Getting Started with GRUFF

Introduction

Most articles in this book focus on interesting applications of Linked Open Data (LOD). But this chapter describes some simple steps on how to use a triple store, how to load linked open data, and how to create SPARQL queries with a graphical query builder. This should allow users new to these topics to better understand the methods and techniques and thus to better understand the more complex examples later in the book.

What are triples and what is RDF?

For completeness we'll introduce the concept of triples here but we assume that the readers of this book are familiar with the RDF stack. The Resource Description Framework (RDF) language is used to express data about resources, where "resources" can be interpreted to be anything (a web page, a person, an idea, etc.). The basic building block is the triple, consisting of subject, predicate, object. The subject is a URI, the predicate is some property that is defined for the type (class) of the subject, and the object is either a typed literal or the URI of some other subject. Let's look at a couple of assertions that express data about resources:

```
bb:YogiBerra rdf:type bio:Person  .

bb:YogiBerra bb:playsPosition bb:Catcher  .

bb:YogiBerra bb:careerHomeRuns 358 .
```

The first assertion says that a "resource," Yogi Berra, whose URI is defined in the bb: namespace, is of type Person (where the meaning of Person is defined in the `bio: namespace`), he played the catcher position (where the meaning of `playsPosition` and `Catcher` are defined in the `bb: namespace`), and he had `358` career home runs (where the meaning of `careerHomeRuns` is defined in `the bb: namespace`). This is what data looks like in RDF: triples expressed as a subject, a predicate, and an object, separated by spaces, and concluded with a period.

The above description comes from a little mini course in RDF that can be found on the Franz website (http://www.franz.com).

What is a triple store? An introduction to AllegroGraph

Most Linked Open Data comes in the form of files containing RDF triples. In order to work efficiently with triples you need to have a *triple store database,* that is specialized for storing the triples data format. A good triple store allows you to store triples and index them for fast retrieval, to perform SPARQL queries, and to reason dynamically or through materialization. AllegroGraph is such a triple store with some additional unique capabilities.

AllegroGraph Provides:

- All essential enterprise capabilities you expect in a major relational database: ACID transactions (Atomicity, Consistency, Isolation and Durability), backup/restore, point in time recovery, security, replication, warm fail over, clustering, triple level security.
- Geospatial reasoning, temporal reasoning, and social network analysis. These features are all directly accessible in SPARQL.
- Business rules with an ISO compatible Prolog compiler
- Server side JavaScript stored procedures [faulty parallelism. Why not delete Deploy?]
- Gruff, a powerful visualization tool which allows user friendly navigation of triples. Gruff's graphical query editor allows easy composition of SPARQL queries.
- The ability to automatically discover patterns by highlighting nodes and turning them into SPARQL queries

The example

In our example we are going to work with a data set that we extracted from DBPedia, the triple version of the Wikipedia. We took all the information about movies and actors, producers and directors and stored that in a single file (N-Triples Format). This file can be downloaded from our website (see instructions below)

We are going to use a powerful visual navigation tool called Gruff. Gruff is one of the interfaces to AllegroGraph and it allows you to create a new triple store, download triple files to populate the store, and then query triples or display triples on the screen. Gruff comes in two forms, a standalone version that includes a basic version of AllegroGraph, and the server edition. You will want the server edition if you are working with hundreds of millions to billions of triples.

If you just want to look at a few million triples and you don't have easy access to a Linux Server, then you can just install the standalone version. We are going to use the standalone version in this tutorial.

Installing and starting the standalone version Gruff:

Visit `http://www.franz.com/agraph/gruff` and go to the download section. Assuming you have a 64-bit Windows machine you should download Gruff v5.0.x for AG 3.3.

Extract the file that you downloaded into a convenient location (which we will refer to as the *Gruff directory*).

Go into the Gruff directory and double click `'gruff.exe'`

Downloading the dataset for our example:

The data for or example is in:
http://www.franz.com/franzdownload/agraph/allegrograph/data/actors.ntriples

Please unzip it and place it in a convenient place.

Creating a new triple store:

Creating a triple store is now about as easy as starting an Excel spreadsheet.

