Compressed Linear Algebra for Large Scale Machine Learning

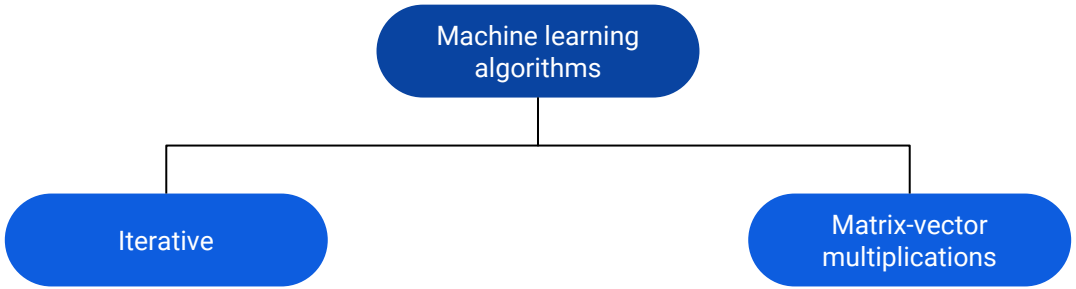
Ahmed Elgohary, Matthias Boehm, Peter J. Haas, Frederick R. Reiss, Berthold Reinwald

IBM Research - Almaden; San Jose, CA, USA
University of Maryland; College Park, MD, USA

