The Safe Lambda Calculus - D.Phil. Erratum

William Blum

August 21, 2021

1 August 19, 2021

1. Remove lines 12-22 page 54. (Reported by Samuel Frontull). The remark that a version of the 'no-variable capture lemma' holds without the safe typing convention 3.13 (Lemma 3.17) is false and the argument is incorrect.

2. In Section 3.1.2, in Lemma 3.26 (Properties of safe reduction), the last two bullets (iii) and (iv) (uniqueness of β_s normal form and Church Rosser property) should be moved after the definition of 'long-safety' (Definition 3.31) with the added condition that we work in the stricter long-safe calculus ($\vdash_l$) rather than the safe calculus $\vdash_s$.

   These two properties may still be true in $\vdash_s$, but the proof given in the thesis only applies to the stricter 'long' fragment of the safe calculus from Def 3.23– which is also the one used in the original TLCA paper. The relevant sentence in the proof is "Because a beta-redex can always be "widen" into consecutive beta-redexes of the shape of those in Def. 3.23", which is only true for long-safe terms.

   The proofs of (i) and (ii) are still valid for $\vdash_s$ (with respect to β_s-normal forms).

   A remark should be added to show why, unlike in $\vdash_l$, safe reduction in $\vdash_s$ does not necessarily produce β-normal forms and therefore β-normal forms and β_s-normal forms do not coincide: Samuel Frontull's counterexample $y : o \vdash_s (\lambda x^0 y^0 . x) y$ cannot be safely reduced because $\lambda y^0 . x$ is not an almost-safe application and therefore is not a safe-redex per Definition 3.21.

   After Definition 3.21 of a safe redex we could mention that when restricted to the long-safe fragment, it coincides with the definition from the TLCA paper.

   Definition 3.23 and Lemma 3.24 (contracting a safe redex preserves safety) remain valid in the relaxed variant of the safe calculus.

3. Small typo on line 2 with extra character 'm' in Definition 3.23.

4. *Remark about the two versions of the safe calculus:* The notions of 'Safe redex' and 'Safe beta reduction' defined in the thesis are defined on a more relaxed version of safe lambda calculus than the one I originally introduced in the TLCA paper. When restricted to the long-safe fragment (Definition 3.31) of the safe calculus, they coincide

with the definition from the TLCA paper: the system of rules of the calculus from the TLCA paper are precisely the 'long safe' rules the thesis.

Recall that in the long safe system, the abstraction rule requires the body of the lambda to be a safe term, unlike the more relaxed version of Table 3.1 where the body can just be an 'almost safe application'.

The rules from Table 3.1 capture the 'incremental binding of variables' aspect of the original safety restriction (i.e., if you consider AST tree where consecutive lambdas are merged into a single bulk AST node, then for safe terms you can retrieve the binder node of any variable by traversing the path from the variable to the root of the AST and stopping at the first lambda node with order greater than the order of the variable.) This is the version that gets studied in the last chapter of the thesis, and the one that gets characterized semantically by 'P-incremental' strategies.

Observe that a long safe term that is not safe can always be turned into an eta-equivalent safe term: by just eta-expanding the body of abstractions that are *almost* safe applications. Such eta-expansion has the side-effect of instantiating fresh variable names, which partially defeats the benefit of the safety restriction. So in a way, the safety terms from Table 3.1 are not 'as safe' as the one from Def 3.31.

Yet, it is still the preferred version of the calculus in the thesis because, contrary to the stricter long variant, this definition is sound with respect to eta-reduction (Proposition 3.37). This becomes important in Chapter 6 where I look at the game semantics model of the calculus. Such models are extensional (eta-expansion or eta-reduction do not alter the semantics of a term) and therefore they cannot possibly capture syntactic differences between long-safe term and safe terms. Hence, the game semantic characterization result from Chapter 5 (Theorem 5.19) concerns the more relaxed version of the calculus from Table 3.1 rather than just long-safety. The characterization result, however, implies that long safe terms are also denoted by P-incremental strategies.

The variable-capture avoidance guarantees from 'Lemma 3.15' still apply to both versions of the calculi. But if one looks for a safe calculus with a sounder notion of safe-reduction, as in 'does not get stuck at terms that could be further reduced with regular β-reduction', then long-safety is the one to pick. On the other hand, to study the denotational game semantics of the language, the definition from Table 3.1 is more appropriate.

2 Acknowledgment

Credits to Samuel Frontull who reported conter-examples that led to this erratum.

3 Reference

[1] D.Phil. Thesis: The Safe Lambda Calculus 2009, Oxford University Research Archive `https://ora.ox.ac.uk/objects/uuid:537d45e0-01ac-4645-8aba-ce284ca02673/download_file?file_format=pdf&safe_filename=Blum%2Bthesis&type_of_work=Thesis`

[2] TLCA paper: The Safe Lambda Calculus (extended version) With C.-H. Luke Ong. Technical report. `https://william.famille-blum.org/research/tlca07-long.pdf`