Tema 2

- Arhitectura sistemelor de calcul -

Anton Puiu
Grupa 332CA
Facultatea de Automatică şi Calculatoare
Universitatea Politehnica Bucureşti
anton.puiu@email.com

Abstract

Acest document conţine descrierea implementării ecuaţiei matriceale:

$$C = B \times A^t + A^2 \times B$$

unde A este o matrice *superior triunghiulară*, iar B o matrice oarecare, ambele pătratice, în următoarele variante:

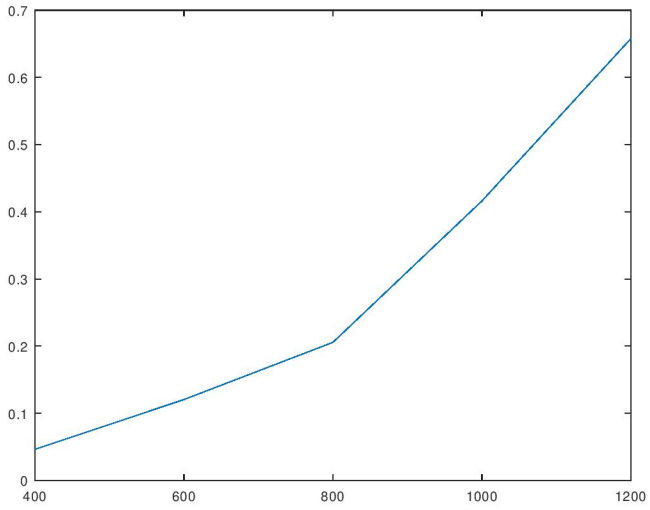
1. Utilizînd biblioteca **BLAS**
2. Neoptimizată
3. Optimizată la nivel de cod
4. Optimizată la nivel de compilator, fara opţiuni suplimentare
5. Optimizată la nivel de compilator, cu opţiuni suplimentare

1 Varianta BLAS

Avînd in vedere documentaţia bilbiotecii BLAS, se observă faptul că pentru ecuaţia matriceală dată, o funcţie potrivită ar fi dtrmm. Astfel, în implementare au fost necesare **trei** apeluri ale acestei funcţii pentru rezolvarea ecuaţiei, asociate următoarelor operaţii:

1. $D = B \times A^t$
2. $E = A \times B$
3. $E = A \times E$

urmînd ca dupa obţinerea tuturor rezultatelor intermediare, reţinute în matricele D şi E, rezultatul final, reprezentat de suma acestor doua matrice, să fie reţinut în matricea C.
Performanţele obţinute utilizînd aceasta bibliotecă pot fi observate în graficul de mai jos.



2 Varianta neoptimizată

În continuare vor fi utilizate următoarele notaţii:

$$BAt = B \times A^t$$
$$AB = A \times B$$

în care se consideră matricele A şi B de dimensiune 4×4, fără a restrînge generalitatea problemei, de forma:

$$A = \begin{pmatrix} a11 & a12 & a13 & a14 \\ 0 & a22 & a23 & a24 \\ 0 & 0 & a33 & a34 \\ 0 & 0 & 0 & a44 \end{pmatrix}; \quad B = \begin{pmatrix} b11 & b12 & b13 & b14 \\ b21 & b22 & b23 & b24 \\ b31 & b32 & b33 & b34 \\ b41 & b42 & b43 & b44 \end{pmatrix};$$

Astfel, rezultatul operaţiei matriceale $B \times A^t$ este următorul:

$$BAt = \begin{pmatrix} a11\,b11 + a12\,b12 + a13\,b13 + a14\,b14 & a22\,b12 + a23\,b13 + a24\,b14 & a33\,b13 + a34\,b14 & a44\,b14 \\ a11\,b21 + a12\,b22 + a13\,b23 + a14\,b24 & a22\,b22 + a23\,b23 + a24\,b24 & a33\,b23 + a34\,b24 & a44\,b24 \\ a11\,b31 + a12\,b32 + a13\,b33 + a14\,b34 & a22\,b32 + a23\,b33 + a24\,b34 & a33\,b33 + a34\,b34 & a44\,b34 \\ a11\,b41 + a12\,b42 + a13\,b43 + a14\,b44 & a22\,b42 + a23\,b43 + a24\,b44 & a33\,b43 + a34\,b44 & a44\,b44 \end{pmatrix};$$

avînd următoarea formula de calcul:

$$BAt_{ij} = \sum_{k=j}^{n} a_{jk} \times b_{ik}$$

iar rezultatul operaţiei $A \times B$ este:

$$AB = \begin{pmatrix} a11\,b11 + a12\,b21 + a13\,b31 + a14\,b41 & a11\,b12 + a12\,b22 + a13\,b32 + a14\,b42 & a11\,b13 + a12\,b23 + a13\,b33 + a14\,b43 & a11\,b14 + a12\,b24 + a13\,b34 + a14\,b44 \\ a22\,b21 + a23\,b31 + a24\,b41 & a22\,b22 + a23\,b32 + a24\,b42 & a22\,b23 + a23\,b33 + a24\,b43 & a22\,b24 + a23\,b34 + a24\,b44 \\ a33\,b31 + a34\,b41 & a33\,b32 + a34\,b42 & a33\,b33 + a34\,b43 & a33\,b34 + a34\,b44 \\ a44\,b41 & a44\,b42 & a44\,b43 & a44\,b44 \end{pmatrix};$$