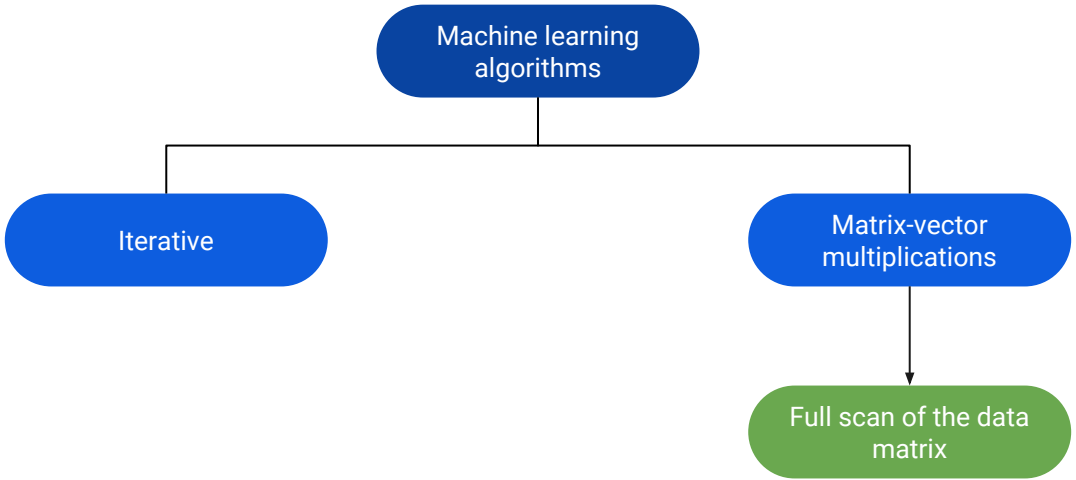
Presented by: Issa Memari

25/1/2018

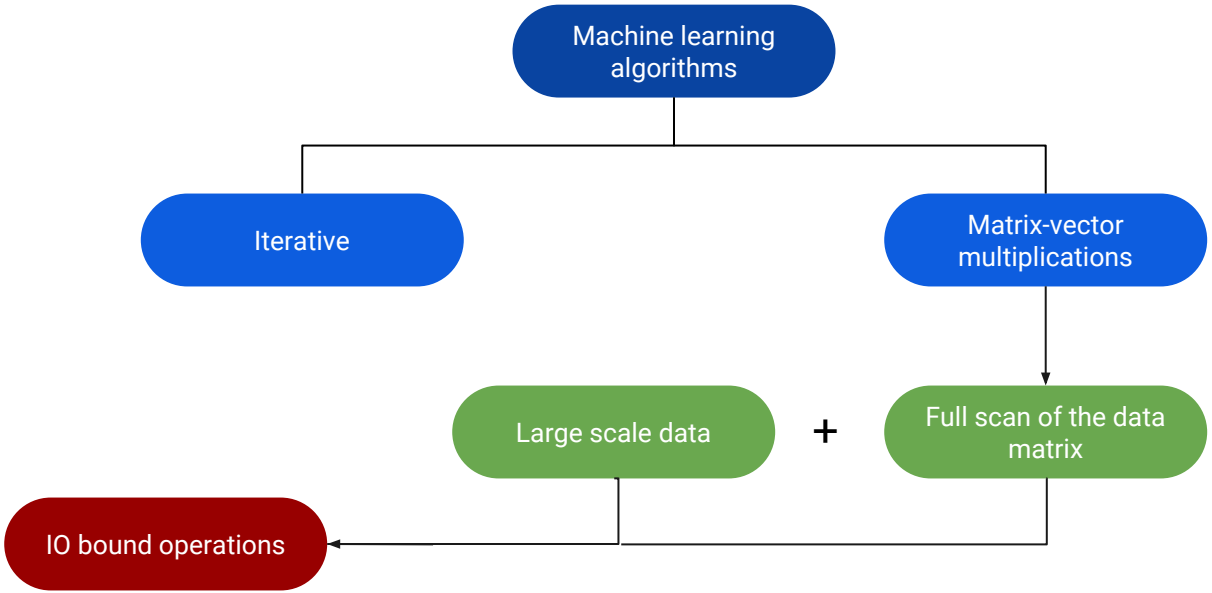
Motivation


Machine learning algorithms
Iterative
Matrix-vector multiplications

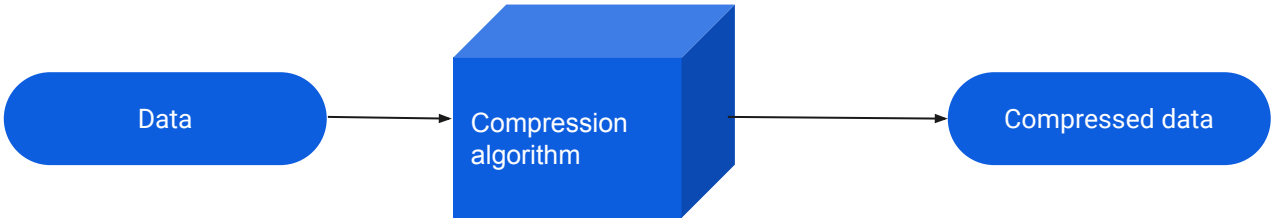
Motivation


Machine learning algorithms
Iterative
Matrix-vector multiplications
Full scan of the data matrix

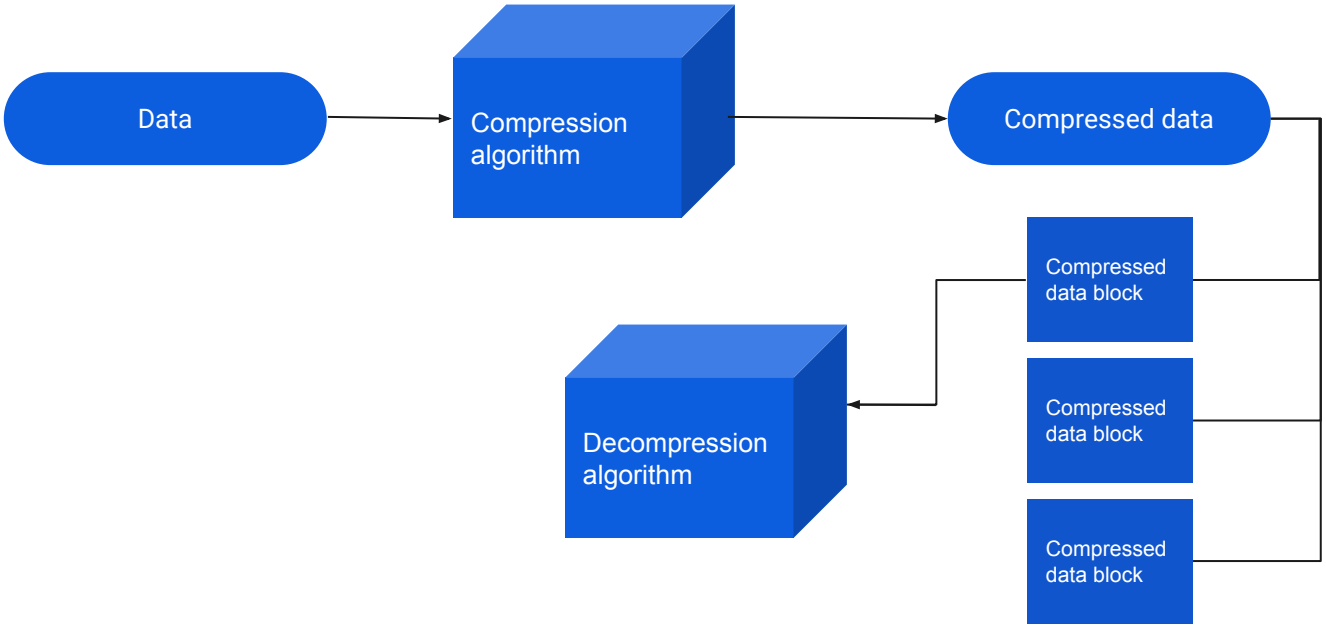
Motivation


Machine learning algorithms
Iterative
Matrix-vector multiplications
Large scale data
+
Full scan of the data matrix
IO bound operations

