Accepting/Rejecting Propositions From Accepted/Rejected Propositions: A Unifying Overview

Ricardo Caferra,* Nicolas Peltier†
Leibniz-IMAG Grenoble, France

Looking at inference as a way of transforming information so as to make it more easily usable (or interpretable) allows to consider *accepted* and *rejected* propositions as equally relevant and naturally gives a *bipolar* view of reasoning. The four possibilities of transforming information from accepted or rejected propositions into accepted or rejected ones are analyzed and examples illustrating them are given. This analysis is not only interesting per se but can also be useful in increasing capabilities of existing theorem provers. A unified framework based on former work by the authors is extended by incorporating the idea of theory-anti-subsumption related to Plotkin's generalization. Working on some technical details of this framework should allow automated reasoning tools to deal with different ways of connecting accepted and rejected propositions. © 2008 Wiley Periodicals, Inc.

1. INTRODUCTION

In this work, we look at inference as a way of making information more easily usable (in general with respect to a given goal).

This point of view captures a large variety of inferences, for example, when proving theorems, information *already contained* in the premises is made more and more explicit in the form of steps of a proof. In inductive logic, generalizations allow to express in compact form a lot of particular cases that would have been intractable otherwise. Other kinds of inference fit this view as well (e.g., statistical inference).

Consequently, bipolarity is studied here by analyzing the different ways of relating *accepted* and *rejected* propositions and entailment[a] relationships between them. Accepted and rejected propositions are near (but conceptually different) to what is frequently called *positive* and *negative* information. If a syntactic, deductive

*Author to whom all correspondence should be addressed: e-mail: Ricardo.Caferra@imag.fr.
†e-mail: Nicolas.Peltier@imag.fr.

[a]"Entailment" is used here with its intuitive meaning. It covers, for example, the formal definition in Refs. 11 and 12, but it is not intended to cope with problems of material, strict and relevant conditionals (see Ref. 13). This is only a terminological warning and we can get rid of it.

INTERNATIONAL JOURNAL OF INTELLIGENT SYSTEMS, VOL. 23, 999–1020 (2008)
 Published online in Wiley InterScience (www.interscience.wiley.com). • DOI 10.1002/int.20304

approach is emphasized, positive (negative) information corresponds to the premises. If a semantic view is adopted, positive and negative information denote precisely the properties (relations) of (between) objects that hold (for positive information) and do not hold (for negative information) in the intended interpretation. In general, accepted propositions will correspond to relationships that hold, but rejecting proposition is different from prefixing them with a negation connective (though rejection is intuitively, at least, obviously related to negation): relations that hold can be rejected for different reasons (known to be useless, in jurisprudence, . . .)

Our goal is to investigate *information extraction* about relationship that holds (or does not hold) from other relationship that holds (or does not hold). Connections between these different entailments are also analyzed.

Our thesis is that studying these different ways of entailment is not only interesting per se, but enables as a consequence to increase the capabilities of present theorem provers. The authors did elsewhere some work confirming this thesis (see e.g., Refs. 1–8).

Such a unified approach leads naturally to a use of abduction that can be fruitful, for example, in discarding lemmata or in choosing strategies, major issues in theorem proving. In this aim, we introduce the theory-anti-subsumption (τ-anti-subsumption), something similar to Plotkin's generalization.[9,10]

To the best of our knowledge, no other proposals in this direction have been published.

We consider the four following possibilities (terminology corresponding to rejection and negation identification is also given):

1.	accepted $\longrightarrow$ accepted	*info* $+$ $\longrightarrow$ *info* $+$
2.	accepted $\longrightarrow$ rejected	*info* $+$ $\longrightarrow$ *info* $-$
3.	rejected $\longrightarrow$ rejected	*info* $-$ $\longrightarrow$ *info* $-$
4.	rejected $\longrightarrow$ accepted	*info* $-$ $\longrightarrow$ *info* $+$

At this point, a natural question arises: *Why do not simply consider positive and negative information and* ***consequence relations*** *between them?*

The answer is as natural as the question: because $\mathcal{B}$ logically follows from $\mathcal{A}$ (independently of the fact that $\mathcal{A}$ and $\mathcal{B}$ be in positive or negative form) iff the relation denoted by $\mathcal{B}$ holds in all the interpretations in which hold those denoted by $\mathcal{A}$, or if $\mathcal{B}$ can be deduced in a proof system from $\mathcal{A}$ (see eg, Refs. 14 and 15). But what about *other* kinds of relationships between $\mathcal{A}$ and $\mathcal{B}$?[b]

Abduction provides a convincing example of relationship other than logical consequence: in order to accept a hypothesis H as an explanation of a set of facts E in the context of a theory (already accepted knowledge, . . .) K, it is required that $K \cup H \models E$ **and** that $K \not\models H$. $\not\models$ cannot be simulated by $\models$ in undecidable theories.

The examples illustrating 1. to 4. have been chosen in order to put bridges between these different entailments. A big part of what can be considered new in this work is strongly related to (a rather constructive view of) negation.

[b]Of course, this question does not exclude the *use* of the notion of logical consequence in defining these different ways of relating $\mathcal{A}$ and $\mathcal{B}$ (see section 4).

Case 1 is obviously the best known (deduction) and we will illustrate it by using deduction in a rather nonstandard way (that underlines bipolarity), by giving an example of *n-validity* (ie, formulae valid only in universes of cardinality less or equal to *n*).

Case 2 corresponds to what is called *dis-inference*, that is, characterizing formulae that surely *cannot be deduced* from other formulae. A similar notion (but without elaborating a calculus) has been studied by H. B. Curry. He called it *nondemonstrability*[16,c] or *invalidity*.[17]

It is, together with 3 below, almost unknown in comparison with 1.

Case 3 corresponds to the so-called *rejection rules* which allow to demolish (destroy, reject) propositions on the ground of already rejected ones. Rejection is particularly important in experimental sciences. Rejection was extensively studied (via the notion of *anticonsequence*) by the Polish school (Łukasiewicz, Tarski, and their followers). Independently, the similar notion that of *refutability* has been developed. It consists in adding *counteraxioms* to the axioms of a (standard) formal system. It should be pointed out that rejection (refutability) *cannot* be simulated simply by negation (of formulae to be rejected) and contraposition (of inference rules), see section 4. We shall show below that anticonsequence fits to a definition of negation given by D. Gabbay in.[18] Astonishingly, this relationship does not seem to have been remarked (not even by D. Gabbay). Examples clarify this idea, particularly from pure equality first-order logic.

