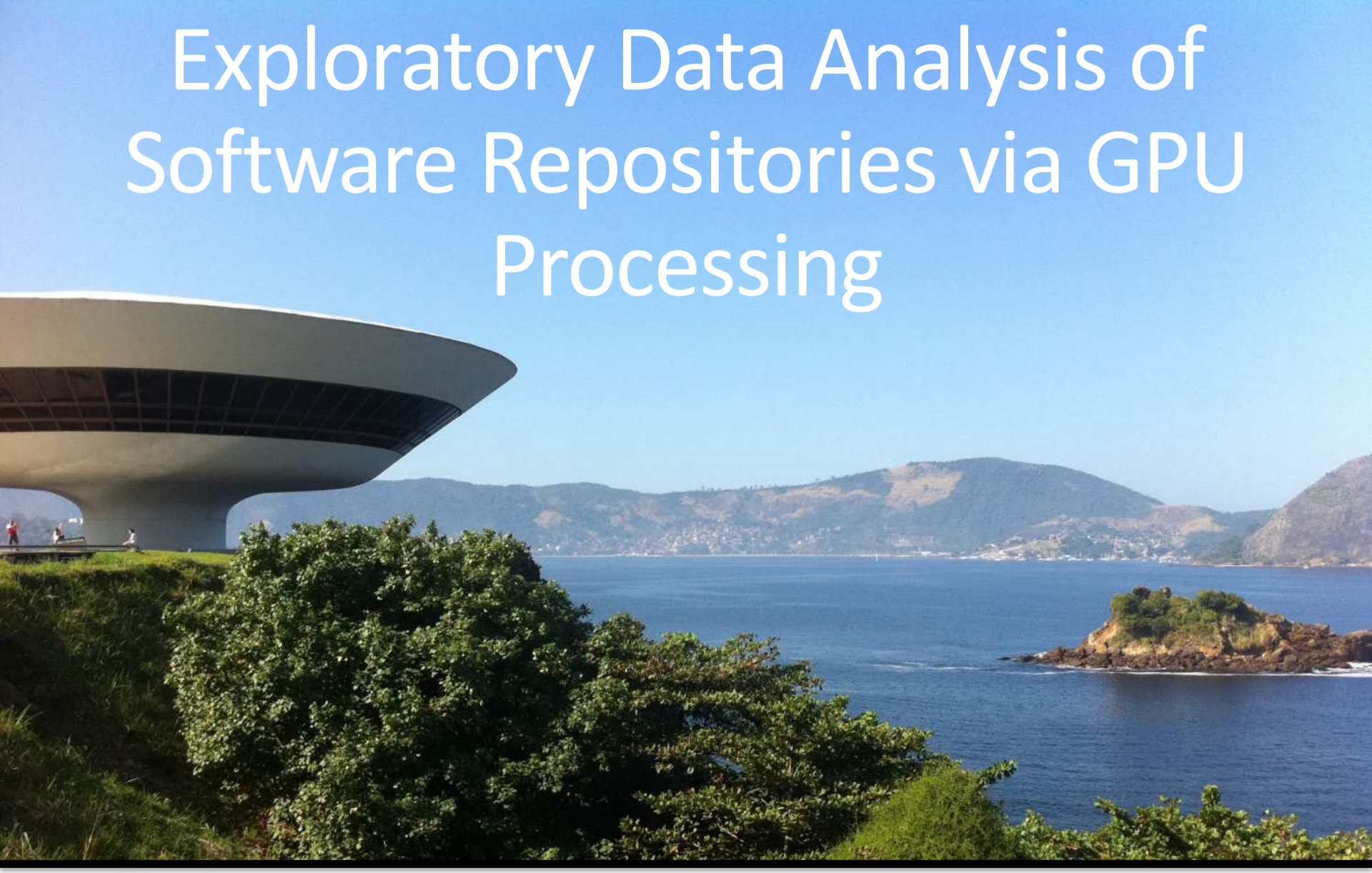
Exploratory Data Analysis of Software Repositories via GPU Processing



Instituto de Computação – UFF

Jose Ricardo
Esteban Clua
Leonardo Murta

Anita Sarma

University of Nebraska Lincoln

Introduction

Introduction

- Software development leaves behind the activity logs for mining relationships
  - Commits in a version system
  - Tasks in a issue tracker
  - Communication
- Finding them is not a trivial task
  - There is an extensive amount of data to be analyzed
  - Data is typically stored across different repositories
- Creating the right query is not a trivial task

