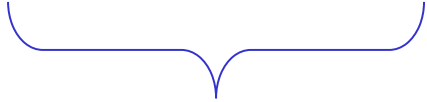


# A Universal Turing Machine

# A limitation of Turing Machines:

Turing Machines are "hardwired"



they execute  
only one program

Real Computers are re-programmable

Solution: Universal Turing Machine

Attributes:

- Reprogrammable machine
- Simulates any other Turing Machine

Universal Turing Machine  
simulates any Turing Machine  $M$

Input of Universal Turing Machine:

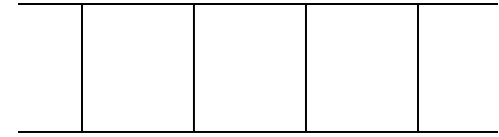
Description of transitions of  $M$

Input string of  $M$

Three tapes

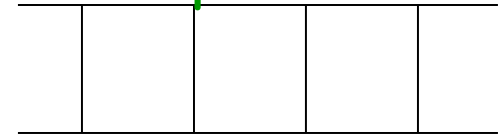


Tape 1



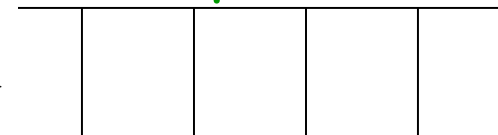
Description of  $M$

Tape 2



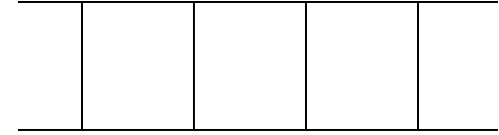
Tape Contents of  $M$

Tape 3



State of  $M$

Tape 1



Description of  $M$

We describe Turing machine  $M$   
as a string of symbols:

We encode  $M$  as a string of symbols

# Alphabet Encoding

Symbols:

*a*

*b*

*c*

*d*

...



Encoding:

1

11

111

1111

# State Encoding

States:  $q_1$        $q_2$        $q_3$        $q_4$        $\dots$



Encoding: 1      11      111      1111

# Head Move Encoding

Move:  $L$        $R$



Encoding: 1      11



# Transition Encoding

Transition:  $\delta(q_1, a) = (q_2, b, L)$

Encoding:

1 0 1 0 1 1 0 1 1 0 1

separator

# Turing Machine Encoding

Transitions:

$$\delta(q_1, a) = (q_2, b, L)$$

$$\delta(q_2, b) = (q_3, c, R)$$

Encoding:

1 0 1 0 1 1 0 1 1 0 1 0 0 1 1 0 1 1 1 0 1 1 1 0 1 1

separator

# Tape 1 contents of Universal Turing Machine:

binary encoding  
of the simulated machine  $M$

Tape 1

---

1 0 1 0 11 0 11 0 10011 0 1 10 111 0 111 0 1100...

---



A Turing Machine is described  
with a binary string of 0's and 1's

Therefore:

The set of Turing machines  
forms a language:

each string of this language is  
the binary encoding of a Turing Machine

# Language of Turing Machines

$L = \{$  1010110101, (Turing Machine 1)  
101011101011, (Turing Machine 2)  
1110101111010111, .....  
..... }

# Countable Sets

Infinite sets are either:

Countable

or

Uncountable

## Countable set:

There is a one to one correspondence (injection)  
of  
elements of the set  
to  
Positive integers (1,2,3,...)

Every element of the set is mapped to a positive number  
such that no two elements are mapped to same number



**Example:** The set of even integers  
is countable

Even integers:  
(positive) 0, 2, 4, 6, ...

Correspondence:

Positive integers: 1, 2, 3, 4, ...

$2n$  corresponds to  $n + 1$

**Example:** The set of rational numbers  
is countable

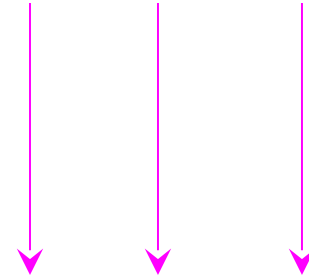
Rational numbers:  $\frac{1}{2}, \frac{3}{4}, \frac{7}{8}, \dots$

# Naive Approach

Rational numbers:

Nominator 1  
 $\frac{1}{1}, \frac{1}{2}, \frac{1}{3}, \dots$

Correspondence:



Positive integers:

1, 2, 3, ...

