Technische Universität München
Fakultät für Informatik
Lehrstuhl für Grundlagen der Software-Zuverlässigkeit und theoretische Informatik
Prof. Dr. Javier Esparza

Thesis (M.Sc.)
A solver for a theory of strings

(in cooperation with mgm technology partners GmbH)

Motivation

An increasing number of areas, including hardware and software quality assurance, require some form of automated reasoning by way of solving constraint-satisfaction problems (CSPs). CSPs in general have a long history in computer science. However, despite the fact that strings occur almost everywhere in computing, CSPs involving strings have only recently attracted attention in the research community. Since the set of allowed values of string-variables (their *domains*) are best characterized via regular expressions, regular languages and finite automata naturally come into play. The process of solving a string-CSP requires operations on such automata.

The goal of this thesis consists in devising and implementing a versatile string-CSP solver. The work will be carried out in close collaboration between TUM and the quality assurance division of mgm technology partners, a software company based in Munich.

mgm Technology partners offers the possibility to work as Werkstudent for the company during the period of the Master's Thesis.

Your Tasks

- familiarize with the existing literature on string solvers,
- define suitable fragments of the theory of strings,
- devise decision procedures for solving the satisfiability (SAT) problem for fragments of the theory of strings,
- implement the decision procedures in Java,
- refine the decision procedures so that they become incremental, and
- integrate the code of the decision procedures into a versatile string-CSP solver.

Requirements

- basic knowledge of *Java*
- some familiarity with *object-oriented* software engineering
- basic knowledge about *propositional logic*
- basic knowledge of *regular expressions* and the theory of *finite automata*
- capability of thinking in *algorithms* and turning them into functional code

Some familiarity with CSPs and graph-algorithms would be an asset.

Keywords

Algorithms – automata theory – constraint satisfaction problems – decision procedures – Java – satisfiability (SAT) – theory of strings

References

Reference Type: Electronic Source
Record Number: 4358
Author: anonymous
Year: 2010
Title: The Regular Expression Generator
Producer: Microsoft Corporation
Keywords: regular expressions
string constraints
string theory
test data generation
Abstract: By using the regular expression generator, you can generate strings that match a defined pattern. You can use the regular expression generator with any data column that has a data type that accepts a string. These data types are char, varchar, varchar(max), text, nchar, nvarchar, nvarchar(max), ntext, and sysname, and user-defined types that are based on these types. You can also use the regular expression generator with common language runtime user-defined types.
URL: http://msdn.microsoft.com/en-us/library/aa833197.aspx

Reference Type: Conference Proceedings
Record Number: 4409
Author: Belhaouari, Hakim; Peschanski, Frédéric
Year of Conference: 2008
Title: A Constraint Logic Programming Approach to Automated Testing
Editor: Banda, M. Garcia de la; Pontelli, E.
Conference Name: ICLP 2008, LNCS 5366
Pages: 754--758
Keywords: constraint satisfaction problems (CSPs)
string constraints
string theory
Abstract: In this paper we present a new constraint solver for the automated generation of test cases from specifications. The specification language is inspired by the contract-oriented programming extended with a finite state machines. Beyond the generation of correct argument values for method calls, we generate full test scenarios thanks to the symbolic animation of the specifications. We propose a flexible CSP architecture that can operate not only on integer or bounded domains but also on arbitrary types. An original notion of type builder is used to establish the link between the type semantics and the CSP framework. We illustrate this with a string builder that can automatically generate string instances depending on combinations of constraints.

