COLIN MCLARTY

Stable surjection logic

Diagrammes, tome 22 (1989), p. 45-57
<http://www.numdam.org/item?id=DIA_1989__22__45_0>

DIAGRAMMES VOLUME 22, 1989

ACTES DES JOURNEES E.L.I.T.

(UNIV. PARIS 7, 27 JUIN-2 JUILLET 1988)

STABLE SURJECTION LOGIC

Colin McLarty

Abstract: La stable surjection logic est une extension de la logique des horn clauses en adjoutant des type introduction axioms qui font l'effet des quantificateurs existentiels moyennant des regles d'inférence toutes proches à la résolution de Prolog. En limitant les type introduction clauses dans une maniere précise on obtient une logique convenable à les theories définissables par limites projectives finies.

Prolog works with horn clauses, or definite program clauses, which we can look at as sequents in a predicate language with no connectives, written as $\Gamma \longrightarrow B$ where Γ is a list of atomic formulas and B an atomic formula. In effect this paper will extend the class of sequents we use to include existential quantifiers, but we do this without quantifiers or other connectives. The resulting logic is called stable surjection logic, or SSr-logic. This includes the logic of partial algebraic theories (also called left exact theories, and finite limit theories) as a special case. The special case is called finite limit logic, or FL-logic. These theories include most of the extensions of the theory of categories that arise in logic and computer science: categories with finite limits, or with finite coproducts, or cartesian closed, or toposes. They also include the theory of sketches, or finite sum sketches, and so on. (Finite limit logic was called "left exact logic" in [7].)

Partial algebraic theories are usually described as typed theories with typed operators and typed partial operators, where the domain of definition of any partial operator is given by an equation. For example, the theory of a category is given with two types, Ob for objects and Ar for arrows. The operators Dom:Ar$\longrightarrow$Ob, Cod:Ar$\longrightarrow$Ob take each arrow to its domain and its codomain respectively, and id:Ob$\longrightarrow$Ar takes each object to its identity arrow. Composition is partially defined for pairs of arrows. In fact, the composite f g is defined if and only if Cod(f)=Dom(g).

Michel Coste has given a syntactic presentation of these theories in [2]. His approach is, for example, to replace the composition operator by a relation "h is the composite of f and g", and use unique existential quantification to say things like "if Dom(f)=Cod(g) then there exists a unique arrow h such that h is the composite of f and g." But his approach does not use unique existential quantifiers freely. In Coste's approach an expression "$(\exists !x)A(x)$" is a well formed formula if and only if the sequent "$A(x), A(y) \longrightarrow x=y$" is provable. Wellformedness is defined

reçu le 05/09/1989

simultaneously with provability so the set of well formed formulas of a theory is generally not recursive.

The syntactic presentation offered here is based eliminates partial operators in favor of additional types. So I describe categories using three types: Ob and Ar as before, and Cp for composable pairs. There are projection operators p_1:Cp$\longrightarrow$Ar and p_2:Cp$\longrightarrow$Ar, and means for saying "for every pair of arrows f and g with Dom(f)=Cod(g) there exists a unique composable pair c with p_1(c)=f and p_2(c)=g." Then composition is a total operator from Cp to Ar.

This logic can be interpreted in the usual way in sets, or in other categories described below. The special case for partial algebraic theories can be interpreted in any category with all finite limits. In each kind of interpretation we get the desired soundness and completeness results. The central result is that any theory in the this logic has a conservative extension to first order logic.

II

A language in SSr-logic is given by:

1) A collection of types, denumerably many variables of each type, and any collection of constants of each type.
2) A collection of typed relation symbols. We write $R \subseteq T_1 x \ldots x T_n$ to show R is an n-ary relation with the given typing. But we do not have product types here. This is just a notation for typing.
3) A collection of typed operator symbols. Write $f{:}T_1 x \ldots x T_n \longrightarrow T_0$ to show the typing of an operator symbol f.

