Cantor's Theorem

Theorem 7 *The set of all subsets of* $\mathbb{N}$ $(2^{\mathbb{N}})$ *is infinite and not countable.*

- Suppose $(2^{\mathbb{N}})$ is countable with $f : \mathbb{N} \to 2^{\mathbb{N}}$ being a bijection.[a]
- Consider the set $B = \{k \in \mathbb{N} : k \notin f(k)\} \subseteq \mathbb{N}$.
- Suppose $B = f(n)$ for some $n \in \mathbb{N}$.

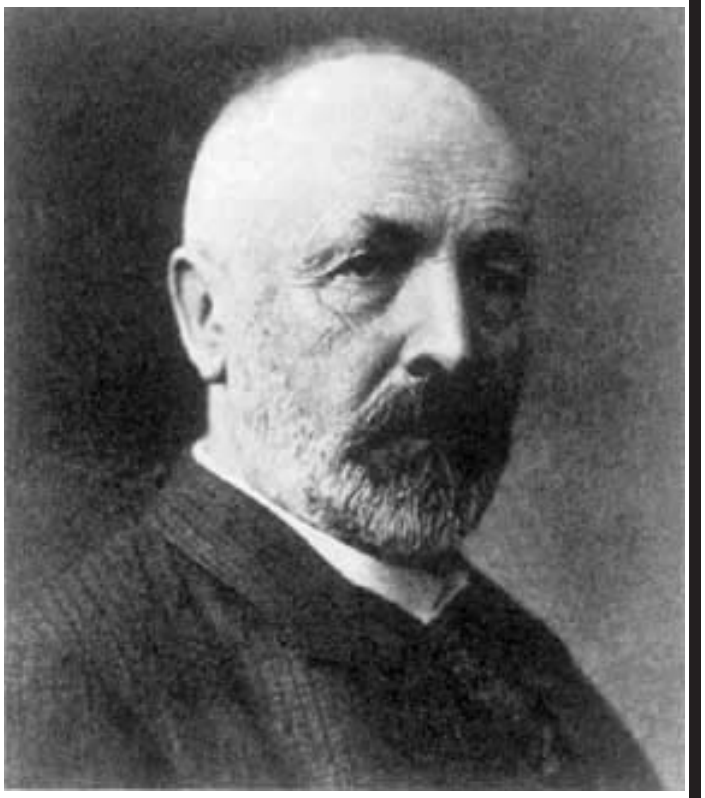
[a] Note that $f(k)$ is a subset of $\mathbb{N}$.

The Proof (concluded)

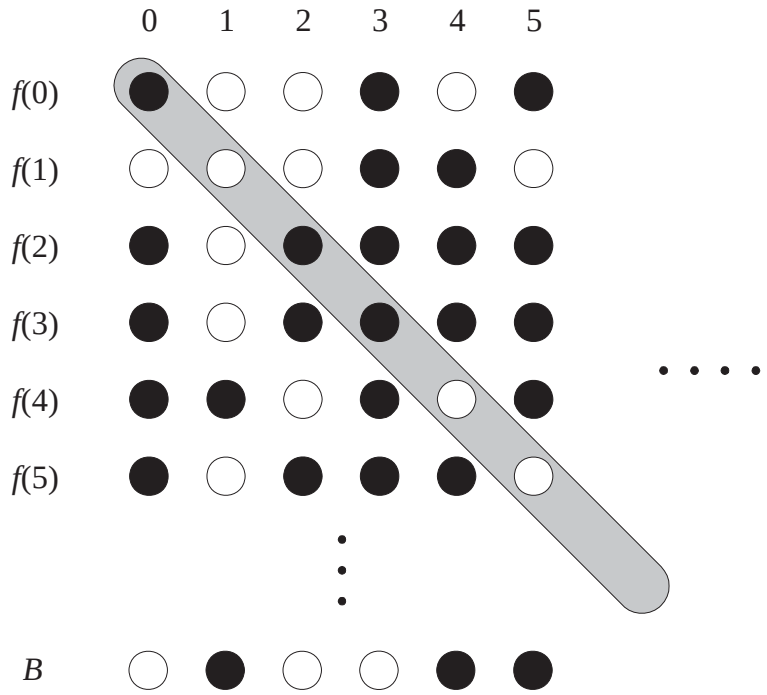
- If $n \in f(n) = B$, then $n \in B$, but then $n \notin B$ by B's definition.
- If $n \notin f(n) = B$, then $n \notin B$, but then $n \in B$ by B's definition.
- Hence $B \neq f(n)$ for any n.
- f is not a bijection, a contradiction.

