Laborator 2 – Stefan Sebastian 235

Programare paralela si distribuita

Analiza cerintelor

Se cere sa se implementeze un program, atat in Java cat si in C++, care aplica un operator binar generic pe elementele a doua matrici si afiseaza rezultatul intr-un fisier. Programul va contine o implementare a tipului de date numar complex iar matricile folosite vor avea operanzi generici. Sa se realizeze o analiza comparativa a timpului de rulare pentru diversi operatori folosind tipurile de date double si numar complex.

Constrangeri

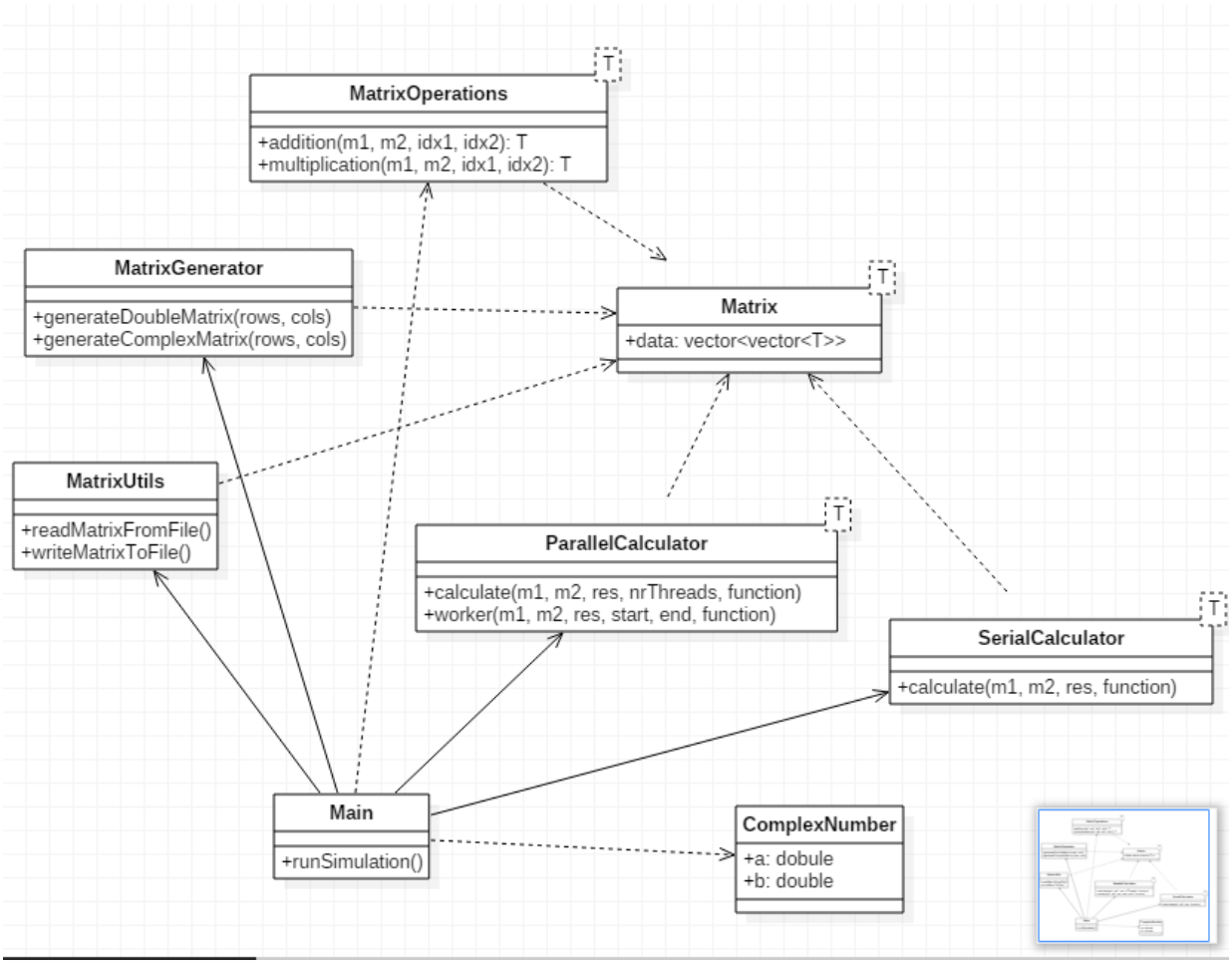
Numarul de threaduri folosit trebuie sa fie un parametru care poate fi modificat inainte de inceperea executiei. Impartirea datelor pe threaduri trebuie sa urmeze o distributie echilibrata. Programul va afisa pe ecran timpul de executie al operatiilor.