In case 4, rejected solutions for a problem (constraints) are used in order to define the accepted ones. They can be also used in order to guide (generally in an efficient way) the search of solutions. It can be qualified of a kind of Sherlock Holmes' approach[d]: "*When you have eliminated the impossible, whatever remains, however improbable, must be the truth*." Of course, this is possible in a safe way only in closed (or fully specifiable) contexts.

We use freely notions that have became almost "common knowledge" in Artificial Intelligence and Automated Deduction such as substitution, unification, Herbrand universe, literal, clause, resolution rule, clausal logic, etc. For formal details, the interested reader can consult Ref. 8. For the notions introduced by the authors elsewhere, the corresponding definitions are briefly recalled.

2. DEDUCTION OR: FROM ACCEPTED PROPOSITIONS TO ACCEPTED PROPOSITIONS

The example below shows how methods normally used in deduction (here the well-known tableaux method) can illustrate the fact that accepted and rejected propositions are, in general, intermingled (i.e., a "bipolarity view"). It is a finer view of the trivial fact that in general, arbitrary formulae are not valid (or equivalently, contradictory).

[c]Of course, though *nondemonstrability* is correct, one would have termed it, with modern terminolgy, *unprovability*. We respect in this work Curry's choice.

[d]Or, more pedantic, corresponds to apophatic (*apophatique* in French) definitions, that is, entities defined by saying what these entities *are not*.

Let us consider the question whether $\models \mathcal{A} \Rightarrow \forall x\, P(x)$ with $\mathcal{A}$ an instance of Leibniz's law stating that x, y, z are three *different* objects with property P.

$$\begin{aligned}\mathcal{A}: \quad & [\exists x \exists y \exists z (P(x) \wedge P(y) \wedge P(z)) \\ & \wedge (Q(x) \Leftrightarrow \neg Q(y)) \\ & \wedge (R(x) \Leftrightarrow \neg R(z)) \\ & \wedge (S(y) \Leftrightarrow \neg S(z))]\end{aligned}$$

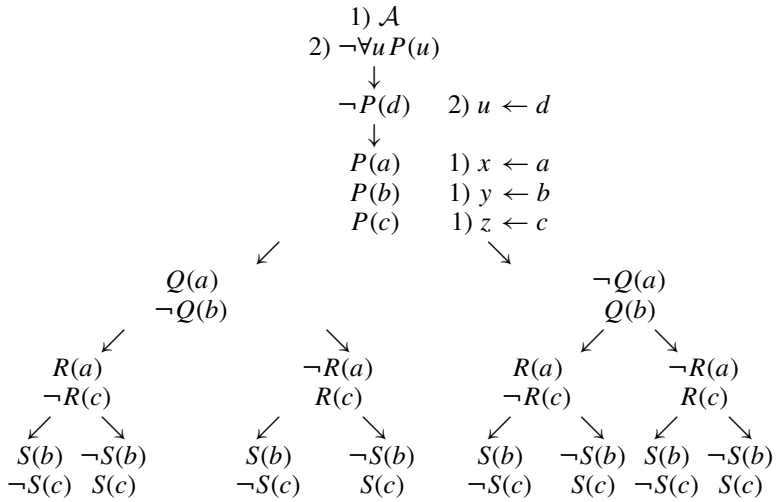
The (correct) answer is "no." But this example has been chosen to illustrate our view on inference. *More information* can be extracted from the inference process and added to the answer "no."

This implication is a three-valid formula (a formula is *n-valid*, $n \in \mathbb{N}$, iff it is valid in universes D, s.t. $card(D) \leq n$ and can be falsified in universes D s.t. $card(D) > n$).

An easy modification of the tableaux method (see e.g., Refs. 19 and 20) allows to obtain the more informative answer we are looking for.

In order to keep soundness, tableaux method requires to instanciate different existential quantifiers with fresh constants (i.e., constants not appearing in the branch being expanded). A (simple) metareasoning mechanism allows to close (under certain cardinalities constraints) otherwise open branches.

A theorem prover incorporating such a feature would clearly improve *qualitatively* the existing ones.



card(D) = 1

$a = b = c = d$ contradiction branch 1.

card(D) = 2

$a = b = c$ contradiction branches 1.1 and 1.2
$a = b = d$ contradiction branch 1.

$a = c = d$ contradiction branch 1.
$b = c = d$ contradiction branch 1.

card(D) = 3

$a = b$ contradiction branches 1.1 et 1.2
$a = c$ contradiction branches 1.1.1; 1.1.2; 1.2.1 and 1.2.2
$a = d$ contradiction branch 1.
$b = c$ contradiction branches 1.1.1.1; 1.1.1.2; 1.1.2.1; 1.1.2.2; 1.2.1.1; 1.2.1.2; 1.2.2.1 and 1.2.2.2
$b = d$ contradiction branch 1.
$c = d$ contradiction branch 1.

In universes D with **card(D)** $\geq$ **4**, no branch can be closed.

3. DIS-INFERENCE OR: FROM ACCEPTED PROPOSITIONS TO REJECTED PROPOSITIONS

The following method dealing with this problem has been introduced and developed elsewhere (see e.g., Refs. 2, 3, 5, 7, and 9). It has been applied to model building, semantic guiding of theorem provers (semantic resolution), logic program verification, ... The addressed problem is strongly related to the very deep one of *negation* (see e.g., Refs. 16 and 18) in the following sense. To say that a statement is false is to say that it is not true. Now, if we take a constructive point of view, this amounts to say that it is not provable. Otherwise stated, a statement is false iff no proof of it exists. Of course, it is not always possible to decide whether this requirement is fulfilled (there exist undecidable theories).[e]

A set of rules has been proposed by the authors in the aforementioned works (for details, see e.g., Ref. 8) in order to infer statements that *surely* are not derivable from others in a particular calculus. Naturally, we have called these rules *dis-inference* rules.

The key technical idea allowing to get this goal is simple: conditions called *constraints* are set in order to prevent the application of the inference rules of the calculus. The constraints are pure equational formulae interpreted in the algebra of finite trees: "=" denotes *syntactic* (and not semantic) equality, that is, the equation $s = t$ (s and t first-order terms) holds iff s is (syntactically) identical to t. It is known that the satisfiability problem is decidable for such constraints.[21,22] Any decidable fragment, if useful, could have been sensibly used as constraints.

These conditions (e.g., $x \neq a \wedge \forall x.x \neq f(b, c)$) restrict the domain of the variables in the formula to which the constraint is associated. The potential variable values eliminated by the constraints are precisely those allowing application of the inference rules. When the considered formulae are clauses (i.e., disjunctions of

[e]From an historical point of view is worth mentioning that some authors have given the same importance to provable and refutable formulae. A good example are the so-called *representation systems* by R. Smullyan.[23] Roughly speaking these systems take into account not only theorems (as formal systems) but also refutable or contra-valid (Smullyan) sentences.

atomic formulae or negation of atomic formulae with all the variables appearing in them universally quantified) what is obtained are called *c-clauses*.

