The Diagonalization Method

Purushothama B R

Department of Computer Science and Engineering
National Institute of Technology Goa ,INDIA

December 7, 2020

I want to Begin With... Michael Jordan's Quote

I've missed more than 9000 shots in my career. I've lost almost 300 games. 26 times, I've been trusted to take the game winning shot and missed. I've failed over and over and over again in my life. And that is why I succeed.

Fundamentals

- The proof of the undecidability of A_{TM} uses a technique called **diagonalization.**

Fundamentals

- The proof of the undecidability of A_{TM} uses a technique called **diagonalization.**
- It is discovered by mathematician Georg Cantor in 1873.

Fundamentals

- The proof of the undecidability of A_{TM} uses a technique called **diagonalization.**
- It is discovered by mathematician Georg Cantor in 1873.
- Cantor was concerned with

Problem

How to measure the sizes of the Infinite sets?

- The proof of the undecidability of A_{TM} uses a technique called **diagonalization.**
- It is discovered by mathematician Georg Cantor in 1873.
- Cantor was concerned with

Problem

How to measure the sizes of the Infinite sets?

- If we have two infinite sets, how can we tell whether one is larger than the other or

- The proof of the undecidability of A_{TM} uses a technique called **diagonalization.**
- It is discovered by mathematician Georg Cantor in 1873.
- Cantor was concerned with

Problem

How to measure the sizes of the Infinite sets?

- If we have two infinite sets, how can we tell whether one is larger than the other or whether they are of the same size?

Is the problem easy for Finite Sets?

Is the problem easy for Finite Sets?

- Answering the question is easy!!!

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set,

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set, and the resulting number is its size.

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set, and the resulting number is its size.
- However,if we try to count the elements of an infinite set,

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set, and the resulting number is its size.
- However,if we try to count the elements of an infinite set,

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set, and the resulting number is its size.
- However,if we try to count the elements of an infinite set,
  - We will never finish!!!

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set, and the resulting number is its size.
- However,if we try to count the elements of an infinite set,
  - We will never finish!!!
- We **can't use the counting method**

Is the problem easy for Finite Sets?

- Answering the question is easy!!!
- Simply count the elements in a finite set, and the resulting number is its size.
- However,if we try to count the elements of an infinite set,
  - We will never finish!!!
- We **can't use the counting method** to determine the relative sizes of infinite sets.

Example

Example

- Take the set of even integers and the set of all strings over $\{0, 1\}$.

Example

- Take the set of even integers and the set of all strings over $\{0, 1\}$.
- Both sets are infinite and thus larger than any finite set.

Example

- Take the set of even integers and the set of all strings over $\{0, 1\}$.
- Both sets are infinite and thus larger than any finite set.
- **But is one of the two larger than the other?**

Example

- Take the set of even integers and the set of all strings over $\{0, 1\}$.
- Both sets are infinite and thus larger than any finite set.
- **But is one of the two larger than the other?**
- **How can we compare their relative size?**

Cantor Did It!!!

Cantor Did It!!!

- Cantor proposed a rather **nice solution to this problem.**

His Observation

Cantor Did It!!!

- Cantor proposed a rather **nice solution to this problem.**

His Observation

- Two finite sets have the same size

Cantor Did It!!!

- Cantor proposed a rather **nice solution to this problem.**

His Observation

- Two finite sets have the same size
  - If the elements of one set can be paired with the elements of the other set.

Cantor Did It!!!

- Cantor proposed a rather **nice solution to this problem.**

His Observation

- Two finite sets have the same size
  - If the elements of one set can be paired with the elements of the other set.

- This method compares the sizes without resorting to counting.

Cantor Did It!!!

- Cantor proposed a rather **nice solution to this problem.**

His Observation

- Two finite sets have the same size
  - If the elements of one set can be paired with the elements of the other set.

- This method compares the sizes without resorting to counting.
- We can extend this idea to infinite sets.

Let us put it precisely!!!

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is,

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.
- Say **A and B are the same size if there is a one-to-one, onto function** $f : A \to B$.

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.
- Say **A and B are the same size if there is a one-to-one, onto function** $f : A \to B$.
- A function that is both one-to-one and onto is called a **correspondence.**

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.
- Say **A and B are the same size if there is a one-to-one, onto function** $f : A \to B$.
- A function that is both one-to-one and onto is called a **correspondence.**
- In a correspondence, every element of A maps to a unique element of B and

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.
- Say **A and B are the same size if there is a one-to-one, onto function** $f : A \to B$.
- A function that is both one-to-one and onto is called a **correspondence.**
- In a correspondence, every element of A maps to a unique element of B and each element of B has a unique element of A mapping to it.

Let us put it precisely!!!

- Assume that we have sets A and B and a function f from A to B.
- **f is one-to-one** if it never maps two different elements to the same placethat is, if $f(a) = f(b)$ whenever $a = b$.
- **f is onto** if it hits every element of B that is, if for every $b \in B$ there is an $a \in A$ such that $f(a) = b$.
- Say **A and B are the same size if there is a one-to-one, onto function** $f : A \to B$.
- A function that is both one-to-one and onto is called a **correspondence.**
- In a correspondence, every element of A maps to a unique element of B and each element of B has a unique element of A mapping to it.

Simple Way of Pairing

A correspondence is simply a way of pairing the elements of A with the elements of B.

Example

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$
- Let **E** be the set of even numbers $\{2, 4, 6, \ldots\}$

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$
- Let **E** be the set of even numbers $\{2, 4, 6, \ldots\}$
- Using Cantors definition of size,

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$
- Let **E** be the set of even numbers $\{2, 4, 6, \ldots\}$
- Using Cantors definition of size, we can see $\mathbb{N}$ and **E** have the same size.

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$
- Let **E** be the set of even numbers $\{2, 4, 6, \ldots\}$
- Using Cantors definition of size, we can see $\mathbb{N}$ and **E** have the same size.
- The correspondence f mapping $\mathbb{N}$ and **E**

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$
- Let **E** be the set of even numbers $\{2, 4, 6, \ldots\}$
- Using Cantors definition of size, we can see $\mathbb{N}$ and **E** have the same size.
- The correspondence f mapping $\mathbb{N}$ and **E** is simply $f(n) = 2n$.

