Commutation

Proposal for a Course in the Advanced Track of ISR 2017

Julian Nagele Vincent van Oostrom

January 31, 2017

University of Innsbruck, Department of Computer Science, Austria

Content Commutation is the basis of many results in computer science, both within rewriting, e.g. confluence, standardisation, normalisation, and relative termination, and outside of it, e.g. sorting, correctness of program transformations [2] (cf., e.g., the work of Schmidt–Schauß), and bisimulation up-to [3]. Motivated by the latter examples, we show how they can be reformulated as commutation problems for corresponding rewrite systems. To automate deciding commutation of given rewrite systems we proceed in two steps. First, we show how to reduce the *global* property of commutation to various *local* commutation properties, e.g. local decreasingness, lazy commutation, and local Dyck. Next, we provide various ways to establish such local commutation properties of *term* rewrite systems, by means of an analysis of the critical pairs between the term rewrite systems. We illustrate automation by reformulating several concrete problems as commutation problems, which are then given to automatic commutation tools.

Organisation The course will be partitioned into 4 sessions of 45 minutes:

1. Motivating examples. Commutation as a property of abstract rewrite systems. Commutation via local commutation.
   Lecturer: Vincent van Oostrom

2. Local commutation via critical pair analysis of term rewrite systems. Various critical pair criteria sufficient for local commutation.
   Lecturer: Julian Nagele

3. Commutation tools for first-order term rewrite systems. Illustration of the tools on motivating first-order examples.
   Lecturer: Julian Nagele

4. Limitations and possible extensions of current commutation techniques. Illustration of commutation techniques on motivating higher-order examples.
   Lecturer: Vincent van Oostrom

Each session will have its own exercises, with the hands-on exercises being located in the 2nd and 3rd session.

Learning Goals The main goal is to acquaint students with the state of the art in commutation, in particular they should gain insight into automatable criteria for showing commutation of first-order term rewrite systems. Moreover we aim to enable students to recognise commutation properties when they appear in various settings and contexts.

Prerequisites We assume basic knowledge of rewriting, More precisely students should be familiar with the first seven chapters of the book by Baader and Nipkow [1].

More advanced notions and techniques in rewriting not covered there, such as strategies, higher-order term rewriting, and decreasing diagrams, will be explained in the course, based on the book by Terese [4].

To partake in the hands-on exercises students should have OCaml[1] (version 3.12 or newer) installed on their laptop.

References

[1] F. Baader and T. Nipkow. *Term Rewriting and All That.* Cambridge University Press, 1998.

[2] G. Huet. Confluent reductions: Abstract properties and applications to term rewriting systems. *Journal of the Association for Computing Machinery*, 27(4):797–821, 1980.

[3] D. Pous. Up-to techniques for weak bisimulation. In *Automata, Languages and Programming, 32nd International Colloquium, ICALP 2005, Lisbon, Portugal, July 11-15, 2005, Proceedings*, volume 3580 of *Lecture Notes in Computer Science*, pages 730–741. Springer, 2005.

[4] Terese. *Term Rewriting Systems*, volume 55 of *Cambridge Tracts in Theoretical Computer Science.* Cambridge University Press, 2003.

[1] https://ocaml.org/