
SANSA: Hands-on, Demonstrations & Applications



This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 809965.



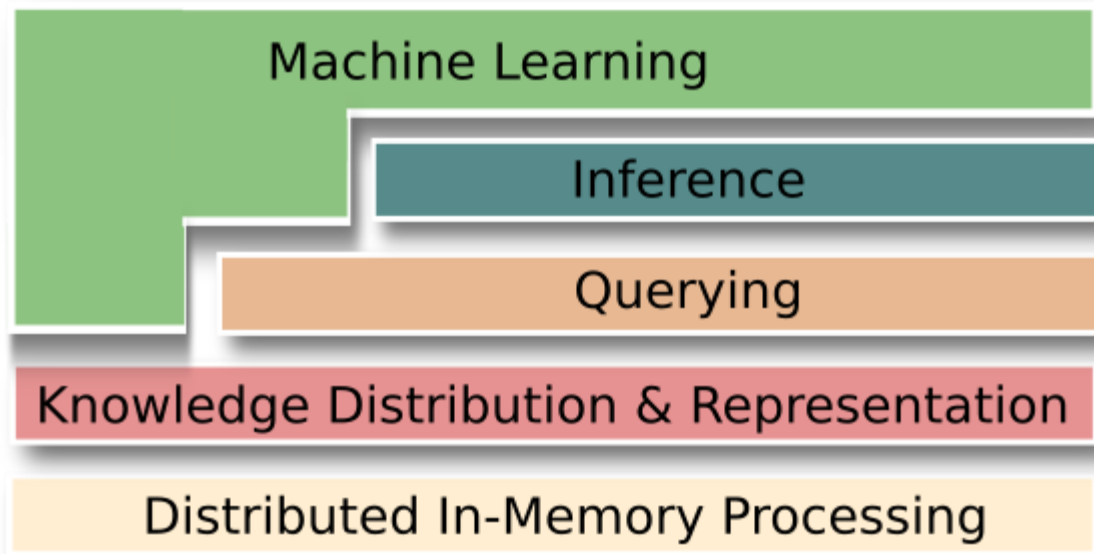
Having a local single-node SANSA

- ❖ From a the SANSA-Notebooks github repository
<<https://github.com/sansa-stack/sansa-notebooks>>
- ❖ Requirements:
 - docker
 - docker-compose
- ❖ Only **5** commands:
 - `git clone https://github.com/sansa-stack/sansa-notebooks`
 - `cd sansa-notebooks/`
 - `make`
 - `make up`
 - `make load-data`
 - Goto: <http://localhost/>

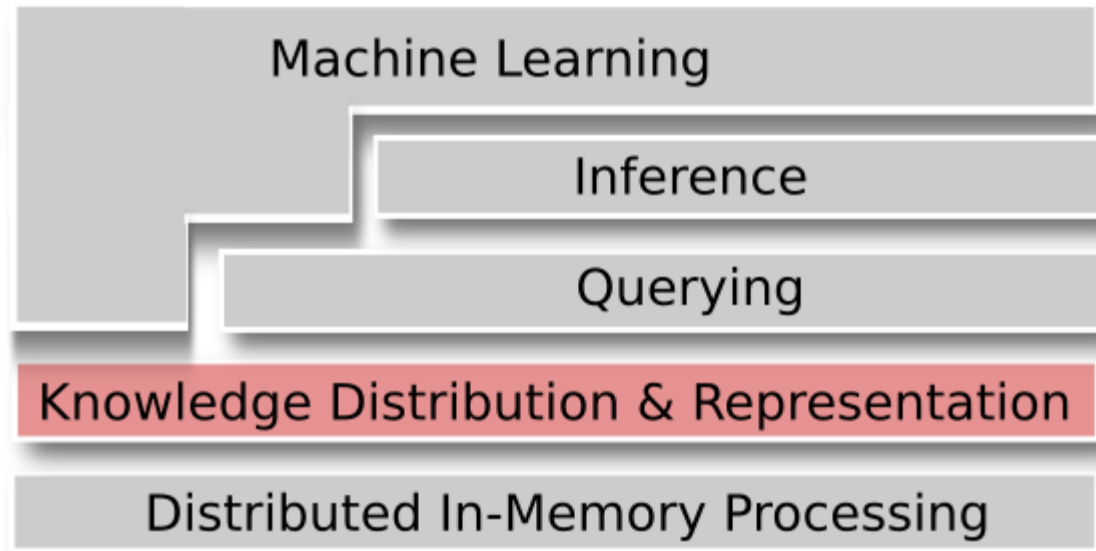


Hands-on Layer by Layer

SANSA Layers



SANSA: Read Write Layer



SANSA: Read Write Layer

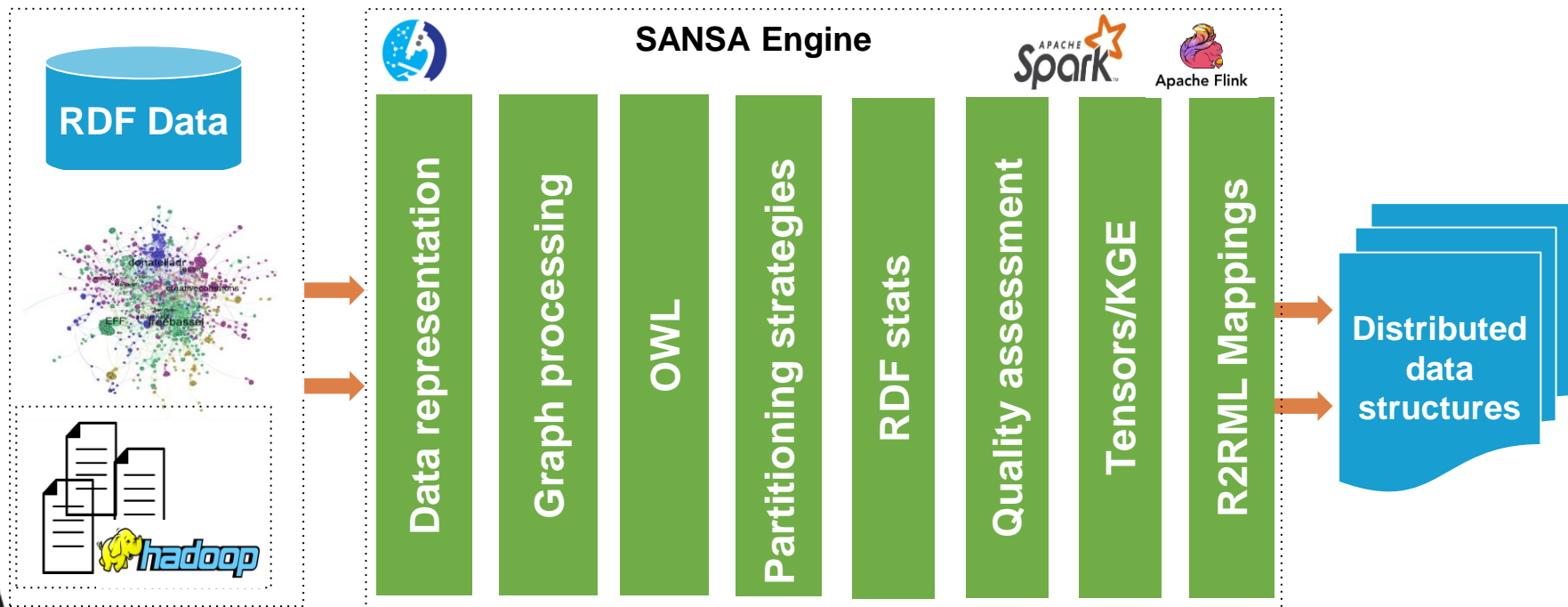
- ❖ Ingest RDF and OWL data in different formats using Jena / OWL API style interfaces
- ❖ Represent data in multiple formats
 - (e.g. RDD, Data Frames, GraphX, Tensors)
- ❖ Allow transformation among these formats
- ❖ Compute dataset statistics and apply functions to URIs, literals, subjects, objects → Distributed LODStats



```
val triples = spark.rdf(Lang.NTRIPLES)(input)
triples.find(None,
Some(NodeFactory.createURI("http://dbpedia.org/ontology/influenced")), None)
val rdf_stats_prop_dist = triples.statsPropertyUsage()
```



SANSA: Read Write Layer features



SANSA: OWL Support

- ❖ Distributed processing of OWL axioms
- ❖ Support for Manchester OWL & functional syntax
- ❖ Derived distributed data structures:
 - E.g. matrix representation of subclass-of axioms to compute its closure via matrix operations

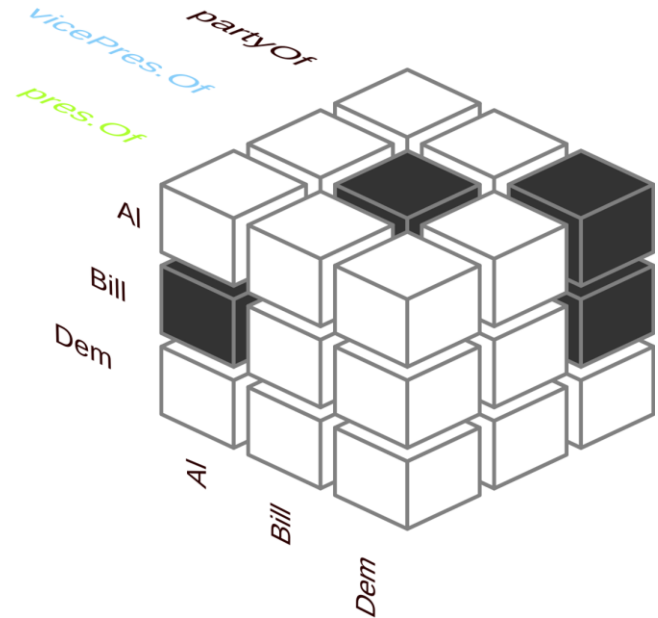
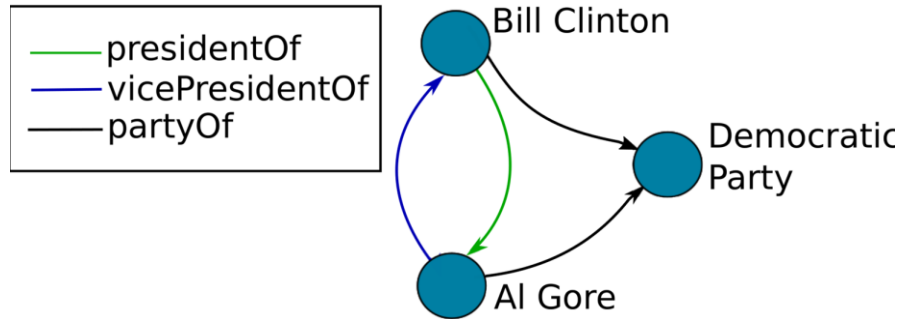


```
val rdd = spark.owl(Syntax.MANCHESTER)("file.owl")  
// get all subclass-of axioms  
val sco = rdd.filter(_.isInstanceOf[OWLSubClassOfAxiom])
```

