Theory and Practice of SMT Solving - Seminar report

Luís Felipe Ramos Ferreira

The paper presents a novel tool called Z3-Noodler, which is a fork of the Z3 SMT theorem prover where the main string solver is substituted by Noodler, a string theory solver based on the *stabilization-based procedure* for solving string formulae. The procedure depends on the usage of nondeterministic finite automata (NFA), and so the authors used a C++ library caled Mata for handling operations over this kind of structure.

Since the solver was developed over Z3, several tools inside it's environment were used, such as the SMT-LIB parser, formula preprocessing, linear integer arithmetic solver, SAT solver, and also the whole DPLL($\mathcal{T}$) architecture. In regards of the formula rewriting step, some modifications were made in order to benefit the decision procedure used in Noodler. Some rewritings made by default were not benefitial for the new solver, so they were disabled. For example, rules that state the membership of a string term to a regular language are efficiently handled by Noodler, but they were initially preprocessed and changed by the default Z3 solver.

In general, Noodler interacts with Z3 as follows. First, it receives a satisfying boolean assignment from Z3's SAT solver and then removes useless assignments for dealing with a simpler formula. Noodler then converts the conjunction of string literals to a LIA constraint and feeds it back to the solver as a theory lemma. Noodler also implements, as aforementioned, a decision procedure based on NFAs called *stabilization-based procedure*.

Regarding the concepts around string theory handled by the solver, Noodler uses different ideas for each step. Noodler applies *Axiom Saturation*, which saturates the input formula before the SAT solver starts generating assignments. This makes the solver avoid checking SAT assignments that are clearly false. Another saturation, now regarding string predicates, is to replace some formulas with equivalent ones that are more suitable for the approach, such as length/regular constraints. One example is the replacement of $\neg\text{contains}(s, \text{"}abc\text{"})$ by $s \notin \Sigma^* abc \Sigma^*$.

The main decision procedure used by Noodler is the *stabilization-based procedure*. Without much detail, the idea behind this approach is to initially convert the regular constraints in the formula into NFAs, and then apply, during the construction, determinizations and minimizations in the automata. For quadratic equations, in particular, a decision procedure called *Nielsen transformation* is used, which constructs a graph from the formula and reason about the satisfia-

bility of the formula over that graph.

During preprocesing, Noodler decreases the number of equations by applying conversion to regular constraints, propagation of epsilons (empty word), under-approximation rules, etc. Another thing Noodler does is to check for trivial unsatisfiable parts of the formula for an eanly termination. Noolder current supports a lot of string functions and predicates such as `replace`, `contains`, `substr`, `at`, `indexOf`, etc. The predicates `replace_all` and `to/from_int` are not supported. In general, the decision procedure implemented in Z3-Noodler can handle the chain-free fragment of string theory, with unbounded disequations and regular constraints, and quadratic equations. For the parts of the theory outside of this fragment, the implementation is sound but not compthe parts of the theory outside of this fragment, the implementation is sound but not complete.

The results presented in the paper show that Z3-Noodler is capable of competing against other string solvers in the state of the art, such as cvc5, Z3, OSTRICH, amog others. In particular, three sets of data were used for benchmarking, denominated `Regex`(regular membership and length constraints), `Equations` (word equations and length constraints) and `Predicates-small`(also includes a few predicates). On the `Regex` group, Z3-Noodler outperforms most of the tools, mainly because of the nature of the usage of NFAs for handling regular constraints. In the `Equations` group, the tool also performed well, being more efficient than most of the other tools. `Predicates-small`, however, is domintaed by cvc5, and Z3-Noodler didn't perform very well, but the results could be improved by applying more axiom saturation for predicates.

In general, we can conclude that Z3-Noodler is a very efficient approach for solving a fragment of string theory formulae, being particularly efficient when the input formula contains regular constraints such as regular membership, which allows for the usage of powerful automata algorithms implemented in the Mata library. However, the tool also contains the weakness of not being able to handle efficiently formulae with a large amount of predicates, limiting it's practical usage.