Undecidable Problems

An Overview

Graham Campbell

Summer 2017

Undecidability

There are two common settings in which ones speaks of undecidability.

Definition (Independence from axioms)

A single statement is called **undecidable** iff neither it, nor its negation, can be deduced using the rules of logic from the set of axioms being used.

Undecidability

There are two common settings in which ones speaks of undecidability.

Definition (Independence from axioms)

A single statement is called **undecidable** iff neither it, nor its negation, can be deduced using the rules of logic from the set of axioms being used.

Definition (Decision problem)

A family of problems with yes/no answers is called **undecidable** iff there is no algorithm that terminates with the correct answer for every problem in the family.

Connection

There is a connection between he two notations of undecidability.

Theorem

Fix a decision problem and axiom system A such that:

1. *if Y_i is provable in A, then the answer to i is yes,*
2. *if $\neg Y_i$ is provable in A, then the answer to i is no.*

Under these assumptions, if the decision problem is undecidable in the 2nd sense, then at least one of its instance statements Y_i is undecidable in the 1st sense.

Connection

There is a connection between he two notations of undecidability.

Theorem

Fix a decision problem and axiom system A such that:

1. *if Y_i is provable in A, then the answer to i is yes,*
2. *if $\neg Y_i$ is provable in A, then the answer to i is no.*

Under these assumptions, if the decision problem is undecidable in the 2nd sense, then at least one of its instance statements Y_i is undecidable in the 1st sense.

Proof:

If every Y_i could be proved or disproved in A, then the decision problem could be solved by a program that generates all theorems deducible from A until it finds either Y_i or $\neg Y_i$. □

The Halting Problem

Definition (The halting problem)

Input: A program p and a natural number x.
Output: Does p eventually halt when run on input x?

Theorem

The halting problem is undecidable.

The Halting Problem

Definition (The halting problem)

Input: A program p and a natural number x.
Output: Does p eventually halt when run on input x?

Theorem

The halting problem is undecidable.

Proof:
Encoding programs as natural numbers, suppose that there were an algorithm for deciding when program p halts on input x. Then, we could write a new program H such that H halts on every input x iff program x does not halt on input x. Taking $x = H$, we have a contradiction: H halts on H iff H does not halt on H. □

Listable Sets

Definition (Computable)

Let $A \subseteq \mathbb{N}$. We call A **computable** (**recursive**) iff there exists an algorithm that takes an input $n \in \mathbb{N}$, and decides whether or not $n \in A$.

Listable Sets

Definition (Computable)

Let $A \subseteq \mathbb{N}$. We call A **computable** (**recursive**) iff there exists an algorithm that takes an input $n \in \mathbb{N}$, and decides whether or not $n \in A$.

Definition (Listable)

Let $A \subseteq \mathbb{N}$. We call A **listable** (**recursively enumerable**) iff there is a program that when left running forever, eventually prints out exactly the elements of A.

Lemma

All **computable** *sets are* **listable**.

Membership Problem

Definition (Membership in a listable set A)

Input: Natural number n.
Output: Is $n \in A$?

Theorem

There exists a listable set A for which the membership problem is undecidable.

Membership Problem

Definition (Membership in a listable set A)

Input: Natural number n.
Output: Is $n \in A$?

Theorem

There exists a listable set A for which the membership problem is undecidable.

Proof:
Let A be the set of programs that halt. Then, A is listable. But, the undecidability of the halting problem implies that A is not computable, solved the membership problem for A is undecidable. □

Logic

Definition (Propositional logic)

Propositional logic consists of a set of atomic propositional symbols, with the logical operators of **negation**, **conjunction**, **disjunction**, **implication**, and **equivalence**.

Logic

Definition (Propositional logic)

Propositional logic consists of a set of atomic propositional symbols, with the logical operators of **negation**, **conjunction**, **disjunction**, **implication**, and **equivalence**.

Definition (First order logic)

First order logic extends propositional logic, allowing **quantification** over **variables**, and adds **functions** and **relations**.

Logic

Definition (Propositional logic)

Propositional logic consists of a set of atomic propositional symbols, with the logical operators of **negation**, **conjunction**, **disjunction**, **implication**, and **equivalence**.

Definition (First order logic)

First order logic extends propositional logic, allowing **quantification** over **variables**, and adds **functions** and **relations**.

Definition (Model)

A first-order structure that satisfies all sentences in a given theory is said to be a model of the theory.

Completeness

Definition (Universally valid)

Fix a finite set of axioms. A statement is said to be **universally valid** iff it is true in every structure satisfying the axioms.

