Towards a property graph generator for benchmarking

Arnau Prat-Pérez
Joan Guisado-Gámez
Xavier Fernández-Salas

Davide Basilio Bartolini
Siegfried Depner
Petr Koupy

Why a property graph generator?

- Graph-based analysis is becoming more and more popular

Why a property graph generator?

- For the field to advance, **many benchmarking initiatives** have appeared

LDBC
Social Network Benchmark

LDBC
Graphalytics

gMark

SWAT
LUBM

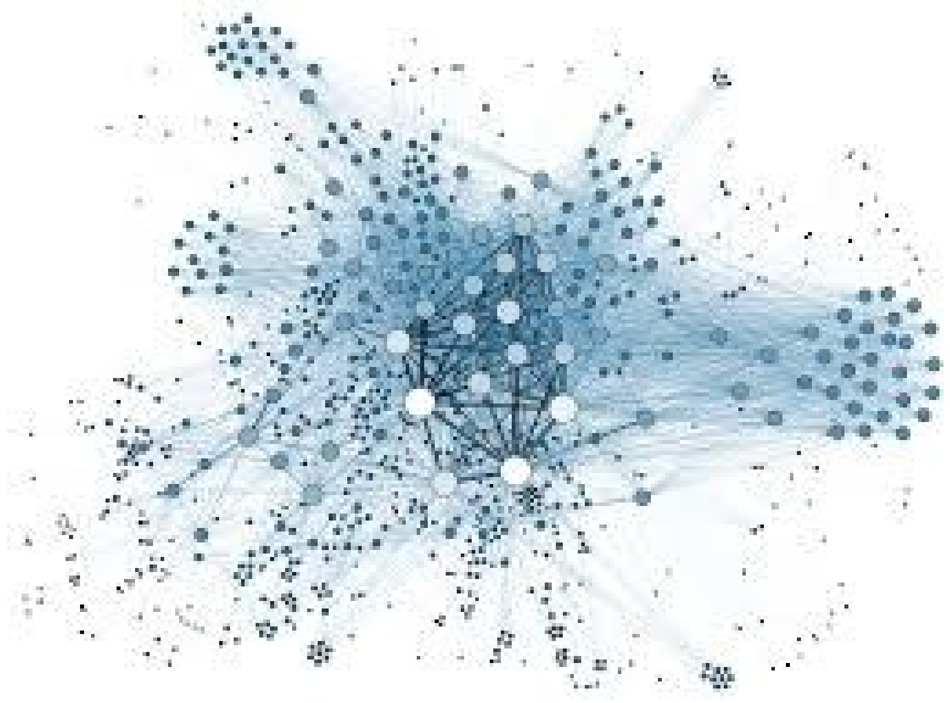
GAP

GRAPH 500

LinkBench

Why a property graph generator?

- Benchmarks need datasets, preferably **real ones**



Why a property graph generator?

- But ...

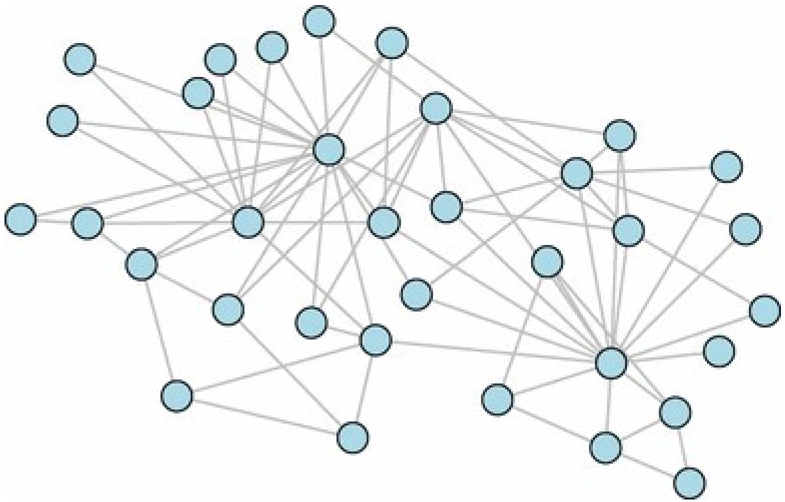
Why a property graph generator?

