Undecidability

Beulah A.

AP/CSE

Revisit - TM

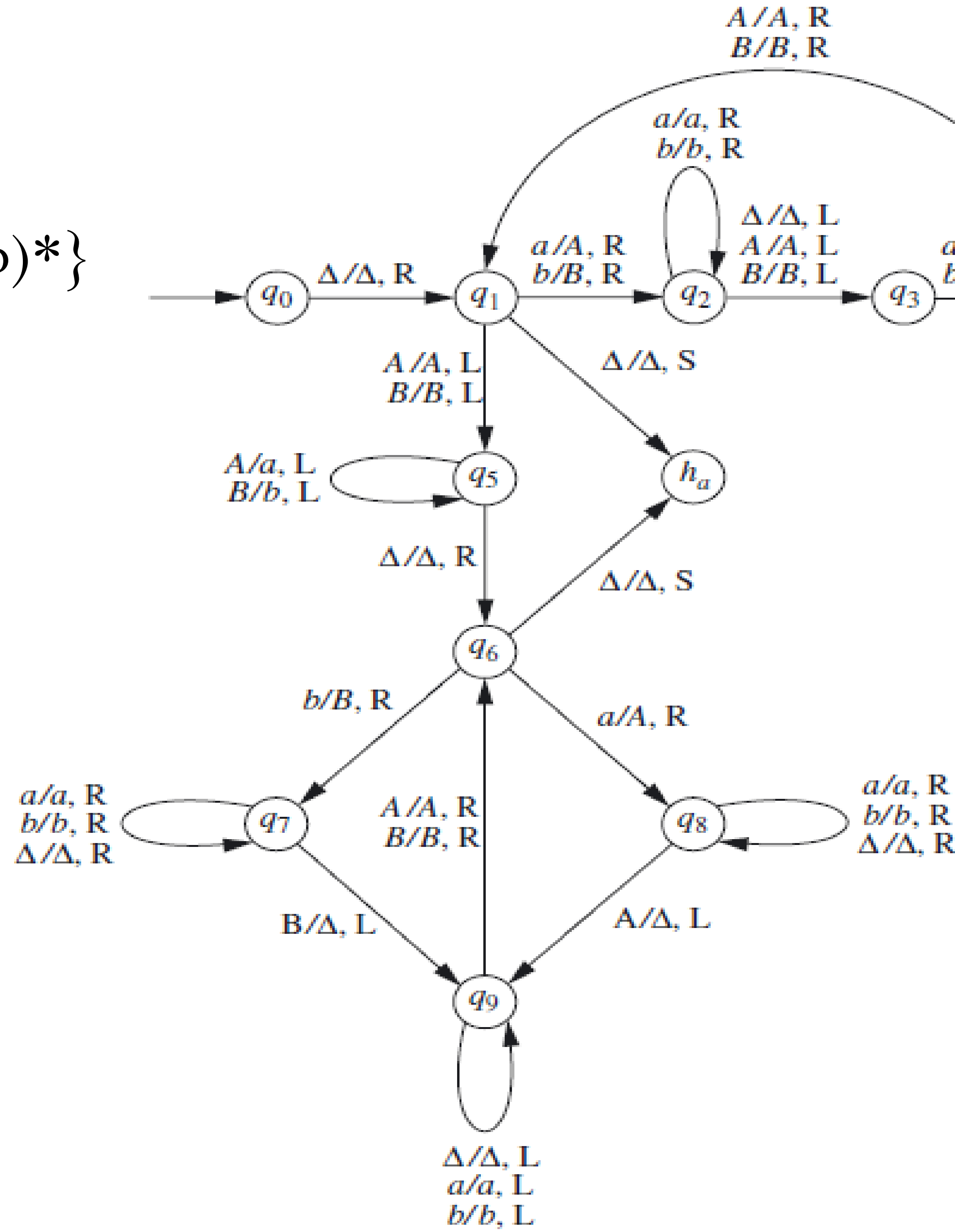
- A Turing machine (TM) is a 7-tuple

  $$M = (Q \cup \{h_a, h_r\}, \Sigma, \Gamma, \delta, q_0, B, F) \text{ where}$$

  - Q – A finite set of states of the finite control. Q + h_a and h_r
  - Σ – A finite set of input symbols
  - Γ – A set of tape symbols, with Σ being a subset
  - q_0 - The start state, in Q
  - B - The blank symbol in Γ, *not* in Σ (should not be an input symbol)
  - F - The set of final or accepting states

