Intégration et interrogation de données

Askomics

F. Legeai

INRA

27 septembre 2018

Intégration et interrogation de données

Askomics

F. Legeai

INRA

27 septembre 2018

Outline

1. A nightmare of data
   - Why it is difficult
   - Why RDF and SPARQL are relevant
2. RDF
   - RDF graph of resources
   - Serialization
3. the SPARQL language
4. Askomics
   - Integrating data
   - Querying data
5. Modeling
   - AskoR
6. Conclusion
7. Principles of federated queries

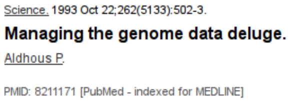
Big data and the deluge of life science data

Big data

Datasets so **large** or **complex** that traditional data processing is inadequate [Laney2001]

Life science: data deluge [Aldhous1993]

- computerized biomedical data
- Genomics and bioinformatics

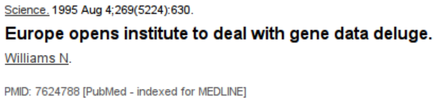


Science. 1993 Oct 22;262(5133):502-3.

Managing the genome data deluge.

Aldhous P.

PMID: 8211171 [PubMed - indexed for MEDLINE]



Science. 1995 Aug 4;269(5224):630.

Europe opens institute to deal with gene data deluge.

Williams N.

PMID: 7624788 [PubMed - indexed for MEDLINE]

Complexity of life science data: distributed

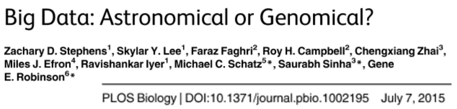
- 1500+ biological databases [Galperin2015]
- Lack of interoperability
- Some efforts of unified access (BioMart, InterMine...)

Annotation@EBI
MetaCyc
HumanCyc
KEGG Kyoto Encyclopedia of Genes and Genomes
WikiPathways Pathways for the People
INOH
prosite
OBO open biomedical ontologies
Pfam
swissprot
pir.georgetown.edu
PIR
RCSB PDB PROTEIN DATA BANK
ABCDB
RefSeq
ProDom
UniGene
ORGANIZED VIEW OF THE TRANSCRIPTOME
EMBL NUCLEOTIDE SEQUENCE DATABASE
Genbank
BIOBASE
TRANSFAC · PathoDB
TRANSPATH · S/MARt DB
TRANSCompel · Cytomer
UniProt the universal protein resource
SGD
Saccharomyces Genome Database
OMIM Online Mendelian Inheritance in Man
Johns Hopkins University
SIGENAE

What to expect for 2025?

Our estimation is that genomics is one of the most demanding domain in terms of

- data acquisition
- data storage
- data distribution
- data analysis



Big Data: Astronomical or Genomical?

Zachary D. Stephens[1], Skylar Y. Lee[1], Faraz Faghri[2], Roy H. Campbell[2], Chengxiang Zhai[3], Miles J. Efron[4], Ravishankar Iyer[1], Michael C. Schatz[5]*, Saurabh Sinha[3]*, Gene E. Robinson[6]*

PLOS Biology | DOI:10.1371/journal.pbio.1002195 July 7, 2015

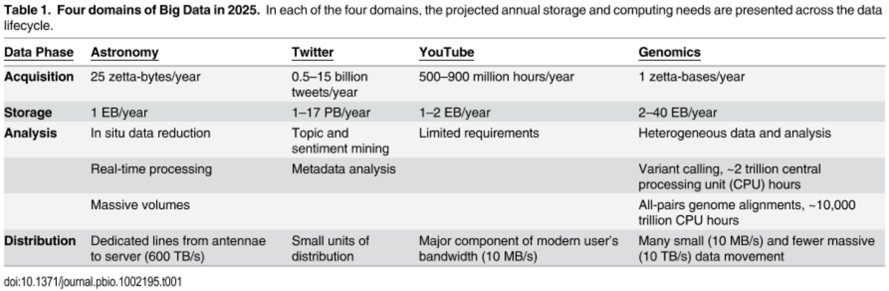
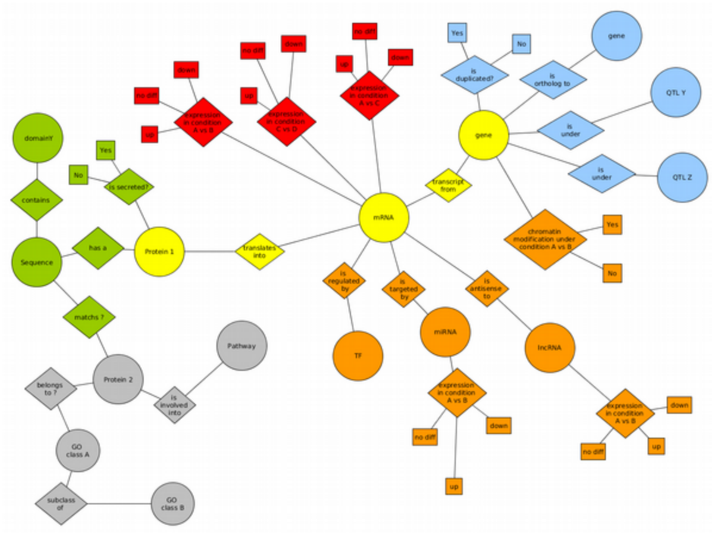


Table 1. Four domains of Big Data in 2025. In each of the four domains, the projected annual storage and computing needs are presented across the data lifecycle.

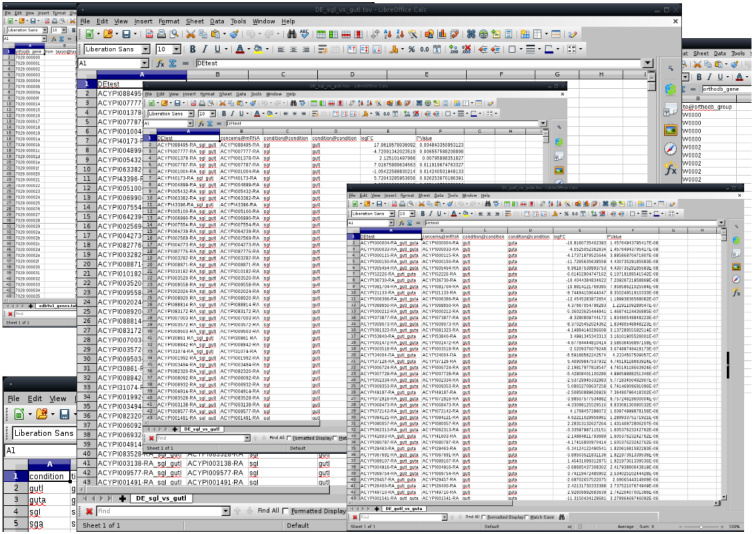
Data Phase	Astronomy	Twitter	YouTube	Genomics
Acquisition	25 zetta-bytes/year	0.5–15 billion tweets/year	500–900 million hours/year	1 zetta-bases/year
Storage	1 EB/year	1–17 PB/year	1–2 EB/year	2–40 EB/year
Analysis	In situ data reduction	Topic and sentiment mining	Limited requirements	Heterogeneous data and analysis
	Real-time processing	Metadata analysis		Variant calling, ~2 trillion central processing unit (CPU) hours
	Massive volumes			All-pairs genome alignments, ~10,000 trillion CPU hours
Distribution	Dedicated lines from antennae to server (600 TB/s)	Small units of distribution	Major component of modern user's bandwidth (10 MB/s)	Many small (10 MB/s) and fewer massive (10 TB/s) data movement

