Resizing Rules - their use and semantic justification

Talk by Vladimir Voevodsky
from Institute for Advanced Study in Princeton, NJ.

September 11 , 2011

A resizing rule is an introduction rule which gives the user an ability to place an object which a priory belongs to a universe [U2] into a smaller universe [U1] by proving some property of this object.

Just as the univalence axiom may be seen as a generalization of extensionality axioms resizing rules may be seen as generalizations of impredicativity conditions.

The need for resizing rules first arises in the univalent approach to the formalization of mathematical notions due to the use of a defined type [hProp] instead of the declared type [Prop].

Disclaimer: I could not find an example of a situation where the use of such a rule would be completely unavoidable.

What is clear however is that without some such rules things get extremely complicated and to properly experiment with the arising complexity one needs full universe polymorphism and explicit control over universe variables in the system neither of which is currently available in Coq.

Let me start by recalling some fundamental definitions and notations of the univalent approach.

```
(* Identity Types. Idenity types are introduced in Coq.Init.Datatypes by
the line: *)

Inductive identity ( A : Type ) ( a : A ) : A -> Type :=
  identity_refl : identity _ a a .

(* We introduce our notation: *)

Notation paths := identity .
Notation idpath := identity_refl .

(* We also introduce our version of dependent sum in the form of
a record to enable the use of the machinery of canonical structures with
it: *)

Record total2 { T: Type } ( P: T -> Type ) :=
  tpair { pr21 : T ; pr22 : P pr21 } .

Implicit Arguments tpair [ T ] .
```

```
(* Contractible types *)

Definition iscontr ( T : Type ) :=
  total2 ( fun cntr : T => forall t : T , paths t cntr ) .

(* h-levels of types *)

Fixpoint isofhlevel ( n : nat ) ( X : Type ) :=
match n with
O => iscontr X |
S m => forall x x' : X , isofhlevel m ( paths x x' )
end .

(* Types of h-level 1 - "propositions" *)

Definition isaprop  := isofhlevel 1 .

Definition hProp := total2 ( fun X : Type => isaprop X ) .

Lemma proofirrelevance ( X : Type ) ( is : isaprop X ) : forall x x' : X
, paths x x' .

Lemma invproofirrelevance ( X : Type ) ( ee : forall x x' : X , paths x
x' ) : isaprop X .
```

```
(* Types of h-level 2 - "sets" *)

Definition isaset := isofhlevel 2 .

Lemma uip { X : Type } ( is : isaset X ) { x x' : X } ( e e' : paths x
x' ) : paths e e' .

Lemma invuip { X : Type } ( uipx : forall x x' : X , forall e e' : paths
x x' , paths e e' ) : isaset X .

Definition hSet:= total2 (fun X : Type => isaset X ) .
```

In the discussion of resizing rules I will use a hypothetical modification of Coq type system with the additional "pseudo-type" [Univ] whose terms are names of universes and with full universe polymorphism of all definitions.

In such a version of Coq language the definitions of [hProp] and [hSet] given above take the form:

Definition hProp (U:Univ) := total2 (fun X:U⇒isaprop X).

Definition hSet (U:Univ) := total2 (fun X:U⇒isaset X).

Hence, for two universes [U1 U2] we have different types of propositions and of sets in [U1] and [U2] respectively.

We know from standard paradoxes that we must distinguish between sets in [U1] and sets in [U1]. However the standard practice of type theory (impredicativity of [Prop]) suggests that it is safe to identify the types of propositions in different universes and also to consider the type of propositions in any universe as a member of the smallest universe [UU].

This can be achieved through the following resizing rules:

$$\textbf{RR1} \qquad \frac{\Gamma \vdash is : isaprop\, X}{\Gamma \vdash X : UU}$$

$$\textbf{RR2} \qquad \frac{U : Univ}{\vdash (hProp\; U) : UU}$$

While these rules do not make the types of propositions in different universes to be definitionally equal they allow one to consider only the type [hProp UU] in all constructions.

To illustrate how these rules affect the behavior of various constructions let me start with the [hProp] analog of [inhabited].

Informally speaking, we want, for any [U:Univ] and [X:U] to construct a term [ishinh X] of [hProp] and a function [hinhpr:X→ishinh X] which is universal among functions from [X] to propositions.

