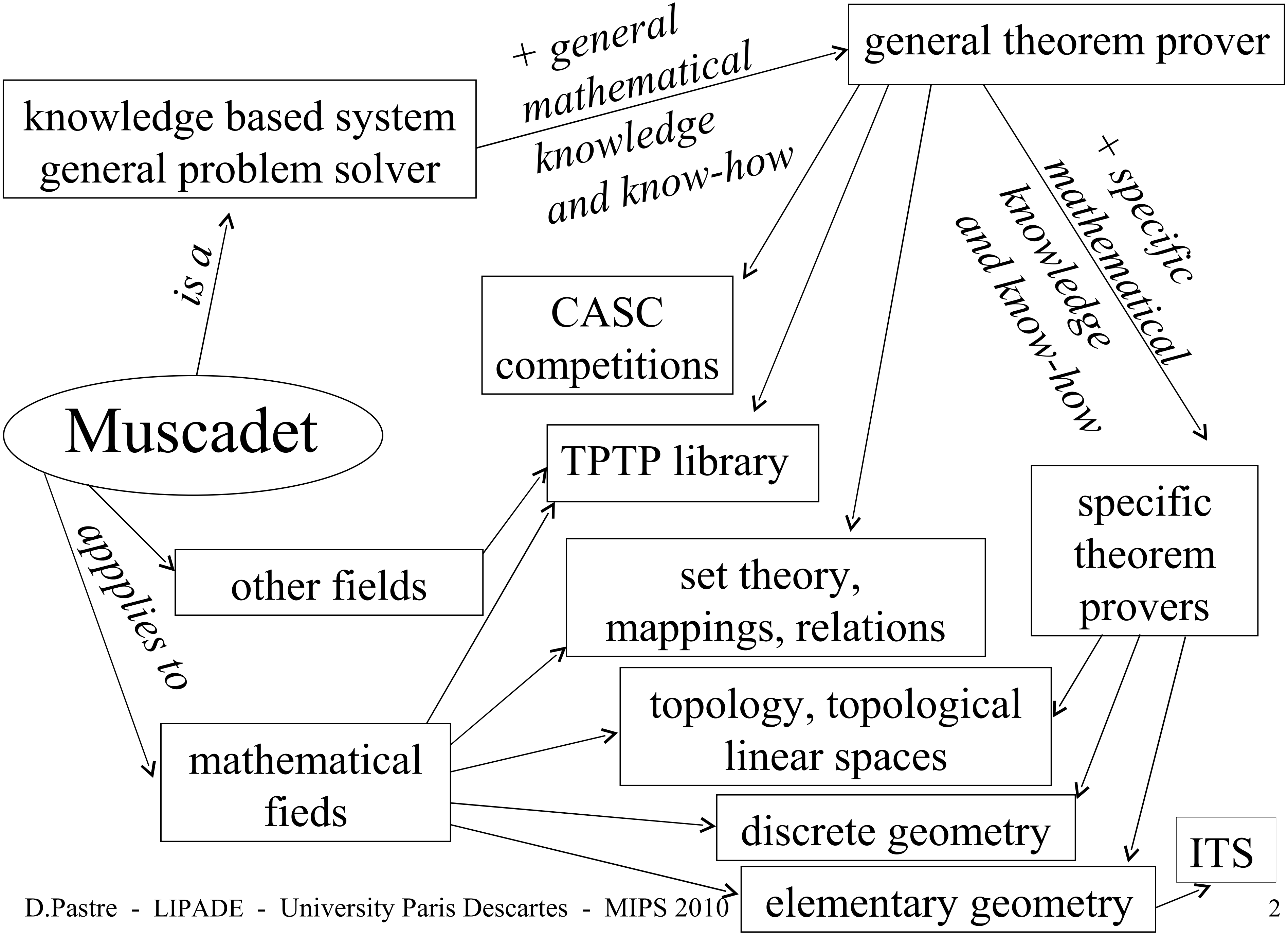
Natural Proof Search and Proof Writing

Dominique Pastre
University Paris Descartes

Workshop on Mathematically Intelligent Proof Search
10 july 2010


general theorem prover
+ general mathematical knowledge and know-how
knowledge based system
general problem solver
is a
CASC competitions
+ specific mathematical knowledge and know-how
Muscadet
TPTP library
applies to
specific theorem provers
other fields
set theory, mappings, relations
topology, topological linear spaces
mathematical fieds
discrete geometry
ITS
elementary geometry

TPTP and CASC

Thousands of Problems for Theorem Provers

(from 1993, Geoff Sutcliffe and Christian Suttner)

~ 7000 FOF problems (first order)   6800 CNF problems (clauses)

in logic, mathematics, computer science, science and engeneering, social sciences, ...