- But ...

OR

Why a property graph generator?

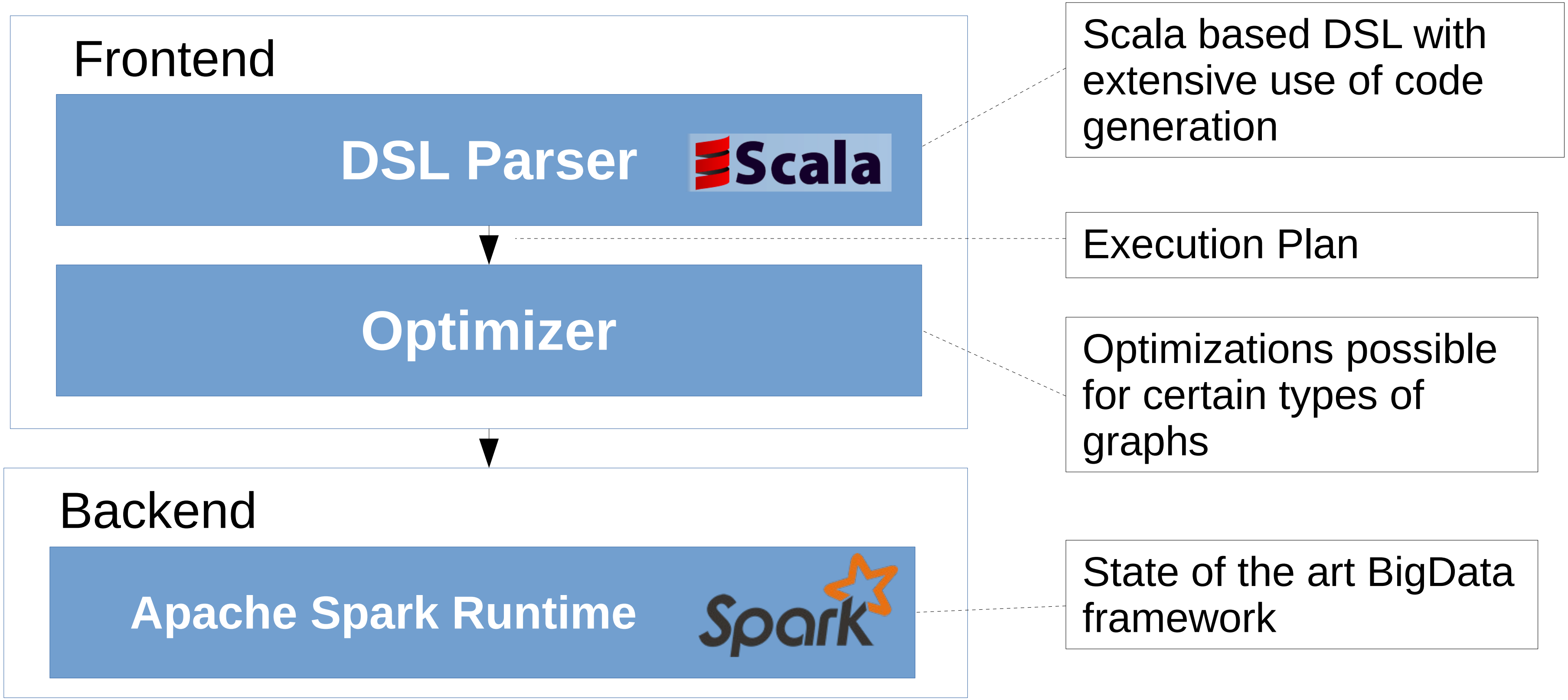
- Synthetic graph generators
- However, each benchmark has specific data needs
  - each benchmark designer implements its own
  - **time consuming task** sometimes **reinventing the wheel**



Why a property graph generator?