Georg Cantor (1845–1918)

Kac and Ulam (1968), "[If] one had to name a single person whose work has had the most decisive influence on the present spirit of mathematics, it would almost surely be Georg Cantor."



Cantor's Diagonalization Argument Illustrated



A Corollary of Cantor's Theorem

Corollary 8 *For any set T, finite or infinite,*

$$|\,T\,| < |\,2^T\,|.$$

- The inequality holds in the finite T case as $k < 2^k$.
- Assume T is infinite now.
- $|\,T\,| \leq |2^T|$.
  - Consider $f(x) = \{x\} \in 2^T$.
  - f maps a member of $T = \{\,a, b, c, \ldots\,\}$ to the corresponding member of $\{\,\{\,a\,\}, \{\,b\,\}, \{\,c\,\}, \ldots\,\} \subseteq 2^T$.

The Proof (concluded)

- $|T| \neq |2^T|$.
  - Use the same argument as Cantor's theorem.

A Second Corollary of Cantor's Theorem

Corollary 9 *The set of all functions on* $\mathbb{N}$ *is not countable.*

- It suffices to prove it for functions from $\mathbb{N}$ to $\{0, 1\}$.
- Every function $f : \mathbb{N} \to \{0, 1\}$ determines a subset of $\mathbb{N}$:

$$\{n : f(n) = 1\} \subseteq \mathbb{N},$$

  and vice versa.
- So the set of functions from $\mathbb{N}$ to $\{0, 1\}$ has cardinality $|\, 2^{\mathbb{N}} \,|$.
- Cantor's theorem (p. 144) then implies the claim.

Existence of Uncomputable Problems

- Every program is a finite sequence of 0s and 1s, thus a nonnegative integer.[a]
- Hence every program corresponds to some integer.
- The set of programs is therefore countable.

[a]Different binary strings may be mapped to the same integer (e.g., "001" and "01"). To prevent it, use the lexicographic order as the mapping or simply insert "1" as the most significant bit of the binary string before the mapping (so "001" becomes "1001"). Contributed by Mr. Yu-Chih Tung (R98922167) on October 5, 2010.

Existence of Uncomputable Problems (concluded)

- A function is a mapping from integers to integers.
- The set of functions is not countable by Corollary 9 (p. 150).
- So there are functions for which no programs exist.[a]

[a] As a nondeterministic program may not compute a function, we consider only deterministic programs for this sentence. Contributed by Mr. Patrick Will (`A99725101`) on October 5, 2010.

Universal Turing Machine[a]

- A **universal Turing machine** U interprets the input as the *description* of a TM M concatenated with the *description* of an input to that machine, x.
  - Both M and x are over the alphabet of U.
- U simulates M on x so that

$$U(M; x) = M(x).$$

- U is like a modern computer, which executes any valid machine code, or a Java Virtual machine, which executes any valid bytecode.

[a]Turing (1936).

The Halting Problem

- **Undecidable problems** are problems that have no algorithms.
  - Equivalently, they are languages that are not recursive.
- We knew undecidable problems exist (p. 151).
- We now define a concrete undecidable problem, the **halting problem**:

$$H = \{M; x : M(x) \neq \nearrow\}.$$

  - Does M halt on input x?

H Is Recursively Enumerable

- Use the universal TM U to simulate M on x.
- When M is about to halt, U enters a "yes" state.
- If $M(x)$ diverges, so does U.
- This TM accepts H.

H Is Not Recursive[a]

- Suppose H is recursive.
- Then there is a TM M_H that *decides* H.
- Consider the program $D(M)$ that calls M_H:

```
1: if M_H(M; M) = "yes" then
2:     ↗; {Writing an infinite loop is easy.}
3: else
4:     "yes";
5: end if
```

[a] Turing (1936).

H Is Not Recursive (concluded)

- Consider $D(D)$:
  - $D(D) = \nearrow \Rightarrow M_H(D; D) =$ "yes" $\Rightarrow D; D \in H \Rightarrow$ $D(D) \neq \nearrow$, a contradiction.
  - $D(D) =$ "yes" $\Rightarrow M_H(D; D) =$ "no" $\Rightarrow D; D \notin H \Rightarrow$ $D(D) = \nearrow$, a contradiction.

