La Meccanizzazione della Matematica

Marco Maggesi

6 marzo 2015

Seminario di Logica e Filosofia della Scienza

Dipartimento di Lettere e Filosofia - Università di Firenze

Cos’è la meccanizzazione?

Formalizzazione:

- Scrivere l’**enunciato** di un teorema in un linguaggio completamente formale.
- Scrivere la **dimostrazione** del teorema usando un insieme prefissato di regole e assiomi.

Meccanizzazione:

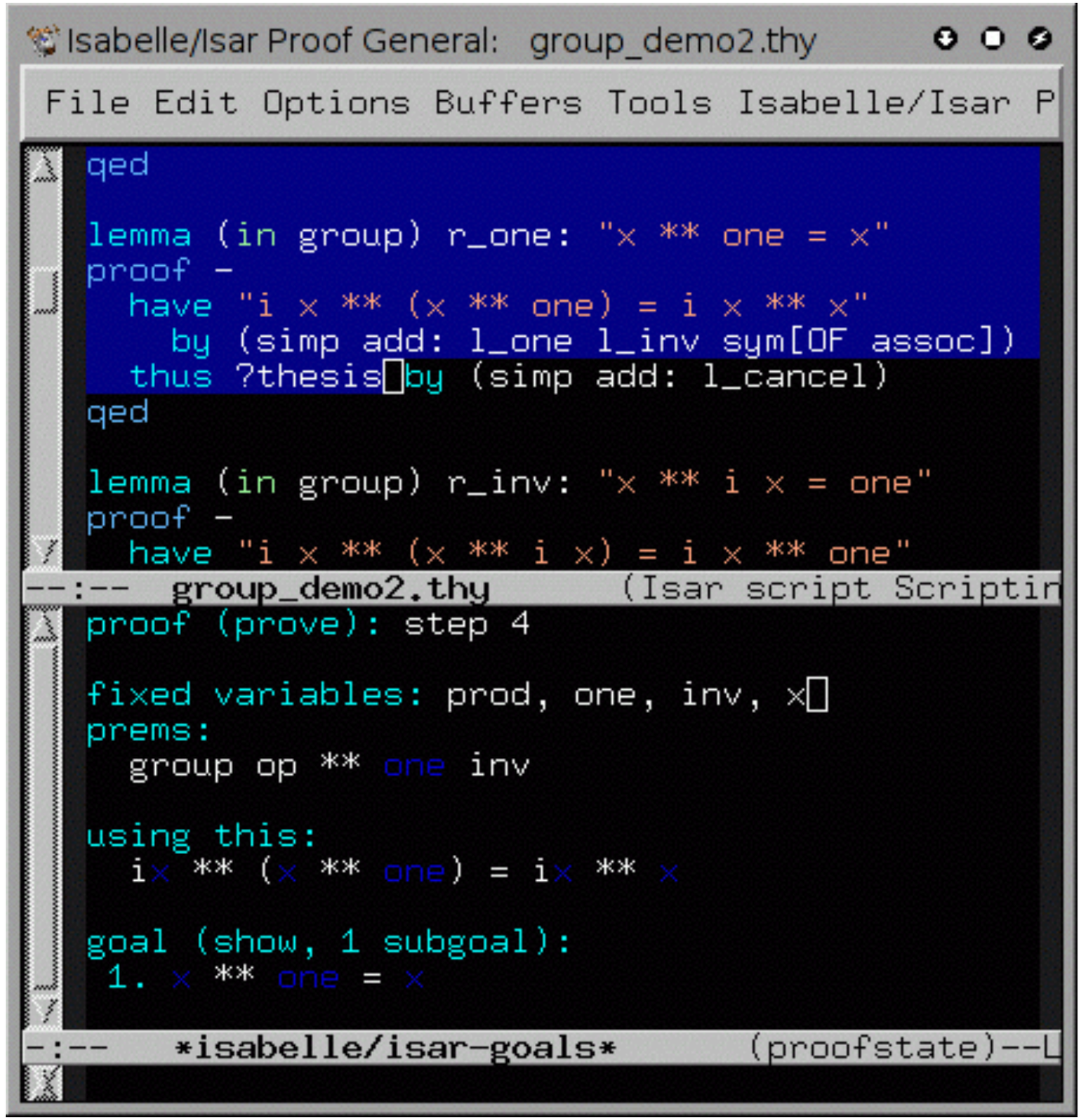
- Verificare **algoritmicamente** la dimostrazione formale usando un computer.