Example

L = {xx | x∈ (a,b)*}



Parsing example with h_a, h_r

$$
\begin{array}{llllllllll}
q_0\Delta aba & \vdash & \Delta q_1 aba & \vdash & \Delta A q_2 ba & \vdash^* & \Delta Aba q_2 \Delta & \vdash & \Delta Ab q_3 a \\
 & \vdash & \Delta A q_4 bA & \vdash & \Delta q_4 AbA & \vdash & \Delta A q_1 bA & \vdash & \Delta AB q_2 A \\
 & \vdash & \Delta A q_3 BA & \vdash & \Delta A h_r BA & \text{(reject)} & & & \\
 & & & & & & & & \\
q_0\Delta ab & \vdash & \Delta q_1 ab & \vdash & \Delta A q_2 b & \vdash & \Delta Ab q_2 \Delta & \vdash & \Delta A q_3 b \Delta \\
 & \vdash & \Delta q_4 AB & \vdash & \Delta A q_1 B & \vdash & \Delta q_5 AB & \vdash & q_5 \Delta aB \\
 & \vdash & \Delta q_6 aB & \vdash & \Delta A q_8 B & \vdash & \Delta A h_r B & \text{(reject)} & \\
 & & & & & & & & \\
q_0\Delta aa & \vdash & \Delta q_1 aa & \vdash & \Delta A q_2 a & \vdash & \Delta Aa q_2 \Delta & \vdash & \Delta A q_3 a \Delta \\
 & \vdash & \Delta q_4 AA & \vdash & \Delta A q_1 A & \vdash & \Delta q_5 AA & \vdash & q_5 \Delta aA \\
 & \vdash & \Delta q_6 aA & \vdash & \Delta A q_8 A & \vdash & \Delta q_9 A & \vdash & \Delta A q_6 \Delta \\
 & \vdash & \Delta A h_a \Delta & \text{(accept)} & & & & &
\end{array}
$$

Recursive (R) and Recursively Enumerable(RE) Languages

Decidability vs. Undecidability

- There are two types of TMs (based on halting):

(Recursive)

TMs that *always* halt, no matter accepting or non-accepting ≡ **DECIDABLE** PROBLEMS

(Recursively enumerable)

TMs that *are guaranteed to halt* only on acceptance. If non-accepting, it may or may not halt (i.e., could loop forever).

- **Undecidability:**
  - Undecidable problems are those that are not recursive

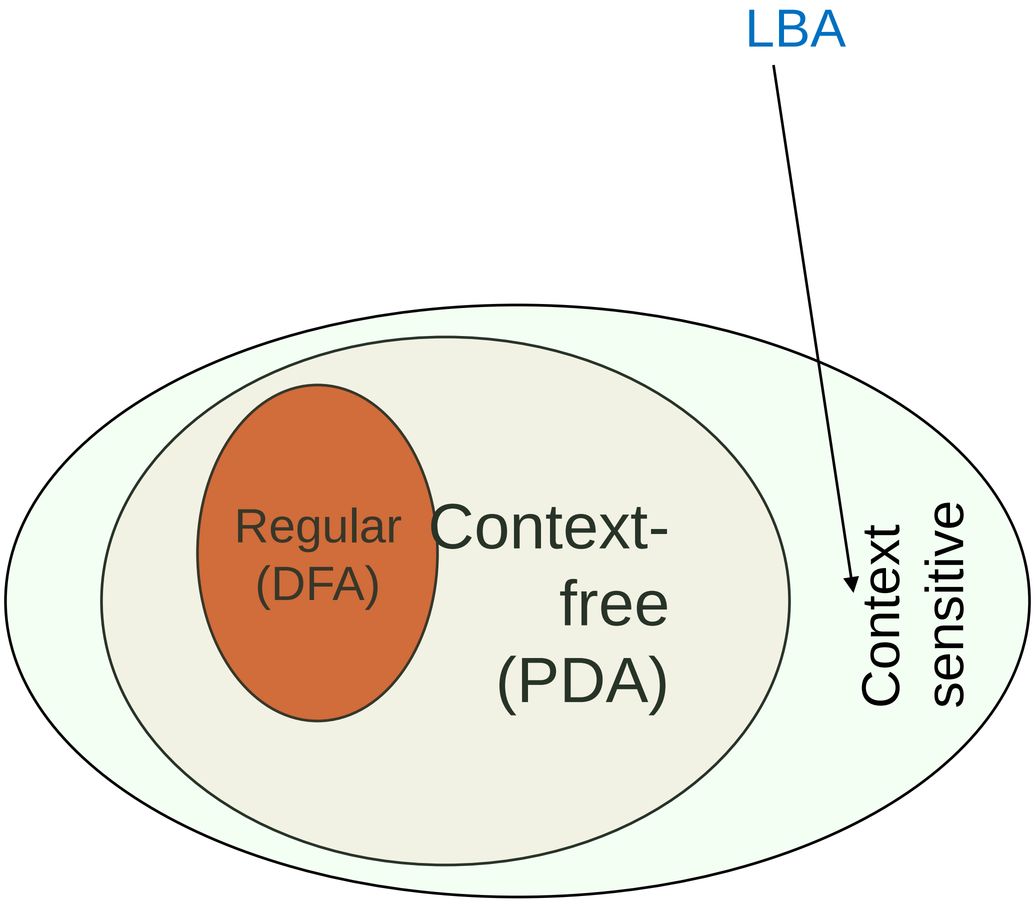
Recursive Language

- A Language L is Recursive if and only if there is a TM that decides L.
  - Let $M=(Q, \Sigma, \Gamma, \delta, q_0, B, H)$ such that
    - $H= \{h_a,h_r\}$
    - $L \subseteq \Sigma^*$ Is a language
    - Assume that the initial configuration of the TM is (q_0,w)
    - M decides L if, for all strings $w \in \Sigma^*$
      - Either $w \in L$, in which case M accepts w
      - Or $w \notin L$, then M rejects w

