Data Quality Assessment and Anomaly Detection Via Map / Reduce and Linked Data: A Case Study in the Medical Domain

Stephen Bonner, Andrew Stephen McGough, Ibad Kureshi, John Brennan, Georgios Theodoropoulos, Laura Moss, David Corsar and Grigoris Antoniou.

Durham University

University of Glasgow

1495 UNIVERSITY OF ABERDEEN

University of HUDDERSFIELD

NHS

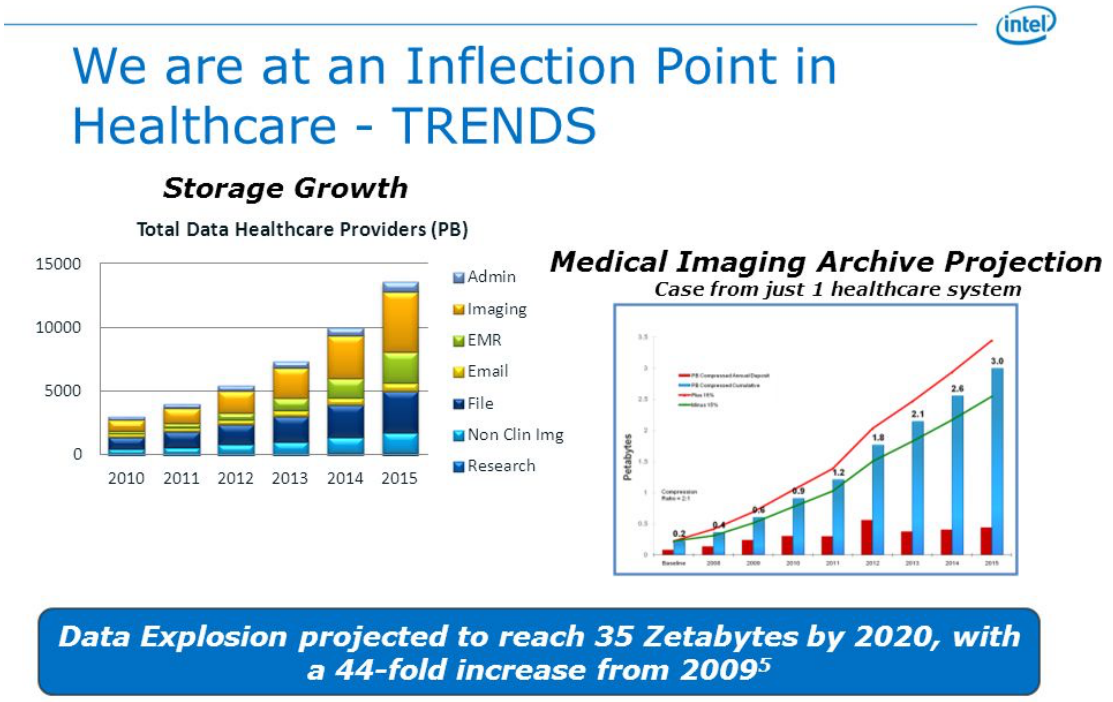
EPSRC
Engineering and Physical Sciences Research Council

Motivating Examples

- **Big Data and Healthcare**

- Big data is a rapidly growing area of interest which offers the potential for great advances but also comes with significant challenges. The healthcare fields have the potential to be one of the *biggest contributors to*, and *benefactors from*, the big data phenomenon.

- In 2012, Intel estimated the overall volume of worldwide healthcare data to be 500 petabytes; this figure is predicted to grow exponentially.



Motivating Examples

- **Big Data and Healthcare**

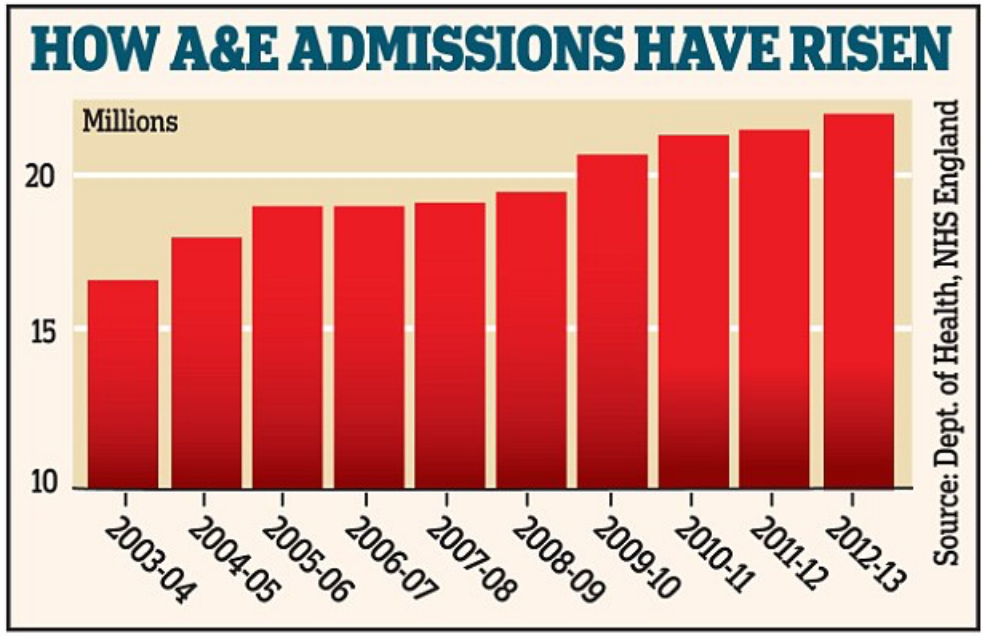
- However, previous studies have estimated that error rates in medical databases can range from 2.3% to 26.9%. Big data does not always mean good data!