1	$\forall y \neg P(y)$	Assumption
2	$\exists x P(x)$	
3	u | $P(u)$	
4	$\forall y \neg P(y)$	R, 1
5	$\neg P(u)$	$\forall$E, 4
6	$\bot$	$\neg$E, 3,5
7	$\bot$	$\exists$E, 2, 3–6
8	$\neg \exists x P(x)$	$\neg$I, 2–7

Nota: la **formalizzazione è un prerequisito della meccanizzazione**, quindi parleremo di meccanizzazione per indicare l’intero processo.

Esempio:

La schermata di una sessione di lavori con Isabelle/HOL



Formalizzato o formalizzabile?

- La taglia dell dimostrazioni formali, rende impossibile *in pratica* la loro costruzione esplicita.

- *“Un testo matematico convenzionale è la descrizione informale di una dimostrazione formale.”*

- La descrizione informale convince il lettore dell’esistenza della dimostrazione formale che rimane un oggetto *virtuale*.

- Con l’avvento dei computer, la costruzione della dimostrazione formale diventa realmente possibile

Dimostrazione automatica e dimostrazione interattiva

- **Dimostrazione automatica:** L'utente propone una congettura e il programma cerca di dimostrarla o di confutarla.

- **Dimostrazione interattiva:** L'utente interagisce con il programma per creare la dimostrazione.

- Ogni sistema moderno usa una combinazione di questi due approcci.

Come codificare il ragionamento?

- **Stile imperativo:** mediante comandi sequenziali che modificano lo stato del sistema.

- **Stile dichiarativo:** mediante un linguaggio descrittivo della dimostrazione da costruire.

- Molti sistemi cercano di combinare le due alternative.

FAQ sulla meccanizzazione

- **È teoricamente possibile?**

  “Certe deduzioni sfuggono ad alla logica simbolica.”
  “È impossibile per motivi teorici (es. per il teorema di Gödel).”

- **È affidabile?**

  “Qualsiasi software contiene dei ‘bug’, anche un dimostratore di teoremi.”

- **È utile?**

  “A che serve un teorema meccanizzato?”
  “Perché dimostrare al computer teoremi già dimostrati?”

- **È fattibile in pratica?**

  “Richiede troppo tempo per essere realizzato.”
  “Richiede troppe risorse di calcolo.”

È teoricamente possibile?

- Risposta: **Sì!**
- Es: Russell & Whitehead, *Principia Matematica*

***54·43.** $\vdash :. \alpha, \beta \in 1 . \supset : \alpha \cap \beta = \Lambda . \equiv . \alpha \cup \beta \in 2$

Dem.

$\vdash . *54{\cdot}26 . \supset \vdash :. \alpha = \iota`x . \beta = \iota`y . \supset : \alpha \cup \beta \in 2 . \equiv . x \neq y .$

[*51·231] $\equiv . \iota`x \cap \iota`y = \Lambda .$

[*13·12] $\equiv . \alpha \cap \beta = \Lambda$ (1)

$\vdash . (1) . *11{\cdot}11{\cdot}35 . \supset$

$\vdash :. (\exists x, y) . \alpha = \iota`x . \beta = \iota`y . \supset : \alpha \cup \beta \in 2 . \equiv . \alpha \cap \beta = \Lambda$ (2)

$\vdash . (2) . *11{\cdot}54 . *52{\cdot}1 . \supset \vdash .$ Prop

From this proposition it will follow, when arithmetical addition has been defined, that $1 + 1 = 2$.

Primi esempi

1910 Russell, Whitehead “Principia Mathematica”

1967 de Bruijn crea Automath

1973 Trybulec crea Mizar

1977 Jutting formalizza il libro di Landau in Automath

README di Jutting

The present five volumes contain the translation of Landau's "Grundlagen der Analysis" into the formal language AUT-QE. The text has been checked mechanically on the Burroughs B6700 at the University of Technology at Eindhoven, and produced afterwards as computer output.

