Hilbert's Tenth Problem

John Lindsay Orr

Department of Mathematics
Univesity of Nebraska–Lincoln

September 15, 2005

Outline

1. Introduction
   - Disclaimer
   - History and Statement of the Problem
2. Sketch of Proof
   - Turing Machines and Decidability
   - Diophantine Sets
   - Universal Diophantine Equations
3. Going Into the Details
   - Working with Diophantine Sets
   - Coding n-tuples

Outline

Disclaimer

- I don’t know what I’m talking about!
- This guy does: Yuri Matiyasevich, “*Hilbert’s Tenth Problem*”

Outline

Hilbert's Problems

- Hilbert's twenty-three problems
- Second International Congress of Mathematicians held in Paris, 1900
- Included Continuum Hypothesis and Riemann Hypothesis
- Included general projects such as "Can physics be axiomatized"?

Hilert's Tenth Problem

10. Determination of the Solvability of a Diophantine Equation

Given a diophantine equation with any number of unknown quantities and with rational integral numerical coefficients: *To devise a process according to which it can be determined by a finite number of operations whether the equation is solvable in rational integers.*

A diophantine equation is a polynomial equation of the form

$$D(x_1, \ldots, x_m) = 0$$

where D is a polynomial with integer coefficients.

Example.

$$x^2 + y^2 - z^2 = 0$$

Example.

$$x^3 + y^3 - z^3 = 0$$

Can we find an algorithm which you can then present with any diophantine equation, $D(x_1, \ldots, x_m) = 0$, and be sure that you will get a "Yes" or "No" answer as to whether the equation has solutions over $\mathbb{N}^m$?

Can we find an algorithm which you can then present with any diophantine equation, $D(x_1, \ldots, x_m) = 0$, and be sure that you will get a "Yes" or "No" answer as to whether the equation has solutions over $\mathbb{N}^m$?

The Answer: **NO, WE CAN'T**

Notes

- Determining solvability isn't the same as finding a solution
- This wouldn't answer Fermat's Last Theorem
- By $\mathbb{N}$ I mean $\{0, 1, 2, 3, \ldots\}$
- By "solution" I almost always mean "solution in $\mathbb{N}$," not in $\mathbb{Z}$.

Notes

- Determining solvability isn't the same as finding a solution
- This wouldn't answer Fermat's Last Theorem
- By $\mathbb{N}$ I mean $\{0, 1, 2, 3, \ldots\}$
- By "solution" I almost always mean "solution in $\mathbb{N}$," not in $\mathbb{Z}$.

Notes

- Determining solvability isn't the same as finding a solution
- This wouldn't answer Fermat's Last Theorem
- By $\mathbb{N}$ I mean $\{0, 1, 2, 3, \ldots\}$
- By "solution" I almost always mean "solution in $\mathbb{N}$," not in $\mathbb{Z}$.

Notes

- Determining solvability isn't the same as finding a solution
- This wouldn't answer Fermat's Last Theorem
- By $\mathbb{N}$ I mean $\{0, 1, 2, 3, \ldots\}$
- By "solution" I almost always mean "solution in $\mathbb{N}$," not in $\mathbb{Z}$.

Why Only Over $\mathbb{N}$?

Over $\mathbb{N}$:

$$D(x_1, x_2, \ldots, x_n) = 0$$

Over $\mathbb{Z}$:

$$\begin{aligned} D(x_1,x_2,\ldots,x_n)^2 \\ +(y_{1,1}^2 + y_{1,2}^2 + y_{1,3}^2 + y_{1,4}^2 - x_1)^2 \\ +(y_{2,1}^2 + y_{2,2}^2 + y_{2,3}^2 + y_{2,4}^2 - x_2)^2 \\ \vdots \\ +(y_{n,1}^2 + y_{n,2}^2 + y_{n,3}^2 + y_{n,4}^2 - x_n)^2 = 0 \end{aligned}$$

Also study diophantine equation with parameters

$$D(a_1, \ldots, a_n, x_1, \ldots, x_m) = 0$$

and ask for which values of $(a_1, \ldots, a_n)$ does the equation have a solution.

Example.

$$ax - by - 1 = 0$$

Outline

What is a Turing Machine?

- It's a model for a computer
- Church-Turing Thesis says it models any computer

What does it look like?

- The machine scans a (singly) infinite tape
- The machine takes states from $X = \{x_1, \ldots, x_m\}$.
- The tape holds values from $Y = \{y_1, \ldots, y_n\}$

What is a Turing Machine?

- It's a model for a computer
- **Church-Turing Thesis** says it models any computer

What does it look like?

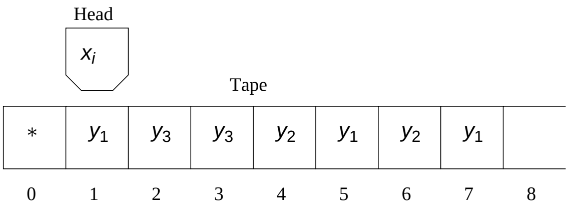
- The machine scans a (singly) infinite tape
- The machine takes states from $X = \{x_1, \ldots, x_m\}$.
- The tape holds values from $Y = \{y_1, \ldots, y_n\}$

What is a Turing Machine?

