavier.

The above development towards a new decision tree for the eight coins problem is shown in Figure 3.3, as the first new solution of the problem in this thesis. The second new solution of the eight coins problem is developed in the next subsection. We critically compare all these solutions in Section 3.7.

3.2.1.2 One Step ahead with the Eight Coins Problem

With this newly developed solution too we start with consisting only four coins, keeping two in a group. Suppose A, B, C, and D are these coins where A and B are in a group, and C and D in the other, as it is in the case of developing a new solution of the eight coins problem in the previous subsection (see Figures 3.3 and 3.4). Here we reach into three possibilities, splitting the case of inequality in either *less* and *greater,* other than the case of equality as it is (in the previous subsection). This method is unlike to the previous method in the way that here we always consider only two coins in a group while making comparisons.

Following the case of equality, we may conclude that all the four coins A, B, C, and D are true coins, and the false coin belongs to the set of remaining four coins E, F, G, and H only, with their possibilities either heavier or lighter. Therefore, this branch of equality follows eight possible leaves with both the possibilities of heavier and lighter for coins E, F, G and H. The other two branches (of *less* and *greater*, following the root of the tree) share the remaining eight possibilities, with four each as stated below.

If the weight-sum of coins A and B is less than the weight-sum of coins C and D (i.e. $A+B < C+D$), then ultimately this branch is supposed to conclude with possibilities either A is lighter, or B is lighter, or C is heavier, or D is heavier. On the contrary, if the weight-sum of coins A and B is greater than the weight-sum of coins C and D (i.e. A+B > C+D), then we conclude with either of the following possibilities: either A is heavier, or B is heavier, or C is lighter, or D is lighter.

Certainly, following the case of equality of the root of the tree, we introduce coins from the remaining set. Instead of introducing all of them, we apply a trick by introducing any three of them along with a true coin. Say, at this step coins E and F form a group, and coins G and A (a true coin) from the other group. Three possible cases may arise.

Case 1: If E+F = G+A, then surely each of all these coins is a true coin, resulting only H as the false coin, either heavier or lighter. Then we compare H with A and conclude accordingly. If the weight of H is less than that of A, then H is lighter; otherwise, H is a heavier coin.

Case 2: If E+F < G+A, then there are three possibilities: Either E is lighter, or F is lighter, or G is heavier (as A is a true coin). Therefore, subsequently, we compare E and F to conclude the following. If E < F, then E is lighter (as F cannot be heavier). On the other hand, if E > F, then F is lighter (as E cannot be heavier). Otherwise, if the weight of the E is same as the weight of the F (i.e. E = F), then G is heavier.

Case 3: If E+F > G+A, then there are three possibilities: Either E is heavier, or F is heavier, or G is lighter (as A is a true coin). Therefore, subsequently, we compare E and F (same as before) to conclude the following. If E < F, then F is heavier (as E cannot be lighter). On the other hand, if E > F, then E is heavier (as F cannot be lighter). Otherwise, if the weight of the E is same as that of F (i.e. E = F), then G is lighter (as A is identified as a true coin in the earlier step).

Now, we consider the case of A+B < C+D, following the root of the tree. As a result, we may conclude that (*i*) either A is lighter, or B is lighter, or C is heavier, or D is heavier, and (*ii*) each of the remaining four coins, that are E, F, G, and H, is a true coin. Therefore, in a similar way, we apply a trick, where we consider three of the four probable false coins along with a true coin from the remaining (true coins). In addition, we interchange the sides of a pair of coins (say B and C) of the root vertex in making the new groups as follows. Here we keep A and C in a group and B and E (a true coin) in the other.

Three cases may arise:

Case 1: If A+C < B+E, then definitely A is the false coin with possibility lighter. This is because (*i*) E is a true coin, (*ii*) D is no longer there in this comparison, and (*iii*) coins B and C have changed their sides, although the inequality relation is same as before.