Completeness

Definition (Universally valid)

Fix a finite set of axioms. A statement is said to be **universally valid** iff it is true in every structure satisfying the axioms.

Definition (Completness)

A deductive system is called **complete** iff every **universally valid** formula is the conclusion of some formal deduction from its axioms.

Completeness

Definition (Universally valid)

Fix a finite set of axioms. A statement is said to be **universally valid** iff it is true in every structure satisfying the axioms.

Definition (Completness)

A deductive system is called **complete** iff every **universally valid** formula is the conclusion of some formal deduction from its axioms.

Theorem (Godel's completeness)

In a consistent first-order theory, every **universally valid** *statement is deducible in a finite number of steps.*

Entscheidungsproblem

Definition (Entscheidungsproblem)

Input: A sentence s in the language a first order logic, possibly with a finite number of extra axioms, $\mathcal{F}$.
Output: Is s universally valid (true in every model of the axioms of $\mathcal{F}$)?

Entscheidungsproblem

Definition (Entscheidungsproblem)

Input: A sentence s in the language a first order logic, possibly with a finite number of extra axioms, $\mathcal{F}$.
Output: Is s universally valid (true in every model of the axioms of $\mathcal{F}$)?

It was known to Hilbert that there is a single first order logic $\mathcal{F}_0$ without special axioms such that if the Entscheidungsproblem for $\mathcal{F}_0$ is decidable, then so is the Entscheidungsproblem for any first order logic. But, Church and Turing independently proved that the Entscheidungsproblem for $\mathcal{F}_0$ was **undecidable**!

Post Correspondence 1

Imagine a rectangular block with a (potentially different) finite string of a's and b's written a long the top and bottom.

When finitely many such blocks are laid side by side, we can consider the strings along the top and bottom, concatenated.

Post Correspondence 1

Imagine a rectangular block with a (potentially different) finite string of a's and b's written a long the top and bottom.

When finitely many such blocks are laid side by side, we can consider the strings along the top and bottom, concatenated.

Definition (Post correspondence problem)

Input: A finite collection of blocks, labelled as above.
Output: Given an unlimited supply of blocks, one can form the concatenation of a non-empty finite sequence of blocks. Does the concatenated top string equal the concatenated bottom string?

Post Correspondence 2

Theorem

The Post correspondence problem is undecidable.

Post Correspondence 2

Theorem

The Post correspondence problem is undecidable.

Proof:

It is possible to show this by embedding the halting problem in it.

Namely, with some work, it is possible, given a computer program p, to construct an instance of the Post correspondence problem that has a positive answer iff p halts. □

Tiling Problem 1

Wang tiles are unit squares in a plane, with sides parallel to the axes, such that each side of the squares has been assigned a colour. Tiles can be translated, but not rotated or reflected.

A tiling of the plane into such squares is valid iff whenever two squares share an edge, the colours match, as in the game of dominoes.

Tiling Problem 1

Wang tiles are unit squares in a plane, with sides parallel to the axes, such that each side of the squares has been assigned a colour. Tiles can be translated, but not rotated or reflected.

A tiling of the plane into such squares is valid iff whenever two squares share an edge, the colours match, as in the game of dominoes.

Definition (Tiling problem)

Input: A finite collection of Wang tiles.
Output: Is there a valid tiling of the entire plane, using only translated copies of the tiles?

Tiling Problem 2

Wang also conjectured that if a tiling exists for a given finite collection, then there exists a **periodic tiling**, one that is invariant under translations by the vectors in a $(n\mathbb{Z})^2$. This conjecture would imply decidability of the tiling problem.

Tiling Problem 2

Wang also conjectured that if a tiling exists for a given finite collection, then there exists a **periodic tiling**, one that is invariant under translations by the vectors in a $(n\mathbb{Z})^2$. This conjecture would imply decidability of the tiling problem.

Berger later showed that the tiling problem was **undecidable** by embedding the halting problem as a sub problem of the tiling problem. Combining this with Wang's observation, shows that if there exist finite collection that can tile a plane, it can only be aperiodic.

Graph Homomorphisms 1

Definition (Vertex set)

Let G be a graph. Then $V(G)$ is the vertex set of G.

Definition (Graph homomorphism)

Fix finite graphs G and H. A **homomorphism** from H to G is a map $V(H) \to V(G)$ such that every edge of H maps to an edge of G.

Graph Homomorphisms 1

Definition (Vertex set)

Let G be a graph. Then $V(G)$ is the vertex set of G.

Definition (Graph homomorphism)

Fix finite graphs G and H. A **homomorphism** from H to G is a map $V(H) \to V(G)$ such that every edge of H maps to an edge of G.

