

CSCB36 Final Review

- natural number \mathbb{N} in this course include 0
- strings are finite i.e. $S=101110\dots$ is never allowed
- if Σ is $\{0,1\}$, then Σ^* is the set of all finite strings over Σ i.e. $\{101, 011, 01110, 1, \dots\}$
- Σ is finite as well
- The Hamming distance is only defined when $|x|=|y|$ and $d_H(x,y)$ is the number of different character in x and y
- A language is a set of strings over Σ and a subset of Σ^*
- $\bar{L} = \Sigma^* \setminus L$
- Language concatenation : $L_1 = \{1,11\}, L_2 = \{0,00\}$, then $L_1 \cdot L_2 = \{10,100,110,1100\}$
- The empty set \emptyset concatenate with any L is empty set i.e. $L \cdot \emptyset = \emptyset \cdot L = \emptyset$
- $|L_1 \cdot L_2|$ may not $= |L_1| + |L_2|$ due to duplication
- $L^n = \begin{cases} \{\epsilon\} & n=0 \\ L \cdot L^{n-1} & n>0 \end{cases}$
- $\emptyset^0 = \{\epsilon\}$ but $\emptyset^n = \emptyset$, $n>0$
- $L^* = L^0 \cup L^1 \cup \dots$ i.e. $\epsilon \in L^*$ and $\emptyset^* = \{\epsilon\}$
- formaly L , $L = \{w \in \{0,1\}^* \mid w = l^p, p \text{ is prime}\}$
- $M = (Q, \Sigma, \delta, S, F)$, $S \in Q, F \subseteq Q$
- $L(M)$ is the language of M e.g. $L(M) = \{w \in \Sigma^* \mid \dots\}$
- Regular language: can be recognized by some DFA
- Theorem: For any DFA M , there is an NFA that accepts the same language i.e. $L(M) = L(N)$ and vice-versa.
- pumping lemma
 - 1. we can write x as uvw
 - 2. $|v| > 0$
 - 3. $|uv| < n$
 - 4. $uv^k w \in L, \forall k \in \mathbb{N}$

• Steps to prove a language is not regular

1. Assume for a contradiction that L is regular
2. Choose n_0
3. find $x \in L$ and $|x| > n_0$
4. write $x = uvw$ and $|v| > n_0, |uv| < n_0$
5. show $uv^k w \notin L$

6. Contradiction

• given any DFA M , we can build a regular expression R s.t $L(R) = L(M)$

• Proving two L are the same we need to show $L_1 = L_2$ iff $L \subseteq L_2$ and $L_2 \subseteq L_1$, apply to regex as well

• if R_1, R_2 differ by only the subexpressions S_1 and S_2 , and $S_1 \equiv S_2$ then $R_1 \equiv R_2$

• CFG $G = (V, \Sigma, P, S)$

• A context-free language is generated by some CFG

• PDA $M = (Q, \Sigma, \Gamma, \delta, S, F)$

• $\forall L, L$ can be expressed as $L(G)$ for some CFG $G \Leftrightarrow L$ can be expressed as $L(M)$ for some non-deterministic PDA M

• if a language is an ambiguous CFL, any PDA that recognizes it must be non-deterministic

• $T(x) = \text{True}$; we set x to be true

• $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$

• $x \rightarrow y \rightarrow z : x \rightarrow (y \rightarrow z)$

• if every row P is 1 and Q is 1, then $P \Rightarrow Q$

• $P \Leftrightarrow Q$

• $P \Leftrightarrow Q \iff P \wedge Q \vee \neg P \wedge \neg Q$

• 2 normal form

① Conjunctive Normal Form (CNF): $(\vee \vee) \wedge (\vee \vee) \wedge (\vee \vee)$

② Disjunctive Normal Form (DNF): $(\wedge \wedge) \vee (\wedge \wedge) \vee (\wedge \wedge) \dots$

• complete set: $\{\neg, \vee, \wedge\}, \{\neg, \vee\}, \{\neg, \wedge\}$

• incomplete set: $\{\rightarrow\}, \{\vee, \wedge\}$

- To prove a set is complete, we use structural induction to show that we can use it to generate every propositional formula.

Practically, we just need to show our set can generate \neg and either \vee or \wedge .

To show a set of connectives is not complete, we need to find a property of every formula generated using only these formulas, and then show that this property is not true for every formula.

- $P \text{ NAND } Q$: only $1,1 \Rightarrow 0$ and $\Rightarrow 1$ otherwise $\neg x: x \text{ NAND } x \quad x \vee y: \neg x \text{ NAND } \neg y \Leftrightarrow (\neg x \text{ NAND } x) \text{ NAND } (\neg y \text{ NAND } y)$
- $P \text{ xor } Q$: two the same $\Rightarrow 0$, otherwise $\Rightarrow 1$ if all input is F, every eng step is F
- O : $O P = O$ $\neg x: x \rightarrow \neg x \quad x \vee y: (x \rightarrow \neg x) \rightarrow y$

first-order language LN

- ① infinite # of variable
- ② finite # of constant
- ③ finite # of predicates

Atomic formula e.g.

$$\begin{pmatrix} A(0,1,1) \\ E(y,x) \\ A(0,1,0) \end{pmatrix}$$

- Free and bound variables
 - Interpretation
- Domain: the value that the constant can take
 predicates def
 value of constant
 valuation: The value that we give to the free variables

- $\neg \exists x, P \Leftrightarrow \forall x, \neg P$
- $\neg \forall x, P \Leftrightarrow \exists x, \neg P$

Now, that's an easy example, so let's see a slightly harder one:

$$\begin{aligned}
 & P(x) \vee \exists x, ((\forall x Q(x) \wedge R(x)) \rightarrow \neg \exists x, S(x)) \\
 \text{can be rewritten as:} \\
 & \Leftrightarrow P(x) \vee \exists x, ((\forall x Q(x) \wedge R(x)) \rightarrow \neg \exists x, S(x)) \\
 & \Leftrightarrow P(x) \vee \exists x, ((\forall y Q(y) \wedge R(y)) \rightarrow \neg \exists z, S(z)) \\
 & \Leftrightarrow P(x) \vee \exists x, ((\forall y Q(y) \wedge R(y)) \rightarrow \forall z, \neg S(z)) \\
 & \Leftrightarrow \exists t, P(t) \vee (\forall y (Q(y) \wedge R(y)) \rightarrow \forall z, \neg S(z)) \\
 & \Leftrightarrow \exists t, \exists y (P(t) \vee (Q(y) \wedge R(y)) \rightarrow \forall z, \neg S(z)) \\
 & \Leftrightarrow \exists t \exists y \forall z, (P(t) \vee (Q(y) \wedge R(y)) \rightarrow \neg S(z))
 \end{aligned}$$

note: $(\forall x a) \rightarrow b \Leftrightarrow \neg (\forall x a) \vee b$
 $\Leftrightarrow (\exists x \neg a) \vee b$
 $\Leftrightarrow \neg \forall x (\neg a) \vee b$
 $\Leftrightarrow \neg \forall x (a) \vee b$
 $\Leftrightarrow \exists x a \rightarrow b$

$$\begin{aligned}
 & \forall x, \exists y, (P(x) \vee Q(y)) ? \\
 & \Leftrightarrow \exists y, \forall x, (P(x) \vee Q(y))
 \end{aligned}$$

$$\begin{aligned}
 & \cancel{\forall x, \exists y, \exists z, (Q(y) \vee z) \wedge (\neg x \vee y \vee z)} \\
 & \exists z, \forall x, \exists y, (Q(y) \vee z) \wedge (\neg x \vee y \vee z)
 \end{aligned}$$