Kleene's Slash and Existence of Values of Open Terms in Type Theory

Jan M. Smith

Department of Computer Science, University of Göteborg/Chalmers

S-412 96 Göteborg, Sweden

smith@cs.chalmers.se

1 Introduction

For most typed languages, a closed object of a type can be computed to a value, that is, to a term beginning with a constructor. An object depending on free variables, however, cannot in general be computed to a value: the normal form of such an object may begin with a selector or a variable.

In this paper we will give a necessary and sufficient condition on the type of a free variable for an open object containing that variable to be computable to a value. The condition is formulated in terms of the type theoretic interpretation of Kleene's slash [8] given in Smith [13], and corresponds to the condition in [8] satisfied by formulas which slashes themselves, that is, to formulas satisfying $C \mid C$. The existence of values of open objects is of interest, for instance, in partial evaluation [7] and pattern matching [2, 4].

When optimizing programs extracted from proofs, an important role is played by sets corresponding to Harrop formulas [6] since they are "without computational content" [12, 16]. We will define what it means for a set to be without computational content and then show that a set is without computational content if and only if it slashes itself; the sets satisfying this condition strictly contain the Harrop sets introduced in [13].

In the formulation of Martin-Löf's type theory that we are considering, the judgemental equality $a = b \in A$ is understood as definitional equality; it is by having an intensional equality that it becomes possible to obtain results concerning computations when interpreting Kleene's slash. Lambek and Scott [9] used Kleene's slash in a type theory, formulated within category theory and with an extensional equality; in that approach, results of the kind we have in this paper cannot be obtained.

In the interpretation in [13] of Kleene's slash for arithmetic, Martin-Löf's type theory without universes was used. Since the only way to obtain dependent sets when not having a universe is by the propositional equality, no really, from the computational point of view, interesting dependent sets can be constructed in the type theory considered in [13]. In this paper, which is focused on computations,

we leave out the propositional equality but add a universe and extend the results in [13] to this theory. The main point of a universe is, in this context, that it can be used to define dependent sets by recursion. However, the definitions and results below are of interest already in the case of no dependent sets. Besides the universe, the only basic sets we consider are the natural numbers N, the one element set $\top$ and the empty set $\bot$, but the definitions and results below can easily be extended to a theory which include other sets defined by strictly positive inductive definitions.

We follow the notation in [11] and, by the terminology of Martin-Löf's type theory, I will use the word "set" instead of "type," which commonly is used in connection with programming languages.

2 Definition of slash in type theory

In arithmetic, slash is a relation $\Gamma \mid A$ between a list Γ of closed formulas and a closed formula A; this is in type theory translated to a relation $\Gamma \mid t \in A$ where Γ is a context and $t \in A$ a judgement in the context Γ.

$\Gamma \mid t \in A$ is defined by an inductive definition of the same kind as the definition of the computability predicate $Comp_A(t)$ when using Tait's method [15] to prove that closed typable terms are normalizable. In fact, the computability predicate can be obtained from the slash by letting Γ be the empty context, that is, $Comp_A(t) = \mid t \in A$. The possibility of introducing a parameter similar to the context Γ in the definition of the computability predicate has also been realized by Hallnäs [5]. For a more detailed discussion of the type theoretic interpretation of slash, see [13], and for normalization proofs of Martin-Löf's type theory, I refer to [10, 14, 3].

The definition of $\Gamma \mid t \in A$ is by the following clauses.

1. $\Gamma \mid t \in A+B$ if $\Gamma \vdash t \in A+B$, $\Gamma \vdash t = inl(a) \in A+B$ for some term a such that $\Gamma \mid a \in A$, or $\Gamma \vdash t = inr(b) \in A+B$ for some term b such that $\Gamma \mid b \in B$.

2. $\Gamma \mid t \in \Pi(A, B)$ if $\Gamma \vdash t \in \Pi(A, B)$, there exists a term $b(x)$ such that $\Gamma, x \in A \vdash b(x) \in B(x)$, for all terms a, $\Gamma \mid a \in A$ implies $\Gamma \mid b(a) \in B(a)$, and $\Gamma \vdash t = \lambda x.b(x) \in \Pi(A, B)$.

