Indecidibilità della logica predicativa diadica

Eugenio G. Omodeo

UNIVERSITÀ
DEGLI STUDI DI TRIESTE

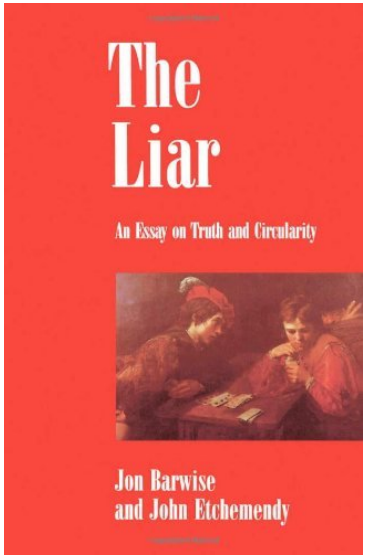
Dip. Matematica e Geoscienze — DMI

Trieste, 24.05.2022

Il paradosso del mentitore (Epimenide di Creta, 600 a.C.)

Lettera di San Paolo a Tito I, 12-13:

> *"[11]A questi tali bisogna chiudere la bocca, perché mettono in scompiglio intere famiglie, insegnando per amore di un guadagno disonesto cose che non si devono insegnare. [12]Uno dei loro, proprio un loro profeta, già aveva detto: «I Cretesi son sempre bugiardi, male bestie, ventri pigri». [13]Questa testimonianza è vera. Perciò correggili con fermezza,"*



Affermazioni *auto-referenziali*

Questa frse contiene tre erori.

Affermazioni *auto-referenziali*

Questa frse contiene tre erori.

$$\{ x \mid x \notin x \}$$

Affermazioni *auto-referenziali*

Questa frse contiene tre erori.

$$\{ x \mid x \notin x \} \stackrel{?}{\in} \{ x \mid x \notin x \}$$

Auto-referenza e computabilità

"*Il fenomeno di un programma che agisce sulla propria descrizione viene talvolta chiamato* auto-referenza *ed è fonte di molti risultati fondamentali in teoria della computabilità. L'intero punto della diagonalizzazione,* [···]*, è in effetti quello di ottenere un'auto-referenza contraddittoria.*"

[DSW94, pag. 98]

Auto-referenza e computabilità

"Il fenomeno di un programma che agisce sulla propria descrizione viene talvolta chiamato auto-referenza *ed è fonte di molti risultati fondamentali in teoria della computabilità. L'intero punto della* diagonalizzazione, *[···], è in effetti quello di ottenere un'auto-referenza contraddittoria."*

[DSW94, pag. 98]


Not logged in Talk Contributions Create account Log i
Visit the main page
WIKIPEDIA
The Free Encyclopedia
Article Talk Read Edit View history
Search Wikipedia
Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store
Entscheidungsproblem
From Wikipedia, the free encyclopedia
In mathematics and computer science, the Entscheidungsproblem (pronounced [ɛntˈʃaɪ̯dʊŋspʁoˌbleːm], German for "decision problem") is a challenge posed by David Hilbert and Wilhelm Ackermann in 1928.[1] The problem asks for an algorithm that considers, as input, a statement and answers "Yes" or "No" according to whether the statement is universally valid, i.e., valid in every structure satisfying the axioms.

Ipotesi di partenza

Supponiamo vi sia un algoritmo in grado di rispondere *sempre*, in modo *corretto*, al quesito

$$\mathcal{K} \overset{?}{\models} \exists\, y\; q(y\,,\, \vec{\boldsymbol{n}})\,.$$

finito

Dapprima

Ipotesi di partenza

Supponiamo vi sia un algoritmo in grado di rispondere *sempre*, in modo *corretto*, al quesito

$$\mathcal{K} \overset{?}{\models} \exists y\, q(y, \vec{n})\,.$$

finito

Dapprima

1) 'trasferiremo' questo assunto dalle clausole Horn al dominio $\mathbb{N}$;

poi

Ipotesi di partenza

Supponiamo vi sia un algoritmo in grado di rispondere *sempre*, in modo *corretto*, al quesito

$$\mathcal{K} \overset{?}{\models} \exists y\, q(y, \vec{n})\,.$$

finito

Dapprima

1) 'trasferiremo' questo assunto dalle clausole Horn al dominio $\mathbb{N}$;

poi

2) ne trarremo una contraddizione, donde l'importante *risultato limitativo*:

Teorema di Church e sua dimostrazione

$\longrightarrow$

Teorema di Church, 1936

Non c'è algoritmo in grado di stabilire se valga o meno

$$\vdash \quad \alpha$$

per ogni dato enunciato α nel calcolo predicativo del $1^{\underline{o}}$ ordine.

Teorema di Church e sua dimostrazione

$\longrightarrow$

Teorema di Church, 1936

Non c'è algoritmo in grado di stabilire se valga o meno

$$\vdash \quad \alpha$$

per ogni dato enunciato α nel calcolo predicativo del $1^{\underline{o}}$ ordine.

Dim.:

Una volta ottenuta la contraddiz. cercata, otterremo per il teor. di completezza e in base al principio di deduzione, l'indecidibilità del problema

$$\vdash \quad \underbrace{(\wedge \mathcal{K}) \ \rightarrow \ \exists\, y \; q(y\,,\, \vec{\boldsymbol{n}})}_{\alpha} \ .$$

(Per il teor. di Church, occorre vi siano relatori di arità almeno 2).

Chiariamo l'assunto iniziale:

"*un algoritmo è in grado di dirci se* $\mathcal{K} \models \exists\, y\; q(y\,,\, \vec{n})$ *o no*"

Stiamo supponendo che:

- $\mathcal{K}$ stia per una base (finita) di clausole di Horn;

Chiariamo l'assunto iniziale:

" *un algoritmo è in grado di dirci se* $\mathcal{K} \models \exists\, y\; q(y\,,\, \vec{n})$ *o no* "

Stiamo supponendo che:

- $\mathcal{K}$ stia per una base (finita) di clausole di Horn;
- $\neg\, q(\,Y\,,\, \vec{n}\,)$ per una *meta* (o '*goal*') mono-letterale, ove

Chiariamo l'assunto iniziale:

" *un algoritmo è in grado di dirci se* $\mathcal{K} \models \exists\, y\; q(y\,,\, \vec{n})$ *o no* "

Stiamo supponendo che:

- $\mathcal{K}$ stia per una base (finita) di clausole di Horn;
- $\neg\, q(Y\,,\, \vec{n}\,)$ per una *meta* (o '*goal*') mono-letterale, ove
- $\vec{n}$ sta per una sequenza

$$\underline{n_1}\,, \ldots, \underline{n_a}\,,$$

con $a \geqslant 1$, $a+1$ arità di q , di NUMERALI

Chiariamo l'assunto iniziale:
" *un algoritmo è in grado di dirci se* $\mathcal{K} \models \exists\, y\; q(y\,,\, \vec{n})$ *o no* "

Stiamo supponendo che:

- $\mathcal{K}$ stia per una base (finita) di clausole di Horn;
- $\neg\, q(Y\,,\, \vec{n}\,)$ per una *meta* (o '*goal*') mono-letterale, ove
- $\vec{n}$ sta per una sequenza

$$\underline{n}_1\,, \ldots, \underline{n}_a\,,$$

con $a \geqslant 1$, $a+1$ arità di q , di NUMERALI intesi così:

$$\begin{aligned} \underline{0} \quad &=_{\text{Def}} \quad 0\,, \\ \underline{m+1} \quad &=_{\text{Def}} \quad \mathsf{s}(\,\underline{m}\,) \end{aligned}$$

(per m che varia in $\mathbb{N}$);

Chiariamo l'assunto iniziale:
" *un algoritmo è in grado di dirci se* $\mathcal{K} \models \exists\, y\; q(y\,,\, \vec{n})$ *o no* "

