Performance Analysis of Parallel Algorithms

Asist. Felician ALECU
Catedra de Informatică Economică, ASE Bucureşti

The main reason of parallelization a sequential program is to run the program faster. When running a parallel program on a real parallel system there is an overhead coming from processors load imbalance and from communication times needed for changing data between processors and for synchronization. This is the reason why the execution time of the program will be greater than the theoretical value.
Keywords: *parallel processing, execution time, parallel speedup, efficiency, costs, parallelization.*

Timpul de execuţie al unui algoritm secvenţial (T_s), se exprimă ca funcţie de dimensiunea datelor de intrare. Timpul de execuţie paralelă (T_p) reprezintă timpul scurs din momentul în care programul paralel este lansat în execuţie până când ultimul procesor îşi încheie execuţia. Acesta depinde însă nu numai de dimensiunea intrărilor ci şi de arhitectura calculatorului paralel şi de numărul de procesoare existente în sistem.

În general, timpii de execuţie sunt exprimaţi cu ajutorul numărului de operaţii elementare ce sunt executate, în cazul cel mai defavorabil, pentru rezolvarea unei probleme de dimensiune dată.

Exprimarea timpului de execuţie în funcţie de dimensiunea problemei poate conduce la obţinerea unor expresii complicate care sunt reduse la forme mai simple cu ajutorul cu ajutorul limitelor asimptotice (O - limita asimptotică superioară, Ω - limita asimptotică inferioară, Θ - ordinul exact).

Indicatorii de performanţă asociaţi sistemelor paralele sunt accelerarea, eficienţa şi costul paralel.

Accelerarea paralelă se notează cu S şi reprezintă câştigul de viteză ce se obţine datorită execuţiei algoritmului pe un sistem paralel în comparaţie cu timpul de execuţie în varianta secvenţială. Acest indicator se exprimă ca raport între timpul de execuţie în cazul cel mai defavorabil al celui mai rapid algoritm secvenţial care rezolvă o anumită problemă şi timpul de execuţie necesar unui algoritm paralel pentru a soluţiona aceeaşi problemă pe un sistem cu p procesoare identice cu cel al calculatorului secvenţial: $S = \frac{T_s}{T_p}$

Teoretic, accelerarea paralelă nu poate avea o valoare mai mare ca p, numărul de procesoare. O accelerare paralelă egală cu numărul de procesoare din sistem poartă numele de accelerare liniară.

În conformitate cu legea lui Amdahl, chiar si într-un sistem paralel ideal este foarte dificil de obţinut o accelerare paralelă egală cu numărul de procesoare datorită faptului că în cadrul oricărui program există o fracţie α care nu poate fi paralelizată şi care trebuie executată secvenţial. Restul de $(1 - \alpha)$ paşi de calcul se pot executa în paralel pe procesoarele disponibile în sistem. Astfel, timpul paralel de execuţie şi accelerarea paralelă devin:

$$T_p = T_s \cdot \alpha + T_s \cdot (1-\alpha)/p$$

$$S = \frac{T_s}{T_p} = \frac{T_s}{T_s \cdot \alpha + T_s \cdot (1-\alpha)/p} = \frac{1}{\alpha + (1-\alpha)/p} = \frac{p}{\alpha \cdot (p-1) + 1}$$

Atunci când $p \to \infty$ avem: $\lim_{p\to\infty} S = \frac{1}{\alpha}$

Din acest motiv, accelerarea maximă care se poate obţine atunci când o fracţie α a programului nu poate fi paralelizată este $1/\alpha$ indiferent de numărul de procesoare din sistem. De exemplu, dacă în cazul unui sistem cu 4 procesoare o fracţie de 20% dintr-un program

nu poate fi paralelizată, timpul paralel de execuţie si accelerarea paralelă vor fi:

$$T_p = T_s \cdot 0.2 + T_s \cdot 0.8/4 = 0.4 \cdot T_s$$

$$S = \frac{T_s}{0.4 \cdot T_s} = \frac{1}{0.4} = 2.5$$

Timpul paralel de execuţie va fi 40% din cel secvenţial ceea ce înseamnă că programul paralel va fi doar de 2.5 ori mai rapid decât cel secvenţial (figura 1).