Comments

- Two levels of interpretations of M:[a]
  - A sequence of 0s and 1s (data).
  - An encoding of instructions (programs).
- There are no paradoxes with $D(D)$.
  - Concepts should be familiar to computer scientists.
  - Feed a C compiler to a C compiler, a Lisp interpreter to a Lisp interpreter, a sorting program to a sorting program, etc.

[a] Eckert and Mauchly (1943); von Neumann (1945); Turing (1946).

Cantor's Paradox (1899[a])

- Let T be the set of all sets.[b]
- Then $2^T \subseteq T$ because 2^T is a set.
- But we know[c] $|\,2^T\,| > |\,T\,|$ (p. 148)!
- We got a "contradiction."
- Are we willing to give up Cantor's theorem?
- If not, what is a set?

---

[a] In a letter to Richard Dedekind. First published in Russell (1903).

[b] Recall this ontological argument for the existence of God by St Anselm (1033–1109) in the 11th century: If something is possible but is not part of God, then God is not the greatest possible object of thought, a contradiction.

[c] Really?

Self-Loop Paradoxes[a]

Russell's Paradox (1901): Consider $R = \{A : A \notin A\}$.

- If $R \in R$, then $R \notin R$ by definition.
- If $R \notin R$, then $R \in R$ also by definition.
- In either case, we have a "contradiction."[b]

Eubulides: The Cretan says, "All Cretans are liars."

Liar's Paradox: "This sentence is false."

Hypochondriac: a patient (like Gödel) with imaginary symptoms and ailments.

---

[a] E.g., Hofstadter, *Gödel, Escher, Bach: An Eternal Golden Braid* (1979) or Quine, *The Ways of Paradox and Other Essays* (1966).

[b] Gottlob Frege (1848–1925) to Bertrand Russell in 1902, "Your discovery of the contradiction [...] has shaken the basis on which I intended to build arithmetic."

Self-Loop Paradoxes (continued)

Sharon Stone in *The Specialist* (1994): "I'm not a woman you can trust."

***Spin City* (1996–2002):** "I am not gay, but my boyfriend is."

***Numbers* 12:3, Old Testament:** "Moses was the most humble person in all the world [$\cdots$]" (attributed to Moses).

Self-Loop Paradoxes (concluded)

***The Egyptian Book of the Dead*:** "ye live in me and I would live in you."

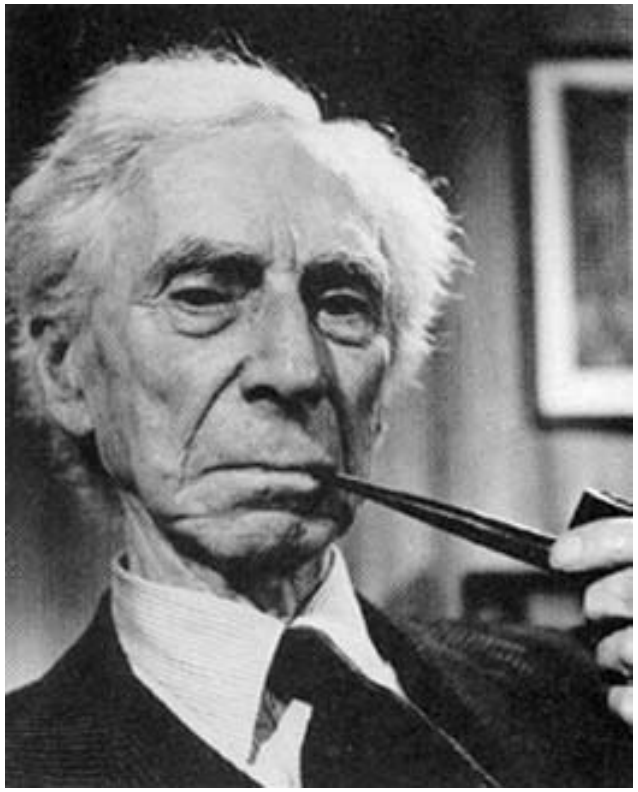
***John* 17:21, New Testament:** "just as you are in me and I am in you."

***Pagan & Christian Creeds* (1920):** "I was moved to Odin, myself to myself."

Soren Kierkegaard in *Fear and Trembling* (1843): "to strive against the whole world is a comfort, to strive with oneself is dreadful."