Case 2: If A+C = B+E, then surely D is the false coin with possibility heavier. This is because in this case (*i*) all the coins A, B, C, and E are true coins, and (*ii*) D is the only dropped coins in this comparison following the earlier case of inequality.

Case 3: If A+C > B+E, then certainly either C is heavier, or B is lighter, as only this pair of coins have changed their sides and inequality relation has also changed. Therefore, one more step is required to find out the false coin that we achieve by comparing B with E, which is a true coin. Only two cases may arise as follows. If B < E, then B is a lighter coin; otherwise, they must have the same weight to conclude that C is heavier.

The remaining branch of the root of the case of A+B > C+D is treated in a similar way. Here we may conclude that either coin A is heavier, or B is heavier, or C is lighter, or D is lighter. Therefore, similar to the previous step of inequality following the root of the tree, we consider the four coins A, B, C, and E, and group them identically in the same way. Here, coin **E** is a true coin. Three probable cases may arise.

Case 1: If A+C > B+E, then definitely A is the false coin with possibility heavier. This is because (*i*) E is a true coin, (*ii*) D is no longer there in this comparison, and (*iii*) coins B and C have changed their sides, although the inequality relation is same as before.

Case 2: If A+C = B+E, then surely D is the false coin with possibility lighter. This is because in this case (*i*) all the coins A, B, C, and E are true coins, and (*ii*) D is the only dropped coins in this comparison following the earlier case of inequality.

Case 3: If A+C < B+E, then certainly either C is lighter, or B is heavier, as only this pair of coins have changed their sides and inequality relation has also changed. Therefore, one more step is required to find out the false coin that we achieve by comparing B with E, which is a true coin. Only two cases may arise as follows. If B > E, then B is a heavier coin; otherwise, they must have the same weight to conclude that C is lighter.

This completes the development of the second new solution of the eight coins problem. The solution is shown in the form of a decision tree in Figure 3.4. In the next section, we consider all these solutions, existing, naive, or newly developed in this thesis for their relative comparisons on several parameters.

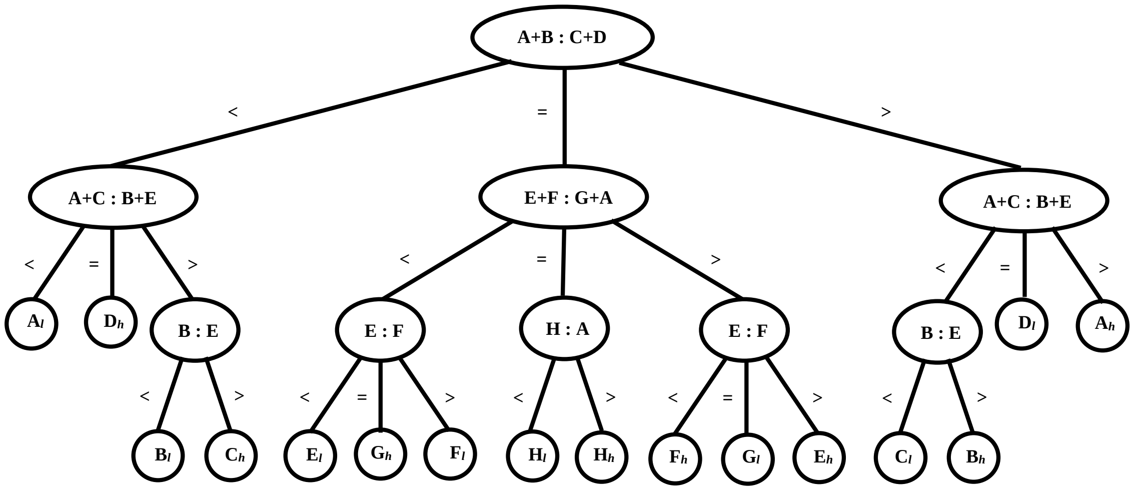


Figure 3.4: The second new solution of the eight coins problem in the form of a decision tree.

