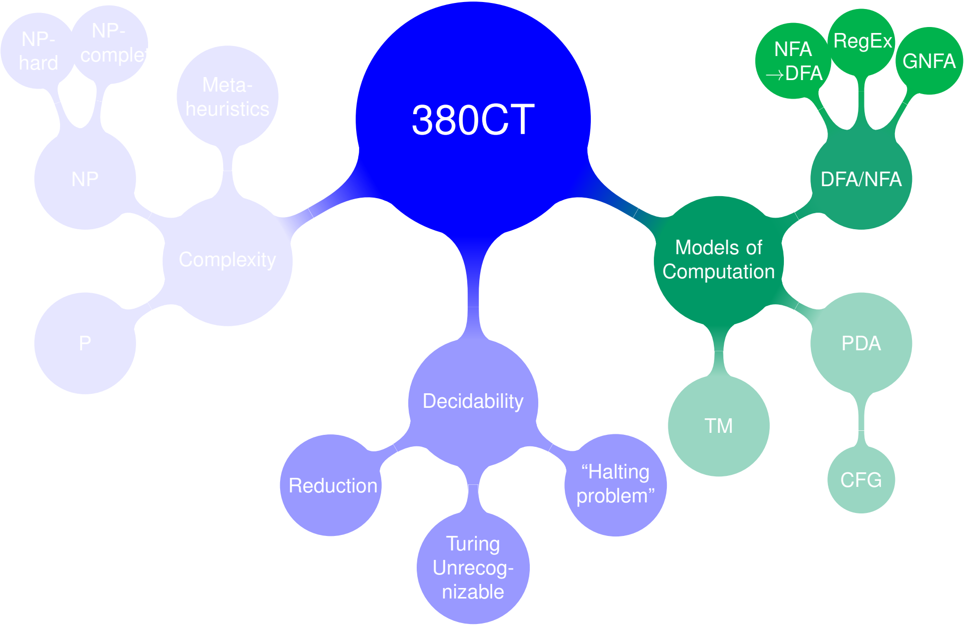
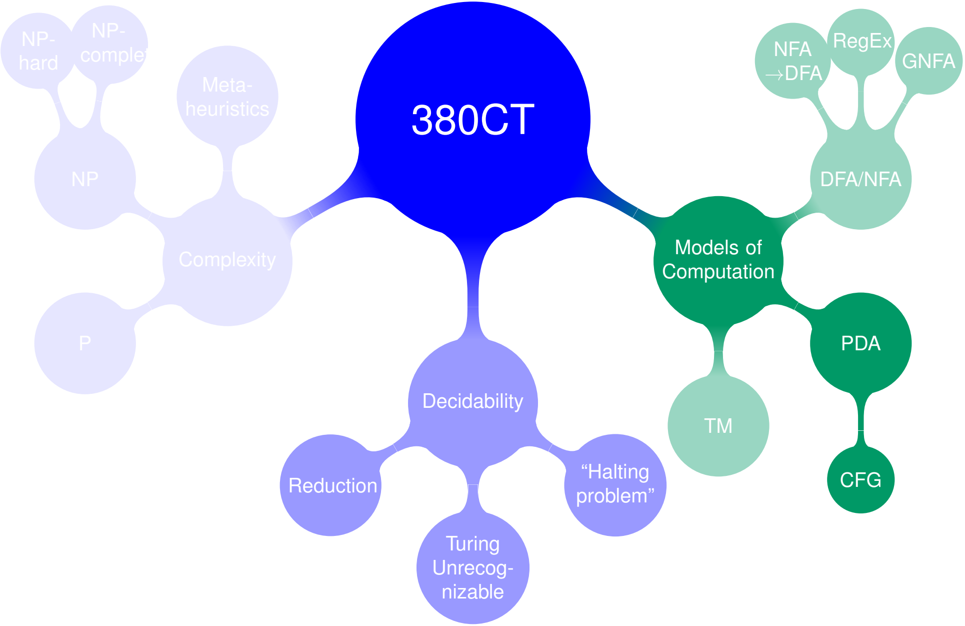
Decidability

