Sparse Boolean Linear Algebra on GPGPU*

*Note: Sub-titles are not captured in Xplore and should not be used

1st Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

3rd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

4th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

5th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

6th Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Abstract is very abstract. Abstract is very abstract.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

One of the techniques to effectively solve a data analysis problem is to reduce it to linear algebra operations over vectors and matrices for appropriate values set. That gives one well studied for years mathematical apparatus, as well as the possibility to evaluate this problem with *zero cost* by linear algebra libraries, which utilize modern hardware, provide various optimization techniques and allow quickly and safely prototype solution in code with predefined building blocks.

Particularly, in graph data analysis such reduction is well presented, since a graph could be converted to the matrix and with specially defined *semiring* it could be effectively process in linear algebra fashion. Examples of problems solved in this way are all-pairs shortest path, breadth-first search, maximal independent set problems [**?**].

Since practical data often come with huge size and sparse form, what is also applicable for the graphs, it requires special processing tools for analysis, what appeals to sparse linear algebra libraries. Huge amount data analysis typically requires to process it as small chunks with the fixed or rarely changed set of instructions, what relates to the *single instruction, multiple data* (SIMD) model. Although modern CPUs allow to exploit SIMD optimizations, GPGPU gives much more power in such kind data processing at cost of implementation challenges. GPGPU programming introduces heterogeneous device model into system, memory traffic and data operations limitations, as well as requires to take into account vendor-specific capabilities.

At this moment there are already presented modern open-source and proprietary sparse linear algebra libraries on GPGPU for general and most common operations and values type However, because of its high specificity, sparse linear boolean algebra on GPGPU is still not presented. Boolean algebra allows to address problems over finite set of values, for example, reachability or relational queries for some graphs [**?**].

In this work we present the sparse linear boolean algebra operations implementation as a sand-alone self-sufficient programming libraries for the two most popular GPGPU platforms: OpenCL and NVIDIA Cuda. Cuda is a GPGPU technology for NVIDIA devices, which allows to employ some platform-specific facilities, such as unified memory, and make an architectural assumptions, what gives more optimizations space at cost of portability. OpenCL is platform agnostic API standard, which allows to run computations on different platforms, such as multi-threaded CPUs, GPUs, and FPGA. Our implementation relies on modern sparse matrices processing techniques, as well as exploits some optimizations, related to the boolean data processing.

II. Related work

Existing libraries, algorithns, frameworks.

III. Implementation Details

Details on implementation.
Architecture.

IV. Evaluation

Evaluation of the proposed implemenation(s).

Identify applicable funding agency here. If none, delete this.

V. Conclusion

Conclusion and future work.

Acknowledgment

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g”. Avoid the stilted expression “one of us (R. B. G.) thanks …”. Instead, try “R. B. G. thanks…”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

References