Theoretische Informatik

Laurin Seeholzer

September 15, 2026

Contents

1 Introduction

Leibnitz proposed that some time in the future, mathematics will be developed enough to model and solve every possible problem. Then came Russel

Definition 1.1. Russels Paradox: Let S be the set of sets that do not contain themselves.

$$S = X | X \notin X$$

Russels Paradox poses the following Question: Does S contain intself. The answer is paradoxical

$$S \notin S \implies S \in S$$

$$S \in S \implies S \notin S$$

Remark 1.2. Hilbert style deduction allows you to solve Russels paradox by first stating what the axioms and interference rules of the System are.

Theorem 1.3. Goedels incompletness: There are mathematical statements that are true but can not possibly be proven to be so.

Definition 1.4. Turing-Machine (TM): A theoretical machine consisting of an infinity strand of paper, a perfect pen and a set of rules. Every step includes reading current position, modifying it, moving left/right.

Remark 1.5. It can be shown that every algorithm can be modeld as a Turing-Machine and vice-versa.

Theorem 1.6. Turing-Theorem: There are problems that can not be automatically solved using an algorithm/TM.

Proof. Let $w_{i,1}, w_{i,2}, w_{i,3} \dots$ be the output of Turing Machine i for inputs $1, 2, 3 \dots$. We define a new Turing Machine M_x such that $w_{x,j} \neq w_{j,j}$ hence the output of M_x differs from the output of every other Turing machine for at least one input. Observe that this is a valid Turing Machine but it can not possibly be contained in our current list of Turing Machines. Hence, we reach a contradiction: if M_x were in our enumeration at some index x, its output on input x would be $w_{x,x} \neq w_{x,x}$, which is logically impossible. Therefore, our assumption that we can construct such a machine M_x must be false. This implies there is no computable general algorithm capable of determining the output $w_{j,j}$ for all Turing machines j, proving that there exist uncomputable functions and that the Halting Problem is undecidable. □

Example 1.7. Halting-problem: Does algorithm A terminate?

Definition 1.8. P vs. NP: The p/np problem poses the question if all in polynomial time verifiable problems (NP) have a polynomial time solution (P).