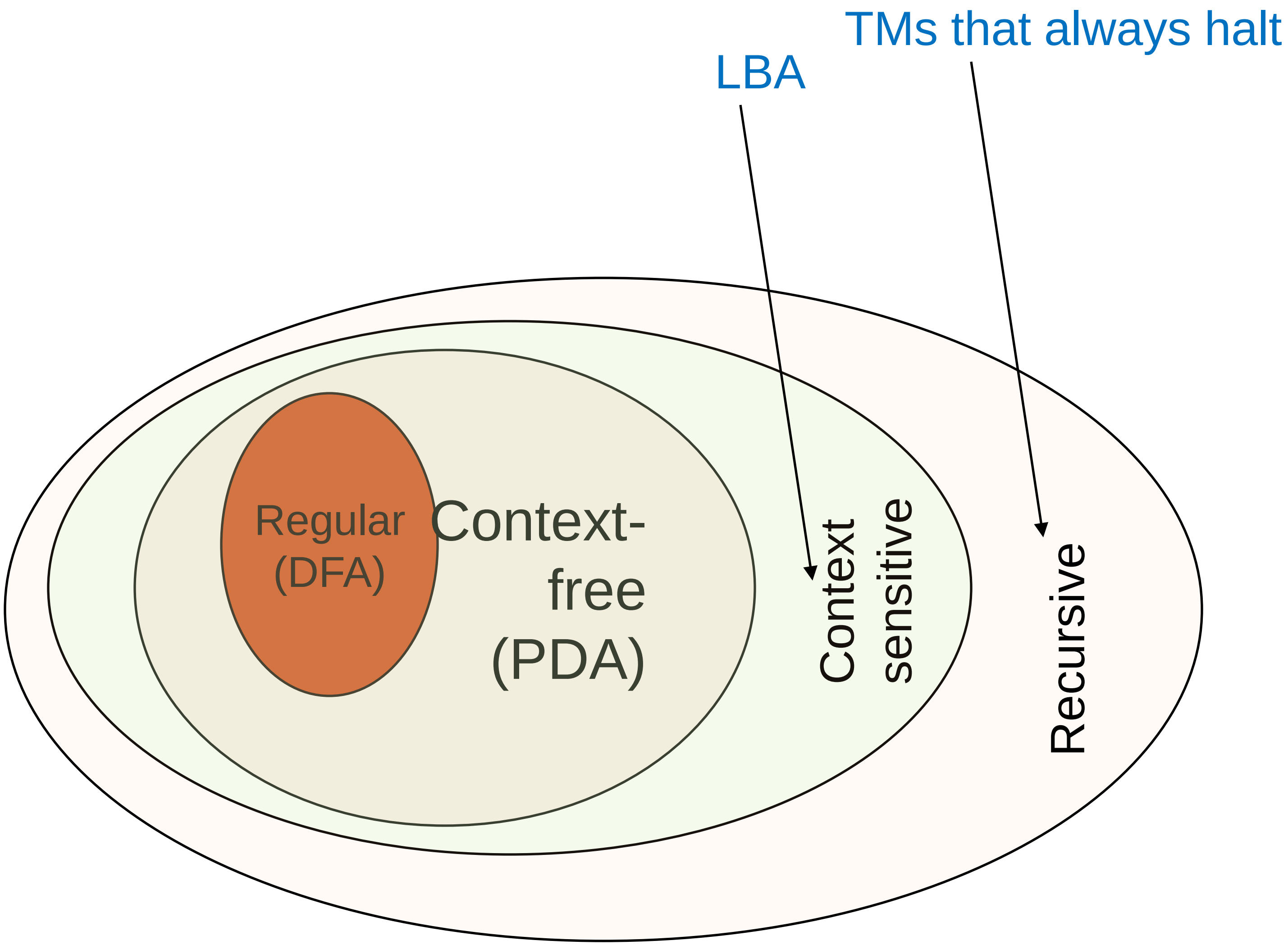
Recursive Enumerable Language

- A Language L is Recursive Enumerable if and only if there is a TM that semidecides L.
  - Let $M=(Q, \Sigma, \Gamma, \delta, q_0, B, H)$ such that
    - $H= \{h_a,h_r\}$
    - $L \subseteq \Sigma^*$ Is a language
    - Assume that the initial configuration of the TM is (q_0,w)
    - M semidecides L if, for all strings $w \in \Sigma^*$
      - Either $w \in L$, in which case M accepts w
      - Or $w \notin L$, then M does not halt

