 **ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ**

**ФАКУЛТЕТ ПО КОМПЮТЪРНИ СИСТЕМИ И УПРАВЛЕНИЕ**

***КУРСОВ ПРОЕКТ***

ПО

Паралелно програмиране

**Алгоритми за паралелно програмиране с Java в Cloud**

Разработил: Йордан Павлов Павлов

Факултетен номер: 121211031

Проверил: проф. Боровска

*София 2015*

**Съдържание:**

Описание....................................................................................................................................3

Цел..............................................................................................................................................4

Системни изисквания...............................................................................................................5

Необходим софтуер..................................................................................................................6

Структурна схема......................................................................................................................7

Стъпки на изпълнение..............................................................................................................8

Реализация.................................................................................................................................9

Примерни резултати...............................................................................................................10

Анализ на резултатите............................................................................................................12

**Описание**

Проекта представлява Java Enterprise приложение, инсталирано на апликейшън сървър (Tomcat 7), извършващо умножение на матрици. Умножението се извъшва по два начина, последователно и паралелно (многонишково). Матриците се подават, чрез “влачене” (drag and drop) или избиране на текстови файлове. Всеки от файловете трябва да съдържа в себе си по една матрица. Ограничението при избор на файлове е броят на редовете от първата матрица да съответстват на броя на колоните от втората матрица. Файловете са пращат към сървър, където е инсталирано уеб приложението. Сървъра обработва файловете, умножава получените матрици последователно и паралелно, засича времето за умножение и връща резултат. При паралелното изпълнение, за умножението на ред с колона, се създава нова нишка, която отговаря за изчислението. Резултатът е текстови файл, в който е записано времето за умножение според различните подходи и резултантната матрица.

**Цел**

Целта на проекта е да се направи демонстрация на възможностите за парализирано изпълнение на операции с Java. Компилираният код се изпълнявва на отдалечен сървър.

Да се изготви статистика, сравнение и анализ на операциите изпълнявани последователно и паралелно.

**Системни изисквания**

* опреционна ситема Windows, Linux или Macintosh
* Java Runtime Environment 6
* Tomcat 7 или подобен Java Application Server
* WEB браузър потдържащ HTML 5

**Необходим софтуер**

Инсталацията на необходимия софтуер се извършва в следния ред, като се предполага, че преди това е инсталирана опреционна ситема Windows, Linux или Macintosh.

1. Изтегля се и се инсталира Java Runtime Environment 6 от следния адрес:

<http://www.oracle.com/technetwork/java/javase/downloads/jre6-downloads-1637595.html>

1. Изтегля и се инсталира Tomcat 7 от:

<http://tomcat.apache.org/download-70.cgi>

1. Стартира се Tomcat чрес стартиране на “**startup.sh”** за Linux и MAC или **“startup.bat”** за Windows, който се намира в **“\bin”** директорията.
2. Копира се **“parallel-programming-project.war”** файла в **“\webapps**” директорията.

**Структурна схема**

Структура:

**/src**

**/tu/sofia/parallel/programming/project**

**/multiplicator**

ConcurrentMatrixMultiplyingTask.java

MatrixMultiplicator.java

**/servlet**

MatrixFileGeneratorServlet.java

MatrixMultiplyServlet.java

**/WebContent**

**/css**

basic.css

bootstrap.css

dropzone.css

**/images**

spritemap.png

[spritemap@2x.png](mailto:spritemap@2x.png)

**/js**

dropzone.js

jquery-2.1.3.min.js

/**WEB-INF**

web.xml

index.html

pom.xml

**Стъпки на изпълнение**

1. За да достъпим приложението, трябва да използваме WEB браузър, поддържащ HTML 5 и да въведем адреса на приложението. Адресът на приложението се изчислява така:

http://<host>:<port>/parallel-programming-project/

* Ако WEB приложението е инсталирано на **application server** с порт **8080**, който се намира на **локалната машина**, то тогава адресът за достъп е:

<http://localhost:8080/parallel-programming-project/>

1. За да започне умножението на матрици, до полето **“Drop files to upload”** трябва да се хванат и завлекът файловете съдържащи матрици. Алтернативен начин за изпълнение, е чрез клик върху полето **“Drop files to upload”** и избор на файловете от менюто.

