Advanced Algorithms: Homework 3

Ivani Patel

October 20, 2022

1. Let D be a device that keeps sending out messages. Each message contains two parts A and B where the intruder can not 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.

   (a) Given that intruder can not observe A but he can observe B. And information leaked from A to B.
   Step 1: Define A as private variable and B as public variable. Define C which is finite set of messages.

   $$A =< a_1, a_2, ..., a_k >$$
   $$B =< b_1, b_2, ..., b_k >$$

   Step 2: Draw a bipartite graph G taking A and B as nodes.

   $$A(10bits) \Rightarrow B(10bits)$$

   Step 3:In time t to find how many times the message is leaked. Foe each A there is only one B corresponding to it.

   Step 4: Take source node S and end node T where S connects all A nodes and T connects all B nodes. Take M as the sum of edges from the maximum flow algorithm.

   Step 5: Information flow will become $\log_2 M$. For the average of bits actually leaked from A to B we calculate $|logM - (1/n)C|$

2. 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$x_1, x_2, ..., x_7$ 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 combination 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 $(y > 12x_1 - z)$ then $x_2 := 3x_7 - 15$ else $x_5 := 18x_4 - 6x_7 + 6$. 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 $x_1, x_2, ..., x_7$ passed to the function that can make the function return a negative integer.

(a) **Step 1**: Translate each statement into an linear constraint, if it is a linear combination of integer variables. i.e

$$x_1 := x_1 + x_2$$

$$\implies x_1' := x_1 + x_2$$

and all other xn's no change.

Step 2: If it is an if-then-else statement where the condition is a comparison between two linear constraints, split it into two integer constraints. Then repeat assigning new value to xn'.

Step 3: Change RHS to non-negative by moving all elements into RHS to LHS or multiply (-1) on both sides.

Step 4: Combine all statements in program into a big ILP instance. For each constraint, add artificial variables: $\alpha_1, \alpha_2, ..., \alpha_n$ and then we get ILP instance.

Step 5: To get the value of min x observe that

If the minimum of the $x < 0$, then the program returns negative integer is true.

Otherwise, the program returns positive integer is false.

3. 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 = K1, ···, Km for some m and each $Ki \subseteq K$ and $Ki \cap Kj = \phi$ whenever i 6= j. I want to design an algorithm, for a given K, to code (or transform or represent) each such P into a number CP such that the code C is optimal; i.e.,

(1). C is 1-1,

(2). CP $\in 1, ..., BK$,

where BK is the number of all such P 's for the given K.

(a) **Step 1**: As the Bell Number can count the possible partitions of a set, the number of set P's can be calculated by B_k. i.e. if $K = \{1, 2, 3\}$, k =3, $B_k = 5$.

$$k = \{1, 2, 3\} B_3 = 5$$

$$p_1 = \{\{1\}, \{2\}, \{3\}\} \to C_{p_1}$$

$$p_2 = \{\{1, 2\}, \{3\}\} \to C_{p_2}$$

$$p_3 = \{\{1\}, \{2, 3\}\} \to C_{p_3}$$

$$p_4 = \{\{1, 3\}, \{2\}\} \to C_{p_4}$$

Step 2: Use binary number n to represent a C_p. As C is 1-1, we need at least $[\log_2 B_k]$ bits memory to store the total number of C_p.

For instance, if K = 5, $B_k = 15$, we need at least $[\log_2 15] = 4$ bits.

Step 3: All C_p can be represented by a binary number which size is n bits.

4. 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?

   (a) If Boolean formula contains 2k variables, we will have 2^k nodes.
   $2^k = 2048$
   $2^k = 2^{11}$
   $\therefore k = 11$
   We know that for k = 11 we shall have 2k Boolean variables
   $\therefore N = 2k = 2 * 11$
   $\therefore N = 22$

5. 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?

   (a) **Method 1**: 40 students can be considered as 40 nodes of graph which can be represented in form of Boolean expression. 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. So we should have $\log_2 40$ variables as value of n is 40.

   $\therefore \log_2 40 = 5.32$,
   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 can be stored.

   Method 2: We can use binary number to represent students. One binary number for one student and we need to get n bits to store 40 students. So,

   $$2^n \geq 40$$

   $$n \geq 6$$

   So we need at lease 6 bits to store 40 students.