- Tool that, **given some "graph specification", produces a synthetic graph with the specified characteristics**
- DataSynth
  - https://github.com/DAMA-UPC/DataSynth
  - Written in Scala
  - Uses Apache Spark

Architecture Overview

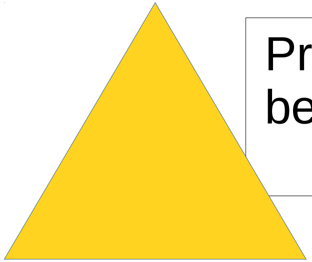


What features should DataSynth have?

- But what characteristics should a property graph generator be able to reproduce?

What features should DataSynth have?

- But what characteristics should a property graph generator be able to reproduce?



Properties and correlations/dependencies between them

- e.g. name is correlated with country

What features should DataSynth have?

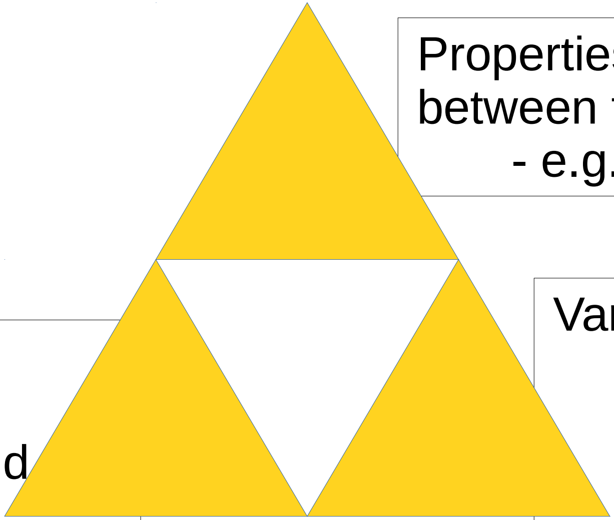
- But what characteristics should a property graph generator be able to reproduce?

Properties and correlations/dependencies between them
- e.g. name is correlated with country

Variate Structure
- degree distributions
- community structure
- low diameter
- large connected component
- etc.

What features should DataSynth have?

- But what characteristics should a property graph generator be able to reproduce?



Properties and correlations/dependencies between them
- e.g. name is correlated with country

Property-Structure correlations/depencies
- e.g. Chinese people tend connect to Chinese people
- represented as a P(X,Y) of observing X and Y on a randomly picked edge.

Variate Structure
- degree distributions
- community structure
- low diameter
- large connected component
- etc.

What features should DataSynth have?

- But what characteristics should a property graph generator be able to reproduce?

Properties and correlations/dependencies between them
- e.g. name is correlated with country

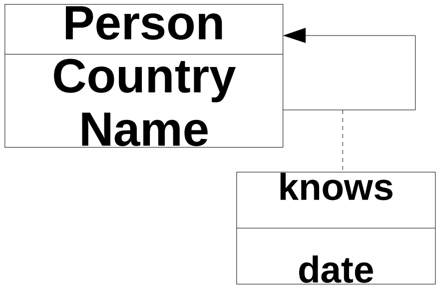
Property-Structure correlations/depencies
- e.g. Chinese people tend connect to Chinese people
- represented as a P(X,Y) of observing X and Y on a randomly picked edge.

Variate Structure
- degree distributions
- community structure
- low diameter
- large connected component
- etc.

But...

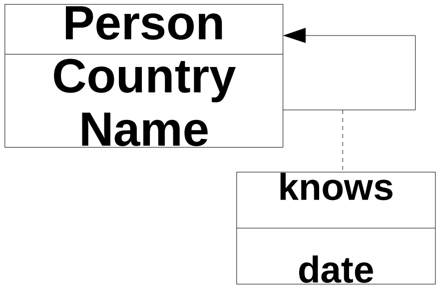
- Having a single algorithm for generating so many things seems too complex
  - Properties and property correlations
  - Ralistic graph structure
  - Property-structure correlations
- There are tens of metrics to measure the structure of a graph, which ones to take (which possibly depend on the algorithms used)?