The method searches for refutation of a set of clauses and simultaneously for models (similarly to the standard use of tableaux, to which the approach has also been applied–see Refs. 6 and 8). It allows to build models for some classes of set of clauses without missing refutational completeness.

We recall here some of its more representative rules in order to give a taste of the approach ($\bar{s}, \bar{t}, \ldots$ denote, as usual, tuples of terms, *var*(*exp*) denotes the set of variables in expression *exp* and c'_2 a clause). Obviously, a unit c-clause is a c-clause containing only one atomic (or negated atomic) formula, that is, a literal.

For each of the inference rules used by the resolution method (and consequently by the resolution-based theorem provers), we have defined a corresponding *dis-inference* rule. We stick to the standard resolution terminology by adding 'c' for "constraint" and 'dis' for "dis-inference", for example, *bc-disresolution* correspond to binary resolution; *unit-bc-disresolution* to unit binary resolution; *bc-dis-resolvent* to binary resolvent, etc.

Syntax is self-explaining.

DEFINITION 1 (bc-disresolution). *Let*
$c_1 : [\![\neg P(\bar{t}) : \mathcal{X}]\!]$ *be a unit c-clause*
and
$c_2 : [\![P(\bar{s}) \vee c'_2(\bar{y}) : \mathcal{Y}]\!]$ *a c-clause*:
The rule
unit bc-disresolution (or bc-disresolution for short, bc for ***b****inary* ***c****onstrained) on* c_2 *with* c_1 *and* $P(\bar{s})$ *is defined as (* $\bar{x} = var(\mathcal{X}) \cup var(\neg P(\bar{t}))$*)*:

$$\frac{[\![\neg P(\bar{t}) : \mathcal{X}]\!] \qquad [\![P(\bar{s}) \vee c'_2(\bar{y}) : \mathcal{Y}]\!]}{[\![P(\bar{s}) \vee c'_2(\bar{y}) : \mathcal{Y} \wedge \forall \bar{x}.[\neg \mathcal{X} \vee \bar{s} \neq \bar{t}]]\!]}$$

■

Example 1.

$$c_1 : [\![P(f(x)) : \top]\!]$$

$$c_2 : [\![\neg P(y) \vee Z(y) : \top]\!]$$

($\top$ is used to specify that the variables in the clause are **un**constrained)

The c-clause $c_3 : [\![\neg P(y) \vee Z(y) : \forall x.f(x) \neq y]\!]$ is the *unit c-dis-resolvent of* c_1 *and* c_2.

The c-clause c_3 can be reduced using simplification rules (on the signature $\Sigma = \{b, f\}$) to

$$c_3 : [\![\neg P(y) \vee Z(y) : y = b]\!]$$

or equivalently

$$c_3 : [\![\neg P(b) \vee Z(b) : \top]\!]$$

The c-clause c_2 is equivalent to the conjunction of c_3 and the c-clause $Z(f(x))$ obtained by c-resolution (i.e. standard resolution applied to c-clauses) between c_1 and c_2. ■

Given two clauses C and D, we say that C *subsumes* D iff there exists a substitution σ such that $\sigma C \subseteq D$.

If clauses are considered as disjunctions (instead of sets) of literals, the definition becomes:

Given two clauses C and D, we say that C *subsumes* D iff $card(C) \leq card(D)$ and there exists a substitution σ such that σC is a subclause (with the obvious meaning) of D.

DEFINITION 2 (unit bc-dissubsumption). *The* unit bc-dissubsumption *rule puts constraints avoiding a c-clause c_1 to be subsumed by a unit c-clause c_2. It is formally defined as follows ($\bar{x} = var(\mathcal{X}) \cup var(P(\bar{t}))$):*

$$\frac{c_1 : [\![P(\bar{t}) \vee c' : \mathcal{Y}]\!] \qquad c_2 : [\![P(\bar{s}) : \mathcal{X}]\!]}{c_3 : [\![P(\bar{t}) \vee c' : \mathcal{Y} \wedge \forall \bar{x}.[\neg \mathcal{X} \vee \bar{s} \neq \bar{t}]]\!]}$$

■

Example 2. Let S be the set of c-clauses:

$$c_1 : [\![P(f(x)) : \top]\!]$$
$$c_2 : [\![R(a) : \top]\!]$$
$$c_3 : [\![P(x) \vee R(x) : \top]\!]$$
$$c_4 : [\![\neg P(x) \vee \neg R(x) \vee R(a) : \top]\!]$$

By bc-dissubsumption (with c_2), c_3 is reduced to (c_3 is deleted):

$$c_5 : [\![P(x) \vee R(x) : x \neq a]\!]$$

By bc-dissubsumption between c_1 and c_5, we obtain (c_5 is deleted):

$$c_6 : [\![P(x) \vee R(x) : x \neq a \wedge \forall z.x \neq f(z)]\!]$$

The formula $x \neq a \wedge \forall z.x \neq f(z)$ does not have a solution on the signature $\Sigma = \{a, f\}$. All the Herbrand's term on the signature Σ are of the form either a or $f(z)$. It is therefore impossible to satisfy on this universe the constraints imposed on x, the formula is then reduced to $\perp$. c_6 is therefore always true and can be deleted from S. S is reduced to : $\{c_1, c_2, c_4\}$.

This simplification is not possible using the standard subsumption rule. ■

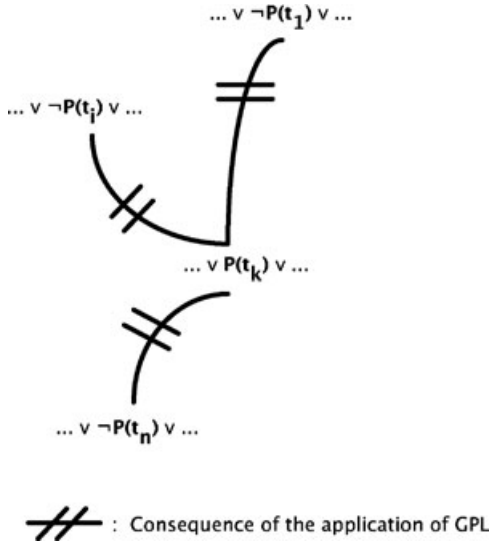
Maybe the most representative rule of the method is the one producing pure literals.

