Complementarity of a Natural Deduction Knowledge-Based Prover and Resolution-Based Provers in Automated Theorem Proving

Dominique Pastre

Crip5 - Université René Descartes - Paris

Abstract

MUSCADET is a knowledge-based theorem prover based on natural deduction. The results obtained during the CASC competitions of theorem provers show its complementarity with regard to resolution-based provers. This paper presents some MUSCADET proofs of theorems proposed at the last two competitions (2005 and 2006) and points out some of the characteristics which may account for its successes.

Key words: automated theorem proving, natural deduction, knowledge-based system

1 Introduction

Bledsoe (1971) attempted to speed up automated theorem proving by using natural-deduction-like heuristics. His first prover (named "PROVER") applied these natural-deduction-like heuristic techniques before sending the (sub)theorems being proved to a resolution-based program. The proofs were shorter and obtained more quickly and more frequently than by resolution alone. Often it was not even necessary to call on resolution. Then, in (Bledsoe, 1972, 1974) and after, resolution was no longer used at all.

Since then, many strategies have been used to improve resolution-based theorem provers. They have become powerful and few provers have been based on natural deduction.

In 2005, we observed that at least one theorem could be proved by MUSCADET [1] (Pastre, 1989, 1993, 2001b) a theorem prover based on natural de-

[1] available from http://www.math-info.univ-paris5.fr/∼pastre/muscadet

duction, whereas it could not be proved by any of the participating resolution-based theorem provers, within the time limit imposed by the CASC competitions (Pelletier, Sutcliffe, Suttner, 2002; Sutcliffe, Suttner, 2006). In 2006, this occurred for three new problems.

This does not mean that MUSCADET is better than other provers. Several theorems were proved by all the entrant systems except MUSCADET. This only tends to show that MUSCADET may be complementary to other provers. MUSCADET still needs to be improved. The improvements will be obtained by new good heuristics and know-how, not by greater computer speed. The analysis of the results of the competitions also shows that MUSCADET is faster than other provers on the problems it can solve. In case of success, the proof is obtained at least as quickly [2] as with other provers, and much more quickly in many cases. If the system fails to obtain the proof it often quickly stops itself. The cases of "timeout" are generally due to infinite creations of objects, or to too many sub-theorems.

For a long time, many theoretical results regarding resolution were published but the new strategies were not very satisfying from a practical point of view. Moreover, theorems were generally given to provers as sets of clauses instead of first-order formulas. There may be one reason for this : although translating sets of first-order formulas into sets of clauses is easily automatizable, writing a *good* set of clauses was not so easy. There may be several possibilities and some of them may fit for the resolution based provers better. This is why it was better to do the transformation by hand.

In the TPTP Library (Sutcliffe, Suttner, 1998), created in 1993, all problems were expressed as sets of clauses up to 1997. Nowadays, more than two thirds of the problems are still given as sets of clauses [3] . In the CASC competitions (Sutcliffe, Suttner, 2006), there are five divisions. For four of them, comprising nine categories, problems are given as sets of clauses. In only one division, divided into two categories, problems are given as sets of first order formulas. The consequence is that resolution-based provers may compete in all the divisions, or be specialized in one or another, while provers that do not work with clauses can compete only in the FOF (First Order Formula) division, which is the most general division of them all [4] .

Moreover, the library keeps growing with the contributions of researchers. As more researchers work with resolution-based provers, more new TPTP problems are better adapted to resolution-based provers than to natural deduction ones, even if they are expressed as first order formulas.

[2] with only one exception (example of section 4.1)
[3] Although TPTP incorporated more new FOF (First Order Formulas) problems than CNF (Clause Normal Form) problems in 2004 and 2005 (but not in 2006)
[4] and has been promoted to the primary place in CASC in 2006

2 Main characteristics of MUSCADET

2.1 Facts, rules and metarules

MUSCADET is a knowledge-based system. It works with facts to which it applies rules.

Facts are the conclusion to be proved, the hypotheses, the objects of a theorem or a sub-theorem to be proved, links to concepts that appeared in the initial conjecture or in the definitions of the preceding concepts, sub-theorems, definitions, axioms and lemmas, and all the properties which give relevant information during the proof searching process. At the beginning, there is no hypothesis and the conclusion to be proved is the first-order formula of the initial conjecture.

Rules are written in the form

rule <name> : if <list of conditions> then <list of actions>

By applying rules, hypotheses are added, objects may be created, the conclusion may be replaced by another one, the theorem to be proved may be split into one or more, independent or not, sub-theorems to be proved.

Some rules are general and express logical or mathematical knowledge and usual mathematical know-how. Here are some examples of such rules. [5]

rule $\forall$: if the conclusion is $\forall X P(X)$
then create a new object $X1$ and the new conclusion is $P(X1)$

rule $\rightarrow$: if the conclusion is $H \rightarrow C$
then add the hypothesis H and the new conclusion is C

rule `stop1` : if the conclusion is one of the hypotheses
then the new conclusion is `true`

rule $\wedge$: if the conclusion is a conjunction
then successively prove all the elements of the conjunction

rule `defconcl` : if the predicate of the conclusion has a definition
then replace the conclusion by its definition

[5] Prolog conventions are used : variables start with upper-case letters whereas constants start with low-case letters. Moreover, in this paper, to make it more readable, I also use the same conventions for predicates. This is not allowed in Prolog, hence neither in MUSCADET or in TPTP which are written in Prolog.
In MUSCADET, $P(X)$ is written C and $P(X1)$ is obtained by replacing X by $X1$ in C.
In TPTP the predicate "apply" is used to write apply(P, X) instead of $P(X)$.

Other rules are automatically built by metarules from the definitions, lemmas and universal hypotheses.
For example, from the definition of inclusion

$$\forall A\ \forall B(A \subset B \leftrightarrow\ \forall X(X \in A \rightarrow X \in B)$$

the following rule is built :

rule $\subset$: if $A \subset B$ and $X \in A$ are hypotheses
then add the hypothesis $X \in B$ if it is not yet a hypothesis.

From the definition [6] of intersection

$$\forall A\ \forall B\ \forall X(X \in A \cap B \leftrightarrow X \in A \wedge X \in B)$$

the following rules are built :

rule $\cap 1$: if $A \cap B : C$ and $X \in C$ are hypotheses
then add the hypothesis $X \in A$ if it is not yet a hypothesis
rule $\cap 5$: if $A \cap B : C$ and $X \in C$ are hypotheses
then add the hypothesis $X \in B$ if it is not yet a hypothesis
rule $\cap 3$: if $A \cap B : C$, $X \in A$ and $X \in B$ are hypotheses
then add the hypothesis $X \in C$ if it is not yet a hypothesis

where $A \cap B : C$ expresses that C is the intersection of A and B. This means that $A \cap B$ has already been introduced.

Some actions are elementary, such as replacing the conclusion by its definition by the `defconcl` rule above. A conclusion of the form

$$A \subset B$$

will simply be replaced by

$$\forall X(X \in A \rightarrow X \in B)$$

Other actions are more sophisticated and are defined by packs of rules, such as adding a hypothesis which is defined by the following rules :

to add a hypothesis H :
- if H is already a hypothesis or is of the form $X{=}X$ then do nothing
- if H is a conjunction
then successively add all the elements of the conjunction
- if H is $\forall X P(X)$ then create local rules for this theorem

... [others examples will be given in the next sections]

- in all other cases add H as a new hypothesis.

Note that the hypotheses which are added are only elementary, disjunctive and existential hypotheses. Conjunctive hypotheses are split before being added and universal hypotheses are treated as definitions or lemmas and replaced by rules. Disjunctive and existential hypotheses are first stored as hypotheses

[6] Muscadet also accepts the following notation $A \cap B = \{X \mid X \in A \wedge X \in B\}$ but it cannot be used in the TPTP context.

without being subjected to any particular treatment because their treatment may be useless or even expansive. Therefore it will be done only later if necessary.

2.2 Elimination of functional symbols

The MUSCADET strategies are designed to work with predicates rather than with functional symbols. In a formula, MUSCADET "eliminates" functional symbols by giving names to the terms. Thus they become objects that will replace the terms in the predicative formula. Consequently, there remains no hypothesis or conclusion such as $p(f(a))$ but instead the hypothesis or conclusion $p(b)$ where b is a constant defined by the hypothesis $f(a):b$. The symbol ":" is used to express that b is the object $f(a)$, and the formula $f(a):b$ will be handled as if it were a predicative formula.

These transformations cannot be done directly on the initial statement of the theorem to be proved, since some of the functional expressions will become hypotheses while others will become conclusions. Moreover, those expressions may contain variables that make things more complicated since they also appear in some definitions. So the expressions are first transformed by using a new quantifier noted "!", which means

"for the only ... equal to ..." .

