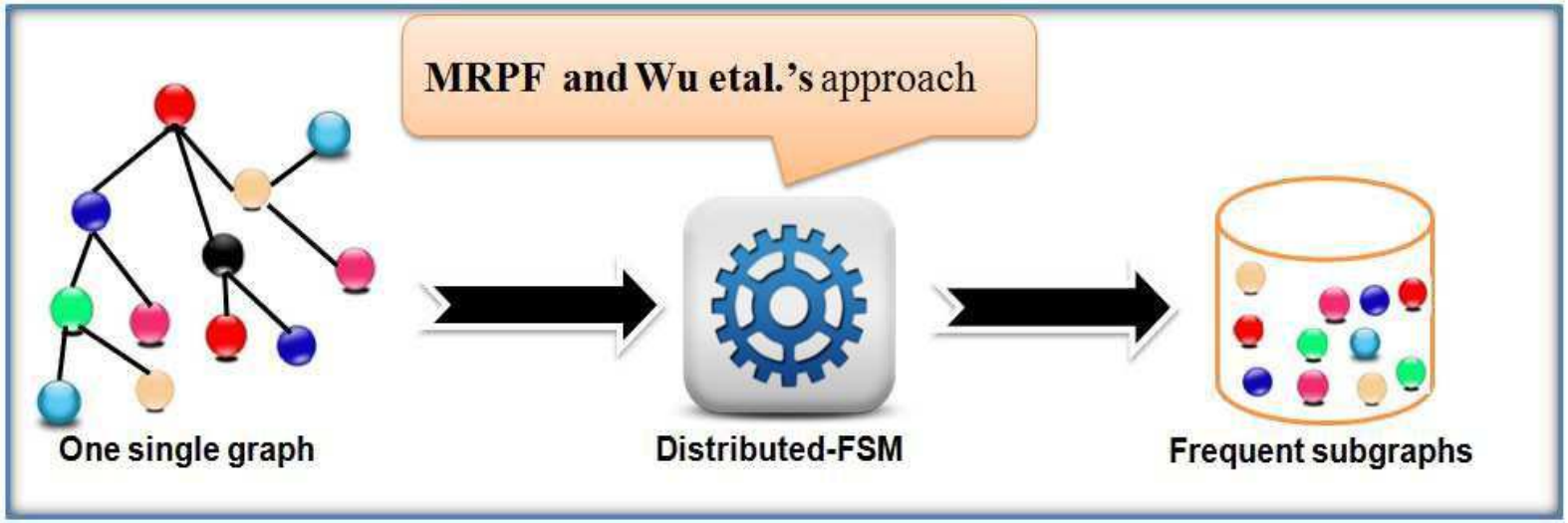
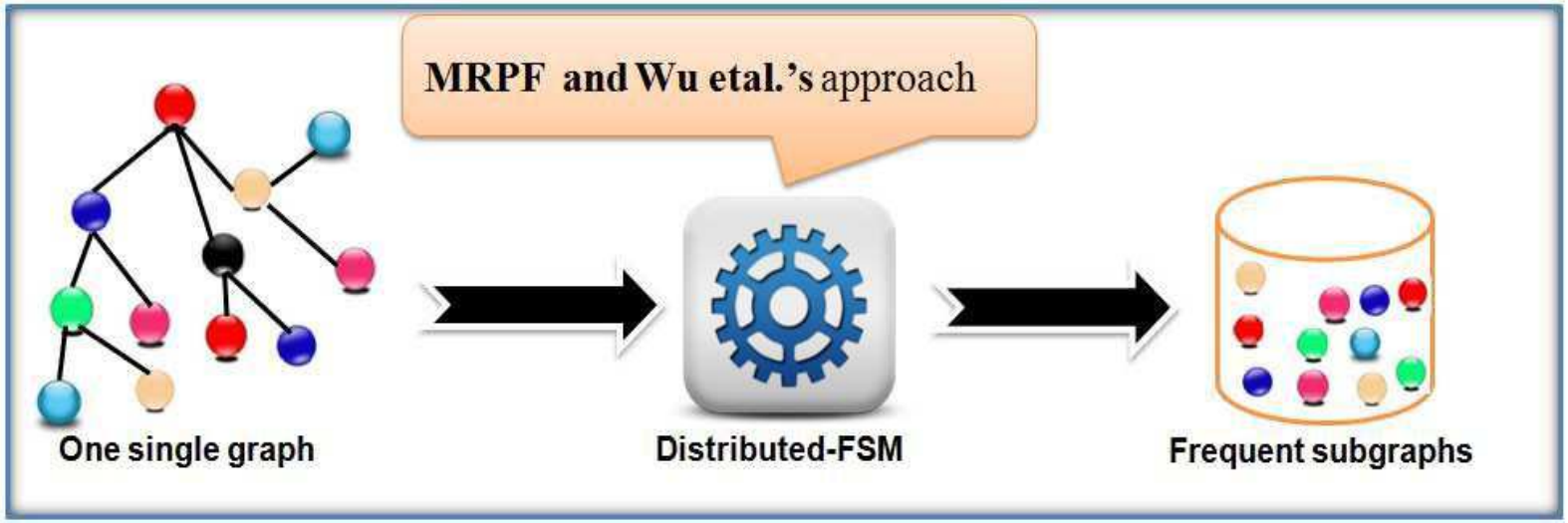
