G52MAL
Machines and Their Languages
Lecture 8
Proving Languages Not to Be Regular

Henrik Nilsson

University of Nottingham
Are there Non-regular Languages?

The regular languages are those that can be recognized by *finite* automata; i.e. machines with finite memory.
Are there Non-regular Languages?

The regular languages are those that can be recognized by *finite* automata; i.e. machines with finite memory.

Are there languages that are not regular?
Are there Non-regular Languages?

The regular languages are those that can be recognized by \textit{finite} automata; i.e. machines with finite memory.

Are there languages that are not regular?

Yes! One example:

$$L = \{0^i1^i \mid i \in \mathbb{N} \}$$
Are there Non-regular Languages?

The regular languages are those that can be recognized by *finite* automata; i.e. machines with finite memory.

Are there languages that are not regular? Yes! One example:

\[ L = \{0^i 1^i \mid i \in \mathbb{N}\} \]

Why? Intuitively: Need to count *arbitrarily far* to check if any given word is accepted. We cannot count arbitrarily far if we only have a finite memory!
Could A Computer Decide $L$? (1)

How can we check if a word belongs to a non-regular language like

$$L = \{0^i1^i \mid i \in \mathbb{N}\}$$
Could A Computer Decide $L$? (1)

How can we check if a word belongs to a non-regular language like

$$L = \{0^i1^i \mid i \in \mathbb{N}\}$$

Can a computer do it?
Could A Computer Decide $L$? (1)

How can we check if a word belongs to a non-regular language like

$$L = \{0^i1^i \mid i \in \mathbb{N}\}$$

Can a computer do it? Can you write a program to check if a given word $w \in L$? Would it work?
Could A Computer Decide $L$? (1)

How can we check if a word belongs to a non-regular language like

$$L = \{0^i1^i \mid i \in \mathbb{N}\}$$

Can a computer do it? Can you write a program to check if a given word $w \in L$? Would it work?

- In theory, no! Anything we physically build is necessarily finite.
Could A Computer Decide $L$? (1)

How can we check if a word belongs to a non-regular language like

$$L = \{0^i 1^i \mid i \in \mathbb{N}\}$$

Can a computer do it? Can you write a program to check if a given word $w \in L$? Would it work?

- In theory, no! Anything we physically build is necessarily finite.
- In practice, of course! It doesn’t take that many bits to count as far as we could possibly want.
Could A Computer Decide $L$? (2)

Example: 128 bits = 16 bytes. Assume a computer running at 1000 GHz, counting one symbol each $10^{-12}$ s.
Could A Computer Decide $L$? (2)

Example: 128 bits = 16 bytes. Assume a computer running at 1000 GHz, counting one symbol each $10^{-12}$ s.

How long before we need more bits to count further?
Example: 128 bits = 16 bytes. Assume a computer running at 1000 GHz, counting one symbol each $10^{-12}$ s.

How long before we need more bits to count further?

About $10^{19}$ years, or 780 million times the currently estimated age of the universe (13.8 billion years).
Could A Computer Decide $L$? (3)

As an aside, the question if we can write a *program* to decide $L$ is more subtle:
Could A Computer Decide $L$? (3)

As an aside, the question if we can write a

**program** to decide $L$ is more subtle:

- A *programming language specification*
  can conceivably be very abstract and not mention any specific limits on sizes.
Could A Computer Decide $L$? (3)

As an aside, the question if we can write a program to decide $L$ is more subtle:

- A *programming language specification* can conceivably be very abstract and not mention any specific limits on sizes.

- A correct program can then be expressed in that it in theory could count arbitrarily far.
As an aside, the question if we can write a program to decide $L$ is more subtle:

- A *programming language specification* can conceivably be very abstract and not mention any specific limits on sizes.
- A correct program can then be expressed in that it in theory could count arbitrarily far.
- However, when this program is run we would sooner or later hit some limitation either due to the *implementation* of the language or due to the hardware we are running it on.
Today’s Lecture

Bottom line: In practice, we can, up to a point, treat a computer as if it has infinite memory if it suits us.
Today’s Lecture

Bottom line: In practice, we can, up to a point, treat a computer as if it has infinite memory if it suits us.

But how can we tell if a language is regular or not (i.e., if a DFA suffices to recognise it) or if we need a more general machine?
Today’s Lecture

Bottom line: In practice, we can, up to a point, treat a computer as if it has infinite memory if it suits us.

But how can we tell if a language is regular or not (i.e., if a DFA suffices to recognise it) or if we need a more general machine?

That’s the topic of today’s lecture.
Today’s Lecture

Bottom line: In practice, we can, up to a point, treat a computer as if it has infinite memory if it suits us.

But how can we tell if a language is regular or not (i.e., if a DFA suffices to recognise it) or if we need a more general machine?

That’s the topic of today’s lecture.

Key observation: Because a Finite Automaton has limited memory, any sufficiently long word in the language must contain repetitive patterns.