Limits of Turing Machines

Dr. Chuck Rocca
roccac@wcsu.edu
http://sites.wcsu.edu/roccac

WESTERN CONNECTICUT STATE UNIVERSITY
MACRICOSTAS SCHOOL OF ARTS & SCIENCES

Table of Contents

Table of Contents

Algorithm

Definition

An *algorithm* is a computational process that is describable in terms of a Turing machine.

Connected Graphs Example

M= "On input $\langle G \rangle$, the string encoding of a graph G:

0. Check that $\langle G \rangle$ is in the correct format
1. Select the first node in G and mark it.
2. Repeat the following until no new nodes are marked:
3. For each node in G, mark it if is is attached by an edge to a node that is already marked.
4. Scan all the nodes of G, if they are all marked, *accept*; otherwise *reject*."

Decidable Unions

Given two *decidable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the union of the languages can be decided by:
M=“On input w:

0. Check that w is in the correct format.
1. Run M_1 on w. If it accepts, *accept*.
2. Run M_2 on w. If it accepts, *accept*.
3. Otherwise *reject*”

Recognizable Unions

Given two *Turing recognizable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the union of the languages can be recognized by:
M=“On input w:

0. Check that w is in the correct format.
1. Run M_1 and M_2 alternately on w step by step. If either accepts, *accept*.
2. If both halt or reject, *reject*.”

Decidable vs. Recognizable

Definition

A Language is *Turning-decidable* or simply *decidable* if some Turing machine decides it; the machine always reaches an accept or reject state. Given any word there is a TM that can tell if the word is or is not in the language.

Decidable vs. Recognizable

Definition

A Language is *Turning-decidable* or simply *decidable* if some Turing machine decides it; the machine always reaches an accept or reject state. Given any word there is a TM that can tell if the word is or is not in the language.

Definition

A Language is *Turning-recognizable* if some Turing machine recognizes it; in this case the machine reaches an accept state, reject state, or it may loop (fail to accept). There is a TM that accepts words in the language, but may fail to reach a verdict if a word is not in the language.

Decidable Intersections

Given two *decidable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the intersections of the languages can be decided by:
M="On input w:

0. Check that w is in the correct format.
1. Run M_1 and M_2 on w. If they both accept, *accept*.
2. Otherwise *reject*"

Decidable Intersections

Given two *decidable* languages L_1 and L_2 and corresponding Turing machines M_1 and M_2 the intersections of the languages can be decided by:
M="On input w:

0. Check that w is in the correct format.
1. Run M_1 and M_2 on w. If they both accept, *accept*.
2. Otherwise *reject*"

Why didn't we say "Run M_1 and M_2 alternately on w step by step?"

Decidable Complements

Given a *decidable* language L_1 and corresponding Turing machine M_1 the complement of the languages can be decided by:
M="On input w:

0. Check that w is in the correct format.
1. Run M_1 on w. If it accepts, *reject*.
2. Otherwise *accept*"

Table of Contents

Deterministic Finite Automaton

Theorem

The set

$$A_{DFA} = \{\langle B, w \rangle \mid B \text{ is a DFA that accepts string } w\}$$

is a decidable language.

Deterministic Finite Automaton

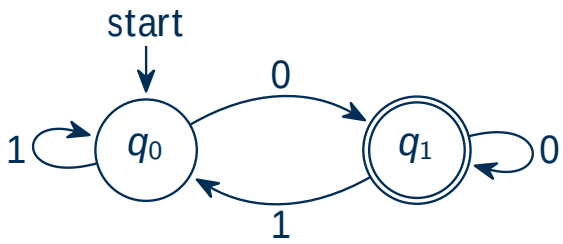
M = \On input $\hbar B; wi$, where B is a DFA and w is a string:

0. Check the format of the input.
1. Simulate B on input w.
2. If the simulation ends in an accept state, *accept*; otherwise, *reject*."

Deterministic Finite Automaton

M = \On input $\langle B, w \rangle$, where B is a DFA and w is a string:

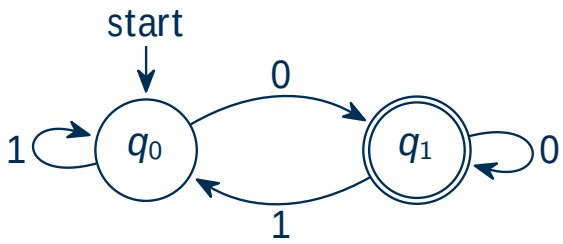
0. Check the format of the input.
1. Simulate B on input w.
2. If the simulation ends in an accept state, *accept*; otherwise, *reject*."

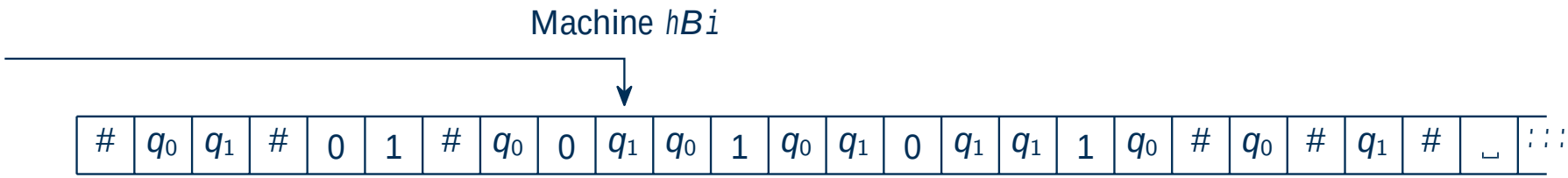


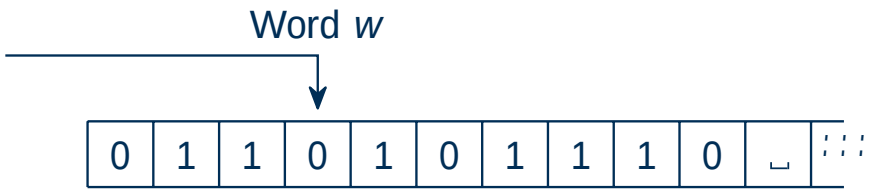
Deterministic Finite Automaton

M = \On input $\langle B, w \rangle$, where B is a DFA and w is a string:

0. Check the format of the input.
1. Simulate B on input w.
2. If the simulation ends in an accept state, *accept*; otherwise, *reject*."







