14/01/2021

Y3878747
exam Candidate number

Computer Science BSc
Theory 3
Closed Examination

Question 1:

(i) $S \Rightarrow Sb \Rightarrow Sbb \Rightarrow aaabb \Rightarrow aabab \Rightarrow abaab \Rightarrow ababa$

(ii) The language ~~[illegible]~~ $L(G)$ is the set of all strings ~~[illegible]~~ over the alphabet $\Sigma = \{a, b\}$, where all strings contain ~~only~~ three as, ~~and many~~ as well as any number of bs (zero or more bs).

(iii) $b^*ab^*ab^*ab^*$

(iv) $S \rightarrow BABABAB$

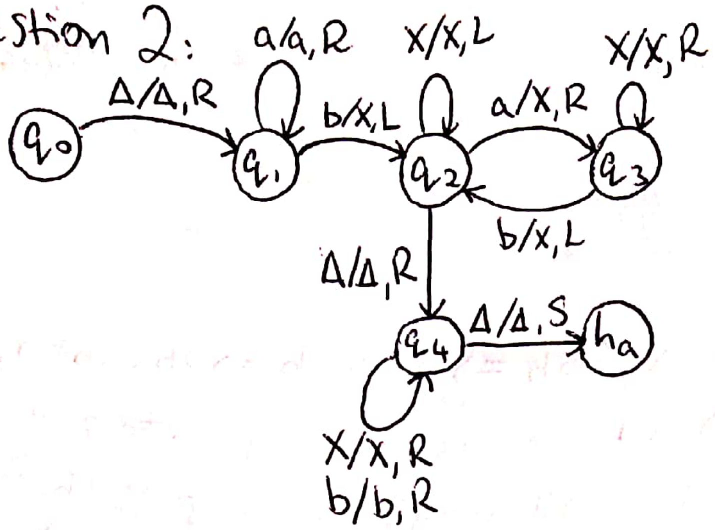
$A \rightarrow a$

$B \rightarrow Bb \mid \Lambda$

THE3
Exam
Y3878747

Question 2:

(i)



(ii) $q_0\Delta aabbb \vdash \Delta q1aabbb \vdash \cancel{\Delta q1a}\ \Delta a q1abbb \vdash \Delta aaq1bbb$
$\vdash \Delta aq2aXbb \vdash \Delta aXq3Xbb \vdash \Delta aXXq3bb \vdash \Delta aXq2XXb$
$\vdash \Delta aq2XXXb \vdash \Delta q2aXXXb \vdash \Delta Xq3XXXb \vdash \Delta XXq3XXXb$
$\vdash \Delta XXXq3Xb \vdash \Delta XXXXq3b \vdash \Delta XXXq2XX \vdash \Delta XXq2XXX$
$\vdash \Delta Xq2XXXX \vdash \Delta q2XXXXX \vdash q2\Delta XXXXX \vdash \Delta q4XXXXX$
$\vdash^5 \Delta XXXXXq4\Delta \vdash \Delta XXXXXha\Delta$

(iii) aaab

(iv) $L(M) = (a^n b^m \mid n \geq 0 \text{ and } m \geq 1 \text{ and } m > n)$

M's behaviour consists of crossing off the first bit detects on input. If it is the first input symbol, ~~it~~ the accept state is reached provided the rest of the input consists only of bs. If it is preceeded by as, one is crossed off (the a closest to the first b) and subsequently the ~~a~~ b ~~closest~~ to the a just crossed off is crossed off. This crossing off a b for every a ensures that the number of as never exceedes the number of bs for the input ~~state~~ string to reach the accept state.

THE 3
Exam
Y3878747

Question 3

(i)

$$f_M(n) = \begin{cases} 1 & \text{if } n = 4m \text{ for any natural number } m \\ & \text{(i.e. } n \text{ is a multiple of } 4\text{),} \\ \text{undefined} & \text{otherwise} \end{cases}$$

(ii) A total function is a function defined for all possible input values. Thus, a TM computing a total function will accept every input. The Turing Machine M reaches the accepting state if the input string represents a zero, or if the the input string's first two symbols collectively represent a 2. It doesn't read the rest of the symbols of the input string. Therefore, all strings representing values greater than or equal to 2 will be accepted. Even Λ representing 0 will be accepted. However, this TM does not accept an input string 1 which represents the natural number 1. Therefore not all possible inputs reach an accepting state, producing an output. Therefore f_M is not a total function.