A literal L is said to be *pure* in a set of c-clauses S iff its complementary (i.e., $\neg L$) does not appear in clauses of S. Our method exploits in one of its key rules

the notion of pure literal in a set of clauses. It is clear that a pure literal in a set of clauses is a model of the clause to which it belongs (it can be evaluated to true independently of the rest of the interpretation). Therefore, a natural idea in order to build a model of a set of clauses S is to try to produce for each clause a pure literal in it. The way our method does this is by setting conditions (coded in the constraints), restricting the domain of variables in the arguments of a literal $P(\bar{t})$ in order to avoid unification with the arguments of a literal $\neg P(\bar{t})$. This generated literal is added to S. The pure literal thus generated is not a logical consequence of S, but consistent with S: if L is pure in S then $S \cup L$ is satisfiable iff S is satisfiable.

The idea underlying this rule can be more easily grasped by considering the "connexion graph" of a set of clauses. The edges of the graph connect complementary atomic (negated atomic) formulae corresponding to potential applications of binary resolution.

What GPL does is to "break" these links, by putting constraints avoiding application of the resolution rule, thus transforming $P(t_k)$ in a pure literal.



DEFINITION 3 (GPL rule). *Let S be a set of c-clauses and $c : [\![P(\bar{t}) \vee c' : \mathcal{X}]\!]$ be a c-clause in S.*

The GPL rule *(**G**enerating **P**ure **L**iterals) is defined as follows:*

$$\frac{[\![P(\bar{t}) \vee c' : \mathcal{X}]\!] \qquad S}{[\![P(\bar{t}) : \mathcal{X}_{\mathsf{pure}}]\!]}$$

where $\mathcal{X}_{\mathsf{pure}} = \bigwedge_{[\![\neg \mathrm{P}(\bar{\mathrm{s}}) \vee \mathrm{r} : \mathcal{Y}]\!] \in S} (\forall \bar{\mathrm{y}}. \neg \mathcal{Y} \vee \bar{\mathrm{s}} \neq \bar{\mathrm{t}}) \wedge \mathcal{X}$ *and where* $\bar{\mathrm{y}}$ *are all the variables in* $var([\![\neg \mathrm{P}(\bar{\mathrm{s}}) \vee \mathrm{r} : \mathcal{Y}]\!])$ ■

The following example shows how the method works (for details, see Ref. 5).

Example 3. Let S be the following set of c-clauses (a, b, and c denote constants):

$$c_1 : [\![P(x) \vee R(x) : \top]\!]$$

$$c_2 : [\![\neg P(a) \vee \neg P(b) : \top]\!]$$

$$c_3 : [\![\neg R(c) : \top]\!]$$

The literal $P(x)$ is not pure in S, since there exists in a clause of S (c_2) complementary literals ($\neg P(a)$ and $\neg P(b)$).

But by adding to the literal $P(x)$ the constraints preventing the application of the resolution rule between $P(x)$ and $\neg P(a)$ (resp. $\neg P(b)$), we get the unit c-clause:

$$[\![P(x) : x \neq a \wedge x \neq b]\!].$$

This pure literal in S is *added* to S (c_1 is conserved in order to keep refutational completeness). ■

The dis-rules are called *model contruction rules*, or *mc-rules* for short, and the (constrained) standard refutation rules used by the resolution method *refutation rules* or *r-rules* for short.

$\square\!\!\!\square$ denotes the constrained empty clause (i.e., two complementary literals with nonempty intersection of variables domains).

The following nondeterministic algorithm search for refutations and models of a set of c-clauses.

Procedure RAMC1:
% **R**efutation **A**nd **M**odel **C**onstruction – non deterministic version %
INPUT:
A finite set of c-clauses S
OUTPUT:
UNSATISFIABLE **or** SATISFIABLE **or** (SATISFIABLE & A Herbrand model).
begin
  repeat
    choose an mc-rule or an r-rule ρ;
    modify S according to ρ;
  until $\square\!\!\!\square \in S$ **or** no inference rule modifies S
  if $\square\!\!\!\square \in S$
  then return(UNSATISFIABLE)
  else if S is a set of unit c-clauses
    then return(SATISFIABLE & S)
    else return(SATISFIABLE)
end.

4. ANTICONSEQUENCE OR: FROM REJECTED PROPOSITIONS TO REJECTED PROPOSITIONS

The concept of explanation in empirical sciences was mentioned in the section 1 as a convincing example of the relevance of the notion to be treated in the present section.

Social rules rejecting some unacceptable (from a social point of view) behaviors offer another good example.

Obviously, a particular behavior entailing a rejected one must be (at least rationally) rejected as well.

We start by recalling the well-known *consequence relation* (Tarski) $\mathcal{C}_n$ that verifies the following conditions (see e.g., Ref. 12):

X: set of wff (well formed formulae) of a formal language
$(T1)$ $X \subseteq \mathcal{C}_n(X)$
$(T2)$ if $X \subseteq Y$ then $\mathcal{C}_n(X) \subseteq \mathcal{C}_n(Y)$
$(T3)$ $\mathcal{C}_n(\mathcal{C}_n(X)) \subseteq \mathcal{C}_n(X)$
(PF) $\mathcal{C}_n(X) = \bigcup\{\mathcal{C}_n(Y) \mid Y \subseteq X \ ; Y : \textit{finite}\}$

4.1. Anticonsequence

Rejecting propositions on the ground of other rejected propositions comes back to Aristote and has been deeply studied by Łukasiewicz and the Polish school.

X: denotes a set of rejected propositions.
$\mathcal{C}_n^{-1}(X)$: denotes the set of propositions rejected on the basis of X
A proposition y is rejected on the basis of X iff there is a proposition in X consequence of y, in symbols:

$y \in \mathcal{C}_n^{-1}(X)$ iff $\exists x \in X.x \in \mathcal{C}_n\{y\}$
Similarly to Tarski's axioms, it can be proven:
$(T^{-1}1)$ $X \subseteq \mathcal{C}_n^{-1}(X)$
$(T^{-1}2)$ if $X \subseteq Y$ then $\mathcal{C}_n^{-1}(X) \subseteq \mathcal{C}_n^{-1}(Y)$
$(T^{-1}3)$ $\mathcal{C}_n^{-1}(\mathcal{C}_n^{-1}(X)) \subseteq \mathcal{C}_n^{-1}(X)$
(PF^{-1}) $\mathcal{C}_n^{-1}(X) = \bigcup\{x \in \mathcal{C}_n^{-1}(Y) \mid Y \subseteq X \ ; Y : \textit{finite}\}$

A short and very good presentation of accepting/rejecting rules is given in Ref. 12 (see also Ref. 24): a user of a language can observe rejecting (sentences) rules, besides accepting (sentences) rules. In the standard semantics of **true** and **false**, accepted sentences are recognized as being **true** and rejected one as **not being true**, that is, either false (bivalent semantics) or **no truth-value** if other truth values are accepted.