Proiectare + detalii implementare C++

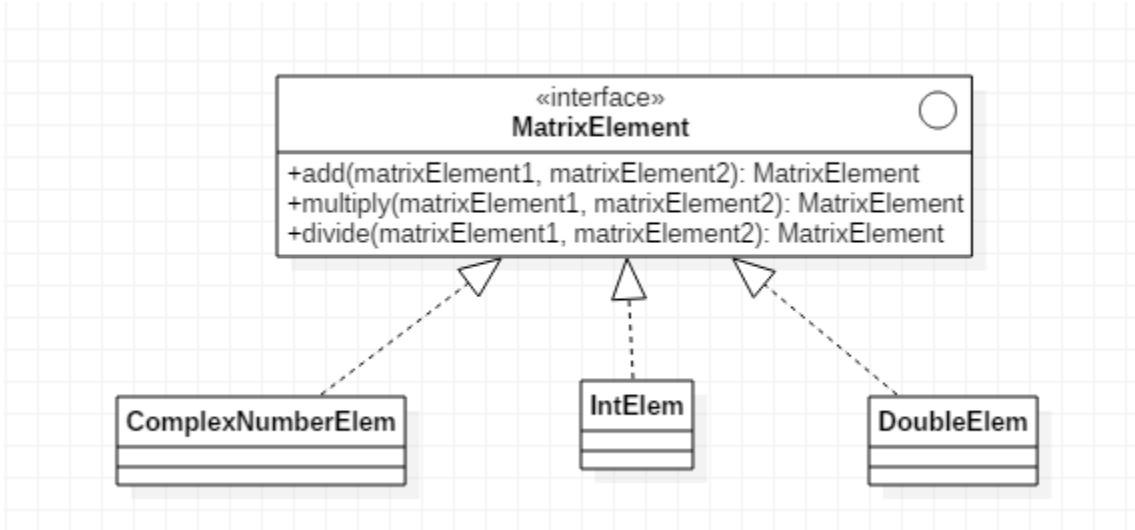
Am folosit clasa Matrix cu un template pentru a reprezenta o matrice generica. Operatiile paralele sunt prelucrate de clasa ParallelCalculator care primeste matricile, numarul de threaduri si un std::function reprezentand un operator ce va fi aplicat intre elementele matricilor. Parametrii acestei functii sunt 2 matrici si 2 numere intregi reprezentand o pozitie pe matrice. Am ales aceasta abordare pentru a permite si operatia de inmultire a matricelor. Functia are de asemenea un template si returneaza un obiect de tipul dat prin acesta. Operatiile seriale sunt prelucrate de clasa SerialCalculator care primeste de asmenea un std::function de acelasi tip ca si ParallelCalculator.

MatrixOperations contine un set de functii de aceeasi forma cu cele acceptate de metodele de prelucrare a matricilor care sunt folosite pentru testarea programului. MatrixGenerator defineste metode pentru a genera matrici de diferite tipuri, populate cu numere aleatoare. MatrixUtils contine un set de metode utilitare pentru lucrul cu matrici precum citirea si scrierea acestora in fisiere.

