Review
CSCI 338

Test 2 Logistics

1. During class on Wednesday 3/27.
2. You can bring your book and any notes you would like, but no electronic devices.
3. You may assume anything proven in class or on homeworks.
4. Four questions:
   1) Show a language is decidable (5 points).
   2) Show a language is not decidable (10 points).
   3) Show a language is not decidable (5 points).
   4) Show a language is not decidable (1-2 points).

Claim: $FINITE_{TM} = \{\langle M\rangle : M$ is a TM and accepts a finite number of strings$\}$ is undecidable.

Proof: Suppose $FINITE_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
    1. Run N on ω and <u>accept</u> if N does.
2. Run H on $\langle M_2\rangle$.
3. If H accepts, <u>reject</u>. If H rejects, <u>accept</u>.

$L(M_2)$ is infinite
$\Updownarrow$
N accepts ω

If N accepts ω, $L(M_2) = \Sigma^*$ (infinite) and S accepts. If N does not accept ω, $L(M_2) = \emptyset$ (finite) and S rejects. Therefore, S is a decider for A_{TM}, which is a contradiction, so $FINITE_{TM}$ is undecidable.

Claim: $EVEN_{TM} = \{\langle M \rangle : M$ is a TM and $L(M)$ contains all even length strings$\}$ is undecidable.

Proof: Suppose $EVEN_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
   1. If x has odd length, <u>reject</u>.
   2. If x has even length, run N on ω.
   3. If N accepts ω, <u>accept</u>. If N rejects ω, <u>reject</u>.
2. Run H on $\langle M_2 \rangle$.

$L(M_2)$ contains all even length strings

$\Updownarrow$

N accepts ω

Claim: $EVEN_{TM} = \{\langle M\rangle : M$ is a TM and $L(M)$ contains all even length strings$\}$ is undecidable.

Proof: Suppose $EVEN_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
   1. If x has odd length, ~~reject~~. **accept**
   2. If x has even length, run N on ω.
   3. If N accepts ω, accept. If N rejects ω, reject.
2. Run H on $\langle M_2\rangle$.

$L(M_2)$ contains all even length strings

$\Updownarrow$

N accepts ω

Claim: $EVEN_{TM} = \{\langle M \rangle : M$ is a TM and $L(M)$ contains all even length strings$\}$ is undecidable.

Proof: Suppose $EVEN_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. If x has odd length, <u>reject</u>.
    2. If x has even length, run N on ω.
    3. If N accepts ω, <u>accept</u>. If N rejects ω, ~~reject~~. **accept**
2. Run H on $\langle M_2 \rangle$.

$\boxed{L(M_2) \text{ contains all even length strings} \Updownarrow N \text{ accepts } \omega}$

Claim: $EVEN_{TM} = \{\langle M\rangle : M$ is a TM and $L(M)$ contains all even length strings$\}$ is undecidable.

Proof: Suppose $EVEN_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
   1. ~~If x has odd length, reject.~~
   2. If x has even length, run N on ω.
   3. If N accepts ω, accept. If N rejects ω, reject.
2. Run H on $\langle M_2\rangle$.

$L(M_2)$ contains all even length strings

$\Updownarrow$

N accepts ω

Claim: $EVEN_{TM} = \{\langle M \rangle : M$ is a TM and $L(M)$ contains all even length strings$\}$ is undecidable.

Proof: Suppose $EVEN_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. Run N on ω and <u>accept</u> if N does.
2. Run H on $\langle M_2 \rangle$.
3. If H accepts, <u>accept</u>. If H rejects, <u>reject</u>.

$L(M_2)$ contains all even length strings

$\Updownarrow$

N accepts ω

If N accepts ω, $L(M_2) = \Sigma^*$(contains all even length strings) and S accepts. If N does not accept ω, $L(M_2) = \emptyset$ (does not contain all even length strings) and S rejects. Thus, S is a decider for A_{TM}, which is a contradiction, so $EVEN_{TM}$ is undecidable.

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. Run N on ω and <u>accept</u> if N does.
2. Run H on $\langle M_2 \rangle$.
3. If H accepts, <u>accept</u>. If H rejects, <u>reject</u>.

M_2 halts on ε

$\Updownarrow$

N accepts ω

If N accepts ω, M_2 halts on ε and S accepts. If N does not accept ω, M_2 does not halt on ε and S rejects. Therefore, S is a decider for A_{TM}, which is a contradiction, so $HALT - \varepsilon_{TM}$ is undecidable.

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)} is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. Run N on ω and ~~accept~~ if N accepts.
       reject
2. Run H on $\langle M_2 \rangle$.