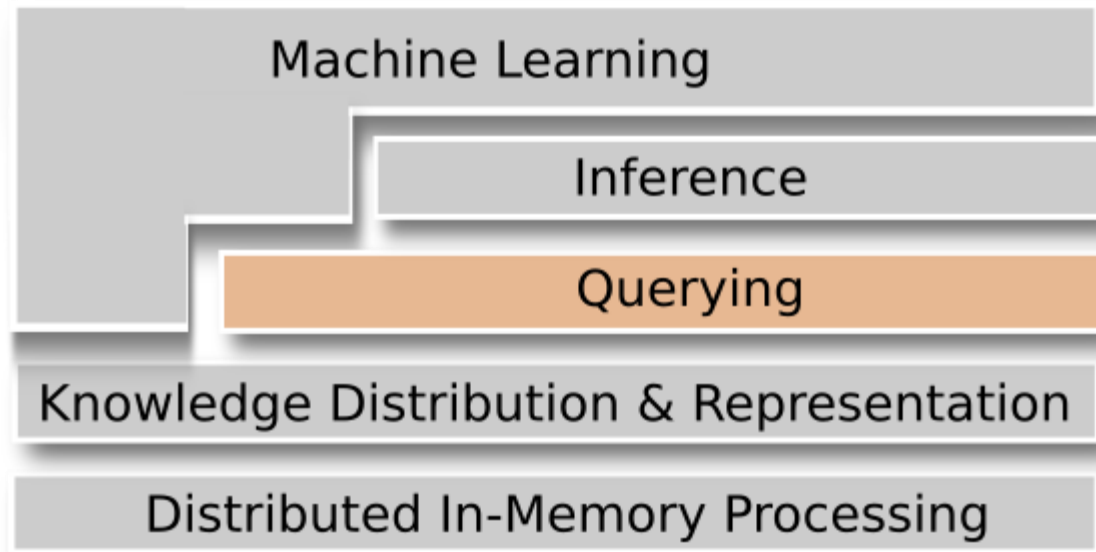


RDF to Tensors (experimental)

- T_{ijk} is 1 if triple (i-th entity, k-th predicate, j-th entity) exists and 0 otherwise



SANSA: Query Layer



SANSA: Query Layer

- ❖ To make generic queries efficient and fast using:
 - Intelligent indexing
 - Splitting strategies
 - Distributed Storage
- ❖ SPARQL query engine evaluation
 - (SPARQL-to-SQL approaches, Virtual Views)
- ❖ Provision of W3C SPARQL compliant endpoint



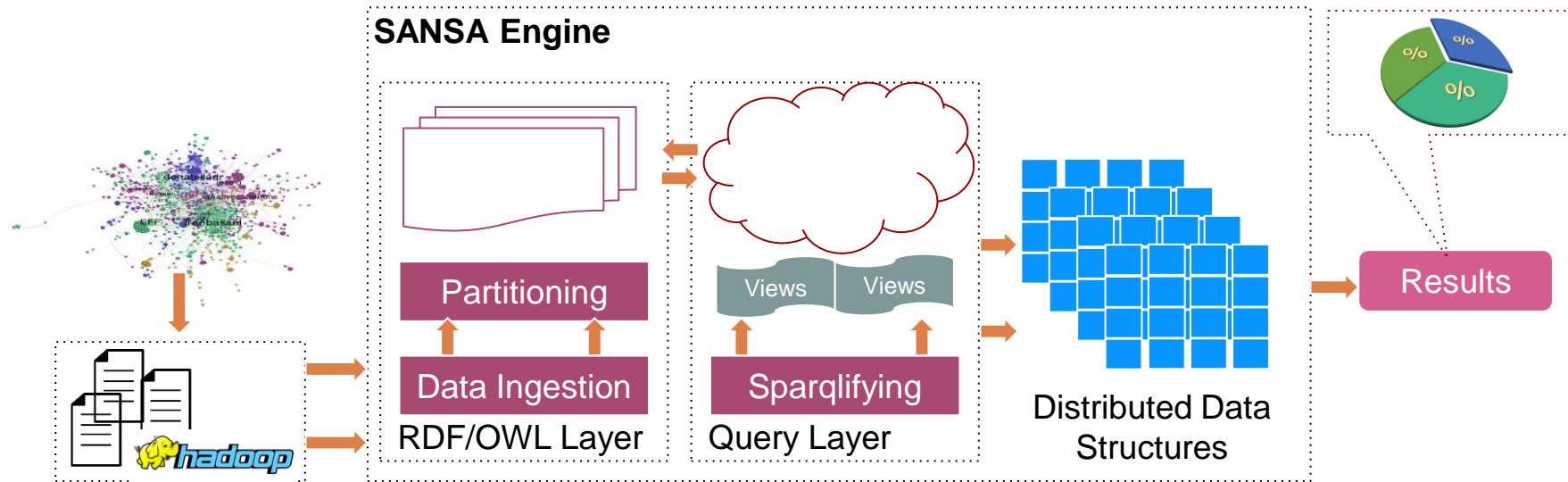
```
val triples = spark.rdf(Lang.NTRIPLES)(input)
```

```
val sparqlQuery = "SELECT * WHERE {?s ?p ?o} LIMIT 10"
```

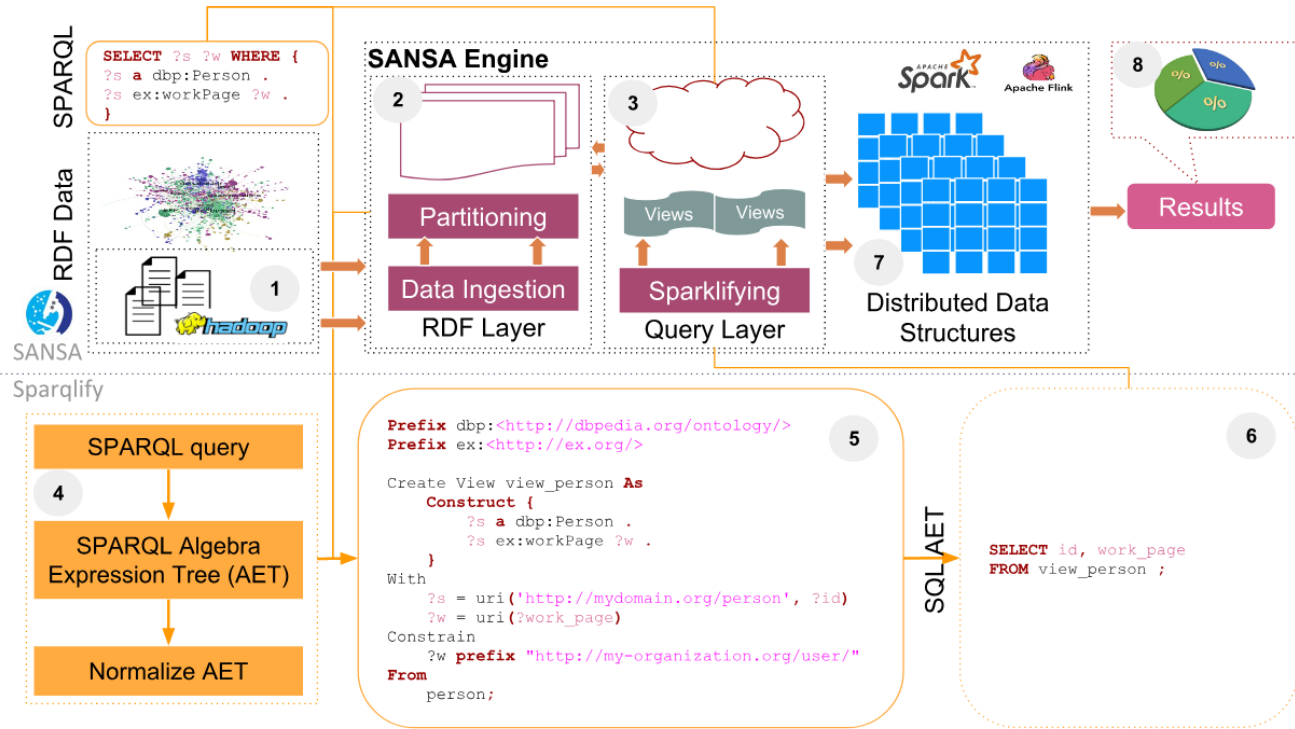
```
val result = triples.sparql(sparqlQuery)
```



