How to generate large-scale data from small-scale real-world data sets?

Gang Lu

Beijing Academy of Frontier Science and Technology

BigDataBench Tutorial, ASPLOS 2016
Atlanta, GA, USA

Motivation

- Benchmarking big data systems
  - The first thing is to obtain BIG data
- Obtaining REAL big data sets?
  - Large companies possess a lot of data
    - Confidentiality issue (User privacy)
  - Replicating big data sets is rather expensive

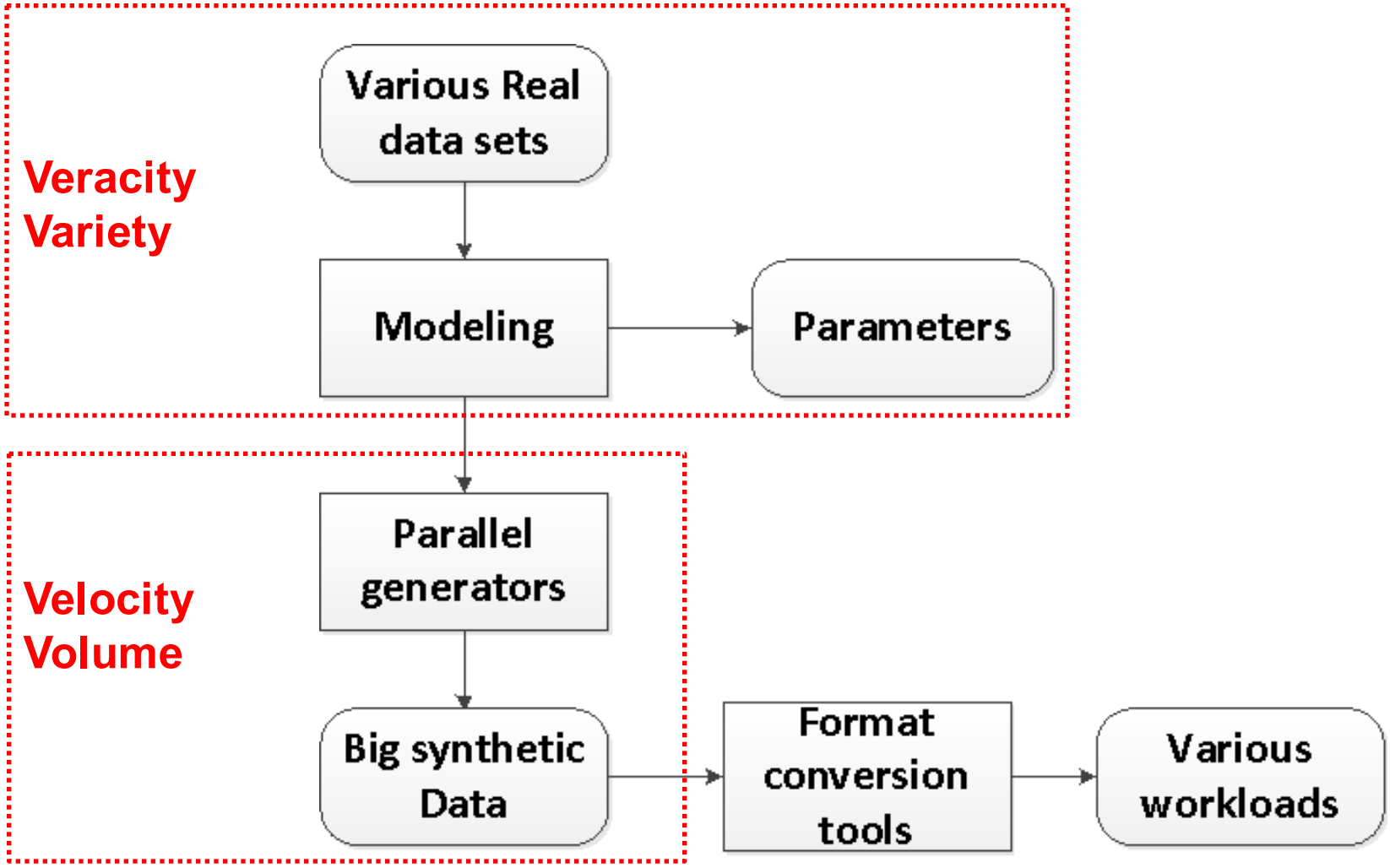
Is it possible to obtain large scale synthetic data?

