SPANG Documentation

spang

Aug 15, 2020

Contents

CHAPTER 1

Overview

An increasing number of biological databases have been made available in the form of Resource Description Framework (RDF) and accessible through SPARQL endpoints, forming together a worldwide platform of biological data integration across the web. However, writing SPARQL codes for querying the databases often becomes a burden for biologists; thus, an easy-to-use querying interface is necessary.

Here, we developed a command-line SPARQL client, SPANG. SPANG can dynamically generate typical SPARQL queries depending on command-line options and arguments. SPANG supports interprocess communication to pass the variable bindings between queries, enabling execution of combination of queries and integration of data across the multiple SPARQL endpoints. SPANG can also search local RDF files besides remote data stores. These features provide the users an easy ways to integrate distributed data described in RDF, thus enhances the integrative analysis of biological data on the Semantic Web platform.

CHAPTER 2

Use in command line

2.1 Installation

```
$ git clone git@github.com/hchiba1/spang.git
$ cd spang
$ npm install
$ npm link
```

2.2 Usage

```
Usage: spang2 [options] <SPARQL_TEMPLATE>

Options:
  -V, --version                output the version number
  -f, --format <FORMAT>        tsv, json, n-triples (nt), turtle (ttl), rdf/xml␣
↪(rdfxml), n3, xml, html (default: "tsv")
  -e, --endpoint <ENDPOINT>    target endpoint
  -S, --subject <SUBJECT>      shortcut to specify subject
  -P, --predicate <PREDICATE>  shortcut to specify predicate
  -O, --object <OBJECT>        shortcut to specify object
  -F, --from <FROM>            shortcut to search FROM specific graph (use alone or␣
↪with -[SPOLN])
  -N, --number                 shortcut of COUNT query (use alone or with -[SPO])
  -G, --graph                  shortcut to search Graph names (use alone or with -
↪[SPO])
  -a, --abbr                   abbreviate results using predefined prefixes
  -q, --show_query             show query and quit
  -L, --limit <LIMIT>          LIMIT output (use alone or with -[SPOF])
  -l, --list_nick_name         list up available nicknames and quit
  --param <PARAMS>             parameters to be embedded (in the form of "--param␣
↪par1=val1,par2=val2,...")
```

(continues on next page)

(continued from previous page)

```
  -h, --help                    output usage information
```

2.3 Shotcut mode

Only ten triples are obtained from the target endpoint.

```
spang2 -L 10
```

List of graphs are obtained.

```
spang2 -G
```

To obatain the list of target endpoints,

```
spang2 -l
```

```
spang2 -F http://ddbj.nig.ac.jp/ontologies/taxonomy/ -L 10
```

```
spang2 -S http://ddbj.nig.ac.jp/ontologies/taxonomy/Taxon
```

```
spang2 -O http://ddbj.nig.ac.jp/ontologies/taxonomy/Taxon -L 10
```

```
spang2 -O http://ddbj.nig.ac.jp/ontologies/taxonomy/Taxon -P rdf:type -L 10
```

```
spang2 -O http://ddbj.nig.ac.jp/ontologies/taxonomy/Taxon -P rdf:type -N
```

```
spang -S taxon:Taxon
```

```
spang -S taxon:Taxon -a
```

```
spang2 -S taxon:Taxon -P rdf:type -L10 -a
```

```
spang2 test/tax/count_species.rq -a --param name=Primates
```

CHAPTER 3

Use in JavaScript

- Download `spfmt_bundled.js` and use it in your HTML.

```
<script src="/js/spfmt_bundled.js"></script>
```

- Then you can use `spfmt.reformat`.

```
spfmt.reformat("SELECT * WHERE {?s ?p ?o}");
/*
SELECT *
WHERE {
    ?s ?p ?o .
}
*/
```

- You can also call `spfmt_bundled.js` through the jsDelivr service.

```
    <textarea id="sparql-text" rows=5></textarea>
    <button id="reformat-button">Reformat</button>
    <textarea id="sparql-text-after" rows=5></textarea>

    <script src="https://cdn.jsdelivr.net/gh/hchiba1/spang@master/js/spfmt_bundled.js
↪"></script>
    <script type="text/javascript">
     window.onload = () => {
         var textArea =
             document.querySelector("#reformat-button").addEventListener('click',␣
↪(event) => {
                 document.querySelector("#sparql-text-after").value =
                     spfmt.reformat(document.querySelector("#sparql-text").value);
             });
     };
    </script>
```

CHAPTER 4

SPANG library

CHAPTER 5

SPANG API

We developed a SPANG library API to search for the set of SPARQL queries. SPANG library is accessible through SPANG Web API. The following command returns the list of libraries.

```
curl https://spang-portal.dbcls.jp/api/library
```

