Mining Large Datasets: Case of Mining Graph Data in the Cloud

Sabeur Aridhi

PhD in Computer Science

with Laurent d'Orazio, Mondher Maddouri and Engelbert Mephu Nguifo

16/05/2014

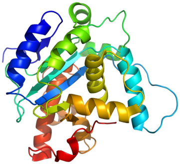
Context and motivations

Application domains

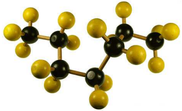
- Computer networks,
- Social networks,
- Bioinformatics,
- Chemoinformatics.

Graph representation

- Data modeling.
- Identifying relationship patterns and rules.



Protein structure



Chemical compound

Social network

Context and motivations

Mining graph data

- Graph mining aims to find patterns, hidden relations and behaviors in data.

Context and motivations

Mining graph data

- Graph mining aims to find patterns, hidden relations and behaviors in data.

Mining graph goals

- Computing graph properties:
  - Density, diameter, radius, ...
- Mining substructures from graph databases.
  - Substructures: paths, trees, subgraphs.
  - Frequent Subgraph Mining (FSM) task.

Context and motivations

Availability of graph data

- Exponential growth in both size and number of graphs in databases.

Context and motivations

Availability of graph data

- Exponential growth in both size and number of graphs in databases.
- Availability of graph data sources:
  - The protein data bank (PDB) contains 95280 of protein 3D structures.
  - Facebook loads 60 terabytes of new data every day **[Thusoo 2010]**.
  - Google processes 20 petabytes of data per day **[Dean 2008]**.

Context and motivations

Availability of graph data

- Exponential growth in both size and number of graphs in databases.
- Availability of graph data sources:
  - The protein data bank (PDB) contains 95280 of protein 3D structures.
  - Facebook loads 60 terabytes of new data every day **[Thusoo 2010]**.
  - Google processes 20 petabytes of data per day **[Dean 2008]**.
- 3Vs of Big Data (Volume, Velocity and Variety).

Context and motivations

Availability of graph data

- Exponential growth in both size and number of graphs in databases.
- Availability of graph data sources:
  - The protein data bank (PDB) contains 95280 of protein 3D structures.
  - Facebook loads 60 terabytes of new data every day **[Thusoo 2010]**.
  - Google processes 20 petabytes of data per day **[Dean 2008]**.
- 3Vs of Big Data (Volume, Velocity and Variety).
- Availability of cloud computing environments.

Context and motivations

In this work

- We are interested to FSM from graph databases.

Context and motivations

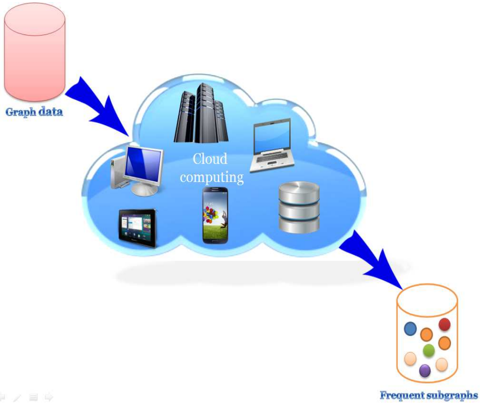
In this work

- We are interested to FSM from graph databases.

Frequent subgraph mining algorithms

- Various approaches of FSM.
- Existing approaches are mainly:
  - Tested on centralized computing systems.
  - Evaluated on relatively small databases.
- Few works for FSM in the cloud.

Goals



Questions

- Distributed FSM from large graph database.
- Data/computation distribution.
- Tuning cloud parameters.

Outline

1. Background
2. Contributions
3. Conclusion

Outline

1 Background
- Graph mining
- Cloud computing
- Frameworks for large data processing in the cloud
- Related works

2 Contributions

3 Conclusion

Outline

1. Background
   - Graph mining
   - Cloud computing
   - Frameworks for large data processing in the cloud
   - Related works
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion
   - Contributions
   - Prospects

Background

Graph

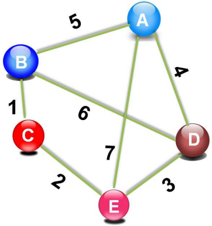
A graph is denoted as $G = (V, E)$ where V is a set of nodes and E is a set of edges.