Goals

- Generating synthetic data to satisfy 4V properties of big data
  - Volume
  - Velocity
  - Variety
  - Veracity

Big Data Generator Suite(BDGS)

Architecture of BDGS



Veracity and Variety

- From real world data, we can get:

Veracity	Variety		
Data set	Data type	Data source	Application domains
Wikipedia Entries	un-structured	text data	Search engine
Amazon Movie Reviews	semi-structure	text data	E-commence
Google Web Graph	un-structured	graph data	Search engine
Facebook Social Graph	un-structured	graph data	Social network
E-commence Transaction	structured	table data	E-commence
Profsearch Person Resume	semi-structured	table data	Search engine

Original size of real data sets

Use BDGS to scale up these data sets

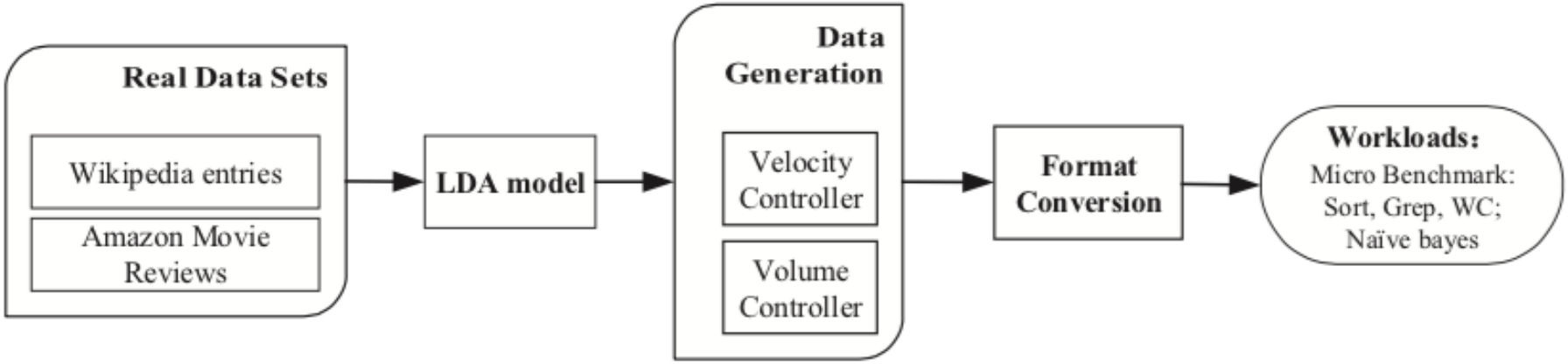
data sets	data size
Wikipedia Entries	4,300,000 English articles
Amazon Movie Reviews	7,911,684 reviews
Google Web Graph	875713 nodes, 5105039 edges
Facebook Social Network	4039 nodes, 88234 edges
E-commence Transaction	table1: 4 columns, 38658 rows. table2: 6 columns, 242735 rows
Person Resumes Data	278956 resumes

What does BDGS provide?