Querying via SPARQL & Partitioning



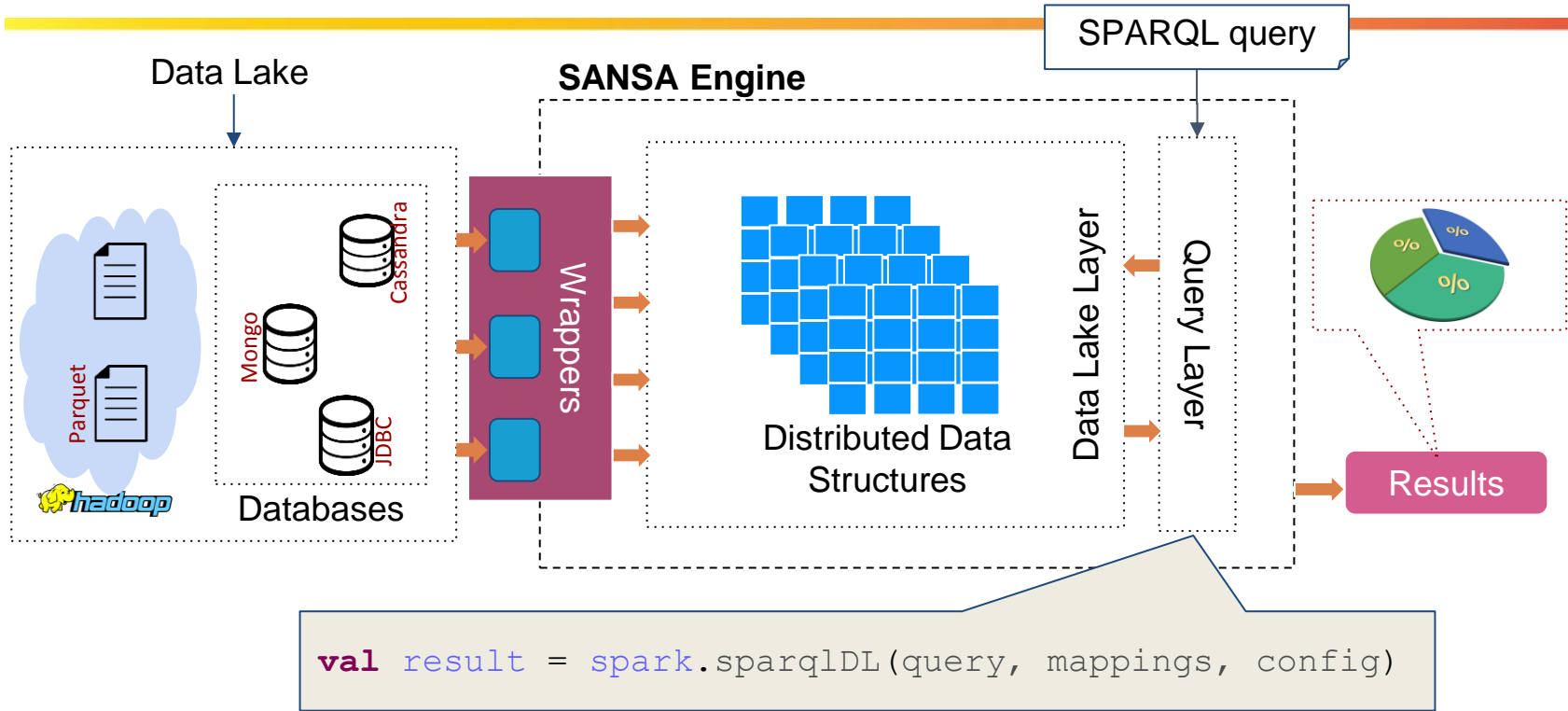
Querying via SPARQL & Partitioning



SANSA-DataLake

- ❖ A solution for the virtual Big Data integration: **Semantic Data Lake**
 - Directly query original data without prior transformation/loading
- ❖ Scalable cross-source query execution (join)
- ❖ Extensible (programmatically)
 - Do not reinvent the wheel: use existing engine connectors (wrappers)

Querying via Semantic Data Lake



SANSA-DataLake

SELECT ?b ?fn ?ln ?in
WHERE {

?b bibo:isbn ?i .
?b dc:title ?t .
?b schema:author ?a .
?b bibo:editor ?e .

?a foaf:firstName ?fn .
?a foaf:lastName ?ln .
?a vivo:Institute ?in .

?in rdfs:label ?n1 .
?in rdf:type vivo:Institute .

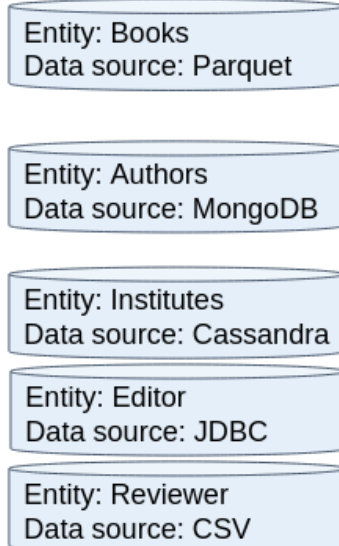
?e rdfs:label ?n2 .
?e rdf:type saws:Editor .

?r schema:reviews ?b .

Query Decomposition

Mappings
(RML)

Relevant Source Detection



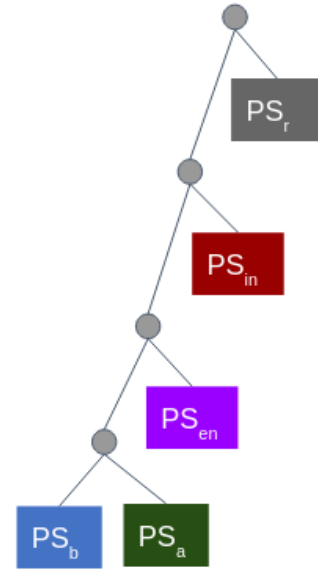
Data access

ParSet Definitions

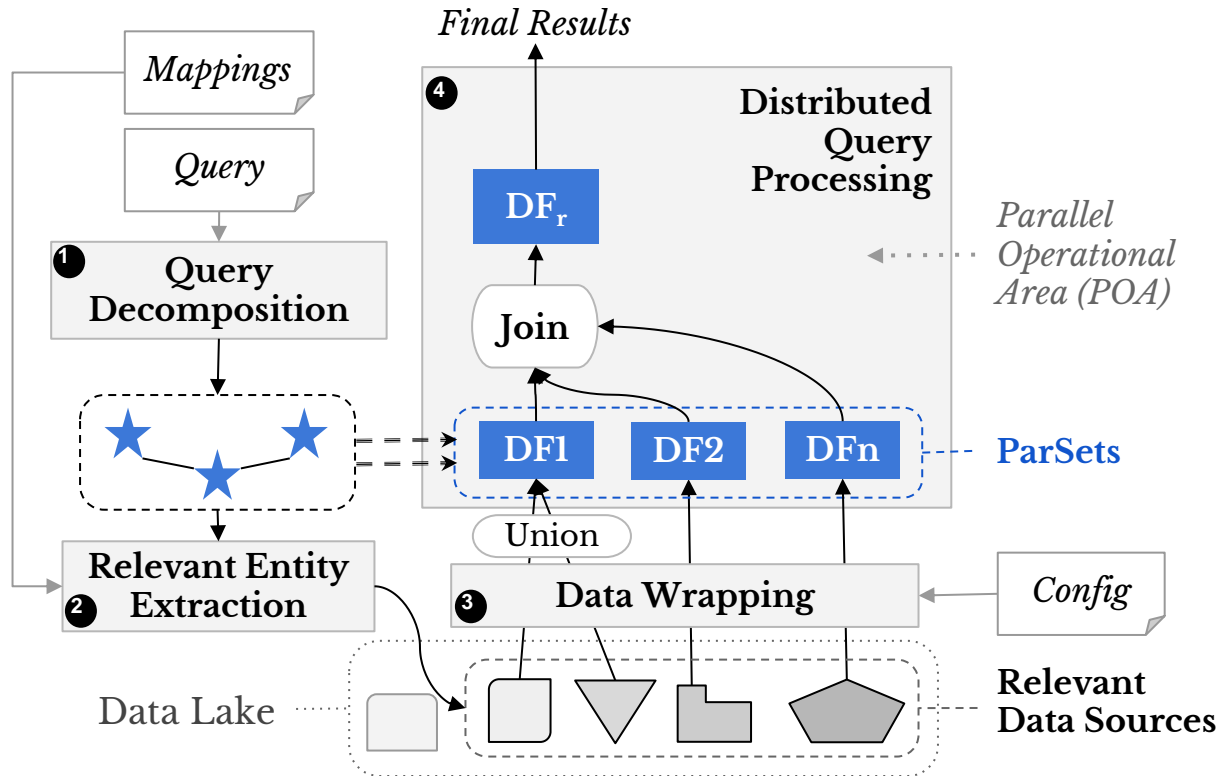


Cost Model

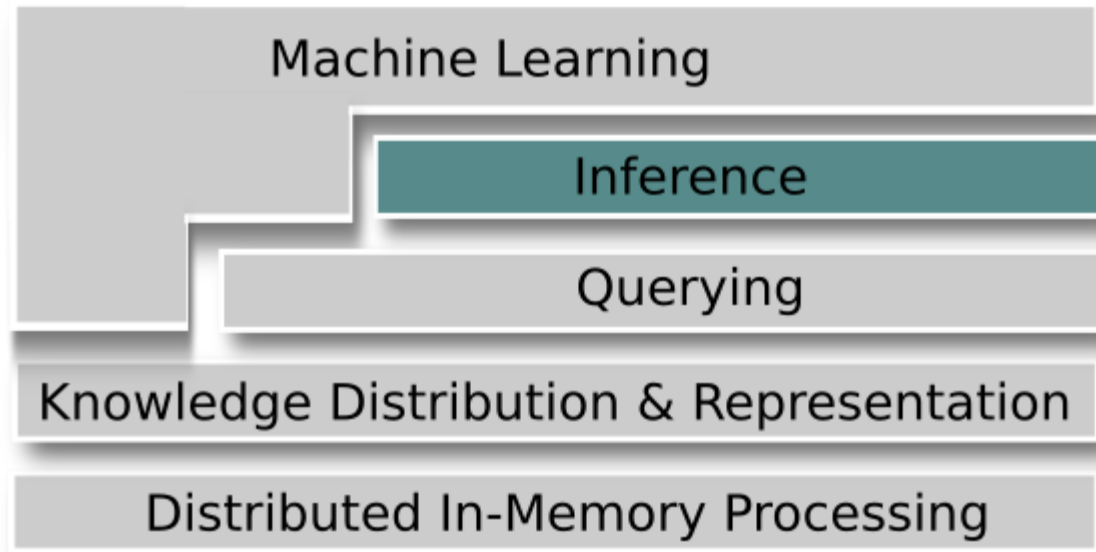
Query Planning and execution



SANSA-DataLake



SANSA: Inference Layer



SANSA: Inference Layer

- The volume of semantic data growing rapidly
- **Implicit** information needs to be derived by reasoning
 - **Reasoning**
 - the process of deducing implicit information from existing RDF data by using W3C Standards for Modelling: RDFS or OWL fragments

SANSA: Inference Layer

- **Reasoning** can be performed in two different strategies:
 - The **forward chaining** strategy derives and stores the derived RDF data back into original RDF data storage for late queries from applications (**data-driven**).
 - The **backward chaining** strategy derives implicit RDF data on the fly during query process (**goal-directed**).
- The **forward chaining** strategy has lower query response time and high load time.