3.2.2 Relative Comparisons among the Solutions

In this thesis, we have considered the eight coins problem from its interest from a combinatorial optimization point of view. An existing solution to this problem is already there in the literature [25, 71] (see Figure 3.1). A naive solution of the same is shown in Figure 3.2, and two new solutions are developed in this thesis, as shown in Figures 3.3 and 3.4. In this section, we compare all these solutions based on their maximum number of comparisons towards a solution, their external path length, the total number of comparisons required, consideration of a maximum number of coins involved in comparison, the average height of the decision tree, and so and so forth. All these results are shown in Table 3.1. Now, we analyze the parameters considered in comparing the solutions included in this thesis. The height of the best existing solution of the eight coins problem is four where the desired decisions, either a coin is heavier or lighter, are all available at the fourth level in the form of leaf vertices of the decision tree. Therefore, at most three comparisons are required starting from the root of the tree. This result of a maximum number of comparisons is equally good for the cases of two newly developed solutions introduced in this thesis, in the worst-case; naturally, the naive solution does not follow it.

The *external path length* is a measure of a tree, which is the sum of the number of branches (or edges) traversed in going from the root *once* to every leaf in the tree [24]. In this respect, the existing solution and the first new solution developed in this thesis are equally good. Needless to mention that in this respect, the naive solution is worse though the second new solution developed in this thesis is better, as four leaf vertices (as decisions) are achieved at the third level.

In terms of total number of comparisons, i.e. the number of internal vertices (other than leaf vertices only), both the existing solution and the naive solution are equally good, but in developing each of the new solutions introduced in this thesis, we have reduced by 25% of the number of comparisons in comparing to that of the earlier two.

Someone may think that the number of coins present in comparison may not be an important issue, though the notion tells that the maximum number of comparisons (or even the height of the tree) may reduce if we involve more coins in a comparison. If only pairs of coins are considered in comparisons, as done in computing the naive solution, the height of the tree increases, though we do know that we should not involve all the coins in comparison, as discussed earlier. Incidentally, our first new solution involves a maximum of six coins in comparison, whereas the second new solution involves only four.

We know that the height of the tree is one more than the maximum number of comparisons involved, starting from the root of the tree, in computing all the desired levels. On the other hand, the average height of the tree is a parameter obtained by computing the average external path length over the desired leaves in a tree. Each solution of the eight coins problem present in this thesis results in obtaining 16 leaf vertices, with eight heavier possibilities and eight lighter possibilities of the given eight coins of the problem. Thus, the average external path length (or average height of the tree) is the average distance from the root of each of the leaf vertices of the tree. In this respect, our first new solution is as good as the existing one, though the second new solution is certainly much better.

Table 3.1: The relative merits and demerits of different solutions of the eight coins problem based on different parameters of making comparisons.

Parameters of making comparisons	**Existing solution (in Figure 3.1)**	**A naive solution (in Figure 3.2)**	**The first new solution (in Figure 3.3)**	**The second new solution (in Figure 3.4)**
Maximum number of comparisons	3	5	3	3
External path length	48	56	48	44
Total number of comparisons	12	12	9	9
Maximum number of coins in a comparison	6	2	6	4
Height of the tree	4	6	4	4
Average height of the tree	3	3.5	3	2.75

At the end of this discussion, we may conclude that our first new solution is very close to the existing one, though here the total number of comparisons is just nine, whereas that of the existing solution is twelve. Our second new solution outperforms all these; its average height is just 2.75, though the maximum number of coins involved in a comparison case has been just four. Therefore, up to this point of time, we may come to an end (or infer) that the second new solution (that is in Figure 3.4) is the best solution of the eight coins problem.

3.3 Algorithms for Solving *n* Coins Problem

In this section in developing our algorithm, we minimally modify the classical solution, shown in Figure 3.1, to compute a better solution in terms of comparisons as shown in Figure 3.4. Now to minimize the number of comparisons, for the equality case at the root we do not compare the coins G and H. This is because one out of G and H must be a counterfeit coin, and so they are of unequal weight. We remove this redundant comparison and compare G with A (which is a known correct coin). If they are of unequal weight, it means that G is the faulty coin, and we find out whether it is heavier or lighter from this comparison only. Otherwise, it is understood that H is faulty. In which case we compare H with the known correct coin A and find out whether it is heavier or lighter.