- Left unchecked, low quality data can lead to sub-standard patient treatment and have substantially adverse affects on research findings.

- We started to wonder that if we could remove these errors, would it lead to better quality medical research?

NHS Data Details

- **So What Data Are We Talking About Here?**
- We are processing NHS critical unit time series data, ECG, EEG, etc, stored in RDF format. 1 patients data, for 1 day, is approximately 3 million triples.
- Massive datasets - NHS in England deals with over ***1 million patients every 36 hours***.
- Admissions increasing year on year across the NHS.
- NHS has an annual budget of ***£115.4 billion***, spending lots on new, sensor rich, equipment.
- Looking for way to remove errors in datasets.



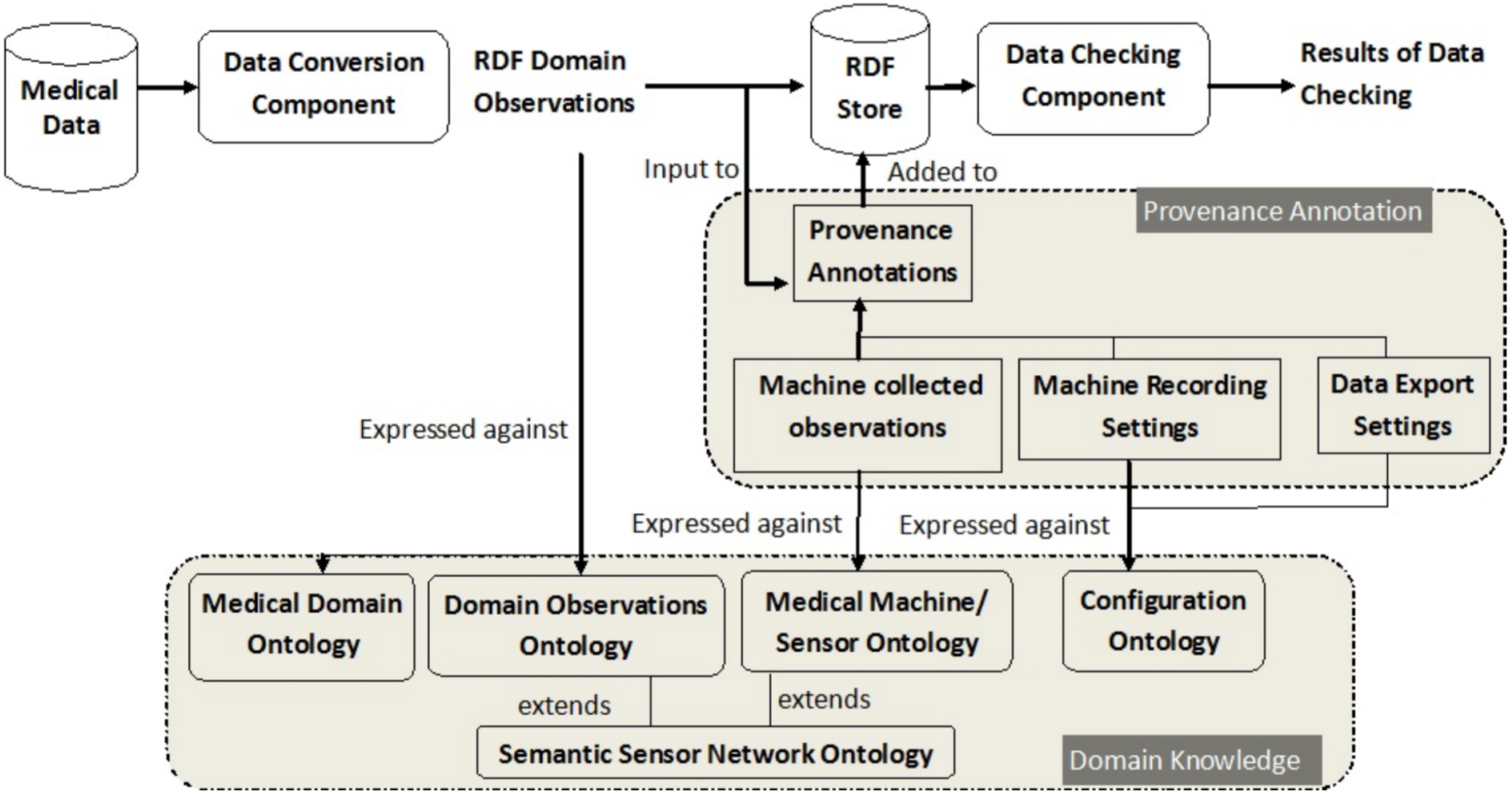
NHS

Previous Work

- **Our Previous Approach To Tackle This Problem**
- Dr Moss's previous work described an approach and supporting framework, written in Jena, for the identification of errors in medical datasets using linked data and semantic web technologies.
- This work enhanced the dataset with additional provenance metadata from the web of linked data.
- Eight SPARQL queries were created to assess the data quality.
- While the Jena based SPARQL approach was highly successful in identifying errors in the data, it was unacceptably slow for real world data (taking several hours per patient), and unable to scale.
- We aimed to improve on this by creating a Map / Reduce based algorithm. Focusing on Hadoop as the NHS department we work with requested this.

