UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
INSTITUTO DE MATEMÁTICA

VÍTOR LUÍS DOS SANTOS TRINDADE

Comparação de resolvedores SMT como geradores de exemplos para modelagens B

Prof. Anamaria Martins Moreira, Ph.D.
Orientador

Rio de Janeiro, Novembro de 2017

Comparação de resolvedores SMT como geradores de exemplos para modelagens B

Vítor Luís dos Santos Trindade

Projeto Final de Curso submetido ao Departamento de Ciência da Computação do Instituto de Matemática da Universidade Federal do Rio de Janeiro como parte dos requisitos necessários para obtenção do grau de Bacharel em Informática.

Apresentado por:

Vítor Luís dos Santos Trindade

Aprovado por:

Prof. Anamaria Martins Moreira, Ph.D.

Prof. Nome do participante banca 1, D.Sc.

Prof. Nome do participante banca 2, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

Novembro de 2017

RESUMO

Comparação de resolvedores SMT como geradores de exemplos para modelagens B

Vítor Luís dos Santos Trindade

Novembro/2017

Orientador: Anamaria Martins Moreira, Ph.D.

B é um método formal usado para a modelagem de especificações e desenvolvimento rigoroso de software. BETA (Bbased testing approach) é uma ferramenta de geração de testes a partir de uma especificação em B. Uma parte do processo de criação de testes em BETA é, dado um teste, obter um conjunto de valores válidos para as entradas do teste, ou a informação que o teste não possui valores válidos. A ferramenta utilizada atualmente para encontrar valores válidos apresenta um problema de explosão de estados na geração de modelos. Como alternativa, buscamos o auxílio de resolvedores SMT (Satisfiability modulo theories), ferramentas capazes de resolver problemas de satisfabilidade em diversas combinações de teorias para obter esse conjunto de valores válidos.

ABSTRACT

Comparação de resolvedores SMT como geradores de exemplos para modelagens B

Vítor Luís dos Santos Trindade

Novembro/2017

Advisor: Anamaria Martins Moreira, Ph.D.

The state's duty is not limited to a mere judicial response, but requires the provision of effective protection that meets the constitutional principle of reasonable duration of the procedure laid down in the Constitution. However, the delay in the delivery of legal protection remains one of contemporary evils of civil procedure.

Lista de Figuras

Lista de Tabelas

Tabela 4.1: Dados preeliminares de {log} 8

Tabela 4.2: Dados preeliminares de Alt-Ergo 9

Tabela 4.3: Dados preeliminares de CVC4 10

Tabela 4.4: Dados preeliminares de Z3 11

Tabela 4.5: Dados preeliminares de ProB 12

Lista de Abreviaturas e Siglas

SAT	Problema da satisfatibilidade booleana ou boolean satisfiability problem
SMT	Satisfiability Modulo Theories
SMT-LIB	Satisfiability Modulo Theories Library

Lista de Símbolos

SAT	Problema da satisfatibilidade booleana ou boolean satisfiability problem
SMT	Satisfiability Modulo Theories
SMT-LIB	Satisfiability Modulo Theories Library

Sumário

Resumo i

Abstract ii

Lista de Figuras iii

Lista de Tabelas iv

Lista de Abreviaturas e Siglas v

Lista de Símbolos vi

1 Introdução 1

1.1 Objetivos 1

1.1.1 Objetivo Geral 1

1.1.2 Objetivos Específicos 1

1.1.3 Estrutura do Artigo 1

2 Fundamentação Teórica 2

2.1 Método B e Beta 2

2.2 SMT Solvers e SMT-LIB 4

3 Metodologia **6**

4 Resultados dos Dados Preeliminares **8**

Capítulo 1

Introdução

1.1 Objetivos

1.1.1 Objetivo Geral

Temos como objetivo geral a avaliação de diferentes ferramentas em respeito a sua capacidade de prover conjuntos de valores válidos (Expandir)

1.1.2 Objetivos Específicos

Para poder utilizar as ferramentas primeiramente é preciso determinar como transformar uma máquina B em algo passível de ser entendido pela ferramenta, se possível de forma automática, para que BETA possa se comunicar com a ferramenta diretamente.
Determinar quais são as ferramentas disponíveis para os nossos objetivos.
Definir que tipo de ferramenta é a mais vantajosa para o problema e, se possível, determinar a melhor ferramenta.

1.1.3 Estrutura do Artigo

Última coisa a ser feita

Capítulo 2

Fundamentação Teórica

2.1 Método B e Beta

O Método B é uma metodologia formal para especificação, projeto e codificação de software [**?**]. O primeiro passo do método B se dá através da criação de máquinas abstratas, que especificam o comportamento do sistema. Essa máquina passa por vários refinamentos até que seja compreensível para um computador, ao nível de o último passo do refinamento é a implementação.