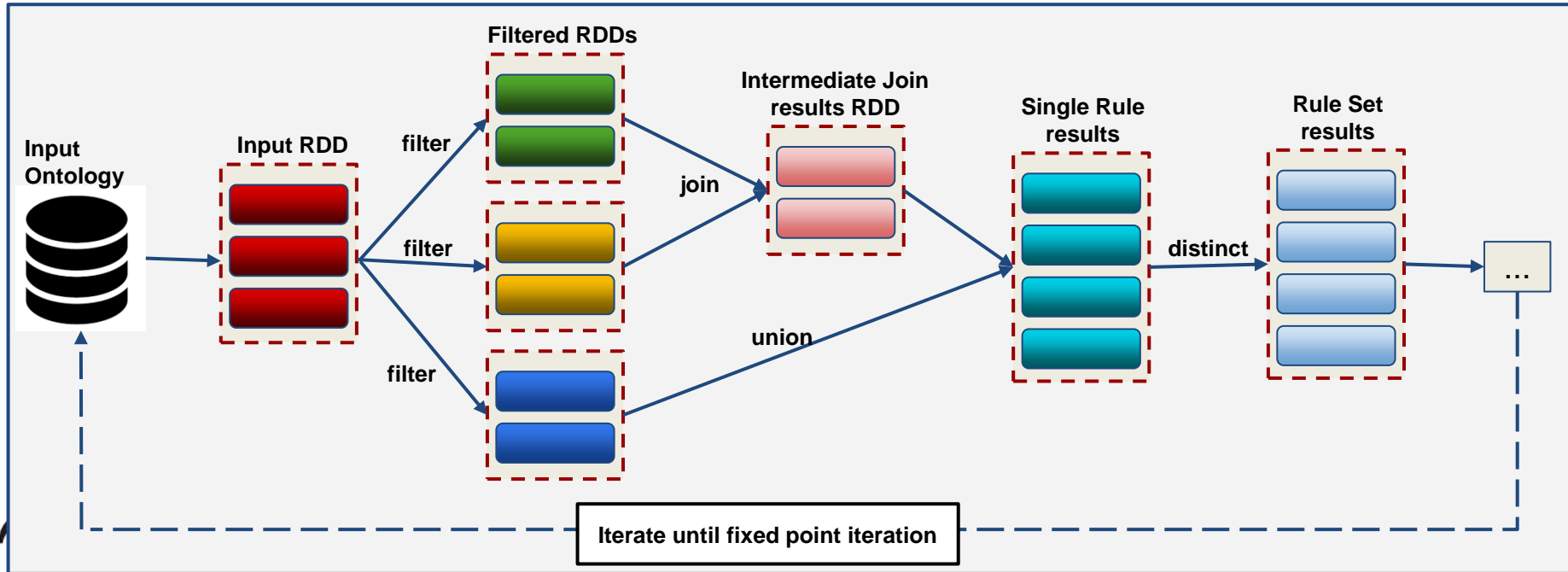
SANSA: Inference Layer

- Parallel in-memory inference via rule-based forward chaining
- Beyond state of the art: dynamically build a **rule dependency graph** for a rule set
 - Adjustable performance
 - Allows domain-specific customisation



SANSA: Inference Layer

Parallel RDFS Reasoning Algorithm based on Spark



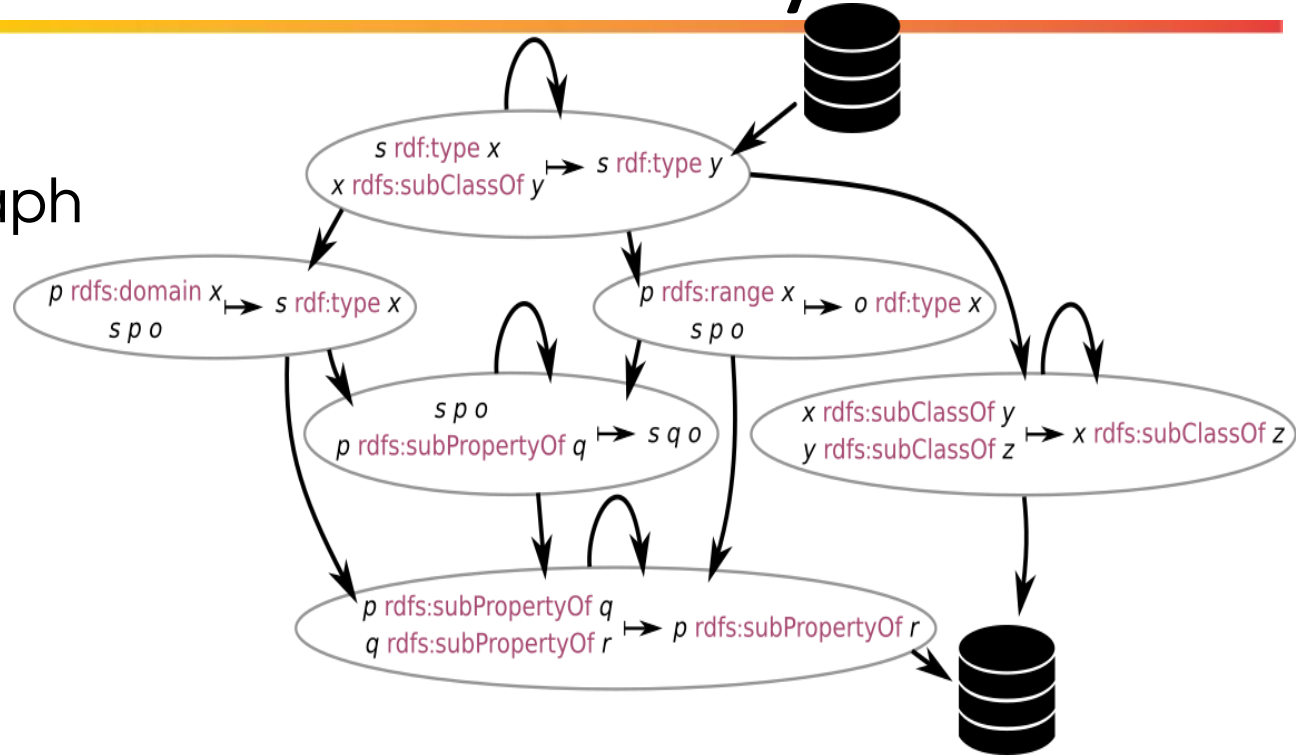
SANSA: Inference Layer

Some RDFS inference rules

- $(X \text{ R } Y), (R \text{ subPropertyOf } Q) \rightarrow (X \text{ Q } Y)$
- $(X \text{ R } Y), (R \text{ domain } C) \rightarrow (X \text{ type } C)$
- $(X \text{ type } C), (C \text{ subClassOf } D) \rightarrow (X \text{ type } D)$

SANSA: Inference Layer

RDFS rule
dependency graph
(simplified)



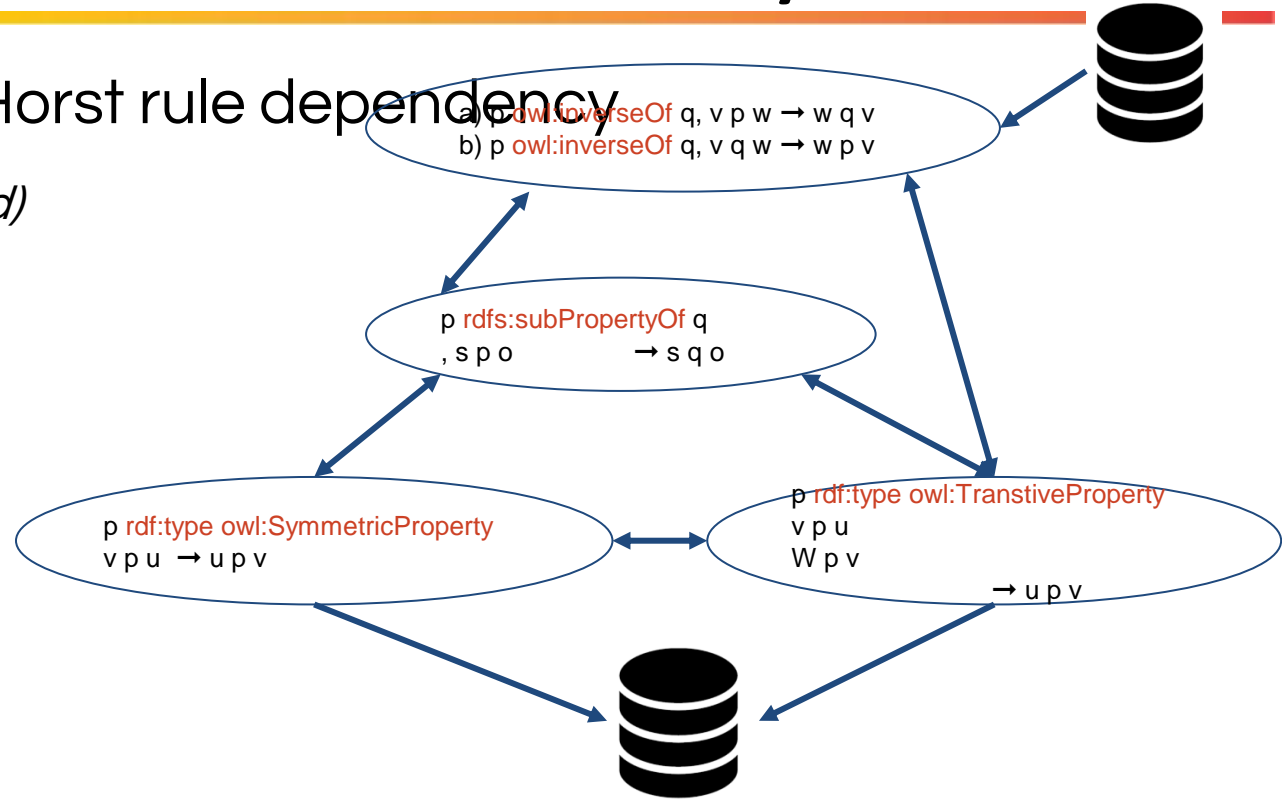
SANSA: Inference Layer

Some OWL Horst Rules:

- $p \text{ owl:inverseOf } q, v p w \rightarrow w q v$
- $p \text{ owl:inverseOf } q, v q w \rightarrow w p v$

SANSA: Inference Layer

- Part of OWL Horst rule dependency graph (*simplified*)



