# Звіт

## Постановка задачі

Реалізована реалізація вирішення СЛАУ на розріджених матрицях.

Є можливість вибору методу декомпозиції матриці:

1. Холецького – 34 представлення симетричної додатноозначеної матриці у вигляді A=LLT де L — нижня трикутна матриця з додатними елементами на діагоналі
2. QR – представлення матриці у вигляді добутку унітарної та правої трикутної матриці
3. LU – представлення матриці у вигляді добутку нижньої трикутної матриці та верхньої трикутної матриці

Також реалізовано 3 способи впорядкування матриці:

1. symrcm (RCM-впорядкованість) – повертає такий вектор впорядкованості для симетричної матриці, який буде концентрувати ненульові елементи поблизу діагоналі.
2. symamd (симетрична впорядкованість) – повертає такий вектор впорядкованості стовпців для симетричної позитивно певної матриці S, який матиме більш розріджене LLT-розкладання
3. null (жодного) – програма працюватиме з оригінальною матрицею

## Технології використані у реалізації

Java – об'єктно-орієнтована мова програмування

CUDA – програмно-апаратна архітектура паралельних обчислень, яка дозволяє істотно збільшити обчислювальну продуктивність завдяки використанню графічних процесорів фірми Nvidia

MatrixMarket – формат для зберігання матриць, він передбачає зберігання впорядкованого за індексом рядка та стовпця в вихідній матриці лінійного масиву трійок "елемент, індекс рядка, індекс стовпця"

## Опис налаштування середовища для запуску рішення

### Налаштування Java проекту

У проект необхідно додати фреймворк Maven, і згодом у pom файл додати у секцію properties:

<jcuda.version>10.1.0</jcuda.version>

Варто зауважити що наразі вказана версія 10.1.0, але від вказаної версії буду залежати наші подальші дії.

У секцію dependencies:

<dependency><groupId>org.jcuda</groupId><artifactId>jcuda</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcublas</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcufft</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcusparse</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcusolver</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcurand</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jnvgraph</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcudnn</artifactId><version>${jcuda.version}</version></dependency>

<dependency><groupId>org.jcuda</groupId><artifactId>jcublas</artifactId><version>${jcuda.version}</version></dependency>

### Завантаження драйверів для CUDA

Необхідно перейти на сторінку:

<https://developer.nvidia.com/cuda-toolkit-archive>

І завантажити драйвера що відповідають версії jcuda що ми використовуємо у проекті.

### Додавання бібліотеки обробки MatrixMarket

Для роботи проекту також необхідна бібліотека MatrixMarketCSR з пакету de.javagl.matrixmarketreader, для цього необхідно викачати її вихідні коди з репозиторію:

<https://github.com/javagl/MatrixMarketReader>

Збілдити .jar артефакт, та згодом підключити його у бібліотеку