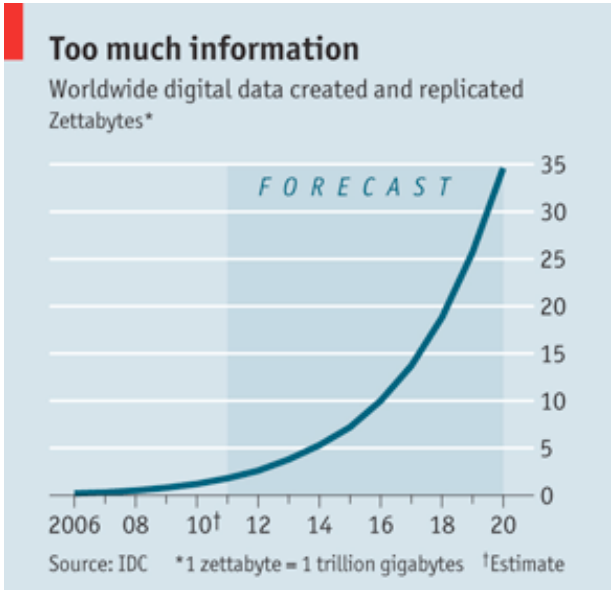
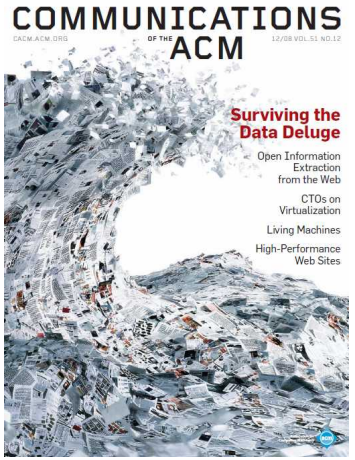
Big Data Mining Services and Knowledge Discovery Applications on Clouds

Domenico Talia
DIMES, Università della Calabria & DtoK Lab
Italy
talia@dimes.unical.it

Data Availability or Data Deluge?

- Some decades ago the main problem was the **shortage of information**, now the challenge is
  - the **very large volume of information** to deal with and
  - the **associated complexity** to process it and to extract significant and useful parts or summaries.





Talk outline

- Big problems and Big data
- Using Clouds for data mining
- A collection of services for scalable data analysis
- Data mining workflows
- ***Data Mining Cloud Framework*** *(**DMCF**)*
- ***JS4Cloud*** for programming service-oriented workflows.
- Final comments.

Complex Big Problems …

- ✓ Big and complex problems need to be solved by Cloud, HPC systems, and large scale distributed computing systems.
- ✓ We need also **scalable** algorithms, techniques, and systems.

… and Big Data

- ✓ Data sources are larger and larger and ubiquitous (Web, sensor networks, mobile devices, telescopes, social media, bio labs, large scientific instruments, …).

- ✓ Large data sources in many fields **cannot be read by humans**

so

- ✓ The huge amount of data available today requires smart data analysys techniques to help people to deal with it.

Data Analysis

- ... Although storing data is a big issue today, a vital issue is analyse, mine, and process data for **making it useful**.
- We are interested in the **value of data**.

Distributed Data Intensive Apps

- In this scenario, Cloud computing systems provide an effective **computational** and **data storage support.**

- Clouds (and also HPC systems, Manycore systems) allow for running **distributed data intensive applications** and **data mining in large and distributed data sets**.

- Clouds can be **used in integrated platforms** through **service interfaces for manage large data sources and process them.**

Goals (1)

- **KDD** and **data mining** techniques are used in many domains to extract useful knowledge from big datasets.

- **KDD** applications range from
  - Single-task applications
  - Parameter-sweeping applications/ regular parallel applications
  - Complex applications (Workflow-based, distributed, parallel).

- **Cloud Computing** can be used to provide developers and end-users with computing and storage services and scalable execution mechanisms **needed to efficiently run all these classes of applications.**

Goals (2)

- Using Cloud services for **scalable execution of data analysis workflows.**
- Defining a programming environment for data analysis: ***Data Mining Cloud Framework (DMCF).***
- Implementing a visual programming interface and the script-based ***JS4Cloud*** language for implementing service-oriented workflows.
- Evaluating the performance of data mining workflows on **DMCF.**

Data analysis as a service

- **PaaS** (*Platform as a Service*) can be an appropriate model to build frameworks that allow users to design and execute data mining applications.

- **SaaS** (*Software as a Service*) can be an appropriate model to implement scalable data mining applications.

- Those two cloud service models can be effectively exploited for delivering **data analysis tools and applications as services**.

Service-Oriented Data Mining

- Knowledge discovery (KDD) and data mining (DM) are:
  - **Compute- and data-intensive processes/tasks**
  - **Often based on distribution of data, algorithms, and users.**
- Large scale service-oriented systems (like Clouds) can integrate both distributed computing and parallel computing, thus they are useful platforms.
- They also offer
  - **security, resource information, data access and management, communication, scheduling, SLAs, …**

Services for Distributed Data Mining

- By exploiting the SOA model it is possible to define **basic services for supporting distributed data mining tasks/ applications.**

- Those services can address all the aspects of data mining and in knowledge discovery processes
  - **data selection and transport services,**
  - **data analysis services,**
  - **knowledge models representation services, and**
  - **knowledge visualization services.**

Services for Distributed Data Mining

- It is possible to design services corresponding to

Data Mining Applications and KDD processes

This level includes the previous tasks and patterns composed in **multi-step workflows**.

Distributed Data Mining Patterns

This level implements, as services, patterns such as **collective learning**, **parallel classification** and **meta-learning** models.

Single Data Mining Tasks

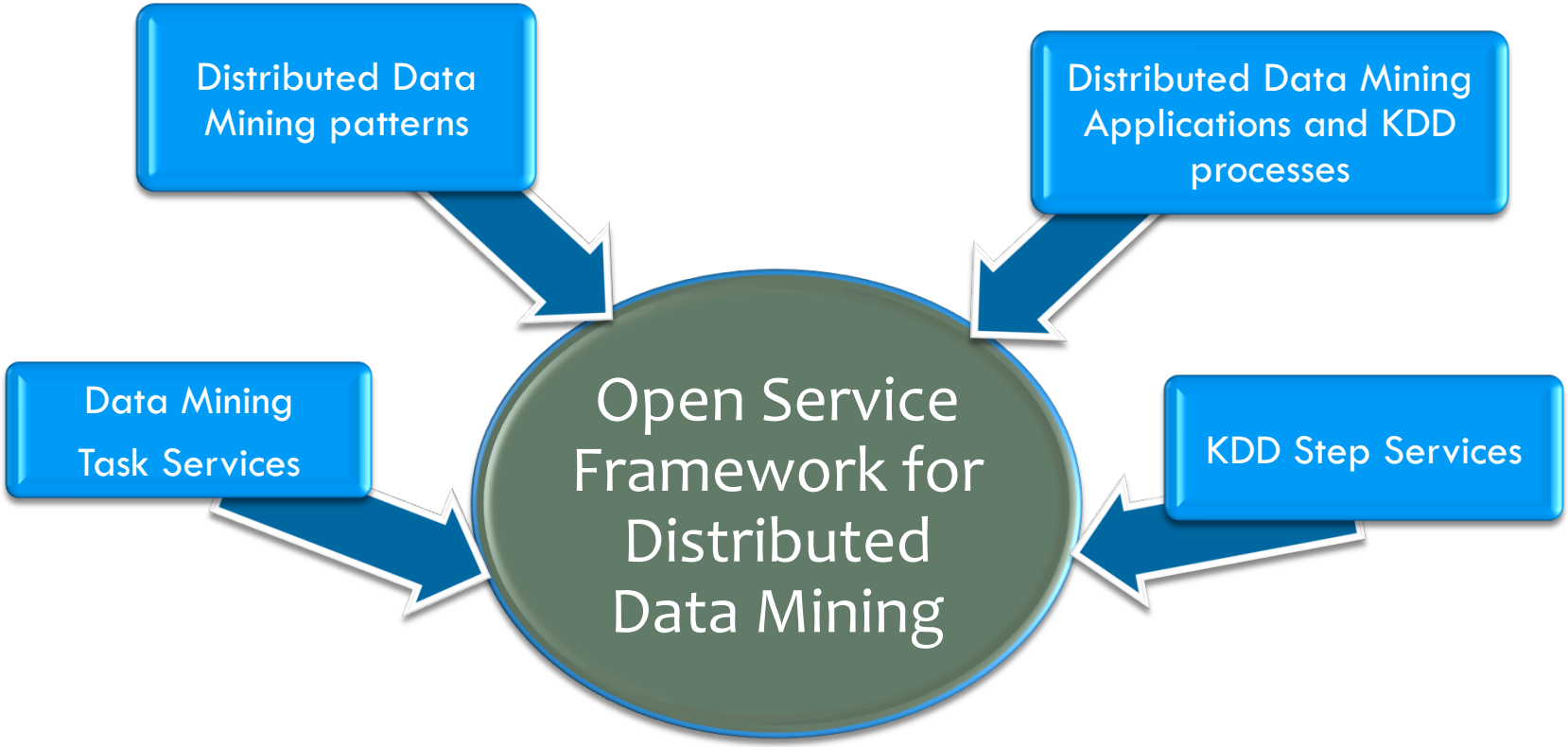
Here are included tasks such as **classification**, **clustering**, and **association rules discovery**.

Single KDD Steps

All steps that compose a KDD process such as **preprocessing**, **filtering**, and **visualization** are expressed as services.

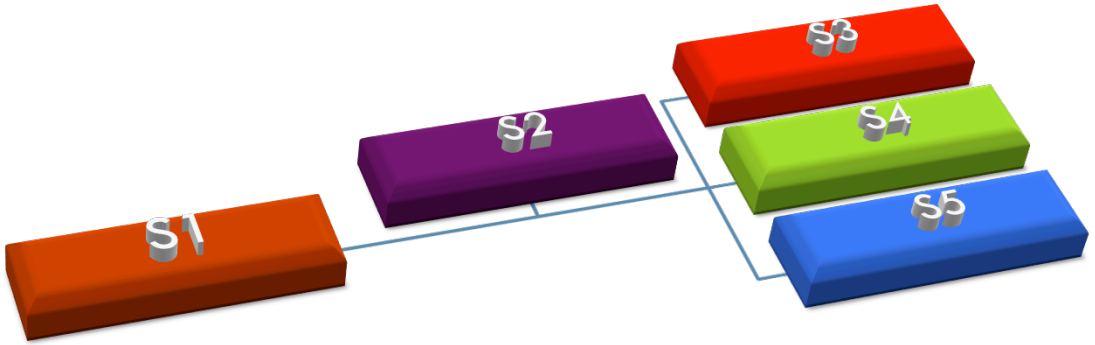
Services for Distributed Data Mining

- This collection of data mining services implements an



Services for Distributed Data Mining

- It allows developers to program distributed KDD processes as a **composition of single and/or aggregated services** available over a service-oriented infrastructure.



A Service-oriented Cloud workflow

- Those services should exploit other basic Cloud/Web services for data transfer, replica management, data integration and querying.

