A reflective proof system for reasoning in contexts

Pierre E. Bonzon

University of Lausanne
1015 Lausanne, Switzerland
pbonzon@hec.unil.ch[1]

Abstract

We consider the problem of building an automated proof system for reasoning in contexts. Towards that goal, we first define a language of contextual implications, and give its operational semantics under the form of a natural deduction system using explicit context assertions. We show that this proof system has an equivalent straightforward logic program, which in turn can be reified, i.e. defined as an outer meta-level context, and thus applied to itself. More powerful reasoning models (e.g. those involving theory lifting) can be then implemented by applying the same logic program on extended meta-level contexts containing specialized axioms.

As a theoretical application, we consider the task of concept learning. In order to achieve generality (i.e. abstracting solution classes from problem instances), we argue that concept learning goals should aim at the discovery of meta-level operators representing the sequence of inference steps leading to object-level moves or actions. We illustrate this idea with the definition of a learning model based on partial deduction with respect to theory lifting.

Introduction

As argued by John McCarthy, the new logic agenda is to formalize contexts, make context a parameter, build context hierarchies, and then use non-monotonic reasoning for upward inheritance. Following recent work (Buvac, Buvac, and Mason 1995; Buvac 1996), formal axiomatized theories for both a propositional logic and a quantificational logic of contexts have been proposed. The second of these two proposals is of particular interest for AI applications, since, as opportunely reminded (Buvac 1996), " these applications require the expressive power of first order logics". But without an effective automated proof system, axiomatized theories are barely of any use when it comes to implement applications. Things start to look better however when both the non-logical axioms of a theory and its corresponding proof system (the later encompassing its logical axioms and inference rules) can be represented as a *logic program*, as one can take advantage of the powerful machinery which is available today to interpret and/or compile Prolog programs.

The first goal of the work reported here was therefore to identify the subset of the sound and complete 2-sorted predicate calculus with equality defined by Buvac that could be used for that purpose. Towards that goal, we first define a language of contextual implications and give its operational semantics under the form of a *natural deduction* system using explicit context assertions. We refrain from any attempt to define a model semantics, and do not hypothesize any soundness and/or completness properties. In contrast to previous proposals, the corresponding assumption free proof system is neither flat nor general, i.e enforces only the backward direction of the usual flatness axiom schema. This derived proof system has an equivalent straightforward logic program, which in turn can be reified, i.e. defined as an outer meta-level context, and then applied to itself. More powerful reasoning models (e.g. those involving *theory lifting*) can be thus implemented by applying the same logic program on extended meta-level contexts containing specialized axioms. Other, more specific models (e.g. those involving *private beliefs*) will require both extended meta-level contexts and a modified logic program relaxing further what was left from the flatness axiom schema.

A second goal was to explore theoretical applications. It is an easy matter, in any rule system, to keep track of successive rule applications into so-called *deduction traces*. Usually, these deduction traces represent sequences of object-level rule applications. But here, in addition, the reflective properties just mentioned mean that generalized deduction traces can also represent inference steps (i.e. meta-level rule applications). Taken in conjunction with the general theory lifting mechanism (McCarthy 1993) that can itself be reified and included into a meta-level context, these capabilities open the door

to a new kind of learning models. Instead of looking for the definition of object-level operators or concepts (thus facing the difficult problem of abstracting solutions classes from problem instances), learning goals can aim at the discovery of meta-level operators representing the sequence of inference steps leading to object-level moves or actions. These meta-level operators can be easily obtainable from generalized deduction traces. As a preliminary report on this ongoing work, we illustrate this idea with the definition of a learning model based on partial deduction with respect to theory lifting. This particular system can be used to both produce and later reuse partially instantiated deduction traces corresponding to generic meta-level operators applicable in various contexts.

A Restricted Logic

We shall rely on a few basic definitions, including those for terms and atomic formulas of first order logic. We will also assume the usual definitions of a substitution θ, of an instance $e\theta$ or $E\theta$ (of a single expression e or of a set of expressions E) and of a most general unifier (mgu) of a set of expressions.

