CS 6041
Theory of Computation

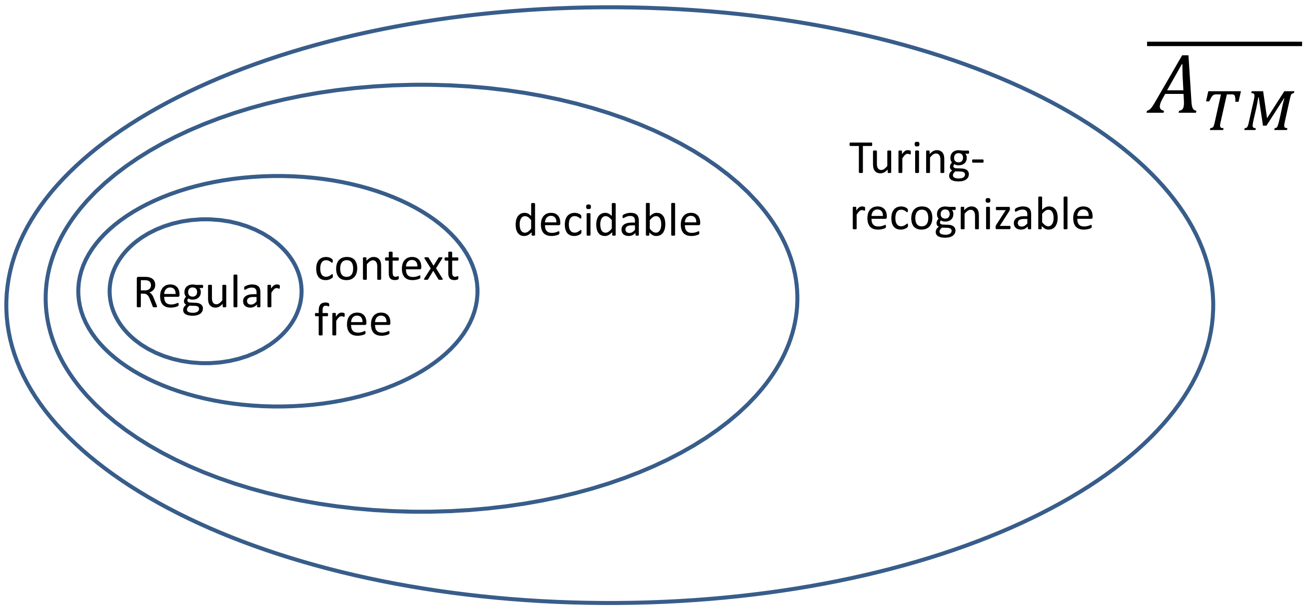
Conclusion

Kun Suo

Computer Science, Kennesaw State University

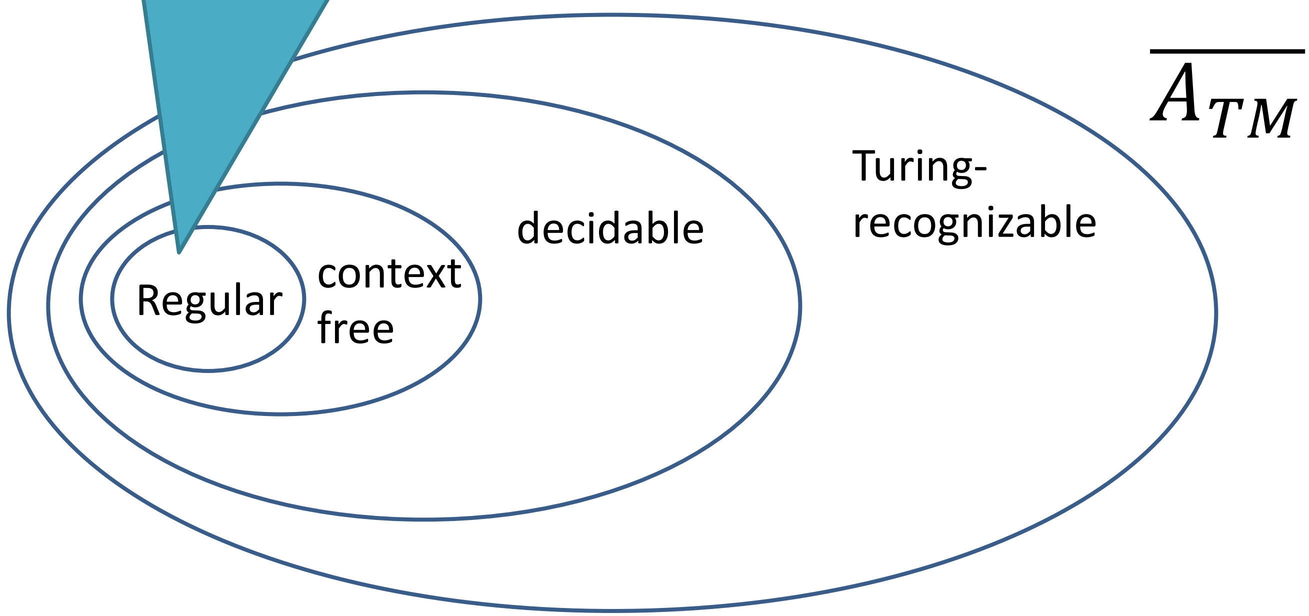
https://kevinsuo.github.io/

Language universe in a big picture



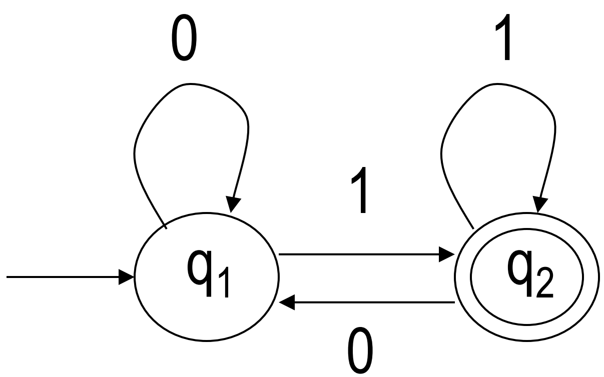
Language Map

DFA: deterministic finite automata
NFA: non-deterministic finite automata
RL: regular language



Deterministic finite automata (DFA)

- Finite automaton is a 5-tuple $M=(Q,\Sigma,\delta,q_0,F)$
  - Q: finite set called states
  - Σ: finite set called the alphabet
  - $\delta: Q\times\Sigma\rightarrow Q$, transition function
  - $q_0\in Q$: start state
  - $F\subseteq Q$: accept states



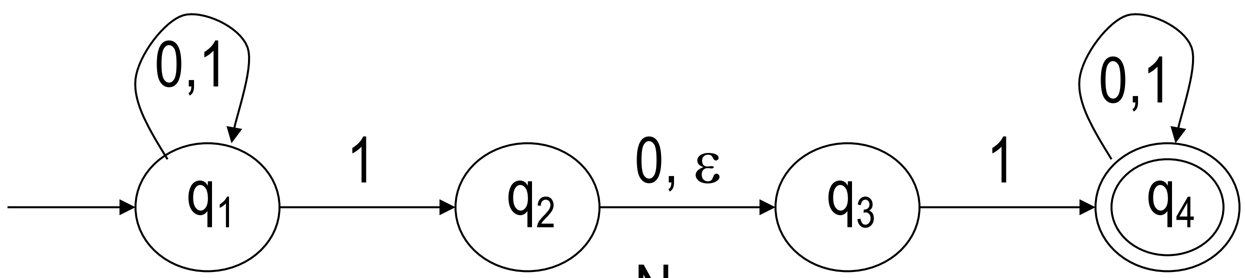