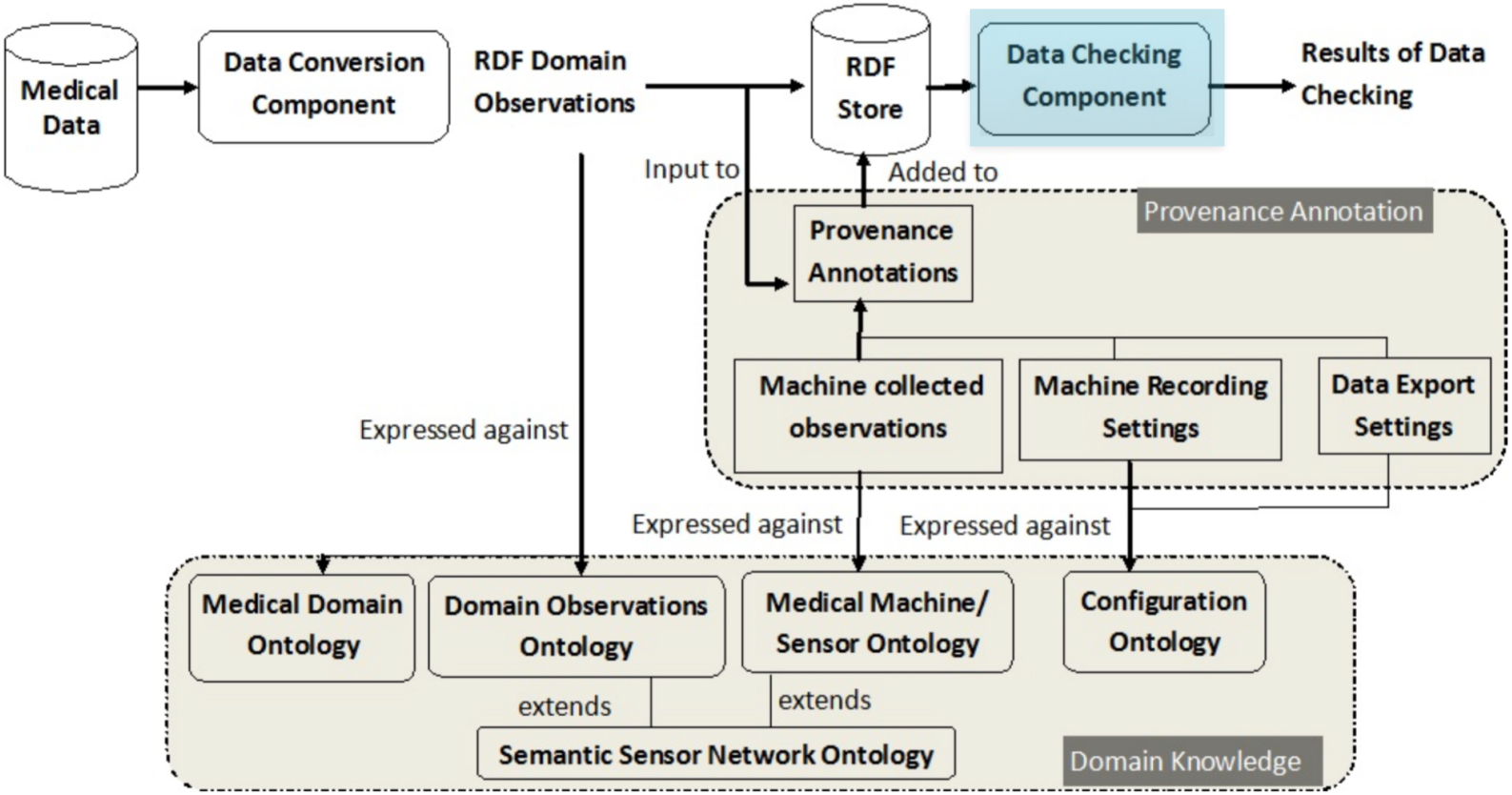
Previous Work

- **Overview Of The Previous Approach**



Previous Work

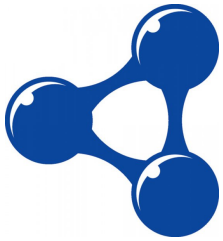
- **Overview Of The Previous Approach**

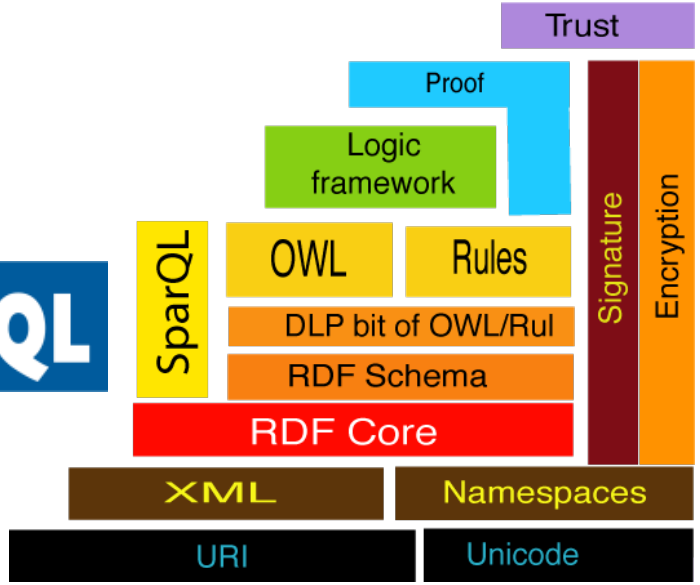


Background and Related Work

- **What is the Semantic Web?**

- The semantic web has the aim of capturing some of the meaning behind data. Currently it is implemented as series of layers -
- **RDF** - The Resource Description Framework. Represents relationships between data as Subject, Predicate and Object.
- **OWL** - Web Ontology Language. Allows for the creation of ontologies.
- **SPARQL** - SPARQL Protocol and RDF Query Language. Used to query data stored as RDF. Very similar syntax to SQL.





Motivation and Analysis

- **Data Analysis -**

- We utilised some graph theory techniques to analyse the RDF data before designing our solution.

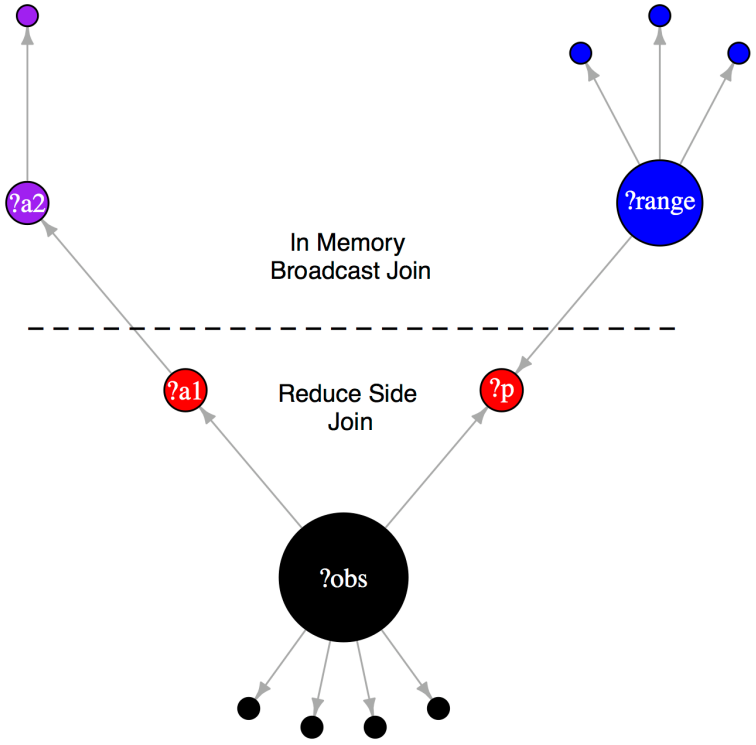
- We analysed an RDF dataset of 2,733,290 unique triples, which resulted in a graph of 595,597 unique vertices and 2,733,290 edges.