Syntax

Let us first define a language L of *contextual implications*. Following (Buvac 1996), we consider a two-sorted predicate calculus (without identity), and denote by C the set of terms of the context sort, and by A the set of atomic formulas of the discourse sort. Our language L is then defined as the least set satisfying the equation $L=A\cup L\rightarrow L\cup ist(C, L)$. The subset $F \subset L$ whose formulas do not contain implications defines the language of *contextual facts.*

Definition (context assertion): A context assertion is a formula $\alpha{:}\Phi$ where α is a term denoting a context and Φ is a a set of contextual implications, i.e. $\Phi \subset L$, with all variables appearing in either α or Φ universally quantified

e.g.
```
block(s0):[on(a,b), on(b,c)].
block(S):[on(X,Y)->above(X,Y)].
c0:[ist(C,P->Q)->ist(C,P)->ist(C,Q)].
```

N.B We adopt the PROLOG convention of using capital letters to represent variables and following the results of ambivalent logic (Kalsbeek and Jiang 1995) we do not distinguish between ordinary (e.g. `X,Y,C`) and meta variables (e.g. `P,Q`) representing arbitrary formulas.

Given a context assertion $\alpha{:}\Phi$ and a context C, we will say that assertion $\alpha{:}\Phi$ *relates to* (or partially defines) context C if C can be unified with α.

Definition (axiom instance): If C is a context and $\alpha{:}\Phi$ is a context assertion related to C, with $C\theta= \alpha\theta$ (i.e. θ is a mgu of C and α), then any formula P wich can be unified with a formula $\phi\theta \in \Phi\theta$ is called an *axiom instance* for context C.

Proof Systems

We shall assume a basic understanding of the management of assumptions in natural deduction, see for instance (Lalement 1993).

Notation

instance(C,P) $\Leftrightarrow$ P is an axiom instance for C, i.e in Prolog, `instance(C,P):- C:L,member(P,L).`

C: $\Leftrightarrow$ context C is assumed, i.e. C is an active assumption

[C:] $\Leftrightarrow$ if $\exists$ at most one active assumption, namely C: , then it can be discharged; else if there is no active assumption, then C: can be assumed and simultaneously discharged (i.e. there is no actual discharge)

1. Natural deduction of contextual facts

Applicable inference rules are defined as follows:

axiom	$\dfrac{\text{instance(C,P)}}{\text{ist(C,P)}}$
entering context	$\dfrac{\text{C:, ist(C,P)}}{\text{P}}$
implication elimination (modus ponens)	$\dfrac{\text{P}\rightarrow\text{Q , P}}{\text{Q}}$
leaving context	$\dfrac{\text{[C:] P}}{\text{ist(C,P)}}$

Conspicuously absent from this set of inference rules is the converse of modus ponens, i.e. the rule

implication introduction	$\dfrac{\text{[P] Q}}{\text{P}\rightarrow\text{Q}}$

In full generality, this rule is required to ensure the completness of the system. However, since we will restrict ourselves to the derivation of *contextual facts*, this rule can be dispensed with altogether (by *reductio ad absurdum*: such a rule is required by *modus ponens* only; furthermore, as implications associate to the right, it is required to establish its first premiss $P \rightarrow Q$ in order to derive Q which otherwise cannot be derived (1); as its premiss must hold, Q must hold when P is assumed; Q must hold either by *axiom*, which contradicts (1), or by *modus ponens*, which in turn leads into an infinite regress).

Definition (deduction relation $\vdash$): A fact P is said to be derivable in the (*inner*) context C, noted $C \vdash P$, iff $\exists$ a derivation (in the usual sense) of ist(C,P) with no active assumptions. Since we restrict ourselves to the derivation of contextual facts, our deduction relation $\vdash$ is defined over $C \times F$.

N.B. Extended derivation of formulas with arbitrary nesting depth will require the introduction of *outer* contexts reifying $\vdash$ (see below).

Let us now consider the following assumption free proof system and logic program:

2. Assumption free proof system

axiom	$\dfrac{\text{instance(C,P)}}{\text{ist(C,P)}}$
modus ponens	$\dfrac{\text{ist(C, P}\rightarrow\text{Q)}\ ,\ \text{ist(C, P)}}{\text{ist(C,Q)}}$
semi-flatness	$\dfrac{\text{ist(C1,P1)}}{\text{ist(C, ist(C1,P1))}}$

3. Logic program

```
(3.1)  ist(C,P) :- instance(C,P).
(3.2)  ist(C,Q) :- ist(C,P->Q),ist(C,P).
(3.3)  ist(C,ist(C1,P1)) :- ist(C1,P1).
```

Proposition (equivalence): Proof systems 1., 2. and 3. are equivalent, i.e. if a formula is derivable in one system, then it is also derivable in the other two systems.

