RDAVIDWebService: a versatile R interface to DAVID

Cristóbal Fresno[1,2] and Elmer A. Fernández[1,2]

[1]Bio-science Data Mining Group, Catholic University of Córdoba, Córdoba, Argentina.
[2]CONICET, Argentina

October 17, 2016

Abstract

Summary: the RDAVIDWebService package provides a class-based interface from R programs/scripts to fully access/control the Database for Annotation, Visualization and Integrated Discovery (DAVID), without the need for human interaction on its website (david.abcc.ncifcrf.gov). The library enhances DAVID capabilities for Gene Ontology analysis by means of GOstats-based direct acyclic graph conversion methods, in addition to the usual many-genes-to-many-terms visualization.
Availability: RDAVIDWebService is available as an R package from the Bioconductor project (www.bioconductor.org) and on the authors' website (www.bdmg.com.ar).
Contact: cfresno@bdmg.com.ar or efernandez@bdmg.com.ar

1 Introduction

One of the most accessed systems for functional genomics/proteomics analysis is the database for annotation, visualization and integrated discovery (DAVID), a web-based online bioinformatics resource (http://david.abcc.ncifcrf.gov) that aims to provide tools for functional interpretation of large lists of genes/proteins (**?**). Its access is carried mainly through a website. There is also a uniform resource locator (URL)-based application programming interface (API), to query DAVID programmatically, accessible through DAVIDQuery R package (**?**). However, the URL-API has limited capabilities, such as URL length and only works with the default settings. In the year 2012, a web service interface was made available allowing full access and control on all its functionalities except visualization (**?**). Although, it is possible to handle DAVID web services (DWS) through R, it requires high programming skills. In addition, query results are very difficult to manage since they are XML (SOAP package, **?**) or Java objects (rJava package, **?**).
Here we provide a versatile class-based R interface to access DAVID. It is an R wrapper to all DWS functionalities, with several new features such as off-line processing (allows using previously queried saved reports) and native R class

data types. Additionally it overcomes DWS visualization constraints, providing the usual many-genes-to-many-terms feature, and enhances DAVID capabilities for Gene Ontology (GO, **?**) analysis by means of GOstats-based (**?**) direct acyclic graph (DAG) conversion methods. Therefore, it expands DAVID features by allowing new developments through one of the most used computer languages in Bioinformatics, R (**?**).

2 Implementation

The package implements *RDAVIDWebService*, a reference class object by means of R5 paradigm, for DWS communication through a Java client (*RDAVIDWebServiceStub*). This allows the establishment of a unique user access point (see Figure **??**).

Figure 1: R-to-Java class diagram. `RDAVIDWebService` uses rJava to access/control DAVID web service through `RDAVIDWebServiceStub` and parse the reports back to R by `DAVIDParser`. Note that operation signatures have been omitted for simplicity.

In order to reduce Java-to-R handshaking due to parsing data structures (a time consuming computational task), the provided Java-based file report client was enhanced (*DAVIDParser*) to allow formatting of all the DAVID outputs into appropriate RDAVIDWebService S4 R classes(see Figure **??**, and section **??**). This speeds up the bottleneck data importation process. In addition, allows locally saving DAVID query to file for further analysis, as well as using website generated reports. Thus, permits using web services and website query results interchangeably.

3 Features

1. *Ease of Use*: it provides a uniform framework to access DAVID analysis straight from R without the need of ad hoc parsing queried reports.

