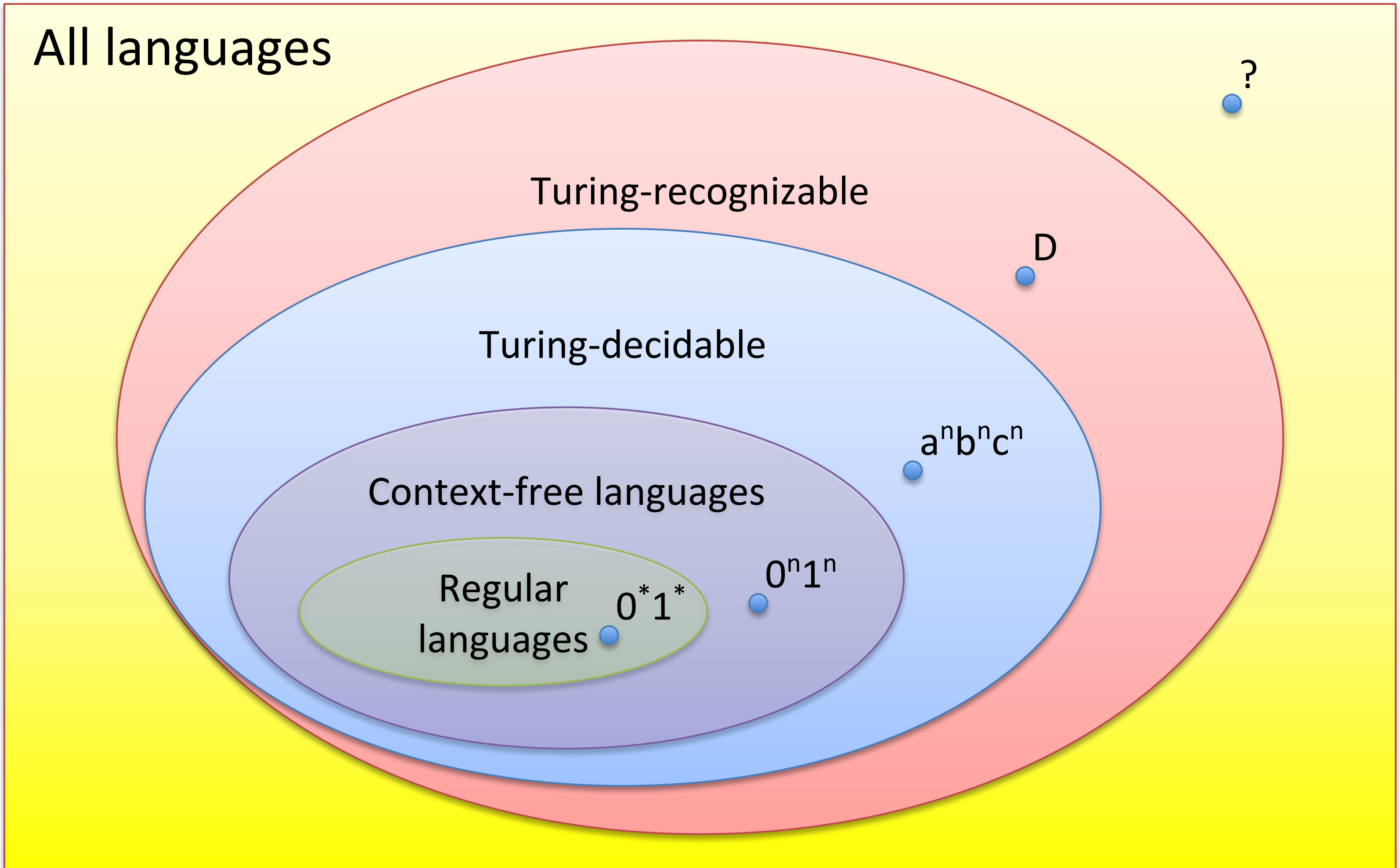
The Halting Problem

Sipser 4.2 (pages 173-182)

Taking stock



Are there problems a computer can't solve?!

- But they seem so powerful…

- What about software verification?
  - Given a program and a specification of what it should do, can we check if it is correct?

What about deciding TMs?

- $A_{TM} = \{<M,w> \mid M \text{ is a TM and } M \text{ accepts } w\}$
- Theorem 4.11: A_{TM} is undecidable!
- Is it even recognizable?

What about deciding TMs?

- $A_{TM} = \{<M,w> \mid M$ is a TM and M accepts $w\}$
- Theorem 4.11: A_{TM} is undecidable!
- Is it even recognizable?
- Let U = "On input $<M,w>$, where M is a TM:
  1. Simulate M on input w.
  2. If M ever enters its accept state, *accept*;
     if M ever enters its reject state, *reject*."