doi:10.1371/journal.pbio.1002195.t001

Data analysis needs data integration



Data everywhere!


DE_sgl_vs_gutl.tsv - LibreOffice Calc
File Edit View Insert Format Sheet Data Tools Window Help
Liberation Sans
DE_sgl_vs_gutl
DE_gutl_vs_guta
Find All
Formatted Display
Sheet 1 of 1
Default

	A
1	DEtest
2	ACYPI088495
3	ACYPI007777
4	ACYPI001378
5	ACYPI007787
6	ACYPI001004
7	ACYPI40173-R
8	ACYPI004899
9	ACYPI005432
10	ACYPI063382
11	ACYPI43396-R
12	ACYPI005100
13	ACYPI006990
14	ACYPI007554
15	ACYPI064239
16	ACYPI002569
17	ACYPI004273
18	ACYPI082776
19	ACYPI003282
20	ACYPI008871
21	ACYPI010182
22	ACYPI003520
23	ACYPI009558
24	ACYPI002024
25	ACYPI008920
26	ACYPI088814
27	ACYPI083172
28	ACYPI007003
29	ACYPI003572
30	ACYPI009503
31	ACYPI30861-R
32	ACYPI008842
33	ACYPI31074-R
34	ACYPI001992
35	ACYPI003494
36	ACYPI082320
37	ACYPI006092
38	ACYPI006932
39	ACYPI004914
40	ACYPI083528-RA_sgl_gutl
41	ACYPI003138-RA_sgl_gutl
42	ACYPI009577-RA_sgl_gutl
43	ACYPI001491-RA_sgl_gutl

	A
1	condition
2	gutl
3	guta
4	sgl
5	sga

Data are distributed

Definition: Entity

Anything that can be identified (i.e. the things we can talk about)

- some informations about an entity in a repository
- other informations about the same entity in another repository

your question requires to combine entity descriptions from multiple datasets

Only possible if:

- Entities are identified
- Datasets use the same identifier to describe the same entity

Good luck with your spreadsheets! :-)

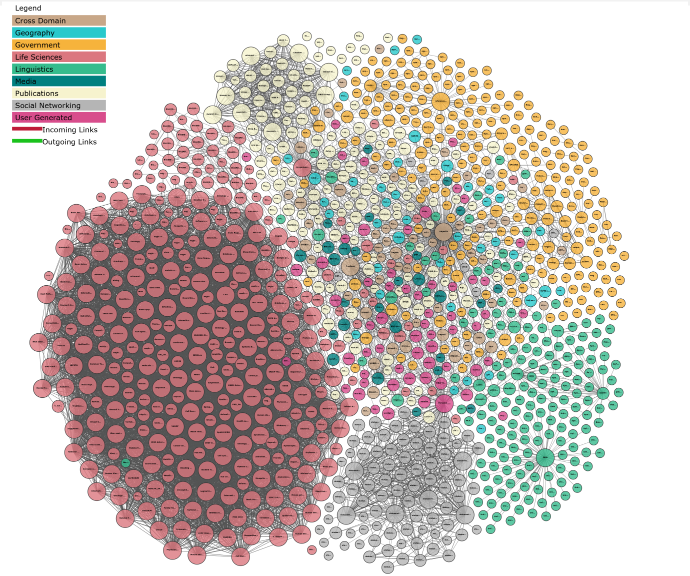
Data description involves hierarchies

- Knowledge has been formalized (e.g. in ontologies)
- Query engines (more or less) gracefully handle
  - linking data and ontologies
  - reasoning on the ontologies

The good news (1/3)

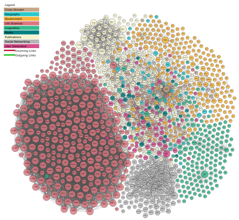
Semantic Web technologies (RDF, SPARQL, OWL) address all of these

Linked open data (in 2017)



Linked open data cloud, by M. Schmachtenberg, C. Bizer, A. Jentzsch and R. Cyganiak http://lod-cloud.net/

Linked Open Data



The not so good news: Linked data are here... but still have to be adopted by end users

“Real” users

- do not contribute (yet) their data to the LOD cloud
- do not use the LOD cloud for analyzing their own data (yet)

“Real” users need to retain their ability to interact with **their** data

The RDF format

- Identify things
  - Describe them
  - their characteristics
  - their relations to other things
  - their categories
- Possibly combine descriptions from multiple places
- Elements of the descriptions should themselves be identiable
- Support rich querying and reasoning on descriptions

"Now! *That* should clear up a few things around here!"

Resources (aka entities)

Resource (= entity)

Resource: anything in the universe

Two kinds of resource

- Referent: resource with an identity
  - identified by an **URI or IRI)**
  - there can be 0, 1 or more IRI identifying the same resource
- Litteral: resource defined by its value
  - represented as a string (e.g. `"foo"` or `'foo'`)
  - each litteral has 1 **datatype** that defines its nature and the range of possible values
    - default=string
    - string, integer, float, date, dateTime, boolean...
    - use XSD

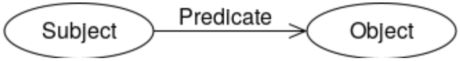
Litteral

- Represented by a string (delimited by single or double-quotes; escape if necessary)
- Its **datatype** (default = string) explains how to interpret
  - `"12"` = the string with two characters (`'1'` and `'2'`)
  - `"12"^^xsd:integer` = the string representing the number twelve
  - `"12.0"^^xsd:float` = the string representing the number twelve
  - `"true"` = the string with four characters
  - `"true"^^xsd:boolean` = the string representing the boolean true
  - `"1"^^xsd:boolean` = the string representing the boolean true
  - `"2017-08-24T18:54:42"^^xsd:dateTime`
- String litterals can be qualified by a **language tag**
  - `"ADN"@fr` or `"DNA"@en`