Deterministic finite automata (DFA)

- How to design deterministic finite automata
- Relationship of regular language and DFA
- Regular languages are closed under regular operations
  - Union, $A \cup B$
  - Concatenation, $A \cap B$
  - Star, A^*
  - Complement, $\bar{A}$
  - Boolean operation, $\mathrm{AND}{:}\wedge,\ OR{:}\vee,\ XOR{:}\oplus$

Nondeterministic finite automaton (NFA)

$N = (Q,\Sigma,\delta,q_0,F)$, where

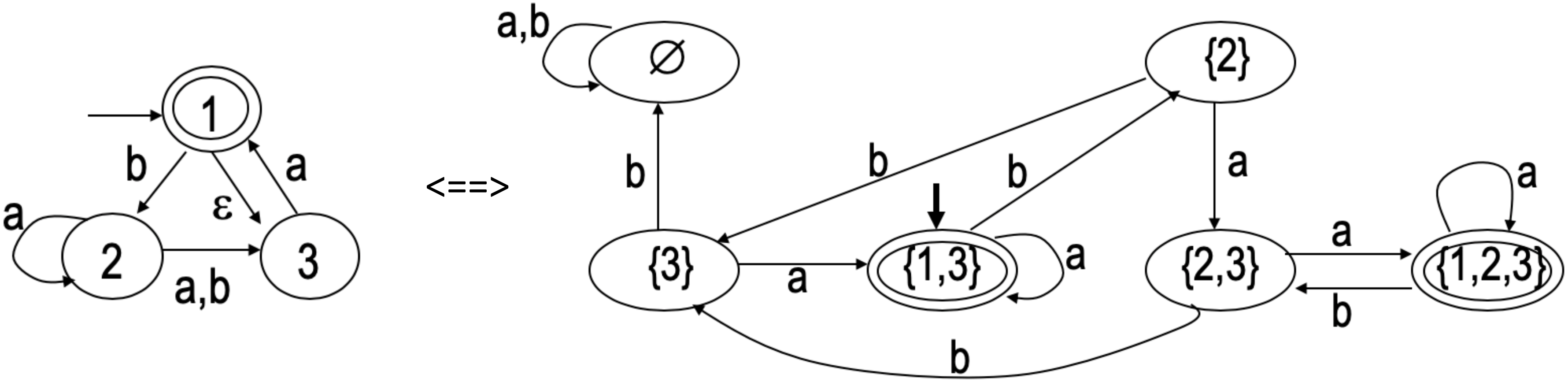
- Q: finite set of states
- Σ: finite alphabet as input; ($\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$)
- $\delta: Q \times \Sigma_\varepsilon \rightarrow P(Q)$, transition function
- $q_0 \in Q$: start state
- $F \subseteq Q$: accept state set