Nondeterministic Finite Automaton

Theorem

The set

$$A_{NFA} = \{\langle B, w\rangle \mid B \text{ is an NFA that accepts string } w\}$$

is a decidable language.

N="On input $\langle B, w\rangle$, where B is a NFA and w is a string:

1. Convert B to a DFA C.
2. Run M from the previous theorem on input $\langle C, w\rangle$.
3. If the simulation ends in an accept state, *accept*; otherwise, *reject*."

Regular Expressions

Theorem

The set

$$A_{REX} = \{\langle B, w \rangle \mid B \text{ is a regex that accepts string } w\}$$

is a decidable language.

P="On input $\langle B, w \rangle$, where B is a RegEx and w is a string:

1. Convert B to a NFA C.
2. Run N from the previous theorem on input $\langle C, w \rangle$.
3. If the simulation ends in an accept state, *accept*; otherwise, *reject*."

Empty Languages

Theorem

The set

$$E_{DFA} = \{\langle A \rangle \mid A \text{ is a DFA and } L(A) = \emptyset\}$$

is a decidable language.

T="On input $\langle A \rangle$, the string encoding of DFA A:

1. Select the start state in A and mark it.
2. Repeat the following until no new states are marked:
3. For each state in A, mark it if there is a transition from a marked state.
4. Scan the accept states of A, if any are marked, *reject*; otherwise *accept*."

Equal Languages

Theorem

The set

$$EQ_{DFA} = fhA, B\, i\, jA \text{ and } B \text{ are DFAs and } L(A) = L(B)g$$

is a decidable language.

- Note, DFAs are closed under *unions*, *intersections,* and *compliments*.
- Construct the *symmetric di erence* of A and B, C A *XOR* B or

$$L(C) = \left(L(A) \setminus \overline{L(B)}\right) [\left(\overline{L(A)} \setminus L(B)\right).$$

- Check if C is empty using T from the previous theorem.

CFLs

Theorem

The set

$$A_{CFG} = \{\langle G, w\rangle \mid G \text{ is a CFG that generates the string } w\}$$

is a decidable language.

$S =$ "On input $\langle G, w\rangle$, where G is a CFG and w is a string:

1. Convert G to Chomsky Normal Form.
2. List all derivations with $2n - 1$ steps, $n = |w|$; except for n=0, then list derivations with one step. (Why $2n - 1$?)
3. If w is generated, *accept*; otherwise *reject*."

CFLs

Theorem

The set

$$E_{CFG} = \{\langle G \rangle \mid G \text{ is a CFG and } L(G) = \emptyset\}$$

is a decidable language.

R="On input $\langle G \rangle$, the string encoding of CFG G:

1. Mark the terminals in G.
2. Repeat the following until no new variables get marked:
3. Mark any variable A where $A \to U_1 U_2 \cdots U_k$ is in G and the U_i are all marked.
4. If the start variable of G is not marked, *accept*; otherwise *reject*."

CFLs

Theorem

The set

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

is not a decidable language.

CFLs

Theorem

The set

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

is not a decidable language. (Proof held until after Chapter 5.)

CFLs

Theorem

The set

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

is not a decidable language. (Proof held until after Chapter 5.)

- Recall, CFLs are not necessarily closed under intersections and complementation.

CFLs

Theorem

The set

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

is not a decidable language. (Proof held until after Chapter 5.)

- Recall, CFLs are not necessarily closed under intersections and complementation.
- $L(G) = \{a^m b^n c^n \mid m, n \quad 0\}$

CFLs

Theorem

The set

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

is not a decidable language. (Proof held until after Chapter 5.)

- Recall, CFLs are not necessarily closed under intersections and complementation.
- $L(G) = \{a^m b^n c^n \mid m, n \quad 0\}$
- $L(H) = \{a^n b^n c^m \mid m, n \quad 0\}$

CFLs

Theorem

The set

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

is not a decidable language. (Proof held until after Chapter 5.)

- Recall, CFLs are not necessarily closed under intersections and complementation.
- $L(G) = \{a^m b^n c^n \mid m, n \quad 0\}$
- $L(H) = \{a^n b^n c^m \mid m, n \quad 0\}$
- $L(G) \setminus L(H) = \{a^n b^n c^n \mid n \quad 0\}$ is not context-free by the pumping lemma