Proof (by rule application cases).

a) $1. \Rightarrow 2.$
- *axiom* in both systems are literally equivalent
- *modus ponens* in 1. either directly or indirectly requires *entering context* (otherwise its first premise $P \rightarrow Q$ leads to an infinite regress); consider these two steps together with the subsequent step of *leaving context*; these combined steps necessarily rely on a context assumption, say *C:*, with *ist(C,P→Q)* previously established; provided that there can be at most one active context assumption when leaving, the second premiss *P* either relies on the same assumption, with *ist(C,P)* previously established, or does not depend on any assumption; in this last case, *P* must be previously established and *P=ist(C1,P1)*; in any case, discharging *C:* will establish *ist(C,Q)* from *ist(C,P→Q)* previously established and either *ist(C,P)* or *P* previously established; as *ist(C, P)* follows by *semi-flatness* from *P* whenever *P=ist(C1,P1)*, these combined steps reduce to *modus ponens* in 2.
- *entering context* is required by *modus ponens* only, and this case is covered above
- if *P = ist(C1,P1)* then *leaving context* reduces to *semi-flatness*; otherwise *leaving context* can only require *modus ponens*, and this case is covered above

b) $2. \Rightarrow 1.$
- *axiom* in both systems are literally equivalent
- *modus ponens* in 2. expands in 1. into a sequence of *entering context*, *modus ponens* and *leaving context*
- *semi-flatness* reduces to *leaving context* (without actual discharge).

c) $2. \Leftrightarrow 3.$
system 2. and logic program 3. are literally equivalent (i.e.syntactic variant) □.

Relating 2. to the complete system of (Buvac 1996), by discarding non applicable axioms and inference rules and taking into account explicit context assertions, shows that in contrast to this system ours is neither flat nor general. *Semi-flatness* as defined in system 2. retains only the backward direction of the flatness axiom schema. If, as suggested in (McCarthy 1993), we regard ist(C,P) as analogous to C→P, then semi-flatness is mirrored by the valid formula (C1→P1) →(C→C1→P1), whereas the converse formula is not valid.

N.B. Rejecting half of flatness protects logic program 3. from harmful indirect left recursion. Yet, systems such as Prolog which rely on a uniform depth first search strategy will still get into infinite recursion. As a way out, one may implement an *iterative deepening search* as follows:

```
search(ist(C,Q)\N):-ist(C,Q)\N;
                    N1 is N+1,
                    search(ist(C,Q)\N1).
```

with

```
ist(C,Q)\N :- instance(C,Q);
              N>0, N1 is N-1,
              ist(C,P->Q)\N1,
              ist(C,P)\N1.
ist(C0,ist(C,P))\N :- ist(C,P)\N.
```

and the new top level call

```
ist(C,Q) :- search(ist(C,Q)\1).
```

As postulated in (McCarthy 1993), our language *L* allows for relations among contexts to be expressed as sentences in the language. In particular, ist(C,P) formulas could "always be considered as themselves asserted within a context", leading thus to introduce *outer* contexts. In contrast to previous similar definitions (e.g. Weyhrauch 1980; Bowen and Kowalski 1982; Giunchiglia, Serafini and Simpson 1992; Attardi and Simi 1995) that are stated as inference rules allowing for the deduction of higher-order (or meta-level) formulas, the following definitions *characterize* a given outer context c0 with respect to the deduction relation.

Let c0 be a non empty context (i.e. such that there exists at least one assertion related to c0).

Definitions

(***reflecting up***): if $\forall$ C,P (C $\vdash$ P $\Rightarrow$ c0 $\vdash$ ist(C,P)) then c0 *reflects up* relation $\vdash$

(***reflecting down***): if $\forall$ C,P (c0 $\vdash$ ist(C,P) $\Rightarrow$ C $\vdash$ P) then c0 *reflects down* relation $\vdash$

(***reifying***): if c0 reflects relation $\vdash$ both up and down, then c0 is a *reification* of relation $\vdash$

Proposition (reification): If c0:[ist(C, P$\rightarrow$Q) $\rightarrow$ ist(C,P) $\rightarrow$ ist(C,Q)] is the only context assertion related to c0, then c0 is a reification of $\vdash$.

