

Decidability

Prof. (Dr.) K.R. Chowdhary
Email: kr.chowdhary@gmail.com

Formerly at department of Computer Science and Engineering
MBM Engineering College, Jodhpur

Thursday 13th November, 2025

K.R. Chowdhary

Theory of Computation

Automata, Formal Languages, Computation and Complexity

Focuses on pedagogy in its writing, that represents a refreshing approach

Ensures comprehensive and enjoyable learning

Undergone a rigorous classroom testing



© 2025

Get 20% off with this code: **SPRAUT**

Available on Springer Nature Link

[link.springer.com/book/
9789819762347](https://link.springer.com/book/9789819762347)

Please note that promotional coupons are only valid for English-language Springer, Apress, and Palgrave Macmillan books & eBooks and are redeemable on link.springer.com only. Titles affected by flood book price laws, forthcoming titles and titles temporarily not available on Springer Nature Link are excluded from promotions, as are reference works, handbooks, encyclopedias, subscriptions, or bulk purchases. The currency in which your order will be invoiced depends on the billing address associated with the payment method used, not necessarily your home currency. Regional VAT/tax may apply. Promotional prices may change due to exchange rates. Promotions are valid for individual customers only. Bookellers, book distributors, and institutions such as libraries and corporations, please visit springernature.com/contact-us. Promotions do not work in combination with other discounts or gift cards.

Decision Problems

We may be interested in following questions:

“Is a number number perfect square?”

“Is number prime?”

“Does a graph has cycle?”

“Does the computation of TM halt before 25th transition?”

Each of these general question describe a decision problem. A decision problem P is a set of *related questions* p_i , each of which has es/no answer. For example:

p_0 : Is 0 a perfect square?

p_1 Is 1 a perfect square?

p_2 is 2 a perfect square?

...

Each of the p_i is an instance of the problem P . The solution of a decision problem P is an algorithm that determines the answer of every question $p \in P$. A decision problem is decidable, if it has a solution.

An algorithm that solves decision problem should be:

- Complete: correct answer is given for every problem instance
- Mechanistic: finite sequence of instructions, each can be carried out without requirement of insight, ingenuity, or guesswork.

- Deterministic: With identical input, the same computation is carried.

A procedure having the properties of complete, Mechanistic, and deterministic, is called *effective procedure*. A standard TM is an effective algorithm if it is, Mechanistic, deterministic, and complete. However, it is complete only if, it halts on every input.

Decidable languages

Decidable language-3

- $A_{EQDFA} = \{ \langle A, B \rangle \mid A \text{ and } B \text{ are DFAs and } L(A) = L(B) \}$
- Equivalence problem: Test whether two DFAs recognize the same language.
- **Theorem:** A_{EQDFA} is decidable languages
- $F =$ “On input $\langle A, B \rangle$, where A and B are DFAs:
 - 1 construct DFA $L(C) = (L(A) \cap \overline{L(B)}) \cup (L(B) \cap \overline{L(A)})$
($A = B \Rightarrow C = \phi$)
 - 2 Run $TM T$ for deciding A_{EQDFA} on input $\langle C \rangle$
 - 3 if T accepts, **Accept**; otherwise **reject**. ”

Acceptance problem: A_{TM}

Input: A TM 's description $\langle M \rangle$ and a string w for input to M .

Output: Yes/No indicating if M eventually enters q_{accept} on input w .

- Acceptance of language consisting of tuples: $\langle M, w \rangle$,
 $A_{TM} = \{ \langle M, w \rangle \mid M \text{ is a Turing description and } M \text{ accepts input } w \}$.
Is A_{TM} Turing recognizable?
- **Defn. Turing Recognizable:** A language A_{TM} is “Turing recognizable” if there exists a $TM M$ such that for all w :
 - If $\langle M, w \rangle \in A_{TM}$ then M eventually enter q_{accept}
 - If $\langle M, w \rangle \notin A_{TM}$ then M eventually enter q_{reject} or M loops for ever.

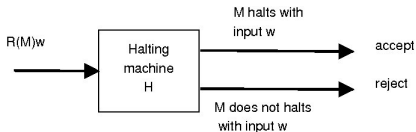
Halting Problem

- **Theorem:** A_{TM} is Turing recognizable.
- $U =$ “On input $\langle M, w \rangle$, where M is a TM and w is a string:
 - 1 Simulate M on input w
 - 2 If M ever enters accepts state, then **U accepts**; if M enters its reject state, **U rejects**”
 - U is universal TM
 - U keeps looping if M neither accepts or rejects
 - However, if $U \equiv M, A_{TM}$ is unsolvable (**i.e., undecidable**)

- A problem is **decidable** if some TM decides (solves) it.

