Seminari lògica: Gödel i Turing

Marc Herault i Edgar Moreno

Primavera 2022

Però que volem de veritat?

Però que volem de veritat?

- Completesa: tot el que és cert es pot provar.

Però que volem de veritat?

- Completesa: tot el que és cert es pot provar.
- Consistència: no hi ha proposicions contradictòries. Principi d'explosió.

Però que volem de veritat?

- Completesa: tot el que és cert es pot provar.
- Consistència: no hi ha proposicions contradictòries. Principi d'explosió.
- Decidibilitat: donat una proposició és pot decidir (mitjançant un algorisme) si aquesta és certa.

Primer teorema de Gödel

Teorema (1931)

Tot sistema lògic consistent (recursivament enumerable) capaç de fer aritmètica bàsica (amb el enters) és incomplet.

Interpretació

$\mathbb{R}$ o $\mathbb{C}$ sí que es poden definir de forma completa.
Podem incloure altres axiomes per a que el nostre conjunt de axiomes original sigui complet, però no ho serà el nou sistema.

Sketch de la prova

Nombre de Gödel

Donem als caràcters bàsics $(0S =< + \times ()x * \exists \forall \vee \wedge \neg)$ un natural únic sense 0.

Ho estenem a cadenes intercalant 0.

Ho estenem a llistes de cadenes intercalant 00.

Sketch de la prova

Nombre de Gödel

Donem als caràcters bàsics $(0S =< + \times ()x * \exists \forall \vee \wedge \neg)$ un natural únic sense 0.
Ho estenem a cadenes intercalant 0.
Ho estenem a llistes de cadenes intercalant 00.

Regla de deducció

Si S_1 i S_2 impliquen per regla lògica (R) S escrivim nRm on $n = G(S_1, S_2)$ i $m = G(S_1, S_2, S)$.
Tota cosa demostrable és un axioma o es pot provar amb un nombre finit de R començant amb els axiomes. Ara tota prova te un nombre de Gödel.
Definim Proof(x, y) si $x = G(\text{prova de S})$ i $y = G(S)$.

Sketch de la prova

Autoreferència

Per $n \in \mathbb{N}$ i $F(y)$ una fòrmula que depen de un paràmetre (F pot ser: y és primer) definim:

$$q(n, G(F)) \iff \neg \text{Proof}(n, G(F(G(F))))$$

Donat n, m podem provar $q(n, m)$ o $\neg q(n, m)$ en un nombre finit de passos.
Definim $P(x) = \forall y, q(y, x)$
$P(G(P)) = \forall y, q(y, G(P))$

Segon teorema de Gödel

Teorema (1931)

Cap sistema lògic consistent es capaç de fer aritmètica bàsica és capaç de provar la seva pròpia consistència.

Que és un algorisme?

Màquina de Turing

Model bàsic però equivalent als llenguatges de programació moderns.
$T = \{\Gamma, b, \Sigma, Q, q_0, A, \delta\}$
$\Gamma =$ alfabet, $b \in \Gamma$ el blanc, $\Sigma \subset \Gamma$ els símbols d'entrada, Q estats, $q_0 \in Q$ estat inicial, $A \subset Q$ estats acceptadors, $\delta : Q \times \Gamma \to Q \times \Gamma \times \{-1, 1\}$

Que es pot computar?

Llenguatge reconegut

Donada T el seu llenguatge $L_T \subseteq \Sigma^*$ és el conjunt de paraules d'entrada tal que la maquina T para (arriba a estat acceptador) en temps finit (no importa la complexitat).

Llenguatge decidible

L és un llenguatge decidible si existeix maquina de Turing T tal que $L = L_T$.

Cositas

Teorema: Maquina mestre

Hi ha una maquina de Turing que donada una màquina de Turing T (codificada de alguna manera adient, numerem màquines igual que feiem proposicions) i una entrada x simula la execució $T(x)$.

Cositas

Teorema: Maquina mestre

Hi ha una maquina de Turing que donada una màquina de Turing T (codificada de alguna manera adient, numerem màquines igual que feiem proposicions) i una entrada x simula la execució $T(x)$.

Turing Completesa

Multitud de coses equivalents a una màquina de Turing:

- C++
- Qualsevol llenguatge amb if's, salts i memòria infinita.
- PowerPoint
- Game of Life (Conway)
- Minecraft
- Certes colònies de formigues

Entscheidungsproblem

Problema de decisió (Hilbert i Ackermann 1928)

Hi ha un algorisme que donada una proposició respon en temps finit si aquest és conseqüència lògica de un conjunt d'axiomes.