Example

- Let $\mathbb{N}$ be the set of natural numbers $\{1, 2, 3, \ldots\}$
- Let **E** be the set of even numbers $\{2, 4, 6, \ldots\}$
- Using Cantors definition of size, we can see $\mathbb{N}$ and **E** have the same size.
- The correspondence f mapping $\mathbb{N}$ and **E** is simply $f(n) = 2n$.

n	$f(n)$
1	2
2	4
3	6
$\vdots$	$\vdots$

Bizarre!!!—No.

Bizarre!!!—No.

- Of course, this example seems bizarre.

Bizarre!!!—No.

- Of course, this example seems bizarre.
- **E** seems smaller than $\mathbb{N}$ because **E** is a proper subset of $\mathbb{N}$

Bizarre!!!—No.

- Of course, this example seems bizarre.
- **E** seems smaller than $\mathbb{N}$ because **E** is a proper subset of $\mathbb{N}$
- But pairing each member of $\mathbb{N}$ with its own member of **E** is possible.

Bizarre!!!—No.

- Of course, this example seems bizarre.
- **E** seems smaller than $\mathbb{N}$ because **E** is a proper subset of $\mathbb{N}$
- But pairing each member of $\mathbb{N}$ with its own member of **E** is possible.
- So we declare these two sets to be the same size.

Countable Sets

Countable Sets

Countable

A set A is countable **if either it is finite** or **it has the same size as N .**

Stranger Example

Stranger Example

- Let $Q = \{\frac{m}{n} | m, n \in \mathbb{N}\}$

Stranger Example

- Let $Q = \{\frac{m}{n} | m, n \in \mathbb{N}\}$
- Q is a set of positive rational numbers.

Stranger Example

- Let $Q = \{\frac{m}{n} | m, n \in \mathbb{N}\}$
- Q is a set of positive rational numbers.
- Q seems to be much larger than $\mathbb{N}$.

Stranger Example

- Let $Q = \{\frac{m}{n} | m, n \in \mathbb{N}\}$
- Q is a set of positive rational numbers.
- Q seems to be much larger than $\mathbb{N}$.
- Yet according to our definition,

Stranger Example

- Let $Q = \{\frac{m}{n} | m, n \in \mathbb{N}\}$
- Q is a set of positive rational numbers.
- Q seems to be much larger than $\mathbb{N}$.
- Yet according to our definition, these two sets are the same size.

Stranger Example

- Let $Q = \{\frac{m}{n} | m, n \in \mathbb{N}\}$
- Q is a set of positive rational numbers.
- Q seems to be much larger than $\mathbb{N}$.
- Yet according to our definition, these two sets are the same size.
- Let us try to give **correspondence** with $\mathbb{N}$ to show that Q is countable.

Easy way to do it!!!

Easy way to do it!!!

- List all the elements of Q.

Easy way to do it!!!

- List all the elements of Q.
- Then we pair the first element on the list with the number 1 from $\mathbb{N}$,

Easy way to do it!!!

- List all the elements of Q.
- Then we pair the first element on the list with the number 1 from $\mathbb{N}$,
- The second element on the list with the number 2 from $\mathbb{N}$, and so on.

Easy way to do it!!!

- List all the elements of Q.
- Then we pair the first element on the list with the number 1 from $\mathbb{N}$,
- The second element on the list with the number 2 from $\mathbb{N}$, and so on.
- We must ensure that every member of Q appears only once on the list.

Easy way to do it!!!

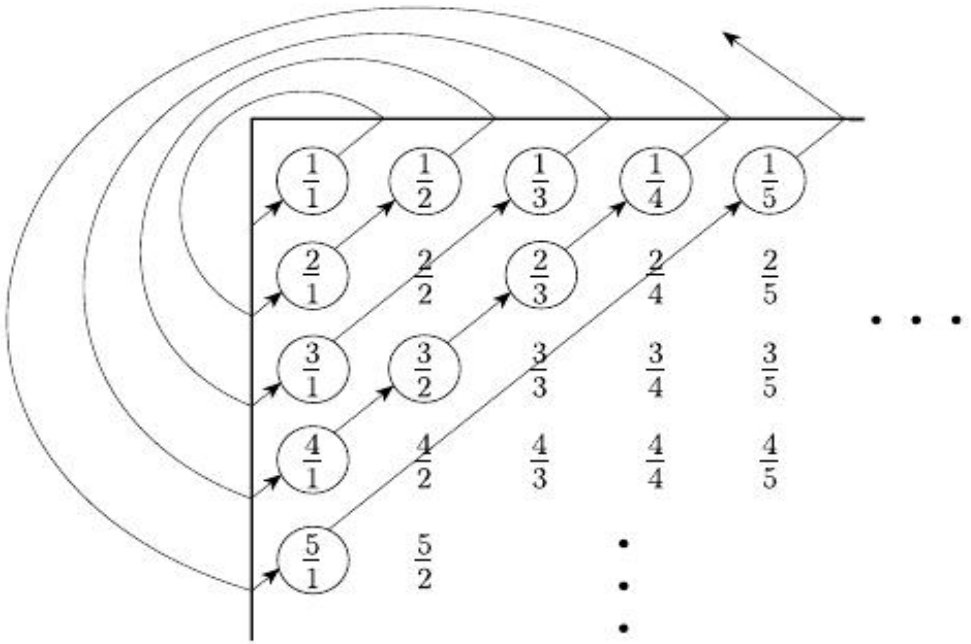
- List all the elements of Q.
- Then we pair the first element on the list with the number 1 from $\mathbb{N}$,
- The second element on the list with the number 2 from $\mathbb{N}$, and so on.
- We must ensure that every member of Q appears only once on the list.
- To get this list,

Easy way to do it!!!

- List all the elements of Q.
- Then we pair the first element on the list with the number 1 from $\mathbb{N}$,
- The second element on the list with the number 2 from $\mathbb{N}$, and so on.
- We must ensure that every member of Q appears only once on the list.
- To get this list, we make an infinite matrix containing all the positive rational numbers.

A correspondence of N and Q

A correspondence of N and Q



Turn Matrix into List: One of the Way

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$
- In the third diagonal, a complication arises!!!

