smrtlinkr

Nigel Delaney

October 9, 2016

`smrtlinkr` is an R package that allows users to communicate with our internal SmrtLink servers[1] to query the database, submit jobs, or obtain information about finished jobs from reports. This document provides a tutorial of the basic functionality, though more features are either available in the package or can be upon request.

Installation

From your personal computer or a PacBio RStudio server such as `bayes:8787` you can install the smrtlinkr package using the following commands:

```
install.packages("devtools")
library(devtools)
install_github("PacificBiosciences/smrtlinkr")
```

Querying Jobs and Datasets

Broadly speaking, SmrtLink servers contain two types of information:

- **Data** such as subreadsets, reference fasta file, etc. This is the raw input into its analysis pipelines.
- **Jobs** which are analyzes and processes that take data as input, and produce output such as summary reports, new BAM files with consensus reads, etc.

Interacting with the SMRTLink servers entails a serious of commands that can examine the available data on a server, submit a new job by referencing the data on the server as inputs, or examine the results/reports of a job[2].

Most commands in `smrtlinkr` for querying data of some are of the form `fetchXXX(... , server, port)` and require a server and a port to be specified. These values are particular to each unique server (smrtlink-alpha, smrtlink-beta, etc.) though by default the commands will query smrtlink-internal on port 8081.

Querying the database

Both data and jobs are identified as being of particular types, e.g. a reference data type or a subreadset data type. To access data we typically need to know the type and id of an item. The following code displays all the available types of data, and shows how we can use this type information to find all the E. coli reference sequences available on smrtlink-beta.

[1] A complete list of SmrtLink servers is available from: https://confluence.pacificbiosciences.com/display/SL/On-site+SMRT+Link+servers

[2] A more detailed description of the SMRTLink REST API can be found here: http://smrtflow.readthedocs.io/en/latest/

Querying available data

First to list all the available data types:

```
library(smrtlinkr)
# View all the available DataSet types
df = fetchDataSetTypes()
```

Available Dataset Types

- references
- ccsreads
- contigs
- subreads
- barcodes
- ccsalignments
- hdfsubreads
- alignments
- gmapreferences

If you know the type of data you are looking for, you can easily load the list of datatypes and query it in R. For example, the code below allows us to find all the E. coli reference sequences known to SMRTLink beta.

Example: Query location of E. coli reference sequences

```
# List all the reference data types that are known to the server
df = fetchDataSetsByType("references", server = "smrtlink-beta", port = 8081)
# Now let's print the path of all the E. coli ones.
kable(df[grep("coli", df$name), c("name", "path")])
```

	name	path
5	ecoliK12_pbi_March2013	/pbi/dept/secondary/siv/smrtlink/smrtlink-beta/smrtsuite_166987/userdata/jobs
24	A6_6K_ecoli_draft_assembly	/pbi/dept/secondary/siv/smrtlink/smrtlink-beta/smrtsuite_166987/userdata/jobs
129	RL_UMD_Ecoli	/pbi/dept/secondary/siv/smrtlink/smrtlink-beta/smrtsuite_166987/userdata/jobs
155	ecoli_split_1000	/pbi/dept/secondary/siv/references/ecoli_split_1000/ecoli_split_1000.referencese
244	ecoliK12_pbi_March2013	/pbi/dept/secondary/siv/references/ecoliK12_pbi_March2013/ecoliK12_pbi_Ma
361	Ecoli_Gold_5kb_panel_080715	/pbi/dept/secondary/siv/smrtlink/smrtlink-beta/smrtsuite_166987/userdata/jobs
449	rh_ecoli	/pbi/dept/secondary/siv/smrtlink/smrtlink-beta/smrtsuite_166987/userdata/jobs

For both references and subreadsets, there are convenience functions `fetchSubreadSetInfo` and `fetchReferenceSetInfo` which allow you to obtain more information about a particular dataset. For example to find out information about a particular reference dataset