3.3.1 Extension of the Algorithm to Solve *n* > 8 (Powers of 2) Coins Problem

The *n* coins problem is solved using the procedure *n-coin_problem* as described here in this section. This procedure takes, as inputs, the starting index of then coins, and the number of coins. This procedure, then calls the *less_than* function or the *greater_than* function depending upon the weight of the pans. The function *less_than* is called when in the first comparison (i.e. done at the root of the decision tree) the weight of the left pan is less than that of the right pan. Similarly, the function *greater_than* is called when the left pan is heavier than the other. These functions are recursively called within each of the functions until only two suspected coins (of whom one is certainly a counterfeit coin) are left.

As the tree structure goes down, each of these procedures eliminates some coins at each stage, until the number of suspected coins is two [4]. At this stage, these functions take the help of the *check* function. This *check* function helps to identify the counterfeit coin between those two suspected coins, with the help of a known correct coin. When the procedure *n-coin_problem* is being called recursively, and the number of remaining coins is only two or only four, then the procedure calls the *2-coin_problem* or the *4-coin_problem*, respectively (instead of calling the *n-coin_problem* again). Basically, these two procedures act as the base case when we are calling the *n-coin_problem* recursively. Now, let us look at the functioning of the generalized algorithm for solving *n* coins problem, where *n* is a power of 2, which is greater than eight.

Input: An integer number *n* of coins (out of which only one coin is counterfeit, either heavier or lighter)**.**

Output: The index z of the counterfeit coin, where z is an integer (location of the false coin) and declaring whether it is heavier or lighter**.**

In developing the algorithm that solves the eight coins problem, as shown in Figure 3.4, we now consider each of the procedures as outlined above in isolation and explain how they work. We start with the main procedure, i.e. the *n-coin_problem*.

3.3.2 Procedure *n-coin_problem*

We have been given n coins, out of which only one coin is counterfeit (or false). To find the counterfeit coin (and to know whether it is heavier or lighter), we use the decision tree structure. We call the *n-coin_problem* with *start* = 1 and the number of coins = n. Here, *start* indicates the starting index of the coins to be considered (or compared), and n is the size of the problem. The total number of possible outcomes in cases of n coins problem is equal to $2n$. We first determine the number of coins to be kept on each of the pans of the equal arm balance [4]. This is given by $3x$, where $x = n/8$. Thus, $6x$ coins are compared initially. We can have three possible cases as follows:

1. Left pan is lighter than the right pan. This case is handled by the *less_than* function. Since the left pan is lighter, it means that any one of the $3x$ coins kept on the left pan is lighter, or any one of the $3x$ coins kept on the right pan is heavier. From this condition, we can reach $2\times3x = 6x$ number of possible outcomes as leaf nodes. The counterfeit coin exists among the coins indexed by *start* and $6x$. The left pan contains the coins indexed from the *start* through $3x$, and the right pan contains the coins indexed from $3x+1$ through $6x$. The first coin on the left pan is indexed by *S_left* (i.e. at the beginning it is same as a *start*). Similarly, the first coin of the right pan is indexed by *S_right*, (i.e. at the beginning it is the coin same as $3x+1$).

2. Left pan is heavier than the right pan. This case is handled by the *greater_than* function. Since the right pan is lighter, it means that any one of the $3x$ coins kept on the left pan is heavier, or any one of the $3x$ coins kept on the right pan is lighter. Likewise, from this condition, we can reach $2\times3x = 6x$ number of possible outcomes as leaf nodes. The indexing is the same as in the previous case.