Doesn't work:

we will never count  
numbers with nominator 2:

$\frac{2}{1}, \frac{2}{2}, \frac{2}{3}, \dots$

# Better Approach

$$\frac{1}{1} \quad \frac{1}{2} \quad \frac{1}{3} \quad \frac{1}{4} \quad \dots$$

$$\frac{2}{1} \quad \frac{2}{2} \quad \frac{2}{3} \quad \dots$$

$$\frac{3}{1} \quad \frac{3}{2} \quad \dots$$

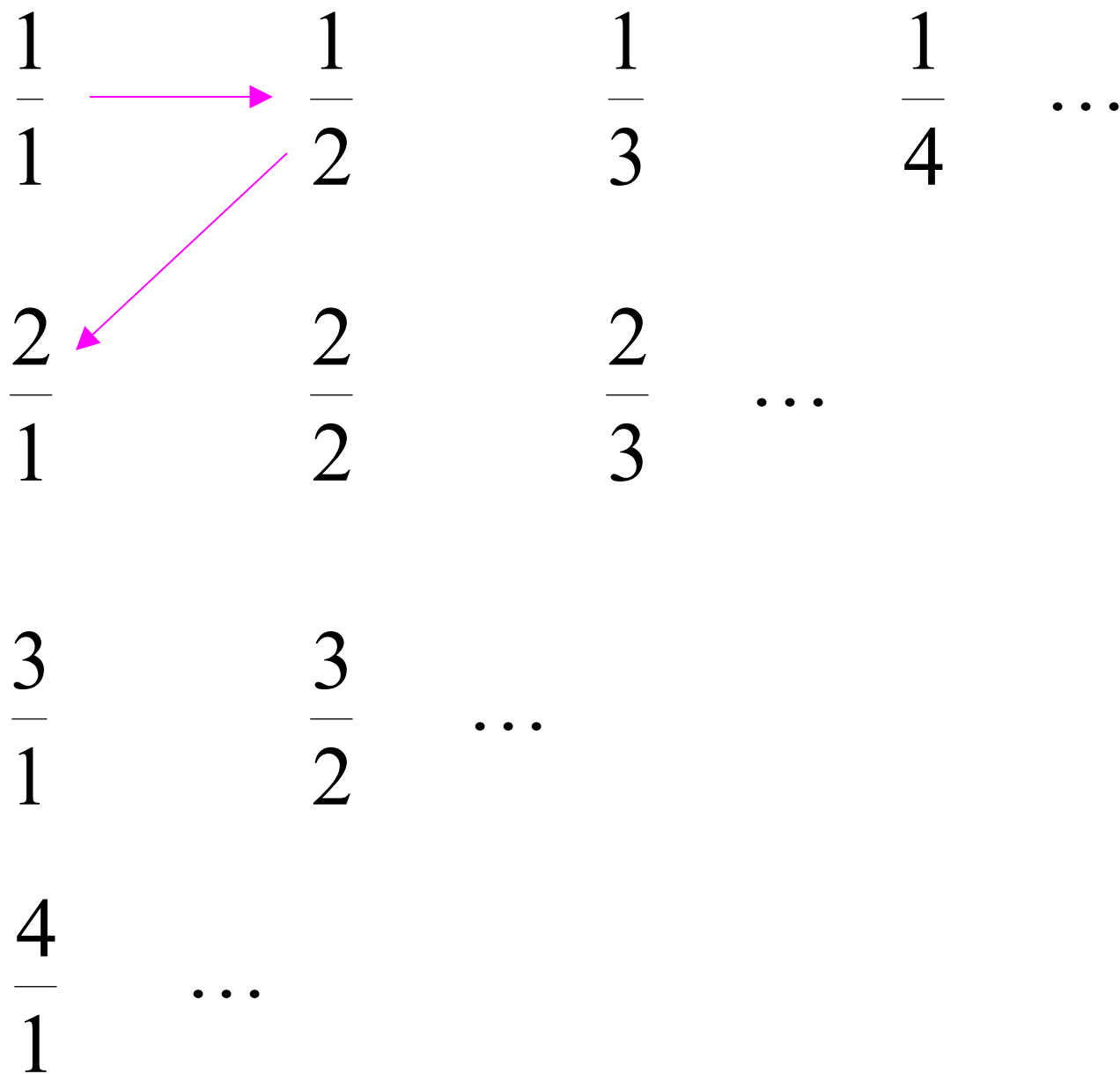
$$\frac{4}{1} \quad \dots$$

$$\frac{1}{1} \xrightarrow{\text{pink arrow}} \frac{1}{2} \quad \frac{1}{3} \quad \frac{1}{4} \quad \dots$$

