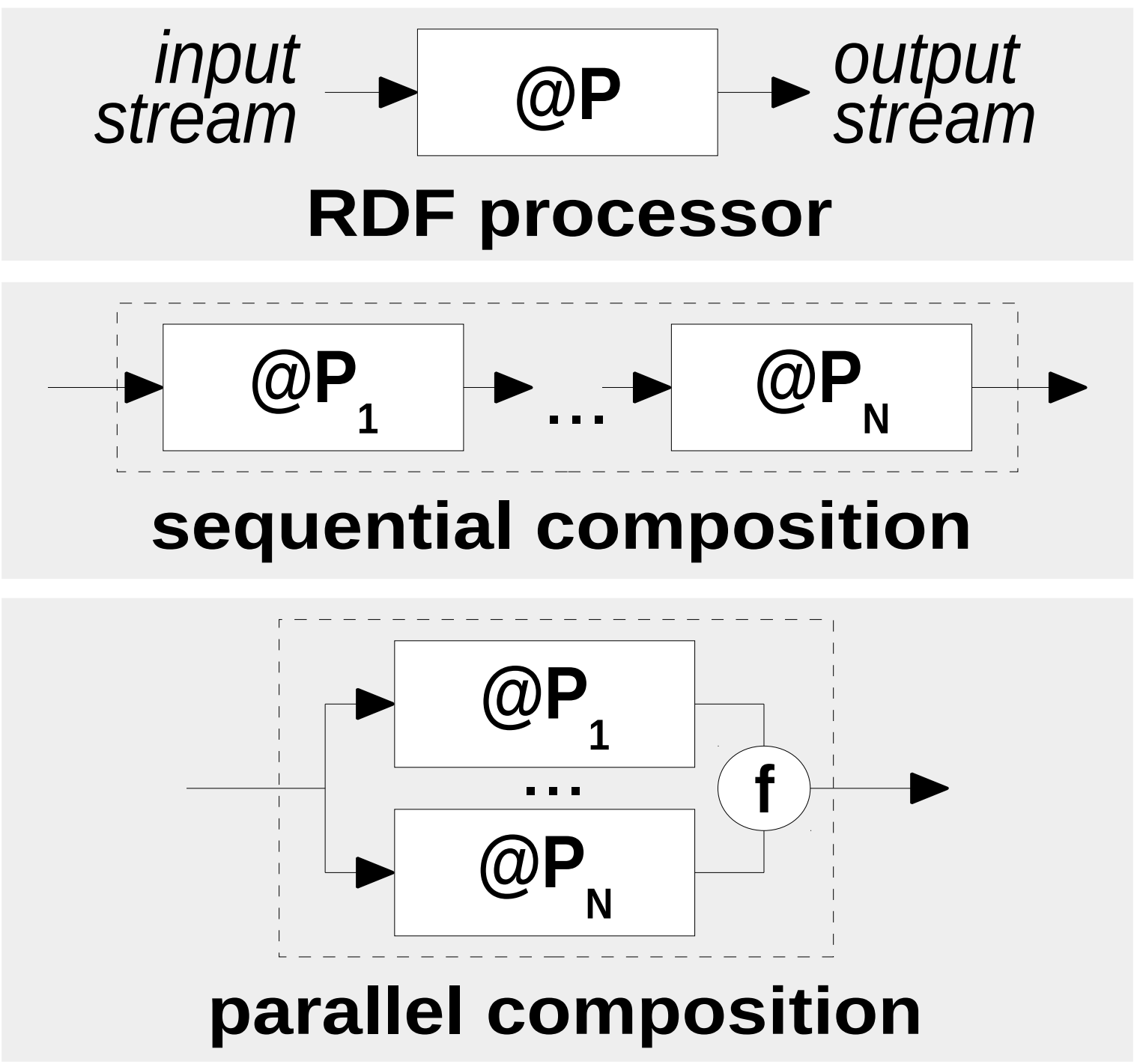
RDFpro

The Swiss-Army tool for RDF and Named Graph manipulation – http://rdfpro.fbk.eu/

Overview

RDFpro is an **extensible**, **general-purpose**, **open source** (public domain) Java tool for **processing large RDF datasets** on a **commodity machine** leveraging **streaming** and **sorting** techniques.