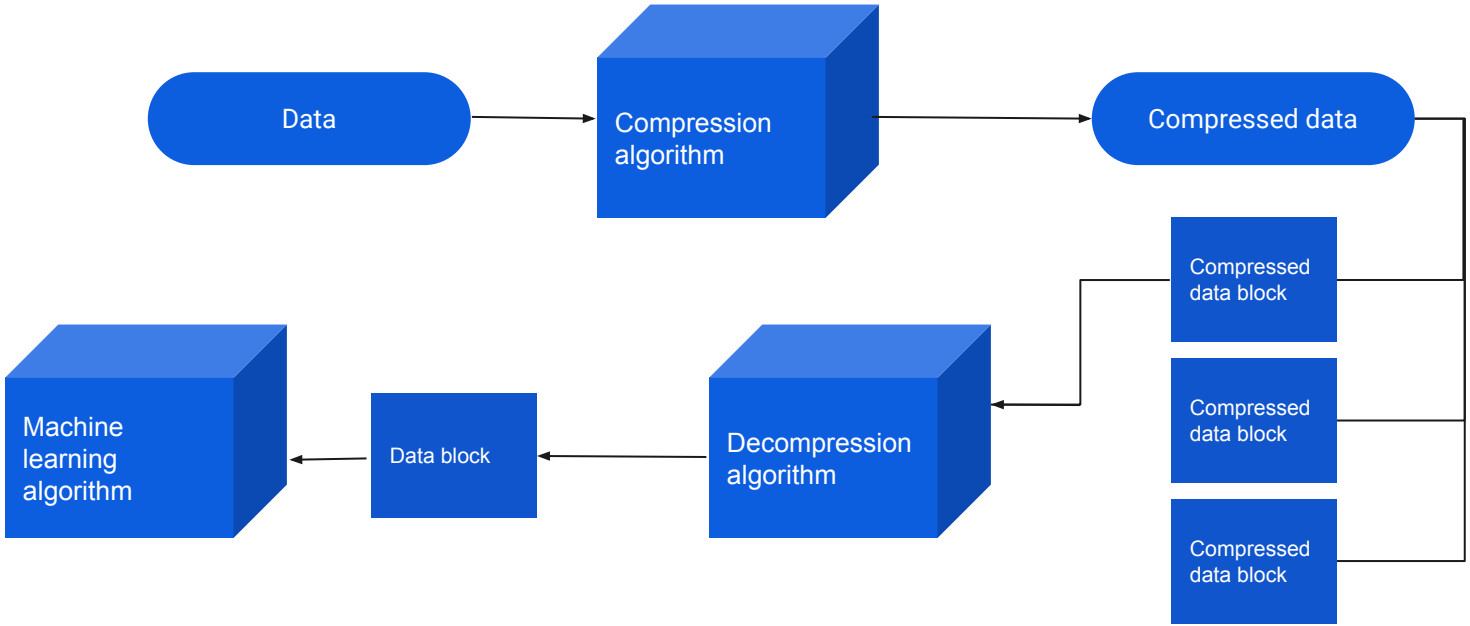
Solution: Fit more data into memory



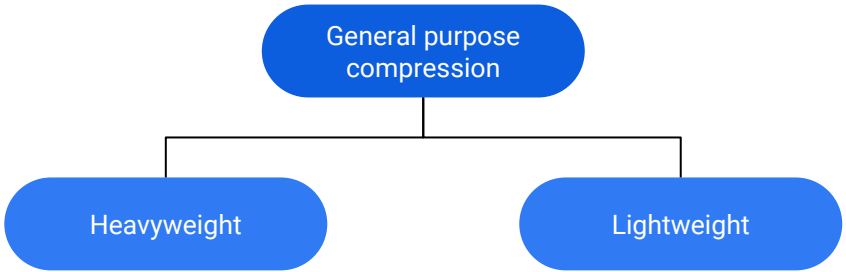
Solution: Fit more data into memory



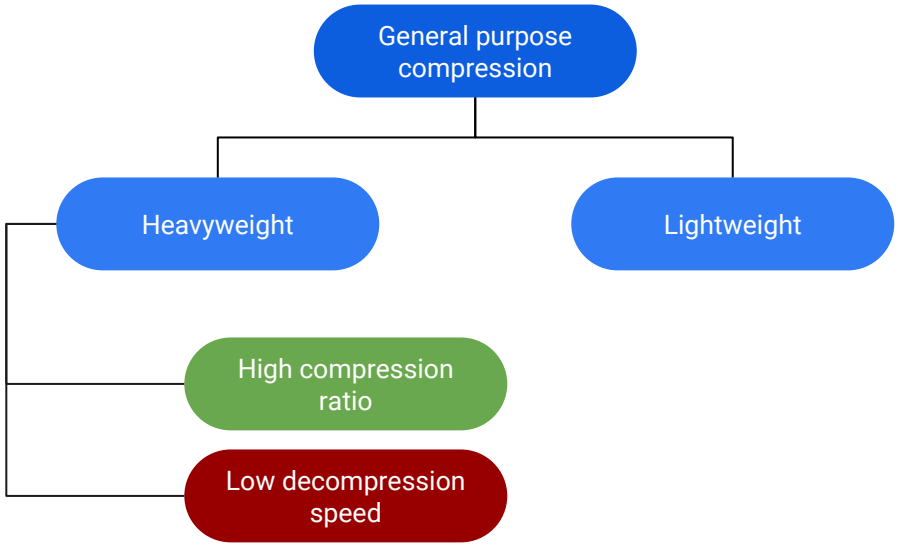
Solution: Fit more data into memory



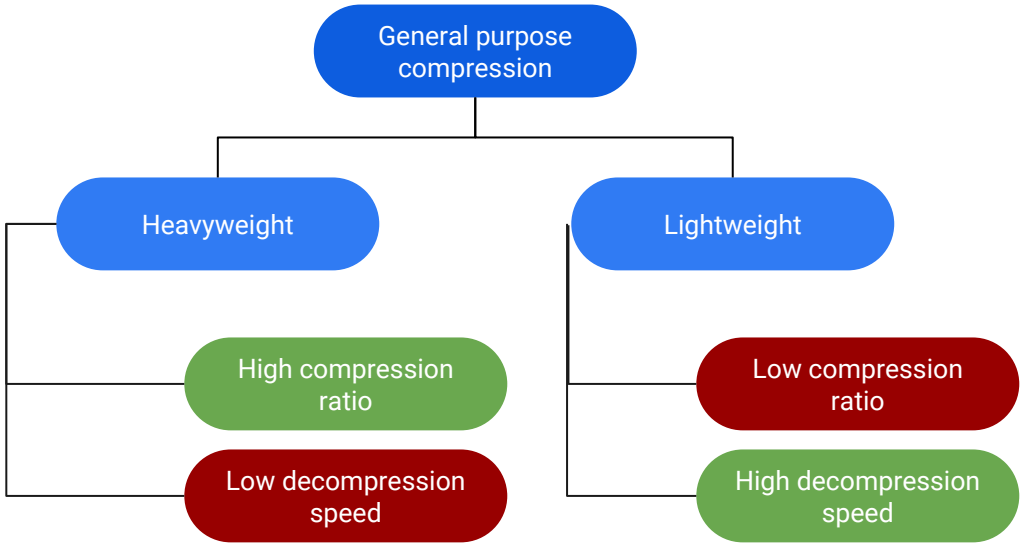
Compression techniques



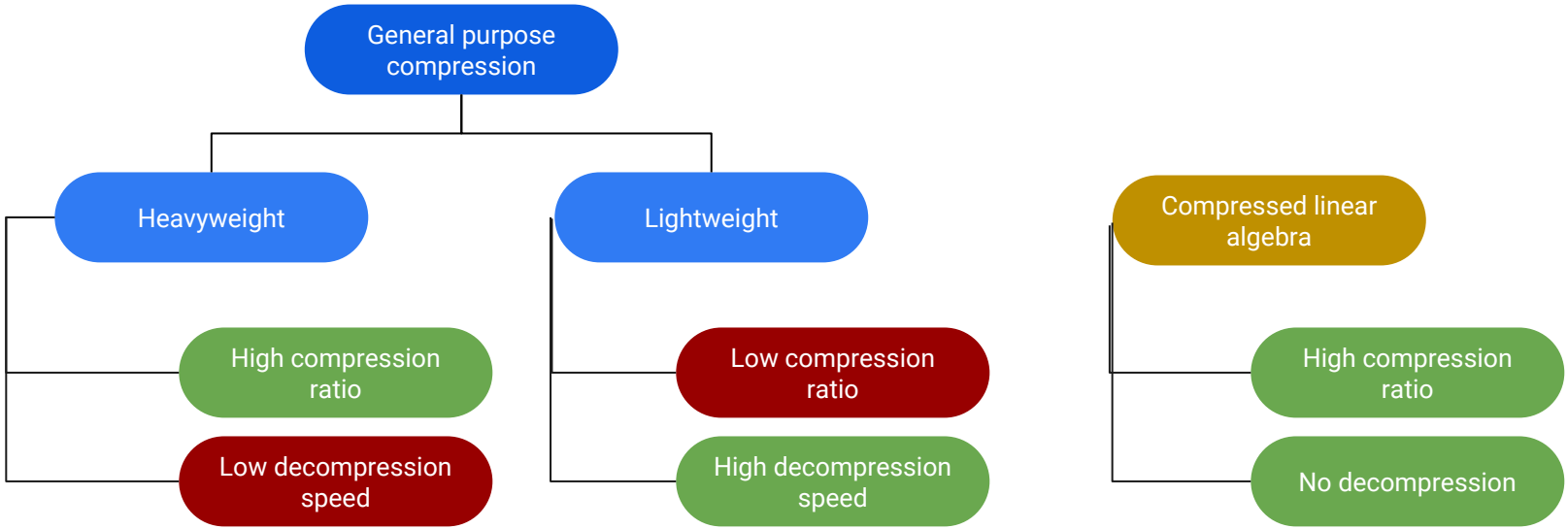
Compression techniques



Compression techniques

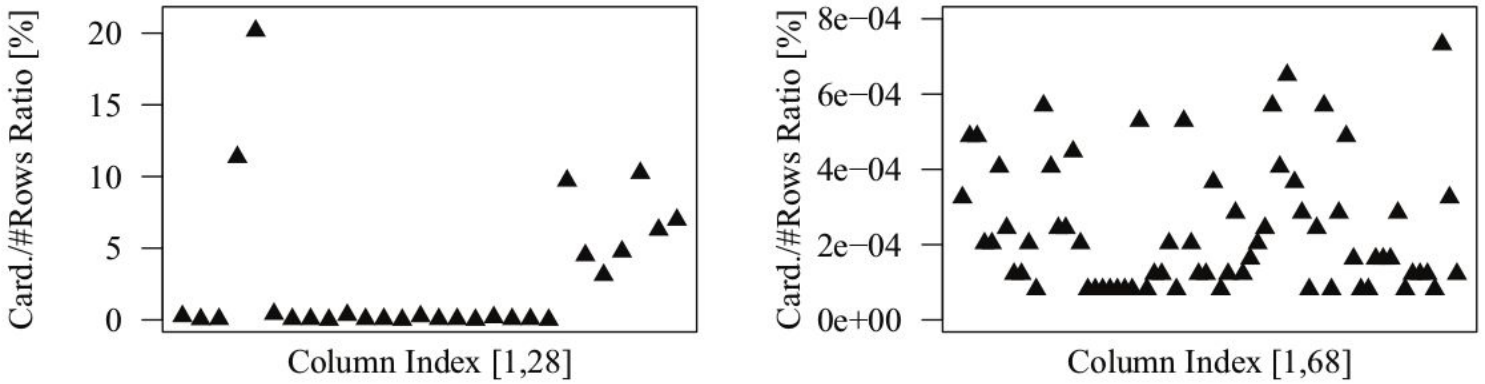


Compression techniques



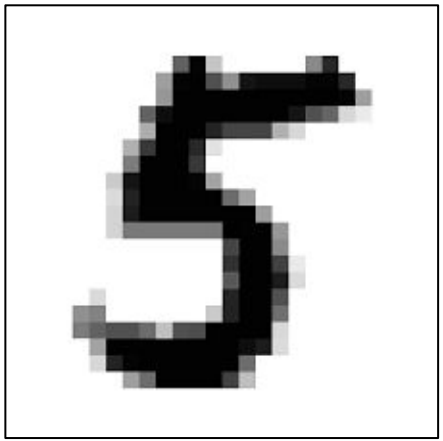
Column-wise compression: Motivation

1. Low column cardinalities



Column-wise compression: Motivation

2. Non-uniform sparsity across columns



Column-wise compression: Motivation

