TODO list

William Blum

June 24, 2008

1 Space complexity

1. Express the size of η-long expansion with respect to the size of the original term. Probably something like: $|\eta\text{-nf}(M : \tau)| = \mathcal{O}(|M|.|\tau|)$, which would mean a size polynomial in the size of the input (=size of term + size of its type).

2. In fact beta-normalization is PSPACE(-complete) at order 3! This means that although by Sec. 2:3 there are order-3 terms needing arbitrarily large number of variables names, in order to reduce any given order 3 terms, we only need to introduce a number of fresh variable names that is at most exponential in the size of the formula (since encoding a variable name requires space logarithmic in the total number of names).

3. At order 3, it should be possible to prove the PSPACE complexity result by showing that the maximum size of the stack of the DPDA encoding the game semantics of the term (see Lukes' paper on observation equivalence of order-3 finitary IA) is polynomial in the size of the term.

4. With safety, no names need to be introduced at any order. Thus in a certain sense we save some space, but only logarithmic in the number of variable names that we would have needed to introduced without the safety constraint.

5. When the order is not bound, normalization of safe term is non-elementary. But at this level "TIME=SPACE". So safety probably does not give anything in term of space complexity.

6. The last two points raise the following question: When the order is bound, does it really cost anything to introduce fresh variable names? For instance, is it possible to find say an order-4 (unsafe) term for which an unbounded number of variable names need to be introduced in order to be able to reduce it (using any possible reduction strategy)? No answer at the moment.

7. I expect the complexity of normalization for order-r simply-typed terms where $r \geq 4$ to be something like $(r-3)$-EXPSPACE. What would safety give use in that case?

8. Space complexity of beta normalization at order 4 for safe terms? EXPSPACE? Safety allows us to not have to encode variable names.

9. Space complexity of beta eta equality at order 4 for safe terms?

2 Miscellaneous ideas

1. A general idea to develop: "safety appears at order 3 but kicks in at order 4". For instance the simplest closed beta normal form is order 3 (kierstead term). But at order 3, the economy of variable names does not buy you anything since normalization is already PSPACE-complete! In fact we hope to prove that safety at order 4 causes a gain in term of space complexity for the normalization problem. Moreover I suspect that De Miranda's result on the nonnecessity of safety for generation word languages at order 2 can be extended to order 3.

2. find a generic encoding of any non-elementary time turing machine in the safe lambda calculus (using a non-uniform representation of numbers)

3. PB: How many variable names are required to generate all the closed beta normal form of a given type?

   In the lambda calculus, it is unbounded. Take the order-3 type

   $$\tau \cong (((o,o),o),(o^n \to o),o,o) \ .$$

   For any $n \in \mathbb{N}$ the following beta normal term:

   $$M \cong \lambda fgx.f(\lambda x_1.f(\lambda x_2.f(\lambda \ldots f(\lambda \ldots f(\lambda x_{n-1}.gx_1 \ldots x_{n-1}) \ldots)$$

   is of type τ and uses n distinct variable names. It is easy to see that any other β-normal form using less that n distinct names cannot be alpha converted to M since the function g expect as argument n variables bound at n different places.

   In the safe lambda calculus, however, the number of variable names required to generate all the closed beta normal forms is bounded by $size(\tau)-1$ where $size$ is defined by $size(o) = 1$ and $size(\tau \to \mu) = size(\tau)+size(\mu)$.

4. In a call-by-value programming language, safety does not buy you anything because the reduction strategy is such that no reduction occurs under a lambda abstraction. Consequently, when evaluating a (closed) program, β-reduction will never require variable renaming!

   There is still a potential, however, for applications in the domain of partial evaluation. Indeed, partial evaluation strategies proceed by contracting redex within the body of a lambda abstraction. The safety constraint allows you to perform these reductions without having to introduce new variable names.

3 Type inhabitation

Let us say that a type is UNSAFE if it is inhabited, and every inhabitant is unsafe; further we say that an unsafe type is UNIQUELY UNSAFE if it has a unique inhabitant.

1. All types with one atom are safe (either not inhabited at all or inhabited in the safe lambda calculus). What about multiple atoms?

   For any atom a, we can construct the types $\tau(n,a)$ as follows: $\tau(0,a) = a$ and $\tau(n+1,a) = \tau(n,a) \to a$. It is easy to see that for all $k \in \mathbb{N}$ and atom a, $\tau(2k,a)$ is not inhabited by any term (even unsafe) and $\tau(2k+1,a)$ is safely inhabited.