N_1

Nondeterministic finite automaton (NFA)

- Equivalence of NFAs and DFAs: Every nondeterministic finite automaton has an equivalent deterministic finite automaton



Regular language

- A language is called a RL if some DFA recognizes it
- A language is called a RL if some NFA recognizes it
- Use NFA to prove RL are closed under regular operations
  - Union, $A \cup B$
  - Concatenation, $A \cap B$
  - Star, A^*
  - Complement, $\bar{A}$
  - Boolean operation, $\mathrm{AND}{:}\wedge,\ OR{:}\vee,\ XOR{:}\oplus$

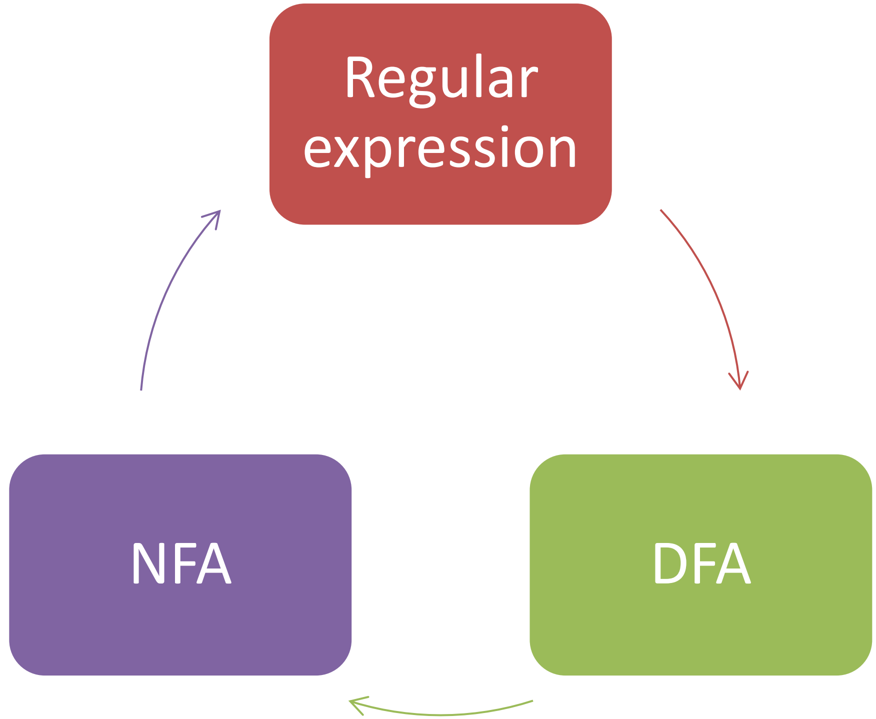
Regular expression

- Regular expressions are those describing languages by using regular operations

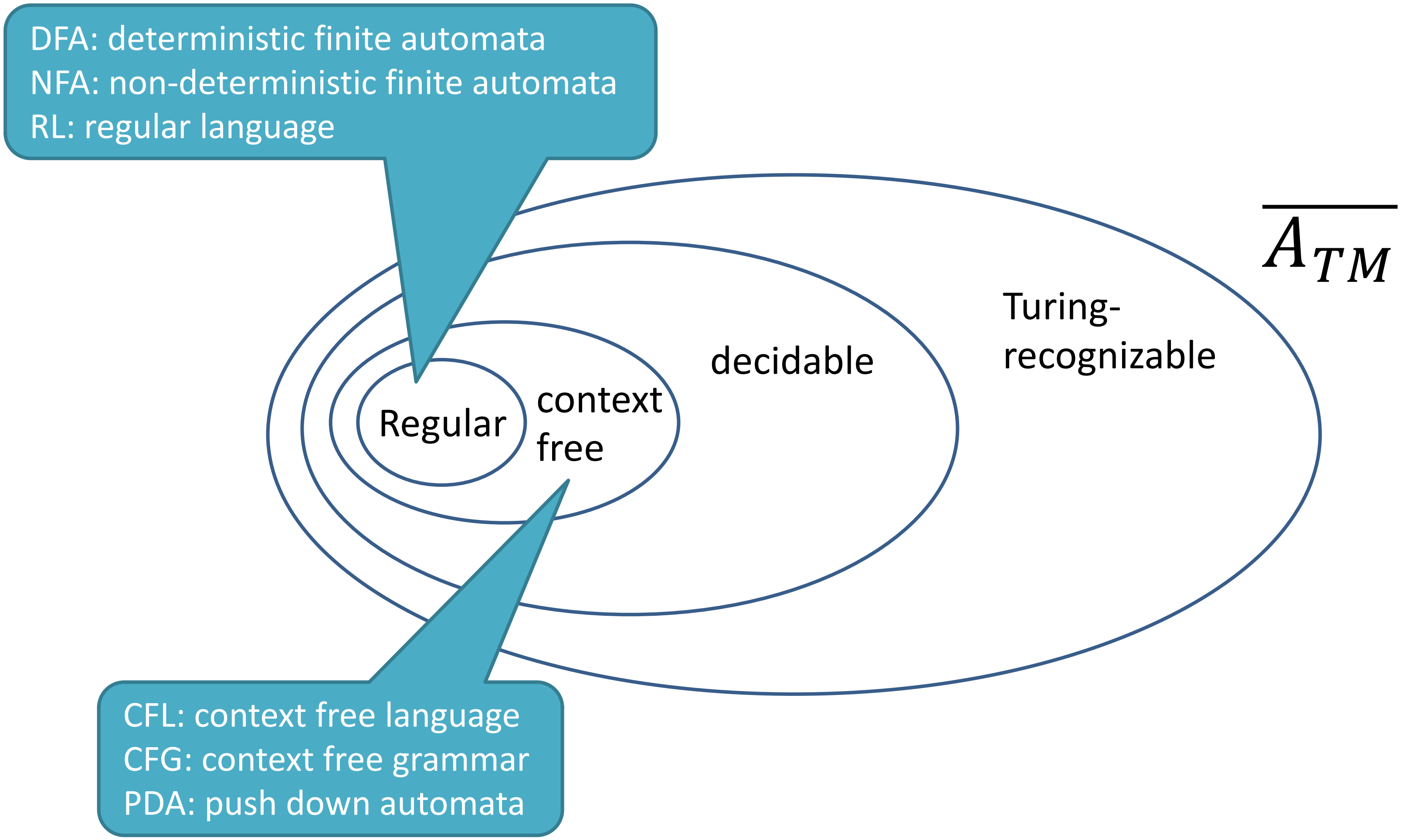
- R is regular expression if R is
  - a, where $a \in \Sigma$;
  - ε;
  - $\emptyset$;
  - $(R_1 \cup R_2)$, where R_1 and R_2 are all regular expressions;
  - $(R_1 R_2)$, where R_1 and R_2 are all regular expressions;
  - (R_1^*), where R_1 is regular expression.

Regular expression