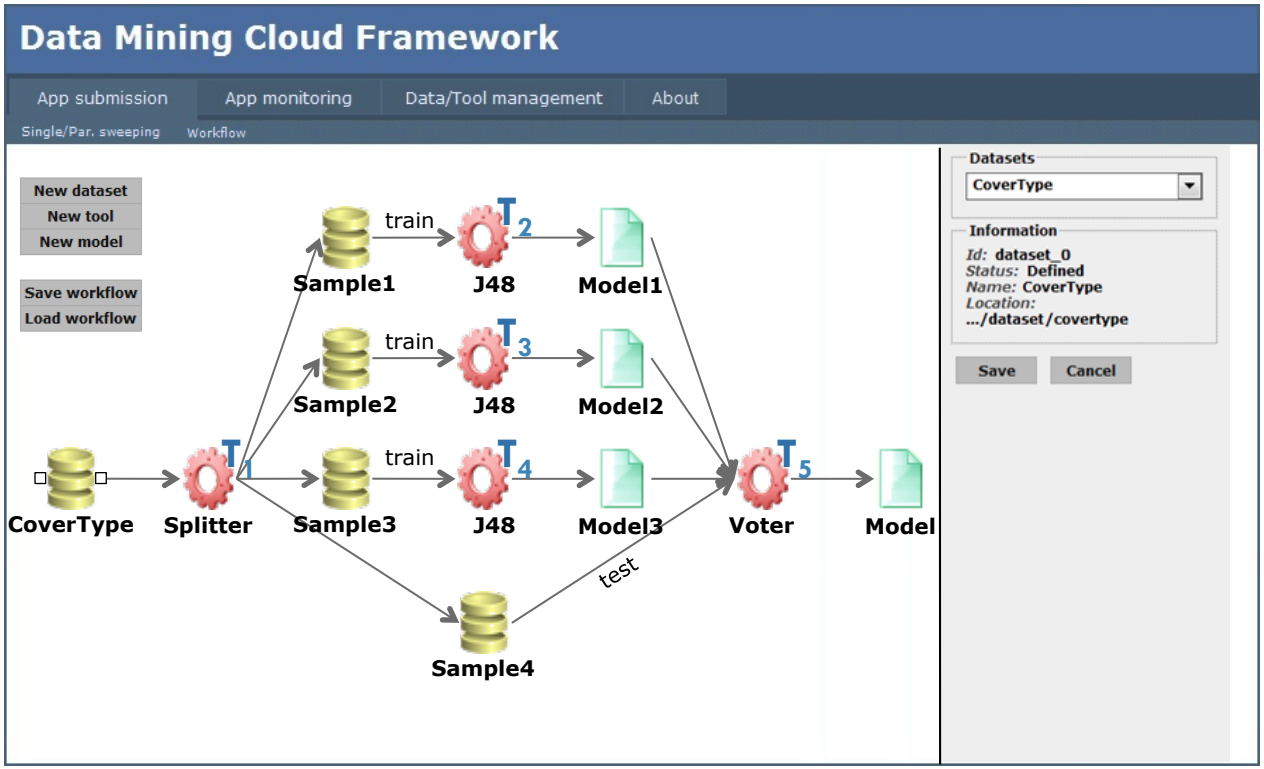
Services for Distributed Data Mining

- By exploiting the Cloud services features it is possible to develop **data mining services accessible every time and everywhere** (remotely and from small devices).

- This approach can produce not only service-based distributed data mining applications, but also
  - **Data mining services for communities**/virtual organizations.
  - Distributed **data analysis services on demand**.
  - A sort of **knowledge discovery eco-system** formed of a large numbers of decentralized data analysis services.

The Data Mining Cloud Framework

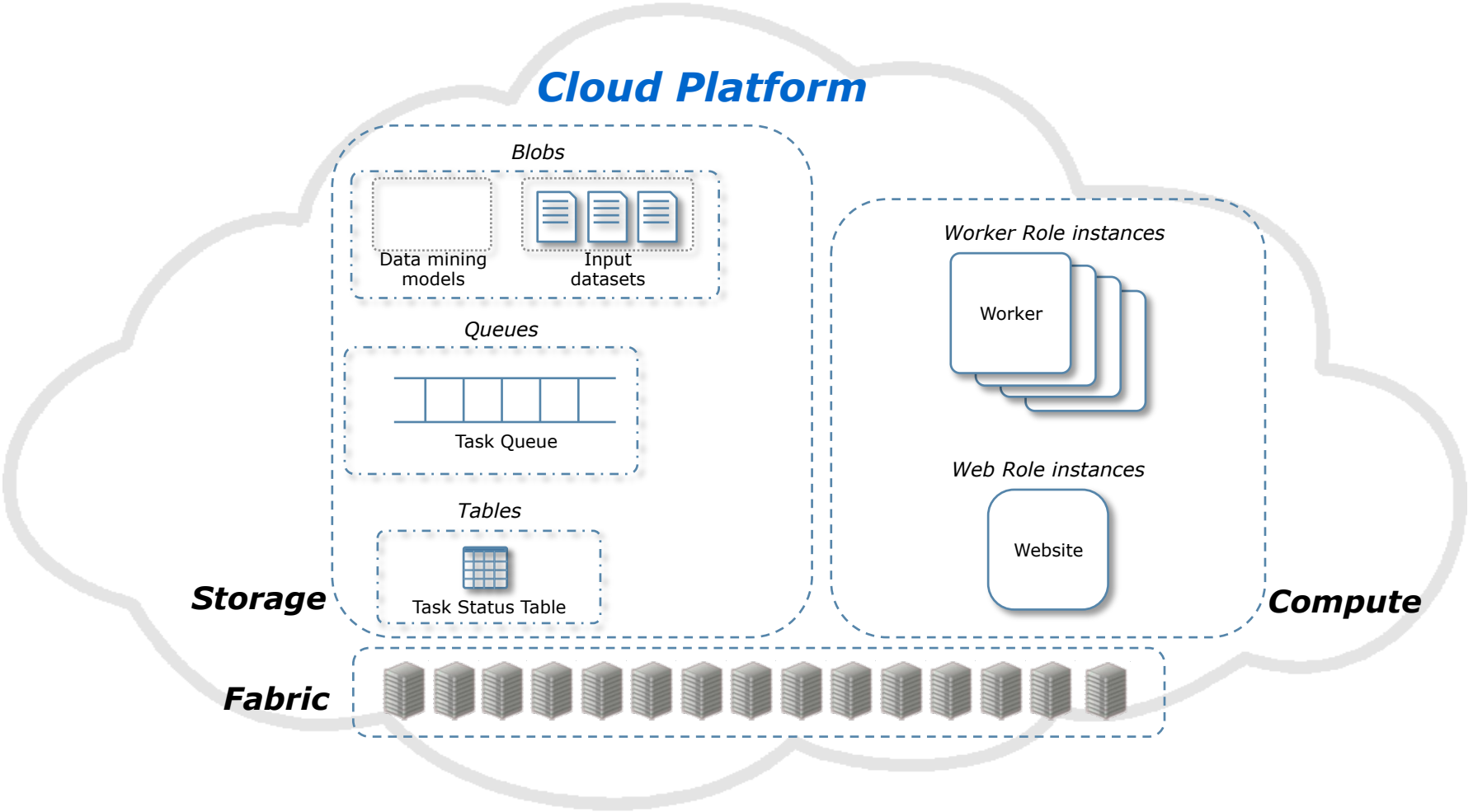
- **Data Mining Cloud Framework** supports *workflow-based KDD applications*, expressed (visually and by a language) as a graphs that link together data sources, data mining algorithms, and visualization tools.



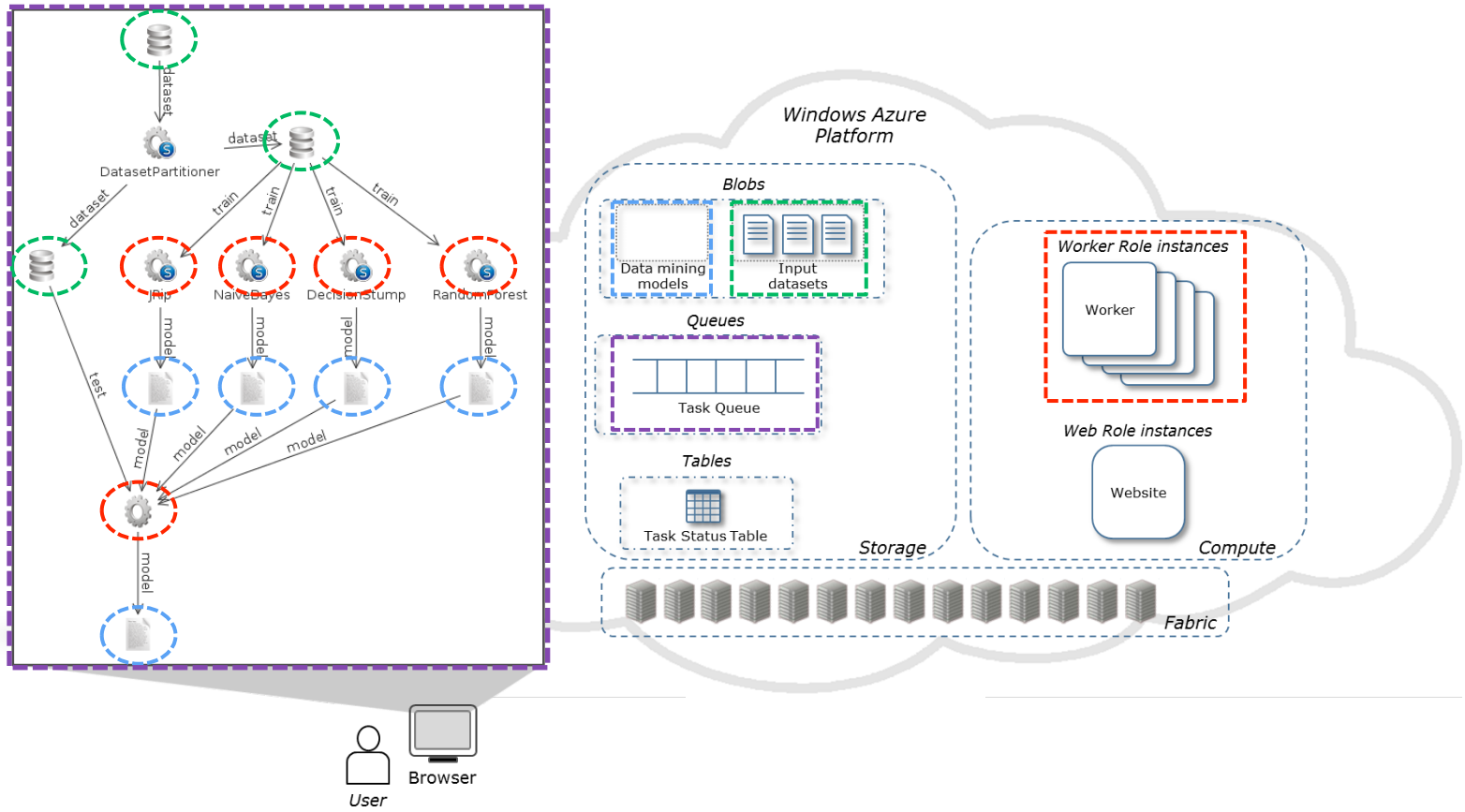
Architecture Components

- **Compute** is the computational environment to execute Cloud applications:
  - *Web role*: Web-based applications.
  - *Worker role*: batch applications.
  - *VM role*: virtual machine images.
- **Storage** provides scalable storage elements:
  - *Blobs*: storing binary and text data.
  - *Tables*: non-relational databases.
  - *Queues*: communication between components.
- **Fabric controller** links the physical machines of a single data center:
  - *Compute* and *Storage* services are built on top of this component.