Related Work

- Several research try to help in project explorations:
  - **Tesseract** and **CodeBook**: interactive investigation among files, developers, and commits through a graph of relationships, providing answer for specific questions
  - **Information Fragment:** allows users to compose queries and views from tasks, change sets, etc. to explore relationships between entities

Problems

- Different approach focusing on a particular development aspect (confirmatory analysis)
  - Allows exploration specific relationships set a priori
- Normally restrict the data to be analyzed in order to be feasible
- Most of them operate at a coarse grain (file)

Dominoes

- Approach that enables interactive exploratory data analysis at varying levels of granularity using GPU
- Organizes data from software repositories into multiple matrices
  - Each matrix is treated as Dominoes tile
  - Tiles can be combined through operations to generate derived tiles
    - Transposition, multiplication, addition, …

Dominoes

Commit #	Developer	Description
C_1	Alice	Change type of function parameter to compute the radius (Circle) and how to render it (in Shape)
C_2	Carlos	Change the side of Cone and how to render it
C_3	Alice	Change how a Shape is rendered
C_4	Alice	Calculation of how circumference and area are calculated using PI. Required modification on how to draw a Shape
C_5	Bob	Modify the height calculation of a cylinder and how it is rendered

Developer

Commit

5x3

Commit

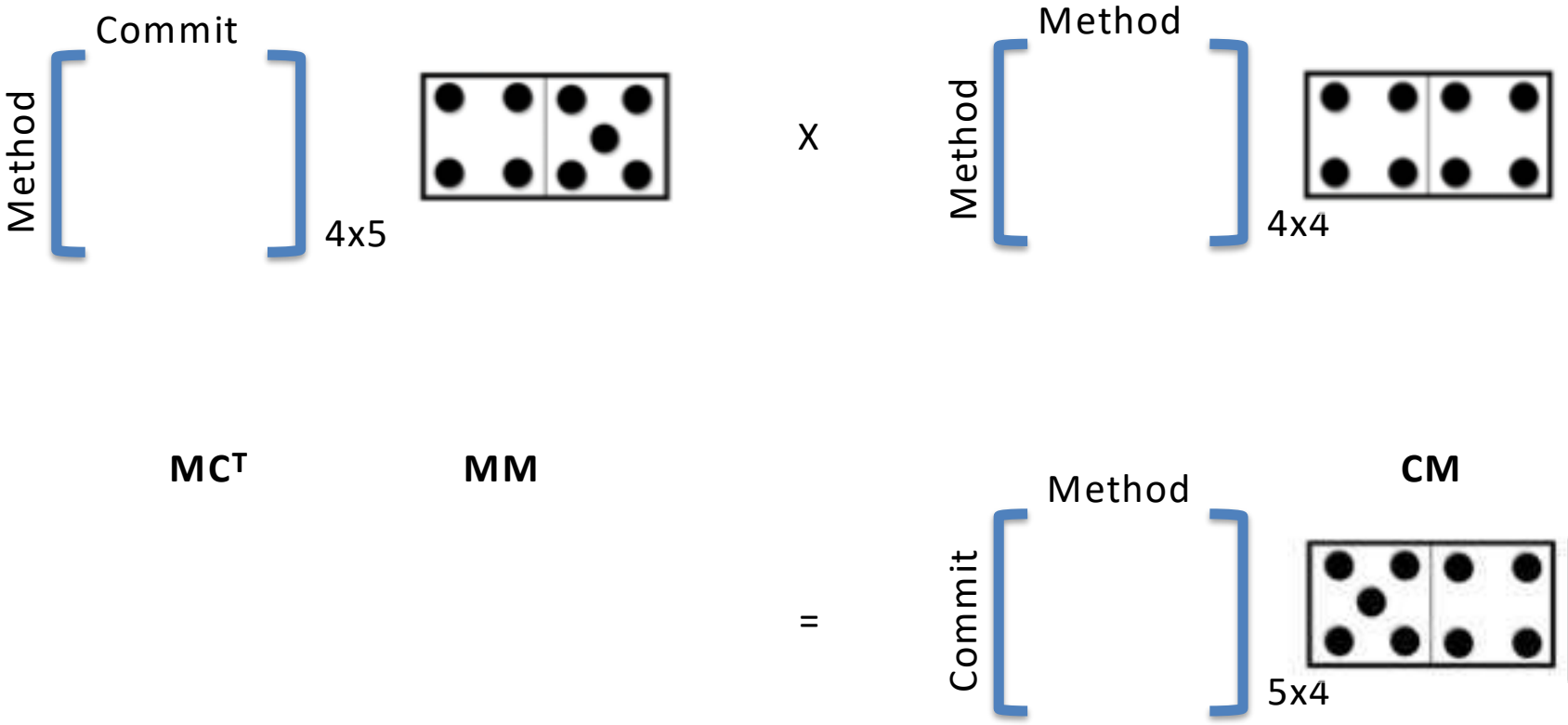
Commit #	Circle circumf()	Cylinder area()	Cone area()	Shape draw()
C_1	1	1	0	1
C_2	0	0	1	1
C_3	0	0	0	1
C_4	1	1	1	1
C_5	0	1	0	1