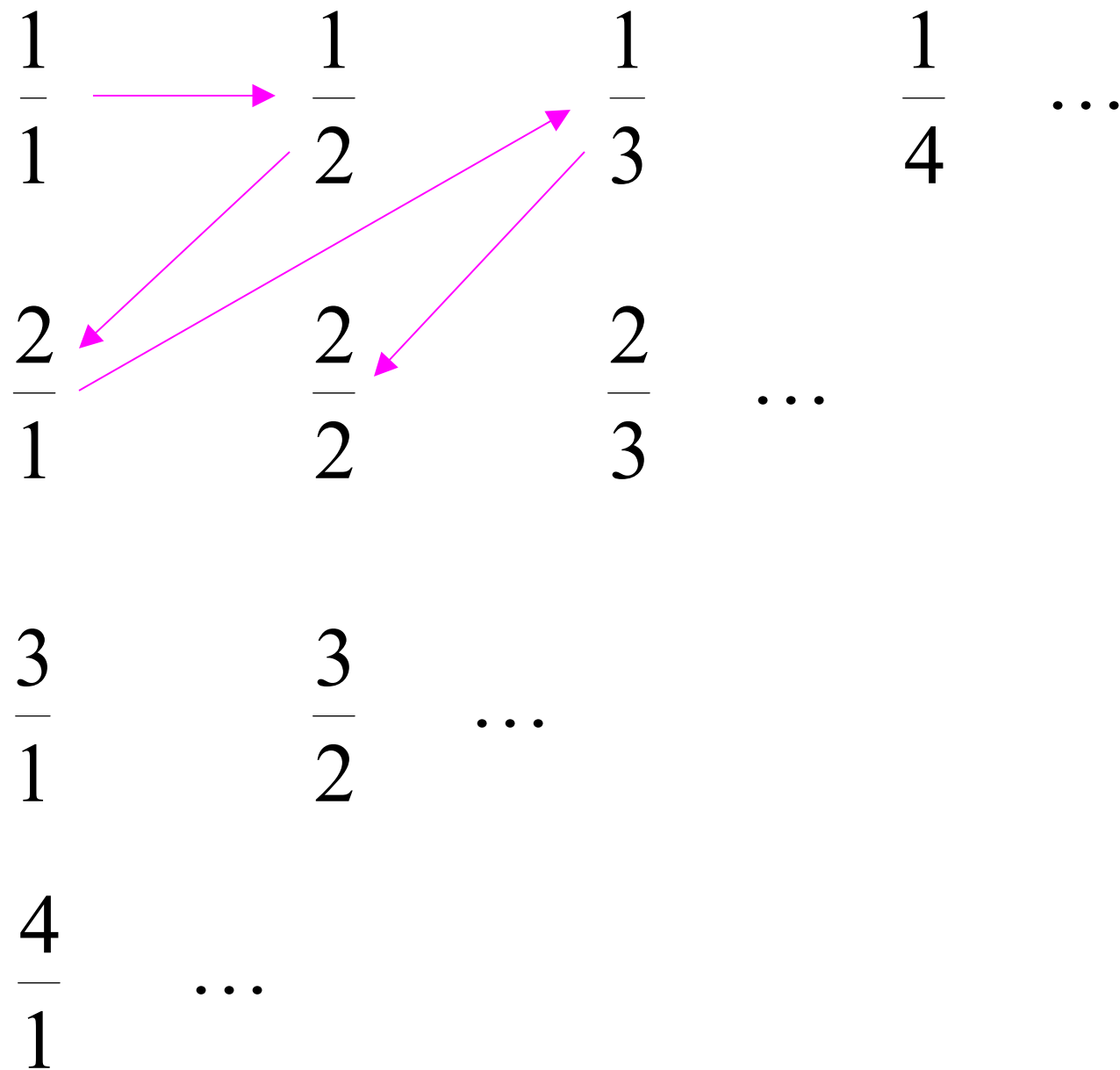
$$\frac{2}{1} \quad \frac{2}{2} \quad \frac{2}{3} \quad \dots$$