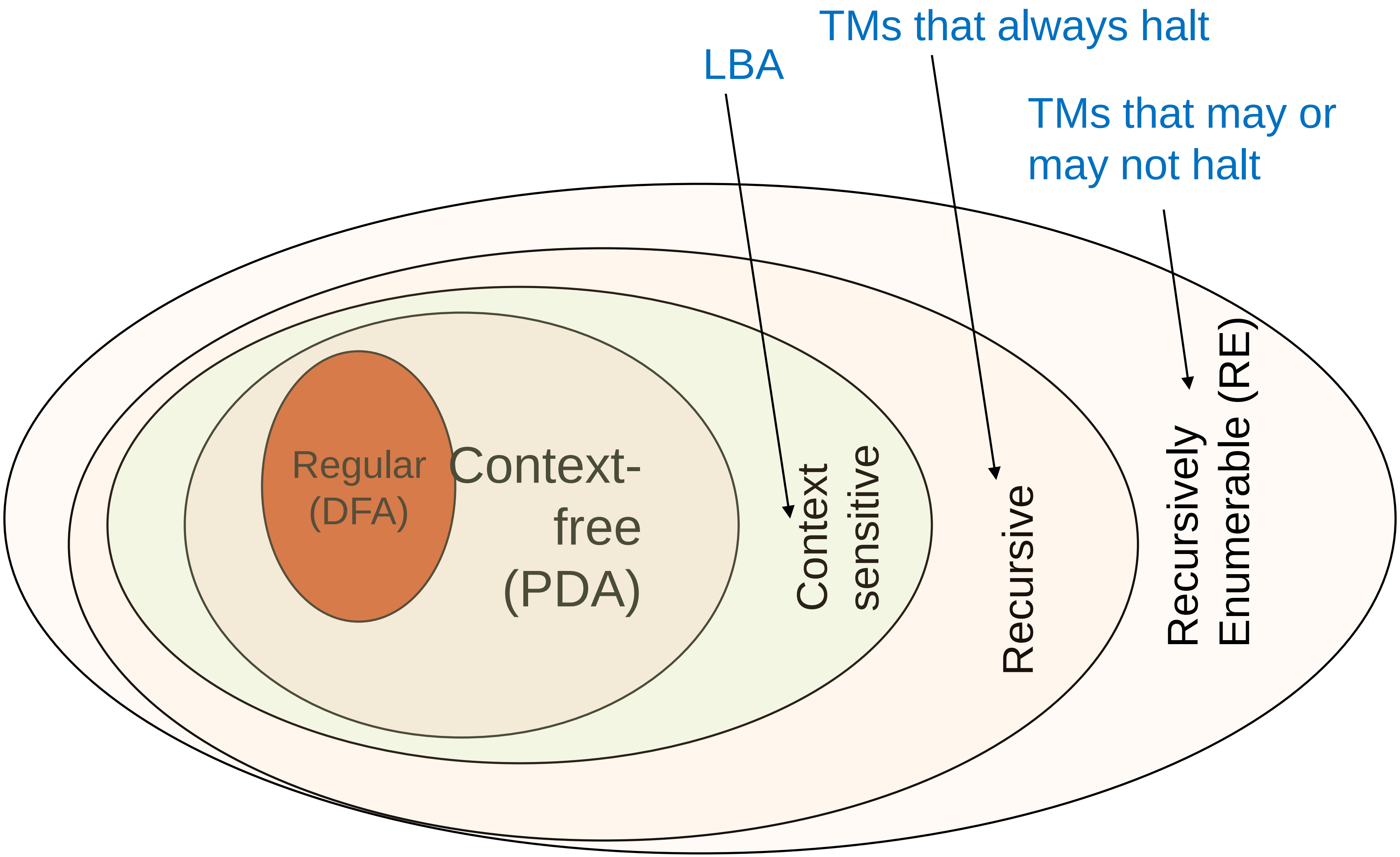
Recursive, RE, Undecidable languages



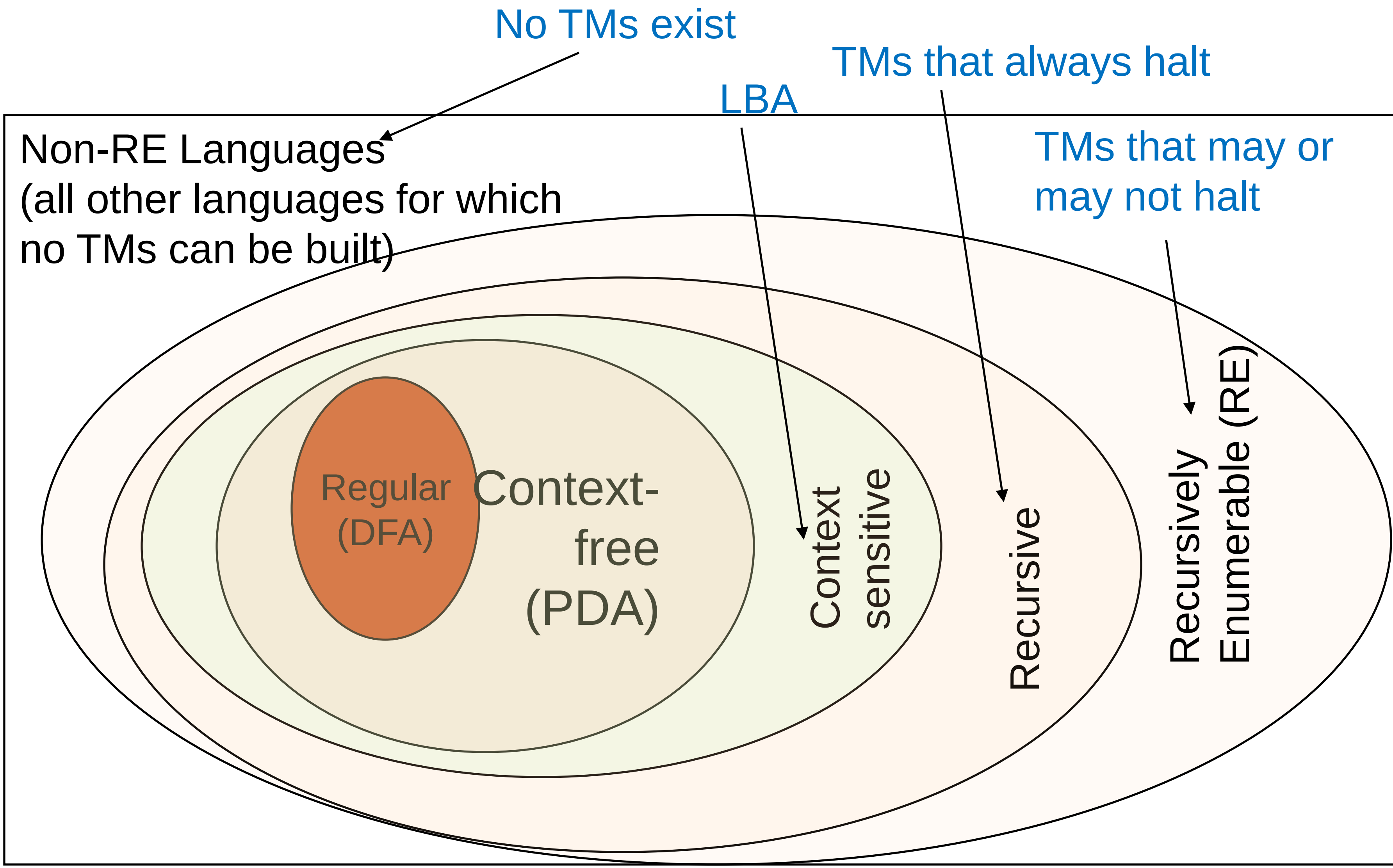
Recursive, RE, Undecidable languages



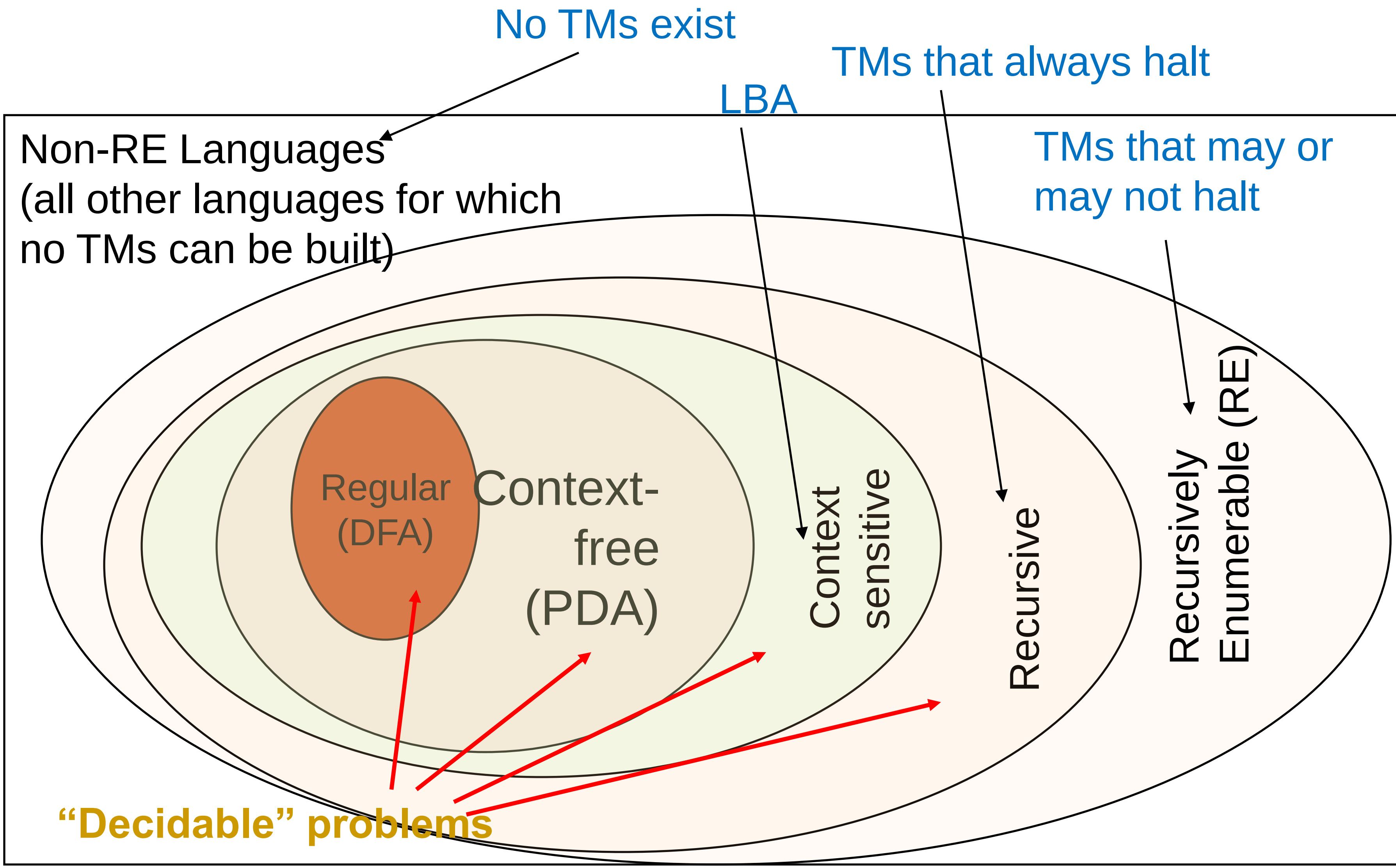
Recursive, RE, Undecidable languages