RDF triple

RDF triple = <subject, predicate, object>

- **subject**: the resource being described
- **predicate**: the relation (from the subject to the object)
- **object**: the value of the predicate for the subject
  (one of them if the predicate can have several values for the subject; in this case use as many triples as necessary)



RDF triple

- an IRI can be the subject of many triples
- an IRI can the the object of many triples
- an IRI can be the subject of some triples and the object of other triples
- a litteral can only be an object

RDF graph

RDF graph = set of RDF triples

- Set
  - No order among the triples
  - Redundant triples are ignored
- Structure = labeled directed graph
  - nodes = resources (subject or object of as least one triple)
  - edges = triples
    - start = subject IRI
    - end = object IRI
    - edge label = property IRI

The graph itself can be seen as a resource (**named graph**)

- identified by its IRI
- useful for describing who generated the graph, when, how, from which data...
- not mandatory (default graph)

RDF in practice

RDF can be stored:

- in files (several formats: n3, turtle, xml-rdf,...)
- in triplestores ($\sim$ DB): Virtuoso, fuseki, sesame & RDF4J, Oracle, 4store,...
- within HTML: RDFa

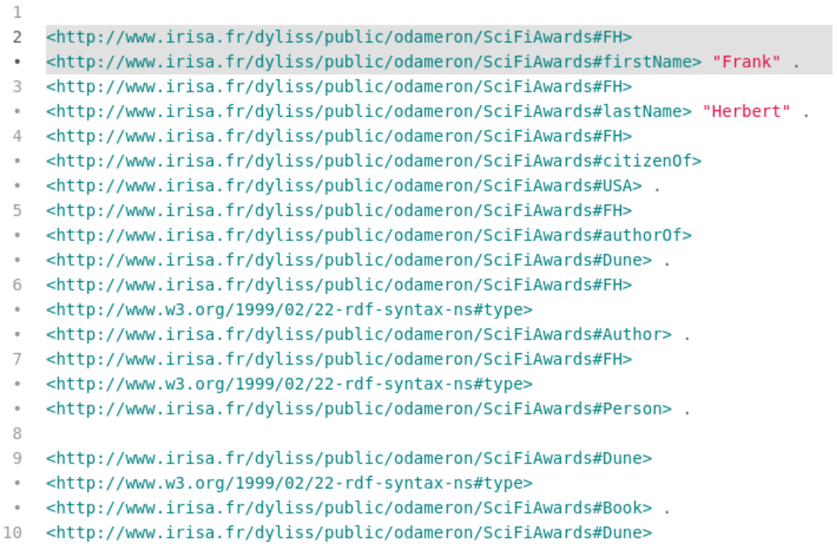
RDF is supported by most languages (and most triplestores have bindings)

RDF format: N-triples

- 1 triple / line
  - URI delimited by angle brackets ("<" and ">")
  - subject, predicate and object separated by space
  - line ends with a colon (".")
- no order between triples
- order within triple matters (obviously)

Straightforward, but not easy to read/write (for humans)

RDF format: N-triples



```

<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#firstName> "Frank" .
<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#lastName> "Herbert" .
<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#citizenOf> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#USA> .
<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#authorOf> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Dune> .
<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Author> .
<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Person> .

<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Dune> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Book> .
<http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Dune>
```

RDF format: Turtle

- N-triples syntax still possible
- comments start with “#” until end of line
- prefix declaration for CURIE
  - `@PREFIX prefixName: <URI template> .`
  - (also a triple)
  - usually at begining of file
- 1 triple / line
  - URIs can be full URIs (<...>) or CURIE (no delimiters)
  - a triple can mix both
  - line ends with “;” ⇒ next line has
    - same subject (no need to repeat)
  - line ends with “,” ⇒ next line has
    - same subject (no need to repeat)
    - same property (no need to repeat)

RDF format: Turtle

```
@prefix sf: <http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .

#
# FRANK HERBERT
#
sf:FH sf:firstName "Frank" .
sf:FH sf:lastName "Herbert" .
sf:FH sf:citizenOf sf:USA .
sf:FH sf:authorOf sf:Dune .
sf:FH rdf:type sf:Author .
sf:FH rdf:type sf:Person .

sf:Dune rdf:type sf:Book .
sf:Dune rdf:type sf:SciFiBook .
sf:Dune sf:title "Dune" .
sf:Dune sf:wonAward sf:Nebula1965 .
sf:Dune sf:wonAward sf:Hugo1966 .

sf:Nebula1965 rdf:type sf:NebulaAward .
sf:Hugo1966 rdf:type sf:HugoAward .
```

Other formats

N3

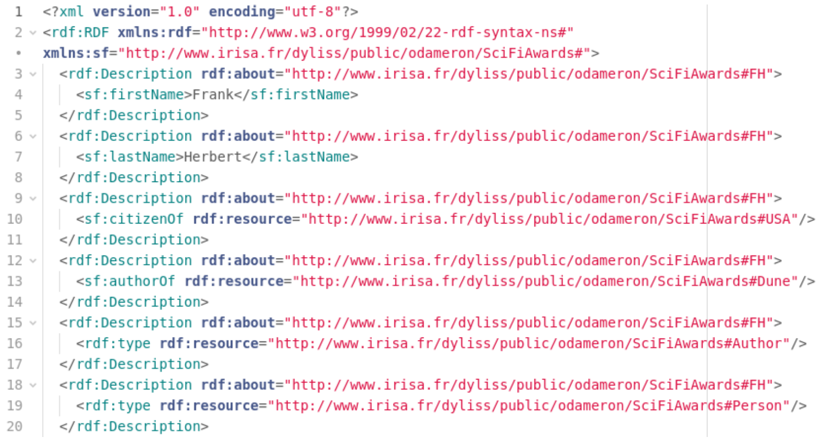
Do not confuse with N-Triples!