History

	version	FOF		CNF
1993	1.0.0			2295
1997	2.0.0	217 (dont 5 SET)		3060
1999	2.2.0	670 (... 308 ...)		3334
2010	4.0.1	6983	1374	6800

The CADE ATP System Competition are held at each CADE conference (organized by Geoff Sutcliffe)

About 20 "sound, fully automatic, classical logic order ATP systems" each year attend CASC competitions

About 50 systems are regularly tested on TPTP problems

Vampire, the best system is based on the resolution principle

Results of TPTP and CASC show

- the superiority of resolution based provers (Vampire, E, iProver), accordingly to the number of problems solved,
- but also the complementarity of resolution based provers and some other provers (Zenon, Muscadet, Infinox), which may prove theorems which no other prover is able to prove)
- Muscadet had in 2007 and 2008 the highest SOTA (a new ranking measure created in 2007 in CASC competitions, which measure the systems' ability to solve problems that few other systems can prove)

A knowledge-based system

Facts

- hypotheses
- conclusion to be proved
- objects
- subtheorems
- definitions, axiomes, lemma
- ...
- all sort of facts which give relevant information during the proof searching progress

Rules

- logic and mathematics
- built from definitions and axioms
- dynamically built from hypotheses

Metarules

Inference rules

Rule "∀" : **to prove** $\forall x\ P(x)$

(i.e. **if** the conclusion of the theorem being proved is $\forall x\ P(x)$)

take any x1

(i.e. **create** an objet x1)

et **prove** P(x1)

(i.e. **replace** the conclusion to be proved by P(x1))

Rule "$\Rightarrow$" : **to** prove $A \Rightarrow B$, assume A and prove B

("assume A" consists to add A as a new hypothesis,
by splitting it if it is a conjunction,
and by doing some specific treatments in some other cases)

Rule "$\wedge$" : **to** prove $A_1 \wedge A_2 \wedge ... \wedge A_n$
prove all the A_i one after the other

Rule "stop" : **if** a new hypothesis has been added,
which is the conclusion to be proved
then the theorem is proved

Rule "stop_$\vee$" : **if** the conclusion is a disjunction $A_1 \vee A_2 \vee ... \vee A_n$
and **if** one of the A_i has been added as a new hypothesis
then the theorem is proved

Rule "hyp_$\vee$" : **if** $A \vee B$ is a hypothesis among others
and **if** C is to be proved
then prove $(A \Rightarrow C) \wedge (B \Rightarrow C)$

Rule "hyp_$\exists$" : **if** $\exists x\ P(x)$ is a hypothesis
and **if** there is still no hypothesis of the form P(y)
then create x1 and assume P(x1)

Rule "concl_$\wedge$" : **to prove** $\exists x\ P(x)$,
search for x such that P(x)

More precisely :

To prove $\exists x\ (C_1(x) \wedge C_2(x) \wedge \ldots \wedge C_n(x))$

search for an object y such that, with present hypotheses, for all i between 1 and n, $C_i(y)$ was verified (easy case) or proved (by a recursive call to the prover)

Rule "def_concl_1" : **if** P(X) is the conclusion to be proved
and **if** a definition of predicate P is known
then replace P(X) by this definition

Rule "def_concl_2" : **if** A:F(B) is a hypothesis
where F is a functionnal symbol
which is defined as F(B) = {Y | P(Y)}
or y *R* F(B) ⇔ P(Y)
and **if** X *R* A has to be proved
then replace the conclusion X *R* A by P(X)

the quantifier !
"for the only ... such that ..."

Rule "elim_func" : **if** the expression P(F(A)) occurs
where F is a functional symbol
then replace it by !B:f(A), P(B)

where !B:f(A), P(B) means for the only B equal to f(A), p(B) is true

!B:f(A), P(B) is equivalent to $\forall B[f(A):B \Rightarrow p(B)]$
and to $\exists B\,[f(A):B \wedge P(B)]$

The first expression is better for conclusions (positive position),
Rule "concl_!" : to prove !B:f(A), P(B),
create B1, **add** the hypothesis B1:f(A) and **prove** P(B1)

The second one is better for hypotheses (negative position), no such hypothesis is added, at the place we have the *super-action*

To add !B:f(A), P(B) **create** an objet B1and **add** the hypothesis P(B1)

Super-actions

Super-actions are defined as packs of rules, they may be recursive.