$p(f(a))$ is replaced by

$$!A{:}f(a), p(A)$$

which means

"for the only A equal to $f(a)$ then $p(A)$ holds"

where A is a variable.

This mechanism is recursive. In the example given in the next section, the formula

$$\text{inv}(f, a \cap b) = \text{inv}(f, a) \cap \text{inv}(f, b)$$

is replaced by

$$!A{:}a \cap b, !B{:}\text{inv}(f, A), !C{:}\text{inv}(f, a), !D{:}\text{inv}(f, b), !E : C \cap D, B = E \quad .$$

This work is done by the (recursive) rules of the `elifun` action which is called for the first conclusion to be proved (as well as for the definitions and lemmas) by the `elifun` rule.

Then the expressions

$$!A{:}\text{<term>}, \text{<property>}$$

are handled by rules specific to hypotheses or to the conclusion, or to building rules from definitions.

Here are examples of such rules :

rule ! : if the conclusion is of the form $!Y{:}F(...), P(Y)$
then if there is already a hypothesis $F(...){:}Y1$
then the new conclusion is $P(Y1)$
else create a new object
add the hypothesis $F(...)$:<this new object>
and the new conclusion is P(<this new object>)
to add a hypothesis H :
if H is of the form $!Y{:}F(...), P(Y)$
then if there is already a hypothesis $F(...){:}Y1$
then add the hypothesis $P(Y1)$
else create a new object
and add the hypotheses $F(...)$:<this newobject>
and P(<this new object>)

2.3 Equality handling

Because of elimination of functional symbols, equality may occur in hypotheses only as equality of objects. Each time two objects of a (sub)theorem to be proved are found to be equal, one of them is replaced everywhere in the sub-theorem by the other, then it is removed, and so is the equality.

In the example given in the next section, during the proof searching process of the second sub-theorem, the object t is in $f^{-1}(b1)$ and in $f^{-1}(b2)$, so that t has an image u in $b1$ and an image v in $b2$.
The following hypotheses have been added :
$f(t){:}u$, $f(t){:}v$,
$u \in b1$ hence $u \in b$ since $b1 \subset b$,
$v \in b2$ hence $v \in b$ since $b2 \subset b$.
The uniqueness of the image of t in b implies that $u = v$.
Then v is replaced by u and we have the new hypothesis $u \in b2$.
Now the same object u belongs to $b1$ and to $b2$, hence to $b1 \cap b2$ and it is possible to conclude that t belongs to $f^{-1}(b1 \cap b2)$.

2.4 Negation handling

MUSCADET works with positive properties as much as possible. Rules built from definitions are made to work with positive rather than with negative properties. Not only does MUSCADET not add negative properties, but it provides general rules to eliminate negations in most cases.
If the conclusion is a negation $\neg A$, then a new hypothesis A is added and the

new conclusion is `false`. This is one of the few cases of proofs by contradiction, the (sub-)theorem will be proved if a contradiction is found (hypothesis `false` added) or if a new conclusion appears, for example after removing a negative hypothesis due to the following rule :

if a hypothesis is a negation $\neg A$
and if the conclusion to be proved is `false`
then the hypothesis is removed and the new conclusion is A.

In other cases, negative hypotheses may be rewritten, for example
$\neg\neg A := A$
$\neg(A \rightarrow B) := A \wedge \neg B$ which gives two hypotheses A and $\neg B$
$\neg(A \leftrightarrow B) := (A \wedge \neg B) \vee (\neg A \wedge B)$
There are also some rare built rules which test negative hypotheses (see section 3.2).

There are two instances where MUSCADET is required to handle negations :
- firstly if the statement of the theorem to be proved contains itself negations, for example

$(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$ (TPTP problem SYN046+1),

- secondly if the handled concepts concern negations, for example complements or differences of sets, empty or disjoint sets.

The example given in section 3.2 concerns the difference of sets, the definition of which is

$\forall E\ \forall A(X \in (E - A) \leftrightarrow X \in E \wedge \neg(X \in A))$

We will see the four rules which are built from these definitions. Three of them do not contain any negation. This was obtained by moving formulas in conditions, actions, hypotheses and conclusion parts. The fourth rule contains only a negative hypothesis in the condition part.

2.5 Examples

Some detailed examples of simple proofs of easy theorems may be found in (Pastre, 1993, 2001a,b). Proofs of other theorems may be found in (Pastre, 1993, 2001c, 2002). The two following sections give detailed proofs of more difficult theorems and provide comments.

The examples of section 3 come from CASC-20 (2005) where one theorem was solved only by MUSCADET, and two theorems were proved by MUSCADET and only one other entrant system but in much more time.

TPTP problems SETxxx+4 were proposed by myself but they have been incorpored in the library since 1999. SET601+3 comes from (Trybulec, 1989).

3 Examples of MUSCADET proofs from CASC-20 (2005)

3.1 Handling images and inverse images : a property of inverse images

Here is an example of a theorem which was proved by MUSCADET during CASC-20 (2005), but by none of the other provers.

Theorem SET757+4. *If f maps A into B and X and Y are two subsets of B, then the inverse image by f of the intersection of X and Y is equal to the intersection of the inverse images by f of X and of Y.*

that is, in usual mathematical notation $f^{-1}(X \cap Y) = f^{-1}(X) \cap f^{-1}(Y)$

Its formal statement in first order predicate calculus is

$$\forall F\ \forall A\ \forall B\ \forall X\ \forall Y$$
$$(\mathrm{maps}(F, A, B) \wedge X \subset B \wedge Y \subset B$$
$$\rightarrow \mathrm{inv}(F, X \cap Y, A) =_{\mathrm{set}} \mathrm{inv}(F, X, A) \cap \mathrm{inv}(F, Y, A))$$

with the following definitions [7]

$$\forall F\ \forall A\ \forall B(\mathtt{maps}(F, A, B) \leftrightarrow$$
$$(\forall X(X \in A \rightarrow \exists Y(Y \in B \wedge \mathtt{apply}(F, X, Y)))$$
$$\wedge\ \forall X\ \forall Y1\ \forall Y2(X \in A \wedge Y1 \in B \wedge Y2 \in B \rightarrow$$
$$(\mathtt{apply}(F, X, Y1) \wedge \mathtt{apply}(F, X, Y2) \rightarrow Y1 = Y2))))$$

(every element in the domain A of F has an image in the range B and this image is unique).

$$\forall A\ \forall B(A =_{\mathrm{set}} B \leftrightarrow A \subset B \wedge B \subset A)$$
$$\forall F\ \forall A\ \forall B\ \forall X(X \in \mathrm{inv}(F, B, A) \leftrightarrow X \in A \wedge \exists Y(Y \in B \wedge \mathrm{apply}(F, X, Y))$$

From these definitions the following rules were built :

rule $=_{\mathrm{set0}}$: if $A =_{\mathrm{set}} B$ is a hypothesis
then add the hypothesis $A \subset B$ [8]

rule $=_{\mathrm{set1}}$: if $A =_{\mathrm{set}} B$ is a hypothesis
then add the hypothesis $B \subset A$

rule `maps1`: if $\mathrm{map}(F, A, B)$, $X \in A$ are hypotheses
then add the hypothesis $\exists Y(Y \in B \wedge \mathrm{apply}(F, X, Y))$

rule `maps2`: if $\mathrm{maps}(F, A, B)$, $X \in A$, $Y1 \in B$, $Y2 \in B$,
$\mathrm{apply}(F, X, Y1)$ and $\mathrm{apply}(F, X, Y2)$ are hypotheses
then add the hypothesis $Y1 = Y2$

rule `inv1`: if $\mathrm{inv}(F, B, A)$:C and $X \in C$ are hypotheses
then add the hypothesis $X \in A$

rule `inv2`: if $\mathrm{inv}(F, B, A)$:C and $X \in C$ are hypotheses

[7] the definitions of $\subset$ and $\cap$ have already been given in section 2.1

[8] for all rules, add "if it is not yet a hypothesis"

then add the hypothesis $\exists Y(Y \in B \wedge \text{apply}(FX, Y))$

rule `inv3`: if $\text{inv}(F, B, A)$:C, $Y \in B$ and $\text{apply}(F, X, Y)$ are hypotheses
then add the hypothesis $X \in C$

Here is the MUSCADET proof [9] . The initial theorem to be proved is numbered 0. Its conclusion is the given conjecture :

$\forall F\ \forall A\ \forall B\ \forall X\ \forall Y$
$(\text{maps}(F, A, B) \wedge X \subset B \wedge Y \subset B$
$\rightarrow \text{inv}(F, X \cap Y, A) =_{\text{set}} \text{inv}(F, X, A) \cap \text{inv}(F, Y, A))$

five applications of the $\forall$ rule remove the universal quantifier and create corresponding objects [10] f, a, b, $b1$, $b2$
and the new conclusion is