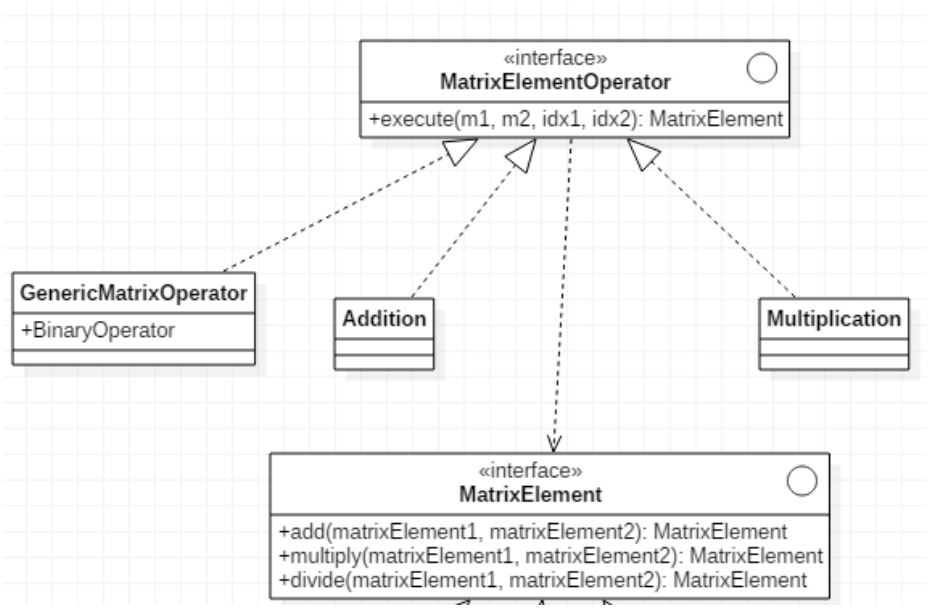
Impartirea datelor pe threaduri am facut-o prin considerarea celulelor ca fiind numerotate si ‘spargerea’ matricei in (nrLin * nrCol) / nrThreads bucati. Metoda worker primeste un punct de start si de sfarsit, le determina coordonatele si aplica std::function primit pe celulele din intervalul dat.



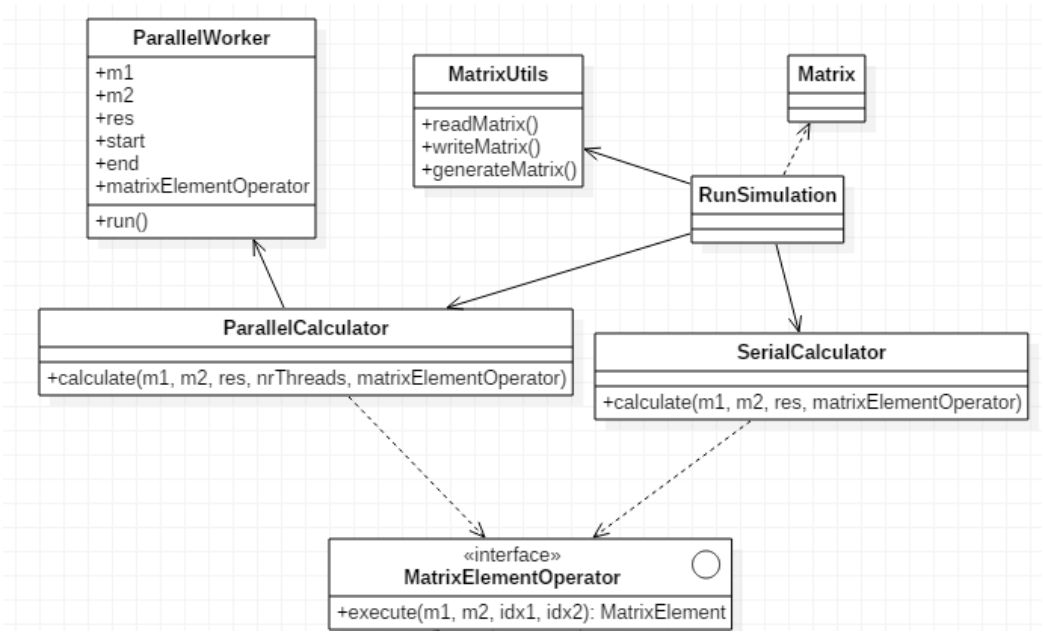
Proiectare + detalii de implementare Java