Stiamo supponendo che:

- $\mathcal{K}$ stia per una base (finita) di clausole di Horn;
- $\neg\, q(\,Y\,,\, \vec{n}\,)$ per una *meta* (o '*goal*') mono-letterale, ove
- $\vec{n}$ sta per una sequenza

$$\underline{n_1}\,, \ldots, \underline{n_a}\,,$$

con $a \geqslant 1$, $a+1$ arità di q , di NUMERALI intesi cosí:

$$\begin{aligned} \underline{0} \quad &=_{\text{Def}} \quad 0\,, \\ \underline{m+1} \quad &=_{\text{Def}} \quad \mathsf{s}(\,\underline{m}\,) \end{aligned}$$

(per m che varia in $\mathbb{N}$);

- $0_{/0}$ ed $\mathsf{s}_{/1}$ siano i soli costrutti di Herbrand.

Osservazioni preliminari

Il fatto che $\mathcal{K}$ sia di Horn ci garantisce l'esistenza del *modello minimo*; grazie a ciò,

Osservazioni preliminari

Il fatto che $\mathcal{K}$ sia di Horn ci garantisce l'esistenza del *modello minimo*; grazie a ciò,

$$\mathcal{K} \models \exists y\, q(y, \vec{\boldsymbol{n}})$$

equivale all'esistenza di un $\boldsymbol{y}$ in $\mathbb{N}$ tale che

$$\mathcal{K} \models q(\underline{\boldsymbol{y}}, \underline{\boldsymbol{n}_1}, \ldots, \underline{\boldsymbol{n}_a}) .$$

Osservazioni preliminari

Il fatto che $\mathcal{K}$ sia di Horn ci garantisce l'esistenza del *modello minimo*; grazie a ciò,

$$\mathcal{K} \models \exists y\, q(y, \vec{n})$$

equivale all'esistenza di un y in $\mathbb{N}$ tale che

$$\mathcal{K} \models q(\underline{y}, \underline{n_1}, \ldots, \underline{n_a}) .$$

Ai nostri fini basterà come

$$\vec{n} \quad \text{una coppia di forma} \quad \underline{m}, \underline{m} ;$$

stiamo, in effetti, per ricorrere alla cosiddetta '**diagonalizzazione**'.

Do per scontato che possiamo rappresentare, tramite un numero naturale, qualsiasi sequenza

$$g_0, \quad \ldots, \quad g_M$$

certificante che g_M è funzione *ricorsiva generale*[1] ai sensi della definizione che ne abbiamo data in precedenza. In altre parole:

[1] $\therefore$ computabile parziale

Do per scontato che possiamo rappresentare, tramite un numero naturale, qualsiasi sequenza

$$g_0, \quad \ldots, \quad g_M$$

certificante che g_M è funzione *ricorsiva generale*[1] ai sensi della definizione che ne abbiamo data in precedenza. In altre parole:

1a) Data una tal seq. di g_i, possiamo calcolare un numero che la rappresenti.

[1] $\therefore$ computabile parziale

Do per scontato che possiamo rappresentare, tramite un numero naturale, qualsiasi sequenza

$$g_0, \quad \ldots, \quad g_M$$

certificante che g_M è funzione *ricorsiva generale*[1] ai sensi della definizione che ne abbiamo data in precedenza. In altre parole:

1a) Data una tal seq. di g_i, possiamo calcolare un numero che la rappresenti.

1b) Di converso, dato un m in $\mathbb{N}$, possiamo concretam. stabilire se m codifichi o meno una tal seq., della quale...

[1] $\therefore$ computabile parziale

Do per scontato che possiamo rappresentare, tramite un numero naturale, qualsiasi sequenza

$$g_0\,, \quad \ldots, \quad g_M$$

certificante che g_M è funzione *ricorsiva generale*[1] ai sensi della definizione che ne abbiamo data in precedenza. In altre parole:

1a) Data una tal seq. di g_i, possiamo calcolare un numero che la rappresenti.

