NaCo Exam Solution

Instructor: Dr Giuditta Franco

December the 19th, 2017

1. Let a membrane system have $[_1 c[_2\ ]_2]_1$ as initial configuration, and rules $R_1 = \{a \to b_1 b_2,\ cb_1 \to cb_1{}',\ b_2 \to b_2 e_{in}\ |_{b_1}\}$. Input skin membrane is $[_1\ ]_1$ and output membrane is $[_2\ ]_2$.

   Show first 4 computational steps, starting from the input a^3. Then explain which function $f(n)$ is computed by the system, if the number n is encoded by a^n.

   Sol.

Step 0. $[_1 a^3 c\ [_2\ ]_2]_1$;

Step 1. $[_1 c b_1^3 b_2^3\ [_2\ ]_2]_1$;

Step 2. $[_1 c b_1^2 b_1' b_2^3\ [_2\ e^3\ ]_2]_1$;

Step 3. $[_1 c b_1 {b_1'}^2 b_2^3\ [_2\ e^6\ ]_2]_1$;

Step 4. $[_1 c {b_1'}^3 b_2^3\ [_2\ e^9\ ]_2]_1$. HALT (no more rules may be applied)

   We notice that input a^3 produces output e^9. Analogously, one may verify that a^n produces e^{n^2}, then the system computes the function $f(n) = n^2$.

2. Given a metabolic system with rules $r_1 : c \to a$, $r_2 : a \to b$, $r_3 : a \to bc$, $r_4 : b \to bc$, $r_5 : b \to ab$, and corresponding flux maps: $u_1 = f_1(a,b,c) = ab, u_2 = f_2(a,b,c) = c^2, u_3 = f_3(a,b,c) = 2a, u_4 = f_4(a,b,c) = a, u_5 = f_5(a,b,c) = c$, compute its state X[1], by starting from the initial state X[0] = (2,2,2). Explain the computational step to pass from X[0] to X[1].

   Sol. By EMA:

$$X[1] = A \times U[0] + X[0]$$

   In this case: $A = \begin{pmatrix} 1 & -1 & -1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 \end{pmatrix}$, $U[0] = \begin{pmatrix} 4 \\ 4 \\ 4 \\ 2 \\ 2 \end{pmatrix}$, and $X(0) = \begin{pmatrix} 2 \\ 2 \\ 2 \end{pmatrix}$, therefore $X(1) = \begin{pmatrix} 0 \\ 10 \\ 4 \end{pmatrix}$.

3. *Prove* that any regular language is decidable.

   Sol. $L \in REG \Rightarrow \exists M \in FSA$ recognizing L, that is, M answers to the membership question in a number of steps equal to the length of the input string.

Alternative Sol. Same proof seen in class to show that a monotonic grammar generates a decidable language may be applied for a grammar of type 3 (which is monotonic as well).

4. Enunciate the first theorem of Shannon.

   Sol. The theorem claims that no code of an information source may reach an average encoding length smaller than the entropy of the source:

   $$H(X,p) \leq L_C$$

   where C is any code (surjective function from the encodings to the data) of an information source (X,p), L_C is its average length, defined as $\sum_{w \in C} |w| p(w)$, and $H(X,p)$ is the entropy of source (X,p).

5. Exhibit one non-decidable language from the class RE.

   Sol. Given an alphabet, first enumerate all possible words over the alphabet $\alpha_i, i \in \mathbb{N}$, and all possible grammars (of type 0) $G_i, i \in \mathbb{N}$. Given this algorithm: $\forall (i,j) \in \mathbb{N} \times \mathbb{N}$ `if` G_i `has generated` α_i `in` j `steps, then` $\alpha_i \in K$, we have defined an RE language

   $$K = \{\alpha_i / \alpha_i \in L(G_i)\}.$$

   We prove that $\overline{K} = \{\alpha_i \ / \ \alpha_i \notin L(G_i)\}$ is not in RE (that is, the language is outside RE). Indeed, by contradiction, if $\overline{K}$ would be in RE (since $RE = \mathcal{L}_0$) then it would exists $d \in \mathbb{N}$ such that $\overline{K} = L(G_d)$. Contradiction is obtained by the question: $\alpha_d \in \overline{K}$? In fact,

   $$\alpha_d \in \overline{K} \Leftrightarrow \alpha_d \in L(G_d) \Leftrightarrow \alpha_d \in K \Leftrightarrow \alpha_d \notin \overline{K}.$$

   By the Post theorem, to be decidable an RE language needs to have its complement contained in RE. Therefore, K is non-decidable language from the class RE.

6. Solve only one of the following exercises:

   (a) Explain the quaternary recombination algorithm, and prove its correctness.

   (b) Explain one DNA computing algorithm solving SAT, both in formal and implementation terms.

   Sol (a). The quaternary recombination algorithm generates an n-dimensional DNA library of binary strings starting from one pool P_0 containing four specific strings (all α-prefixed and β-suffixed for given oligos α and β): $I_1 = X_1X_2X_3X_4X_5 \ldots X_n$, $I_2 = Y_1Y_2Y_3Y_4Y_5 \ldots Y_n$, $I_3 = X_1Y_2X_3Y_4X_5 \ldots$, $I_4 = Y_1X_2Y_3X_4Y_5 \ldots$.