3. $\Gamma \mid t \in \Sigma(A, B)$ if $\Gamma \vdash t \in \Sigma(A, B)$, there exist terms a and b such that $\Gamma \mid a \in A$, $\Gamma \mid b \in B(a)$, and $\Gamma \vdash t = \langle a, b \rangle \in \Sigma(A, B)$.

4. $\Gamma \mid t \in N$ if $\Gamma \vdash t \in N$ and $\Gamma \vdash t = \overline{n} \in N$ for some numeral $\overline{n}$.

5. $\Gamma \mid t \in \top$ if $\Gamma \vdash t \in \top$ and $\Gamma \vdash t = tt \in \top$.

6. $\Gamma \mid t \in \bot$ does not hold for any t.

When including a universe U in type theory, the definition of slash must be extended to the set U and the decoding set $Set(a)$ of an element a in U. The

definition of slash for these sets is made simultaneously, reflecting that the definitions of U and $Set(a)$ depend on each other.

7. $\Gamma \mid t \in U$ and $\Gamma \mid s \in Set(t)$ if $\Gamma \vdash t \in U$, $\Gamma \vdash s \in Set(t)$ and one of the following clauses holds.
   (a) $\Gamma \vdash t = a \widehat{+} b \in U$ for some terms a and b, $\Gamma \vdash a \in U$ and $\Gamma \vdash b \in U$, such that $\Gamma \mid a \in U$ and $\Gamma \mid b \in U$. $\Gamma \mid s \in Set(t)$ then means that $\Gamma \mid s \in Set(a) + Set(b)$.
   (b) $\Gamma \vdash t = \widehat{\Pi}(a, b) \in U$ for some terms a and b, $\Gamma \vdash a \in U$ and $\Gamma,\ x \in Set(a) \vdash b(x) \in U$, such that $\Gamma \mid a \in U$ and, for all terms u, $\Gamma \mid u \in Set(a)$ implies $\Gamma \mid b(u) \in U$. $\Gamma \mid s \in Set(t)$ then means that $\Gamma \mid s \in \Pi(Set(a), (x)Set(b(x)))$.
   (c) $\Gamma \vdash t = \widehat{\Sigma}(a, b) \in U$ for some terms a and b, $\Gamma \vdash a \in U$ and $\Gamma,\ x \in Set(a) \vdash b(x) \in U$, such that $\Gamma \mid a \in U$ and, for all terms u, $\Gamma \mid u \in Set(a)$ implies $\Gamma \mid b(u) \in U$. $\Gamma \mid s \in Set(t)$ then means that $\Gamma \mid s \in \Sigma(Set(a), (x)Set(b(x)))$.
   (d) $\Gamma \vdash t = \widehat{N} \in U$. $\Gamma \mid s \in Set(t)$ means that $\Gamma \mid s \in N$.
   (e) $\Gamma \vdash t = \widehat{\top} \in U$. $\Gamma \mid s \in Set(t)$ means that $\Gamma \mid s \in \top$.
   (f) $\Gamma \vdash t = \widehat{\bot} \in U$. $\Gamma \mid s \in Set(t)$ means that $\Gamma \mid s \in \bot$.

Note that when not having a universe, there are no dependent sets and the definition of $\Gamma \mid t \in A$ is then an inductive definition on the structure of the set A, which simply is built up from the basic sets by $+$, $\to$ and $\times$.

The main result for the interpretation of slash in type theory is the following theorem.

Theorem 1. *Let Δ be a context $x_1 \in D_1, \ldots, x_m \in D_m(x_1, \ldots, x_{m-1})$ and $d_1, \ldots, d_m$ terms such that $\Gamma \mid d_i \in D_i(d_1, \ldots, d_{i-1})$, $0 < i \leq m$. Then $\Delta \vdash a(x_1, \ldots, x_m) \in A(x_1, \ldots, x_m)$ implies $\Gamma \mid a(d_1, \ldots, d_m) \in A(d_1, \ldots, d_m)$.*

The proof is by induction on the length of the derivation of $\Delta \vdash a(x_1, \ldots, x_m) \in A(x_1, \ldots, x_m)$ and follows closely a normalization proof for type theory based on Tait's method; in fact, the context Γ will act just as a parameter in the proof and the usual normalization proof is obtained when Γ is the empty context. The proof for type theory without a universe is given in [13]; the proof with a universe is a straightforward extension of that, given the following lemma.

