MNC: Structure-Exploiting Sparsity Estimation for Matrix Expressions

Johanna Sommer[3], Matthias Boehm[2], Alexandre Evfimievski[1], Berthold Reinwald[1], Peter Haas[4]

[1] IBM Research – Almaden [2] Graz University of Technology
[3] IBM Germany [4] UMass Amherst

SIGMOD 2019
Research 16: Machine Learning

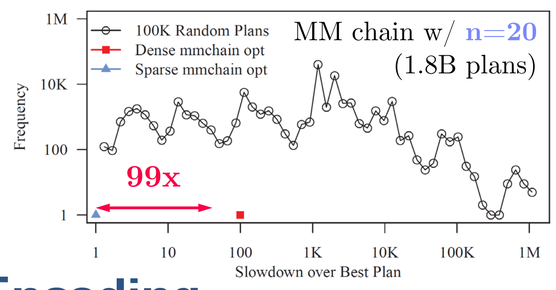
Contact: Matthias Boehm
m.boehm@tugraz.at

Motivation

Sparsity Estimation

- **Ubiquitous Sparse Matrices**
  - Sparse inputs: NLP, graph, recommenders, scientific computing
  - Data preprocessing: binning/recoding + one-hot encoding/feature hashing
  - Sparse intermediates: selection predicates, dropout
  - Transformation matrices: random reshuffling, sampling
  - ➔ Automatic application of sparse operations

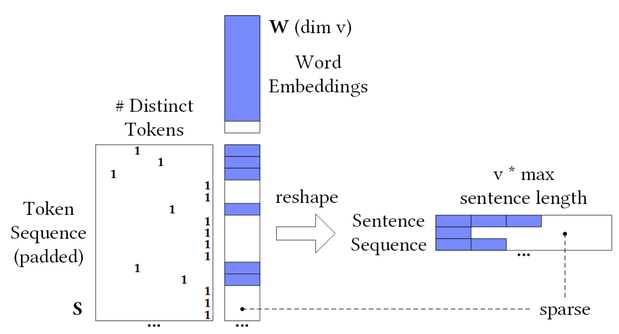
- **Problem of Sparsity Estimation**
  - **Compilation:** memory and cost estimation (local vs distributed, rewrites, resource allocation, operator fusion)
  - **Runtime:** format decisions and memory pre-allocation



Example: NLP Sentence Encoding

- **Problem**
  - Encoding a sequence of words (padded to max sentence length) into sequence of word embeddings
  - Input for SentenceCNN

- **Implementation**
  - Permutation matrix multiply
  - Reshape into sentences



Sparsity Estimators

Assumptions

- **A1: No Cancellation Errors**
- **A2: No Not-A-Number (NaN*0=NaN)**

Common assumptions
➔ Boolean mat. mult.
m x n by n x l
$(s_C = \frac{nnz(C)}{ml})$

Existing Sparsity Estimators

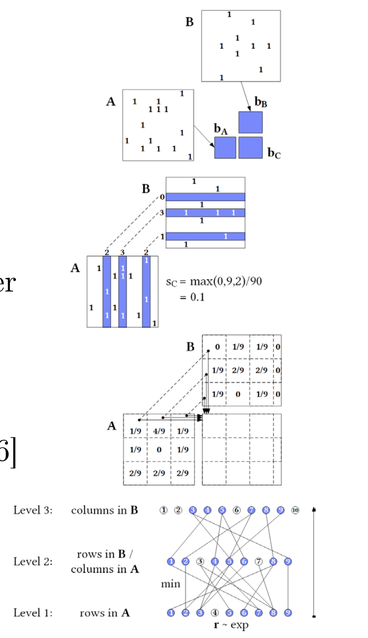
- **#1 Naïve Metadata Estimators**
  - Estimate sparsity from the inputs (e.g., SystemML)
  - $\hat{s}_C = 1 - (1 - s_A s_B)^n$
  - $\hat{s}_C = \min(1, s_A n) \cdot \min(1, s_B n)$

- **#2 Naïve Bitset Estimator**
  - Convert inputs to bitsets and perform Boolean mm
  - **Examples:** SciDB [SSDBM'11], cuSparse, MKL

- **#3 Sampling**
  - Sampling of aligned columns of A and rows of B
  - Sparsity estimated via max of count-products
  - **Examples:** MatFast [ICDE'17], improvements in paper

- **#4 Density Map**
  - Store sparsity per b x b block (default b = 256)
  - MM-like estimator (E_{ac} for *, $s_A + s_B - s_A s_B$ for +)
  - **Examples:** SpMacho [EDBT'15], AT Matrix [ICDE'16]