Hierarchy of Languages

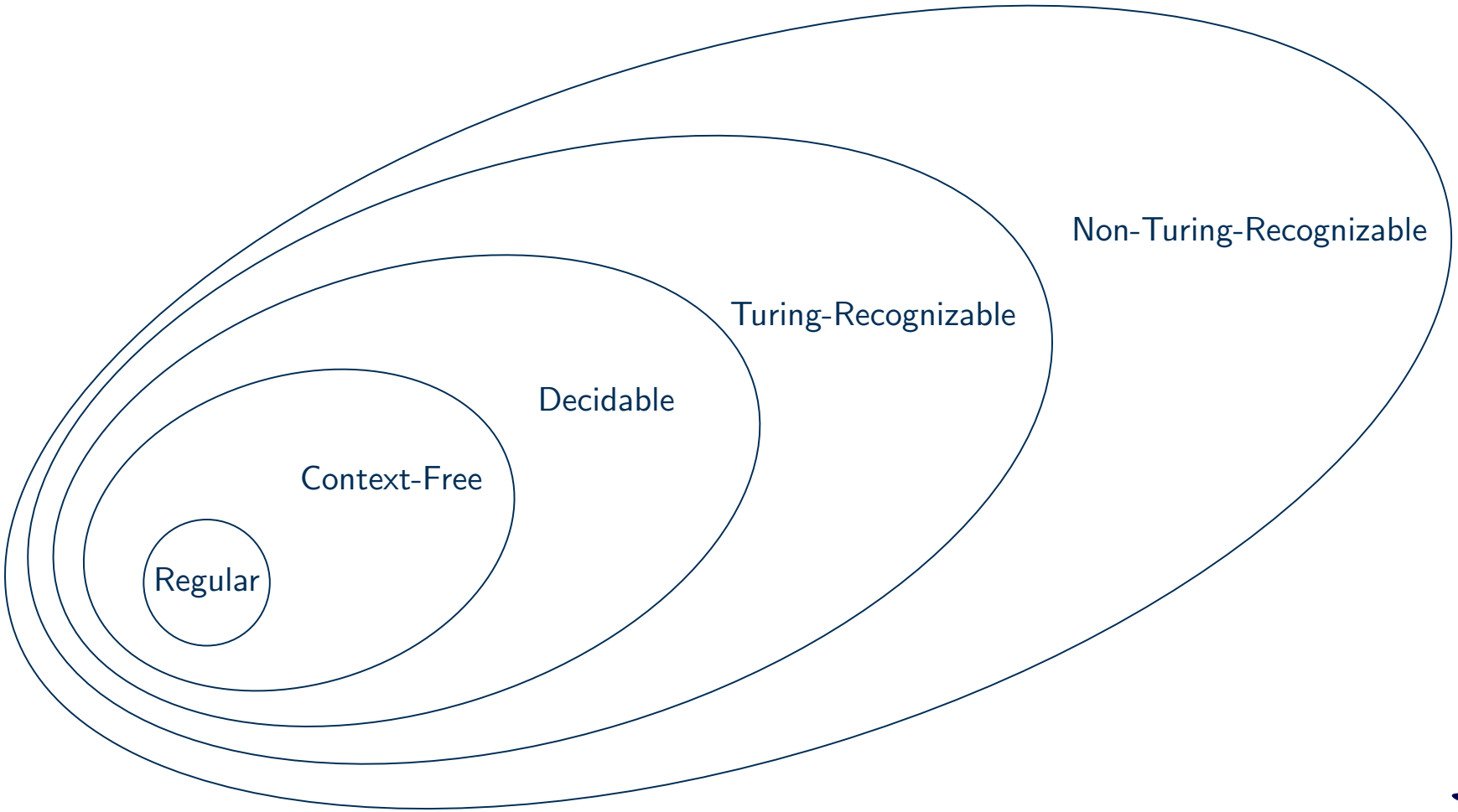


Table of Contents

Cardinality: Naive Definition

Definition

The *cardinality* of a set is the number of elements in the set and two sets have the same cardinality if they have the same number of elements.

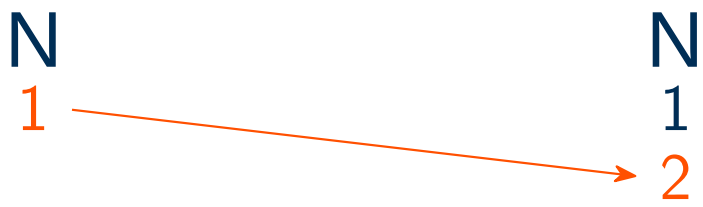
$$A = f1, 2, 3, 4, 5g$$
$$B = fa, b, c, d, eg$$
$$C = f!, @, \#, \$, \%, \hat{}g$$
$$N = f1, 2, 3, 4, ...g$$
$$Z = f0, \quad 1, \quad 2, \quad 3, ...g$$
$$Q = fa/bja, b \ 2 \ Z \ and \ b \ 6= \ 0g$$

Cardinality: Improved Definition

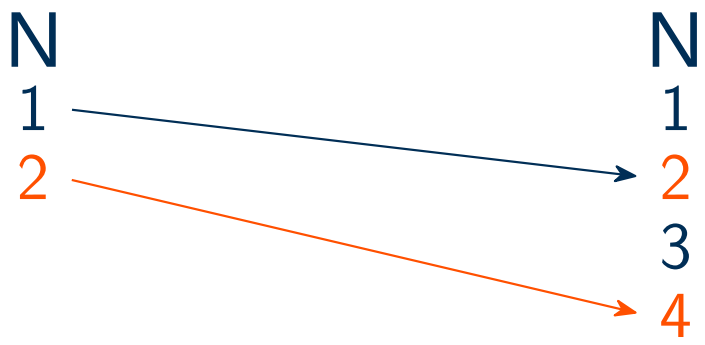
Definition

The *cardinality* of a finite set is the number of elements in the set. A set is *in nite* if there is a one-to-one correspondence between the set and a proper subset of the set. And, two sets have the same cardinality if there exists a one-to-one correspondence between their elements.