Correspondingly, for such a language, there are two distinguished consequence relations: truth preserving and falsity preserving ones.

Ref. 25 contains a short, clear, and exhaustive (to the best of the author's knowledge) synthesis on the problem of rejection and refutation systems (i.e., their associated deductive systems).

As an example, we give a system for *non-theorems* of CPC (Classical Propositional Calculus) borrowed to Ref. 26.[f]

Example 4. **Axioms**

$(A1)$ $P \Rightarrow \neg P$ (P: atomic formula)

$(A2)$ $\neg P \Rightarrow P$ (P: atomic formula)

Inference Rules

$(R1)$

a)

$$\frac{\alpha}{P \Rightarrow \alpha}$$

b)

$$\frac{\alpha}{\neg P \Rightarrow \alpha}$$

P atomic, $P \notin varprop(\alpha)$ ($varprop(\alpha)$: set of propositional variables in the formula α)

$(R2)$

$$\frac{\alpha \Rightarrow \beta}{\alpha \Rightarrow (\alpha \Rightarrow \beta)}$$

$(R3)$

$$\frac{\alpha \Rightarrow \beta}{(\gamma \Rightarrow \alpha) \Rightarrow \beta}$$

$(R4)$

$$\frac{\neg\alpha \Rightarrow \beta}{(\alpha \Rightarrow \gamma) \Rightarrow \beta}$$

$(R5)$

$$\frac{\neg\alpha \Rightarrow \beta}{\alpha}$$

⋮

There are eight inference rules. Only three of them are needed in the example.

Remark 1. A substitution rule cannot be allowed since, many non-tautologies become tautologies through substitution. Axioms cannot be replaced by schemata.

[f]Systems for nonclassical logics have been also proposed. In Refs. 25 and 27, for example, refutation systems are proposed for some propositional modal logic.

Example 5. $\not\vdash (P \Rightarrow Q) \Rightarrow (Q \Rightarrow P)$

1. $\neg P \Rightarrow P$ (A2)
2. P (R5)
3. $Q \Rightarrow P$ (R1)
4. $Q \Rightarrow (Q \Rightarrow P)$ (R2)
5. $(P \Rightarrow Q) \Rightarrow (Q \Rightarrow P)$ (R3)

■

As an answer to the objection: *"Notion of rejection is inessential in systems with quantifiers because of instead of rejecting closed propositions one could assert its negation,"* the following example is proposed in Ref. 28.

In intuitionistic logic, (1) and (2) below hold

(1) $\neg(P \vee Q) \Rightarrow \neg P \wedge \neg Q$

(2) $P \wedge \neg P \Rightarrow Q$

but excluded middle:

(3) $P \vee \neg P$ (for all P)

does not.

Now, by substitution in (3), we can obtain:

(4) $\forall x\, P(x) \vee \neg \forall x\, P(x)$

and, if as suggested in the objection, in order to reject (3), we add to the logical system the negation of one instance of it (i.e., a negation of (4)), we obtain

(5) $\neg(\forall x\, P(x) \vee \neg \forall x\, P(x))$

That is to say, by using (1) and modus ponens

(6) $\neg \forall x\, P(x) \wedge \neg(\neg \forall x\, P(x))$

that is, according to (2) (and modus ponens) Q is theorem. Otherwise stated, arbitrary formulae are theorems. The proposed alternative way to treat rejection switchs a consistent system to an inconsistent one.

On the other hand, there exists propositions (e.g., in empirical sciences), the truth or falsehood, one does not pronounce on. They can be rejected *without asserting their negation*. ■

4.2. Anticonsequence and Some Treatments of Negation

Two treatments of negation interest us here because of their relationship with anticonsequence.

4.2.1. Non-Demonstrability (Unprovability) and Refutability

The chapter 6 of Ref. 16 is devoted to the study of negation.

After asking the question "What is meant by the negation of an elementary statement of a formal system?," Curry studies five forms of negation.

The so-called *nondemonstrability* and *refutability* are worth recalling here.

A statement is false just when no proof exists. This approach is similar to the one grounding dis-inference rules (see section 3). This point of view is acceptable

from a nonconstructive standpoint, but it is not acceptable from a constructive one. To establish constructively, the falsity of a statement amounts to give an effective process showing that the given statement does not have the structural characteristic of a theorem.

A point to be remarked is that demonstrability is a *monotonic* relation (see section 4).

This is not the case for nondemonstrability (i.e., in adding premisses, a nondemonstrable statement may become demonstrable).

The notion of refutability (according to Curry, this term has been borrowed to Carnap) consists in adding to the axioms the so-called *counteraxioms*.

If an elementary statement B is refutable on the basis of counteraxioms and is deducible from the elementary statement A (using deductive rules of the system) then A is refutable.

One can easily see that this approach correspond to Lukasiewicz's one (see section 4.1). It has been developed independently by Curry.

When interpreting negation as refutability it *becomes monotonic* (see T^{-1} axioms section 4.1).

Example 6. In order to illustrate refutability, Curry considers in Ref. 16 a rudimentary system for number theory $(< \mathbb{N}, \{s\}, \{=\} >)$ and the logical system (extended with a counteraxiom):

Axiom: $0 = 0$

Inference Rules:

$(R1)$

$$\frac{x = y}{s(x) = s(y)}$$

$(R2)$

$$\frac{x = y}{y = x}$$

$(R3)$

$$\frac{x = y \quad y = z}{x = z}$$

The theorems of this theory are all the elementary statements of the form $x = x$

Counteraxiom: $4 = 6$[g]

We can show in this system that e.g. $1 = 2$ is *refutable*:

1. $1 = 2$
2. $2 = 3$ 1. and $(R1)$
3. $1 = 3$ 1., 2. and $(R3)$

[g]Obviously, s denotes 'successor' and 1, 2, 3, ... denote $s(0)$, $s(s(0))$, $s(s(s(0)))$, ...

4. $2 = 4$ 3. and $(R1)$
5. $3 = 5$ 4. and $(R1)$
6. $4 = 6$ 5. and $(R1)$ Counteraxiom

and that e.g. $4 = 5$ is also *refutable*:

1. $4 = 5$
2. $5 = 6$ 1. and $(R1)$
3. $4 = 6$ 1., 2. and $(R3)$ Counteraxiom

One of the main interest of this example is that it shows that in this system, refutability **cannot** be simulated by deduction and contraposition,[h] for example, $4 \neq 5$ cannot be deduced in adding $4 \neq 6$, eliminating the counteraxiom $4 = 6$ and changing the inference rules to:

$(R1_c)$

$$\frac{s(x) \neq s(y)}{x \neq y}$$

$(R2_c)$

$$\frac{y \neq x}{x \neq y}$$

$(R3'_c)$

$$\frac{x \neq z \quad x = y}{y \neq z}$$

$(R3''_c)$

$$\frac{x \neq z \quad y = z}{x \neq y}$$

4.2.2. *Negation as Syntactical Inconsistency*

In the study by Gabbay,[18] two definitions are given of what a negation in a logical system is.