* WEB приложението предлага и генератор на квадратни матрици, който може да бъде достъпен така:

http://<host>:<port>/parallel-programming-project/matrix-generator/?size=<s>&radix=<r>

* Параметрите **size=<s>** и **radix=<r>**, получават положителни, целочисени числа. Чрез size се задава размера на матрицата, а чрез radix се задава диапазона на генерираните числа. В случай, че не се зададът параметрите, ще бъдат използвани стойности по подразбиране, който са съответно size=100 и radix=100. След като изпълнението на генератора приключи, генерираната матрицата ще бъде автоматично свалена като текстови файл с име **“generated-matrix.txt”**.

Пример:

<http://localhost:8080/parallel-programming-project/matrix-generator/?size=500&radix=100>

1. След като се изпълнят горните стъпки, ще се отвори нов прозорец, от който автоматично ще се изтегли текстови файл, съдържащ информация за необходимото време за изчисление на резултата и получената матрица. Възможно е при първоначално изпълнение на програмата, WEB браузърът да блокра отваращият се прозорец. За правилното изпълнение на WEB приложението, трябва да се разреши отварянето на изкачащи прозорци за този адрес.

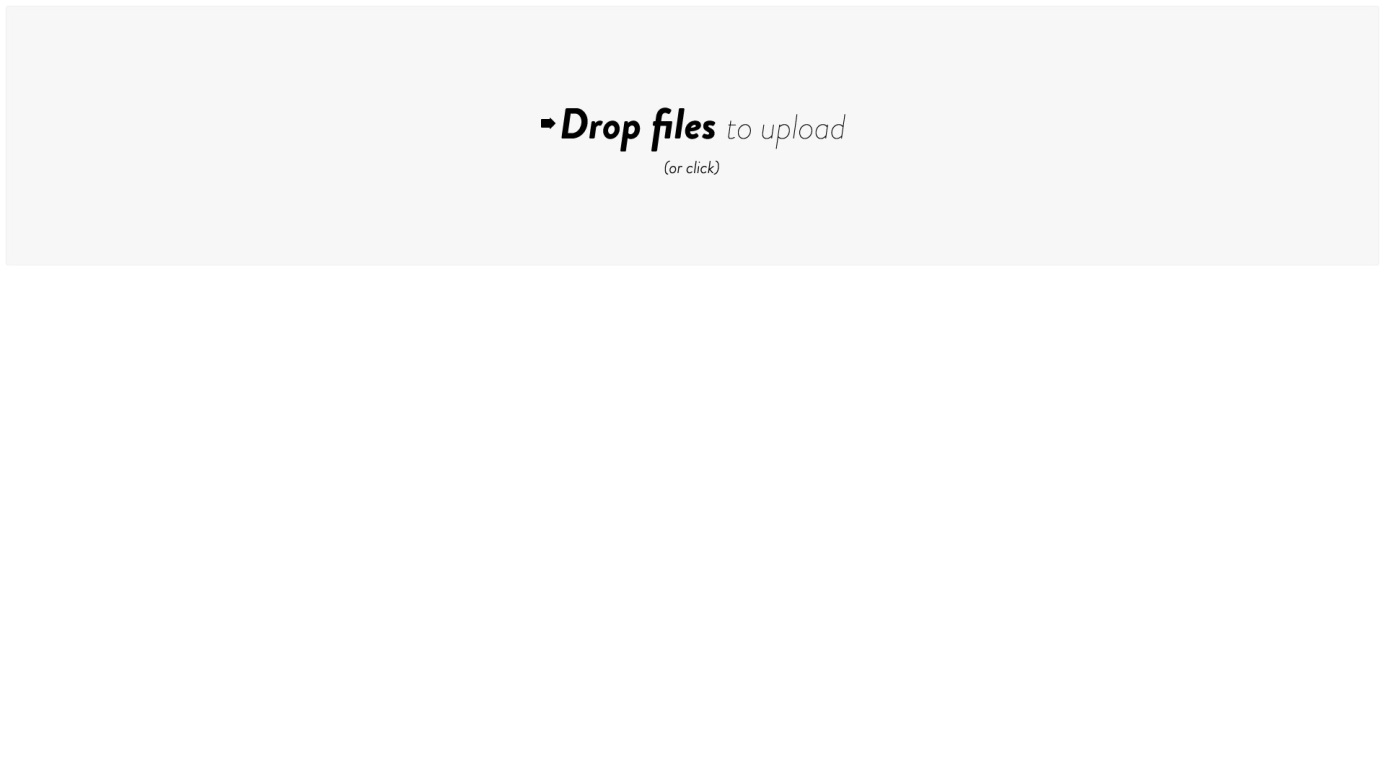
**Реализация**

Приложението се състои от 4 Java класа.

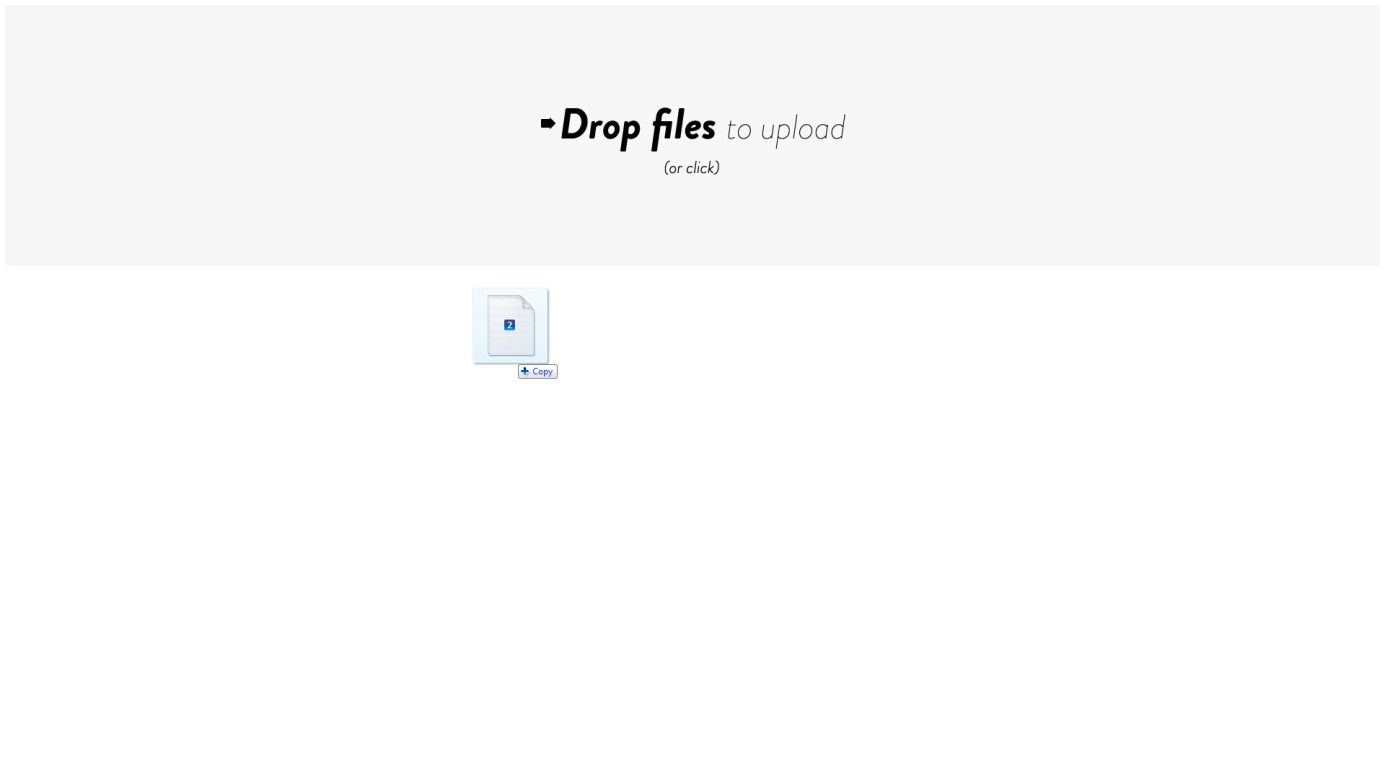
* **MatrixMultiplicator** – съдържа логиката за последователно и паралелно умножението на матрици.
* **ConcurrentMatrixMultiplyingTask** – съдържа логиката за всяка една нишка използвана при паралелното умножение на матрици. За всяка колона от първата матрица се прави отделна инстанция (нишка) на този клас.
* **MatrixMultiplyServlet** – съдържа логиката свързваща входните заявки към сървъра и навигираща процесите на приложението.
* **MatrixFileGeneratorServlet** – съдържа логиката за генериране на текстови файлове съдържащи матрици с произволни числа в тях.