SANSA: Inference Layer

- SANSA-Inference Layer support RDFs and OWL-Horst reasoning in Triples and OWLAxioms
- Triple based forward chaining:

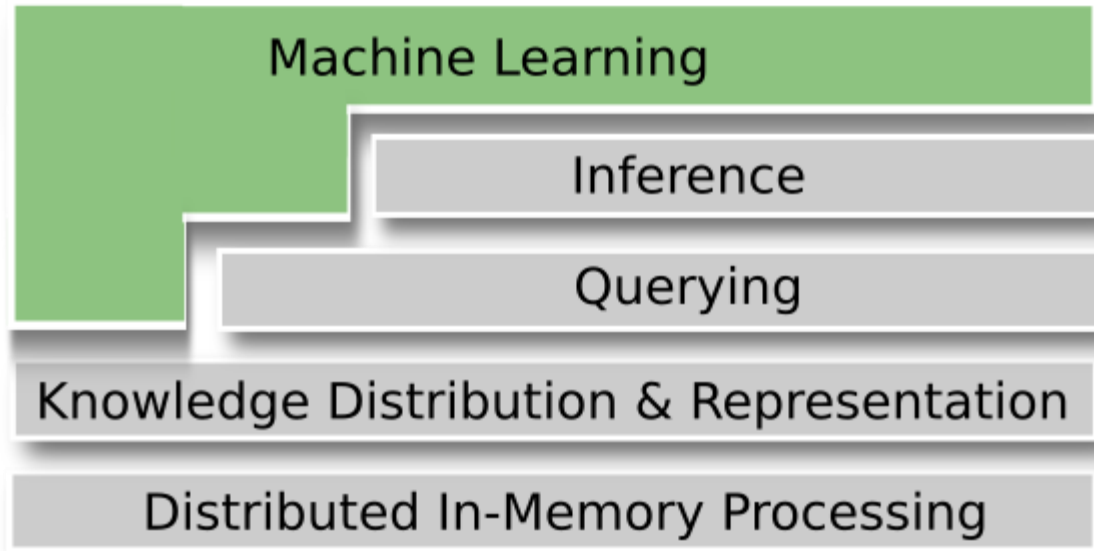
```
// load triples from disk
val graph = RDFGraphLoader.loadFromDisk(spark, input, parallelism)
val reasoner = new ForwardRuleReasonerOWLHorst(spark.sparkContext)
// compute inferred graph
val inferredGraph = reasoner.apply(graph)
```

SANSA: Inference Layer

- Axiom based forward chaining:

```
// load axioms from disk
var owlAxioms = spark.owl(Syntax.FUNCTIONAL)(input)
// create reasoner and compute inferred graph
val inferredGraph = profile match {
    case RDFS => new ForwardRuleReasonerRDFS(spark.sparkContext,
parallelism)(owlAxioms)
    case OWL_HORST => new
ForwardRuleReasonerOWLHorst(spark.sparkContext, parallelism)(owlAxioms)
    case _ => throw new RuntimeException("Invalid profile: '" +
profile + "'")
}
```

SANSA: ML Layer

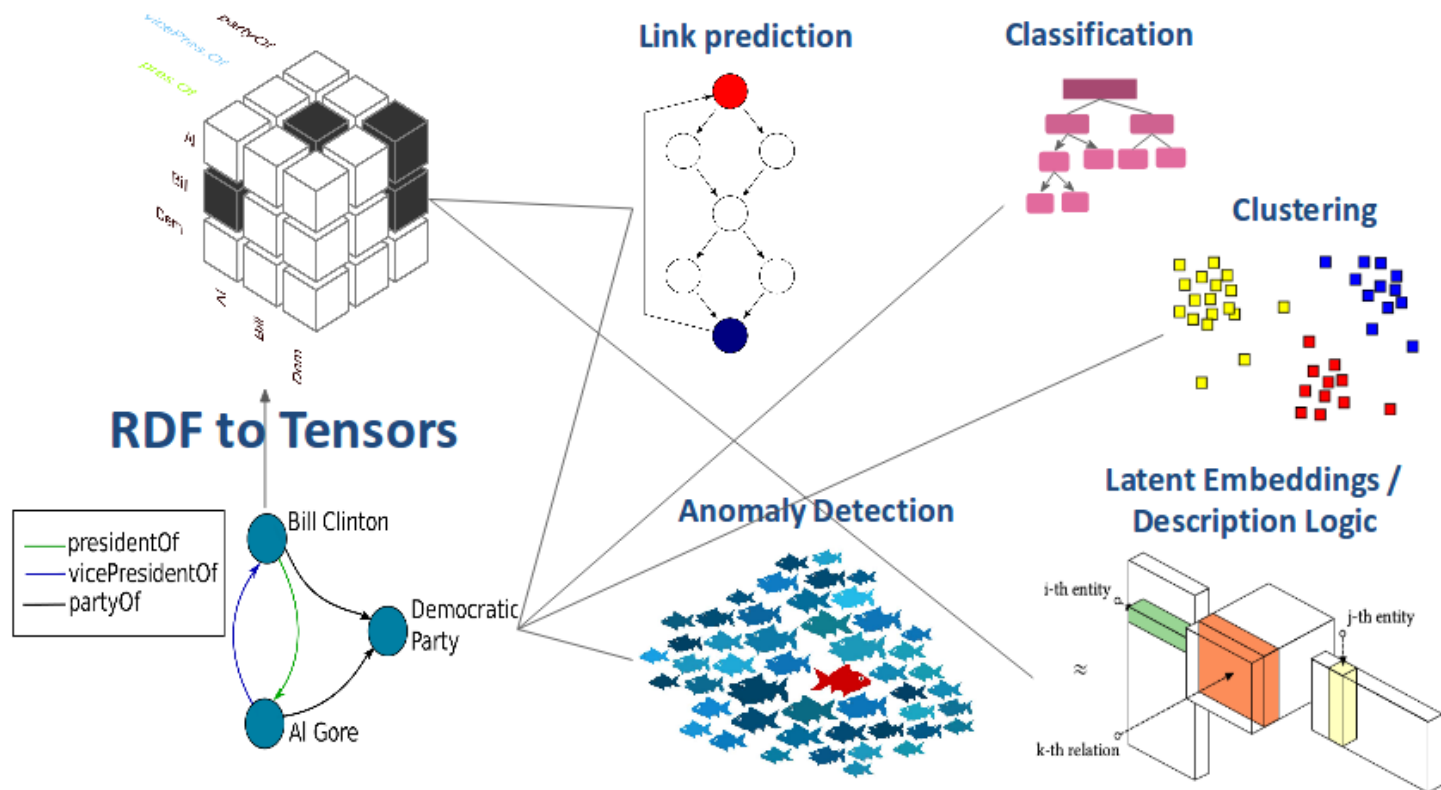


SANSA: ML Layer

- ❖ Distributed Machine Learning (ML) algorithms that work on RDF data and make use of its structure / semantics
- ❖ Algorithms:
 - Graph Clustering
 - Power Iteration,
 - BorderFlow,
 - Link based
 - Modularity based clustering
 - Association rule mining (AMIE+ = mining horn rules from RDF data using partial completeness assumption and type constraints)
 - Outlier detection
 - KG Kernels



Available Scope



RDF By Modularity Clustering

- A Hierarchical clustering method
- Starts with each vertex in its own community
- Iteratively join pairs of community choosing the one with greatest increase in the optimizing function Q
 - Optimization function identifies the significant community structure
- The cut off point maximal value of Q.
- Scales as the square of the network size

```
RDFByModularityClusteringAlg(spark.sparkContext,  
numIterations, input, output)
```

```
spark.stop
```



Power Iteration Clustering

- Simple and fast version of spectral clustering technique
- Efficient and scalable in terms of time $O(n)$ and space
- Applying PowerIteration to the row normalized affinity matrix
- Partitioning clustering algorithm
 - Outputs one-level clustering solution

```
val lang = Lang.NTRIPLES
```

```
val triples = spark.rdf(lang)(input)
```

```
val graph = triples.asStringGraph()
```

```
val cluster = RDFGraphPowerIterationClustering(spark, graph,  
output, k, maxIterations)  
cluster.saveAsTextFile(output)
```



BorderFlow Clustering

local graph clustering algorithm

Designing for directed and undirected weighted graphs

Clusters in BorderFlow:

Maximal intra-cluster density

Minimal outer-cluster density

```
val lang = Lang.NTRIPLES
val triples = spark.rdf(lang)(input)
val graph = triples.asGraph()

val borderflow = algName match {
  case "borderflow" => BorderFlow(spark, graph, output, outputevlsoft, outputevlhard)
  case "firsthardening" => FirstHardeninginBorderFlow(spark, graph, output, outputevlhard)
  case _ =>
    throw new RuntimeException("'" + algName + "' - Not supported, yet.")
}
```



Link Based Clustering

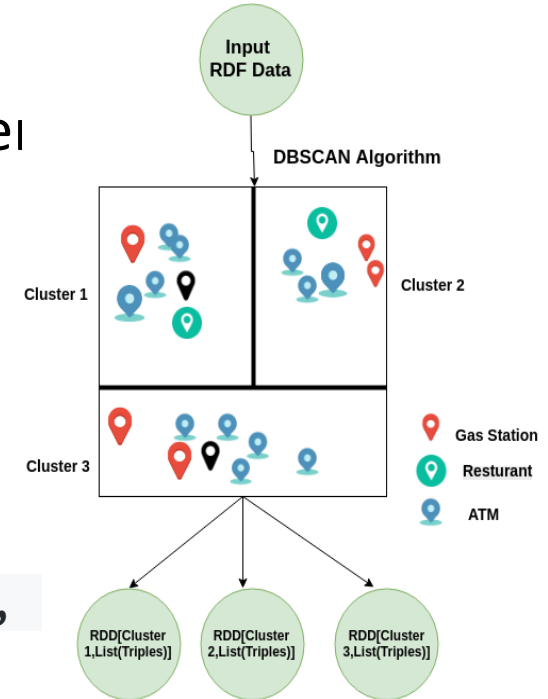
- Hierarchical link clustering method
- Bottom up approach of hierarchical called the “agglomerative”
- Clusters are created recursively
- The similarity S between links can be given by e.g. Jaccard similarity

1. `val lang = Lang.NTRIPLES`
2. `val triples = spark.rdf(lang)(input)`
3. `val graph = triples.asStringGraph()`
4. `AlgSilviaClustering(spark, graph, output, outpoteval)`

DBSCAN Clustering

- Clustering of POIs on the basis of spatially closed coordinates.
- POIs located in same geolocation are cluster of their categories.

```
1. val lang = Lang.NTRIPLES
2. val triples = spark.rdf(lang)(input)
3. val graph = triples.asStringGraph()
4. AlgSilviaClustering(spark, graph, output,
    outputeval)
```



Numerical Outliers Detection

- Detecting numerical outliers in large RDF dataset.
- Spark minHashLSH used to create the cohort of similar class.
- A scalable approach to find the outliers in a massive dataset.
- Numerical outliers detected in the data are useful for improving the quality of RDF Data

RDF Graph Kernels

Given an RDF graph constructing a tree for each instance and counting the number of paths in it.