Terms and atomic formulas are defined the usual way. There are only atomic formulas. A <u>sequent</u> is an expression of the form $\Gamma \longrightarrow$ B where Γ is a list of formulas and B is a formula. A theory in the language is specified by axioms:

4) There will be a set of sequents called <u>axioms of the theory</u>.
5) Any type T may be given one <u>type instantiation axiom</u>, written

$$[E(v_1,\ldots v_n) \mid F(t,v_1,\ldots v_n)]$$

where $E(v_1,\ldots v_n)$ is a list of formulas involving at most the variables v_1 through v_n and $F(t,v_1,\ldots v_n)$ is a list of formulas with the same variables plus a variable t of type T. Intuitively, this will act like a sequent

$$E(v_1,\ldots v_n) \longrightarrow (\exists t)F(t,v_1,\ldots v_n)$$

6) We will use the <u>axioms of equality</u>

$$\longrightarrow x=x$$

$$x=y,\ A \longrightarrow A^x_y$$

where x and y are variables of the same type, A is a formula, and A^x_y is the result of substituting y for x in A.

The rules of inference for natural deduction in SSr-logic are:

7) Thinning.

$$\frac{\Gamma,\ A \longrightarrow B}{\Gamma \longrightarrow B}$$

whenever A occurs in Γ. We will also allow permutation of formulas in Γ.

8) Cut.

$$\frac{\Gamma,\ A \longrightarrow B \qquad \Gamma' \longrightarrow A}{\Gamma,\ \Gamma' \longrightarrow B}$$

10) For each type instantiation axiom

$$[E(v_1\ldots v_n) \mid F(t,v_1\ldots v_n]$$

there is an <u>instantiation rule</u>:

$$\frac{\Gamma,\ E(t_1\ldots t_n),\ F(t,t_1\ldots t_n) \longrightarrow B}{\Gamma,\ E(t_1\ldots t_n) \longrightarrow B}$$

for any terms t_1 through t_n of suitable type, where the variable t does not occur in the lower sequent. (Intuitively, any conclusion you could draw from

$E(t_1 \ldots t_n)$ plus the existence of t with $F(t, t_1 \ldots t_n)$ follows from $E(t_1 \ldots t_n)$ alone.)

These rules are sound for interpretation in sets, if domains are required to be nonempty. To allow for possibly empty domains, or for the more general interpretation in categories we need a restriction on variables, as usual in categorial logic. For the cut rule we require that, for every variable in A there is some variable of the same type in the lower sequent. For the instantiation rule we require the same for every variable in $F(t, t_1 \ldots t_n)$ except "t." The instantiation rule is intended to eliminate "t."

A derivation in a theory is the usual sort of natural deduction tree, where the top of every branch is a trivial sequent $\Gamma, A \longrightarrow A$ or a substitution instance of an axiom of the theory or an axiom of equality.

III

A finite limit type instantiation clause consists of a particular kind of type instantiation axiom plus two axioms:

$$[\ E(v_1 \ldots v_n)\ |\ p_1(t)=v_1, \ \ldots\ p_n(t)=v_n\ ] \longrightarrow E(p_1(t), \ \ldots\ p_n(t))$$

$$p_1(t)=p_1(s), \ \ldots\ p_n(t)=p_n(s) \longrightarrow t=s$$

where p_1 through p_n are any operators of suitable type, and all variables are of suitable type for the formulas. Suppose v_1 through v_n are of types T_1 through T_n respectively, while t is of type T. Then intuitively the type instantiation clause forces T to be precisely the subtype of the product $T_1 x \ldots x T_n$ that satisfies the conditions in $E(v_1 \ldots v_n)$. Call p_1 through p_n the projection operators of the clause. The last axiom says that a value t in type T is fully determined when you know the n-tuple $\langle p_1(t), \ldots p_n(t) \rangle$, so you might as well identify t with that n-tuple. The type instantiation axiom says every n-tuple satisfying E gives a value in T, and the other axiom says all values in T satisfy E. This is made precise at the end of this paper.

An <u>FL-theory</u> is a theory in SSr-logic such that every type instantiation axiom is part of a finite limit type instantiation clause. <u>FL-logic</u> is just the logic of FL-theories.