- **#5 Layered Graph [J.Comb.Opt.'98]**
  - Nodes: rows/columns in mm chain
  - **Edges:** non-zeros connecting rows/columns
  - Assign r-vectors ~exp and propagate via min



Apache SystemML™
github.com/apache/systemml
github.com/tugraz-isds/systemds

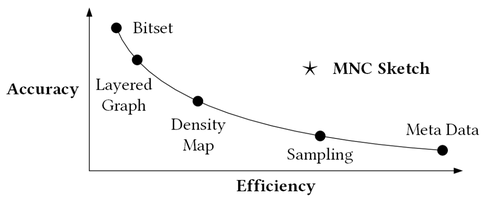
MNC Sketch

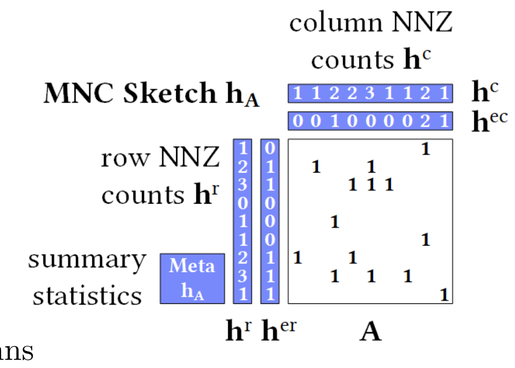
Basic Data Structure

- **#1 Row/Column NNZs**
  - Count vectors for nnz per row/col
  - h^r=rowSums(A!=0); h^c=colSums(A!=0)

- **#2 Extended Row/Column NNZs**
  - h^{er} = rowSums((A!=0)*(h^c=1))
  - h^{ec} = colSums((A!=0)*(h^r=1))

- **#3 Summary Statistics**
  - Max nnz per row/column
  - # of non-empty / # of half-full rows/columns





Sparsity Estimation

+ **Additional Operations**
(reorg and element-wise ops in the paper)

- **Basic Estimator**
  - DensityMap-like estimator over column/rows slices

- **#1 Exact Sparsity Estimation**
  - Under assumptions A1 and A2, exact sparsity estimation possible

$s_C = \hat{s}_C = h_A^c h_B^r / (ml)$
if $\max(h_A^r) \leq 1 \vee \max(h_B^c) \leq 1$

Sparsity Estimation, cont.

- **#2 Lower and Upper Bounds**
  - Lower bound ➔ clipping of bad estimates
  - Upper bound ➔ improved estimates

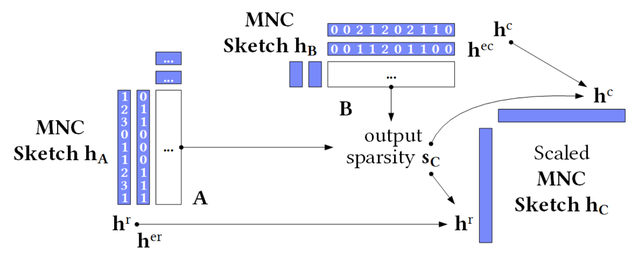
$s_C \geq |h_A^r > n/2| \cdot |h_B^c > n/2| / (ml)$
$s_C < nnz(h_A^r) \cdot nnz(h_B^c) | / (ml)$

- **#3 Extended NNZ Counts**
  - Compose estimate from exactly known and estimated quantities (i.e., number non-zeros)

$\hat{s}_C = (h_A^{ec} h_B^r + (h_A^c - h_A^{ec}) h_B^{er} + E_{dm}(h_A^c - h_A^{ec}, h_B^r - h_B^{er}, p) \cdot p/(ml))$
$p = (nnz(h_A^r) - |h_A^r = 1|) \cdot (nnz(h_B^c) - |h_B^c = 1|)$

Sketch Propagation

- **Basic Sketch Propagation**
  - Estimate sparsity $\hat{s}_C$
  - Propagate h_A^r and h_B^c scaled to sparsity $\hat{s}_C$ ➔ E.g., $\sum h_C^r \approx \hat{s}_C ml$
  - **Assumption:** Structure preserving



- **Probabilistic Rounding**
  - Ultra-sparse matrices ➔ basic rounding would introduce bias
  - $h_C^r = round(h_A^r \cdot \hat{s}_C ml / \sum h_A^r)$ w/ probability $h_C^r - \lfloor h_C^r \rfloor$ of rounding h_C^r up

- **Exact Sketch Propagation**
  - If A or B are fully diagonal ➔ propagate h_B or h_A, respectively