M_2 halts on ε

$\Updownarrow$

N accepts ω

Claim: $HALT-\varepsilon_{TM} = \{\langle M\rangle : M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT-\varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
    1. If $x \neq \varepsilon$, reject.
    2. If $x = \varepsilon$, run N on ω and accept if N accepts.
2. Run H on $\langle M_2\rangle$.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ accepts } \omega \end{array}}$$

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)} is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. If $x \neq \varepsilon$, ~~reject~~. **accept**
    2. If $x = \varepsilon$, run N on ω and accept if N accepts.
2. Run H on $\langle M_2 \rangle$.

$$M_2 \text{ halts on } \varepsilon \Updownarrow N \text{ accepts } \omega$$

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
   1. ~~If $x \neq \varepsilon$, reject.~~
   2. If $x = \varepsilon$, run N on ω and accept if N accepts.
2. Run H on $\langle M_2 \rangle$.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ accepts } \omega \end{array}}$$

Claim: $HALT - \varepsilon_{TM} = \{\langle M\rangle : M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N, \omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
   1. If $x \neq \varepsilon$, reject.
   2. If $x = \varepsilon$, run N on ω and accept if N ~~accepts~~. **rejects**
2. Run H on $\langle M_2\rangle$.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ accepts } \omega \end{array}}$$

Claim: $HALT-\varepsilon_{TM}=\{\langle M\rangle : M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT-\varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

S = on input $\langle N,\omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
   1. If $x\neq\varepsilon$, <u>reject</u>.
   2. If $x=\varepsilon$, run N on ω and <u>accept</u> if N ~~accepts~~. **<u>rejects</u>**
2. Run H on $\langle M_2\rangle$.

M_2 halts on ε

$\Updownarrow$

N accepts ω

Doesn't work! N not accepting is not the same as N rejecting!

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)} is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides $\boldsymbol{HALT_{TM}}$:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. Run N on ω and <u>accept</u> if N does.
2. Run H on $\langle M_2 \rangle$.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ halts on } \omega \end{array}}$$

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)} is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides $\boldsymbol{HALT_{TM}}$:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
   1. Run N on ω and <u>accept</u> if N does.
2. Run H on $\langle M_2 \rangle$.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ halts on } \omega \end{array}}$$

Doesn't work! M_2 does not (always) halt when N halts on ω!

Claim: $HALT - \varepsilon_{TM} = \{\langle M \rangle : M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT - \varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides $HALT_{TM}$:

S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
    1. Run N on ω.
    2. <u>Accept</u> if N accepts and <u>reject</u> if N rejects.
2. Run H on $\langle M_2 \rangle$.
3. If H accepts, <u>accept</u>. If H rejects, <u>reject</u>.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ halts on } \omega \end{array}}$$

Claim: $HALT-\varepsilon_{TM} = \{\langle M\rangle\text{:}\, M$ is a TM and halts on empty input (i.e. ε)$\}$ is undecidable.

Proof: Suppose $HALT-\varepsilon_{TM}$ is decidable and let TM H be its decider.

Build a TM S that decides $HALT_{TM}$:

S = on input $\langle N, \omega\rangle$

1. Construct TM M_2 on input $\langle x\rangle$:
   1. Run N on ω.
   2. <u>reject</u>.
2. Run H on $\langle M_2\rangle$.
3. If H accepts, <u>accept</u>. If H rejects, <u>reject</u>.

$$\boxed{\begin{array}{c} M_2 \text{ halts on } \varepsilon \\ \Updownarrow \\ N \text{ halts on } \omega \end{array}}$$

Prove that $ALMOST\_ALL_{DFA} = \{\langle A, \omega \rangle : A \text{ is a DFA and } L(A) = \Sigma^* \backslash \{\omega\}\}$ is decidable. Remember to show that your algorithm meets the requirements of a decider.

You are allowed to use any deciders we learned in class or on homework!

Prove that $ANY_{TM} = \{\langle M \rangle : M$ is a TM and $L(M) \neq \emptyset\}$ is undecidable.

Prove that $ALMOST\_ALL_{DFA} = \{\langle A, \omega\rangle : A \text{ is a DFA and } L(A) = \Sigma^* \backslash \{\omega\}\}$ is decidable. Remember to show that your algorithm meets the requirements of a decider.