Definition (Homomorphisms density)

Fix finite graphs G and H. The **homomorphism density** $t(H, G)$ is the probability that a uniformly chosen, random map $V(H) \to V(G)$ is a homomorphism.

Graph Homomorphisms 2

Known Inequality

For every finite graph G:

$$t(K_3, G) \leq 2t(K_2, G)^2 - t(K_2, G)$$

Graph Homomorphisms 2

Known Inequality

For every finite graph G:

$$t(K_3, G) \leq 2t(K_2, G)^2 - t(K_2, G)$$

Definition (Inequalities between graph homomorphism densities)

Input: Finite graphs $H_1, ..., H_k$ and integers $a_1, ..., a_k \in \mathbb{Z}$.
Output: Does $a_1 t(H_1, G) + \cdots + a_k t(H_k, G)$ hold for all finite graphs G?

Hatami and Norine showed this problem is **undecidable**.

Abstract Games

Definition (Game framework)

Given $m \in \mathbb{Z}^+$, and a computable function $W : \mathbb{N}^m \to \{A, B\}$, we can consider the two-player "game of no chance", in which:

1. The players alternately choose natural numbers, starting with A, and ending after m numbers $x_1, ..., x_m$ have been chosen.
2. There is perfect information: both players know the rules and can see all previously made choices.
3. The winner is $W(x_1, x_2, ..., x_m)$.

Many games can be fit into this framework.

Winning Strategy 1

Theorem

In this game framework, given m and W, exactly one of two players has a winning strategy.

Winning Strategy 1

Theorem

In this game framework, given m and W, exactly one of two players has a winning strategy.

Proof:

Simply recursively simulate the game (Minimax Algorithm). □

Winning Strategy 1

Theorem

In this game framework, given m and W, exactly one of two players has a winning strategy.

Proof:

Simply recursively simulate the game (Minimax Algorithm). □

Theorem

In this game framework, given m and W, it is impossible to decide which player has a winning strategy.

Winning Strategy 1

Theorem

In this game framework, given m and W, exactly one of two players has a winning strategy.

Proof:

Simply recursively simulate the game (Minimax Algorithm). □

Theorem

In this game framework, given m and W, it is impossible to decide which player has a winning strategy.

Proof:

Given program p, consider the one-move game in which A chooses a positive integer x_1 and wins iff p halts within the first x_1 steps. A has a winning strategy iff p halts, which is undecidable. □

Winning Strategy 2

Definition (Simple sets)

A set S is **simple** iff $S \subset \mathbb{N}$ and S is **listable**.

Winning Strategy 2

Definition (Simple sets)

A set S is **simple** iff $S \subset \mathbb{N}$ and S is **listable**.

Lemma

There exists a simple set S whose complement $\overline{S}$ infinite, but contains no infinite listable set.

Winning Strategy 2

Definition (Simple sets)

A set S is **simple** iff $S \subset \mathbb{N}$ and S is **listable**.

Lemma

There exists a simple set S whose complement $\overline{S}$ infinite, but contains no infinite listable set.

Proof: Omitted.

Winning Strategy 2

Definition (Simple sets)

A set S is **simple** iff $S \subset \mathbb{N}$ and S is **listable**.

Lemma

There exists a simple set S whose complement $\overline{S}$ infinite, but contains no infinite listable set.

Proof: Omitted.

Theorem

There exists a three-move game in which player B has a wining strategy, but not a computable winning strategy.

Winning Strategy 3

Proof:

Fix a simple set S such that whose complement $\overline{S}$ infinite, but contains no infinite listable set, that exists due to our lemma.

Winning Strategy 3

Proof:

Fix a simple set S such that whose complement $\overline{S}$ infinite, but contains no infinite listable set, that exists due to our lemma.

Consider the three-move game in which A wins iff $x_1 + x_2 = g(x_3)$.

Winning Strategy 3

Proof:

Fix a simple set S such that whose complement $\overline{S}$ infinite, but contains no infinite listable set, that exists due to our lemma.

Consider the three-move game in which A wins iff $x_1 + x_2 = g(x_3)$.

Player B's winning strategy is to find $t \in \overline{S}$ with $t > x_1$, and to choose $x_2 = t - x_1$.

Winning Strategy 3

Proof:

Fix a simple set S such that whose complement $\overline{S}$ infinite, but contains no infinite listable set, that exists due to our lemma.

Consider the three-move game in which A wins iff $x_1 + x_2 = g(x_3)$.

Player B's winning strategy is to find $t \in \overline{S}$ with $t > x_1$, and to choose $x_2 = t - x_1$.

