Z3-Noodler: An Automata-based String Solver

Yu-Fang Chen [1] David Chocholatý [2] **Vojtěch Havlena** [2]
Lukáš Holík [2] Ondřej Lengál [2] Juraj Síč [2]

[1] Academia Sinica, Taiwan

[2] Brno University of Technology, Czech Republic

8 April 2024 (TACAS'24)

Artifacts Available EAPLS V1.1

Artifacts Evaluated EAPLS Reusable V1.1

String Constraint Solving

- strings are everywhere: fundamental datatype in modern PLs
  $\leadsto$ **reasoning about strings is crucial**

String Constraint Solving

- strings are everywhere: fundamental datatype in modern PLs

  ⇝ **reasoning about strings is crucial**

- string contraint solving
  - satisfiability of formulae over string constraints (variables over Σ^*)
  - various types of constraints

$$\underbrace{x = yz \land y \neq u}_{\textit{(dis)equations}} \land \overbrace{x \in (ab)^* a^+ (b|c)}^{\textit{regular constraints}} \land \overbrace{|x| = 2|u| + 1}^{\textit{length constraints}} \land \underbrace{\text{contains}(u, \text{replace}(z, b, c))}_{\textit{more complex operations}}$$

String Constraint Solving

- strings are everywhere: fundamental datatype in modern PLs

  ⇝ **reasoning about strings is crucial**

- string contraint solving
  - satisfiability of formulae over string constraints (variables over Σ^*)
  - various types of constraints

$$\underbrace{x = yz \land y \neq u}_{\textit{(dis)equations}} \land \overbrace{x \in (ab)^* a^+ (b|c)}^{\textit{regular constraints}} \land \overbrace{|x| = 2|u| + 1}^{\textit{length constraints}} \land \underbrace{\text{contains}(u, \text{replace}(z, b, c))}_{\textit{more complex operations}}$$

- wide-ranging applications
  - analysis of string manipulating programs [BlakeDJ'19]
  - vulnerabilities of web applications [ErikssonSDR'23]
  - Amazon cloud access control policies [Rungta'22]

String Constraint Solving

- strings are everywhere: fundamental datatype in modern PLs
  ⇝ **reasoning about strings is crucial**

- string contraint solving
  - satisfiability of formulae over string constraints (variables over Σ^*)
  - various types of constraints

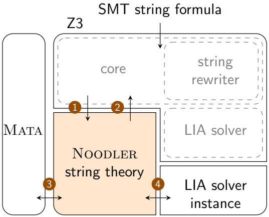
$$\underbrace{x = yz \land y \neq u}_{\textit{(dis)equations}} \land \overbrace{x \in (ab)^* a^+ (b|c)}^{\textit{regular constraints}} \land \overbrace{|x| = 2|u| + 1}^{\textit{length constraints}} \land \underbrace{\text{contains}(u, \text{replace}(z, b, c))}_{\textit{more complex operations}}$$

- wide-ranging applications
  - analysis of string manipulating programs [BlakeDJ'19]
  - vulnerabilities of web applications [ErikssonSDR'23]
  - Amazon cloud access control policies [Rungta'22]

- tool support
  - CVC5, Z3, Z3str4, OSTRICH, **Z3-Noodler**

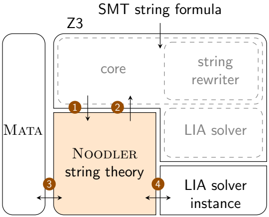
Z3-NOODLER: Highlight

- based on SMT solver Z3
  - replacement of Z3's string theory solver
  - modified string theory rewriter
  - stabilization-based decision procedure



Z3-NOODLER: Highlight

- based on SMT solver Z3
  - replacement of Z3's string theory solver
  - modified string theory rewriter
  - stabilization-based decision procedure

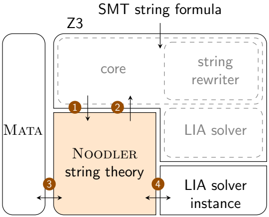


- heavily using nondeterministic finite automata
  - MATA library for efficient operations

[9 April TACAS]

Z3-NOODLER: Highlight

- based on SMT solver Z3
  - replacement of Z3's string theory solver
  - modified string theory rewriter
  - stabilization-based decision procedure

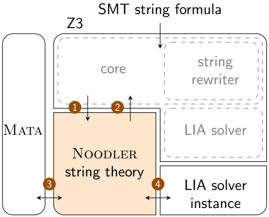


- heavily using nondeterministic finite automata
  - MATA library for efficient operations [9 April TACAS]

- support of various predicate/functions defined by SMT-LIB
  - (dis)equations, length and regular constraints
  - string functions/predicates (`replace`, `indexof`, ...)
  - string conversions (since v1.1) (`from_int`, `to_int`, ...)