- Interesting characteristics from studying the vertices listed by in degree. As the table shows, all nodes with a high in degree have a low out degree.

Node Name	d_v^{in}	d_v^{out}
<Patient/15>	158300	1
<PhysiologicalObservation>	158299	0
<ssn/Observation>	158299	0
<PatientData/Reading>	158299	0
<PhysiologicalObservationValue>	158299	0
<ObservationValue>	158299	0
<PhysiologicalSensorOutput>	158299	0
<SensorOutput>	158299	0
<ObservationCollection>	39536	0
<Timepoint>	39536	0

Motivation and Analysis

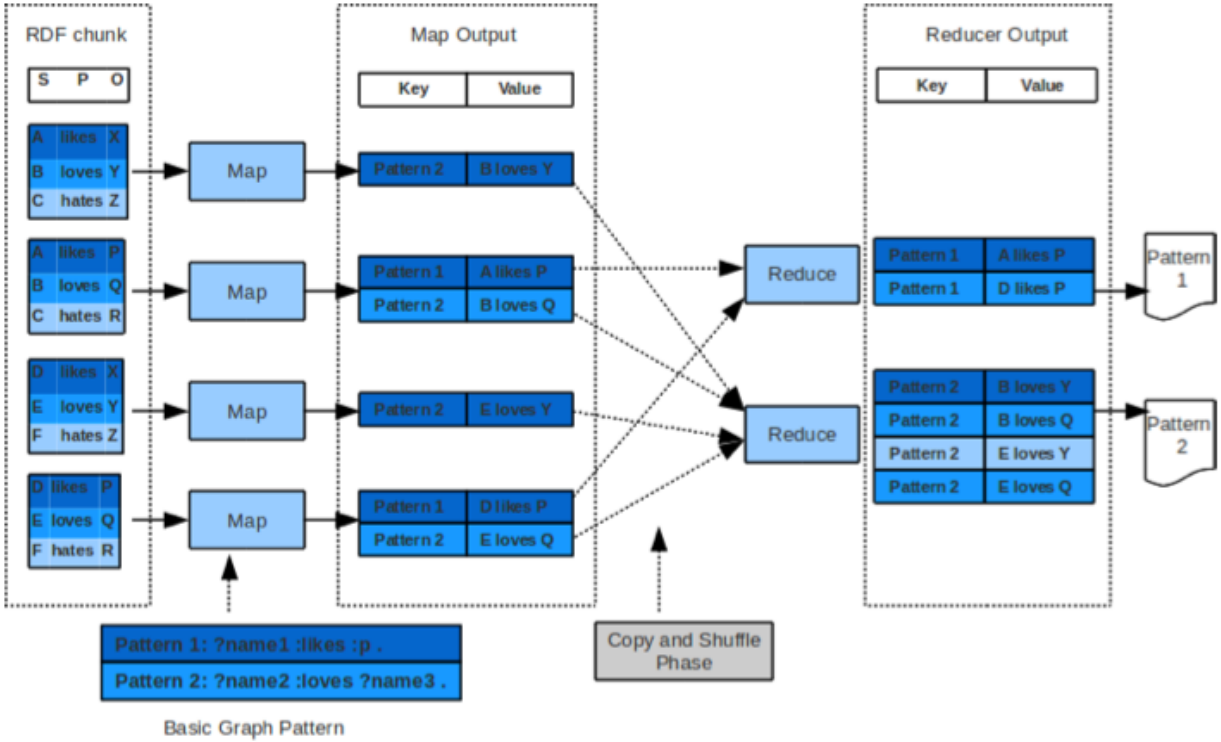
- **Data Analysis -**
- Also interesting to consider the SPRQL queries as a graph.
- Good illustration of the large number of joins required for each query. Have to complete this logic for each valid set of triples.
- This posses a problem for a Hadoop based approach as Hadoop struggles with joins.
- However several techniques have emerged to complete joins via Map / Reduce. It's an active area of research within the community.