So, we have computable winning strategy $x_2 = w(x_1)$. However, this yields an infinite listable subset $\{x_1 + w(w_1) \mid x_1 \in \mathbb{N}\}$ of $\overline{S}$, so it must be the case that computable w cannot exist. □

Infinite Chess

Definition (Infinite chess problem)

Input: A finite list of chess pieces and their starting positions on a $\mathbb{Z} \times \mathbb{Z}$ chess board.
Output: Can White force mate?

Infinite Chess

Definition (Infinite chess problem)

Input: A finite list of chess pieces and their starting positions on a $\mathbb{Z} \times \mathbb{Z}$ chess board.
Output: Can White force mate?

Open research: It is unknown if this is **undecidable**!

Infinite Chess

Definition (Infinite chess problem)

Input: A finite list of chess pieces and their starting positions on a $\mathbb{Z} \times \mathbb{Z}$ chess board.
Output: Can White force mate?

Open research: It is unknown if this is **undecidable**!

Theorem

One can decide if White can mate in $n < \infty$ moves, given some n and starting configuration.

Infinite Chess

Definition (Infinite chess problem)

Input: A finite list of chess pieces and their starting positions on a $\mathbb{Z} \times \mathbb{Z}$ chess board.
Output: Can White force mate?

Open research: It is unknown if this is **undecidable**!

Theorem

One can decide if White can mate in $n < \infty$ moves, given some n and starting configuration.

Proof:
This can be shown by encoding each instance if the problem as a first-order sentence in the **Presburger arithmetic**. □

Groups

Let G and H be groups. We recap some definitions.

Definition (Conjugacy)

Let $g_1, g_2 \in G$. Then $g_1 \sim g_2$ iff $\exists h \in G, g_1 = hg_2h^{-1}$.

Definition (Isomorphism)

$G \cong H$ iff there exists a group isomorphism $G \to H$.

Groups

Let G and H be groups. We recap some definitions.

Definition (Conjugacy)

Let $g_1, g_2 \in G$. Then $g_1 \sim g_2$ iff $\exists h \in G, g_1 = hg_2h^{-1}$.

Definition (Isomorphism)

$G \cong H$ iff there exists a group isomorphism $G \to H$.

We ask, is there an algorithm to:

1. recognise the identity of a group?
2. decide if two elements are conjugate?
3. decide of two groups are isomorphic?

Finitely Presented Groups

Finite Presentation

We can describe the symmetric group of order 6:

$$S_3 = \langle r, t \mid r^3 = 1, r^2 = 1, trt^{-1} = r^{-1} \rangle$$

Geometrically, we have the group of symmetries of an equilateral triangle as being generated by a 120° rotation r, and a reflection t.

Finitely Presented Groups

Finite Presentation

We can describe the symmetric group of order 6:

$$S_3 = \langle r, t \mid r^3 = 1, r^2 = 1, trt^{-1} = r^{-1} \rangle$$

Geometrically, we have the group of symmetries of an equilateral triangle as being generated by a 120° rotation r, and a reflection t.

Definition (Finitely presented group)

Any group arising in this way is called a **finitely presented group**.

Finitely Presented Groups

Finite Presentation

We can describe the symmetric group of order 6:

$$S_3 = \langle r, t \mid r^3 = 1, r^2 = 1, trt^{-1} = r^{-1} \rangle$$

Geometrically, we have the group of symmetries of an equilateral triangle as being generated by a 120° rotation r, and a reflection t.

Definition (Finitely presented group)

Any group arising in this way is called a **finitely presented group**.

Definition (Word)

An element of a finitely presented group can be specified by giving a **word** in the generators: a finite sequence of the generators.

The Word Problem

Definition (Word problem)

Input: A word w in the generators of G.
Output: Does w represent 1_G?

The Word Problem

Definition (Word problem)

Input: A word w in the generators of G.
Output: Does w represent 1_G?

The decidability of the word problem actually depends only the isomorphism type of the group, not on the presentation. There are classes of groups in which the word problem is decidable (eg finite groups, finitely presented Abelian groups).

The Word Problem

Definition (Word problem)

Input: A word w in the generators of G.
Output: Does w represent 1_G?

The decidability of the word problem actually depends only the isomorphism type of the group, not on the presentation. There are classes of groups in which the word problem is decidable (eg finite groups, finitely presented Abelian groups).

Theorem (Novikov and Boone)

There exists a finitely presented group for which the word problem is undecidable.

The Conjugacy Problem

Definition (The conjugacy problem)

Input: Words w_1, w_2 in the generators of group G.
Output: Is $w_1 \sim w_2$?

The Conjugacy Problem

Definition (The conjugacy problem)

Input: Words w_1, w_2 in the generators of group G.
Output: Is $w_1 \sim w_2$?

