Hilbert’s Problems

- In 1900 delivered address to International Congress of Mathematicians, identified 23 problems to (try to) solve in the next century.

- 10th problem:

  Devise an algorithm that tests whether a polynomial with integral coefficients has an integral root.

\[
\begin{align*}
  x^2 - 2x + 1 & \quad \text{has root } x = 1 \\
  x^2 - x + 1 & \quad \text{no integral roots} \\
  5x^2 + 6x + 1 & \quad \text{no roots}
\end{align*}
\]
Defining Algorithm

Hilbert's Problems

- In 1900 delivered address to International Congress of Mathematicians, identified 23 problems to (try to) solve in the next century.

- 10th problem:
  
  Devise an algorithm that tests whether a polynomial with integral coefficients has an integral root.

Today we know: no algorithm exists !!!
In 1936:
- Alonzo Church defines algorithms via $\lambda$-calculus
- Alan Turing defines algorithms via (decidable) TM’s

**Church-Turing thesis:**

Intuitive notions of algorithms = TM algorithms

In 1970, Yuri Matijasevič (building on work of Martin Davis, Hilary Putnam, and Julia Robinson):

There is no algorithm for Hilbert’s 10th problem.
Hilbert’s 10th problem in our terminology:

\[ D = \{ p \mid p \text{ is a polynomial with an integral root} \}, \]

where \( p \in \{0,1,\ldots,9,x,+,-,\}^*. \)

\[ \text{e.g. } p = x^{25} + 9x^{15} + 7 \]

Is it Turing-recognizable?

**YES:**
- try all possible roots:
  - 0, 1, -1, 2, -2, 3, -3, 
  - if one evaluates to 0, then accept
- otherwise keep going

Hilbert’s 10th problem is asking whether this language is T-decidable?

**NO**
(see previous slide)
A typical CS problem phrased in formal-languages way:

\[ E = \{ [x] \mid x \text{ satisfies required properties} \}, \]

where \([x]\) is a string encoding of the input.

\(\langle x \rangle\) \text{ - encoding of the input}

\textbf{Is it Turing-recognizable?}

\textbf{E.g.} \(X\) - a graph and two vertices s.t \text{property: is t reachable from s?}