Example "**add a hypothesis**"

To add-hyp H

if H is already a hypothesis or if H is of the form X=X
then do nothing

if H is of the form A∧B alors add-hyp A ad add-hyp B

if H is of the form ∀X P or A⇒B
then create rules locale to this (sub)theorem

if H is of the form *for the only* Y *such that* Y:F(X), P(X))
and if there is not already a hypothesis of the form Y:F(X)
then crete a new object Y1 add add-hyp Y1:F(X)
else add H as a new hypothesis

...

Rules relating to concepts defined by the user

The *predicate* P gives rules of the form :

Rule "Pi" : **if** P(...) is a hypothesis

alors ...

This is automatically done by metarules

<u>example</u> :

formal definition :

$$A \subset B \Leftrightarrow \forall x \, (x \in A \Rightarrow x \in B)$$

rule :

Rule "$\subset$" : **if** $A \subset B$ and $x \in A$ are hypotheses

alors add the hypothesis $x \in B$

Le *functional symbol* F gives rules of the form :

Rule "Fi" : **if** Y:F(...) and $X \in Y$ are hypotheses

then ...

<u>example</u> :

formal definition :

$$\mathcal{P}(A) = \{ X \mid X \subset A \}$$

rule :

Rule "$\boldsymbol{\mathcal{P}}$" : **if** B:$\mathcal{P}$(A) and $x \in B$ are hypotheses

then add the hypothesis $x \subset A$

other example

formal definition : $A \cap B = \{x \mid x \in A \wedge x \in B\}$

rules :

Rule "$\cap$11" : if C:$A \cap B$ and $x \in C$ are hypotheses
then add the hypothesis $x \in A$

Rule "$\cap$12" : if C:$A \cap B$ and $x \in C$ are hypotheses
then add the hypothesis $x \in B$

Rule "$\cap$2" : if C:$A \cap B$, $x \in A$ and $x \in B$ are hypotheses
then add the hypothesis $x \in C$

Remark : la rule "$\cap$2" is not of the form

if $x \in A$ and $x \in B$ are hypotheses
alors add the hypothesis $x \in A \cap B$

which would be expansive

Power set of the intersection of two sets

Theorem to be proved $\forall A \forall B(\mathcal{P}(A \cap B) =_{set} \mathcal{P}(A) \cap \mathcal{P}(B))$

Definition of intersection

$$A \cap B = \{X \mid X \in A \land X \in B\}$$

Definition of power set

$$\mathcal{P}(A) = \{ X \mid X \subset A \}$$

Definition of set equality

$$A =_{set} B \Leftrightarrow A \subset B \land B \subset A$$

Definition of inclusion

$$A \subset B \Leftrightarrow \forall X (X \in A \Rightarrow X \in B)$$

rules	objects	hypotheses	conclusion
			$\forall A \forall B(\mathcal{P}(A \cap B) =_{set} \mathcal{P}(A) \cap \mathcal{P}(B))$
$\forall$	a, b		$\mathcal{P}(a \cap b) =_{set} \mathcal{P}(a) \cap \mathcal{P}(b)$
elim_func	c, pc	$c{:}a \cap b$, $pc{:}\mathcal{P}(c)$	
and	pa, pb	$pa{:}\mathcal{P}(a)$, $pb{:}\mathcal{P}(b)$	
concl_!	pd	$pd{:}pa \cap pb$	$pc =_{set} pd$
def_concl1			$pc \subset pd \wedge pd \subset pc$
$\wedge$	gives Theorem 1 and Theorem 2		
Theorem 1			$pc \subset pd$
def_concl1			$\forall X (X \in pc \Rightarrow X \in pd)$
$\forall$	x		$x \in pc \Rightarrow x \in pd$
$\Rightarrow$		$x \in pc$	$x \in pd$
$\mathcal{P}$		$x \subset c$	
def_concl2			$x \in pa \wedge x \in pb$
$\wedge$	gives Theorem 11 and Theorem 12		

Theorem 11 rule	objects	hypotheses	conclusion
...	...		$x \in pa$
defconcl2			$x \subset a$
defconcl1			$\forall X\ (X \in x \Rightarrow X \in a)$
$\forall$ and $\Rightarrow$	t	$t \in x$	$t \in a$
$\subset$		$t \in c$	
$\cap 11$		$t \in a$	Theorem 11 proved
Theorem 12			
	...	...	$x \in pb$
			...
			Theorem 12 proved
up			Theorem 1 proved