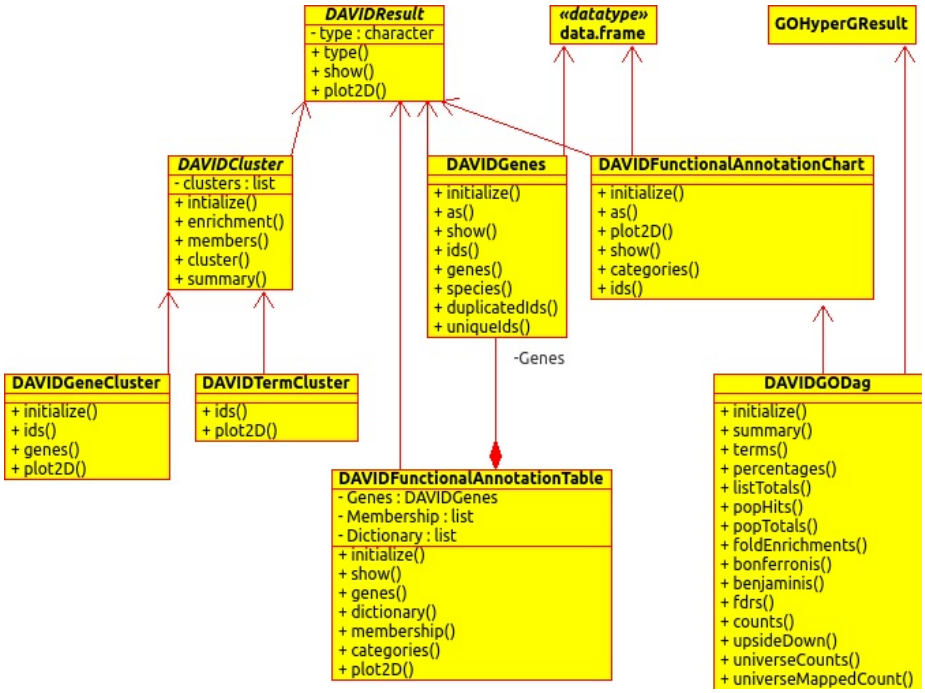

Figure 2: Hierarchy R class diagram. Note that operation signatures have been omitted for simplicity.

2. *Data import/export*: results from DAVID can be accessed through R or also generated on the website. In both cases they are stored in the same format for later use. This permits ON/OFF-line processing capabilities within R. Hence, empowering queried reports generated anytime and anywhere for processing, without the need to redo the uploading to DAVID.

3. *Visualization*: customizable many-genes-to-many-terms 2D relationship views are also available with the ggplot2 package (**?**).

4. *Gene Ontology structure*: DAVID set enrichment analysis (SEA) or modular enrichment (MEA) results can be mapped into GOstats-based direct acyclic graphs. This enables visualization of EASE score-based enriched biological process (BP), molecular function (MF) and cellular component (CC) GO terms in the DAGs. Thus, the exploration and analysis of blurred pattern presence is facilitated, compared to the usual tabular format.

4 Package overview

The package can be conceptually divided into two parts:

- **Connectivity**: a wrapper to DAVID web service for the basic work flow: (1) upload of gene/protein ids as gene/background list/s; (2) check

DAVID's status for mapped/unmapped genes in the uploaded list/s or lookup the available categories, etc. (3) select the background/species and categories to use in the present analysis, and (4) get the different reports which includes Functional Annotation Chart/Table/Clustering and so on.

- **Exploration**: on/off-line report import of the different results into R objects hierarchy (see Figure **??**), ready to use them with the use favourite CRAN (**?**) or Bioconductor (**?**) package/s. In addition, DWS capabilities are enhanced with the incorporation of the usual many-gene-to-many terms 2D relationship visualization available at DAVID's website, and the new feature which generates the induced Gene Ontology GOstats direct acyclic graph, in order to get the big biological picture, as we will show in section **??**.

4.1 Connectivity example

RDAVIDWebService requires a registered DAVID user (this is a prerequisite to use DWS). By means of the registered institutional e-mail, the user can build a `DAVIDWebService` object and establish a connection. Then, a gene list should be uploaded providing a name and type of list. Here, the one provided in the DAVID website is used (`demoList1` with Affymetrix® identifiers).
Note: the following code will not run unless you change the `"user@inst.org"` e-mail by the user registered DAVID account.

```
R> library("RDAVIDWebService")
R> david<-DAVIDWebService$new(email="user@inst.org")
R> data(demoList1)
R> result<-addList(david, demoList1,
+ idType="AFFYMETRIX_3PRIME_IVT_ID",
+ listName="demoList1", listType="Gene")
R> result

$inDavid
[1] 0.9695122

$unmappedIds
[1] "34902_at"   "1937_at"    "35996_at"   "32163_f_at" "32407_f_at"
```

The `result` output shows that 96.95% of the complete `demoList1` are recognized `$inDavid`. In addition, this object also contains the five `$unmappedIds`. On the other hand, the status of the connection is saved in `david` object.

```
R> david

DAVIDWebService object to access DAVID's website.
User email: user@inst.org
Available Gene List/s:
       Name Using
1 demoList1     *
Available Specie/s:
              Name Using
1 Homo sapiens(155)     *
Available Background List/s:
          Name Using
1 Homo sapiens     *
```

In this example, 155 genes corresponding to *Homo sapiens* are present in `demoList1`. The complete genome is selected as the default background but, the user can upload their own by modifying `listType="Background"`. If required, the user can select which annotation category to use, e.g. `GOTERM_BP_ALL`, `GOTERM_MF_ALL` and `GOTERM_CC_ALL`.

```
R> setAnnotationCategories(david, c("GOTERM_BP_ALL",
+ "GOTERM_MF_ALL", "GOTERM_CC_ALL"))
```