IV

For example, to axiomatize category theory in FL-logic we use:

1) Three types: Ob for objects, Ar for arrows, and Cp for composable pairs of arrows.
2) No relation symbols.
3) Four operators:

   $Dom:Ar \longrightarrow Ob$ taking arrows to their domains,
   $Cod:Ar \longrightarrow Ob$ for codomains,
   $id:Ob \longrightarrow Ar$ taking objects to their identity arrows,
   $m:Cp \longrightarrow Ar$ taking a composable pair to its composite arrow.

   We will write id_A for id(A), and abbreviate Dom(f) and Cod(g) as Df and Cg respectively when they occur in subscripts.
8) There is a type instantiation clause for the type Cp:

$$[\ Dom(f)=Cod(g)\ |\ p_1(c)=f,\ p_2(c)=g\ ]$$

$$\longrightarrow Dom(p_1(c))=Cod(p_2(c))$$

$$p_1(c)=p_1(c'),\ p_2(c)=p_2(c') \longrightarrow c=c'$$

   So Cp is intuitively the subtype of ArxAr satisfying the equation.
12) The axioms of the theory of categories are:

$$\longrightarrow Dom(id_A)=A$$

$$\longrightarrow Cod(id_A)=A$$

$$p_1(c)=f,\ p_2(c)=id_{Df} \longrightarrow m(c)=f$$

$$p_1(c)=id_{Cg},\ p_2(c)=g \longrightarrow m(c)=g$$

$$\longrightarrow Dom(m(c))=Dom(p_2(c))$$

$\longrightarrow \mathrm{Cod}(m(c))=\mathrm{Cod}(p_1(c))$

and an associativity axiom that displays the major disadvantage to FL-logic:

$$p_1(c)=f,\ p_2(c)=m(c'),\ p_1(c')=g,\ p_2(c')=h,$$
$$p_1(c'')=m(c'''),\ p_2(c'')=h,\ p_1(c''')=f,\ p_2(c''')=g$$
$$\longrightarrow\quad m(c) = m(c'')$$

where A is a variable of object type; f,g, and h are of arrow type; and c, c', c" and c"' are of composable pair type. Intuitively, c = <f,m(c')> and c' = <g,h> and so on. Actually the right hand side can be shortened slightly at the cost of readability but no matter how you do it, to talk about composition in the FL-logic theory of categories you have to posit composable pairs with all the desired components.

To see how this theory works we can prove that an arrow that acts like an identity on the left is an identity arrow. Take a constant $\underline{f}$ of arrow type and add a new axiom:

$$p_1(c)=\underline{f},\ p_2(c)=g \longrightarrow m(c)=g$$

In words, if $\underline{f}$ is the first member of a composable pair, then the composite equals the second member. The derivation starts by using cut with a trivial sequent and an axiom of equality to give

$$\mathrm{Dom}(\underline{f})=\mathrm{Cod}(id_{Df}),\ m(c)=\underline{f},\ m(c)=id_{Df} \longrightarrow \underline{f}=id_{Df}$$

Use cut with the axiom $p_1(c)=\underline{f},\ p_2(c)=id_{Df} \longrightarrow m(c)=\underline{f}$. Then use the new axiom on $\underline{f}$ to cut $m(c)=id_{Df}$. Thinning gives:

$$\mathrm{Dom}(\underline{f})=\mathrm{Cod}(id_{Df}),\ p_1(c)=\underline{f},\ p_2(c)=id_{Df},\ \longrightarrow \underline{f}=id_{Df}$$