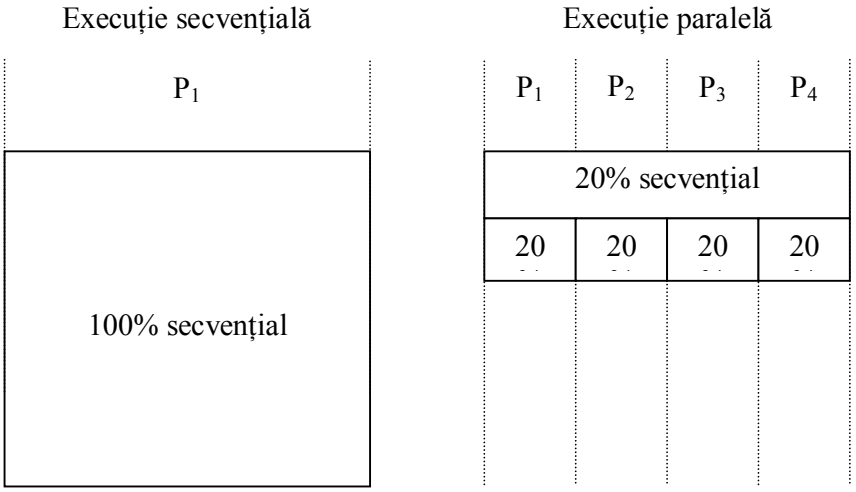

Fig. 1. Execuţia unui program în cadrul unui sistem ideal

Accelerarea maximă care poate fi obţinută este 1/0.2 = 5 ceea ce semnifică faptul că timpul paralel de execuţie nu va fi niciodată mai mic de 20% din cel secvenţial, indiferent de numărul de procesoare din sistem.

Legea lui Amdahl exprimă în mod clar necesitatea minimizării fracţiei α ce nu poate fi paralelizată prin stabilirea unei limite superioare a acelerării paralele

Atunci când programul paralel este executat pe un sistem real, apar o serie de costuri suplimentare generate de comunicaţia dintre procesoare (schimb de date, sincronizare) cât şi de încărcarea neechilibrată a procesoarelor. Din acest motiv timpul paralel de execuţie poate depăşi valoarea teoretică calculată (Fig. 2).

Eficienţa paralelă se defineşte ca fiind egală cu raportul dintre accelerarea paralelă şi numărul de procesoare utilizate: $E = \frac{S}{p}$

Datorită faptului că accelerarea are o valoare limitată la numărul de procesoare, eficienţa paralelă poate lua valori numai între 0 şi 1.

Eficienţa paralelă cuantifică măsura în care procesoarele din sistem au efectuat operaţii utile de calcul de-a lungul execuţiei programului paralel.

Costul paralel înglobează timpii consumaţi de toate procesoarele sistemului paralel pentru obţinerea soluţiei problemei: $C_p = p \cdot T_p$

În general prin cost înţelegem suma tuturor timpilor consumaţi pentru rezolvarea unei probleme. Din acest motiv costul secvenţial va fi întotdeauna egal cu timpul secvenţial de execuţie deoarece acesta reprezintă timpul total consumat pentru rularea programului: $C_s = T_s$

Eficienţa paralelă mai poate fi exprimată cu ajutorul raportului dintre costul secvenţial şi cel paralel:

$$E = \frac{S}{p} = \frac{T_s \big/ T_p}{p} = \frac{T_s}{p \cdot T_p} = \frac{C_s}{C_p}$$

Un algoritm paralel se consideră a fi optimal din punct de vedere al costului dacă costul paralel al acestuia este proporţional cu timpul de execuţie al celui mai rapid algoritm serial care rezolvă respectiva problemă: $C_p = k \cdot T_s$

Folosind relaţia anterioară, eficienţa paralelă devine:

$$E = \frac{C_s}{C_p} = \frac{T_s}{p \cdot T_p} = \frac{T_s}{k \cdot T_s} = \frac{1}{k} = O(1)$$

Rezultă că un algoritm cost-optimal are o eficienţă de ordinul *O(1)*, constantă în raport cu

dimensiunea problemei şi cu numărul de procesoare utilizate pentru rezolvarea acesteia.

Costul suplimentar de calcul paralel reprezintă suma timpilor de inactivitate şi a celor consumaţi de către procesoarele sistemului paralel pentru operaţiile de comunicaţie şi sincronizare, operaţii fără de care obţinerea rezultatelor finale ale programului nu ar fi posibilă. Acest cost nu există pentru varianta secvenţială a algoritmului deoarece în sistem există un singur procesor.

În figura 2 se prezintă modul în care se va executa un program în cadrul unui sistem paralel real format din patru procesoare, în condiţiile în care 20% din program nu se poate paraleliza, încărcarea procesoarelor nu este echilibrată iar timpii de comunicaţie nu mai sunt neglijaţi. Se observă faptul că, faţă de cazul ideal, timpul paralel de execuţie este mai mare datorită influenţei celor doi factori anterior menţionaţi.