$$\frac{3}{1} \quad \frac{3}{2} \quad \dots$$

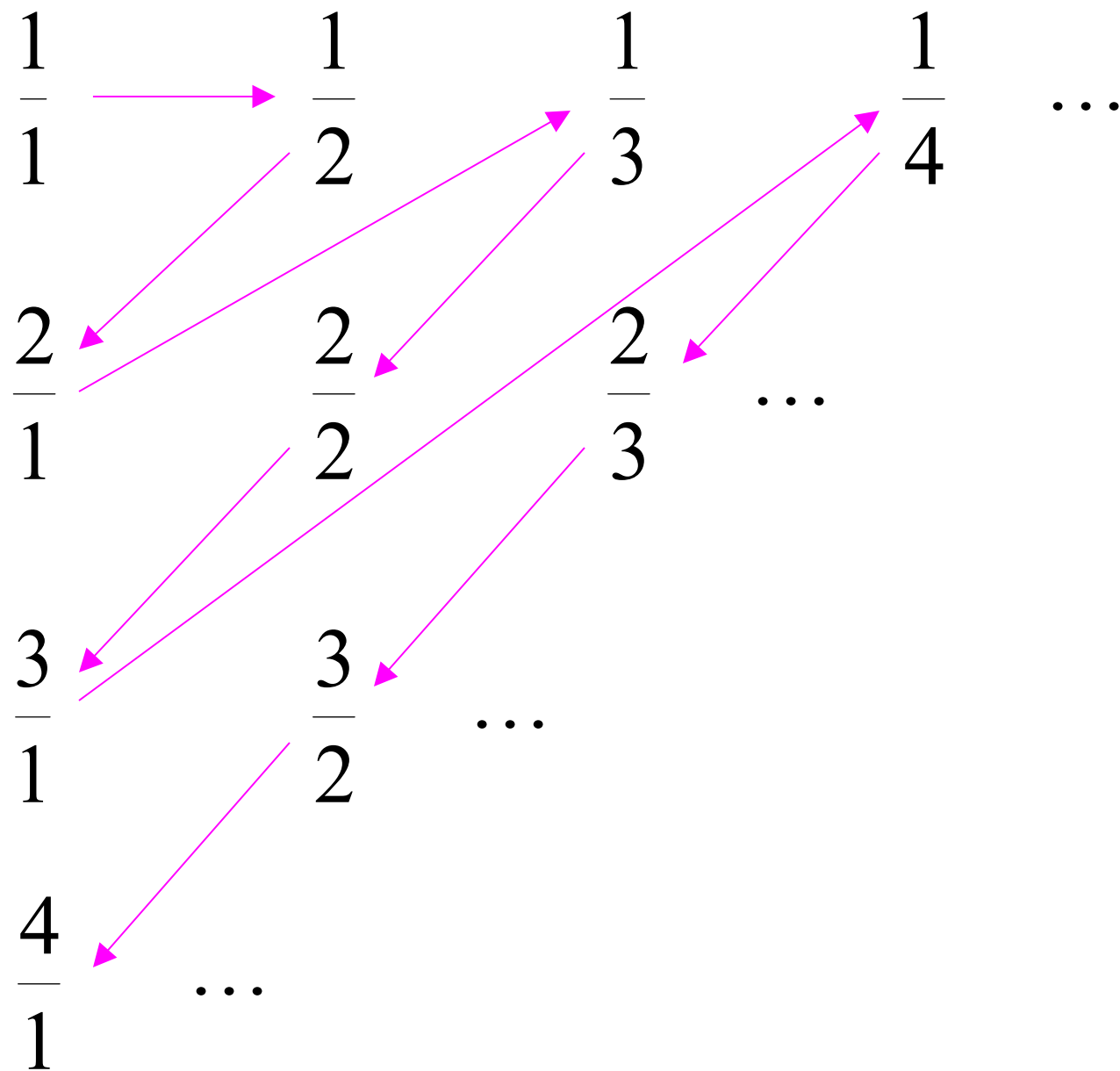
$$\frac{4}{1} \quad \dots$$











Rational Numbers:

$$\frac{1}{1}, \frac{1}{2}, \frac{2}{1}, \frac{1}{3}, \frac{2}{2}, \dots$$

Correspondence:

Positive Integers:

$$1, 2, 3, 4, 5, \dots$$

We proved:

the set of rational numbers is countable  
by describing an enumeration procedure  
(enumerator)  
for the correspondence to natural numbers

## Definition

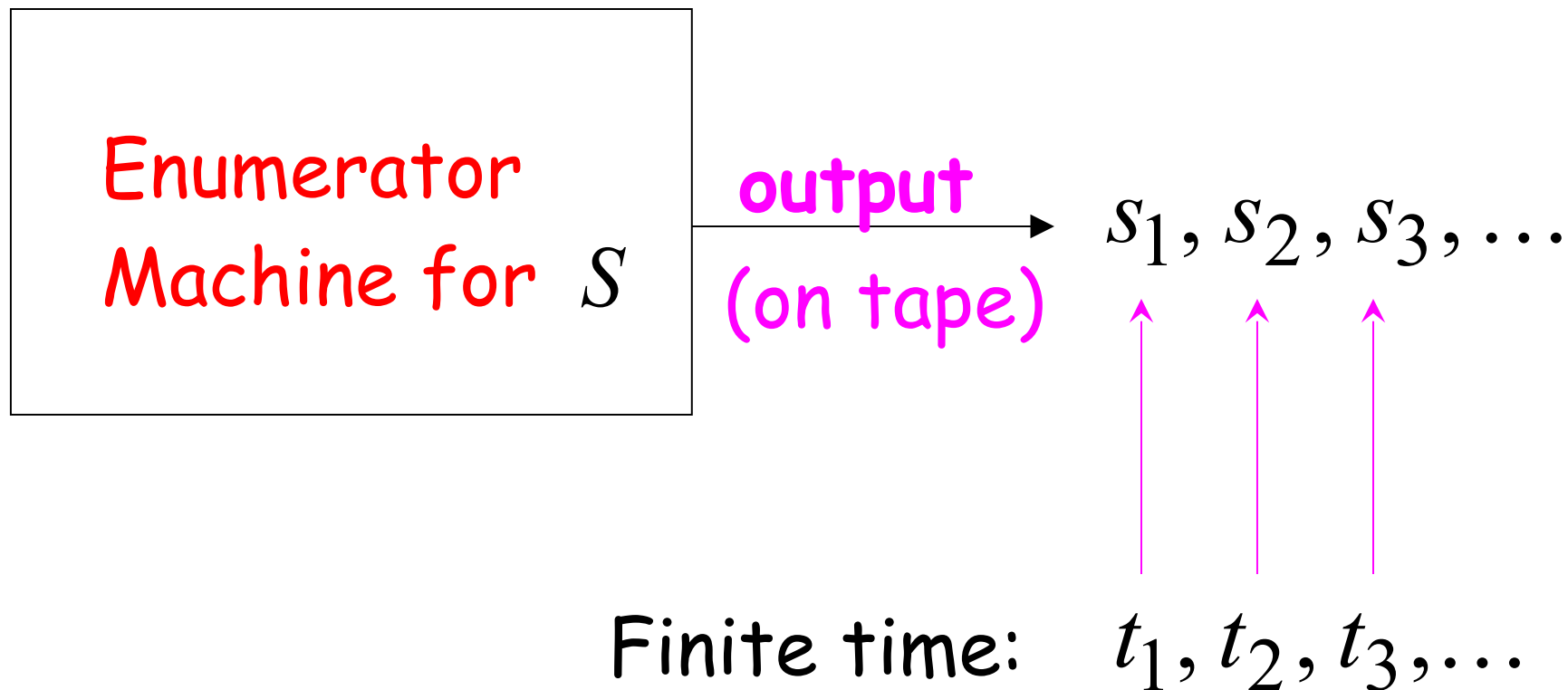
Let  $S$  be a set of strings (Language)