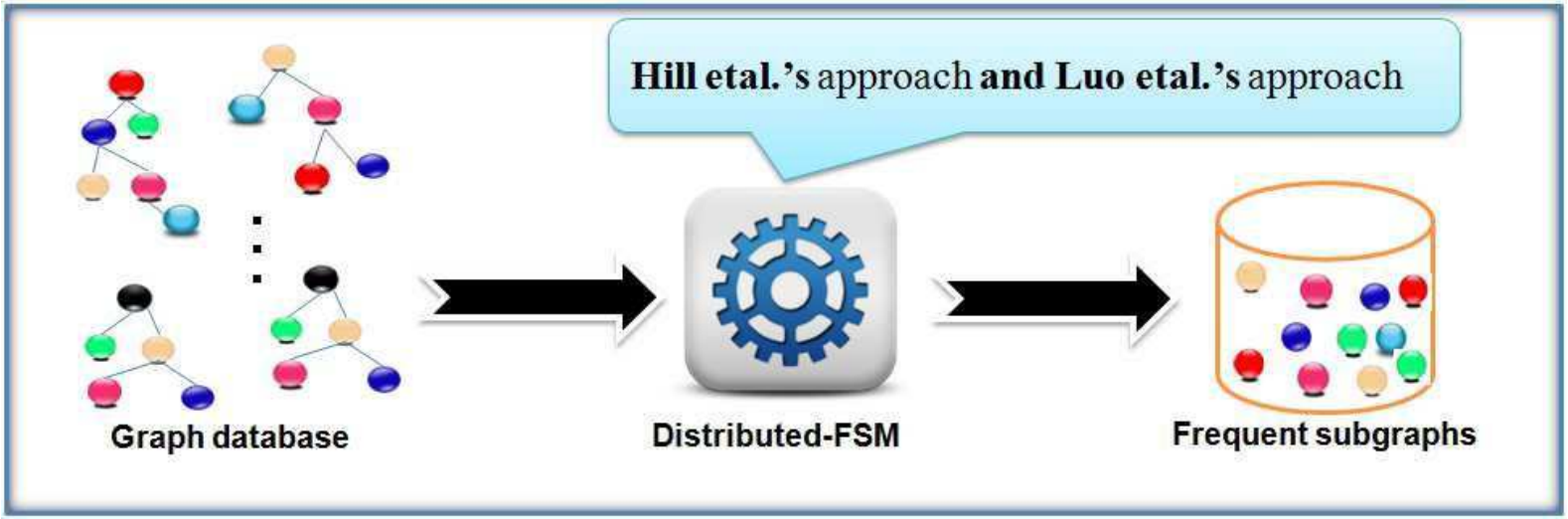
Outline