Example: Getting information about a particular reference set

```
# Use the reference ID and server information to get more information
refId = df$id[20]
```

```r
# Now query for more info
info = fetchReferenceSetInfo(refId, server = "smrtlink-beta", port = "8081")
info
```

```
## $name
## [1] "gencode_pc_transcripts_V19"
##
## $updatedAt
## [1] "2016-01-21T04:08:41.124Z"
##
## $path
## [1] "/home/UNIXHOME/gconcepcion/for_grabs/ayang/fasta/gencodeV19/gencode_pc_transcripts_V19/referenc
##
## $ploidy
## [1] "haploid"
##
## $tags
## [1] ""
##
## $uuid
## [1] "2baeeaa4-d15a-4c86-8995-eb839d241639"
##
## $totalLength
## [1] 194630920
##
## $projectId
## [1] 1
##
## $numRecords
## [1] 94365
##
## $version
## [1] "3.0.1"
##
## $id
## [1] 2465
##
## $md5
## [1] "56221a83c7ec0e026c6ed1417f2831c5"
##
## $jobId
## [1] 1739
##
## $createdAt
## [1] "2016-01-21T04:08:41.124Z"
##
## $organism
## [1] "gencode_pc_transcripts_V19"
##
## $userId
## [1] 1
##
## $datasetType
## [1] "PacBio.DataSet.ReferenceSet"
```

```
##
## $comments
## [1] "reference dataset comments"
```

Example: Getting information about a particular SubreadSet

```
# Get an id for a particular subreadset
id = fetchDataSetsByType("subreads")$id[1]
# Now query for more info
fetchSubreadSetInfo(id)
```

```
## $name
## [1] "dacetoxidans/0005"
##
## $updatedAt
## [1] "2016-07-11T23:24:38.284Z"
##
## $path
## [1] "/pbi/dept/secondary/siv/testdata/SA3-DS/dacetoxidans/2530923/0005/Analysis_Results/m140906_1621
##
## $tags
## [1] ""
##
## $instrumentName
## [1] "42175"
##
## $uuid
## [1] "62c7a6be-76f4-ee87-9fed-ad5497c1c5ad"
##
## $totalLength
## [1] 810158639
##
## $projectId
## [1] 1
##
## $numRecords
## [1] 740276
##
## $wellSampleName
## [1] "Daceto Nat 6.67pM P6C4"
##
## $bioSampleName
## [1] "Daceto Nat 6.67pM P6C4"
##
## $version
## [1] "3.0.1"
##
## $id
## [1] 95
##
## $md5
## [1] "eaa8740ed1c7c1af981af003e06a2618"
```

```
##
## $jobId
## [1] 194
##
## $createdAt
## [1] "2016-07-11T23:24:38.284Z"
##
## $wellName
## [1] "C01"
##
## $cellIndex
## [1] 4
##
## $userId
## [1] 1
##
## $metadataContextId
## [1] "m140906_162143_42175_c100676202550000001823129611271444_s1_p0"
##
## $runName
## [1] "2014-09-05_NGAT-175_P6C4-Base-Mod-Training"
##
## $datasetType
## [1] "PacBio.DataSet.SubreadSet"
##
## $comments
## [1] ""
```

Examining Jobs

Similarly to datasets, jobs in SMRTLink also have a "type", the code below lists all the available job types.

```
# View all the available job types
d = fetchJobTypes()
kable(d)
```

jobTypeId	description
export-datasets	Export PacBio XML DataSets to ZIP file
convert-rs-movie	Import RS metadata.xml and create an HdfSubread DataSet XML file
mock-pbsmrtpipe	Mock Pbmsrtpipe Job used for Development purposes
pbsmrtpipe	Run a secondary analysis pbsmrtpipe job.
simple	Simple Job for debugging and development
conditions	Create a multi-analysis job pipeline by running a pbsmrtipe Condition JSON driven Pipeline
convert-fasta-barcodes	Import fasta reference and create a generated a Reference DataSet XML file.
merge-datasets	Merge PacBio XML DataSets (Subread, HdfSubread datasets types are supported)
import-datastore	Import a PacBio DataStore JSON file
convert-fasta-reference	Import fasta reference and create a generated a Reference DataSet XML file.
import-dataset	Import a Pacbio DataSet XML file