Question 4:

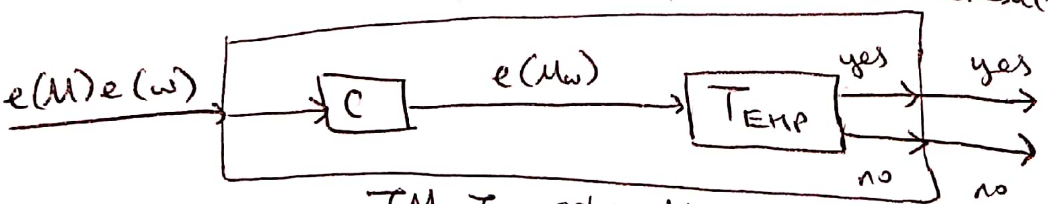
We reduce the halting problem (HP) to the existential halting problem (EHP), which seeks to solve the question of if there exists an input string for M, a machine inputted into EHP, upon which M can reach a halting configuration. To reduce$_E$ means to show that the decidability of EHP implies the decidability of ~~H~~ HP.

Supposing that EHP is decidable: a ~~TM~~ Turing Machine T_{EHP} exists which solves the EHP. Then, T_{EHP} takes as input the encoding e(M) of ~~TM~~ any TM M, and accepts M if there exists an input string on which M reaches a halting configuration. Else, T_{EHP} rejects.

Using T_{EHP}, T_{HP}, a TM solving the halting problem (if the input machine M will ever reach a halting configuration on an input w [both M and w (where w is a string over M's input alphabet) are inputs to T_{HP}]), can be constructed. On input e(M), T_{HP} produces e(M_w) — a machine constructed off of M ~~...~~ and with input w. T_{HP} produces M_w such that the first thing M_w does is to erase its own input off ~~the~~ its tape, and write w on the tape, and run ~~...~~ M on w. Upon these actions, M may either accept, reject or loop on w. Next, T_{HP} runs T_{EHP} on input e(M_w). T_{HP} accepts if T_{EHP} accepts, otherwise, T_{HP} rejects.

We have constructed M_w such that if ~~...~~ M halts on no inputs, M_w will ~~...~~ not halt on any input. Comparatively, if M halts on one input, M_w will halt on all inputs.

T_{HP} runs T_{EHP} on input e(M_w) and returns the result:



TM T_{HP} solving HP

Since ~~...~~ M only accepts w if and only if ~~...~~ M_w accepts w, T_{HP} accepts ~~...~~ e(M)e(w) if and only if M halts on input w. Thus T_{HP} solves the (general) halting problem. However, ~~...~~ the (general) halting problem is unsolvable, as proven by Alan Turing. Therefore a Turing machine ~~solving one~~ solving EHP, as we assumed, cannot exist, so our assumption was incorrect. Thus EHP is undecidable.

THE 3
Exam
Y3878747

Question 5

(i)

$$f_M(w) = \begin{cases} \Lambda & \text{if } w = u\,aa\,v, \text{ for } u, v \in \Sigma^*, \\ w & \text{otherwise} \end{cases}$$

(ii)

$$\tau_M(n) = \begin{cases} 2n+4 & \cancel{\text{if input } w \text{ (of length } n\text{) contains } aa}, \\ \cancel{2n+3} & \cancel{\text{otherwise}} \end{cases}$$

(iii)

$$s_M(n) = n + 2$$

THE 3
Exam
Y3878747

Question 6:

(i) $q_0 \Delta aaa \vdash \Delta q1 aaa \vdash \Delta a q1 aa \vdash \Delta aa q1 a$
$\vdash \Delta a q2 aa \vdash \Delta q2 aaa \vdash \Delta a q3 aa \vdash \Delta aa q3 a \vdash \Delta aaa q3 \Delta$
$\vdash \Delta aaa h \Delta$

(ii)

$$T_M(n) = \begin{cases} n+1 & \text{if } n \leq 1, \\ 3n+1 & \text{otherwise} \end{cases}$$