- N3 = turtle + additional features
  - Named graphs
  - Inference rules
  - ...
- not a W3C recommendation

RDF-XML

- XML serialization of RDF
- a major pain to read or edit these by hand

RDF format: RDF-XML



```xml
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:sf="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#">
  <rdf:Description rdf:about="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH">
    <sf:firstName>Frank</sf:firstName>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH">
    <sf:lastName>Herbert</sf:lastName>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH">
    <sf:citizenOf rdf:resource="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#USA"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH">
    <sf:authorOf rdf:resource="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Dune"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH">
    <rdf:type rdf:resource="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Author"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#FH">
    <rdf:type rdf:resource="http://www.irisa.fr/dyliss/public/odameron/SciFiAwards#Person"/>
  </rdf:Description>
```

SPARQL Endpoints

- SPARQL queries are executed against RDF datasets, consisting of RDF graphs.
- A SPARQL endpoint accepts queries and returns results via HTTP.
- Endpoints can be generic (i.e.web accessible datasets) or specific to a dataset
- The results of SPARQL queries can be returned and/or rendered in a variety of formats (XML, JSON, RDF, HTML)

SPARQL queries

SPARQL = query language similar to SQL
Variable names start with a question mark

```
 @prefix foaf: <http://xmlns.com/foaf/0.1/> .
 @prefix mus: <http://www.irisa.fr/dyliss/public/mus#> .

SELECT ?player ?instr
 WHERE {
   ?player foaf:name Alice
   ?player mus:plays ?instr .
 }
```

SPARQL queries (2)

```
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix mus: <http://www.irisa.fr/dyliss/public/mus#> .

SELECT ?player ?instr
 WHERE {
   ?player foaf:name Alice
   ?player mus:plays ?instr
   ?instr mus:instrclass corde .
 }
```

SPARQL queries (3)

```
SELECT ?player ?instr
 WHERE {
   ?player foaf:name Alice
   ?player mus:plays ?instr
   ?instr mus:instrclass corde
   ?player info:age ?age
   FILTER (?age <= 24) .
    }
```

SPARQL queries (4)

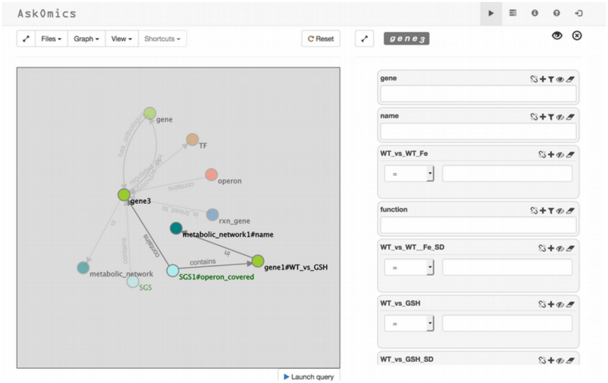
+ many other features

- OPTIONAL
- UNION
- ORDER BY
- DISTINCT
- ...

AskOmics: reconciling domain experts with Semantic Web

AskOmics is usefull for:

- Integrating data
- Querying data



AskOmics (1) integrating data with RDF

Integrating data

- Import your data files
  - CSV or TSV
  - RDF
  - GFF
- Import public knowledge bases (GO, Reactome, NCBI taxon,...)
- Declare (remote) SPARQL endpoints (in progress)

The good news (2/3)

Don't worry about RDF, AskOmics generates it from your csv

AskOmics (2) generating user-friendly SPARQL queries

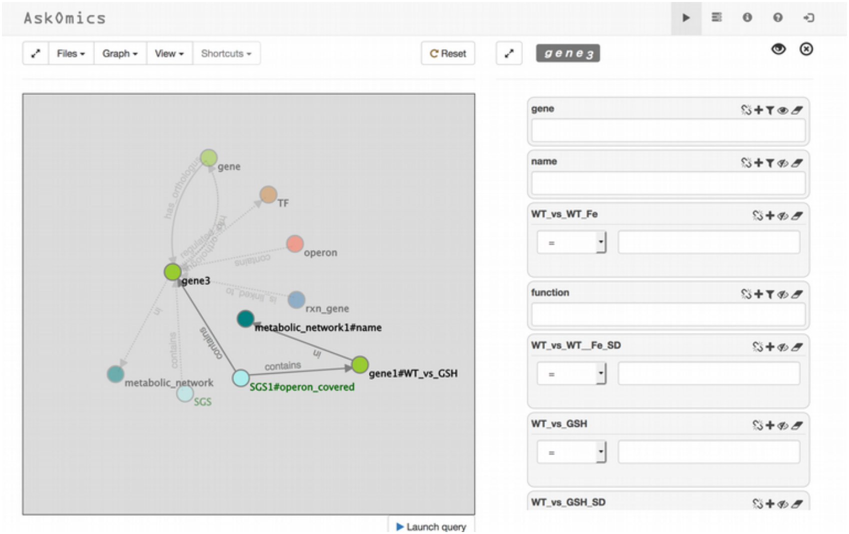
Querying data

- Graph-based user-friendly SPARQL query composer
  - based on an abstraction of your data
    - depends on the data structure (small)
    - not on the data themselves (possibly huge)
  - can be enriched (shortcuts and virtual links)
  - modular design (select/deselect datasets)
- Span multiple SPARQL endpoints (in progress)
- You do not have to see the SPARQL code
- Save the query result (obviously) in RDF or TSV

The good news (3/3)

Don't worry about SPARQL queries, AskOmics generates them for you (and runs them as well)

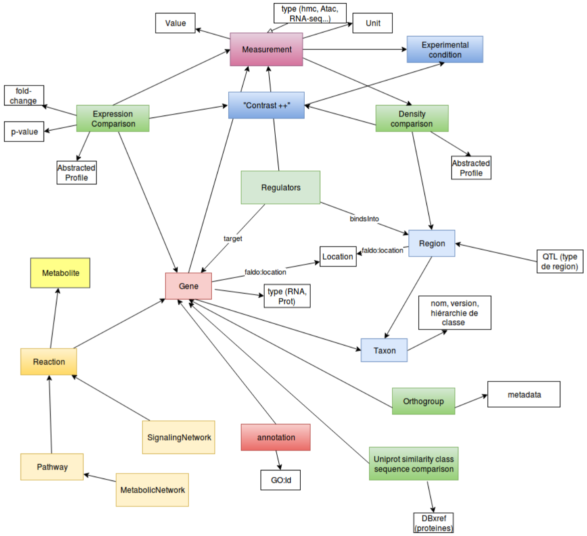
Querying data



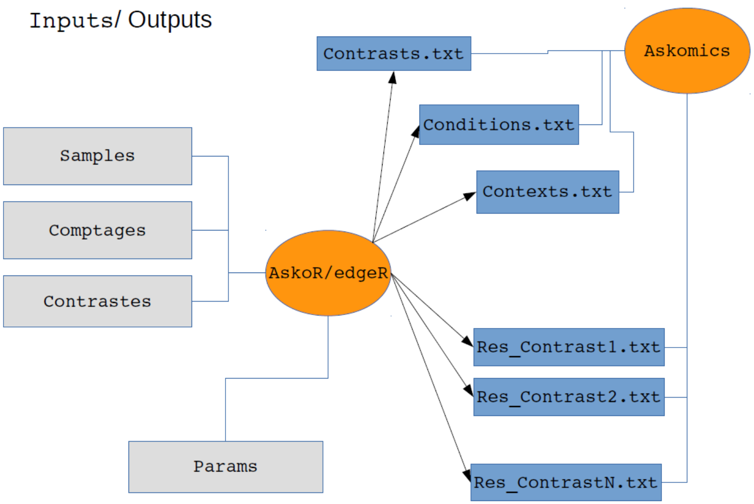
AskOmics: new features in latest release