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$
- In the third diagonal, a complication arises!!!
- It contains $\frac{3}{1}$, $\frac{2}{2}$ and $\frac{1}{3}$.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$
- In the third diagonal, a complication arises!!!
- It contains $\frac{3}{1}$, $\frac{2}{2}$ and $\frac{1}{3}$.
- If we simply added these to the list, we would

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$
- In the third diagonal, a complication arises!!!
- It contains $\frac{3}{1}$, $\frac{2}{2}$ and $\frac{1}{3}$.
- If we simply added these to the list, we would we would repeat $\frac{1}{1} = \frac{2}{2}$.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$
- In the third diagonal, a complication arises!!!
- It contains $\frac{3}{1}$, $\frac{2}{2}$ and $\frac{1}{3}$.
- If we simply added these to the list, we would we would repeat $\frac{1}{1} = \frac{2}{2}$.
- Skip an element when it would cause repetition.

Turn Matrix into List: One of the Way

- Begin the list with all the elements in the first row.
- That isnt a good approach because the first row is infinite,
- So the list would never get to the second row.
- Instead we list the **elements on the diagonals,**
- which are superimposed on the diagram, starting from the corner.
- The first diagonal contains the single element $\frac{1}{1}$.
- And the second diagonal contains the two elements $\frac{2}{1}$ and $\frac{1}{2}$
- In the third diagonal, a complication arises!!!
- It contains $\frac{3}{1}$, $\frac{2}{2}$ and $\frac{1}{3}$.
- If we simply added these to the list, we would we would repeat $\frac{1}{1} = \frac{2}{2}$.
- Skip an element when it would cause repetition.
- Continuing in this way we obtain list of all elements of Q.

Oh... I can do it for any sets....

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,
- You might think that any two infinite sets can be shown to have the same size.

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,
- You might think that any two infinite sets can be shown to have the same size.
- After all, you need only demonstrate a correspondence,

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,
- You might think that any two infinite sets can be shown to have the same size.
- After all, you need only demonstrate a correspondence,
- This example shows that surprising correspondences do exist.

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,
- You might think that any two infinite sets can be shown to have the same size.
- After all, you need only demonstrate a correspondence,
- This example shows that surprising correspondences do exist.
- **However, for some infinite sets, no correspondence with N exists.**

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,
- You might think that any two infinite sets can be shown to have the same size.
- After all, you need only demonstrate a correspondence,
- This example shows that surprising correspondences do exist.
- **However, for some infinite sets, no correspondence with N exists.**
- These sets are simply too big.

Oh... I can do it for any sets....

- After seeing the correspondence of N and Q,
- You might think that any two infinite sets can be shown to have the same size.
- After all, you need only demonstrate a correspondence,
- This example shows that surprising correspondences do exist.
- **However, for some infinite sets, no correspondence with N exists.**
- These sets are simply too big.
- Such sets are called **uncountable.**

Uncountable...

Uncountable...

- The **set of real numbers** is an example of an uncountable set.

Uncountable...

- The **set of real numbers** is an example of an uncountable set.
- A real number is one that has a decimal representation.

Uncountable...

- The **set of real numbers** is an example of an uncountable set.
- A real number is one that has a decimal representation.
- The numbers $\pi = 3.1415926...$ and

Uncountable...

- The **set of real numbers** is an example of an uncountable set.
- A real number is one that has a decimal representation.
- The numbers $\pi = 3.1415926...$ and $\sqrt{2} = 1.4142135...$ are examples of real numbers.

Uncountable...

- The **set of real numbers** is an example of an uncountable set.
- A real number is one that has a decimal representation.
- The numbers $\pi = 3.1415926...$ and $\sqrt{2} = 1.4142135...$ are examples of real numbers.
- Let $\mathbb{R}$ be the set of real numbers.

Uncountable...

- The **set of real numbers** is an example of an uncountable set.
- A real number is one that has a decimal representation.
- The numbers $\pi = 3.1415926...$ and $\sqrt{2} = 1.4142135...$ are examples of real numbers.
- Let $\mathbb{R}$ be the set of real numbers.
- **Cantor proved that $\mathbb{R}$ is uncountable.**

Uncountable...

- The **set of real numbers** is an example of an uncountable set.
- A real number is one that has a decimal representation.
- The numbers $\pi = 3.1415926...$ and $\sqrt{2} = 1.4142135...$ are examples of real numbers.
- Let $\mathbb{R}$ be the set of real numbers.
- **Cantor proved that $\mathbb{R}$ is uncountable.**
- In doing so, **he introduced the diagonalization method.**

Theorem by Cantor

Theorem by Cantor

Theorem 4.17

$\mathbb{R}$ is uncountable.

Proof

Proof

- In order to show that R is uncountable,

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.
- Our job is to show that f fails to work as it should.

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.
- Our job is to show that f fails to work as it should.
- For it to be a correspondence,

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.
- Our job is to show that f fails to work as it should.
- For it to be a correspondence, f must pair all the members of N

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.
- Our job is to show that f fails to work as it should.
- For it to be a correspondence, f must pair all the members of N with all the members of R.

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.
- Our job is to show that f fails to work as it should.
- For it to be a correspondence, f must pair all the members of N with all the members of R.
- But we will find an x in R that is not paired with in N

Proof

- In order to show that R is uncountable,
  - Cantor showed that no correspondence exists between $\mathbb{N}$ and $\mathbb{R}$.
- **Proof is by contradiction.**
- Supose that there is a correspondence f existed between N and R.
- Our job is to show that f fails to work as it should.
- For it to be a correspondence, f must pair all the members of N with all the members of R.
- But we will find an x in R that is not paired with in N
- Which will be our contradiction.

How to find that x?: Idea

How to find that x?: Idea

- We find this x is by **actually constructing it.**

How to find that x?: Idea

- We find this x is by **actually constructing it.**
- We choose each digit of x to make x different from

How to find that x?: Idea

- We find this x is by **actually constructing it.**
- We choose each digit of x to make x different from one of the real numbers that is paired with an element of N.