Towards proving undecidability

- Cantor 1873: How can we tell whether one infinite set is "larger" than another?
- A function f from A to B is
  - One-to-one if $f(x) \neq f(y)$ if $x \neq y$
    - (f never maps two elements to the same value)
  - Onto if every element of B is hit
- A correspondence is a function that is both one-to-one and onto

Countable sets

- Sets A and B have the same size if:
  - A and B are finite with the same number of elements
  - A and B are infinite with a correspondence between them
- A set is countable if it is finite or has the same size as $\mathbf{N}$
  - (*natural numbers* $1, 2, 3, \ldots$)

For example…

- E = { even natural numbers } is countable
- Define $f : \mathbf{N} \rightarrow E$ as $f(n) = 2n$

n	f(n)
1	2
2	4
3	6
…	…

Diagonalization

- Theorem 4.17: R is uncountable.
- Proof:

  By contradiction. Assume there is a correspondence. We find a real number $x \neq f(n)$ for any natural number n.

n	f(n)
1	3.**1**4159…
2	55.5**5**555…
3	0.12**3**45…
4	0.500**0**0…
…	…

Put your thinking caps on!

- How do we show that:
  - Σ^* is countable
    - assume $\Sigma = \{0,1\}$
  - B = { all infinite binary sequences } is uncountable

Uh-oh…

- Theorem 4.18: Some languages are not Turing-recognizable.
- Proof:

  Let L be the set of all languages over Σ. Define a correspondence from L to B.

Uh-oh...

- Theorem 4.18: Some languages are not Turing-recognizable.
- Proof:

Let L be the set of all languages over Σ. Define a correspondence from L to B.

$$\Sigma^* = \{\ \varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}$$

Uh-oh...

- Theorem 4.18: Some languages are not Turing-recognizable.
- Proof:

Let L be the set of all languages over Σ. Define a correspondence from L to B.

$$\Sigma^* = \{\ \varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}$$

$$X_A = \quad 0 \quad 1 \quad 0 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad \ldots$$

Uh-oh…

- Theorem 4.18: Some languages are not Turing-recognizable.
- Proof:

  Let L be the set of all languages over Σ. Define a correspondence from L to B.

$$\Sigma^* = \{\ \varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, \ldots\}$$

$$A = \{\ 0, 00, 01, 000, 001, \ldots\}$$

$$X_A = \ 0\ \ 1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ \ldots$$

The Halting Problem

- A_{TM} = {<M,w> | M is a TM and M accepts w}
- Theorem 4.11: A_{TM} is undecidable.

The Halting Problem

- A_{TM} = {<M,w> | M is a TM and M accepts w}
- Theorem 4.11: A_{TM} is undecidable.
- Proof:

  By contradiction. Assume H is a decider for A_{TM}.

The Halting Problem

- A_{TM} = {<M,w> | M is a TM and M accepts w}
- Theorem 4.11: A_{TM} is undecidable.
- Proof:

By contradiction. Assume H is a decider for A_{TM}. We construct a TM

D = "On input <M>, where M is a TM:

1. Run H in input <M, <M>>
2. If H accepts, reject; if H rejects, accept."

Huh?

$$D(<M>) = \begin{cases} accept & \text{if } M \text{ does not accept } <M> \\ reject & \text{if } M \text{ accepts } <M> \end{cases}$$

Huh?

$$D(\langle M \rangle) = \begin{cases} accept & \text{if } M \text{ does not accept } \langle M \rangle \\ reject & \text{if } M \text{ accepts } \langle M \rangle \end{cases}$$

- What if we run D on $\langle D \rangle$?

Huh?

$$D(<M>) = \begin{cases} accept & \text{if } M \text{ does not accept } <M> \\ reject & \text{if } M \text{ accepts } <M> \end{cases}$$

- What if we run D on $<D>$?
- Then

$$D(<D>) = \begin{cases} accept & \text{if } D \text{ does not accept } <D> \\ reject & \text{if } D \text{ accepts } <D> \end{cases}$$

Huh?

$$D(<M>) = \begin{cases} accept & \text{if } M \text{ does not accept } <M> \\ reject & \text{if } M \text{ accepts } <M> \end{cases}$$

- What if we run D on $<D>$?
- Then

$$D(<D>) = \begin{cases} accept & \text{if } D \text{ does not accept } <D> \\ reject & \text{if } D \text{ accepts } <D> \end{cases}$$

- Contradiction! Then H cannot exist and A_{TM} is undecidable.

Where is the diagonalization?

- Running a machine on its description

	$<M_1>$	$<M_2>$	$<M_3>$	$<M_4>$	…
M_1	*accept*		*accept*		
M_2	*accept*	*accept*	*accept*	*accept*	
M_3					
M_4	*accept*	*accept*			
…					

Where is the diagonalization?

- Running H on a machine and its description

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	…
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
…					

Where is the diagonalization?

- Adding D to the picture

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	…	$\langle D \rangle$	…
M_1	***accept***	*reject*	*accept*	*reject*			
M_2	*accept*	***accept***	*accept*	*accept*			
M_3	*reject*	*reject*	***reject***	*reject*			
M_4	*accept*	*accept*	*reject*	***reject***			
…							
D	***reject***	***reject***	***accept***	***accept***		**?**	
…							

So have we found a language that is not Turing-recognizable?

- Theorem 4.22: A language is decidable iff it and its complement are recognizable

So have we found a language that is not Turing-recognizable?

- Theorem 4.22: A language is decidable iff it and its complement are recognizable

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

Updating the picture