- A language is regular if some deterministic finite automaton recognizes it
- A language is regular if and only if some nondeterministic finite automaton recognizes it
- A language is regular if and only if some regular expression describes it

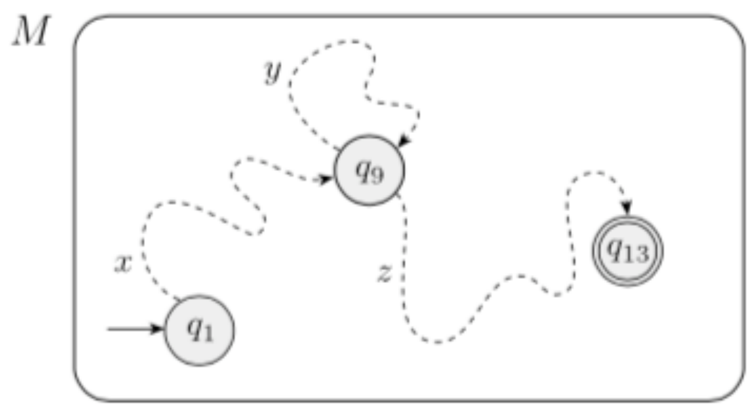


Language universe in a big picture



Non-regular languages

- ***(Pumping lemma)*** A is RL, then there is a number p (pumping length), where if $s \in A$ and $|s| \geq p$, then $s = xyz$, satisfying the following:
  1) $\forall i \geq 0$, $xy^iz \in A$;
  2) $|y| > 0$;
  3) $|xy| \leq p$.



Context-free languages (CFL)

- Context-free grammar (CFG) is a 4-tuple

  $G=(V,\Sigma,R,S)$,

  1) V: finite variable set

  2) Σ: finite terminal set

  3) R: finite rule set

     ($A \to w$, $w \in (V \cup \Sigma)^*$)

  4) $S \in V$: start variable

- How to design context-free grammar

Ambiguity in CFL

- If a grammar generates the ***same*** string in several ***different*** ways, we say that the string is derived *ambiguously* in that grammar.