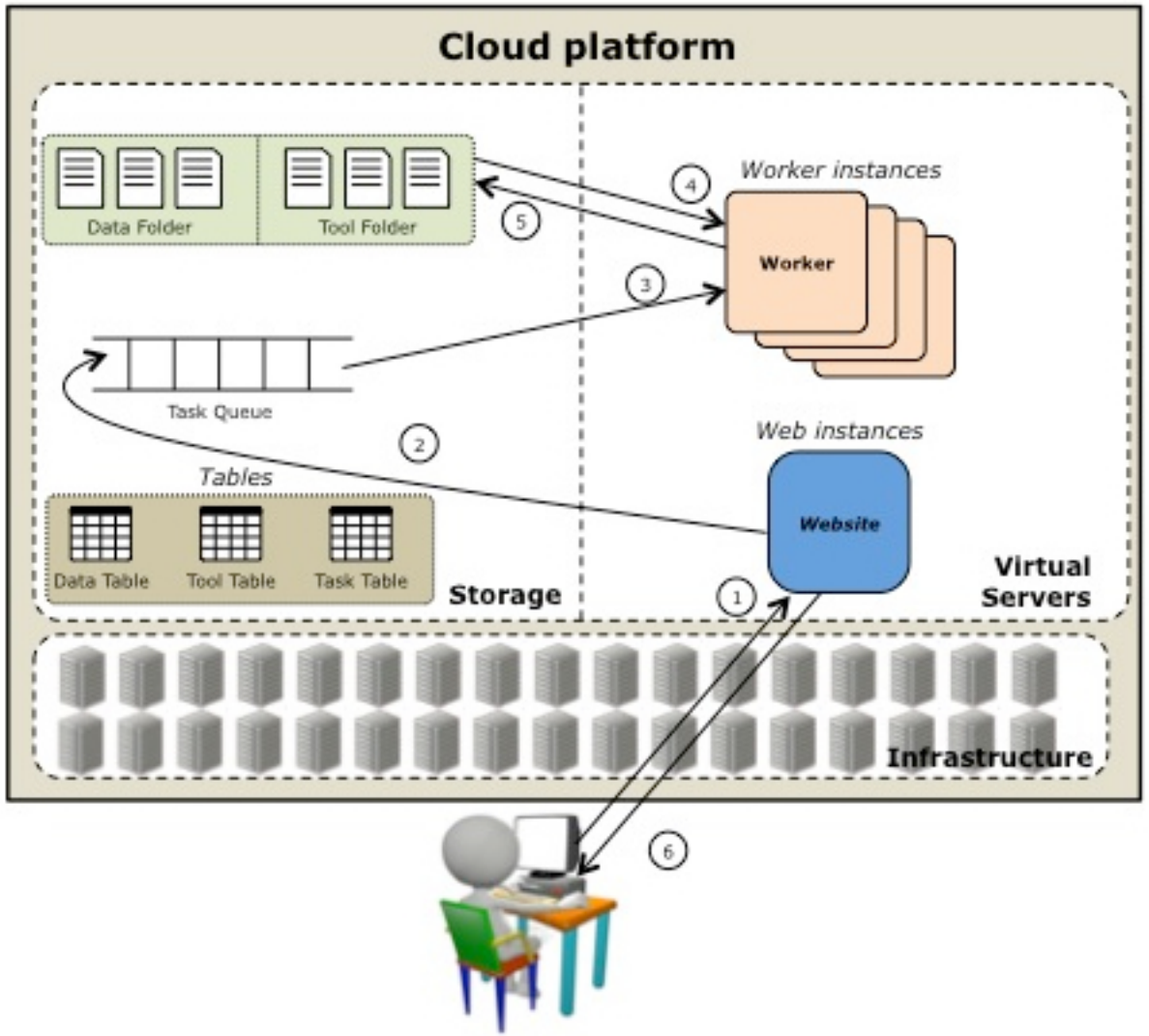
The Data Mining Cloud Framework: Architecture



The Data Mining Cloud Framework - Mapping

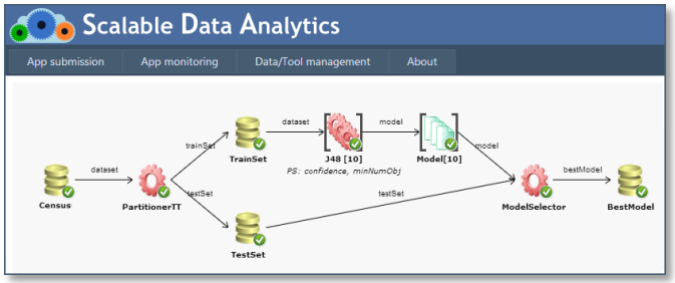


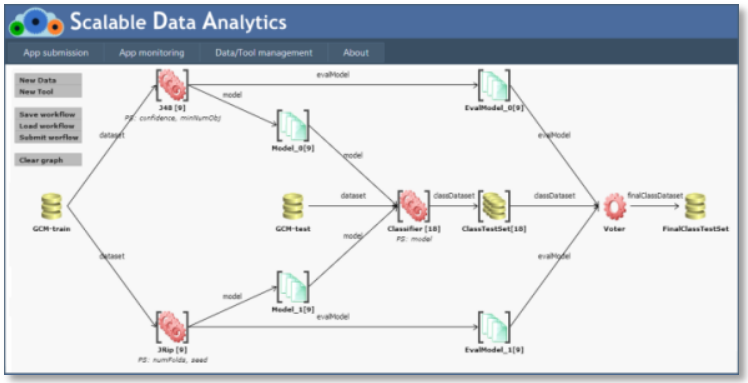
The Data Mining Cloud Framework – Execution



Example applications (1)

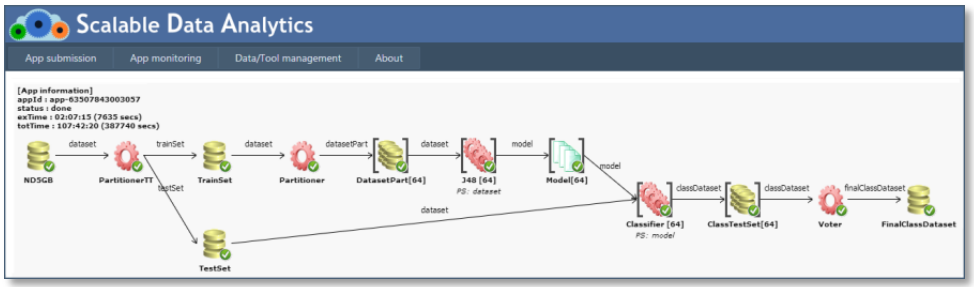
Finance: Prediction of personal income based on census data





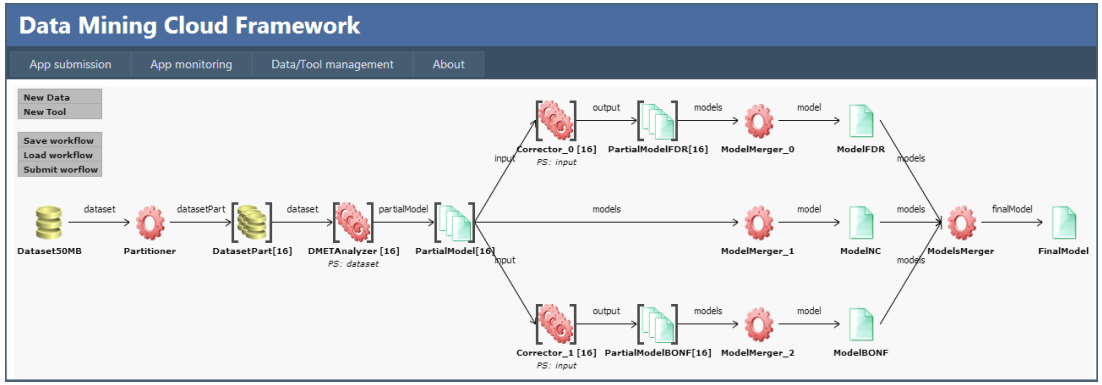
E-Health: Disease classification based on gene analysis

Networks: Discovery of network attacks from log analysis.

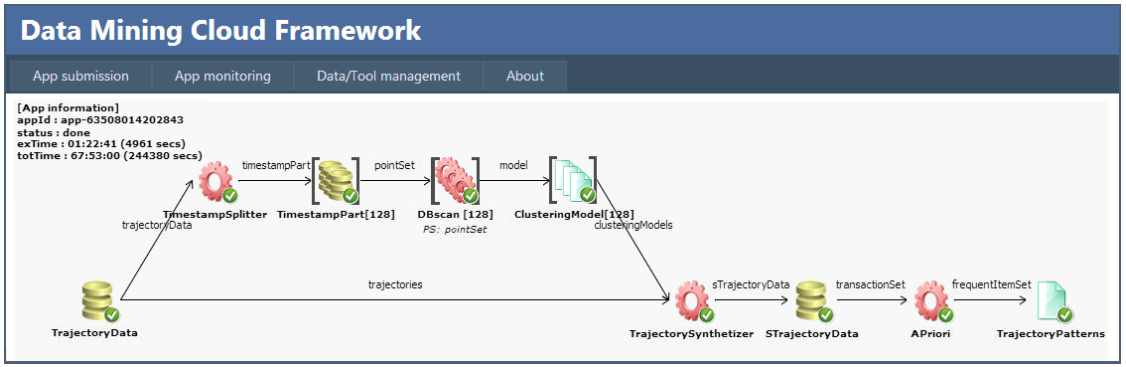


Example applications (2)

Biosciences: drug metabolism associations in pharmacogenomics.

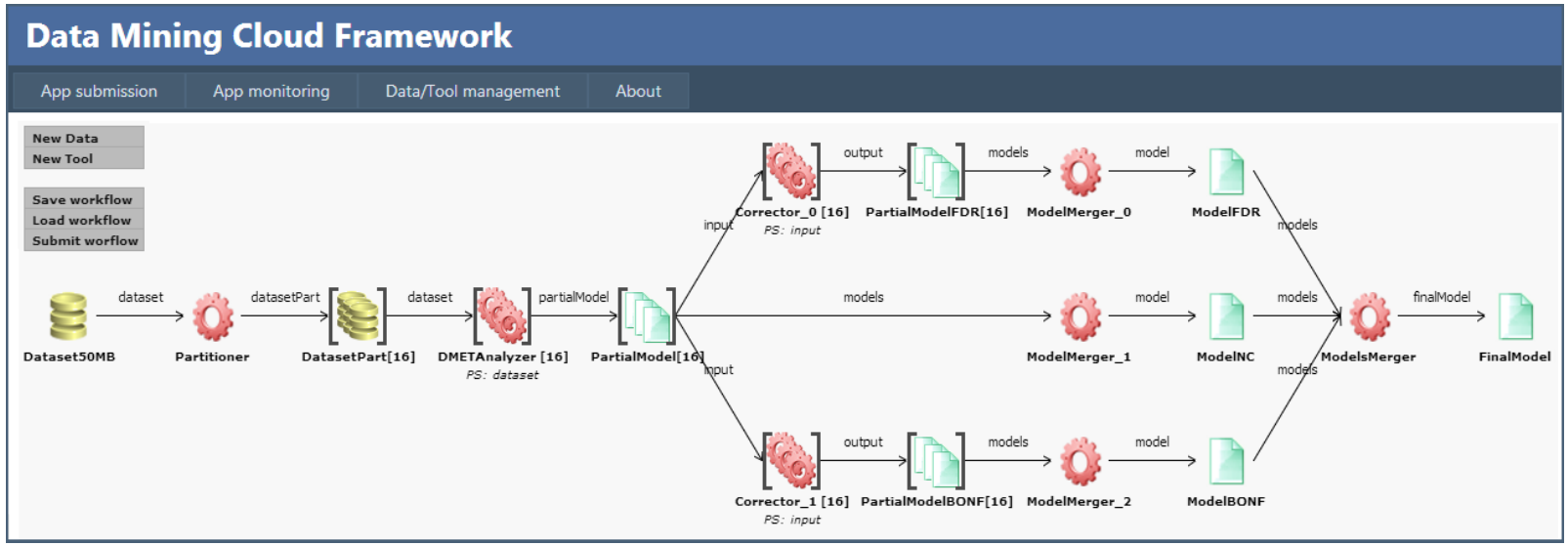


Smart City: Car trajectory pattern detection applications.