First we create a new triple store: *File -> New Triple-Store*

Because we work with a standalone version we use the name of your local machine, Gruff will probably fill it in for you already. As you can see my laptop is called `JansSamsung`. Note that you don't have to fill in a port number. For the Store Folder you will type in the full name of the triple store you are going to create. Make sure it is not an existing directory because it will overwrite that.

Once you click ok, the database will ask you how many triples you expect. Just accept the default. This number is only important if you know you are going to use millions of triples.

Loading data into Gruff:

Now we are going to load the file with movies and actors in Gruff.

File->Load Triples->Load N-Triples

Gruff will ask if you want to load the triples from a file or from the web. Chose '`file`' for this tutorial.

And find the place where you stored the file actors.ntriples downloaded from the Franz Inc website. Select it and load. You will see a yellow bar for a few seconds and if that bar disappears the data is ready to be used.

Displaying some triples on the screen:

To quickly test the data was loaded. From the Gruff menu:

Display->Display All Triples Up To A Limit.

And after a few seconds you'll see activity on your screen. Use the wheel on your mouse (or shift- or shift+,) to zoom in and out and then press the letter 'r' to reformat the screen.

Just for fun you might want to click on a node and go to the Tabular View. Click around a little bit to become familiar with the data. Go back to the Graph View (*View->Graph View*) or press the letter 'g'.

Now we are going to delete all the information from the screen by *Remove->Remove-All-Nodes* (Don't worry, it won't delete any triples, it will just remove the nodes from the screen)

Creating a freetext index and find Kevin Bacon:

In many cases you start exploring a set of files by typing in some of the concepts that you know that might be in the file. For that we need text indexing (i.e. Key Word Search, like Google).

Display -> Edit Free Text Predicates

A widget will pop up, just select all and then click ok.

Now we want to find Kevin Bacon:

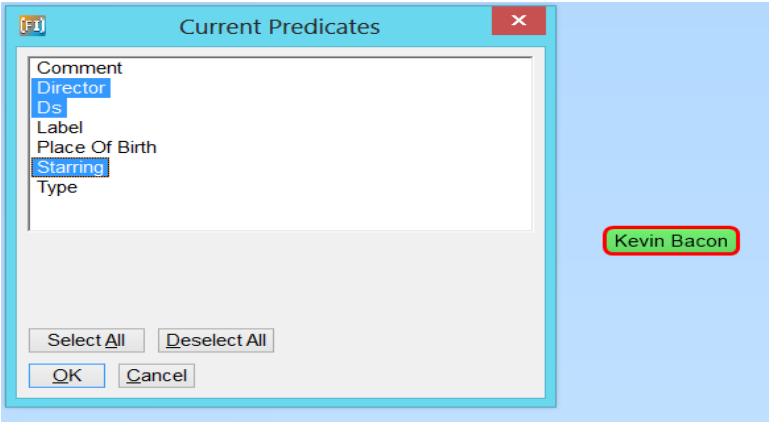
Display-> Display Triples by Freetext Index (or press ';') and type `Kevin Bacon` in the search field.

Browse through the results and choose `Kevin Bacon` and select *ok*.