Although users may think of a job as the name of a SMRTLink procedure (e.g. Resequencing) internally all such jobs are of type `pbsmrtpipe` and so this is the type that will typically be queried to examine results on

a smrtlink server. For example

Example: List all resequencing jobs on smrtlink-beta

```
# Let's get all the pbsmrtpipe jobs
d = fetchJobsByType("pbsmrtpipe", server = "smrtlink-internal", port = 8081)

# Let's only get the resequencing jobs, the type of pbsmrtpipe job is identified
# in the comment column

# Grep the comment column to find resequencing
reseqRows = grep("resequencing", d$comment)
# Now let's list 5 of these jobs names and ids
d[reseqRows[10:15], c("name", "id")]
```

```
##                                                    name   id
## 2856            A03_poc_LVP12-D11-5247_Flea_Plate_LP4_10min 4831
## 2857            A01_poc_LVP12-D11-5247_Flea_Plate_LP4_10min 4832
## 2858            A02_poc_LVP12-D11-5247_Flea_Plate_LP4_10min 4833
## 2859 B12_poc_LVP12-D11-5247_1G1.25RSB_4FMP60mMArg_LP4_10min 4834
## 2860 B11_poc_LVP12-D11-5247_1G1.25RSB_6FMP20mMArg_LP4_10min 4835
## 2861            A03_poc_LVP12-D11-5247_Flea_Plate_LP4_10min 4836
```

Example: Get Path of subreads and reference for resequencing jobs.

Given a set of resequencing jobs, we might want to for example get the underlying reference and subread sets that were used, we can do this with the following function:

```
paths = fetchSubreadsAndReferencesForResequencingJobs(d[reseqRows[10:15], ])
kable(paths)
```

name	subread
A03_poc_LVP12-D11-5247_Flea_Plate_LP4_10min	/pbi/collections/320/3200035/r54010_20160829_
A01_poc_LVP12-D11-5247_Flea_Plate_LP4_10min	/pbi/collections/320/3200035/r54010_20160829_
A02_poc_LVP12-D11-5247_Flea_Plate_LP4_10min	/pbi/collections/320/3200035/r54010_20160829_
B12_poc_LVP12-D11-5247_1G1.25RSB_4FMP60mMArg_LP4_10min	/pbi/collections/320/3200035/r54010_20160829_
B11_poc_LVP12-D11-5247_1G1.25RSB_6FMP20mMArg_LP4_10min	/pbi/collections/320/3200035/r54010_20160829_
A03_poc_LVP12-D11-5247_Flea_Plate_LP4_10min	/pbi/collections/320/3200035/r54010_20160829_

Submitting Jobs

`smrtlinkr` can also be used to create jobs on SMRTLink, allowing users to programatically create jobs and then view them on the SMRTlink server. Below is example code which shows how to create a resequencing job using the `postReseqJob` function. You can obtain a subreadset ID and referenceset ID by looking at your smrtlink server (see data management) or you can also grab it programatically as shown below.

```
# To create a job, we'll need a job name, a reference ID and a subreadset ID
refName = "pBR322_HindIII_scr_F_EcoRI_tc6_R_unrolled6x"
subreadName = "15881P_SB_1G1.25R_A4k"

### OBTAIN A REFERENCE SET ID ###
# We can get the reference ID either by looking at SMRTLink or just querying here
refs = fetchDataSetsByType("references")
# Grep to find the ID of the one we want
refID = refs$id[grep(refName, refs$name)]

### OBTAIN A SUBREAD SET ID ###
# Same situation
subreads = fetchDataSetsByType("subreads")
subID = subreads$id[grep(subreadName, subreads$name)]

# Now we just submit the job, calling it whatever we want
postReseqJob("Submitted from R!", refID, subID)
```

```
## Response [http://smrtlink-internal.nanofluidics.com:8081/secondary-analysis/job-manager/jobs/pbsmrtp
##   Date: 2016-10-11 01:21
##   Status: 201
##   Content-Type: application/json; charset=UTF-8
##   Size: 2.2 kB
## {
##   "name": "Submitted from R!",
##   "updatedAt": "2016-10-10T18:21:19.804-07:00",
##   "path": "/pbi/dept/secondary/siv/smrtlink/smrtlink-internal/userdata/j...
##   "state": "CREATED",
##   "uuid": "54214f63-f99e-4300-836f-8cf24da335d7",
##   "jobTypeId": "pbsmrtpipe",
##   "id": 5281,
##   "comment": "pbsmrtpipe pbsmrtpipe.pipelines.sa3_ds_resequencing",
##   "createdAt": "2016-10-10T18:21:19.804-07:00",
## ...
```