3. Both the pans are equal. Since the pans are equal, it means that the $6x$ coins compared at the root are correct. The counterfeit coin then lies in the $2x$ number of coins yet to be considered. In this case, i.e. in the case of equality, the n coins problem gets reduced to $2x$ coins problem. To solve this $2x$ coins problem, we call the *n-coin_problem* recursively, with the number of coins now equal to $2x$. Since the first $6x$ coins are correct, now $start = 6x+1$ and $n = 2x$. From this case of equality, we can reach $2\times2x = 4x$ number of possible outcomes as leaf nodes.

3.3.3 Procedure *less_than* Function

We retain the first x coins in the left pan (starting from *S_left*) and interchange the next x number of coins with the first x coins in the right pan (starting from *S_right*) and retain the next x coins on the right pan as it is. Thus, in this comparison, the last x coins of both the pans are not considered. Rather, these coins are removed from the comparison at this stage. Again, in this case, there are three possible outcomes:

(i) The left pan is lighter: The counterfeit coin exists among the first x number of coins in the left pan or the middle x number of coins in the right pan. Thus, we must update the value of *S_right*. It now points to the first coin of the middle x number of coins of the right pan. The value of *S_left* remains as it is.

(ii) The left pan is heavier: The counterfeit coin exists among the middle x number of coins in the left pan or the first x number of coins in the right pan. Thus, we must update the value of *S_left*. It now points to the first coin of the middle x number of coins of the left pan. The value of *S_right* remains as it is.

(iii) Both pans are equal: The counterfeit coin exists among the last x number of coins of both the left and the right pan. Thus, the value of both *S_left* and *S_right* needs to be updated. *S_left* now points to the first coin of the last x coins of the left pan, and *S_right* now points to the first coin of the last x coins of the right pan.

Now, the value of x is halved as we move down one level in the decision tree structure. If the value of x is equal to ½, only two coins are left undecided. At this point, we call the *check* function with these two undecided coins and a known correct coin, as it has been considered for the three coins G, H, and A, where A is a correct coin (see the case of equality in Figure 3.4 following the root of the tree).If the value of *x* is more than ½, the *less_than* function is called recursively with the updated values of *S_left* and *S_right*. Case (iii) does not occur more than once. This is because all the undecided coins (among which the counterfeit coin exists) are compared with the next subsequent levels.

3.3.4 Procedure *greater_than* Function

This procedure is very much the mirror image of the earlier procedure. Here we retain the first *x* coins in the left pan (starting from *S_left*) and interchange the next *x* number of coins with the first *x* coins in the right pan (starting from *S_right*) and retain the next *x* coins of the right pan as it is. Thus, in this comparison, the last *x* coins of both the pans are not considered. Again, in this case, there are three possible outcomes:

(i) The left pan is heavier: The counterfeit coin exists among the first *x* number of coins in the left pan or the middle *x* number of coins in the right pan. Thus, we must update the value of *S_right*. It now points to the first coin of the middle *x* number of coins of the right pan. The value of *S_left* remains as it is.

(ii) The left pan is lighter: The counterfeit coin exists among the middle *x* number of coins in the left pan or the first *x* number of coins in the right pan. Thus, we must update the value of *S_left*. It now points to the first coin of the middle *x* number of coins of the left pan. The value of *S_right* remains as it is.

(iii) Both pans are equal: The counterfeit coin exists among the last *x* number of coins of both the left and the right pan. Thus, the value of both *S_left* and *S_right* needs to be updated. *S_left* now points to the first coin of the last *x* coins of the left pan, and *S_right* now points to the first coin of the last *x* coins of the right pan.

Now, the value of *x* is halved as we move down one level in the decision tree structure. If the value *of x is* equal to ½, only two coins are left undecided. At this point, we call the *check* function with these two undecided coins and a known correct coin. The *check* function takes the three coins as parameters, in the following order– lighter or correct coin (for less than the condition it would be the coin indexed by *S_right*, and for greater than condition it would be the coin indexed by *S_right*+1), heavier or correct coin (for less than the condition it would be the coin indexed by *S_left*+1, and for greater than condition it would be the coin indexed by *S_left*), and known correct coin. This *check* function then finds out the counterfeit coin by comparing these three coins. If the value of *x* is more than ½, the *greater_than* function is called recursively with the updated values of *S_left* and *S_right*. In a similar way, as in the case of a *less_than* function, here also the third case (i.e. Case (iii) above) does not occur more than once. This is because all the undecided coins (among which the counterfeit coin exists) are compared with the next subsequent levels [5].

