Judge Documentation

Release 0.1

Martin Pool

Sep 27, 2017

Contents

Judge compares the run time for two programs and tell you if there’s a statistically significant difference.

CHAPTER 1

Quick start

Judge is stored in bzr on Launchpad:

```
$ bzr branch lp:judge
```

Suppose you're working on a program and have made some changes, and you want to know if you made it faster or slower. Judge will help:

```
$ judge 'bzr2.3 status' 'bzr2.5 status'
    n        mean      sd       min       max cmd
   50     332.8ms       8.1     320.6     355.1 bzr2.3 st
   50     255.0ms       7.4     245.0     283.1 bzr2.5 st
  -77.787ms  -23.4% p=0.000
difference is significant at 95.0% confidence (p=0.000):
```

Judge does two main things:

1. It runs the commands alternately several times, and measures the time they take. Judge discards the first run of each of the commands, since it may be overly affected by caches warming up.
2. It does a small bit of mathematics to provide an opinion on whether the observed difference is likely to be repeatable. Specifically it uses Welch's t-test for a two-sample unpooled t-test with unequal variances.

CHAPTER 2

Specifying commands

The first two arguments to judge are the command lines to test.

Note that the commands are executed directly, not through a shell, so you cannot use shell metacharacters such as quoting and redirection.

Note also that while Unix shells will expand variables within quoted strings, they typically will not expand ~ for $HOME and they do no expansion within single quotes.

```
judge "ls ~/dir1" "ls ~/dir2"             # probably not what you want
judge 'ls $HOME/dir1' 'ls $HOME/dir2'     # nor this
judge "ls $HOME/dir1" "ls $HOME/dir2"     # works
```

Judge logs the output from the commands into judge.log in the current directory. You can use this to check that the commands did what you intended them to do. This file is replaced on each run.

CHAPTER 3

Interpreting the results

There are three possible outcomes from a Judge trial: an insubstantial result, an insignificant result, or a significant substantial result.

Insubstantial result

There is no substantial difference in mean run time between the two commands (by default, less than a 1% difference):

```
 n        mean        sd        min        max cmd
50      306.7ms       0.5      304.1      307.3 sleep 0.3
50      306.9ms       0.3      306.0      307.2 sleep 0.3
+0.121ms   +0.0% p=0.155
difference is not substantial (+0.0% change)
```

In this case the distributions of values may greatly overlap, so it's hard to tell whether the specific difference between the means arose by chance or not. However, for the purposes of benchmarking programs, just knowing the means were very similar tells you that there's probably no difference.

Insignificant result

If difference in mean run time is small compared to the variability of the two programs, and it's hard to be sure the difference did not just arise by chance.

```
 n        mean         sd       min        max cmd
50        7.7ms       0.4       5.8        8.6 sleep 0.001
50        7.8ms       0.2       7.5        8.6 sleep 0.0011
+0.091ms   +1.2% p=0.084
difference is probably not significant at 95.0% confidence (p=0.084)
```

In this case you may want to repeat the test with a larger number of runs, or look for a benchmark that will make the effect you're trying to measure stronger, or perhaps look for other sources of extrinsic variability.

Substantial significant result

There was a difference that was both significant (compared to the variance of the measurements) and substantial (compared to the absolute values)

```
   n        mean       sd       min       max cmd
  50     106.9ms       0.2     106.6     107.6 sleep 0.1
  50     206.9ms       0.2     205.9     207.7 sleep 0.2
+99.957ms  +93.5%
difference is significant at 95.0% confidence (p=0.000):
```

CHAPTER 4

Cautions

Judge tries to help you generate a statistically robust measurement, but you should still be careful in using the results:

Cautions about benchmark commands

- The commands, hardware, or test data in your experiment might not be representative of the cases your users care about.
- The commands may not be doing what you think they are: check `judge.log` to see their output.
- The commands might be dominated by startup time, and that may not be what you really care about at the moment.
- The 'hot' case of running the commands repeatedly may not be representative of the case you care about.

Cautions about interpreting the results

- Resist the temptation to re-roll until you do get an apparently significant result. With enough experiments, you will eventually get an apparently significant difference, but this doesn't mean there actually is a difference.
- The measured times may be perturbed by some other activity on the machine. Judge runs the commands repeatedly and alternately to reduce the chance that any particular command will be affected by other tasks, but it is certainly possible.
- The fact that a result seems unlikely to have arisen by chance (low p-value) is no guarantee that it did not in fact arise by chance.
- Do not attempt to over-interpret the p-value: the best pratice is to set a level in advance at which the results will be considered significant and then they either reach that or not. It's normal for it to vary from one run to another. Don't think of it as being "nearly significant": if there is not enough data to get a significant result, you may not know how close you are.

- Judge only looks at the difference in mean run time. However, it may be that what matters most for your situation is something else, perhaps the 95% worst-case result, or the chance of exceeding a particular time budget.

Cautions about the statistics

- The t-test assumes the variables are normally distributed but this may not be true.
- The author of Judge is not a statistics expert and either the code or the documentation may be wrong. (Patches and bug reports are very welcome.)

CHAPTER 5

Future work

- Make more of the parameters configurable.
- Measure other attributes than just wall clock time (cpu and system time, io, etc.) Linux reports more in <http://stackoverflow.com/questions/7205806/is-getrusage-broken-in-linux-2-6-30>.
- Add `setup.py`.
- Make Judge installable by pip.
- Perhaps control how many tests to run by either fitting them in a particular overall budget, or by looking at the variability of the first few runs.
- Log the raw data in a reusable form.
- Control the log directory.
- Report whether the commands fail.
- Allow specifying non-measured cleanup or teardown commands, to establish test data and to allow caches to either be filled or emptied.
- Perhaps take some better approach to the insubstantial-difference case to help understand whether the difference is significant or not.

  It might be better to compute: what is the p-value for the hypothesis the population means differ by more than 1%?
- Allow running just one command, and calculate the mean/sd/etc.

CHAPTER 6

See also

FreeBSD's ministat, written by Poul Henning-Kamp, applies a similar statistical test. The main differences are that ministat draws ascii-art histograms, and Judge takes care of running the programs while ministat reads in externally measured numbers.

CHAPTER 7

Licence

Judge is Copyright 2011 Martin Pool.

Judge is licensed under the Apache License, Version 2.0.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.