Now, that everything is in order, the user can get the different reports to use right away or to save into a file for future recall. For example the Functional Annotation Clustering can be obtained on-line on `termCluster` object, or as `termClusterReport1.tab` file by invoking:

```
R> termCluster<-getClusterReport(david, type="Term")
R> getClusterReportFile(david, type="Term",
+ fileName="termClusterReport1.tab")
```

4.2 Exploration example

Hereafter, we asume that at some point `demoList1` has been used and every report saved into files (data available in the package).
A user can obtain the functional annotation cluster report of `demoList1` and inspect the results using the following code:

```
R> library("RDAVIDWebService")
R> fileName<-system.file("files/termClusterReport1.tab.tar.gz",
+    package="RDAVIDWebService")
R> untar(fileName)
R> termCluster<-DAVIDTermCluster(untar(fileName, list=TRUE))
R> termCluster

DAVID Result object
Result type:  AnnotationCluster
Number of cluster:  28

R> head(summary(termCluster))
```

```
  Cluster Enrichment Members
1       1      2.904      14
2       2      2.135       4
3       3      2.059      10
4       4      1.977      14
5       5      1.501       4
6       6      1.347       4
```

Here, `termCluster` is an object from class `DAVIDTermCluster` with the corresponding `AnnotationCluster` report data of `demoList1`, where 28 clusters are found. Then, `head(summary(termCluster))` can provide a superficial view of the `Enrichment` Score reached in each cluster and how many `Members` are present. The user can visually explore the 2D view of a particular cluster (e.g. the second on Figure **??**).

```
R> clustNumber<-2
R> plot2D(termCluster, clustNumber)
```

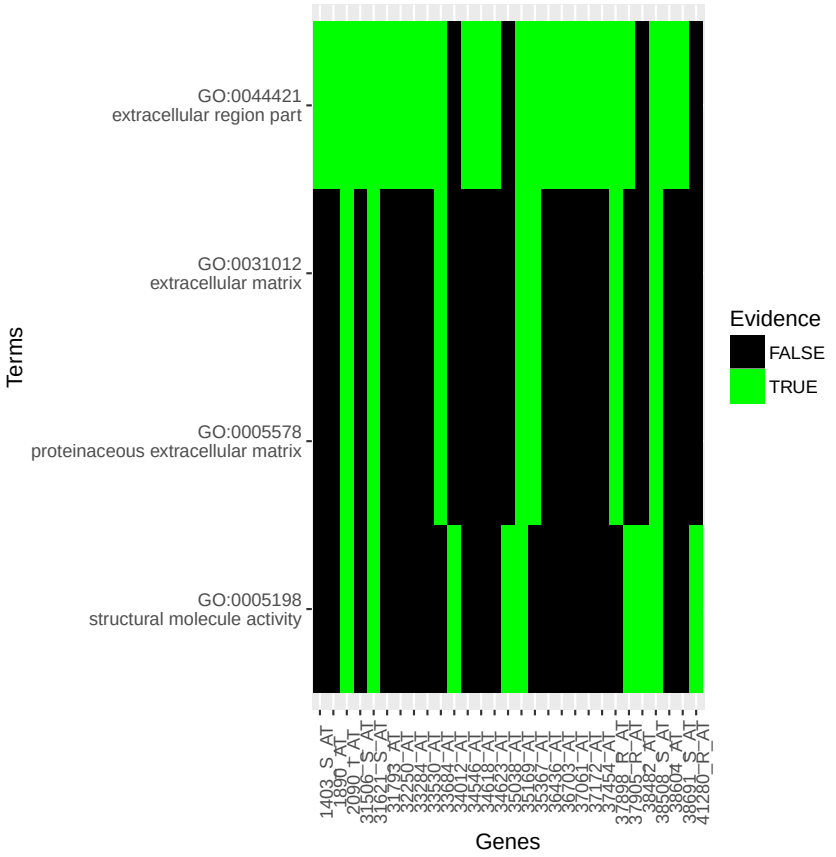

Figure 3: Functional annotation cluster exploration, using a 2D-view of the evidence of the four term/category members present in the second cluster.