Volume 1 contains the logic presupposed by Landau (which includes a naive concept of sets, and the theory of equivalence classes), and the chapters 1 and 2 of Landau's book (natural numbers and positive rationals).

Volume 2 contains chapter 3 (the positive reals).

Volume 3 contains the translation of Landau's chapter 4 (the real number system) where he introduces the negative reals by "creating" a copy of the positive reals. This "creation" is translated by a number of axioms.

Volume 4 gives an alternative version of chapter 4, where arbitrary reals are introduced as equivalence classes of pairs of positive reals, thus avoiding extra axioms.

Volume 5 contains chapter 5 (the complex numbers).

For a more detailed account of the language AUT-QE, and of the translation we refer to

L.S. van Benthem Jutting
Checking Landau's "Grundlagen" in the AUTOMATH system.
Mathematical Centre Tracks 83. Amsterdam 1979.

Esempio di codice Automath

```
all:=all"l"(cut,p):'prop'
some:=some"l"(cut,p):'prop'
one:=one"e"(cut,p):'prop'
ksi@satz116:=refis(cut,ksi):is(ksi,ksi)
eta@[i:is(ksi,eta)]
satz117:=symis(cut,ksi,eta,i):is(eta,ksi)
eta@[zeta:cut][i:is(ksi,eta)][j:is(eta,zeta)]
satz118:=tris(cut,ksi,eta,zeta,i,j):is(ksi,zeta)
+1119
@[x0:rat][y0:rat][m:more(x0,y0)]
t1:=ec3e23(is"rt"(x0,y0),more(x0,y0),less(x0,y0),sa
tz81b(x0,y0),m):not(less(x0,y0))
-1119
ksi@[x0:rat][ux:urt(ksi,x0)][y0:rat][m:more(y0,x0)]
satz119:=th3"l.imp"(lrt(ksi,y0),less(y0,x0),t1".
1119"(y0,x0,m),
[t:lrt(ksi,y0)]cutapp2a(y0,t,x0,ux)):urt(ksi,y0)
```

Esempio di codice Misar

```
theorem
 for a being Real holds sin.(a-PI) = -sin.a & cos.(a-PI) = -cos.a
   proof let th be Real;
    thus sin.(th- PI) = sin.(th+(-PI)) by XCMPLX_0:def 8
  .= (sin.(th)) * (cos.(-PI)) + (cos.(th)) * ( sin.(-PI))  by SIN_COS:79
  .= (sin.(th)) * (cos.(PI))  + (cos.(th)) * (sin.(-PI))   by SIN_COS:33
  .= (sin.(th)) * (cos.(PI))  + (cos.(th)) * (-(sin.(PI))) by SIN_COS:33
  .= -(1)*sin.(th) by SIN_COS:81,XCMPLX_1:175 .= -sin.th;
 thus cos.(th- PI) = cos.(th+(-PI)) by XCMPLX_0:def 8
  .= (cos.(th)) * (cos.(-PI)) - (sin.(th)) * (sin.(-PI))   by SIN_COS:79
  .= (cos.(th)) * (cos.(PI))  - (sin.(th)) * (sin.(-PI))   by SIN_COS:33
  .= (cos.(th)) * (cos.(PI))  - (sin.(th)) * (-(sin.(PI))) by SIN_COS:33
  .= -(1)*cos.(th) by SIN_COS:81,XCMPLX_1:175 .= -cos.th;
end;
```

Alcuni esempi notevoli di teoremi meccanizzati

- Teorema dei numeri primi.
- Teorema della curva chiusa di Jordan.
- Teorema fondamentale dell'algebra (intuizionista).
- Teoremi di incompletezza di Gödel.
- Teorema dei 4 colori.
- Teorema di Feit-Thompson.
- Teorema di Hales (congettura di Keplero).

È fattibile in pratica?

Molti teoremi basilari in matematica sono stati meccanizzati:

- Analisi reale e complessa, algebra lineare, …
- Logica, combinatoria, computer science, …
- Verifica di hardware, software, protocolli, …

Un semplice esempio: il Teorema di Euclide

Teorema (Euclide). La successione dei numeri primi

2, 3, 5, 7, 11, 13, 17, ...