Theorem 2

rule	objects	hypotheses	conclusion
...	...		$pd \subset pc$
defconcl1			$\forall X\ (X \in pd \Rightarrow X \in pc)$
$\forall$ and $\Rightarrow$	x	$x \in pd$	$x \in pc$
$\cap$1 and 2		$x \in pa,\ x \in pb$	
$\mathcal{P}$(twice)		$x \subset a,\ x \subset b$	
defconcl1			$x \subset c$
(twice)			$\forall X\ (X \in x \Rightarrow X \in c)$
$\forall$ and $\Rightarrow$	t	$t \in x$	$t \in c$
$\subset$(twice)		$t \in a,\ t \in b$	
$\cap$2		$t \in c$	
stop			Theorem 2 proved
up			Theorem 0 proved

Details for elim_funct and concl_!

objects	hypotheses	conclusion
		$\mathcal{P}(a \cap b) =_{set} \mathcal{P}(a) \cap \mathcal{P}(b)$
		$!C{:}a \cap b \ !Pa{:}\mathcal{P}(a) \ !Pb{:}\mathcal{P}(b) \ !PC{:}\mathcal{P}(C) \ !Pab{:}Pa \cap Pb \ ! \ PC =_{set} PP$
c	$c{:}a \cap b$	$!Pa{:}\mathcal{P}(a) \ !Pb{:}\mathcal{P}(b) \ !PC{:}\mathcal{P}(c) \ !Pab{:}Pa \cap Pb \ ! \ PC =_{set} PP$
pa	$pa{:}\mathcal{P}(a)$	$!Pb{:}\mathcal{P}(b) \ !PC{:}\mathcal{P}(c) \ !Pab{:}pa \cap Pb \ ! \ PC =_{set} PP$
pb	$pb{:}\mathcal{P}(b)$	$!PC{:}\mathcal{P}(c) \ !Pab{:}pa \cap pb \ ! \ PC =_{set} PP$
pc	$pc{:}\mathcal{P}(c)$	$!Pab{:}pa \cap pb \ ! \ pc =_{set} PP$
pd	$pd{:}pa \cap pb$	$pc =_{set} pd$

* * * theorem to be proved
![A, B]:equal_set(power_set(intersection(A, B)), intersection(power_set(A), power_set(B)))

* * * * * * theoreme 0 * * * * * *
*** newconcl(0, ..., 1)
explanation : initial theorem -- action ini
create object(s) z2 z1
*** newconcl(0, equal_set(power_set(intersection(z1, z2)), intersection(power_set(z1), power_set(z2))), 2)
*** because concl((0, ..., 1)
*** explanation : the universal variable(s) of the conclusion is(are) instantiated
-- rule !
*** newconcl(0, seul(intersection(z1, z2)::A, seul(power_set(A)::D, seul(power_set(z1)::B, seul(power_set(z2)::C, seul(intersection(B, C)::E, equal_set(D, E)))))), 3)
*** because concl(0, ..., 2)
*** explanation : elimination of the functional symbols of the conclusion
for example, p(f(X)) is replaced by only(f(X)::Y, p(Y))
-- elifun
*** addhyp(0, intersection(z1, z2)::z3, 4), newconcl(0, ...), 4)
*** because concl(0, ..., 3)
*** explanation : creation of object z3 and of its definition
-- rule concl_only

....

.... newconcl(0, equal_set(z4, z7), 8)
*** explanation : creation of object z7 and of its definition

-- rule concl_only

*** newconcl(0, subset(z4, z7)&subset(z7, z4), 9)
*** because concl(0, equal_set(z4, z7), 8)
*** explanation : the conclusion equal_set(z4, z7) is replaced by its definition(fof equal_set)
--- rule def_concl_pred

* * * * * * creation * * * * * * sub-theoreme 0-1 * * * * *
all the hypotheses of (sub)theorem 0 are hypotheses of subtheorem 0-1
*** newconcl(0-1, subset(z4, z7), 10)
*** because concl(0, subset(z4, z7)&subset(z7, z4), 9)
*** explanation : to prove a conjunction, prove all the elements of the conjunction
--- action proconj
*** newconcl(0-1, ![A]: (member(A, z4)=>member(A, z7)), 11)
*** because concl(0-1, subset(z4, z7), 10)
*** explanation : the conclusion subset(z4, z7) is replaced by its definition(fof subset)
--- rule def_concl_pred
create object(s) z8
*** newconcl(0-1, member(z8, z4)=>member(z8, z7), 12)
*** because concl((0, ![A]: (member(A, z4)=>member(A, z7))), 11)
*** explanation : the universal variable(s) of the conclusion is(are) instantiated
--- rule !
*** addhyp(0-1, member(z8, z4), 13)
*** newconcl(0-1, member(z8, z7), 13)
*** because concl(0-1, member(z8, z4)=>member(z8, z7), 12)
*** explanation : to prove H=>C, assume H and prove C
--- rule =>