The instantiation rule for type Cp gives:

$$\mathrm{Dom}(\underline{f})=\mathrm{Cod}(id_{Df}),\ \longrightarrow \underline{f}=id_{Df}$$

and using the second axiom of category theory to cut gives:

$$\longrightarrow \underline{f}=id_D$$

V

Consider a natural deduction system for typed first order logic, as in [6] chapter 5, or [4] modified as follows: Each quantifier rule is given for each type of quantifier. We allow derivations using sequents as axioms of the theory, so we use the cut rule restricted to cases where the cut formula is a subformula of some formula occuring in the axioms of the theory. The axioms of equality are given as:

$$\longrightarrow x=x$$
$$x=y,\ F \longrightarrow F^{x}_{y}$$

where x an y are any variables of the same type, and F is an atomic formula. These suffice for the usual first order logic with equality. If you are thinking of SSr-logic with the restriction on variables in cut and instantiation, then make the same restriction on cut in the first order logic. Such a system of first order logic will have a generalized hauptsatz: Any sequent that has a derivation in a theory has some derivation using only subformulas of formulas in the sequent and formulas in axioms of the theory.

To extend a theory in SSr-logic to one in first order logic keep the same types, relation symbols, and operators. Extend the definition of a formula to include all the first order connectives. Keep the axioms of the theory, but for each type instantiation axiom

$$[\ E(v_1 \ldots v_n)\ |\ F(t,v_1 \ldots v_n)\ ]$$

add a new sequent as an axiom of the theory:

$$E(v_1 \ldots v_n) \longrightarrow (\exists t)\&F(t,v_1 \ldots v_n)$$

where $\&F(t,v_1 \ldots v_n)$ is the conjunction all the formulas in $F(t,v_1 \ldots v_n)$.

It is easy to see that any derivation in the theory in SSr-logic can be transformed into a first order derivation, with each use of an instantiation rule replaced by a use of the existential quantifier rule and the new axiom.

Conversely consider any sequent of the SSr theory that has a derivation in the extended first order theory. By the generalized hauptsatz it has a derivation using only subformulas of formulas in the sequent and formulas in the axioms of the extended theory. By construction none of these uses any connectives except the axioms

$$E(v_1 \ldots v_n) \longrightarrow (\exists t)\&F(t,v_1 \ldots v_n)$$

We can suppose the conjunction is bracketed from the left. Thus the only formulas with connectives that can appear are of the form

$$(\exists t)\&F(t,v_1 \ldots v_n)$$

or conjunctions of initial segments of formulas in the list F from some such axiom. We abbreviate the existentially quantified formulas as "$(\exists t)F$."

In the derivation any cut with an existential formula as cut formula

$$\frac{\Gamma,\ (\exists t)F \longrightarrow B \qquad \Gamma' \longrightarrow (\exists t)F}{\Gamma,\Gamma' \longrightarrow B}$$

can either be eliminated or else rewritten in the form

$$\cfrac{\dfrac{\Gamma,\ (\exists t)F \longrightarrow B \qquad E(v_1 \ldots v_n) \longrightarrow (\exists t)F}{\Gamma,\ E(v_1 \ldots v_n) \longrightarrow B}}{\Gamma,\Gamma' \longrightarrow B}$$

where the double line indicates further cuts that do not use existential formulas as cut formulas. Here $E(v_1 \ldots v_n) \longrightarrow (\exists t)F$

is one of the axioms of the first order theory. The proof is a simple induction on the subderivation leading to $\Gamma,\Gamma' \longrightarrow (\exists t)F$. There are three cases:

1) The top sequent on the leftmost branch is trivial, having $(\exists t)F$ in the antecedent and as consequent.
2) The top sequent in the leftmost branch is an axiom of the theory.
3) At some point on the leftmost branch there is a step

$$\frac{\Gamma'' \longrightarrow F}{\Gamma'' \longrightarrow (\exists t)F}$$

In this case we can replace $(\exists t)F$ with F everywhere in the derivation, down to the cut we are working on and use F (here taken as a conjunction) as the cut formula. There are minor complications keeping track of the variable "t" but they can be handled.

Much simpler reasoning shows we can further assume no conjunction is ever used as a cut formula, and so no conjunction occurs as consequent of any sequent in the derivation.