URL: http://www.springerlink.com/content/q4h0647146273813/fulltext.pdf

Reference Type: Report
Record Number: 4386
Author: Bjoerner, Nikolaj; Tillmann, Nikolai; Voronkov, Andrei
Year: 2008
Title: Path Feasibility Analysis for String-Manipulating Programs
City: Redmond, WA, USA
Institution: Microsoft Corporation
Keywords: Satisfiability Modulo Theories (SMT)
string constraints
test data generation
string theory
Abstract: We discuss the problem of path feasibility for programs manipulating strings using a collection of standard string library functions. We prove results on the complexity of this problem, including its undecidability in the general case and decidability of some special cases. In the context of test-case generation, we are interested in an efficient finite model finding method for string constraints. To this end we develop a two-tier finite model finding procedure. First, an integer abstraction of string constraints are passed to an SMT (Satisfiability Modulo Theories) solver. The abstraction is either unsatisfiable, or the solver produces a model that fixes lengths of enough strings to reduce the entire problem to be finite domain. The resulting fixed-length string constraints are then solved in a second phase. We implemented the procedure in a symbolic execution framework, report on the encouraging results and discuss directions for improving the method further.
Notes: "our objective is really to find small strings that can be supplied as unit tests"
URL: http://research.microsoft.com/apps/pubs/default.aspx?id=70656
https://mailserver.di.unipi.it/ricerca/proceedings/ETAPS09/papers/5505/55050307.pdf

Reference Type: Conference Proceedings
Record Number: 4450
Author: Emmi, Michael; Majumdar, Rupak; Sen, Koushik
Year of Conference: 2007
Title: Dynamic Test Input Generation for Database Applications
Conference Name: Proc. 2007 international symposium on software testing and analysis (ISSTA '07)
Keywords: automatic test generation
constraint solver for string theory: algorithm
string constraints: equality, disequality, regular expressions
union-find algorithms

Abstract: We describe an algorithm for automatic test input generation for database applications. Given a program in an imperative language that interacts with a database through API calls, our algorithm generates both input data for the program as well as suitable database records to systematically explore all paths of the program, including those paths whose execution depend on data returned by database queries. Our algorithm is based on concolic execution, where the program is run with concrete inputs and simultaneously also with symbolic inputs for both program variables as well as the database state. The symbolic constraints generated along a path enable us to derive new input values and new database records that can cause execution to hit uncovered paths. Simultaneously, the concrete execution helps to retain precision in the symbolic computations by allowing dynamic values to be used in the symbolic executor. This allows our algorithm, for example, to identify concrete SQL queries made by the program, even if these queries are built dynamically.

The contributions of this paper are the following. We develop an algorithm that can track symbolic constraints across language boundaries and use those constraints in conjunction with a novel constraint solver to generate both program inputs and database state. We propose a constraint solver that can solve symbolic constraints consisting of both linear arithmetic constraints over variables as well as string constraints (string equality, disequality, as well as membership in regular languages). Finally, we provide an evaluation of the algorithm on a Java implementation of MediaWiki, a popular wiki package that interacts with a database backend.
URL: http://cs.ucla.edu/~mje/docs/issta2007-paper.pdf
Author Address: Michael Emmi
UC Los Angeles
mje@cs.ucla.edu
Rupak Majumdar
UC Los Angeles
rupak@cs.ucla.edu
Koushik Sen
UC Berkeley
ksen@cs.berkeley.edu

Reference Type: Conference Proceedings
Record Number: 4410
Author: Golden, Keith; Pang, Wanlin
Year of Conference: 2003
Title: Constraint reasoning over strings
Editor: Rossi, Francesca
Conference Name: Principles and practice of constraint programming-CP 2003: 9th international Conference
Conference Location: Kinsale, Ireland
Date: Sept./Oct. 2003
Keywords: string constraints

string theory
Abstract: This paper discusses an approach to representing and reasoning about constraints over strings. We discuss how many string domains can often be concisely represented using regular languages, and how constraints over strings, and domain operations on sets of strings, can be carried out using this representation.
URL: http://ti.arc.nasa.gov/static/asanicms/pub-archive/archive/0529.pdf
Author Address: 1 Computational Science Division, NASA Ames Research Center, Moffett Field, CA 94035
2 QSS Group Inc., NASA Ames Research Center, Moffett Field, CA 94035

Reference Type: Thesis
Record Number: 4412
Author: Hooimeijer, Pieter
Year: 2010
Title: Decision Procedures for String Constraints
Academic Department: Computer Science Dept.
City: Virginia
University: University of Virginia
Thesis Type: PhD
Keywords: string constraints
string theory
URL: http://www.cs.virginia.edu/~weimer/students/pieter-phd-proposal.pdf
Author Address: Pieter Hooimeijer
Department of Computer Science
151 Engineer's Way
P.O. Box 400740
Charlottesville, VA 22904-4740