3. Tall and skinny matrices

Column encoding formats

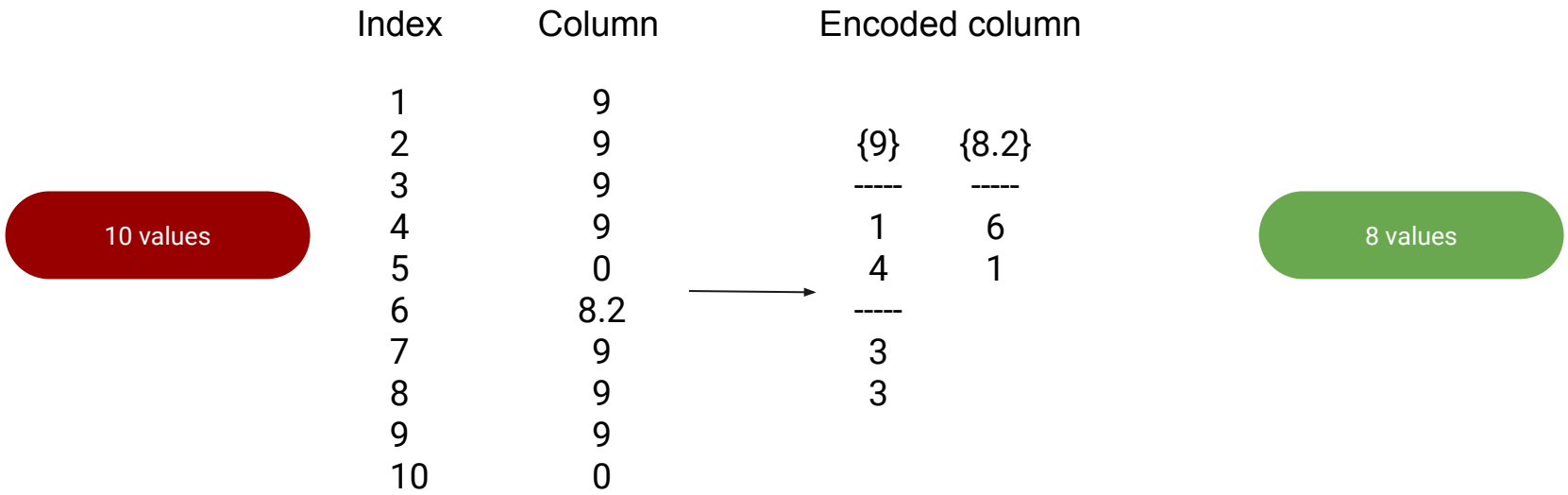
1. Run-Length Encoding (RLE)
2. Offset-List Encoding (OLE)
3. Uncompressed Columns (UC)

Column encoding formats: Run-Length Encoding

10 values

Index	Column
1	9
2	9
3	9
4	9
5	0
6	8.2
7	9
8	9
9	9
10	0

Column encoding formats: Run-Length Encoding



Column encoding formats: Offset-List Encoding

10 values

Index	Column
1	9
2	9
3	9
4	9
5	0
6	8.2
7	9
8	9
9	9
10	0

Column encoding formats: Offset-List Encoding

10 values

Index	Column
1	9
2	9
3	9
4	9
5	0
6	8.2
7	9
8	9
9	9
10	0

{9}	{8.2}
1	6
2	
3	
4	
7	
8	
9	

10 values ☹️

RLE vs. OLE

10 values

Index	Column
1	0
2	1.7
3	0
4	1.7
5	0
6	1.7
7	0
8	2
9	0
10	0

RLE vs. OLE

10 values

10 values

Index	Column
1	0
2	1.7
3	0
4	1.7
5	0
6	1.7
7	0
8	2
9	0
10	0

{1.7}	{2}
-----	-----
2	8
1	1

2	
1	

2	
1	

RLE vs. OLE

10 values

10 values

6 values

Index	Column
1	0
2	1.7
3	0
4	1.7
5	0
6	1.7
7	0
8	2
9	0
10	0

RLE encoded column

{1.7}	{2}
2	8
1	1
2	
1	
2	
1	

OLE encoded column

{1.7}	{2}
2	8
4	
6	

Column encoding formats: Uncompressed Columns

10 values

Index	Column
1	10
2	20
3	30
4	40
5	50
6	60
7	70
8	80
9	90
10	100

Column encoding formats: Uncompressed Columns

10 values

Index	Column	Encoded column
1	10	10
2	20	20
3	30	30
4	40	40
5	50	50
6	60	60
7	70	70
8	80	80
9	90	90
10	100	100