- Text generator
- Graph generator
- Table generator

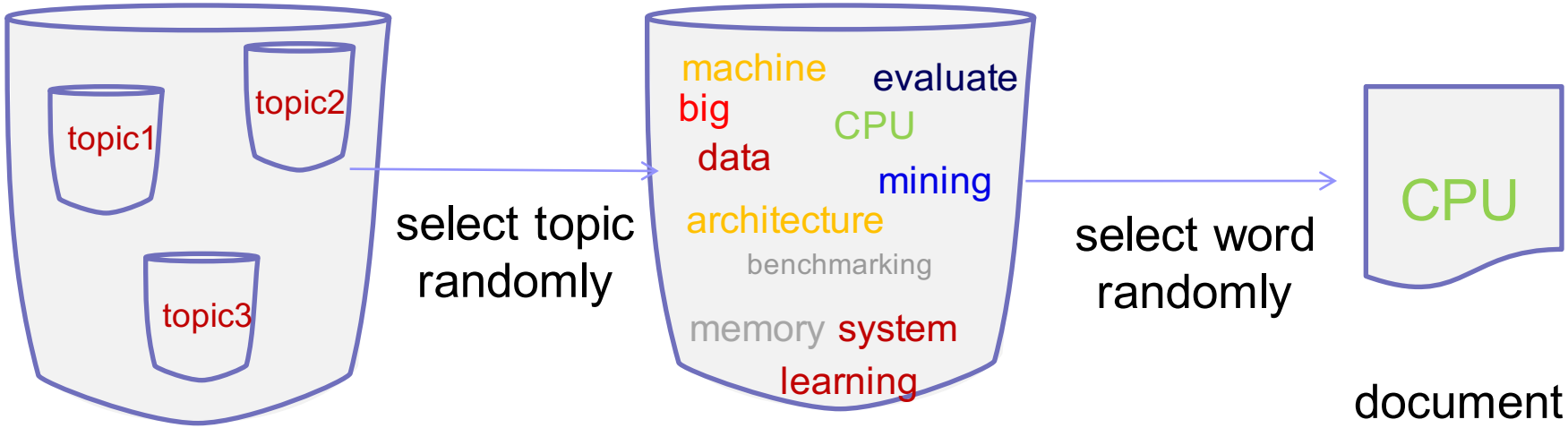
Text generator

- Use LDA (Latent Dirichlet Allocation) (David M Blei, et al.) to generate text corpus.
  - Topic model
    - To model the information of semantic level
  - Widely used in machine learning and natural language processing



