1 Indicatori de performantele ai calculului paralel

Datorita complexitatii calculului paralel este greu de a stabili o masura a performantelor care sa stabileasca real si absolut performantele unui sistem cu arhitectura paralela.
Totusi sunt utilizati citiva indici care masoara diferite aspecte globale.

1.1 Rata de executie -

masoara rata de producere a unor rezultate in unitatea de timp.
Uzual se foloseste :

- **MIPS** - milioane de instrucrtiuni pe secunda care masoara numarul de instructiuni pe care le executa pe secunda o unitate de prelucrare momo sau multiprocesor. Un asfesl de indice este inadecvat pentru o masina SIMD care executa aceeasi instructiune pe mai multe fluxuri de date.
- MOPS - milioane de operatii pe secunda care masoara operatiile efectuate de unitatile de prelucrare. O astfel de unitate de masura nu tine seama de lungimea cuvintului si nici de natura operatiilor.
- MFLOPS - milioane de operatii cu virgula mobila pe secunda (Giga) care masoara operatiile cu virgula mobila efectuate (Tera) de unitatile de prelucrare.O astfel de masura este adecvata numai pentru aplicatii numerice dar nu reprezinta nici un fel de masura pentru prelucrari alfanumerice sau pentru aplicatii bazate pe intaligenta artificiala.
- MLIPS – milioane de inferente logice pe secunda care masoara numarul de inferente logice realizate in aplicatii de IA

1.2 Viteza de prelucrare - V_p

$$V_p = \frac{T_1}{T_p} \quad \text{unde}$$

- T_1 - timpul necesar pentru prelucrare utilizind un singur procesor
- T_p - timpul necesar pentru prelucrare utilizind p procesoare

Cu alte cuvinte, V_p reprezinta raportul intre prelucrarea secventiala si cea paralela care scoate in evidenta cresterea in viteza datorita paralelismului.
$p*T_p >> T_1$
deoarece se consuma timp cu sincronizarea, comunicarea si "overhead" cerut de interactiunea intre procesoare.

$$1 \leq Vp < p$$

1.3 Eficienta Ep

Este definita ca raportul intre viteza de prelucrare si numarul de procesoare.

$$E_p = \frac{V_p}{p} = \frac{T_1}{p*T_p} < 1$$

Eficienta este o masura a eficientei costului.

1.4 Redundanta Rp

Este definita ca raportul intre

- numarul total al operatiilor Op necesar efectuarii calculului cu p procesoare si
- numarul de operatii O1 necesar efectuarii calculului cu un singur procesor.

$$R_p = \frac{O_p}{O_1}$$

reflecta timpul pierdut cu overhead-ul.

1.5 Utilizarea U_p

Este definita ca raportul dintre
numarul de operatii O_p necesar efectuarii calculului cu p procesoare si
numarul de operatii care ar fi putut fi efectuate cu p procesoare in timpul Tp

$$U_p = \frac{O_p}{p*T_p} \leq 1$$

2 Modele matematice ale calculului paralel

Este foarte important a stabili modele ale calculului paralel in vederea stabilirii limitei calculului paralel.
Fie T_n timpul necesar executiei a
n taskuri de k tipuri diferite.
Fiecare tip consta din n_i taskuri necesitind t_i secunde fiecare
In acest caz

$$T_n = \sum_{i=1}^{k} (n_i*t_i) \quad \text{iar} \quad n = \sum_{i=1}^{k} n_i$$

Prin definitie, rata de executie R este numarul de operatii efectuate in unitatea de timp

$$Rn = \frac{n}{T_n} = \frac{n}{\sum_{i=1}^{k} (n_i*t_i)}$$

Fie $f_i=n_i/n$

$$\sum_{i=1}^{k} f_i = 1$$

$$R_i = \frac{1}{t_i}$$

In acest caz rezulta

$$R_n = \frac{n}{\sum_{i=1}^{k} (n_i * t_i)} = \frac{1}{\sum_{i=1}^{k} (n_i/n * t_i)} = \frac{1}{\sum_{i=1}^{k} (f_i/R_i)}$$

Aceasta relatie reprezinta "bottleneck" - "gituirea", limitarea in structurile paralele.
Rata de executie a unui task poate domina performantele masinii.

