Alt-Ergo*

Alt-Ergo is a SMT solver developed by Université Paris-Sud and the OCamlPro company. Since its first release in 2006, Alt-Ergo has been mainly designed for discharging proof obligations generated by software development frameworks. In particular, some of its features directly come from the Why/Why3 platforms for deductive program verification (also developed at Université Paris-Sud) which provide a rich specification language based on a polymorphic type system à la ML.

In order to directly handle proof tasks from this system, Alt-Ergo has a native input language for a polymorphic first-order logic and built-in capabilities to reason about parametric user-defined data-structures. The solver also supports quantifiers reasoning (based on E-matching), the free theory of equality, the theory of (integer and rational) arithmetic, enumerations, record data types and the theory of arrays. Recently, a procedure for the theory of floating-point arithmetic has been integrated. Last but not least, Alt-Ergo integrates a powerful mechanism for reasoning about associative-commutative function symbols.

In order to work closely with the SMT community, we have implemented in Alt-Ergo 2.2 a new frontend for a polymorphic conservative extension of the SMT-LIB 2 standard, presented at SMT2018[3].

Alt-Ergo is written in OCaml. Each module is implemented in a modular way as a set of (parameterized) modules. Most of the code (except very few parts like the CDCL algorithm and hashconsing used for maximal sharing) is written in a purely applicative programming style.

Alt-Ergo provides decision procedures for reasoning in the combination of the following built-in theories: the free theory of equality with uninterpreted symbols, linear arithmetic over integers and rationals, fragments of non-linear arithmetic, polymorphic functional arrays with extensionality, enumerated datatypes, record datatypes, associative and commutative (AC) symbols, floating-point arithmetic [5], and fixed-size bit-vectors with concatenation and extraction operators. Universal quantifiers are handled using the usual e-matching technique, extended to deal with type variables.

The main differences of Alt-Ergo from other SMT solvers are:

- Shostak combination. The algorithm which implements the equation reasoning for convex theories is reminiscent of Shostak combination called CC(X) [2]. Its extension to handle associative and commutative user-defined symbols is called AC(X) [1].

- Non-linear arithmetic. To reason about non-linear integer arithmetic, Alt-Ergo implements an algorithm which relies on the extension and collaboration of the AC(X) framework and interval calculus to handle NIA axioms in a built-in way [4].

- Polymorphism. The historical input language of Alt-Ergo is a first-order logic with some built-in theories and polymorphic data types. We recently added a partial support for the SMT-LIB 2 standard extended with ML-style prenex polymorphism[3].

- Tableaux-like and CDCL procedures. Since its first versions, Alt-Ergo integrates a Tableaux like SAT solver modulo theories implemented in a purely functional programming style. We recently worked on a new SAT solver that combines the efficiency of a CDCL engine with the nice properties of the Tableaux-like solver (construction of a small Boolean model, interaction with theories and instantiation engine using a small set of literals and terms).

*This work is partially supported by VOCaL (ANR-15-CE25-0008) and SOPRANO (ANR-14-CE28-0020) ANR projects.
• Graphical User Interface. To the best of our knowledge, Alt-Ergo is the only SMT solver equipped with a graphical user interface[6].

Current team members
• Guillaume Bury, OCamlPro SAS
• Sylvain Conchon, LRI, Université Paris Sud, CNRS, INRIA Saclay–Île-de-France
• Albin Coquereau, OCamlPro SAS
• Mohamed Iguernelala, OCamlPro SAS
• Steven de Oliveira, OCamlPro SAS

Former contributors
• Francois Bobot, CEA List
• Evelyne Contejean, CNRS, INRIA Saclay–Île-de-France, LRI Université Paris Sud
• Claire Dross, AdaCore
• Stephane Lescuyer, Prove&Run
• Alain Mebsout, OCamlPro SAS
• Other contributors (Post-doc, internship) : David Baudet, Denis Cousineau, Kylian Ji, Frédéric Lang, Arthur Milchior, Samia Nabili, Samuel Risbourg, Thibaut Tachon

For more informations visit our website at https://alt-ergo.ocamlpro.com/ and our public repo on github : https://github.com/OCamlPro/alt-ergo

References