How to find that x?: Idea

- We find this x is by **actually constructing it.**
- We choose each digit of x to make x different from one of the real numbers that is paired with an element of N.
- In the end, we are sure that x is different from any real number that is paired.

Illustration with an Example

Illustration with an Example

- Suppose that the correspondence f exists.

Illustration with an Example

- Suppose that the correspondence f exists.
- Let $f(1) = 3.14159...$, $f(2) = 55.55555...$, $f(3) = ...$, and so on.

Illustration with an Example

- Suppose that the correspondence f exists.
- Let $f(1) = 3.14159...$, $f(2) = 55.55555...$, $f(3) = ...$, and so on.
- Then f pairs the number 1 with 3.14159... ,

Illustration with an Example

- Suppose that the correspondence f exists.
- Let $f(1) = 3.14159...$, $f(2) = 55.55555...$, $f(3) = ...$, and so on.
- Then f pairs the number 1 with 3.14159... ,the number 2 with 55.55555..., and so on.

Illustration with an Example

- Suppose that the correspondence f exists.
- Let $f(1) = 3.14159...$, $f(2) = 55.55555...$, $f(3) = ...$, and so on.
- Then f pairs the number 1 with 3.14159... ,the number 2 with 55.55555..., and so on.
- Few values of a hypothetical correspondence f between N and R.

Illustration with an Example

- Suppose that the correspondence f exists.
- Let $f(1) = 3.14159...$, $f(2) = 55.55555...$, $f(3) = ...$, and so on.
- Then f pairs the number 1 with 3.14159... ,the number 2 with 55.55555..., and so on.
- Few values of a hypothetical correspondence f between N and R.

n	$f(n)$
1	$3.14159...$
2	$55.55555...$
3	$0.12345...$
4	$0.50000...$
$\vdots$	$\vdots$

Construction

Construction

- We construct the desired x by giving its decimal representation.

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$
  - Arbitrarily, we let it be 4.
- To ensure that $x \neq f(2)$,

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$
  - Arbitrarily, we let it be 4.
- To ensure that $x \neq f(2)$,
  - we let the second digit of x be anything

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$
  - Arbitrarily, we let it be 4.
- To ensure that $x \neq f(2)$,
  - we let the second digit of x be anything different from the second fractional digit 5 of $f(2) = 55.555555....$

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$
  - Arbitrarily, we let it be 4.
- To ensure that $x \neq f(2)$,
  - we let the second digit of x be anything different from the second fractional digit 5 of $f(2) = 55.555555....$
  - Arbitrarily, we let it be 6.

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$
  - Arbitrarily, we let it be 4.
- To ensure that $x \neq f(2)$,
  - we let the second digit of x be anything different from the second fractional digit 5 of $f(2) = 55.555555....$
  - Arbitrarily, we let it be 6.
- The third fractional digit of $f(3) = 0.12345...is3$,

Construction

- We construct the desired x by giving its decimal representation.
- It is a number between 0 and 1
- So all its significant digits are fractional digits following the decimal point.
- Our objective is to ensure that $x \neq f(n)$ for any n.
- To ensure that $x \neq f(1)$,
  - We let the first digit of x be anything different from the first fractional digit 1 of $f(1) = 3.14159....$
  - Arbitrarily, we let it be 4.
- To ensure that $x \neq f(2)$,
  - we let the second digit of x be anything different from the second fractional digit 5 of $f(2) = 55.555555....$
  - Arbitrarily, we let it be 6.
- The third fractional digit of $f(3) = 0.12345...is3$,
  - So we let x be anything different, say, 4.

Contd..

Contd..

- Continuing in this way **down the diagonal of the table for** f **,**

Contd..

- Continuing in this way **down the diagonal of the table for** f **,**
- We obtain all the digits of x

Contd..

- Continuing in this way **down the diagonal of the table for** f **,**
- We obtain all the digits of x

n	$f(n)$
1	$3.\underline{1}4159\ldots$
2	$55.5\underline{5}555\ldots$
3	$0.12\underline{3}45\ldots$
4	$0.500\underline{0}0\ldots$
$\vdots$	$\vdots$

$$x = 0.4641\ldots$$

Concluding Remarks

Concluding Remarks

- We know that x is not $f(n)$ for any n

Concluding Remarks

- We know that x is not $f(n)$ for any n
  - Because it differs from $f(n)$ in the n^{th} fractional digit.

Concluding Remarks

- We know that x is not $f(n)$ for any n
  - Because it differs from $f(n)$ in the n^{th} fractional digit.
- A slight problem arises because certain numbers, such as 0.1999 . . . and 0.2000 . . . , are equal

Concluding Remarks

- We know that x is not $f(n)$ for any n
  - Because it differs from $f(n)$ in the n^{th} fractional digit.
- A slight problem arises because certain numbers, such as $0.1999\ldots$ and $0.2000\ldots$, are equal even though their decimal representations are different.

Concluding Remarks

- We know that x is not $f(n)$ for any n
  - Because it differs from $f(n)$ in the n^{th} fractional digit.
- A slight problem arises because certain numbers, such as $0.1999\ldots$ and $0.2000\ldots$, are equal even though their decimal representations are different.
- We avoid this problem by never selecting the digits 0 or 9 when we construct x.

Corollary

Corollary

Some languages are not Turing-recognizable.

Contd..

Contd..

- The set of all Turing machines is countable .

Contd..

- The set of all Turing machines is countable .
  - Because each Turing machine M has an encoding into a string $< M >$.

Contd..

- The set of all Turing machines is countable .
  - Because each Turing machine M has an encoding into a string $< M >$.
- Omit those strings that are not

Contd..

- The set of all Turing machines is countable .
  - Because each Turing machine M has an encoding into a string $< M >$.
- Omit those strings that are not legal encodings of Turing machines, we can obtain a list of all Turing machines.

Contd...

Contd...

- **Goal:** To show that the set of all languages is uncountable.

Contd...

- **Goal:** To show that the set of all languages is uncountable.
  - Observe that the set of all infinite binary sequences is uncountable.

Contd...

- **Goal:** To show that the set of all languages is uncountable.
  - Observe that the set of all infinite binary sequences is uncountable.
  - An infinite binary sequence is an unending sequence of $0's$ and $1's$.

Contd...