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

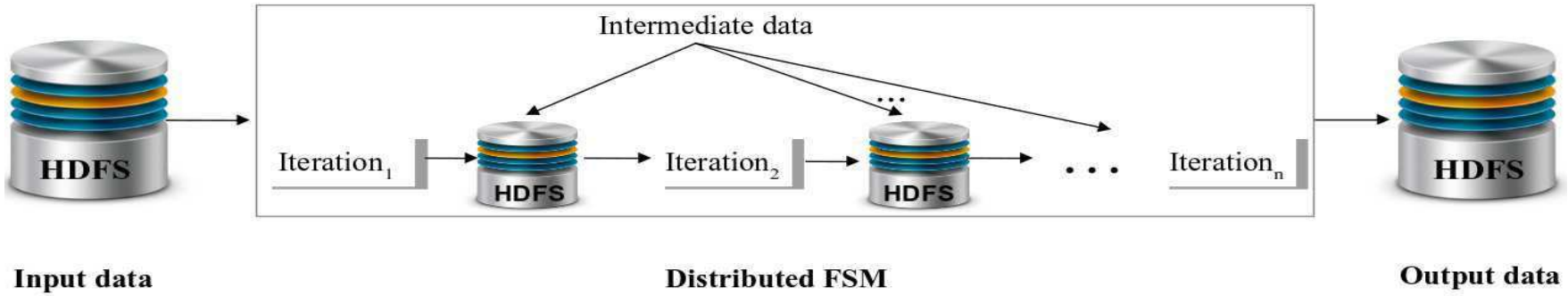
In this work

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

Background

In this work

- We focus on distributed FSM techniques from large graph databases.
- Two crucial problems with existing approaches:
  1. No data partitioning according to data characteristics.
  2. Construct the final set of frequent subgraphs iteratively.



Outline

Outline

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

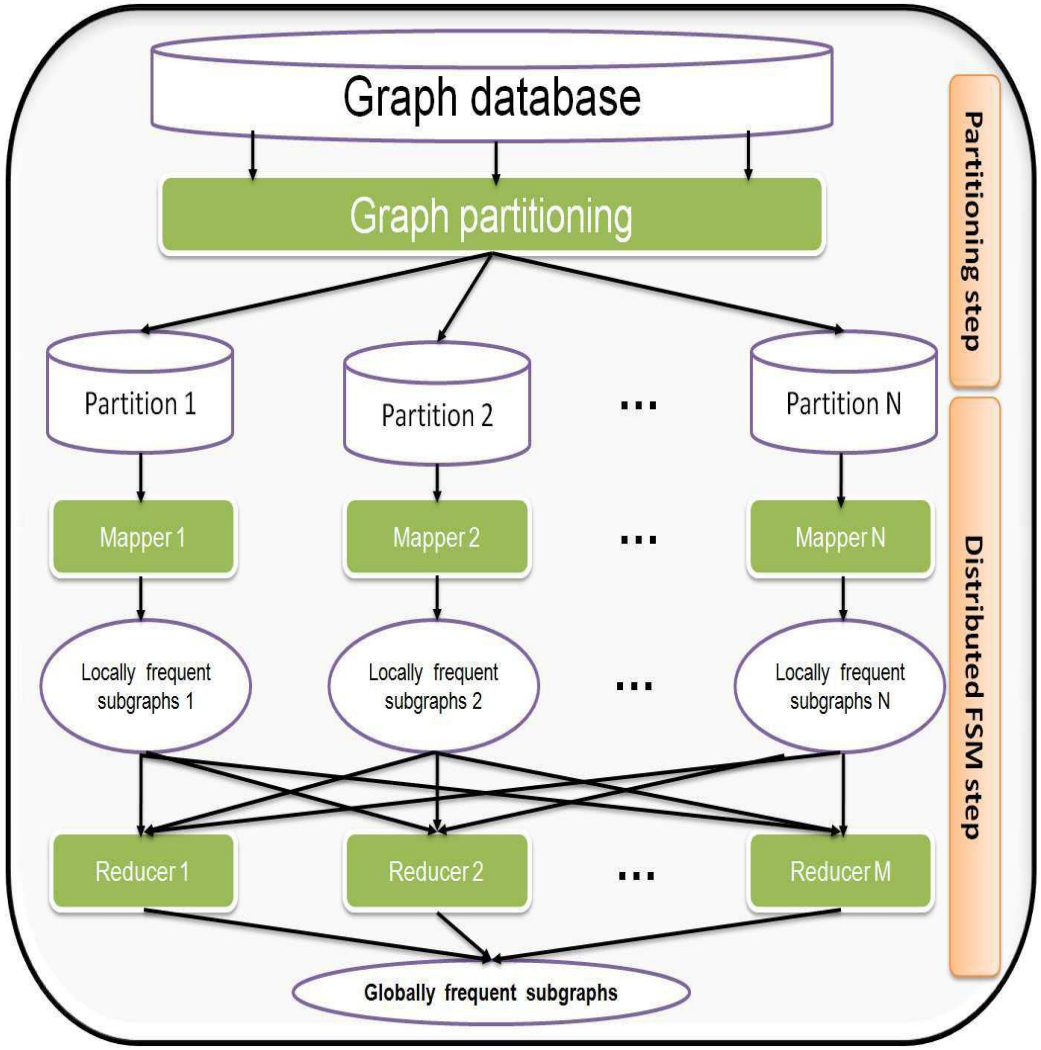
For a given minimum support threshold $\theta \in [0,1]$ and a tolerance rate $\tau \in [0,1]$, G' is *locally frequent subgraph* at site i if $Support(G', Part_i(DB)) \geq ((1-\tau) \cdot \theta)$.