Bertrand Russell (1872–1970)

Karl Popper (1974), "perhaps the greatest philosopher since Kant."



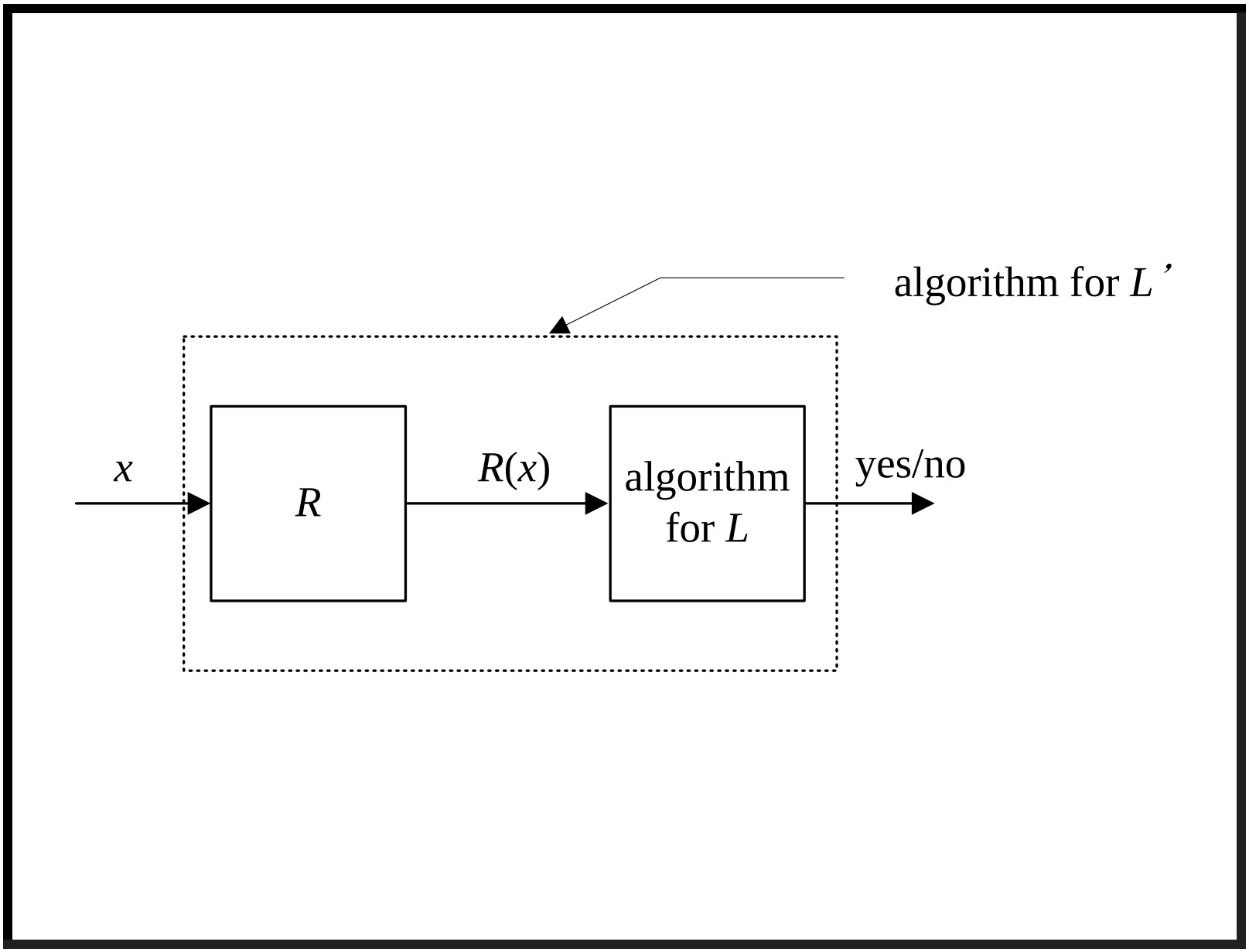
Reductions in Proving Undecidability

- Suppose we are asked to prove that L is undecidable.
- Suppose L' (such as H) is known to be undecidable.
- Find a computable transformation R (called **reduction**) from L' to L such that[a]

$$\forall x \ \{x \in L' \text{ if and only if } R(x) \in L\}.$$

- Now we can answer "$x \in L'$?" for *any* x by asking "$R(x) \in L$?" because they have the same answer.
  - L' is said to be **reduced** to L.

