Qualifier Exam — FOUNDATIONS
SAMPLE, October 1995

1. In this question be sure to distinguish between a language and a grammar or automaton generating it. Justify all your answers at a level of detail suitable for a knowledgeable reader.

   (a) Explain how each grammar and each automaton can be viewed as a finite set of finite strings.

   (b) Define the notion of *regular expression.*

   (c) Sketch an algorithm that when given a finite-state automaton converts it into an equivalent regular expression. Give an upper bound on the number of unions and concatenations in the output regular expression.

   (d) Consider (without necessarily describing it) an algorithm that, given a context-free grammar G in Chomsky Normal Form and a string x, determines whether $x \in L(G)$. Give an upper bound on the number of set products that the algorithm computes as a function of the sizes of G and x. Give an upper bound on the number of set products computed by the algorithm.

2. (a) Define the notions of *problem*, *poly-time reductions* between problems, and *NP-completeness.*

   (b) Let us say that a boolean expression φ with an even number of variables is *symmetric-satisfiable* if there is a truth-valuation that satisfies φ, and under which exactly half of φ's variables are assigned the value **true**. Show that symmetric satisfiability is NP-complete. (You may use without proof well-known examples of NP-complete problems.)

3. Let C be the set of descriptions of those Turing machines that converge for some input.[1]

   (a) Is the set C decidable?

   (b) Is it semi-decidable (i.e. recursively enumerable)?

   (c) Let D be the set of descriptions of those Turing machines that converge for no input. Is D semi-decidable?

4. (a) Is the following formula of first order logic provable?

   $$\forall x\,(\mathbf{P}(x) \to \mathbf{P}(\mathbf{f}(x))) \to \forall x\,(\mathbf{P}(x) \to \mathbf{P}(\mathbf{f}(\mathbf{f}(x)))$$

   If it is, exhibit a complete formal proof for it.[2] If it is not, give a structure $\mathcal{S}$ in which the formula is false, where the universe of $\mathcal{S}$ has exactly two elements.

[1]Descriptions of Turing machines can be taken to be some textual listing of the transition tables, coded in binary using ASCII.

[2]You may refer to any standard formalism for first order logic with which you are familiar, such as a sequential calculus, a natural-deduction system, or a tableaux system, but you must state which formalism you use.

(b) Same question for the formula $\forall x\,(\mathbf{P}(x) \vee \mathbf{Q}(\mathbf{f}(x))) \;\rightarrow\; (\forall x\,\mathbf{P}(x) \vee \forall x\,\mathbf{Q}(\mathbf{f}(x)))$

(c) Raymond Smullyan has observed that in every party he has been to, there was always some person p such that if p drank, then everyone drank. Using $\mathbf{D}(p)$ to express "person p drinks", write a first order formula that conveys Smullyan's observation. Is your formula provable? Answer as in the previous parts.

5. It is known that the provability of formulas of first order logic is undecidable.

   (a) Is there a program that, when given a formula φ as input, returns 1 if φ is valid (that is, true in all structures) and 0 if it is not?

   (b) Is there a program that when given a formula φ as input, returns 1 if φ is valid, and does not terminate if it is not?

   (c) Is there a program that, when given a formula φ as input, returns 1 if φ is valid, and 0 if there is a finite structure in which φ is false?

   Explain your answers convincingly.