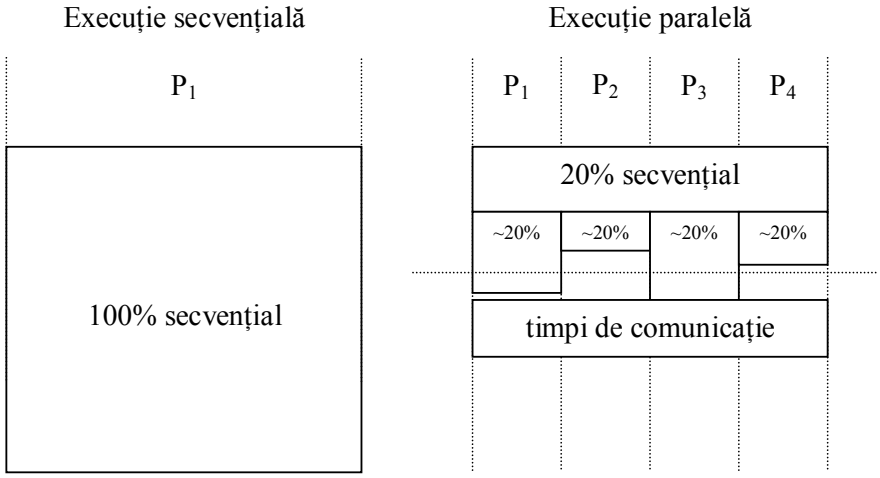

Fig. 2. Execuţia unui program în cadrul unui sistem real

Costul suplimentar de calcul paralel se poate exprima cu ajutorul următoarei formule:

$$C_{\sup l} = C_p - C_s = p \cdot T_p - T_s$$

Pentru ducerea la bun sfârşit a unei sarcini de calcul este necesar ca procesoarele sistemului să comunice între ele. Transferul de date între două procesoare ale sistemului este o operaţiune mare consumatoare de timp care poate genera inactivitate la nivelul elementelor de procesare. Din acest motiv, algoritmii paraleli ar trebui să fie proiectaţi astfel încât majoritatea acceselor la memorie să nu necesite transportul datelor solicitate prin reţeaua de interconectare.

Existenţa în cadrul unui program a unui punct de sincronizare între mai multe procesoare va genera timpi de aşteptare deoarece execuţia programului va continua numai după ce toate procesoarele au atins punctul respectiv.

Principală cauză pentru care la nivelul elementelor de procesare se înregistrează timpi de inactivitate o constituie încărcarea neechilibrată a procesoarelor care survine atunci când procesoarele din sistem au asociate sarcini de calcul de dimensiuni inegale. Într-un astfel de caz, unele procesoare îşi vor încheia execuţia mai devreme şi vor rămâne în aşteptare în timp ce altele vor fi încă în faza de execuţie.

O altă sursă care generează timpi de inactivitate şi implicit o încărcare neechilibrată a procesoarelor o reprezintă variaţia gradului de paralelism pe parcursul execuţiei programului, unde prin gradul de paralelism al unui program paralel înţelegem numărul de procesoare care sunt utilizate pentru execuţia acestuia.

Chiar şi prezenţa în cadrul unui program a unei părţi ce nu poate fi paralelizată atrage după sine o astfel de încărcare neechilibrată a procesoarelor deoarece fracţia respectivă va trebui să fie executată secvenţial, de către un singur procesor. O astfel de situaţie este exprimată de către legea lui Amdahl.

Pentru ca un program paralel să fie executat cât mai rapid este necesar să se reducă cât mai mult posibil fracţia de program ce nu se

poate paraleliza şi în plus să se asigure o distribuție cât mai uniformă a sarcinii de calcul pe elementele de procesare în condițiile în care se are în vedere minimizarea timpilor de comunicație

Pentru obținerea unor algoritmi paraleli cât mai eficienți este necesar ca aceştia să îndeplinească o serie de condiții:

- să fie proiectați astfel încât să folosească un număr de procesoare mai mic de decât dimensiunea problemei, mai ales în situațiile în care este necesar să se prelucreze un volum considerabil de date; mai mult, algoritmul nu trebuie să fie dependent de o anumită configurație a calculatorului paralel pe care va fi executat;
- să se obțină un timp paralel de execuție cât mai mic şi considerabil mai bun decât cel corespunzător celui mai rapid algoritm secvențial care rezolvă aceeaşi problemă; în general ne aşteptăm ca timpul paralel de execuție să scadă în situația în care numărul de procesoare disponibile în sistem creşte;
- costul paralel trebuie să fie optimal.

Bibliografie

[Lad04] S. Ladd, *Guide to Parallel Programming*, Springer-Verlag, 2004

[Kar03] G. Karniadakis, R. M. Kirby, *Parallel Scientific Computing in C++ and MPI : A Seamless Approach to Parallel Algorithms and their Implementation*, Cambridge University Press, 2003

[Gra03] A. Grama et al, *An Introduction to Parallel Computing: Design and Analysis of Algorithms*, Addison Wesley, 2nd edition, 2003

[Dod02] Gh. Dodescu, B. Oancea, M. Raceanu, *Procesare paralelă*, Editura Economica, Bucureşti, 2002

[Gro02] W. Gropp et al, *The Sourcebook of Parallel Computing*, Morgan Kaufmann, 2002