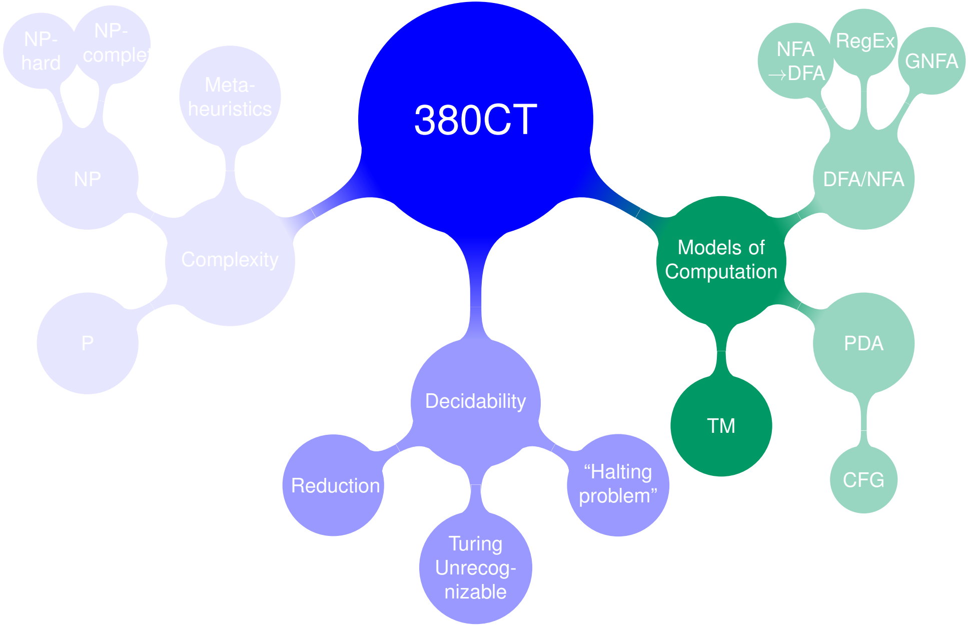
Dr Kamal Bentahar

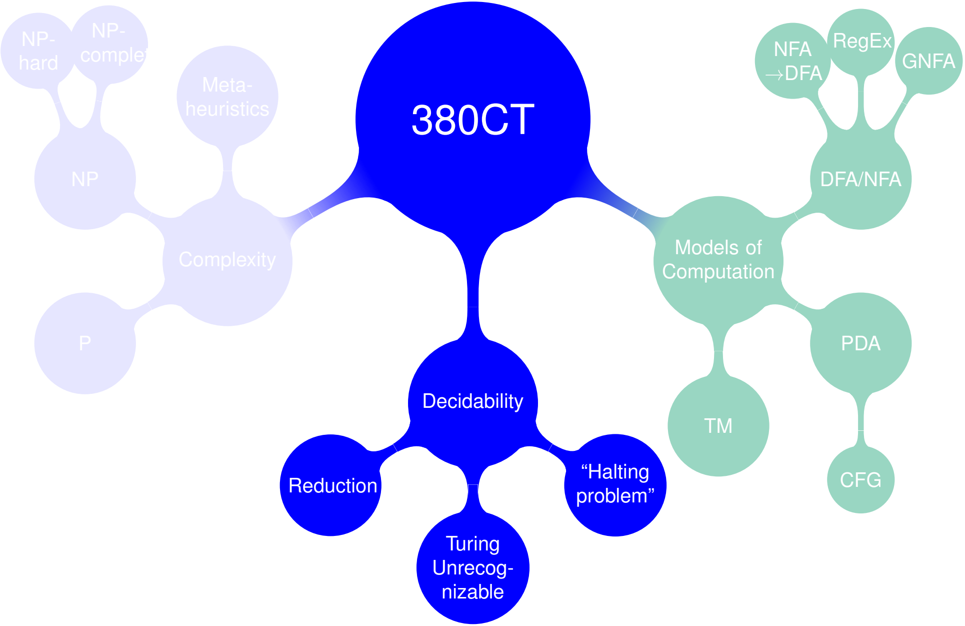
School of Engineering, Environment and Computing
Coventry University