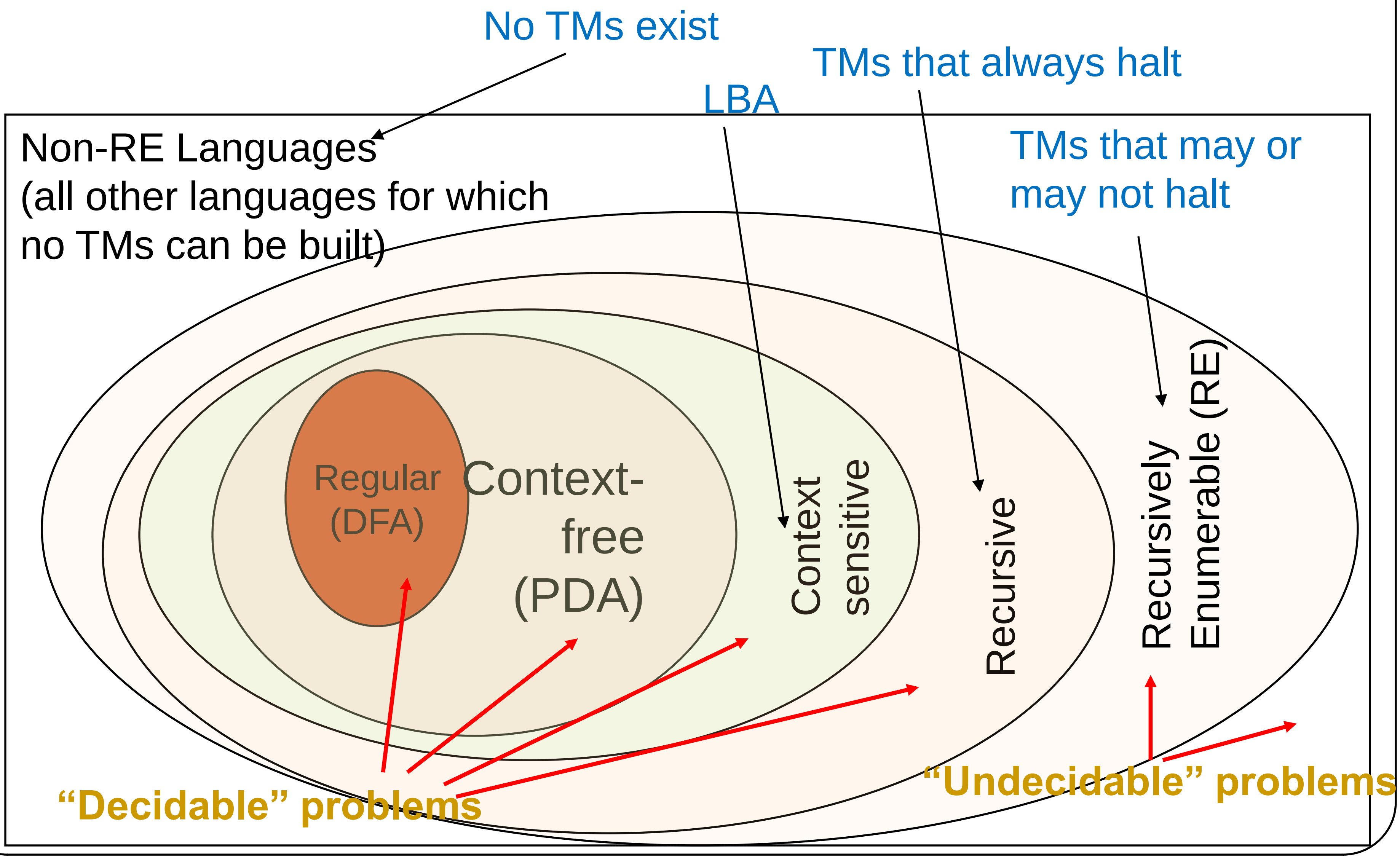
Recursive, RE, Undecidable languages



Recursive, RE, Undecidable languages

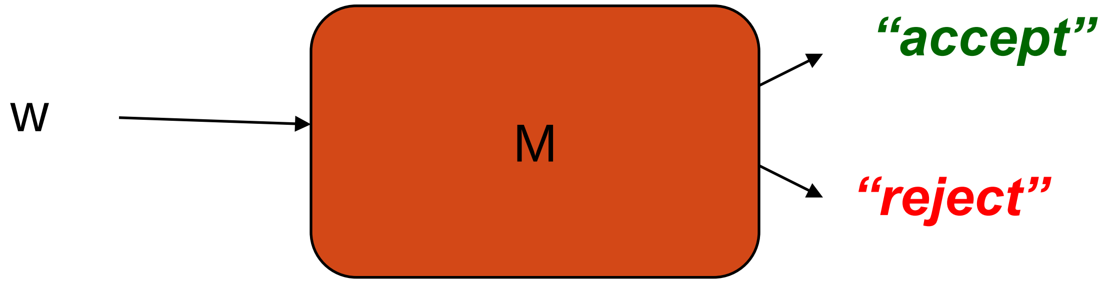


Recursive, RE, Undecidable languages

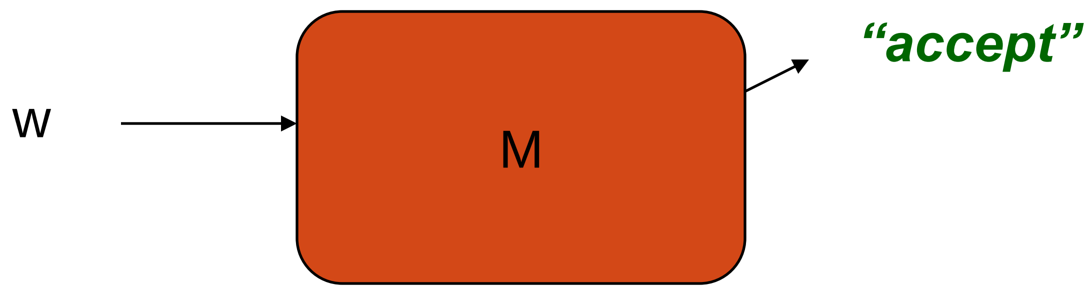