   Let P_1 and P_2 be two copies of the pool $P_0 = \{\alpha I_1 \beta,\ \alpha I_2 \beta,\ \alpha I_3 \beta,\ \alpha I_4 \beta\}$.

   for $i = 2,3,4,5$ **do**

   - `perform` $XPCR_{X_i}$ on P_1 and $XPCR_{Y_i}$ on P_2;
   - `mix` $P := P_1 \cup P_2$;
   - $(P_1, P_2) :=$ `split`(P)

The above algorithm is correct, as for any binary assigment $\alpha_1\alpha_2\ldots\alpha_n$, there exists a sequence of recombination rules r_{α_j} (implemented by $XPCR_{\alpha_j}$) which generates it starting from the four axioms. Such a sequence may be infered by the following algorithm.

Let us call l_i the initial sequence containing $\alpha_{i-1}\alpha_i$ as subsequence, for $i = 2, \ldots, n$, and let c, s_1, s_2 be string variables.

$c := l_2$
`for` $j = 2, \ldots, n-1$
apply $r_{\alpha_j} : \ c,\ l_{j+1} \ \longrightarrow \ s_1,\ s_2;$
$c := s_1$

Sol (b). Namely, we choose to explain the Sakamoto's algorithm solving SAT. (This algorithm could have been explained much more shortly, as we have done in class).

An instance 3-SAT(n, m) is solved first by generating DNA molecules containing, for each clause, one out of its three literals (that is, molecules with m literals, one for each clause $C_i = l_1^{(i)} \vee l_2^{(i)} \vee l_3^{(i)}$, $i = 1, 2, \ldots, m$), and then by extracting all non-contradictory assignments for the literals. These are the solutions of the instance.

`INPUT:` $P = \{\alpha C_1, \overline{C_1} L_1 C_2, \ldots, \overline{C_{m-1}} L_{m-1} C_m, \overline{C_m} L_m \beta \ \mid \ |C_i| = |\overline{C_i}| = 20$ long sticky ends, $|\alpha| = |\beta| = 20\ L_i \in \{l_1^{(i)}, l_2^{(i)}, l_3^{(i)}\}, |L_i| = 20,\ \gamma_z \in L_i,\ i = 1, \ldots, m, \gamma_z$ restriction site of the enzyme Enz$\}$.

`ALGORITHM:`

(a) $P := Lig(C(P))$; \\ Assembly of 3^m assignments satisfying all clauses
(b) $P = PCR(\alpha, \overline{\beta})(P)$; \\ material amplification
(c) $P = H(P)$; \\ heating to get single filaments
(d) $P = Enz(C(P))$; \\ sudden cooling, to form hairpins and immediate cut by the enzyme
(e) $P = El_{40(m+1)}(P))$; \\ selection of filaments of initial lenght
(f) $P = PCR(\alpha, \overline{\beta})(P)$; \\ material amplification
(g) `if` $El(P) \neq \emptyset$ `then X= Yes else X=NO`

`OUTPUT:` Value of X.

7. Given the following pattern: $(ab)^n + (bc)^m$ with $n, m \in \mathbb{N}$, provide the corresponding regular expression, the FSA recognizer, and a Chomsky grammar generating it.

Sol. Regual expression: $(ab)^* + (bc)^*$. Finite state automaton, recognizing $(ab)^* + (bc)^*$:
$q_0 a \to q_1$
$q_1 b \to q_2$
$q_2 a \to q_1$
$q_0 b \to q_3$
$q_3 c \to q_4$
$q_4 b \to q_3$
q_0 initial and final state, q_2 and q_4 final states.

Grammar (of type 3, directly deduced by the automaton above) generating $(ab)^* + (bc)^*$:
$q_0 \to a q_1$
$q_1 \to b q_2$
$q_1 \to b$
$q_2 \to a q_1$
$q_0 \to b q_3$

$q_3 \to cq_4$
$q_3 \to c$
$q_4 \to bq_3$
$q_0 \to \lambda$
where $q_0 = S$, and q_1, q_2, q_3, q_4 are non-terminal symbols.

Alternative grammar:
$S \to \lambda$
$S \to abS$
$S \to ab$
$S \to S'$
$S' \to bcS'$
$S' \to bc$

8. Give a definition for:

   - Computation: Process performed on a physical system, assuming a finite number of states (namely, an initial and a final one), by means of a list of instructions and a termination criterion.

   - Natural Computing: It is an instance of computation inspired by or performed by natural systems.

   - Information: A set of data which may be stored, transformed, and transmitted.
     The information of an event is a function of its probability/distribution.
   - Enzymatic paradox: Enzymes (are proteins which) catalize/activate biochemical reactions, producing enzymes. Which of them comes first?

   - (Given a dictionary D and a genome G) Average genomic positional coverage: The average (over all the positions of the genome) number of words of D that cover each position p. A position p of genome G is covered by the word $G[i, j] \in D$, if $i \leq p \leq j$.

   - Minimal Forbidden Length: Let Σ denote the DNA alphabet, and G a genome. $MFL = \min\{k : D_k(G) \neq \Sigma^k\}$, where $D_k(G)$ is the dictionary of k-mers occurring in the genome.

   - Optimal code: A code C is optimal if no code C' exists with a smaller average length (in other terms, $L_{C'} > L_C$ for all other codes C').

   - Evolutionary computing: Computation inspired by evolution theories, both Darwinian and Genetic drift.
     More in details, we are given with a *fitness function* F and a *generation mechanism* G. Over an initial population, evaluate the fitness function. While F is under a threshold, *select* a subset and *expand* it by G. Stop when F exceeds the threshold, or a prefixed number of steps has been executed.
     - `Initialize` S with a set S_0 of possible solutions
     - `Evaluate` a fitness level F over S, and `while` F is under a given threshold, do
       * Select a subset S' of S
       * Expand S' into a population S according to G
     - **Output** the population S reaching the fitness threshold.