However, in Figure **??**, the four term/category members of this cluster share

all the ids at "extracellular region part" (upper row). But, as we go down towards the bottom row (structural molecule activity) only nine ids have evidence related to it. In this view, the hierarchical structure of GO is not considered nor the members that are enriched or not (default option). Therefore, the user can extend DAVID's features obtaining the associated induced DAG structure of the cluster (`DAVIDGODag`) and contextualize it using GOstats functionalities (`plotGOTermGraph`, see Figure **??**).

```
R> davidGODag<-DAVIDGODag(members(termCluster)[[clustNumber]],
+    pvalueCutoff=0.1, "CC")
R> plotGOTermGraph(g=goDag(davidGODag),
+    r=davidGODag, max.nchar=40, node.shape="ellipse")
```

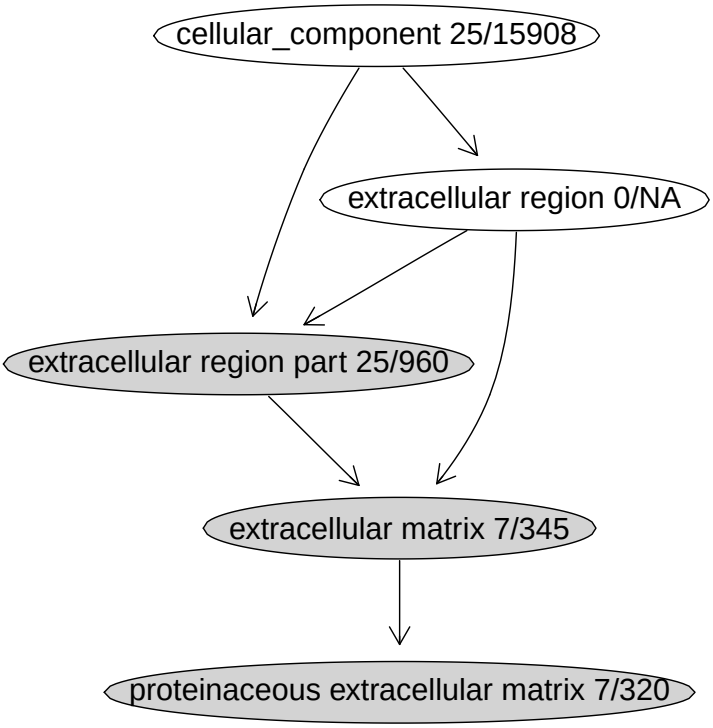

Figure 4: Gene Ontology direct acyclic graph induced by cluster two in figure **??**. Terms with an EASE score < 0.1 are shown in grey. In addition, the ratio between genes on the list vs. background reference is displayed. Since no information is available regarding the other terms into the mapped GO structure from the cluster, NAs (not available) are introduced when required.

5 Trouble shooting

Sometimes apache Axis' default parameters needs to be changed in order to use RDAVIDWebService. For example if DAVID web service is bussy the default timeout would not be enought. Then you can inspect it and change it calling:

```
R> getTimeout(david)

[1] 30000

R> setTimeout(david, 50000)
R> getTimeout(david)

[1] 50000
```

Other reported parameter that might need to be changed is the transport protocol if you get the following error when uploading a gene list:

```
org.apache.axis2.AxisFault: Transport error: 501 Error: Not Implemented
```

Then you can change on the client side the appropiate parameter value:

```
R> setHttpProtocolVersion(david, "HTTP/1.0")
R> getHttpProtocolVersion(david)

[1] "HTTP/1.0"
```

Acknowledgements

Funding: this work was supported by the National University of Villa Maria [31/0186 to E.F. and 31/0187 to E.F.] and Catholic University of Córdoba, Argentina.

Session Info

```
R> sessionInfo()

R version 3.3.1 (2016-06-21)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: OS X 10.9.5 (Mavericks)

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
 [1] grid      stats4    parallel  stats     graphics  grDevices
 [7] utils     datasets  methods   base

other attached packages:
 [1] Rgraphviz_2.18.0        GO.db_3.4.0
 [3] RDAVIDWebService_1.12.0 ggplot2_2.1.0
```

```
 [5] GOstats_2.40.0          Category_2.40.0
 [7] Matrix_1.2-7.1          AnnotationDbi_1.36.0
 [9] IRanges_2.8.0           S4Vectors_0.12.0
[11] Biobase_2.34.0          graph_1.52.0
[13] BiocGenerics_0.20.0

loaded via a namespace (and not attached):
 [1] Rcpp_0.12.7             splines_3.3.1
 [3] munsell_0.4.3           colorspace_1.2-7
 [5] xtable_1.8-2            lattice_0.20-34
 [7] plyr_1.8.4              tools_3.3.1
 [9] gtable_0.2.0            AnnotationForge_1.16.0
[11] DBI_0.5-1               genefilter_1.56.0
[13] digest_0.6.10           survival_2.39-5
[15] RBGL_1.50.0             GSEABase_1.36.0
[17] rJava_0.9-8             bitops_1.0-6
[19] RCurl_1.95-4.8          RSQLite_1.0.0
[21] scales_0.4.0            XML_3.98-1.4
[23] annotate_1.52.0
```