Motivation and Analysis

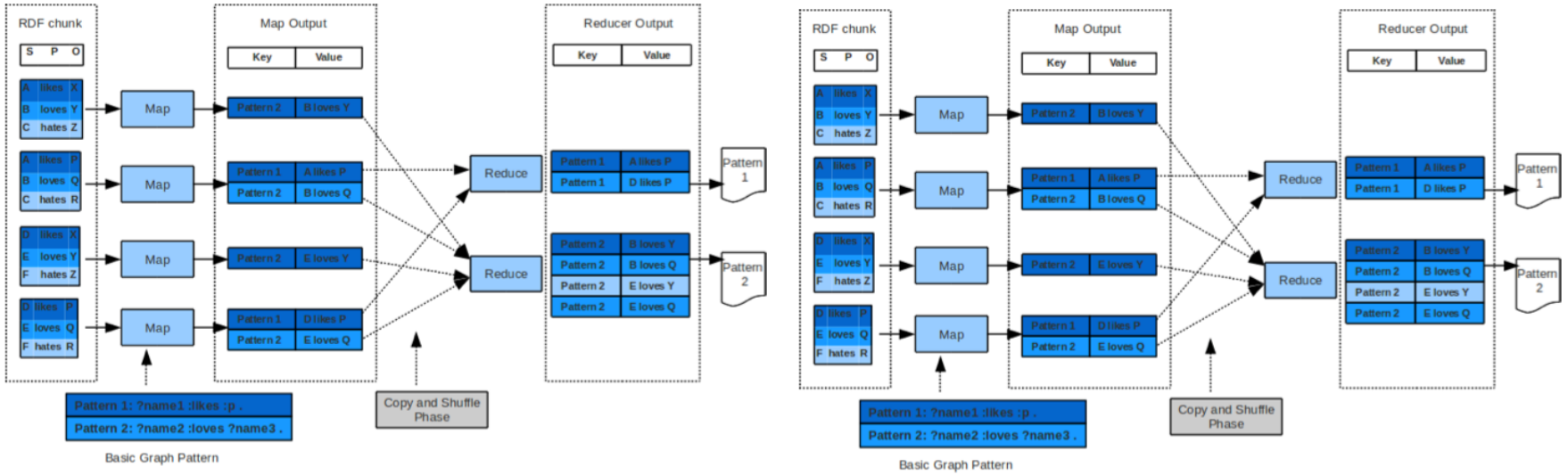
- **Hadoop Joining Techniques -**
- Three most popular called Map-Side, Reduce-Side and Cascade joins.
- **Reduce Side:**

$$(P(a,b) \bowtie Q(b,c)),$$



Motivation and Analysis

- **Cascade Joins:** Used when more then one join is required.



$$(R(a,b) \bowtie S(b,c) \bowtie T(c,d)).$$

$$(R(a,b) \bowtie S(b,c) \rightarrow IntermediateResult(a,b,c)).$$

$$(IntermediateResult(a,b,c) \bowtie T(c,d) \rightarrow FinalResult(a,b,c,d)).$$

- Takeaway is that this is a slow and inefficient process in Hadoop world.

Theoretical Model

- **Our Key Optimisations**

- **Super Query** - We noticed that the vast majority of the joins required by the eight different queries were identical. Rather then recompute all the joins for each query, we collect all the triples required by all the queries at once. Saving the need for the re-computation of joins.

- **Triple Group Creation -** We created a method to partition the complete RDF graph into smaller sub-graphs. We chose triple groups, these are triples which share a number of common join keys in the original SPARQL queries.