è infinita.

Dimostrazione. Supponiamo che la successione sia finita e consideriamo il prodotto di tutti i numeri primi più 1

$$Q := p_1 p_2 \ldots p_N + 1.$$

Ci domandiamo: Q è primo?

No, perché è più grande di tutti i numeri primi.

Sì, perché non è divisibile da nessun numero primo.

Contraddizione: la successione dei numeri primi è infinita.

CVD

Meccanizzazione del teorema di Euclide in Isabelle/Isar

```
theorem euclid: "¬finite prime" proof
  assume fin: "finite prime" def q ≡ "prod prime + 1"
  hence "q > 1" using prod_gt_one zero_not_prime fin by auto
  hence "q ∈ composed ∨ q ∈ prime" using prime_or_composed by auto
  thus False proof
    have le_q: "∀p.p ∈ prime ⟹ p < q"
    unfolding q_def using prod_le zero_not_prime fin by force
    assume "q ∈ prime" hence "q ∉ prime" using le_q by auto
    thus False by contradiction
  next
    assume "q ∈ composed"
    then obtain p where "p ∈ prime" "p < q" "p dvd q"
    using composed_dvd_prime by auto
    from ‘p ∈ prime‘ fin dvd_prod have "p dvd prod prime" by auto
    with ‘p dvd q‘ q_def dvd_plus_right have "p dvd 1" by blast
    thus False using ‘p ∈ prime‘ prime_not_invertible by auto
  qed
qed
```

È utile?

Quali sono le dimostrazioni informali sospette?

- Quelle particolarmente lunghe e involute. La classificazione dei gruppi semplici finiti.

- Quelle che coinvolgono calcoli che non possono essere svolti a mano. Il teorema dei quattro colori, La dimostrazione di Hales della congettura di Keplero.

- Quelle di natura molto tecnica, per le quali il ragionamento rigoroso è faticoso. Proof theory, verifica di hardware, di software, di protocolli, …

Errori di matematici famosi

Un libro di Maurice Lecat, *Erreurs de Mathématiciens des origines à nos jours* (1935) contiene oltre 130 pagine di errori commessi da eminenti matematici fino al 1900.

Non c'è dubbio che se un testo analogo venisse compilato ai nostri giorni, si estenderebbe per parecchi volumi.

Il teorema dei quattro colori

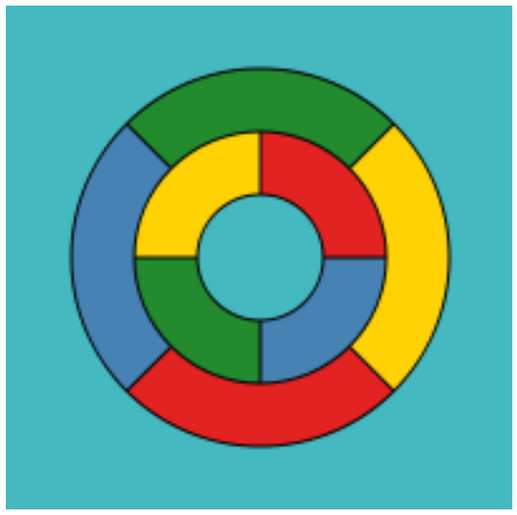
Problema:
Colorare una carta geografica con soltanto quattro colori

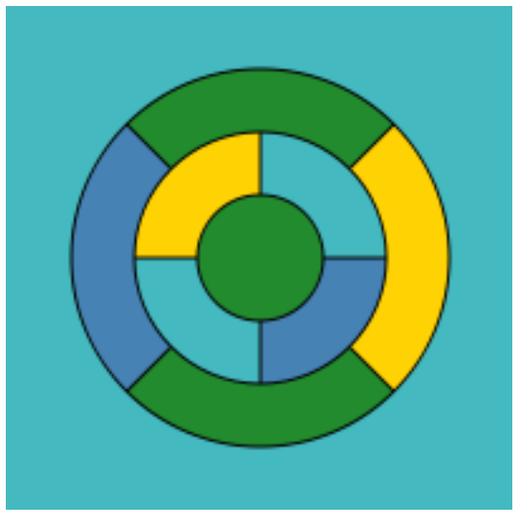
Esempio di colorazioni

Due colorazioni della stessa mappa:

La prima colorazione impiega 5 colori.