- **Goal:** To show that the set of all languages is uncountable.
  - Observe that the set of all infinite binary sequences is uncountable.
  - An infinite binary sequence is an unending sequence of $0's$ and $1's$.
- Let **B** be the set of all infinite binary sequences.

Claim

B is uncountable.

Contd...

- **Goal:** To show that the set of all languages is uncountable.
  - Observe that the set of all infinite binary sequences is uncountable.
  - An infinite binary sequence is an unending sequence of $0's$ and $1's$.
- Let **B** be the set of all infinite binary sequences.

Claim

B is uncountable.

- **This can be proved by giving similar proof as we had discussed before.**

Contd..

Contd..

- Let L be the set of all languages over alphabet Σ.

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.
- Thus is, it means that the two sets are the same size.

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.
- Thus is, it means that the two sets are the same size.
- Let $\Sigma^* = \{s_1, s_2, s_3, \ldots\}$

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.
- Thus is, it means that the two sets are the same size.
- Let $\Sigma^* = \{s_1, s_2, s_3, \ldots\}$
- Each language $A \in L$ has a unique sequence in B.

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.
- Thus is, it means that the two sets are the same size.
- Let $\Sigma^* = \{s_1, s_2, s_3, \ldots\}$
- Each language $A \in L$ has a unique sequence in B.
- The i^{th} bit of that sequence is a 1 if $s_i \in A$

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.
- Thus is, it means that the two sets are the same size.
- Let $\Sigma^* = \{s_1, s_2, s_3, \ldots\}$
- Each language $A \in L$ has a unique sequence in B.
- The i^{th} bit of that sequence is a 1 if $s_i \in A$ and is a 0 if $s_i \in A$

Contd..

- Let L be the set of all languages over alphabet Σ.
- I want to show that L **is uncountable**
- I can do that by giving a correspondence with B.
- Thus is, it means that the two sets are the same size.
- Let $\Sigma^* = \{s_1, s_2, s_3, \ldots\}$
- Each language $A \in L$ has a unique sequence in B.
- The i^{th} bit of that sequence is a 1 if $s_i \in A$ and is a 0 if $s_i \in A$
- This is called characteristic sequence of A.

Example

Example

- If A is the language of all strings starting with a 0 over the alphabet $\{0, 1\}$

Example

- If A is the language of all strings starting with a 0 over the alphabet $\{0, 1\}$
- Its characteristic sequence would be

$$\begin{aligned}
\Sigma^* &= \{\ \varepsilon,\ \ 0,\ \ 1,\ \ 00,\ \ 01,\ \ 10,\ \ 11,\ \ 000, 001,\ \cdots\ \}\ ; \\
A &= \{\ \ \ \ 0,\ \ \ \ 00,\ \ 01,\ \ \ \ \ \ \ \ 000, 001,\ \cdots\ \}\ ; \\
\chi_A &= \ \ 0\ \ 1\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ \cdots\ \ .
\end{aligned}$$

Concluding remarks

Concluding remarks

- The function $f : L \to B$,

Concluding remarks

- The function $f : L \to B$,
  - Where $f(A)$ equals the characteristic sequence of A, is one-to-one and onto.

Concluding remarks

- The function $f : L \to B$,
  - Where $f(A)$ equals the characteristic sequence of A, is one-to-one and onto.
- Therefore, as B is uncountable,

Concluding remarks

- The function $f : L \to B$,
  - Where $f(A)$ equals the characteristic sequence of A, is one-to-one and onto.
- Therefore, as B is uncountable, L is uncountable as well.

Concluding remarks

- The function $f : L \rightarrow B$,
  - Where $f(A)$ equals the characteristic sequence of A, is one-to-one and onto.
- Therefore, as B is uncountable, L is uncountable as well.
- We have shown that the set of all languages cannot be put into a correspondence with the set of all Turing machines.

Concluding remarks

- The function $f : L \rightarrow B$,
  - Where $f(A)$ equals the characteristic sequence of A, is one-to-one and onto.
- Therefore, as B is uncountable, L is uncountable as well.
- We have shown that the set of all languages cannot be put into a correspondence with the set of all Turing machines.
- We conclude that some languages are not recognized by any Turing machine.

We are equipped!!!

We are equipped!!!

- Note

$$A_{TM} = \{< M, w > | M \text{ is a TM and M accepts } w\}.$$

We are equipped!!!

- Note

$$A_{TM} = \{< M, w > | M \text{ is a TM and M accepts } w\}.$$

Theorem 4.11

A_{TM} is undecidable.

Focus on Proof Sketch

Focus on Proof Sketch

- We assume that A_{TM} is decidable and

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.
- Suppose H is decider for A_{TM}

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.
- Suppose H is decider for A_{TM}
- On input $< M, w >$, where M is a TM and w is a string

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.
- Suppose H is decider for A_{TM}
- On input $< M, w >$, where M is a TM and w is a string
  - H halts and accepts

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.
- Suppose H is decider for A_{TM}
- On input $< M, w >$, where M is a TM and w is a string
  - H halts and accepts if M accepts w.

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.
- Suppose H is decider for A_{TM}
- On input $< M, w >$, where M is a TM and w is a string
  - H halts and accepts if M accepts w.
  - Furthermore, H halts and rejects M fails to accept w

Focus on Proof Sketch

- We assume that A_{TM} is decidable and obtain a contradiction.
- Suppose H is decider for A_{TM}
- On input $< M, w >$, where M is a TM and w is a string
  - H halts and accepts if M accepts w.
  - Furthermore, H halts and rejects M fails to accept w
- In other words, we assume that H is a TM, where

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

Contd..

Contd..

- Now we construct a new Turing machine **D** with

Contd..

- Now we construct a new Turing machine **D** with H as a subroutine.

Contd..

- Now we construct a new Turing machine **D** with H as a subroutine.
- New TM calls H to determine

Contd..

- Now we construct a new Turing machine **D** with H as a subroutine.
- New TM calls H to determine
  - What M does when the input to M is **its own description M .**

Contd..

- Now we construct a new Turing machine **D** with H as a subroutine.
- New TM calls H to determine
  - What M does when the input to M is **its own description M .**
- Once D has determined this information, it does the opposite.