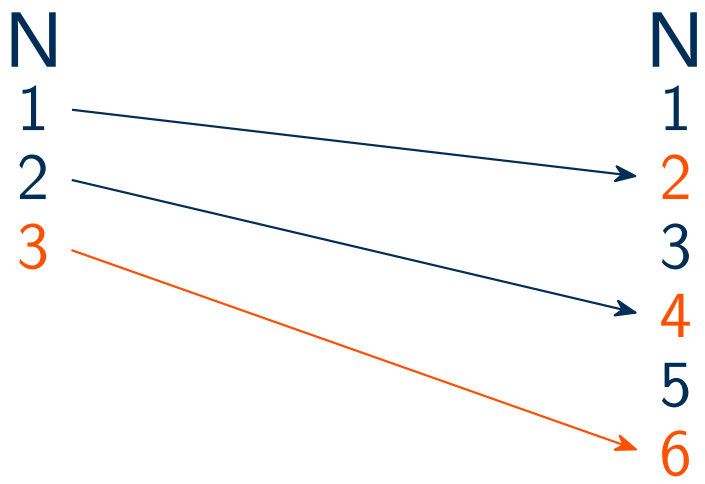
N to 2N



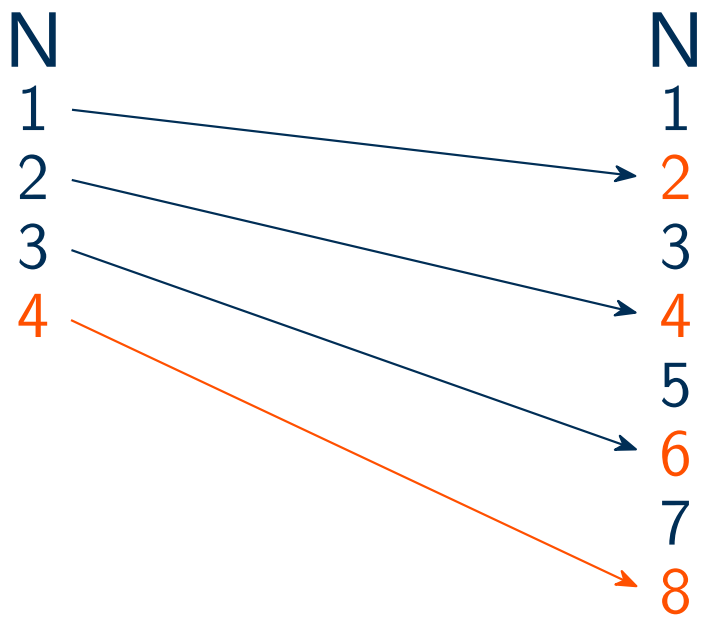
N to 2N



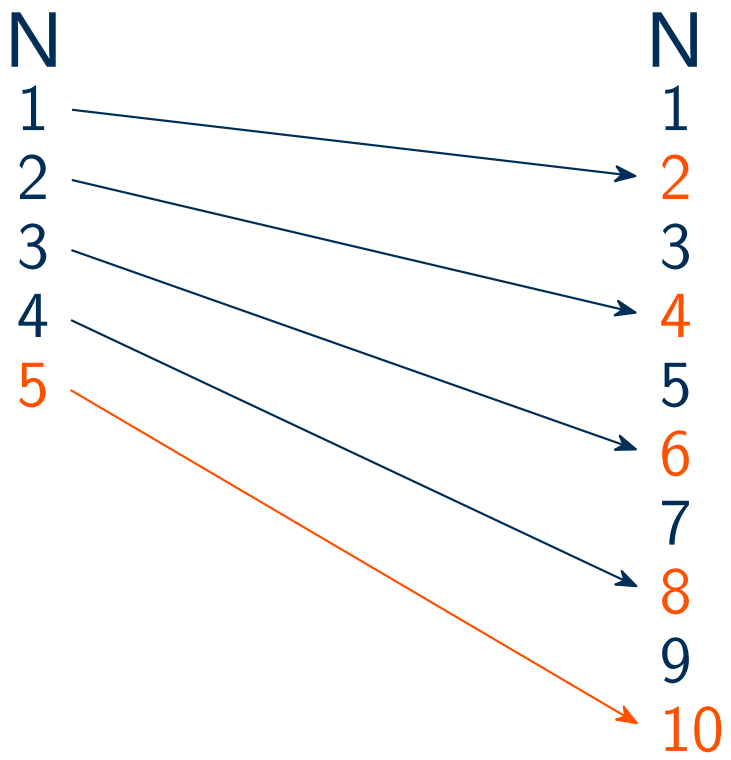
N to 2N



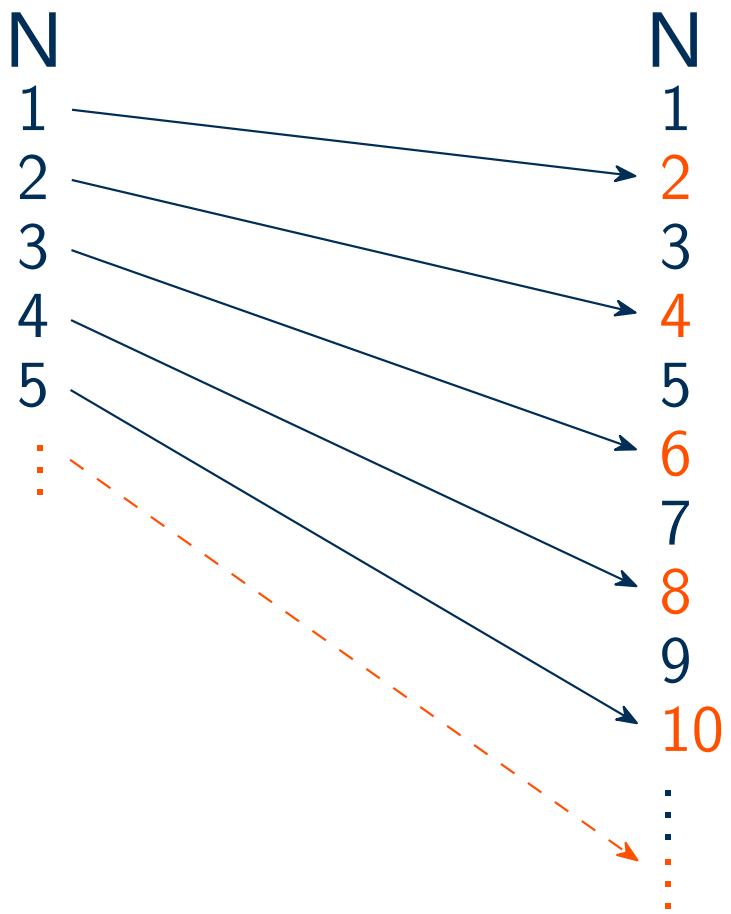
N to 2N



N to 2N



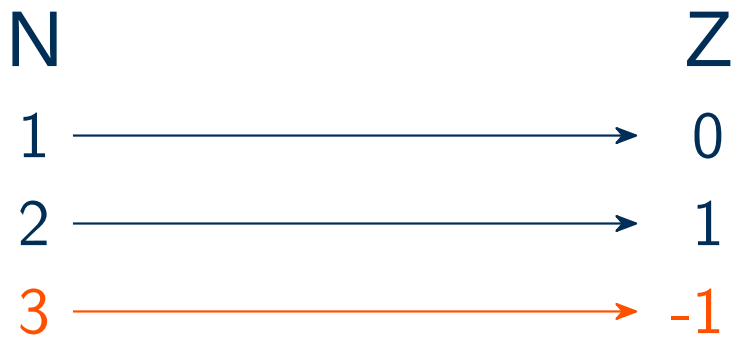
N to 2N



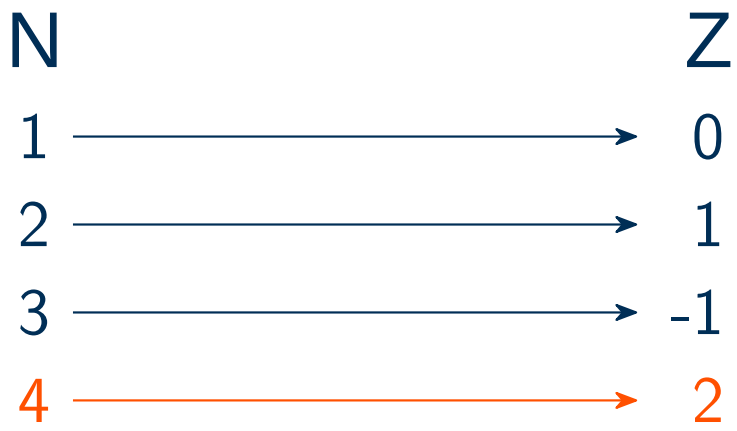
N to Z

N to Z

N to Z

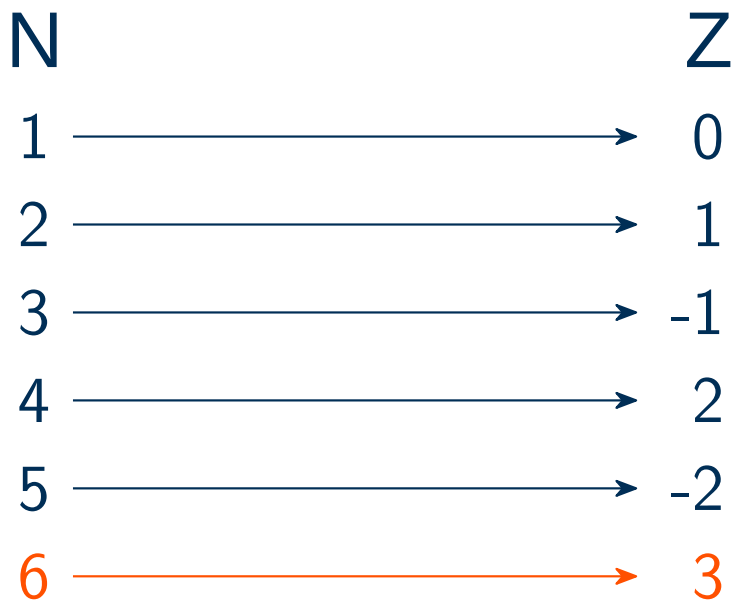


N to Z

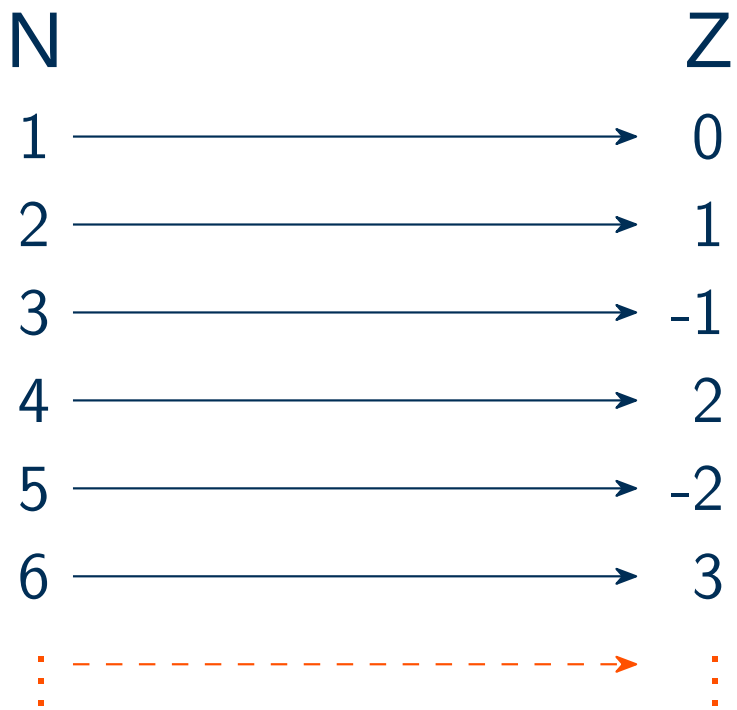


N to Z

N to Z



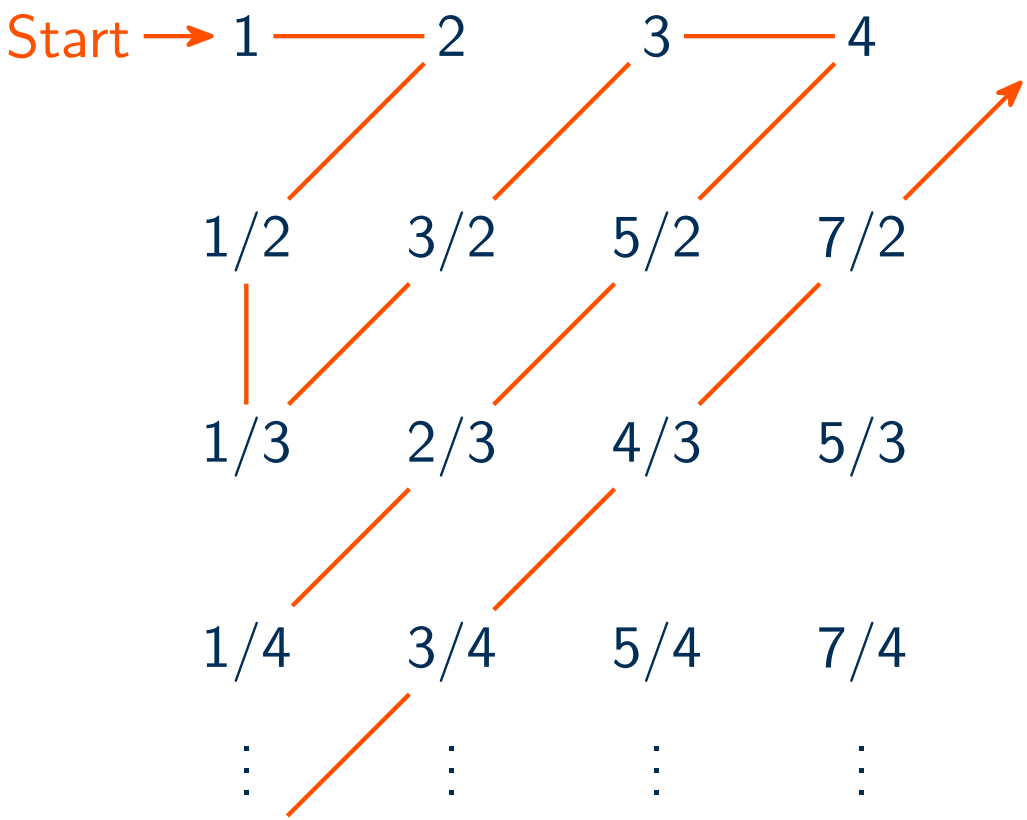
N to Z



N to Q^+

1	2	3	4
1/2	3/2	5/2	7/2
1/3	2/3	4/3	5/3
1/4	3/4	5/4	7/4
⋮	⋮	⋮	⋮

N to Q^+



Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.7$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.74$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.741$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.7412$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.74128$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.741287$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.741287\mathbf{3}$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.74128731$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.741287318$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.7412873187\ldots$$