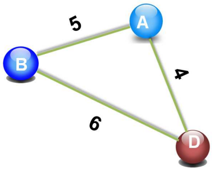
Subgraph

A graph $G' = (V', E')$ is a subgraph of another graph $G = (V, E)$ iff: $V' \subseteq V$, and $E' \subseteq E \cap (V' \times V')$.

Density

The density of a graph $G = (V, E)$ is calculated by $density(G) = \frac{2 \cdot |E|}{(|V| \cdot (|V| - 1))}$.





Outline

1 Background
- Graph mining
- Cloud computing
- Frameworks for large data processing in the cloud
- Related works

2 Contributions
- Distributed subgraph mining in the cloud

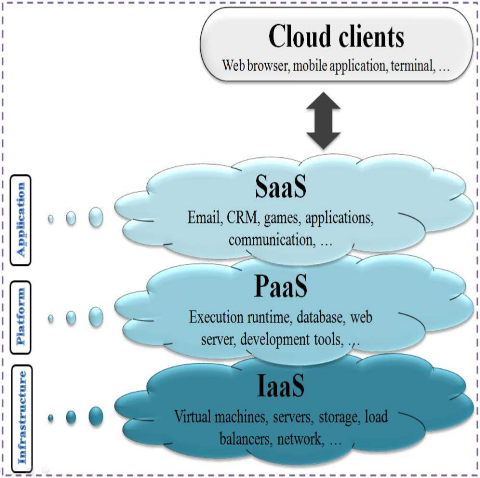
3 Conclusion
- Contributions
- Prospects

Background

Cloud computing

- Large number of computers that are connected via Internet.
- Applications delivered as services.
- Hardware and system software delivered as services.
- Pay as you go.
- Cloud services can be rapidly and elastically provisioned.

Background



Service models

- Software as a Service (SaaS).
- Platform as a Service (PaaS),
- Infrastructure as a Service (IaaS),

Outline

1. Background
   - Graph mining
   - Cloud computing
   - Frameworks for large data processing in the cloud
   - Related works
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion
   - Contributions
   - Prospects

Background

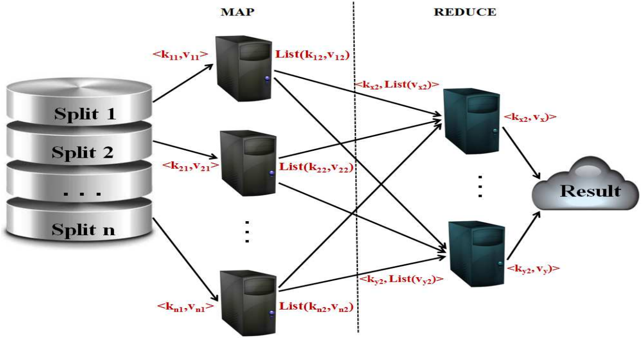
MapReduce framework

- A framework for processing huge datasets.
- Large number of computers and task/node failures.

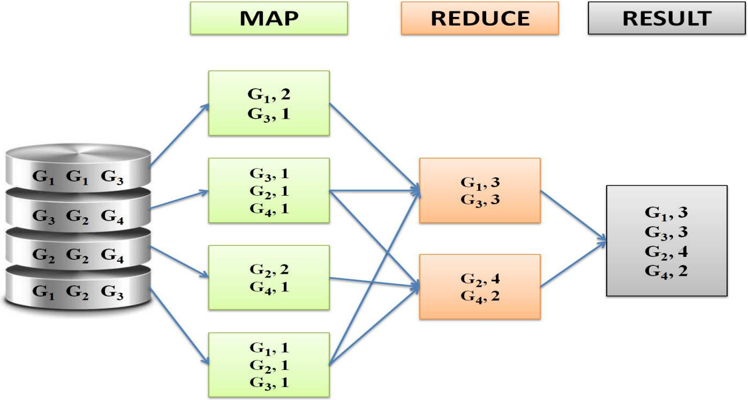
Background

MapReduce framework

- A framework for processing huge datasets.
- Large number of computers and task/node failures.