RDF processor

sequential composition

parallel composition

Addressed problem

1. tool support for RDF processing is fragmented and users often have to integrate heterogeneous tools even for simple workflows;
2. tools scaling to large datasets often require complex, distributed infrastructures such as Hadoop

RDFpro solution

- simple pipes & filter computation model supporting arbitrary sequential / parallel composition of RDF processors for different tasks
- out-of-the-box processors implementing common tasks
- possibility to plug-in new processors for custom task

- non-distributed computation based on streaming and sorting techniques for processing large datasets not fitting into memory
- multi-threaded implementation for vertical scalability

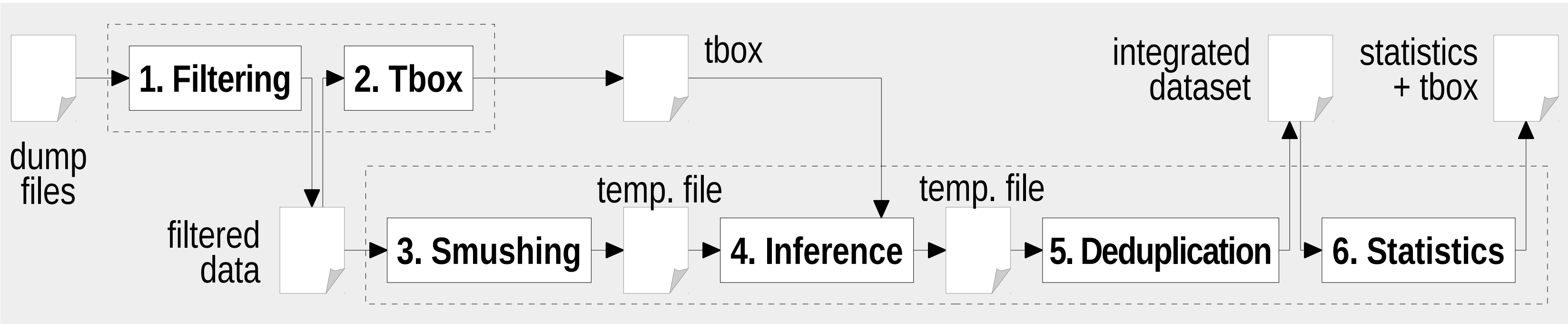
Supported RDF Processing Tasks

@read, @write	read/write data in multiple (compressed) formats
@download	download data from a SPARQL endpoint via queries
@upload	upload data to a SPARQL endpoint via INSERT DATA
@unique	RDF quad deduplication and set/multiset operations
@tbox. @stats	TBox and VOID statistics extraction
@transform	RDF quad filtering and replacement with JavaScript / Groovy scripting support
@mapreduce	MapReduce-like computation (multi-threaded, non-distributed), supporting map and reduce scripts
@smush	owl:sameAs smushing
@rdfs	RDFS inference with selectable rules
@rules	inference with SPARQL-like rules (OWL 2 RL support)

Concrete Use Case

Task: integrate relevant data from Freebase, GeoNames and DBpedia EN, ES, IT and NL, performing smushing, RDFS inference, provenance tracking with Named Graphs, and VOID statistics computation.

Pipeline of RDFpro processors:



Results (Intel Core I7 860 Linux pc, 16 GB ram, 500 GB 7200 rpm hd):