Recursive Languages & Recursively Enumerable (RE) languages

- Any TM for a <u>Recursive</u> language is going to look like this:



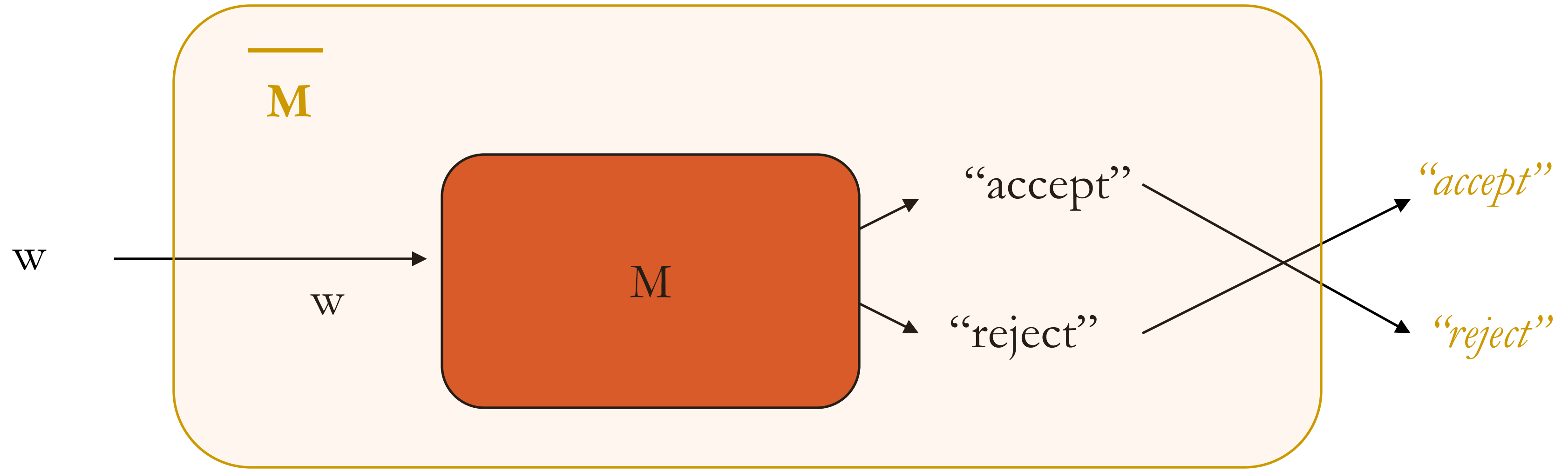
- Any TM for a <u>Recursively