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

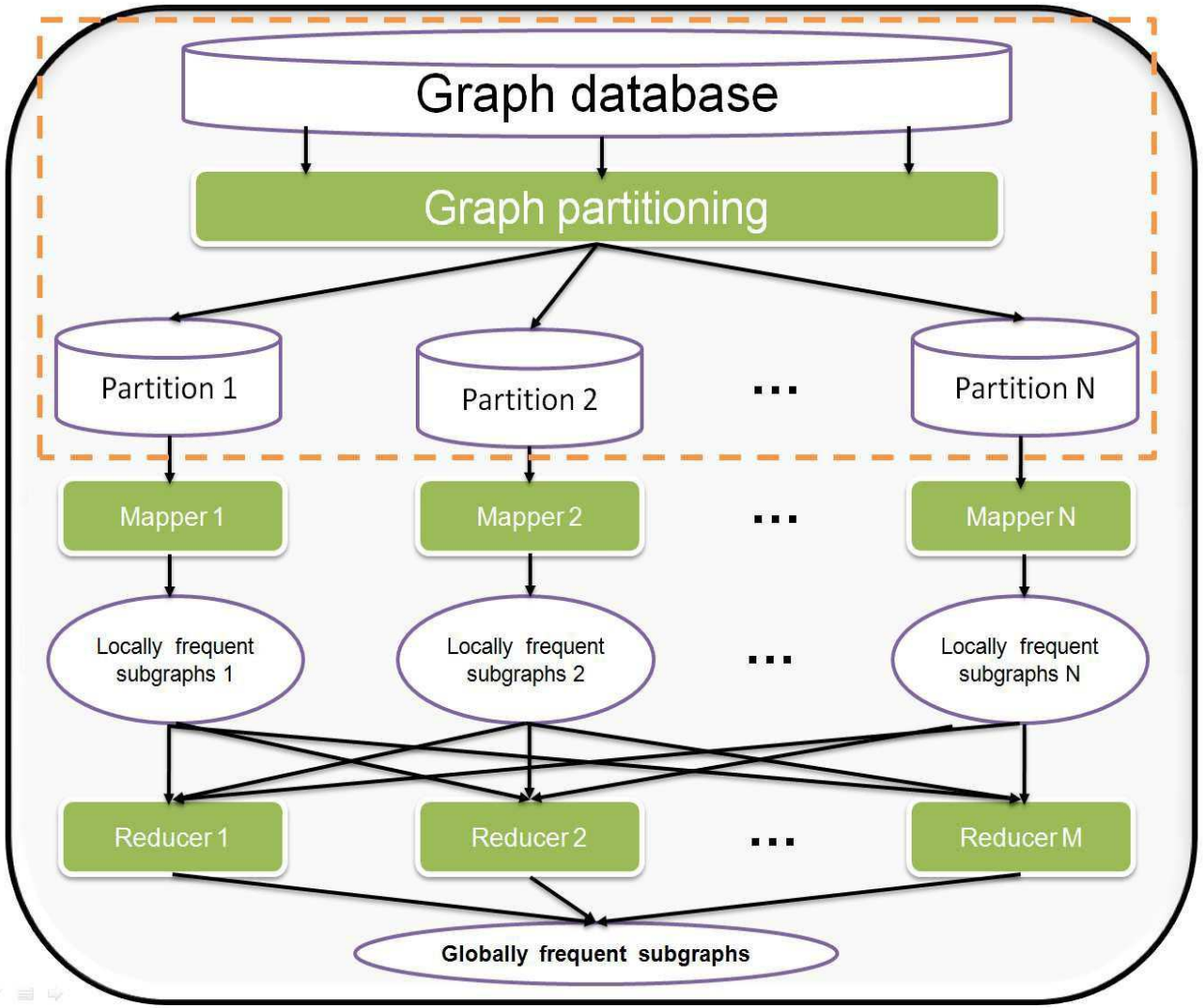


Approach overview

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.