14/11/2016


380CT
Complexity
NP
NP-hard
NP-complete
P
Meta-heuristics
Decidability
Reduction
Turing Unrecog-nizable
“Halting problem”
Models of Computation
DFA/NFA
NFA →DFA
RegEx
GNFA
PDA
CFG
TM


380CT
Complexity
NP
NP-hard
NP-complete
Meta-heuristics
P
Decidability
Reduction
Turing Unrecog-nizable
"Halting problem"
Models of Computation
DFA/NFA
NFA →DFA
RegEx
GNFA
TM
PDA
CFG


380CT
Models of Computation
DFA/NFA
NFA →DFA
RegEx
GNFA
PDA
CFG
TM
Decidability
Reduction
Turing Unrecog-nizable
“Halting problem”
Complexity
NP
NP-hard
NP-complete
Meta-heuristics
P


380CT
Complexity
NP
NP-hard
NP-complet
Meta-heuristics
P
Decidability
Reduction
Turing Unrecog-nizable
“Halting problem”
Models of Computation
DFA/NFA
NFA →DFA
RegEx
GNFA
TM
PDA
CFG

Last time. . .

Turing recognizable/decidable languages

- A language is (Turing) **recognizable** if some TM *recognizes* it.
- A language is (Turing) **decidable** if some TM *decides* it.
  (All branches of a NTM need to reject for it to reject a string.)

Last time...

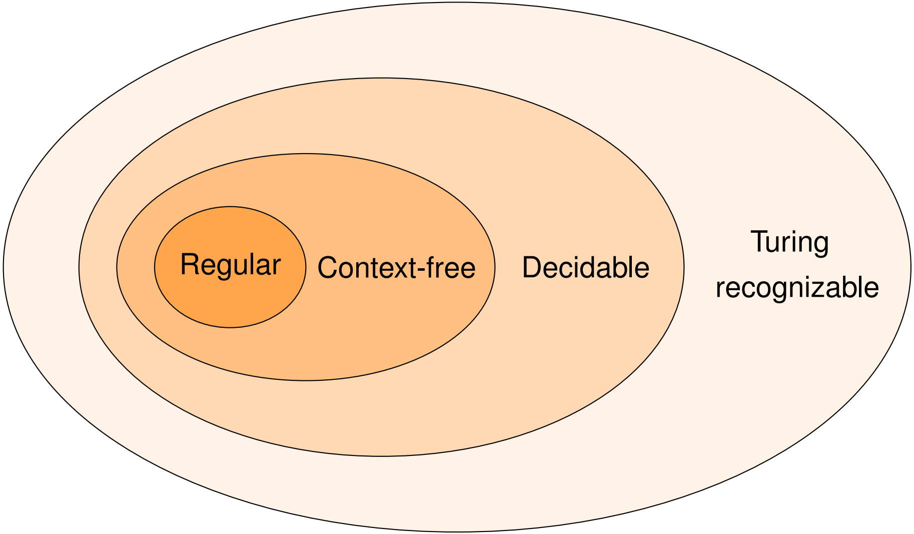
Turing recognizable/decidable languages

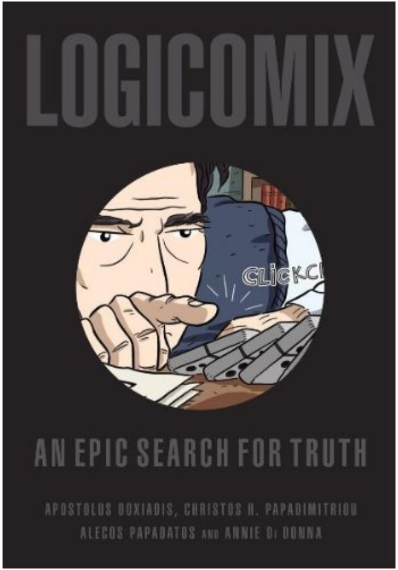
- A language is (Turing) **recognizable** if some TM *recognizes* it.
- A language is (Turing) **decidable** if some TM *decides* it.
  (All branches of a NTM need to reject for it to reject a string.)