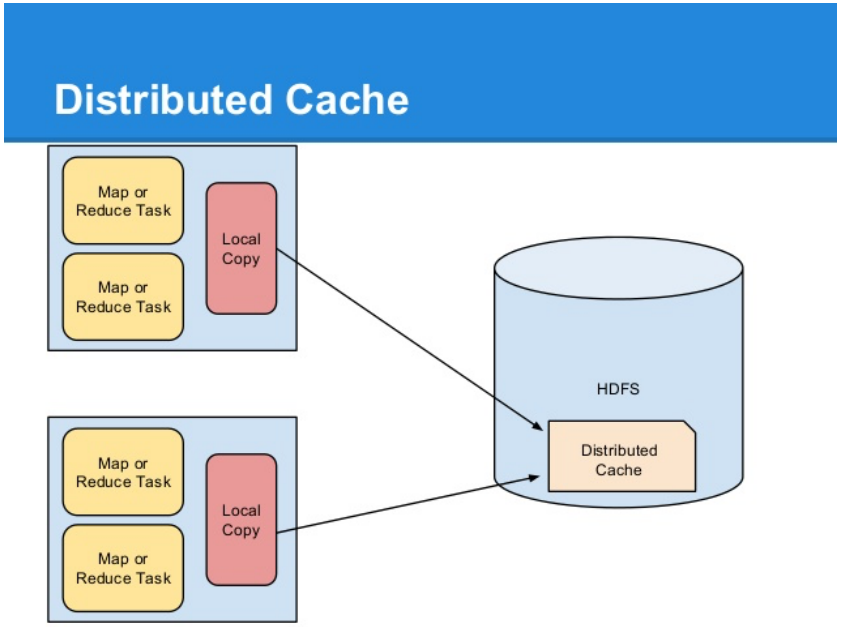
```
SELECT ?obs ?p ?htime ?max ?value WHERE{
?range a med:AcceptableRange .
?range med:clinicalRangeMax ?max .
?range pd:hasParameter ?p .

?obs a mo:PhysiologicalObservation;
?obs ssn:observedProperty ?p .
?obs ssn:observationResultTime ?time .
?obs pd:atHumanTime ?htime .
?obs ssn:observationResult ?a1
```

Triple Group Name	Number Of Elements
?obs	3,426,188
?range	518
?cs	446

Theoretical Model

- **Triple Group Creation -**
- Basically a way of partitioning the complete RDF subgraph into smaller, more manageable graphs. Based on common join keys.
- Once we have split the complete dataset into triple groups, we can push the smaller ones in a special feature of Hadoop called the distributed cache. This allows a small set of data to be pushed to all the nodes in the cluster.
- This avoids the endless cycle of cascade reduce joins. Much more efficient.
- Only limited space in the distributed cache. So we only push the small, but frequently joined to, triple groups into it.



Hadoop Implementation

- **Our new Hadoop implementation -**
- Is a pure MR implementation, requiring no additional software.
- The complete query, including all the joins is completed in just two complete MR iterations:
- **Selection Phase** is used as a data reduction phase along with the identification of the triples which will become part of the in-memory Broadcast join. These triples are exchanged between nodes as part of the join phase.
- **Join Phase** performs the final joins of the in-memory data with the locally held data, along with the generation of the final results from the query.
- Compared to alternative approaches, this is much quicker and more efficient.

Experimental Evaluation and Results

- **Hardware:**
- Testing performed on a small development Hadoop cluster, comprising a head node with eight data nodes.
- Software stack of CentOS 6.5 64-Bit, Java OpenJDK 1.7.0 51 and Hadoop 1.2.1.
- All nodes had identical hardware – an Intel Q8400 quad-core processor, 8GB of Memory, a 250GB (7200 RPM) HDD and communicated via a dedicated Gigabit switch.

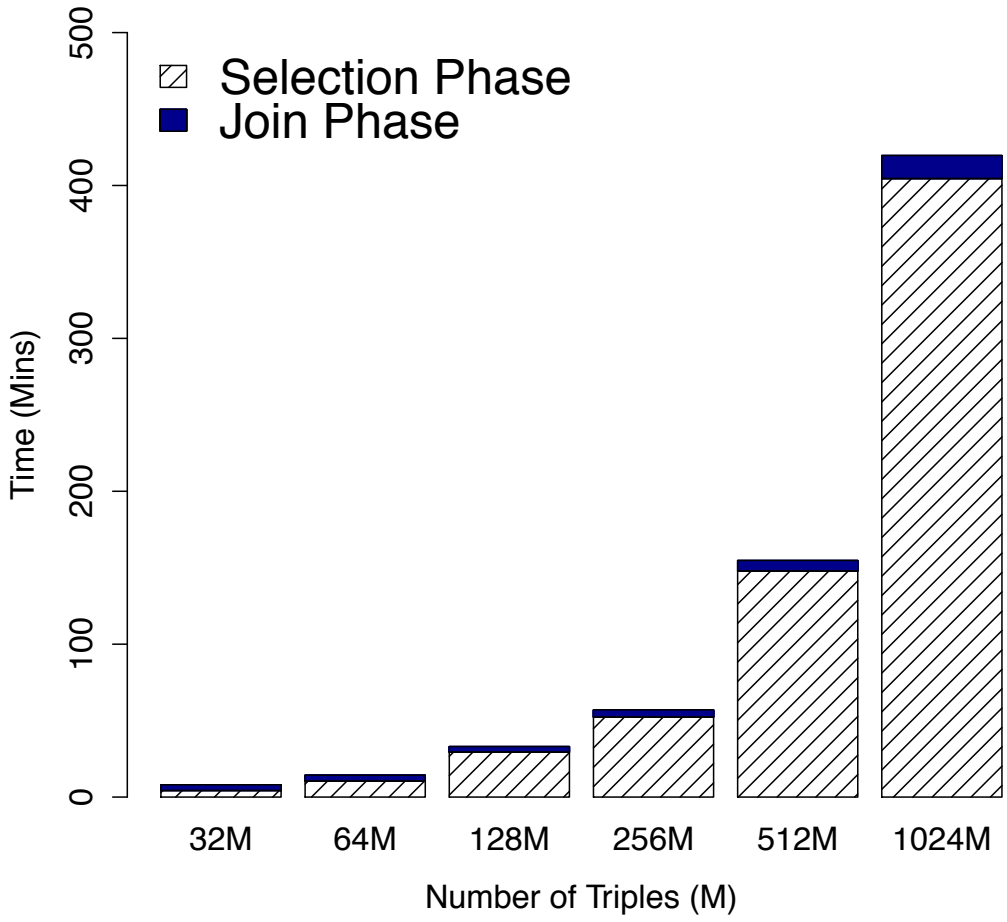
- **Input Data:**
- Due to the sensitivity of the original datasets, only a small medical dataset was available so additional records were synthetically generated.
- The algorithm used for the data generation retains the structure and distribution from the real world dataset, but inserts new randomly generated values for the variable triple elements.
- Statistically mimics the structure and distribution of the existing datasets.