$\text{maps}(f, a, b)\ \wedge\ b1 \subset b\ \wedge\ b2 \subset b) \rightarrow$
$\text{inv}(f, b1 \cap b2, a) =_{\text{set}} \text{inv}(f, b1, a) \cap \text{inv}(f, b2, a)$

. . . . . rule $\forall$

elimination of functional symbols
new conclusion

$\text{maps}(f, a, b) \wedge b1 \subset b \wedge b2 \subset b) \rightarrow$
$!A{:}b1 \cap b2,\ !B{:}\text{inv}(f, A, a),$
$!C{:}\text{inv}(f, b1, a), !D{:}\text{inv}(f, b2, a),\ !E{:}C \cap D,$
$B =_{\text{set}} E$

. . . . . rule elifun

separation of hypotheses and conclusion
add hypotheses $\text{maps}(f, a, b)$, $b1 \subset b$ and $b2 \subset b$
new conclusion

$!A{:}b1 \cap b2, !B{:}\text{inv}(f, A, a),$
$!C{:}\text{inv}(f, b1, a), !D{:}\text{inv}(f, b2, a), !E{:}C \cap D,$
$B =_{\text{set}} E$

. . . . . rule $\rightarrow$

three applications of the ! *rule create objects and their definitions*
add objects : $b3$, $a0$, $a1$, $a2$ and $a3$
add hypotheses $b1 \cap b2 : b3$, $\text{inv}(f, b3, a) : a0$,
$\text{inv}(f, b1, a){:}a1$, $\text{inv}(f, b2, a) : a2$, $a1 \cap a2 : a3$
new conclusion $a0 =_{\text{set}} a3$

. . . . . rule !

definition of the conclusion
new conclusion $a0 \subset a3 \wedge a3 \subset a0$

. . . . . rule defconcl

[9] Some useless actions have been removed.

[10] the system names the objects `o, o1, o2, o3`, and so on, but I give here the names f, a, b, $b1$ and so on, which makes the proof more easily readable by a human reader.

as the conclusion is now a conjunction, the theorem 0 is split into two sub-theorems 1 and 2 which will be proved one after the other

. . . . . SUB-THEOREM 1
the conclusion of the first sub-theorem is the first sub-formula of the conjunction ; hypotheses and other facts are copied from theorem 0 to sub-theorem 1
new conclusion $a0 \subset a3$ . . . . . creation of sub-theorem 1
definition of the conclusion
new conclusion $\forall A(A \in a0 \rightarrow A \in a3)$ . . . . . rule defconcl
add object x
new conclusion $x \in a0 \rightarrow x \in a3$ . . . . . rule $\forall$
add hypothesis $x \in a0$
new conclusion $x \in a3$ . . . . . rule $\rightarrow$
add hypothesis $x \in a$ . . . . . rule inv1
add hypothesis $\exists A(A \in b3 \wedge \text{apply}(f, x, A))$
x has an image in $b1 \cap b2$ because it belongs to $f^{-1}(b1 \cap b2)$
. . . . . rule inv2
treatment of the existential hypothesis : creation of an object y in $b1 \cap b2$ which is the image of x
add object y
add hypotheses $y \in b3$ and $\text{apply}(f, x, y)$
add treated-hypothesis $\exists A(A \in b3 \wedge \text{apply}(f, x, A))$
this fact is used to memorize the fact that the hypothesis has been treated (it cannot be removed because the rule which has added it would add it again, and infinitely loop)
. . . . . rule $\exists$
add hypothesis $y \in b1$ (*since* $y \in b1 \cap b2$) . . . . . rule $\cap 1$
add hypothesis $y \in b$ (*since* $b1 \subset b$) . . . . . rule $\subset$
add hypothesis $y \in b2$ (*since* $y \in b1 \cap b2$) . . . . . rule $\cap 2$
add hypothesis $x \in a1$
(x *belongs to* $f^{-1}(b1)$ *since its image* y *belongs to* $b1$)
. . . . . rule inv3
add hypothesis $x \in a2$
(x *belongs to* $f^{-1}(b2)$ *since its image* y *belongs to* $b2$)
. . . . . rule inv3
add hypothesis $x \in a3$
(x *belongs to* $f^{-1}(b1) \cap f^{-1}(b2)$ *since its belongs to* $f^{-1}(b1)$ *and to* $f^{-1}(b2)$)
. . . . . rule $\cap 3$
the last hypothesis added is the actual conclusion to be proved, then the sub-theorem 1 is proved and its conclusion is put at `true` *to memorize the fact that it is proved*
new conclusion (of theorem 1) `true` . . . . . rule stop1
theorem 1 proved

since sub-theorem 1 is proved, the first sub-formula of the conclusion of theo-

rem 0 is removed
new conclusion (of theorem 0) $a3 \subset a0$
and the second sub-theorem is now being proved

. . . . . SUB-THEOREM 2
new conclusion $a3 \subset a0$
. . . . . creation of sub-theorem 2
definition of the conclusion
new conclusion $\forall A(A \in a3 \rightarrow A \in a0)$ . . . . . rule defconcl
add object t
new conclusion $t \in a3 \rightarrow t \in a0$ . . . . . rule $\forall$
add hypothesis $t \in a3$
new conclusion $t \in a0$ . . . . . rule $\rightarrow$
add hypothesis $t \in a1$
(*since t belongs to* $f^{-1}(b1) \cap f^{-1}(b2)$ *it belongs to* $f^{-1}(b1)$)
. . . . . rule $\cap$1
add hypothesis $t \in a2$ (*and to* $f^{-1}(b2)$)
. . . . . rule $\cap$2
add hypothesis $t \in a$ (*and to the domaine a*)
. . . . . rule inv3
add hypothesis $\exists A(A \in b1 \wedge \text{apply}(f, t, A))$
(*t has an image in b*1 *since it belongs to* $f^{-1}(b1)$)
. . . . . rule inv2
add hypothesis $\exists A(A \in b2 \wedge \text{apply}(f, t, A))$
(*and an image in b*2 *since it belongs to* $f^{-1}(b2)$)
. . . . . rule inv2
treatment of the first existential hypothesis of the sub-theorem :
*creation of u, image of t in b*1
add object u
add hypotheses $u \in b1$ and $\text{apply}(f, t, u)$
add treated-hypothesis $\exists A(A \in b1 \wedge \text{apply}(f, t, A))$
. . . . . rule $\exists$
add hypothesis $u \in b$
(u *belongs to b since* $b1 \subset b$) . . . . . rule $\subset$
treatment of the second existential hypothesis of the sub-theorem :
*creation of v, image of t in b*2
add object v
add hypothesis $v \in b2$ and $\text{apply}(f, t, v)$
add treated-hypothesis $\exists A(A \in b2 \wedge \text{apply}(f, t, A))$
. . . . . rule $\exists$
add hypothesis $v \in b$
v *belongs to b since* $b2 \subset b$ . . . . . rule $\subset$
add hypothesis $u = v$ (*since the image is unique*) . . . . . rule maps2
replace v by u propagate and remove v
add hypothesis $u \in b2$ . . . . . rule =

add hypothesis $u \in b3$
(u *belongs to* $b1 \cap b2$ *since* u *belongs to* $b1$ *and to* $b2$)
. . . . . rule $\cap 3$
add hypothesis $t \in a0$
(t *belongs to* $f^{-1}(b1 \cap b2)$ *since its image* u *belongs to* $b1 \cap b2$)
. . . . . rule inv3
new conclusion (of theorem 2) `true` . . . . . rule stop1
theorem 2 proved

since sub-theorem 2 is proved, there is no more formula to prove in the conclusion of theorem 0, so it is proved
new conclusion (of theorem 0) `true` . . . . . rule concl$\wedge$
theorem 0 proved

3.2 Handling concepts with negation : a theorem about complements

Here is an example of a theorem which was proved by MUSCADET in less than 0.01 second during the last competition (2005), and also by Vampire (Riazanov,Voronkov, 2002) but in 99 seconds, and by none of the other provers.

Theorem SET012+4. *If A is a subset of E then the complement in E of its complement is equal to itself,*

Its formal statement in first order predicate calculus is
$$\forall E\ \forall A(A \subset E \rightarrow (E - (E - A)) =_{\text{set}} A)$$

The definition of complement in a set (or difference) is
$$\forall E\ \forall A(X \in (E - A) \leftrightarrow X \in E \wedge \neg(X \in A)$$

The rules built from this definition are the following :
rule `diff1` : if $(E - A) : B$ and $X \in B$ are hypotheses
then add the hypothesis $X \in E$ [11]
rule `diff2` : if $(E - A) : B$, $X \in B$ and $X \in A$ are hypotheses
then add the hypothesis `false`
The hypothesis `false` means that there is a contradiction in the hypotheses, hence the sub-theorem to which such a rule may be applied is proved.
rule `diff3` : if $(E - A) : B$, $X \in E$ and $\neg(X \in A)$ are hypotheses
then add the hypothesis $X \in B$

[11] for all rules, add "if it is not yet a hypothesis"

rule `diff4` : if $(E - A) : B$ and $X \in E$ are hypotheses
then add the hypothesis $X \in A \vee X \in B$.