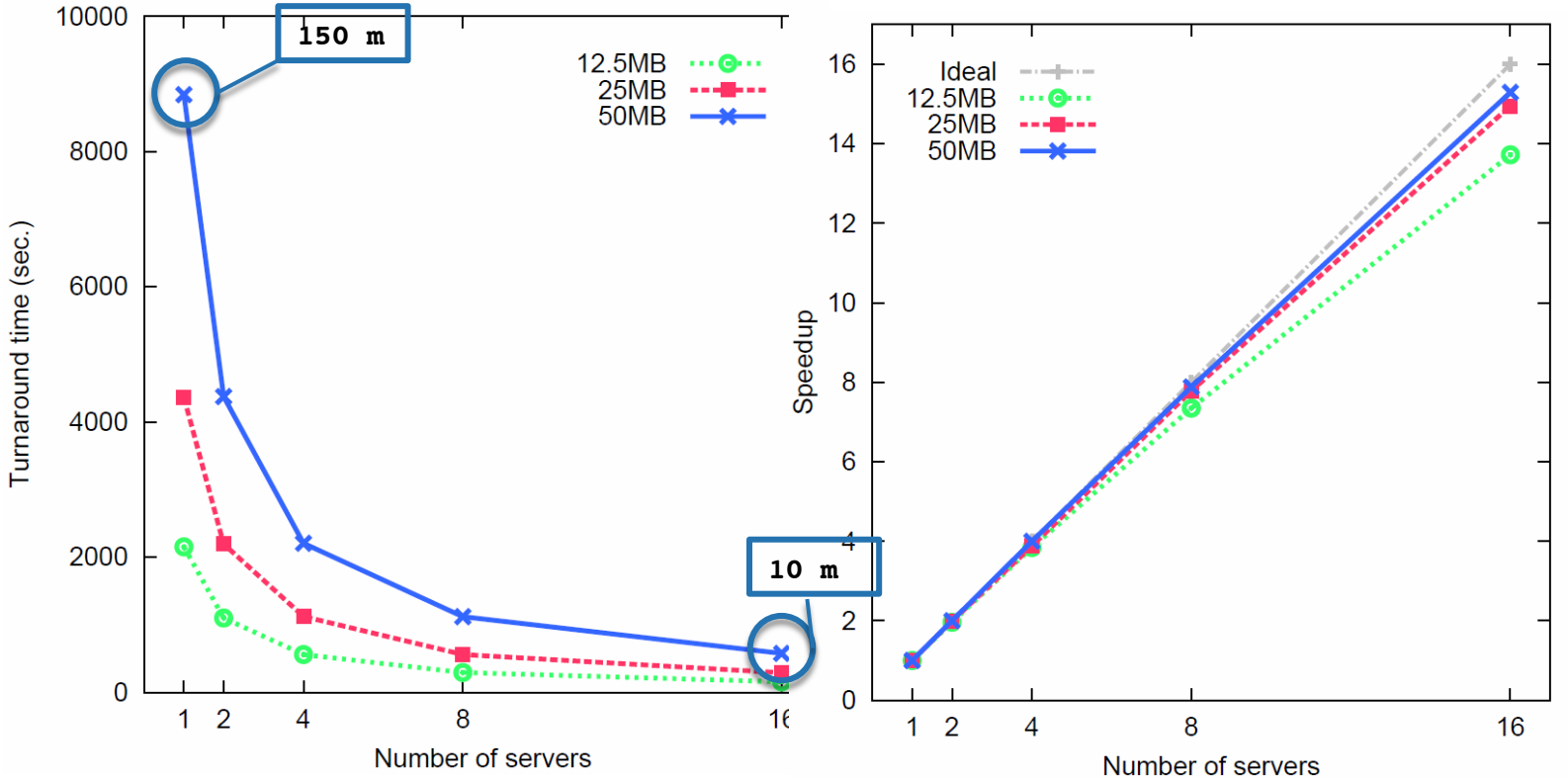


The Cloud4SNP workflow

- **DMET** (Drug Metabolism Enzymes and Transporters) has been designed specifically to test drug metabolism associations in pharmacogenomics case-control study.
- Cloud4SNP [3] is a Cloud implementation of **DMET** by using the ***DMCF***.



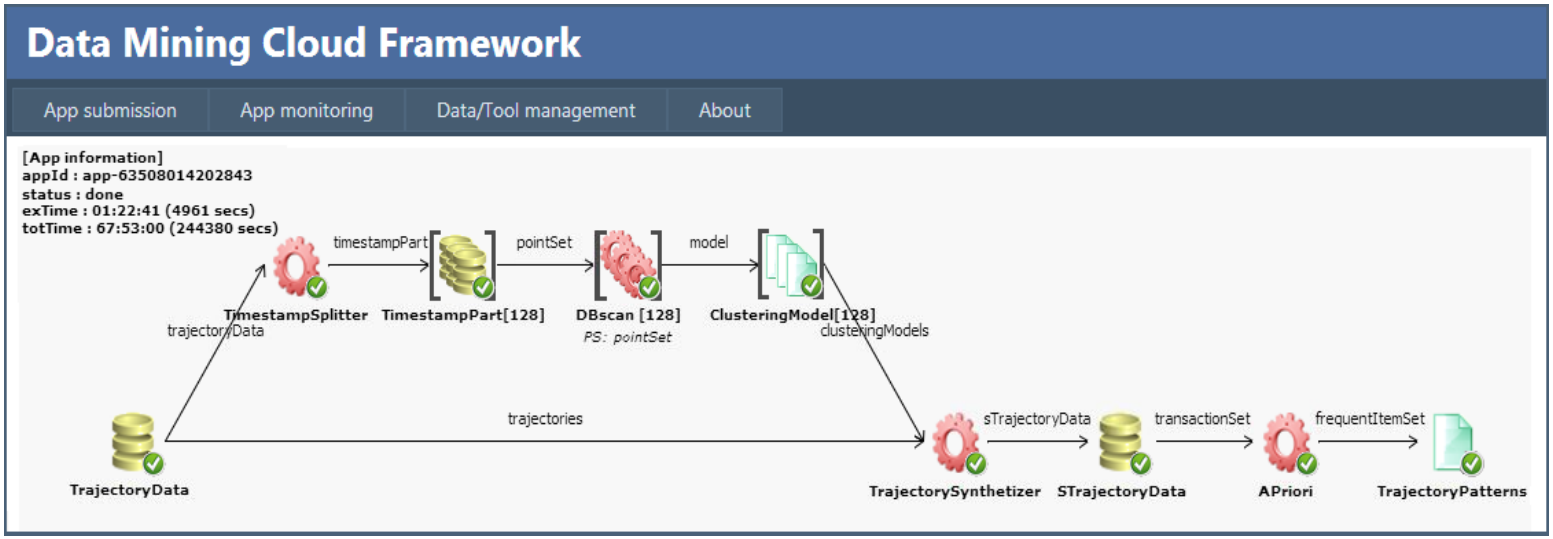
Performance evaluation



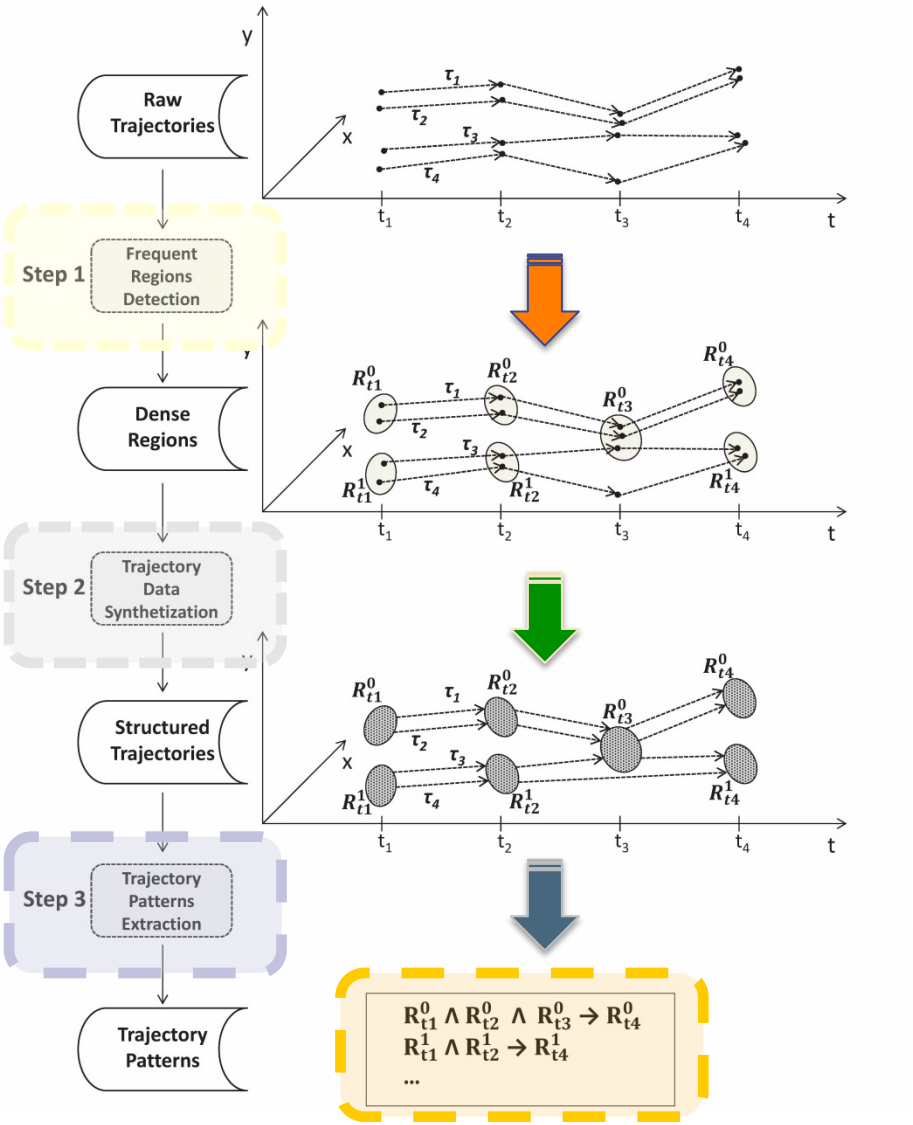
Trajectory Pattern Detection

- Analyze trajectories of mobile users to discover movement patterns and rules.

- A workflow that integrate frequent regions detection, trajectory data synthetization and trajectory pattern extraction.



Application Main Steps



Frequent Regions Detection.

- Detect areas more densely passed through
- Density-based clustering algorithm (DB-Scan)
- Further analysis: movement through areas

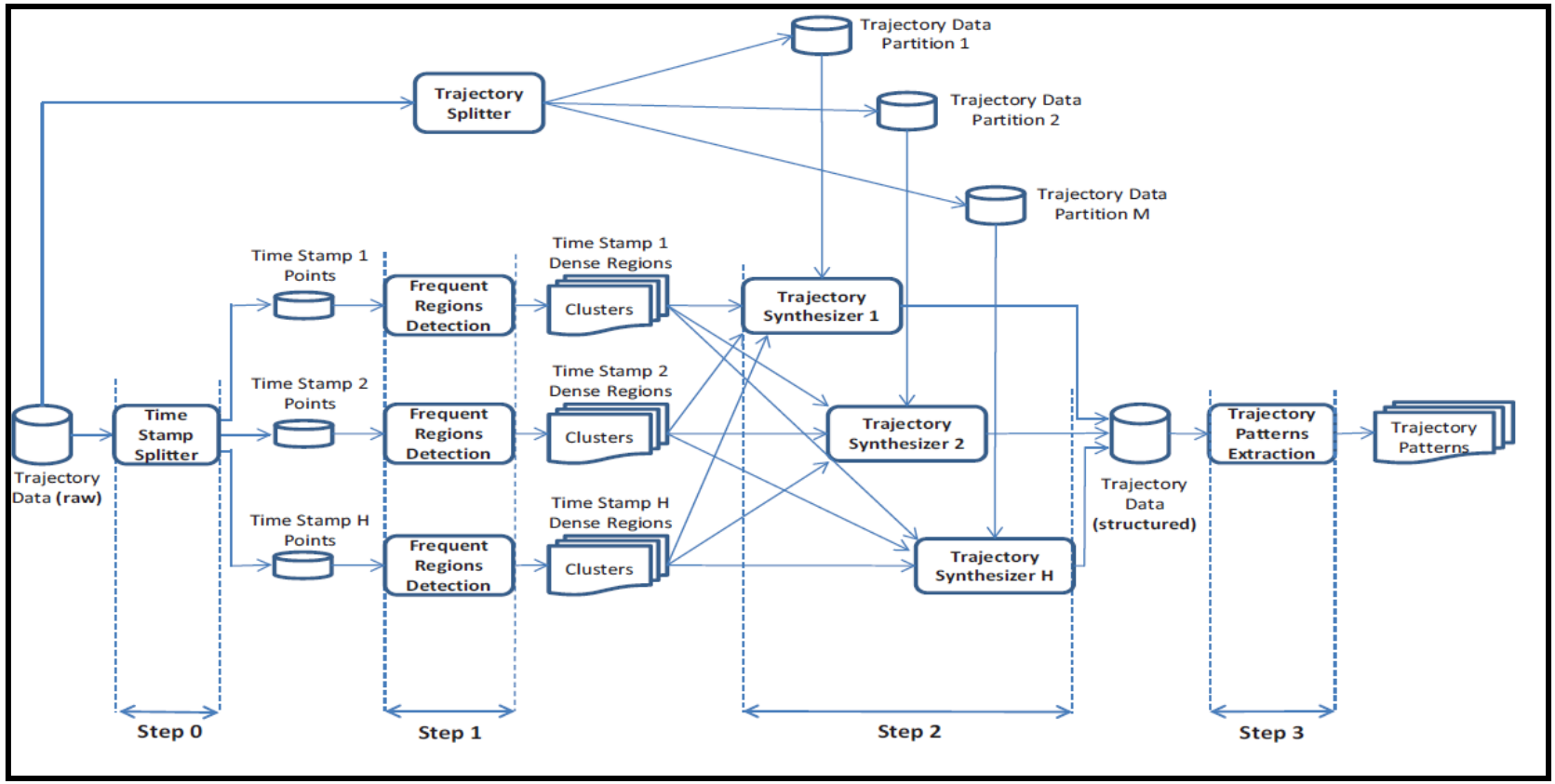
Trajectory Data Synthetization.