Cantor Diagonalization

N	R
1	0.65395501314
2	0.73800613014
3	0.05050813247
4	0.10810350448
5	0.04587954758
6	0.66716666577
7	0.73243627345
8	0.27311930829
9	0.17177211903
10	0.45518277788
⋮	⋮

New Number:

$$x = 0.7412873187\ldots$$

Avoiding 0's and 9's when replacing digits since

$$0.19999\ldots = 0.20000\ldots$$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$
$$P(\mathbb{N}) = \{;, \{1\}, \{2, 3\}, \{4\}, \{8\}, \{7, 19, 83\}, \{101, 23, 7\}, \ldots\}$$

Power Sets

$$\mathrm{N} = f1, 2, 3, 4, 5, \ldots g$$
$$P\left(\mathrm{N}\right) = fs_1, s_2, s_3, s_4, s_5 \ldots g$$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \in \mathbb{N} \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$
- $2 \in s_2$, then $2 \notin X$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$
- $2 \in s_2$, then $2 \notin X$
- $2 \notin s_2$, then $2 \in X$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$
- $2 \in s_2$, then $2 \notin X$
- $2 \notin s_2$, then $2 \in X$
- $3 \in s_3$, then $3 \notin X$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$
- $2 \in s_2$, then $2 \notin X$