An **enumerator** for  $S$  is a Turing Machine  
that generates (prints on tape)  
all the strings of  $S$  one by one

and

each string is generated in finite time

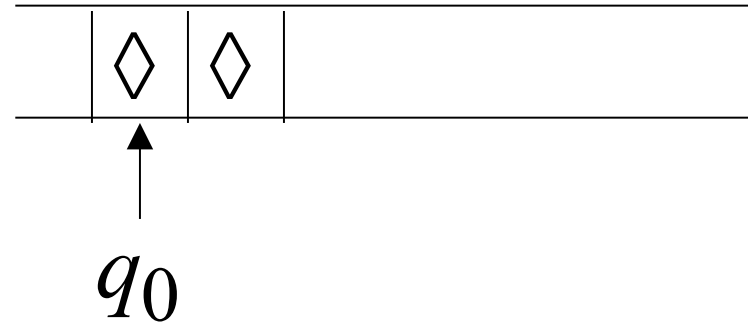
strings  $s_1, s_2, s_3, \dots \in S$



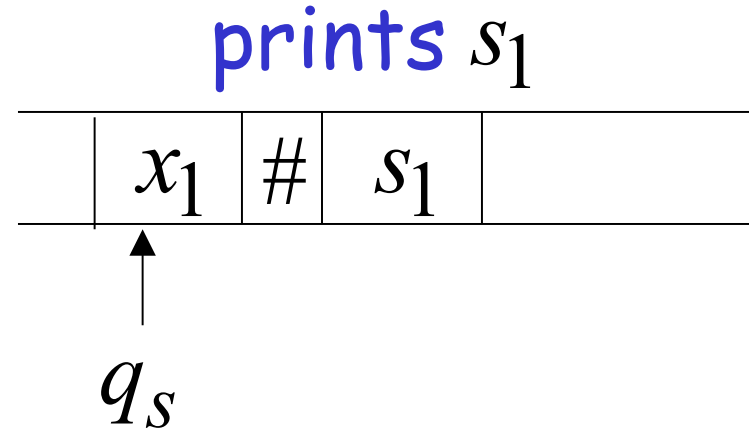
# Enumerator Machine

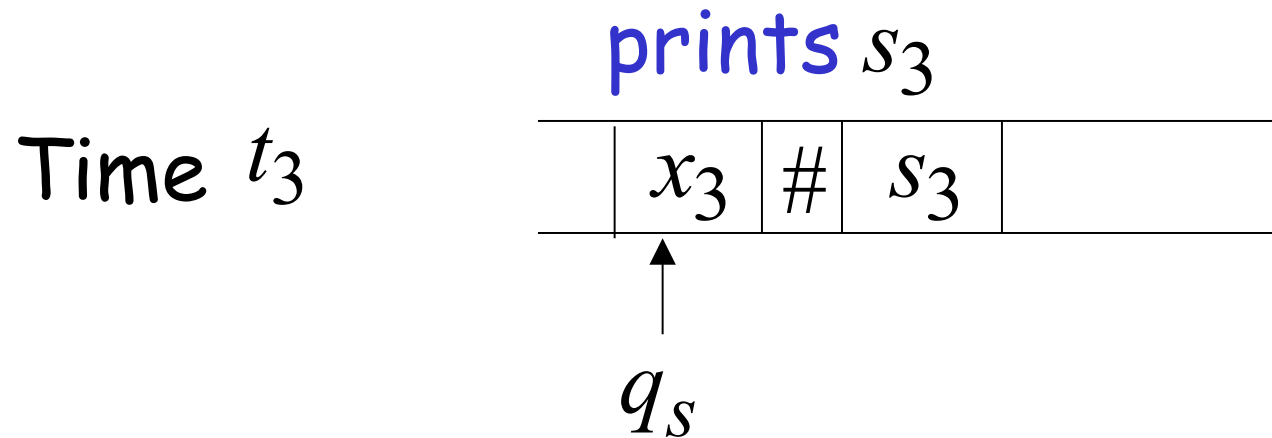
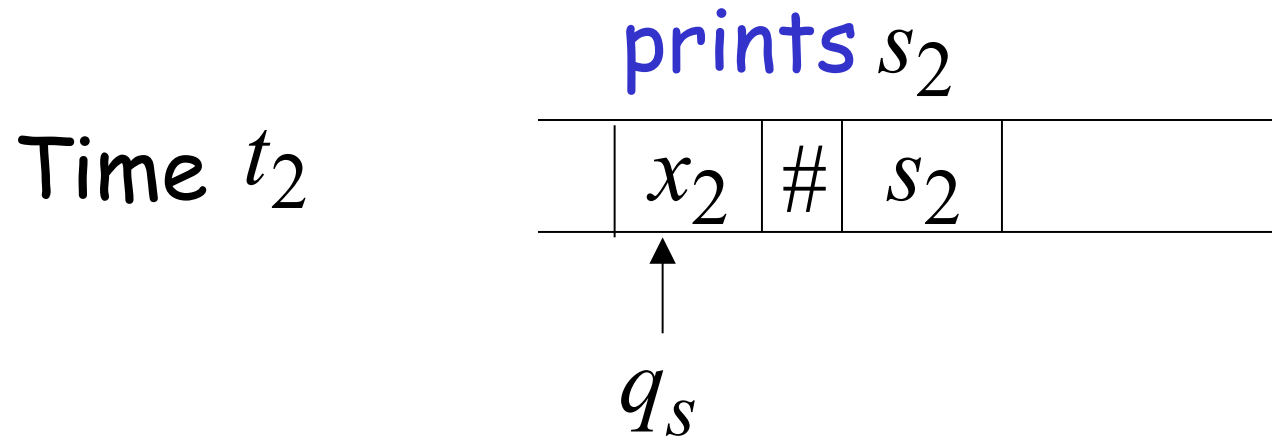
## Configuration

Time 0



Time  $t_1$





## Observation:

If for a set  $S$  there is an enumerator,  
then the set is countable

The enumerator describes the  
correspondence of  $S$  to natural numbers



Example: The set of strings  $S = \{a,b,c\}^+$   
is countable

Approach:

We will describe an enumerator for  $S$

## Naive enumerator:

Produce the strings in lexicographic order:

$$s_1 = a$$

$$s_2 = aa$$

$$\vdots \quad aaa$$

$$aaaa$$

.....

Doesn't work:

strings starting with  $b$   
will never be produced

# Better procedure: **Proper Order** (Canonical Order)

1. Produce all strings of length 1
2. Produce all strings of length 2
3. Produce all strings of length 3
4. Produce all strings of length 4
- ⋮

Produce strings in  
Proper Order:

$s_1 =$	<i>a</i>	}	length 1
$s_2 =$	<i>b</i>		
$\vdots$	<i>c</i>		
	<i>aa</i>	}	length 2
	<i>ab</i>		
	<i>ac</i>		
	<i>ba</i>		