Lemma 1. *Let Δ be a context and $d_1, \ldots, d_m$ terms as in theorem 1. Then $\Delta \vdash A(x_1, \ldots, x_m) = B(x_1, \ldots, x_m)$, $\Delta \vdash a(x_1, \ldots, x_m) \in A(x_1, \ldots, x_m)$ and $\Gamma \mid a(d_1, \ldots, d_m) \in A(d_1, \ldots, d_m)$ implies $\Gamma \mid a(d_1, \ldots, d_m) \in B(d_1, \ldots, d_m)$.*

This lemma is needed for the induction step in the proof concerning the rule

$$\frac{a \in A \qquad A = B}{a \in B}$$

For the theory without a universe, this lemma is easy to prove by structural induction on the set A. When including a universe, the lemma is no longer trivial, but a proof of it can be obtained from the proof of proposition 1 in Coquand [3], which is a corresponding result for the computability predicate. Coquand's proof is by a computability argument and, again, Γ can be introduced as a parameter in that proof without any changes in the proof.

An alternative way of proving the lemma is to use a crucial result in [3] on the uniqueness of values together with the predicativity of Martin-Löf's set theory. Predicativity means here that if a set is introduced then its parts must already have been defined; for instance, if at a certain stage the set $\Pi(A, B)$ is introduced then we must at earlier stages know that A is a set and that $B(a)$ is a set for all $a \in A$. The predicativity can be seen directly from the rules of forming sets, but there is also a metamathematical proof of this in Aczel [1] for set theory with one universe, that is, the theory we are considering. Predicativity gives that we obtain a well-ordering from the transitive closure of the relation $<$ defined by

(i) $D < D + E$,

(ii) $E < D + E$,

(iii) $D < \Pi(D, E)$,

(iv) $E(d) < \Pi(D, E)$ for all $d \in D$,

(v) $D < \Sigma(D, E)$ and

(vi) $E(d) < \Sigma(D, E)$ for all $d \in D$.

In [3] it is shown that if two sets on constructor form are definitionally equal then the constructors are the same and the parts are definitionally equal; for instance, if $\Pi(A, B) = C(D, E)$ then C is Π, $A = D$ and $x \in A \vdash B(x) = E(x)$. Using this result and the fact that $\Gamma \mid a \in A$ implies that A is definitionally equal to a set on constructor form, we can use the above well-ordering and straightforwardly prove the lemma by transfinite induction.

From theorem 1 we get

Corollary 1 *Let $z \in C \mid c \in C$. Then $z \in C \vdash a(z) \in A$ implies $z \in C \mid a(c) \in A$.*

3 Sets without computational content

Kleene [8] showed that if a formula C satisfies the extended disjunction and existence properties, that is, for all formulas A, B and $A(x)$,

$$C \vdash A \vee B \text{ implies } C \vdash A \text{ or } C \vdash B \tag{ED}$$

$$C \vdash \exists x A(x) \text{ implies } C \vdash A(t) \text{ for some term } t \tag{EE}$$

then $C \mid C$. We will show a corresponding result for type theory; the extended disjunction and existence properties for a formula will then correspond to that

a set is without computational content. Intuitively, a set C is without computational content if $z \in C \vdash t \in A$ implies that t has a value even if A is a set with several constructors.

For the remaining of this section we will assume that every element of a set with only one constructor can be expanded to constructor form; for function sets this corresponds η-conversion.

η-rule for $\top$

$$\frac{c \in \top}{c = tt \in \top}$$

η-rule for Π

$$\frac{c \in \Pi(A, B)}{c = \lambda x.apply(c, x) \in \Pi(A, B)}$$

η-rule for Σ

$$\frac{c \in \Sigma(A, B)}{c = \langle fst(c), snd(c) \rangle \in \Sigma(A, B)}$$

In the theory we are considering, the sets with more than one constructor are disjoint unions, the set of natural numbers and the universe. Hence, we say that a set C is *without computational content* if it satisfies following clauses.