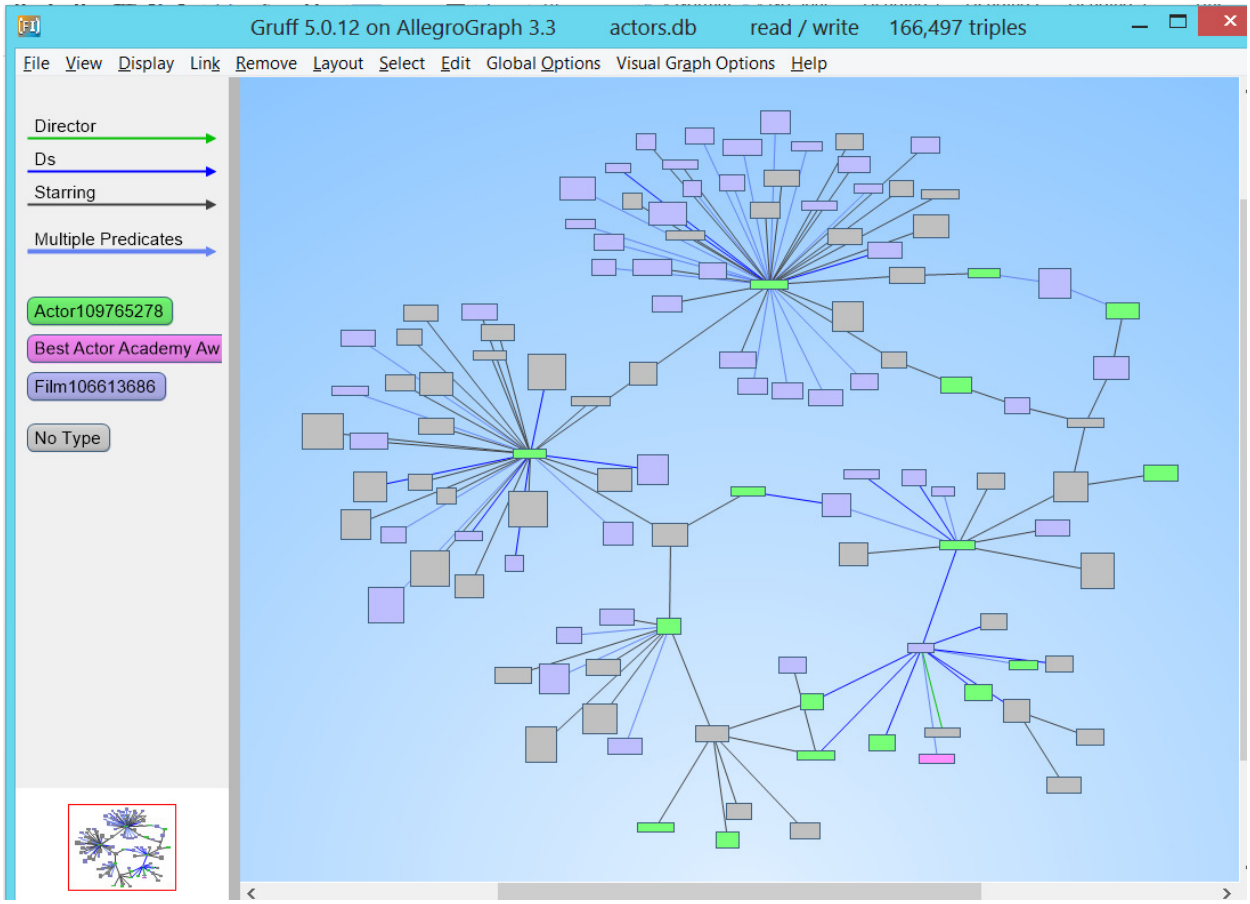
So now you have one node on the screen that we are going to use in the next section

Exploring the Graph in the Graph View:

So now we have Kevin Bacon on the screen and we want to see some triples where Kevin is the subject or object. The first thing we want to do is to select the predicates that we want to see on the screen. Type the letter 'p' and you'll see a list of predicates. Choose `DS`, `Director`, and `Starring` and click OK



Now select the Kevin Bacon node and press the letter 'f'. You'll see a lot of new nodes come up. Click on a movie and see how that expands by pressing the letter 'f'. Click a few times on nodes and you'll see that the screen gets crowded with nodes and links. Zoom in a little bit (use the wheel on your mouse or shift-.) and press the letter 'r' to reformat (In the Layout Menu you see all the types of reorganization of the screen provided by Gruff). Below is a screen shot that should look similar on your machine.



Note how you see on the left side the names of the predicates and classes used as well as the corresponding colors.

There are other ways to explore the graph on the screen. Press the letter 'z' a few times until only the Kevin Bacon node is on the screen (you might have to zoom back to see his name again).

For example: right click on Kevin and play with the first two options (Display a Linked Node from Menus, Display Linked Nodes from a Tree) to show triples on the screen.

Finding the Shortest Path Between Two Nodes:

One significant advantage to using a triple store is to let the database find connections between nodes.