- Improved distribution
  - docker-compose
  - Genostack (thank you, GenOuest)
- Collaborative mode with public/private graphs management (authentification)
- Internal Modules
  - Gene Ontology
  - BioPax-v3 (Reactome,...)
- External modules integration
  - Metabolic Pathways (generated by Aureme)
  - Expression Analysis (generated by Askor)

Managing your own data

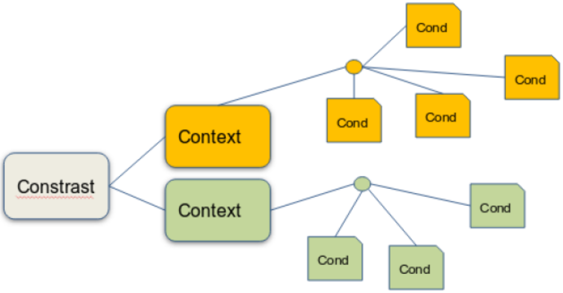


AskoR - The scheme



AskoR - 3 cons

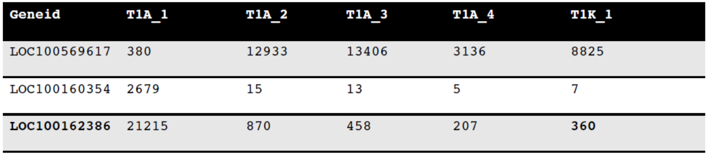
Contrastes, contextes et conditions



AskoR - Inputs

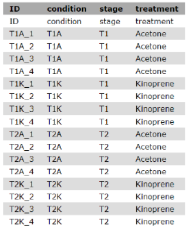
Inputs / Outputs

- Comptages



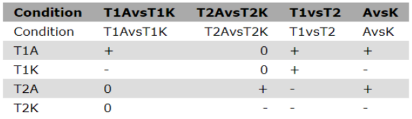
Geneid	T1A_1	T1A_2	T1A_3	T1A_4	T1K_1
LOC100569617	380	12933	13406	3136	8825
LOC100160354	2679	15	13	5	7
LOC100162386	21215	870	458	207	**360**

- Fichier de description des échantillons



ID	condition	stage	treatment
ID	condition	stage	treatment
T1A_1	T1A	T1	Acetone
T1A_2	T1A	T1	Acetone
T1A_3	T1A	T1	Acetone
T1A_4	T1A	T1	Acetone
T1K_1	T1K	T1	Kinoprene
T1K_2	T1K	T1	Kinoprene
T1K_3	T1K	T1	Kinoprene
T1K_4	T1K	T1	Kinoprene
T2A_1	T2A	T2	Acetone
T2A_2	T2A	T2	Acetone
T2A_3	T2A	T2	Acetone
T2A_4	T2A	T2	Acetone
T2K_1	T2K	T2	Kinoprene
T2K_2	T2K	T2	Kinoprene
T2K_3	T2K	T2	Kinoprene
T2K_4	T2K	T2	Kinoprene

- Contrastes

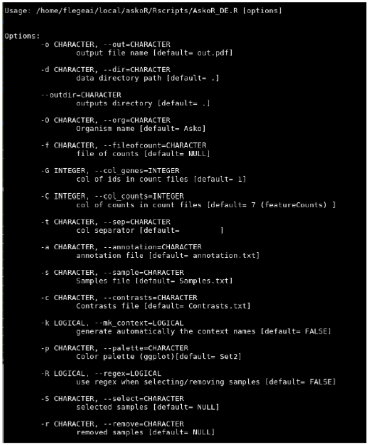


Condition	T1AvsT1K	T2AvsT2K	T1vsT2	AvsK
Condition	T1AvsT1K	T2AvsT2K	T1vsT2	AvsK
T1A	+	0	+	+
T1K	-	0	+	-
T2A	0	+	-	+
T2K	0	-	-	-

AskoR - Commands

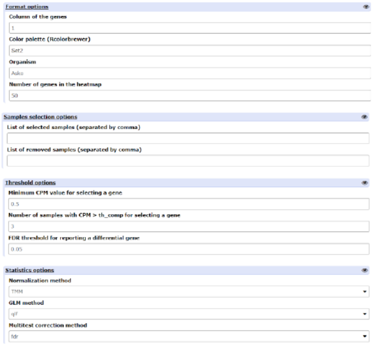
AskoR

Ligne de commande



```
Usage: /home/flegeai/local/askoR/Rscripts/AskoR_DE.R [options]

Options:
        -o CHARACTER, --out=CHARACTER
                output file name [default= out.pdf]

        -d CHARACTER, --dir=CHARACTER
                data directory path [default= .]

        --outdir=CHARACTER
                outputs directory [default= .]

        -O CHARACTER, --org=CHARACTER
                Organism name [default= Asko]

        -f CHARACTER, --fileofcount=CHARACTER
                file of counts [default= NULL]

        -G INTEGER, --col_genes=INTEGER
                col of ids in count files [default= 1]

        -C INTEGER, --col_counts=INTEGER
                col of counts in count files [default= 7 (featureCounts) ]

        -t CHARACTER, --sep=CHARACTER
                col separator [default=         ]

        -a CHARACTER, --annotation=CHARACTER
                annotation file [default= annotation.txt]

        -s CHARACTER, --sample=CHARACTER
                Samples file [default= Samples.txt]

        -c CHARACTER, --contrasts=CHARACTER
                Contrasts file [default= Contrasts.txt]

        -k LOGICAL, --mk_context=LOGICAL
                generate automatically the context names [default= FALSE]

        -p CHARACTER, --palette=CHARACTER
                Color palette (ggplot)[default= Set2]

        -R LOGICAL, --regex=LOGICAL
                use regex when selecting/removing samples [default= FALSE]

        -S CHARACTER, --select=CHARACTER
                selected samples [default= NULL]

        -r CHARACTER, --remove=CHARACTER
                removed samples [default= NULL]
```

Galaxy



Format options

Column of the genes	1
Color palette (Rcolorbrewer)	Set2
Organism	Asko
Number of genes in the heatmap	50

Samples selection options

List of selected samples (separated by comma)	
List of removed samples (separated by comma)	