Method

4x5

Dominoes

- Dominoes' tiles resembles a Dominoes game, where the user can play with to build deeper relationships



Dominoes
Basic Building Tiles

- **[class|method] (ClM)**: composition among class and method
- **[commit|method] (CM)**: relationship between commits and methods
- **[developer|commit] (DC**): relationship between developers and their commits
- **[bug|commit] (BC**): relationship between commits and bugs

Dominoes
Some Derived Building Tiles

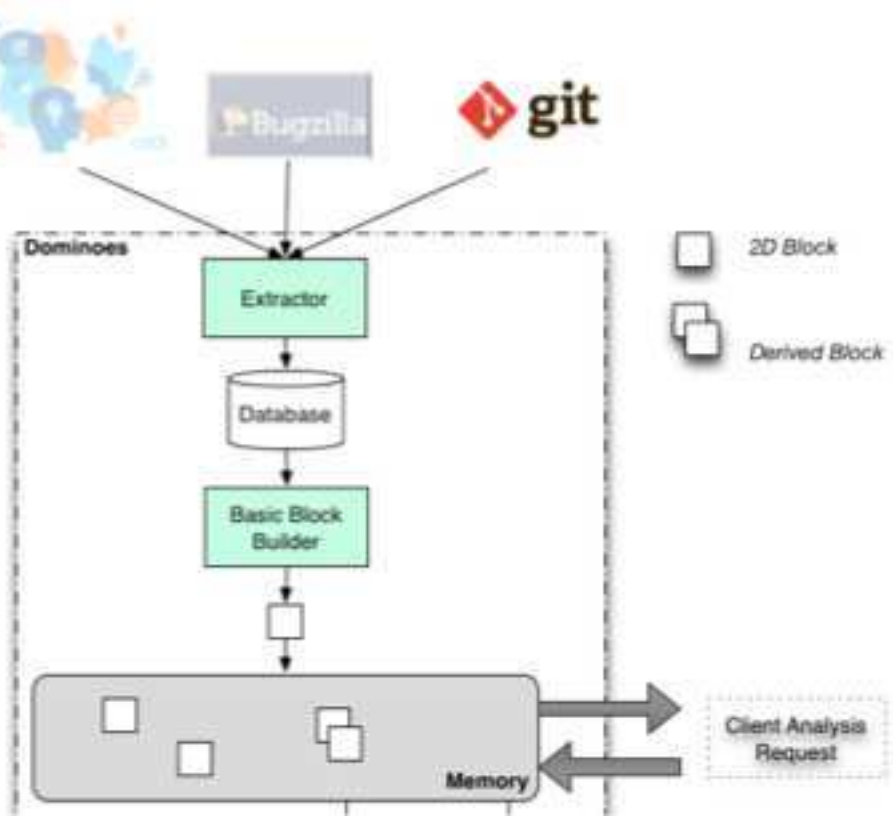
- **[method|method] (MM = $CM^T \times CM$)**: represents method dependencies
- **[class|class] (ClCl = $ClM \times MM \times ClM^T$)**: represents class dependencies
- **Bug-Method (BM = BC × CM)**: represents the methods that were changed to fix each bug. This matrix could be used to identify which methods are “buggy”

Drawback

- Exploration of such relationships at fine-grain is more accurate, however requires huge amount of data to be processed