Entscheidungsproblem

Problema de decisió (Hilbert i Ackermann 1928)

Hi ha un algorisme que donada una proposició respon en temps finit si aquest és conseqüència lògica de un conjunt d'axiomes.

Teorema (Church i Turing independement, 1936)

No existeix l'algorisme.

Entscheidungsproblem

Problema de decisió (Hilbert i Ackermann 1928)

Hi ha un algorisme que donada una proposició respon en temps finit si aquest és conseqüència lògica de un conjunt d'axiomes.

Teorema (Church i Turing independement, 1936)

No existeix l'algorisme.

Church–Turing thesis

Teorema: El λ-càlcul (Church) i les màquines de Turing computen les mateixes funcions.
Assumpció: Les maquines de Turing capturen el que entenem per algorisme.

Halting Problem

Teorema

No hi ha cap màquina de Turing que decideixi si una màquina de Turing amb una entrada donada para.

Halting Problem

Teorema

No hi ha cap màquina de Turing que decideixi si una màquina de Turing amb una entrada donada para.

Demostració (autoreferència !?!?!?)

Sigui T la màquina que decideix el Halting Problem.

```
def g():
    if T(g):
        while(true)
```

Halting Problem

Teorema

No hi ha cap màquina de Turing que decideixi si una màquina de Turing amb una entrada donada para.

Demostració (autoreferència !?!?!?)

Sigui T la màquina que decideix el Halting Problem.

```
def g():
    if T(g):
        while(true)
```

Entscheidungsproblem

Podem expressar el concepte de que una màquina pari com una premissa en el nostre sistema lògic, i per tant no hi ha algorisme que la decideixi.

Halting Problem ho soluciona tot

Per pensar: Si tinguessiu una màquina de Turing que resol Halting Problem, resoleu la conjectura de Collatz.

- conjectura de Goldbach?
- hipòtesi de Riemann?

Busy Beaver numbers

Joc n-busy beaver

Donar una màquina de Turing de n estats i un halting state. Ha de parar si l'entrada és tot 0.

n-BB

La màquina que més 1 escriu al joc n-busy beaver.

Σ

$\Sigma(n)$ la màxima puntuació conseguible al n-busy beaver.
Creix més ràpid que qualssevol funció computable.

Busy beaver number

Values of $\Sigma(n, m)$						
m \ n	2-state	3-state	4-state	5-state	6-state	7-state
2-symbol	4	6	13	4098 ?	$> 3.5 \times 10^{18\,267}$	$> 10^{10^{10^{10^{18\,705\,353}}}}$
3-symbol	9	$\geq 374\,676\,383$	$> 1.3 \times 10^{7036}$	?	?	?
4-symbol	≥ 2050	$> 3.7 \times 10^{6518}$	?	?	?	?
5-symbol	$> 1.7 \times 10^{352}$	?	?	?	?	?
6-symbol	$> 1.9 \times 10^{4933}$	?	?	?	?	?

Tan fort com Halting problem!

Existeix una màquina de Turing de $27 + 1$ estats que es para si i només si la conjectura de Goldbach (tot parell > 2 és suma de 2 primers) és falsa. Sabent $BB(27)$ pots resoldre la conjectura de Goldbach, simplement fes anar la màquina $BB(27)$ passos i mira si ha parat.

Navier-Stokes

Equacions de Navier Stokes

EDPs que modelen com (creiem) que evolucionen els fluids.

Pregunta (10^6 dòlars)

Les solucions són existeixen, són úniques i "boniques"? Realment són un model correcte?

Navier-Stokes

Equacions de Navier Stokes

EDPs que modelen com (creiem) que evolucionen els fluids.

Pregunta (10^6 dòlars)

Les solucions són existeixen, són úniques i "boniques"? Realment són un model correcte?

Estratègia

Si som capaços de construir una màquina de Turing (Maquina mestre) utilitzant aquestes equacions sabem que el seu comportament serà indecidible...
Poques esperances de que les solucions estiguin ben definides i semblants.

Feina de la UPC! Robert Cardona i Eva Miranda tenen els últims resultats en el camp obert per Terence Tao (i altres).