- If a grammar generates some string ambiguously, we say that the grammar is ***ambiguous***.

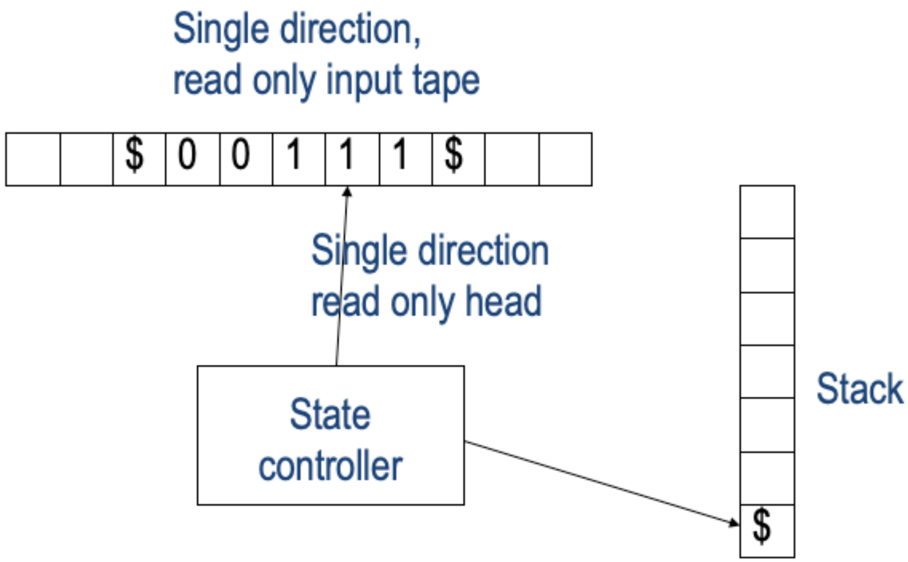
Chomsky normal form (CNF)

- CNF: only allow CFG in the following forms
  - $S \rightarrow \varepsilon$
  - $A \rightarrow BC$
  - $A \rightarrow a$

- How to transfer CFG into CNF

Pushdown Automata (PDA)

- PDA $M=(Q,\Sigma,\Gamma,\delta,q_0,F)$, where
  1) Q: set of states
  2) Σ: input alphabet, $\Sigma_\varepsilon=\Sigma\cup\{\varepsilon\}$
  3) **Γ: stack alphabet, $\Gamma_\varepsilon=\Gamma\cup\{\varepsilon\}$**
  4) $\delta: Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon\rightarrow P(Q\times\Gamma_\varepsilon)$,
     transition function
  5) $q_0\in Q$: start state
  6) $F\subseteq Q$: accept state set