3 Legea lui Amdhal

Inca din 1967 Gene Amdahl a introdus o lege care ii poarta numele privind rata de executie. El propune o formula prin care stabileste o limita a cresterii de viteza a structurilor paralele in raport cu structurile monoprocesor.
Fie:

Rh	rata de executie pe structura paralela
Rl	rata de executie pe structura monoprocesor
f	procentul secventei de instructiuni executata in paralel
1-f	procentul secventei de instructiuni executata secvential

In acest caz formula de baza este :

$$R(f) = \frac{1}{f/Rh + (1-f)/Rl}$$

Rl - poate fi stabilita de I/E sau operatii de comunicatie sau executie secvente critice care se exclud reciproc
Rh -poate fi stabilita de prelucrari concurente pe procesoare distincte

O forma redusa a acestei formule ar putea fi exprimata astfel :
Fie p procesoare
L lungimea toata a programului
S lungimea sectiunii secventiale

$$R = \frac{\text{timp calcul procesor}}{\text{timp calcul pentru p procesoare}} = \frac{L}{S + (L-S)/p}$$

pentru p >>> rezulta R = L / S

Indiferent de numarul de procesoare cresterea de viteza este limitata de caracteristicile intrinseci ale programului.

Exemplu:
Fie un sistem de calcul multiprocesor care poate executa 100 Mflops utilizand procesoarele de care dispune si poate executa 1 Mflops activitatile secventiale.
Care este performanta sistemului cand executa un program care contine 90% din cod care se poate executa in paralel si 10% din cod care se executa secvential

Rezolvare:
Utilizand legea lui Amdahl:

$$R(f) = \frac{1}{f/Rh + (1-f)/Rl}$$

$$R(f) = \frac{1}{0.9/100 + 0.1/1} = \frac{1}{0.109} = 9.17 \text{ Mflops}$$

Sa consideram ca masina executa 1Gflops utilizand procesoarele de care dispune si poate executa 1 Mflops activitatile secventiale.

$$R(f) = \frac{1}{0.9/1000 + 0.1/1} = \frac{1}{0.1009} = 9.91 \text{ Mflops}$$

Exemplu:

Un algoritm secvential prelucreaza o matrice A[n,n] in $n^3/3 - n^2/6$ unitati de timp. Fiind date p procesoare, timpul necesar este $n(n^2-1)/2p$
1 Sa se calculeze Vp
2 Care este eficienta algoritmului cand n >>>
3 Pentru ce valoare a lui n algoritmul se apropie cu 5% de eficienta maxima

$$Vp = \frac{n^3/3 - n^2/6}{n(n^2-1)/2p} = \frac{n\,p\,(2n-1)}{3\,(n^2-1)}$$

$$Ep = \frac{Vp}{P} = \frac{n\,(2n-1)}{3\,(n^2-1)}$$

Cand n >>> rezulta Ep = 2/3

$$Ep = \frac{Vp}{P} = \frac{n(2n-1)}{3(n^2-1)} = \frac{95}{100}$$

$200n^2-100n=285\ n^2 - 285$
$85\ n^2+100n -285=0$

n= 2.670067 deci n=2

4 Limite ale calculului paralel

In 1986 Jack Worlton a studiat limitele calculului paralel utilizind un model care aproximeaza operarea unui multiprocesor.
El presupune ca un program paralel consta din :

- sectiuni de prelucrare distribuite pe diverse procesoare;
- sectiuni de sincronizare;
- sectiuni care se executa secvential si constituie un "overhead".