**Примерни резултати**

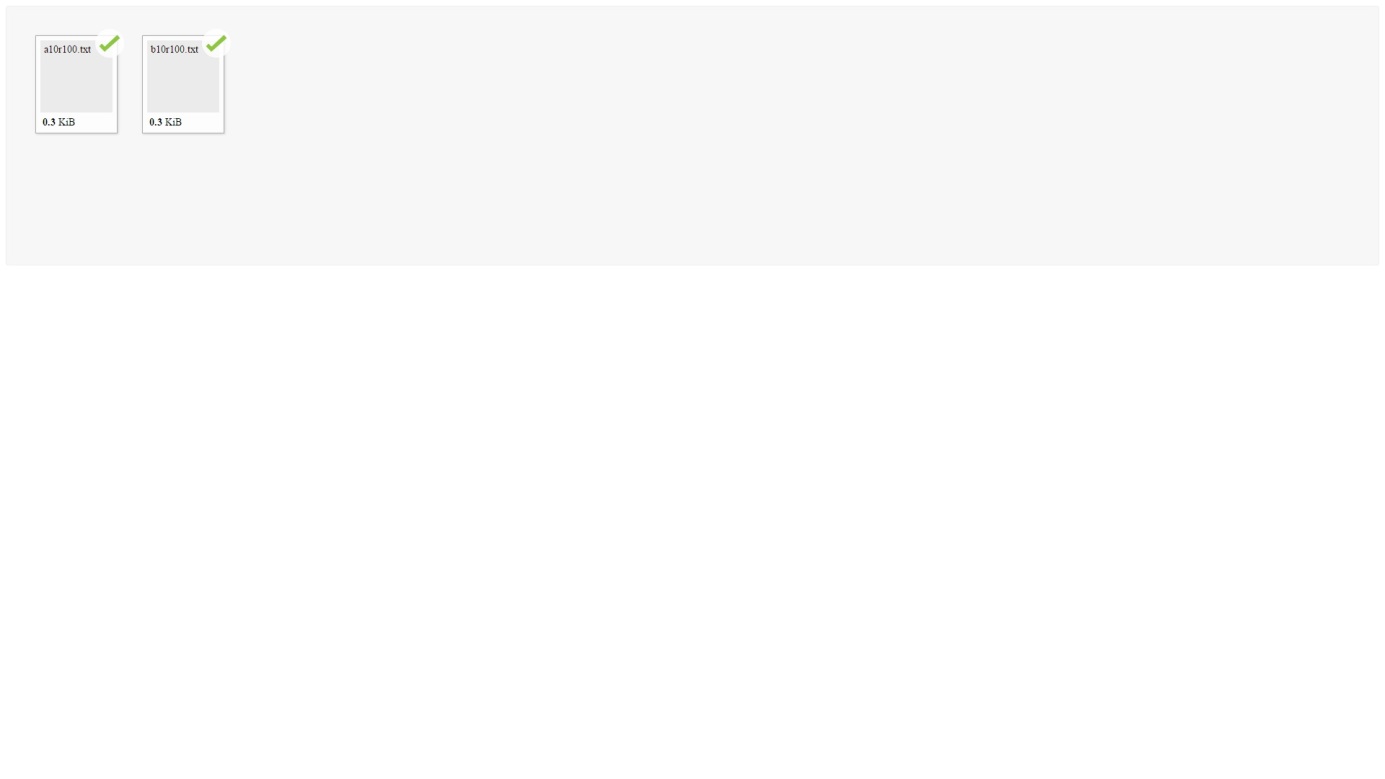
Начален екран:



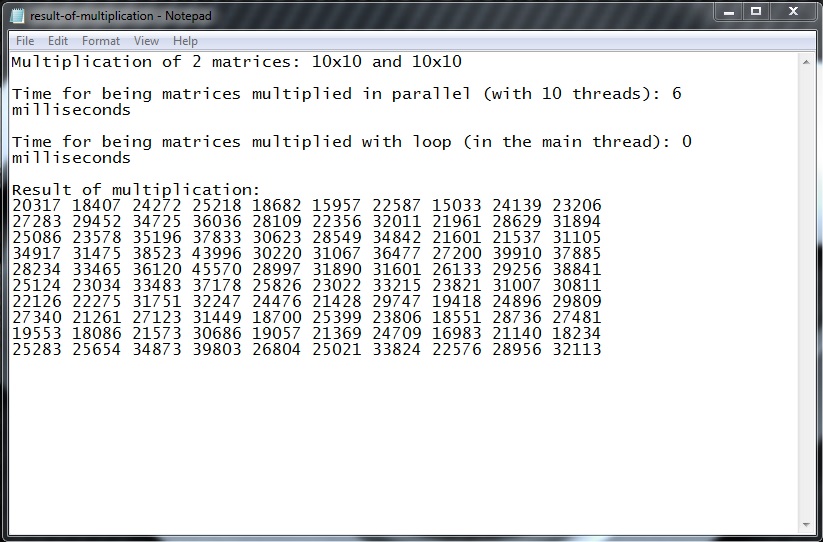
Влачене на файлове и изпращане към сървъра:



Успешно изпратени файлове:



Резултат от изпълнението:



**Анализ на резултатите**

Системни параметри:

* Локална машина:
  + Windows x64
  + Intel i3 M 370 2.40 GHz
  + 4 GB RAM
* Cloud:
  + Linux
  + 1 core
  + 2 GB RAM

Тестови данни:

Използват се квадратни матрици със следните размери и числа:

* + 10x10 с произволни числа в диапазона до 100
  + 10x10 с произволни числа в диапазона до 1000
  + 100x100 с произволни числа в диапазона до 100
  + 100x100 с произволни числа в диапазона до 1000
  + 1000x1000 с произволни числа в диапазона до 100
  + 1000x1000 с произволни числа в диапазона до 1000

Резултати:

* Локална машина:

* Cloud:

Извод:

Очаквано при малък размер на входните данни последователното умножение е по-бързо от паралелното. При размер от 1000x1000 паралелното изпълнение се извършва по-бързо от последователното. Локалната машина разполага с 4 физически процесора, а машината в cloud-a, само с 1 физически процесор. Това обяснява защо времето за паралелна обработка в cloud-a, при матрица с размер 1000x1000 и с числа в диапазона до 100, е приблизително равно на времето за последователна обработка. При увеличаване на диапазона на числата, паралелната обработка в cloud-a е по-бърза от последователната.