3.3.5 Procedure *4-coin_problem*

This procedure solves the *n-coin_problem* when the problem is reduced to $n = 4$, i.e. there are only four undecided coins out of which one is counterfeit. The parameters passed to this function are a *start* (i.e. the index of the coin from which the four undecided coins occur) and the index of a correct coin. It compares the coins indexed by *start* and *start*+1. If they are equal, then it means that these two coins are correct, and the problem reduces to two coins problem. Then the *2-coin_problem* is called with a *start* = *start*+2 and the index of a correct coin. Otherwise, if the weight of the initial two coins is unequal, then the *check* function is called with these two coins to find out the counterfeit coin.

3.3.6 Procedure *2-coin_problem*

This solves the *n-coin_problem* when the problem is reduced to $n = 2$, i.e. there are only two undecided or unchecked coins. The parameters passed to this function are a *start* (i.e. the index of the coin from which two undecided coins occur) and the index of a correct coin. This gives the index of the counterfeit coin and mentions whether it is heavier or lighter.

3.3.7 Procedure *check* Function

The *check* function takes three coins as parameters in the following order– lighter or corrects coin, heavier or correct coin, and a known correct coin. This procedure tells whether the first coin passed is lighter (which may be the lighter coin or a correct coin), or the second coin passed is heavier (which may be the heavier coin or a correct coin), or the reverse. This is done by comparing the weights of the first and second coin passed with the known correct coin passed.

3.4 Extension of the Algorithm to Solve $n \geq 6$ (Even) Coins Problem

In this version of the algorithm, we first find the smallest possible value y depending upon the given value of n such that $y \geq n$, where y is a power of 2. If $y = n$, this version of the algorithm exactly matches to the steps of the algorithm stated in Section 3.1. Needless to mention that if $n = 8$, then $x = 1$ and the number of coins compared at the root of the decision tree is 6 keeping the first 3 coins on the left pan and the next 3 coins on the right pan. Similarly, for $n = 16$, $x = 2$ and the number of coins compared is equal to 12 keeping the first 6 coins on left pan and the next 6 coins on right pan, for $n = 32$, $x = 4$ and the number of coins compared is equal to 24 keeping the first 12 coins on the left pan and the next 12 coins on the right pan, and so on [4].

On the other hand, if n is not equal to y, rather $n < y$ (where $2^{p-1} < n < 2^p = y$, for some integer value of $p \geq 3$), then the number of coins compared in each pan is $2x$, where $x = y/8$. As, for example, if $n = 10$, then $y = 16$ and $x = 2$, and the total number of coins compared at the root of the decision tree is 8 keeping the first 4 coins on the left pan and the next 4 coins on the right pan, if $n = 22$, then $y = 32$ and $x = 4$, and the total number of coins compared at the root of the tree is 16 keeping the first 8 coins on left pan and the next 8 coins on right pan, and so on.

If the left and the right pans are equal in weight at the first comparison made at the root of the tree, then the counterfeit coin lies among the remaining $n–2^{p-1}$ coins of the n coin problem. For example, if the decision follows the equality branch of the tree for $n = 10$, then the counterfeit coin belongs to the remaining 2 coins, for $n = 22$, the counterfeit coin belongs to the remaining 6 coins, and so on. Then following this equality branch of the decision tree, we either recursively call procedure *n-coin_problem*, or straightway call procedure *4-coin_problem*, or *2-coin_problem*, as the situation arises.