(iii) $S_M(n) = n+2$

THE 3
Exam
Y3878747

Question 7:

(i) 2^n is of order $n!$ if there are positive integers C and n_0, such that $2^n \le C \cdot n!$, for all $n \geq n_0$.

induction ~~xxxx~~ hypothesis
let $n_0 = 4$ and $C = 2$, such that $2^n \le C \cdot n!$
induction basis:
let us substitute $n = 4$ into the inequality.

$$2^n \le C \cdot n!$$
$$2^n \le 2 \cdot (n!)$$
$$2^4 \le 2 \cdot (4!)$$
$$16 \le 2 \cdot (24)$$
$$16 \le 48$$

induction step: (by the induction hypothesis)

$$(2^n)(n+1) \le C \cdot (n!)(n+1)$$
$$2^n(2 + (n-1)) \le C \cdot (n+1)!$$
$$\underbrace{2^{n+1} + 2^n(n-1)}_{> 2^{n+1}} \le C \cdot (n+1)!$$

(we know that $n+1 \geq 5$ as $n_0 = 4$ in hypothesis) So we know $n > 2$, thus we can infer that $2^n(n+1)$ is greater than 2^{n+1})

∴ can conclude $2^n \in O(n!)$

(ii) Suppose that $n^4 - 2n \in O(n^3)$. Then positive integers C and n_0 exist, such that $n^4 - 2n \le C \cdot n^3$, for all $n \geq n_0$. Let m be the maximum value ~~n_0~~ between n_0 or $(C+1)$.

Then $m > C$. And:
$$n^4 - 2n \le C \cdot n^3$$
$$m^4 - 2m \le C \cdot m^3$$
$$m(m^3 - 2) \le (C \cdot m^2)m \quad \text{(dividing by m as we know m is positive)}$$
$$m^3 - 2 \le C \cdot m^2$$
$$m^3 \le C \cdot m^2 \quad \text{(as long as } m > 1 \text{, the -2 doesn't affect the inequality so discard it)}$$
$$m \cdot (m^2) \le C \cdot (m^2) \quad \text{(can divide by positive square values and inequality)}$$
$$\boxed{m \le C}$$
however, this is a contradiction to the inequality stated above: $m > C$.

Hence the assumption that $n^4 - 2n \in O(n^3)$ is false. ∴ $n^4 - 2n \notin O(n^3)$

THE 3
exam
Y3878747

Question 8:

The ~~Satisfy~~ satisfiability of a whole expression in DNF can be narrowed down to if any clause (where a clause is defined as a sub-expression; in the case of a DNF, a ~~disjunct~~ each clause is seperated by a disjunction) in the whole expression is satisfiable by a certain assignment of truth values to the variables in the clause, making the clause a tautology as well as the whole expression itself (due to its DNF nature).

A DNF expression D is a tautology if and only if a disjunct D_i exists which evaluates to true. ~~To~~ In order to be more precise, a key observation could be that D is a tautology if and only if D contains a clause which does not contain two complementary literals (where one is a variable, and the other ~~is its negation~~ the former's negation). If this is not the case, the whole expression of D will be made false as none of the clauses would evaluate to true, regardless of boolean variables set (e.g the expression $x \wedge \neg x$ would never evaluate to true no matter ~~what~~ if we assign true or false to x.)

A Turing Machine can check in polynomial time whether there exists at least one clause which does not contain ~~two~~ a pair of complementary literals. ~~The machine scans~~ For each clause (starting from the beginning of the expression (the left) to the end (the right most clause of the expression)), the Machine scans from the beginning of the clause to the end of it, marking a literal and moving to the right until either the complementary literal is found, or the end of the conjunct is reached. If complementary literals aren't found in a clause, D ~~is boolean~~ is satisfiable and the procedure terminates. If complementary literals are found, the TM moves onto checking the next clause ~~of~~ in the expression. If ~~then~~ all the clauses in the expression have been checked, and all of them contain complementary literals, the procedure terminates with the answer that D is unsatisfiable.

Sources used:
- THEORY 3 lectures and problem sheets and past papers
- Introduction to Languages and the theory of computation textbook.