Actually, the word problem can be viewed as a subproblem of the conjugacy problem, consisting of the instance for which w_2 is the empty word. Thus, the conjugacy problem is at least as hard as the word problem for G.

The Conjugacy Problem

Definition (The conjugacy problem)

Input: Words w_1, w_2 in the generators of group G.
Output: Is $w_1 \sim w_2$?

Actually, the word problem can be viewed as a subproblem of the conjugacy problem, consisting of the instance for which w_2 is the empty word. Thus, the conjugacy problem is at least as hard as the word problem for G.

Theorem

There exists a finitely presented group for which the conjugacy problem is undecidable.

Properties of Groups

One can ask about algorithms that accept a finite presentation as input, and try to decide if the group it defines has a given property.

Properties of Groups

One can ask about algorithms that accept a finite presentation as input, and try to decide if the group it defines has a given property.

Markov Properties

For a wide range of natural properties, the decision problem turns out to be undecidable! Examples of such properties are: being trivial, finite, Abelian, nilpotent, solvable, or free.

Properties of Groups

One can ask about algorithms that accept a finite presentation as input, and try to decide if the group it defines has a given property.

Markov Properties

For a wide range of natural properties, the decision problem turns out to be undecidable! Examples of such properties are: being trivial, finite, Abelian, nilpotent, solvable, or free.

Theorem

The group isomorphism problem is undecidable.

Properties of Groups

One can ask about algorithms that accept a finite presentation as input, and try to decide if the group it defines has a given property.

Markov Properties

For a wide range of natural properties, the decision problem turns out to be undecidable! Examples of such properties are: being trivial, finite, Abelian, nilpotent, solvable, or free.

Theorem

The group isomorphism problem is undecidable.

Proof:
Deciding triviality is a subproblem of the problem of deciding whether two finite presentations define isomorphic groups. □

Hilbert's Tenth Problem 1

This is one of the 23 problems Hilbert famously published after a lecture in 1900, asking for an algorithm to decide the solvability of diophantine equations.

Hilbert's Tenth Problem 1

This is one of the 23 problems Hilbert famously published after a lecture in 1900, asking for an algorithm to decide the solvability of diophantine equations.

Definition (Hilbert's tenth problem)

Input: A multivariable polynomial $f \in \mathbb{Z}[x_1, ..., x_n]$.
Output: Does there exist an $\overrightarrow{a} \in \mathbb{Z}^n$ with $f(\overrightarrow{a}) = 0$?

This was eventually proved undecidable by Matiyasvich.

Hilbert's Tenth Problem 1

This is one of the 23 problems Hilbert famously published after a lecture in 1900, asking for an algorithm to decide the solvability of diophantine equations.

Definition (Hilbert's tenth problem)

Input: A multivariable polynomial $f \in \mathbb{Z}[x_1, ..., x_n]$.
Output: Does there exist an $\overrightarrow{a} \in \mathbb{Z}^n$ with $f(\overrightarrow{a}) = 0$?

This was eventually proved undecidable by Matiyasvich.

Definition (Diophantine)

We call $A \subseteq \mathbb{Z}$ **diophantine** iff there exists a polynomial $p(t, \overrightarrow{x}) \in \mathbb{Z}[t, x_1, ..., x_n]$ s.t. $A = \{a \in \mathbb{Z} \mid \exists \overrightarrow{x} \in \mathbb{Z}, p(a, \overrightarrow{x}) = 0\}$.

Hilbert's Tenth Problem 2

Theorem

A subset of $\mathbb{Z}$ is diophantine iff it is listable!

Hilbert's Tenth Problem 2

Theorem

A subset of $\mathbb{Z}$ is diophantine iff it is listable!

Proof:

Clearly diophantine sets are listable. The converse was shown by Matiyasvich by expressing exponentiation in diophantine terms. □

Hilbert's Tenth Problem 2

Theorem

A subset of $\mathbb{Z}$ is diophantine iff it is listable!

Proof:
Clearly diophantine sets are listable. The converse was shown by Matiyasvich by expressing exponentiation in diophantine terms. □

Corollary

Hilbert's tenth problem is undecidable.

Hilbert's Tenth Problem 2

Theorem

A subset of $\mathbb{Z}$ is diophantine iff it is listable!

Proof:
Clearly diophantine sets are listable. The converse was shown by Matiyasvich by expressing exponentiation in diophantine terms. □

Corollary

Hilbert's tenth problem is undecidable.

Proof:
Recall from that there are simple A for which there is no algorithm to decide if a given integer belongs to A. With this in mind, the result is immediate from our above theorem. □