Experiments

SparsEst: Estimation Benchmark

B1 Structured Matrix Products

Name	Expression	Data
B1.1 NLP	X W	Syn1
B1.2 Scale	diag(λ) X	Syn2
B1.3 Perm	table(s1,s2) X	Syn2
B1.4 Outer	C R	Syn3/Syn4
B1.5 Inner	R C	Syn4/Syn3

B2 Real Matrix Operations

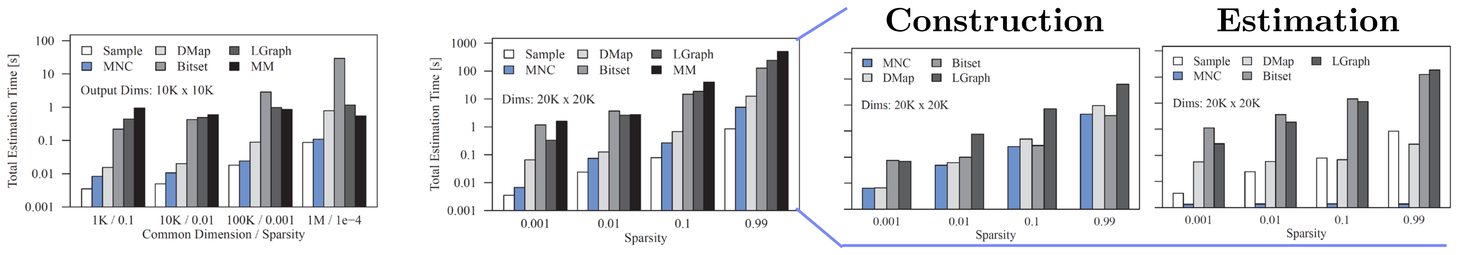
Name	Expression	Data
B2.1 NLP	X W	AMin A
B2.2 Project	X P	Cov
B2.3 CoRefG	G t(G)	AMin R
B2.4 EmailG	G G	Email
B2.5 Mask	M * X	Mnist1m

B3 Real Matrix Expressions (Chain)

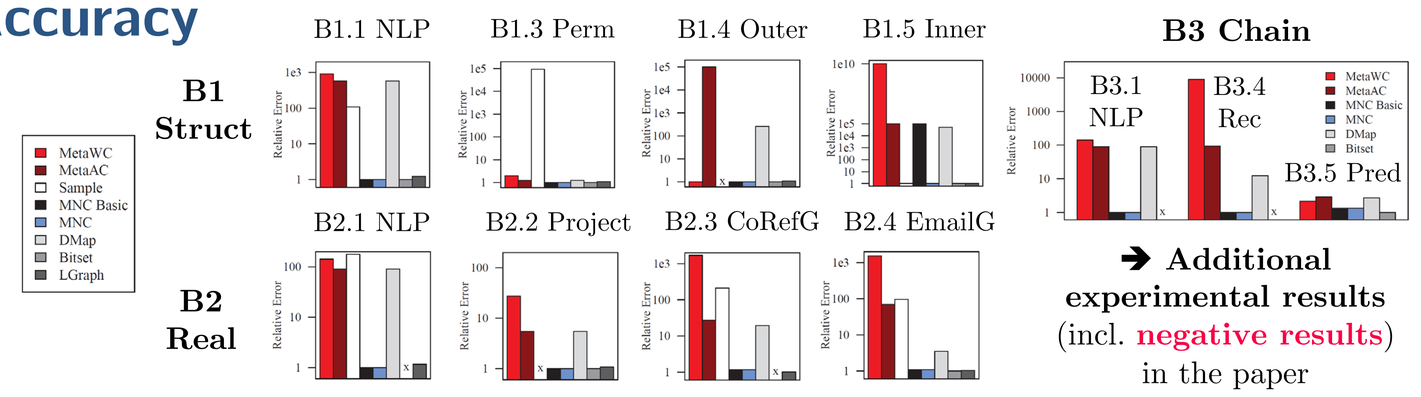
Name	Expression	Data	
B3.1 NLP	reshape(X W)	AMin A	25.1M x 2.5M, **3.9e-7**
B3.2 Scale&Shift	t(S) t(X) diag(w) X S B	Mnist1m	1M x 784, **0.25**
B3.3 Graph	P G G G G	AMin R	3.1M x 3.1M, **2.6e-6**
B3.4 Recomm.	(P X != 0) * (P L t(R))	Amazon	8M x 2.3M, **1.2e-6**
B3.5 Predicate	X * (R * S + T) != 0	Mnist1m	

Runtime Overhead

Experimental Setup: 2-socket Xeon E5-2620 (24 vcores, 128GB memory), single-threaded estimators, multi-threaded mm



Accuracy



➔ **Additional experimental results** (incl. **negative results**) in the paper