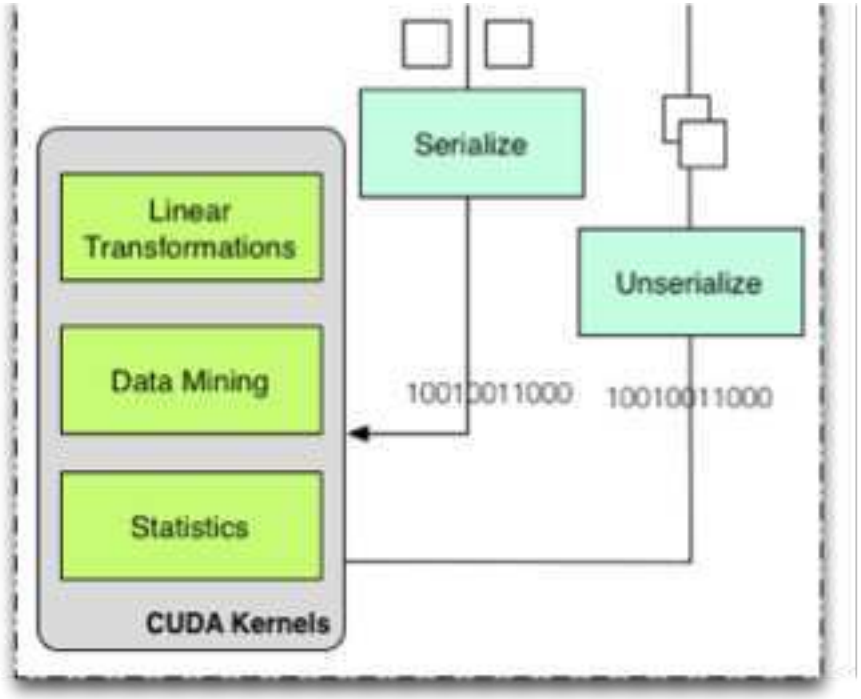
- Exploratory analysis operations are implemented over matrices using the GPU

Dominoes Architecture



- **Extractor module** gather information from repository and save to database
- **Basic block builder** is responsible to generate building blocks relationship from database

Dominoes Architecture



- Operations are performed in GPU using a **Java Native Interface** call
- Derived and basic building block **still in memory** for future use

Dependencies

- Many questions depend upon finding dependencies against methods (**MM (Method|Method** tile) to be answered

- How to find methods dependencies?
  - Syntactic analysis
  - Semantic analysis
  - Inference

Inference

- Based on how frequently files or methods were modified together
  - **Support**: proportion of transactions in the dataset that contains the item set

	A	B	C	D
A	**2**	2	1	2
B	2	**3**	1	3
C	1	1	**2**	2
D	2	3	2	**5**

Support

- Artifact **B** were modified with artifact **D** three times
- Transitive relationship
- $\mathbf{MM = CM^T \times CM}$

Inference

– **Confidence:** metric that defines how close a dataset should be modified together, given that a specific artifact is being modified.

	A	B	C	D
A	**1**	1	0.5	1
B	0.6	**1**	0.3	1
C	0.5	0.5	**1**	1
D	0.4	0.6	0.4	**1**

Confidence

- Modifying artifact **D** implies modifying **B** with 100% of confidence
- However, modifying **B** implies modifying **D** with 60% of confidence

$$\mathrm{M}^{\mathrm{conf}}[i,j] = \frac{\mathrm{M}^{\mathrm{sup}}[i,j]}{\mathrm{M}^{\mathrm{sup}}[i,i]}$$

Results

- Using Apache Derby to evaluate Dominoes.
  - Repository data from 08/11/2004 to 01/23/2014
  - Evaluation regarding support ✕ confidence
  - Evaluation regarding time processing

  - Comprises:
    - 7,578 commits
    - 36 distinct developers
    - 34,335 file changes
    - 305,551 method changes

Support ✕ Confidence

- Top 5 logical dependencies in terms of support with biggest difference in confidence.
  - Interface/Implementation case

Artifact A	Artifact B	Support	Conf. (A-B)	Conf. (B-A)
DataDictionary.java	DataDictionaryImpl.java	79	88%	37%
DD_Version.java	DataDictionaryImpl.java	45	78%	21%
LanguageConnectionContext.java	GenericLanguageConnectionContext.java	44	86%	48%

Support ✕ Confidence

- Top 5 logical dependencies in terms of support with biggest difference in confidence.
  - Composition case