   Take three distinct atoms a, b and c. For any $i,j,k \in \mathbb{N}$, we write $\sigma(i,j,k)$ to denote the type

   $$\sigma(i,j,k) \cong (\tau(i,a) \to \tau(j,b)) \to (\tau(j,b) \to \tau(k,c)) \to \tau(i,a) \to \tau(k,c) \ .$$

   We now give examples of types together with some inhabitant (safe if one is known):

   (a) $\sigma(i,j,k)$ is inhabited by the term $\lambda xyzw.y(xz)$ which is unsafe if and only if $i < j$.

   (b) For any odd number k, $\sigma(i,j,k)$ is safely inhabited. Indeed if $k = 1$ then it is inhabited by the safe term

   $$\lambda xyzw^{c}.w$$

   and otherwise it is inhabited by

   $$\lambda xyzw^{\tau(c,k-1)}.w(In(k-2))$$

   where $In(k-2)$ denotes the safe inhabitant of type $\tau(c,k-2)$ (which exists since $k-2$ is odd.

   (c) The single-atom type $(\tau(1,a) \to \tau(2,a)) \to (\tau(2,a) \to \tau(4,a)) \to \tau(1,a) \to \tau(4,a)$ is inhabited by the safe term

   $$\lambda x^{\tau(1,a)\to\tau(2,a)} y^{(\tau(2,a)\to\tau(4,a))} z^{\tau(1,a)} w^{\tau(3,a)}.x(\lambda x^{a}.x)(\lambda x^{a}.x) \ .$$

   (d) $\sigma(1,2,4)$ is inhabited by the safe term:

   $$\lambda xyzw.y(\lambda g.x(\lambda a.a)(\lambda b.b))(\lambda f.f(\lambda c.c))$$

   where $x : ((a,a),(b,b),b)$, $y : (((b,b),b),(((c,c),c),c),c)$, $g : (b,b)$, $f : ((c,c),c)$. (Safe inhabitant found by Luke.)

   (e) $\sigma(1,3,4)$ is inhabited by the safe term

   $$\lambda x^{(\tau(1,a)\to\tau(3,b))} y^{(\tau(3,b)\to\tau(4,c))} z^{\tau(1,c)} w^{\tau(3,c)}.y(x(\lambda u^{a}.u)) \ .$$

(f) $\sigma(2,3,4)$ is inhabited by the safe term $\lambda xyzw.y(\lambda f.f(\lambda b.b))(\lambda g.g(\lambda c.c))$ where $f : ((b,b),b)$, $g : ((c,c),c)$.

(g) It is easy to show that $\sigma(0,2,0)$ (order 4) is not safely inhabited: Its only inhabitant up to β, η, α conversion is $\lambda xyzw.y(xz)$ which is unsafe. Same for $\sigma(0,2,4)$.

(h) Luke's example: $(a,(((a,b),b),c),c)$ (order 4). Its only inhabitant up to β, η, α conversion is $\lambda x.\lambda y.y(\lambda z.zx)$ which is unsafe.

(i) At order 3: The type $(((a,c),b),((c,b),a),a)$ is inhabited by only one term: $\lambda fg.g(\lambda x.f(\lambda y.c))$ which is unsafe.

(j) At order 3 with only 2 atoms: $(((b,a),b),((a,b),a),a)$ is only inhabited by the following family of unsafe terms:

$$\begin{array}{ll}
\lambda fg.g(\lambda x_1.f(\lambda y_1.x_1)) & \\
\lambda fg.g(\lambda x_1.f(\lambda y_1.g(\lambda x_2.y_1))) & \\
\lambda fg.g(\lambda x_1.f(\lambda y_1.g(\lambda x_2.f(\lambda y_2.x_i))) & \text{where } i=1,2 \\
\lambda fg.g(\lambda x_1.f(\lambda y_1.g(\lambda x_2.f(\lambda y_2.g(\lambda x_3.y_i))) & \text{where } i=1,2 \\
\ldots &
\end{array}$$

(k) At order 2 all closed β-normal terms are safe so there is no unsafe type at this order.

2. Luke's conjecture: "there is no uniquely unsafe type that is generated with only two atoms."

3. Conjecture: Even for terms that are not unsafe but that are intrinsically safe, the variable-capture permitting lemma still holds.

4 Conjectures

1. Proof of decompositionality of O-incrementally justified strategy. If this result can be proved then we can build a category of O-ij strategies.