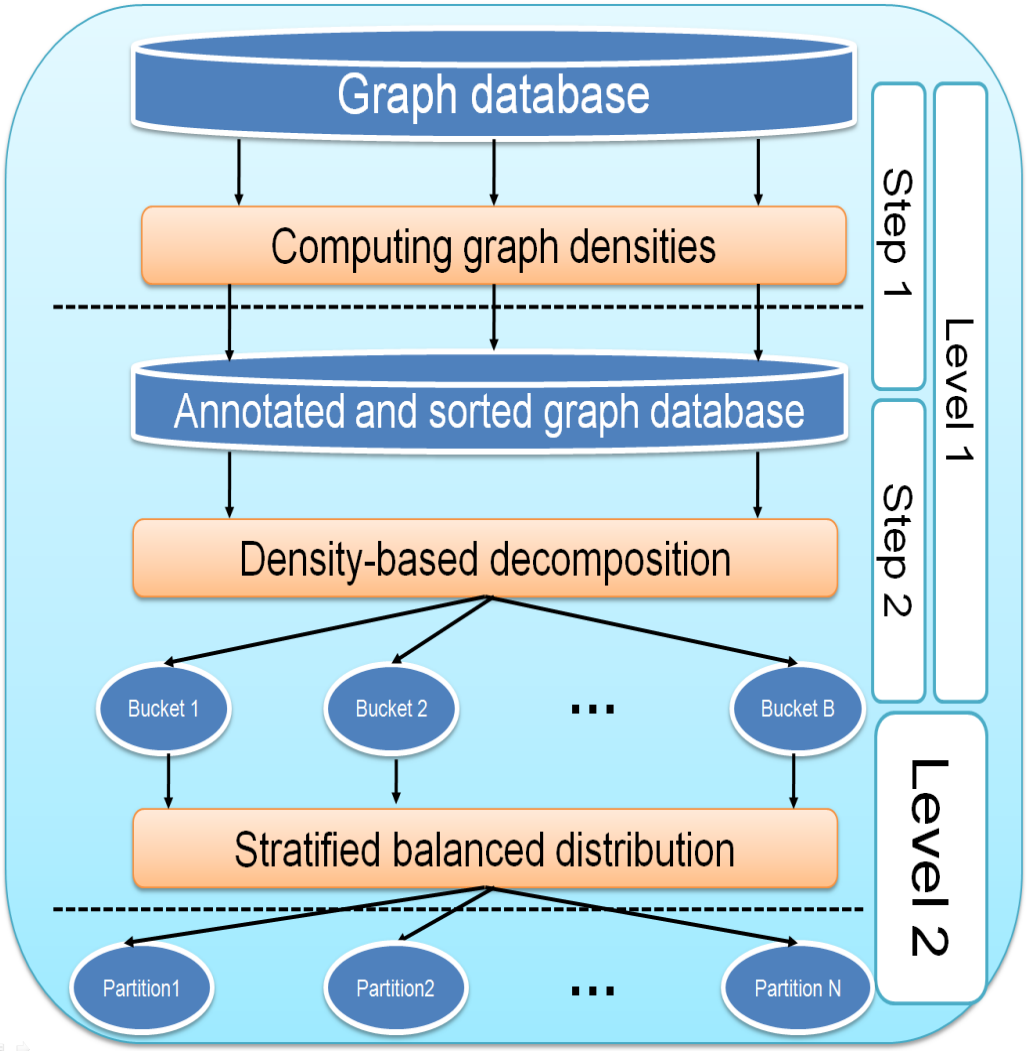
MRGP

- 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.

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

- 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.