Riorganizzando la colorazione si arriva ad impiegarne 4.





Storia del problema dei quattro colori

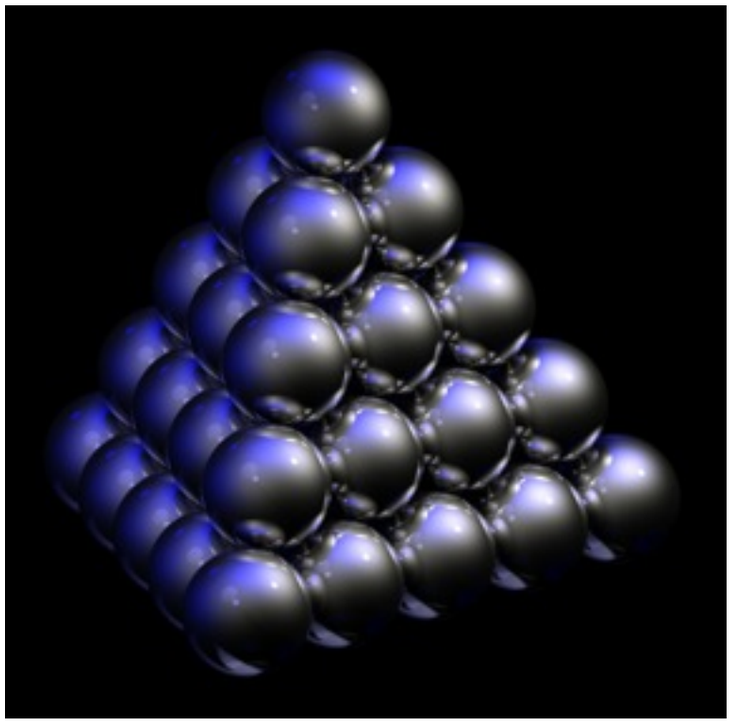
- 1852 viene formulata la congettura.
- 1879 la prima dimostrazione dovuta a Kempe.
- 1890 Heaywood trova un errore nella dimostrazione di Kempe.
- 1976 Appel e Haken danno una dimostrazione che impiega una verifica al computer.
- 2005 Meccanizzazione di Gontier in Coq.

Teorema di Feit-Thompson

- Enunciato: Tutti i gruppi finiti sono risolubili.
- 1911, congetturato da Burnside.
- 1962, dimostrazione di Feit e Thompson (255 pagine!)
- 2012, meccanizzazione di Gontier in Coq (6 anni di lavoro).

Teorema di Hales (congettura di Keplero)

Il miglior modo di impilare delle sfere nello spazio in modo che occupino il minimo spazio è seguendo un reticolo esagonale.



Storia della congettura

- 1611, Keplero formula la congettura
- 1831, Gauss dà una prima soluzione valida per un reticolo “regolare”
- 1900, Hilbert include la congettura nella sua famosa lista di 23 problemi.
- 1953, Fejes Tóth riduce la dimostrazione ad un numero finito (ma molto grande) di sottoproblemi.
- 1998, Hales annuncia una dimostrazione basata pesantemente sull’uso del computer.
- 2003, Annals of Mathematics pubblica la parte “standard” della dimostrazione. Hales inizia il il progetto FlysPecK.
- 2014, Hales e i suoi collaboratori annunciano la dimostrazione formale.

Metodi formali

Obiettivo: assicurare il corretto funzionamento di un dispositivo critico. (Hardware, software, protocolli, …)

Procedimento (*'certificazione'*):

- Scrivere formalmente la specifica del sistema.

- Descriverne formalmente l'implementazione.
  (Algoritmo, diagramma, grafo, …)

- Dimostrare il teorema:
  "l'implementazione soddisfa la specifica".

Il teorema meccanizzato si chiama ***certificato***.

Esempi di grandi gruppi e progetti di certificazione

- NASA Formal Methods Research Program
- Verifica del calcolo in virgola mobile presso INTEL
- L4, un microkernel certificato
- CompCert, un compilatore C certificato
- Envisage, certificazione di software Java

Bugs: Alcuni esempi notevoli

- 1985: La macchina per radioterapia Therac-25 è responsabile della morte di almeno 6 pazienti.