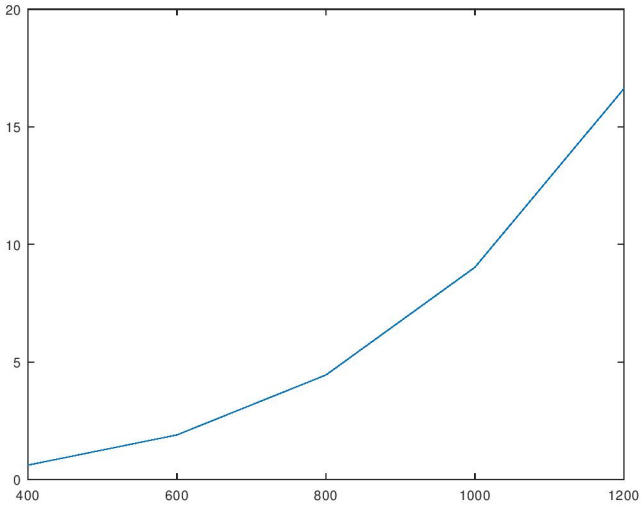
pentru care formula de calcul este următoarea:

$$AB_{ij} = \sum_{k=i}^{n} a_{ik} \times b_{kj}$$

Pentru a obţine termenul $A^2 \times B$, similar cu implementarea **BLAS**, se poate aplica formula AB_{ij}, în care iniţial se utilizeaza matricele A şi B, urmînd ca aceasta formulă să fie reaplicată, avînd în locul matricei B rezultatul calculat anterior.
Pentru obţinerea rezultatului ecuaţiei iniţiale, se realizează operaţia de adunare a matricelor între matricea BAt şi cea calculată anterior.

Avînd în vedere faptul că pentru calculul matricelor BAt şi AB sunt necesare matricele A şi B, am putut realiza acest calcul în aceeaşi buclă **for**.

Performanţele obţinute utilizînd formulele anterioare pot fi observate în graficul următor.

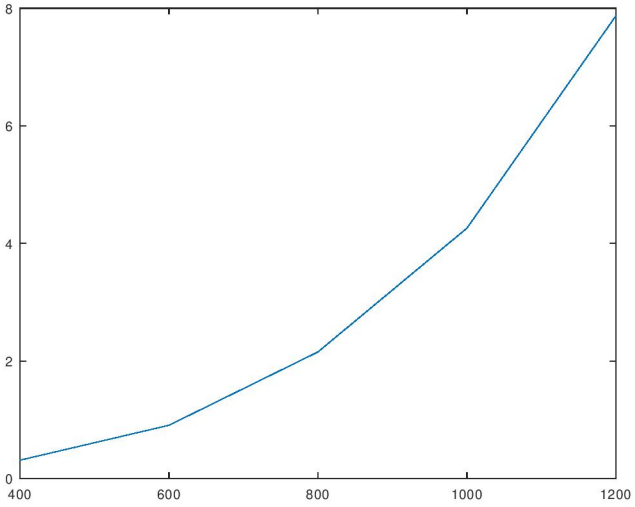


3 Varianta optimizată

Spre deosebire de varianta neoptimizată, se pot observa următoarele modificări:

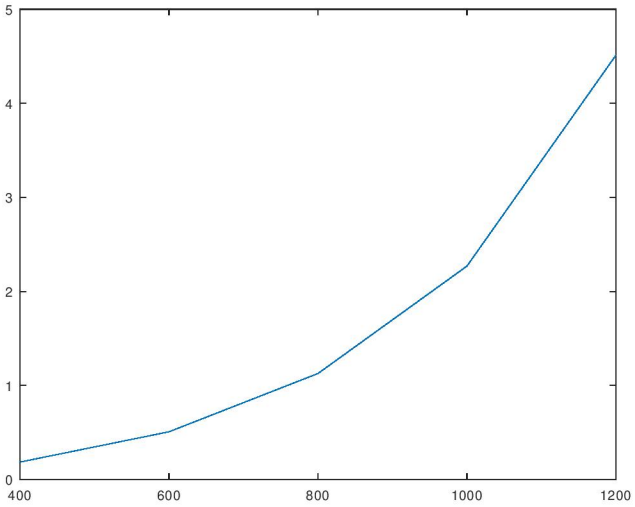
- Utilizarea unei variabile *index*, în care a fost reţinută valoarea $i * N$, astfel evitînd recalcularea acestei valori în diferite părti ale programului
- Utilizarea variabilelor de tip **pointer**, în defavoarea accesului la memorie prin intermediul variabilei ce reprezintă matricea, astfel economisind **o înmulţire şi o adunare** pentru *fiecare acces în matrice*
- Utilizarea cuvîntul cheie **register**, pentru a asigura viteza de acces egală cu viteza de lucru a procesorului pentru toate variabilele menţionate anterior, împreuna cu variabila în care se reţine suma din formulele menţionate anterior.

În următorul grafic se poate observa impactul pe care l-au avut optimizările făcute, chiar dacă nu s-a ţinut cont de **memoria cache**.



4 Varianta optimizată la nivel de compilator, fără opţiuni suplimentare

Se poate observa că performanţa obţinută utilizînd optimizările la nivel de compilator au crescut de aproximativ două ori, la fel ca în cazul variantei optimizate.



5 Varianta optimizată la nivel de compilator, cu opţiuni suplimentare

Pentru a spori performanţa obţinută anterior, am ales utilizarea următoarelor opţiuni:

- **-funroll-loops**, pentru a elimina cazul în care o instrucţiune de salt condiţionat este executată de procesor
- **-funsafe-math-optimizations**, întrucît asociativitatea elementelor nu este prezentă.

Rezultatele obţinute sunt prezente în graficul următor.