Computability and Complexity

Department of Computer Science, University of Copenhagen, Julia Lawall

Written exam for 4 hours, January 26th, 2011

All usual aids such as books, notes, exercises and the like may be used during the exam, but no calculators, computers, cell phones or similar equipment. The exam may be answered in pencil.

This exam set has 2 pages. It consists of 5 questions, and a total of 12 parts. Within each question, the parts are equally weighted. All parts are worth 8 or 9 points. Do not spend too much time on any question. Some parts have subparts, which are designed to help you structure your answer.

Question 1: (24 points)

Part 1: Show that the language $\{a^n b^i \mid n \leq i \leq 2n\}$ is not regular.

Part 2: Show that the language $\{a^n b^i \mid n \leq i \leq 2n\}$ is context free.

Part 3: Show that the language $\{a^n b^n c^i \mid n \leq i \leq 2n\}$ is not context free.

Question 2: (24 points)

Part 1: Consider the extension of a pushdown automaton to a 2-stack pushdown automaton. A 2-stack pushdown automaton is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ where Q, Σ, Γ and F are all finite sets, and

1. Q is the set of states.
2. Σ is the input alphabet.
3. Γ is the stack alphabet.
4. $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \times \Gamma_\varepsilon \to \mathcal{P}(Q \times \Gamma_\varepsilon \times \Gamma_\varepsilon)$ is the transition function.
5. $q_0 \in Q$ is the start state.
6. $F \subseteq Q$ is the set of accept states.

Define a 2-stack pushdown automaton that recognizes $\{a^n b^n c^i \mid n \leq i \leq 2n\}$.

Part 2: Show that for every 2-stack pushdown automaton there is an equivalent deterministic single-tape Turing machine. (Hint: Given the various components of a 2-stack pushdown automaton, show how to define each of the 7 components of a deterministic single-tape Turing machine. Explain briefly each of part of your construction.)

Part 3: Show that for every single-tape deterministic Turing machine there is an equivalent 2-stack pushdown automaton. (Hint: Use one stack to hold what is to the left of the tape head and another stack to hold what is to the right of the tape head.)

Question 3: (16 points)

Part 1: Consider the problem of determining whether a Turing machine M on an input string w ever attempts to move its head left.

> **Subpart 1:** Formulate this problem as a language.
>
> **Subpart 2:** Show that this language is **decidable**.

Part 2: Consider the problem of determining whether a Turing machine M on an input string w ever attempts to move its head left when it is on the leftmost tape cell.

> **Subpart 1:** Formulate this problem as a language.
>
> **Subpart 2:** Show that this language is **undecidable**.

Question 4: (18 points)

Part 1: Consider the language *INTERSECT*, defined as follows:

$$\mathit{INTERSECT} = \{\langle \{A_1, \ldots, A_n\}, k\rangle \mid A_1, \ldots, A_n \text{ are sets and } k \text{ is an integer, such that there exists a set } C \text{ of size } k \text{, such that for every } 1 \leq i \leq n,\ A_i \cap C \neq \emptyset\}$$

> **Subpart 1:** Give an example of a string that is in this language, and another example of a string that is not in this language.
>
> **Subpart 2:** Is *INTERSECT* in P or is it NP complete? Prove your answer.

Part 2: Consider the language *SUB-SUM*, defined as follows:

$$\mathit{SUB\text{-}SUM} = \{\langle \{w_1, \ldots, w_n\}, \{v_1, \ldots, v_n\}, w, v\rangle \mid \text{there exists a set } S \subseteq \{1, \ldots, n\} \text{, such that } \Sigma_{i\in S}\, w_i \leq w \text{ and } \Sigma_{i\in S}\, v_i \geq v\}$$

> **Subpart 1:** Give an example of a string that is in this language, and another example of a string that is not in this language.
>
> **Subpart 2:** Is *SUB-SUM* in P or is it NP complete? Prove your answer.

Question 5: (18 points)

Part 1: Say that two Boolean formulas are equivalent if they have the same set of variables and are true on the same set of assignments to those variables. A Boolean formula is minimal if no shorter Boolean formula is equivalent to it. Let *MIN-FORMULA* be the collection of minimal Boolean formulas. Show that *MIN-FORMULA* is in PSPACE (the polynomial space complexity class).

Part 2: Let B be the language of properly nested parentheses and brackets. For example, `([()()]()[])` is in B, but `([)]` is not. Show that B is in L (the log space complexity class).

END OF THE EXERCISES.