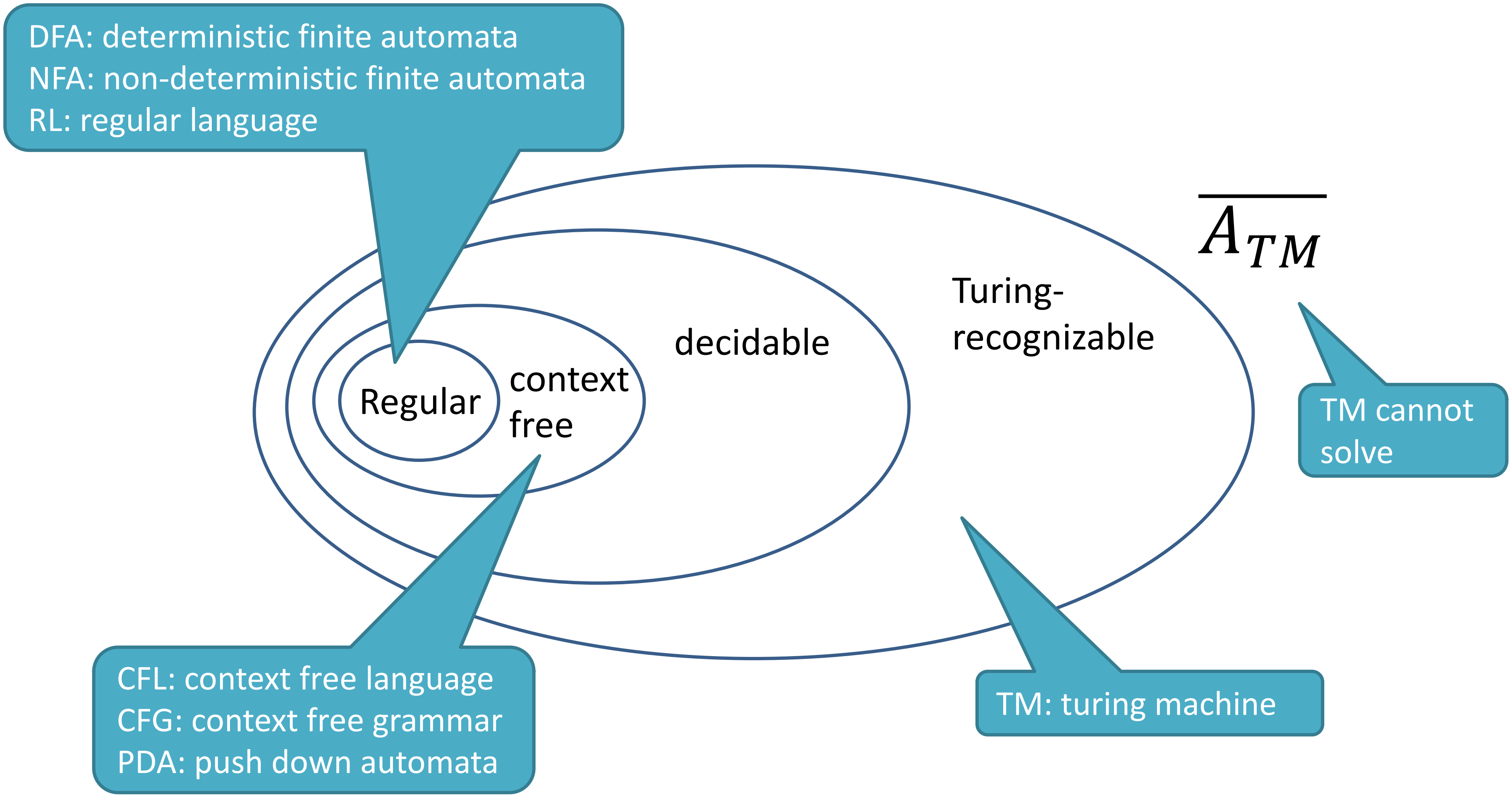


Equivalence of PDA and CFG

- A language is context free if and only if some pushdown automaton recognizes it

- A language is CFL $\Leftrightarrow$ some PDA recognizes it

Language universe in a big picture



Non-context-free language (Non-CFL)

- ***(Pumping lemma)*** Suppose A is CFL,

  then there exist a number p(the pumping length) where,

  if $s \in A$ and $|s| \geq p$, then $s = uvxyz$,

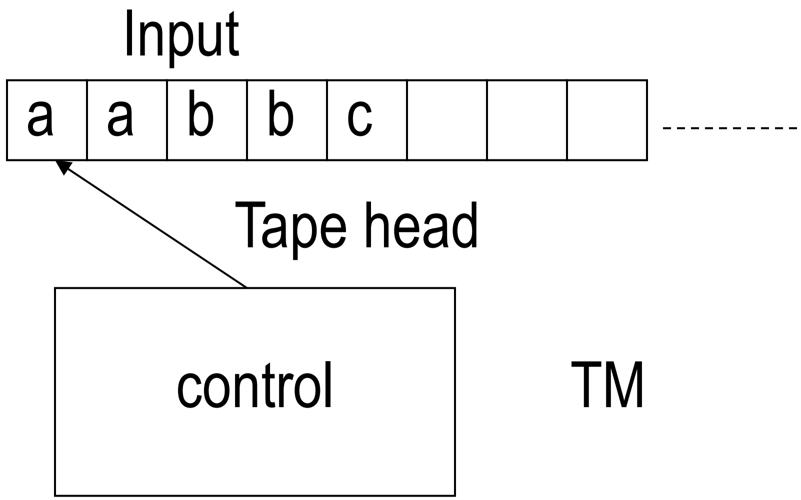
- Satisfying the following

  1) $\forall i \geq 0$, $uv^i xy^i z \in A$;

  2) $|vy| > 0$;

  3) $|vxy| \leq p$.

Turing machine



	Turing machine	Finite automata	Pushdown automata
Header	Read and write	Only read	Only read
Header move	Left and right	Only right	Only right
Input	infinite	finite	finite
Output	Accept and reject, also non-halt	Accept and reject	Accept and reject, also non-halt (loop in stack)

Turing machine

- Turing-recognizable vs. Turing-decidable
  - Accept/reject/non-halt
  - Accept/reject

- Variants of TMs
  - Multitape Turing machine
  - Nondeterministic Turing machine

Decidable problems concerning regular languages

- Acceptance problem for DFAs
  - whether a DFA accepts a string
- Acceptance problem for NFAs
  - whether a NFA accepts a string
- Regular expression decidability
  - Whether a regular expression generates a string
- Emptiness testing for DFAs
  - Whether a DFA is empty
- Equivalence of DFAs
  - Whether two DFAs recognize the same language

Decidable problems concerning context-free languages

- CFG generation decidability
  - Whether a CFG generates a particular string
- Emptiness testing for CFGs
  - Whether a CFG is empty
- Equivalence of CFGs
  - Whether two CFGs recognize the same language
- CFL decidability
  - Whether a CFL is decidable