[a] Contributed by Mr. Tai-Dai Chou (J93922005) on May 19, 2005.


algorithm for L'
x
R
$R(x)$
algorithm for L
yes/no

Reductions in Proving Undecidability (concluded)

- If L were decidable, "$R(x) \in L$?" becomes computable and we have an algorithm to decide L', a contradiction!
- So L must be undecidable.

Theorem 10 *Suppose language L_1 can be reduced to language L_2. If L_1 is not recursive, then L_2 is not recursive.*

More Undecidability

- $H^* = \{M : M \text{ halts on all inputs}\}$.
  - We will reduce H to H^*.
  - Given the question "$M; x \in H$?", construct the following machine (this is the reduction):[a]

$$M_x(y)\ \{M(x);\}$$

  - M halts on x if and only if M_x halts on all inputs.
  - In other words, $M; x \in H$ if and only if $M_x \in H^*$.
  - So if H^* were recursive (recall the box for L on p. 165), H would be recursive, a contradiction.

[a] Simplified by Mr. Chih-Hung Hsieh (D95922003) on October 5, 2006. M_x ignores its input y; x is part of M_x's code but not M_x's input.

More Undecidability (concluded)

- $\{M; x : \text{there is a } y \text{ such that } M(x) = y\}$.
- $\{M; x : \text{the computation } M \text{ on input } x \text{ uses all states of } M\}$.
- $\{M; x; y : M(x) = y\}$.

Complements of Recursive Languages

The **complement** of L, denoted by $\bar{L}$, is the language $\Sigma^* - L$.

Lemma 11 *If L is recursive, then so is $\bar{L}$.*

- Let L be decided by M, which is deterministic.
- Swap the "yes" state and the "no" state of M.
- The new machine decides $\bar{L}$.[a]

[a] Recall p. 111.

Recursive and Recursively Enumerable Languages

Lemma 12 (Kleene's theorem) *L is recursive if and only if both L and $\bar{L}$ are recursively enumerable.*

- Suppose both L and $\bar{L}$ are recursively enumerable, accepted by M and $\bar{M}$, respectively.
- Simulate M and $\bar{M}$ in an *interleaved* fashion.
- If M accepts, then halt on state "yes" because $x \in L$.
- If $\bar{M}$ accepts, then halt on state "no" because $x \notin L$.
- Note that either M or $\bar{M}$ (but not both) must accept the input and halt.

A Very Useful Corollary and Its Consequences

Corollary 13 *L is recursively enumerable but not recursive, then $\bar{L}$ is not recursively enumerable.*

- Suppose $\bar{L}$ is recursively enumerable.
- Then both L and $\bar{L}$ are recursively enumerable.
- By Lemma 12 (p. 170), L is recursive, a contradiction.

Corollary 14 *$\bar{H}$ is not recursively enumerable.*[a]

[a] Recall that $\bar{H} = \{M; x : M(x) = \nearrow\}$.

R, RE, and coRE

RE: The set of all recursively enumerable languages.