The Church-Turing Thesis – Algorithms

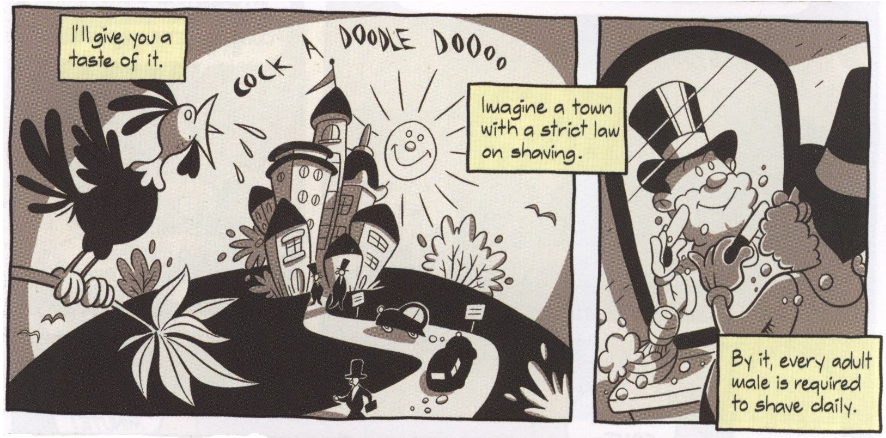
Intuitive concept of algorithms = Turing machine algorithms

Venn diagram



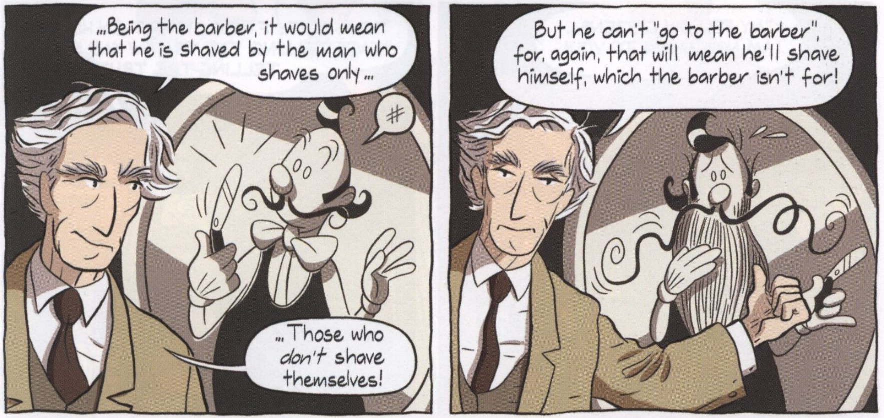

LOGICOMIX
CLICKCL
AN EPIC SEARCH FOR TRUTH
APOSTOLOS DOXIADIS, CHRISTOS H. PAPADIMITRIOU
ALECOS PAPADATOS AND ANNIE DI DONNA


In my life to date, I have written dozens of books and hundreds of articles... I've given thousands of lectures.
But I suspect my name will survive, if it does at all...
...For a confounded paradox I discovered that year.
A paradox that brought Logic upside down...

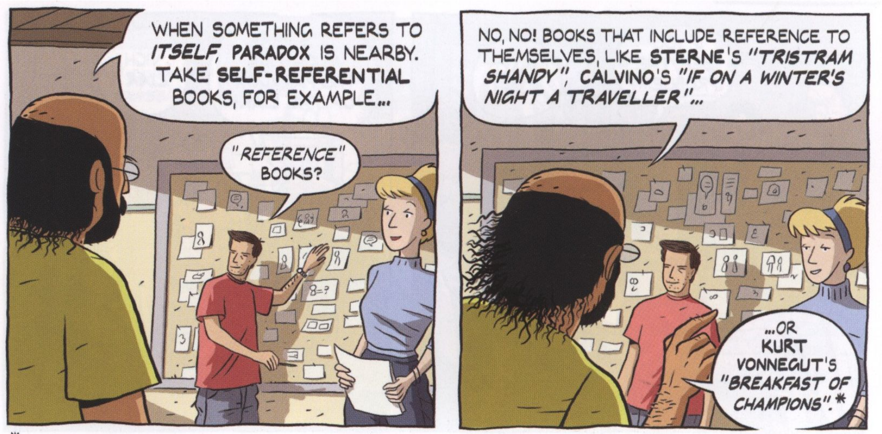

I'll give you a taste of it.
COCK A DOODLE DOOOoo
Imagine a town with a strict law on shaving.
By it, every adult male is required to shave daily.