- each point is substituted by the dense region it belongs to.
- trajectory representations is changed from movements between points into movements between frequent regions

Trajectory Pattern Extraction.

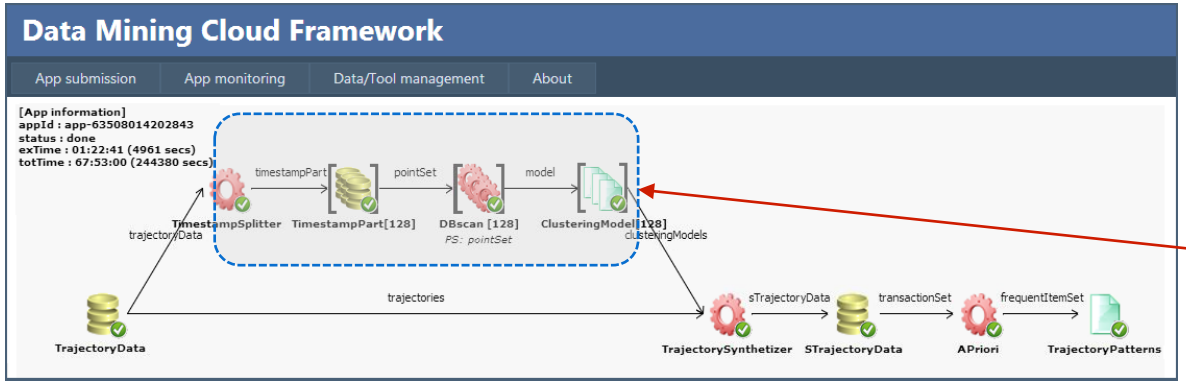
- Discovery of patterns from structured trajectories
- T-Apriori algorithm, i.e. ad-hoc modified version of Apriori

Application Workflow



Workflow Implementation

- Workflow implementing the trajectory pattern detection algorithm
  - Each node represents either a data source or a data mining tool
  - Each edge represents an execution dependency among nodes
  - Some nodes are labeled by the array notation
    - Compact way to represent multiple instances of the same dataset or tool
    - Very usfeul to build complex workflows (data/task parallelism, parameter sweeping, etc.)

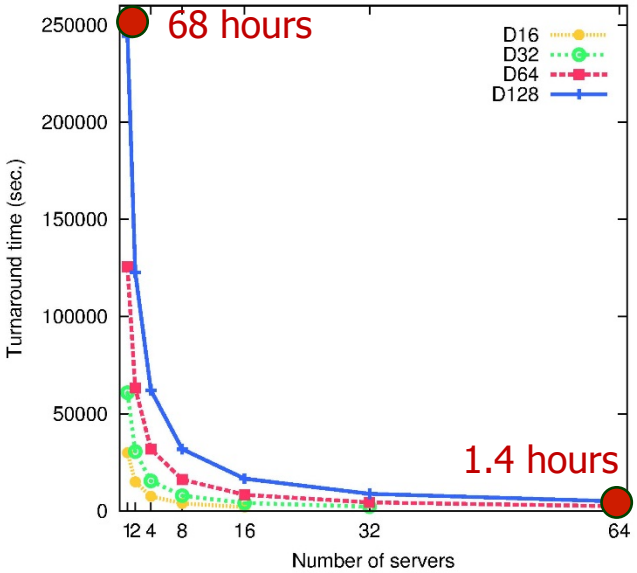


128 parallel tasks !!!

Experimental Evaluation

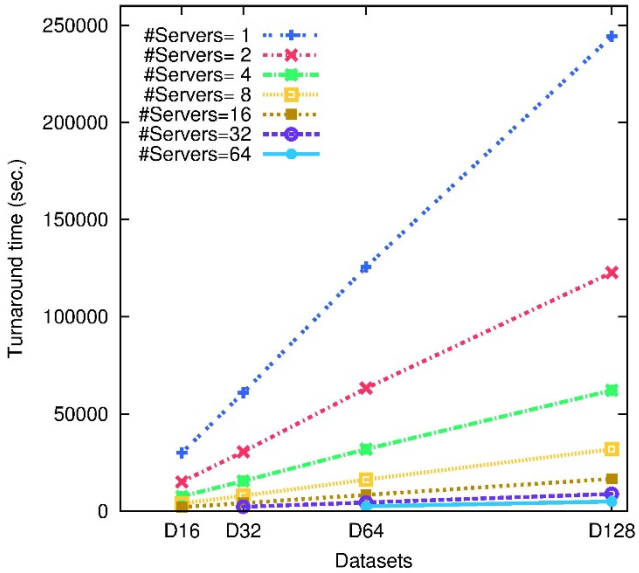
Turnaround time

- vs the number of servers (up to 64), for different data sizes



  - comparison parallel\sequential execution
  - D_{16} (D_{128}): it reduces from 8.3 (68) hours to about 0.5 (1.4) hours

- vs several data sizes (up to 128 timestamps), for different number of servers

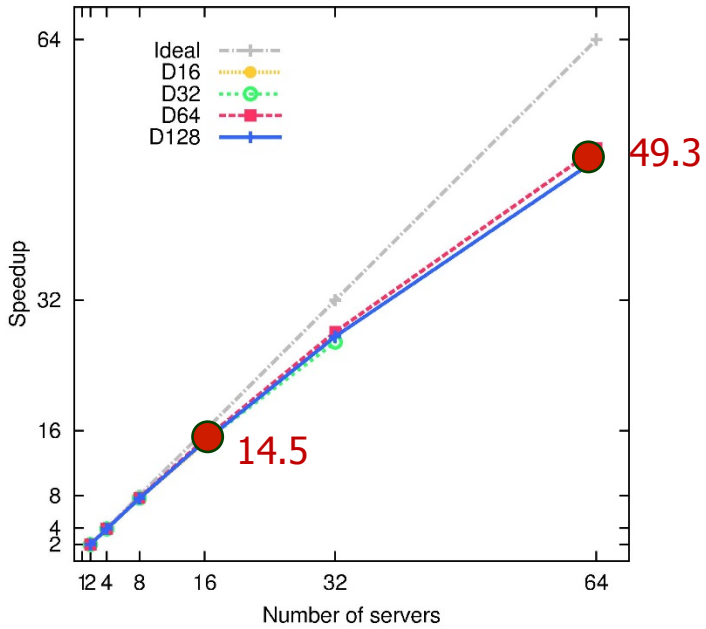


  - it proportionally increases with the the input size
  - it proportionally decreases with the increase of computing resources

Experimental Evaluation

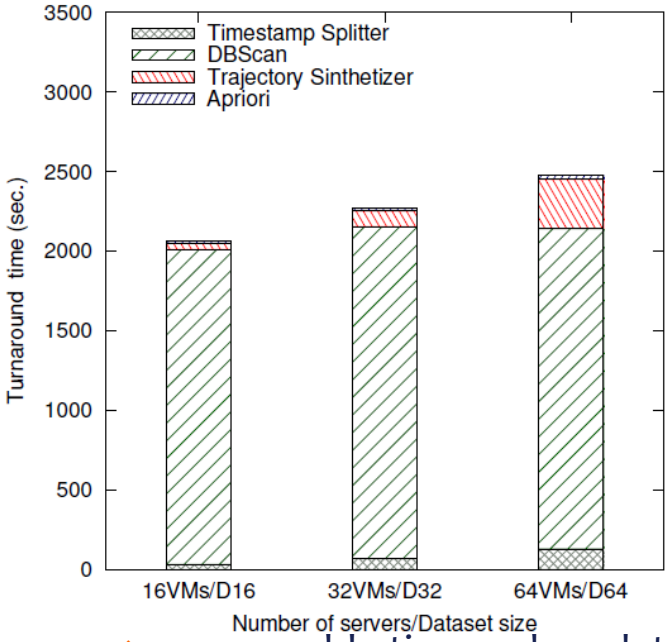
Scalability indicators

speed-up



- notable trend, up to the case of 16 nodes
- good trend for higher number of nodes (influence of the sequential steps)

scale-up



- comparable times when data size and #servers increase proportionally
- DBSCAN step (parallel) takes most of the total time
- other steps (sequential) increases with larger datasets

Script-based workflows

- We extended DMCF adding a ***script-based data analysis programming model*** as a more flexible programming interface.
- Script-based workflows are an effective alternative to graphical programming.
- A script language allows experts to program complex applications more rapidly, in a **more concise** way and with **higher flexibility**.
- The idea is to offer a script-based data analysis language as an **additional and more flexible programming interface** to skilled users.

The JS4Cloud script language

- ***JS4Cloud*** (*JavaScript for Cloud*) is a language for programming data analysis workflows.

- Main benefits of JS4Cloud:
  - it is based on a well known scripting language, so users **do not have to learn a new language** from scratch;
  - it implements a **data-driven task parallelism** that automatically spawns ready-to-run tasks to the available Cloud resources;
  - it exploits **implicit parallelism** so application workflows can be programmed in a totally sequential way (no user duties for work partitioning, synchronization and communication).

JS4Cloud functions

JS4Cloud implements three additional functionalities, implemented by the set of functions:

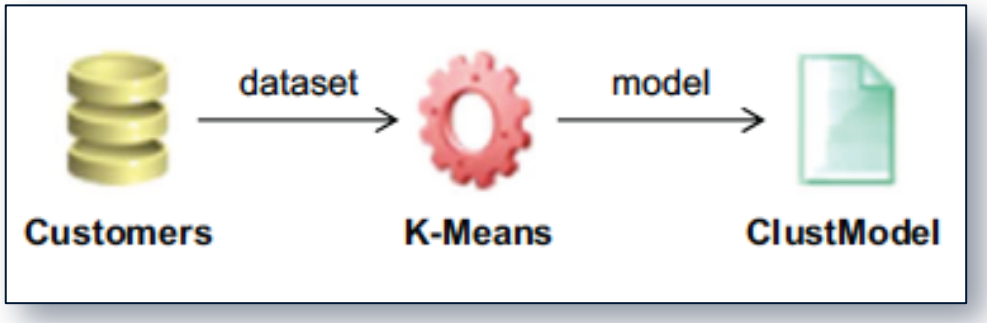
- **`Data.get`**, for accessing one or a collection of datasets stored in the Cloud;
- **`Data.define`**, for defining new data elements that will be created at runtime as a result of a tool execution;
- **`Tool`**, to invoke the execution of a software tool available in the Cloud as a service.

Functionality	*Function*	*Description*
Data Access	Data.get(<*dataName*>);	Returns a reference to the data element with the provided name.
	Data.get(new RegExp(<*regular expression*>));	Returns an array of references to the data elements whose name match the regular expression.
Data Definition	Data.define(<*dataName*>);	Defines a new data element that will be created at runtime.
	Data.define(<*arrayName*>,<*dim*>);	Define an array of data elements.
	Data.define(<*arrayName*>,[<dim_1>,...,<dim_n>]);	Define a multi-dimensional array of data elements.
Tool Execution	<*toolName*>(<par_1>:<val_1>,...,<par_n>:<val_n>);	Invokes an existing tool with associated parameter values.