Proof: Construct the following Turing machine to decide $ALMOST\_ALL_{DFA}$:

M = on input $\langle A, \omega\rangle$

1. Run A on ω and <u>reject</u> if A accepts.
2. Construct DFA B that only accepts ω.
3. Construct DFA C so that $L(C) = L(A) \cup L(B)$.
4. Run the decider for ALL_{DFA} on C.
5. If the decider accepts, <u>accept</u>. If it rejects, <u>reject</u>.

Every DFA is guaranteed to halt on all input so step 1 will halt. Constructing B and C will occur in finite time, and running a decider is guaranteed to halt. Thus, M is guaranteed to halt and is a decider.

Prove that $ANY_{TM} = \{\langle M \rangle : M$ is a TM and $L(M) \neq \emptyset\}$ is undecidable.

Proof: Suppose ANY_{TM} is decidable and let TM H be its decider.

Build a TM S that decides A_{TM}:

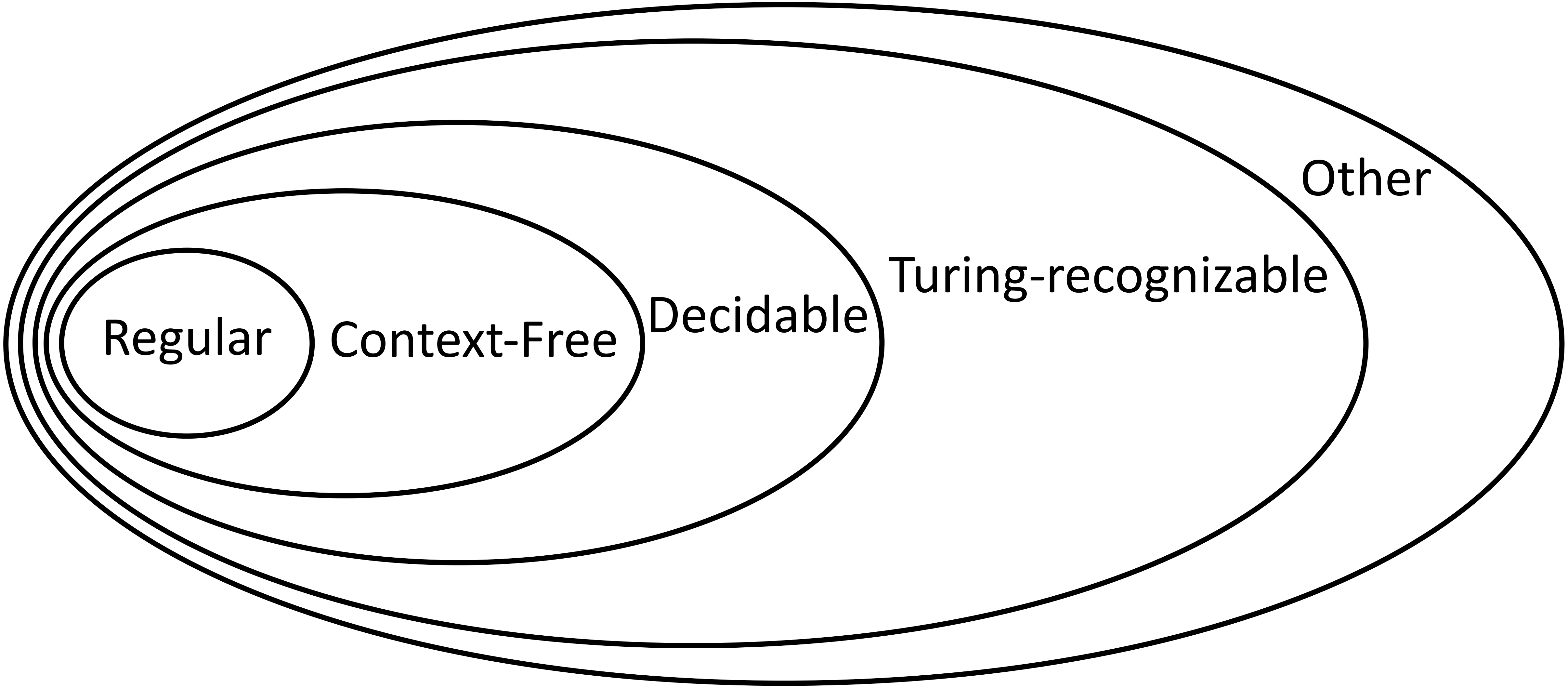
S = on input $\langle N, \omega \rangle$

1. Construct TM M_2 on input $\langle x \rangle$:
   1. Run N on ω and accept if N does.
2. Run H on $\langle M_2 \rangle$.
3. If H accepts, accept. If H rejects, reject.

$$\boxed{\begin{array}{c} L(M_2) \neq \emptyset \\ \Updownarrow \\ N \text{ accepts } \omega \end{array}}$$

If N accepts ω, $L(M_2) = \Sigma^*$ and S accepts. If N does not accept ω, $L(M_2) = \emptyset$ and S rejects. Thus, S is a decider for A_{TM}, which is a contradiction, so ANY_{TM} is undecidable.

