Report on a paper submitted to
the Journal of Functional Programming

entitled

Fair Enumeration Combinators

by Max New, Burke Fetscher, Jay McCarthy and Robert Bruce Findler.

General assessment of the paper

The paper proposes a new framework of *enumerations*, i.e. bijections between natural numbers and elements of some given type, with an application to property-based software testing. A notion of *fairness* is introduced, meant to capture the intuitive balance in compound enumeration exploration, supported with a few natural examples of fair and unfair enumerations. The supplementary material includes a Coq model for a limited subset of the core library combinators, along with additional tests and proofs of correctness. Finally, the developed implementation is compared with several other enumeration techniques on a Redex automated testing benchmark[1]. Fair enumerations are argued to constitute effective testing tools, contrary to their unfair counterparts.

General comments

The article starts with a brief introduction to the subject of property-based software testing with an emphasis on enumeration-based techniques. The main object of interest, i.e. *enumerations*, is introduces on an informal, programmer-like level. This informal narration of the paper continues unfortunately to explaining the main components of the framework, including BELOW/E, OR/E and other key combinators. Such a presentation, without a precise mathematical treatment, makes it quite difficult to read the paper. The informal, almost storytelling-like narrative is obfuscating the presented ideas – an example being the generalisation of Szudzik's pairing function to higher dimensions. Instead of a mathematical formula or pseudo-code with a correctness comment, the reader is given a two paragraph intuition behind the generalisation idea (see page 10, section 5, paragraphs 5 and 6).

The following section 6, contains a formal model of the enumeration framework. The model occupies two full pages and stands in contrast to the previous exposition. Readers unfamiliar with Coq or Racket are most likely not going to follow the presentation at this point. Despite some hopes, 12 pages into the paper, the definition of fairness is still not given. Moreover, the reader is asked to ignore the fairness part of the model (see section 6, paragraph 2, line 3). Such an exposition is not only bad style, but strengthens the feeling of an informal discussion at this point, rather than a rigorous treatment of the considered semantics. Fortunately, the given model is generally commented, hence it is possible to grasp the main idea of the intended framework's inner workings. That being said, some parts of the model are not discussed, e.g. the meaning (semantic) of a substitution applied to an enumeration (see rule FIX/E).

Paragraph 9 in the same section, starts with an explanation that the actual Coq model is simpler than the model presented in the paper. A detailed difference between the two is not given, although the Coq model is stated to suffice for proving some results about fairness. The expressive power of the Coq model is not further discussed. Without it, the following theorems, as stated, are unverifiable to the reader. The results themselves are a rather simple and intuitive exercise. The supplied Coq code does indeed compile although, interestingly, not with the newest 8.5pl2 Coq version.

On page 15, the anticipated definition of fairness is given in terms of the trace combinator. The definition depends on the semantics of TRACE/E, however has a nice intuition in terms of

[1] http://docs.racket-lang.org/redex/benchmark.html

equilibrium points. The following results are easy consequences of the assumed semantics, in particular Szudzik's pairing function and the alternating OR/E combinator.

In the reminder of the paper, some empirical facts are reported, comparing different approaches and techniques of enumeration-based testing. Assumed methods of different enumeration tactics are tested against a specific automated testing benchmark. A partial ordering involving the subsets of detected bugs is introduced, creating a final ranking of approaches. According to the test results, ad-hoc and fair enumerations found more bugs than other, unfair ones. The experiment is supported with detailed figures and suitable statistical data.

It is debatable whether fairness, as formally defined, is a suitable measure of the intuitive fairness as discussed throughout the paper. An enumeration e is said to be f-fair, if f defines the equilibria of e. It means that each fair enumeration is f-fair for some recursive function f. In particular, an enumeration that reaches equilibria in $A(n, n)$ for $n \in \mathbb{N}$ where A is the Ackermann function, is also fair. Clearly, this is an exaggeration but highlights the fact that fair (intuitively speaking) enumerations ought to be f-fair for some slowly growing function, e.g. polynomial. This is precisely the case for enumerations explicitly given in the paper, where each of them is f-fair for some low degree polynomial f.

It is not discussed in the paper what kind of restrictions were actually imposed on the fair enumerations in the conducted experiment. In other words, how fair were the enumerations in question. The above discussed subtlety implies that it would be theoretically possible to design a set of fair enumeration with arbitrary fast growing fairness functions and draw the precise opposite conclusions as in the paper – the unfair enumerations could be indistinguishable from the fair ones in any practical amount of time and space. For that reason, I suggest a stronger criterion for fair enumerations incorporating the asymptotic growth rate of fairness functions. Such a definition would also add precise meaning to the informal notions of 'mildly' or 'brutally' (un)fair functions.

Conclusions

The paper presents a new library for enumeration-based testing. Unfortunately, the paper is written in an intuitive narration, with little solid and precise mathematical treatment in the paper itself. Interesting formalisms are available in the supplementary material. Without them, the paper cannot be considered complete. The definition of fairness is, as discussed earlier, far from being a robust theory and presents little interests to the reader unfamiliar with either Coq or Racket. Readers interested in functional programming or enumeration-based testing in general, as myself, will find the paper hard to read and appreciate, without actually using the developed library. In conclusion, the submission presents a nice library for enumeration-based testing, but unfortunately little beyond that.