The Great Book of Zebra

The Zebra Project

September 9, 2015

Preface

This book is a collaborative work from the https://github.com/fsvieira/zebrajs project community and everyone is invited to participate.

The list of contributors is at the contributors section 1.4 and your name can be there too :D.

This is a work in progress.

Chapter 1

Introduction

1.1 The Zebra-machine (ZM)

This is the official book of Zebra-machine (ZM). Here you will find anything you need to understand in deep the ZM, the book covers both theoretical and practical definitions.

Zebra-machine (ZM)is a logical symbolic computation query system, given a set of computational definitions it will answer questions about them, therefor ZMis better suited for software validation and constrain satisfaction problems.

1.2 The Zebra-machine (ZM)

As mentioned before ZM is a logical symbolic computation query system, and it consists of two parts the definitions and the query, both parts share the same language of ZM terms, which is defined by a certain formal syntax, and a set of transformation rules.

The ZM language of terms are defined as:

1. c: c is a terminal symbol called constants. Constants are ZM terms.
2. $'p$: p is a variable, variables are ZM terms.
3. $(p_0 \dots p_n)$: its a n-tuple of ZM terms called z-tuples. Z-tuples are ZM terms.
4. Nothing else is a ZM term.

A ZM computation is expressed as 4-tuple $(\sigma, \delta, q, \alpha)$ where:

1. σ is a set of terminal symbols (constants),

2. δ is a set of z-tuples (definitions),
3. q is a z-tuple (query),
4. α is the set of possible computational answers to query q based on *delta* definitions.

1.2.1 ZM Operations

Unification ($\circ$, binary operation) defined as $q \circ p$ where q and p are ZM terms. Unification is defined by the rules:

1. $p \circ p \implies p$

   p unifies with itself, resulting on itself.

2. $'p \circ q \iff 'p = q$

   $'p$ is a variable and q is a ZM term, they unify iff $'p = q$.

3. $q \circ' p \iff 'p = q$

   $'p$ is a variable and q is a ZM term, they unify iff $'p = q$.

4. $(p_0 \ldots p_n) \circ (q_0 \ldots q_n) \iff (p_0 \circ q_0 \ldots p_n \circ q_n)$

   $(p_0 \ldots p_n)$ z-tuple only unifies with other z-tuple if they have same size and all sub ZM terms unify.

5. Anything else fails to unify.

Not-unify ($\setminus$, binary operation) The not-unify operation describes a "set" excluding a ZM term. Let q and p be ZM terms therefor

$$q \setminus p \iff q \neq p$$

TODO: make operations like unify.

Examples:

- $yellow \setminus blue$,

  $yellow$ and $blue$ are constants and $yellow \neq blue$.

- $(blue\ yellow) \setminus (yellow\ blue)$,

  $(blue\ yellow) \neq (yellow\ blue)$

- $(blue\ 'p) \setminus (yellow\ blue)$,

  $'p$ is a variable and since $(blue'p) \neq (yellow\ blue)$ then $'p \neq blue$

- $'p \setminus' q$,

  $'p$ and $'q$ are variables, $'p \neq' q$.

1.3 Computing Examples

Unification

1. $yellow \circ yellow \implies yellow$, succed.
2. $blue \circ yellow$, fail: can't unify constants with diferent value.
3. $yellow \circ (yellow)$, fail: can't unify constant and tuple.
4. $(blue\ yellow) \circ (blue\ yellow) \implies (blue \circ blue\ yellow \circ yellow) \implies (blue\ yellow)$, succed.

1.4 Contributors

- Filipe Vieira, https://github.com/fsvieira