Team Amalgam

SE390 Research Plan

Joseph Hong, Chris Kleynhans, Ming-Ho Yee, Atulan Zaman
{yshong,cpkleynh,m5yee,a3zaman}@uwaterloo.ca

November 26, 2012

Abstract

SE390 Requirements Specification for Team Amalgam

Contents

1 Introduction

1.1 Purpose

This document is the requirements specification for Moolloy version 0.3, a computer-based system that will be developed by Team Amalgam. This system will be developed over the course of the fourth-year design project, which consists of the courses SE390, SE490, and SE491.

Furthermore, as the system is built on top of Moolloy, the scope of this document includes the requirements specification for Moolloy. The intended audience for this document will be researchers interested in the *guided improvement algorithm* for multi-objective optimization, or any other relational logic optimization problems.

This document allows us, and any other researchers, to develop benchmarks and tests for Moolloy. Furthermore, interested researchers may also use this document to write extensions and modify the software.

1.2 Scope

The software product described in this document will be referred to as *Moolloy v0.3*, or *Moolloy* for short. Henceforth, any references to the existing Moolloy versions will be referred to with their version numbers, or as the *original Moolloy*.

Moolloy is an implementation of the *guided improvement algorithm*, which produces Pareto-optimal solutions to general multi-objective optimization problems. Although Moolloy is built on top of a SAT solver, it is not a SAT solver.

Multi-objective optimization is an interest to many fields of science and engineering. In particular, we are interested in problems in aerospace, civil engineering, and software engineering. Many of these problems cannot be solved by Moolloy v0.2, as the input space is too large. Our work is to optimize Moolloy so it can handle problems of this scale.

Our focus is on optimizing Moolloy and how it calls the SAT solver. We are not concerned with optimizing the SAT solver itself.

1.3 Definitions, acronyms, abbreviations

Definition 1. A solution is said to be *Pareto-optimal* if and only if it is not *dominated* by any other solution. A solution a dominates a solution b if all metrics of a are greater than or equal to their corresponding metrics of b, and

there exists some metric of a that is strictly greater than its corresponding metric of b.

Definition 2. The set of all Pareto-optimal solutions is called the *Pareto-front*.

Definition 3. A *multi-objective optimization (MOO) problem* is a problem with multiple constraints, as well as multiple goals to optimize over.

Definition 4. An *exact* solution to a multi-objective optimization problem is the Pareto-front.

Definition 5. *Discrete* in this document means that there is a countable number of configurations for every problem. This is in contrast to the continuous case. A synonym for discrete is *combinatorial*, but we will only use the former term.

Definition 6. By *general-purpose*, we mean that Moolloy can solve any multi-objective optimization problem, as opposed to a specific one.

Definition 7. *SAT*, or *boolean satisfiability*, is a problem that asks whether a given Boolean formula can be assigned values such that its evaluation is true. In other words, it asks if a given Boolean formula can be satisfied.

1.4 References

Our work is an extension of the original Moolloy, which was described by Rayside, Estler, and Jackson [1].

1.5 Overview

The rest of this document describes the environment, interface, and functionality of Moolloy. We also discuss assumptions about the user, as well as other constraints and external dependencies. Finally, we describe the functional and nonfunctional requirements for the computer-based system.

2 Overall Description

2.1 Product Perspective

2.2 Product Functions

Moolloy takes as its input a multi-objective optimization problem and returns the Pareto-front.

2.3 User Characteristics

We assume that all users are already familiar with multi-objective optimization, including such terms as *Pareto-optimal* and *Pareto-front*. Furthermore, we assume users are familiar with how SAT solvers are used to find these solutions.

The user should also be familiar with expressing the problem in Moolloy's domain specific language, as well as interpreting the results.

2.4 General Constraints

2.5 Assumptions and Dependencies

As Moolloy is implemented in Java, the user will need to have the Java Runtime installed on his or her environment.

3 Specific Requirements

References

[1] D. Rayside, H.-C. Estler, D. Jackson, "The Guided Improvement Algorithm for Exact, General-Purpose, Many-Objective Combinatorial Optimization," MIT CSAIL, Tech. Rep., 2009.