But it's not obligatory to shave yourself...
...For those who don't want to, there is a barber.
In fact, the law decrees: "Those who don't shave themselves are shaved by the barber."


"Those who don't shave themselves are shaved by the barber." It sounds innocuous... However, if taken literally, it leads straight to paradox!
For, you see, the question arises:
"Who will shave the *barber?*"
?
He obviously cannot choose to shave *himself*, for...

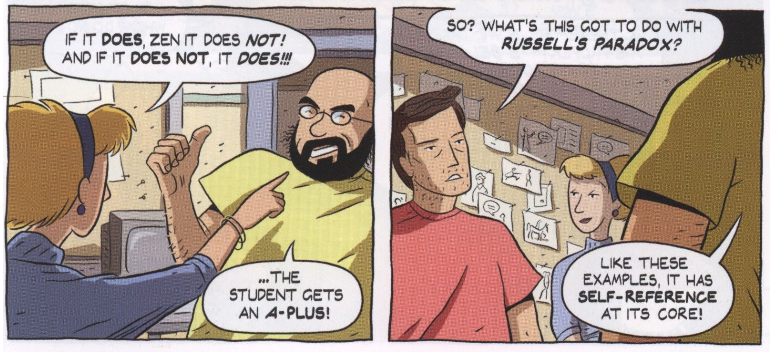

...Being the barber, it would mean that he is shaved by the man who shaves only...
#
...Those who don't shave themselves!
But he can't "go to the barber", for, again, that will mean he'll shave himself, which the barber isn't for!


"MY FELLOW CITIZENS...
I AM NOW LYING TO YOU!"
!
THINK OF IT: IF HE IS LYING, THEN HE IS IN FACT TELLING THE TRUTH! AND IF HE IS TELLING THE TRUTH...
HE IS LYING!
?



* Of course **LOGICOMIX** is also self-referential.


SUPPOSE NOW YOU MAKE A COMPLETE CATALOGUE OF ALL BOOKS THAT ARE NOT SELF-REFERENTIAL!
IT WILL BE A BIG CATALOGUE, ZIS...
SURE! BUT THE QUESTION IS...
?
WILL IT CONTAIN ITSELF?
GOT IT!


IF IT DOES, ZEN IT DOES NOT! AND IF IT DOES NOT, IT DOES!!!
...THE STUDENT GETS AN A-PLUS!
SO? WHAT'S THIS GOT TO DO WITH RUSSELL'S PARADOX?
LIKE THESE EXAMPLES, IT HAS SELF-REFERENCE AT ITS CORE!

Notation for encodings of objects

We need to **encode** objects so that TMs can operate on them.

We denote the encoding of *object* by writing it between *angled brackets*:

$$\langle \textit{object} \rangle$$

Decidable problems – examples

- Problems about regular languages

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w \rangle \mid B \text{ is a DFA that accepts input string } w\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
- Problems about context-free languages

Decidable problems – examples

- Problems about regular languages
  - $A_{\mathrm{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\mathrm{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\mathrm{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\mathrm{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
- Problems about context-free languages
  - $A_{\mathrm{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\mathrm{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\mathrm{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\mathrm{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\mathrm{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
- Problems about context-free languages
  - $A_{\mathrm{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
  - $E_{\mathrm{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
- Problems about context-free languages
  - $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
  - $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
- Problems about context-free languages
  - $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
  - $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$

Is $EQ_{\text{CFG}} = \{\langle G, H\rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H)\}$ decidable?

Decidable problems – examples

- Problems about regular languages
  - $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
  - $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
  - $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
  - $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
- Problems about context-free languages
  - $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
  - $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$

Is $EQ_{\text{CFG}} = \{\langle G, H\rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H)\}$ decidable?

No!

Undecidability

- Computers seem so powerful – can they solve all problems?

Undecidability

- Computers seem so powerful – can they solve all problems?
- Theorem: Computers are limited in a fundamental way.

Undecidability

- Computers seem so powerful – can they solve all problems?
- Theorem: Computers are limited in a fundamental way.
- One type of unsolvable problem: given a computer program and a precise specification of what that program is supposed to do, verify that the program performs as specified.