Computability Hierarchy



Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega\rangle : M$ is a TM and M does not halt on $\omega\}$ is not Turing-recognizable.

Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega \rangle$: M is a TM and M does not halt on ω} is not Turing-recognizable.

Proof: Suppose $\overline{HALT_{TM}}$ was Turing-recognizable. Let T be its recognizer (i.e., T will accept if a TM does ***not*** halt on some input).

Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega\rangle : M$ is a TM and M does not halt on ω} is not Turing-recognizable.

Proof: Suppose $\overline{HALT_{TM}}$ was Turing-recognizable. Let T be its recognizer (i.e., T will accept if a TM does ***not*** halt on some input).

Consider S on $\langle N, \omega\rangle$:

1. Run N on ω.
2. accept.

$\boldsymbol{HALT_{TM}}$ **recognizer!**

Unrecognizable Language

Claim: $\overline{HALT_{TM}} = \{\langle M, \omega\rangle : M$ is a TM and M does not halt on ω} is not Turing-recognizable.

Proof: Suppose $\overline{HALT_{TM}}$ was Turing-recognizable. Let T be its recognizer (i.e., T will accept if a TM does ***not*** halt on some input).

Consider S on $\langle N, \omega\rangle$:

1. Run N on ω.
2. accept.

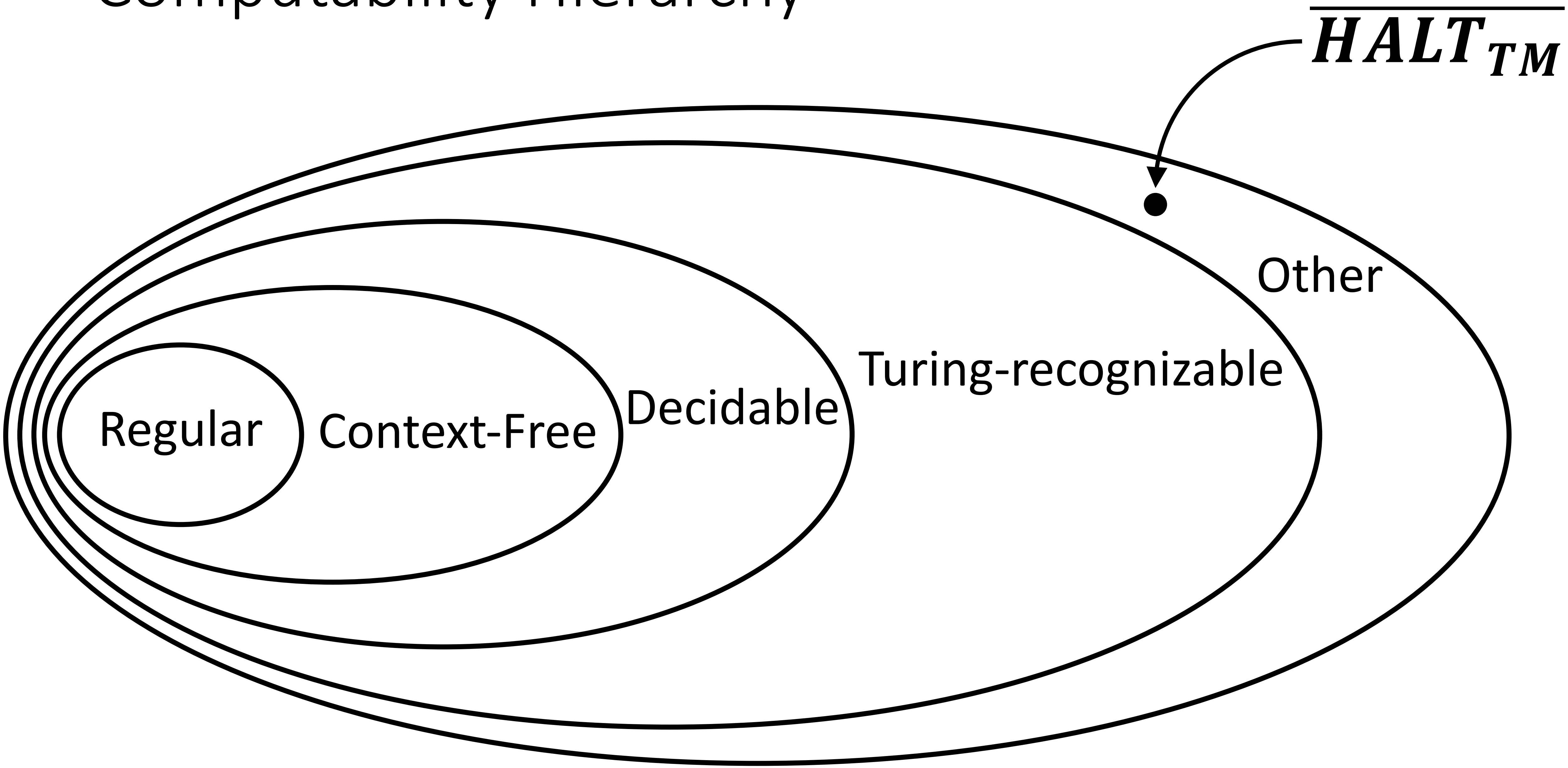
$\boldsymbol{HALT_{TM}}$ **recognizer!**

