

Project 4.1: Sparse Matrix Arithmetics & Project 6.7: Multi-key LSD Sort



Xie Li, Yi-Nan Xu and
Yuan-Hang Zhang^a

University of
Chinese Academy of Sciences

June 29, 2017

^aCo-last author.

OUTLINE

INTRODUCTION

Styling and Coding Tools

TASK 1: SPARSE MATRIX LIBRARY

Framework Overview

Modern Updates

Personal Notes

TASK 2: MULTI-KEY SORT

Framework Overview

Observations and Analysis

INTRODUCTION

Both programs are implemented in C++. Source code can be obtained from:

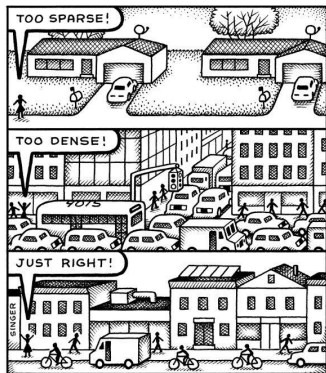
- <https://github.com/qq1024407215/SparseMatLib>
 - <https://github.com/poemonsense/MultiKeySort>
- Used modern C++ features (templates, smart pointers, classes, operator overloading)...
- Usefulness:
- Sparse matrices: for *compressed sensing*, high-dimensional data, data forecasts and sparse graphs...
 - Multi-key sort: databases (ORDER BY)

PROBLEM DESCRIPTION

以“带行逻辑链接信息”的三元组顺序表表示稀疏矩阵，实现两个矩阵相加、相减和相乘的运算。稀疏矩阵的输入形式采用三元组表示，而运算结果的矩阵则以通常的阵列形式列出。

【选做内容】

- 按教科书 5.3.2 节中的描述方法，以十字链表表示稀疏矩阵。



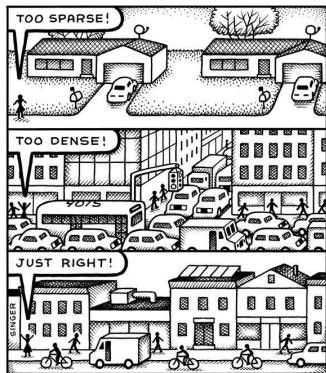
"No Exit: Goldilocks Evaluates Density"
©2011 Andy Singer

PROBLEM DESCRIPTION

以“带行逻辑链接信息”的三元组顺序表表示稀疏矩阵，实现两个矩阵相加、相减和相乘的运算。稀疏矩阵的输入形式采用三元组表示，而运算结果的矩阵则以通常的阵列形式列出。

【选做内容】

- 按教科书 5.3.2 节中的描述方法，以十字链表表示稀疏矩阵。
- 增添矩阵求逆的运算，包括不可求逆的情况。在求逆之前，先将稀疏矩阵的内部表示改为十字链表。



"No Exit: Goldilocks Evaluates Density"
©2011 Andy Singer

NEEDS ANALYSIS

- Operations: addition, subtraction, multiplication

NEEDS ANALYSIS

- Operations: addition, subtraction, multiplication
- Extra credit: inverse, transpose, linear equation system solver...

NEEDS ANALYSIS

- Operations: addition, subtraction, multiplication
- Extra credit: inverse, transpose, linear equation system solver...
- **BLAS**, *basic linear algebra subprograms*

A specification for common linear algebra operations such as vector addition, scalar multiplication, dot products, linear combinations, and matrix multiplication.

Some widely used implementations: NVIDIA[®] cuSPARSE & cuBLAS, OpenBLAS, Intel[®] Math Kernel Library (MKL) / Sparse BLAS, etc.

NEEDS ANALYSIS

- Operations: addition, subtraction, multiplication
- Extra credit: inverse, transpose, linear equation system solver...
- **BLAS**, *basic linear algebra subprograms*

A specification for common linear algebra operations such as vector addition, scalar multiplication, dot products, linear combinations, and matrix multiplication.