Undecidability

- Computers seem so powerful – can they solve all problems?
- Theorem: Computers are limited in a fundamental way.
- One type of unsolvable problem: given a computer program and a precise specification of what that program is supposed to do, verify that the program performs as specified.
- $\rightarrow$ **Software verification** is, in general, not solvable by computer!

Undecidability – the **Halting Problem**

Consider

$$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Undecidability – the **Halting Problem**

Consider

$$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Proof by contradiction: suppose a decider D exists such that

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

Undecidability – the **Halting Problem**

Consider

$$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Proof by contradiction: suppose a decider D exists such that

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

Now run it on itself:

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

Undecidability – the **Halting Problem**

Consider

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Proof by contradiction: suppose a decider D exists such that

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

Now run it on itself:

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

Does it accept or reject?

Undecidability – the **Halting Problem**

Consider

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Proof by contradiction: suppose a decider D exists such that

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

Now run it on itself:

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

Does it accept or reject?

It rejects if it accepts, and it accepts if it doesn't accept!!

Undecidability – the **Halting Problem**

Consider

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Proof by contradiction: suppose a decider D exists such that

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

Now run it on itself:

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

Does it accept or reject?

It rejects if it accepts, and it accepts if it doesn't accept!!

There is a problem with the assumption that such a D exists.

Undecidability – the **Halting Problem**

Consider

$$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$$

Proof by contradiction: suppose a decider D exists such that

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

Now run it on itself:

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

Does it accept or reject?

It rejects if it accepts, and it accepts if it doesn't accept!!

There is a problem with the assumption that such a D exists.

The acceptance problem A_{TM} therefore cannot be decidable.

Undecidability – the **Halting Problem**

The Halting Problem is

$$HALT_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \textit{ halts} \text{ on input } w\}$$

and is also undecidable. Proof uses "reduction" to A_{TM}.

Last time. . .

1. Acceptance problems

(red: undecidable, blue: decidable)

Last time. . .

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$

(red: undecidable, blue: decidable)

Last time. . .

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems

(red: undecidable, blue: decidable)

Last time. . .

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$

(red: undecidable, blue: decidable)

Last time. . .

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
   2.2 $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\mathrm{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\mathrm{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\mathrm{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\mathrm{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
   2.2 $E_{\mathrm{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$
3. Language equality problems

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
   2.2 $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$
3. Language equality problems
   3.1 $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
   2.2 $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$
3. Language equality problems
   3.1 $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
   3.2 $EQ_{\text{CFG}} = \{\langle G, H\rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H)\}$

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
   2.2 $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$
3. Language equality problems
   3.1 $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
   3.2 $EQ_{\text{CFG}} = \{\langle G, H\rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H)\}$
4. Halting problem

(red: undecidable, blue: decidable)

Last time...

1. Acceptance problems
   1.1 $A_{\text{DFA}} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
   1.2 $A_{\text{NFA}} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts input string } w\}$
   1.3 $A_{\text{CFG}} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates string } w\}$
   1.4 $A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
2. Language emptiness problems
   2.1 $E_{\text{DFA}} = \{\langle A\rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$
   2.2 $E_{\text{CFG}} = \{\langle G\rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$
3. Language equality problems
   3.1 $EQ_{\text{DFA}} = \{\langle A, B\rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B)\}$
   3.2 $EQ_{\text{CFG}} = \{\langle G, H\rangle \mid G \text{ and } H \text{ are CFGs and } L(G) = L(H)\}$
4. Halting problem
   4.1 $HALT_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \textbf{ halts} \text{ on input } w\}$

(red: undecidable, blue: decidable)

The **reducibility** concept

How do we show that $\mathit{HALT}_{\mathrm{TM}}$ is undecidable?

The **reducibility** concept

How do we show that $HALT_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\mathrm{TM}}$ is decidable $\Longrightarrow$ A_{TM} is decidable.

The **reducibility** concept

How do we show that $\mathit{HALT}_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $\mathit{HALT}_{\mathrm{TM}}$ is decidable $\implies$ A_{TM} is decidable.

Idea: use $\mathit{HALT}_{\mathrm{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $\mathit{HALT}_{\mathrm{TM}}$.