*** addhyp(0-1, subset(z8, z3), 14)
*** because hyp(0-1, power_set(z3)::z4, 5), hyp(0-1, member(z8, z4), 13), obj_ct(0-1, z8)
*** explanation : rule if (hyp(A, power_set(D)::B, _), hyp(A, member(C, B), _), obj_ct(A, C))then addhyp(A, subset(C, D), _)
built from the definition of power_set (fof power_set)
-- rule power_set
*** newconcl(0-1, member(z8, z5)&member(z8, z6), 15)
*** because concl(0-1, member(z8, z7), 13), hyp(0-1, intersection(z5, z6)::z7, 8)
*** explanation : definition intersection
-- rule defconcl2

* * * * * * creation * * * * * * sub-theoreme 0-1-1 * * * * *

....

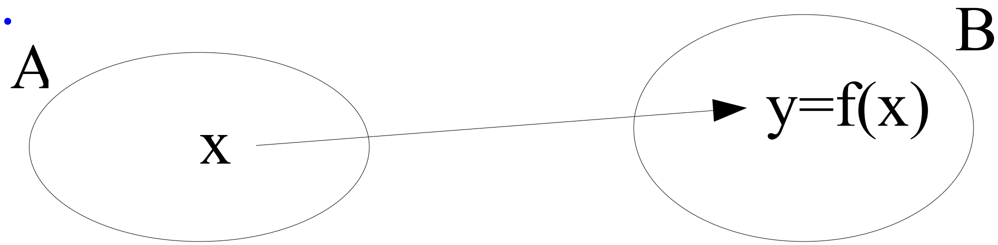
*** newconcl(0-1-1, true, 23)
*** because hyp(0-1-1, member(z9, z1), 22), concl(0-1-1, member(z9, z1), 20)
*** explanation : the conclusion member(z9, z1) to be proved is a hypothesis
-- rule stop_hyp_concl

....

Processing of the existential hypotheses

Systematically creating objects could be expansive. So, the processing of existential hypotheses has a low priority and these hypotheses are handled one after the other, in the order when they appeared, and all the other rules are tried again before processing the next one.

Example : If f maps A into B, then each element of A has an image in B.

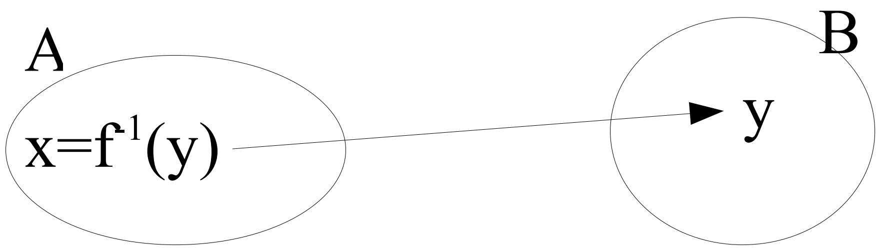


Special case, if f maps A into A :

$$a \rightarrow a_1=f(a) \rightarrow a_2=f(a_1) \rightarrow a_3=f(a_2) \rightarrow \ldots$$

All that can be deduced from the l'hypothèse $a_i=f(a_{i-1})$ is deduced before the creation of a_{i+1}.

If moreover f is surjective, each element of B has an antecedent in A.



Special case, if f maps A onto A :

$$\ldots \to a_4=f^{-1}(a_2) \to a_2=f^{-1}(a) \to a \to a_1=f(a) \to a_3=f(a_1) \to \ldots$$

an image and an antecedent are created alternately.

Moreover, if there are several mappings, images and antecedents are created alternately for all mappings.

Reordering rules

The rules which may create more specific objects must have higher priority than others

Metarule : if the rule R may create an element a such that P
the rule R' may create an element b such that Q
P is more general than Q

then R' must be applied before R

More precisely, the metarule is the following (of which it is a restriction) :

if the rule R contains the action add-hyp $\exists x \in A\ C$
the rule R' contains the action add-hyp $\exists x' \in A\ C'$
C' is a conjunction of terms and one of them is equal to C modulo x and x'