Literals in RDF can only occur as objects in triples and therefore have no out-going edges in the RDF graph.

Apply Term Frequency-Inverse Document Frequency (TF-IDF) for vectors.

1. `val rdfFastGraphKernel = RDFFastGraphKernel(spark, triples, "http://swrc.ontoware.org/ontology#affiliation")`
2. `val data = rdfFastGraphKernel.getMLLibLabeledPoints`
3. `RDFFastTreeGraphKernelUtil.predictLogisticRegressionMLLIB(data,`
4. `iteration)`



Rule Mining

- Association Rule Mining under Incomplete Evidence (AMIE)
- Atoms : are facts with the subject and object position substituted with variables e.g. (isChildOf(?a,?b))
- Rules : made up of atoms having a head (one atom) and the body (multiple atoms)
- The body $\{B_1, \dots, B_n\}$ predicts the head $r(x, y)$
- A Rule can be written as
$$B_1 \wedge B_2 \wedge \dots \wedge B_n \Rightarrow r(x, y)$$
- OR
$$\vec{B} \Rightarrow r(x, y)$$



Projects Powered by SANSA



BIG DATA EUROPE

Empowering Communities
with Data Technologies

Big Data Europe

Integrating Big Data, software & communities for addressing
Europe's societal challenges

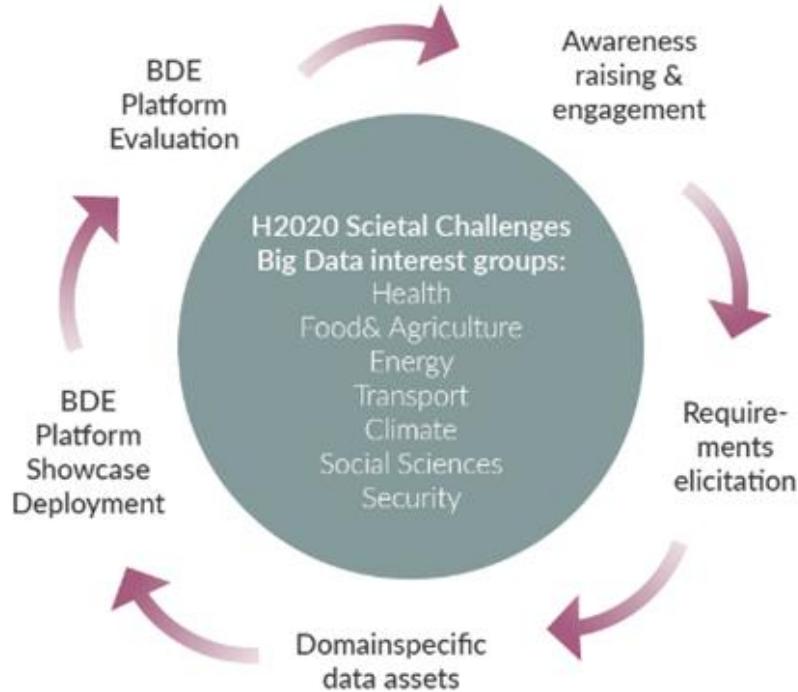
<<https://www.big-data-europe.eu/>>



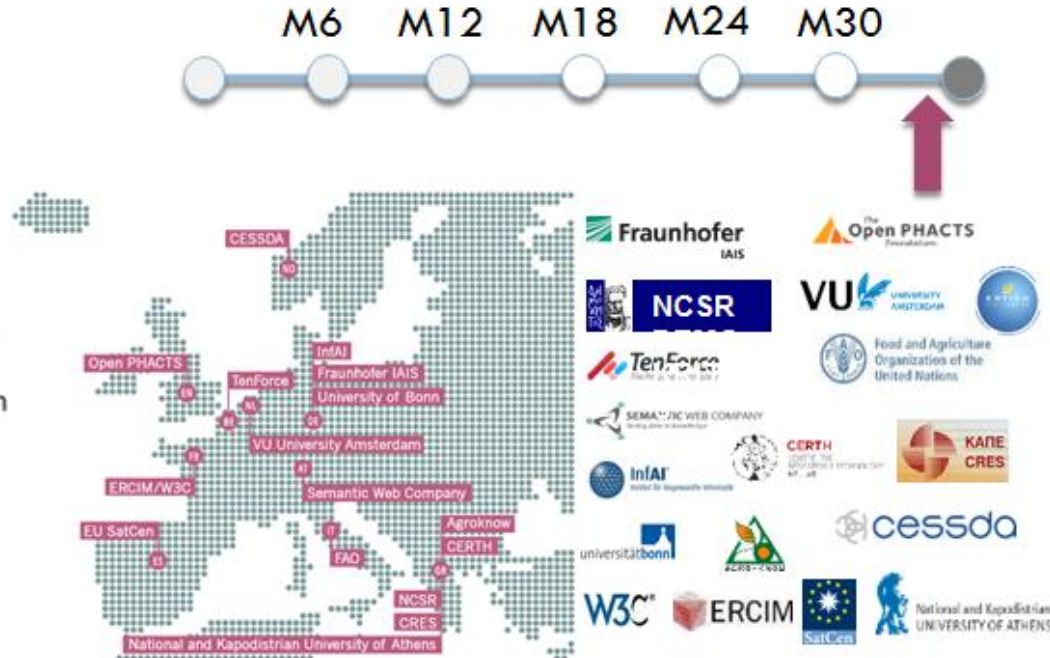
LEARNING, APPLYING, MULTIPLYING BIG DATA ANALYTICS