1. If $z \in C \vdash t \in A + B$ then $z \in C \vdash t = inl(a) \in A+B$ for some term a such that $z \in C \vdash a \in A$, or $z \in C \vdash t = inr(b) \in A+B$ for some term b such that $z \in C \vdash b \in B$.
2. If $z \in C \vdash t \in N$ then $z \in C \vdash t = \overline{n} \in N$ for some numeral $\overline{n}$.
3. If $z \in C \vdash t \in U$ then one of the following holds.
   (a) $z \in C \vdash t = a \widehat{+} b \in U$ for some terms a and b such that $z \in C \vdash a \in U$ and $z \in C \vdash b \in U$.
   (b) $z \in C \vdash t = \widehat{\Pi}(a, b) \in U$ for some terms a and b such that $z \in C \vdash a \in U$ and $z \in C, x \in Set(a) \vdash b(x) \in U$.
   (c) $z \in C \vdash t = \widehat{\Sigma}(a, b) \in U$ for some terms a and b such that $z \in C \vdash a \in U$ and $z \in C, x \in Set(a) \vdash b(x) \in U$.
   (d) $z \in C \vdash t = \widehat{N} \in U$.
   (e) $z \in C \vdash t = \widehat{\top} \in U$.
   (f) $z \in C \vdash t = \widehat{\bot} \in U$.

The definition of slash and corollary 1 give

Corollary 2 *If $z \in C \mid z \in C$ then C is without computational content.*

Examples of sets without computational content are sets corresponding to Harrop formulas. The Harrop sets H_Γ in a context Γ are inductively defined by

(i) $\top$ is in H_Γ,

(ii) if A is in H_Γ and $B(x)$ in $H_{\Gamma,\, x\in A}$, then $\Sigma(A,B)$ is in H_Γ,

(iii) if A is a set in Γ and $B(x)$ in $H_{\Gamma,\, x\in A}$, then $\Pi(A,B)$ is in H_Γ.

If H is a Harrop set in the empty context we simply say that H is a Harrop set.

Theorem 2 *If H is a Harrop set then $z \in H \mid z \in H$.*

Proof. By induction on the definition of a Harrop set in a context Γ it is straightforward to show that if H is a Harrop set in the context Γ then $\Gamma,\ z \in H \mid z \in H$. Alternatively, the theorem can be obtained from theorem 3 in [13], which is a corresponding result in the more general situation of no η-rules.

Corollary 2 and theorem 2 give

Corollary 3 *Harrop sets are without computational content.*

There are more sets than the Harrop sets which satisfy $z \in C \mid z \in C$ and, hence, are without computational content; two examples are $\bot \to N$ and $\Pi(N, (n)F(n))$ where

$$\begin{cases} F(0) \;=\; \top \\ F(succ(n)) \;=\; N \to F(n). \end{cases}$$

Formally, we have to use the universe to introduce the family F:

$$F(n) \;=\; Set(natrec(n, \widehat{\top}, (x,y)\widehat{N \to} y)).$$

The next theorem gives, as a special case, the converse of corollary 2.

Theorem 3 *Let C be without computational content. Then $z \in C \vdash a \in A$ implies that $z \in C \mid a \in A$.*

Proof. We first discuss the simpler case of type theory without a universe; we can in this case use structural induction on the formation of the set $A(x_1, \ldots, x_{m-1})$ in the context $x_1 \in G_1,\ \ldots,\ x_m \in G_m(x_1, \ldots, x_{m-1})$ to prove that

> *if* $z \in C,\ x_1 \in G_1,\ \ldots,\ x_m \in G_m(x_1, \ldots, x_{m-1}) \vdash a(x_1, \ldots, x_{m-1}) \in A(x_1, \ldots, x_{m-1})$ *and* $g_1, \ldots,\ g_m$ *are terms such that* $z \in C \mid g_i \in G_i(g_1, \ldots, g_{i-1})$, $0 < i \leq m$, *then* $z \in C \mid a(g_1, \ldots, g_m) \in A(g_1, \ldots, g_m)$.

I exemplify the proof for A equal to $\Pi(D,E)$ and $D + E$ and, to simplify notation, we leave out the context $x_1 \in G_1,\ \ldots,\ x_m \in G_m(x_1, \ldots, x_{m-1})$ and the corresponding substitutions of slashable terms $g_1, \ldots,\ g_m$.

Let $z \in C \vdash a \in \Pi(D,E)$. By the η-rule for Π, $z \in C \vdash a = \lambda x.apply(a,x) \in \Pi(D,E)$. Since $z \in C,\ x \in D \vdash apply(a,x) \in E(x)$, the induction hypothesis gives that $z \in C \mid d \in D$ implies $z \in C \mid apply(a,d) \in E(d)$. Hence, by the definition of slash, $z \in C \mid a \in \Pi(D,E)$.