1b) Di converso, dato un m in $\mathbb{N}$, possiamo concretam. stabilire se m codifichi o meno una tal seq., della quale...

1c) ...in caso affermativo, possiamo individuare:

- la lunghezza $M+1$,
- ciascuna componente g_i;

[1] $\therefore$ computabile parziale

1d) . . . in caso affermativo . . .

- siamo anche in grado di ricavare da $\boldsymbol{m}$ una base $\mathcal{B}$ di clausole di Horn che *rappresenti* la funzione parziale[2]

$$\boldsymbol{x} \quad \mapsto \quad \boldsymbol{g}_M(\boldsymbol{x}, 0, \ldots, 0)\,.$$

tramite un predicato diadico

$$p(Y, X)\,,$$

[2] Ai nostri fini correnti, non serve considerare funzioni $\boldsymbol{g}_M$ di arità > 1: quando $\boldsymbol{g}_M$ è a piú argomenti, la rimpiazziamo con la funzione qui indicata, ottenibile da essa per composiz. con funzioni iniziali.

1d) . . . in caso affermativo . . .

- siamo anche in grado di ricavare da m una base $\mathcal{B}$ di clausole di Horn che *rappresenti* la funzione parziale[2]

$$x \mapsto g_M(x, 0, \ldots, 0).$$

tramite un predicato diadico

$$p(Y, X),$$

nel senso che per ogni x in $\mathbb{N}$ e ogni y in $\mathbb{N}$:

$\mathcal{B} \models \exists y\, p(y, \underline{x})$ sse $g_M(x, 0, \ldots, 0)$ ha un valore,
$\mathcal{B} \models p(\underline{y}, \underline{x})$ sse $g_M(x, 0, \ldots, 0) = y$.

[2] Ai nostri fini correnti, non serve considerare funzioni g_M di arità > 1: quando g_M è a piú argomenti, la rimpiazziamo con la funzione qui indicata, ottenibile da essa per composiz. con funzioni iniziali.

2) Contraddizione tramite il metodo diagonale

Il celebre **teorema di universalità** di Turing (1936), riferito a

- la nozione di ricorsione generale da noi qui adottata e
- al caso delle funzioni ricorsive generali *monadiche*,

ci dice che

Esistono una base $\mathcal{U}$ di clausole di Horn e un relatore triadico u tali che per ogni coppia $\boldsymbol{m}$, $\boldsymbol{x}$ di naturali tale che $\boldsymbol{m}$ codifichi una seq.

$$\boldsymbol{g}_0, \quad \ldots, \quad \boldsymbol{g}_M$$

come sopra, le affermazioni

$$\text{“ } \mathcal{U} \models \exists y \, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{x}}) \text{ ”}$$

e

$$\text{“ } \boldsymbol{g}_M(\boldsymbol{x}, 0, \ldots, 0) \text{ ha un valore ”}$$

siano equivalenti una all'altra.

Esistono una base $\mathcal{U}$ di clausole di Horn e un relatore triadico u tali che per ogni coppia $\boldsymbol{m}$, $\boldsymbol{x}$ di naturali tale che $\boldsymbol{m}$ codifichi una seq.

$$\boldsymbol{g}_0, \quad \ldots, \quad \boldsymbol{g}_M$$

come sopra, le affermazioni

$$\text{“ } \mathcal{U} \models \exists y \, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{x}}) \text{ ”}$$

e

$$\text{“ } \boldsymbol{g}_M(\boldsymbol{x}, 0, \ldots, 0) \text{ ha un valore ”}$$

siano equivalenti una all'altra. COME? 