Reference Type: Personal Communication
Record Number: 4414
Author: Hooimeijer, Pieter; Veanes, Margus
Year: 2010
Title: An Evaluation of Automata Algorithms for String Analysis
City: Redmond City
Publisher: Microsoft Research
Date: August 2010
Keywords: string constraints
string theory
Abstract: There has been significant recent interest in automated reasoning techniques, in particular constraint solvers, for string variables. These techniques support a wide range of clients, ranging from static analysis to automated testing. The majority of string constraint solvers rely on finite automata to support regular expression constraints. For these approaches, performance depends critically on fast automata operations such as intersection, complementation, and

determinization. Existing work in this area has not yet provided conclusive results as to which core algorithms and data structures work best in practice.

In this paper, we study a comprehensive set of algorithms and data structures for performing fast automata operations. Our goal is to provide an apples-to-apples comparison between techniques that are used in current tools. To achieve this, we re-implemented a number of existing techniques. We use an established set of regular expressions benchmarks as an indicative workload. We also include several techniques that, to the best of our knowledge, have not yet been used for string constraint solving. Our results show that there is a substantial performance difference across techniques, which has implications for future tool design.
URL: http://research.microsoft.com/pubs/133121/MSR-TR-2010-90.pdf
Author Address: Pieter Hooimeijer1 and Margus Veanes2
1 University of Virginia
pieter@cs.virginia.edu
2 Microsoft Research
margus@microsoft.com

Reference Type: Conference Proceedings
Record Number: 4411
Author: Hooimeijer, Pieter; Weimer, Westley
Year of Conference: 2010
Title: Solving string constraints lazily
Conference Name: ASE '10 Proceedings of the IEEE/ACM international conference on Automated software engineering
Keywords: constraint satisfaction problems (CSPs)
string constraints
string theory
Abstract: Decision procedures have long been a fixture in program analysis, and reasoning about string constraints is a key element in many program analyses and testing frameworks. Recent work on string analysis has focused on providing decision procedures that model string operations. Separating string analysis from its client applications has important and familiar benefits: it enables the independent improvement of string analysis tools and it saves client effort.

We present a constraint solving algorithm for equations over string variables. We focus on scalability with regard to the size of the input constraints. Our algorithm performs an explicit search for a satisfying assignment; the search space is constructed lazily based on an automata representation of the constraints. We evaluate our approach by comparing its performance with that of existing string decision procedures. Our prototype is, on average, several orders of magnitude faster than the fastest existing implementation
URL: https://qosbox.cs.virginia.edu/~weimer/p/weimer-ase2010-lazy-preprint.pdf
Author Address: Pieter Hooimeijer
Department of Computer Science
151 Engineer's Way

P.O. Box 400740
Charlottesville, VA 22904-4740

Reference Type: Electronic Source
Record Number: 4385
Author: Jha, Susmit; Seshia, Sanjit A.; Limaye, Rhishikesh
Year: 2009
Title: On the Computational Complexity of Satisfiability Solving for String Theories
Producer: UC Berkeley
Access Year: 2010
Last Update Date: March 16, 2009
Keywords: bit-vector SMT solver
byte-blast approach
Satisfiability Modulo Theories (SMT)
string constraints
string theory
Abstract: Satisfiability solvers are increasingly playing a key role in software verification, with particularly effective use in the analysis of security vulnerabilities. String processing is a key part of many software applications, such as browsers and web servers. These applications are susceptible to attacks through malicious data received over network. Automated tools for analyzing the security of such applications, thus need to reason about strings. For efficiency reasons, it is desirable to have a solver that treats strings as first-class types. In this paper, we present some theories of strings that are useful in a software security context and analyze the computational complexity of the presented theories. We use this complexity analysis to motivate a byte-blast approach which employs a Boolean encoding of the string constraints to a corresponding Boolean satisfiability problem.
Notes: "The frequent use of string operations in these applications has motivated several groups to explore the possibility of designing a constraint solver which treats strings as first-class types."

"We identify a set of core predicates and functions. Many other more complicated string-manipulating functions can be expressed as some simple composition of these functions. We use these predicates and functions to define a theory of strings."