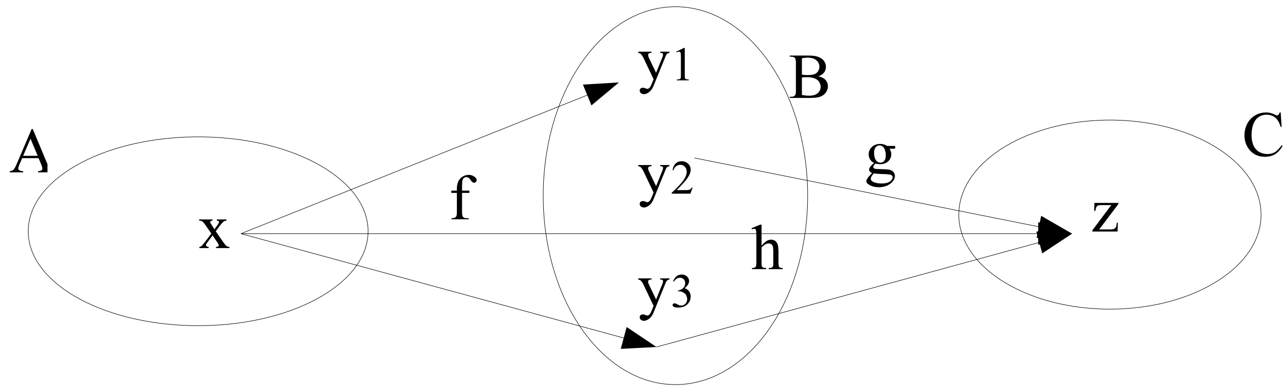
then apply R' before R

Example

If f maps A into B, then each element in A has an image in B.

If f maps A onto A dans B, then each element in B has an pre-image in A.

If h is the composition (from A into C) of f, mapping A into B, and of g, mapping B into C, and if z=h(x), then there is an element y in B such that y=f(x) and z=g(y)



Then $y_1=y_3$ and, if g is injective, $y_2=y_3$.
Rather than creating y_1, then y_2 and y_3, it is better to only create y_3 which verifies the three properties.

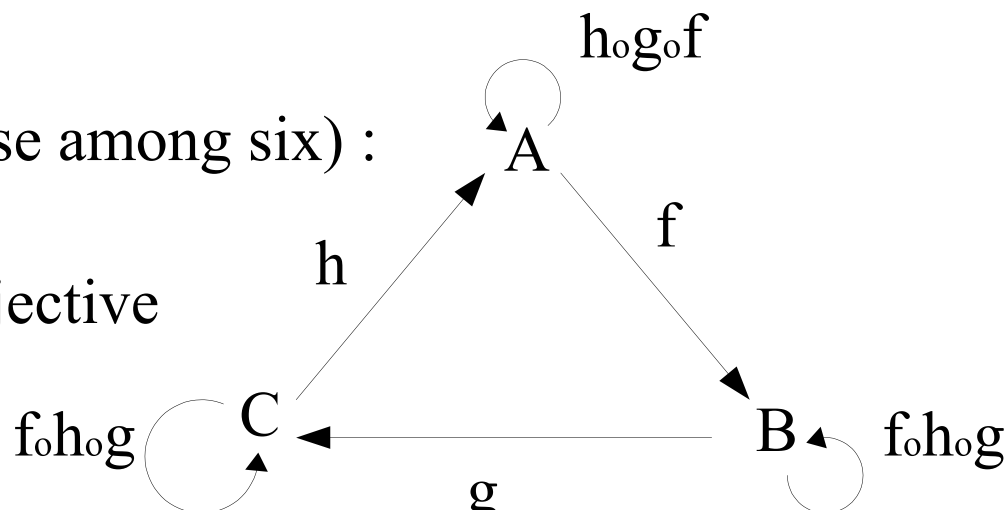
Example in set theory

Theorem : Consider three mappings f, g, h from A into B, B into C, C into A; if among the three mappings $h \circ g \circ f$, $g \circ f \circ h$, $f \circ h \circ g$, two are injective (resp. surjective) and the third is surjective (resp. injective), then f, g and h are one-to-one.

For example (one case among six) :