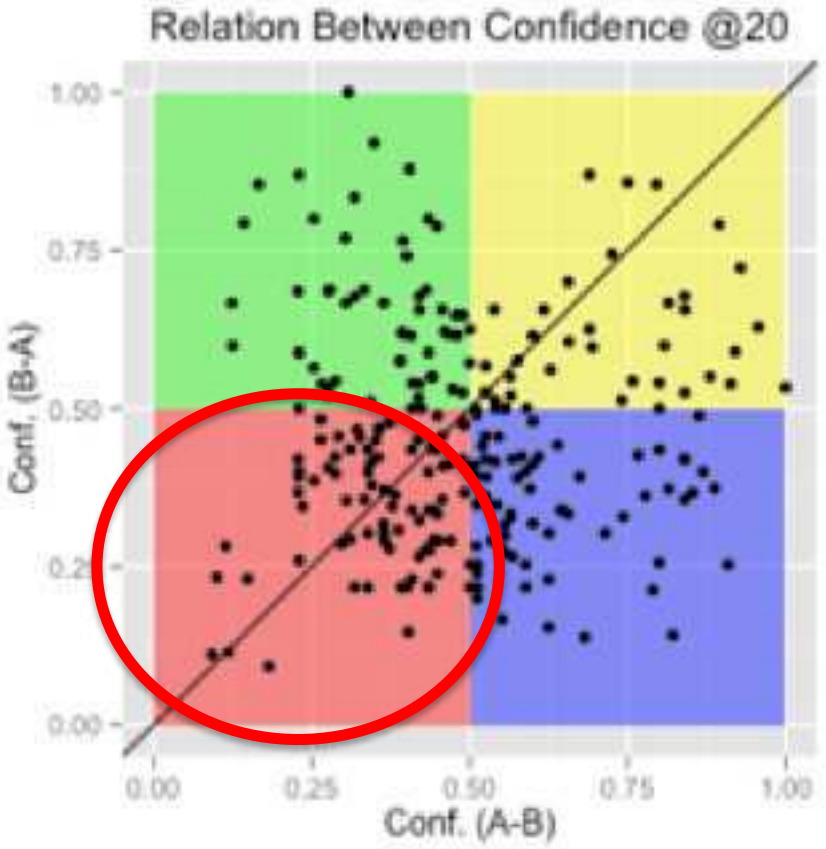
Artifact A	Artifact B	Support	Conf. (A-B)	Conf. (B-A)
DRDAConnThread.java	DRDAStatement.java	37	22%	68%

Support ✕ Confidence

- Top 5 logical dependencies in terms of support with biggest difference in confidence.
  - Class specialization case

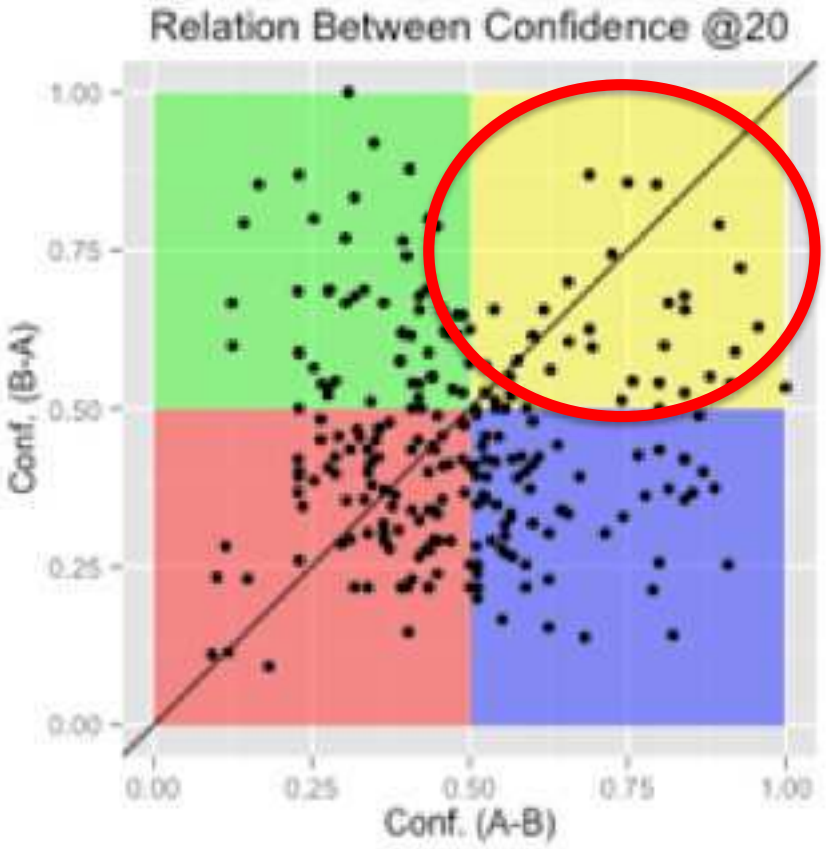
Artifact A	Artifact B	Support	Conf. (A-B)	Conf. (B-A)
ResultSetNode.java	SelectNode.java	36	54%	45%