"Constraint solvers are widely used in verification and validation of software and hardware systems."

"Analysis of string processing software is an important problem [...]. This makes it essential to develop verification techniques that can efficiently handle constraints over strings."
URL: http://arxiv.org/PS_cache/arxiv/pdf/0903/0903.2825v1.pdf
Author Address: EECS Department, UC Berkeley

{jha,sseshia,rhishi}@eecs.berkeley.edu

Reference Type: Conference Proceedings
Record Number: 4381
Author: Kiezun, Adam; Ganesh, Vijay; Guo, Philip J.; Hooimeijer, Pieter; Ernst, Michael D.
Year of Conference: 2009
Title: HAMPI: A Solver For String Constraints
Conference Name: ISSTA 2009: ACM International Symposium on Testing and Analysis
Conference Location: Chicago, Illinois, USA
Date: 19-23 July 2009
Keywords: context-free languages
regular languages
satisfiability (SAT)
string constraints
string theory
Abstract: Many automatic testing, analysis, and verification techniques for programs can be effectively reduced to a constraint-generation phase followed by a constraint-solving phase. This separation of concerns often leads to more effective and maintainable tools. The increasing efficiency of off-the-shelf constraint solvers makes this approach even more compelling. However, there are few effective and sufficiently expressive off-the-shelf solvers for string constraints generated by analysis techniques for string-manipulating programs.

We designed and implemented Hampi, a solver for string constraints over fixed-size string variables. Hampi constraints express membership in regular languages and fixed-size context-free languages. Hampi constraints may contain context-free-language definitions, regular-language definitions and operations, and the membership predicate. Given a set of constraints, Hampi outputs a string that satisfies all the constraints, or reports that the constraints are unsatisfiable.

Hampi is expressive and efficient, and can be successfully applied to testing and analysis of real programs. Our experiments use Hampi in: static and dynamic analyses for finding SQL injection vulnerabilities in Web applications; automated bug finding in C programs using systematic testing; and compare Hampi with another string solver. Hampi's source code, documentation, and the experimental data are available at http://people.csail.mit.edu/akiezun/hampi.
URL: http://people.csail.mit.edu/akiezun/issta54-kiezun.pdf

Reference Type: Report
Record Number: 4379
Author: Veanes, Margus; de Halleux, Peli; Tillmann, Nikolai
Year: 2009

Title: Rex: Symbolic Regular Expression Explorer
City: Redmond, WA, USA
Institution: Microsoft Research
Type: technical report
Report Number: Microsoft Research Technical Report MSR-TR-2009-137
Keywords: regular expressions
finite automata
satisfiability modulo theories
string constraints
string theory
Abstract: Constraints in form [of] regular expressions over strings are ubiquitous. They occur often in programming languages like Perl and C#, in SQL in form of LIKE expressions, and in web applications. Providing support for regular expression constraints in program analysis and testing has several useful applications. We introduce a method and a tool called Rex, for symbolically expressing and analyzing regular expression constraints. Rex is implemented using the SMT solver Z3, and we provide experimental evaluation of Rex.
URL: http://research.microsoft.com/pubs/102927/rex-TR.pdf
Author Address: {margus,jhalleux,nikolait}@microsoft.com

Reference Type: Conference Proceedings
Record Number: 4423
Author: Veanes, M.; de Halleux, P.; Tillmann, N.
Year of Conference: 2010
Title: Rex: Symbolic Regular Expression Explorer
Conference Name: Software Testing, Verification and Validation (ICST), 2010 Third International Conference on ...
Pages: 498--507
Date: 6-10 April 2010
Keywords: constraint satisfaction problems (CSPs)
satisfiability modulo theories (SMT): Z3
string constraints
string theory
Abstract: Constraints in form [of] regular expressions over strings are ubiquitous. They occur often in programming languages like Perl and C#, in SQL in form of LIKE expressions, and in web applications. Providing support for regular expression constraints in program analysis and testing has several useful applications. We introduce a method and a tool called Rex, for symbolically expressing and analyzing regular expression constraints. Rex is implemented using the SMT solver Z3, and we provide experimental evaluation of Rex.