- It's a model for a computer
- Church-Turing Thesis says it models any computer

What does it look like?

- The machine scans a (singly) infinite tape
- The machine takes states from $X = \{x_1, \ldots, x_m\}$.
- The tape holds values from $Y = \{y_1, \ldots, y_n\}$

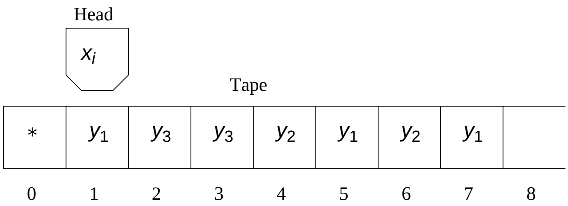


What is a Turing Machine?

- It's a model for a computer
- Church-Turing Thesis says it models any computer

What does it look like?

- The machine scans a (singly) infinite tape
- The machine takes states from $X = \{x_1, \ldots, x_m\}$.
- The tape holds values from $Y = \{y_1, \ldots, y_n\}$

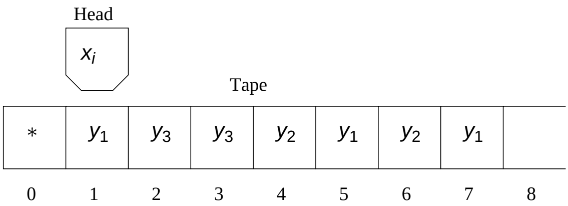


What is a Turing Machine?

- It's a model for a computer
- Church-Turing Thesis says it models any computer

What does it look like?

- The machine scans a (singly) infinite tape
- The **machine** takes states from $X = \{x_1, \ldots, x_m\}$.
- The tape holds values from $Y = \{y_1, \ldots, y_n\}$

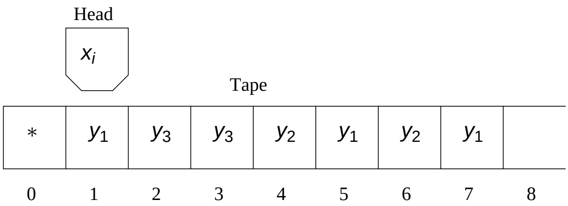


What is a Turing Machine?

- It's a model for a computer
- Church-Turing Thesis says it models any computer

What does it look like?

- The machine scans a (singly) infinite tape
- The machine takes states from $X = \{x_1, \ldots, x_m\}$.
- The **tape** holds values from $Y = \{y_1, \ldots, y_n\}$



How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x, y)$ to the cell
4. moves in direction $D(x, y)$
5. enters state $S(x, y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1, 0, 1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states.**

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x, y)$ to the cell
4. moves in direction $D(x, y)$
5. enters state $S(x, y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1, 0, 1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states**.

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x, y)$ to the cell
4. moves in direction $D(x, y)$
5. enters state $S(x, y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1, 0, 1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states**.

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x, y)$ to the cell
4. moves in direction $D(x, y)$
5. enters state $S(x, y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1, 0, 1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states.**

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x, y)$ to the cell
4. moves in direction $D(x, y)$
5. enters state $S(x, y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1, 0, 1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states.**

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x,y)$ to the cell
4. moves in direction $D(x,y)$
5. enters state $S(x,y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1,0,1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states**.

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x,y)$ to the cell
4. moves in direction $D(x,y)$
5. enters state $S(x,y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1,0,1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states.**

How a Turing Machine Works

At each step the machine:

1. scans the current cell while in state x
2. reads the value (y) from that cell
3. writes a value $W(x,y)$ to the cell
4. moves in direction $D(x,y)$
5. enters state $S(x,y)$

So the machine is determined by three finite functions:

$$W : X \times Y \longrightarrow Y, \qquad D : X \times Y \longrightarrow \{-1, 0, 1\}, \text{ and } S : X \times Y \longrightarrow X$$

The machine also has a single **initial state** x_1 and some **final states**.

How to Program a Turing Machine

Build simple machines that do basic operations, like:

- LEFT or RIGHT
- WRITE(y)
- READ(y)
- STOP or NEVERSTOP

Learn how to compose machines:

if (M_1) {
  M_2
}

or

while (M_1) {
  M_2
}

How to Program a Turing Machine

Build simple machines that do basic operations, like:

- LEFT or RIGHT
- WRITE(y)
- READ(y)
- STOP or NEVERSTOP

Learn how to compose machines:

if (M_1) {
  M_2
}

or

while (M_1) {
  M_2
}

How to Program a Turing Machine

Build simple machines that do basic operations, like:

- LEFT or RIGHT
- WRITE(y)
- READ(y)
- STOP or NEVERSTOP

Learn how to compose machines:

if (M_1) {
  M_2
}

or

while (M_1) {
  M_2
}

How to Program a Turing Machine

Build simple machines that do basic operations, like:

- LEFT or RIGHT
- WRITE(y)
- READ(y)
- STOP or NEVERSTOP

Learn how to compose machines:

if (M_1) {
  M_2
}

or

while (M_1) {
  M_2
}

How to Program a Turing Machine

Build simple machines that do basic operations, like:

- LEFT or RIGHT
- WRITE(y)
- READ(y)
- STOP or NEVERSTOP

Learn how to compose machines:

if (M_1) {
  M_2
}

or

while (M_1) {
  M_2
}

How to Program a Turing Machine

Build simple machines that do basic operations, like:

- LEFT or RIGHT
- WRITE(y)
- READ(y)
- STOP or NEVERSTOP

Learn how to compose machines:

if (M_1) {
  M_2
}

or

while (M_1) {
  M_2
}

Say that a set $S \subseteq \mathbb{N}$ is Turing decidable if there is a Turing machine M such that, whenever M is started with initial data on the tape encoding a $n \in \mathbb{N}$:

- M halts in state q_2 if $n \in S$
- M halts in state q_3 if $n \notin S$

How to Answer Hilbert's Tenth Problem

Imagine indexing all possible diophantine equations in some order. E.g. $D_1, D_2, D_3, \ldots$.

Let $S = \{k \ : \ D_k \text{ has a solution}\}$.

Hilbert's 10th problem becomes:

Question

Is S Turing decidable?

Say that a set $S \subseteq \mathbb{N}$ is Turing semidecidable if there is a Turing machine M such that, whenever M is started with initial data on the tape encoding a $n \in \mathbb{N}$:

- if $n \in S$ then M eventually halts
- if $n \notin S$ then M never halts

Lemma

If S is Turing decidable then S and S^c are Turing semidecidable.

Proof.

Let M be a machine that decides S. To semidecide S use the machine:

$$\textbf{if}\ (\ M\ )\ \{\ \text{STOP}\ \};\ \text{NEVERSTOP}$$

To semidecide S^c use the machine:

$$\textbf{if}\ (\ M\ )\ \{\ \text{NEVERSTOP}\ \}\ \text{STOP};$$

□

Lemma

If S is Turing decidable then S and S^c are Turing semidecidable.

Proof.

Let M be a machine that decides S. To semidecide S use the machine:

if (M) { STOP }; NEVERSTOP

To semidecide S^c use the machine:

if (M) { NEVERSTOP } STOP;

□

Lemma

If S is Turing decidable then S and S^c are Turing semidecidable.

Proof.

Let M be a machine that decides S. To semidecide S use the machine:

$$\textbf{if}\ (\ M\ )\ \{\ \text{STOP}\ \};\ \text{NEVERSTOP}$$

To semidecide S^c use the machine:

$$\textbf{if}\ (\ M\ )\ \{\ \text{NEVERSTOP}\ \}\ \text{STOP};$$

□

Theorem

The set S is Turing decidable if and only if S and S^c are Turing semidecidable.

Outline

Definition

Say that a set $S \subseteq \mathbb{N}^k$ is diophantine if there exists a diophantine equation

$$D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$$

such that $(a_1, \ldots, a_k) \in S$ if and only if $D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$ has a solution in N^n.

Definition

Say that a set $S \subseteq \mathbb{N}^k$ is diophantine if there exists a diophantine equation

$$D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$$

such that $(a_1, \ldots, a_k) \in S$ if and only if $D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$ has a solution in N^n.

Example. The set

$$\{(a, b) \ : \ gcd(a, b) = 1\}$$

is diophantine.

Definition

Say that a set $S \subseteq \mathbb{N}^k$ is diophantine if there exists a diophantine equation

$$D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$$

such that $(a_1, \ldots, a_k) \in S$ if and only if $D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$ has a solution in N^n.

Example. The set

$$\{(a, b) \ : \ gcd(a, b) = 1\}$$

is diophantine. (Take $D(a, b, x, y) = ax - by - 1$.)

Example. The set

$$\{a \ : \ a \text{ is not a prime}\}$$

is diophantine.

Proof. Let

$$D(a, x, y) = (x + 2)(y + 2) - a$$

In fact, the set

$$\{a \;:\; a \text{ is a prime}\}$$

is diophantine.

Factoid. A set $S \subseteq \mathbb{N}$ is diophantine if and only if S is the set of non-negative values taken by some integer-coefficient polynomial as its variables range over $\mathbb{N}$.

Thus, incredibly,

$$\{\text{prime numbers}\} = \mathbb{N} \cap \{D(x_1, \ldots, x_n \;:\; x1, \ldots, x_n \in \mathbb{N}\}$$

Lemma

Every diophantine set is Turing semidecidable.

Proof.

S has a diophantine representation

$$D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$$

Initialize the tape with $(a_1, \ldots, a_k) \in \mathbb{N}_k$, and run:

foreach $x = (x_1, \ldots, x_n) \in \mathbb{N}^n$ {
  if($D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$) {
    STOP
  }
}

□

Theorem

Every Turing semidecidable set is diophantine.

