Logica

1: Paradossi (meta)linguistici

Claudio Sacerdoti Coen

<sacerdot@cs.unibo.it>

Universitá di Bologna

27/09/2019

Outline

Paradossi ($\approx$ antinomie)

Antinomia: una conclusione inaccettabile, che deriva da premesse accettabili per mezzo di un ragionamento accettato"

Talvolta
Antinomia = $<<$ definizione precedente $>>$
Paradosso = conclusione contraria all'intuizione che deriva da premesse accettabili per mezzo di un ragionamento accettato

Nel corso parlerò di paradossi intendendo antinomie.

Falso paradosso (trova l'errore):

$$x = 1 \Rightarrow x^2 = x \Rightarrow x^2 - 1 = x - 1 \Rightarrow$$
$$\Rightarrow (x+1)(x-1) = x - 1 \Rightarrow x+1 = 1 \Rightarrow x = 0 \Rightarrow 1 = 0$$

Linguaggio naturale

Linguaggio naturale (italiano, francese, arabo, ...) alla base della comunicazione e del ragionamento umano.

Massimamente espressivo; spesso esteso e specializzato:

- Filosofia
- Diritto
- Politica
- Matematica “informale”

Possiamo usarlo anche per descrivere procedure di calcolo (informatica) e dimostrazioni (logica)?

Ambiguità del linguaggio naturale

Il linguaggio naturale è:

1. Ambiguo
2. Fortemente dipendente dal contesto

"*La vecchia porta la sbarra*"

"*Lucia ha perso la testa . . .*"

if la vecchia porta la sbarra **then**
  amputa(gamba,dx)
else
  amputa(gamba,sx)

Paradossi del linguaggio naturale

Il linguaggio naturale:

1. Ammette paradossi

“Io mento”

io mento se e solamente se ciò che dico non è vero
io mento se e solamente se “io mento” non è vero
io mento se e solamente se io non mento

A cosa è dovuto il paradosso?

Paradossi del linguaggio naturale

Il linguaggio naturale:

1. Ammette paradossi

Aggettivo autologico = aggettivo che si applica a se stesso (p.e. polisillabico)
Aggettivo eterologico = aggettivo che non si applica a se stesso (p.e. monosillabico)

“Eterologico è eterologico”

eterologico è eterologico sse non si applica a se stesso
eterologico è eterologico sse eterologico non è eterologico

A cosa è dovuto il paradosso?

Paradossi del linguaggio naturale

Il linguaggio naturale:

1. Ammette paradossi

“Definizione: sia x il più piccolo numero non definibile in meno di 1000 parole”

x è definito sse x è il più piccolo numero non definibile in meno di 1000 parole

x è definito (in meno di 1000 parole) sse x non è definito (in meno di 1000 parole)

A cosa è dovuto il paradosso?

Paradossi del linguaggio naturale

La causa di tutti i paradossi precedenti è comune:

1. l'uso meta-linguistico del linguaggio naturale
2. l'auto-applicazione di un concetto meta-linguistico a se stesso
3. l'uso della negazione per concludere qualcosa e la sua negazione

meta-linguistico = applicato al linguaggio, che parla del linguaggio
Cfr: meta-motore di ricerca = motore di ricerca che cerca su altri motori di ricerca
Cfr: meta-teoria = teoria che spiega un'altra teoria

Paradossi del linguaggio naturale

Come evitare i paradossi del linguaggio naturale?

1. Non si può impedire l'uso della negazione
2. Non si può impedire l'auto-applicazione
3. Come impedire l'uso meta-linguistico del linguaggio naturale?

Logica matematica (formale):

si abbandona il linguaggio naturale in favore di linguaggi artificiali (linguaggi logici, linguaggi di programmazione)

Paradossi in matematica

Nel linguaggio matematico:

1. È semplice introdurre ulteriori paradossi
2. È necessario (e possibile?) evitare i paradossi

Paradosso di Russell: “sia $X = \{Y \mid Y \notin Y\}$”

$X \in X$ sse $X \notin X$

A cosa è dovuto il paradosso?

