We say that a function is in the class $\mathcal{NC}$ if the function can be computed in polylogarithmic time by polynomially many processors.

At the start of the computation of such a function, each symbol of the input string could be read by a different processor, and at the end, each symbol of the output string could be written by a different processor.

Answers to True/False Questions, Part II

A *deterministic context free language* (DCFL) is a language accepted by a deterministic push-down automaton (DPDA).

1. True or False. T = true, F = false, and O = open, meaning that the answer is not known science at this time. In the questions below, $\mathcal{P}$ and $\mathcal{NP}$ denote $\mathcal{P}$-TIME and $\mathcal{NP}$-TIME, respectively.

(lxvi) **F** Every subset of a regular language is regular.

(lxvii) **F** Let L be the language over $\{a, b, c\}$ consisting of all strings which have more a's than b's and more b's than c's. There is some PDA that accepts L.

(lxviii) **T** Every subset of any enumerable set is enumerable.

(lxix) **T** If L is a context-free language which contains the empty string, then $L\backslash\{\lambda\}$ must be context-free.

(lxx) **T** If L is any language, there is a reduction of L to the halting problem. (Warning: this is a trick question. Give it some serious thought.)

(lxxi) **T** The computer language C++ has Turing power.

(lxxii) **T** Let Σ be the binary alphabet. Every $w \in \Sigma^*$ which starts with 1 is a binary numeral for a positive integer. Let $Sq : \Sigma^* \to \Sigma*$ be a function which maps the binary numeral for any integer n to the binary numeral for n^2. Then Sq is an $\mathcal{NC}$ function.

(lxxiii) **T** If L is any $\mathcal{P}$-TIME language, there is an $\mathcal{NC}$ reduction of the Boolean circuit problem to L.

(lxxiv) **T** If an abstract Pascal machine can perform a computation in polynomial time, there must be some Turing machine that can perform the same computation in polynomial time.

(lxxv) **T** The binary integer factorization problem is co-$\mathcal{NP}$.

(lxxvi) **F** Let L be any $\mathcal{RE}$ language which is not decidable, and let M_L be a machine which accepts L.
(a) If there are no strings of L of length n, let $T(n) = 0$.
(b) Otherwise, let $T(n)$ be the largest number of steps it takes M_L to accept any string in L of length n.
Then T is a recursive function.

(lxxvii) **O** There is a polynomial time reduction of the subset sum problem to the binary factorization problem.

(lxxviii) **F** The language of all palindromes over $\{a, b\}$ is an LR language.

(lxxix) **F** The Simplex algorithm for linear programming is polynomial time.

(lxxx) **F** Remember what a *fraction* is? It's a string consisting of a decimal numberal, followed by a slash, followed by another decimal numeral whose value is not zero. For example, the string "14/37" is a fraction. Each fraction has a value, which is a number. For example, "2/4" and "1/2" are different fractions, but has the same value. For any real number x, the set of fractions whose values are less than x is $\mathcal{RE}$.

(lxxxi) **F** The union of any two deterministic context-free languages must be a DCFL.

(lxxxii) **F** The intersection of any two deterministic context-free languages must be a DCFL.

(lxxxiii) **T** The complement of any DCFL must be a DCFL.

(lxxxiv) **T** Every DCFL is generated by an LR grammar.

(lxxxv) **T** The membership problem for a DCFL is in the class $\mathcal{P}$-TIME.

(lxxxvi) **T** If $h : \Sigma_1 \to \Sigma_2^*$ is a function, $L_1 \in \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, $h(L_1) = L_2$ and L_1 is regular, then L_2 must be regular.

(lxxxvii) **F** If $h : \Sigma_1 \to \Sigma_2^*$ is a function, $L_1 \in \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, $h(L_1) = L_2$ and L_2 is regular, then L_1 must be regular.

(lxxxviii) **T** If $h : \Sigma_1 \to \Sigma_2^*$ is a function, $L_1 \in \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, $L_1 = h^{-1}(L_2)$ and L_2 is regular, then L_1 must be regular.