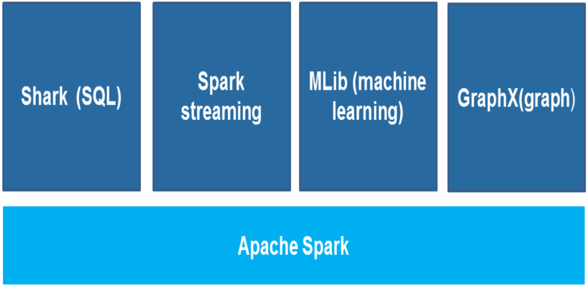
Background


MAP
REDUCE
RESULT
G1 G1 G3
G3 G2 G4
G2 G2 G4
G1 G2 G3
G1, 2
G3, 1
G3, 1
G2, 1
G4, 1
G2, 2
G4, 1
G1, 1
G2, 1
G3, 1
G1, 3
G3, 3
G2, 4
G4, 2
G1, 3
G3, 3
G2, 4
G4, 2

Background

SPARK framework

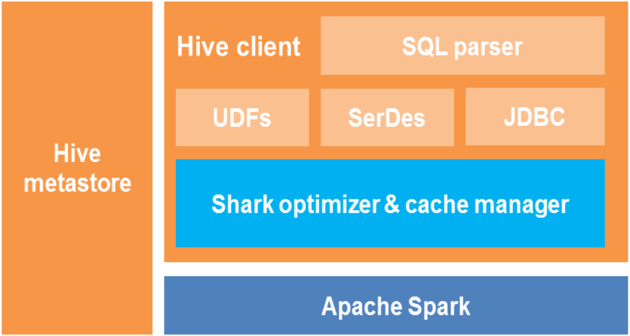
- A general engine for large-scale data processing.
- Combine SQL, streaming, and complex analytics.
- It offers several high-level operators that make it easy to build parallel applications.



Background

SHARK framework

- A distributed SQL query engine for Hadoop.
- Based on SPARK and uses the existing Hive client and metastore.



Outline

1. Background
   - Graph mining
   - Cloud computing
   - Frameworks for large data processing in the cloud
   - Related works
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion
   - Contributions
   - Prospects

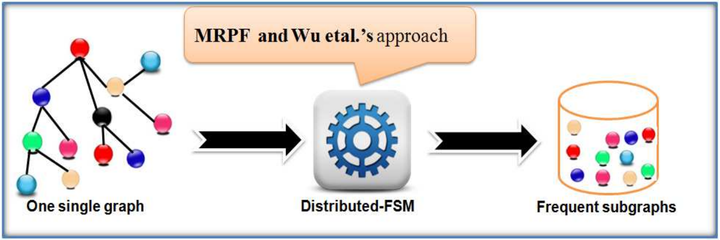
Background

Cloud-based FSM techniques

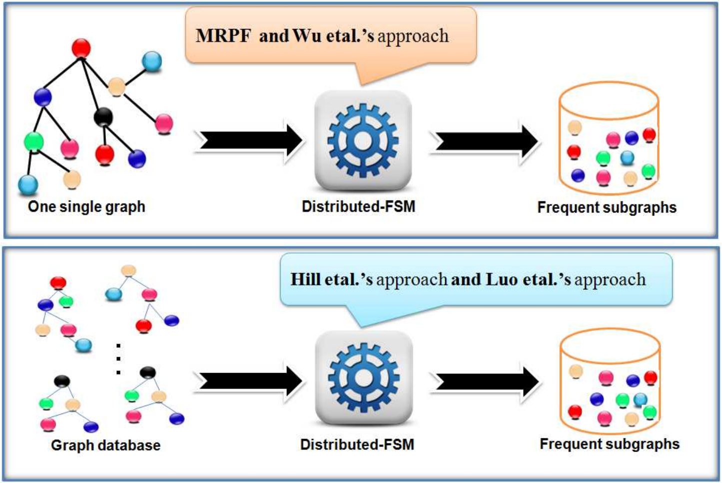
Cloud-based FSM approaches from:

1. Single large graphs (MRPF **[Liu 2009]** and Wu *etal.'s* approach **[Wu 2010]**).
   - MRPF **[Liu 2009]**, and
   - Wu *etal.'s* approach **[Wu 2010]**.
2. Massive graph databases (Hill *etal.'s* **[Hill 2012]** and Luo *etal.'s* **[Luo 2011]**).
   - Hill *etal.'s* **[Hill 2012]**, and
   - Luo *etal.'s* **[Luo 2011]**.

Background


MRPF and Wu etal.'s approach
One single graph
Distributed-FSM
Frequent subgraphs

Background


MRPF and Wu etal.'s approach
One single graph
Distributed-FSM
Frequent subgraphs
Hill etal.'s approach and Luo etal.'s approach
Graph database
Distributed-FSM
Frequent subgraphs

Background

In this work

- We focus on distributed FSM techniques from large graph databases.

Background

In this work

- We focus on distributed FSM techniques from large graph databases.
- Three crucial problems with existing approaches:

Background

In this work

- We focus on distributed FSM techniques from large graph databases.
- Three crucial problems with existing approaches:
  1. No data partitioning according to data characteristics.

Background

In this work

- We focus on distributed FSM techniques from large graph databases.
- Three crucial problems with existing approaches:
  1. No data partitioning according to data characteristics.
  2. Do not include the monetary aspect of cloud computing.

Background

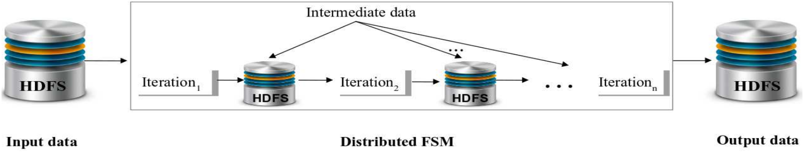
In this work

- We focus on distributed FSM techniques from large graph databases.
- Three crucial problems with existing approaches:
  1. No data partitioning according to data characteristics.
  2. Do not include the monetary aspect of cloud computing.
  3. Construct the final set of frequent subgraphs iteratively.

Background

In this work

- We focus on distributed FSM techniques from large graph databases.
- Three crucial problems with existing approaches:
  1. No data partitioning according to data characteristics.
  2. Do not include the monetary aspect of cloud computing.
  3. Construct the final set of frequent subgraphs iteratively.



Outline

1. Background
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion

Outline

1. Background
   - Graph mining
   - Cloud computing
   - Frameworks for large data processing in the cloud
   - Related works
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion
   - Contributions
   - Prospects

Problem formulation

Notations

- $DB = \{G_1, \ldots, G_K\}$ is a large scale graph database,
- $SM = \{M_1, \ldots, M_N\}$ is a set of distributed machines,
- $\theta \in [0, 1]$ is a minimum support threshold,
- $Part(DB) = \{Part_1(DB), \ldots, Part_N(DB)\}$ is a partitioning of the database over SM such that
  - $Part_j(DB) \subseteq DB$ is a non-empty subset of DB,
  - $\bigcup_{i=1}^{N} \{Part_i(DB)\} = DB, and,$
  - $\forall i \neq j, Part_i(DB) \cap Part_j(DB) = \emptyset.$

Problem formulation

Globally frequent subgraph

For a given minimum support threshold $\theta \in [0,1]$, G' is *globally frequent subgraph* if $Support(G', DB) \geq \theta$.

Problem formulation

Globally frequent subgraph

For a given minimum support threshold $\theta \in [0,1]$, G' is *globally frequent subgraph* if $Support(G', DB) \geq \theta$.