Experimental Evaluation and Results

- This Figure shows the query performance of our new approach on the 8 node cluster across a range of data sizes.

- Interesting to note how the run time is split between the two stages - majority of the time spent in the selection phase.

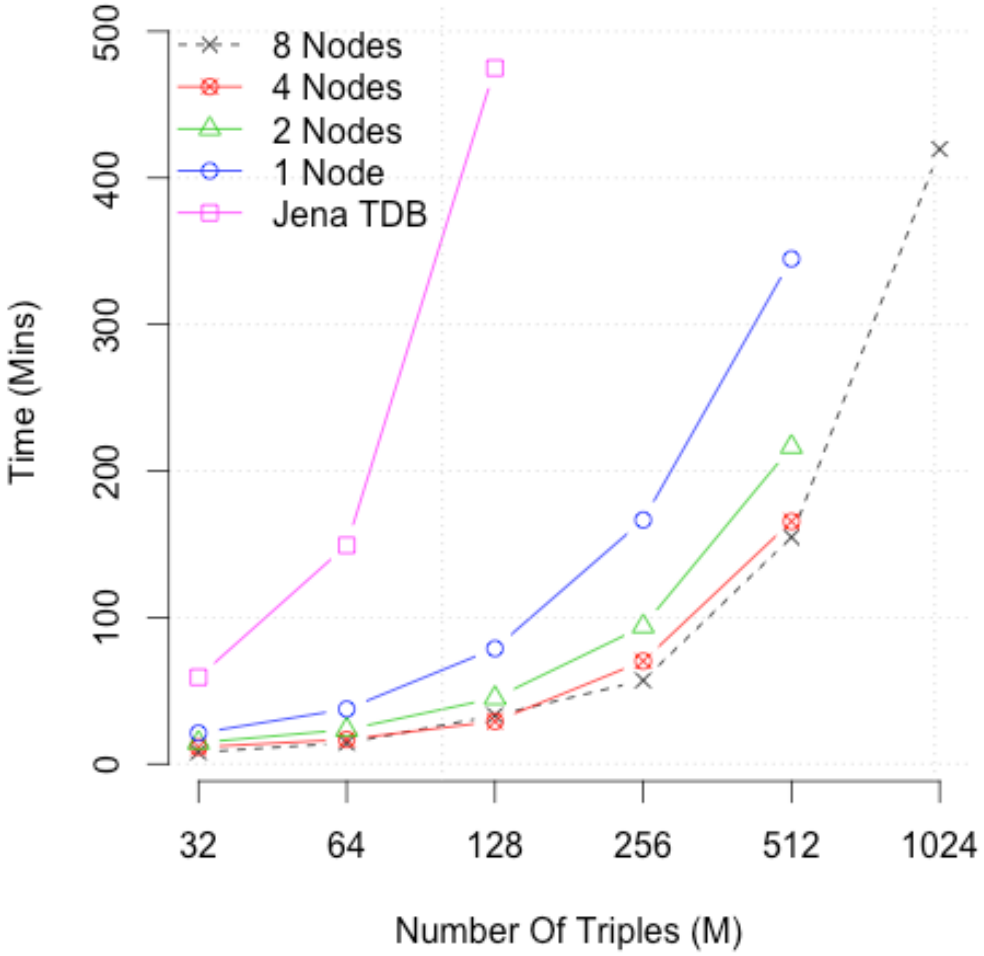
- Increase in runtime not quite linear due to increased network traffic



Experimental Evaluation and Results

- This Figure shows how our new approach scales across a range of cluster sizes. The original Jena implementation is included for comparison.