Gabbay considers a logical system defined by a *provability relation* $\vdash$ corresponding to the Scott's consequence relation (see Refs. 11 and 29) defined by the three properties below:

A, B denote wff (well-formed formulae).
$\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ denote finite sets of wff.
$\mathcal{A}, \mathcal{B} : \mathcal{A} \cup \mathcal{B}$
$\mathcal{A}, B : \mathcal{A} \cup \{B\}$

[h]Refutability is indicated by putting $\neq$ in the place of $=$

$\mathcal{A} \vdash \mathcal{B}$ if $\mathcal{A} \cap \mathcal{B} \neq \emptyset$ (reflexivity)
if $\mathcal{A} \vdash \mathcal{B}$ then $\mathcal{A}, \mathcal{A}' \vdash \mathcal{B}, \mathcal{B}'$ (monotony)
if $\mathcal{A} \vdash B, \mathcal{C}$ and $\mathcal{A}, B \vdash \mathcal{C}$ then $\mathcal{A} \vdash \mathcal{C}$ (transitivity or cut)

Scott's $\vdash$ is equivalent to Tarski's C_n (see Refs. 11 and 12). Relations $\vdash$ and C_n are called *sequential consequence relation* and *inferential consequence relation*, respectively.[12]

The Gabbay's definition we are interested here is grounded on the intuition that A and $\neg A$[i] are wff mutually exclusive. Therefore, a fixed set of *unwanted* wffs $\theta^{\neg} \neq \emptyset$ must be specified and it is said that for any A, A *is negated by* Δ, according to the following definition:

$\Delta \vdash \neg A$ iff $\exists y \in \theta^{\neg}$ such that $\Delta, A \vdash y$

The ideas underlying Gabbay's definition of negation and that of anticonsequence are the same. If a proposition Q is rejected and Q is entailed by another proposition P, then P must be rejected as well. Otherwise (by definition of entailment), Q should be accepted, contradicting the assumed premise.

5. CONSTRAINTS OR: FROM REJECTED PROPOSITIONS TO ACCEPTED PROPOSITIONS

Example 7. The old and well-known $n-$queens problem (see e.g., Refs. 30 and 31), that is, to place n queens in a $n \times n$ chessboard in such a way that they cannot attack each other if the standard chess rules are respected, can be used to illustrate the entailment from rejected to accepted propositions.

We give a solution of the four-queens as obtained by a constraint solver (or a constraint programming language).

The (closed) world:

```
1<= L1 <=4, 1<= L2 <=4, 1<= L3 <=4, 1<= L4 <=4,
1<= C1 <=4, 1<= C2 <=4, 1<= C3 <=4, 1<= C4 <=4,
```

The rejected positions (propositions):

```
\% not in the same line:

L1#L2, L1#L3, L1#L4, L2#L3, L2#L4, L3#L4,

\% not in the same column:

C1#C2, C1#C3, C1#C4, C2#C3, C2#C4, C3#C4,

\% not in the same (\) diagonals:
```

[i]We use here '$\neg$' to denote negation but Gabbay uses a more neutral one: "$*$."

```
L1-C1#L2-C2, L1-C1#L3-C3,
L1-C1#L4-C4, L2-C2#L3-C3, L2-C2#L4-C4,
L3-C3#L4-C4,

\% not in the same (/) diagonals:

L1+C1#L2+C2,
L1+C1#L3+C3, L1+C1#L4+C4, L2+C2#L3+C3,
L2+C2#L4+C4, L3+C3#L4+C4 ;
```

The accepted positions (propositions):

```
{ L1 = 4, C1 = 3, L2 = 1, C2 = 2, L3 = 3, C3 = 1,
  L4 = 2, C4 = 4 }

{ L1 = 4, C1 = 3, L2 = 1, C2 = 2, L3 = 2, C3 = 4,
  L4 = 3, C4 = 1 }

{ L1 = 4, C1 = 2, L2 = 3, C2 = 4, L3 = 2, C3 = 1,
  L4 = 1, C4 = 3 }

{ L1 = 4, C1 = 2, L2 = 2, C2 = 1, L3 = 3, C3 = 4,
  L4 = 1, C4 = 3 }
```

...

The first solution, for example, can be visualized:

	$Q2$		
			$Q4$
$Q3$			
		$Q1$	

■

Example 8.

The four-queens (n-queens in general) problem could have been specified using positive instead of negative information.

The (closed) world:

```
1<= L1 <=4, 1<= L2 <=4, 1<= L3 <=4, 1<= L4 <=4,
1<= C1 <=4, 1<= C2 <=4, 1<= C3 <=4, 1<= C4 <=4,
```

The accepted positions (propositions):

```
(L1 = L2 + 1 and C1 = C2 + 2) or (L1 = L2 +2 and
 C1 = C2 + 1) or
(L1 = L2 -1 and C1 =C2 -2) or (L1 = L2 - 2 and
 C1 = C2 - 1);

(L1 = L3 + 1 and C1 = C3 + 2) or (L1 = L3 + 2 and
 C1 = C3 + 1) or
(L1 = L3 - 1 and C1 = C3 - 2) or (L1 = L3 - 2 and
 C1 = C3 - 1);
```

$\ddots$

Of course, the solutions are the same.

Remark 2. It is not always possible to capture (or more precisely to describe) the same information with positive and negative formulae. We use, as example, the case of constrained clause (see section 3). The c-clause $[\![P(x, y) : x \neq y]\!]$ denotes on the Herbrand universe over the signature $\Sigma = \{a^{(0)}, f^{(1)}\}$ (see e.g., Ref. 8) a set of ground (unit) clauses requiring to be described in positive-form *infinitely many* ground literals:

$P(a, f(a)), P(f(a), a), P(f(a), f(f(a))), P(f(f(a)), f(a)), \ldots$

The problem of reasoning from negative information is an important one addressed in automated deduction. It is one of the research problems (problem number 7) in Ref. 32: *Does there exist an inference rule . . . that reasons from inequalities . . . in contrast to reason from equalities?*

In the next section, we contend that the anticonsequence concept can contribute quite naturally to a theory of reasoning *from* inequalities.

6. TOWARD A UNIFIED APPROACH

The method recalled in section 3 allows in its present form to deal with three of the four possibilities analyzed in this work: accepted/accepted and accepted/rejected as already mentioned and rejected/accepted because it is an intrinsically constraint-based method.