Contd..

- Now we construct a new Turing machine **D** with H as a subroutine.
- New TM calls H to determine
  - What M does when the input to M is **its own description M .**
- Once D has determined this information, it does the opposite.
- That is, it rejects if M accepts and accepts if M does not accept.

Description of D

D = "On input $\langle M \rangle$, where M is a TM:

1. Run H on input $\langle M, \langle M \rangle \rangle$.
2. Output the opposite of what H outputs. That is, if H accepts, *reject*; and if H rejects, *accept*."

No Confusion

No Confusion

- Notion of running a machine on its own description!

No Confusion

- Notion of running a machine on its own description!
- That is similar to running a program with itself as input,

No Confusion

- Notion of running a machine on its own description!
- That is similar to running a program with itself as input, something that does occasionally occur in practice.

No Confusion

- Notion of running a machine on its own description!
- That is similar to running a program with itself as input, something that does occasionally occur in practice.
- For example, a compiler is a program that translates other programs.

No Confusion

- Notion of running a machine on its own description!
- That is similar to running a program with itself as input, something that does occasionally occur in practice.
- For example, a compiler is a program that translates other programs.
- A compiler for the language Python may itself be written in Python

No Confusion

- Notion of running a machine on its own description!
- That is similar to running a program with itself as input, something that does occasionally occur in practice.
- For example, a compiler is a program that translates other programs.
- A compiler for the language Python may itself be written in Python
- So, running that program on itself would make sense.

Summary

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

Question?

Question?

- **What happens when we run D with its own description D as input?**

Question?

- **What happens when we run D with its own description D as input?**
- We get,

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

Question?

- **What happens when we run D with its own description D as input?**
- We get,

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

- No matter what D does, it is forced to do the opposite.

Question?

- **What happens when we run D with its own description D as input?**
- We get,

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

- No matter what D does, it is forced to do the opposite.
- Which is obviously a contradiction.

Question?

- **What happens when we run D with its own description D as input?**
- We get,

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

- No matter what D does, it is forced to do the opposite.
- Which is obviously a contradiction.
- Thus, neither TM D nor TM H can exist.

Steps of Proof

Steps of Proof

- Assume that a TM H decides A_{TM} .

Steps of Proof

- Assume that a TM H decides A_{TM} .
- Use H to build a TM D that takes an input $< M >$

Steps of Proof

- Assume that a TM H decides A_{TM} .
- Use H to build a TM D that takes an input $< M >$
  - Where **D accepts its input** $< M >$

Steps of Proof

- Assume that a TM H decides A_{TM} .
- Use H to build a TM D that takes an input $< M >$
  - Where **D accepts its input** $< M >$ exactly when **M does not accept** its input $< M >$

Steps of Proof

- Assume that a TM H decides A_{TM} .
- Use H to build a TM D that takes an input $< M >$
  - Where **D accepts its input** $< M >$ exactly when **M does not accept** its input $< M >$
- Finally, run D on itself.

Machine Takes on the Following...

Machine Takes on the Following...

- The machines takes on the following actions.

Machine Takes on the Following...

- The machines takes on the following actions.
  - H accepts $< M, w >$ exactly when M accepts w.

Machine Takes on the Following...

- The machines takes on the following actions.
  - H accepts $< M, w >$ exactly when M accepts w.
  - D rejects $< M >$ exactly when M accepts $< M >$.

Machine Takes on the Following...

- The machines takes on the following actions.
  - H accepts $< M, w >$ exactly when M accepts w.
  - D rejects $< M >$ exactly when M accepts $< M >$.
  - D rejects $< D >$ exactly when D accepts $< D >$.

Where is Diagonalization?

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**
- In these tables,

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**
- In these tables,
  - We list all TM's down the rows, $M_1, M_2, \ldots,$

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**
- In these tables,
  - We list all TM's down the rows, $M_1, M_2, \ldots,$
  - All their descriptions across the columns $< M_1 >, < M_2 >, \ldots,$

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**
- In these tables,
  - We list all TM's down the rows, $M_1, M_2, \ldots,$
  - All their descriptions across the columns $< M_1 >, < M_2 >, \ldots,$
- The entries tell whether the machine in a given row accepts the input in a given column.

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**
- In these tables,
  - We list all TM's down the rows, $M_1, M_2, \ldots,$
  - All their descriptions across the columns $< M_1 >, < M_2 >, \ldots,$
- The entries tell whether the machine in a given row accepts the input in a given column.
- The entry is accept if the machine accepts the input

Where is Diagonalization?

- Where is the diagonalization in the proof of this Theorem.
- Let us see behavior for **TM's H and D.**
- In these tables,
  - We list all TM's down the rows, $M_1, M_2, \ldots,$
  - All their descriptions across the columns $< M_1 >, < M_2 >, \ldots,$
- The entries tell whether the machine in a given row accepts the input in a given column.
- The entry is accept if the machine accepts the input
  - But is blank if it rejects or loops on that input.

Entry i, j is accept if M_i accepts $< M_j >$

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*		*accept*		
M_2	*accept*	*accept*	*accept*	*accept*	
M_3					$\cdots$
M_4	*accept*	*accept*			
$\vdots$		$\vdots$			

Contd..

Contd..

- In the following figure, the entries are the results of running H on inputs corresponding to Previous figure.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$			$\vdots$		

Contd..

- In the following figure, the entries are the results of running H on inputs corresponding to Previous figure.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$		$\vdots$			

- So if M_3 does not accept input M_2 ,

Contd..

- In the following figure, the entries are the results of running H on inputs corresponding to Previous figure.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$		$\vdots$			

- So if M_3 does not accept input M_2 ,
  - The entry for row M_3 and column $< M_2 >$ is reject

Contd..

- In the following figure, the entries are the results of running H on inputs corresponding to Previous figure.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*	
M_4	*accept*	*accept*	*reject*	*reject*	
$\vdots$		$\vdots$			

- So if M_3 does not accept input M_2 ,
  - The entry for row M_3 and column $< M_2 >$ is reject because H rejects input $< M_3, < M_2 >>$.