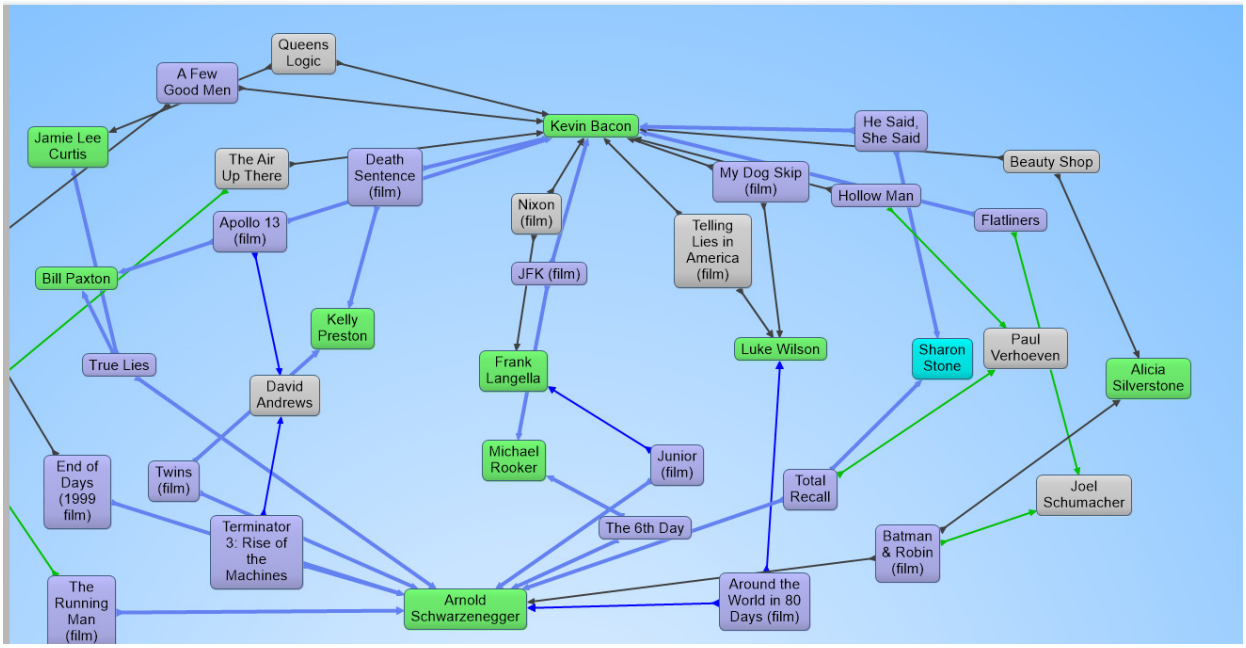
First let us clean up the screen by removing all triples from the screen (*Remove -> Remove All Triples*)

Then use ';' to find Kevin Bacon (or use *Display->Display Triples by Freetext Query*)

Do this again to find Arnold Schwarzenegger (just type `Arnold` and you'll find him).

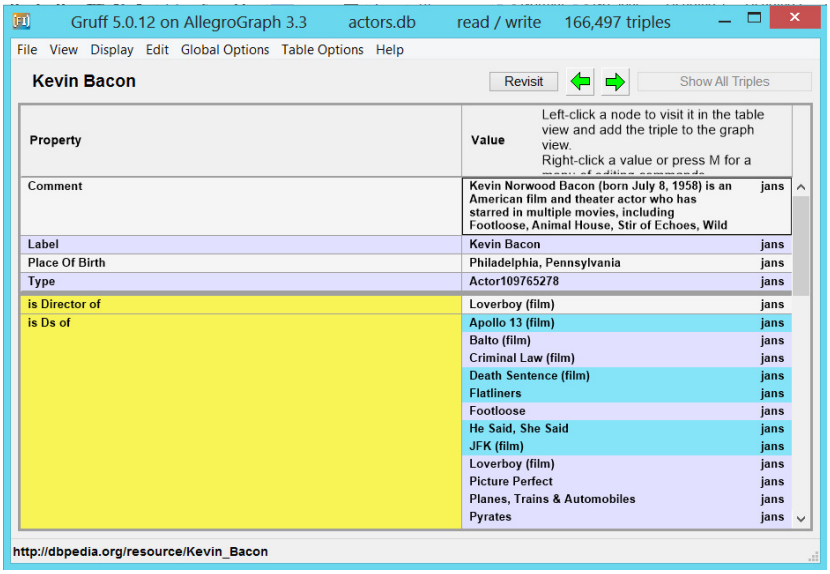
You should now have two nodes on the screen; `Arnold Schwarzenegger` and `Kevin Bacon`.