DataSynth's approach



TIME

DataSynth's approach

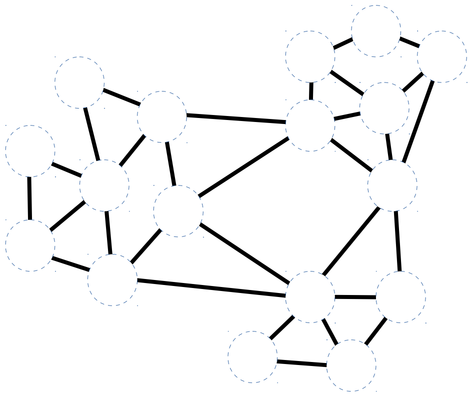


node property generation

Id	Country
1	China
2	Japan
3	China
...	...
17	Germany

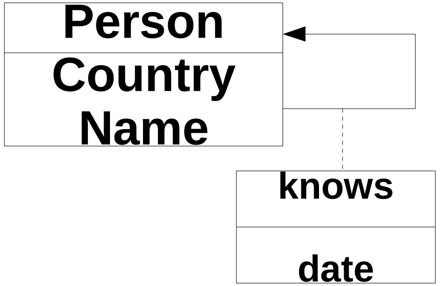
Id	Name
1	Lee
2	Hiroshi
3	Yang
...	...
17	Wolfgang

structure generation



TIME

DataSynth's approach

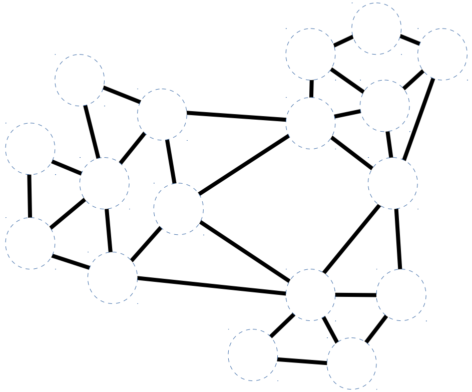


node property generation

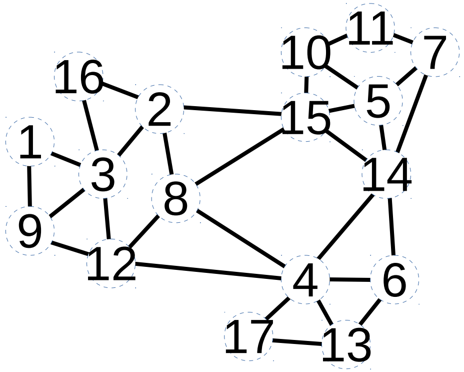
Id	Country
1	China
2	Japan
3	China
...	...
17	Germany

Id	Name
1	Lee
2	Hiroshi
3	Yang
...	...
17	Wolfgang

structure generation



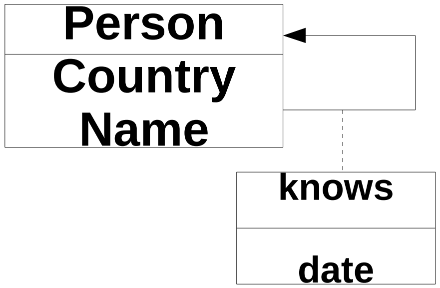
Matching preserving given joint probability distributions