Corollary

A set is diophantine $\iff$ it is Turing semidecidable.

Theorem

Every Turing semidecidable set is diophantine.

Corollary

A set is diophantine $\iff$ *it is Turing semidecidable.*

Summary

What have we learned?

$$
\begin{aligned}
& S \text{ is decidable} \\
\iff & S, S^c \text{ are semidecidable} \\
\iff & S, S^c \text{ are diophantine}
\end{aligned}
$$

So one way to show a set is not decidable is to show that one of S or S^c is not diophantine.

Outline

Definition

The integer-coefficient polynomial

$$U(a_1, \ldots, a_k, c, y_1, \ldots, y_w)$$

is a universal diophantine polynomial if, for any diophantine equation

$$D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$$

we can find a code $c \in \mathbb{N}$ such that

$$\exists x_1, \ldots, x_n \text{ with } D(a_1, \ldots, a_k, x_1, \ldots, x_n) = 0$$

$$\iff$$

$$\exists y_1, \ldots, y_w \text{ with } U(a_1, \ldots, a_k, c, y_1, \ldots, y_w) = 0$$

Theorem

For each k, there exists a universal diophantine equation

$$U_k(a_1, \ldots, a_k, c, y_1, \ldots, y_w)$$

Let

$$H_0 = \{c \ : \ U_0(c, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$$

This is our "enumeration of the solvable diophantine equations".

We shall show that H_0 is diophantine and H_0^c is not!

Theorem

For each k, there exists a universal diophantine equation

$$U_k(a_1, \ldots, a_k, c, y_1, \ldots, y_w)$$

Let

$$H_0 = \{c \ : \ U_0(c, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$$

This is our "enumeration of the solvable diophantine equations".

We shall show that H_0 is diophantine and H_0^c is not!

Theorem

For each k, there exists a universal diophantine equation

$$U_k(a_1, \ldots, a_k, c, y_1, \ldots, y_w)$$

Let

$$H_0 = \{c \ : \ U_0(c, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$$

This is our "enumeration of the solvable diophantine equations".

We shall show that H_0 is diophantine and H_0^c is not!

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0$ has a solution$\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (First part) Write $D(k, y_1, \ldots, y_w) = U_1(k, k, y_1, \ldots, y_w)$. Then

$$k \in H_1 \iff D(k, y_1, \ldots, y_w) \text{ has a solution}$$

Thus H_1 is diophantine.

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (First part) Write $D(k, y_1, \ldots, y_w) = U_1(k, k, y_1, \ldots, y_w)$. Then

$$k \in H_1 \iff D(k, y_1, \ldots, y_w) \text{ has a solution}$$

Thus H_1 is diophantine.

Let $H_1 = \{k \; : \; U_1(k, k, y_1, \dots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (First part) Write $D(k, y_1, \dots, y_w) = U_1(k, k, y_1, \dots, y_w)$. Then

$$k \in H_1 \iff D(k, y_1, \dots, y_w) \text{ has a solution}$$

Thus H_1 is diophantine.

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (First part) Write $D(k, y_1, \ldots, y_w) = U_1(k, k, y_1, \ldots, y_w)$. Then

$$k \in H_1 \iff D(k, y_1, \ldots, y_w) \text{ has a solution}$$

Thus H_1 is diophantine.

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (Second part) If H_1^c were diophantine there would be a code, k, for it. But then ask: Does $U_1(k, k, y_1, \ldots, y_w) = 0$ have a solution?
If "yes" then $k \in H_1$. But k is the code for the set H_1^c so in general:

$$U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution } \iff a \in H_1^c$$

Thus, $k \in H_1^c$. Contradiction!
If "no" then $k \in H_1^c$. But likewise $a \notin H_1^c$. Contradiction!

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (Second part) If H_1^c were diophantine there would be a code, k, for it. But then ask: Does $U_1(k, k, y_1, \ldots, y_w) = 0$ have a solution?

If "yes" then $k \in H_1$. But k is the code for the set H_1^c so in general:

$$U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution } \iff a \in H_1^c$$

Thus, $k \in H_1^c$. Contradiction!

If "no" then $k \in H_1^c$. But likewise $a \notin H_1^c$. Contradiction!

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (Second part) If H_1^c were diophantine there would be a code, k, for it. But then ask: Does $U_1(k, k, y_1, \ldots, y_w) = 0$ have a solution?

If "yes" then $k \in H_1$. But k is the code for the set H_1^c so in general:

$$U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution } \iff a \in H_1^c$$

Thus, $k \in H_1^c$. Contradiction!

If "no" then $k \in H_1^c$. But likewise $a \notin H_1^c$. Contradiction!

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (Second part) If H_1^c were diophantine there would be a code, k, for it. But then ask: Does $U_1(k, k, y_1, \ldots, y_w) = 0$ have a solution?
If "yes" then $k \in H_1$. But k is the code for the set H_1^c so in general:

$$U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution } \iff a \in H_1^c$$

Thus, $k \in H_1^c$. Contradiction!
If "no" then $k \in H_1^c$. But likewise $a \notin H_1^c$. Contradiction!