Now select Arnold, press ‘shift-f’, and drag the cursor to Kevin and click and you should see something like the picture below

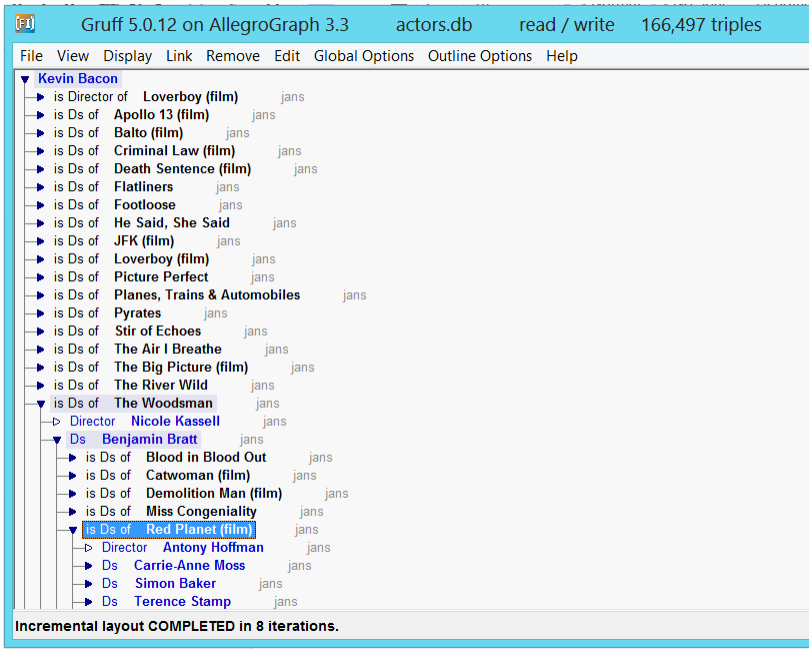


6. Exploring a Graph in the Tab View and Outline View

First let us discuss the Tabular View. Assuming that you see Kevin Bacon still on the screen, double click on his name (or press the letter ‘t’) and you are in the tabular view. See the picture below. It is kind of self evident on how to navigate through this view. Note that there is a thick grey line in the middle. Above the grey line you have triples that start with Kevin, below the grey line you have triples where Kevin is in the object position of the triple.



Another way to explore the triples is to use the outline view. Click on Kevin again and hit the letter 'O' and you'll see this. Note that black text means that you are going deeper into the hierarchy, the blue text means that you see triples that point back at Kevin. Just play with it and you'll soon understand intuitively.



Writing a SPARQL Query in Query View:

Gruff will help you write SPARQL Queries. If you know SPARQL to some extent then you can go to the query view by pressing the 'w' or *View->Query View*. Select the SPARQL bullet and then just for fun type this little query that will select a hundred random triples from the triple store.

```
Select * where { ?x ?y ?z . } limit 100
```

First click on the '`Do Query`' button and then click on the Create Visual Graph button.

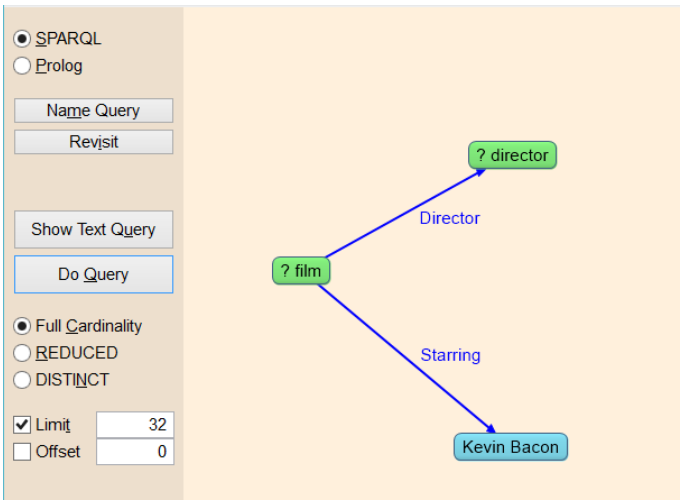
Writing a SPARQL query with the Graphical Query Builder:

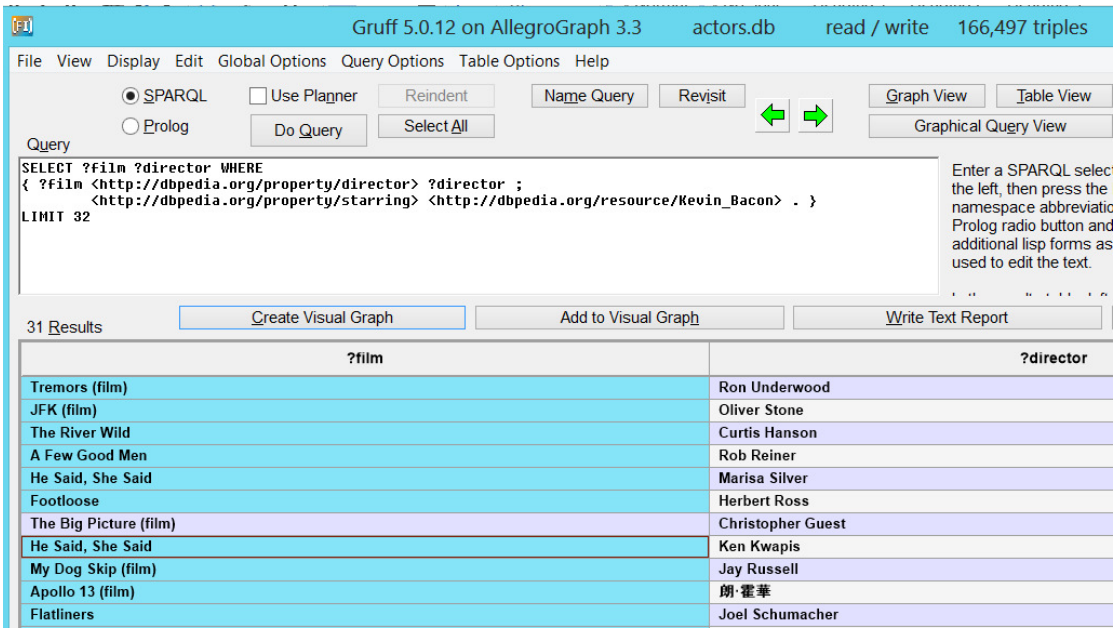
Now you don't need to type SPARQL Queries, you can also build them graphically. Here is a brief example.

The query that we are going to build: "Who directed the movies that Kevin Bacon starred in?"

1. Go to the Query Editor view (press 'e' or *View->Graphical Query View*)
2. Use *Display->Display Triples by Freetext Query* to find 'Kevin Bacon'
3. Right click on the screen and choose the first option: `create variable node`. Give it the name '`film`' (although you could any name you like, do not add the ? in front of the variable, Gruff will do that for you)
4. Right click on the variable '`? film`' and choose the first option: `Add Predicate Link`

5. Drag the cursor to Kevin Bacon and you get the option to choose the name of the link. Choose the option '`Predicates of Object Kevin Bacon`'. AllegroGraph already knows all the predicates pointing to Kevin Bacon so now you just have to choose. Please choose '`starring`'
6. Now right click on the screen again and create a new variable node, call it '`director`'
7. Right click on '`? film`' and '`Add Predicate Link`'. Drag the cursor to "`? director`". This time choose from '`All Predicates`' and choose '`director`'.
8. Now click on the `do-query` button and you'll see how a query gets created and executed.





Conclusion

We have shown you a simple way to create a triple store, navigate the triples, and create queries. Please note that we only used Gruff with the built in triple store. For larger data sets and working with SPARQL 1.1 you will want to try the combination of the AllegroGraph server and the client side Gruff.