Undecidable and unrecognizable

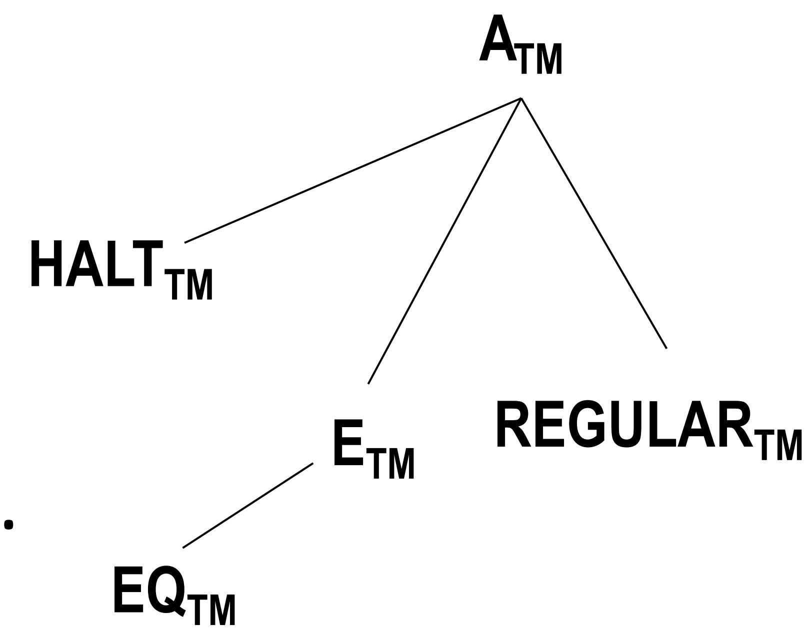
- A_{TM} is undecidable
  - Diagonalization method

- $\overline{A_{TM}}$ is not Turing-recognizable

	DFA	CFG	TM
Acceptance	√	√	×
Emptiness	√	√	×
Equivalence	√	×	×

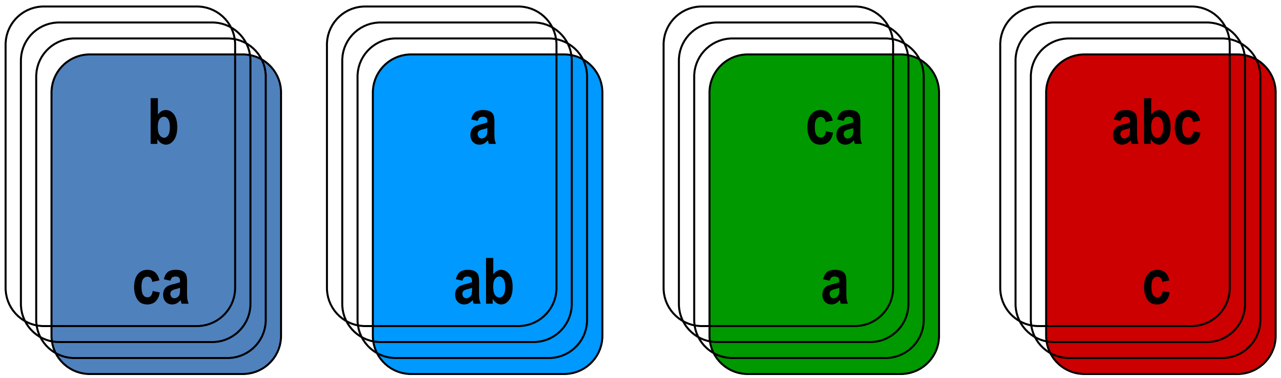
Halting problem and reducibility

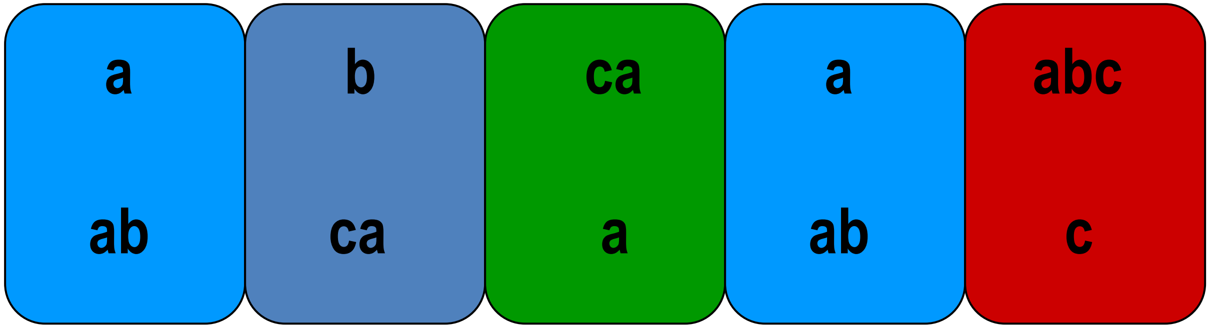
- TM halting problem:
  - whether a Turing machine halts (by accepting or rejecting) on a given input



- $HALT_{TM}$ is undecidable
- E_{TM} is undecidable
- $REGULAR_{TM}$ is undecidable.
- EQ_{TM} is undecidable.