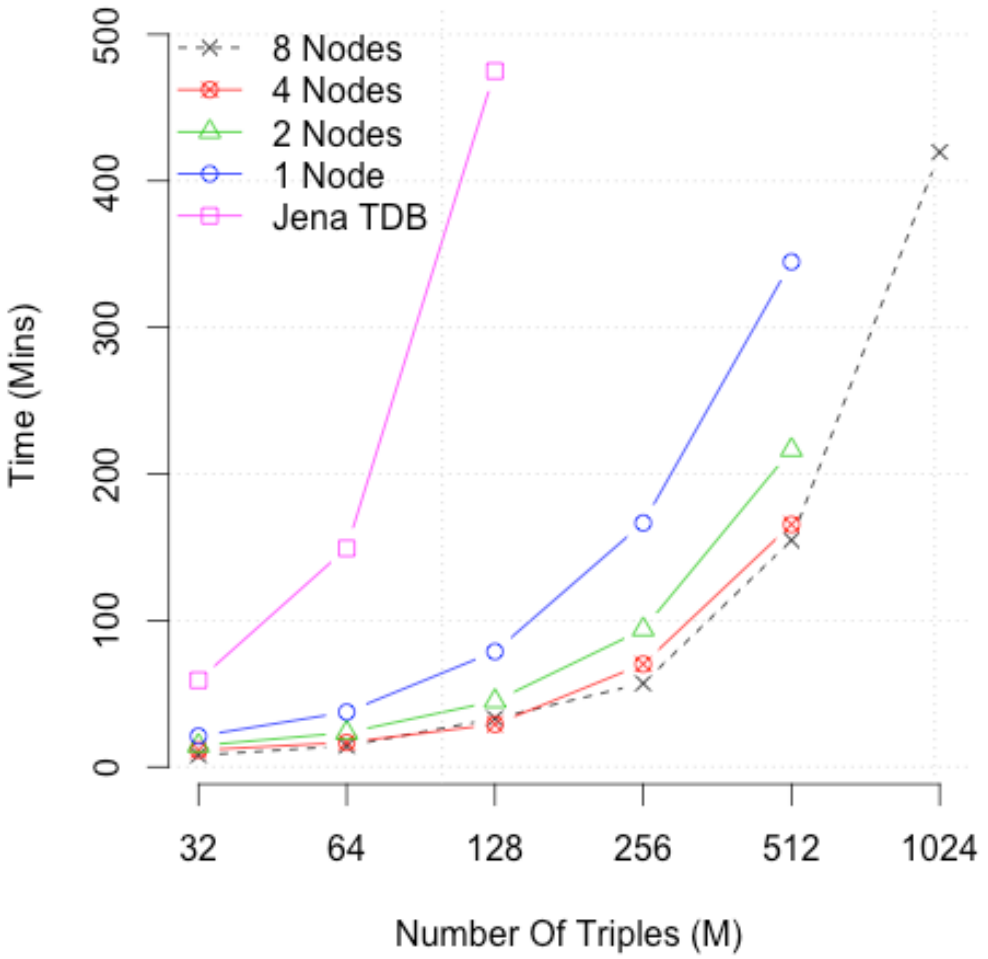
- Note that the pink line is the previous approach. Jena cannot scale past 128M triples.
- Crucially, this new approach demonstrates a five times speedup over the Jenna approach.
- We expect on better hardware that an increase in cluster size would result in better performance. Hadoop is severally network constrained.



Experimental Evaluation and Results

- **What does this mean in terms of patient record numbers?**

- 3 million RDF triples is approximately one patients records for one day.
- Jena implementation only able to process 43 patient day records.
- Our approach, using the 8 node cluster, can process 341 patient days in the less time.



Conclusions and Further Work

- **Conclusions:**
- This work has presented a novel methodology of processing clinical care medical datasets, in linked data format, to assess for accuracy and validity.
- With medical scientists increasing their research into predictive and prescriptive modelling it is of upmost importance that the available datasets are accurate and error free.
- However, it is also important that these datasets can be made available in a timely manner – especially significant for diagnosis.
- The two key optimisations are the creation of the super query and the identification and distribution of triple groups.
- Crucially, this new approach demonstrates a five times speedup over the Jenna approach.

Conclusions and Further Work

- **Further Work:**
- Hope to start deploying this software in hospitals in Scotland. Need to make it more user friendly.
- Currently this work has been tailored specifically to process medical RDF datasets and pre-determined SPARQL queries.
- Future research is needed to investigate the possibility of creating a generic framework for processing RDF datasets via Hadoop.
- The key aspect of this generic framework would be that the optimisations explored here would be automatically performed, possibly by a machine learning based approach.

- **Thanks for listening and any Questions?**

Theoretical Model

- **Triple Group Creation -** We formally define triple group TGs as:

$$TG_n = \{(v, *, *) \in J \cup (*, *, v) \in J : |\Phi_v| > 1, v \in V\},$$

- where TGn is a subgraph of the complete SPARQL query, J is the set of all Base Graph Patterns within the WHERE clause of the SPARQL query, V is the set of all variables found within the WHERE clause, (v,$*$,$*$) denotes that v is the subject of a BGP, likewise ($*$,$*$,v) implies v is the object, and Φ is defined as:

$$\Phi_v = \{v \in V : (v, *, *) \in J \ , \ (*, *, v) \in J\},$$

- i.e. the set of all BGP's which contain v.
- Once we have split the complete dataset into triple groups, we can push the smaller ones in a special feature of Hadoop called the distributed cache. This allows a set of data to be pushed to all the nodes in the cluster.