Support ✕ Confidence



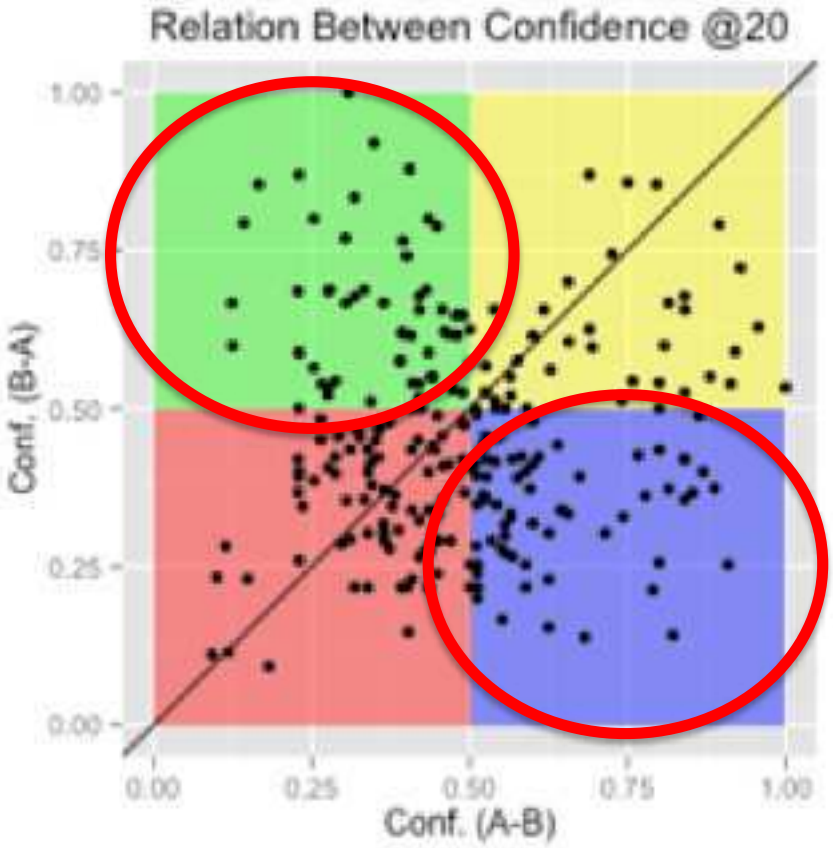
Weak bidirectional dependencies (less than 0.5)

Support ✕ Confidence



Strong bidirectional dependencies (above than 0.5)

Support ✕ Confidence



Unidirectional dependencies with highest divergence among confidence

Time

- Evaluation time (support and confidence).
  - [file|commit] (34,335 x 7,578)
    - CPU: 696 minutes | GPU: 0.7 minutes
  - [method|commit] (305,551 x 7,578)
    - CPU: N/A | GPU: 5 minutes

NVidia GeForce GTX580.
CPU Intel Core 2 Quad Q6600

Conclusions

- Dominoes is an exploratory tool that allows relationship manipulations
  - Basic and derived building block
- The use of support solely is not accurate
  - Need to identify the direction of relationships through confidence
- Using confidence for threshold is more natural as it represents normalized values

