The Rewrite Engines Competitions: A RECtrospective

Francisco Durán
Univ. of Málaga, Spain

Hubert Garavel
INRIA Grenoble, France
Rewrite Engines Competition (REC)

- Side event of **WRLA** (Workshop on Rewriting Logic and its Applications)

- 4 editions of REC so far:
  - **REC1** (2006)  G. Denker, C. Talcott, G. Rosu et al.
  - **REC2** (2008)  F. Durán, M. Roldán, E. Balland et al.
  - **REC3** (2010)  F. Durán, M. Roldán, J.C. Bach et al.
  - **REC4** (2018)  H. Garavel, M.A. Tabikh, I. Arrada
Term rewrite systems

- A simple, yet powerful model of computation
  - variable and function symbols
  - rewrite rules: \( \text{add}(x, \text{succ}(y)) \rightarrow \text{succ}(\text{add}(x, y)) \)
  - conditional rules: \( \text{exp}(x, 0) \rightarrow 1 \) if \( \text{neq}(x, 0) \)

- Different levels of complexity
  - basic features: many-sorted, confluent, terminating
  - advanced features:
    - nondeterministic rewriting
    - axioms (associativity/commutativity)
    - rewrite strategies
Implementations of term rewriting

■ Term rewrite engines
  ► ASF+SDF, CafeOBJ, Maude, Rascal, Stratego/XT
  ► support of advanced rewriting features

■ Functional languages
  ► Clean, Haskell, OCaml, Opal, SML

■ Algebraic languages for concurrency
  ► LOTOS, mCRL2

■ Imperative and object-oriented languages
  ► LNT, Scala, Tom
Evolution of competition scope

- First editions **REC1, REC2, REC3**
  - focus on term rewrite engines
  - small number of tools: ASF+SDF, Maude, Stratego/XT
  - exploration of advanced features

- Latest edition **REC4**
  - encompass all implementations of term rewriting
  - large set of languages/compilers/interpreters
  - restriction to **basic features**
    deterministic, confluent, terminating specifications
## Participating tools in the REC editions

| language (tool)   | web site                                      | REC1 | REC2 | REC3 | REC4 |
|------------------|-----------------------------------------------|------|------|------|------|
| ASF+SDF          | [http://www.meta-environment.org](http://www.meta-environment.org) | ×    | ×    |      |      |
| CafeOBJ          | [http://cafeobj.org](http://cafeobj.org)      |      |      |      | ×    |
| Clean            | [http://clean.cs.ru.nl](http://clean.cs.ru.nl) |      |      |      | ×    |
| Haskell (GHC)    | [http://www.haskell.org](http://www.haskell.org) |      |      |      | ×    |
| LNT (CADP)       | [http://cadp.inria.fr](http://cadp.inria.fr)   |      |      |      | ×    |
| Lotos (CADP)     | [http://cadp.inria.fr](http://cadp.inria.fr)   |      |      |      | ×    |
| Maude            | [http://maude.cs.illinois.edu](http://maude.cs.illinois.edu) | ×    | ×    | ×    | ×    |
| mCRL2            | [http://www.mcrl2.org](http://www.mcrl2.org)   |      |      |      | ×    |
| OCaml            | [http://www.ocaml.org](http://www.ocaml.org)   |      |      |      | ×    |
| Opal (OCS)       | [http://github.com/TU-Berlin/opal](http://github.com/TU-Berlin/opal) |      |      |      | ×    |
| Rascal           | [http://www.rascal-mpl.org](http://www.rascal-mpl.org) |      |      |      | ×    |
| Scala            | [http://www.scala-lang.org](http://www.scala-lang.org) |      |      |      | ×    |
| SML (MLton)      | [http://www.mlton.org](http://www.mlton.org)   |      |      |      | ×    |
| SML (SML/NJ)     | [http://www.smlnj.org](http://www.smlnj.org)   |      |      |      | ×    |
| Stratego/XT      | [http://www.metaborg.org](http://www.metaborg.org) | ×    | ×    |      | ×    |
| TermWare         | [http://gradsoft.ua/index_eng.html](http://gradsoft.ua/index_eng.html) |      |      |      | ×    |
| Tom              | [http://tom.loria.fr](http://tom.loria.fr)     | ×    | ×    |      | ×    |
| TXL              | [http://txl.ca](http://txl.ca)                |      |      | ×    |
Evolution of competition procedures

First edition **REC1**
- problems coded in the input language of each tool
- programming skills played a major role

Next editions **REC2** and **REC3**
- problems coded in a common language **REC-2008**
- manual or automated translations from **REC-2008** to the input language of each tool

Latest edition **REC4**
- problems coded in a common language **REC-2017**
- 17 automated translators developed for **REC-2017**
Sample REC-2017 specification

**REC-SPEC** simple

**SORTS** % abstract data domains

  Bool Nat

**CONS** % primitive operations

  true : -> Bool
  false : -> Bool
  zero : -> Nat
  succ : Nat -> Nat

**OPNS** % defined functions

  and : Bool Bool -> Bool
  plus : Nat Nat -> Nat

**VARS** % free variables

  A B : Bool
  M N : Nat

**RULES** % function definitions

  and (A, B) -> B if A ->< true
  and (A, B) -> false if A ->< false
  plus (zero, N) -> N
  plus (succ (M), N) -> succ (plus (M, N))

**EVAL** % terms to be evaluated

  and (true, false)
  plus (succ (zero), succ (zero))

**END-SPEC**
Available REC benchmarks

- A growing collection of benchmarks
  - divided into 4 categories
  - certain benchmarks are parameterized
  - at present: 43 models, 85 instances

| category                        | REC1  | REC2  | REC3  | REC4  |
|---------------------------------|-------|-------|-------|-------|
| source language                 | tool-specific | REC-2008 | REC-2008 | REC-2017 |
| unconditional term rewrite systems | (5) 7 | (5) 12 | (7) 26 | (19) 43 |
| conditional term rewrite systems | (9) 25 | (8) 18 | (6) 17 | (24) 42 |
| rewriting modulo equations      | (4) 9  | (4) 6  | (4) 6  | (0) 0  |
| rewriting modulo strategies     | (0) 0  | (1) 1  | (1) 3  | (0) 0  |
| TOTAL                           | (18) 41 | (18) 37 | (18) 52 | (43) 85 |
Conclusion

■ Rewrite Engines Competitions
  ▶ 4 editions so far, gradually evolving over years
  ▶ performance assessment of rewriting implementations
  ▶ a collection of benchmarks in REC-2017 language

■ Future steps
  ▶ 4 tools (at least) being enhanced following REC4
  ▶ new benchmarks under development
  ▶ improvements to the REC-2017 language
    • simpler notations ("=" rather than "-><-", etc.)
    • predefined libraries: Bool, Nat, Int, etc.
    • meta-programming using Awk scripts