Proof

a) *reflecting up* follows by *semi-flatness* in 2.

b) *reflecting down* follows by rule application cases in 2.: ist(c0,ist(C,Q)) derives from *semi-flatness* or *modus ponens*; thus either ist(C,Q), or we have

$$\frac{\text{ist(c0, P'}\rightarrow\text{ ist(C,Q)) , ist(c0, P')}}{\text{ist(c0, ist(C,Q))}} \quad (1)$$

and in turn, ist(c0, P'$\rightarrow$ ist(C,Q)) in (1) necessarily derives from *modus ponens*, i.e. we have

$$\frac{\text{ist(c0, P''}\rightarrow\text{P'}\rightarrow\text{ ist(C,Q)) , ist(c0, P'')}}{\text{ist(c0, P'}\rightarrow\text{ ist(C,Q))}} \quad (2)$$

The first premiss in (2), i.e. ist(c0, P''$\rightarrow$P'$\rightarrow$ ist(C,Q)), derives from *axiom* (if it were to derive from another *modus modens*, this would lead to an infinite regress), i.e. we have ist(c0, ist(C, P$\rightarrow$Q) $\rightarrow$ ist(C,P) $\rightarrow$ ist(C,Q)) which leads to P'= ist(C,P) and P'' = ist(C, P$\rightarrow$Q); finally, the second premiss in (1) and (2), i.e. ist(c0, P') and ist(c0, P''), both derive from *semi-flatness* (it they were to derive from *modus ponens*, this would lead to an infinite regress); both P' and P'' must thus be derivable, and ist(C,Q) follows by *modus ponens* on P' and P'' □.

Theoretical Relevance and/or Applications

By definition, if c0 is a reification of $\vdash$ then it has the same deductive power (modulo a modality), i.e. the derivation of any P in any inner context C can be replaced by the derivation of ist(C,P) in the outer context c0. The following is an example using previously given context assertions and system 2.

1 ist(c0, ist(C,P$\rightarrow$Q) $\rightarrow$ ist(C,P) $\rightarrow$ ist(C,Q))
(by *axiom*)
2 ist(block(s0), on(X,Y) $\rightarrow$ above(X,Y))
(by *axiom*)
3 ist(c0, ist(block(s0), on(X,Y) $\rightarrow$ above(X,Y)))
(by *semi-flatness* on 2)
4 ist(c0, ist(block(s0), on(X,Y)) $\rightarrow$ ist(block(s0), above(X,Y)))
(by *modus ponens* on 1 and 3)
5 ist(block(s0), on(a,b))
(by *axiom*)
6 ist(c0, ist(block(s0), on(a,b)))
(by *semi-flatness* on 5)
7 ist(c0, ist(block(s0), above(a,b)))
(by *modus ponens* on 4 and 6)

More interesting cases arise when c0 corresponds to non conservative extensions of $\vdash$, i.e embodies stronger principles (Giunchiglia and Serafini 1994), thus allowing for the derivation (modulo a modality) of formulas that would otherwise not be derivable.

Theory lifting

As an example of a derivation that requires an extended outer context c0, let us consider the following assertions implemcnting *theory lifting* (McCarthy 1993) :

```
above_theory:[on(X,Y) -> above(X,Y)].
block(s0):[on(a,b)].
block(S) :[ist(above_theory,P)
            -> ist(block(S),P)].

c0:[ist(C,P->Q)  -> ist(C,P) -> ist(C,Q),
    ist(C, ist(A,P) -> ist(C,P))
     ->ist(A,P)
        -> ist(C,P)].
```

According to the *specific lifting axiom* now asserted within `block(S)`, any rule `P` stated within `above_theory` holds within `block(S)`. To actually enforce this (or any other) specific axiom, a new *lifting inference rule* is asserted within outer context `c0`. It must be noted that in contrast to (McCarthy 1993) and as postulated in (Attardi and Simi 1994), our implementation of theory lifting relies on an explicit outer context. However, in contrast to Attardi and Simi (whose lifting axiom asserted within c0 is bound to the `above_theory` context), our lifting inference rule is truly general.