☞ Osserviamo che il seguente procedimento è algoritmico e in corrispondenza a qualsiasi coppia m , x fornitagli in ingresso o emette un responso (da intendersi come affermativo) oppure non dà risultato:

☞ Osserviamo che il seguente procedimento è algoritmico e in corrispondenza a qualsiasi coppia m , x fornitagli in ingresso o emette un responso (da intendersi come affermativo) oppure non dà risultato:

0. estrai $\mathcal{B}$ e $p_{/2}$ dal dato m (se l'estraz. è impossibile, cicla pure all'infinito), come descritto sopra al punto **1d)**;

☞ Osserviamo che il seguente procedimento è algoritmico e in corrispondenza a qualsiasi coppia m , x fornitagli in ingresso o emette un responso (da intendersi come affermativo) oppure non dà risultato:

0. estrai $\mathcal{B}$ e $p_{/2}$ dal dato m (se l'estraz. è impossibile, cicla pure all'infinito), come descritto sopra al punto **1d)**;

1. elenca le dimostrazioni che utilizzano $\mathcal{B}$ come insieme (finito) di premesse;

☞ Osserviamo che il seguente procedimento è algoritmico e in corrispondenza a qualsiasi coppia m , x fornitagli in ingresso o emette un responso (da intendersi come affermativo) oppure non dà risultato:

0. estrai $\mathcal{B}$ e $p_{/2}$ dal dato m (se l'estraz. è impossibile, cicla pure all'infinito), come descritto sopra al punto **1d)**;

1. elenca le dimostrazioni che utilizzano $\mathcal{B}$ come insieme (finito) di premesse;

2. per ciascuna delle dimostrazioni, via via che vengono elencate, controlla se l'ultimo passo della dimostrazione sia un enunciato della forma

$$p(\underline{y}, \underline{x});$$

☞ Osserviamo che il seguente procedimento è algoritmico e in corrispondenza a qualsiasi coppia m , x fornitagli in ingresso o emette un responso (da intendersi come affermativo) oppure non dà risultato:

0. estrai $\mathcal{B}$ e $p_{/2}$ dal dato m (se l'estraz. è impossibile, cicla pure all'infinito), come descritto sopra al punto **1d)**;

1. elenca le dimostrazioni che utilizzano $\mathcal{B}$ come insieme (finito) di premesse;

2. per ciascuna delle dimostrazioni, via via che vengono elencate, controlla se l'ultimo passo della dimostrazione sia un enunciato della forma

$$p(\underline{y}, \underline{x});$$

3. ed emetti il responso—fermandoti—se questo è il caso;

☞ Osserviamo che il seguente procedimento è algoritmico e in corrispondenza a qualsiasi coppia m , x fornitagli in ingresso o emette un responso (da intendersi come affermativo) oppure non dà risultato:

0· estrai $\mathcal{B}$ e $p_{/2}$ dal dato m (se l'estraz. è impossibile, cicla pure all'infinito), come descritto sopra al punto **1d)**;

1· elenca le dimostrazioni che utilizzano $\mathcal{B}$ come insieme (finito) di premesse;

2· per ciascuna delle dimostrazioni, via via che vengono elencate, controlla se l'ultimo passo della dimostrazione sia un enunciato della forma

$$p(\underline{y}, \underline{x});$$

3· ed emetti il responso—fermandoti—se questo è il caso;

4· altrimenti protrai l'attività **1·** .

☞ Visto che di un procedim. algoritmico qui si tratta, la funzione parziale $u(m, x)$ ch'esso computa può essere certificata computabile tramite un'opportuna seq. di funzioni ricorsive generali,

⊣

☞ Visto che di un procedim. algoritmico qui si tratta, la funzione parziale $u(m, x)$ ch'esso computa può essere certificata computabile tramite un'opportuna seq. di funzioni ricorsive generali, dalla quale sappiamo come ottenere la base voluta, $\mathcal{U}$, di clausole di Horn:

⊣

☞ Visto che di un procedim. algoritmico qui si tratta, la funzione parziale $u(m, x)$ ch'esso computa può essere certificata computabile tramite un'opportuna seq. di funzioni ricorsive generali, dalla quale sappiamo come ottenere la base voluta, $\mathcal{U}$, di clausole di Horn: volendo evitare ogni ricorso alla tesi di Church-Turing, possiamo effettivam.—se sfidati—*implementare* $\mathcal{U}$.