Pentru a reprezenta matrici cu elemente generice am definit o interfata MatrixElement care defineste o serie de operatii de baza dintre tipurile de date care pot fi salvate in matrice.



Pentru a defini operatiile care pot fi aplicate pe elementele matricilor am folosit interfata MatrixElementOperator care, similar cu functia din C++, primeste ca parametrii 2 matrici si 2 indici (reprezentand pozitia) pentru a permite si operatia de inmultire a matricilor. Pentru a defini o operatie noua se poate instantia GenericMatrixOperator, o clasa care implementeaza aceasta interfata si aplica un operator pe cele 2 elemente ale matricilor, folosind un BinaryOperator sau o functie lambda. O alta metoda este definirea unei alte clase care implementeaza MatrixElementOperator.



Impartirea datelor pe threaduri precum si restul claselor si metodelor folosite sunt similare cu varianta in C++ a programului.

Cazuri de testare

Testarea s-a facut pe un procesor intel i5-5200U, 2.2GHz, 2 cores, 4 threads.

Inmultire

Limbaj	Tip de date	Dimensiune	Nr Threads	Timp serial	Timp paralel
C++	Double	1000x1000	2	11	7
C++	Double	1000x1000	4	11	11
C++	Double	1000x1000	6	11	6
C++	Double	1000x1000	8	19	22
Java	Double	1000x1000	2	63	322
Java	Double	1000x1000	4	40	456
Java	Double	1000x1000	6	22	346
Java	Double	1000x1000	8	400	21
C++	Complex	1000x1000	2	48	25
C++	Complex	1000x1000	4	39	17
C++	Complex	1000x1000	6	40	21
C++	Complex	1000x1000	8	41	24
Java	Complex	1000x1000	2	220	38
Java	Complex	1000x1000	4	455	130
Java	Complex	1000x1000	6	28	221
Java	Complex	1000x1000	8	176	21

Operatorul 1/(1/a + 1/b)

Limbaj	Tip de date	Dimensiune	Nr Threads	Timp serial	Timp paralel
C++	Double	1000x1000	2	14	13
C++	Double	1000x1000	4	17	10
C++	Double	1000x1000	6	14	9
C++	Double	1000x1000	8	14	9
Java	Double	1000x1000	2	267	26
Java	Double	1000x1000	4	439	21
Java	Double	1000x1000	6	485	28
Java	Double	1000x1000	8	335	17
C++	Complex	1000x1000	2	391	43

C++	Complex	1000x1000	4	383	25
C++	Complex	1000x1000	6	381	28
C++	Complex	1000x1000	8	385	27
Java	Complex	1000x1000	2	170	333
Java	Complex	1000x1000	4	503	175
Java	Complex	1000x1000	6	146	305
Java	Complex	1000x1000	8	256	435

Analiza comparativa

Operatiile pe tipuri primitive (double) tind sa fie mai rapide decat operatiile pe tipuri definite de noi (numere complexe), diferenta se observa in special in timpul operatiilor seriale.

Pentru majoritatea testelor implementarea C++ scoate timpi mai buni decat implementarea Java.

Diferenta dintre operatorii folositi se observa la Java unde au fost nevoie de instructiuni suplimentare pentru a implementa operatia 1 / (1 / a + 1 / b). Am folosit metode separate pentru fiecare tip de date pentru a putea defini conceptul de “1”.