All together we can assume the only role of quantifiers and conjunction in the derivation is in the pattern

$$\dfrac{\dfrac{\Gamma,\ (\exists t)F,\ F(t,t_1\ldots t_n) \longrightarrow B}{\Gamma,\ (\exists t)F \longrightarrow B} \qquad E(t_1\ldots t_n) \longrightarrow (\exists t)F}{\Gamma,\ E(t_1\ldots t_n) \longrightarrow B}$$

where the double line indicates steps conjoining the formulas in the list F and applying the existential quantifier rule. But then notice we can add $E(t_1\ldots t_n)$ to the antecedent of every left-most sequent above this step, and still have a derivation of the same last sequent; and then the step as shown is precisely what the instantiation rules of SSr-logic allow us to do without using the formula $(\exists t)F$ at all. So the SSr sequent, assumed derivable in the first order extension, is derivable in SSr-logic.

An interpretation of a theory in SSr-logic is given by the usual data:

1) For each type T there is a set I(T).
2) For each relation symbol $R \subseteq T_1 x \ldots x T_n$, I(R) is a subset of the product $I(T_1)x\ldots I(T_n)$.
3) For each operator symbol $f:T_1 x\ldots xT_n \longrightarrow T_0$, I(f) is a function from the product $I(T_1)x\ldots xI(T_n)$ to $I(T_0)$.

Interpretations for all terms, and extensions of atomic formulas are defined as usual.

We say a sequent $\Gamma \longrightarrow B$ is <u>true</u> in the interpretation if it is satisfied by every sequence, in the usual way. A type instantiation clause $[\ E(v_1\ldots v_n)\ |\ F(t,v_1\ldots v_n)\ ]$ is <u>sound</u> in the interpretation if and only if the first order sequent

$$E(v_1\ldots v_n) \longrightarrow (\exists t)F$$

is satisfied in the usual way.

A <u>model</u> of a theory in SSr-logic is an interpretation that makes every axiom true and every type instantiation clause sound. It is easy to see that this is precisely the same thing as a model for the first order extension of the theory. Since the first order extensions are conservative, SSr-logic is sound and complete for this interpretation.

SSr-logic can also be interpreted in any category with all finite limits where surjections are stable under pullback; and the special case of FL-logic is interpretable, sound, and complete for all categories with finite limits. (A surjection is an arrow which does not factor through any proper subobject of its codomain. See [6] p.74. In particular, any monic surjection is iso.)

Formulas of SSr-logic are given extensions in the usual way. Where the above definition calls for sets and subsets, we now call for objects and subobjects. Equations are interpreted by equalizers. Given a formula whose variables are included among $v_1 \ldots v_n$, with types $T_1 \ldots T_n$ respectively, we define the *extension* of the formula *over the variables* $v_1 \ldots v_n$ by the usual pullback. This gives a subobject of $I(T_1) x \ldots x I(T_n)$. We must mention the variables explicitly because we may need to include variables which do not actually appear in the formula. Given any list Γ of formulas all of whose variables are included among $v_1 \ldots v_n$, the *extension of* Γ over those variables is the intersection of the extensions of the formulas in Γ, over those variables. Suppose a sequent $\Gamma \longrightarrow B$ contains exactly the variables $v_1 \ldots v_n$. We say the sequent is *true* if the extension of Γ over $v_1 \ldots v_n$ is contained in the extension of B over those same variables. Notice that so far we have only used the finite limit structure of the category we interpret in.

A type instantiation axiom [$E(v_1 \ldots v_n)$ | $F(t, v_1 \ldots v_n)$] is *sound* if the top arrow in this pullback is a surjection:

$$\begin{array}{ccc} P.B. & \xrightarrow{\qquad\qquad\qquad} & [E] \\ \downarrow & & \downarrow \\ [F] \rightarrowtail I(T) x I(T_1) x \ldots I(T_n) & \longrightarrow & I(T_1) x \ldots I(T_n) \end{array}$$

where [E] is the extension of $E(t_1 \ldots t_n)$, and [F] the extension of

$F(t,t_1 \dots t_n)$ each over the obvious variables.