- $2 \notin s_2$, then $2 \in X$
- $3 \in s_3$, then $3 \notin X$
- $3 \notin s_3$, then $3 \in X$

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$
- $2 \in s_2$, then $2 \notin X$
- $2 \notin s_2$, then $2 \in X$
- $3 \in s_3$, then $3 \notin X$
- $3 \notin s_3$, then $3 \in X$
- etc.

Power Sets

$$\mathbb{N} = \{1, 2, 3, 4, 5, \ldots\}$$

$$P(\mathbb{N}) = \{s_1, s_2, s_3, s_4, s_5 \ldots\}$$

$$\begin{array}{ccc} 1 & \longrightarrow & s_1 \\ 2 & \longrightarrow & s_2 \\ 3 & \longrightarrow & s_3 \\ 4 & \longrightarrow & s_4 \\ 5 & \longrightarrow & s_5 \\ 6 & \longrightarrow & s_6 \\ 7 & \longrightarrow & s_7 \\ \vdots & \dashrightarrow & \vdots \end{array}$$

Define a new set X as follows

$$X = \{n \mid n \notin s_n\}$$

Thus, if

- $1 \in s_1$, then $1 \notin X$
- $1 \notin s_1$, then $1 \in X$
- $2 \in s_2$, then $2 \notin X$
- $2 \notin s_2$, then $2 \in X$
- $3 \in s_3$, then $3 \notin X$
- $3 \notin s_3$, then $3 \in X$
- etc.

And therefore $\forall n : X \neq s_n$.

A Theorem on Power Sets

Theorem

Given a set S, the cardinality of $P(S)$ is always greater than the cardinality of S; there are in nitely many in nities.