By manipulations analogous to those executed in the preceding example, MUSCADET creates objects e, a, b and c with their properties stored as hypotheses :

$a \subset e$
$(e - a) : b$
$(e - b) : c$

and the conclusion to be proved is

$c =_{\text{set}} a$

which is replaced by its definition

$c \subset a \wedge a \subset c$.

Two sub-theorems are to be proved.

PROOF OF SUB-THEOREM 1
The conclusion

$c \subset a$

is replaced by its definition

$\forall X(X \in c \rightarrow X \in a)$

an object x is created such that

$x \in c$

and the conclusion is

$x \in a$.

The rules `diff1` and `diff4` add the hypotheses

$x \in e$ and $x \in a \vee x \in b$.

After this splitting of this disjunctive hypothesis, two sub-theorems have to be proved

SUB-THEOREM 11 with the hypothesis $x \in a$, which is the conclusion to be proved, and

SUB-THEOREM 12 with the hypothesis $x \in b$, and the `diff2` rule brings the contradiction

PROOF OF SUB-THEOREM 2
The conclusion

$a \subset c$

is replaced by its definition. Then an object y is created such that

$y \in a$

and the conclusion is

$y \in c$.

The `diff1` and `diff4` rules the hypotheses

$y \in e$ and $y \in b \vee y \in c$.

then two sub-theorems 21 and 22 are proved as before.

The following theorem

Theorem SET601+3.p.
$\forall A\ \forall B\ \forall C\ ((A \cap B) \cup (B \cap C) \cup (C \cap A) =_{\text{set}} (A \cup B) \cap (B \cup C) \cap (C \cup A))$

was proved by MUSCADET in less than 0.01 second at CASC-20 (2005). It was also proved by Prover9 (McCune, 2005) in less than 0.01 second. Three other provers also proved it, but in at least 146 seconds.

MUSCADET splits this theorem into five final sub-theorems. We can see that Prover9 also splits this theorem since, as the author wrote in his CASC system description, "a preprocessing step attempts to reduce the problem to independent subproblems".

Here, these splittings are particularly efficient.

The proof of MUSCADET looks like an elementary proof given by an unexperienced mathematician. More advanced mathematicians would give a more direct and smarter proof by using their knowledge about the distributivity of intersections and unions.

The rules applied by MUSCADET are both those which were described in previous sections and other rules built from the definition of union.

definition : $\forall A\ \forall B\ \forall X(X \in A \cup B \leftrightarrow X \in A \vee X \in B)$

built rules :
rule $\cup 1$: if $A \cup B : C$ and $X \in C$ are hypotheses
then add the hypothesis $X \in A \vee X \in B$ [12]
rule $\cup 2$: if $A \cup B : C$ and $X \in A$ are hypotheses
then add the hypothesis $X \in C$
rule $\cup 3$: if $A \cup B : C$ and $X \in B$ are hypotheses
then add the hypothesis $X \in C$

Note that the last two rules directly lead to elementary belongings whereas the first rule leads to disjunctive hypotheses which, in turn, will lead to splittings.

[12] for all rules, add "if it is not yet a hypothesis"

The initial theorem is split into two sub-theorems by splitting the equality of sets into two inclusions.

SUB-THEOREM 1
to prove the conclusion
$(a \cap b) \cup (b \cap c) \cup (c \cap a) \subset (a \cup b) \cap (b \cup c) \cap (c \cup a)$
where a, b, c are objects and all the intersections and unions such as $a \cap b$ have been created as new objects with their own names, an object x is taken in $(a \cap b) \cup (b \cap c) \cup (c \cap a)$ and it must be proved that x belongs to $(a \cup b) \cap (b \cup c) \cap (c \cup a)$

Belonging to the union leads, in two steps, to three sub-theorems which are easily proved.
- split of $x \in (a \cap b) \cup (b \cap c) \cup (c \cap a)$
- SUB-THEOREM 11 : $x \in (a \cap b) \cup (b \cap c)$
  -split
  -SUB-THEOREM 111 : x belongs to $a \cap b$ hence to a and to b
  hence to $a \cup b$, $c \cup a$, $b \cup c$ and to their intersection
  -SUB-THEOREM 112 : x belongs to $b \cap c$ hence to b and to c
  hence to $(b \cup c)$, $(a \cup b)$, $c \cup a)$ and to their intersection
- SUB-THEOREM 12 : x belongs to $c \cap a$ hence to c and to a,
  hence to $c \cup a$, $b \cup c$, $a \cup b$ and to their intersection

SUB-THEOREM 2
to prove the conclusion
$(a \cup b) \cap (b \cup c) \cap (c \cup a) \subset (a \cap b) \cup (b \cap c) \cup (c \cap a)$
an object y is taken in $(a \cup b) \cap (b \cup c) \cap (c \cup a)$,
y belongs to $a \cup b$, $b \cup c$ and $c \cup a$,
and the belonging to these unions leads to four final sub-theorems :
- split of $y \in a \cup b$
- SUB-THEOREM 21 : y belongs to a
  - split of $y \in b \cup c$
  - SUB-THEOREM 211 : y belongs to b hence to $a \cap b$
  and to $(a \cap b) \cup (b \cap c) \cup (c \cap a)$
  - SUB-THEOREM 212 : y belongs to c hence to $c \cap a$
  and to $(a \cap b) \cup (b \cap c) \cup (c \cap a)$
- SUB-THEOREM 22 : y belongs to b
  - $y \in b \cup c$ does not have to be split
  - split of $y \in c \cup a$
  - SUB-THEOREM 221 : y belongs to c hence to $(b \cap c)$
  and to $(a \cap b) \cup (b \cap c) \cup (c \cap a)$
  - SUB-THEOREM 222 : y belongs to a hence to $(c \cap a)$
  and to $(a \cap b) \cup (b \cap c) \cup (c \cap a)$

Here is another example of a theorem which was proved by MUSCADET in less than 0.01 second during CASC-20 (2005), and also by Vampire (Riazanov,Voronkov, 2002) but in 240 seconds, and by none of the other provers.

Theorem SET703+4
The union of two singletons $\{A\}$ *and* $\{B\}$ *is equal to the unordered pair* $\{A, B\}$

Its formal statement is
$\forall A\ \forall B\ (\{A\} \cup \{B\} =_{\text{set}} \{A, B\})$

The definitions of singleton $\{A\}$ and unordered-pair $\{A, B\}$ are
$\forall A\ \forall X(X \in \{A\} \leftrightarrow X = A)$
$\forall A\ \forall B\ \forall X(X \in \{A, B\} \leftrightarrow X = A \vee X = B)$

The rules built from these definitions are
rule `singleton1`: if $\{A\} : S$ and $X \in S$ are hypotheses
then add the hypothesis $X = A$
rule `singleton2`: if $\{A\} : S$ is a hypothesis
then add the hypothesis $A \in S$
rule `pair1`: if $\{A, B\} : P$ and $X \in P$ are hypotheses
then add the hypothesis $X = A \vee X = B$
rule `pair2`: if $\{A, B\} : P$ is a hypothesis
then add the hypothesis $A \in P$
rule `pair3`: if $\{A, B\} : P$ is a hypothesis
then add the hypothesis $B \in P$

PROOF

The conclusion of the first theorem to be proved is the statement of the conjecture.

add objects a and b
new conclusion $\{a\} \cup \{b\} =_{\text{set}} \{a, b\}$ ..... rule $\forall$
new conclusion
$!A{:}\{a\},\ !B{:}\{b\},\ !C{:}A \cup B,\ !D{:}\{a, b\},\ C =_{\text{set}} D$ ..... rule elifun
add object $a1$, $b1$, $a1b1$ and ab
add hypotheses $\{a\} : a1$, $\{b\} : b1$, $a1 \cup b1 : a1b1$ and $\{a, b\} : ab$
new conclusion $a1b1 =_{\text{set}} ab$ ..... rule !
add hypothesis $a \in a1$ ..... rule singleton2
add hypothesis $a \in a1b1$ ..... rule $\cup$2
add hypothesis $b \in b1$ ..... rule singleton2
add hypothesis $b \in a1b1$ ..... rule $\cup$3

add hypothesis $a \in ab$ . . . . . rule pair2
add hypothesis $b \in ab$ . . . . . rule pair3
definition of the conclusion
new conclusion $a1b1 \subset ab \wedge ab \subset a1b1$ . . . . . rule defconcl