Let $H_1 = \{k \ : \ U_1(k, k, y_1, \dots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (Second part) If H_1^c were diophantine there would be a code, k, for it. But then ask: Does $U_1(k, k, y_1, \dots, y_w) = 0$ have a solution?
If "yes" then $k \in H_1$. But k is the code for the set H_1^c so in general:

$$U_1(k, k, y_1, \dots, y_w) = 0 \text{ has a solution } \iff a \in H_1^c$$

Thus, $k \in H_1^c$. Contradiction!
If "no" then $k \in H_1^c$. But likewise $a \notin H_1^c$. Contradiction!

Let $H_1 = \{k \; : \; U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution}\}$

Claim. H_1 is a diophantine set but H_1^c is not.

Proof. (Second part) If H_1^c were diophantine there would be a code, k, for it. But then ask: Does $U_1(k, k, y_1, \ldots, y_w) = 0$ have a solution?
If "yes" then $k \in H_1$. But k is the code for the set H_1^c so in general:

$$U_1(k, k, y_1, \ldots, y_w) = 0 \text{ has a solution} \iff a \in H_1^c$$

Thus, $k \in H_1^c$. Contradiction!
If "no" then $k \in H_1^c$. But likewise $a \notin H_1^c$. Contradiction!

Let $H_0 = \{k \ : \ U_0(k, y_1, \dots, y_v) = 0 \text{ has a solution}\}$

$$\begin{aligned} & U_1(a, k, y_1, \dots, y_w) = 0 \text{ has a solution} \\ \iff & D_k(a, x_1, \dots, x_n) = 0 \text{ has a solution} \\ \iff & W(x_1, \dots, x_n) = 0 \text{ has a solution} \\ \iff & U_0(c(a, k), y_1, \dots, y_v) = 0 \text{ has a solution} \end{aligned}$$

Thus

$$c(k, k) \in H_0 \iff k \in H_1$$
$$c(k, k) \in H_0^c \iff k \in H_1^c$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$

$$U_1(a, k, y_1, \ldots, y_w) = 0 \text{ has a solution}$$
$$\iff D_k(a, x_1, \ldots, x_n) = 0 \text{ has a solution}$$
$$\iff W(x_1, \ldots, x_n) = 0 \text{ has a solution}$$
$$\iff U_0(c(a, k), y_1, \ldots, y_v) = 0 \text{ has a solution}$$

Thus

$$c(k, k) \in H_0 \iff k \in H_1$$
$$c(k, k) \in H_0^c \iff k \in H_1^c$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \;:\; U_0(k, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$

$$\begin{aligned} & U_1(a, k, y_1, \ldots, y_w) = 0 \text{ has a solution} \\ \iff & D_k(a, x_1, \ldots, x_n) = 0 \text{ has a solution} \\ \iff & W(x_1, \ldots, x_n) = 0 \text{ has a solution} \\ \iff & U_0(c(a, k), y_1, \ldots, y_v) = 0 \text{ has a solution} \end{aligned}$$

Thus

$$\begin{aligned} c(k, k) \in H_0 &\iff k \in H_1 \\ c(k, k) \in H_0^c &\iff k \in H_1^c \end{aligned}$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$

$$
\begin{aligned}
& U_1(a, k, y_1, \ldots, y_w) = 0 \text{ has a solution} \\
\iff & D_k(a, x_1, \ldots, x_n) = 0 \text{ has a solution} \\
\iff & W(x_1, \ldots, x_n) = 0 \text{ has a solution} \\
\iff & U_0(c(a,k), y_1, \ldots, y_v) = 0 \text{ has a solution}
\end{aligned}
$$

Thus

$$
\begin{aligned}
c(k,k) \in H_0 &\iff k \in H_1 \\
c(k,k) \in H_0^c &\iff k \in H_1^c
\end{aligned}
$$

Fact. $c(a,k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0$ has a solution$\}$

$$\begin{aligned} & U_1(a, k, y_1, \ldots, y_w) = 0 \text{ has a solution} \\ \iff & D_k(a, x_1, \ldots, x_n) = 0 \text{ has a solution} \\ \iff & W(x_1, \ldots, x_n) = 0 \text{ has a solution} \\ \iff & U_0(c(a, k), y_1, \ldots, y_v) = 0 \text{ has a solution} \end{aligned}$$

Thus

$$\begin{aligned} c(k, k) \in H_0 &\iff k \in H_1 \\ c(k, k) \in H_0^c &\iff k \in H_1^c \end{aligned}$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \ : \ U_0(k, y_1, \dots, y_v) = 0 \text{ has a solution}\}$

$$\begin{aligned} & U_1(a, k, y_1, \dots, y_w) = 0 \text{ has a solution} \\ \iff & D_k(a, x_1, \dots, x_n) = 0 \text{ has a solution} \\ \iff & W(x_1, \dots, x_n) = 0 \text{ has a solution} \\ \iff & U_0(c(a, k), y_1, \dots, y_v) = 0 \text{ has a solution} \end{aligned}$$