- **Level 1** vector-vector: $\mathbf{y} \leftarrow \alpha \mathbf{x} + \mathbf{y}$

Some widely used implementations: NVIDIA[®] cuSPARSE & cuBLAS, OpenBLAS, Intel[®] Math Kernel Library (MKL) / Sparse BLAS, etc.

NEEDS ANALYSIS

- Operations: addition, subtraction, multiplication
- Extra credit: inverse, transpose, linear equation system solver...
- **BLAS**, *basic linear algebra subprograms*

A specification for common linear algebra operations such as vector addition, scalar multiplication, dot products, linear combinations, and matrix multiplication.

- **Level 1** vector-vector: $\mathbf{y} \leftarrow \alpha \mathbf{x} + \mathbf{y}$
- **Level 2** vector-matrix: $\mathbf{y} \leftarrow \alpha \mathbf{A} \mathbf{x} + \beta \mathbf{y}$

Some widely used implementations: NVIDIA[®] cuSPARSE & cuBLAS, OpenBLAS, Intel[®] Math Kernel Library (MKL) / Sparse BLAS, etc.

NEEDS ANALYSIS

- Operations: addition, subtraction, multiplication
- Extra credit: inverse, transpose, linear equation system solver...
- **BLAS**, *basic linear algebra subprograms*

A specification for common linear algebra operations such as vector addition, scalar multiplication, dot products, linear combinations, and matrix multiplication.

- **Level 1** vector-vector: $\mathbf{y} \leftarrow \alpha \mathbf{x} + \mathbf{y}$
- **Level 2** vector-matrix: $\mathbf{y} \leftarrow \alpha \mathbf{A} \mathbf{x} + \beta \mathbf{y}$
- **Level 3** matrix-matrix: $\mathbf{C} \leftarrow \alpha \mathbf{A} \mathbf{B} + \beta \mathbf{C}$

Some widely used implementations: NVIDIA[®] cuSPARSE & cuBLAS, OpenBLAS, Intel[®] Math Kernel Library (MKL) / Sparse BLAS, etc.

OUR IMPLEMENTATION

- Storage structures
 - Array1D: 1-D vector
 - Array2D: `<vector<vector<int>>`
 - CrossList : row-linked, column-linked nodes
 - Triplet: naive 3-tuples
 - RowLinkTriMat: row-linked 3-tuples
 - CSRStore: **Compressed Spase Row**
- Methods
 - Overloaded operators: $+$, \times , $=$...
 - Inversion
 - Determinant (co-factor expansion) + classical adjoint
 - Iterative solvers: **CG (conjugate gradient)**

CSR: COMPRESSED SPARSE ROW

$$\begin{pmatrix} 1 & 7 & 0 & 0 \\ 0 & 2 & 8 & 0 \\ 5 & 0 & 3 & 9 \\ 0 & 6 & 0 & 4 \end{pmatrix}$$

0 2 4 7 9	row offsets
0 1 1 2 0 2 3 1 3	column indices
1 7 2 8 5 3 9 6 4	values

- Entries from a row required to be consecutively located in this data structure

CSR: COMPRESSED SPARSE ROW

$$\begin{pmatrix} 1 & 7 & 0 & 0 \\ 0 & 2 & 8 & 0 \\ 5 & 0 & 3 & 9 \\ 0 & 6 & 0 & 4 \end{pmatrix}$$

0	2	4	7	9	row offsets				
0	1	1	2	0	2	3	1	3	column indices
1	7	2	8	5	3	9	6	4	values

- Entries from a row required to be consecutively located in this data structure
- Reduced number of memory accesses

CSR: COMPRESSED SPARSE ROW

$$\begin{pmatrix} 1 & 7 & 0 & 0 \\ 0 & 2 & 8 & 0 \\ 5 & 0 & 3 & 9 \\ 0 & 6 & 0 & 4 \end{pmatrix}$$

