Why Regression Suite?

Lakshmi Krishnamurthy

v1.0, 10 June 2012

Introduction

Regression Suite aims to incorporate performance tests as part of routine unit regression tests. In particular, the following are the design objectives:

- Generate execution time distributions and statistics
- Enhanced design to allow precise measurement of execution times
- Estimate initialization and event-specific delays
- Produce elaborate and customized regression details and outputs that are compact, queryable, and processable (in particular, fields that can be used to generate statistics).

Specifically, all these are suited for tests of analytics modules (financial analytics in particular, but other analytics too).

RegressionSuite Features

The key objective behind the design of the RegressionSuite is that it enable/ease incorporation of testing to beyond simple unitary unit testing (pass/fail), and allow for (in this edition) clock time based performance measurements. To that end, the following is a comprehensive suite of design goals and features that it handles:

- Execution Time Distribution: A first objective is the measurement of the execution time (clock time, as opposed to CPU time – to gauge the response in a given setting/environment) distribution. Measurement is done by isolating the actual regression run from the preparation and processing (more on this later). Further start-up is treated separately from the routine run, again with a view to measurement isolation. Central statistical measures such as mean and variance are generated – so are the extremal measures (minimum and maximum execution times).
- Enhanced Unit Testing: The next goal is to enhance the unit testing by extending the typical unit regression tests in a couple of ways. First, RegressionSuite is intended for

generating a variety of inputs to span the full range (this can, of course, cause it to be prohibitively expensive for practical use in many situations – strategies for input variance reduction may be employed as suitable – as in identifying the parameter-correlated valid input ranges). Next RegressionSuite also needs to distinguish between situations where the unit tests fail, but the regression is deemed to have succeeded (e.g., over specific inputs). Further RegressionSuite will be extended to capture both the execution time distribution and the unit test success/failure over the given input ranges. As noted earlier, measurement of execution times makes most sense if done by excluding the preparation and output processing times, unit tests need to account for that as well (more on this on the frame-work discussion).

- Pluggable and automated frame-work: RegressionSuite is built using interfaces and frame-works, yet with simple process controls, invocation freedom (minimal constraints), and without side-effects. Both the regression set/modules as well as the unit regressors are built around interfaces, and the core invocation/execution functionality is delegated to the frame-work. The frame-work also generates the regression statistics and other details.
- Regression Details: Details are as elaborate or brief, and their emission is controlled typically at multi-levels – at the frame-work level, at the individual module level, and at the unit regressor level. Field level details are built in by the unit regressor, and there are no limitations on the what constitutes a detail. Being named values, the details are built to be compact/efficient, parseable, and procesesable (XML is not the format of choice). Of course distribution may be generated on any of the number parsable detail fields.