Sa definim :

ts	timpul de sincronizare;
to	sectiune de overhead
t	media timpului de executie a unui task
N	numarul de task-uri (intre puncte de sincronizare)
P	numarul de procesoare

Timpul de executie secvential al celor N task-uri este :

$$T1 = N*t$$

Timpul de executie a celor N task-uri executate pe p procesoare este :

$$Tn,p = ts + (\lceil n / p \rceil)*(t+to)$$

Viteza de prelucrare

$$Vn,p = T1 / Tn,p = \frac{N*t}{t_s + (\lceil N / p \rceil)*(t+t_0)} =$$

$$= \frac{1}{t_s / (N*t) + (1/N)*(\lceil N / p \rceil)*(1+(t_0/t))}$$

Observatii:
Pentru a creste viteza de prelucrare trebuie sa avem in vedere reducerea efectului sincronizarii, overhead-ului si a numarul de pasi $\lceil N / p \rceil$

- ***efectul sincronizarii*** cauzat de ts/(N*t) poate fi redus fie prin micsorarea timpului de sincronizare ts sau prin marirea intervalului intre sincronizari N*t
- ***efectul overhead-ului*** cauzat de to/t poate fi redus prin reducerea timpului de overhead to sau prin cresterea granularitatii task-urilor.

Cresterea granularitatii task-urilor ajuta la reducerea atit a efectului sincronizarii cit si a overhead-ului.

- ***numarul pasilor de calcul*** $\lceil N / p \rceil$ poate fi redus prin cresterea numarului de procesoare si avind un numar de task-uri multiplu de procesoare.

Eficienta

$$E_{n,p} = \frac{V_{n,p}}{p}$$

Presupunind ca avem un numar mare de task-uri si ca overhead-ul este neglijabil relativ la granularitate avem :

p >>>> N fix

$$V_{n,p} = \frac{N}{t_s/t} \quad \text{si} \quad E_{n,p} = 0$$

N >>>> p fix

$$V_{n,p} = \frac{P}{1 + t_0/t} \quad \text{si} \quad E_{n,p} = \frac{1}{1 + t_0/t}$$

5 Nivelurile de paralelism

Paralelismul poate fi examinat la multe niveluri in functie de complexitatea dorita. Frecvent gasim descrierea paralelismului la nivelurile:

- job; $J = \{ JOB_1, \ldots\ldots, JOB_m \}$
- task; $JOB_i = \{ Ti_1, \ldots\ldots..., Ti_n \}$
- proces; $Tik = \{ Pik1, Pik2, ..., Pikp \}$
- variabila;
- bit.

5.1 Nivel de job

Se executa doua sau mai multe programe independente pe resurse de procesare distincte.

5.2 Nivel de task-uri

JOB={T1, ..., Tn}.

Fiecare Ti se executa pe cite un procesor distinct insa exista o relatie intre Ti si Tj in ceea ce priveste

transferul de date sau completarea functiilor de prelucrare realizate in cadrul job-ului.

Exemplu:
Consideram un robot care are mai multe grade de libertate. Pentru fiecare grad de libertate exista un procesor. Programul de conducere a robotului este partitionat in task-uri care se ocupa de cite un grad de libertate al robotului.
Taskurile se executa in paralel, dar interactioneaza pentru miscarea robotului.