Definition ishinh (U U':Univ)(X:U) :=

forall P:hProp U', (X→P)→P .

Definition hinhpr (U U':Univ)(X:U) :=

fun x⇒fun P⇒fun f:X→P⇒f x.

Let me first make a comment on the universe parameters of ishinh.

The first of these two parameters is "polymorphic" i.e. if [U1$\subset$U2] then the universe polymorphism rules imply that for [X : U1] one has [ishinh U1 U' X = ishinh U2 U' X] where = is definitional equality.

The second universe parameter *is not* polymorphic in this sense. For [U1'<U2'] there is a function [ishinh U U1' X $\rightarrow$ ishinh U U2' X] but this function is not even an equivalence unless resizing rules are assumed.

Let us now compare other properties of this construction depending on whether we assume RR1 and RR2 or not.

Without resizing rules:

- Universe level: [ishinh U U' X] is a proposition typed to [max U U'+1].
- Universality property: projection to [ishinh U U' X] is universal for functions to types in [hProp U'].

With RR1 RR2:

- Universe level: [ishinh U U' X] is a proposition typed to [UU] for any [U U'].
- Universality property: projection to [ishinh U UU X'] is universal for functions to types in [hProp U'] for all [U'].

Let us now consider a more complicated construction - the set-quotient of a type by a relation. Here we have two different definitions. The first one is just a type-theoretic reformulation of the usual definition of quotient as the set of equivalence classes.

Let [X:U1] be a type and [R:X→X→hProp U2] a relation on [X]. Let [A:X→hProp U1'] be a subtype of [X]. We first define a property [iseq-class R A].

The first component of this property is that the carrier of [A] which is defined as [total2 (fun x⇒A x)] is non-empty. For this we use our [ishinh] relative to a universe [U2']. The other two components of this property are straightforward and we obtains a definition [iseqclass R A U2'] of being an equivalence class which depends on one non-polymorphic universe parameter.

Taking the type of subtypes which are equivalence classes we get set-quotient of [X] with respect to a [R]:

Definition setquot (U1 U2 U1' U2':Univ)(X:U1)(R:X→X→hProp U2)

:= total2 (fun A:X→hProp U1'⇒iseqclass R A U2').

Again [U1 U2] are polymorphic universe parameters and [U1' U2'] are non-polymorphic ones.

The main properties of this construction with and without resizing rules compare as follows.

Without resizing rules:

- Universe level: [setquot U1 U2 U1' U2' X R] is a set typed to [max U1 U2 U1'+1 U2'+1].
- Universality property: not provable.

With RR1 and RR2:

- Universe level: [setquot U1 U2 U1' U2' X R] is a set typed to [U1] for any [U2 U1' U2'].
- Universality property (existence and uniqueness): the projection to [setquot U1 UU UU UU X R] is universal for functions to sets in all universes which are compatible with [R] .

There is another construction of [setquot] which is computationally better than the one given above. It proceeds as follows. First one defines a type [compfun X R Y] of functions from [X] to [Y] compatible with the relation [R]. Evaluation defines a function:

ff X R U1':X→forall Y:hSet U1', ((compfun X R Y)→Y)

Using [ishinh U2'] we can define the image of this function and we set

Definition setquot' (U1 U2 U1' U2':Univ)(X:U1)(R:X→X→hProp U2)

:= Im U2' (ff X R U1').

Again we have two polymorphic and two non-polymorphic universe parameters. The properties of this construction with and without the rules RR1 and RR2 compare as follows:

Without resizing rules:

- Universe level: [setquot' U1 U2 U1' U2' X R] is a set typed to [max U1 U2 U1'+1 U2'+1].
- Universality property (existence): for functions to sets in [U1'] compatible with [R].
- Universal property (uniqueness): for functions to sets [U2'].

With RR1 and RR2 :

- Universe level: [setquot U1 U2 U1' U2' X R] is a set typed to [max U1 U1'+1].
- Universality property (existence): for functions to sets in [U'] compatible with [R] for all [U'].
- Universal property (uniqueness): for functions to sets in [U'] for all [U'].