$h \circ g \circ f$ injective

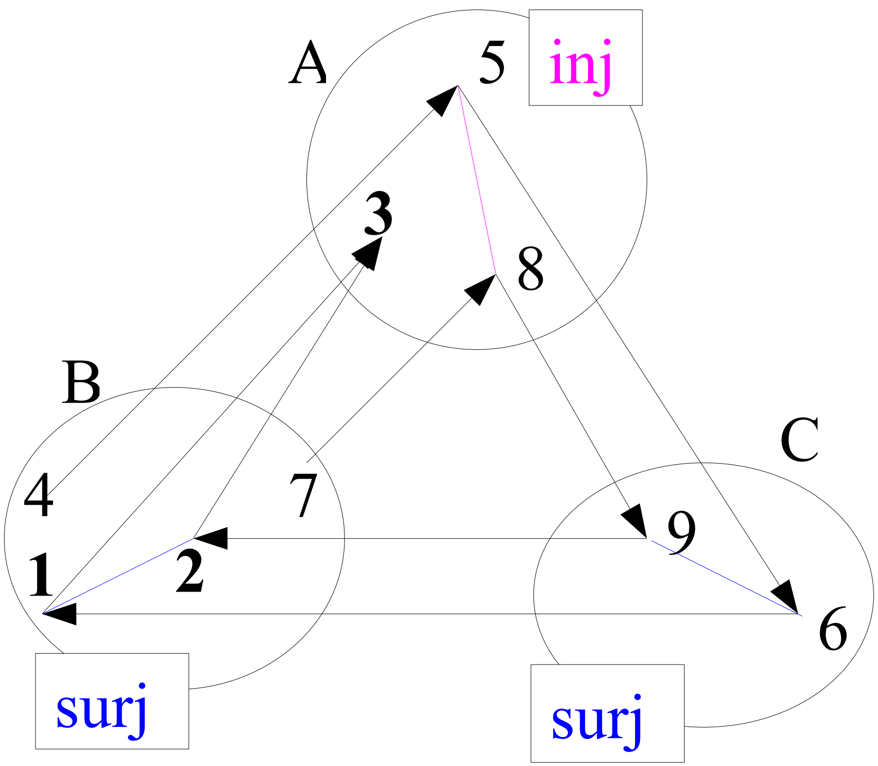
$g \circ f \circ h$ and $f \circ h \circ g$ surjective



Case $h \circ g \circ f$ injective, $g \circ f \circ h$ and $f \circ h \circ g$ surjective (one case among six)

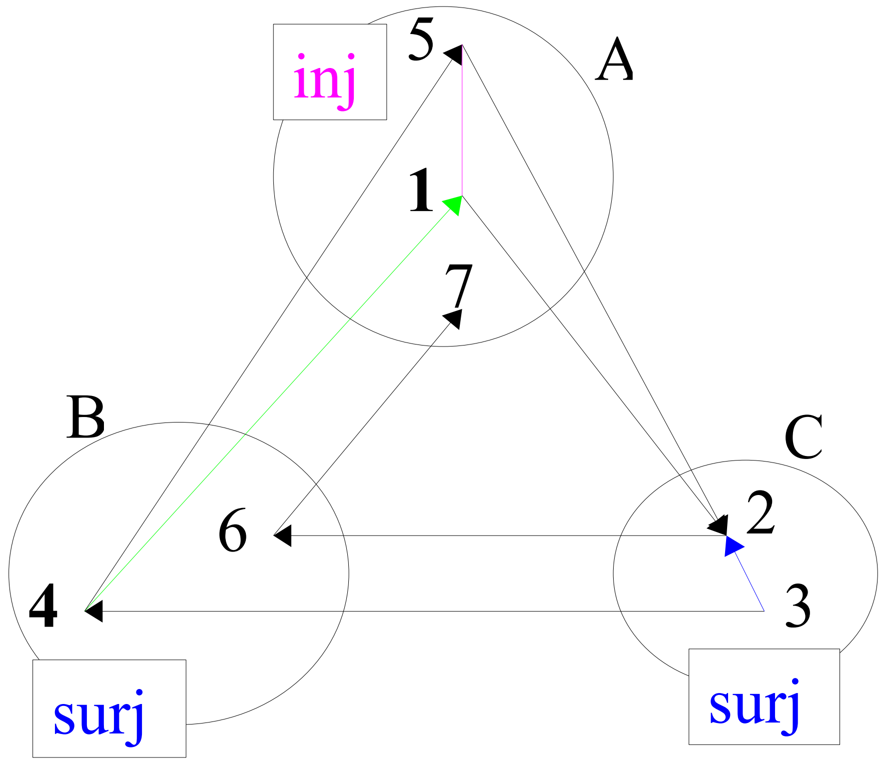
<u>h injective</u>

if 1 and 2 have the same image 3, then they are equal



<u>h surjective</u>

4 is a pre-image de 1 because 1 is equal to its image 5



Proof of theorem $\neg \exists X \forall Y (Y \in X \Leftrightarrow Y \notin Y)$
($X = \{ Y \mid Y \notin Y \}$ is not a set)

by the resolution principle : clauses

$$\left| \begin{array}{l} Y \notin a \vee Y \notin Y \\ Y \in Y \vee Y \in a \end{array} \right. \Big\rangle\ \square$$

by Muscadet :

concl : $\neg \exists X \forall Y (Y \in X \Leftrightarrow Y \notin Y)$

hyp : $\exists X \forall Y (Y \in X \Leftrightarrow Y \notin Y)$

concl : false

object : a

local rules :