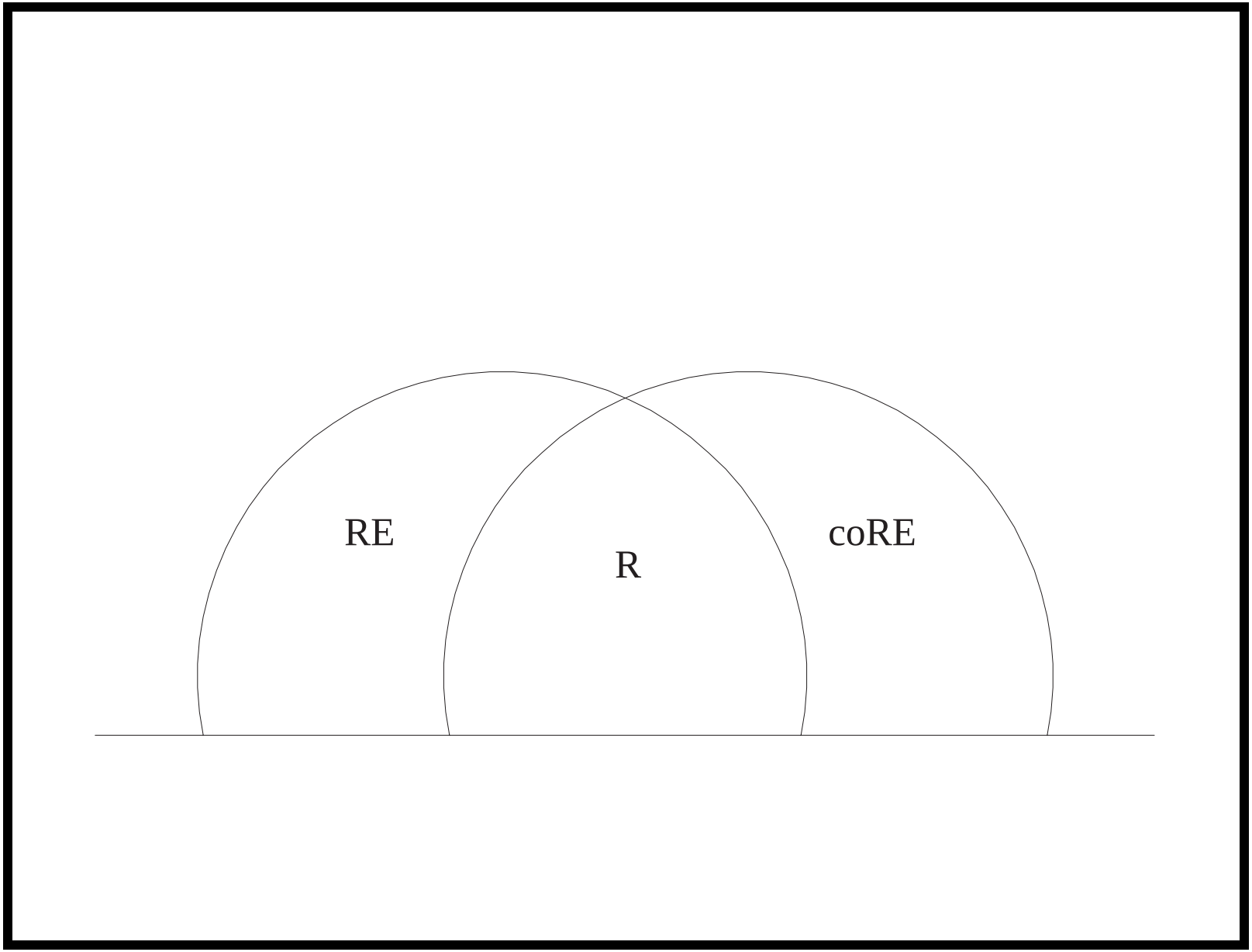
coRE: The set of all languages whose complements are recursively enumerable.

R: The set of all recursive languages.

- Note that coRE is not $\overline{\text{RE}}$.
  - $\text{coRE} = \{ L : \overline{L} \in \text{RE} \} = \{ \overline{L} : L \in \text{RE} \}$.
  - $\overline{\text{RE}} = \{ L : L \notin \text{RE} \}$.

R, RE, and coRE (concluded)

- $\mathrm{R} = \mathrm{RE} \cap \mathrm{coRE}$ (p. 170).
- There exist languages in RE but not in R and not in coRE.
  - Such as H (p. 155, p. 156, and p. 171).
- There are languages in coRE but not in RE.
  - Such as $\bar{H}$ (p. 171).
- There are languages in neither RE nor coRE.


RE
R
coRE

Undecidability in Logic and Mathematics

- First-order logic is undecidable (answer to Hilbert's "*Entscheidungsproblem*" (1928)).[a]
- Natural numbers with addition and multiplication is undecidable.[b]
- Rational numbers with addition and multiplication is undecidable.[c]

---

[a] Church (1936).
[b] Rosser (1937).
[c] Robinson (1948).

Undecidability in Logic and Mathematics (concluded)

- Natural numbers with addition and equality is decidable and complete.[a]
- Elementary theory of groups is undecidable.[b]

---

[a] Presburger's Master's thesis (1928), his only work in logic. The direction was suggested by Tarski. Mojżesz Presburger (1904–1943) died in a concentration camp during World War II.

[b] Tarski (1949).

Julia Hall Bowman Robinson (1919–1985)

Alfred Tarski (1901–1983)

Boolean Logic

It seemed unworthy of a grown man
to spend his time on such trivialities,
but what was I to do? $[\cdots]$
The whole of the rest of my life might be
consumed in looking at
that blank sheet of paper.
— Bertrand Russell (1872–1970),
Autobiography, Vol. I (1967)

Boolean Logic[a]

Boolean variables: $x_1, x_2, \ldots$.