2. Compilation of a safe HORS to a HOPDA (see the note of my failed attempt).

3. Observational equivalence of Safe IA_4 beta-normal form: I conjecture that it is decidable by a reduction to the DPDA equivalence problem following ideas used by Murawski et al. to show decidability of obs. equ. of $IA_3 + Y_0$. I have shown that observational equivalence is characterized by equality of the set of complete O-ij plays. Furthermore, up to order 3, because of O-i.j., the dagger operator $(\cdot)^\dagger$ for strategies can only stack up and iterate copies of σ. Indeed O is the only player who is able to change the current thread, but the O-ij condition forces him to either stay in the same thread or create a new thread, but he cannot switch to a previously created thread. This property should allow us to construct

a DPDA recognizing $\mathcal{O}(id_A)$ for any order 3-game A. It then remains to show that all the constructs of IA can be handled. Following Murawksi et al. we can decompose the application case into multiplicative application+contraction. Since we deal with beta normal forms, we consider applications of the form $f M_1 \ldots M_q$ for some variable f of order at most 3. The set $comp(\mathcal{O}(\llbracket f M_1 \ldots M_q \rrbracket))$ boils down to the computation of $comp(\mathcal{O}(\llbracket f \rrbracket^\dagger))$.

We claim that for every P/O-i.j. strategy σ the following property holds:

$$\mathcal{O}(\sigma^\dagger) \subseteq \mathcal{O}(\sigma)^\dagger$$

By unfolding the definition of dagger, this amounts to showing that for any legal position s, if all its prefixes ending with an O-question are O-i.j then for any initial move m in s, all the prefixes of $s \upharpoonright m$ ending with an O-question are O-i.j: Let q be an O-question occurring in $s \upharpoonright m$. It is easy to show by induction that for any legal position s and any initial move m, $\llcorner s \upharpoonright m \lrcorner$ is a subsequence of $\llcorner s \lrcorner$. Thus if $s_{\leqslant q}$ is O-i.j. then so is $(s \upharpoonright m)_{\leqslant q}$. Instantiating the previous equation with $\sigma \leftarrow \mathcal{O}(\sigma)$ gives:

$$\mathcal{O}(\sigma^\dagger) = \mathcal{O}(\mathcal{O}(\sigma)^\dagger)$$

Hence by the previous remark concerning the dagger operation at order 3, and following Murawksi et al.'s construction, it should be possible to build a DPDA recognizing $comp(\mathcal{O}(\llbracket f \rrbracket^\dagger))$ from a DPDA recognizing $comp(\mathcal{O}(\llbracket f \rrbracket))$.

5 Thesis TODO

1. Safe combinators. What is the counterpart of the S, K, I combinators for the safe lambda calculus? S is not safe. In fact we can construct a family of combinators S^m_m, for $m, n \in \mathbb{N}$ which can generate any term of the safe lambda calculus.

2. chapter HORS, safety, PDA

3. mention that computation tree are just a generalization of Bohm trees to term that are not necessarily in β-nf. For PCF they are a generalization of the Evaluation tree of Abramsky (see “Full abstraction of PCF” paper).

5.1 To mention in further works

1. What is the UNTYPED version of the safe lambda calculus? Which notion of order to use? Order of the minimal instantiation of its principal type?

2. Safe System F: is there a notion of safety for the second-order lambda calculus?

3. Can we encode an alternating TM in the safe lambda calculus? (I need to have a look at the encoding of TM in MLL by Mairson ("PTIME in MLL")
4. Is the safe lambda calculus of any use?
5. Proof theoretic consequences of safety?
6. At order 3, normalization in the simply-typed lambda calculus is PSPACE-complete. What about the safe lambda calculus?
7. what is the largest fragment of the simply-typed lambda calculus for whic capture permitting substitution is sound for reducing redexes? (suggested by Thorsten Altenkirch).

5.2 Application of traversal theory

1. Potential application: observational equivalence of terms that are not necessarily in beta-nf.

6 various ideas

1. Ideas to develop:

   Non-deterministic safe = non-deterministic unsafe.

   Deterministic = determinacy for the Proponent

   Deterministic *safe* = some determinacy for the Opponent (because the incremental binding of variables constrains the O-view.)

2. Is the safe lambda calculus of any use?

7 To read/learn

1. Finite model Theorem (Statman+Dowek): $\forall M.\exists^{\text{finite}}\mathcal{M}.\forall N.\mathcal{M} \models M = N$ iff $M \equiv N$