Decidability

Linz 6th, Chapter 12: Limits of Algorithmic Computation, page 309ff
A property $P$ of strings is said to be \textit{decidable} if the set of all strings having property $P$ is a recursive set; that is, if there is a total Turing machine that accepts input strings that have property $P$ and rejects those that do not.

\[ P \text{ is decidable } \iff \{x \mid P(x)\} \text{ is recursive.} \]

\[ A \text{ is recursive } \iff \text{“}x \in A\text{” is decidable,} \]
Consider problems with answer YES or NO

Examples:

• Does Machine $M$ have three states?
• Is string $w$ a binary number?
• Does DFA $M$ accept any input?
A problem is decidable if some Turing machine solves (decides) the problem.

Decidable problems:

- Does Machine $M$ have three states?
- Is string $w$ a binary number?
- Does DFA $M$ accept any input?
The Turing machine that solves a problem answers **YES** or **NO** for each instance.

Input problem instance → Turing Machine → **YES**

Input problem instance → Turing Machine → **NO**
The machine that decides a problem:

• If the answer is YES then halts in a yes state

• If the answer is NO then halts in a no state

These states may not be final states
Turing Machine that decides a problem

YES and NO states are halting states
Difference between Recursive Languages and Decidable problems

For decidable problems:

The YES states may not be final states
[What does the author mean?]

No harm in assuming YES states are final.

Decidable = Recursive
Some problems are undecidable:

which means:

there is no Turing Machine that solves all instances of the problem
A simple undecidable problem:

The membership problem
The Membership Problem

Input:  
• Turing Machine $M$, and
• String $w$

Question:  
Does $M$ accept $w$?  

$w \in L(M)$?
Theorem: The membership problem is undecidable

(there are $M$ and $w$ for which we cannot decide whether $w \in L(M)$ )

Proof: Assume for contradiction that the membership problem is decidable
Assume there exists a Turing Machine $H$ that decides/solves the membership problem.

$M \xrightarrow{w} H$  

$H \xrightarrow{\text{YES}} M \text{ accepts } w$

$H \xrightarrow{\text{NO}} M \text{ rejects } w$
Let $L$ be a recursively enumerable language.

Let $M$ be the Turing Machine that accepts $L$.

We will prove that $L$ is also recursive: we will describe a Turing machine that accepts $L$ and halts on any input.
Turing Machine that accepts $L$ and halts on any input

$M$ accepts $w$?

- YES $\Rightarrow$ accept $w$
- NO $\Rightarrow$ reject $w$
Therefore, $L$ is recursive.

Since $L$ is chosen arbitrarily, we have proven that every recursively enumerable language is also recursive.

But there are recursively enumerable languages which are not recursive.

Contradiction!!!!
Therefore, the membership problem is undecidable

END OF PROOF
Another famous undecidable problem:

The halting problem
ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. Turing.

[Received 28 May, 1936.—Read 12 November, 1936.]

The "computable" numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. Although the subject of this paper is ostensibly the computable numbers,
The Halting Problem

Input: • Turing Machine $M$, and
      • String $w$

Question: Does $M$ halt on input $w$?
Theorem:
The halting problem is undecidable
(there are \( M \) and \( w \) for which we cannot decide whether \( M \) halts on input \( w \) )

Proof: Assume for contradiction that the halting problem is decidable
Thus, there exists Turing Machine $H$ that solves the halting problem.

$M$ \rightarrow \quad YES \quad \rightarrow \quad M$ halts on $w$

$w$ \rightarrow \quad NO \quad \rightarrow \quad M$ doesn't halt on $w$
Construction of $H$

Input: initial tape contents

$w_M$ $w$

Encoding of $M$ String $w$

$H$

$q_0$

$q_y$ YES

$q_n$ NO
Construct machine $H'$:

If $H$ returns YES then loop forever

If $H$ returns NO then halt
$H'$

$H$

$w_M \xrightarrow{w} q_0 \xrightarrow{\text{YES}} q_y \xrightarrow{\text{YES}} q_a \xrightarrow{\text{Loop forever}} q_b \xrightarrow{\text{NO}} q_n \xrightarrow{\text{NO}} q_0$
Construct machine $\hat{H}$:

Input: $w_M$ (machine $M$)

If $M$ halts on input $w_M$

Then loop forever

Else halt
Run machine \( \hat{H} \) with input itself:

Input: \( w_{\hat{H}} \) (machine \( \hat{H} \) )

If \( \hat{H} \) halts on input \( w_{\hat{H}} \)

Then loop forever

Else halt
\( \hat{H} \) on input \( w_{\hat{H}} \):

If \( \hat{H} \) halts then loops forever

If \( \hat{H} \) doesn't halt then it halts

NONSENSE !!!!!
Therefore, we have contradiction

The halting problem is undecidable

END OF PROOF
Another proof of the same theorem:

If the halting problem were decidable then every recursively enumerable language would be recursive.

If $L = \{M_w \# w \mid M_w \text{ halts on } w\}$ were recursive, then every r.e. set is recursive.
Theorem:

The halting problem is undecidable

Proof: Assume for contradiction that the halting problem is decidable
Thus, there exists Turing Machine $H$ that solves the halting problem.

$M$ \rightarrow $H$ \rightarrow YES \rightarrow $M$ halts on $w$

$w$ \rightarrow $H$ \rightarrow NO \rightarrow $M$ doesn't halt on $w$
Let $L$ be a recursively enumerable language.

Let $M$ be the Turing Machine that accepts $L$.

We will describe a Turing machine that accepts $L$ and halts on any input, proving that $L$ is also recursive.
Turing Machine that accepts $L$ and halts on any input

\[ M \:] \quad \text{halts on } w? \]

- Run $M$ with input $w$
  - Halts on final state: accept $w$
  - Halts on non-final state: reject $w$

- $M$ halts on $w$?
  - NO: reject $w$
  - YES: accept $w$
Therefore \( L \) is recursive

Since \( L \) is chosen arbitrarily, we have proven that every recursively enumerable language is also recursive

But there are recursively enumerable languages which are not recursive

Contradiction!!!!
Therefore, the halting problem is undecidable