JS4Cloud patterns

Single task

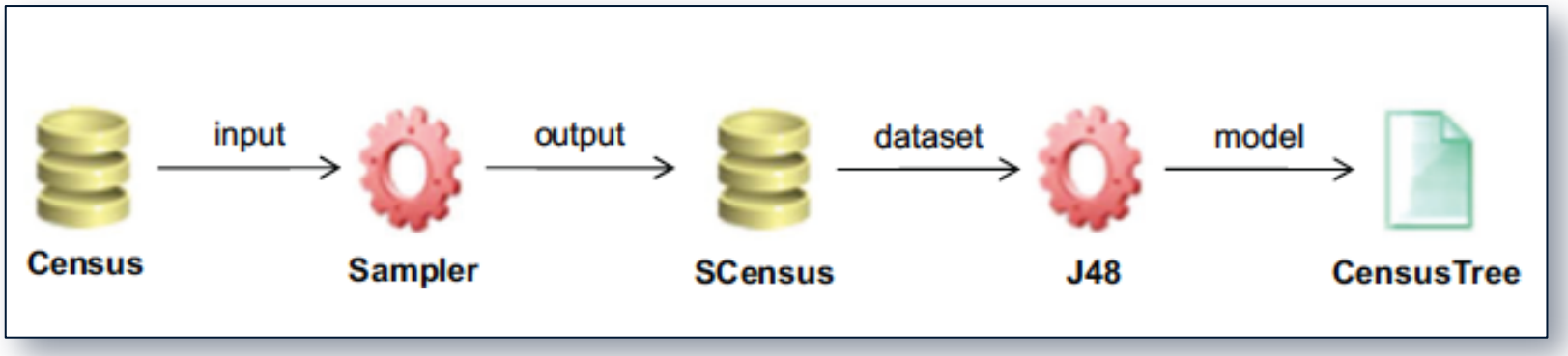
```
var DRef = Data.get("Customers");
var nc = 5;
var MRef = Data.define("ClustModel");
K-Means({dataset:DRef, numClusters:nc, model:MRef});
```



JS4Cloud patterns

Pipeline

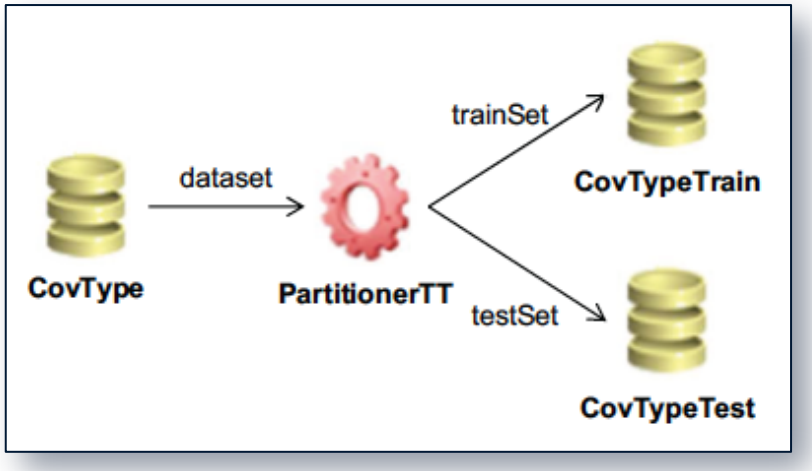
```
var DRef   = Data.get("Census");
var SDRef  = Data.define("SCensus");
Sampler({input:DRef, percent:0.25, output:SDRef});
var MRef   = Data.define("CensusTree");
J48({dataset:SDRef, confidence:0.1, model:MRef});
```



JS4Cloud patterns

Data partitioning

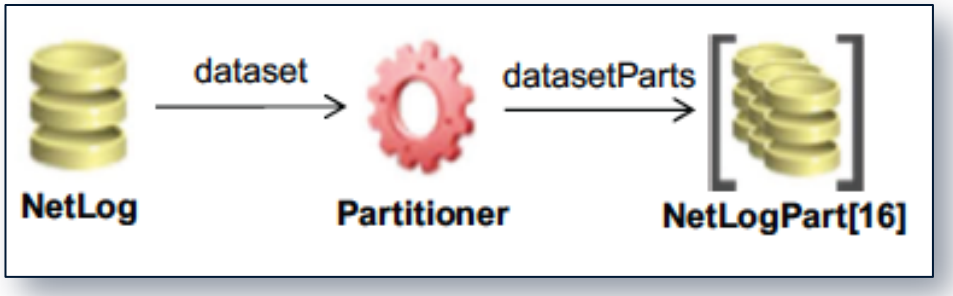
```
var DRef  = Data.get("CovType");
var TrRef = Data.define("CovTypeTrain");
var TeRef = Data.define("CovTypeTest");
PartitionerTT({dataset:DRef, percTrain:0.70,
               trainSet:TrRef, testSet:TeRef});
```



JS4Cloud patterns

Data partitioning

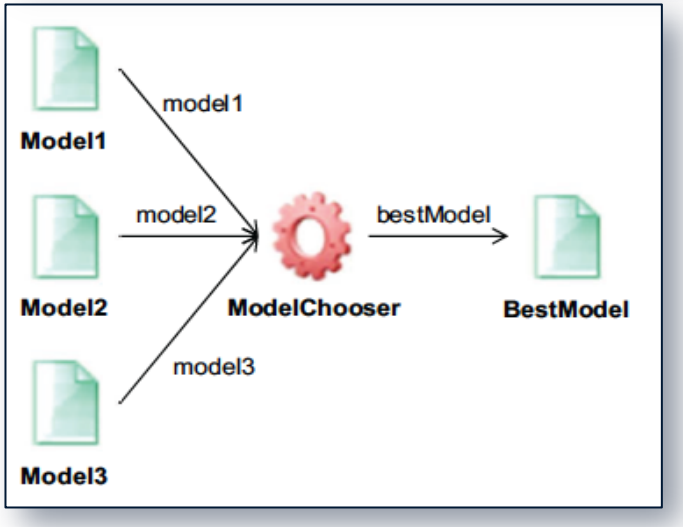
```
var DRef = Data.get("NetLog");
var PRef = Data.define("NetLogParts", 16);
Partitioner({dataset:DRef, datasetParts:PRef});
```



JS4Cloud patterns

Data aggregation

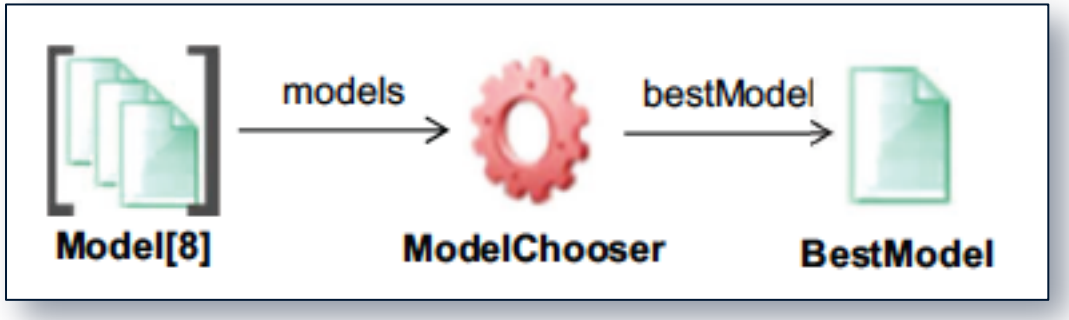
```
var M1Ref = Data.get("Model1");
var M2Ref = Data.get("Model2");
var M3Ref = Data.get("Model3");
var BMRef = Data.define("BestModel");
ModelChooser({model1:M1Ref, model2:M2Ref,
              model3:M3Ref, bestModel:BMRef});
```



JS4Cloud patterns

Data aggregation

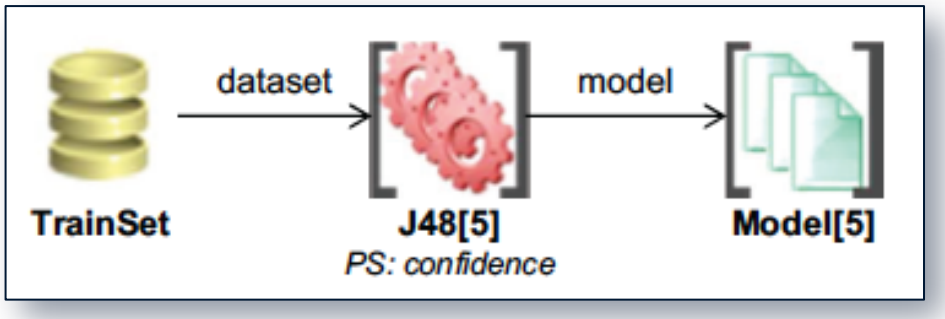
```
var MsRef = Data.get(new RegExp("^Model"));
var BMRef = Data.define("BestModel");
ModelChooser({models:MsRef, bestModel:BMRef});
```



JS4Cloud patterns

Parameter sweeping

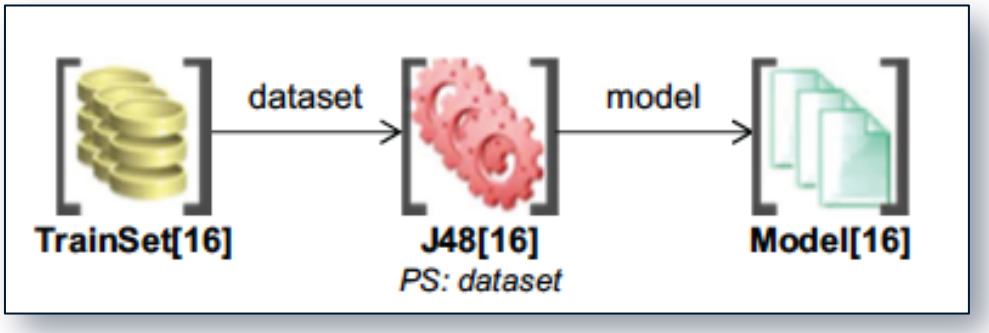
```
var TRef = Data.get("TrainSet");
var nMod = 5;
var MRef = Data.define("Model", nMod);
var min = 0.1;
var max = 0.5;
for(var i=0; i<nMod; i++)
   J48({dataset:TRef, model:MRef[i],
        confidence:(min+i*(max-min)/(nMod-1))});
```



JS4Cloud patterns

Input sweeping

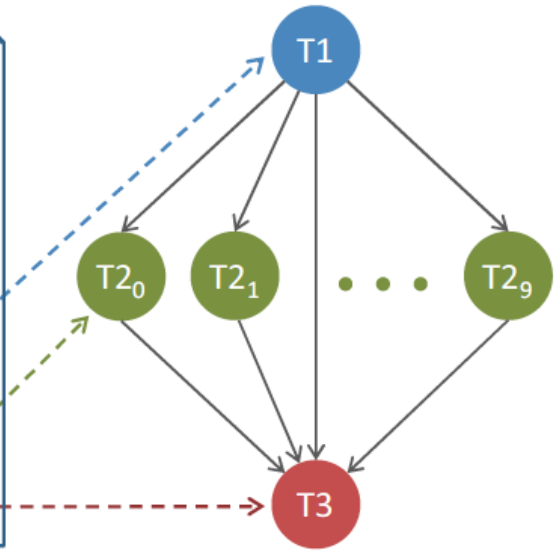
```
var nMod = 16;
var MRef = Data.define("Model", nMod);
for(var i=0; i<nMod; i++)
   J48({dataset:TsRef[i], model:MRef[i],
        confidence:0.1});
```