Thus

$$c(k, k) \in H_0 \iff k \in H_1$$

$$c(k, k) \in H_0^c \iff k \in H_1^c$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$

$$
\begin{aligned}
& U_1(a, k, y_1, \ldots, y_w) = 0 \text{ has a solution} \\
\iff & D_k(a, x_1, \ldots, x_n) = 0 \text{ has a solution} \\
\iff & W(x_1, \ldots, x_n) = 0 \text{ has a solution} \\
\iff & U_0(c(a, k), y_1, \ldots, y_v) = 0 \text{ has a solution}
\end{aligned}
$$

Thus

$$
\begin{aligned}
c(k, k) \in H_0 &\iff k \in H_1 \\
c(k, k) \in H_0^c &\iff k \in H_1^c
\end{aligned}
$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Let $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0 \text{ has a solution}\}$

$$\begin{aligned} & U_1(a, k, y_1, \ldots, y_w) = 0 \text{ has a solution} \\ \iff & D_k(a, x_1, \ldots, x_n) = 0 \text{ has a solution} \\ \iff & W(x_1, \ldots, x_n) = 0 \text{ has a solution} \\ \iff & U_0(c(a,k), y_1, \ldots, y_v) = 0 \text{ has a solution} \end{aligned}$$

Thus

$$c(k,k) \in H_0 \iff k \in H_1$$
$$c(k,k) \in H_0^c \iff k \in H_1^c$$

Fact. $c(a, k)$ is a diophantine polynomial $\Rightarrow H_0^c$ is not diophantine!

Summary

We have seen that

- $H_0 = \{k : U_0(k, y_1, \ldots, y_v) = 0$ has a solution$\}$ is not Turing decidable.
- The elements of H_0 are in one-to-one correspondence with the solvable diophantine equations.
- Thus, there is no algorithm to decide which diophantine equations are solvable and which are not.

Summary

We have seen that

- $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0$ has a solution$\}$ is not Turing decidable.
- The elements of H_0 are in one-to-one correspondence with the solvable diophantine equations.
- Thus, there is no algorithm to decide which diophantine equations are solvable and which are not.

Summary

We have seen that

- $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0$ has a solution$\}$ is not Turing decidable.
- The elements of H_0 are in one-to-one correspondence with the solvable diophantine equations.
- Thus, there is no algorithm to decide which diophantine equations are solvable and which are not.

Summary

We have seen that

- $H_0 = \{k \ : \ U_0(k, y_1, \ldots, y_v) = 0$ has a solution$\}$ is not Turing decidable.
- The elements of H_0 are in one-to-one correspondence with the solvable diophantine equations.
- Thus, there is no algorithm to decide which diophantine equations are solvable and which are not.

Outline

Unions and Intersections

Let $S_1, S_2 \subseteq \mathbb{N}^k$ be diophantine sets with representations

$(a_1, \ldots, a_k) \in S_1 \iff D_1(a_1, \ldots, a_k, x_1, \ldots, x_m) = 0$ has a solution

and

$(a_1, \ldots, a_k) \in S_1 \iff D_2(a_1, \ldots, a_k, y_1, \ldots, y_n) = 0$ has a solution

Then $S_1 \cup S_2$ and $S_1 \cap S_2$ are diophantine sets.

Proof. Consider

$$D_1(a_1, \ldots, a_k, x_1, \ldots, x_m) D_2(a_1, \ldots, a_k, y_1, \ldots, y_n) = 0$$

and

$$D_1(a_1, \ldots, a_k, x_1, \ldots, x_m)^2 + D_2(a_1, \ldots, a_k, y_1, \ldots, y_n)^2 = 0$$

Unions and Intersections

Let $S_1, S_2 \subseteq \mathbb{N}^k$ be diophantine sets with representations

$(a_1, \ldots, a_k) \in S_1 \iff D_1(a_1, \ldots, a_k, x_1, \ldots, x_m) = 0$ has a solution

and

$(a_1, \ldots, a_k) \in S_1 \iff D_2(a_1, \ldots, a_k, y_1, \ldots, y_n) = 0$ has a solution

Then $S_1 \cup S_2$ and $S_1 \cap S_2$ are diophantine sets.

Proof. Consider

$$D_1(a_1, \ldots, a_k, x_1, \ldots, x_m) D_2(a_1, \ldots, a_k, y_1, \ldots, y_n) = 0$$

and

$$D_1(a_1, \ldots, a_k, x_1, \ldots, x_m)^2 + D_2(a_1, \ldots, a_k, y_1, \ldots, y_n)^2 = 0$$

Some Basic Diophantine Sets

The set $\{(a, b) \; : \; aRb\}$ is diophantine when "R" is one of the relations:

- $a = b$ (consider "$\exists x$ s.t. $x + (a - b)^2 = 0$")
- $a < b$ (consider "$\exists x$ s.t. $a + x + 1 = b$")
- $a | b$ (consider "$\exists x$ s.t. $ax = b$")

