Assignment -3

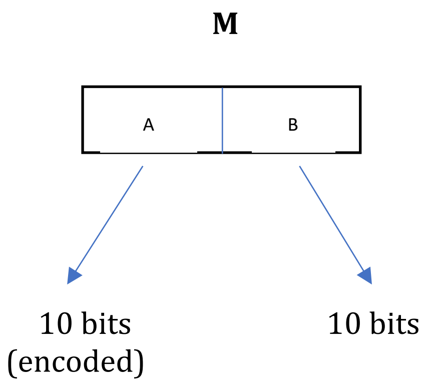
Advanced Algorithms (CPT_S 515)

Name:	Coral Jain
WSU ID:	011632780

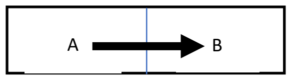
- Let D be a device that keeps sending out messages. Each message contains two parts A and B where the intruder cannot observe A, but he can observe B. Suppose that each of A and B takes 10 bits so, each message is exactly 20 bits. Before the developers sell the device to the public (including the intruder), they run some experiments trying to make sure that there isn't any information leakage that is more than 1 bit from A to B in a message. The experiments are done by run the device for a long time and obtain a large and finite set C of messages. Please design a program that can estimate the average number of bits actually leaked from A to B in a message drawn from C.

Solution:

For this problem, we are given that a message M of device D is a combination of two parts i.e. A and B. Both A and B are of size 10 bits. Also, it is given to us that an intruder cannot observe A, but he can observe B, which means that A is encoded to A' and B is unencoded. This can be represented as given below:



Now we know that there is a leakage of some bits from A to B:



We can consider that as A is encoded to a different kind of representation, which can be a Boolean expression or a formula. So, when A is represented in a different form it can be expressed as A' which can be a combination of Boolean variables that are used to fulfil the functionality of A. Hence, from A' we can figure out the number of bits that part A contains in M.

From the set C which is a collection of many such M messages of different sizes, we can pick any M and know from A part of it to know that number of bits it has. For example, if A is a message of 9 bits, there is a leakage of 10-1=1 bit, if it is a message of 6 bits, there is a leakage of 10-6 = 4 bits. For every run, the leaked number of bits can be identified, and number of bits can be calculated. After retrieving a message M from C, these steps can be as followed as a part of an efficient algorithm:

- ⇨ Get part A from M.
- ⇨ Retrieve Boolean expression of A.
- ⇨ Calculate the number of variables required
- ⇨ Calculate the number of bits.
- ⇨ Figure out the number of bits leakage by subtracting from 10.
- ⇨ Calculate the same by running it over a fixed set of messages in a loop.
- ⇨ Calculate the average for the above.

- Consider a C-function that has integer variables as arguments and integer as return type:

  int myFunction(int x1, int x2, ..., int x7)
  In the function with arguments x1,x2,...,x7 which are integer variables, there are only 10 lines of code, where each line is in the form of an assignment variable := Exp to an integer variable where Exp is a linear combi- nation of integer variables (e.g., y:=2x1+3x2-5) or an if-then-else statement where the condition is a comparison between two linear constraints on integer variables and the assignments in the if-then-else statement are in the form of variable := Exp shown above (e.g., if (y>12x1-z) then x2:=3x7-15 else x5:=18x4-6x7+6. The first line of the function declares three integer variable x, y, z, while the last line is to return the value x back. Please design a program that can verify whether there are values for x1, x2,..,x7 passed to the function that can make the function return a negative integer.