The previous slide shows that the rules RR1 RR2 are not quite sufficient to make [setquot'] to be really well behaved since it is still typed to a universe whose level depends on [U1']. There is also an issue with how the universality property is proved for [setquot'] for universes larger than [U1']. To make this definition to work smoothly one needs yet another resizing rule:

$$(RR3) \quad \frac{U : Univ \quad \Gamma \vdash X_1 : U \quad \Gamma \vdash is : isaset\, X_2 \quad \Gamma \vdash f : surjection\, X_1\, X_2}{\Gamma \vdash X_2 : U}$$

Let me outline now a set-theoretic model which satisfies the rules RR1-RR3. Being set-theoretic, this model does not satisfy the univalence axiom. I expect it to be possible to construct a modification of the univalent model which both satisfies the univalence axiom and justifies the resizing rules but so far it is work in progress.

I would also like to make an informal conjecture that resizing rules which are semantically justifiable do not affect normalization properties of the theory.

The model takes values in a category of sets $\mathcal{C}$ defined relative to ZFC with $\omega + 1$ universes i.e. there is an infinite sequence of universes U_i such that $U_i \in U_{i+1}$ a universe U_ω which contains all of the U_i's and a universe $U_{\omega+1}$.

The set of objects of $\mathcal{C}$ is the set $U_{\omega+1}$. Morphisms between are functions between sets.

Being a category of sets this category is locally cartesian closed (lcc). Therefore the general machinery developed in my "Notes on Type Theories" for construction of models of type theory in lcc categories is applicable. This machinery requires one to fix first a "universe" in the category i.e. a morphism $p : \tilde{U} \to U$. For any such morphism one constructs a well-defined up to a canonical isomorphism contextual category (in the sense of Thomas Streicher) denoted $CC(\mathcal{C}, p)$.

Various type theoretic structures on $CC(\mathcal{C}, p)$ such as dependent sums, dependent products, Martin-Lof identity types and universes can be introduced by specifying appropriate pull-back squares based on p.

In our case we take U to be the set of isomorphism classes of well-ordered sets in U_ω and $\tilde{U}$ to be the set of pairs of such a class and an element in its canonical representative with p being the forgetting function.

The diagram required for the definition of relevant type-theoretic structures on $CC(\mathcal{C}, p)$ are all easily constructed using axiom of choice.

Important for us point is how we interpret universes of the type system. To do it we send the i-th universe of the type system UU_i to the set of isomorphism classes of well ordered sets in U_i.

Let us see now how this model justifies the resizing rules:

RR1 If [X] satisfies the condition [isaprop] then so does the set $M(X)$ which it is mapped to by the model and therefore $M(X)$ is either empty or has one element. In each case there is only one isomorphism class of well orderings on $M(X)$ and it belongs to all U_i's.

RR2 The model of [hProp UU_i] is the set of isomorphism classes of sets in U_i which satisfy [isaprop]. By the previous comment this is a two element set for any UU_i which has only one up to an isomorphism well-ordering and belongs to all UU_j's.

RR3 The fact that RR3 holds in this model follows from a simple argument based on the fact that if $f : X \to Y$ is a surjection then cardinality of Y is $\leq$ cardinality of X.

Here are few more examples of resizing rules which are validated by the well-ordered sets model and are expected to be validated by the modified univalent model:

$$(RR0) \quad \frac{U : Univ \quad \Gamma \vdash X_1 : U \quad \Gamma \vdash is : id\, X_1\, X_2}{\Gamma \vdash X_2 : U}$$

$$(RR4) \quad \frac{U : Univ \quad \Gamma \vdash X : U}{\Gamma \vdash (\Sigma_{X' : U}\ ishinh\, (\, id\, X\, X')) : U}$$

$$(RR5) \quad \frac{U : Univ \quad \Gamma \vdash X_1 : U \quad \Gamma \vdash is : weq\, X_1\, X_2}{\Gamma \vdash X_2 : U}$$

One can also consider a resizing rule of the form:

$$\frac{U_1\ \ U_2 : Univ \qquad \Gamma \vdash X_1 : U_1}{\Gamma \vdash X_1 : U_2}$$

This rule is clearly equivalent to [Type : Type] rule and therefore is not justifiable by any model with values in a consistent theory. In practice however it might be reasonable to give the user of a system an option to introduce any resizing rules just as we let the user to introduce any axioms.

There will be a full year program on Univalent Foundations at the Institute for Advanced Study in 2012-2013 co-organized by Steve Awodey, Thierry Coquand and myself. For information on the program see `http://www.math.ias.edu`.