Z3-NOODLER: Highlight

- based on SMT solver Z3
  - replacement of Z3's string theory solver
  - modified string theory rewriter
  - stabilization-based decision procedure



- heavily using nondeterministic finite automata
  - MATA library for efficient operations [9 April TACAS]

- support of various predicate/functions defined by SMT-LIB
  - (dis)equations, length and regular constraints
  - string functions/predicates (`replace`, `indexof`, . . .)
  - string conversions (since v1.1) (`from_int`, `to_int`, . . .)

- good for regex-intensive and equation-intensive formulae
  - paradise for the stabilization-based procedure

String Theory Core

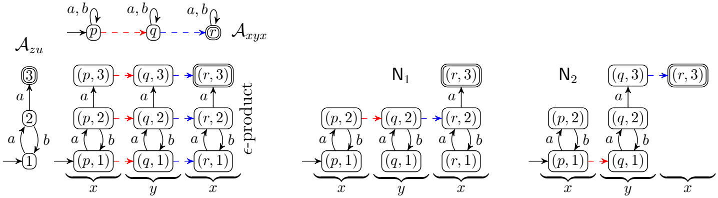
- axiom saturation: axioms for preds/funcs + lengths axioms
  - e.g., $s \notin \Sigma^* abc \Sigma^*$ for $\neg\texttt{contains}(s, \text{"}abc\text{"})$; $|t_1.t_2| = |t_1| + |t_2|$

String Theory Core

- axiom saturation: axioms for preds/funcs + lengths axioms
  - e.g., $s \notin \Sigma^* abc \Sigma^*$ for $\neg\texttt{contains}(s, "abc")$; $|t_1.t_2| = |t_1| + |t_2|$
- preprocessing: transforming the string constraint to a suitable form

String Theory Core

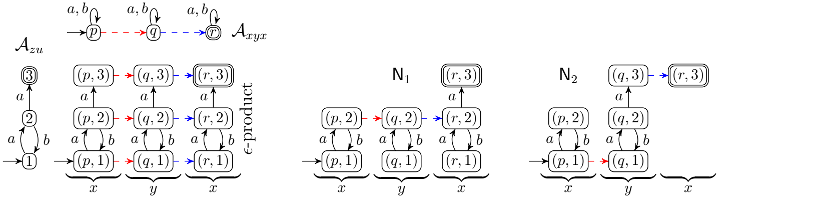
- axiom saturation: axioms for preds/funcs + lengths axioms
  - e.g., $s \notin \Sigma^* abc \Sigma^*$ for $\neg\texttt{contains}(s, "abc")$; $|t_1 . t_2| = |t_1| + |t_2|$
- preprocessing: transforming the string constraint to a suitable form
- stabilization-based procedure [ChenCHHLS'23, BlahHHCCLS'23]
  - iterative refinement of variables' languages
  - based on noodlification of NFAs representing variable languages
  - lazy generation of stable solutions; complete for chain-free fragment



$xyx = zu \land u \in (baba)^* a \land z \in a(ba)^*$

String Theory Core

- axiom saturation: axioms for preds/funcs + lengths axioms
  - e.g., $s \notin \Sigma^* abc \Sigma^*$ for $\neg\texttt{contains}(s, "abc")$; $|t_1.t_2| = |t_1| + |t_2|$
- preprocessing: transforming the string constraint to a suitable form
- stabilization-based procedure [ChenCHHLS'23, BlahHHCCLS'23]
  - iterative refinement of variables' languages
  - based on noodlification of NFAs representing variable languages
  - lazy generation of stable solutions; complete for chain-free fragment



$xyx = zu \land u \in (baba)^* a \land z \in a(ba)^*$

- Nielsen transformation
  - Nielsen graph $\rightsquigarrow$ counter automaton
  - transition saturation + LIA formulae generation