Código 2.1: Uma possível máquina B para uma ATM

```
/* ATM
 * Author:
 * Creation date: 08/10/14
 */
MACHINE
    ATM
VARIABLES
    account_id, account_balance
INVARIANT
    account_id : INT &
    account_id > 0 &
    account_balance : INT &
    account_balance >= 0
INITIALISATION
    account_id :: INT || account_balance := 0
OPERATIONS
    deposit(mm) =
    PRE
        mm : INT & mm > 0
    THEN
        account_balance := account_balance + mm
    END;

    withdraw(mm) =
    PRE
        mm : INT & mm > 0 & account_balance >= mm
    THEN
        account_balance := account_balance - mm
    END;

    bb <-- balance = bb := account_balance

END
```

(Botar Referência para de onde saiu esse código)Uma máquina B pode possuir as seguintes cláusulas

- MACHINE: Nome da máquina
- SETS: Conjuntos utilizados, tanto abstratos como enumerados
- CONSTANTS: Constantes a serem usadas
- PROPERTIES: Descreve as constantes e conjuntos
- VARIABLES: Variáveis locais
- INVARIANTS: Propriedades da variáveis locais que são sempre verdadeiras. Uma variável nunca deve ferir um invariante.
- INITIALISATION: Estado inicial da máquina. Todas as variáveis devem possuir um valor inicial
- OPERATIONS: Operações da máquina. É dividido em precondição e corpo da operação. Uma operação só pode ser executada se as precondições forem cumpridas.

BETA é uma abordagem para geração de casos de teste de unidade a partir de especificações formais escritas na notação do Método B[**?**]. A partir de do invariante do estado, das precondições e de outros predicados de uma operação, BETA gera testes para cada operação de uma máquina B usando como critério de cobertura o particionamento do espaço de entrada.A ferramenta utilizada atualmente por BETA para gerar os valores de entrada dos teste para cobrir o espaço de entrada, ProB, lida com faixas de valores de modo ineficiente, e só é utilizável em faixas muito reduzidas (a faixa de inteiros default de ProB é de -1 a 3). Qualquer tentativa de ampliar a faixa pode resultar em explosão combinatória.

2.2 SMT Solvers e SMT-LIB

Existe um problema chamado problema da satisfatibilidade booleana (conhecido também como boolean satisfiability problem ou SAT), que consiste em determinar se existe pelo menos uma valoração que satisfaz uma fórmula booleana. Para

resolver esse problema existem ferramentas, entre elas os resolvedores SAT (SAT solvers), que se utilizam de algoritmos de backtracking. SMT (Satisfiability Modulo Theories) é uma extensão de SAT que permite a adição de mais teorias além da booleana, como arrays e funções não interpretadas, e o problema continua sendo se existe uma valoração que satisfaz a fórmula, embora as variáveis não sejam mais necessariamente booleanas. Para facilitar a comparação, integração e comunicação entre solvers SMT foi criado o Satisfiability Modulo Theories Library (SMT-LIB), que provê diversos recursos, entre eles a linguagem SMT-LIB, criada com objetivo de ser a linguagem padrão para solvers SMT. Essa linguagem é utilizada em SMT-COMP, uma competição anual de benchmarks para solvers SMT.

Código 2.2: Uma exemplo de uma sessão interativa com um solver usando SMT-LIB omitindo as respostas de sucesso[?]

```
>(set-option :produce-modelstrue)
>(set-logic QF_LIA)
>(declare-fun x () Int)
>(declare-fun y () Int)
>(assert (= (+ x y) 9))
>(assert (= (+ (* 2 x) (* 3 y)) 22))
>(check-sat)
sat
>(get-value (x))
((x 5))
>(get-value ((- x y) (+ 2 y)))
(((- x y) 1)((+ 2 y) 8)
```

Capítulo 3

Metodologia

Foram escolhidas ferramenta com um grande número de usuários e ferramentas usadas em casos parecidos em outras linguagens. Essas ferramentas são log, Alt-Ergo, CVC4 e Z3. Foram levantados então dados preeliminares dessas ferramentas e também de ProB, a ferramenta usada atualmente por BETA, baseados apenas na documentação de cada um deles, nas categorias de:

- Teorias suportadas: Se a ferramenta é capaz de lidar com todas as teorias envolvidas no teste;
- Documentação: Em termos de relevância para um usuário, amplitude e disponibilidade e, pois ela precisa ser informativa para quem for usar a ferramenta e não só para desenvolvedores, precisa descrever todo o uso, da instalação a execução, e precisa ser de livre e fácil acesso.Quanto melhor a documentação, mais fácil a interação com a ferramenta. Uma ferramenta com documentação fraca ou incompleta obriga a tentativa e erro.
- Em desenvolvimento: Aplicações em desenvolvimento são capazes de melhorar e possuem pouca probabilidade de serem abandonados. É possível para sugerir novas melhorias para uma ferramenta em desenvolvimento para melhor atender nossas necessidades.
- Retorna casos que satisfazem: Se é capaz de, dado as restrições do teste,

retornar qual o conjunto de valores que satisfazem essas restrições e permitem a avaliação do teste. Esse é um ponto principal para o nosso uso. Caso não haja o retorno dos casos que satisfazem, a ferramenta não possui utilidade.