Enumerable</u> (RE) language is going to look like this:



Closure Properties of Recursive (R) and Recursively Enumerable(RE) Languages

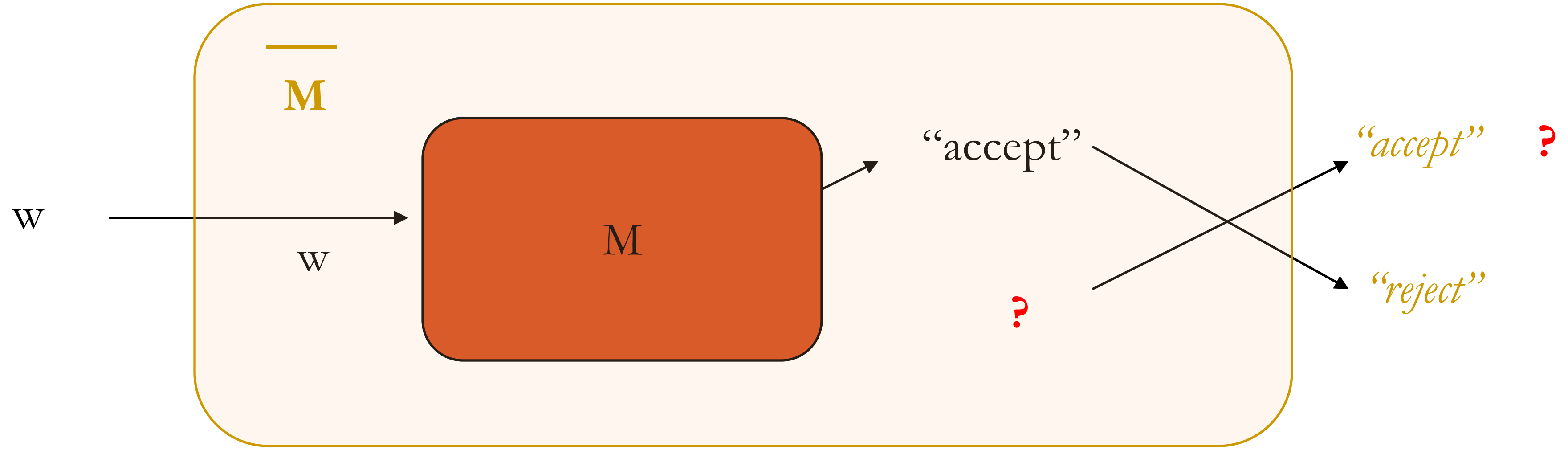
Recursive Languages are closed under <u>complementation</u>