Literals: x_i, $\neg x_i$.

Boolean connectives: $\vee, \wedge, \neg$.

Boolean expressions: Boolean variables, $\neg\phi$ (**negation**), $\phi_1 \vee \phi_2$ (**disjunction**), $\phi_1 \wedge \phi_2$ (**conjunction**).

- $\bigvee_{i=1}^{n} \phi_i$ stands for $\phi_1 \vee \phi_2 \vee \cdots \vee \phi_n$.
- $\bigwedge_{i=1}^{n} \phi_i$ stands for $\phi_1 \wedge \phi_2 \wedge \cdots \wedge \phi_n$.

Implications: $\phi_1 \Rightarrow \phi_2$ is a shorthand for $\neg\phi_1 \vee \phi_2$.

Biconditionals: $\phi_1 \Leftrightarrow \phi_2$ is a shorthand for $(\phi_1 \Rightarrow \phi_2) \wedge (\phi_2 \Rightarrow \phi_1)$.

---

[a] George Boole (1815–1864) in 1847.

Truth Assignments

- A **truth assignment** T is a mapping from boolean variables to **truth values** `true` and `false`.
- A truth assignment is **appropriate** to boolean expression ϕ if it defines the truth value for every variable in ϕ.
  - $\{x_1 = \texttt{true}, x_2 = \texttt{false}\}$ is appropriate to $x_1 \vee x_2$.
  - $\{x_2 = \texttt{true}, x_3 = \texttt{false}\}$ is not appropriate to $x_1 \vee x_2$.

Satisfaction

- $T \models \phi$ means boolean expression ϕ is true under T; in other words, T **satisfies** ϕ.
- ϕ_1 and ϕ_2 are **equivalent**, written

  $$\phi_1 \equiv \phi_2,$$

  if for any truth assignment T appropriate to both of them, $T \models \phi_1$ if and only if $T \models \phi_2$.

Truth Tables

- Suppose ϕ has n boolean variables.
- A **truth table** contains 2^n rows.
- Each row corresponds to one truth assignment of the n variables and records the truth value of ϕ under that truth assignment.
- A truth table can be used to prove if two boolean expressions are equivalent.
  - Just check if they give identical truth values under all appropriate truth assignments.

A Truth Table

p	q	$p \wedge q$
0	0	0
0	1	0
1	0	0
1	1	1

A Second Truth Table

p	q	$p \vee q$
0	0	0
0	1	1
1	0	1
1	1	1

A Third Truth Table

p	$\neg p$
0	1
1	0

De Morgan's Laws[a]

- **De Morgan's laws** say that

$$\begin{aligned} \neg(\phi_1 \wedge \phi_2) &\equiv \neg\phi_1 \vee \neg\phi_2, \\ \neg(\phi_1 \vee \phi_2) &\equiv \neg\phi_1 \wedge \neg\phi_2. \end{aligned}$$

- Here is a proof of the first law:

ϕ_1	ϕ_2	$\neg(\phi_1 \wedge \phi_2)$	$\neg\phi_1 \vee \neg\phi_2$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

[a] Augustus DeMorgan (1806–1871) or William of Ockham (1288–1348).

Conjunctive Normal Forms

- A boolean expression ϕ is in **conjunctive normal form** (**CNF**) if

$$\phi = \bigwedge_{i=1}^{n} C_i,$$

  where each **clause** C_i is the disjunction of zero or more literals.[a]

  – For example,

$$(x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (x_2 \vee x_3).$$

- Convention: An empty CNF is satisfiable, but a CNF containing an empty clause is not.

[a] Improved by Mr. Aufbu Huang (R95922070) on October 5, 2006.

Disjunctive Normal Forms

- A boolean expression ϕ is in **disjunctive normal form** (**DNF**) if

$$\phi = \bigvee_{i=1}^{n} D_i,$$

  where each **implicant** D_i is the conjunction of zero or more literals.

