UNIVERSITY OF ILLINOIS AT CHICAGO

MASTER'S PROJECT

Generating Useful Sequences for Integration Testing

Author:
Kush SHAH

Supervisor:
Dr. Mark GRECHANIK

A report submitted in fulfilment of the requirements for the degree of Master of Science

in the

Department of Computer Science

April 2014

Declaration of Authorship

I, Kush Shah, declare that this report titled, 'Generating Useful Sequences for Integration Testing' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a master's degree at this University.
- Where any part of this report has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this report is entirely my own work.
- I have acknowledged all main sources of help.
- Where the report is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

UNIVERSITY OF ILLINOIS AT CHICAGO

Abstract

Mark Grechanik
Department of Computer Science

Master of Science

Generating Useful Sequences for Integration Testing

by Kush SHAH

The main aim of the project is to generate meaningful and optimized sequences from the huge number of sequence pools available from the execution traces and state carving applied on them.

This activity can be divided into various sub tasks:

→ Parsing the json out put of ASSIST and generating an input file for the Frequent Closed Sequence Miner(BIDE)[3].

→ Applying BIDE using different thresholds on the same data and receiving different outputs from BIDE.

→ Finding out the frequent sequences in all order and quantity present in the original data using BIDE output.

→ Optimize the sequences using some heuristics(shortest sewuence, ranked shortest sequence,etc.) so as to get the most information out of it.

→ Revert the output back to ASSIST in JSON fromat.

Acknowledgements

The acknowledgements and the people to thank go here, don't forget to include your project advisor…

Contents

List of Figures

List of Tables

Abbreviations

LAH	**L**ist **A**bbreviations **H**ere

Physical Constants

Speed of Light	c	$=$	$2.997\ 924\ 58 \times 10^8$ ms$^{-\mathrm{S}}$ (exact)

Symbols

a	distance	m
P	power	W (Js^{-1})
ω	angular frequency	rads^{-1}

For/Dedicated to/To my…

Chapter 1

Introduction

1.1 Contemporary Softwares

Comapnies today are embracing *agile software development* to a greater extent for their softwares.

The two main facets of agile technologies are *incremental* and *iterative*. They incorporate the ever changing requirements and aim to incorporate all. The very nature of the development pratice being incremental requires the employees to have sprints and/or scrums for the projects. It is basically a short period in time after which the developers and managers meet and incrementally build upon the requirements or even change them if required.

Due to this incremntal and iterative approach, testing plays a very important role in the process. More often the releases are between very small amount of time periods like two weeks or may be less.

This creates a lot of versions with varied requirements and as the software is build incrementally and has smaller and broken up modules, *Integration Testing* is very important and supposed to be be sound and intuitive.

1.2 Integration Testing

Integration testing involves testing components and/or modules in there entirety to check the correctness of the software.

As discussed earlier, integration testing becomes important for agile development methods but it is also important in any software development methods. Integration Testing is a kind of black box testing used to test, as the name suggests the integrity of the software. Shared data and processes are simulated so that components being tested are tested to behave and interact correctly among each other.

At one end of the spectrum of integration testing we can view acceptance testing, which integrates all components of a system and are evaluated according to the requirements. Whereas on the other end of the spectrum lies unit testing where a units(functions) belonging logically or structurally together(eg. class) are tested individually and then combined with similar other units to confirm functional correctness using test harnesses.

Having said this, one needs a sound balance between both the approaches as a perfect integration testing would involve testing methodologies of both the extremes. As the system grows larger, it becomes exponentially difficult to test all the combinations possible. Thus it becomes more important for a better integration testing. Using the unit testing paradigm makes the testing very sluggish as it grows and randomly checking components to work together may be insensible because they may or may not interact in the system. Thus, generating good integration tests involves not only a significant effort to compose differents simulation of states of various components but also a good knowledge of the components and their internals and an acute intuitive niche of combining different components.

1.3 Problem

Thus from the above information we can conclude that setting up an Integration Testing mechanism is an overhead and is a burdern financially on the system at the initial phases but several case studies have proved that they increase the quality of a software in the long run.

It has become more important to have integration test suites up and running from a ver early stage looking at the Test Driven Development approach in the agile methodologies.

One looses a count and scale of test suites once the project starts to grow at a faster pace as the suites grow exponentially wit the increase in number of components becasue of the many combinations possible. There is also a factor of time as in agile development releases are too nearly placed with respect to time. Thus there is an urgent need to make this process as automated as possible.

This project contributes in a very small way to the larger project and research carried out by Dr. Mark GRECHANIK in the from of Automatic SynthesiS of Integration Software Tests(ASSIST)(explained in the next Chapter).

Chapter 2

Automatic SynthesiS of Integration Software Tests(ASSIST)

2.1 Overview

Welcome to this LaTeX Thesis Template, a beautiful and easy to use template for writing a thesis using the LaTeX typesetting system.

If you are writing a thesis (or will be in the future) and its subject is technical or mathematical (though it doesn't have to be), then creating it in LaTeX is highly recommended as a way to make sure you can just get down to the essential writing without having to worry over formatting or wasting time arguing with your word processor.