On the other hand, the notions in section 4 do not provide any method to *compute* formulae to be rejected on the ground of other rejected formulae. A criterium to *verify* that a formula must be rejected corresponds to the definitions given there.

A natural question is therefore: *"Can the method described in section 3 be extended in order to cover also the rejected/rejected case?"*

The problem of *computing* rejected propositions from a set of rejected ones is similar (but dual in some sense) to the one addressed in inductive generalization[9,10] and in inductive logic programming (see e.g., Ref. 33) where, roughly speaking, from formulae denoting particular cases of a relation, a *general* (i.e., taking into account *the full domain* of the relation) specification is searched.

The main difference being that in these works, the particular cases are *accepted* formulae. In contrast, we start with *rejected* ones. But, if we use the anticonsequence notion, in both cases from a formula B we are looking for a formula A such that $A \models B$.

Aside the case of B contradictory (in such special case any contradictory formula A will work), the problem is basically the same.

What can be the usefulness of such a unified approach?

As a natural example, consider a system mechanizing mathematics and a user looking for proving (or refuting) a conjecture. In addition to the abilities to prove (via its deductive capabilities) or disprove (via its capabilities of building counter-examples, or equivalently models), he or she can be greatly helped by using *already-rejected* conjectures that seem to have relationship with the conjecture at hand.

Maybe the best to illustrate the suggested approach to the problem is to refer to the already mentioned research problem number 7 in Ref. 32 and two examples there.

A well-known inference rule for reasoning with equalities (implemented in all resolution-based theorem provers) is *paramodulation*. It allows in only *one operation* to instantiate variables (universally quantified) *and* to replace equals by equals (a formal definition can be found in, e.g., Ref. 8).

For example, applying paramodulation on clauses (1) and (2) below (a, b, c denote constants, x, y, z denote variables, implicitly universally quantified):

(1) $\underline{a + b} = c$

(2) $\underline{(x + y)} + z = x + (y + z)$

it can be obtained (using a substitution $\{x \leftarrow a, y \leftarrow b\}$ to unify the underlined expressions and replacing in (2) $a + b$ by c):

$c + z = a + (b + z)$

But paramodulation does not apply in reasoning *from inequalities* (otherwise stated, it is not possible to soundly replace *unequals* by *unequals*). L. Wos[32] asks for a (obviously *sound*) rule similar to paramodulation for doing so.

As an example, he points out the impossibility to deduce by paramodulation:

$a + (b + z) \neq c + z$

from

$(a + b) \neq c$

in arithmetic (structure Ar below).

$\mathsf{Ar} = < \mathbb{N}, \{0, succ, +, \times\}, \{=, \leq\} >$

Now in a system incorporating refutability (anticonsequence), it is natural to reject $a + (b + z) = c + z$ on the basis of rejection of $(a + b) = c$ (we use freely standard theorems of arithmetic):

$a + (b + z) = c + z$

$(a + b) + z = c + z$

$(a + b) + z - z = c + z - z$

$(a + b) + (z - z) = c + (z - z)$
$(a + b) + 0 = c + 0$
$(a + b) = c$ (already rejected)

Of course, this is not an answer to Wos' wish because reasoning is not from $(a + b) \neq c$, but the *effect* can be considered as near to Wos' wish: here inequality *has guided* the search.

It should be clear that our only contention is that the anticonsequence notion is perfectly adapted to the reasoning *from* negative information (in this case, rejected and negative information coincides) in order to make explicit negative information.

As subsumption of a clause D by a clause C is a sufficient condition for logical consequence, that is, if C subsumes D then $C \models D$. This is easy to see: if a clause C subsumes a clause D then C is *more general* than D because C contains less (or the same number) of disjuncts and possibly variables in places where D contains ground terms (the converse cannot be true, by definition of substitution). Taking into account that variables in clauses are *universally quantified*, clearly D is a particular case of C. Therefore, verification of the claimed logical consequence is a trivial matter.

In order to reasoning from rejected to rejected propositions (or from negative to negative information), we propose to use a τ-anti-subsumption rule. It allows to start reasoning from (possibly a disjunction of) inequalities.

We say that a clause C τ-*anti-subsumes* a clause D iff there exists a substitution σ such that $\sigma C \subseteq_\tau D$ where $\subseteq_\tau$ means subset modulo the theory τ.

Otherwise stated, once applied σ and the τ axioms $C \subseteq D$.

The τ-anti-subsumption rule can be seen as a way of 'discovering' (by introducing *virtual contrapositives*) logical consequences of given inequalities. This rule will be less useful when inequalities are obtained by negating universally quantified sentences because the generalization it includes will find again the sentence where the inequality comes from.

Presently, we study the properties of an algorithm of τ-generalization in order to implement an automated inference system able to deal with the four entailment possibilities studied in this work. This algorithm can be obtained by adapting and generalizing Plotkin's propositions for generalization.[9,10]

Two main differences w.r.t. Plotkin's work:

- The generalization (not necessarily the least one) is from only one formula.
- Generalization must be done modulo a theory.

Concerning the example above, generalization can be done in two steps:
1. $(a + b) = c$
is Ar-anti-subsumed (Ar for arithmetic) by
2. $(a + b) + z = c + z$
(with substitution $\sigma : \{z \leftarrow 0\}$).
2. can be Ar rewritten in:
3. $a + (b + z) = c + z$

L. Wos insists on the soundness of the inference in reasoning from inequalities, and he warns about the (unsound) inference step from

$a \times b \neq c$

to

$a \times (b \times z) \neq c \times z$

Using τ-anti-subsumption, we **cannot** reject $a \times (b \times z) = c \times z$ on the basis of $a \times b = c$ because in Ar

$a \times (b \times z) = c \times z \not\models a \times b = c$

or, if we want to reject $a \times (b \times z) = c \times z$ on the basis of $a \times b = c$ we need to introduce the constraint $z \neq 0$.

Maybe the reader is uncomfortable with the notion of rejection as defined in section 4, because it is too much linked to the one of deduction. Indeed, a formula ϕ is rejected from an already rejected formula ψ in a set of axioms A iff ψ is a logical consequence of $A \cup \{\phi\}$. Particularly, it is impossible to reject a proposition from axioms only. We think that our GPL rule can be easily adapted to provide a natural extension of the rejection notion.

Example 9.

Assume the formula $A \models B$.

If B is rejected, then it is clear using the definitions in section 4 that A must be rejected, since B is a logical consequence of A. If we only know that A is rejected then B cannot be rejected using the aforementioned definition. But if we use the GPL rule, assuming that $\neg A$ holds (i.e., A is rejected) then $A \Rightarrow B$ is redundant (subsumed by $\neg A$) thus B does not occur any more in the axioms. Hence, $\neg B$ is trivially pure and B may be rejected.