Paradossi in matematica

Come evitare il paradosso di Russell:

1. Non è possibile formare liberamente $\{Y \mid P(Y)\}$ dove P è una proprietà qualunque (assioma errato di comprensione).
   p.e. $\{Y \mid Y$ è un ragazzo ed è biondo$\}$
2. Ma è possibile selezionare elementi da un insieme esistente X (assioma di separazione): $\{Y \in X \mid P(Y)\}$
   p.e. $\{Y \in$ insieme degli studenti $\mid Y$ è biondo$\}$
   se si sa già che la collezione di tutti gli studenti forma un insieme
3. La collezione di tutti gli insiemi NON è un insieme (ma una classe propria) (No all'uso meta-linguistico della nozione di insieme)

Quindi $\{Y \in$ classe di tutti gli insiemi $\mid Y \notin Y\}$ non è un insieme, ma una classe propria. Quindi $Y \notin Y$.

“Paradossi in informatica”

Linguaggi funzionali higher order (O’Caml, Haskell, Lisp, Scheme, Miranda, ...):
una funzione può prendere in input/dare in output altre funzioni (uso meta-linguistico delle funzioni)

Linguaggi imperativi e/o ad oggetti (C, Pascal, C++, Java, ...):
una funzione/metodo può prendere in input/dare in output puntatori/reference a funzioni/oggetti (e quindi metodi)

Linguaggio assembler:
è possibile passare l’esecuzione a un indirizzo (parola) qualsiasi

Uso meta-linguistico delle funzioni inevitabile!
Auto-applicazione inevitabile
Negazione inevitabile $\Rightarrow$ “paradossi” inevitabili

“Paradossi” in informatica

Supponiamo che tutte le funzioni $f, g, h, \ldots$ scrivibili in un linguaggio di programmazione P dato un input, restituiscano un output in un tempo finito (totalità)

Sia `f(g) = not(g(g))`

Allora `f(f) = not(f(f))` !!!

Assurdo. Pertanto:

- O f non è scrivibile in P
  e quindi P è altamente inespressivo
- Oppure f non è totale
  ovvero $f(f)$ diverge (= non restituisce alcun output in un tempo finito)

Ovvero le funzioni dei linguaggi di programmazione non sono funzioni matematiche.

"Paradossi" in informatica

Supponiamo per assurdo che esista un programma *f* che, dato un programma *g*, determini se *g* converga su *x* (↓ ovvero restituisca un output in tempo finito) o diverga su *x* (↑):

`f(g,x) = true` iff `g(x)` ↓

Sia `h(g) = if f(g,g) then` ↑ `else` ↓

Esempio:

```
h(g) = if f(g,g) then while(true) do nothing
       else return 0
```

`h(h)` ↑ sse `f(h,h) = true` sse `h(h)` ↓

Assurdo: non esiste nessun programma che decida se un altro diverga

"Paradossi" in informatica

Consideriamo un linguaggio di programmazione non tipato (p.e. Perl).
Sia T l'insieme di tutti i valori possibili (interi, booleani, funzioni, etc.)
Supponiamo che una funzione del nostro linguaggio sia una funzione matematica e viceversa.

$T = \{0, 1\} \cup T^T$
(T contiene almeno i booleani e le funzioni da un T qualunque a un T qualunque)

Assurdo! in quanto $|T| < 2 + |T^T|$ (teorema della diagonalizzazione di Cantor)

Quindi ogni linguaggio di programmazione non può esprimere tutte le funzioni matematiche!

Metodo di diagonalizzazione Cantor

La dimostrazione del teorema di Cantor è rimandata alla lezione su funzioni e relazioni in teoria degli insiemi.

Conclusioni

- Esiste una classe molto ampia di antinomie linguistiche dovute a
    1. l'uso meta-linguistico del linguaggio naturale
    2. l'auto-applicazione di un concetto meta-linguistico a se stesso
    3. l'uso della negazione per concludere qualcosa e la sua negazione
- Esse si manifestano come paradossi ineludibili in informatica
- Esse tendono a manifestarsi in matematica, dove vanno evitate
- I linguaggi logici (e linguaggi di programmazione) usati per rendere il linguaggio non ambiguo e privo di antinomie
- Matematica formale = matematica espressa in un linguaggio logico (formale)