10 values

Column co-coding

Index	Column 1	Column 2
1	1.7	1.7
2	2	6
3	0	0
4	2	6
5	2	0
6	1.7	1.7
7	0	0
8	3	6
9	1.7	1.7
10	2	6

20 values

Column co-coding

16 values

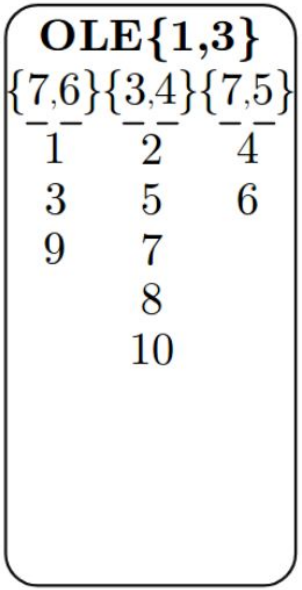
20 values

Index	Column 1	Column 2
1	1.7	1.7
2	2	6
3	0	0
4	2	6
5	2	0
6	1.7	1.7
7	0	0
8	3	6
9	1.7	1.7
10	2	6

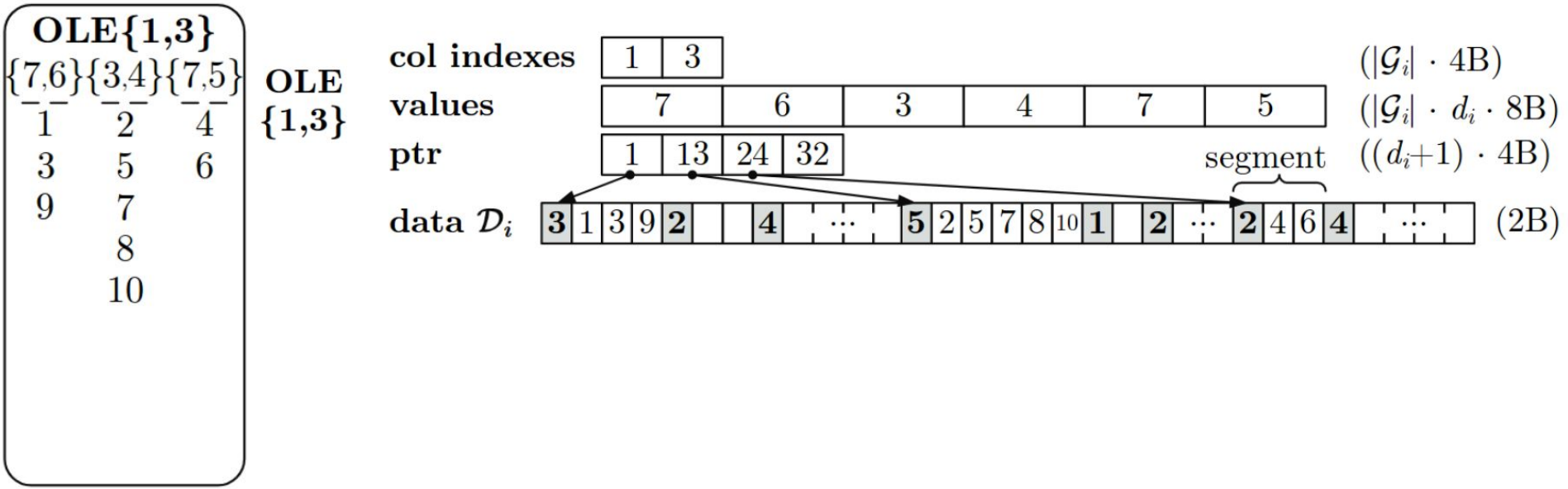
OLE encoded group

{1.7,1.7}	{2,6}	{2,0}	{3,6}
1	2	5	8
6	4		
9	10		

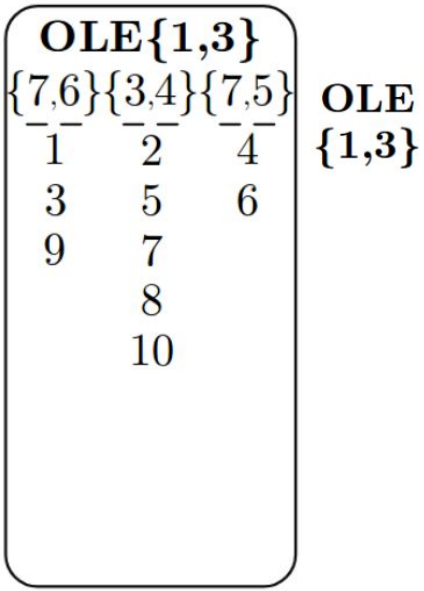
Data layout: OLE

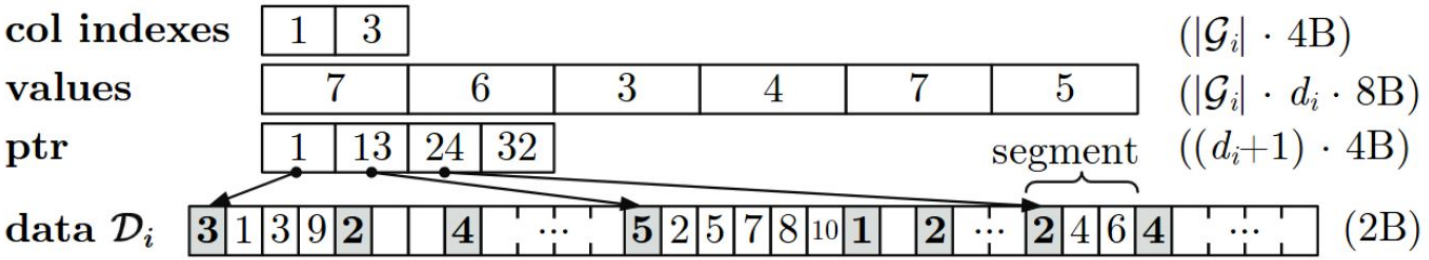