Big Data Europe

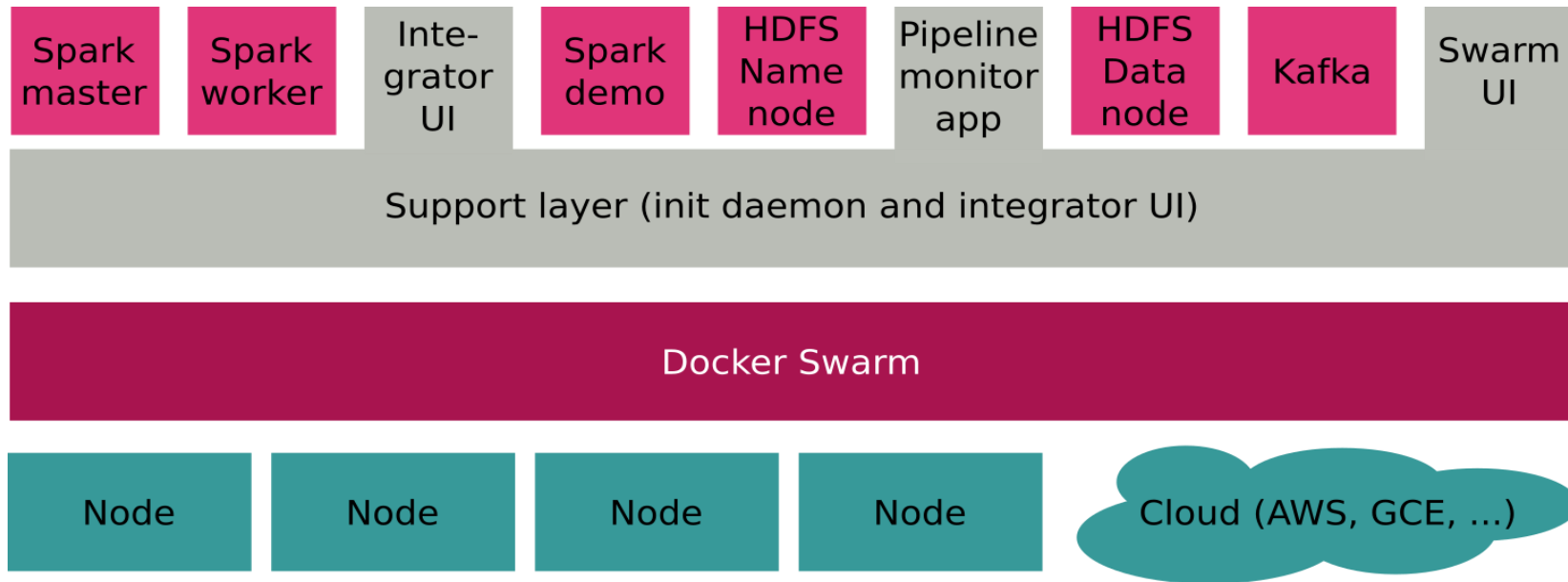


Coordination & Support Action 2015-17



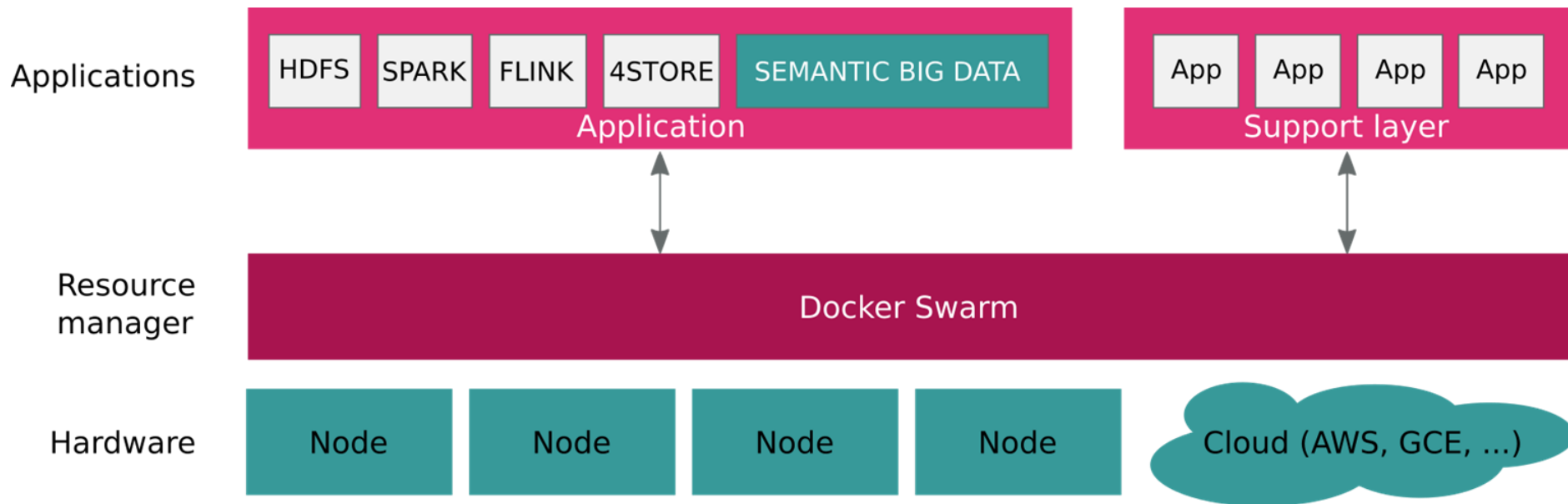


Platform Architecture

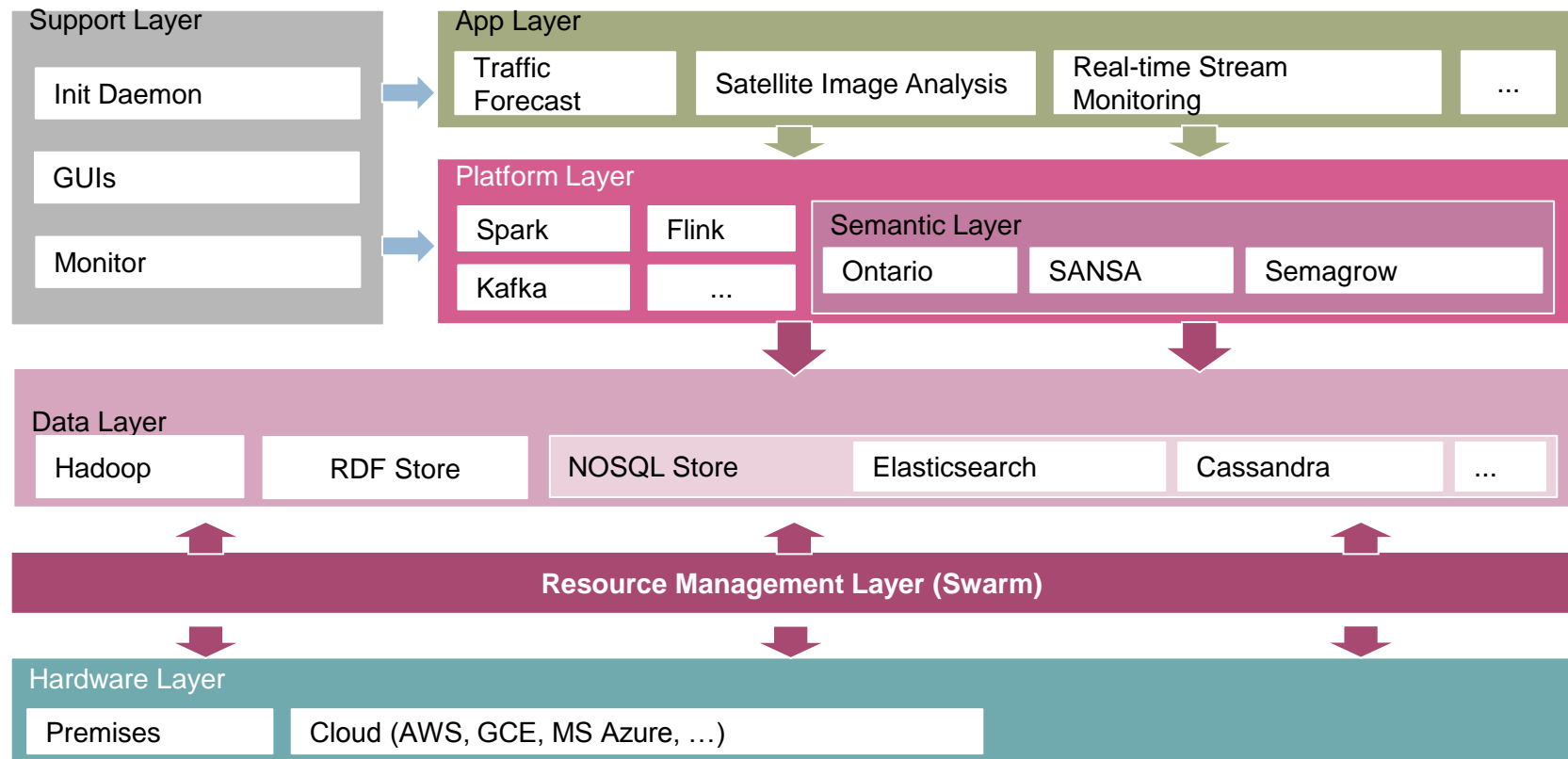




Platform Architecture



Platform Architecture





SPECIAL

SPECIAL

Scalable Policy-aware Linked Data Architecture For Privacy, Transparency and Compliance

<<https://www.specialprivacy.eu>>

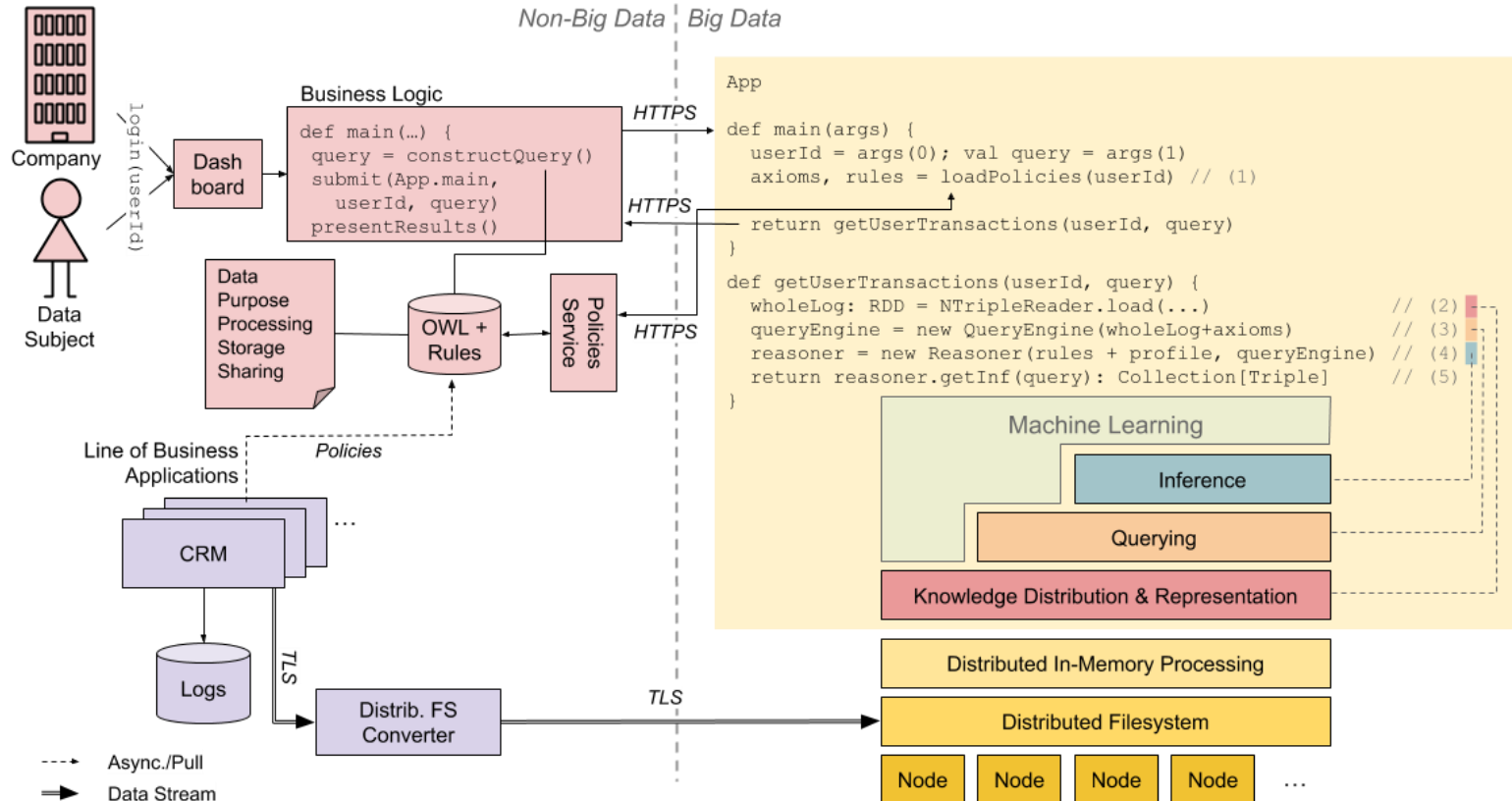


LEARNING, APPLYING, MULTIPLYING BIG DATA ANALYTICS

Overview

- ❖ Addresses the contradiction between Big Data innovation and privacy-aware data protection
- ❖ It allows citizens and organisations to share more data, while guaranteeing data protection compliance
- ❖ Combine and significantly extend big data architectures to handle Linked Data, harness them with sticky policies as well as scalable queryable encryption

SPiRiT





SLIPO

Scalable Linking and Integration of Big POI data

<www.slipo.eu>



Context



SLIPO

❖ Points Of Interest (POI)

- Our lives evolve around locations → POIs are locations which exhibit an interest and embed metadata
- They are part of the trinity with roads and addresses. [What is there? How do I get there? Where do I find something?]

❖ Data Economy leaders are producing, using and offering POIs



Challenges



Same POI, different geometries (points, polygons).



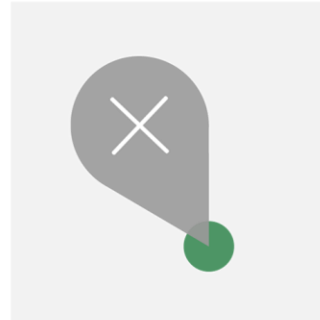
Different POIs, same location.