0	2	4	7	9	row offsets				
0	1	1	2	0	2	3	1	3	column indices
1	7	2	8	5	3	9	6	4	values

- Entries from a row required to be consecutively located in this data structure
- Reduced number of memory accesses
- More vectorization-oriented optimizations

CSR: COMPRESSED SPARSE ROW

$$\begin{pmatrix} 1 & 7 & 0 & 0 \\ 0 & 2 & 8 & 0 \\ 5 & 0 & 3 & 9 \\ 0 & 6 & 0 & 4 \end{pmatrix}$$

0	2	4	7	9	row offsets				
0	1	1	2	0	2	3	1	3	column indices
1	7	2	8	5	3	9	6	4	values

- Entries from a row required to be consecutively located in this data structure
- Reduced number of memory accesses
- More vectorization-oriented optimizations
 - **CSRL [ISCAS 2014]**: 一种提高 SpMV 向量化性能的新型稀疏矩阵存储格式. 刘芳芳, 杨超, 2014.

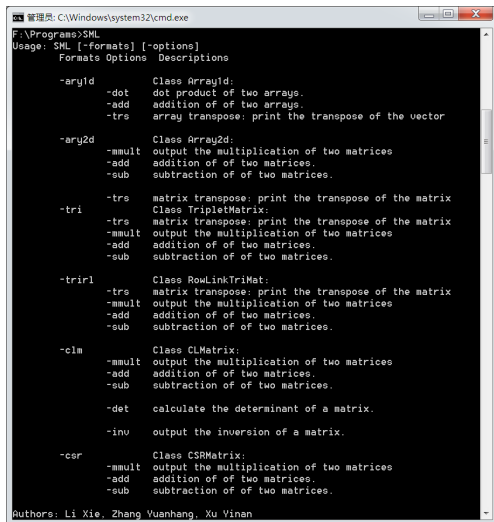
CSR: COMPRESSED SPARSE ROW

$$\begin{pmatrix} 1 & 7 & 0 & 0 \\ 0 & 2 & 8 & 0 \\ 5 & 0 & 3 & 9 \\ 0 & 6 & 0 & 4 \end{pmatrix}$$

0	2	4	7	9	row offsets				
0	1	1	2	0	2	3	1	3	column indices
1	7	2	8	5	3	9	6	4	values

- Entries from a row required to be consecutively located in this data structure
- Reduced number of memory accesses
- More vectorization-oriented optimizations
 - **CSRL [ISCAS 2014]**: 一种提高 SpMV 向量化性能的新型稀疏矩阵存储格式. 刘芳芳, 杨超, 2014.
 - **CSR(r) [NUDT 2016]**: 一种适合向量化的稀疏矩阵存储格式. 谢佩珍等, 2016.

COMMAND-LINE INTERFACE



```
管理员: C:\Windows\system32\cmd.exe
F:\Programs>SML
Usage: SML [-formats] [-options]
      Formats Options  Descriptions

-ary1d      Class Array1d:
            -dot      dot product of two arrays.
            -add      addition of of two arrays.
            -trs      array transpose: print the transpose of the vector

-ary2d      Class Array2d:
            -mmult     output the multiplication of two matrices
            -add      addition of of two matrices.
            -sub      subtraction of of two matrices.

            -trs      matrix transpose: print the transpose of the matrix

-tri        Class TripletMatrix:
            -trs      matrix transpose: print the transpose of the matrix
            -mmult     output the multiplication of two matrices
            -add      addition of of two matrices.
            -sub      subtraction of of two matrices.

-tri1l      Class RowLinkTriMat:
            -trs      matrix transpose: print the transpose of the matrix
            -mmult     output the multiplication of two matrices
            -add      addition of of two matrices.
            -sub      subtraction of of two matrices.

-clm        Class CLMatrix:
            -mmult     output the multiplication of two matrices
            -add      addition of of two matrices.
            -sub      subtraction of of two matrices.

            -det      calculate the determinant of a matrix.

            -inv      output the inversion of a matrix.

-csr        Class CSRMatrix:
            -mmult     output the multiplication of two matrices
            -add      addition of of two matrices.
            -sub      subtraction of of two matrices.

Authors: Li Xie, Zhang Yuanhang, Xu Yinan
```