- 1994: L'istruzione FDIV di uno dei primi modelli di Pentium non funziona correttamente all'Intel un danno economico di 475 milioni di dollari.

- 2003: Blackout del Nord America. Il più grande di tutti i tempi, convogli 55 milioni di persone.

- 2012: La finanziaria Knight Capital perde 440 milioni di dollari a causa di un software finanziario.

- 2013: Un bug nella centralina delle vetture Toyota è responsabile di almeno 89 morti.

Impatto economico dei bug

- Alcuni studi sul costo economico dei bug:
  - [NIST 2002] *"I **danni** dovuti ai bug dei computer costano circa 60 miliardi di dollari ogni anno"*.
  - [Università di Cambridge 2013] *"Il **debug** del software costa 312 miliardi di dollari ogni anno".*

È affidabile?

- Perché una dimostrazione al computer dovrebbe essere più affidabile di una tradizionale?
- La logica del dimostratore è consistente?
- L'implementazione è corretta?
- Ci sono dei *bug*?

Architettura dei dimostratori

Approccio di De Brujin:

La correttezza del dimostratore deve dipendere da un *kernel* particolarmente semplice e piccolo.

Il kernel deve essere molto più semplice dell'intero sistema.

Un esempio notevole: HOL Light

HOL Light ha un kernel particolarmente semplice:

- 10 regole di inferenza
- 3 assiomi
- 1 principio di definizione
- 430 righe di codice

$$\frac{a}{\vdash a = a}$$

$$\frac{\Gamma \vdash a = b;\ \Delta \vdash b' = c}{\Gamma \cup \Delta \vdash a = c}$$

$$\frac{\Gamma \vdash f = g;\ \Delta \vdash a = b}{\Gamma \cup \Delta \vdash f\,a = g\,b}$$

$$\frac{x;\ \Gamma \vdash a = b}{\Gamma \vdash \lambda x.\ a\ = \lambda x.\ b}\ \text{(if } x \text{ is not free in } \Gamma\text{)}$$

$$\frac{(\lambda x.\ a)\,x}{\vdash (\lambda x.\ a)\,x = a}$$

$$\frac{p{:}bool}{p \vdash p}$$

$$\frac{\Gamma \vdash p;\ \Delta \vdash p' = q}{\Gamma \cup \Delta \vdash q}$$

$$\frac{\Gamma \vdash p;\ \Delta \vdash q}{(\Gamma \setminus q) \cup (\Delta \setminus p) \vdash p = q}$$

Certificare il certificatore

Idea: costruire un dimostratore *'autocertificato'*.

- **Problema:** il teorema di Gödel indica che questo è impossibile (almeno che il dimostratore non sia inconsistente).
- **Soluzione:** fornire una significativa *'approssimazione'*:
  È stato costruito un modello di HOL Light in se stesso e sono stati meccanizzati i teoremi:

  - $\mathrm{HOL} \vdash \mathit{Con}(\mathrm{HOL} - \{\infty\})$
  - $\mathrm{HOL} + \mathrm{I} \vdash \mathit{Con}(\mathrm{HOL})$

Ostacoli all'adozione della meccanizzazione

- Esistono numerosi dimostratori tra loro incompatibili: ACL2, Agda, Coq, HOL Light, HOL4, Idris, Isabelle, Mizar, NuprI, ProofPower, PVS.

- Esistono numerosi fondamenti della matematica tra cui scegliere:
  - logica classica o intuizionistica?
  - logica del primo ordine o di ordine superiore?
  - con tipi o senza?
  - con tipi semplici o dipendenti?
  - Curry-Howard o no?
  - Univalenza?
  - …

- Il rapporto tra un testo matematico meccanizzato e uno convenzionale (fattore di De Brujin) è tipicamente 4. Ma può variare enormemente.

Conclusioni

- La meccanizzazione della matematica è possibile.
- Ci sono importanti applicazioni pratiche e teoriche.
- Esistono molti dimostratori e molti paradigmi dai quali ancora non emerge un consenso generale.
- La meccanizzazione ha ancora oggi un alto costo economico.