CS320: Problems and Solutions for Day 13, Winter 2023

Problem 1 **(a)** Let G be a grammar that defines the C++ programming language. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary string p of characters from the legal C++ character set.

QUESTION: Does p represent a valid C++ program?

Explain your answer. If such algorithm does not exist, prove it.

Answer: The program should simulate the push-down automaton which accepts all strings that are valid C++ programs; this automaton is obtained by an algorithmic conversion of the grammar G.

(b) Let G be a grammar that defines the C++ programming language. In an introductory programming class, students write C++ programs for verifying if a sequence of five integers is sorted. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary student program p and an arbitrary sequence s of five integers.

QUESTION: Does p correctly verify s?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist, by Rice's Theorem, even for a fixed s. Let r be any program that correctly verifies if a given fixed sequence s of five integers is sorted. (Observe that, for a given fixed sequence, r may simply write the answer and do nothing else.) The property of being equivalent to r is non-trivial, since r has it and many other programs do not.

(c) Let G be a grammar that defines the C++ programming language. In an introductory programming class, students write C++ programs for sorting a sequence of integers of arbitrary length. There are only two grades: pass and fail. A student program passes if it correctly sorts the benchmark input sequence within two seconds of processor time; otherwise the program receives the grade fail. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: An arbitrary student program p and an arbitrary integer sequence s.

QUESTION: Does p pass if tested on s as the benchmark?

Explain your answer. If such algorithm does not exist, prove it.

Answer: Execute the student program p on input s for two seconds. After two seconds have expired, if the program p has arrived to the correct answer, output pass, else output fail.

(d) Let G be a grammar that defines the C++ programming language, as implemented by a major software vendor, named X, and let P be the compiler manufactured by this vendor. At a major university, students are writing C++ compilers in their software design courses, as programming exercises. Company X is interested in hiring students that write good compilers. Company X evaluates a compiler as good if it compiles the same set of programs as its own compiler P. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: Compiler P developed by X, and an arbitrary compiler P_1 written by a student.

QUESTION: Is P_1 a good compiler?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist, by Rice's Theorem, even for a fixed P. The property of being equal to the set of strings accepted by P is non-trivial, and we cannot decide if the language accepted by P_1 has this property.

(e) The scenario is identical to that given in part (d), except that Company X evaluates a compiler as good if it compiles the same set of programs as its own compiler P and, additionally, is never slower than P by a factor greater than 2. Describe the algorithm that should be employed by a program that solves the following problem:

INPUT: Compiler P developed by X, and an arbitrary compiler P_1 written by a student.

QUESTION: Is P_1 a good compiler?

Explain your answer. If such algorithm does not exist, prove it.

Answer: This algorithm does not exist, for the reasons stated in part (d)—the new condition is a conjunction of two non-trivial properties.

Problem 2 **(a)**
Let M_1 be a Turing machine that decides language L_1;
let M_2 be a Turing machine that accepts language L_2;

let M_3 be a Turing machine that accepts language L_3.

Describe a Turing machine M that accepts language:

$$L_1 \cap (L_2 \cup L_3)$$

If such Turing machine does not exist, prove it.

Answer: M simulates M_1 until M_1 halts. If M_1 rejects then M also rejects. If M_1 accepts them M continues as follows. M simulates both M_2 and M_3 in parallel and halts and accepts if and when any of them accepts.

(b)
Let M_1 be a Turing machine that accepts language L_1;
let M_2 be a Turing machine that decides language L_2;
let M_3 be a Turing machine that decides language L_3.

Describe a Turing machine M that accepts language:

$$L_1 \setminus (L_2 \cap L_3)$$

If such Turing machine does not exist, prove it.

Answer: M simulates both M_2 and M_3 in parallel until they both halt. If both M_2 and M_3 accept, then M rejects. Otherwise, M simulates M_1 and accepts if and when M_1 accepts.

(c)
Let M_1 be a Turing machine that decides language L_1;
let M_2 be a Turing machine that accepts language L_2;
let M_3 be a Turing machine that accepts language L_3.

Describe a Turing machine M that decides language:

$$L_1 \setminus (L_2 \cap L_3)$$

If such Turing machine does not exist, prove it.

Answer: Impossible. If M existed, we could set:

$$\begin{aligned} L_1 &= \Sigma^* \\ L_2 &= L_H \\ L_3 &= \Sigma^* \end{aligned}$$

where:

$$L_H = \{(M, w) \mid (M, w) \searrow\}$$

yielding:

$$L_2 \cap L_3 = L_H$$

Hence, M would decide:

$$L \setminus L_H = \overline{L_H}$$

which is impossible, since $\overline{L_H}$ is not even recursively enumerable.

Problem 3 Let M_1 and M_2 be two arbitrary Turing machines over input alphabet Σ. For each of the following six questions, determine if the answer is always **yes**, always **no**, or sometimes **yes**. Justify your answer in each case.

(a) Is $L(M_1) = \emptyset$?

Advice for Answer: Sometimes.

(b) Is $L(M_2) = \Sigma^*$?

Advice for Answer: Sometimes.

(c) Is $L(M_1)$ recursive?

Advice for Answer: Sometimes.

(d) Is $L(M_2)$ recursively enumerable?

Advice for Answer: Always.

(e) Is $L(M_1) = L(M_2)$?

Advice for Answer: Sometimes.

(f) Is $L(M_1) \cup L(M_2)$ recursively enumerable?

Advice for Answer: Always.

Problem 4 Let L be a non-recursive language over the English alphabet $\{a, b, c, \ldots, x, y, z\}$, accepted by a Turing machine M. Describe a Turing machine M' such that M' writes *error* on its tape and halts if and only if its input string does not belong to L. If such M' does not exist, explain why.

Answer: M' does not exist, lest it would accept $\overline{L}$. This would mean that $\overline{L}$ is recursively enumerable. However, since L is not recursive, its complement $\overline{L}$ cannot be recursively enumerable.