	<i>bb</i>		
	<i>bc</i>		
	<i>ca</i>		
	<i>cb</i>		
	<i>cc</i>		
	<i>aaa</i>	}	length 3
	<i>aab</i>		
	<i>aac</i>		
	<i>.....</i>		

**Theorem:** The set of all Turing Machines is countable

**Proof:** Any Turing Machine can be encoded with a binary string of 0's and 1's

Find an enumeration procedure for the set of Turing Machine strings

# Enumerator:

## Repeat

1. Generate the next binary string of 0's and 1's in proper order
2. Check if the string describes a Turing Machine
  - if **YES**: print string on output tape
  - if **NO**: ignore string

# Binary strings

# Turing Machines

0 ignore

1 ignore

00 ignore

01

⋮

1 0 1 0 1 1 0 1 1 0 0

1 0 1 0 1 1 0 1 1 0 1

⋮

1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 0 1  $\xrightarrow{s_2}$  1 0 1 1 0 1 0 1 0 0 1 0 1 0 1 1 0 1

⋮

$\xrightarrow{s_1}$

1 0 1 0 1 1 0 1 1 0 1

End of Proof

## Simpler Proof:

Each Turing machine binary string is mapped to the number representing its value



# Uncountable Sets

We will prove that there is a language  $L$   
which is not accepted by any Turing machine

Technique:

Turing machines are countable

Languages are uncountable

(there are more languages than Turing Machines)

## Theorem:

If  $S$  is an infinite countable set, then  
the powerset  $2^S$  of  $S$  is uncountable.

