The GRAN repository system and reproducibility tools

Gabriel Becker

June 16, 2015

Contents

1 Introduction

GRAN is an open source set of tools for testing and deploying *R* packages as package repositories for both general deployment and result reproduction. It is based on the switchr framework, and allows users to deploy package manifests as validated repositories.It is centered around the R repository mechanism for pacakge distribution. GRAN provides three major areas of functionality:

1. The ability to create one or more *R* repositories by pulling and testing packages from diverse locations (scm, local directory), in a manner conducive to continuous integration
2. Tools for recreating specific environments based on sessionInfos, and for creating lightweight virtual repositories which serve the exact package versions specified in the sessoinInfo
3. Tools for assessing the potential impact of upgrading a package, to assist administrators in keeping systems up-to-date while protecting reproducibility and comparability of results in long-running core applications.

2 Creating GRAN repositories

GRAN repositories are based on *package manifests* (PkgManifest or SeedingManifest objects from the switchr framework)

Given a manifest, initial construction and rebuilding of individual GRAN repositories (referred to as *subrepositories* because GRAN supports a form of branched deployment) is performed via the makeRepo function. For example:

```
> testpkgs = list.files(system.file("testpkgs", package = "GRANBase"), full.names = TRUE)
> man = PkgManifest(name = basename(testpkgs),
+     url =  testpkgs, type = "local")
> repdir = file.path(tempdir(), "repos")
> dir.create(repdir)
> repo = makeRepo(man, repo_name= "stable", basedir = repdir,
+     destination = repdir,
+     cores = 1L, install_test = FALSE, check_test = FALSE)
```

NOTE: In the above code, we disabled the installation and R CMD check-related tests due to not playing well with the CRAN build system. In most cases, these should be TRUE in order to create a validated package repository. Also note that in the output below, the willfail package appears in the repository. This would not be the case if the check test was turned on, as it is engineered as a test case to fail check.

```
> available.packages(repo, type="source")
```

```
           Package     Version  Priority Depends
GRANBase   "GRANBase"  "1.0.4"  NA       "switchr (>= 0.9.4), methods"
GRANstable "GRANstable" "0.9.93" NA       "GRANBase"
deptest    "deptest"   "1.0"    NA       "toypkg"
switchr    "switchr"   "0.9.6"  NA       "methods"
toypkg     "toypkg"    "1.0"    NA       NA
willfail   "willfail"  "1.0"    NA       NA
           Imports                LinkingTo Suggests                       Enhances
GRANBase   "tools, XML, hwriter" NA        "parallel, BiocStyle"          NA
GRANstable NA                    NA        "BiocStyle"                    NA
deptest    NA                    NA        NA                             NA
switchr    "tools"               NA        "BiocInstaller, RJSONIO, RCurl" NA
toypkg     NA                    NA        NA                             NA
willfail   NA                    NA        NA                             NA
```

```
           License        License_is_FOSS License_restricts_use OS_type Archs
GRANBase   "Artistic-2.0" NA              NA                    NA      NA
GRANstable "Artistic-2.0" NA              NA                    NA      NA
deptest    "Artistic-2.0" NA              NA                    NA      NA
switchr    "Artistic-2.0" NA              NA                    NA      NA
toypkg     "Artistic-2.0" NA              NA                    NA      NA
willfail   "Artistic-2.0" NA              NA                    NA      NA
           MD5sum                             NeedsCompilation File
GRANBase   "fe92e19bef8bbab7e6a9ddc479cebdff" "no"             NA
GRANstable "634b6faa0cb742a441edc410e2d88ae4" "no"             NA
deptest    "9bb01da36ceee9b72662f11379bc9535" "no"             NA
switchr    "72d69b816d447bd7df6756b15dfd9c53" "no"             NA
toypkg     "f0861a527b1f24dfb1f781c63036be53" "no"             NA
willfail   "4ddf54a3b8eb0ad0982d825d9fa2416c" "no"             NA
           Repository
GRANBase   "file:///tmp/RtmpT4nkvO/repos/stable/src/contrib"
GRANstable "file:///tmp/RtmpT4nkvO/repos/stable/src/contrib"
deptest    "file:///tmp/RtmpT4nkvO/repos/stable/src/contrib"
switchr    "file:///tmp/RtmpT4nkvO/repos/stable/src/contrib"
toypkg     "file:///tmp/RtmpT4nkvO/repos/stable/src/contrib"
willfail   "file:///tmp/RtmpT4nkvO/repos/stable/src/contrib"
```

We refer readers to the documentation for that function regarding the customization options.

GRAN represents (sub)repositories as *GRANRepository* objects. These objects contain all the information required to build and deploy the repository.