Solution:

Here we are a C program which consists of a function called myFunction which accepts 7 variables as arguments and returns a value of one out of the 3 variables initialized at the start. Thus, we need to figure out by the means of a program whether the values passed as arguments can return the value of x in the form of a negative integer. This question can be solved by the help of a linear programming method as follows:

- ⇨ We need to formulate this problem into a mathematical model and then solve it using the concept linear programming.
- ⇨ If we assume that are the equations are linearly related to each other, we can write all the equations in linear relationship with variable x that gets returned at the end.
- ⇨ Now we must also consider two variables p and q that need to be considered for 2 out of 10 lines of code that is a part of the myFunction.
- ⇨ Now the equation that we need to consider can be given as:

  $x = p + x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + q$

- ⇨ To solve the above equation one of the inequalities that needs to be satisfied should be that variable $x > 0$.
- ⇨ Using the above constraint, we solve for the value of $p + q$.

- ⇨ We also need to remember that there would be a number of values of p + q when we solve all the equations.
- ⇨ Thus, if we add all the values of p + q from all the equations, we must get the value of $(p + q)_{min} = 0$ then the value of x would be -ve.

- Symbolic representation is way to code a finite object. BDD is a way to code a finite set. However, when a power set (a set of finite sets) is given, BDD is not usually efficient. Sometimes, it is a good idea to code an object as a number since a number itself is a string (e.g., 123 is the string ”123”). We now consider a special case. Let K = {1, ..., k} for some k, and consider P be a set of disjoint subsets of K. That is, P = $\{K_1,\cdots,K_m\}$ for some m and each $K_i \subseteq K$ and $K_i \cap K_j = \emptyset$ whenever $i \neq j$. I want to design an algorithm, for a given K, to code (or transform or represent) each such P into a number C_P such that the code C is optimal; i.e.,

(1). C is 1-1,

(2). $C_P \in \{1,...,B_K\}$,
where B_K is the number of all such P’s for the given K.

Solution:

To solve this problem, we need to take help of power set and solve the problem. For this, we consider a power set P of K, denoted as P(K). We are given that K = {1,2……k} is a set and P is a set of disjoint subsets of K. So, we delete the last value or the element from P(K) and find its power set and call it P(PK). After this we calculate value of $P(PK) \cup P(PK) \cup e'$ where e' is the element that we deleted previously. This union will give us a set M.

Now we try to find out each and every disjoint power set of M and then do the same step as above which is to delete the last value/element from the set M. We now find its corresponding power set P(M).

In the next step we calculate the union of P(PK) and $P(PK) \cup e'$ where e' is the element that we deleted previously. Here we also define a counter variable z whose value is 0. For each unique set of disjoint subsets of K from above is assigned a binary value starting from 1 to z.

- Let G be a directed graph of 2048 nodes. When we use a Boolean formula to represent the G, how many Boolean variables are needed in the formula?

Solution:

If a Boolean formula contains $2k$ variables, we will have 2^k nodes.

Here, since we have 2048 nodes, we can represent it as:

⇨ $2^k = 2048$
⇨ $2^k = 2^{11}$
⇨ $\therefore k = 11$

Also, we know that for $k = 11$, we shall have $2k$ Boolean variables i.e. **22 variables**.

- Data types are an abstraction of data and data structures are a way to store the types into memory. In particular, we never store physical objects in memory; in case when we really want to do it, we first represent the physical objects in an abstract representation and store the representation in memory. Here is a problem. There are 40 students in a classroom. I want to design a closet so that any one of the students can be hidden inside. Imagine that the closet is a chunk of computer memory. Then, how big (in bits) closet do you need?

Solution:

The number of students i.e. 40 can be considered to be 40 nodes of a graph which are to be represented in the form of a Boolean expression. For this, we need to consider that for 40 nodes, we must have certain number of variables that need to be taken into consideration in order to represent the students in abstract form.

Since we know that there are $\log_2 n$ number of variables in which the nodes can be represented where n is the number of nodes that need to be represented. Hence, we shall have $\log_2 40$ variables as value of n is 40 here.

Therefore, $\log_2 40 = 5.32$, which can be approximately taken as 6 for the number of bits to be considered since 5 bits will provide less space than required. Hence, we have **6 bits** in which a node or a variable in which it can be stored.