INVERSE OF A MATRIX

```
F:\Programs>SML -clm -inv
input the row and col number of the matrix:
row: 3
col: 3
input the non-zero number of the matrix:
7
input the non-zero elements the matrix (in the format 'row col val'):
1th: 1 2 2
2th: 2 1 1
3th: 2 2 1
4th: 2 3 -1
5th: 3 1 2
6th: 3 2 1
7th: 3 3 -1
original matrix:
CLMatrix Print:
0          2          0
1          1         -1
2          1         -1
Inversion result:
CLMatrix Print:
0          -1          1
0.5        0          -0
0          -2          1
Authors: Li Xie, Zhang Yuanhang, Xu Yinan
```

A RETROSPECTIVE

- 内存泄漏不是说着玩的 (用 `shared_pointer`)
- 大工程的管理
 - Git
 - 依赖和包含关系

PROBLEM DESCRIPTION

按用户给定的进行排序的关键字的优先关系，输出排序结果。约定按 LSD 法进行多关键字的排序。在对各个关键字进行排序时采用两种策略：其一是利用稳定的内部排序法，其二是利用“分配”和“收集”的方法，并综合比较这两种策略。

【选做内容】

- 增添按 MSD 策略进行排序，并和上述两种排序策略进行综合比较。

OUR IMPLEMENTATION

- 使用 `vector` 存储各记录，记录内的数据同样使用 `vector` 存储，数据范围为 $[0, 99]$
- 比较 Merge Sort 与 Radix Sort 的时间复杂度
- 比较 LSD 法与 MSD 法的时间复杂度
- 观察消耗时间与关键字数量、记录数量的关系
- 用 Python 写了一个简单的 GUI，方便处理输入、输出数据，验证结果有效性并导出

IMPLEMENTATIONS

- LSD: 从低优先级关键字开始，依次进行全序列的排序
- MSD: 从高优先级关键字开始进行排序，递归进行，下一次进行排序的范围为前一次关键字相等的范围
- 归并排序 (Merge Sort), 时间复杂度为 $O(n \log n)$
- 基数排序 (Radix Sort), 分配、收集的方法，时间复杂度为 $O(d(n + rd))$

DEMO

多关键字排序 Multi-key ...

多关键字排序

关键字数量:

样本大小:

排序优先级:

radixLSD, radixMSD, mergeLSD, mergeMSD

```
0.171, 0.218, 1.187, 0.39
0.172, 0.234, 1.125, 0.39
0.187, 0.218, 1.093, 0.422
0.172, 0.218, 1.109, 0.39
0.156, 0.234, 1.093, 0.39
0.187, 0.218, 1.093, 0.39
```

Analysis

LSD Radix Sort Average Time: 0.174s
MSD Radix Sort Average Time: 0.223s
LSD Merge Sort Average Time: 1.117s
MSD Merge Sort Average Time: 0.395s

确定

随机 分析 比较 趋势

Finished!

RESULTS

- 10000 条记录，5 个关键字的情况下

LSD Radix Sort	0.027s
MSD Radix Sort	0.066s
LSD Merge Sort	0.211s
MSD Merge Sort	0.075s

表: 不同排序算法的用时比较

- 关键字数量**远小于**记录数，此时基数排序比归并排序快
- 基数排序中，LSD 比 MSD 快是递归的时间消耗造成的，LSD 是不需要递归的
- 归并排序中，LSD 比 MSD 慢，原因是减少了排序的次数，也减少了递归的次数

