# Theory Qualifier
# April 1, 1988

## Instructions

Four hours, closed book. All questions are equal weight.

There are two parts, a *basic part* and an *advanced part*, each consisting of four questions. Of the four advanced questions, there is one from each of the following three areas:

1. recursive function theory and logic,
2. algorithm design and analysis,
3. computational complexity.

In addition, there is one "wild card" question. For a *basic pass*, you must answer all four basic questions satisfactorily. For an *advanced pass*, you must answer all four basic questions and any two of the four advanced questions satisfactorily. Indicate clearly which two advanced questions you are attempting; only two will be graded.

The questions will be graded according to the following criteria, in order of importance:

1. correctness and completeness
2. clarity, precision, and conciseness
3. optimality of result

Partial credit will be awarded where appropriate. Unclear presentations and irrelevancies will be penalized. *Please read all questions carefully before answering.*

**GOOD LUCK!**

## Part I—Basic

1. An *Eulerian circuit* in an undirected graph $G = (V, E)$ is a cycle (not necessarily simple) that contains each edge exactly once.

   (a) Prove that if $G$ has an Eulerian circuit, then each of its vertices must have even degree.

   (b) Prove the converse of (a) by giving an algorithm that finds an Eulerian circuit in a connected graph all of whose vertices have even degree. For full credit, your algorithm must be $O(|E|)$.

2. For each of the following classes of automata, tell whether the deterministic and nondeterministic versions are (i) equivalent, (ii) inequivalent, or (iii) not known to be equivalent or inequivalent:

   (a) finite automata

   (b) two-way finite automata

   (c) pushdown automata

   (d) automata with a one-way read head and $k$ pushdown stores

   (e) linear bounded automata

   (f) polynomial time-bounded Turing machines

   (g) polynomial space-bounded Turing machines

   (h) Turing machines

3. Consider the context-free grammar $G$ with start symbol $S$ and productions

$$\begin{aligned} S &\rightarrow aAB \mid aBA \mid bAA \mid \epsilon \\ A &\rightarrow aS \mid bAAA \\ B &\rightarrow aABB \mid aBAB \mid aBBA \mid bS \end{aligned}$$

   Prove that $L(G)$ is the language consisting of all words that have exactly twice as many $a$'s as $b$'s.

4. Consider the following scheduling problem:

   **Instance:** An undirected graph $G = (V, E)$ whose vertices specify unit-time jobs and whose edges specify that the two jobs cannot be scheduled simultaneously, and positive integers $m$ and $t$.

   **Question:** Can the jobs be scheduled on $m$ identical processors in $t$ time units?

(a) Show that the problem is $NP$-complete for $t \geq 3$.

(b) Give a polynomial time algorithm for $t = 2$.

## Part II—Advanced

1. (recursive function theory and logic)

   (a) Prove that the following 3-valued truth table does not correctly characterize the intuitionistic propositional connectives.

| $p$ | $q$ | $p \to q$ | $p \vee q$ | $p \wedge q$ | $\neg p$ |
|---|---|---|---|---|---|
| $t$ | $t$ | $t$ | $t$ | $t$ | $f$ |
| $t$ | $\perp$ | $\perp$ | $t$ | $\perp$ | $f$ |
| $t$ | $f$ | $f$ | $t$ | $f$ | $f$ |
| $\perp$ | $t$ | $t$ | $t$ | $\perp$ | $\perp$ |
| $\perp$ | $\perp$ | $\perp$ | $\perp$ | $\perp$ | $\perp$ |
| $\perp$ | $f$ | $\perp$ | $\perp$ | $f$ | $\perp$ |
| $f$ | $t$ | $t$ | $t$ | $f$ | $t$ |
| $f$ | $\perp$ | $t$ | $\perp$ | $f$ | $t$ |
| $f$ | $f$ | $\perp$ | $f$ | $f$ | $t$ |

   (b) Prove that *no* truth table with any finite number of truth values can correctly characterize the intuitionistic connectives. (*Hint.* One possible approach is to show that there are infinitely many inequivalent intuitionistic formulas over a single propositional letter $p$.)

2. (algorithm design and analysis)

   (a) Let $M$ be a maximum matching and $N$ a maximal matching in a graph $G = (V, E)$. Prove that $|N| \geq \frac{1}{2}|M|$.

   (b) An *edge cover* in $G$ is a subset of edges that covers every vertex; i.e., every vertex has at least one incident edge from this set. Give an efficient algorithm for finding a smallest edge cover.

   (c) A *vertex cover* in $G$ is a subset of vertices that covers every edge; i.e., at least one endpoint of every edge is in this set. The problem of finding the smallest vertex cover is *NP*-hard. Give an efficient algorithm for finding a vertex cover that is at most twice the cardinality of the optimal. (*Hint.* Use maximal matching.)

   (d) Let $n_e$ denote the size of the smallest edge cover in $G = (V, E)$, and $n_v$ the size of the smallest vertex cover. ¿From (a), (b) and (c) conclude that $n_e + 2n_v \geq |V|$.

3. (computational complexity) Recall that a language $L \subseteq \{0,1\}^*$ is in *RP* if there exists a polynomial $p(\cdot)$ and a polynomial time predicate $R(\cdot,\cdot)$ such that

   (i) $x \in L$ iff for at least half the strings $y \in \{0,1\}^{p(|x|)}$, $R(x,y)$;

   (ii) $x \notin L$ iff for all strings $y \in \{0,1\}^{p(|x|)}$, $\neg R(x,y)$.

   The strings $y \in \{0,1\}^{p(|x|)}$ for which $R(x,y)$ are called *witnesses* for $x$.

   (a) Prove that for all $n \geq 2$ there exists a set $S_n$ of strings of length $p(n)$ such that

      i. $|S_n| = n$

      ii. every $x \in L$ with $|x| = n$ has a witness in $S_n$.

   (b) Does (a) enable you to say anything about the $P \stackrel{?}{=} RP$ question or the $RP \stackrel{?}{=} NP$ question? Elaborate.

4. (wild card)

   (a) Give a polynomial time algorithm for finding a satisfying truth assignment to a satisfiable Boolean formula, given an oracle for the decision problem *SAT*.

   (b) Prove that if a language $L$ over a one-letter alphabet is *NP*-complete, then $P = NP$. (*Hint.* Modify your proof of part (a) and use the reduction from *SAT* to $L$.)