The powerset  $2^S$  contains all possible subsets of  $S$

Example:  $S = \{a, b\}$        $2^S = \{\emptyset, \{a\}, \{b\}, \{a, b\}\}$

## Proof:

Since  $S$  is countable, we can list its elements in some order

$$S = \{s_1, s_2, s_3, \dots\}$$



Elements of  $S$

Elements of the powerset  $2^S$  have the form:

$$\emptyset$$

$$\{s_1, s_3\}$$

$$\{s_5, s_7, s_9, s_{10}\}$$

$\vdots$

They are subsets of  $S$

We encode each subset of  $S$   
with a binary string of 0's and 1's

Subset of $S$	Binary encoding				
	$s_1$	$s_2$	$s_3$	$s_4$	$\dots$
$\{s_1\}$	1	0	0	0	$\dots$
$\{s_2, s_3\}$	0	1	1	0	$\dots$
$\{s_1, s_3, s_4\}$	1	0	1	1	$\dots$

Every infinite binary string corresponds to a subset of  $S$ :

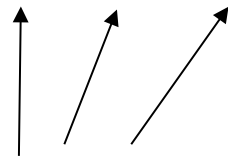
Example: 1 0 0 1 1 1 0 ...

Corresponds to:  $\{s_1, s_4, s_5, s_6, \dots\} \in 2^S$

Let's assume (for contradiction)  
that the powerset  $2^S$  is countable

Then: we can list the elements of the  
powerset in some order

$$2^S = \{t_1, t_2, t_3, \dots\}$$



Subsets of  $S$



# Powerset element

## Binary encoding example

---

$t_1$	1	0	0	0	0	...
-------	---	---	---	---	---	-----

---

$t_2$	1	1	0	0	0	...
-------	---	---	---	---	---	-----

---

$t_3$	1	1	0	1	0	...
-------	---	---	---	---	---	-----

---

$t_4$	1	1	0	0	1	...
-------	---	---	---	---	---	-----

---

...

...

$t$  = the binary string whose bits  
are the complement of the diagonal

$t_1$	1	0	0	0	0	...
$t_2$	1	1	0	0	0	...
$t_3$	1	1	0	1	0	...
$t_4$	1	1	0	0	1	...

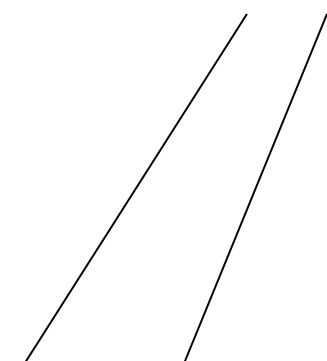
Binary string:  $t = 0011\dots$

(binary complement of diagonal)

The binary string

$$t = 0011\dots$$

corresponds  
to a subset of  $S$ :


$$t = \{s_3, s_4, \dots\} \in 2^S$$

$t$  = the binary string whose bits  
are the complement of the diagonal

$t_1$	1	0	0	0	0	...
$t_2$	1	1	0	0	0	...
$t_3$	1	1	0	1	0	...
$t_4$	1	1	0	0	1	...

$t = 0011 \dots$

Question:  $t = t_1$  ? **NO:** differ in 1<sup>st</sup> bit

$t$  = the binary string whose bits  
are the complement of the diagonal

$t_1$	1	0	0	0	0	...
$t_2$	1	1	0	0	0	...
$t_3$	1	1	0	1	0	...
$t_4$	1	1	0	0	1	...

$t = 0011 \dots$

Question:  $t = t_2$  ? **NO:** differ in 2<sup>nd</sup> bit

$t$  = the binary string whose bits  
are the complement of the diagonal

$t_1$	1	0	0	0	0	...
$t_2$	1	1	0	0	0	...
$t_3$	1	1	0	1	0	...
$t_4$	1	1	0	0	1	...

$t = 0011 \dots$

Question:  $t = t_3$  ? **NO:** differ in 3<sup>rd</sup> bit

Thus:  $t \neq t_i$  for every  $i$

since they differ in the  $i$ th bit

However,  $t \in 2^S \Rightarrow t = t_i$  for some  $i$

Contradiction!!!

