Efficient SpMV on GPGPU

Sebastian Szkoda

Wrocław Centre for Networking and Supercomputing
Aim

The aim of this tutorial is to demonstrate how to choose the best method of SpMV to particular SpM, GPGPU.
Abstract

- Brief introduction to Sparse Matrices (SpM)
- Sparse Matrix Vector Multiplication
- Overview of SpM-storing formats
- How to choose the best SpM-storing format for particular matrix for SpMV operation.
Abstract

- Brief introduction to Sparse Matrices (SpM)
- Sparse Matrix Vector Multiplication
- Overview of SpM-storing formats
- How to choose the best SpM-storing format for particular matrix for SpMV operation.
Abstract

- Brief introduction to Sparse Matrices (SpM)
- Sparse Matrix Vector Multiplication
- Overview of SpM-storing formats
- How to choose the best SpM-storing format for particular matrix for SpMV operation.
Abstract

• Brief introduction to Sparse Matrices (SpM)
• Sparse Matrix Vector Multiplication
• Overview of SpM-storing formats
• How to choose the best SpM-storing format for particular matrix for SpMV operation.
Abstract

- Brief introduction to Sparse Matrices (SpM)
- Sparse Matrix Vector Multiplication
- Overview of SpM-storing formats
- How to choose the best SpM-storing format for particular matrix for SpMV operation.
**Sparse Matrix definition**

- In numerical analysis SpM is a matrix mainly populated with zeros.
Sparse Matrices

Numerous problems lead to Sparse Matrices:

- System of linear equations
- Differential equations by its discretization
- Eigenvalue problems
- Graph matrix representation

Scientific domains:

- Chemistry, weather forecast, physics, biology and engineering
Sparse Matrices

Numerous problems lead to Sparse Matrices:

- System of linear equations
- Differential equations by its discretization
- Eigenvalue problems
- Graph matrix representation

Scientific domains:

- Chemistry, weather forecast, physics, biology and engineering
Sparse Matrices

Numerous problems lead to Sparse Matrices:

- System of linear equations
- Differential equations by its discretization
- Eigenvalue problems
- Graph matrix representation

Scientific domains:
- Chemistry, weather forecast, physics, biology and engineering
Sparse Matrices

Numerous problems lead to Sparse Matrices:

- System of linear equations
- Differential equations by its discretization
- Eigenvalue problems
- Graph matrix representation

Scientific domains:

- Chemistry, weather forecast, physics, biology and engineering
Typical **Sparse** Matrix

- Thousands or millions of columns
- Rarely exceeding 100 non-zero elements per row
- Sparsity pattern:
Typical **Sparse** Matrix

- Thousands or millions of columns
- Rarely exceeding 100 non-zero elements per row
- Sparsity pattern:
Typical **Sparse Matrix**

- Thousands or millions of columns
- Rarely exceeding 100 non-zero elements per row
- Sparsity pattern:
Sparse Matrix Collection

From 1991 until now, Thimoty A. Davis and Yifan Hu from University of Florida has been collecting Sparse Matrices. Database is available at:
http://www.cise.ufl.edu/research/sparse/matrices/

It contains over 2500 problems (February 2014)
Sparse Matrix Vector (SpMV)

- Multiplication of Sparse Matrix and dense Vector is called SpMV.

\[ \hat{A} \cdot \vec{x} \]

- Level 2 numerical kernel routine
- It is main computational part in most cases where sparse matrices appear. For example Iterative methods of solving systems of linear equations. CG, BiCG, GMRES,...
Sparse Matrix Vector (SpMV)

- On modern throughput oriented architectures, efficiency of:

\[ \hat{A} \cdot \hat{x} \]

operation is mainly dependent on data organisation in computational device memory to achieve coalesced memory reads thus increase memory bandwidth what yields maximum efficiency.
SpM storing formats - **COO**

COO – Coordinate format.

Information about row index, column index and value of each nonzero element is stored in three, one-dimensional arrays:

\[
\begin{align*}
\text{Val} & = [1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10] \\
\text{ColInd} & = [0 \ 3 \ 1 \ 4 \ 2 \ 4 \ 2 \ 3 \ 4 \ 4] \\
\text{RowInd} & = [0 \ 0 \ 1 \ 1 \ 2 \ 2 \ 3 \ 3 \ 3 \ 4]
\end{align*}
\]

\[
\begin{bmatrix}
1 & 0 & 0 & 2 & 0 \\
0 & 3 & 0 & 0 & 4 \\
0 & 0 & 5 & 0 & 6 \\
0 & 0 & 7 & 8 & 9 \\
0 & 0 & 0 & 0 & 10
\end{bmatrix}
\]
SpM storing formats - **CRS**

CRS – Compressed sparse row. COO like format with RowInd array replaced by array holding number of previous nnz elements.

\[ \text{Val} = [1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10] \]
\[ \text{ColInd} = [0 \ 3 \ 1 \ 4 \ 2 \ 4 \ 2 \ 3 \ 4 \ 4] \]
\[ \text{RowPtr} = [0 \ 2 \ 4 \ 6 \ 9 \ 10] \]
SpM storing formats - **ELL**

ELL – ELLPACK/ITPACK. For (N,M) SpM with max K nnz elements in row it holds data as two dense matrices. Val (N,K) holds nnz elements, filling less occupied rows with zeros. ColInd holds corresponding column indices.

\[
\text{Val} = \begin{bmatrix}
1 & 2 & 0 \\
3 & 4 & 0 \\
5 & 6 & 0 \\
7 & 8 & 9 \\
10 & 0 & 0 \\
\end{bmatrix},
\text{ColInd} = \begin{bmatrix}
0 & 3 & -1 \\
1 & 4 & -1 \\
2 & 4 & -1 \\
2 & 3 & 4 \\
4 & -1 & -1 \\
\end{bmatrix}
\]
SpM storing formats - **HYB**

**HYB** – Hybrid Format

Is a combination of the ELL and COO formats.