Some Basic Diophantine Sets

The set $\{(a, b) \ : \ aRb\}$ is diophantine when "R" is one of the relations:

- $a = b$ (consider "$\exists x$ s.t. $x + (a - b)^2 = 0$")
- $a < b$ (consider "$\exists x$ s.t. $a + x + 1 = b$")
- $a|b$ (consider "$\exists x$ s.t. $ax = b$")

Some Basic Diophantine Sets

The set $\{(a, b) \ : \ aRb\}$ is diophantine when "R" is one of the relations:

- $a = b$ (consider "$\exists x$ s.t. $x + (a - b)^2 = 0$")
- $a < b$ (consider "$\exists x$ s.t. $a + x + 1 = b$")
- $a|b$ (consider "$\exists x$ s.t. $ax = b$")

Some Basic Diophantine Sets

The set $\{(a, b) \ : \ aRb\}$ is diophantine when “R” is one of the relations:

- $a = b$ (consider “$\exists x$ s.t. $x + (a - b)^2 = 0$”)
- $a < b$ (consider “$\exists x$ s.t. $a + x + 1 = b$”)
- $a|b$ (consider “$\exists x$ s.t. $ax = b$”)

Some Basic Diophantine Sets

The set $\{(a,b) \;:\; aRb\}$ is diophantine when "R" is one of the relations:

- $a = b$ (consider "$\exists x$ s.t. $x + (a-b)^2 = 0$")
- $a < b$ (consider "$\exists x$ s.t. $a + x + 1 = b$")
- $a|b$ (consider "$\exists x$ s.t. $ax = b$")

The set $\{(a,b,c) \;:\; a = rem(b,c)\}$ is diophantine.

Proof.

$$
\begin{aligned}
& a = rem(b,c) \\
\iff & a < c \;\&\; c|b-a \\
\iff & \exists x, y \text{ s.t. } (a+x+1-b)^2 + (cy-(b-a))^2 = 0
\end{aligned}
$$

Some Basic Diophantine Sets

The set $\{(a,b) \ : \ aRb\}$ is diophantine when "R" is one of the relations:

- $a = b$ (consider "$\exists x$ s.t. $x + (a-b)^2 = 0$")
- $a < b$ (consider "$\exists x$ s.t. $a + x + 1 = b$")
- $a|b$ (consider "$\exists x$ s.t. $ax = b$")

The set $\{(a,b,c) \ : \ a = rem(b,c)\}$ is diophantine.

Proof.

$$
\begin{aligned}
& a = rem(b,c) \\
\iff & a < c \ \& \ c|b-a \\
\iff & \exists x, y \text{ s.t. } (a+x+1-b)^2 + (cy-(b-a))^2 = 0
\end{aligned}
$$

Some Basic Diophantine Sets

The set $\{(a, b) \ : \ aRb\}$ is diophantine when “R” is one of the relations:

- $a = b$ (consider “$\exists x$ s.t. $x + (a - b)^2 = 0$”)
- $a < b$ (consider “$\exists x$ s.t. $a + x + 1 = b$”)
- $a | b$ (consider “$\exists x$ s.t. $ax = b$”)

The set $\{(a, b, c) \ : \ a = rem(b, c)\}$ is diophantine.

Proof.

$$
\begin{aligned}
& a = rem(b, c) \\
\iff & a < c \ \& \ c | b - a \\
\iff & \exists x, y \text{ s.t. } (a + x + 1 - b)^2 + (cy - (b - a))^2 = 0
\end{aligned}
$$

Some Basic Diophantine Sets

The set $\{(a,b) \ : \ aRb\}$ is diophantine when "R" is one of the relations:

- $a = b$ (consider "$\exists x$ s.t. $x + (a-b)^2 = 0$")
- $a < b$ (consider "$\exists x$ s.t. $a + x + 1 = b$")
- $a|b$ (consider "$\exists x$ s.t. $ax = b$")

The set $\{(a,b,c) \ : \ a = rem(b,c)\}$ is diophantine.

Proof.

$$\begin{aligned} & a = rem(b,c) \\ \Longleftrightarrow & a < c \ \& \ c|b-a \\ \Longleftrightarrow & \exists x, y \text{ s.t. } (a+x+1-b)^2 + (cy-(b-a))^2 = 0 \end{aligned}$$

The set $\{(a,b,c) \; : \; a \equiv b \ (mod\ c)\}$ is diophantine.

Proof.

$$
\begin{aligned}
& a \equiv b \ (mod\ c) \\
\iff & rem(a,c) = rem(b,c) \\
\iff & \exists v,w \text{ s.t. } v = rem(a,c) \ \& \ w = rem(b,c) \ \& \ w = v \\
\iff & \exists v,w,x,y,x',y',z \text{ s.t. } ((v+x+1-a)^2 + (cy-(a-v))^2)^2 \\
& + ((w+x'+1-b)^2 + (cy'-(b-w))^2)^2 \\
& + (z+(v-w)^2)^2 = 0
\end{aligned}
$$

The set $\{(a,b,c)\ :\ a \equiv b\ (mod\ c)\}$ is diophantine.