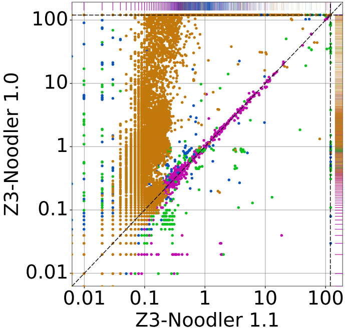
Experiments

- benchmarks from SMT-LIB
  - `QF_S` (18 314)
  - `QF_SLIA` (81 310)

- comparison with SOTA solvers

- comparison with **Z3-NOODLER v1.1**
  - TACAS submission = v1.0
  - various optimizations
  - support of string conversions

- timeout 120 s, memory limit 8 GiB



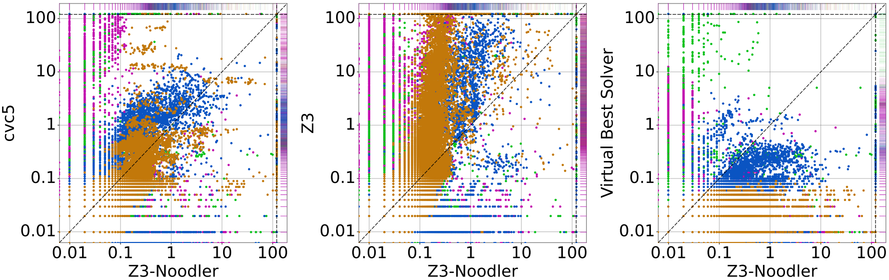
● **Regex** ● **Equations** ● **Predicates-small** ● **PyEx**

Unsolved Instances

	Regex					**Equations**								**Predicates-small**				
	Aut	Den	StrFuzz	Syg	Σ	Kal	Kep	Norn	Slent	Slog	Web	Woo	Σ	StrInt	Leet	StrSm	Σ	PyEx
Included	15995	999	11618	343	28955	19432	587	1027	1128	1976	365	809	25324	16968	2652	1880	21500	23845
Unsupported	0	0	0	0	0	0	0	0	0	0	316	0	316	0	0	0	0	0
Z3-Noodler v1.1	60	**0**	**2**	**0**	**62**	270	**3**	**0**	**1**	**0**	**8**	59	**341**	264	4	137	405	94
cvc5	93	18	703	**0**	814	**1**	240	84	24	**0**	47	54	450	**5**	**0**	**19**	**24**	**19**
Z3	125	116	537	**0**	778	284	309	124	73	31	104	**27**	952	239	**0**	59	298	987
Z3str4	60	4	30	**0**	94	174	254	73	73	16	121	78	789	1102	4	60	1166	570
OSTRICH	**48**	6	218	**0**	272	288	387	**0**	126	6	74	53	934	1059	27	173	1259	12833
Z3str3RE	66	27	185	1	279	144	311	133	87	55	192	118	1040	3231	192	259	3682	17764
Z3-Noodler^{OOPSLA}	86	1	1982	**0**	2069	508	575	**0**	6	**0**	45	256	1390	1627	29	692	2348	13362

- best values in **bold**
- Z3-Noodler outperforms others on **Equations** and **Regex**
- support for `replace_all` is in making

Running Times



- fast on **Equations** and **Regex** (even if compared to VBS)
- often complementary to other solvers
- great in a solver portfolio

● **Regex** ● **Equations** ● **Predicates-small** ● **PyEx**

Conclusion

- string solver Z3-NOODLER based on Z3
- combination of procedures; the stabilization-based procedure
- heavily using nondet. finite automata (MATA) [9 April TACAS]
- fast on equation and regex intensive benchmarks
- Github repo: `https://github.com/VeriFIT/z3-noodler`
- **see the poster**

Conclusion

- string solver Z3-NOODLER based on Z3
- combination of procedures; the stabilization-based procedure
- heavily using nondet. finite automata (MATA) [9 April TACAS]
- fast on equation and regex intensive benchmarks
- Github repo: `https://github.com/VeriFIT/z3-noodler`
- **see the poster**

Future work

- support of `replace_all` WIP
- model generation WIP
- extended support of ¬`contains`

Conclusion

- string solver Z3-NOODLER based on Z3
- combination of procedures; the stabilization-based procedure
- heavily using nondet. finite automata (MATA) [9 April TACAS]
- fast on equation and regex intensive benchmarks
- Github repo: `https://github.com/VeriFIT/z3-noodler`
- **see the poster**

Future work

- support of `replace_all` WIP
- model generation WIP
- extended support of ¬`contains`

Thank You!