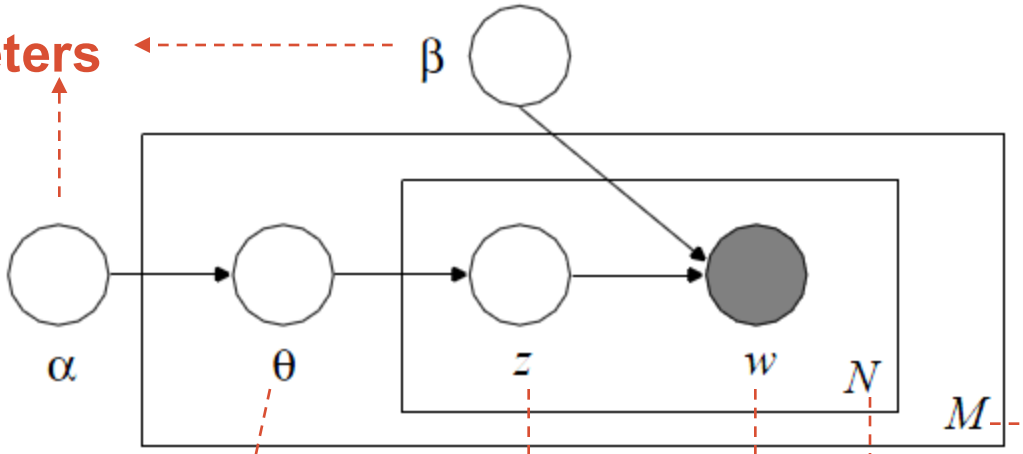
Text generator

- **How to generate a new document**



Progress of generating a new document

Latent dirichlet allocation



Hyper parameters

Number of documents in corpus

Topic proportion

Topic

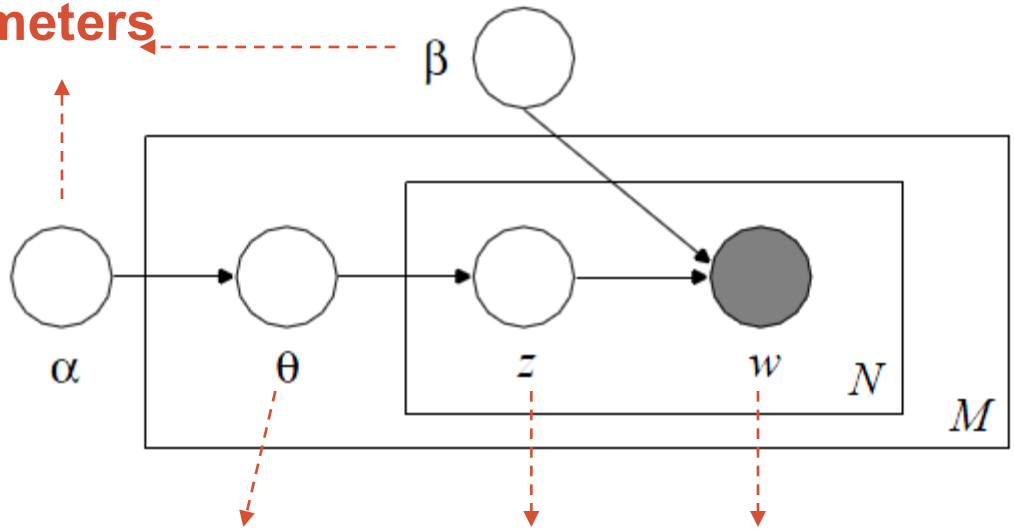
Word

Document length

David M Blei, et al., "Latent dirichlet allocation," the Journal of machine Learning research, vol. 3, pp. 993–1022, 2003.