Proof.

$$
\begin{aligned}
& a \equiv b\ (mod\ c) \\
\iff & rem(a,c) = rem(b,c) \\
\iff & \exists v, w \text{ s.t. } v = rem(a,c)\ \&\ w = rem(b,c)\ \&\ w = v \\
\iff & \exists v, w, x, y, x', y', z \text{ s.t. } ((v + x + 1 - a)^2 + (cy - (a - v))^2)^2 \\
& + ((w + x' + 1 - b)^2 + (cy' - (b - w))^2)^2 \\
& + (z + (v - w)^2)^2 = 0
\end{aligned}
$$

The set $\{(a,b,c) \ : \ a \equiv b \ (mod\ c)\}$ is diophantine.

Proof.

$$
\begin{aligned}
& a \equiv b \ (mod\ c) \\
\iff & rem(a,c) = rem(b,c) \\
\iff & \exists v,w \text{ s.t. } v = rem(a,c) \ \& \ w = rem(b,c) \ \& \ w = v \\
\iff & \exists v,w,x,y,x',y',z \text{ s.t. } ((v+x+1-a)^2 + (cy-(a-v))^2)^2 \\
& \qquad + ((w+x'+1-b)^2 + (cy'-(b-w))^2)^2 \\
& \qquad + (z+(v-w)^2)^2 = 0
\end{aligned}
$$

Exponentiation is Diophantine

Theorem (Matiyasevich, 1970)

The set $\{(a, b, c) \ : \ a = b^c\}$ is diophantine.

Corollary

The set $\{(a, n) \ : \ a = n!\}$ is diophantine.

$$a \text{ is prime } \iff a > 1 \ \& \ gcd(a, (a-1)!) = 1$$

Exponentiation is Diophantine

Theorem (Matiyasevich, 1970)

The set $\{(a, b, c) \ : \ a = b^c\}$ is diophantine.

Corollary

The set $\{(a, n) \ : \ a = n!\}$ is diophantine.

$$a \text{ is prime} \iff a > 1 \ \& \ gcd(a, (a-1)!) = 1$$

Exponentiation is Diophantine

Theorem (Matiyasevich, 1970)

The set $\{(a, b, c) \ : \ a = b^c\}$ is diophantine.

Corollary

The set $\{(a, n) \ : \ a = n!\}$ is diophantine.

$$a \text{ is prime} \iff a > 1 \ \& \ gcd(a, (a-1)!) = 1$$

Outline

Coding n-tuples

$$(a_0, a_1, \ldots, a_n)$$

$$\downarrow$$

$$a = \underbrace{a_0 + a_1 b + a_2 b^2 + \cdots}_{y} + \underbrace{a_k b^k}_{eb^k} + \underbrace{\cdots + a_n b^n}_{xb^{k+1}}$$

$$e = Elem(k, a, b)$$

$$\iff$$

$$\exists x, y \quad \text{s.t.} \quad a = y + eb^k + xb^{k+1} \;\; \& \;\; e < b \;\; \& \;\; y < b^k$$

Coding n-tuples

$$(a_0, a_1, \ldots, a_n)$$

$$\downarrow$$

$$a = \underbrace{a_0 + a_1 b + a_2 b^2 + \cdots}_{y} + \underbrace{a_k b^k}_{eb^k} + \underbrace{\cdots + a_n b^n}_{xb^{k+1}}$$

$$e = Elem(k, a, b)$$

$$\iff$$

$$\exists x, y \quad \text{s.t.} \quad a = y + eb^k + xb^{k+1} \;\; \& \;\; e < b \;\; \& \;\; y < b^k$$

Primes

$$(b+1)^n = \binom{n}{0} + \binom{n}{1} b + \cdots + \binom{n}{k} b^k + \cdots + \binom{n}{n} b^n$$

$$a = \binom{n}{k}$$

$$\iff$$

$$a = Elem(k, (b+1)^n, b) \;\;\&\;\; b = 2^n$$

a is prime

$$\iff$$

$$a > 1 \;\;\&\;\; gcd(a, (a-1)!) = 1$$

Primes

$$(b+1)^n = \binom{n}{0} + \binom{n}{1} b + \cdots + \binom{n}{k} b^k + \cdots + \binom{n}{n} b^n$$

$$a = \binom{n}{k}$$

$$\iff$$

$$a = Elem(k, (b+1)^n, b) \ \ \& \ \ b = 2^n$$

a is prime

$$\iff$$

$$a > 1 \ \ \& \ \ gcd(a, (a-1)!) = 1$$

Primes

$$(b+1)^n = \binom{n}{0} + \binom{n}{1} b + \cdots + \binom{n}{k} b^k + \cdots + \binom{n}{n} b^n$$

$$a = \binom{n}{k}$$

$$\iff$$

$$a = Elem(k, (b+1)^n, b) \;\;\&\;\; b = 2^n$$

a is prime

$$\iff$$

$$a > 1 \;\;\&\;\; gcd(a, (a-1)!) = 1$$