It should be noticed that negation symbol $\neg$ is used for convenience only and should not be confused with standard negation (in this example $\neg B$ could be rejected as well).

7. CONCLUSION

Consequence relations and the corresponding deducibility relations have been deeply and extensively studied for many years now. But other forms of logically connecting formulae ('entailments'), such as *disinference* and *anticonsequence*, addressed in sections 3 and 4, respectively, are practically absent from Artificial Intelligence and Computational Logic literature.

Hopefully, the unified view given in this paper has shown that it is worth-studying forms of 'entailment' other than logical consequence (whose enormous importance does not need to be recalled).

Hints have been given about a simple extension of a method developed by the authors elsewhere, that should allow to define a calculus able to deal with the four possible combinations of accepted rejected propositions in a unique framework.

Working out of the technical details of this calculus is part of ongoing research.

References

1. Caferra R, Zabel N. Extending resolution for model construction. In: Proc Logics AI-JELIA 90. New York: Springer Verlag; 1990. LNAI 478. pp 23–32.

2. Caferra R, Peltier N. Model building and interactive theory discovery. In: Proc TABLEAUX'95. New York: Springer Verlag; 1995. LNAI 918. pp 154–168.
3. Caferra R, Peltier N. Extending semantic resolution via automated model building: Applications. In: Proc IJCAI'95. San Francisco, CA: Morgan Kaufmann; 1995. pp 328–334.
4. Caferra R, Peltier N. A new technique for verifying and correcting logic programs. J Automat Reason 1997;19(3):277–318.
5. Caferra R, Zabel N. A simultaneous search for refutations and models by equational constraints. J Symbolic Comput 1992;13:613–641.
6. Caferra R, Zabel N. Model construction using tableaux extended by equational problems. J Logic Comput 1993;3(1):3–25.
7. Caferra R, Peltier N. Disinference rules, model building and abduction. In: Orlowska E, editor. Logic at work. Essays dedicated to the memory of Helena Rasiowa, chapter 20 (part 5: Logic in Computer Science). Heidelberg, Germany: Physica-Verlag; 1998. pp 331–353.
8. Caferra R, Leitsch A, Peltier N. Automated model building. In: Applied logic series vol 31. Dordrecht, Boston, London: Kluwer; 2004.
9. Plotkin GD. A note on inductive generalization. In: Meltzer B, Michie D, editors. Machine intelligence 5. UK: Edinburgh University Press; 1969. pp 153–163.
10. Plotkin GD. A further note on inductive generalization. In: Meltzer B, Michie D, editors. Machine intelligence 6. UK: Edinburgh University Press; 1971. pp 101–124.
11. Scott D. Completeness and axiomatibility in many-valued logic. In: Henkin L et al., (editors). Proc Tarski Symp. American Mathematical Society. pp 411–435. Proc Symposia Pure Mathematics, 1974, 25.
12. Wójcicki R. Theory of logical calculi. Basic theory of consequence operations. Studies in epistemology, logic, methodology, and philosophy of science. Vol 199 Kluwer Academic Publishers; 1988.
13. Anderson AR, Belnap ND Jr. Entailment, the logic of relevance and necessity, vol 1. Princeton, NJ: Princeton University Press; 1975.
14. Ebbinghaus HD, Flum J, Thomas W. Mathematical logic. In: Undergraduate texts in mathematics. New York: Springer-Verlag; 1984.
15. Etchemendy J. Tarski on truth and logical consequence. J Symbolic Logic 1988;53(1):51–79.
16. Curry HB. Foundations of mathematical logic. New York, Dover Publications; 1977.
17. Curry HB. A theory of formal deducibility. In: Notre Dame mathematical lectures vol 6, 2nd ed. Notre Dame, Indiana; 1957.
18. Gabbay DM. What is negation in a system? In: Drake FR, Truss JK, editors. Logic colloquium '86. Amsterdam Elsevier; 1988. pp 95–112.
19. Smullyan RM. First-Order logic. New York: Springer-Verlag, 1968.
20. Fitting M. First-order logic and automated theorem proving. In: Texts and monographs in computer science. New York Berlin Heidelberg: Springer-Verlag; 1990.
21. Comon H, Lescanne P. Equational problems and disunification. J Symbolic Comput 1989;7:371–475.
22. Mal'cev AI. Axiomatizable classes of locally free algebra of various type. In: The metamathematics of algebraic systems: Collected papers 1936–1967. Chapter 23. Benjamin Franklin Wells: North Holland; 1971. pp 262–281.
23. Smullyan RM. Theory of formal systems. Annals of Mathematics Studies. Princeton University Press, 1961. Revised Edition.
24. Wójcicki R. Dual counterparts of consequence operations. Bull Sect Logic Polish Acad Sci 1973;2(1):54–57.
25. Goranko V. Refutation systems in Modal Logic. Studia Logica 1994;53:299–324.
26. Caicedo X. A formal system for the non-theorems of the propositional calculus. Notre Dame J Formal Logic 1978;19(1):147–151.
27. Goranko V. Proving unprovability in some normal modal logics. Bull Sect Logic. Polish Acad Sci 1991;20(1):23–29.
28. Słupecki J, Bryll G, Wybraniec-Skardowska U. Theory of rejected propositions.I. Studia Logica 1971;29:75–115.

29. Scott D. Rules and derived rules. In: Stenlund S, editor. Logical theory and semantic analysis, essays dedicated to Stig Kanger on his fiftieth birthday. D. Reidel Publishing Company. Dordrecht-Holland, Boston; 1974. pp 147–161.
30. Falkowski B-J, Schmitz L. A note on the queen's problem. Inform Process Letters 1986;23(1):39–46.
31. Selman B, Levesque H, Mitchell D. A new method for solving hard satisfiability problems. In Proc AAAI-92. Cambridge, MA: MIT Press; 1992. pp 440–446.
32. Wos L. Automated reasoning: 33 Basic research problems. Englewood Cliffs, NJ: Prentice Hall; 1988.
33. Muggleton S, de Raedt L. Inductive logic programming: Theory and methods. J Logic Program 1994;19,20:629–679.
34. Apt KR, Bol RN. Logic programming and negation: A survey. J Logic Programming 1994;19,20:9–71.
35. Borkowski L. Jan Łukasiewicz, selected works. Studies in Logic and the Foundations of Mathematics. Nort-Holland, 1970.
36. Dutkiewicz R. The method of axiomatic rejection for the intuitionistic propositional calculus. Studia Logica 1989;48:449–459.
37. Słupecki J, Bryll G, Wybraniec-Skardowska U. The theory of rejected propositions.II. Studia Logica 1972;30:97–139.