On the other hand, if the left pan is lighter than the right pan, then the counterfeit coin lies among the first $4x$ coins, of which one of the first $2x$ coins can be lighter, or one of the next $2x$ coins can be heavier. Then it calls the *less_than* procedure to deal with the left sub-tree of the tree.

If the left pan is heavier than the right pan, then the counterfeit coin lies among the first $4x$ coins, of which one of the first $2x$ coins can be heavier, or one of the next $2x$ coins can be lighter. Then it calls the *greater_than* procedure to deal with the right sub-tree of the tree.

In this way, each time we move down towards the leaf of the decision tree the value of x is divided by 2. When $x = ½$, there are only 2 coins undecided. One of these 2 coins is the counterfeit one. This time the *check* function finds the desired counterfeit coin and declares whether it is heavier or lighter.

3.5 Extension of the Algorithm to Solve $n \geq 7$ (Odd) Coins Problem

In our algorithm, this is an obvious checking whether the given number n of coins is even or odd. If it is even, then it is checked whether it is a power of two. If it is a power of two, as it has been developed in Section 3.2. On the other hand, if the value of n is odd, we do some prior computations. In this case, just one additional checking is made as it has been detailed below [4, 5].

We know that if n is odd, then we can write $n = 2m+1$, where $2m$ is obviously an even number. Then we divide these $2m$ coins into two groups, each having m number of coins; each of the groups of m coins is kept on each of the pans of the equal arm balance. If they are equal, then the counterfeit coin must be the last coin, i.e. the nth indexed coin. But still, we do not know whether it is heavier or lighter than each of the remaining coins. For that, we compare the last coin (i.e. the nth indexed coin) with the first coin, which is a known correct coin. If the last coin is lighter than the correct coin, we conclude that the nth coin is lighter; otherwise, the last coin must be heavier than the correct coin, and we conclude that the nth coin is heavier.

On the other hand, the equal arm balance must show the case of inequality, where m coins in a group are kept on either of the pans, and the counterfeit coin belongs to amongst the first $n–1$ (= $2m$) coins under consideration, which is a case of finding the counterfeit coin for a given set of even number of coins. Incidentally, this algorithm has been developed in section 3.2.

3.6 Applications of the Problem

The application of the problem under consideration is not only limited to coins, but to any kind of valued article that is frequently counterfeited, like ornaments, watches, metal (like gold), jewelry, etc. A *counterfeit* is an imitation that is made usually with the intent to deceptively represent its content or true origin. The word *counterfeit* most frequently describes the forgeries of currency or documents, but can also describe software, pharmaceuticals, clothing, and more recently, motorcycles and cars, especially when these results in patent or trademark infringement.