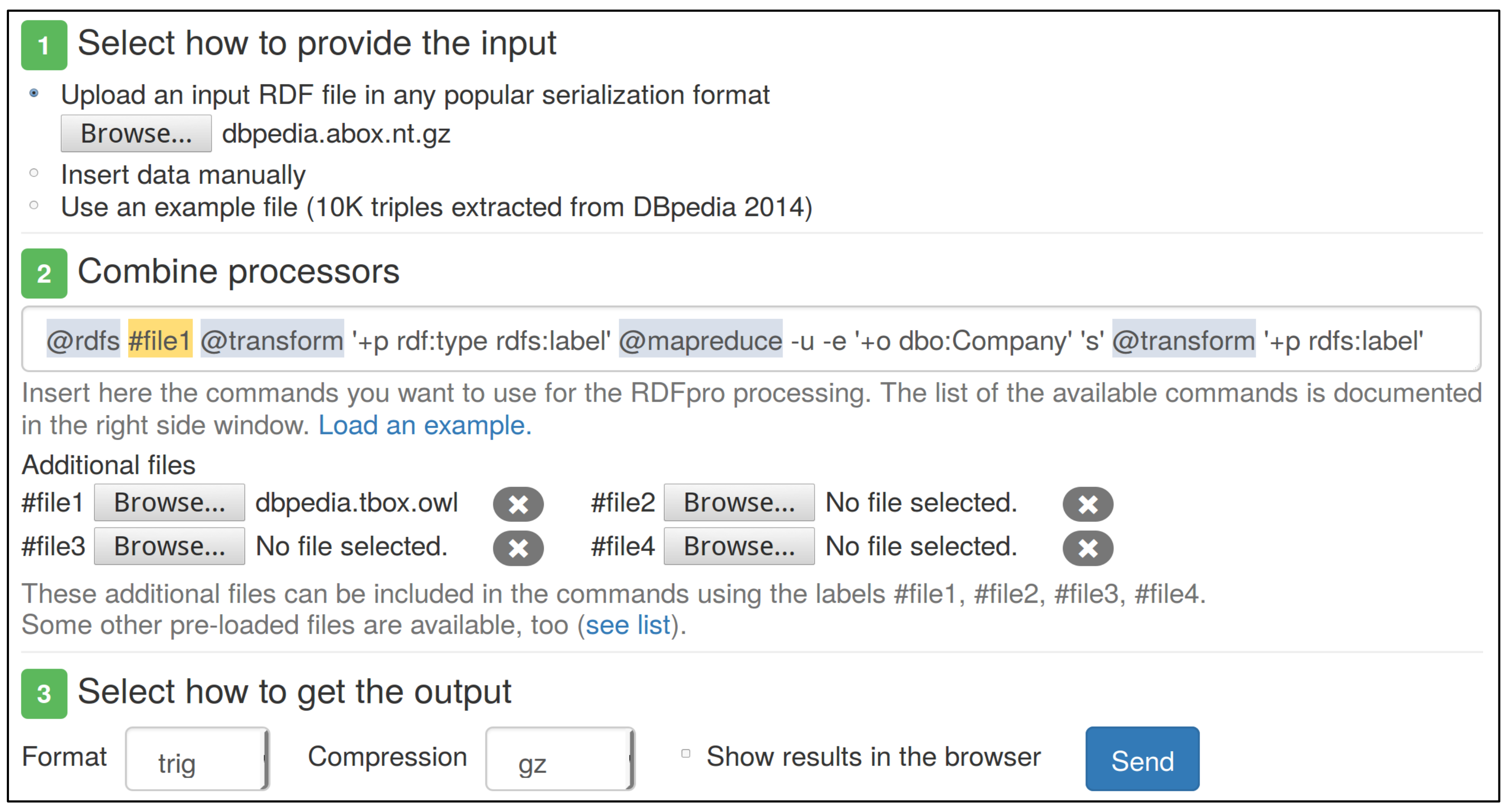
Processing step	Input size		Output size		Throughput		Time
	[Mquads]	[MiB]	[Mquads]	[MiB]	[Mquads/s]	[MiB/s]	[s]
Step 1 - Filtering	3175	31670	770	9871	0.76	7.55	4194
Step 2 - TBox extraction	770	9871	<1	~1	1.87	23.95	412
Step 3 - Smushing	770	9871	800	10538	0.34	4.36	2265
Step 4 - Inference	800	10539	1691	15884	0.21	2.79	3780
Step 5 - Deduplication	1691	15884	964	9071	0.40	3.73	4254
Step 6 - Statistics extract.	964	9072	<1	~3	0.37	3.50	2595
Steps 1-2 aggregated	3175	31670	770	9872	0.74	7.34	4315
Steps 3-6 aggregated	770	9872	964	9080	0.12	1.50	6590