For the +-case, let $z \in C \vdash a \in D + E$. Since C is without computational content, $z \in C \vdash a = inl(d) \in D + E$ for some term d such that $z \in C \vdash d \in D$ or $z \in C \vdash a = inl(e) \in D + E$ for some term e such that $z \in C \vdash e \in E$. Assume that the first case holds. By the induction hypothesis, $z \in C \mid d \in D$. Hence, by the definition of slash, $z \in C \mid a \in D + E$. The second case is handled in the same way.

For type theory with a universe, structural induction cannot be used since sets may then also be of the form $Set(a)$ where $a \in U$. Since C is without computational content, $z \in C \vdash A$ *set* implies that A is definitionally equal to a basic set, a set on +-form, a set on Π-form, or a set on Σ-form; hence, we can use the well-ordering introduced in the proof of lemma 1. The proof of the theorem for set theory with a universe can now proceed as above for the case without a universe, using transfinite induction instead of structural induction.

Since $z \in C \vdash z \in C$, we obtain from theorem 3

Corollary 4 *If C is without computational content then $z \in C \mid z \in C$.*

Corollaries 2 and 4 give a characterization of sets without computational content in terms of slash: a set C is without computational content if and only if $z \in C \mid z \in C$.

Acknowledgement. Normalization proofs for Martin-Löf's type theory with a universe contain subtle details and I would like to thank Thierry Coquand for many discussions on the topic and also for suggesting the alternative proof of lemma 1.

References

[1] Peter Aczel. The strength of Martin-Löf's type theory with one universe. In *Proceedings of the Symposium on Mathematical Logic, Oulu, 1974*, pages 1–32. Report No 2, Department of Philosophy, University of Helsinki, 1977.

[2] Rod Burstall. Proving Properties of Programs by Structural Induction. *Computer Journal*, 12(1):41–48, 1969.

[3] Thierry Coquand. An algorithm for testing conversion in type theory. In *Logical Frameworks*. Cambridge University Press, 1991.

[4] Thierry Coquand. Pattern matching with dependent types. In *In the informal proceeding from the logical framework workshop at Båstad*, June 1992.

[5] Lars Hallnäs. Partial Inductive Definitions. *Theoretical Computer Science*, (87), 1991.

[6] R. Harrop. Concerning formulas of the types A $\rightarrow$ B $\vee$ C, A $\rightarrow$ $(\exists x)$B(x) in intuitionistic formal systems. *Journal of Symbolic Logic*, 25:27–32, 1960.

[7] N. D. Jones, P. Sestoft, and H. Søndergaard. Mix: A self-applicable partial evaluator for experiments in compiler generation. *Lisp and Symbolic Computation*, (2):9–50, 1989.

[8] S. C. Kleene. Disjunction and existence under implication in elementary intuitionistic formalisms. *Journal of Symbolic Logic*, 27:11–18, 1962.

[9] J. Lambek and P. J. Scott. New proofs of some intuitionistic principles. *Zeit. f Math. Logik und Grundlagen d. Math.*, 29:493–504, 1983.

[10] Per Martin-Löf. An Intuitionistic Theory of Types. Technical report, University of Stockholm, 1972.

[11] Bengt Nordström, Kent Petersson, and Jan M. Smith. *Programming in Martin-Löf's Type Theory. An Introduction.* Oxford University Press, 1990.

[12] Christine Paulin-Mohring. *Extraction de Programmes dans le Calcul des Constructions*. PhD thesis, L'Universite Paris VII, 1989.

[13] Jan M. Smith. An interpretation of kleene's slash in type theory. In G. Huet, G. Plotkin, and C. Jones, editors, *Informal Proceedings of the Second Workshop on Logical Frameworks*, pages 337–342. Esprit Basic Research Action, May 1991. To appear in G. Huet and G. Plotkin, editors, Logical Frameworks, Cambridge University Press.

[14] Catarina Svensson. A normalization proof for Martin-Löf's type theory. Licentiate Thesis, Chalmers University of Technology and University of Göteborg, Sweden, March 1990.

[15] W. W. Tait. Intensional interpretation of functionals of finite type I. *Journal of Symbolic Logic*, 32(2):198–212, 1967.

[16] Yukihide Takayama. Extracting redundancy-free programs from proofs in first order arithmetic. *To appear in Journal of Symbolic Computation.*