The **reducibility** concept

How do we show that $HALT_{\text{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\text{TM}}$ is decidable $\Longrightarrow$ A_{TM} is decidable.

Idea: use $HALT_{\text{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\text{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\text{TM}}$.

The **reducibility** concept

How do we show that $HALT_{\text{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\text{TM}}$ is decidable $\implies$ A_{TM} is decidable.

Idea: use $HALT_{\text{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\text{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\text{TM}}$.
- Construct TM D to decide A_{TM}, with D operating as follows:
  $D =$ "On input $\langle M, w \rangle$:

The **reducibility** concept

How do we show that $HALT_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\mathrm{TM}}$ is decidable $\implies$ A_{TM} is decidable.

Idea: use $HALT_{\mathrm{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\mathrm{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\mathrm{TM}}$.
- Construct TM D to decide A_{TM}, with D operating as follows:

  $D =$ "On input $\langle M, w \rangle$:
  1. Run TM H on input $\langle M, w \rangle$.

The **reducibility** concept

How do we show that $HALT_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\mathrm{TM}}$ is decidable $\Longrightarrow$ A_{TM} is decidable.

Idea: use $HALT_{\mathrm{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\mathrm{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\mathrm{TM}}$.
- Construct TM D to decide A_{TM}, with D operating as follows:

  $D =$ "On input $\langle M, w\rangle$:
  1. Run TM H on input $\langle M, w\rangle$.
  2. If H rejects, *reject*.

The **reducibility** concept

How do we show that $HALT_{TM}$ is undecidable?

Proof by contradiction: $HALT_{TM}$ is decidable $\Longrightarrow$ A_{TM} is decidable.

Idea: use $HALT_{TM}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{TM}$.

Proof

- Suppose there exists a TM H that decides $HALT_{TM}$.
- Construct TM D to decide A_{TM}, with D operating as follows:

  $D =$ "On input $\langle M, w\rangle$:
  1. Run TM H on input $\langle M, w\rangle$.
  2. If H rejects, *reject*.
  3. If H accepts, simulate M on w until it halts.

The **reducibility** concept

How do we show that $HALT_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\mathrm{TM}}$ is decidable $\Longrightarrow$ A_{TM} is decidable.

Idea: use $HALT_{\mathrm{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\mathrm{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\mathrm{TM}}$.
- Construct TM D to decide A_{TM}, with D operating as follows:

  $D =$ "On input $\langle M, w\rangle$:
  1. Run TM H on input $\langle M, w\rangle$.
  2. If H rejects, *reject*.
  3. If H accepts, simulate M on w until it halts.
  4. If M has accepted, *accept*; if M has rejected, *reject*."

The **reducibility** concept

How do we show that $HALT_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\mathrm{TM}}$ is decidable $\implies$ A_{TM} is decidable.

Idea: use $HALT_{\mathrm{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\mathrm{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\mathrm{TM}}$.
- Construct TM D to decide A_{TM}, with D operating as follows:

  $D =$ "On input $\langle M, w\rangle$:
  1. Run TM H on input $\langle M, w\rangle$.
  2. If H rejects, *reject*.
  3. If H accepts, simulate M on w until it halts.
  4. If M has accepted, *accept*; if M has rejected, *reject*."
- If H decides $HALT_{\mathrm{TM}}$, then D decides A_{TM}.

The **reducibility** concept

How do we show that $HALT_{\mathrm{TM}}$ is undecidable?

Proof by contradiction: $HALT_{\mathrm{TM}}$ is decidable $\Longrightarrow$ A_{TM} is decidable.

Idea: use $HALT_{\mathrm{TM}}$'s decider to decide A_{TM} – *"reduce"* A_{TM} to $HALT_{\mathrm{TM}}$.

Proof

- Suppose there exists a TM H that decides $HALT_{\mathrm{TM}}$.
- Construct TM D to decide A_{TM}, with D operating as follows:

  $D =$ "On input $\langle M, w \rangle$:
  1. Run TM H on input $\langle M, w \rangle$.
  2. If H rejects, *reject*.
  3. If H accepts, simulate M on w until it halts.
  4. If M has accepted, *accept*; if M has rejected, *reject*."
- If H decides $HALT_{\mathrm{TM}}$, then D decides A_{TM}.
- Since A_{TM} is undecidable, $HALT_{\mathrm{TM}}$ must also be undecidable.