. . . . . SUB-THEOREM 1
new conclusion $a1b1 \subset ab$
. . . . . creation of sub-theorem 1
definition of the conclusion
new conclusion $\forall A(A \in a1b1 \rightarrow A \in ab)$ . . . . . rule defconcl
add object x
new conclusion $x \in a1b1 \rightarrow x \in ab$ . . . . . rule $\forall$
add hypothesis $x \in a1b1$
new conclusion $x \in ab$ . . . . . rule $\rightarrow$
add hypothesis $x \in a1 \vee x \in b1$ . . . . . rule $\cup 1$
treatment of the disjunctive hypothesis $x \in a1 \vee x \in b1$
new conclusion $(x \in a1 \rightarrow x \in ab) \wedge (x \in b1 \rightarrow x \in ab)$
add treated-hypothesis $x \in a1 \vee x \in b1$ . . . . . rule $\vee$

. . . . . SUB-THEOREM 11
new conclusion $x \in a1 \rightarrow x \in ab$
. . . . . creation of sub-theorem 11
add hypothesis $x \in a1$
new conclusion $x \in ab$ . . . . . rule $\rightarrow$
add hypothesis $x = a$ . . . . . rule singleton1
replace a by x propagate and remove a
add hypothesis $x \in ab$ . . . . . rule $=$
new conclusion (of theorem 11) `true` . . . . . rule stop1
theorem 11 proved

new conclusion (of theorem 1) $x \in b1 \rightarrow x \in ab$

. . . . . SUB-THEOREM 12
new conclusion $x \in b1 \rightarrow x \in ab$
. . . . . creation of sub-theorem 12
proved in an analogous manner
theorem 12 proved

new conclusion (of theorem 1) `true` . . . . . rule concl$\wedge$
theorem 1 proved

new conclusion (of theorem 0) $ab \subset a1b1$

. . . . . SUB-THEOREM 2
new conclusion $ab \subset a1b1$

. . . . . creation of sub-theorem 2
definition of the conclusion
new conclusion
$\forall A(A \in ab \rightarrow A \in a1b1)$ . . . . . rule defconcl
add object y
new conclusion $y \in ab \rightarrow y \in a1b1$ . . . . . rule $\forall$
add hypothesis $y \in ab$
new conclusion $y \in a1b1$ . . . . . rule $\rightarrow$
add hypothesis $y = a \vee y = b$ . . . . . rule pair1
treatment of the disjunctive hypothesis $y = a \vee y = b$
new conclusion $(y = a \rightarrow y \in a1b1) \wedge (y = b \rightarrow y \in a1b1)$
add treated-hypothesis $y = a \vee y = b$ . . . . . rule $\vee$

. . . . . SUB-THEOREM 21
new conclusion $y = a \rightarrow y \in a1b1$
. . . . . creation of sub-theorem 21
add hypothesis $y = a$
new conclusion $y \in a1b1$ . . . . . rule $\rightarrow$
replace a by y propagate and remove a
add hypothesis $y \in a1b1$ . . . . . rule =
new conclusion (of theorem 21) `true` . . . . . rule stop1
theorem 21 proved

new conclusion (of theorem 2) $y = b \rightarrow y \in a1b1$

. . . . . SUB-THEOREM 22
new conclusion $y = b \rightarrow y \in a1b1$
. . . . . creation of sub-theorem 22
proved in an analogous manner
theorem 22 proved

new conclusion (of theorem 2) `true` . . . . . rule concl$\wedge$
theorem 2 proved

new conclusion (of theorem 0) `true` . . . . . rule concl$\wedge$
theorem 0 proved

4 Examples of MUSCADET proofs from CASC-J3 (2006)

Some problems of CASC-J3 are taken from two sets of new problems which were new problems of the release v3.2.0 of the TPTP library.

One set of 119 new problems (SET866to999+1 and SEUxxx+1) comes from the Mizar Library (Mizar; Bylinsli, 1989a; Bylinski, 1989b) after they have been translated into the TPTP format (Urban, 2003).

The other set of 29 new problems (SET789to817+4) about relations was recently proposed by myself to be incorporated in the TPTP library.

4.1 A theorem from Mizar

MUSCADET is rather well adapted to the problems translated from the Mizar Library. This is not surprising since they are expressed in a rather natural mathematical manner.

In the FNE[13] category, MUSCADET, although it is not efficient in this category, solved all of them (six, which were easy)

In the FEQ[14] category, it solved half of the problems (five out of ten). In particular it solved problem SEU075+1 which was solved by only one other entrant system (by the preceding version 8.0 of Vampire, winner of CASC-20 in 2005, but not by the current Vampire version 8.1).

Nevertheless, MUSCADET took a lot of time to solve it. The reason is not the difficulty of the proof itself but the fact that many of the given axioms are useless. Only four out of the forty axioms are useful. Many of them lead to the creation of useless objects and hypotheses which lead to other useless objects and hypotheses. Therefore the MUSCADET proof is exceptionally long. As the growth is linear, not exponential, MUSCADET finally managed to find the proof.

Here are the statements of this problem and the useful steps of the MUSCADET proof.

Theorem SEU075+1.

If the domains of g and h are equal and equal to the range of f, and if $g_o f = h_o f$ then $g = h$.

Its formal statement in first order predicate calculus is

$$\forall A\ \forall B(\text{relation}(B) \wedge \text{function}(B) \\ \rightarrow \forall C(\text{relation}(C) \wedge \text{function}(C)$$

[13] The FOF division is divided into two categories, FNE with no equality
[14] ... and FEQ with equality.

$\rightarrow \forall D(\text{relation}(D) \wedge \text{function}(D)$
$\rightarrow (A = \text{range(B)} \wedge \ \text{dom}(C) = A \ \wedge \ \text{dom}(D) = A$
$\wedge \ \text{composition}(B, C) = \text{composition}(B, D)$
$\rightarrow C = D))))$.

The following axioms are given
`d5_funct_1`:
$\forall A(\text{relation}(A) \wedge \text{function}(A)$
$\rightarrow \forall B(B = rng(A)$
$\leftrightarrow \forall C(C \in B \leftrightarrow \exists D(D \in \text{dom}(A) \wedge C = \text{apply}(A, D)))))$
`t9_funct_1`:
$\forall A(\text{relation}(A) \wedge \text{function}(A)$
$\rightarrow \forall B(\text{relation}(B) \wedge \text{function}(B)$
$\rightarrow (\text{dom}(A) = \text{dom}(B)$
$\wedge \forall C(C \in \text{dom}(A) \rightarrow \text{apply}(A, C) = \text{apply}(B, C))$
$\rightarrow A = B)))$
`t23_funct_1`:
$\forall A, B(\text{relation}(B) \wedge \text{function}(B)$
$\rightarrow \forall C(\text{relation}(C) \wedge \text{function}(C)$
$\rightarrow (A \in \text{dom}(B))$
$\rightarrow \text{apply}(\text{composition}(B, C), A) = \text{apply}(C, \text{apply}(B, A)))))$.

From these definitions the following rules were built
rule `d5_funct_1_exists`:
if relation(B), function(B), rng(B):E and $G \in E$ are hypotheses
then add the hypothesis $\exists I \ (!J : dom(B), I \in J) \wedge \text{apply}(B, I) : G)$

rule `t9_funct_1_sc`:
if relation(B), function(B), relation(E), function(E), dom(B):H
and dom(E):H are hypotheses
and if the conclusion is $B = E$
then the new conclusion is
$\forall K((!L{:}\text{dom}(B), K \in L) \rightarrow (!M{:}\text{apply}(B, K),\text{apply}(E, K){:}M))$
(comment : this new conclusion is a sufficient condition for the preceding one)

rule `t23_funct_1`:
if relation(B), function(B), relation(E), function(E), dom(B):H,
$J \in H$, composition(B, E):L, apply(L, J):N and apply(B, J):P
are hypotheses
then add the hypothesis apply(E, P):N

rule `t23_funct_1_exists`:
if relation(B), function(B), relation(E), function(E), dom(B):H, $J \in H$,
composition(B, E):L and apply(L, J):N are hypotheses
and if apply(B, J):P is not a hypothesis

then add the hypothesis $\exists(P, \text{apply}(B, J):P \wedge \text{apply}(E, P):N)$

There are also the general rules
rule concl! :
if the conclusion is $!Y:F(..), B(Y)$
and if there is no hypothesis $Z:F(..)$
then create a new object $Y1$
add the hypothesis $Y1:F(..)$
and the new conclusion is $B(Y1)$

rule concl: :
if the conclusion is $Y:F(..)$
then the new conclusion is $!Z:F(..), Z = Y$

Here is the MUSCADET proof

As decribed in the preceding examples,
- the $\forall$ rule removes the universal quantifiers
and creates the corresponding objects x, $f1$, $f2$ and $f3$,
- the `elifun` rule eliminates the funtional symbols,
and
- the $\rightarrow$ rule separates hypotheses and conclusion.