5.3 Nivel de proces

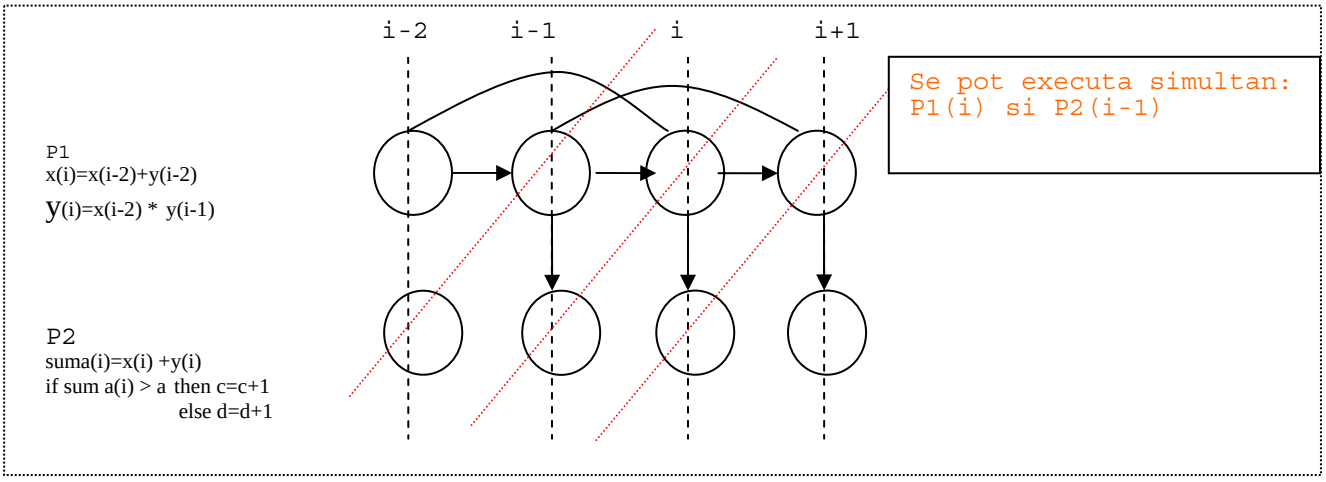
Ti={Pi1, Pi2, ..., Pip}
Task-urile sunt alcatuite din mai multe procese care in general sunt identificate de utilizator sau de catre compilator daca acesta este destinat pentru structuri multiprocesor.
Sa consideram task-ul

```
For i=1 to n
        x(i)=x(i-2)+y(i-2)
        y(i)=x(i-2) * y(i-1)
        suma(i)=x(i) +y(i)
        if s(i) > a then c=c+1
                else d=d+1
```

In cadrul acestui task identificam doua procese:

```
P1(i)
        x(i)=x(i-2)+y(i-2)
        y(i)=x(i-2) * y(i-1)
P2(i)
        suma(i)=x(i) +y(i)
        if s(i) > a then c=c+1
                else d=d+1
```

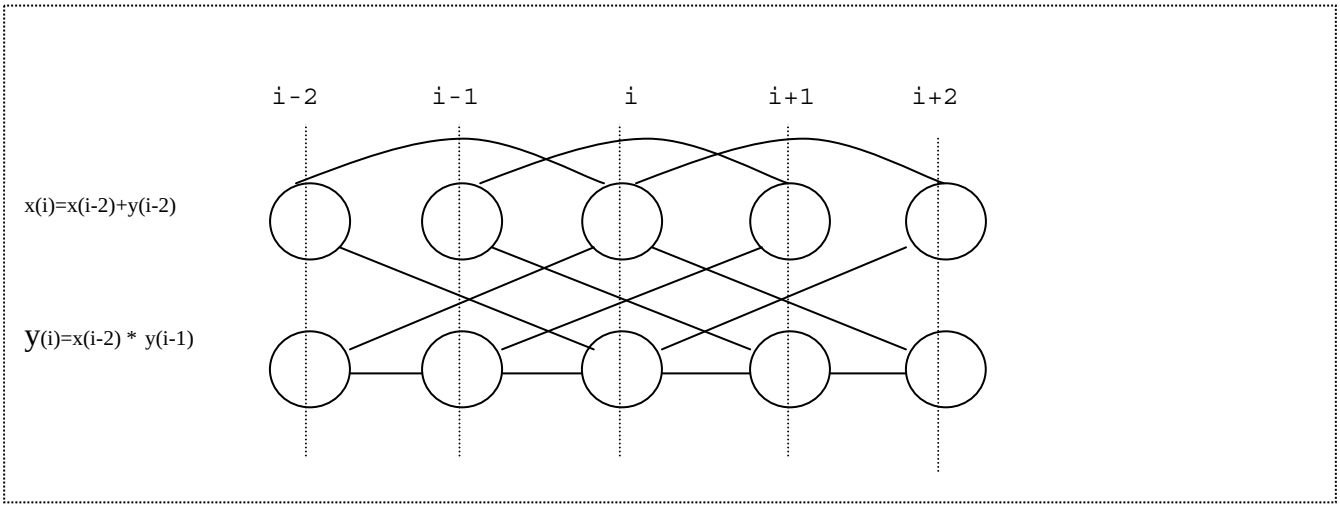
Intre P1(i) si P2(i) exista o dependenta de date, deci nu pot fi efectuate in paralel
Insa P1(i) si P2(i-1) pot fi efectuate in paralel
Interdependenta intre aceste procese este urmatoarea:



5.4 Nivel de variabila

Pi= { Ii1, ….Iik}
Fiecare proces este format dintr-uo multime de instructiuni care pot calcula variabilele de iesire in functie de variabilelel de intrare.
Paralelismul in cadrul unui proces se poate realiza prin calculul simultan a mai multe variabile de iesire.

In exemplul anterior putem considera ca in cadrul procesului P1 variabilele
x(i) si y(i) se pot calcula in paralel
sau
x(i) si y(i+1) se pot calcula in paralel



5.5 Nivel de bit

Toate calculatoarele, cu foarte putine exceptii, utilizeaza unitati aritmetice paralele cu sau fara anticiparea transportului si cu structura pipe line.

6 Relatia intre algoritmi paraleli si arhitecturi paralele

Prelucrarea paralela include atit algoritmi paraleli cit si arhitecturi paralele.
Un algoritm paralel poate fi considerat ca o colectie de procese independente care se executa simultan, procesele comunicind in timpul executiei.
Astfel un algoritm se executa pe unitati functionale hardware care in general constau din

- elemente de procesare;
- module de transfer date.

Ceea ce intereseaza este cum se transpun procesele pe unitatile functionale hardware.
H.T.Kung a fost unul din primii care au studiat relatia intre algoritmi si arhitectura. A stabilit citeva caracteristici si a prezentat corelarea intre ele:

Algoritmi paraleli	Arhitecturi paralele
granularitate modul	complexitate procesor
control concurent	mod de operare
mecanismul datelor	structura memoriei
geometria comunicatiei	retele de comutare
.complexitate algoritm	numar de procesoare; dimensiune memorie

Sa specificam foarte sumar caracteristicile elementelor prezentate

- Granularitate modul - se refera la complexitatea modulului care poate fi job, task, proces sau instructiune.De obicei exista posibilitati de paralelism la o granularitae mare dar care nu este exploatat deoarece implica o comunicatie intensa si o crestere a complexitatii software-ului.La acest nivel se face o analiza intre granularitate si comunicatie pentru a stabili solutia cea mai buna.
- Control concurent se refera la schema de selectie a modulelor pentru executie. Aceasta trebuie sa satisfaca dependenta de date si dependenta de control (asigurarea excluderii mutuale la o aceeasi resura).

  Citeva scheme de control sunt bazate pe:
  - disponibilitatea datelor (data flow)
  - control centralizat (sinchronized)
  - cereri (demand-driven).

De exemplu algoritmii cu matrice se potrivesc foarte bine pe procesoarele sistolice sau pe masive de procesoare iar algoritmii care au transferuri conditionate si alte iregularitati se potrivesc foarte bine pe arhitecturi asincrone cum ar fi multiprocesoare si data-flow.

- Mecanismul datelor se refera la faptul cum sunt utilizati operanzii. Datele furnizate de instructiuni pot fi utilizate ca "date pure" in structurile data-flow sau pot fi depuse in locatii adresabile in masinile Von Neumann.
- Geometria comunicatiei - se refera la sablonul de interactiune intre module. Poate fi regulata sau neregulata.
- Dimensiune algoritm se refera la numarul de operatii necesare pentru implementarea algoritmului. Are influenta asupra numarului de procesoare si dimensiunii memoriei.