Table of Contents

A_{TM} is Undecidable

Theorem

Given the set

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

A_{TM} *is undecidable.*

A_{TM} is Undecidable

Theorem

Given the set

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

A_{TM} *is undecidable.*

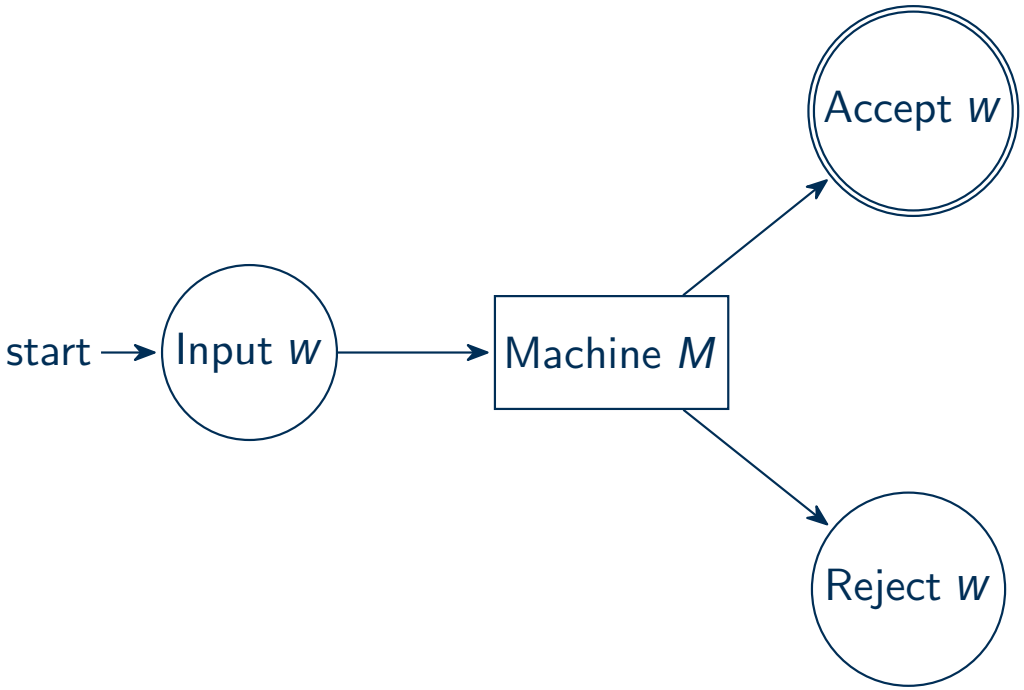
$U =$ " On input $\langle M, w \rangle$, where M is a TM and w is a string:

1. Simulate M on w.
2. If M ever enters its accept state, *accept*; If M ever enters its reject state, *reject*."

This is a *universal Turing machine* and shows that A_{TM} is recognizable.

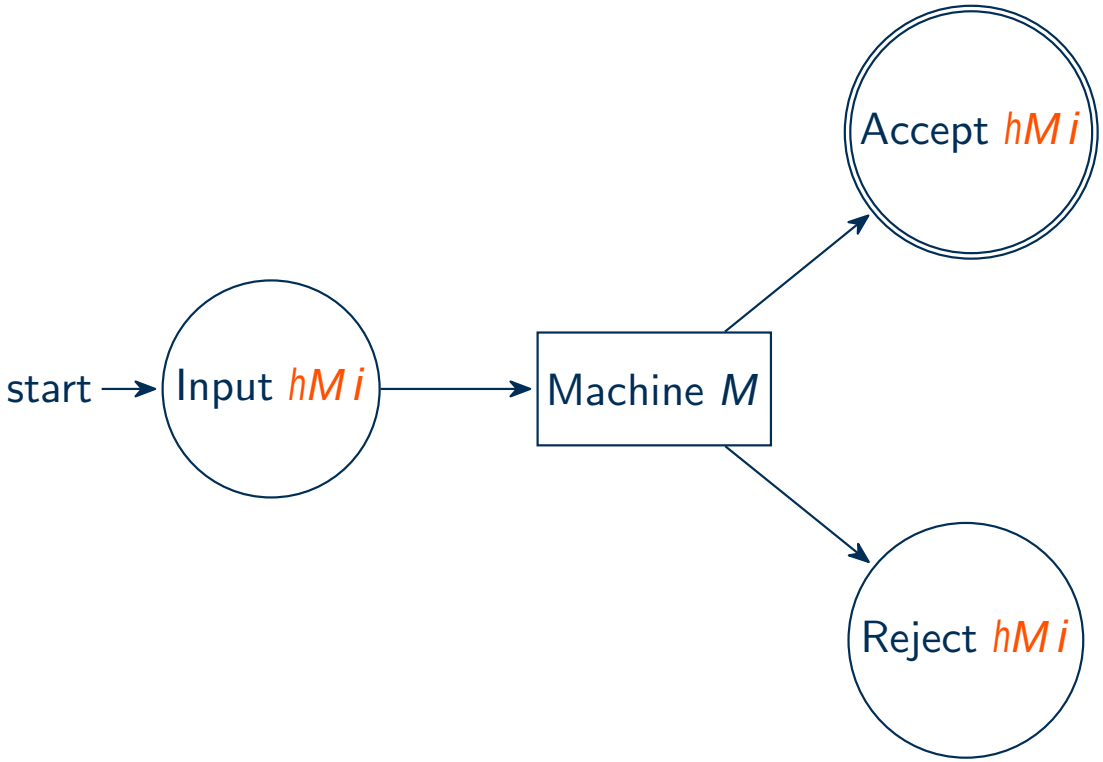
An Undecidable Language

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



An Undecidable Language

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



(Think C++ compiler written in C++.)