Threshold options

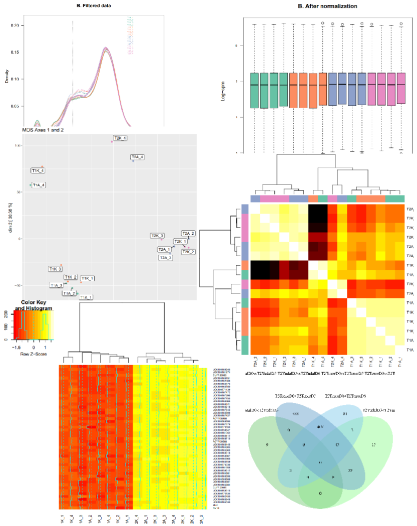
Minimum CPM value for selecting a gene	0.5
Number of samples with CPM > th_comp for selecting a gene	3
FDR threshold for reporting a differential gene	0.05

Statistics options

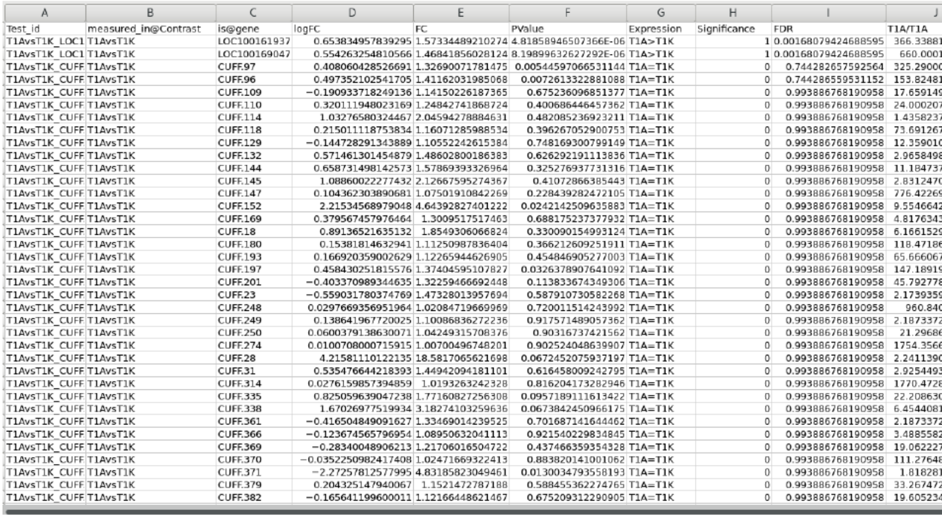
Normalization method	TMM
GLM method	qlf
Multitest correction method	fdr