It is easy to see that if the category in question has stable images, that is every arrow factors into a surjection followed by a monic and surjections are preserved by pullback, then the above is equivalent to saying [E] factors through the image of [F]. And it is routine to see that the rules of SSr-logic, with the variable restriction, are sound for these interpretations. Thus they are sound and complete, since the interpretation in sets is one case of this categorial interpretation.

Consider the special case of FL-logic. We end this paper with the proof that a type instantiation clause for type T, with "t" a variable of that type:

$$[\ E(v_1 \dots v_n)\ |\ p_1(t)=v_1,\ \dots\ p_n(t)=v_n\ ]$$
$$\longrightarrow E(p_1(t),\ \dots\ p_n(t))$$
$$p_1(t)=p_1(s),\ \dots\ p_n(t)=p_n(s) \longrightarrow t=s$$

forces the type T to be interpreted as the extension of the list $E(v_1 \dots v_n)$. But this extension can also be defined by a limit (and the limit of any finite diagram can be given by such an extension, by the usual construction of limits as equalizers of products). So FL-logic as defined here has exactly the same strength as "left exact logic" in [7], and can be interpreted in any category with all finite limits. Formulas and sequents are interpreted as in SSr-logic, and a type instantiation clause is called <u>sound</u> in an interpretation if and only if the type it introduces is interpreted as the extension of the list $E(v_1 \dots v_n)$. It is routine to verify the soundness of FL-logic for this wider class of interpretations, or it follows directly from the soundness for interpretation in sets plus the argument Makkai and Reyes give for the soundness of Horn logic in [M&R p. 96] Completeness follows directly from completeness for sets.

Now the proof that an SSr sound intepretation of a type instantiation clause does force the introduced type to be the desired extension. Suppose we have an SSr interpretation, I, in which this type instantiation clause for type T is sound:

$$[\ E(v_1 \dots v_n)\ |\ p_1(t)=v_1,\ \dots\ p_n(t)=v_n\ ]$$
$$\longrightarrow E(p_1(t),\ \dots\ p_n(t))$$

$$p_1(t)=p_1(s),\ \ldots\ p_n(t)=p_n(s) \longrightarrow t=s$$

That is, the top arrow in the above pullback is a surjection, with [F] the extension of $p_1(t)=v_1,\ \ldots\ p_n(t)=v_n$. The third axiom in the clause forces the the composite arrow along the bottom to be monic. Thus [F] is a subobject of $I(T_1)x\ldots I(T_n)$. Then too, the top arrow is a pullback of a monic, so it is monic. Since it is also surjective, it is iso. Thus [E] factors through [F] as subobjects of $I(T_1)x\ldots I(T_n)$. The second axiom in the clause makes [F] factor through [E], thus they are the same subobject of $I(T_1)x\ldots I(T_n)$. On the other hand, [F] is the extension of $p_1(t)=v_1,\ \ldots\ p_n(t)=v_n$; and that is just the graph of the product arrow $\langle I(p_1),\ldots I(p_n)\rangle : I(T) \longrightarrow I(T_1)x\ldots xI(T_n)$. So [F] is isomorphic to I(T); and I(T) with monic $\langle I(p_1),\ldots I(p_n)\rangle$ is the extension [E] (up to isomorphism).

REFERENCES

[1] Barr, M., and Wells, C., Toposes, Triples and Theories (New York: Springer Verlag, 1985)

[2] Coste, M., "Une approche logique des théories définissable par limites projectives finies," Séminaire Bénabou, Université Paris-Nord, 1976.

[3] Freyd, P., "Aspects of topoi," Bull. Austral. Math. Soc. 7 (1972) pp.1-76 and 467-480.

[4] Grandy, R., Advanced Logic for Applications (Dordrecht: Reidel, 1977)

[5] Johnstone, P., Topos Theory (London: Academic Press, 1977)

[6] Makkai, M., and Reyes, G., First Order Categorical Logic, SLN 611 (New York: Springer Verlag, 1977)

[7] McLarty, C., "Left Exact Logic," Jour. Pure and Applied Algebra 41 (1986) pp.63-66.