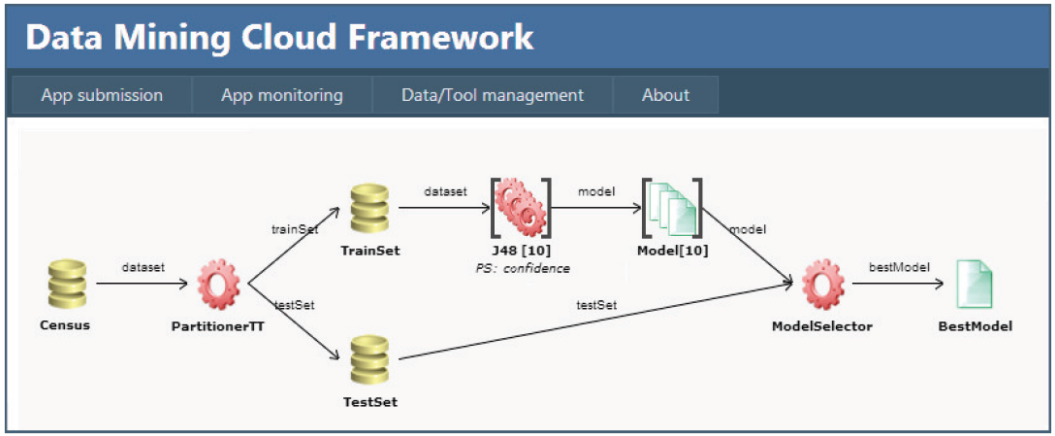


Parallelism exploitation

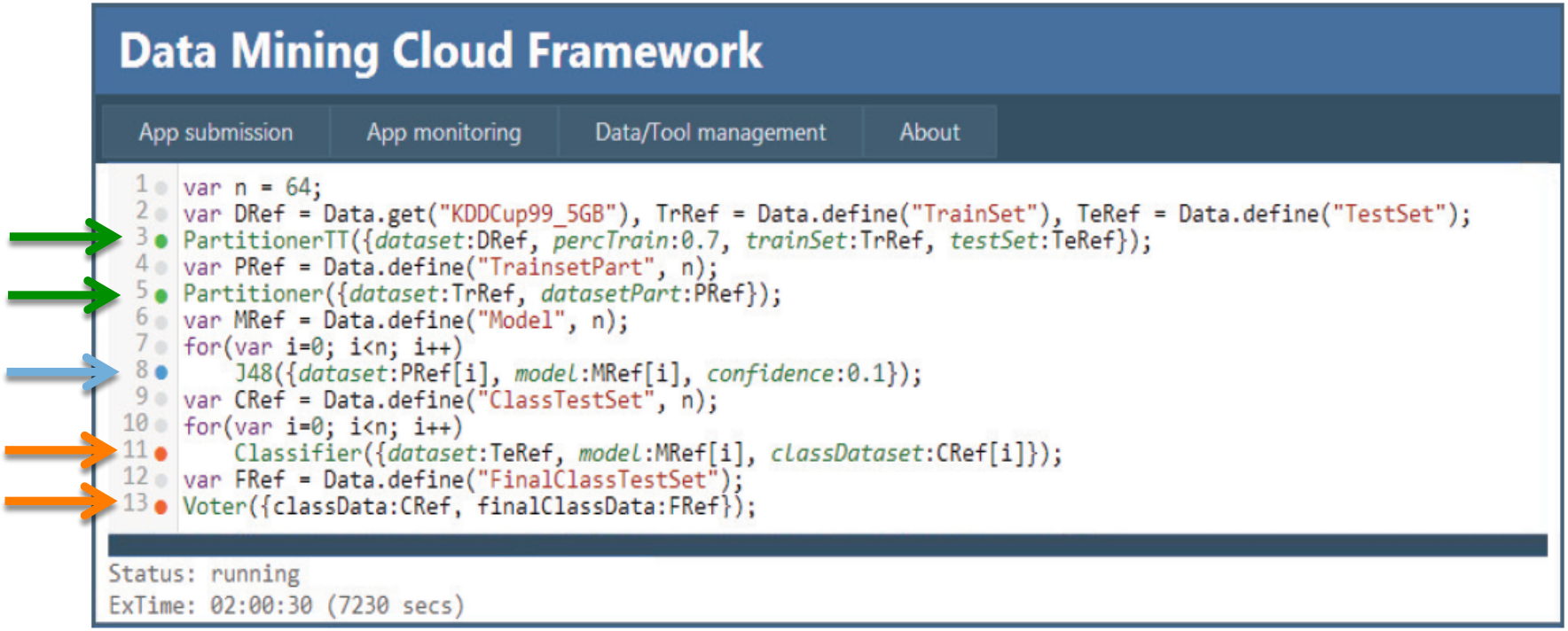
```
var DRef = Data.get("Census");
var TrRef = Data.define("TrainSet");
var TeRef = Data.define("TestSet");
var min = 0.1, max = 0.5; nMod = 10;
var MRef = Data.define("Model", nMod);
var BRef = Data.define("BestModel");

PartitionerTT({dataset:DRef, percTrain:0.70, trainSet:TrRef, testSet:TeRef});
for(int i=0; i<nMod; i++)
   J48({dataset:TrRef, model:Model[i], confidence:(min+i*(max-min)/(nMod-1))});
ModelSelector({testSet:TeRef, model:Model, bestModel:BRef});
```





Monitoring interface

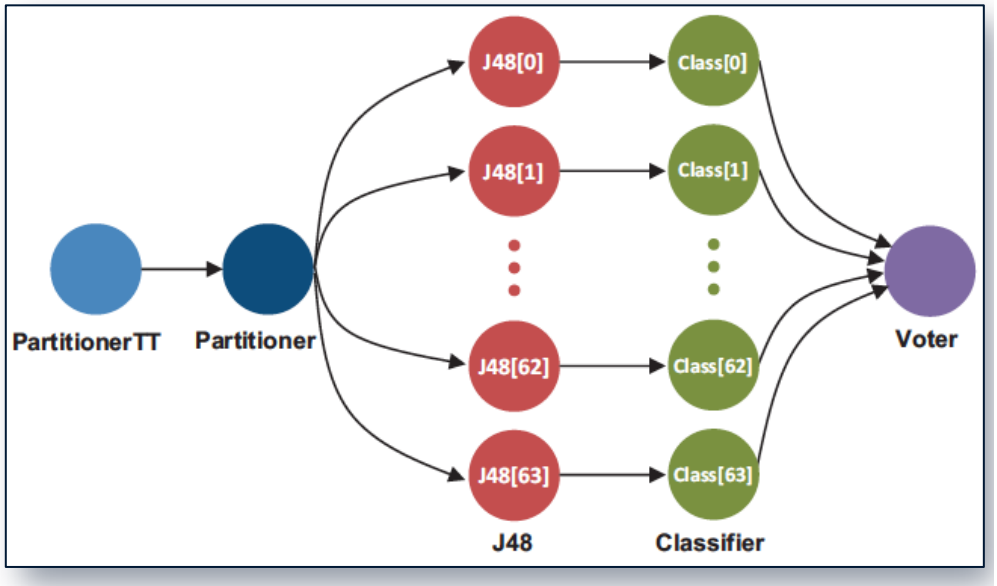


- A snapshot of the application during its execution monitored through the programming interface.

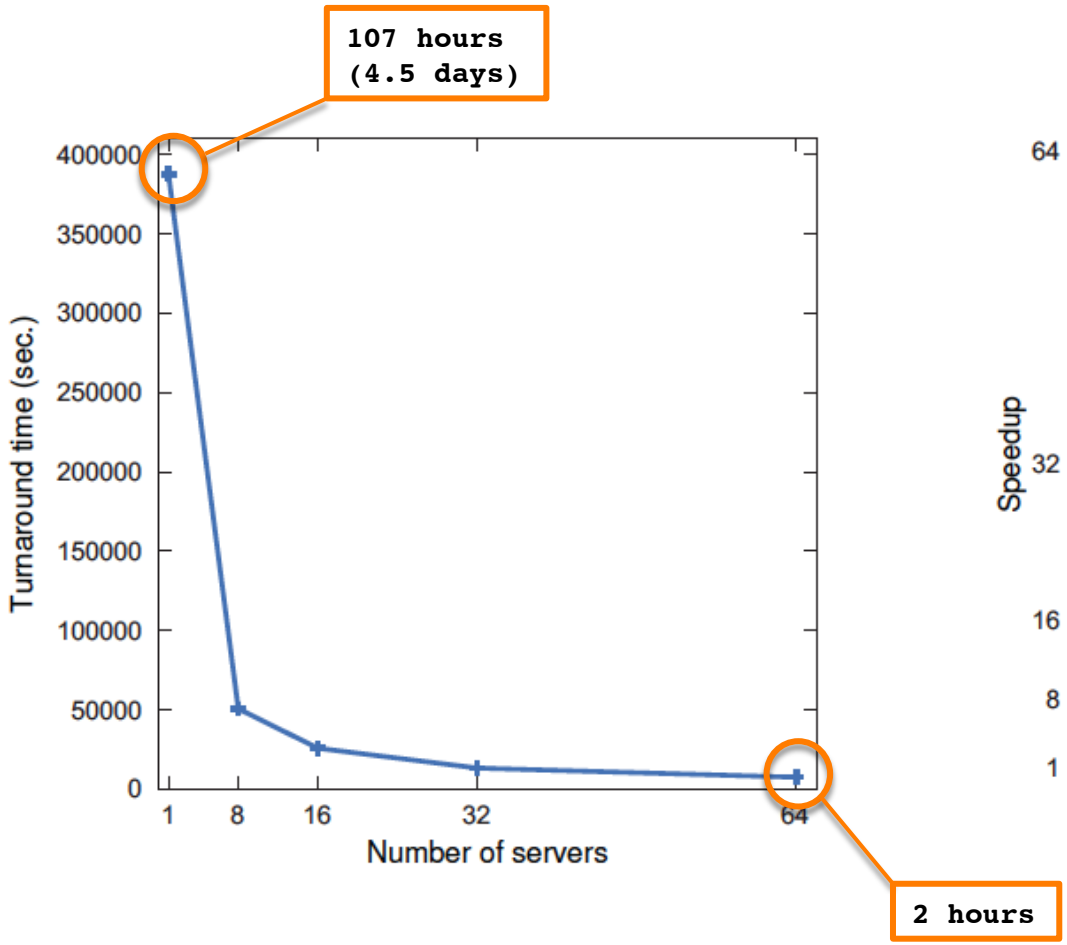
Performance evaluation

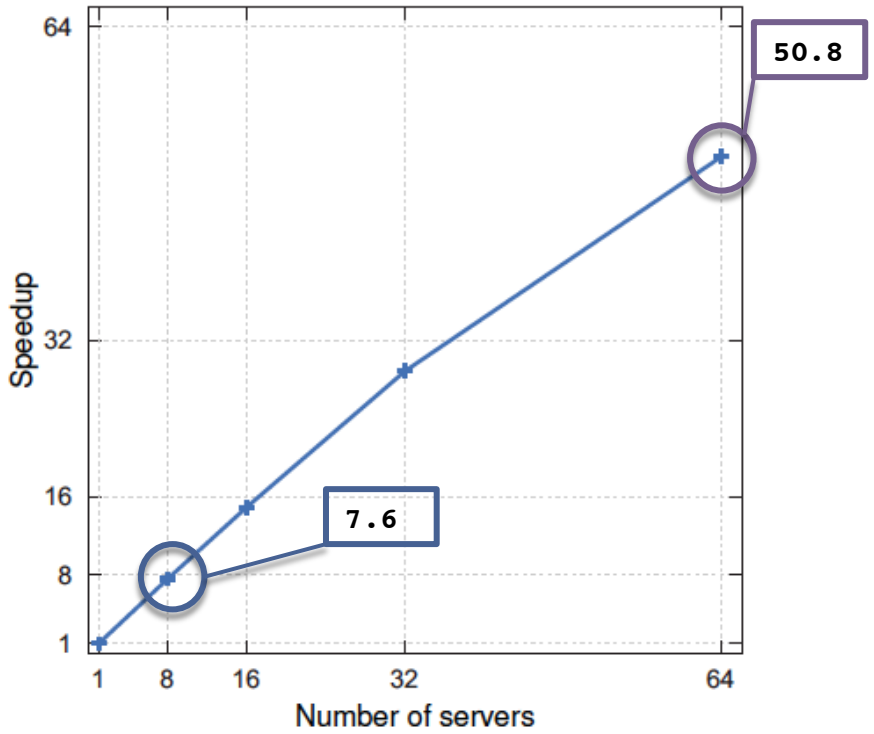
- Input dataset: **46 million tuples**
- Used Cloud: **up to 64 virtual servers** (single-core 1.66 GHz CPU, 1.75 GB of memory, and 225 GB of disk)

```
 1: var n = 64;
 2: var DRef = Data.get("KDDCup99_5GB"),
        TrRef = Data.define("TrainSet"),
        TeRef = Data.define("TestSet");
 3: PartitionerTT({dataset:DRef, percTrain:0.7,
        trainSet:TrRef, testSet:TeRef});
 4: var PRef = Data.define("TrainsetPart", n);
 5: Partitioner({dataset:TrRef, datasetPart:PRef});
 6: var MRef = Data.define("Model", n);
 7: for(var i=0; i<n; i++)
 8:     J48({dataset:PRef[i], model:MRef[i],
           confidence:0.1});
 9: var CRef = Data.define("ClassTestSet", n);
10: for(var i=0; i<n; i++)
11:     Classifier({dataset:TeRef, model:MRef[i],
           classDataset:CRef[i]});
12: var FRef = Data.define("FinalClassTestSet");
13: Voter({classData:CRef, finalClassData:FRef});
```