Locally frequent subgraph

For a given minimum support threshold $\theta \in [0,1]$ and a tolerance rate $\tau \in [0,1]$, G' is *locally frequent subgraph* at site i if $Support(G', Part_i(DB)) \geq ((1-\tau)\cdot\theta)$.

Problem formulation

Globally frequent subgraph

For a given minimum support threshold $\theta \in [0,1]$, G' is *globally frequent subgraph* if $Support(G', DB) \geq \theta$.

Locally frequent subgraph

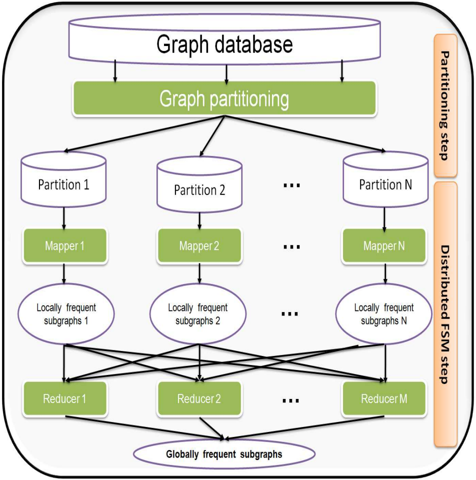
For a given minimum support threshold $\theta \in [0,1]$ and a tolerance rate $\tau \in [0,1]$, G' is *locally frequent subgraph* at site i if $Support(G', Part_i(DB)) \geq ((1-\tau)\cdot\theta)$.

Loss rate

Given S_1 and S_2 two sets of subgraphs with $S_2 \subseteq S_1$ and $S_1 \neq \emptyset$, we define the loss rate in S_2 compared to S_1 by:

$LossRate(S_1, S_2) = \frac{|S_1 - S_2|}{|S_1|}$.

System overview

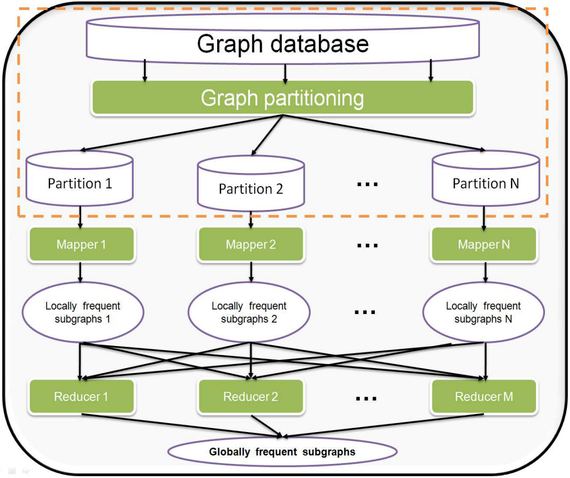


Approach overview

Two-step approach:

1. Partitioning step,
2. Mining step.

Partitioning step



Partitioning step

Partitioning methods

Many partitioning methods are possible. We consider:

1. MRGP: the default MapReduce partitioning method.
2. DGP: a density-based partitioning method.

Partitioning step

Partitioning methods

Many partitioning methods are possible. We consider:

1. MRGP: the default MapReduce partitioning method.
2. DGP: a density-based partitioning method.

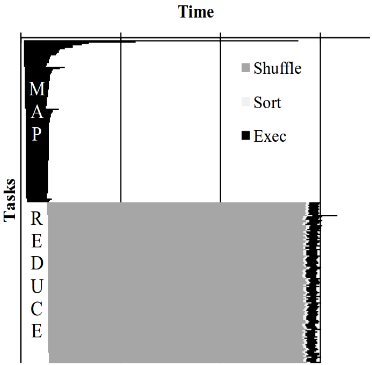
MRGP

- Based on the size on disk.
- *Map-skew* problems (highly variable runtimes).
  - No data characteristics included.

DGP

- Based on graph density.
- May ensures load balancing among machines.
  - May exploit other data characteristics.