Three Ways of Using RDFpro

① **Command line tool**, cross-platform (tested on Linux/Mac/Windows)

```
dkmuser@dkm-server-1:/data/rdfpro-example
$ rdfpro @read dbpedia.abox.nt.gz @rdfs dbpedia.tbox.owl @transform '+p rdf:type rdfs:label'\
>        @mapreduce -u -e '+o dbo:Company' 's' @transform '+p rdfs:label' @write Iabels.nt.gz
14:56:10(I) 27063 TBox triples read (165018 tr/s avg)
14:58:10(I) 72309189 triples read (601518 tr/s avg)
14:58:10(I) 125668722 records to sort (1045435 rec/s avg)
14:59:13(I) 63193206 records from sort (1363391 rec/s avg)
14:59:13(I) 13397105 reductions (289067 red/s avg)
14:59:13(I) 68646 triples written (1499 tr/s avg)
14:59:13(I) Done in 183 s
```

② **Web tool**, for PHP-enabled web servers (demo on RDFpro web site)



1 Select how to provide the input

● Upload an input RDF file in any popular serialization format

Browse... dbpedia.abox.nt.gz

○ Insert data manually

○ Use an example file (10K triples extracted from DBpedia 2014)

2 Combine processors

@rdfs #file1 @transform '+p rdf:type rdfs:label' @mapreduce -u -e '+o dbo:Company' 's' @transform '+p rdfs:label'

Insert here the commands you want to use for the RDFpro processing. The list of the available commands is documented in the right side window. Load an example.

Additional files

#file1	Browse...	dbpedia.tbox.owl	✖	#file2	Browse...	No file selected.	✖
#file3	Browse...	No file selected.	✖	#file4	Browse...	No file selected.	✖

These additional files can be included in the commands using the labels #file1, #file2, #file3, #file4. Some other pre-loaded files are available, too (see list).

3 Select how to get the output

Format: trig   Compression: gz   □ Show results in the browser   Send

③ **Java library**, available on Maven central

```
URI dboCompany = new URIImpl("http://dbpedia.org/ontology/Company");

RDFSource aboxSource = RDFSources.read(true, true, null, null, "dbpedia.tbox.owl");
RDFSource tboxSource = RDFSources.read(true, true, null, null, "dbpedia.abox.nt.gz");
RDFHandler labelsSink = RDFHandlers.write(null, 0, "labels.nt.gz");

RDFProcessor processor = RDFProcessors.sequence(
    RDFProcessors.rdfs(aboxSource, null, false, false),
    RDFProcessors.transform(Transformer.filter((Statement s) -> {
        URI p = s.getPredicate();
        return p.equals(RDF.TYPE) || p.equals(RDFS.LABEL); })),
    RDFProcessors.mapReduce(Mapper.select("s"), Reducer.filter(Reducer.IDENTITY,
        (Statement s) -> s.getObject().equals(dboCompany), null), true));

processor.apply(aboxSource, labelsSink, 1);
```

Powered by RDFpro

- RDF processing in **NewsReader** – http://www.newsreader-project.eu/
- **KnowledgeStore** storage framework – http://knowledgestore.fbk.eu/
- **PIKES** knowledge extraction suite – http://pikes.fbk.eu/

References:

- Corcoglioniti, F., Rospocher, M., Mostarda, M., Amadori, M. *Processing Billions of RDF Triples on a Single Machine using Streaming and Sorting.* In: ACM SAC 2015.
- Corcoglioniti, F., Rospocher, M., Amadori, M., Mostarda, M. *RDFpro: an Extensible Tool for Building Stream-Oriented RDF Processing Pipelines.* In: ISWC Developers Workshop, 2014.
- Corcoglioniti, F., Palmero Aprosio, A., Rospocher, M. *Demonstrating the Power of Streaming and Sorting for Non-distributed RDF Processing: RDFpro.* In: ISWC Posters & Demonstrations, 2015.
- Corcoglioniti, F., Rospocher, M., Cattoni, R., Magnini, B., Serafini, L. *The KnowledgeStore: a Storage Framework for Interlinking Unstructured and Structured Knowledge*. In: IJSWIS, volume 11, 2015.

Supported by:

NewsReader

SEVENTH FRAMEWORK PROGRAMME