Halting problem: Given a TM M and input string $\langle W, \langle m \rangle \rangle$, decide whether M halt on $\langle \langle M \rangle, w \rangle$?

- **The instance of the problem is :** $e_n(M)e_n(w)$. Halting = $\{e_n(M)e(w) | M \text{ halts on } w\}$ is *not recursive*.



Undecidability of A_{TM}

Proof by contradiction:

- Assume that \exists some $TM H$ that decides A_{TM} . That is, H accepts if M accepts w , and H rejects if M rejects w .
- Now we construct a $TM D$ with H as subroutine. This calls H and determine what M does when the input to M is its own description $\langle M \rangle$. However, after determining this, it outputs the opposite. That is, it rejects if M accepts, and vice-versa. Call this as H' .
- Define $D(\langle M \rangle) =$
 - 1 Construct a $TM D$ (having input $\langle M \rangle$) that outputs the opposite of the result of simulating H on input $\langle \langle M \rangle, M \rangle$.
 - 2 Output the opposite of what H outputs, i.e., if H accepts, then reject, and if H rejects, then accept.
- The above can be rewritten as:
- If M accepts its own description $\langle M \rangle$, then
 $H(\langle M \rangle)$ accepts and $\therefore D(\langle M \rangle)$ rejects
- If M rejects its own description $\langle M \rangle$, then
 $H(\langle M \rangle)$ rejects and $\therefore D(\langle M \rangle)$ accepts
- What happens if we run D on its own description $\langle D \rangle$?
- From above: (substitute D for M), we have (see next slide)

Proving Undecidability of A_{TM}

If D accepts $\langle D \rangle$:

$H(D, \langle D \rangle)$ accepts and $D \langle D \rangle$ rejects

If D rejects $\langle D \rangle$:

$H(D, \langle D \rangle)$ rejects and $D \langle D \rangle$ accepts

- Which can be further simplified:

If D accepts $\langle D \rangle$:

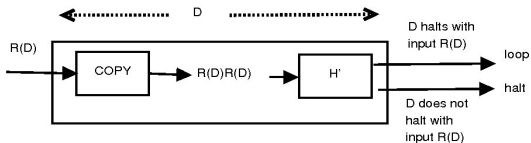
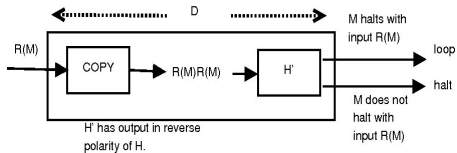
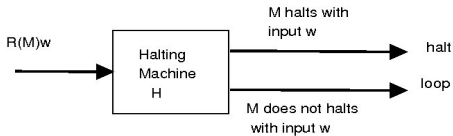
$D \langle D \rangle$ rejects

If D rejects $\langle D \rangle$:

$D \langle D \rangle$ accepts

- Hence, whatever is done, it must do the opposite. So there is a contradiction. So, D cannot exist. But, if H exists, we know how to make D . H cannot exist; so there is no TM that decides A_{TM} .

Proving Undecidability of A_{TM}



if D halts with input $R(D)$ then loop
if D does not halt with input $R(D)$ then halt

A different approach for A_{TM} as undecidable

- Preceding proof uses self-reference and diagonalization.
- To obtain table for diagonalization argument, consider that every $v \in \{0,1\}^*$ represent a TM . If v does not have form $R(M)$, a one-state TM with no transition is assigned to v . Thus, TMs can be listed as $M_0, M_1, M_2, M_3, M_4, M_5, M_6, M_7, \dots$ corresponding to $\epsilon, 0, 1, 00, 01, 10, 11, 000$.
- Consider a table that lists TMs along the horizontal and vertical axes. The i, j th entry in table is:

$$\begin{cases} 1 & \text{if } M_i \text{ halts when run with input } R(M_j) \\ 0 & \text{if } M_i \text{ does not halt when run with input } R(M_j) \end{cases}$$

Diagonal elements are answers to the self-referential questions: Does M_i halt when run with itself?

Undecidable, decidable, recognizable, Unrecognizable:

- A_{CFG} is decidable
- A_{TM} is undecidable
- $L \in P(\Gamma^*)$ is unrecognizable, where $P(\Gamma^*)$ is uncountable
- $A_{TM}^- = \{w \mid M \text{ is a TM and } M \text{ does not accept } w\}$



Chowdhary, K.R. (2025). Decidability, Undecidability, and Unsolvability. In: Theory of Computation. Springer, Singapore.
https://doi.org/10.1007/978-981-97-6234-7_12