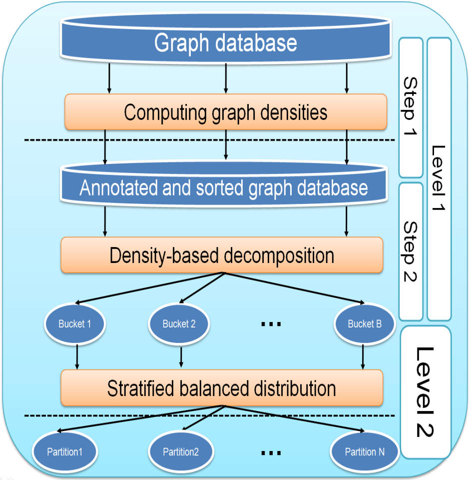
Map-Skew problems



Map-skew

- **Skew:** highly variable task runtimes.
- Origin:
  - Characteristics of the algorithm.
  - Characteristics of the dataset.

Partitioning step: DGP method

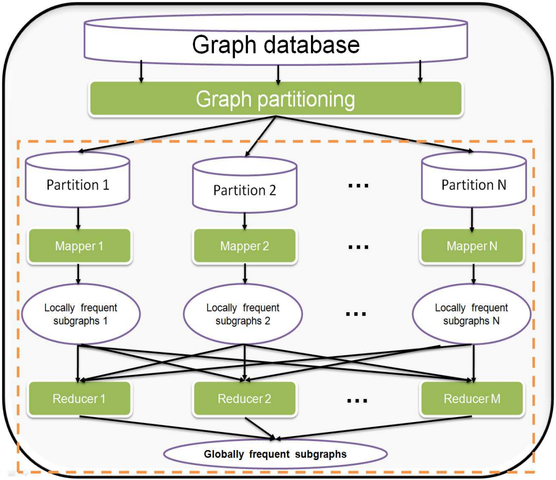


DGP overview

Two-levels approach:

1. Dividing the graph database into *B* buckets,
2. Constructing the final list of partitions.

Distributed FSM step



Distributed FSM step

Distributed FSM step

- A single MapReduce job.
  - **Input:** a set of partitions.
  - **Output:** the set of globally frequent subgraphs.

Distributed FSM step

Distributed FSM step

- A single MapReduce job.
  - **Input:** a set of partitions.
  - **Output:** the set of globally frequent subgraphs.

In the Mapper machine

- We run a subgraph mining technique on each partition in parallel.
- Mapper i produces a set of locally frequent subgraphs.
  - Pairs of $\langle s, Support(s, Part_i(DB))\rangle$.

Distributed FSM step

Distributed FSM step

- A single MapReduce job.
  - **Input:** a set of partitions.
  - **Output:** the set of globally frequent subgraphs.

In the Mapper machine

- We run a subgraph mining technique on each partition in parallel.
- Mapper *i* produces a set of locally frequent subgraphs.
  - Pairs of $\langle s, Support(s, Part_i(DB))\rangle$.

In the Reducer machine

- We compute the set of globally frequent subgraphs
  - Pairs of $\langle s, Support(s, DB)\rangle$.
  - No false positives generated.

Distributed FSM step

Algorithm 1 Map function.

Require: A partitioned graph database $DB = \{Part_1(DB), \ldots, Part_N(DB)\}$, minimum support threshold θ, tolerance rate τ, key $= i$, value= graph partition $Part_i(DB)$

Ensure: Locally frequent subgraphs in $Part_i(DB)$

1: $S_i \leftarrow FSMLocal(Part_i(DB), \theta, \tau)$
2: **for all** s in S_i **do**
3: $\quad EmitIntermediate(s, Support(s, Part_i(DB)))$
4: **end for**

Algorithm 2 Reduce function.