So, we have the hypotheses
relation($f1$)
function($f1$)
relation($f2$)
function($f2$)
relation($f3$)
function($f3$)
rng($f1$):x
dom($f2$):x
dom($f3$):x
composition($f1, f2$):$f4$
composition($f1, f3$):$f4$

and the new conclusion is
$f2 = f3$
Then the `t9_funct_1_sc` rule replaces the conclusion by
$\forall(A\ ((!B:\text{dom}(f2), A \in B) \rightarrow\ (!C:\text{apply}(f2, A), \text{apply}(f3, A):C))$
which is a sufficient condition to have $f2 = f3$.

Then
- the $\forall$ rule creates the object $o5$,

- the $\rightarrow$ rule separate hypotheses and conclusion
and,
- by the rules `concl!` (twice) and `concl:`, we have the hypotheses
$o5 \in o$
apply$(f2, o5)$:$o6$
apply$(f3, o5)$:$o7$
and the conclusion to be proved is
$o7 = o6$

then the `d5_funct_1_exists` rule adds the hypothesis
$\exists A\ ((!B$:dom$(f1), A \in B) \wedge$ apply$(f1, A)$:$o5)$
which is treated by the $\exists$ rule which adds the hypotheses
dom$(f1)$:$o9$
$o8 \in o9$
apply$(f1, o8)$:$o5$
then the `t23_funct_1_exists` rule add the hypothesis
$\exists A$ (apply$(f4, o8)$:A $\wedge$ $(!B$:apply$(f1, o8)$, apply$(f2, B)$:$A))$
which is treated by the $\exists$ rule which adds the object $o70$ and the hypotheses
apply$(o4, o8)$:$o70$
apply$(o2, o5)$:$o70$

the `egaldef` rule adds the hypothesis
$o7 = o6$
which is the conclusion to be proved

4.2 Theorems about relations

These theorems concern properties of relations, expressed in a naïve manner, Six of these problems, all FEQ, were proposed at the CASC-J3 (2006) competition. One of them was proved only by MUSCADET.

Theorem SET796+4. *If $R(a, b)$ then a is the greatest lower bound of the unordered pair $\{a, b\}$*

Formal statement :
$\forall R\ \forall E \forall A \forall B$ (order(R, E) $\wedge$ $A \in E$ $\wedge$ $B \in E$ $\wedge$ apply$(R, A, B))$
$\rightarrow glb(A, \{A, B\}, R, E)$

With the help of the "apply" predicate the definition of "order" may be given as a first order formula :
$\forall R\ \forall E$ (order(R, E)
$\leftrightarrow$
$\forall X(X \in E \leftrightarrow$ apply$(R, X, Y))$

$$\wedge\, \forall X \forall Y\ (X \in E \wedge Y \in E \rightarrow (\mathrm{apply}(R, X, Y) \wedge R(Y, X) \rightarrow X = Y))$$
$$\wedge\, \forall X \forall Y \forall Z (X \in E\ \wedge\ Y \in E\ \wedge\ Z \in E$$
$$\rightarrow (\mathrm{apply}(R, X, Y) \wedge \mathrm{apply}(R, Y, Z) \rightarrow \mathrm{apply}(R, X, Z)))$$

Except the use of predicate "apply", the MUSCADET proof looks like the proof that a human would have given.

4.3 Theorems about ordinal numbers

For problems about ordinal numbers, in addition to the definitions concerning strict order, the following axiom

$\forall X\ \forall Y(\mathrm{apply}(\in, X, Y) \leftrightarrow X \in Y)$

allows to handle "belonging" either as a predicate, or as a constant.
This axiom could be written

$\forall X\ \forall Y(\mathrm{apply}(\mathrm{member}, X, Y) \leftrightarrow \mathrm{member}(X, Y))$

in the TPTP format but, at G. Sutcliffe's request, two different names were used for the constant and the predicate

$\forall X\ \forall Y(\mathrm{apply}(\mathrm{member\_predicate}, X, Y) \leftrightarrow \mathrm{member}(X, Y)$

In one or the other formulation, MUSCADET simply builds and uses the rules :

if $\mathrm{apply}(\in, XE)$ is a hypothesis and $X \in E$ is not a hypothesis
then add the new hypothesis $X \in E$

and

if $X \in E$ is a hypothesis and $\mathrm{apply}(\in, X, E)$ is not a hypothesis
then add the new hypothesis $\mathrm{apply}(\in, X, E)$

The definition of the ordinal numbers is the following where "on" (constant) is the collection of ordinal numbers

$\forall A\ (A \in^{15}\ \mathrm{on}$
$\leftrightarrow \mathrm{set}(A) \wedge\ \mathrm{strict\_well\_order}(\in^{16}, A) \wedge \forall X(X \in^{17} A \rightarrow X \subset A))$

Six problems about ordinal numbers, all FEQ, were proposed at the CASC-J3 (2006) competition. Three of them were proved only by MUSCADET.

One is theorem SET808+4.p

$\forall A\ (A \in \mathrm{on} \rightarrow A \subset \mathrm{on})$

The two others are about the sum of respectively an ordinal number or the successor of an ordinal number. The MUSCADET proofs look like the proofs that a human would have given.

[15] "member" in TPTP format
[16] "member_predicate" in TPTP format
[17] "member"

5 Why MUSCADET may be an efficient system

First, I will point out a major difference between the resolution principle and the natural methods of MUSCADET.

With the resolution principle, there is a very theoretically efficient deduction rule which may generate so many clauses that the proof, even theoretically obtainable, may not be found in a reasonable time. Strategies are written to limit the exponential "combinatorial explosion".

With the natural methods of MUSCADET, there is a high but reasonable number of rules, given or automatically built. The conditions for their application are strict and they usually lead to a linear growth. Some of them may be expansive (infinite creation of objects) or exponential (splittings) but they have low priority and if time is out, this often means that a proof could not be obtained. This may also mean that priorities are not right and prevent an efficient rule to be applied. Strategies must therefore be written to define further rules and metarules, and consequently the number of useful deduced facts.

The efficiency of some MUSCADET characteristics will now be commented.

5.1 Representations

The first order formula of the conjecture to be proved is decomposed into various facts.

First there are the hypotheses and the conclusion to be proved. The simplest of these facts look like clauses but there are differences :

- hypotheses and conclusion are closed formulas;
- a hypothesis P (or $\neg P$) is not handled as a conclusion (or as part of a disjunctive conclusion) $\neg P$ (or P);
- quantifiers are not systematically removed and stay as long as possible near the scope of the quantified variable;
- there are no universal hypotheses; instead there are new local rules;
- a conclusion may be disjunctive; in this case, there are rules to handle it, for example
  - if A is found to be a new hypothesis,
    then the conclusion $A \vee B$ is true
  - if the conclusion is $\neg A \vee B$,
    then the new hypothesis A is added
    and the new conclusion is B

Many objects are created. They will be useful, for example, to simply verify a conclusion of the form $\exists X P(X)$. If such an object does not exist yet a more complex mechanism must be used.

Objects correspond to Skolem constants, but there are no other Skolem functions. We will see in subsection 5.3 the mechanism that replaces them.

Other facts are the definitions and lemmas. Metarules build rules from their formal statements. They contain rewriting rules and build step by step the conditions and the actions of these new rules. The formal statements of the lemmas are thus no longer useful. For the definitions, their statements, after the elimination of functional symbols (see 2.2), are still useful (for the definition of the conclusion). The mechanism is described in (Pastre, 1989). We have seen several such rules in preceding sections.

Hypotheses are never removed. When an existential or disjunctive hypothesis H is treated, a new fact is added to register that this hypothesis has been treated and will prevent its being added again.

5.2 Splittings

As Bledsoe (1971) already pointed out, splitting is very efficient. We also saw in section 3.3 the efficiency of splitting since Prover9 (McCune, 2005) and MUSCADET proved a theorem in less than 0.01 second whereas other provers needed at least 146 seconds or failed.

In MUSCADET, splitting is not only a preprocessing step but it may also be done at all levels of the proof.

Vampire (Riazanov,Voronkov, 2002) uses another sort of splitting rule which applies to clauses (Riazanov,Voronkov, 2001). But this splitting requires to introduce new predicates and thus has to be limited. The authors say "Although the use of splitting results in degradation of performance on the average, there exist many problems which VAMPIRE can solve in reasonable time only with splitting".