Post Correspondence Problem (PCP)

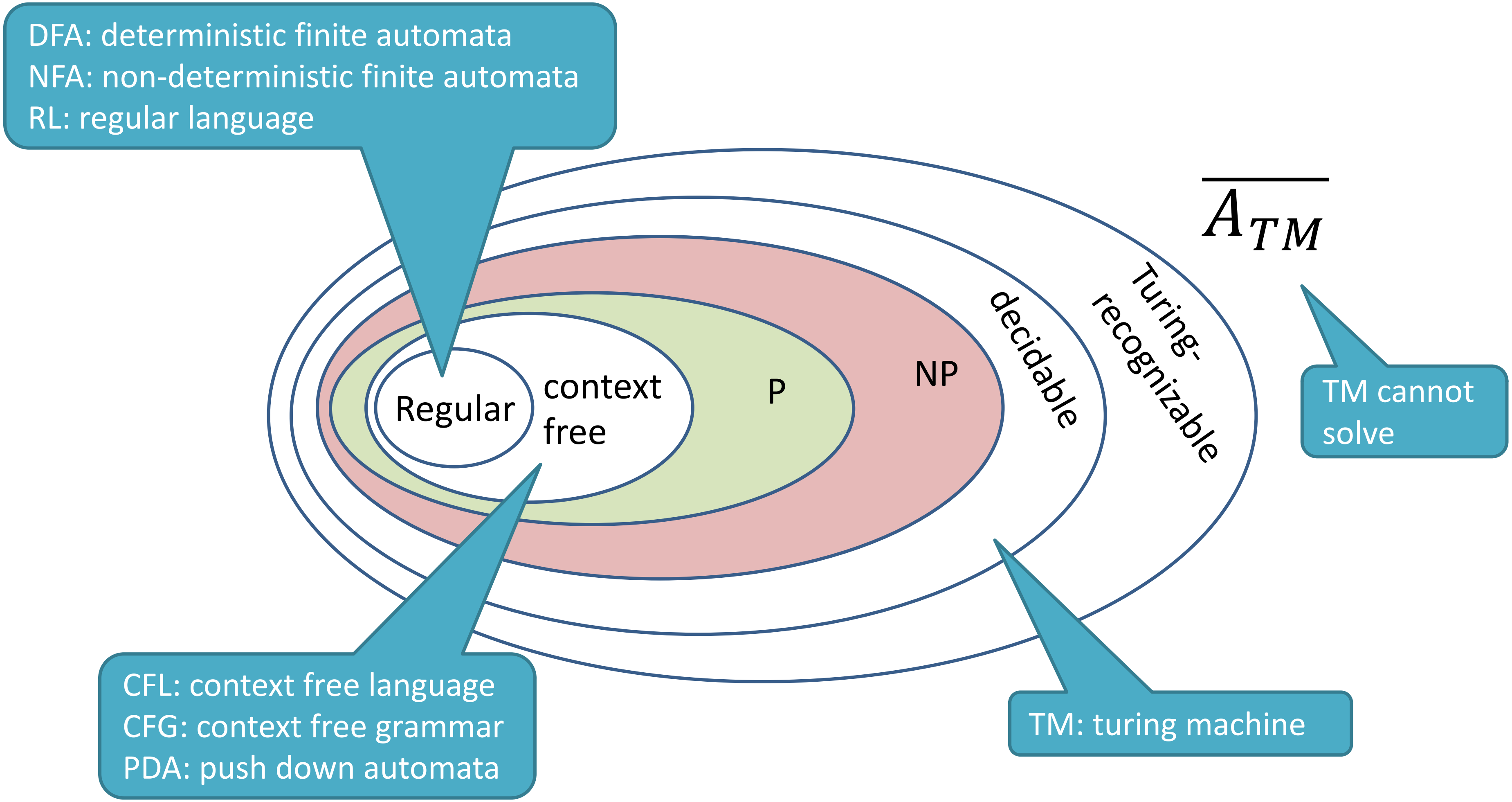




Closure on operations

	Complement $\overline{A}$	Intersection $\cap$	Union $\cup$	Star A^*
Regular/DFA/ NFA	√	√	√	√
CFL/ PDA	×	×	√	√
Turing-decidable TM	√	√	√	√
Turing-recognizable TM	×	√	√	√

Language universe in a big picture



Time complexity

- O() vs. o()
- Polynomial bounds vs. Exponential bounds
- Single-tape TM vs. multitape TM
  - $O(n)$ vs. $O(n^2)$
- Deterministic TM vs. nondeterministic TM
  - $O(n)$ vs. $O(a^n)$
- Class P vs. Class NP
- NP-completeness

Conclusion

Thanks