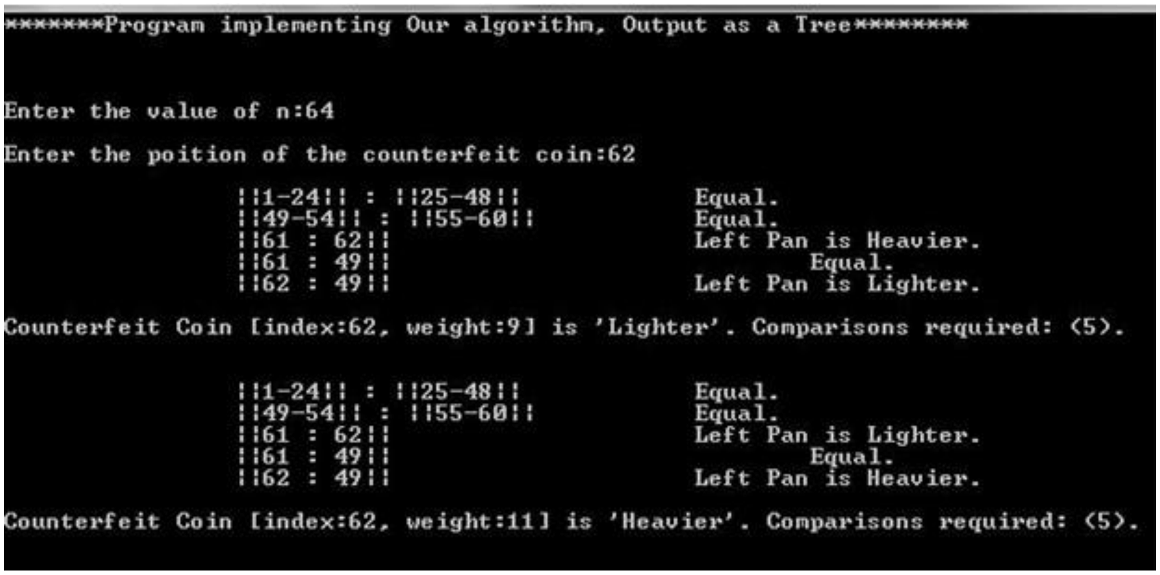
Coin counterfeiting occurs regularly in the antique coin market, but there are various modern forgeries that also make it into general circulation [19, 53]. Counterfeit antique coins are generally made to a very high standard so that they often fool collectors; this is not easy, and many coins still stand out. The importance of solving this problem is more in the theoretical field of computer science and/or mathematics. The very fact that the decision tree structure can be used to solve such problems of large size, by eliminating a part of the solution domain after each step of decision making, is of utmost importance, especially because as our algorithm works for any value of *n*, it does not matter if the value of *n* is not known a priori. This is, in fact, a new innovative work, and important in solving problems using the decision tree structure.

3.7 Experimental Results of the Algorithm

The output of our algorithm for $n = 64$, the counterfeit coin is at 62 is shown in Figure 3.5.

3.8 Summary

In this chapter, we have considered the eight coins problem, one that is well-known in the literature. Only one out of eight coins is a false coin, which is either heavier or lighter. The usual objective of this problem is to find the false coin using a minimum number of comparisons, and in the form of a decision tree.



```
*******Program implementing Our algorithm, Output as a Tree********


Enter the value of n:64

Enter the poition of the counterfeit coin:62

                ||1-24|| : ||25-48||            Equal.
                ||49-54|| : ||55-60||           Equal.
                ||61 : 62||                     Left Pan is Heavier.
                ||61 : 49||                             Equal.
                ||62 : 49||                     Left Pan is Lighter.

Counterfeit Coin [index:62, weight:9] is 'Lighter'. Comparisons required: (5).

                ||1-24|| : ||25-48||            Equal.
                ||49-54|| : ||55-60||           Equal.
                ||61 : 62||                     Left Pan is Lighter.
                ||61 : 49||                             Equal.
                ||62 : 49||                     Left Pan is Heavier.

Counterfeit Coin [index:62, weight:11] is 'Heavier'. Comparisons required: (5).
```

Figure 3.5: Results of our algorithm, which finds the counterfeit coin at 62.

In this chapter, we have developed two new solutions for the eight coins problem, and they are as good as or better than the existing classical solution. Our next target is to find some other equally good or better solutions. Moreover, we would like to consider the twelve coins problem, the sixteen coins problem, and so on, for their probable solutions, and in each only one coin is counterfeit (or false), either heavier or lighter.

At the very beginning, we have made it clear that our objective is to develop an algorithm for the counterfeit coins problem, where the number of coins can be anything, from two to any larger value. We first observe the existing solution for the eight coins problem, and then we modify it slightly to suit our needs. After that, we extract the algorithm behind it and generalize it for all values of *n*, where *n* is an even number. We identify two types of even numbers, ones which are powers of two, and the other is the rest of the numbers, six or more. The algorithm works slightly in different ways for the two types of cases, but the solution is reached in both cases. After developing the algorithm for solving the counterfeit coin problem where the number of coins is even, we widen our algorithm for handling the situation of an odd number of coins problem. This shows that the algorithm developed herein is truly generalized and works for all values of *n* (where *n* is the number of coins). Furthermore, the algorithm terminates after executing for a finite (and predictable) number of steps.