Latent dirichlet allocation

Hyper parameters



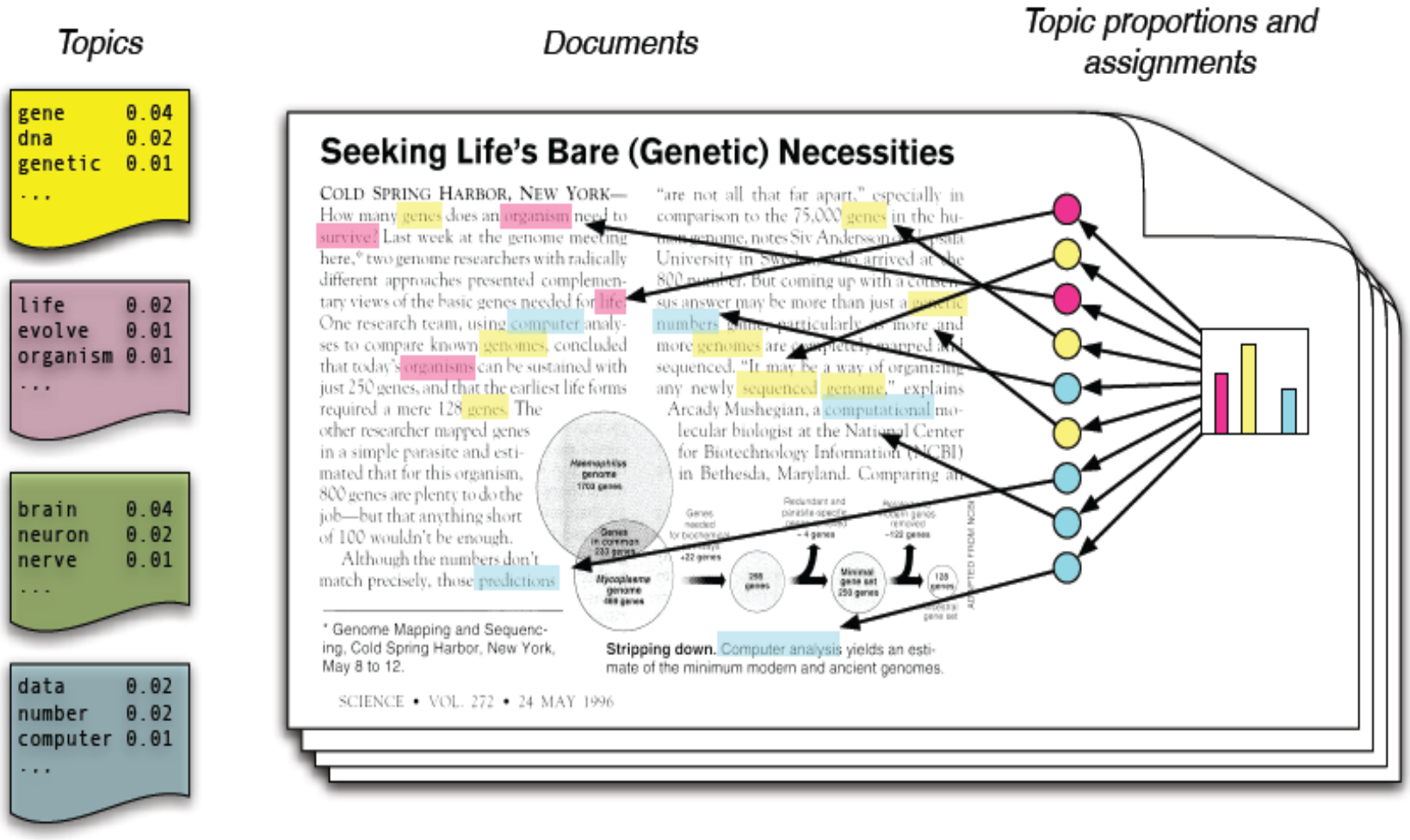
Three-level hierarchical Bayesian model

Dirichlet distribution

Multinomial distribution

$$p(\theta, \mathbf{z}, \mathbf{w} \mid \alpha, \beta) = p(\theta \mid \alpha) \prod_{n=1}^{N} p(z_n \mid \theta) p(w_n \mid z_n, \beta)$$

David M Blei, et al., “Latent dirichlet allocation,” *the Journal of machine Learning research*, vol. 3, pp. 993–1022, 2003.



We can use expectation-maximization algorithm to determine α and β

How to use it to generate texts?

- Go into the directory of *BigDataGeneratorSuite*

```
gen_text_data.sh
  <model name>
  <number of files>
  <number of lines>
  <number of words>
  <output dir >
```

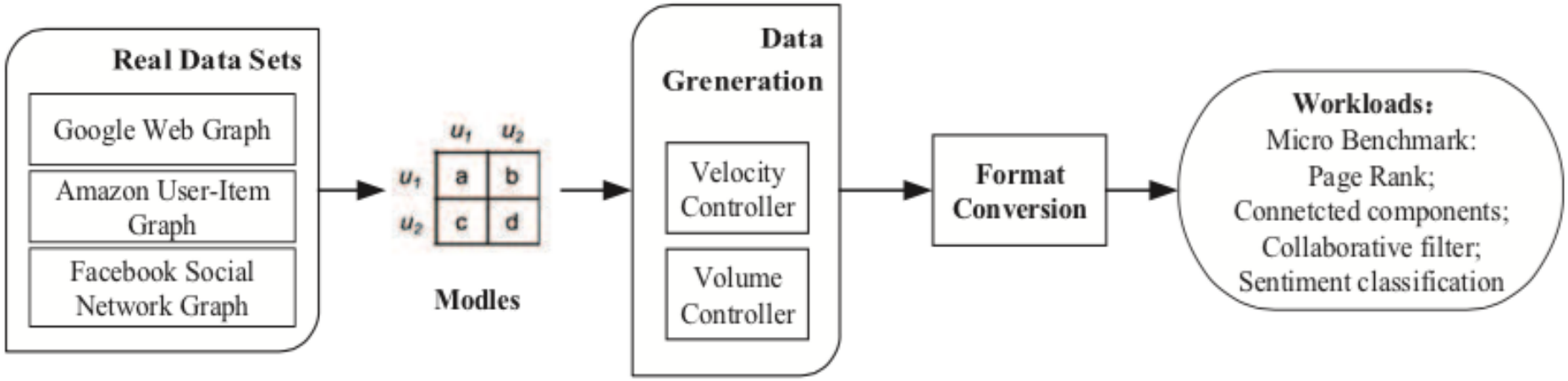
Parameters	Explannation
model name	the name of model used to generate new data (lda_wiki1w or amazonMR)
No. of files	the number of files to be generated
No. of lines	number of lines in each file
No. of words	number of words in each line
Output dir	output director

- An example
  - sh gen_text_data.sh *lda_wiki1w 10 100 1000 gen_data/*

Note: Installation of the GSL-GNU Scientific Library is needed.

Graph generator

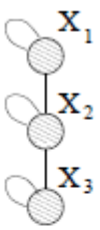
- Use the Stochastic Kronecker Graph model (Jure Leskovec,et al.) to generate graph
  - Used also by Graph 500
  - Different from Graph 500, our graph is application specific, the initiator is obtained from real representative data set of specific applications.

