```
(***************************************************************************)
(*    Projet Formel - Calculus of Constructions V4.10 - Vernacular V2.3    *)
(***************************************************************************)
(*                                                                         *)
(*              Reynolds paradox, with the Type:Type axiom                 *)
(*                                                                         *)
(***************************************************************************)

(* use the Type:Type system, #use"Reyn_tac";V"Log_Rel" *)

(* this file shows an inconsistency is the logical system U-, defined in Girard's
   thesis, and in which the question of the consistency of U- was raised.
   The system U- may be described with the axioms

         (star,star), (k,star), (k,k), (k',k)

   in Barendregt-Berardi's GTS. The reader can check that we use only these axioms
   in the derivation, and not fully the Type:Type axiom.
   The lambda-term we get does not loop to itself by reduction *)

(* Reynolds operator *)

Definition PHI.
Body [A:Type](A->Prop)->Prop.

(* we extend this map functorialy *)

Definition phi:(A,B:Type)(A->B)->(PHI A)->(PHI B) =
[A,B:Type][f:A->B][z:(PHI A)][u:B->Prop](z [x:A](u (f x))).

(* preinitial PHI-algebra. We need the axiom (Type,Type) *)

Definition A0.
Body (A:Type)((PHI A)->A)->A.

Definition iter_A0.
Body [X:Type][f:(PHI X)->X][u:A0](u X f).

Definition intro : (PHI A0)->A0 =
[z:(PHI A0)][A:Type][f:(PHI A)->A](f (phi A0 A (iter_A0 A f) z)).

(* extension of PHI to relations. We can thus consider PHI as a functor
   on sets, that are types with a relations *)

Definition phi2 : (A:Type)(Rel A)->(Rel (PHI A)) =
[A:Type][R:(Rel A)](power (A->Prop) (power A R)).

(* partial equivalence relation defined on A0, so that the set A0,E0
   is an initial PHI-algebra in the category of sets *)

Definition teta : A0 -> (PHI A0) =
        (iter_A0 (PHI A0) (phi (PHI A0) A0 intro)).

Definition E0 : (Rel A0) = [x1,x2:A0](E:(Rel A0))
     (per A0 E) ->
     ((z1,z2:(PHI A0))(phi2 A0 E z1 z2)->(E (intro z1) (intro z2))) ->
     ((x1,x2:A0)(E x1 x2)->(E x1 (intro (teta x2)))) ->
     (E x1 x2).

Definition F0 : (Rel (A0->Prop)) = (power A0 E0).

Definition G0 : (Rel (PHI A0)) = (power (A0->Prop) F0).

(* the goal of what follows is to show that the set A0,E0 is in one-to-one
   correspondance with the set (PHI A0),(phi2 A0 E0), via intro,teta.
   From this, we deduce a contradiction via Cantor-Russell's argument *)

Goal (sym A0 E0).
By reds.
Do res_simpl H0.
Do res_simpl H3.
Do res_simpl H.
Save sym_E0.

Goal (trans A0 E0).
By reds.
Do res_simpl H1.
Do res_exact H5 y.
Do res_simpl H.
Do res_simpl H0.
Save trans_E0.

Goal (per A0 E0).
Resolve per_intro.
Exact sym_E0.
Exact trans_E0.
Save per_E0.

(* intro is a map from (PHI A0),(phi2 A0 E0) to A0,E0 *)

Goal (z1,z2:(PHI A0))(G0 z1 z2)->(E0 (intro z1) (intro z2)).
By reds.
Do res_simpl H1.
Do res_simpl H.
```

```
Do res_simpl H3.
Do res_simpl H4.
Save lemma1.

Goal (per (A0->Prop) F0).
Unfold F0.
Unfold power.
Do res_simpl per_E0.
Do res_simpl per_equiv.
Resolve per_intro.
By (hyps THEN hyps).
Resolve H1.
Resolve H3.
Do res_simpl H.
By (hyps THEN hyps).
Do res_exact H2 (y y0).
Do res_simpl H3.
Resolve H4.
Do res_exact H0 x0.
Do res_simpl H.
Save per_F0.


Goal (per (PHI A0) G0).
Unfold G0.
Unfold power.
Do res_simpl per_F0.
Do res_simpl per_equiv.
Resolve per_intro.
By (hyps THEN hyps).
Resolve H1.
Resolve H3.
Do res_simpl H.
By (hyps THEN hyps).
Do res_exact H2 (y y0).
Do res_simpl H3.
Resolve H4.
Do res_exact H0 x0.
Do res_simpl H.
Save per_G0.

Goal (x1,x2:A0)(E0 x1 x2)->(E0 x1 (intro (teta x2))).
By (hyps THEN hyps).
