Conservative and Definitional Extensions

Revision: May 10, 2016; Rendered: May 25, 2021

Conservative and definitional extension (see, e.g., [Hod97]). Actually, a generalization of definitional extension is presented here. Formalized with the *PIE* system.

1 Conservative Extension

Formula G is a *conservative extension* of formula F if and only if the following biconditional is valid. The right-to-left direction can be expressed as first-order validity, since second-order quantification is only in the antecedent and only existential there. The left-to-right direction in

---

$\mathit{conservative\_extension}(F, G)$

Defined as

$$F \leftrightarrow \mathit{proj}(S, G),$$

where

$$S := \mathsf{free\_predicates}(F).$$

1.1 Examples for Conservative Extensions

$\mathit{f_1}$

Defined as

$$\mathsf{a} \rightarrow \mathsf{b}.$$

$\mathit{ex\_ce_1}$

Defined as

$$\mathit{conservative\_extension}(\mathit{f_1}, (\mathit{f_1} \land (\mathsf{p} \leftrightarrow \mathsf{a}))).$$

This formula is valid: $\mathit{ex\_ce_1}$.

ex_ce$_2$

Defined as

$$\mathit{conservative\_extension}(f_1, (f_1 \land (\mathsf{p} \leftarrow \mathsf{a}))).$$

This formula is valid: *ex_ce*$_2$.

1.2 Counterexamples for Conservative Extensions

ex_ce$_3$

Defined as

$$\mathit{conservative\_extension}(f_1, (f_1 \land (\mathsf{b} \rightarrow \mathsf{p}) \land (\mathsf{p} \rightarrow \mathsf{a}))).$$

Failed to validate this formula: *ex_ce*$_3$.

def_extx(F, G)

Defined as

$$\mathsf{predicate\_definiens}(P, (F, G)),$$

where

$$\begin{array}{l} S_F := \mathsf{free\_predicates}(F), \\ S_G := \mathsf{free\_predicates}(G), \\ S_X := S_G \setminus S_F, \\ \text{singleton\_to\_member(S\_X, P)}. \end{array}$$

2 Implicit Definitional Extensions

We define the following concept: Formula G is an *implicit definitional extension* of formula F by unary predicate p iff

1. p does not occur in F.
2. There exists a formula Dx with no occurrences of p and with no bound occurrences of x such that $G \models \forall x\, px \leftrightarrow Dx$.
3. $F \equiv \exists p\, G$.

That property can be verified with just first-order reasoning: Dx is a definiens of p that can be computed by interpolation. The right-to left direction of the conservative extension property, condition (3), can generally be expressed as first-order validity. Also the left-to-right condition can be expressed as first-order validity, as shown by the following equivalences:

$$\begin{array}{ll} & F \models \exists p\, G[p] \\ \text{iff} & F \models \exists p\, G[p] \land \forall x\, px \leftrightarrow Dx \\ \text{iff} & F \models \exists p\, G[D] \land \forall x\, px \leftrightarrow Dx \\ \text{iff} & F \models G[D], \end{array}$$

where $G[p] = G$ and $G[D]$ stands for G with all occurrences of p replaced by Dx with x instantiated to the argument of p at the respective occurrence. The follwing entailment is another equivvalent way to express the above entailments. It is first-order expressible and might be more convenient since the replacement of the occurrences of p does not have to be explicitly performed:

$$F \models \forall p\, \neg(\forall x\, px \leftrightarrow Dx) \lor G[p].$$

---

predicate_definiens_xyz(P, F)

Defined as

$$\exists P\,(F \land P_X) \rightarrow \neg\exists P\,(F \land \neg P_X),$$

where

$$\begin{array}{l} N := \text{arity of } P \text{ in } F, \\ X := x_1, \ldots, x_N, \\ P_X := P(X). \end{array}$$

A version of predicate_definiens with fixed arguments (as obtained by mac_make_args). It is assumed that these are not used as constants elsewhere.

The following predicate implements the sketched method for verifying the implicit definitional extension property by means of first-order reasoning. The formula arguments are passed to the ppl_ predicates that perform macro expansion. The predicate succeeds iff the property holds and returns a definiens for the argument predicate as binding of D.

```
implicit_definitional_extension(F, G, P, D) :-
        ppl_valid((ex2(P, G)->F), [prover=cm, printing=false]),
        last_ppl_result(true),
        ppl_ipol(predicate_definiens_xyz(P, G),
                 [prover=cm, printing=false]),
        last_ppl_result(D),
        mac_get_arity(P, G, N),
        mac_make_args(N, X),
        mac_make_atom(P, X, P_X),
```

```
ppl_valid( (F -> all2(p,  (all(X,  (P_X<->D))->G))),
           [prover=cm, printing=false]),
last_ppl_result(true).
```

---

f_2

Defined as

$$f_1 \land (\mathsf{p} \rightarrow \mathsf{a}) \land (\mathsf{a} \land \mathsf{b} \rightarrow \mathsf{p}).$$

---

f_3

Defined as

$$f_1 \land (\mathsf{p} \rightarrow \mathsf{a}).$$

Both formulas f_2 and f_3 are conservative extensions of f_1:
This formula is valid: $\mathit{conservative\_extension}(f_1, f_2)$.
This formula is valid: $\mathit{conservative\_extension}(f_1, f_3)$.

We can test the predicate `implicit_definitional_extension` with these calls:

```
?- implicit_definitional_extension(f1, f2, p, D).  % succeeds
?- implicit_definitional_extension(f1, f3, p, D).  % fails
```

Only formula f_2 but not f_3 is an implicit definitional extension of f_1. The following formula is a computed definiens D for

`implicit_definitional_extension(f1, f2, p, D):`

$$\mathsf{a} \land \mathsf{b}.$$

References

[Hod97] Wilfrid Hodges. *A Shorter Model Theory.* Cambridge University Press, Cambridge, 1997.

Index