It is an attempt to reduce ELL potentially unacceptable storage overhead and benefit from its speed on GPGPU's.
SpM storing formats - **CMRS**

CMRS – Compressed Multi-Row Sparse Format.

- The key idea behind compressed multi-row storage is to reduce CRS low SpMV performance for matrices with relatively short rows.
- It groups shorter rows into strips and process them in parallel.
- Data storage requires four arrays, two same as in CRS,
- Array (StripPtr) denotes the first elements of each strip
- Array (RowInStrip) locates rows within a strip
SpM storing formats - CMRS

CMRS – Compressed Multi-Row Sparse Format.

- The key idea behind compressed multi-row storage is to reduce CRS low SpMV performance for matrices with relatively short rows.
- It groups shorter rows into strips and process them in parallel.
- Data storage requires four arrays, two same as in CRS,
- Array (StripPtr) denotes the first elements of each strip
- Array (RowInStrip) locates rows within a strip
SpM storing formats - CMRS

CMRS – Compressed Multi-Row Sparse Format.

- The key idea behind compressed multi-row storage is to reduce CRS low SpMV performance for matrices with relatively short rows.
- It groups shorter rows into strips and process them in parallel.
- Data storage requires four arrays, two same as in CRS,
- Array (StripPtr) denotes the first elements of each strip
- Array (RowInStrip) locates rows within a strip
SpM storing formats - **CMRS**

CMRS – Compressed Multi-Row Sparse Format.

- The key idea behind compressed multi-row storage is to reduce CRS low SpMV performance for matrices with relatively short rows.
- It groups shorter rows into strips and process them in parallel.
- Data storage requires four arrays, two same as in CRS,
  - Array (StripPtr) denotes the first elements of each strip
  - Array (RowInStrip) locates rows within a strip
SpM storing formats - CMRS

CMRS – Compressed Multi-Row Sparse Format.

- The key idea behind compressed multi-row storage is to reduce CRS low SpMV performance for matrices with relatively short rows.
- It groups shorter rows into strips and process them in parallel.
- Data storage requires four arrays, two same as in CRS,
- Array (StripPtr) denotes the first elements of each strip
- Array (RowInStrip) locates rows within a strip
SpM storing formats - **CMRS**

\[
\begin{bmatrix}
1 & 0 & 0 & 2 & 0 \\
0 & 3 & 0 & 0 & 4 \\
0 & 0 & 5 & 0 & 6 \\
0 & 0 & 7 & 8 & 9 \\
0 & 0 & 0 & 0 & 10 \\
\end{bmatrix}
\]

- \( \text{Val} = [1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9 \ 10] \)
- \( \text{ColInd} = [0 \ 3 \ 1 \ 4 \ 2 \ 4 \ 2 \ 3 \ 4 \ 4] \)
- \( \text{StripPtr} = [0 \ 4 \ 9 \ 10] \)
- \( \text{RowInStrip} = [0 \ 0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 1 \ 0] \)
SpM storing formats - Other

- PKT – Pocket format
- DIA – Diagonal format
- HYB – DIA + CSR
- CSR with permutations
- ELL-R, ELLR-T, sliced ELLR-T,
- ...
SpMV GPGPU libraries

- CuSparse – library provided by NVIDIA, supports COO, CSR, CSC, ELL/HYB sparse matrix format.
- Cusp – a library for sparse linear algebra and graph computations on CUDA.
How to choose the best sparse matrix storage format for SpMV operation.
Analysis

Let's consider set of 132 example matrices from University of Florida SpM collection.

Five methods for SpMV:

- HYB from cuSparse 4.1
- CRS form cuSparse 4.1
- Two basic implementations of CRS in vector and scalar variants
- CMRS implementation
Analysis

• For each of 132 matrices, time of SpMV operation was measured for every of 5 SpMV methods.

• The fastest multiplication method is chosen along with two features: mean number of nonzero elements and number of rows.

• Table with data can be found in the bibliography, it includes double and single precision SpMV on Nvidia K20 and GTX480.
Analysis – CRS variants

Number of rows vs. Number of nonzero elements.
Analysis – HYB, CRS
Analysis – HYB, CMRS

Number of rows vs. Number of nonzero elements elements.
Analysis – CRS, HYB, CMRS
Analysis – CRS, HYB, CMRS
Analysis – CRS, HYB, CMRS
Bibliography


• Table with data: Zbigniew Koza et al., Compressed Multiple-Row Storage Format, SIAM J. Sci. Comput., 36-2, C219-C239 (2014)

Thank you for your attention.