Therefore the powerset  $2^S$  is uncountable

End of proof

# An Application: Languages

Consider Alphabet :  $A = \{a, b\}$

The set of all strings:

$$S = \{a, b\}^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, \dots\}$$

infinite and countable

because we can enumerate  
the strings in proper order



Consider Alphabet :  $A = \{a, b\}$

The set of all strings:

$$S = \{a, b\}^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, \dots\}$$

infinite and countable

Any language is a subset of  $S$  :

$$L = \{aa, ab, aab\}$$

Consider Alphabet :  $A = \{a, b\}$

The set of all Strings:

$$S = A^* = \{a, b\}^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, \dots\}$$

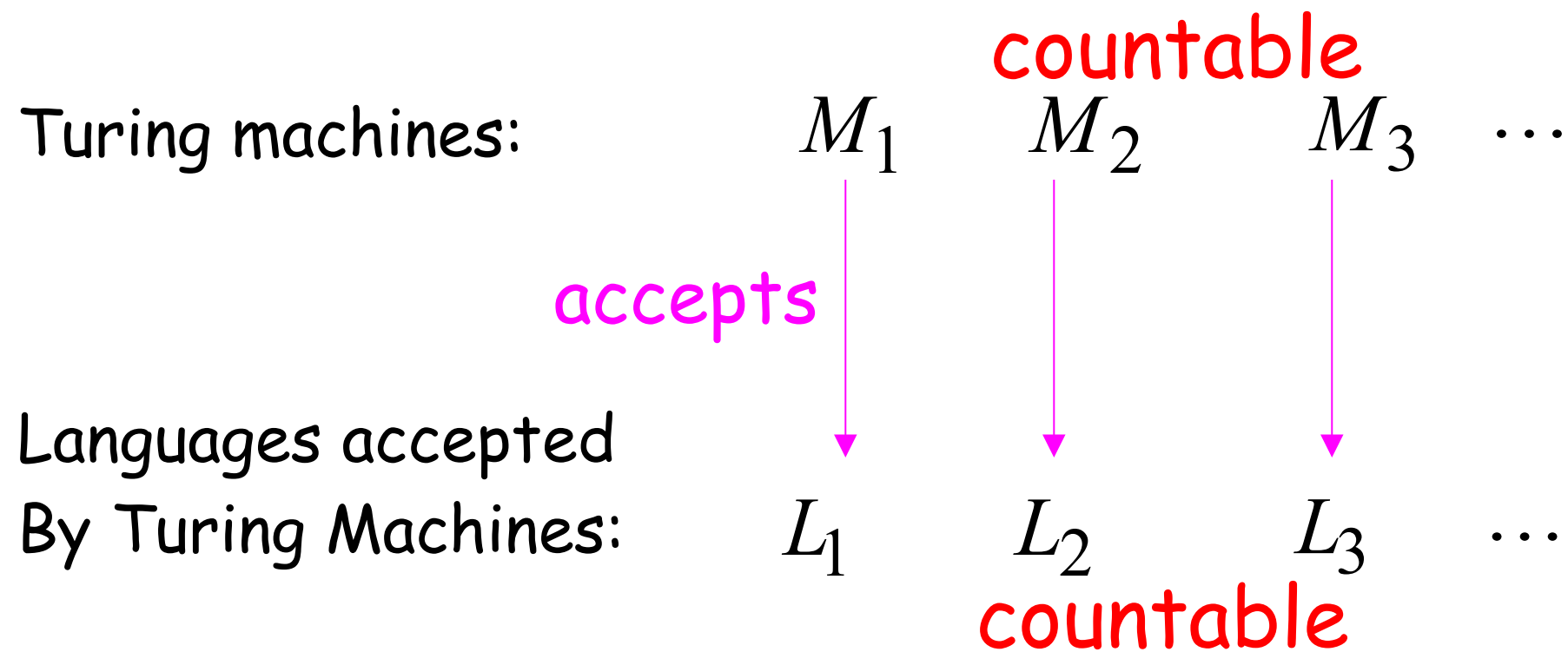
infinite and countable

The powerset of  $S$  contains all languages:

$$2^S = \{\emptyset, \{\varepsilon\}, \{a\}, \{a, b\}, \{aa, b\}, \dots, \{aa, ab, aab\}, \dots\}$$

uncountable

Consider Alphabet :  $A = \{a, b\}$



Denote:  $X = \{L_1, L_2, L_3, \dots\}$       Note:  $X \subseteq 2^S$

countable

$(S = \{a, b\}^*)$

Languages accepted  
by Turing machines:

$X$  countable

All possible languages:  $2^S$  uncountable

Therefore:  $X \neq 2^S$

(since  $X \subseteq 2^S$ , we get  $X \subset 2^S$ )

## Conclusion:

There is a language  $L$  not accepted  
by any Turing Machine:

$$X \subset 2^S \implies \exists L \in 2^S \text{ and } L \notin X$$

# Non Turing-Acceptable Languages

$L$



Turing-Acceptable  
Languages

Note that:  $X = \{L_1, L_2, L_3, \dots\}$

is a *multi-set* (elements may repeat)  
since a language may be accepted  
by more than one Turing machine

However, if we remove the repeated elements,  
the resulting set is again countable since every element  
still corresponds to a positive integer