Data layout: OLE



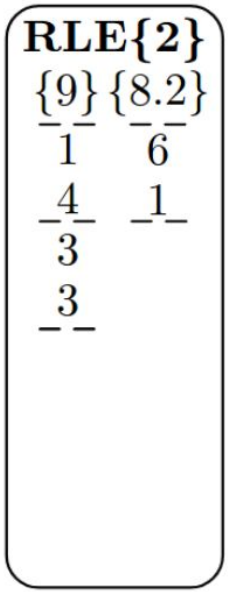
Data layout: OLE



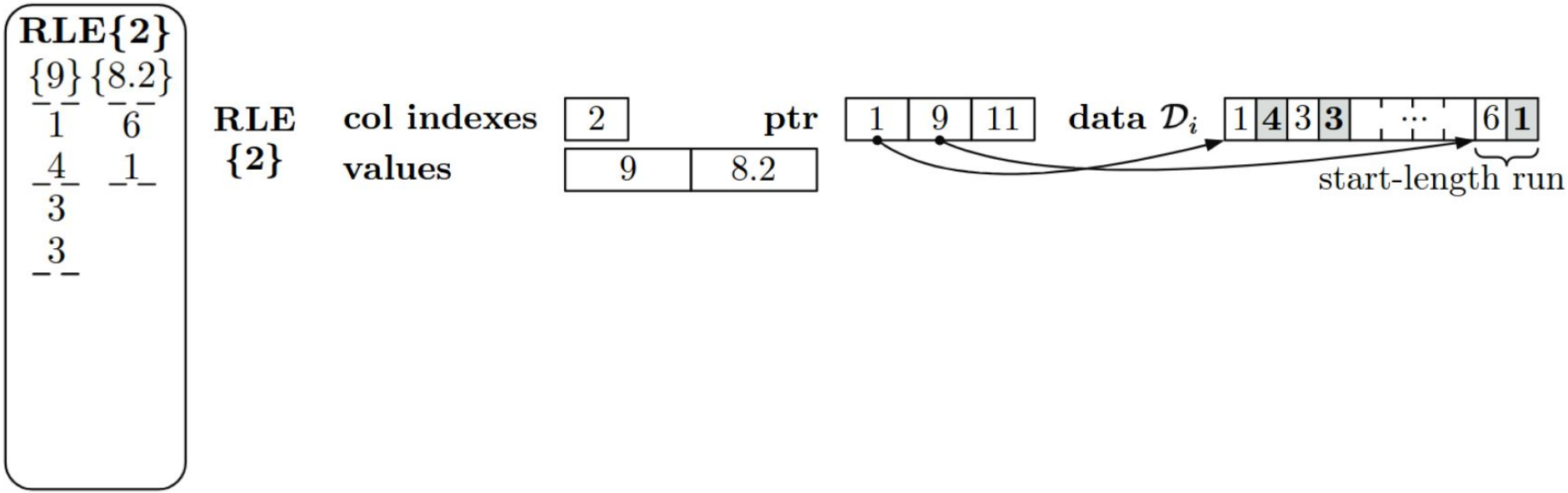


$$S_i^{\mathrm{OLE}} = 4|\mathcal{G}_i| + d_i\big(4 + \alpha|\mathcal{G}_i|\big) + 2\sum_{j=1}^{d_i} b_{ij} + 2z_i,$$

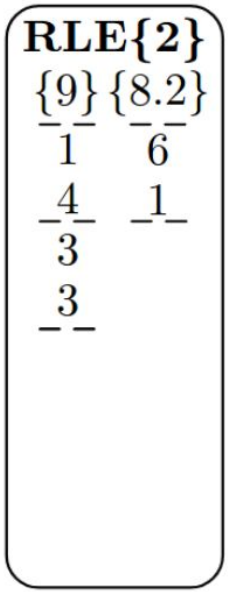
Data layout: RLE

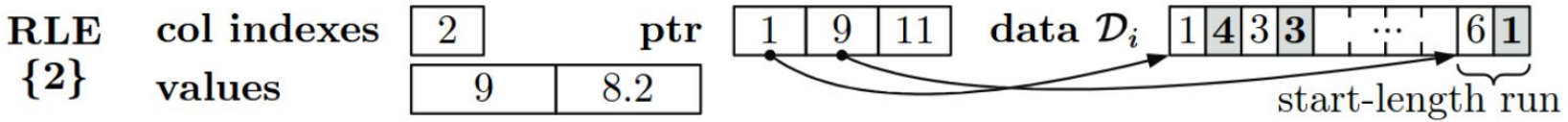


Data layout: RLE



Data layout: RLE





$$S_i^{\mathrm{RLE}} = 4|\mathcal{G}_i| + d_i\big(4 + \alpha|\mathcal{G}_i|\big) + 4\sum_{j=1}^{d_i} r_{ij}\,,$$