- r0 : if $Y \in a$ and $Y \in Y$ then false
- r1 : if $Y \notin Y$ then $Y \in a$
- r2 : for all object Y, $Y \in Y \vee Y \in a$

hyp : $a \in a \vee a \in a$ (rule r2)

$a \in a$ (rule "$\vee$")

false (rule r0) theorem proved (by contradiction)

* * * theorem to be proved
~ ?[B]:![A]: (element(A, B)<=> ~element(A, A))

* * * proof :

* * * * * * theoreme 0 * * * * * *
*** newconcl(0, ~ ?[B]:![A]: (element(A, B)<=> ~element(A, A)), 1)
*** explanation : initial theorem
-- action ini
*** addhyp(0, ?[B]:![A]: (element(A, B)<=> ~element(A, A)), 2), newconcl(0, false, 2)
*** because concl(0, ~ ?[B]:![A]: (element(A, B)<=> ~element(A, A)), 1)
*** explanation : assume ?[B]:![A]: (element(A, B)<=> ~element(A, A)) and search for a contradiction
-- rule concl_not
create object(s) z1
*** addhyp(0, ![A]: (element(A, z1)<=> ~element(A, A)), 3)
*** because hyp(0, ?[B]:![A]: (element(A, B)<=> ~element(A, A)), 2)
*** explanation : treatment of the existential hypothesis
-- rule hyp_exi
*** addhyp(0, element(z1, z1)|element(z1, z1), 4)
*** because obj_ct(0, z1)

*** explanation : the rule r_hyp__3__2or : if obj_ct(A, B) then
addhyp(A, element(B, B)|element(B, z1), _)
is a local rule built from the universal hypothesis
![A]: (element(A, z1)<=> ~element(A, A))
-- rule r_hyp__3__2or
*** addhyp(0, element(z1, z1), 5)
*** because hyp(0, element(z1, z1)|element(z1, z1), 4)
*** explanation : E|E = E
-- rule hyp_or1
*** addhyp(0, false, 6)
*** because hyp(0, element(z1, z1), 5), hyp(0, element(z1, z1), 5), obj_ct(0,z1)
*** explanation : the rule r_hyp__3__ : if (hyp(A, element(B, z1), _),
hyp(A, element(B, B), _), obj_ct(A, B))then addhyp(A, false, _)
is a local rule built from the universal hypothesis
![A]: (element(A, z1)<=> ~element(A, A))
-- rule r_hyp__3__
*** newconcl(0, true, 7)
*** because hyp(0, false, 6), concl(0, false, 2)
*** explanation : the conclusion false to be proved is a hypothesis
-- rule stop_hyp_concl
then the initial theorem is proved
* *

pseudo second order

mathematical definition :

$$\forall R \,(\, \text{transitive}(R) \Leftrightarrow \forall X \, \forall Y \, \forall Z \, (R(X,Y) \wedge R(Y,Z)\,) \Rightarrow R(X,Z)$$

Muscadet definitions :

$$\forall R \,(\, \text{transitive}(R) \Leftrightarrow \forall X \, \forall Y \, \forall Z \, (..[R,X,Y] \wedge ..[R,Y,Z] \;\Rightarrow\; ..[R,X,Z]\,)\,)$$

$$\forall X \, \forall Y \,(\, ..[\text{subset},X,Y] \Leftrightarrow \text{subset}(X,Y)\,)$$

theorem to be proved : **transitive(subset)** mathematical definition :

$\forall R\ (\ \text{transitive}(R,E) \Leftrightarrow$

$\forall X\ \forall Y\ \forall Z\ (X \in E \wedge Y \in E \wedge Z \in E \wedge R(X,Y) \wedge R(Y,Z)\ ) \Rightarrow R(X,Z)\ )$

Muscadet definitions :

$\forall R\ (\ \text{transitive}(R,E)$

$\Leftrightarrow \forall X\ \forall Y\ \forall Z\ (X \in E \wedge Y \in E \wedge Z \in E \wedge ..[R,X,Y] \wedge ..[R,Y,Z] \Rightarrow ..[R,X,Z])\ )$

$\forall X\ \forall Y\ (\ ..[\text{subset},X,Y] \Leftrightarrow \text{subset}(X,Y)\ )$

theorem to be proved : **transitive(subset, $\mathcal{P}$(E))**