Require: Minimum support threshold θ, key=a subgraph s, values=local supports of s

Ensure: Globally frequent subgraphs in DB

1: $GlobalSupportCount \leftarrow 0$
2: **for all** v in $values$ **do**
3: $\quad GlobalSupportCount \leftarrow GlobalSupportCount + v$
4: **end for**
5: $GlobalSupport \leftarrow \frac{GlobalSupportCount}{N}$
6: **if** $GlobalSupport >= \theta$ **then**
7: $\quad Emit(s, GlobalSupport)$
8: **end if**

Experiments

Implementation platform

- Hadoop 0.20.1 release, an open source version of MapReduce.
- A local cluster with five nodes.
  - A Quad-Core AMD Opteron(TM) Processor 6234 2.40 GHz CPU.
  - 4 GB of memory.
- Three existing subgraph miners: gSpan, FSG and Gaston.

Experiments

Implementation platform

- Hadoop 0.20.1 release, an open source version of MapReduce.
- A local cluster with five nodes.
  - A Quad-Core AMD Opteron(TM) Processor 6234 2.40 GHz CPU.
  - 4 GB of memory.
- Three existing subgraph miners: gSpan, FSG and Gaston.

Datasets

- Six datasets composed of synthetic and real ones.
- Different parameters such as: the number of graphs, the average size of graphs in terms of edges and the size on disk.

Experiments

Table: Experimental data.

Dataset	Type	Number of graphs	Size on disk	Average size
DS1	Synthetic	20,000	18 MB	[50-100]
DS2	Synthetic	100,000	81 MB	[50-70]
DS3	Real	274,860	97 MB	[40-50]
DS4	Synthetic	500,000	402 MB	[60-70]
DS5	Synthetic	1,500,000	1.2 GB	[60-70]
DS6	Synthetic	100,000,000	69 GB	[20-100]

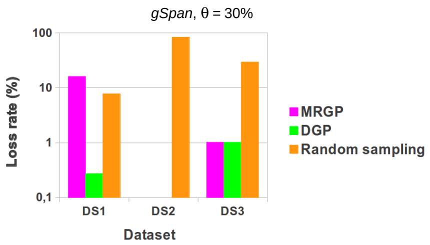
Experiments

Experimental protocol

Three types of experiments:

1. Quality:
   - MRGP vs. DGP.
   - Comparison with random sampling method.
2. Load balancing and execution time:
   - Performance evaluation tests.
   - Scalability tests.
3. Impact of MapReduce parameters.

Experiments: Quality



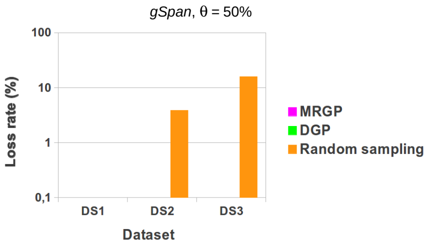
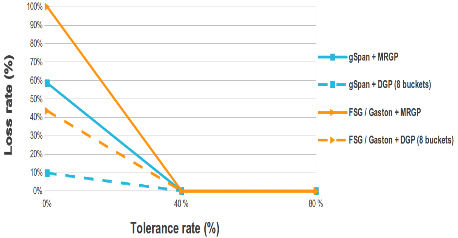
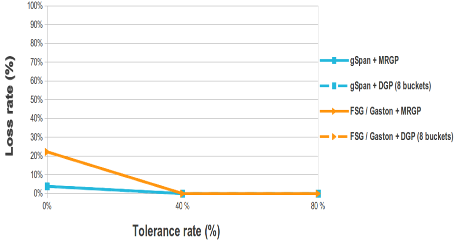


Table: Number of false positives of the sampling method.

Dataset	Support θ (%)	gSpan		FSG		Gaston	
		Number of subgraphs	Number of false positives	Number of subgraphs	Number of false positives	Number of subgraphs	Number of false positives
DS1	30	4421	4078	4401	4078	4401	4078
	50	194	155	174	153	174	153
DS2	30	164	139	144	58	144	58
	50	29	4	12	4	12	4
DS3	30	264	195	258	193	258	193
	50	62	30	59	30	59	30