Add D to the Figure

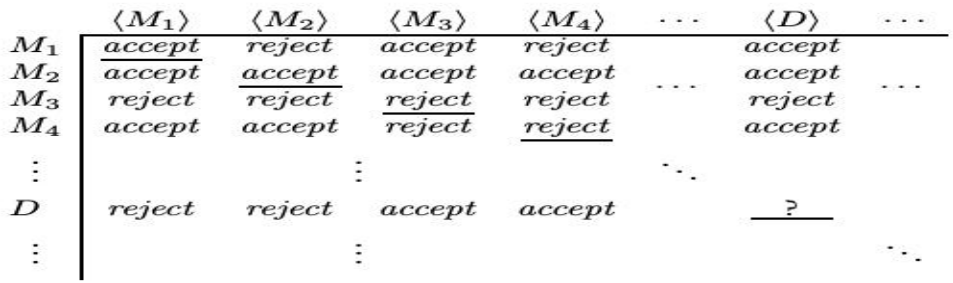
Add D to the Figure

- In the following figure, D is added to previous Figure.

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$		$\vdots$			$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$		$\vdots$					$\ddots$

Add D to the Figure

- In the following figure, D is added to previous Figure.

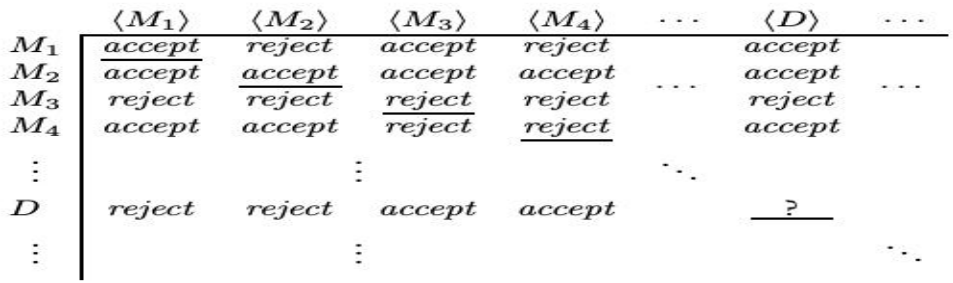


	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$			$\vdots$		$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$			$\vdots$				$\ddots$

- By our assumption, H is a TM and so is D.

Add D to the Figure

- In the following figure, D is added to previous Figure.

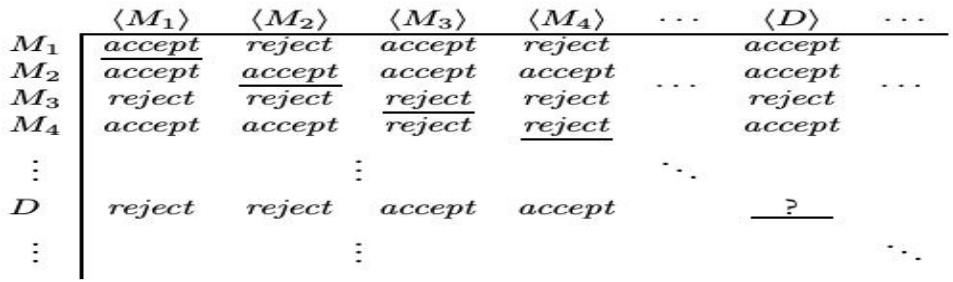


	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$		$\vdots$			$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$		$\vdots$					$\ddots$

- By our assumption, H is a TM and so is D.
- Therefore, it must occur on the list $M_1, M_2, \ldots$ of all TM s.

Add D to the Figure

- In the following figure, D is added to previous Figure.

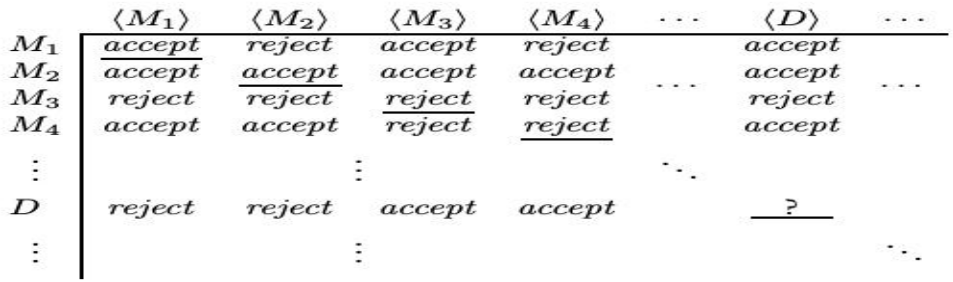


	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$			$\vdots$		$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$			$\vdots$				$\ddots$

- By our assumption, H is a TM and so is D.
- Therefore, it must occur on the list $M_1, M_2, \ldots$ of all TM s.
- Note that D computes the opposite of the diagonal entries.

Add D to the Figure

- In the following figure, D is added to previous Figure.

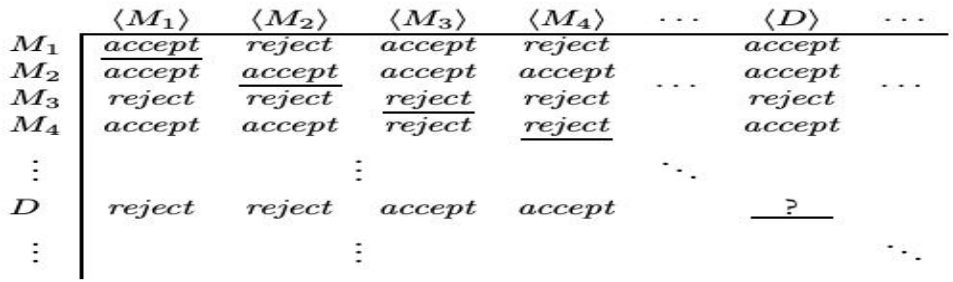


	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$			$\vdots$		$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$			$\vdots$				$\ddots$

- By our assumption, H is a TM and so is D.
- Therefore, it must occur on the list $M_1, M_2, \ldots$ of all TM s.
- Note that D computes the opposite of the diagonal entries.
- The contradiction occurs occurs at the point of the question mark

Add D to the Figure

- In the following figure, D is added to previous Figure.