e.g. P(China,China) ≈ 0.2

TIME

DataSynth's approach

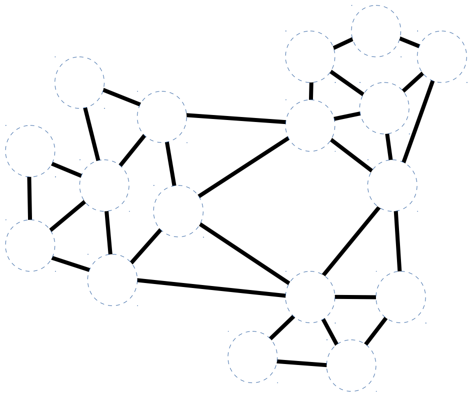


node property generation

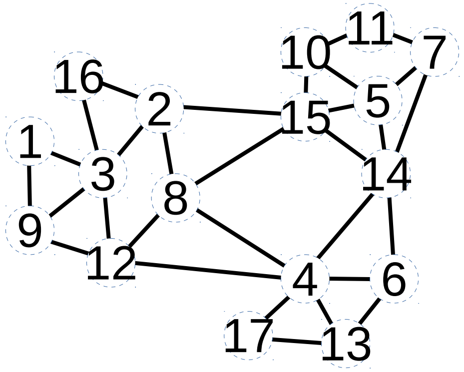
Id	Country
1	China
2	Japan
3	China
...	...
17	Germany

Id	Name
1	Lee
2	Hiroshi
3	Yang
...	...
17	Wolfgang

structure generation



Matching preserving given joint probability distributions



e.g. P(China,China) ≈ 0.2

edge property generation

Id	date
1	30/01/2015
2	4/06/2016
3	12/11/2016
...	...
30	03/03/2017

TIME

DataSynt's Approach

- Pros:
  - Accurate distributions of property values and correlations between properties
  - Does not limit us to a single way of generating the structure of a graph
    - We can use existing techniques and let the door open to new contributions
  - Pay for what we get
- Cons:
  - Heavy relies on a sophisticated matching approach to achieve accurate property-structure correlation

Property Generation

- We have a “Property Table” for each <type,property> pair
- We use a similar technique to that proposed by Myriad [1]
  - Highly parallel
  - Allows in-place data generation
    - Given and **Id** of an entity, I can generate its properties

[1] Alexander Alexandrov, Kostas Tzoumas, and Volker Markl. 2012. Myriad: scalable and expressive data generation. PVLDB 5, 12 (2012), 1890–1893.

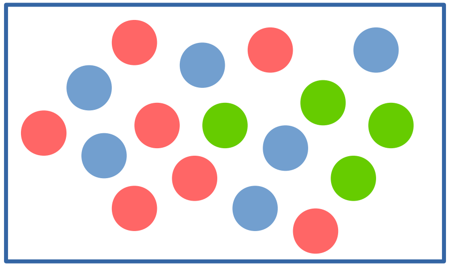
Structure Generation

- We can use existing scalable graph generation techniques: BTER [1], Darwini [2], etc.
- Hadoop implementation of BTER implemented:
  - https://github.com/DAMA-UPC/BTERonH

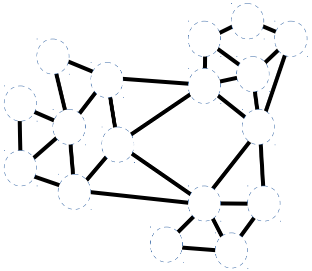
[1] Tamara G Kolda et al. 2014. A scalable generative graph model with community structure. SISC 36, 5 (2014), C424–C452.
[2] Sergey Edunov et al. 2016. Darwini: Generating realistic large-scale social graphs. arXiv:1610.00664 (2016)

Property-to-Structure Matching

Input

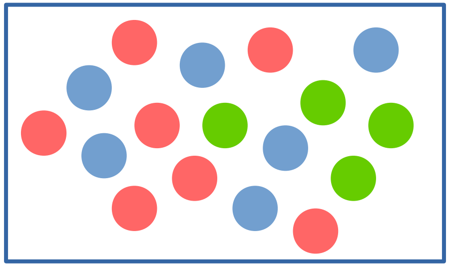


P(X,Y)			
	0.3	0.067	0.067
	0.067	0.33	0.067
	0.067	0.067	0.17



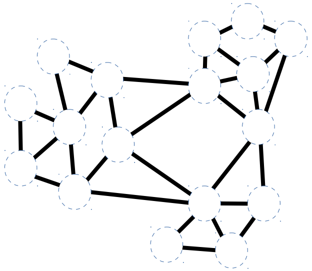
Property-to-Structure Matching

Input



P(X,Y)