RESULTS

- 记录数取值为 {2000, 5000, 10000, 12000, 15000, 18000, 20000, 25000, 30000, 50000}

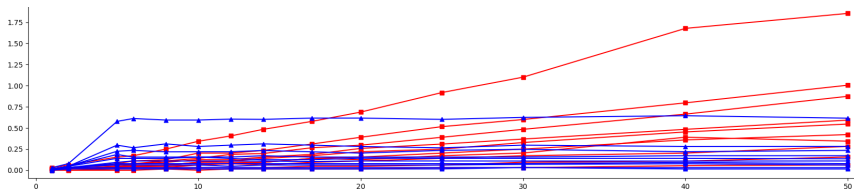


图: 基数排序中, 关键字数量对消耗时间的影响

— LSD — MSD

RESULTS

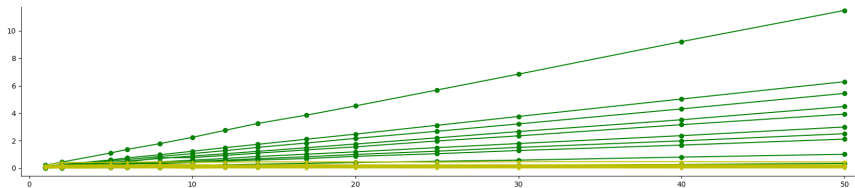


图: 归并排序中, 关键字数量对消耗时间的影响

— LSD — MSD

RESULTS

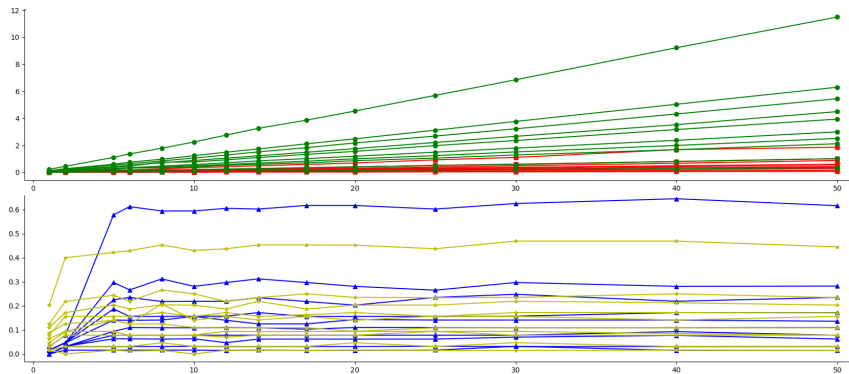


图: 关键字数量对消耗时间的影响 (上图为 LSD)

— 基数排序 — 归并排序

RESULTS

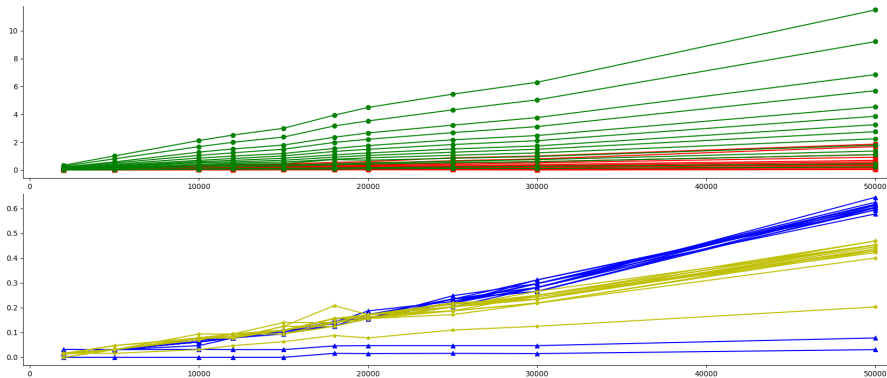


图: 记录数对消耗时间的影响 (上图为 LSD)

— 基数排序 — 归并排序

THANK YOU!

Q&A time