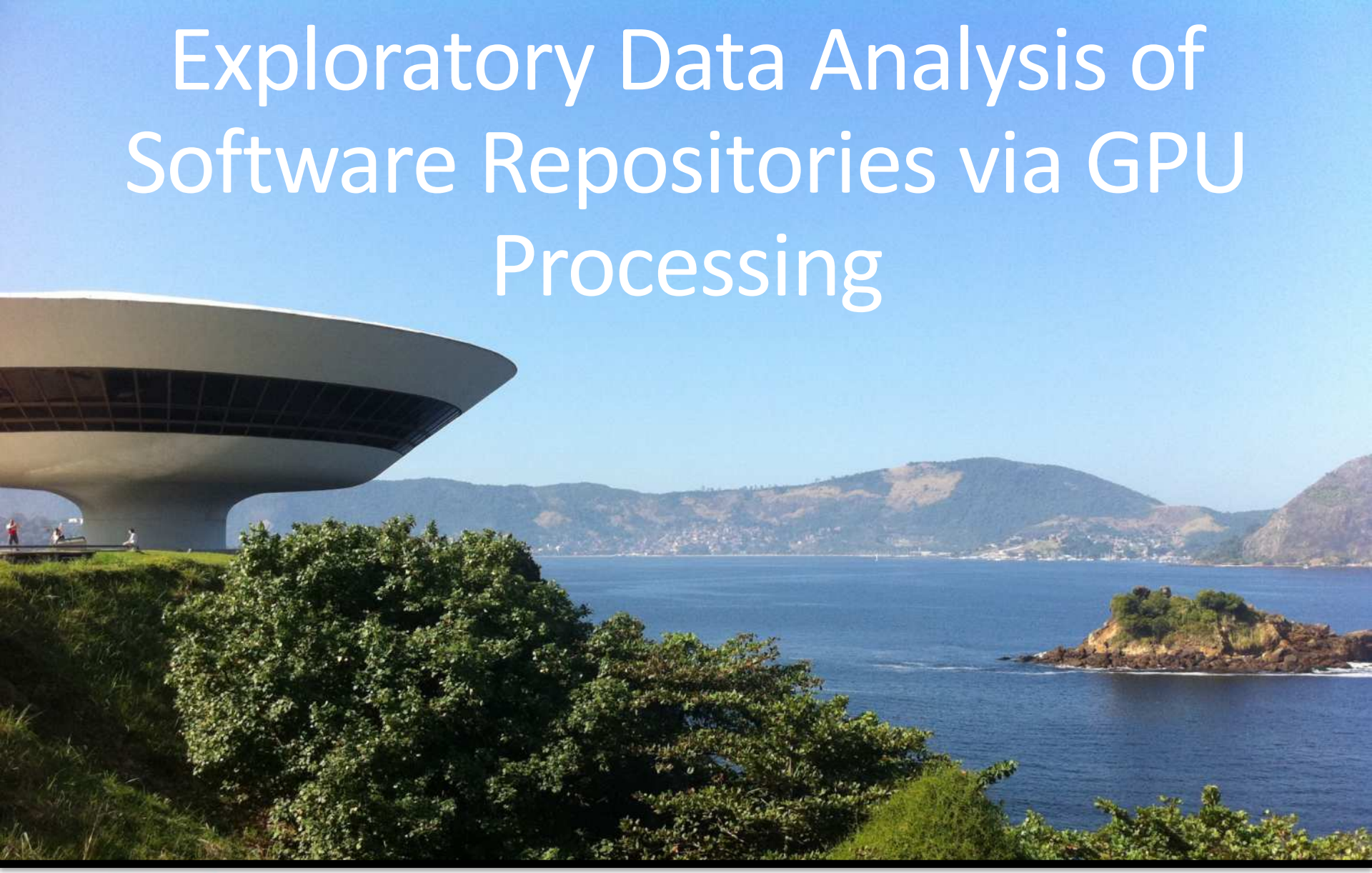
Conclusions

- Employment of GPU allows seamless relationship manipulations at interactive rates
  - Uses matrices underneath to represents building blocks
- Dominoes opens a new realm of exploratory software analysis, as endless combinations of Dominoes' pieces can be experimented in an exploratory fashion

Future Work

- Allow temporal analysis by considering time as third dimension (3D building tiles)
- Develop a GUI prototype
  - Provide real time visualizations for both basic and derived building tiles
- Apply Dominoes to answer different software engineering questions
  - Expertise depth ✕ breadth in a project

Exploratory Data Analysis of Software Repositories via GPU Processing



Instituto de Computação – UFF

Jose Ricardo
Esteban Clua
Leonardo Murta

Anita Sarma

University of Nebraska Lincoln