More undecidable problems!

Using reducibility we can show that the following problems are all undecidable

1. $E_{\text{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")

More undecidable problems!

Using reducibility we can show that the following problems are all undecidable

1. $E_{\text{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")
2. $REGULAR_{\text{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language}\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")

More undecidable problems!

Using reducibility we can show that the following problems are all undecidable

1. $E_{\text{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")
2. $REGULAR_{\text{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language}\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")
3. $EQ_{\text{TM}} = \{\langle M_1, M_2\rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$
   Reduce E_{TM} to it. ("Proof by reduction from E_{TM}")

More undecidable problems!

Using reducibility we can show that the following problems are all undecidable

1. $E_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")
2. $REGULAR_{TM} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language}\}$
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")
3. $EQ_{TM} = \{\langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2)\}$
   Reduce E_{TM} to it. ("Proof by reduction from E_{TM}")
4. Post Correspondence Problem (PCP).
   Reduce A_{TM} to it. ("Proof by reduction from A_{TM}")

Turing-**un**recognizable languages!

Theorem

L is decidable $\iff$ both L and $\overline{L}$ are *Turing-recognizable*

Turing-**un**recognizable languages!

Theorem

L is decidable $\iff$ both L and $\overline{L}$ are *Turing-recognizable*

- Take $L = A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$

Corollary

$\overline{A}_{\mathrm{TM}}$ is *not* Turing-recognizable.

Turing-**un**recognizable languages!

Theorem

L is decidable $\iff$ both L and $\overline{L}$ are *Turing-recognizable*

- Take $L = A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
- We know L is Turing-recognizable.

Corollary

$\overline{A}_{\mathrm{TM}}$ is *not* Turing-recognizable.

Turing-**un**recognizable languages!

Theorem

L is decidable $\iff$ both L and $\overline{L}$ are *Turing-recognizable*

- Take $L = A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
- We know L is Turing-recognizable.
- If $\overline{L} = \overline{A}_{\mathrm{TM}}$ were also Turing-recognizable then A_{TM} would be decidable.

Corollary

$\overline{A}_{\mathrm{TM}}$ is *not* Turing-recognizable.

Turing-**un**recognizable languages!

Theorem

L is decidable $\iff$ both L and $\overline{L}$ are *Turing-recognizable*

- Take $L = A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
- We know L is Turing-recognizable.
- If $\overline{L} = \overline{A}_{\text{TM}}$ were also Turing-recognizable then A_{TM} would be decidable.
- But we know A_{TM} is not decidable!

Corollary

$\overline{A}_{\text{TM}}$ is *not* Turing-recognizable.

Turing-**un**recognizable languages!

Theorem

L is decidable $\iff$ both L and $\overline{L}$ are *Turing-recognizable*

- Take $L = A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM and } M \text{ accepts } w\}$
- We know L is Turing-recognizable.
- If $\overline{L} = \overline{A}_{\text{TM}}$ were also Turing-recognizable then A_{TM} would be decidable.
- But we know A_{TM} is not decidable!
- So $\overline{A}_{\text{TM}}$ cannot be Turing-recognizable.

Corollary

$\overline{A}_{\text{TM}}$ is *not* Turing-recognizable.

Time Complexity

- Being decidable means that an algorithm exists to decide the problem.

Time Complexity

- Being decidable means that an algorithm exists to decide the problem.
- However, the algorithm may still be *practically* ineffective because of its **time** and/or **space** cost.

Time Complexity

- Being decidable means that an algorithm exists to decide the problem.
- However, the algorithm may still be *practically* ineffective because of its **time** and/or **space** cost.

Time Complexity

- Being decidable means that an algorithm exists to decide the problem.
- However, the algorithm may still be *practically* ineffective because of its **time** and/or **space** cost.

Definition

The **running time** or **time complexity** of a TM that always halts is the maximum number of steps $f(n)$ that it makes on any input of length n.

We say that M runs in time $f(n)$; and that M is an $f(n)$-time TM.