# Optimization of a Linear Algebra Approach to OLAP

## 1st general debriefing meeting

Filipe Costa Oliveira[1] Sérgio Manuel Rodrigues Caldas[2]

{a57816[1],a57779[2]}@alunos.uminho.pt
Department of Computer Science
University of Minho

University of Minho, April 2016

**Advisor**: Prof. Alberto Proença **Co-Advisor:** Prof. José Nuno Oliveira

# Table of Contents

# OLAP, understanding the Challenge

## Online Analytical processing (OLAP) systems:

- Perform multidimensional analysis of business data;
- Provides the capability for complex calculations, trend analysis, and sophisticated data modeling;

## Depends on Relational Algebra

- Lack algebraic properties;
- Qualitative and quantitative proofs for all the relational operator;

# Proposed Solution

## Proposed Solution

- proposed by R.Pontes, Benchmarking a Linear Algebra Approach to OLAP (2015).
- Focus on a **typed linear algebra** approach;
- Encodes OLAP functionality solely in terms of Linear Algebra operations;

## Depends on Linear Algebra operations

- Dot Product;
- Khatri-Rao Product;
- Kronecker Product;
- Hadamard Product;
- Transposition;

# Comparing LA to RA approach with TPC-H query plans

## Based on TPC-H Benchmark:

- Consists of a suite of business oriented ad-hoc queries and concurrent data modifications;
- The queries and the data have broad industry-wide relevance;
- Illustrates decision support systems that examine large volumes of data;
- Execute queries with a high degree of complexity;

# Translating TPC-H RA queries to La operations

## TPC-H Query 1

```
SELECT RETURNFLAG, LINESTATUS, sum(QUANTITY)
FROM LINEITEM
WHERE SHIPDATE >= 1998-08-28 AND SHIPDATE <= 1998-12-01
GROUP BY RETURNFLAG, LINESTATUS
```

Translates into:

$$\underbrace{(L_{ReturnFlag} \triangledown L_{LineStatus})}_{\text{projection}} \cdot \underbrace{[L]^{Shipdate>=1998-08-28}_{Shipdate} \cdot [L]^{Shipdate<=1998-12-01}_{Shipdate}}_{\text{selection}} \cdot \underbrace{[\![L]\!]_{Quantity} \cdot !^{\circ}}_{\text{aggregation}}$$

# Understanding the HPC Challenge

There was still lots of practical work to do:

- Define a DSL for the LA language ✓ (03-Fev to 29-Fev)
- Determined Software Requirements
  - Profile the given datasets ✓ (03-Fev to 03-Apr)
- Implement from the start a typed LA solution. ✍ (09-Mar to current)
  - The GNOME® Project GLib package
  - Intel® MKL (Intel® Math Kernel Library)
- Re-evaluate its performance in a real word scenario.

# Understanding the HPC Challenge

## Main Question:

Can our linear algebra solution provide a more efficient solution than its market competitors?

- Already reduced matrices sparsity degree to values near 0,01% vs $1*10^{-25}\%(f.e. : matrixA(10^{34}x10^{34}))$.
- Sparse Matrices will be represented using **BLAS BSR** (Block sparse row format) .
  - Similar to the CSR format.
  - Nonzero entries in the BSR are optimized to produce **square dense blocks**.

# Optimization of a Linear Algebra Approach to OLAP

## 1st general debriefing meeting

Filipe Costa Oliveira[1] Sérgio Manuel Rodrigues Caldas[2]

{a57816[1],a57779[2]}@alunos.uminho.pt
Department of Computer Science
University of Minho

University of Minho, April 2016

**Advisor**: Prof. Alberto Proença **Co-Advisor:** Prof. José Nuno Oliveira

# Appendix

- DSL Operations
- Profile the given datasets
- Our Approach decisions

# Appendix :: DSL Operations

## DSL Operations:

- Transposition - "'";
- Dot Product - "*";
- Khatri-Rao Product - "krao";
- Kronecker Product - "kron";
- Hadamard Product - "><";

## Sample Input file

```
matrix a, b;
matrix(2,2) c,d;
a = csvread('nomeficheiro1.csv', 2);
b = csvread('nomeficheiro2.csv', 3);
d = a kron b;
c  = a krao d;
d = d'..(a><b)*c;
csvwrite('nomedoficheiro.csv',c);
```

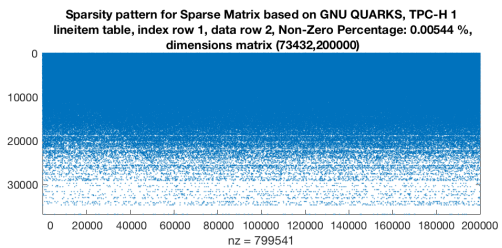
# Appendix :: Profile the given datasets

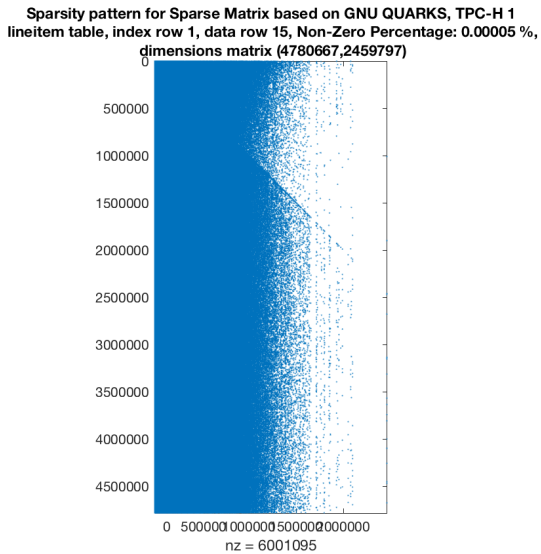
Reduced Matrices Sparsity by translating Base64 Encoding:

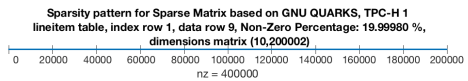
| RA | LA |
|---|---|
| Customer#000000001 | 10235451621031712117752211775221200 |

into **2-way association** between a string and a unique integer identifier using **GLib Quarks**:

| RA | LA |
|---|---|
| QHS1dfqs3BBrTlijJuiLQ1I05sWnHWCiiW1I | 2753 |



# Appendix :: Our Approach decisions

## Intel® MKL (Intel® Math Kernel Library)

- Accelerates math processing routines that increase application performance and reduce development time;
- Includes highly vectorized and threaded Linear Algebra Operations.

## Sparse Matrices Formats Used:

- Sparse BLAS CSR (Compressed Sparse Row) Matrix Storage Format;
- Sparse BLAS CSC (Compressed Sparse Column) Matrix Storage Format;
- Sparse BLAS COO (Coordinate) Matrix Storage Format;
- **Sparse BLAS BSR (Block sparse row format) Matrix Storage Format**;