Outline

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]

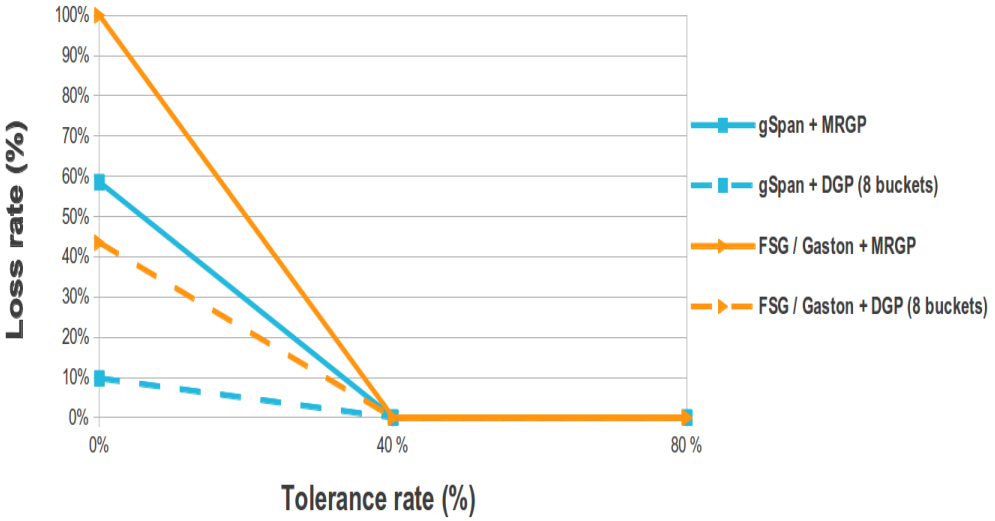
Experiments

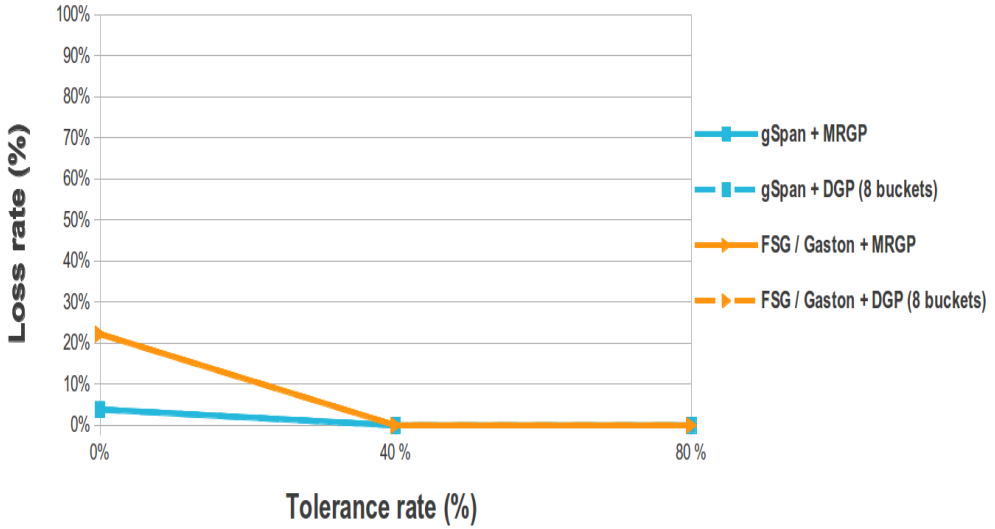
Experimental protocol

Three types of experiments:

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

Experiments: Quality

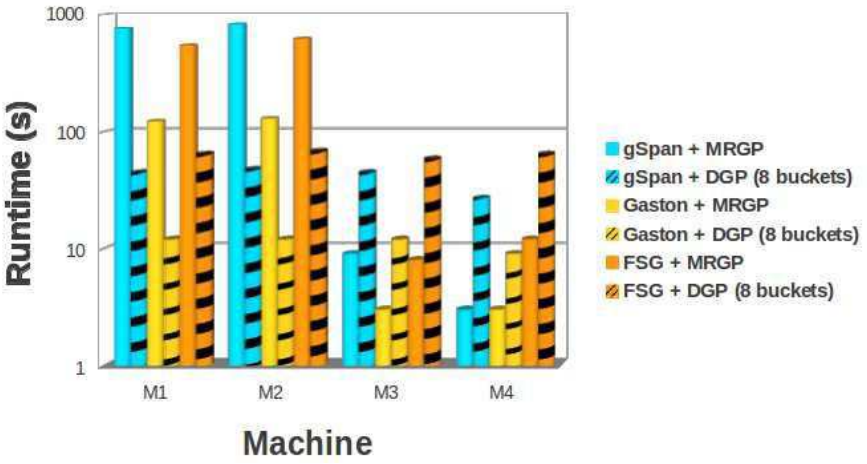


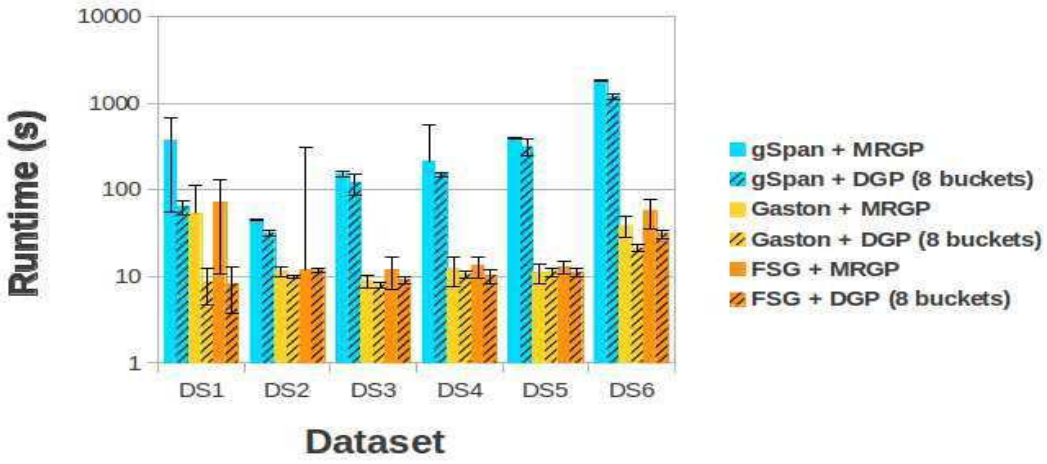


Result quality

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

Experiments: Load balancing and execution time

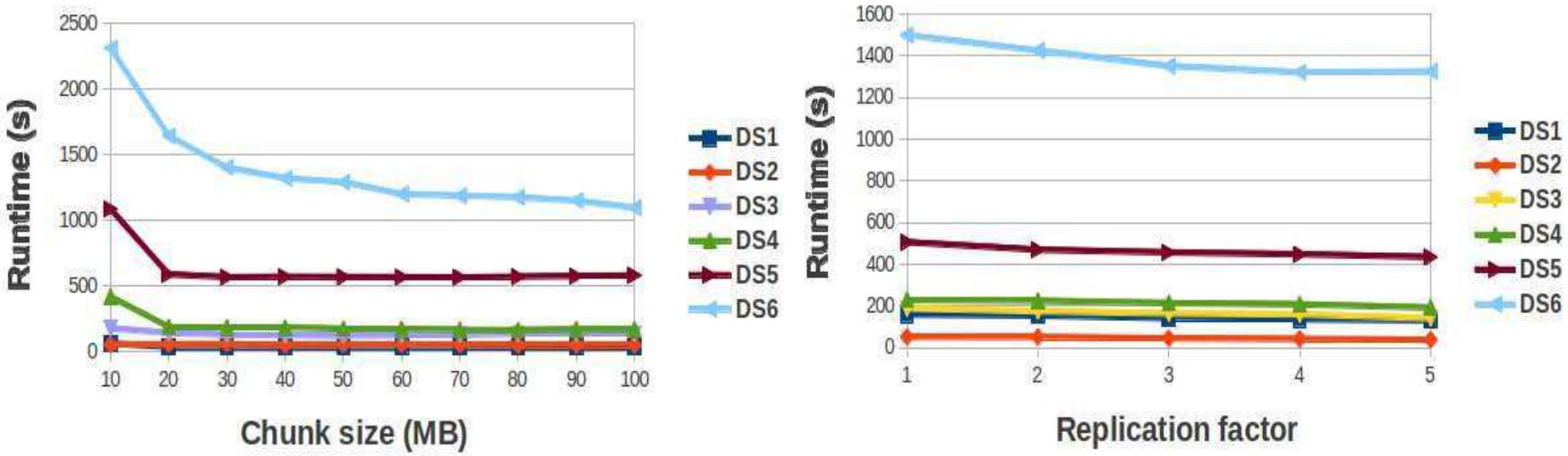




Runtime and workload distribution

- 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

Outline

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

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?

Thank You!