LaTeX is easily able to professionally typeset documents that run to hundreds or thousands of pages long. With simple mark-up commands, it automatically sets out the table of contents, margins, page headers and footers and keeps the formatting consistent and beautiful. One of its main strengths is the way it can easily typeset mathematics, even *heavy* mathematics. Even if those equations are the most horribly twisted and most difficult mathematical problems that can only be solved on a super-computer, you can at least count on LaTeX to make them look stunning.

2.2 Stages

LaTeX is not a WYSIWYG (What You See is What You Get) program, unlike word processors such as Microsoft Word or Apple's Pages. Instead, a document written for LaTeX is actually a simple, plain text file that contains *no formatting*. You tell LaTeX how you want the formatting in the finished document by writing in simple commands amongst the text, for example, if I want to use *italic text for emphasis*, I write the '`\textit{}`' command and put the text I want in italics in between the curly braces. This means that LaTeX is a "mark-up" language, very much like HTML.

2.2.1 A (not so short) Introduction to LaTeX

If you are new to LaTeX, there is a very good eBook – freely available online as a PDF file – called, "The Not So Short Introduction to LaTeX". The book's title is typically shortened to just "lshort". You can download the latest version (as it is occasionally updated) from here:
`http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf`

It is also available in several other languages. Find yours from the list on this page:
`http://www.ctan.org/tex-archive/info/lshort/`

It is recommended to take a little time out to learn how to use LaTeX by creating several, small 'test' documents. Making the effort now means you're not stuck learning the system when what you *really* need to be doing is writing your thesis.

2.2.2 A Short Math Guide for LaTeX

If you are writing a technical or mathematical thesis, then you may want to read the document by the AMS (American Mathematical Society) called, "A Short Math Guide for LaTeX". It can be found online here:
`http://www.ams.org/tex/amslatex.html`
under the "Additional Documentation" section towards the bottom of the page.

2.2.3 Common LaTeX Math Symbols

There are a multitude of mathematical symbols available for LaTeX and it would take a great effort to learn the commands for them all. The most common ones you are likely to use are shown on this page:
`http://www.sunilpatel.co.uk/latexsymbols.html`

You can use this page as a reference or crib sheet, the symbols are rendered as large, high quality images so you can quickly find the LaTeX command for the symbol you need.

2.2.4 LaTeX on a Mac

The LaTeX package is available for many systems including Windows, Linux and Mac OS X. The package for OS X is called MacTeX and it contains all the applications you need – bundled together and pre-customised – for a fully working LaTeX environment and workflow.

MacTeX includes a dedicated LaTeX IDE (Integrated Development Environment) called "TeXShop" for writing your '`.tex`' files and "BibDesk": a program to manage your references and create your bibliography section just as easily as managing songs and creating playlists in iTunes.

2.3 Stage 4 for implementation

If you are familiar with LaTeX, then you can familiarise yourself with the contents of the Zip file and the directory structure and then place your own information into the '`Thesis.cls`' file. Section 2.5 on page 9 tells you how to do this. Make sure you read section 2.7 about thesis conventions to get the most out of this template and then get started with the '`Thesis.tex`' file straightaway.

If you are new to LaTeX it is recommended that you carry on reading through the rest of the information in this document.

2.3.1 About this Template

This LaTeX Thesis Template is originally based and created around a LaTeX style file created by Steve R. Gunn from the University of Southampton (UK), department of Electronics and Computer Science. You can find his original thesis style file at his site, here:
`http://www.ecs.soton.ac.uk/~srg/softwaretools/document/templates/`

My thesis originally used the '`ecsthesis.cls`' from his list of styles. However, I knew LaTeX could still format better. To get the look I wanted, I modified his style and also created a skeleton framework and folder structure to place the thesis files in.

This Thesis Template consists of that modified style, the framework and the folder structure. All the work that has gone into the preparation and groundwork means that all you have to bother about is the writing.

Before you begin using this template you should ensure that its style complies with the thesis style guidelines imposed by your institution. In most cases this template style and layout will be suitable. If it is not, it may only require a small change to bring the template in line with your institution's recommendations.

Appendix A

Appendix Title Here

Write your Appendix content here.

Bibliography

[1] A. S. Arnold, J. S. Wilson, and M. G. Boshier. A simple extended-cavity diode laser. *Review of Scientific Instruments*, 69(3):1236–1239, March 1998.

[2] C. J. Hawthorn, K. P. Weber, and R. E. Scholten. Littrow configuration tunable external cavity diode laser with fixed direction output beam. *Review of Scientific Instruments*, 72(12):4477–4479, December 2001.

[3] Jianyong Wang and Jiawei Han. Bide: Efficient mining of frequent closed sequences. In *Proceedings of the 20th International Conference on Data Engineering*, ICDE '04, pages 79–, Washington, DC, USA, 2004. IEEE Computer Society.

[4] Carl E. Wieman and Leo Hollberg. Using diode lasers for atomic physics. *Review of Scientific Instruments*, 62(1):1–20, January 1991.