Private beliefs

Semi-flat contexts cannot be models for private beliefs: if a modality `bel(X,P)` is introduced to mean that agent `X` believes `P` to be true, then `bel(Y,bel(X,P)):- bel(X,P)` is a counterintuitive inference rule. Let us consider instead the restricted rule

```
ist(b0, bel(X,P)):- instance(X,P)          (1)
```

together with the extended outer context assertion

```
b0: [bel(X,P->Q) -> bel(X,P) -> bel(X,Q),
     (bel(X,A) -> bel(X,B) -> bel(X,C))
       -> bel(Y,bel(X,A))
           -> bel(Y,bel(X,B))
               -> bel(Y,bel(X,C))].
```

in which the second inference rule enables the *nested application* (at any depth) of the first or any other similar rule. Substituting (1) for the semi-flatness rule in logic program 3. will result in a new proof system allowing for the derivation, within outer context `b0`, of `bel` modalities (modulo outer `ist` modalities), i.e. of nested formulas of the form `ist(b0, bel(X,P))` `ist(b0, bel(Y,bel(X,P)))`, and so on.

Reflective contexts

Reflective contexts, which keep track of derivations, involve a new modality `reflect(V,W)` meaning that `W` holds because of the sequence of inference steps `V`. A simple reflective context can be defined as

```
r0:[instance(C,P)->reflect(axiom(C),ist(C,P)),
    reflect(X,ist(C,P->Q))
     -> reflect(Y,ist(C,P))
         -> reflect(mp(X,Y),ist(C,Q))].
```

Its first rule simply reflects an *axiom* instantiation. The second one reflects an application of *modus ponens* reified into the term `mp(X,Y)`, where `X` and `Y` are the reified inference steps needed to derive the antecedents `P->Q` and `P`, respectively. As an example, the call

```
ist(r0,reflect(D,ist(C,above(X,Y))))
```

will result in the following instantiation for `D`

```
D = mp(axiom(block(s0)),axiom(block(s0)))
```

N.B. As `instance(C,P)` is actually a Prolog procedure (defined earlier), in order for such a call to appear in a context assertion, semi-flatness in logic program 3. must be extended with

(3.4) `ist(C,instance(C1,P1)):-instance(C1,P1).`

The inference steps taken during the above derivation have been turned into a *fully instantiated* trace `D`. Conversely `D` can be *forced* back into `r0`, driving the derivation into the same inference steps. Constraining a derivation with a fully instantiated tracc leads to a reduced search space and thus makes `D` equivalent to a *specific* meta-level operator for deriving `above(X,Y)` in context `block(s0)`. Similarly, *partially instantiated* traces would lead to *generic* operators applicable in various contexts, e.g. in any context `block(S)`.

Deriving traces, e.g. by hopping up first to another meta-level context, accounts for the *learning* of an operator. Hopping down again to force these traces is equivalent to *applying* an operator. To illustrate these ideas, we shall consider a particular learning model based on *theory lifting*. Partially instantiated traces will be constituted of embedded sequences of modus ponens applications whose first antecedent only will be specified. Furthermore, this first antecedent will follow from the lifting inference rule introduced earlier (i.e. a process equivalent to *partial deduction* with respect to theory lifting). Our learning model can thus be defined in terms of the following inputs:

- a collection of *domain* theories, e.g. `block(S)`
- a collection of *liftable theories* together with specific *lifting axioms* (e.g. as in `above_theory`)
- a reflective context allowing to carry out *partial deductions* with respect to theory lifting.