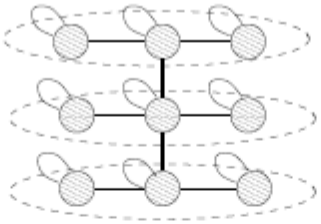


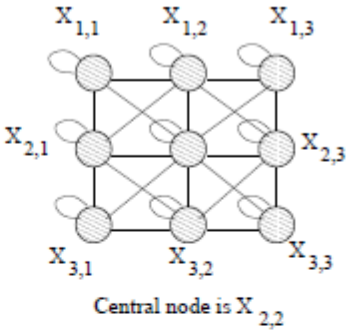
Deterministic Kronecker Graph

1: has edge

0: no edge







(a) Graph K_1 (b) Intermediate stage (c) Graph $K_2 = K_1 \otimes K_1$

1	1	0
1	1	1
0	1	1

self similar

K_1	K_1	0
K_1	K_1	K_1
0	K_1	K_1

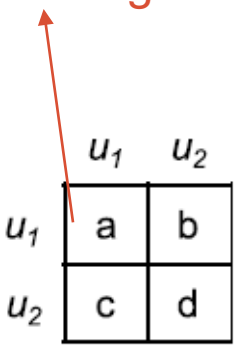
(d) Adjacency matrix of K_1

(e) Adjacency matrix of $K_2 = K_1 \otimes K_1$

Jure Leskovec,et al. , “Kronecker graphs: An approach to modeling networks,” *The Journal of Machine Learning Research*, vol. 11, pp. 985–1042, 2010.

Stochastic Kronecker Graph

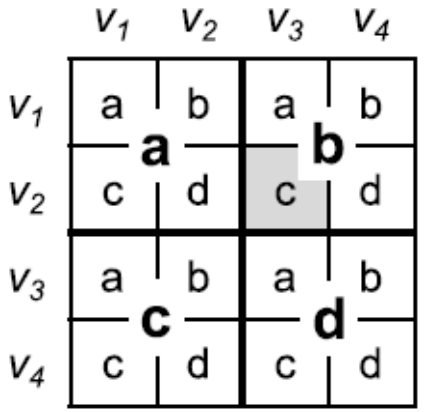
The probability with which the cell generate a edge



(a) 2×2 Stochastic Kronecker initiator $\mathcal{P}_1$

	v_1	v_2	v_3	v_4
v_1	a·a	a·b	b·a	b·b
v_2	a·c	a·d	b·c	b·d
v_3	c·a	c·b	d·a	d·b
v_4	c·c	c·d	d·c	d·d

