Posudek diplomové práce

Matematicko-fyzikální fakulta Univerzity Karlovy

Autor práce	Lukáš Rozsypal
Název práce	Kampa: from a prototype to practical usability
Rok odevzdání	2023
Studijní program	Informatika **Studijní obor** Softwarové systémy

Autor posudku	Tomáš Petříček **Role** Oponent
Pracoviště	Department of Distributed and Dependable Systems

Text posudku:

SUMMARY

The thesis presents an experimental general-purpose programming language Kampa that has been developed by the author. The language has been previously introduced in a Bachelor thesis by the author and so the present thesis focuses on new aspects of the language design, new language features and implementation imrpvements in the new version.

The main topics discussed in the thesis are (i) redesign of the language type system, including improved type inference and new design for generic and dependent types, (ii) support for coroutines and the development of iterator and asynchronous programming libraries that this enables, (iii) new language implementation based on intermediate byte code and (iv) other library developments that are also used as a case study for the practical usability of the language.

EVALUATION

The development of a general-purpose programming language is a major undertaking and the author has been able to complete this task to a very high degree. The libraries developed for the language, including that for asynchronous programming, require a comprehensive and complete language and the author has been able to complete those successfully. The thesis itself is mainly descriptive. It describes the language design in a well-written English, but it does not attempt to provide e.g., a more formal specification or more comprehensive discussion of the design rationale. This is not necessarily a problem as doing so was not the focus of the project, but there are many ways in which this could be made even better.

For example, the design of the dependent types (used also for generic function types) is interesting enough to deserve a more detailed explanation. Similarly, the design of the immutable/readonly/mutable qualifiers is also quite interesting - but it would be even more interesting if the thesis made some suggestions (even if without formal proofs) about what guarantees the design gives to the programmer.

In summary, the thesis has a somewhat broader and less detailed focus (which would perhaps make it more academically interesting), but it clearly presents a significant body of thorough work and shows authors good understanding of the problem of programming language design.

COMMENTS

The thesis claims it focuses on "clean" language design and "simple and orthogonal" design. Those are notoriously elusive terms - it would be interesting to see if the author can come up with some more quantifiable ways of talking about those properties.

The claim that Java "does not allow using these tools [records, references, arrays] separately" is perhaps a bit misleading. Java is designed with a different set of primitive tools in mind (objects) and so it may be unfair to criticise it from the perspective of a different paradigm than its own. I think it can still be criticised - but using a more refined argument.

QUESTIONS

When discussing the Box type, you note that it is not possible to take a pointer (box) of an existing value without copying. What exactly is copied? Does this meen a deep copy (including copying values of all nested Box types)?

How is garbage collected in the implementation? Is it the case that the memory representation in the interpreter (section 3.4) is such that all garbage collection can be delegated to the (Java) runtime in which the interpreter is implemented? Are there any limitations of this approach?

Práci doporučuji k obhajobě.

Práci nenavrhuji na zvláštní ocenění.

Pokud práci navrhujete na zvláštní ocenění (cena děkana apod.), prosím uveďte zde stručné zdůvodnění (vzniklé publikace, významnost tématu, inovativnost práce apod.).

Datum 27. August 2023 **Podpis** T. Petříček