The results is shown below as JSON format.


```
[
  {
    "name": "disgenet",
    "title": "DisGeNET",
    "description": "DisGeNET",
    "endpoint": "http://rdf.disgenet.org/sparql/",
    "schema": "http://www.disgenet.org/web/DisGeNET/menu/rdf#schema",
    "uri": "http://localhost:7070/api/library/disgenet",
    "count": 14
  },
  {
    "name": "wikipathways",
    "title": "WikiPathways",
    "description": "WikiPathways",
    "endpoint": "http://sparql.wikipathways.org/",
    "schema": null,
    "uri": "http://localhost:7070/api/library/wikipathways",
    "count": 6
  },
  ...
]
```


The following command returns the list of query in a library.

```
curl https://spang-portal.dbcls.jp/api/library/disgenet
```

The results is shown in below as JSON format.

```
{
  "disgenet": [
    {
      "name": "disease_gene",
      "title": "Get UniProt IDs for a specific disease, e.g. C0751955 (\"Brain␣
↪Infarction\")",
      "uri": "http://localhost:7070/api/disgenet/disease_gene",
      "endpoint": "http://rdf.disgenet.org/sparql/",
      "param": [
        {
          "name": "arg1",
          "default": "C0751955"
        }
      ]
    },
    {
      "name": "disease_uniprot",
      "title": "Get UniProt IDs for a specific disease, e.g. C0751955 (\"Brain␣
↪Infarction\")",
      "uri": "http://localhost:7070/api/disgenet/disease_uniprot",
      "endpoint": "http://rdf.disgenet.org/sparql/",
      "param": [
        {
          "name": "arg1",
          "default": "C0751955"
        }
      ]
    },
    {
      "name": "gda_alzheimer",
      "title": "Alzheimer's disease-related genes with curated evidences",
      "uri": "http://localhost:7070/api/disgenet/gda_alzheimer",
      "endpoint": "http://rdf.disgenet.org/sparql/",
      "param": [

      ]
    },
    {
      "name": "gda_evidence",
      "title": "Get specific gda, e.g. NCBI gene ID 4204 and C0035372(\"Rett Syndrome\
↪")",
      "uri": "http://localhost:7070/api/disgenet/gda_evidence",
      "endpoint": "http://rdf.disgenet.org/sparql/",
      "param": [

      ]
    },
    ...
  ]
}
```


The following command searches for queries with keywords.

```
curl https://spang-portal.dbcls.jp/api/search?keyword=gene
```

Using SPANG API, users can execute SPARQL query over the Web, with the given parameters.

The following command returns the result of the query, with the given parameter.

```
curl https://spang-portal.dbcls.jp/api/library/disgenet/gene\_disease.rq?arg1=678
```

CHAPTER 6

Adding metadata to SPARQL

CHAPTER 7

Formatting SPARQL

SPANG specifies formatting of SPARQL.

`spfmt` is a SPARQL formatter written in JavaScript.

It can be used in a web site or in the command line.

An example web site: https://hchiba1.github.io/sparql-utils/

7.1 Usage on a web site

- Download `spfmt_bundled.js` and use it in your HTML.

```
<script src="/js/spfmt_bundled.js"></script>
```

- Then you can use `spfmt.reformat`.

```
spfmt.reformat("SELECT * WHERE {?s ?p ?o}");
/*
SELECT *
WHERE {
    ?s ?p ?o .
}
*/
```

- You can also call `spfmt_bundled.js` through the jsDelivr service.

```
    <textarea id="sparql-text" rows=5></textarea>
    <button id="reformat-button">Reformat</button>
    <textarea id="sparql-text-after" rows=5></textarea>

    <script src="https://cdn.jsdelivr.net/gh/hchiba1/sparql-utils@master/spfmt/src/
↪spfmt_bundled.js"></script>
    <script type="text/javascript">
```

(continues on next page)

(continued from previous page)

```
    window.onload = () => {
        var textArea =
            document.querySelector("#reformat-button").addEventListener('click',␣
↪(event) => {
                document.querySelector("#sparql-text-after").value =
                    spfmt.reformat(document.querySelector("#sparql-text").value);
            });
    };
   </script>
```

7.2 Usage in command line

7.2.1 Requirements

- Node.js (>= 11.0.0)
- npm (>= 6.12.0)

7.2.2 Installation

```
$ npm install
$ npm link
```

7.2.3 Usage

```
$ cat messy.rq
SELECT * WHERE          {          ?s ?p ?o }

$ spfmt messy.rq
SELECT *
WHERE {
    ?s ?p ?o .
}
```

7.2.4 Test examples

If you have globally installed mocha

```
$ mocha
```

Otherwise,

```
$ ./node_modules/mocha/bin/mocha
```

7.3 Update spfmt_bundled.js

Update the `spfmt_bundled.js` as follows after editing any other JS codes

```
$ ./node_modules/browserify/bin/cmd.js src/spfmt_browser.js > src/spfmt_bundled.js
```

CHAPTER 8

FAQ

spang2 command fails, even though it previously runs.

Try `npm install` again.

```
cd spang2
npm install
```