Consider V on $\langle N, \omega\rangle$:

1. Run T on $\langle N, \omega\rangle$ and run S on $\langle N, \omega\rangle$ in parallel.
2. If T accepts, reject. If S accepts, accept.

$\boldsymbol{HALT_{TM}}$ **decider!**

Computability Hierarchy



Unrecognizable Language

Claim: A language is decidable $\Leftrightarrow$ it and its complement are Turing-recognizable.

Proof: $\Longrightarrow$ If a language is decidable, its complement is also decidable (just reverse accept/reject conditions) and decidable languages are recognizable.

$\Longleftarrow$ If A and $\bar{A}$ are both Turing-recognizable, let M_1 and M_2 be recognizers for A and $\bar{A}$. Consider the following TM:

M = on input ω

1. Run both M_1 and M_2 on ω in parallel (alternate instructions).
2. If M_1 accepts, <u>accept</u>. If M_2 accepts, <u>reject</u>.

Since $\omega \in A$ or $\bar{A}$, M_1 or M_2 must accept (halts on input). Thus, M is a decider for A.

Beyond Decidability

What if $HALT_{TM}$ were “decidable”?

Beyond Decidability

What if $HALT_{TM}$ were “decidable”?

Goldbach’s Conjecture:

- 280-year-old open problem.
- Every integer ≥ 2 is sum of two primes.

Beyond Decidability

What if $HALT_{TM}$ were "decidable"?

Goldbach's Conjecture:

- 280-year-old open problem.
- Every integer ≥ 2 is sum of two primes.

Consider G on $\langle x \rangle$:

1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to n, reject.
3. Increment n and loop to step 1.

Beyond Decidability

```
public boolean G() {
   int i = 2;
   while (true) {
      boolean found = false;
      for (int n = 1; n < i; n++) {
         for (int m = 1; m < i; m++) {
            if (isPrime(n) && isPrime(m) && m + n = i) {
               found = true;
            }
         }
      }
      if (!found) {
         return false;
      }
      i++;
   }
}
```

Beyond Decidability

What if $HALT_{TM}$ were "decidable"?

Goldbach's Conjecture:

- 280-year-old open problem.
- Every integer ≥ 2 is sum of two primes.

Consider G on $\langle x \rangle$:

1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to n, <u>reject</u>.
3. Increment n and loop to step 1.

What does it mean if G halts?
What does it mean if G does not halt?

Beyond Decidability

What if $HALT_{TM}$ were "decidable"?

Goldbach's Conjecture:

- 280-year-old open problem.
- Every integer ≥ 2 is sum of two primes.

Consider G on $\langle x \rangle$:

1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to n, reject.
3. Increment n and loop to step 1.

What does it mean if G halts? **Goldbach's conjecture is false!**

What does it mean if G does not halt? **Goldbach's conjecture is true!**

Beyond Decidability

What if $HALT_{TM}$ were "decidable"?

Goldbach's Conjecture:

- 280-year-old open problem.
- Every integer ≥ 2 is sum of two primes.

Consider G on $\langle x \rangle$:

1. For $n = 2$, check each pair of prime number $< n$.
2. If no pair sums to n, reject.
3. Increment n and loop to step 1.

What does it mean if G halts? **Goldbach's conjecture is false!**

What does it mean if G does not halt? **Goldbach's conjecture is true!**

Turns out you can do this for lots of open problems over natural numbers (twin prime conjecture,…)