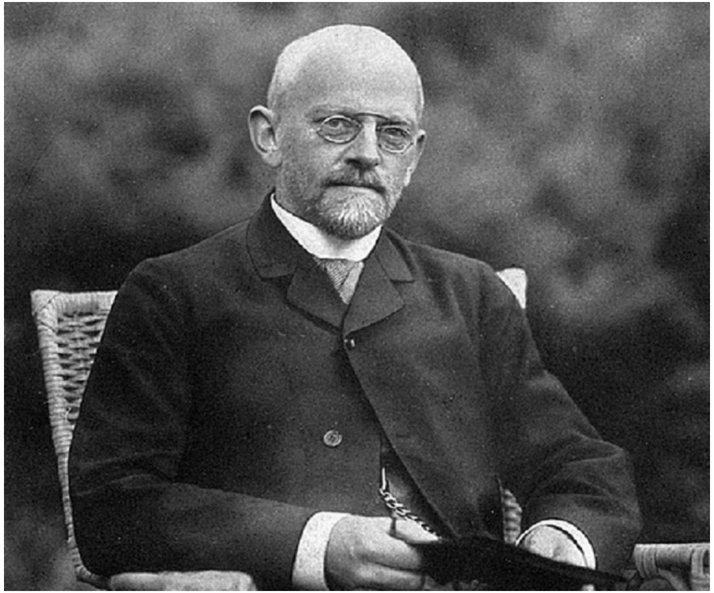


Entscheidungsproblem

Expressing problems and solutions

Problem:

67 * 4.5 = ?

Answer:

67 * 4.5 = 301.5

Problem:

Given two numbers, x and y,
what is (x * y)?

Answer:

?

To express the answer to the multiplication problem, we need to work with a more generic form of solutions: *formal languages*

Problems and instances

Problem:

67 * 4.5 = ?

An instance of the MULT problem

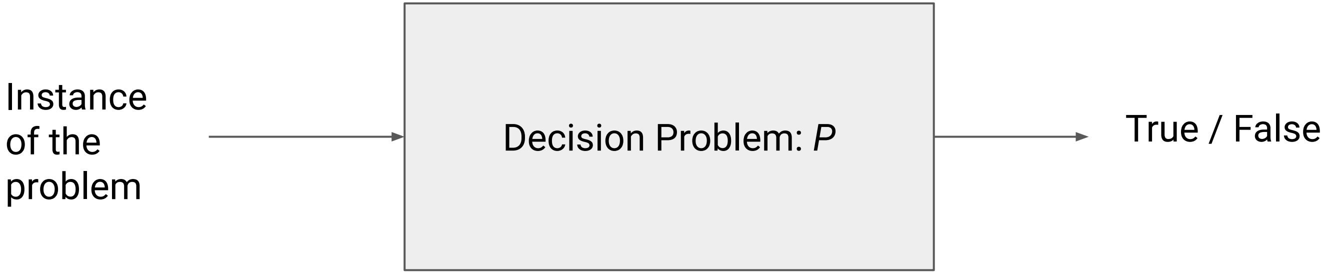
Problem:

Given two numbers, x and y, what is (x * y)?

The problem: MULT

Decision Problems

A decision problem is a function with boolean output.



Decision Problems

~~A decision problem is a function with boolean output.~~

Equivalently, we can express a decision problem using **sets** as:

- Define all possible inputs: A
- Define all desirable inputs: B

$$P(x) = (x \in B)$$

This way of defining a decision problem is somewhat easier because we just need to define the set B.

Example: primality test

- A = all of integer
- B = all of prime numbers

P(n) = true if n is prime

P(n) = false if n is not prime

Decidability of a decision problem

For now, we rely on an informal definition:

(Python) Decidability

A decision problem, P, is decidable if it can be implemented in Python.

A formal definition requires some more machineries:

- *Mathematical logic*
- *Computation*

Mathematical Logic

$\neg(\forall x P(x)) \equiv \exists x(\neg P(x))$
$\neg(\exists x P(x)) \equiv \forall x(\neg P(x))$
$\forall x \forall y Q(x,y) \equiv \forall y \forall x Q(x,y)$
$\exists x \exists y Q(x,y) \equiv \exists y \exists x Q(x,y)$

We will skip the formalism, but it's the syntax and rules of mathematics.

If you want to define something formally in mathematics, you have to rely on logic.

... so we don't make *mistakes*.

Example of mathematical logical definition

We work with predicate logic with integer arithmetics

- Logical connectives: AND, OR, NOT
- Quantifiers: forall ∀, exists ∃
- Variables over integers
- Constants: 0, 1, 2, ...
- Integer addition + and multiplication *

Let's denote this logic as **L**.

Let's define a decision problem, PRIME, using **L**.

- A = all the integers.
- B can be defined by **L**.

DIV(x, y) = ∃n. x = y * n

PRIME(x) = NOT(∃y. (DIV(x, y)
AND NOT(y = 1) AND NOT(y = x)))

So,

B = { x : PRIME(x) }

Ken Q Pu, Faculty of Science, Ontario Tech University

Back to decidability?

- We defined PRIME in mathematical logic **L**.
- Is it (Python-)decidable?
  - Definitely YES

(Python) Entscheidungsproblem

Are ***all*** decision problems expressible by mathematical logic Python-decidable?

No

Python-undecidable #1: Gödel's statement

Gödel's defined the following:

- A numbering system for all logical statements (true and false statements).
- The Gödel's number using the numbering system.
- A logical statement about the Gödel's number.

Gödel basically used a clever recursion in the construction.

He proved that no computation can never determine if Gödel's statement is true or false.

Python-undecidability #2: Diophantine equations

No Python program can ever solve all possible instances of the diophantine equation.

Wait, how is diophantine equation a decision problem?

Does a given diophantine equation have an integer solution?

Ken Q Pu, Faculty of Science, Ontario Tech University

We will revisit the following unanswered questions

1. Why are these decision problems Python-undecidable?
2. How is diophantine equation a decision problem?
3. What if we use a programming language that is "better" than Python?
4. Are there other undecidable problems?
5. How do we solve an undecidable problem?