	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accept*	*reject*	*accept*	*reject*		*accept*	
M_2	*accept*	*accept*	*accept*	*accept*	$\cdots$	*accept*	$\cdots$
M_3	*reject*	*reject*	*reject*	*reject*		*reject*	
M_4	*accept*	*accept*	*reject*	*reject*		*accept*	
$\vdots$			$\vdots$		$\ddots$		
D	*reject*	*reject*	*accept*	*accept*		?	
$\vdots$			$\vdots$				$\ddots$

- By our assumption, H is a TM and so is D.
- Therefore, it must occur on the list $M_1, M_2, \ldots$ of all TM s.
- Note that D computes the opposite of the diagonal entries.
- The contradiction occurs occurs at the point of the question mark where the entry must be the opposite of itself.

A TURING-UNRECOGNIZABLE LANGUAGE

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**,

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**, the language is decidable.

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**, the language is decidable.
- Hence for any undecidable language,

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**, the language is decidable.
- Hence for any undecidable language, either it or

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**, the language is decidable.
- Hence for any undecidable language, either it or its complement is not Turing -recognizable.

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**, the language is decidable.
- Hence for any undecidable language, either it or its complement is not Turing -recognizable.
- We say that a language is **co-Turing-recognizable language,**

A TURING-UNRECOGNIZABLE LANGUAGE

- Now we exhibit a language that isnt even Turing-recognizable.
- A_{TM} will not suffice for this purpose
- Since we showed that A_{TM} is Turing-recognizable.
- Let us focus on a Theorem which says the following
  - If both a **language and its complement are Turing-recognizable**, the language is decidable.
- Hence for any undecidable language, either it or its complement is not Turing -recognizable.
- We say that a language is **co-Turing-recognizable language,**
  - If it is the complement of a Turing-recognizable language.

Theorem: Condition for Decidability

Theorem 4.22

A language is decidable **iff** it is Turing-recognizable and co-Turing-recognizable.

Proof

Proof

- We have two directions to prove.

Proof

- We have two directions to prove.
- First, if A is decidable, It's easy to see that

Proof

- We have two directions to prove.
- First, if A is decidable, It's easy to see that
  - Both A and its complement A are Turing-recognizable.

Proof

- We have two directions to prove.
- First, if A is decidable, It's easy to see that
  - Both A and its complement A are Turing-recognizable.
  - Any decidable language is Turing-recognizable, and

Proof

- We have two directions to prove.
- First, if A is decidable, It's easy to see that
  - Both A and its complement A are Turing-recognizable.
  - Any decidable language is Turing-recognizable, and
  - The complement of a decidable language also is decidable.

Other Direction

Other Direction

- If both A and $\overline{A}$ are Turing-recognizable,

Other Direction

- If both A and $\overline{A}$ are Turing-recognizable,
  - let M_1 be the recognizer for A and

Other Direction

- If both A and $\overline{A}$ are Turing-recognizable,
  - let M_1 be the recognizer for A and
  - M_2 be the recognizer for $\overline{A}$.

Other Direction

- If both A and $\overline{A}$ are Turing-recognizable,
  - let M_1 be the recognizer for A and
  - M_2 be the recognizer for $\overline{A}$.
- The following Turing machine M is a decider for A.

M = "On input w:

1. Run both M_1 and M_2 on input w in parallel.
2. If M_1 accepts, *accept*; if M_2 accepts, *reject*."

Contd..

Contd..

- Running the two machines in parallel means that M has two tapes.

Contd..

- Running the two machines in parallel means that M has two tapes.
- One for simulating M_1 and the other for simulating M_2 .

Contd..

- Running the two machines in parallel means that M has two tapes.
- One for simulating M_1 and the other for simulating M_2 .
- In this case, M takes turns simulating one step of each machine,

Contd..

- Running the two machines in parallel means that M has two tapes.
- One for simulating M_1 and the other for simulating M_2 .
- In this case, M takes turns simulating one step of each machine, which continues until one of them accepts.

Show that M decides A

Show that M decides A

- Every string w is either in A or $\overline{A}$.

Show that M decides A

- Every string w is either in A or $\overline{A}$.
- Therefore, either M_1 or M_2 must accept w.

Show that M decides A

- Every string w is either in A or $\overline{A}$.
- Therefore, either M_1 or M_2 must accept w.
- Because M halts whenever M_1 or M_2 accepts,

Show that M decides A

- Every string w is either in A or $\overline{A}$.
- Therefore, either M_1 or M_2 must accept w.
- Because M halts whenever M_1 or M_2 accepts,
- M always halts and so it is a decider.

Show that M decides A

- Every string w is either in A or $\overline{A}$.
- Therefore, either M_1 or M_2 must accept w.
- Because M halts whenever M_1 or M_2 accepts,
- M always halts and so it is a decider.
- Furthermore, it accepts all strings in A and rejects all strings not in A.

Show that M decides A

- Every string w is either in A or $\overline{A}$.
- Therefore, either M_1 or M_2 must accept w.
- Because M halts whenever M_1 or M_2 accepts,
- M always halts and so it is a decider.
- Furthermore, it accepts all strings in A and rejects all strings not in A.
- So M is a decider for A, and thus A is decidable.

Not a Turing Recognizable Language

Corollary

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

Not a Turing Recognizable Language

Corollary

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

- We know that A_TM is Turing-recognizable.

Not a Turing Recognizable Language

Corollary

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

- We know that A_TM is Turing-recognizable.
- If A_TM also were Turing-recognizable,

Not a Turing Recognizable Language

Corollary

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

- We know that A_TM is Turing-recognizable.
- If A_TM also were Turing-recognizable, A_{TM} would be decidable.

Not a Turing Recognizable Language

Corollary

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

- We know that $A_T M$ is Turing-recognizable.
- If $A_T M$ also were Turing-recognizable, A_{TM} would be decidable.
- We have proved thatthat A_{TM} is not decidable.

Not a Turing Recognizable Language

Corollary

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

- We know that A_TM is Turing-recognizable.
- If A_TM also were Turing-recognizable, A_{TM} would be decidable.
- We have proved thatthat A_{TM} is not decidable.
- So, $\overline{A_{TM}}$ must not be Turing-recognizable.

THANK YOU