Such a reflective context `r1` associates a reflective form of the lifting inference rule with a shortened form of modus ponens, i.e.

```
r1:[instance(C,ist(A,P) -> ist(C,P))
    ->instance(A,P)
      ->reflect(lift(axiom(C),axiom(A)),ist(C,P))
    reflect(X,ist(C,P->Q))
     ->reflect(mp(X,_),ist(C,Q))].
```

The goal of the system is then to discover *control rules* given under the form of partially instantiated traces of applicable inference steps leading to specific facts which follow from the lifting of theories into selected domains. While it is notoriously difficult to abstract solution classes

or concepts from specific facts, applicable inference steps are not bound to specific data instances and thus directly achieve *generality.*

As an example, the call

```
ist(r1,reflect(D,ist(C,above(X,Y))))
```

will lead to the following partial instantiation for `D`

```
D = mp(lift(axiom(block(_)),
            axiom(above_theory)),_)
```

thus defining a generic meta-level operator, which could be seen as an application of the following *control rule*:

> *in order to get specific facts, first select a specific lifting axiom, then instantiate the corresponding liftable theory, and finally conclude by applying some rule from the lifted theory.*

While this control rule itself was not explicitely given, an appropriate instantiation was *learned* under the form of a partially instantiated trace `D`.

Related work

(McCarthy, J. 1993) and (Buvac, S. 1996) have been the primary source of inspiration for this research. Our work is also very much related to (Attardi and Simi 1994), who first formally linked reasoning in context with natural deduction. Yet the system they end up with is not an automated one. The theoretical results reported in (Massacci 1996) rely on a definition of a tableaux calculus which "satisfies the strong confluence property and therefore can be adapted to many search heuristics". This system, however, is restricted to the propositional case.

Conclusions and future work

Through possible extensions of outer contexts mirroring its Prolog proof procedure, our reflective proof system can be tailored to the needs of various inner contexts. As an exemple, *common beliefs* can be seen as a case of theory lifting into private beliefs. On the other hand, the uniform depth first search strategy of Prolog results in its relative inefficiency. This could be alleviated by a direct implementation of a deepening search of logic program 3.

A semantic account of the language is needed in order to get an understanding of the semi-flatness hypothesis. By formalizing the analogy ist(C,P)≡C→P to include ist(C,ist(C1,P1))≡C→C1→P1, it could offer C→C1 as a model for "C *subsumes* C1", (C→C1)∧(C1→C) for "C is *equivalent* to C1", and (C→C1)∨(C1→C) for "C is *compatible* with C1". Any pair of compatible contexts would then satisfy the formula mirroring semi-flatness.

Acknowledgments

Thanks to the referees for their valuable comments. This work has been supported by the Swiss National Research Council under contract n° 2100-045664.95/1.

References

Attardi, G. and Simi, M. 1995. A Formalization of Viewpoints, *Fundamenta Informaticae*, 23 (3).

Attardi, G. and Simi, M. 1994. Proofs in Contexts, In: *Proc. 4th Intl. Conf. on Principles of Knowledge Representation and Reasoning.*

Bowen, K and Kowalski, R. 1982. Amalgamating Language and Metalanguage in Logic Programming. In: K. Clark and S. Tarnlund (eds), *Logic Programming*, Academic Press

Buvac, S.; Buvac, V. and Mason, I.A. 1995. Metamathematics of contexts, *Fundamenta Informaticae,* 23 (3).

Buvac, S. 1996. Quantificational Logic of Context, *Proc. 13th Natl. Conf. on Artificial Intelligence* .

Giunchiglia, F. and Serafini, L. 1994. Multilanguage hierarchical logics, or: how can we do without modal logics, *Artificial Intelligence*, 65, 29-70.

Giunchiglia, F.; Serafini, L. and Simpson, A. 1992. Hierarchical Meta-Logics: Intuitions, Proof Theory and Semantics, *Proc. 3th Intl. Workshop on Meta Programming in Logic*, LNCS vol 649, Springer Verlag .

Guha, R. 1991. *Contexts: A Formalization and some Applications*, Ph.D. Dissertation, Stanford University.

Kalsbeek, M. and Jiang, Y. 1995. A Vademecum of Ambivalent Logic. In: K. Apt and F. Turini (eds.), *Meta-Logics and Logic Programming*, MIT Press.

Lalement R. 1993. *Computation as Logic*, Prentice Hall, 1993.

Massacci, F. 1996. Contextual Reasoning is NP-complete, *Proc. 13th Natl. Conf. on Artificial Intelligence.*

McCarthy, J. 1993. Notes on Formalizing Context, *Proc. 13th Intl. Join Conf. on Artificial Intelligence*

Weyhrauch, R. 1980. Prolegomena to a Theory of Mechanized Reasoning, *Artificial Intelligence*, 13 (1).