Compressed linear algebra: Matrix-vector multiplication

Example on the whiteboard

Compression planning

Compression planning involves three tasks:

1. Estimating column compression ratios
2. Partitioning columns into groups
3. Choosing the encoding format for each group

Estimating column compression ratios

$$S_i^{\mathrm{OLE}} = 4|\mathcal{G}_i| + d_i\left(4 + \alpha|\mathcal{G}_i|\right) + 2\sum_{j=1}^{d_i} b_{ij} + 2z_i,$$

$$S_i^{\mathrm{RLE}} = 4|\mathcal{G}_i| + d_i\left(4 + \alpha|\mathcal{G}_i|\right) + 4\sum_{j=1}^{d_i} r_{ij},$$

Instead of scanning the full data matrix, estimate parameters from a random sample of the data

Partitioning columns into groups

1. Enumerate all possible partitions, infeasible. Bell(13)=4213597
2. Greedy brute force.
3. Bin packing + greedy brute force.

Choosing the encoding format for each group

1. Scan the data matrix and compute actual compressed sizes for chosen groups.
2. For each group, compute compressed size as the minimum of OLE and RLE sizes.
3. If a group is incompressible, keep removing the column with largest estimated compressed size until group is compressible or empty.

Experiments

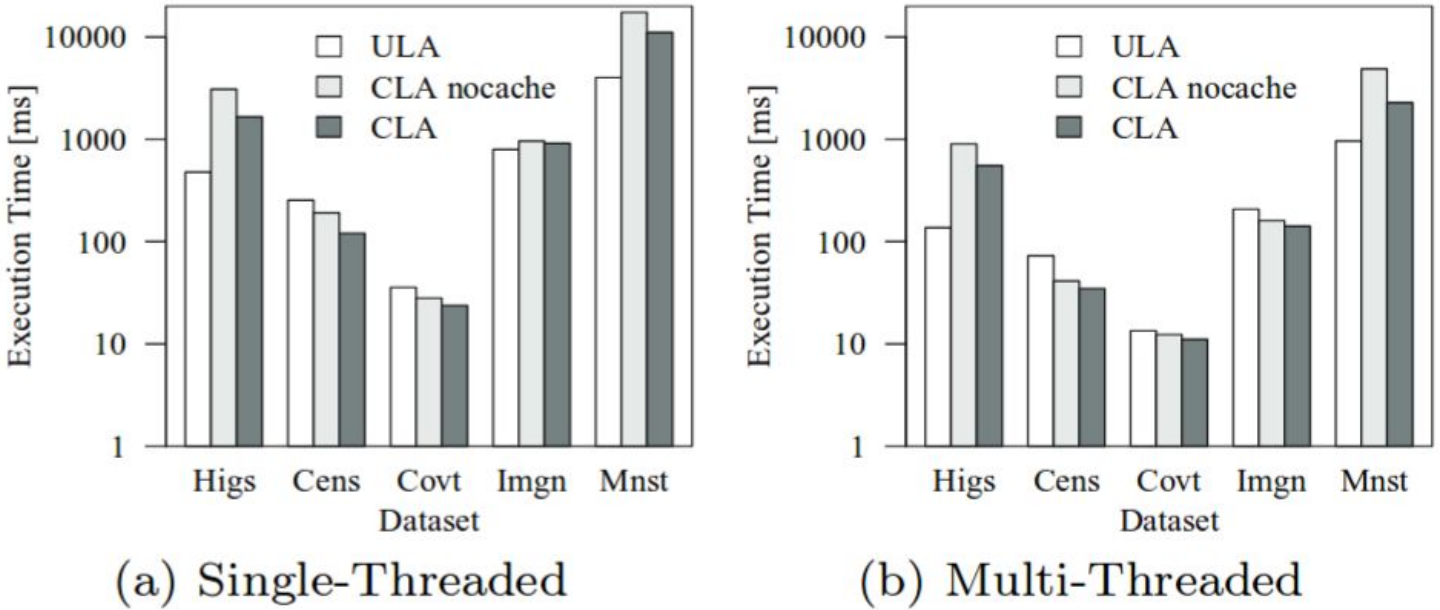
Experiments: Datasets

Dataset	#Rows	#Columns	Sparsity	In-memory size
Higgs	11M	28	0.92	2.5 GB
Census	2.5M	68	0.43	1.3 GB
Covtype	600K	54	0.22	0.14 GB
ImageNet	1.2M	900	0.31	4.4 GB
Mnist8m	8.1M	784	0.25	19 GB

Experiments: Compression ratio

Dataset	Gzip	Snappy	CSR-VI	D-VI	CLA
Higgs	1.93	1.38	1.04	1.9	**2.03**
Census	17.11	6.04	3.62	7.99	**27.46**
Covtype	10.4	6.13	3.56	2.48	**12.73**
ImageNet	5.54	3.35	2.07	1.93	**7.38**
Mnist8m	4.12	2.60	2.53	N/A	**6.14**

Experiments: Matrix-vector multiplication time



Thank you for listening