- Usado por ferramentas de B: Indica que a compatibilidade com B já foi alcançada.
- Tamanho da base de usuários: Ferramentas com base de usuários maiores possuem menos chances de serem abandonados e tendem a se desenvolver mais rápido.
- Se possui API e sua linguagem: Caso possua uma API, é possível que Beta possa se comunicar com ele de modo transparente para o usuário.
- Linguagem de entrada: Modo de entrada padrão.

Capítulo 4

Resultados dos Dados Preeliminares

Todos os dados são referentes ao dia 28/08/2017

Nome	{log}
Descrição	Uma língua de programação com restrições lógicas baseada em Prolog voltada para manipulação de conjuntos.
Website	`http://people.dmi.unipr.it/gianfranco.rossi/setlog.Home.html`
Teorias Suportadas	Sets, Integers.
Documentação	Possui exemplos, manual em `http://people.dmi.unipr.it/gianfranco.rossi/SETLOG/manual_4_9_1.pdf`
Em desemvolvimento	Último update 01/17 no beta, último estável 04/2016.
Retorna casos que satisfazem	Sem informação.
Usado por ferramentas de B	Nenhuma conhecida.
Tamanho da base de usuários	Nenhuma conhecida.
Possui API e sua linguagem	Nenhuma conhecida.
Linguagem de entrada	Baseada em Prolog.

Tabela 4.1: Dados preeliminares de {log}

Comparando as ferramentas é perceptível que a maioria das ferramentas aceita

Nome	Alt-Ergo
Site	`http://alt-ergo.lri.fr/`
Teorias Suportadas	Free theory of quality with uninterpreted symbols, linear arithmetic over integers and rationals, non-linear arithmetic, polymorphic functional arrays, enumerated datatypes, record datatypes, associative and commutative (AC) symbols, fixed-size bit-vectors.
Licença	CeCILL-C.
Documentação	`https://github.com/OcamlPro/alt-ergo/blob/master/INSTALL.md`, muito voltada para instalação e funcionamento interno.
Em desenvolvimento	Última release estável 21/11/2016, sem commits desde então.
Retorna casos que satisfazem	Sem informação.
Usado por ferramentas de B	Atelier-B, Rodin.
Tamanho da base de usuários	Mais de 20 estrelas no github, usado na plataforma Why3.
Possui API e sua linguagem	Sem informação.
Linguagem de entrada	Linguagem Própria, aceita SMT-LIB.

Tabela 4.2: Dados preeliminares de Alt-Ergo

a linguagem padrão SMT-LIB, e que se BETA for capaz de descrever os testes em SMT-LIB ele poderá usar a maioria das ferramentas.

Nome	CVC4
Site	`http://cvc4.cs.stanford.edu/`
Teorias Suportadas	Equality over free (aka uninterpreted) function and predicate symbols, real and integer linear arithmetic, bit-vectors, arrays, tuples, records, user-defined inductive datatypes, strings, finite sets, separation logic.
Licença	CVC4 é All rights reserved, o código fonte é BSD.
Documentação	Boa para API, necessário o uso da documentação do CVC3 na linguagem própria, terceriza parte da documentação SMT-LIB
Em desenvolvimento	Sim, múltiplos commits diários, releases estáveis com uma relativa regularidade.
Retorna casos que satisfazem	Sim.
Usado por ferramentas de B	Rodin.
Tamanho da base de usuários	30 perguntas com a tag CVC4 no stackoverflow e 168 estrelas no repositório do github.
Possui API e sua linguagem	Possui uma API em C++.
Linguagem de entrada	Possui uma linguagem própria e aceita SMT-LIB

Tabela 4.3: Dados preeliminares de CVC4

Nome	Z3
Site	`https://github.com/Z3Prover/z3`
Teorias Suportadas	uninterpreted funcitions, arrays, bitvectors, bitvectors arrays, reals, ints, difference logic over integers, nonlinear, quantifiers. Não suporta sets.
Licença	MIT
Documentação	Documentação simples da API, tutorial introdutório de como usar em SMT-LIB bem detalhado, teoria por teoria https://github.com/Z3Prover/z3/wiki/Documentation
Em desenvolvimento	Múltiplos commits diários, última versão estável lançada em 7/11/2016.
Retorna casos que satisfazem	Sim.
Usado por ferramentas de B	Rodin.
Tamanho da base de usuários	Mais de 1600 perguntas com tag z3 no stackoverflow, mais de 2,200 estrelas no github.
Possui API e sua linguagem	C, C#, Java, Python.
Linguagem de entrada	SMT-LIB.

Tabela 4.4: Dados preeliminares de Z3

Nome	ProB
Site	`https://www3.hhu.de/stups/prob/index.php/Main_Page`
Teorias Suportadas	Todas de B.
Licença	EPL v1.0
Documentação	Bem extenso e completo, https://www3.hhu.de/stups/prob/index.php/Tutorial
Em desenvolvimento	Sim
Retorna casos que satisfazem	Sim.
Usado por ferramentas de B	Rodin.
Tamanho da base de usuários	Siemens, Alstom, AtelierB, Rodin.
Possui API e sua linguagem	Java.
Linguagem de entrada	B.

Tabela 4.5: Dados preeliminares de ProB