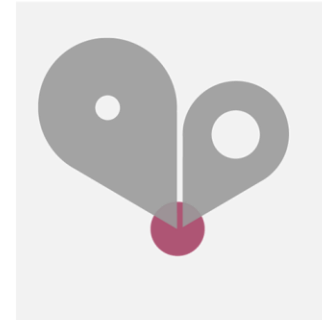
POI (point) within another POI (polygon).



Different POIs belonging to the same chain.



Defunct POI.



Same POI, change of type.

Goals



SLIPO

- Apply well established Linked Data principles and standards (RDF/S, OWL, GeoSPARQL) to define a global and vendor agnostic schema for representing POI data and metadata,
- Develop ETL software, mappings and guidelines for transforming POIs of conventional formats into Linked POI data,
- Develop Interlinking and Fusion software that will be able to integrate POIs from several disperse data sources.



Alethio Collaboration

A comprehensive suite of blockchain analyses for the Ethereum network

<<https://aleth.io/>>

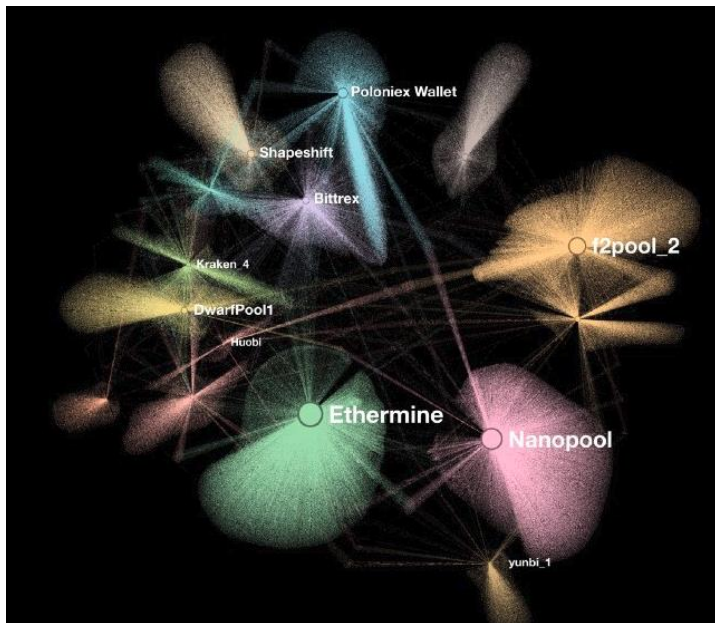
<<http://sansa-stack.net/collaboration-alethio/>>

Alethio + SANSA in a nutshell



- SANSA notebooks were used to process 38 billion triple Ethereum data on a 100 nodes cluster
- The data is filtered and queried in real time
- Characterize movements between groups of Ethereum accounts and aggregate flows over history of blockchain
- RDF represented as a graph using GraphX

Hubs and Authorities in EthOn



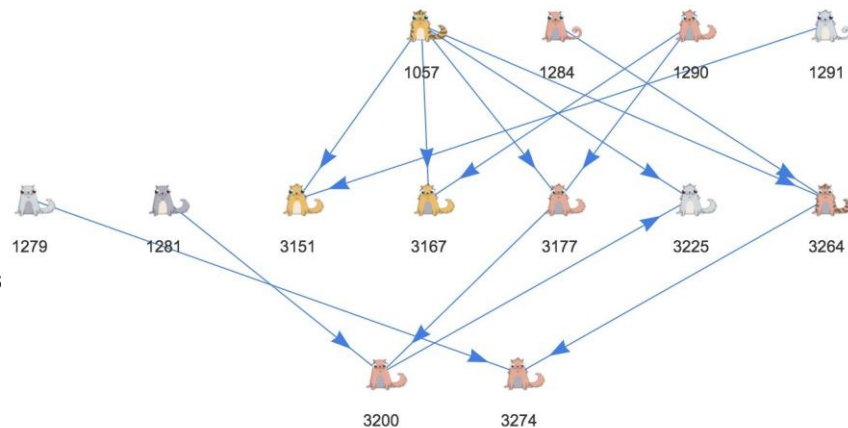
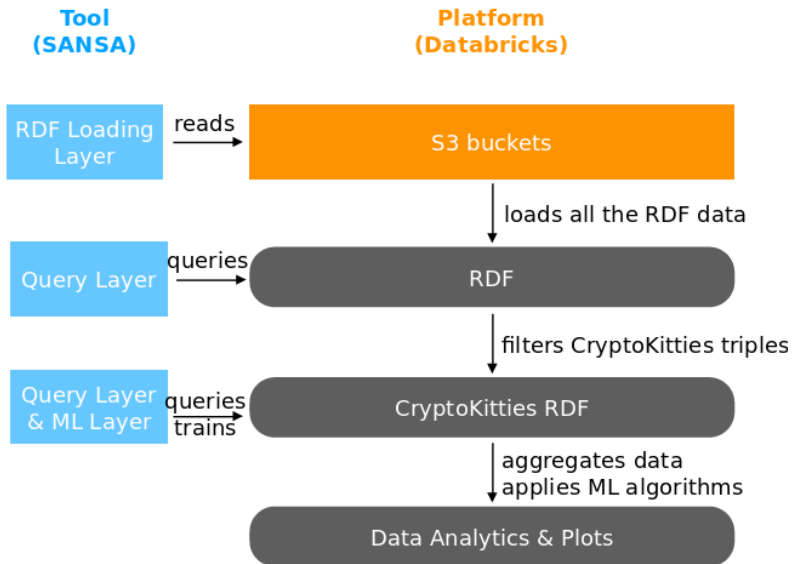
Transaction Network of Top Hubs and Authorities

<https://media.consensys.net/the-hubs-authorities-in-transaction-network-powered-by-sansa-and-graph-analysis-9bf271c2e5f9>

- Analysis of value transaction network graph using
 - Connected Components
 - Page Rank
- Analysis of Typical Behavior Patterns of Exchanges' Deposit Wallets

Profiting from Kitties on Ethereum

- **Context:** Cryptokitties <<https://www.cryptokitties.co/>>
- **Challenge:** Analyzing game performance and customer



Other user-cases



<www.ec-better.eu>

Big Data Earth Observation Technology

SANSA is used for computing statistics and extract knowledge from the metadata files associated to the satellite picture in order to address specific use-cases from the list of challenges of the project.



<<https://qualichain-project.eu>>

QualiChain

SANSA is used to compute analytics on top of educational blockchain to achieve specific tasks such as *e.g.* clusterisation of skills among list of applicants and so on.



<<http://boost40.eu/>>

Towards a European Data Space

SANSA will be used for covering heterogeneity between stakeholders and data providers for better and efficient Data Processing, Data Management and Data Analytics.

10+ more use cases

<http://sansa-stack.net/powered-by/>



Conclusion

Conclusions and Next steps

- ❖ A generic stack for (big) Linked Data
 - Build on top of a state-of-the-art distributed frameworks (**Spark**, **Flink**)
- ❖ Out-of-the-box framework for scalable and distributed semantic data analysis combining semantic web and distributed machine learning for (1) querying, (2) inference and (3) analytics of RDF datasets.
- ❖ Next steps
 - Support for SPARQL 1.1 and other partitioning strategies (Query Layer)
 - Backward chaining and better evaluation (Inference Layer)
 - More algorithms and definition of ML pipelines (ML Layer)



Associated Publications

1. [Distributed Semantic Analytics using the SANSA Stack](#) by Jens Lehmann, Gezim Sejdiu, Lorenz Bühmann, Patrick Westphal, Claus Stadler, Ivan Ermilov, Simon Bin, Muhammad Saleem, Axel-Cyrille Ngonga Ngomo and Hajira Jabeen in Proceedings of 16th International Semantic Web Conference – Resources Track (ISWC'2017), 2017 [\[BibTex\]](#).
2. [The Tale of Sansa Spark](#) by Ivan Ermilov, Jens Lehmann, Gezim Sejdiu, Lorenz Bühmann, Patrick Westphal, Claus Stadler, Simon Bin, Nilesh Chakraborty, Henning Petzka, Muhammad Saleem, Axel-Cyrille Ngonga Ngonga, and Hajira Jabeen in Proceedings of 16th International Semantic Web Conference, Poster & Demos, 2017 [\[BibTex\]](#).
3. [DistLODStats: Distributed Computation of RDF Dataset Statistics](#) by Gezim Sejdiu, Ivan Ermilov, Jens Lehmann, and Mohamed Nadjib-Mami in Proceedings of 17th International Semantic Web Conference, 2018. [\[BibTex\]](#)
4. [STATisfy Me: What are my Stats?](#) by Gezim Sejdiu; Ivan Ermilov; Jens Lehmann; and Mohamed-Nadjib Mami. In *Proceedings of 17th International Semantic Web Conference, Poster & Demos*, 2018.
5. [Profiting from Kitties on Ethereum: Leveraging Blockchain RDF with SANSA](#) by Damien Graux; Gezim Sejdiu; Hajira Jabeen; Jens Lehmann; Danning Sui; Dominik Muhs; and Johannes Pfeffer. In *14th International Conference on Semantic Systems, Poster & Demos*, 2018.
6. [SPIRIT: A Semantic Transparency and Compliance Stack](#) by Patrick Westphal, Javier Fernández, Sabrina Kirrane and Jens Lehmann. In *14th International Conference on Semantic Systems, Poster & Demos*, 2018.
7. [Divided we stand out! Forging Cohorts fOr Numeric Outlier Detection in large scale knowledge graphs \(CONOD\)](#) by Hajira Jabeen; Rajjat Dadwal; Gezim Sejdiu; and Jens Lehmann. In *21st International Conference on Knowledge Engineering and Knowledge Management (EKAW'2018)*, 2018.
8. [Clustering Pipelines of large RDF POI Data](#) by Rajjat Dadwal; Damien Graux; Gezim Sejdiu; Hajira Jabeen; and Jens Lehmann. In *ESWC 2019 (Poster Track)*.





Prof. Jens Lehmann



Dr. Damien Graux



Dr. Hajira Jabeen

THANK YOU !



LEARNING, APPLYING, MULTIPLYING BIG DATA ANALYTICS

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 809965.