Experiments: Quality

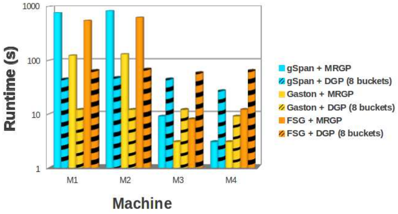


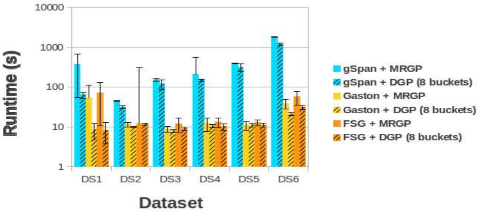


Result quality

- Distributed FSM vs. classic one.
- Low values of loss rate with DGP.

Experiments: Load balancing and execution time

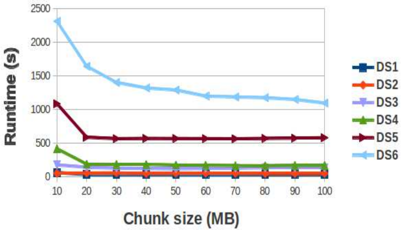


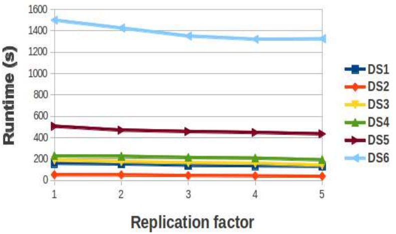


Runtime and workload distribution

- DGP enhances the performance of our approach.
- Balanced workload distribution over the distributed machines.

Experiments: Impact of MapReduce parameters





Chunk size and replication factor

- High runtime values with small chunk size.
- The runtime is inversely proportional to the replication factor.

Outline

1. Background
2. Contributions
3. Conclusion
   - Contributions
   - Prospects

Outline

1. Background
   - Graph mining
   - Cloud computing
   - Frameworks for large data processing in the cloud
   - Related works
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion
   - Contributions
   - Prospects

Conclusion

At a glance

- A MapReduce-based framework for distributing FSM in the cloud.
  - Many partitioning techniques of the input graph database.
  - Many subgraph extractors.
- A data partitioning technique that considers data characteristics.
  - It uses the density of graphs.
  - Balanced computational load over the distributed machines.
- Experiment validation.

Outline

1. Background
   - Graph mining
   - Cloud computing
   - Frameworks for large data processing in the cloud
   - Related works
2. Contributions
   - Distributed subgraph mining in the cloud
3. Conclusion
   - Contributions
   - Prospects

Prospects

Improvements of the cloud-based FSM approach

- Different topological graph properties.
- Relation between database characteristics and the choice of the partitioning technique.

Open questions

- What is the maximum number of buckets and/or partitions?
- What is the size of chunk to use in the partitioning step and in the distributed subgraph mining step?

Prospects

Performance and scalability improvement

- Runtime improvement with task and node failures.
- Ensure minimal loss of information in the case of failures.

Portability improvement

- Extension of our approach to SPARK, SHARK, Open Computing Language (OpenCL) and Message Passing Interface (MPI).

Deployment of the approach

- Study the integration of our approach to recent distributed machine learning toolkits such as the Apache Mahout project and SystemML.

Work in progress

Cost models

- Cost models for distributing frequent pattern mining in the cloud.
    - Application to distributed frequent subgraphs.
- Objective functions that consider the needs of customers:
    - Budget limit,
    - Response time limit, and
    - Result quality limit.

Publications

Journals

- S. Aridhi, L. d'Orazio, M. Maddouri et E. Mephu Nguifo. Un partitionnement basé sur la densité de graphe pour approcher la fouille distribuée de sous-graphes fréquents. Techniques et Science Informatiques. (Accepted)
- S. Aridhi, L. d'Orazio, M. Maddouri and E. Mephu Nguifo. Density-based data partitioning strategy to approximate large scale subgraph mining. Information Systems, Elsevier, ISSN 0306-4379, http://dx.doi.org/10.1016/j.is.2013.08.005, 2014. (In press)

Thank You!