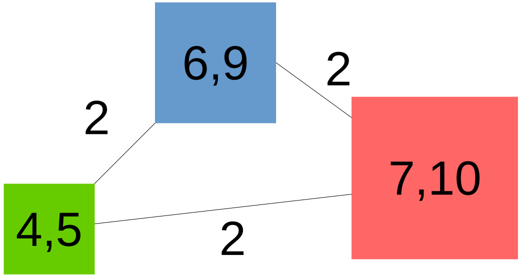
	0.3	0.067	0.067
	0.067	0.33	0.067
	0.067	0.067	0.17



Block Model

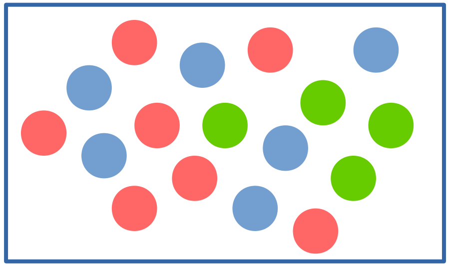
	6
	7
	4

	9	2	2
	2	10	2
	2	2	5



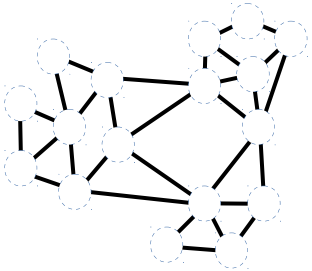
Property-to-Structure Matching

Input



P(X,Y)

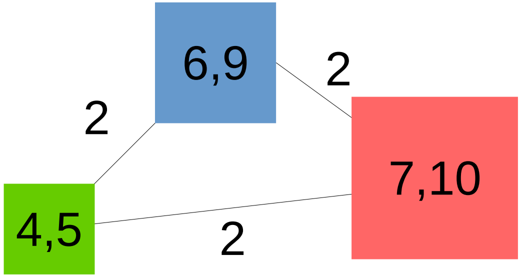
	0.3	0.067	0.067
	0.067	0.33	0.067
	0.067	0.067	0.17



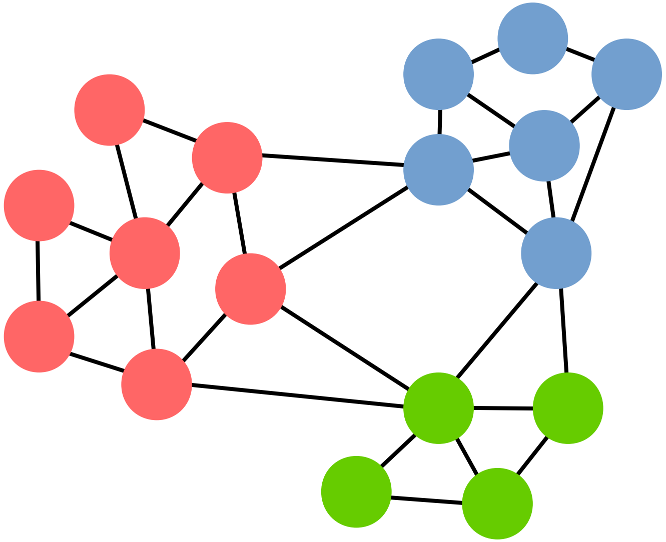
Block Model

	6
	7
	4

	9	2	2
	2	10	2
	2	2	5



Graph Partitioning



Next Steps

- Investigate further on the performance/quality of our Matching approach
  - Multithreaded/Distributed
  - Efficient for high-cardinality values
  - Understand when and when not works well
- Push for the DSL
- Integrate more existing structure generators
  - bi-partite graphs
- Long term: work towards “DGaaS” (Data Generation as a Service)