Contents

1 Decidability

Hello welcome to the section.

Corollary 1.1. *A language is decidable if* $\exists$ *a non deterministic Turing Machine that recognizes it.*

Theorem 1.2. *A language is Turing Recognizable if and only if some enumerator enumerates it.*

Theorem 1.3. *The class of Fontext Free Languages is a proper subset of the Turing Recognizable languages.*

Hilbert's 10th Problem: Given a polynomial with integer coeficients, does there exist an integer root to that polynomial.

$$D = \{p | \text{p is a polynomial over one variable}\}$$

$$F = \{p | \text{p is a polynomial over one or more variables}\}$$

Theorem 1.4. *The class of Turing Recognizable Languages is closed under* $\cup$.

Proof. Let A, B be Turing Recognizable Languages.
$\exists$ Turing Machines $M_A, M_B, L(M_A) = A, L(M_B) = B$.
We want Turing Machine M such that $L(M) = A \cup B$
On input w, M does:
1. run M_A and M_B in parallel on w
-if M_A or M_B then halt and accept
-if M_A and M_B then halt and reject

Claim. $L(M) = A \cup B$

Let $w \in L(M)$ $w \in A \cup B$ etc..... $\square$

Definition 1.5. An algorithm is a well defined sequence of steps to perform a computation. ◇

Definition 1.6. A Universal Turing Machine (u) is a Turing Machine that can simulate running any Turing Machine on an input string. ◇

Acceptance problem for DFAs

Definition 1.7. The acceptance problem for Deterministic Finite Automata is $A_{DFA} = \{< M, w > | M \text{ is a } DFA \text{ and } w \in L(M)\}$. ◇

Theorem 1.8. *A_{DFA} is decidable.*

Proof. On an input $< M, w >$, X does:
1. Simulate running M for $|w|$ transitions.
2. If M is in an accept state, halt and accept.
3. Halt and reject.

$< M, w > \in A_{DFA} \Rightarrow L(M) \Rightarrow X$ Halt and accept $\Rightarrow < M, w > \in L(X)$
$< M, w > \in L(X) \Rightarrow w \in L(M) \Rightarrow < M, w > \in A_{DFA}$
$M = (Q, \Sigma, \delta, q_0, F)$
$|Q| < \infty$
$|\Sigma| < \infty$
$|\delta| < \infty$
Therefore X decides A_{DFA}. □

Acceptance problem for NFAs

Definition 1.9. The acceptance problem for Non-deterministic Finite Automata is $A_{NFA} = \{< M, w > | M \text{ is a } NFA \text{ and } w \in L(M)\}$. ◇

Theorem 1.10. *A_{NFA} is decidable.*

Proof. On an input $< M, w >$, Y does:
1. Construct DFA M' such that $L(M') = L(M)$.
2. Call Turing Machine X with input $< M, w >$ and return what it returns.

Let $< M, w > \in A_{NFA} \Rightarrow w \in L(M) \Rightarrow L(M') \Rightarrow < M', w > \in A_{DFA} \Rightarrow Y$ accepts $< M, w > \Rightarrow < M, w > \in L(Y)$
$< M, w > \in L(Y) \Rightarrow < M', w > \in L(X) = A_{DFA} \Rightarrow w \in L(M') = L(M) \Rightarrow < M, w > \in A_{NFA}$
$M = (Q, \Sigma, \delta, q_0, F)$
$|Q| < \infty$
$\mathcal{P}(Q) < \infty$
$|\Sigma| < \infty$
$|\delta| < \infty$
Therefore Y decides A_{NFA}. □

Generation problem for Regular Expressions

Definition 1.11. The generation problem for regular expressions $A_{rex} = \{< R, w > |R$ is a regular expression and $w \in L(R)\}$. $\diamond$

Theorem 1.12. *A_{rex} is decidable.*

Proof. On an input $< M, w >$, Z does:
1. Construct NFA N such that $L(N) = L(R)$.
2. Call Turing Machine Y with input $< N, w >$ and return what it returns.

Want to prove that $L(Z) = A_{rex}$.
Therefore Y decides A_{NFA}. $\square$

Emptiness problem for DFAs

Definition 1.13. The emptiness problem for DFAs is $E_{DFA} = \{< M > |M$ is a DFA and $L(M) = \emptyset\}$. $\diamond$

Theorem 1.14. *E_{DFA} is decidable.*

2 The Next Section

Hello this is another section.