Computational Complexity

Scribe: Sophia Wang and Sharath Byakod

Date: 23, May, 2019

1 Chomsky Hierarchy

Recursively Enumerable Languages: can be recognized by Turing Machines.

Context Sensitive Languages: can be recognized by Turing Machines with linear space on their tape. Ex: $\{a^n b^n c^n | n \epsilon Z^+\}$

Context Free Languages: can be recognized by NPDAs. Ex: $\{ww^R\}$ and $\{a^n b^n | n \epsilon Z^+\}$

Regular Expressions: can be recognized by DFAs. Ex: $\{a^n | n \epsilon Z^+\}$

Note: a Turing Machine can "recognize a language" if and only if:
TM(x) answers "Yes", if x is in the language, and
TM(x) either answers "No" or runs forever, for all x not in the language.

2 The Halting Problem

Given a Turing Machine M and input x, we are presented with the question: "Does M(x) halt?", *or, does the Turing Machine give an answer instead of running indefinitely?* The following things are true:

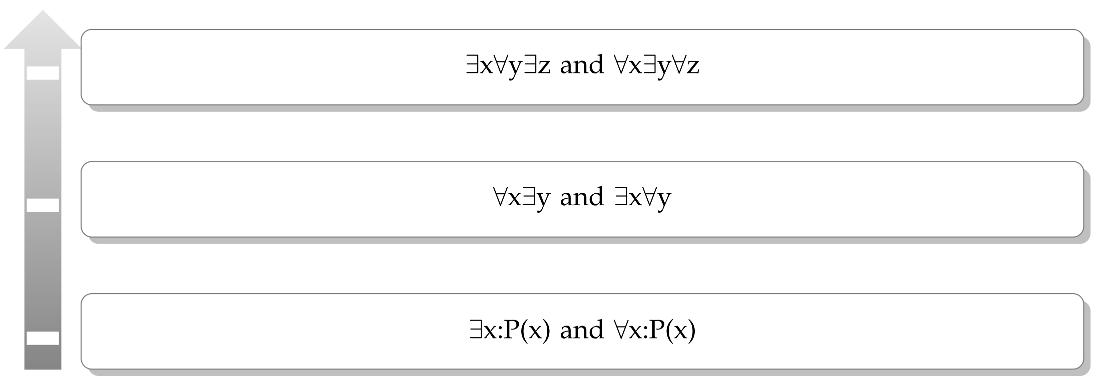
1. Turing Machines are enumerable. Any and all Turing Machines can be uniquely represented by a integer. This makes sense when we consider that a binary string is a series of operations.
2. All x's are enumerable. Turing Machines only run on integers.
3. Universal Turing Machines exist.

Define: A recursive language is any language such that a Turing Machine acting on a x within the language will halt if it results in an "Yes" or a "No".

Define: An alternating Turing Machine is visualized as a tree that alternates between levels of ∃ and ∀ with leaves that are conditional statements.

3 The Arithmetic Hierarchy

∃ = Existential Quantifier, or "there exists".
∃ x: x>5
∀ = Universal Quantifier, or "for all".
∀ x: x>5x
∀ x ∃ y: y < x— d



In the lowest level of the arithmetic hierarchy → ∃x:P(x), *NP* and ∀x:P(x), co-*NP*. It is known that *PRIME* ∈ co-*NP* since it is defined as: for all x, x is not an integer between 1–n, nor does it divide n. However, Pratt's Theorem shows that *PRIME* ∈ *NP*.

Theorem 3.1. *Pratt's Theorem: PRIME ∈ NP*

Proof. If p is prime, then $\mathbb{Z}_p^*$ is cyclic. This means that an element g generates the entire group. Then, we use a non-deterministic Turing Machine to guess g. To prove g is a generator, for all primes that divide $p-1 = q_1^{e_1} * q_2^{e_2} * \ldots, g^{(p-1)/q_i} \not\equiv 1 \pmod{p}$, we need only perform $\log_2 p$ tests on primes of size and order $\log p$. We need to prove this recursively to show that prime factors are truly prime. Thus, Pratt's Certificate of Primality requires the factorization of $n-1$ and the method is best applied to small numbers (numbers n known to have easily factorable $n-1$).