Laborator 5

**Cerința:**

**Laborator 5**

Aceeasi problema ca si la Laborator 4 dar cu urmatoarele modificari:

* Dintre cele p threaduri p1 sunt cititoare si p2 sunt de tip ‘worker’ (preiau din coada si adauga in lista rezultat) – p=p1+p2
* Coada in care se adauga monoamele citite are o capacitate maxima = MAX
* Implementarea pentru producator consumator (writer/reader) trebuie sa foloseasca mecanisme de tip wait notify (nu busy waiting)
* **Sincronizarea operatiile cu lista inlantuita se fac la nivel de nod nu pentru intreaga lista!**
  + Sincronizarile se vor face folosind variabile de tip Lock asociate cu fiecare nod.
  + Daca rezulta un nod cu coeficient nul acest nod se va sterge din lista!

Observatie: Este obligatoriu sa se foloseasca sablonul ‘producator-consumator’ pentru legatura dintre threadurile ‘reader’ si cele de tip ‘worker’ --- wait|notify.

Limbaj: la alegere intre Java si C++

Analiza timpului de executie pentru urmatoarele cazuri:

1. Rezolvare secventiala
2. 10 polinoame fiecare cu gradul maxim 1000 si cu maxim 50 monoame si MAX=20
   1. p = 4, 6, 8 ; si p1=2 , p1=3
   2. secvential ; si p1=2 , p1=3
3. 5 polinoame fiecare cu gradul maxim 10000 si cu maxim 100 monoame si MAX=30
   1. p = 4, 6, 8 ; si p1=2 , p1=3
   2. secvential ; si p1=2 , p1=3

Analiza comparativa a performantei –Lab 4) versus Lab 5)

**Proiectare:**

Pentru ambele variante (secvențială și paralelă) am implementat clasele **Node** și **MyList**. **Node** reprezintă un monom, și este tipul de date al elementelor din lista de tip **MyList**. Conține un coeficient, un exponent, și următorul nod. **MyList** conține **head**, ce reprezintă primul element al listei, și **length**, lungimea sa.

**MyList** are următoarele metode:

* insertNode(node: Node) – inserează un nod în listă
* writeToFile – scrie lista în fișierul rezultat

În plus pentru varianta paralelă, am implementat structura pentru coada de monoame, **MyQueue**. Aceasta conține un **ArrayList** de tip **Node**, ce conține monoamele ce vor fi procesate de consumeri.

**MyQueue** are următoarele metode:

* firstElement() – returnează primul nod din coadă
* enqueue(node: Node) – inserează nodul dat în coadă
* dequeue() – scoate din coadă primul nod
* isEmpty() – verifică dacă coada este goală
* setReadingDoneFlagTrue() – seteaza flag-ul de terminare a citirii ca fiind true si apeleaza **notifyAll()**
* incrementThreadNumber() – incrementeaza numarul de produceri care au terminat

Pentru varianta paralelă, **p1** thread-uri sunt produceri, si citeste fiecare, intre un **start** si un **end**, ca si la laboratoarele trecute. Dupa ce termina de citit bucata lui, fiecare thread incrementeaza variabila **finishedProducers**, printr-o metoda **synchronized**. Daca **finishedProducers** este egal cu **p1**, atunci se notifica faptul ca citirea e gata. **P2** thread-uri sunt consumeri, si extrag noduri din coada, si le adauga polinomului de final.

Pentru sincronizarea thread-urilor, am folosit un **ReentrantLock** la nivel de nod, folosind metodele **lock()** si **unlock()**.

**Timpi de execuție:**

Tip Polinom,Nr Consumer Threads,Nr Producer Threads,Queue Size,Timp Executie,Raport TSecvential/Tparalel   
P1(10 polinoame, grad maxim 1000, maxim 50 monoame),3,2,20,51.76271,0.8651  
,2,2,20,43.4318,1.0310  
,4,2,20,51.53173,0.8690  
,4,4,20,51.79119,0.8646  
,6,2,20,49.20985,0.9101  
,7,1,20,48.86381,0.9164  
P2(5 polinoame, grad maxim 10000, maxim 100 monoame),3,2,30,57.91046,0.7316  
,2,2,30,46.01527,0.9208  
,4,2,30,54.79386,0.7733  
,4,4,30,55.44012,0.7642  
,6,2,30,52.08246,0.8135  
,7,1,30,50.895,0.8324  
P3(10 polinoame, grad maxim 1000, maxim 100 monoame),3,2,30,95.49099,0.6128  
,2,2,30,78.4995,0.7328  
,4,2,30,97.17339,0.6022  
,4,4,30,99.58046,0.5876  
,6,2,30,100.09814,0.5846  
,7,1,30,98.26127,0.5955

**Analiză timp:**

Rapoartele de timp Tsecvential/Tparalel se pot observa în tabelul de timpi de mai sus, pe penultima coloană.

Comparativ cu laboratorul trecut, timpii par a fi mai slabi, cel mai probabil din cauza limitei impuse asupra cozii de noduri.

**Diagramele claselor:**