  – For example,

$$(x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2) \vee (x_2 \wedge x_3).$$

Clauses and Implicants

- The $\bigvee$ of clauses remains a clause.
  - For example,

$$\begin{aligned} & (x_1 \vee x_2) \vee (x_1 \vee \neg x_2) \vee (x_2 \vee x_3) \\ = \quad & x_1 \vee x_2 \vee x_1 \vee \neg x_2 \vee x_2 \vee x_3. \end{aligned}$$

- The $\bigwedge$ of implicants remains an implicant.
  - For example,

$$\begin{aligned} & (x_1 \wedge x_2) \wedge (x_1 \wedge \neg x_2) \wedge (x_2 \wedge x_3) \\ = \quad & x_1 \wedge x_2 \wedge x_1 \wedge \neg x_2 \wedge x_2 \wedge x_3. \end{aligned}$$

Any Expression ϕ Can Be Converted into CNFs and DNFs

$\phi = x_j$**:**

- This is trivially true.

$\phi = \neg\phi_1$ **and a CNF is sought:**

- Turn ϕ_1 into a DNF.
- Apply de Morgan's laws to make a CNF for ϕ.

$\phi = \neg\phi_1$ **and a DNF is sought:**

- Turn ϕ_1 into a CNF.
- Apply de Morgan's laws to make a DNF for ϕ.

Any Expression ϕ Can Be Converted into CNFs and DNFs (continued)

$\phi = \phi_1 \vee \phi_2$ **and a DNF is sought:**

- Make ϕ_1 and ϕ_2 DNFs.

$\phi = \phi_1 \vee \phi_2$ **and a CNF is sought:**

- Turn ϕ_1 and ϕ_2 into CNFs,[a]

$$\phi_1 = \bigwedge_{i=1}^{n_1} A_i, \quad \phi_2 = \bigwedge_{j=1}^{n_2} B_j.$$

- Set

$$\phi = \bigwedge_{i=1}^{n_1} \bigwedge_{j=1}^{n_2} (A_i \vee B_j).$$

[a] Corrected by Mr. Chun-Jie Yang (R99922150) on November 9, 2010.

Any Expression ϕ Can Be Converted into CNFs and DNFs (concluded)

$\phi = \phi_1 \wedge \phi_2$ **and a CNF is sought:**

- Make ϕ_1 and ϕ_2 CNFs.

$\phi = \phi_1 \wedge \phi_2$ **and a DNF is sought:**

- Turn ϕ_1 and ϕ_2 into DNFs,

$$\phi_1 = \bigvee_{i=1}^{n_1} A_i, \quad \phi_2 = \bigvee_{j=1}^{n_2} B_j.$$

- Set

$$\phi = \bigvee_{i=1}^{n_1} \bigvee_{j=1}^{n_2} (A_i \wedge B_j).$$

An Example: Turn $\neg((a \wedge y) \vee (z \vee w))$ into a DNF

$$
\begin{aligned}
& \neg((a \wedge y) \vee (z \vee w)) \\
\overset{\neg(\text{CNF}\vee\text{CNF})}{=} \quad & \neg(((a) \wedge (y)) \vee ((z \vee w))) \\
\overset{\neg(\text{CNF})}{=} \quad & \neg((a \vee z \vee w) \wedge (y \vee z \vee w)) \\
\overset{\text{de Morgan}}{=} \quad & \neg(a \vee z \vee w) \vee \neg(y \vee z \vee w) \\
\overset{\text{de Morgan}}{=} \quad & (\neg a \wedge \neg z \wedge \neg w) \vee (\neg y \wedge \neg z \wedge \neg w).
\end{aligned}
$$

Satisfiability

- A boolean expression ϕ is **satisfiable** if there is a truth assignment T appropriate to it such that $T \models \phi$.
- ϕ is **valid** or a **tautology**,[a] written $\models \phi$, if $T \models \phi$ for all T appropriate to ϕ.
- ϕ is **unsatisfiable** if and only if ϕ is false under all appropriate truth assignments if and only if $\neg\phi$ is valid.

[a] Wittgenstein (1889–1951) in 1922. Wittgenstein is one of the most important philosophers of all time. "God has arrived," the great economist Keynes (1883–1946) said of him on January 18, 1928. "I met him on the 5:15 train." Russell (1919), "The importance of 'tautology' for a definition of mathematics was pointed out to me by my former pupil Ludwig Wittgenstein, who was working on the problem. I do not know whether he has solved it, or even whether he is alive or dead."

Ludwig Wittgenstein (1889–1951)

Wittgenstein (1922), "Whereof one cannot speak, thereof one must be silent."