

Report on the “*Habilitation à Diriger de Recherches*” thesis

*Preuves automatiques de non-terminaison à base de boucles compressées*

submitted by

Etienne Payet, Université de la Réunion

Thomas Jensen

IRISA/CNRS

The research reported in Etienne Payet’s habilitation thesis is an investigation into the problem of analysing the termination properties of computations. Deciding the termination of a program is a problem that is of importance in many areas and it is well known that this cannot be done fully automatically. Instead, various approaches based on static analysis have been proposed. These approaches admit the principle that the analysis may fail to prove termination and declare that it doesn’t know. This is troublesome, as the programmer does not know whether this verdict was given because the program may not terminate or because the analysis was not powerful enough. It is in this case that the work of Payet on non-termination analysis has an important role to play, by trying to prove the dual property of non-termination, thereby narrowing the grey-zone where the analysis is inconclusive.

After a brief introduction to termination analysis in Chapter 1, Chapter 2 presents the first major contribution: a non-termination analysis for logic programs. The method is based on using an unfolding of the logic program under analysis, that has equivalent non-termination behaviour. From the unfolding of the program, the analysis extracts classes of atomic queries that are non-terminating for the program. In its simplest form, the analysis searches for binary programs where the body is more general than the head; this is guaranteed to create non-terminating behaviour. This analysis being rather weak, it is extended with the notion of ‘derivation neutral’ arguments to enable a stronger notion of ‘being more general’. The correctness of the analysis is proved formally and a series of examples complements well the formal definitions. In addition, the approach has been evaluated experimentally on a substantial series of logic programs, showing that the extended analysis gives quite precise information about the termination behaviour of standard logic programs.

Chapter 3 addresses the problem of analysing the non-termination behaviour of constraint logic programs. The goal is still to produce queries for which there is a non-terminating computation. The idea is to extend the somewhat syntactic approach developed for logic programs to a more logical characterisation of the derivation neutral arguments, involving the underlying constraint domain. This allows e.g., to take into account numerical properties when producing non-terminating queries. Payet proposes an algorithm for computing such queries and presents the results of running an experimental implementation on a couple of simple constraint logic programs.

The problem of proving non-termination can also be posed in the general setting of term rewriting systems (TRS), which is the object of Chapter 4. Again, the approach is to adapt the unfold-and-infer developed in the previous chapters for (constraint) logic programs to the term rewriting

**UMR IRISA**

setting. From a given TRS, the analysis will derive a set of additional rewrite rules on which it is easier to identify looping behaviour (essentially by semi-unifying left-hands and right-hands of rules). The problem with this approach is that a naïve generation will produce far too many rules so the contribution of Payet is to propose methods to eliminate rules that are useless for proving non-termination. This improved technique is evaluated experimentally on a set of benchmarks issued from the 2007 international Termination Competition and is observed to identify most of the TRS that do not terminate due to loops.

Chapter 5 describes an investigation into the area of non-termination analysis of object-oriented software in the form of Java programs. This might seem a departure from work in the previous chapters but is actually relies in a substantial manner on the previous work. Indeed, the approach investigated by Payet uses the idea of translating a Java program into a representation in constraint logic form and then deducing non-termination properties of the Java program from a non-termination analysis of the constraint logic program. The translation in question is the path-length analysis developed by Payet *et al.* for analysing termination of Java byte code programs, which derives a CLP that relates the input values of a basic block to its output values. The analysis has a number of restrictions (*eg.*, it is limited to linear relations) that means that certain programs are not analysed exactly. The treatment of data stored in the heap is also an issue on which further development could be undertaken. Nevertheless, experiments show that the analysis is useful for reducing the (complex) task of proving termination, by eliminating from consideration those clauses for which it can be proved that they will *not* be able to contribute to a termination proof.

-o0o-

My overall evaluation of the habilitation thesis presented by Etienne Payet is that it constitutes a detailed research into the issue of non-termination analysis, beginning with the foundational analysis of logic programs on which analyses for other language paradigms have been constructed. This gives a satisfactory mixture of variety and coherence to the work at the same time.

The work presented in this thesis has been well published. In particular, three journal articles have been produced and published in respective journals (in particular, one has appeared in the very prestigious ACM TOPLAS journal).

The research has been put on a solid formal foundation but it should be stressed that Payet does not neglect the practical aspects of the proposed analysis: all the analyses come with an experimental evaluation.

The format of the thesis clearly reflects that it is composed from four previously published articles. While this is fine for a habilitation thesis, I would encourage that a little more harmonisation of notation and cross-referencing be done on the document if it is to be published as a monograph. More details of the path-length analysis of Java byte code should also be included to make such a monograph self-contained.

These stylistic remarks are however minor. The thesis clearly documents that Etienne Payet has provided a series of results on non-termination analysis, which together constitute a contribution that fully justifies that he be authorized to defend his habilitation thesis.

Rennes, 14 October 2009

Thomas Jensen, DR CNRS