Do res_simpl H2.
Do res_simpl H.
Save id_intro_teta.

Goal (z1,z2:(PHI A0))(G0 z1 z2)->(G0 z1 (teta (intro z2))).
By (hyps THEN hyps).
Change (equiv (z1 x) (z2 [x:A0](y (intro (teta x))))).
Do res_simpl H.
Do res_intro H0.
Do res_simpl id_intro_teta.
Save id_teta_intro.

Goal (x1,x2:A0)(E0 x1 x2)->(G0 (teta x1) (teta x2)).
By hyps.
Change ([u,v:A0](G0 (teta u) (teta v)) x1 x2).
Resolve H.
Do res_simpl per_G0.
Do res_simpl per_intro.
Do res_simpl H0.
Do res_exact H1 (teta y).
By (hyps THEN hyps).
Change (equiv (z1 [u:A0](x (intro (teta u))))
          (z2 [u:A0](y (intro (teta u))))).
Do res_simpl H0.
Resolve H1.
Do res_simpl lemma1.
By hyps.
Do res_simpl id_teta_intro.
Save lemma_teta.

Definition psi : (A0->Prop)->A0 =
[u:A0->Prop](intro (F0 u)).

Definition inter : (PHI A0)->A0->Prop =
[C:(PHI A0)][x:A0](P:A0->Prop)(F0 P P)->(C P)->(P x).

Goal (z1,z2:(PHI A0))(G0 z1 z2)->(F0 (inter z1) (inter z2)).
By (hyps THEN hyps).
Do res_intro equiv_intro.

By hyps.
Do cut (equiv (P x) (P y)).
Do res_intro H4.
Resolve H5.
Do res_simpl H1.
Do cut (equiv (z1 P) (z2 P)).
Do res_intro H7.
Exact (H9 H3).
Do res_simpl H.
Do res_simpl H2.
```

```
By hyps.
Do cut (equiv (P x) (P y)).
Do res_intro H4.
Resolve H6.
Do res_simpl H1.
Do cut (equiv (z1 P) (z2 P)).
Do res_intro H7.
Exact (H8 H3).
Do res_simpl H.
Do res_simpl H2.
Save lemma_inter.

(* we follow Cantor-Russell's paradox *)

Definition khi : A0 -> (A0->Prop) = [x:A0](inter (teta x)).

Section paradox.

Variable p:Prop.

Definition u0: A0->Prop = [x:A0](khi x x)->p.

Definition x0: A0 = (psi u0).

Goal (E0 x0 x0).
Unfold x0.
Unfold x0.
Unfold psi.
Unfold psi.
Do res_simpl lemma1.
Do res_simpl per_F0.
Do res_simpl equiv_intro.
Do res_exact H1 x.
Do res_exact H1 y.
Do res_exact H0.
Save lemma4.

Goal (F0 u0 u0).
By hyps.
Do cut (F0 (khi x) (khi y)).
Do cut (equiv (khi x x) (khi y y)).
Do res_simpl H1.
Unfold u0.
Unfold u0.
Do res_simpl equiv_intro.
Exact (H4 (H3 H5)).
Exact (H4 (H2 H5)).
Do res_simpl H0.
Unfold khi.
Unfold khi.
Resolve lemma_inter.
Do res_simpl lemma_teta.
Save lemma3.

Goal (F0 u0 (khi x0)).
Unfold x0.
Unfold khi.
Unfold psi.
Do apply per_F0.
Do res_simpl H.
Do res_exact H1 (inter (F0 u0)).
By hyps.
Do res_simpl equiv_intro.

By hyps.
Do cut (equiv (u0 x) (P y)).
Do res_simpl H6.
Exact (H7 H3).
Do res_simpl H5.

Do apply lemma3.
Do cut (equiv (u0 x) (u0 y)).
Do res_simpl H5.
Resolve H7.
Do res_simpl H3.
Do res_simpl H4.

Do res_simpl lemma_inter.
Do cut (G0 (F0 u0) (teta (intro (F0 u0)))).
Do res_simpl H3.
Do res_simpl id_teta_intro.
Do res_simpl equiv_intro.

Do res_exact H1 x0.
Do res_exact H1 y0.
Do res_simpl H0.

Save lemma5.

Goal (equiv (u0 x0) (khi x0 x0)).
Do apply lemma5.
Resolve H.
Exact lemma4.
Save lemma6.
```

```
Goal (u0 x0).
By hyps.
Do res_simpl lemma6.
Exact (H1 H H).
Save lemma7.

Goal p.
Do res_simpl lemma6.
Exact (lemma7 (H lemma7)).
Save Reynolds.
```

Imprimé en France
par
l'Institut National de Recherche en Informatique et en Automatique