https://github.com/askomics/askoR

AskoR - Graphics

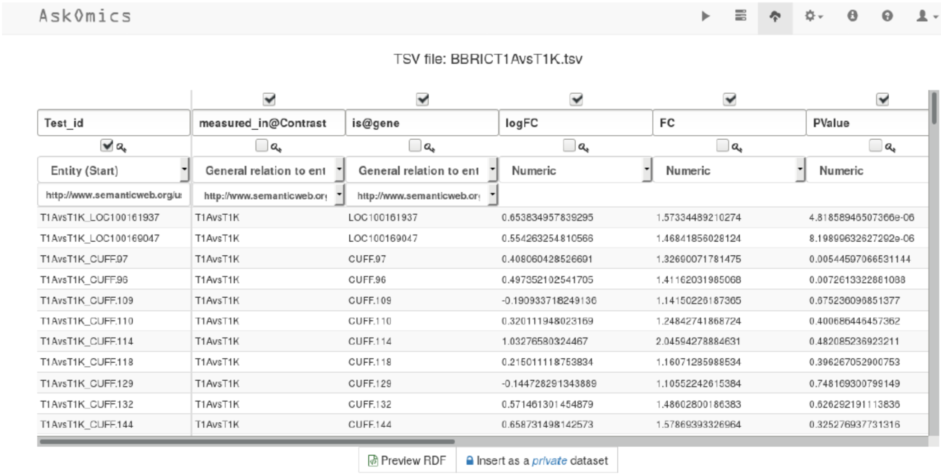


AskoR - Results



A	B	C	D	E	F	G	H	I	J
Test_id	measured_in@Contrast	is@gene	logFC	FC	PValue	Expression	Significance	FDR	T1A/T1A
T1AvsT1K_LOC1	T1AvsT1K	LOC100161937	0.653834957839295	1.57334489210274	4.81858946507366E-06	T1A>T1K	1	0.00168079424688595	366.33881
T1AvsT1K_LOC1	T1AvsT1K	LOC100169047	0.554263254810566	1.46841856028124	8.19899632627292E-06	T1A>T1K	1	0.00168079424688595	660.0001
T1AvsT1K_CUFF	T1AvsT1K	CUFF.97	0.408060428526691	1.32690071781475	0.00544597066531144	T1A=T1K	0	0.744282657592564	325.29000
T1AvsT1K_CUFF	T1AvsT1K	CUFF.96	0.497352102541705	1.41162031985068	0.00726133228810888	T1A=T1K	0	0.744286559531152	153.82481
T1AvsT1K_CUFF	T1AvsT1K	CUFF.109	−0.190933718249136	1.14150226187365	0.675236096851377	T1A=T1K	0	0.993886768190958	17.659149
T1AvsT1K_CUFF	T1AvsT1K	CUFF.110	0.320111948023169	1.24842741868724	0.400686446457362	T1A=T1K	0	0.993886768190958	24.000207
T1AvsT1K_CUFF	T1AvsT1K	CUFF.114	1.03276580324467	2.04594278884631	0.482085236923211	T1A=T1K	0	0.993886768190958	1.4358237
T1AvsT1K_CUFF	T1AvsT1K	CUFF.118	0.215011118753834	1.16071285988534	0.396267052900753	T1A=T1K	0	0.993886768190958	73.691267
T1AvsT1K_CUFF	T1AvsT1K	CUFF.129	−0.144728291343889	1.10552242615384	0.748169300799149	T1A=T1K	0	0.993886768190958	12.359010
T1AvsT1K_CUFF	T1AvsT1K	CUFF.132	0.571461301454879	1.48602800186383	0.626292191113836	T1A=T1K	0	0.993886768190958	2.9658498
T1AvsT1K_CUFF	T1AvsT1K	CUFF.144	0.658731498142573	1.57869393326964	0.325276937731316	T1A=T1K	0	0.993886768190958	11.184737
T1AvsT1K_CUFF	T1AvsT1K	CUFF.145	1.08860022277432	2.12667595274367	0.41072866385443	T1A=T1K	0	0.993886768190958	2.8312470
T1AvsT1K_CUFF	T1AvsT1K	CUFF.147	0.104362303890681	1.07501910842269	0.228439282472105	T1A=T1K	0	0.993886768190958	776.42269
T1AvsT1K_CUFF	T1AvsT1K	CUFF.152	2.21534568979048	4.64392827401222	0.0242142509635883	T1A=T1K	0	0.993886768190958	9.5546642
T1AvsT1K_CUFF	T1AvsT1K	CUFF.169	0.379567457976464	1.3009517517463	0.688175237377932	T1A=T1K	0	0.993886768190958	4.8176343
T1AvsT1K_CUFF	T1AvsT1K	CUFF.18	0.89136521635132	1.8549306066824	0.330090154993124	T1A=T1K	0	0.993886768190958	6.1661529
T1AvsT1K_CUFF	T1AvsT1K	CUFF.180	0.15381814632941	1.11250987836404	0.366212609251911	T1A=T1K	0	0.993886768190958	118.47186
T1AvsT1K_CUFF	T1AvsT1K	CUFF.193	0.166920359002629	1.12265944626905	0.454846905277003	T1A=T1K	0	0.993886768190958	65.666067
T1AvsT1K_CUFF	T1AvsT1K	CUFF.197	0.458430251815576	1.37404595107827	0.0326378907641092	T1A=T1K	0	0.993886768190958	147.18919
T1AvsT1K_CUFF	T1AvsT1K	CUFF.201	−0.403370989344635	1.32259466692448	0.113833674349306	T1A=T1K	0	0.993886768190958	45.792778
T1AvsT1K_CUFF	T1AvsT1K	CUFF.23	−0.559031780374769	1.47328013957694	0.587910730582268	T1A=T1K	0	0.993886768190958	2.1739359
T1AvsT1K_CUFF	T1AvsT1K	CUFF.248	0.0297669356951964	1.02084719669969	0.720011514243992	T1A=T1K	0	0.993886768190958	960.840
T1AvsT1K_CUFF	T1AvsT1K	CUFF.249	0.138641967720025	1.10086836272236	0.917571489057362	T1A=T1K	0	0.993886768190958	2.1873372
T1AvsT1K_CUFF	T1AvsT1K	CUFF.250	0.0600379138630071	1.04249315708376	0.90316737421562	T1A=T1K	0	0.993886768190958	21.29686
T1AvsT1K_CUFF	T1AvsT1K	CUFF.274	0.0100708000715915	1.00700496748201	0.902524048639907	T1A=T1K	0	0.993886768190958	1754.3566
T1AvsT1K_CUFF	T1AvsT1K	CUFF.28	4.21581110122135	18.5817065621698	0.0672452075937197	T1A=T1K	0	0.993886768190958	2.2411390
T1AvsT1K_CUFF	T1AvsT1K	CUFF.31	0.535476644218393	1.44942094181101	0.616458009242795	T1A=T1K	0	0.993886768190958	2.9254493
T1AvsT1K_CUFF	T1AvsT1K	CUFF.314	0.0276159857394859	1.0193263242328	0.816204173282946	T1A=T1K	0	0.993886768190958	1770.4728
T1AvsT1K_CUFF	T1AvsT1K	CUFF.335	0.825059639047238	1.77160827256308	0.0957189111613422	T1A=T1K	0	0.993886768190958	22.208630
T1AvsT1K_CUFF	T1AvsT1K	CUFF.338	1.67026977519934	3.18274103259636	0.0673842450966175	T1A=T1K	0	0.993886768190958	6.4544081
T1AvsT1K_CUFF	T1AvsT1K	CUFF.361	−0.416504849091627	1.33469014239525	0.701687141644462	T1A=T1K	0	0.993886768190958	2.1873372
T1AvsT1K_CUFF	T1AvsT1K	CUFF.366	−0.123674565796954	1.08950632041113	0.921540229834845	T1A=T1K	0	0.993886768190958	3.4885582
T1AvsT1K_CUFF	T1AvsT1K	CUFF.369	−0.28340048906213	1.21706016504722	0.437466359354328	T1A=T1K	0	0.993886768190958	19.062227
T1AvsT1K_CUFF	T1AvsT1K	CUFF.370	−0.0352250982417408	1.02471669322413	0.883820141001062	T1A=T1K	0	0.993886768190958	111.27648
T1AvsT1K_CUFF	T1AvsT1K	CUFF.371	−2.27257812577995	4.83185823049461	0.0130034793558193	T1A=T1K	0	0.993886768190958	1.818281
T1AvsT1K_CUFF	T1AvsT1K	CUFF.379	0.204325147940067	1.1521472787188	0.588455362274765	T1A=T1K	0	0.993886768190958	33.267472
T1AvsT1K_CUFF	T1AvsT1K	CUFF.382	−0.165641199600011	1.12166448621467	0.675209312290905	T1A=T1K	0	0.993886768190958	19.605234

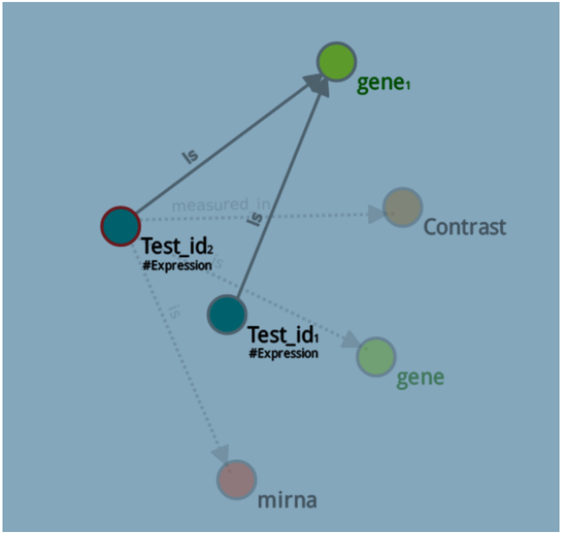
AskoR - Integration in Askomics (1)



TSV file: BBRICT1AvsT1K.tsv

Test_id	measured_in@Contrast	is@gene	logFC	FC	PValue
	✓	✓	✓	✓	✓
✓	☐	☐	☐	☐	☐
Entity (Start)	General relation to ent	General relation to ent	Numeric	Numeric	Numeric
http://www.semanticweb.org/u	http://www.semanticweb.or	http://www.semanticweb.or			
T1AvsT1K_LOC100161937	T1AvsT1K	LOC100161937	0.653834957839295	1.57334489210274	4.81858946507366e-06
T1AvsT1K_LOC100169047	T1AvsT1K	LOC100169047	0.554263254810566	1.46841856028124	8.19899632627292e-06
T1AvsT1K_CUFF.97	T1AvsT1K	CUFF.97	0.408060428526691	1.32690071781475	0.00544597066531144
T1AvsT1K_CUFF.96	T1AvsT1K	CUFF.96	0.497352102541705	1.41162031985068	0.0072613322881088
T1AvsT1K_CUFF.109	T1AvsT1K	CUFF.109	-0.190933718249136	1.14150226187365	0.675236096851377
T1AvsT1K_CUFF.110	T1AvsT1K	CUFF.110	0.320111948023169	1.24842741868724	0.400686446457362
T1AvsT1K_CUFF.114	T1AvsT1K	CUFF.114	1.03276580324467	2.04594278884631	0.482085236923211
T1AvsT1K_CUFF.118	T1AvsT1K	CUFF.118	0.215011118753834	1.16071285988534	0.396267052900753
T1AvsT1K_CUFF.129	T1AvsT1K	CUFF.129	-0.144728291343889	1.10552242615384	0.748169300799149
T1AvsT1K_CUFF.132	T1AvsT1K	CUFF.132	0.571461301454879	1.48602800186383	0.626292191113836
T1AvsT1K_CUFF.144	T1AvsT1K	CUFF.144	0.658731498142573	1.57869393326964	0.325276937731316