⊣

☞ Visto che di un procedim. algoritmico qui si tratta, la funzione parziale $u(m, x)$ ch'esso computa può essere certificata computabile tramite un'opportuna seq. di funzioni ricorsive generali, dalla quale sappiamo come ottenere la base voluta, $\mathcal{U}$, di clausole di Horn: volendo evitare ogni ricorso alla tesi di Church-Turing, possiamo effettivam.—se sfidati—*implementare* $\mathcal{U}$.
(Ed, in effetti, il 'programma universale' è stato da innumerevoli autori specificato in completo dettaglio.) ⊣

Grazie ad $\mathcal{U}$, giungiamo allo scopo –I

Disponendo di un metodo effettivo in grado di dirci, per ogni coppia $\boldsymbol{m}$, $\boldsymbol{x}$, se

$$\mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{x}})$$

oppure no—*questa la nostra ipotesi di partenza*—, possiamo servircene in particolare per stabilire se

$$\mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{m}})$$

o no;

Grazie ad $\mathcal{U}$, giungiamo allo scopo –I

Disponendo di un metodo effettivo in grado di dirci, per ogni coppia $\boldsymbol{m}$, $\boldsymbol{x}$, se

$$\mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{x}})$$

oppure no—*questa la nostra ipotesi di partenza*—, possiamo servircene in particolare per stabilire se

$$\mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{m}})$$

o no; dunque la funzione

$$\boldsymbol{t}(\boldsymbol{m}) = \begin{cases} 1 & \text{se } \mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{m}}) \,, \\ 0 & \text{altrimenti} \,, \end{cases}$$

è computabile totale;

Grazie ad $\mathcal{U}$, giungiamo allo scopo –I

Disponendo di un metodo effettivo in grado di dirci, per ogni coppia $\boldsymbol{m}$, $\boldsymbol{x}$, se

$$\mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{x}})$$

oppure no—*questa la nostra ipotesi di partenza*—, possiamo servircene in particolare per stabilire se

$$\mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{m}})$$

o no; dunque la funzione

$$\boldsymbol{t}(\boldsymbol{m}) = \begin{cases} 1 & \text{se} \quad \mathcal{U} \models \exists y\, \mathrm{u}(y, \underline{\boldsymbol{m}}, \underline{\boldsymbol{m}}) \,, \\ 0 & \text{altrimenti}\,, \end{cases}$$

è computabile totale; partendo da essa otteniamo la

$$\boldsymbol{d}(\boldsymbol{m}) = \begin{cases} 0 & \text{se} \quad \boldsymbol{t}(\boldsymbol{m}) = 0\,, \\ \text{non ha val.} & \text{altrimenti}\,, \end{cases}$$

che sarà ricorsiva generale, in quanto

$$\boldsymbol{d}(\boldsymbol{m}) = \min_{x \geqslant 0} \left(\boldsymbol{t}(\mathrm{proj}_1(\boldsymbol{m}, \mathrm{x})) = 0 \right) .$$

Grazie ad $\mathcal{U}$, giungiamo allo scopo

Sia ora k il numero che codifica una seq.

$$d_0, \quad \ldots, \quad d_D$$

definiente quest'ultima $d(\_)$. Ecco la contraddizione che cercavamo:

$$\begin{array}{rll} d(k) & \text{ha un valore} & sse \\ t(k) & = 0 & sse \\ \mathcal{U} & \nvdash \exists y\, u(y, \underline{k}, \underline{k}) & sse \\ d_D(k) & \text{non ha un valore} & sse \\ d(k) & \text{non ha un valore}. & \dashv \end{array}$$

Alonzo Church (1903–1995), Alan Mathison Turing (1912–1954)

Riferimenti bibliografici


Martin D. Davis, Ron Sigal, and Elaine J. Weyuker.
Computability, complexity, and languages - Fundamentals of theoretical computer science.
Computer Science ad scientific computing. Academic Press, 1994.