There are various versions of this splitting rule. This explains perhaps why Vampire 7.0 proved theorem SET711+4 [18] in less than 0.01 second, as well as MUSCADET, and Vampire 8.0 needed 170 seconds (no other prover succeded), whereas theorem SET601+3, which we have seen in section 3.3, was proved by Vampire 8.0 (146 seconds needed) but not by Vampire 7.0 (timeout).

[18] This theorem states the uniqueness of the inverse image of a one-to-one mapping.

We have already seen in sections 2.2 and 3.1 how a formula of the form $P(F(X))$ [19] is treated when it appears in a conclusion, after it has been instantiated. But such a formula may appear as a sub-formula anywhere in a conjecture or in a definition or lemma and it may contain variables. There is no skolemisation.

$P(F(X))$
might be replaced by
$\forall Y(Y{=}F(X)) \rightarrow P(Y))$
or by
$\exists Y(Y{=}F(X) \wedge P(Y))$
and one or the other of these two formulas could be more adequate, depending on its position in relation to implications and negations when it is handled. It is the reason why the quantifier "!" ("for the only ... equal to ...") was introduced.

$P(F(X))$
is replaced by
$!Y{:}F(X), P(Y)$
which means
"for the only Y equal to $F(X)$, $P(Y)$"

It is only when the sub-formula $!Y{:}F(X), P(Y)$ appears as a conclusion to be proved or as a hypothesis to be added or as the sub-formula to be processed during building rules that this sub-formula is dealt with.

The rule ! : if the conclusion is of the form $!Y{:}F(...), P(Y)$
then if there is already a hypothesis $F(...){:}Y1$
then the new conclusion is $P(Y1)$
else create a new object
add the hypothesis $F(...)$:<this new object>
and the new conclusion is $P($<this new object>$)$
treats "!" in the same manner as $\forall$.

The rule
to add a hypothesis H :
if H is of the form $!Y{:}F(...), P(Y)$
then if there is already a hypothesis $F(...){:}Y1$
then add the hypothesis $P(Y1)$
else create a new object

[19] In this paper $F(X)$ is used for any term containing the free variable Y, $P(Y)$ is used for any formula containing the free variable X

and add the hypotheses $F(...)$:<this new object>
and P(<this new object>)

treats "!" in the same manner as $\exists$ but immediately.

Note that these treatments are done on closed formulas, the created objects are Skolem constants, there is no need for other Skolem functions.

In metarules, building rules from definitions and lemmas, "!" is also treated either as $\forall$ or as $\exists$ according to its position in the sub-formula which is being treated.

We also saw in sections 2.2 and 3.1 that the mechanism is recursive.
A formula $f(g(a))$ leads to objects b and c and to hypotheses $g(a):b$ and $f(b):c$.

The first advantage of this flattening is that all intermediary terms are created as objects and may be considered in a rule, as seen with the rules $\cap i$ and $\cup i$

The second advantage is that if $f(a)$ and $f(b)$ are found to be equal, this will be memorized by the fact that there will be now only one object c to denote both terms, with the hypotheses $f(a):c$ and $f(b):c$. $f(a)$ and $f(b)$ will now have the same properties thanks to the intermediary object c.

On the downside, if the succession of functional symbols is characteristic in a term $f(g(a))$ then it is not easily visible. It is the reason why MUSCADET is not adapted to work in group theory, for example.

5.4 Building efficient rules

The building of rules has already been mentioned several times. It is a complex mechanism which is described in detail in (Pastre, 1989) and it involves many rules. I will here only develop a crucial point which is in relation with the treatment of functional symbols.

Firstly, rules are adapted to the choice of working with positive properties. If we have two sets such that $A \subset B$, the $\subset$ rule (see section 2.1) will be applied each time an element is found to be in A. These elementary properties will certainly be useful. If there are no sets such as $A \subset B$, the concept $\subset$ is probably not pertinent. Surely, it must not have priority.

This set of rules is not complete, but successes largely make up for failures.

Secondly, if a definition or a lemma leads to a statement of the form
$H \rightarrow P(F(X))$

(for example $X \in A \rightarrow X \in A \cup B$)
the following rule could be built
  if H is a hypothesis then add the new hypothesis $P(F(X))$
but this would have introduced the object $F(X)$ which may have nothing to do in the actual context. Moreover, this mechanism could be expansive by introducing $F(X)$, then $F(F(X))$, and so on (for example introducing unions of unions of unions, and so on, of sets),

Instead of this, the formula
  $H \rightarrow P(F(X))$
which is rewritten
  $H \rightarrow !Y{:}F(X), P(Y)$
leads to the condition
  $!Y{:}F(X)$ is a hypothesis
and to the action
  add the hypothesis $P(Y)$
that is to the rule
  if <conditions built from H >
    and $!Y{:}f(X)$ is a hypothesis
  then add the hypothesis $P(Y)$.

So the hypothesis $P(Y)$ is added only if $F(Y)$ already exists. And if $F(Y)$ is pertinent, it probably will be introduced by other rules. This sort of rule is efficient and, above all, it is not dangerous, so it could have a high priority. This explains why MUSCADET is efficient in proving theorems where many concepts F are defined by formulas of the form
  $\forall X(P(F(X)) \leftrightarrow ...)$,
more especially in proving theorems about unions, intersections, mappings, images, power sets, inverse images, etc.

But there are situations where these restrictions are too severe while building rules. This is the case if the functional symbol F does not correspond to a defined concept but to a term which appears only in axioms (for example the functional symbol "growth_rate" in axioms of MGT problems). In these situations, the sub-formulas of the form
  $H \rightarrow P(F(X))$
leads to the building of other rules in which there is not the condition
  $!Y{:}F(X)$ is a hypothesis
but in which there is the action
  add the hypothesis $\exists Y(F(X){:}Y \wedge P(Y))$. The hypothesis will be treated later, like other existential hypotheses.

Another feature of MUSCADET explains its efficiency when handling images, inverse images, as well as non-empty or disjoint sets.
From a sub-formula of the form

$H \rightarrow \exists X P(F(X)))$

where

$P(F(X))$

denotes a sub-formula involving the functional symbol F
(for example every element has an image or every element has an preimage, (see section 3.1))
the following rule might be built :

if <conditions built from H >
then create a new object
and add the hypotheses $F(X)$:<this new object>
and P(<this new object>

But this rule might be expansive. Instead the built rule is just

if <conditions built from H >
then add the hypothesis $\exists X P(X))$

So existential hypotheses are first stored without being treated. They are treated later, one by one, in the order in which they have been added (for example an image, then an preimage, then an image, etc, or if there are several mappings, images by each of them will be created successively).

This explain the successes of MUSCADET in proving theorems :

- SET751+4, proved only by MUSCADET in less than 0.01 second and by Vampire (Riazanov,Voronkov, 2002) in more than 100 seconds during CASC-20 (2005)
- HAL002+1, proved only by MUSCADET and E (Schulz, 2002) in less than 0.01 second and by Vampire in more than 50 seconds.

During the CASC-19 competition (2003), four theorems about mappings were proved only by MUSCADET and by no other provers : SET723+4, SET743+4, SET750+4, SET752+4. In theorem SET743+4, for example, there are three sets, five mappings, and MUSCADET creates 12 elements (6 necessary ones and 6 useless ones).

There is in the TPTP library another, more difficult, theorem about mappings (SET741+4) where there are three sets, nine mappings, injective or surjective, and to prove it, MUSCADET creates 31 elements (16 necessary ones and 15 useless ones).

We saw in section 2.3 the treatment of equality, combined with the elimination of functional symbols. In hypotheses, equalities occur only between constants (objects) and are quickly removed, so as one of the objects, by replacing everywhere one of the constants by the other. Consequently, there are no longer equalities in hypotheses. The equality between two terms is only implicit : they are named by the same constant. This simplification is successful.

A consequence of this fact is that an equality in a hypothesis must not appear in the condition of a rule. Such a rule will not be applied, except during the short time between the addition of this hypothesis and its removal ! So, the treatment of equalities in the building of rules must be specific.

As seen in section 3.4, the definitions of singleton and unordered-pair does not lead to the rules

if $\{A\}$:S and $X = A$ are hypotheses
then add the hypothesis $X \in S$

and

if $\{A, B\}$:P and X=A are hypotheses
then add the hypothesis $X \in P$

but to the rules

if $\{A\}$:S is a hypothesis
then add the hypothesis $A \in S$

and

if $\{A, B\}$:P is a hypothesis
then add the hypothesis $A \in P$

This is successful, not only for theorem SET703+4, which is relatively easy, but also for theorem SET707+4

$$\forall A\ \forall B(\{\{A\}, \{A, B\}\} = \{\{U\}, \{U, V\}\} \leftrightarrow A = U \wedge B = V)$$

the difficulty of which was emphasized by Brown (1986) in his paper about his system based on the fundamental deduction principle.