AskoR - Integration in Askomics (2)



Askomics now

AskOmics now

- Within Dyliss: data integration
- Dyliss ecosystem:
    - INRA
    - INSERM
    - Sanofi
    - received interest from: Fealinx + Theranexus
- Growing adoption by a wider user base
    - Docker image
    - GenoStack (support from GenOuest)
    - interface with Galaxy (developped by INRA)
    - AskoR (developped by INRA)

The future of AskOmics

The future of AskOmics

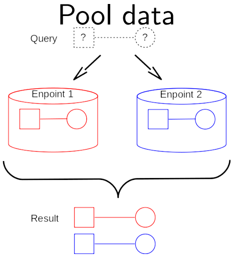
- extend query expressivity
- improve the GUI ergonomy
- improve support for multiple endpoints
  - constellation of AskOmics
  - benefit from the FederatedQueryScaler PRE (with Wimmics)
- reach a larger user base

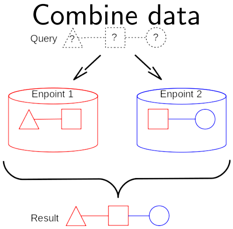
AskOmics received contributions from:

- Meziane Aite
- Arnaud Belcour (intern)
- Charles Bettembourg (postdoc + Sanofi)
- Anthony Bretaudeau (INRA)
- Yvanne Chaussin (intern)
- Olivier Dameron (Univ. Rennes 1)
- Aurélie Evrard (postdoc INRA)
- Olivier Filangi (INRA)
- Xavier Garnier (intern + INRA)
- Maël Kerbiriou (intern)
- Fabrice Legeai (INRA)
- Sylvaine Bitteur (INRA / Agrocampus Ouest) for the logo
- Colleagues from Nantes for their insight on federated queries
    - Pascal Molli
    - Patricia Serrano Alvarado
    - Hala Skaf

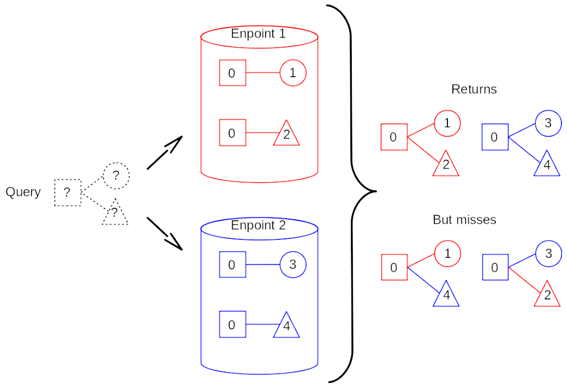
Federated queries principle

- Linked data
  - RDF repositories can be queried in SPARQL via endpoints
  - data from one endpoint can make references to data from another endpoint
- Federated queries span several endpoints
  - the SPARQL engine is responsible for propagating the query and merging the results
  - good news: supported by SPARQL language + query engines
  - not so good news: performances :-(



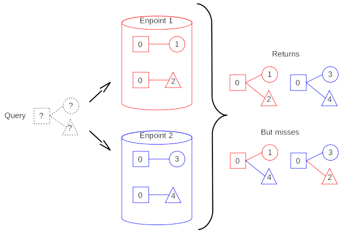


Federated queries difficulty: endpoints not independent



Treating the endpoints independently fails when combining data

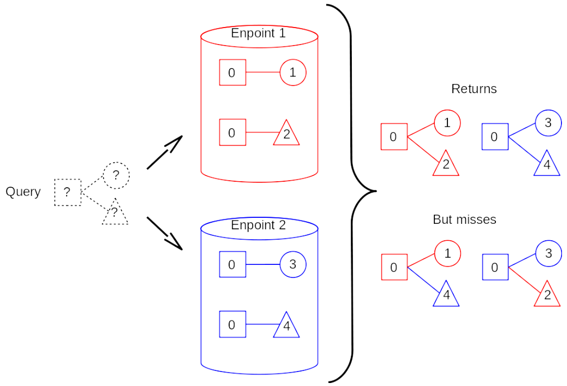
Federated queries difficulty: endpoints can not be merged



Merging the endpoints is not a viable solution either

- each endpoint is potentially big
- merging
    - increases network traffic
    - increases storage consumption
    - decreases query answering performances
    - does not scale up to LOD

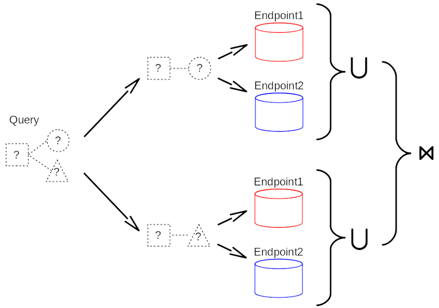
Federated queries difficulty: queries must be decomposed



The query must be decomposed into fragments

so that the endpoints are not treated as if they were disjoint

Federated queries: q. fragmentation increases complexity



Sending each fragment to each endpoint results in

- many subqueries for each endpoint (distant server overload)
- many unions and joins (local engine overload)
- potential transfer of large quantities of data before performing the joins, even if it ultimately few results (network overload)

Processing federated queries: general approach

Decompose the query into fragments

The fewer fragments the better: reduces joins

For each fragment, select the relevant endpoints

The fewer endpoints the better (but no false negatives!): reduces joins

Determine the order for processing the fragments (q. planning)

Start by the most selectives, maybe parallelize, and potentially rewrite the subqueries

These three aspects can be inter-dependent

Processing federated queries: existing solutions

- General low level SPARQL engines: Virtuoso, Fuseki,...
  - well adapted for serving 1 endpoint
  - naive approach for federated queries
- Dedicated high level engines (FedX, HiBISCuS,...)
  - front end to low-level engines
  - hot domain (in general)
  - still not up to Life Science queries