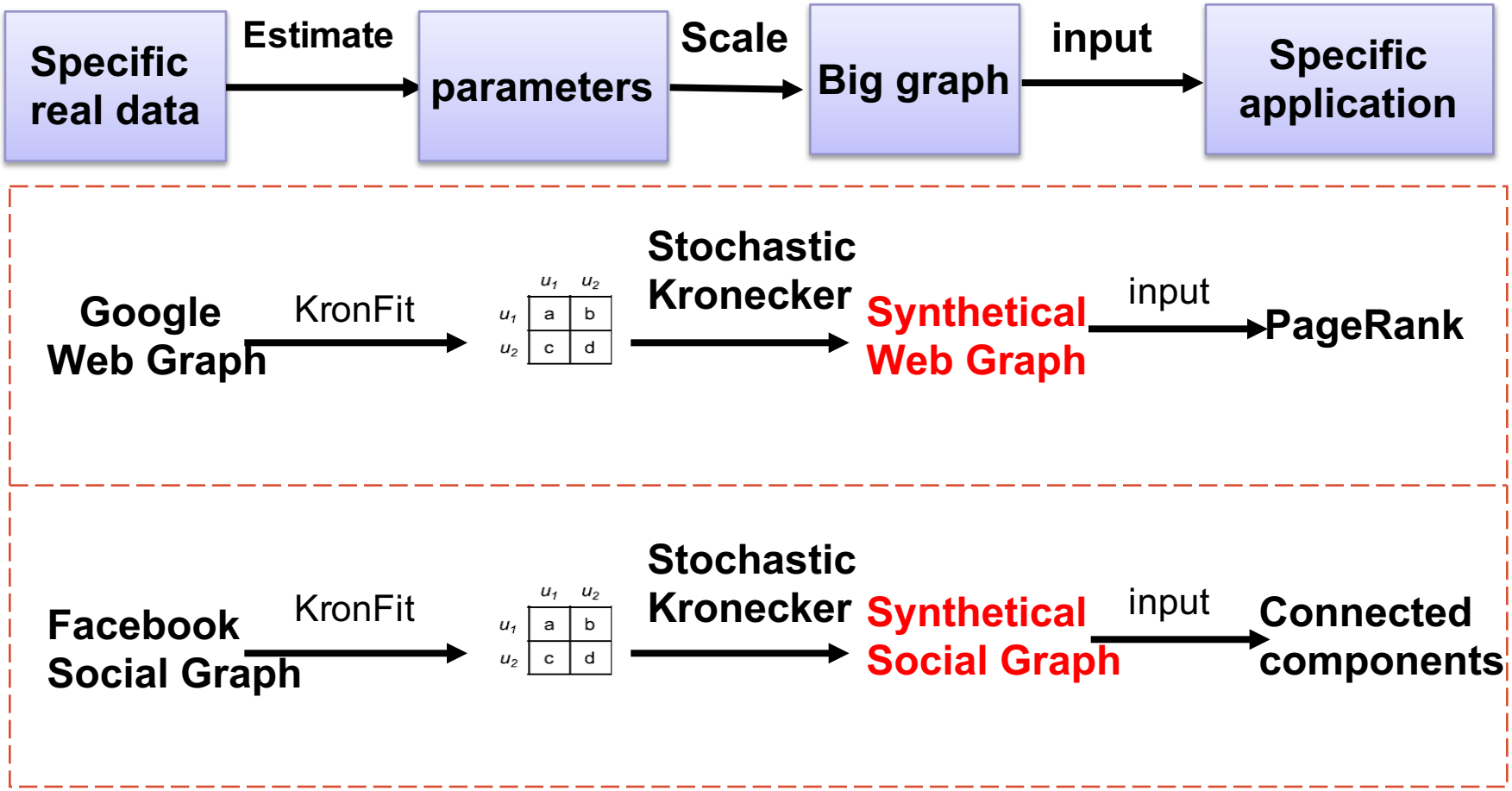
(b) Probability matrix $\mathcal{P}_2 = \mathcal{P}_1 \otimes \mathcal{P}_1$



(c) Alternative view of $\mathcal{P}_2 = \mathcal{P}_1 \otimes \mathcal{P}_1$

Jure Leskovec,et al. , “Kronecker graphs: An approach to modeling networks,” *The Journal of Machine Learning Research*, vol. 11, pp. 985–1042, 2010.

Application-specific



How to use it to generate graphs?

- Go into the directory of *BigDataGeneratorSuite*

```
gen_kronecker_graph
  <output file>
  <matrix>
  <iteration>
  <random seed>
```

Parameters	Explannation
output file	output file name (default:'graph.txt')
matrix	Matrix (in Maltab notation) (default: '0.9 0.5; 0.5 0.1')
iteration	Iteration of using kronecker product (default: 5)
random seed	time seed of random algorithm (default: 0)

- An example
  - sh gen_kronecker_graph -o:../data-outfile/amazon_gen.txt -m:"0.7196 0.6313; 0.4833 0.3601" -i:23

Table generator

- Relational structured table
  - Parallel Data Generation Framework (Tilmann Rabl, et al.)
    - PDGF is also used by BigBench and TPC-DS
    - using XML configuration files for data description and distribution
- Semi-structured resumes
  - choose a mix of fields, each field follows bernoulli distribution

How to use it to generate tables?

- Go into the directory of *BigDataGeneratorSuite*

```
pdgf.jar
-l schema.xml
-l generation.xml
-sf 2000
```

Parameters	Explannation
schema.xml	the schema configuration: the structure of the data and the generation rules
generation.xml	the generation configuration defines the output and post-processing of generated data
sf	A multiple increase in the reference data base 100 000

- An example
  - java -XX:NewRatio=1 -jar pdgf.jar *-l demo-schema.xml -l demo- generation.xml -c -s -sf 2000*


Any
Questions