This is successful also for all theorems relying on a concept which includes, among other things, the assertion of the uniqueness of objects, for example mapping, injection or partition.

6 Related work

Although most theorem provers now use resolution, there are still some provers which use natural deduction and there are even two other provers, besides MUSCADET, which participated in CASC competitions, THINKER (Pelletier, 1998) in 1997 and Dilemna (Björk, 2003) in 2004.

In CASC-14 (1997) THINKER could not prove any theorem containing equalities. This is why, from then on, the organizers decided to split the FOF division into two categories, one category for problems with no equality and one category for problems with equality. Moreover, in THINKER, as described in (Pelletier, 1998), "arbitary functional symbols" are not represented, "the only terms are individual constants (0-place function symbols) and variables".

MUSCADET did not participate in the competition in 1997, but further experiments showed that it would have been better than THINKER for semantic problems containing functional symbols and/or equality (now category FEQ) and not as good as THINKER for syntactic problems expressed in a rather unnatural manner (now category FNE).

In CASC-J2 (2004) Dilemna proved two theorems in the category FEQ and nine theorems in the category FNE.

MUSCADET did not participate in the competition in 2004, but at that time it was able to prove fifteen theorems (not proved by Dilemna) among the FEQ theorems of the competions and only two (also proved by Dilemna) among the FNE theorems of the competition. It is now able (version 2.6) to prove twenty-five and three theorems of the respective categories.

Moreover one FEQ theorem (SET609+3) was already proved by MUSCADET in 2004 and was not proved by any of the entrant systems of that competition.

These results tend to show that the results of MUSCADET, that is, its much better results for FEQ [20] problems than for FNE problems, are not only due to its using natural deduction type methods and heuristics, but also to the fact that its strategies are implemented by rules in knowledge bases, and that it uses metarules to automatically build new rules adapted to the chosen representations of a "theorem to be proved", as described in this paper and illustrated in several examples. All this can be implemented only in a natural

[20] FEQ is presented as the category with equality, but its other characteristic is that it also contains many definitions of concepts which are used in other definitions and axioms, as well in the conjecture and that all these statements are expressed in a natural manner. FNE often contains few very big formulas or formulas expressed in a rather unnatural manner.

deduction context.

7 Conclusion

MUSCADET functions in a manner which is quite different from resolution-based provers. It uses methods based on natural deduction and is a knowledge-based system. We have seen some of these crucial methods and explained why MUSCADET is able to prove some theorems that resolution-based provers are not yet able to prove. We have also seen that, in cases where theorems are also proved by other provers, the MUSCADET proof can be obtained at least as fast as with the other provers and much faster in many cases. However, MUSCADET cannot prove some theorems that resolution-based provers can easily prove.

MUSCADET is efficient for everyday mathematical problems which are expressed in a natural manner, for example in naive set theory. It is not efficient for problems which are defined axiomatically, from a logician's point of view, for instance in the fields of axiomatic geometry or axiomatic set theory.

MUSCADET is efficient to solve problems which involve many axioms, definitions or lemmas. It is not efficient at all to solve problems which involve only one large conjecture and no intermediary definitions.

Improvements will not be due to the increase of computer speed. Most of the time, either MUSCADET succeeds or it fails and stops quickly. The role of heuristics is not to limit the number of deduced facts which generally increases in a linear manner. The role of heuristics is to enlarge the scope of situations that the system is able to handle efficiently. The analysis of the failures of MUSCADET will be pursued in order to help refine its heuristics.

Nevertheless there are two cases of time out. One is due to too many useless splittings, the other concerns problems that contain too many existential axioms, leading to too many objects being created. Until now MUSCADET does not backtrack, as it is more difficult to decide when to backtrack than to refine heuristics so that it chooses the right path. Two potential improvements would lead to better results : to implement backtracking and to improve heuristics to choose the right path, especially in the case of problems in axiomatic theories.

Another way to improve theorem proving is to have provers cooperate. Sutcliffe (2001) has already worked to make the best resolution-based provers cooperate, and the overall performance was better than the performance of each of the components.

It would certainly be possible to make MUSCADET cooperate with a resolution-based prover. Some provers may begin with a preprocessing step, such as attempting to split a problem into sub-problems before clausifying it (Schulz, 2002). MUSCADET could first analyse the given problem and choose between searching itself for a proof or calling on the resolution-based prover. In the event of failure on a sub-problem or after having exceeded a time limit, it could also call on the resolution-based prover for this sub-problem.

Acknowledgements

I would like to thank Geoff Sutcliffe and Christian Suttner for their interest in empirically successful automated reasoning, for having created the TPTP Problem Library and for updating it, and finally for organizing the CASC competitions.

References

Bledsoe W. W., Splitting and Reduction Heuristics in Automatic Theorem Proving, *Journal of Artificial Intelligence*, 2:55–77, 1971.

Bledsoe W. W., Boyer R. S., Henneman W. H., Compter Proofs of Limit Theorems, *Journal of Artificial Intelligence*, 3:27–60, 1972.

Bledsoe W. W., Bruell P., A Man-Machine Theorem-Proving System, *Journal of Artificial Intelligence*, 5:51-72, 1974.

Bledsoe W. W., Non-Resolution Theorem Proving, *Journal of Artificial Intelligence*, 9:1–35, 1977.

Björk M., A First Order Extension of Stalmarck's Method, PhD thesis, Department of Computing Science, Chalmers University of Technology, Gothenburg, Sweden, 2003

Brown F. M., An Experimental Logic based on the Fundamental Deduction Principle, *Journal of Artificial Intelligence*, 30:117–263, 1986

Bylinski C., Some Basic Properties of Sets, *Journal of Formalized Mathematics*, 1(1):i47-53 http://merak.pb.bialystok.pl/mirror/JFM

Bylinski C., Functions and Their Basic Properties, *Journal of Formalized Mathematics*, 1(1):55-65 http://merak.pb.bialystok.pl/mirror/JFM

McCune W., http://www.mcs.anl.gov/∼mccune/prover9, 2005

Mizar, http://www.mizar.org, 1973, 1989

Pastre D., MUSCADET: An Automatic Theorem Proving System using Knowledge and Metaknowledge in Mathematics, *Journal of Artificial Intelligence*, 38(3):257–318, 1989.

Pastre D., Automated Theorem Proving in Mathematics, *Annals on Artificial Intelligence and Mathematics*, 8(3-4):425–447, 1993

Pastre D., Muscadet version 2.3 : User's Manual, http://www.math-info.univ-paris5.fr/∼pastre/muscadet/manuel-en.ps, 16p, 2001a

Pastre D., Muscadet2.3 : A Knowledge-based Theorem Prover based on Natural Deduction, *International Joint Conference on Automated Reasoning - Conference on Automated Deduction*, 685–689, 2001b

Pastre D., Implementation of Knowledge Bases for Natural Deduction, *2nd International Workshop on Implementation of Logics, 8th International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, 49–68, 2001c

Pastre D., Strong and weak points of the MUSCADET theorem prover, *AI Communications*, 15(2-3):147-160, 2002, http://www.cs.miami.edu/∼tptp

Pelletier F. J., Automated Natural Deduction in THINKER, *Studia Logica*, 60:3-43, 1998

Pelletier, F.J., Sutcliffe, G., Suttner, C., The Developmeent of CASC, *AI Communications*, 15(2-3):79–90, 2002, http://www.cs.miami.edu/∼tptp/CASC

Riazanov A., Voronkov A, Splitting without backtracking, *17th Inernational Joint Conference on Artificial Intelligence, IJCAI 01*, 611-617, 2001

Riazanov A., Voronkov A, The Design and Implementation of Vampire, *AI Communications*, 15(2-3):91-110, 2002

Robinson J.A., A machine oriented logic based on the resolution principle, *J.ACM* 12:23-41, 1965

Schulz S., E: A Brainiac Theorem Prover, *AI Communications*, 15(2-3):111-126, 2002

Sutcliffe G., Suttner C.B., The TPTP Problem Library: CNF Release v1.2.1, *Journal of Automated Reasoning* 21(2):177–203,1998, http://www.cs.miami.edu/∼tptp

Sutcliffe G., The Design and Implementation of a Compositional Competition-Cooperation Parallel ATP System, *2nd International Workshop on Implementation of Logics, 8th International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, 92–102, 2001

Sutcliffe, G., Suttner, C., The State of CASC, *AI Communications*, 19(1):35–48, 2006, http://www.cs.miami.edu/∼tptp/CASC

Trybulec A., Tarski Grothendiek Set Theory *Journal of Formalized Mathematics*, 1(1):9-11, 1989, http://merak.pb.bialystok.pl/mirror/JFM

Urban J., Translating Mizar for First Order Theorem Provers, *MKM, Lecture Notes in Computer Science*, 2594:203-215, Springer, 2003