An Undecidable Language

- $A_{TM} = f\langle M, w\rangle \mid M$ is a Turing Machine and M accepts $w g$
- Suppose there's a *decider* H for A_{TM}

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

An Undecidable Language

- $A_{TM} = \{\langle M, w \rangle \mid M \text{ is a Turing Machine and } M \text{ accepts } w\}$
- Suppose there's a *decider* H for A_{TM}
- Define $D(\langle M \rangle) = :H(\langle M, \langle M \rangle \rangle)$

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

An Undecidable Language

- $A_{TM} = fhM, w i jM$ is a Turing Machine and M accepts wg
- Suppose there's a *decider* H for A_{TM}
- Define $D(hM i) = :H(hM, hM i i)$
- D can't decide $hD i$

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

An Undecidable Language

- $A_{TM} = fhM, w\, i\, jM$ is a Turing Machine and M accepts wg
- Suppose there's a *decider* H for A_{TM}
- Define $D(hM\, i) = :H(hM, hM\, i\, i)$
- D can't decide $hD\, i$

$$D(hD\, i) = \begin{cases} accept & \text{if } D \text{ does not accept } hD\, i \\ reject & \text{if } D \text{ accepts } hD\, i \end{cases}$$

-) Neither D nor H can exist and so A_{TM} is undecidable

Machine D

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

Machine D

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

Machine D

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

Machine D

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

Machine D

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

Machine D

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

A_{TM} is Undecidable

Theorem

Given the set

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

A_{TM} *is undecidable.*

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $|\Sigma| < 1$ implies $|\Sigma|$ is countable

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $|\Sigma| < 1$ implies $|\Sigma^*|$ is countable
- Every TM can be represented by a string, $\langle TM \rangle$, in Σ^*

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $|\Sigma| < 1$ implies $|\Sigma^*|$ is countable
- Every TM can be represented by a string, $\langle TM \rangle$, in Σ^*
- There are countably many TMs

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $j\Sigma j < 1$ implies $j\Sigma$ j is countable
- Every TM can be represented by a string, $hTMi$, in Σ
- There are countably many TMs
- A language, A, is a subset of Σ

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $j\Sigma j < 1$ implies $j\Sigma$ j is countable
- Every TM can be represented by a string, $hTMi$, in Σ
- There are countably many TMs
- A language, A, is a subset of Σ
- $L = fAjA$ is a language$g = P(\Sigma)$

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $j\Sigma j < 1$ implies $j\Sigma$ j is countable
- Every TM can be represented by a string, $hTMi$, in Σ
- There are countably many TMs
- A language, A, is a subset of Σ
- $L = fAjA$ is a language$g = P(\Sigma)$
- L is uncountable

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

- $j\Sigma j < 1$ implies $j\Sigma$ j is countable
- Every TM can be represented by a string, $hTMi$, in Σ
- There are countably many TMs
- A language, A, is a subset of Σ
- $L = fAjA$ is a language$g = P(\Sigma)$
- L is uncountable
-) Some languages are not Turing-recognizable

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Alternately:

- $\Sigma = f s_1, s_2, s_3, s_4, \ldots g$

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Alternately:

- $\Sigma = f s_1, s_2, s_3, s_4, \ldots g$
- $B = f$infinite binary sequences$g$

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Alternately:

- $\Sigma = f s_1, s_2, s_3, s_4, \ldots g$
- $B = f$infinite binary sequences$g$
- $L = f$all languages$g = P(\Sigma)$

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Alternately:

- $\Sigma = f s_1, s_2, s_3, s_4, \ldots g$
- $B = f$infinite binary sequences$g$
- $L = f$all languages$g = P(\Sigma)$
- $f : L \quad ! \quad B$

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Alternately:

- $\Sigma = \{s_1, s_2, s_3, s_4, \ldots\}$
- $B = \{\text{infinite binary sequences}\}$
- $L = \{\text{all languages}\} = P(\Sigma)$
- $f : L \to B$
- $\forall A \in L : f(A) = b_1 b_2 b_3 b_4 \ldots \in B$

$$b_i = \begin{cases} 0 & s_i \notin A \\ 1 & s_i \in A \end{cases}$$

Non-Turing Recognizable Languages

Theorem

Some languages are not Turing-Recognizable.

Alternately:

- $\Sigma = \{s_1, s_2, s_3, s_4, \ldots\}$
- $B = \{\text{infinite binary sequences}\}$
- $L = \{\text{all languages}\} = P(\Sigma)$
- $f : L \to B$
- $\forall A \in L : f(A) = b_1 b_2 b_3 b_4 \ldots \in B$

$$b_i = \begin{cases} 0 & s_i \notin A \\ 1 & s_i \in A \end{cases}$$

- $\chi_A = f(A)$ is called the *characteristic sequence* of A

Decidability vs. Recognizably

Definition

A language, $\overline{A}$, is *co-Turing-recognizable* if it is the complement of a Turing-recognizable language A.

Decidability vs. Recognizably

Definition

A language, $\overline{A}$, is *co-Turing-recognizable* if it is the complement of a Turing-recognizable language A.

Theorem

A language is decidable if and only if it is Turing-recognizable and co-Turing-recognizable. (i.e. A and $\overline{A}$ are both recognizable)

Decidability vs. Recognizably

Definition

A language, $\overline{A}$, is *co-Turing-recognizable* if it is the complement of a Turing-recognizable language A.

Theorem

A language is decidable if and only if it is Turing-recognizable and co-Turing-recognizable. (i.e. A and $\overline{A}$ are both recognizable)

Theorem (contrapositive)

A language is non-decidable if and only if it is not Turing-recognizable or not co-Turing-recognizable.

Decidability vs. Recognizably

Definition

A language, $\overline{A}$, is *co-Turing-recognizable* if it is the complement of a Turing-recognizable language A.

Theorem (contrapositive)

A language is non-decidable if and only if it is not Turing-recognizable or not co-Turing-recognizable.

Corollary

The language $\overline{A_{TM}}$ is non-Turing-recognizable.

Table of Contents

Next Class

- Some Undecidable Problems

Next Class

- Some Undecidable Problems
- Specific Undecidable Problem

Next Class

- Some Undecidable Problems
- Specific Undecidable Problem
- Mapping Reducibility

Limits of Turing Machines

Dr. Chuck Rocca
roccac@wcsu.edu
http://sites.wcsu.edu/roccac

WESTERN CONNECTICUT STATE UNIVERSITY
MACRICOSTAS SCHOOL OF ARTS & SCIENCES