REVERSE DEPENDS: SOME PRELIMINARY COMMENTS

DSC 2018 / R FOUNDATION SUMMIT 2018

Dirk Eddelbuettel

9 July 2018

University of Illinois at Urbana-Champaign
Debian and R Projects
Current Situation

CRAN packages since 2017, in thousands, log scale

DSC 2018
Package Growth

- Still mostly-exponential growth at CRAN
- But CRAN team *smaller* than a few years ago
- Pressure on CRAN administrators
- Pressure on package authors
Status Quo

- Expected to run prior to submission
- Not really documented
- “just look at packages tools and utils”
- devtools has one helper function ...
- ... which is a start but too slow / basic for advanced use
No Free Lunch

- Single run on a decent machine now takes more than a workday
- Should be easy-ish to parallelize (given resources)
- But that has not yet happened.
- Is testing support a community thing? R Hub?
New(-ish) package

Quick walk through

• Function to enqueue
• Function to dequeue
• Function to summarize
• No aggregation across time and runs yet
$ git/prrd/inst/scripts/enqueueJobs.r \
  -q tmp/queues/ RcppArmadillo

which calls (after **docopt** argument processing)

```
  prrd::enqueueJobs(opt$PACKAGE, opt$queue)
```

DSC 2018
$ git/prrd/inst/scripts/dequeueJobs.r \
-q tmp/queues/ RcppArmadillo

which calls (after docopt argument processing)

prrd::dequeueJobs(opt$PACKAGE, opt$queue, opt$exclude)
$ git/prrd/inst/scripts/summariseJobs.r \ 
  tmp/queues/Rcpp_2018-06-23/queuefile.sqlite -e -f

which calls (after `docopt` argument processing)

```r
prrd::summariseQueue(opt$package, "", 
  dbfile=opt$QUEUEFILE, 
  extended=opt$extended)
```
Test of RcppArmadillo had 463 successes, 3 failures, and 15 skipped packages.
Average of 23.034 secs relative to 137.058 secs using 6 runners

Failed packages: catSurv, DDPGPSurv, saeRobust

Skipped packages: cliqueMS, dynamichazard, growcurves, growfunctions, joineRML, miceadds, \nMorpho, mrgsolve, nlmixr, pcalg, rFTRLProximal, riskRegression, RStoolbox, Rvcg, RxODE

None still working

None still scheduled

Error summary:

<table>
<thead>
<tr>
<th>package</th>
<th>missingPkg</th>
<th>badInstall</th>
<th>error</th>
<th>fail</th>
<th>warn</th>
<th>note</th>
<th>ok</th>
<th>hasOtherIssue</th>
</tr>
</thead>
<tbody>
<tr>
<td>catSurv</td>
<td>FALSE</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>0</td>
<td>TRUE</td>
</tr>
<tr>
<td>DDPGPSurv</td>
<td>‘mc2d’</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>passes once installed</td>
</tr>
<tr>
<td>saeRobust</td>
<td>FALSE</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>0</td>
<td>FALSE</td>
</tr>
</tbody>
</table>

Results (queues and summaries) pushed to repo rcpp-logs.
Category in CRAN-robot emails

- Useful metric
- How can we support it?
- Comparison across time / runs in prrd
For reverse depends

- Maybe
- Plan ?
- Contributors ?
- CRAN integration ?
No Free Lunch

- Do we need to rethink testing?
  - only packages which themselves are impactful? *(maybe)*
  - only packages which were updated recently? *(maybe not)*
  - only packages which may have failed in the past? *(possibly)*
  - other ways to subsample?

- This both an engineering and a statistics questions so ...
Still No Free Lunch

- Tests really only run the code they cover
- Rcpp has e.g. code generators, we generally do not regenerate in *client packages*
- The *one minute cap* via CRAN Policy means we *suppress tests*
Where to go from here

- Anybody else interested?
- More open questions than answers