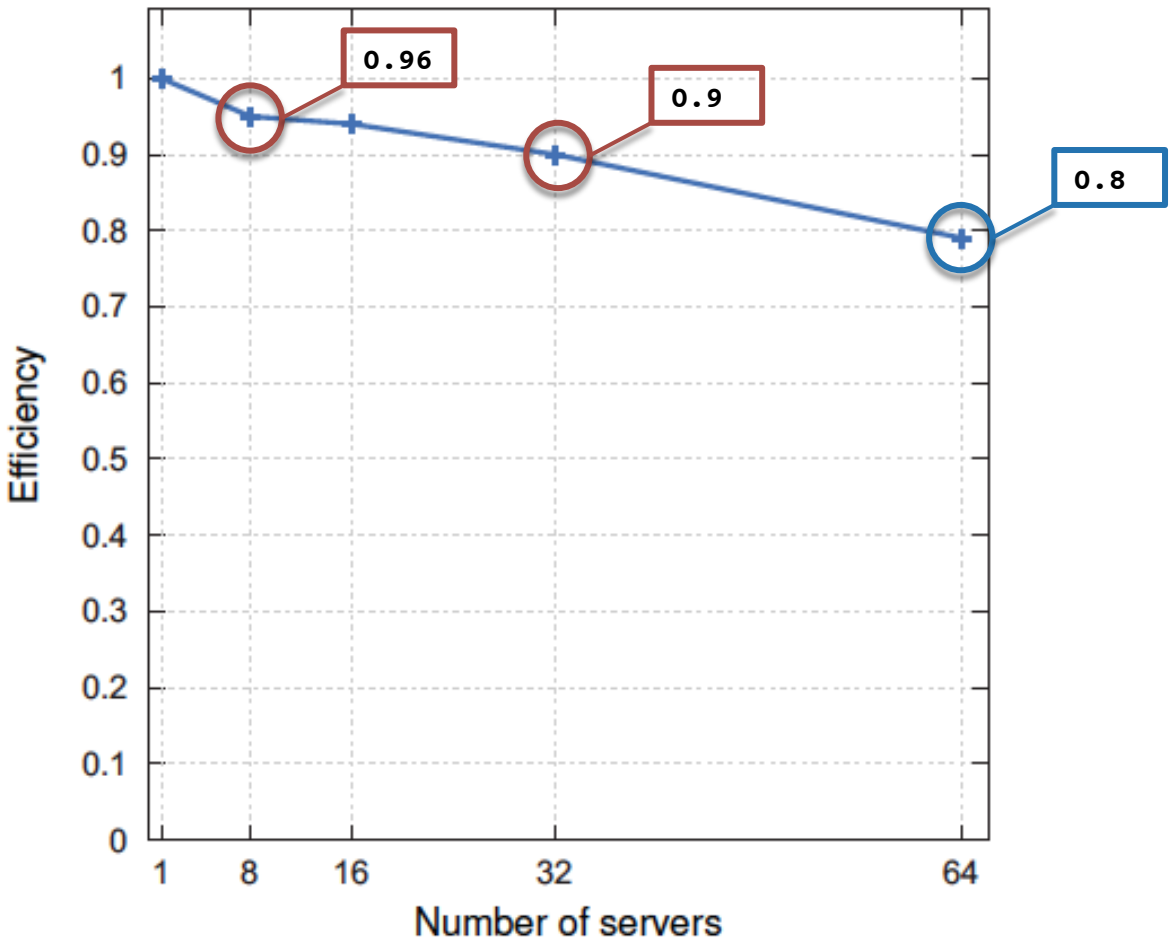


Turnaround and speedup





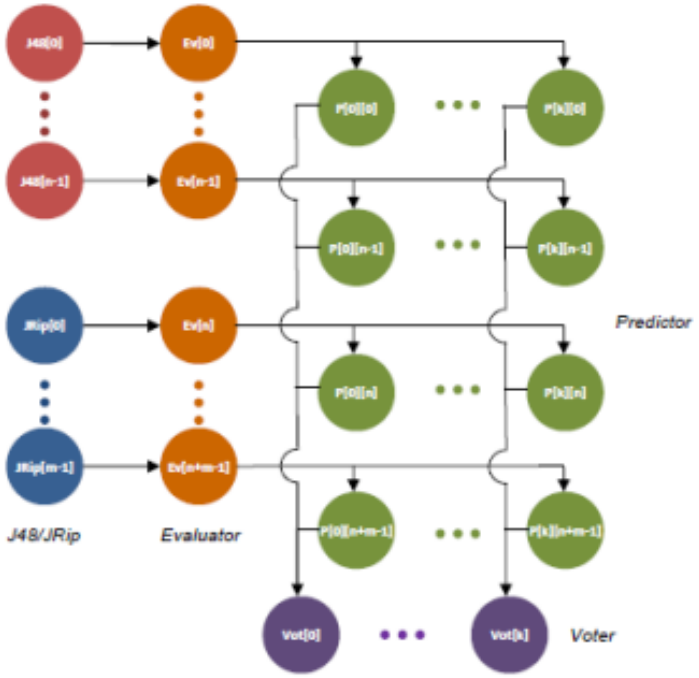
Efficiency


0.96
0.9
0.8
Efficiency
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
1
8
16
32
64
Number of servers

Another application example

- **Ensemble learning workflow** (gene analysis for classifying cancer types)
  - Turnaround time: **162** minutes on **1** server, **11** minutes on **19** servers.
  - Speedup: **14.8**

```
 1: var TrRef = Data.get("GCM-train");
 2: var conf = [0.1, 0.25, 0.5], mno = [2, 5, 10], nfol = [3, 5, 10],
      snum = [1487, 5741, 7699];
 3: var n = conf.length*mno.length, m = nfol.length*snum.length;
 4: var M1Ref = Data.define("Model1", n), M2Ref = Data.define("Model2", m);
 5: for(var i=0; i<conf.length; i++)
 6:    for(var j=0; j<mno.length; j++)
 7:       J48({dataset:TrRef, model:M1Ref[i*mno.length+j], confidence:conf[i],
             minNumObj:mno[j]});
 8: for(var i=0; i<nfol.length; i++)
 9:    for(var j=0; j<snum.length; j++)
10:       JRip({dataset:TrRef, model:M2Ref[i*snum.length+j], numFolds:nfol[i],
             seed:snum[j]});
11: var TeRef = Data.get("GCM-test"), EvM1Ref = Data.define("EvModel1", n),
      EvM2Ref = Data.define("EvModel2", m);
12: for(var i=0; i<n; i++)
13:    Evaluator({dataset:TeRef, model:M1Ref[i], evalModel:EvM1Ref[i]});
14: for(var i=0; i<m; i++)
15:    Evaluator({dataset:TeRef, model:M2Ref[i], evalModel:EvM2Ref[i]});
16: var k = 4;
17: var DRef = Data.get("UnlabGCM", k), CRef = Data.define("ClassGCM",[k,n+m]);
18: for(var i=0; i<k; i++){
19:    for(var j=0; j<n; j++)
20:       Predictor({dataset:DRef[i], model:M1Ref[j], classDataset:CRef[i][j]});
21:    for(var j=0; j<m; j++)
22:       Predictor({dataset:DRef[i], model:M2Ref[j], classDataset:CRef[i][n+j]});
23: }
24: var FRef = Data.define("FinalClassGCM", k), EvMRef = EvM1Ref.concat(EvM2Ref);
25: for(var i=0; i<k; i++)
26:    WeightedVoter({classDataset:CRef[i], evalModel:EvMRef, finalClassDataset:FRef[i]});
```



Final comments

- Data mining and knowledge discovery tools are needed **to support finding what is interesting and valuable** in big data.
- Cloud computing systems can effectively be used as scalable platforms for **service-oriented data mining**.
- Design and programming tools are needed for simplicity and scalability of complex data analysis processes.
- The **DMCF** and its programming interfaces support users in implementing and running scalable data mining.

Ongoing & future work

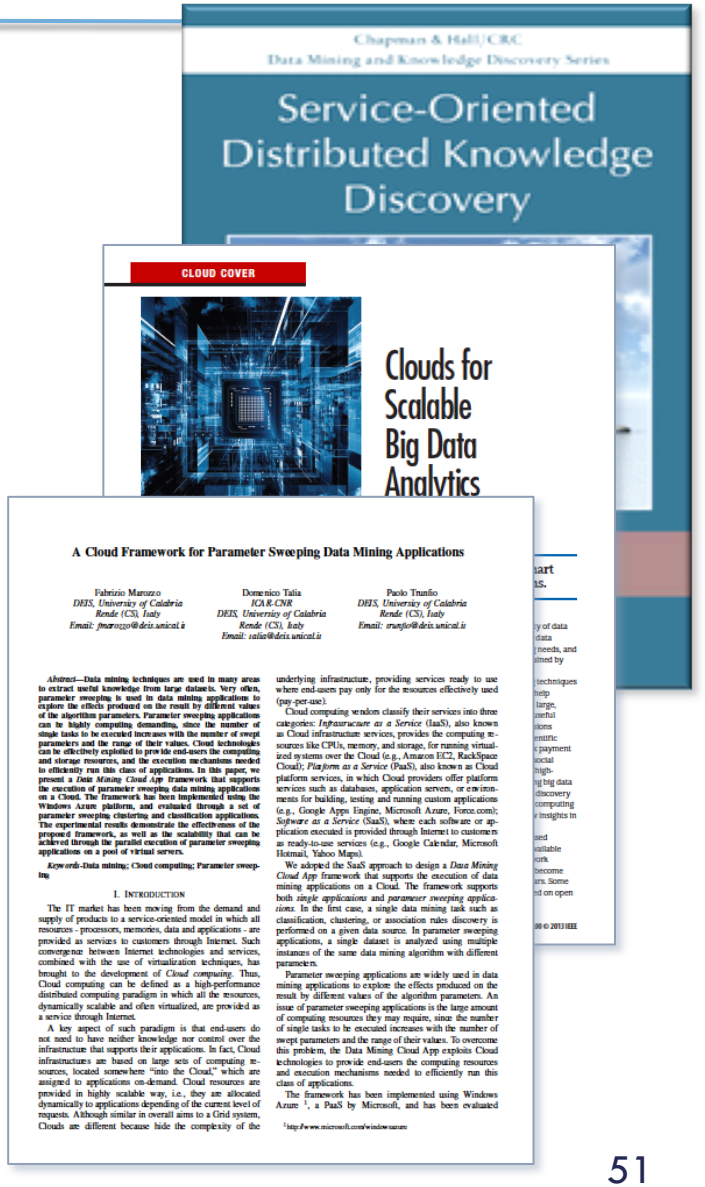
- **DtoK Lab** is a startup that originated from our work in this area.

www.scalabledataanalytics.com

- The **DMCF** system is delivered on public clouds as a high-performance Software-as-a-Service **(SaaS)** to provide innovative data analysis tools and applications.
- Applications in the area of **social data analysis, urban computing, air traffic** and others have been developed by JS4Cloud.

Some publications

- D. Talia, P. Trunfio, *Service-oriented distributed knowledge discovery*, CRC Press, USA, 2012.
- D. Talia, "Clouds for Scalable Big Data Analytics", *IEEE Computer*, 46(5), pp. 98-101, 2013
- F. Marozzo, D. Talia, P. Trunfio, "A Cloud Framework for Parameter Sweeping Data Mining Applications". *Proc. CloudCom 2011*, IEEE CS Press, pp. 367-374, Dec. 2011.
- F. Marozzo, D. Talia, P. Trunfio, "A Cloud Framework for Big Data Analytics Workflows on Azure". In: *Clouds, Grids and Big Data*, IOS Press, 2013.
- F. Marozzo, D. Talia, P. Trunfio, "Using Clouds for Scalable Knowledge Discovery Applications". *Euro-Par Workshops*, LNCS, Springer, pp. 220-227, Aug. 2012.



Thanks

Questions?