- If L is Recursive, $\overline{L}$ is also Recursive



Are Recursively Enumerable Languages closed under <u>complementation</u>? (NO)

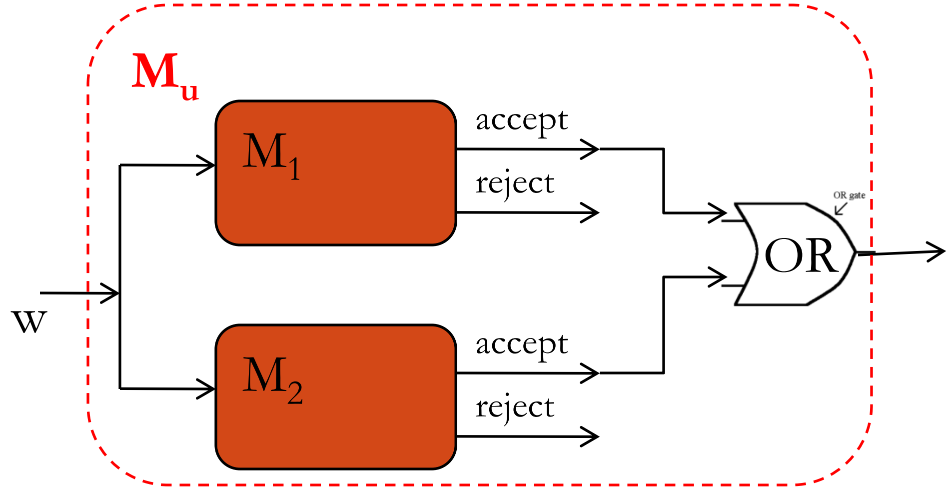
- If L is RE, $\overline{L}$ need not be RE



Recursive Langs are closed under Union

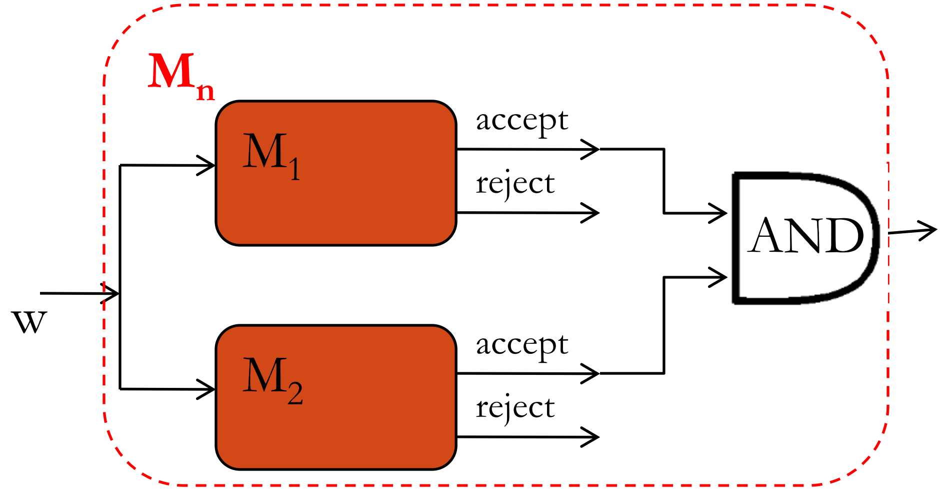
- Let M_u = TM for $L_1 \cup L_2$
- M_u construction:

1. Make 2-tapes and copy input w on both tapes
2. Simulate M_1 on tape 1
3. Simulate M_2 on tape 2
4. If either M_1 or M_2 accepts, then M_u accepts
5. Otherwise, M_u rejects.



Recursive Langs are closed under Intersection

- Let M_n = TM for $L_1 \cap L_2$
- M_n construction:
  1. Make 2-tapes and copy input w on both tapes
  2. Simulate M_1 on tape 1
  3. Simulate M_2 on tape 2
  4. If either M_1 AND M_2 accepts, then M_n accepts
  5. Otherwise, M_n rejects.



Other Closure Property Results

- Recursive languages are also closed under:
  - Concatenation
  - Kleene closure (star operator)
  - Homomorphism, and inverse homomorphism
- RE languages are closed under:
  - Union, intersection, concatenation, Kleene closure

- RE languages are *not* closed under:
  - complementation