Once a GRAN repository is created, its *GRANRepository* object is saved within the created directory structure as the repo.R file. This allows future builds to be invoked by the simpler syntax of passing a *GRANRepository* object or path to a created repository to makeRepo directly:

```
> repo = makeRepo(file.path(repdir, "stable"), cores=1L)
```

The makeRepo function also accepts a build_pkgs argument, which will cause only the specified packages (and their reverse dependencies) to be rebuilt, regardless of changes in version number.

```
> repo2  = makeRepo(repo, build_pkgs=basename(testpkgs)[1],
+     cores = 1L)
```

3 The repository build process

GRAN performs the following steps when creating or updating a repository. At the end of each step, the packages' statuses are updated to reflect the results of that step.

1. Up-to-date copies of package sources are obtained for each package being built, including updating previously checked out versions
2. Packages whose versions have changed since their last successful build, or who are reverse dependencies of such a package, are built without vignettes into a temporary repository via R CMD build.
3. Packages which successfully built, along with their GRAN , CRAN, and *Bioconductor*-based dependencies, are installed into a temporary library location.
4. Packages which successfully installed are built again, with vignettes, into a staging directory.
5. Remaining packages are tested via R CMD CHECK, and their statuses are updated accordingly
6. Packages which meet the requirements set for the repository (CHECK warnings and notes can be all owed, or not) are deployed into the final destination repository
7. The GRAN manifest is updated to reflect the build results
8. An HTML build report is generated from the updated manifest
9. The manifest and *GRANRepository* object are saved
10. The *GRANRepsitory* object is returned

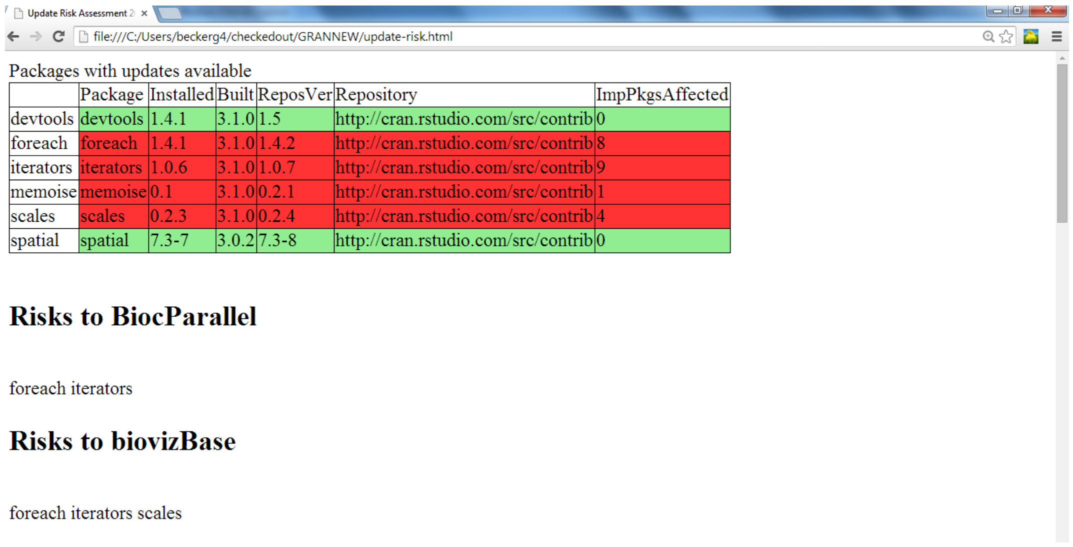
4 Tools for managing repository stability

GRAN also provides tools to navigate the tension between stability and using the most up-to-date version of packages to have the latest bug fixes available.

The identifyRisk function identifies which currently installed packages can be updated, and determines the packages that could possibly be affected by updating the package. In particular, the function allows the user to identify a vector of *important* packages and assesses the risks to each of them (by default, it takes that to be the full set of installed packages).

Risk here has a dual meaning. On the one hand updating a package which an important package depends on incurs the risk of changing the important package's behavior, potentially changing results in a critical application. On the other hand, not updating a such a package may leave important bugfixes un-applied, drawing the results generated when using the important package into question.

buildRiskReport builds an HTML report which lists information about each package with an update available in an easy to digest table. It also provides a list of specific risks to each important package (packages with no risks identified are currently omitted).


Update Risk Assessment
file:///C:/Users/beckerg4/checkedout/GRANNEW/update-risk.html

Packages with updates available

	Package	Installed	Built	ReposVer	Repository	ImpPkgsAffected
devtools	devtools	1.4.1	3.1.0	1.5	http://cran.rstudio.com/src/contrib	0
foreach	foreach	1.4.1	3.1.0	1.4.2	http://cran.rstudio.com/src/contrib	8
iterators	iterators	1.0.6	3.1.0	1.0.7	http://cran.rstudio.com/src/contrib	9
memoise	memoise	0.1	3.1.0	0.2.1	http://cran.rstudio.com/src/contrib	1
scales	scales	0.2.3	3.1.0	0.2.4	http://cran.rstudio.com/src/contrib	4
spatial	spatial	7.3-7	3.0.2	7.3-8	http://cran.rstudio.com/src/contrib	0

Risks to BiocParallel

foreach iterators

Risks to biovizBase

foreach iterators scales

Figure 1: An update risk report