Examen APD CB Main

GG EZ

Grile

Algoritmi paraleli

------------------------------------------------------------------------------------

Ce face functia pthread\_join()?

a.

**Blocheaza thread-ul curent in asteprarea thread-ului dat ca argument**

b.

Opreste thread-ul dat ca argument

c.

Opreste toate thread-urile

d.

Intrerupe thread-ul dat ca argument

------------------------------------------------------------------------------------

Cine controleaza numarul de thread-uri pornite intr-un program Pthread?

a.

Sistemul de operare

b.

Linia de comanda

c.

**Programatorul**

d.

Procesorul

-\_-

------------------------------------------------------------------------------------

Cum se poate masura timpul de executie din linia de comanda?

a.

Pornind aplicatia xclock

b.

Folosind apel la functia localtime

**c.**

**Folosind programul time**

d.

Folosind programul date

[Șterge alegerea mea](https://curs.upb.ro/mod/quiz/attempt.php?attempt=434284&cmid=189836#)

Este C aici cred+ 4

------------------------------------------------------------------------------------

Intr-un program Pthread, o variabila declarata global poate fi vazuta de toate thread-urile / Intr-un program Pthread, o variabila declarata global poate fi modificata de toate thread-urile

a.

adevarat / fals

b.

**adevarat / adevarat**

c.

fals / fals

d.

fals / adevarat

eu zic ca b +2

Intr-un program care foloseste pthreads, daca un thread face lock de 2 ori consecutiv pe acelasi mutex, se produce un deadlock / Intr-un program multithreaded in Java, daca un thread face lock de 2 ori consecutiv pe acelasi obiect, se produce un deadlock

a.

fals / adevarat

b.

**adevarat / fals**

c.

adevarat / adevarat

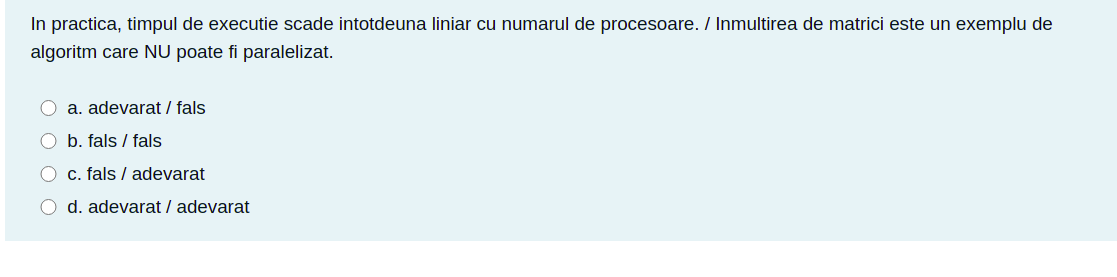
d.

fals / fals

e a? eu zic ca e b

E b sigur

------------------------------------------------------------------------------------



b? nu e a? b, este **b clar**

**Înmulțirea de matrici stiu ca e paralelizabila, ca paralelizezi un for**

------------------------------------------------------------------------------------

Ce se va intampla daca porniti un program Pthread cu mai multe thread-uri decat core-uri?

a.

Programul porneste si creste automat numarul de core-uri

b.

Programul nu va putea porni

c.

Programul nu compileaza

d.

**Programul va functiona normal**

Dd++++

------------------------------------------------------------------------------------

In cadrul modelului genetic Master-Slave,

a.

functia de fitness se evalueaza in paralel si operatorii de selectie si evolutie se aplica pe sub-populatii

b.

functia de fitness se evalueaza la nivelul masterului si operatorii de selectie si evolutie se aplica pe toata populatia

c.

functia de fitness se evalueaza la nivelul masterului si operatorii de selectie si evolutie se aplica pe sub-populatii

d.

**functia de fitness se evalueaza in paralel si operatorii de selectie si evolutie se aplica pe toata populatia**

sigur d?da

-----------------------------------------------------------------------------------

In cadrul Selecției de tip turneu:

a.

Se aleg soluțiile (cromozomii) cele mai bune pentru etapa următoare

b.

**Se aleg soluții (cromozomi) doua cate doua, soluția mai buna este aleasa**

c.

Soluțiile (cromozomii) se aleg in funcție de rank-ul lor

d.

Fiecare soluție (cromozom) are o probabilitate de a fi aleasa, proporțională cu valoarea sa de fitness

la asta cred ca este b nu????????

**R a (asa am gasit in curs)**

▪ Soluția mai bună (din fiecare pereche) este aleasă ‒ adăugată la mating pool pentru etapa de Crossover

-\_-

------------------------------------------------------------------------------------

In cazul algoritmului arbore de alegere a liderului, numărul de mesaje trimise in sistem este:

a.

2N

b.

2N-2

c.

4N

d.

**4N-4**

-\_-

nu e d? Ba da (slide 736) d++

------------------------------------------------------------------------------------

Ce poate cauza un comportament nedeterminist al unui program paralel?

a.

Folosirea unei regiuni critice

**b.**

**Scrierea simultana a doua thread-uri in aceeasi zona de memorie**

c.

Citirea simultana de catre doua thread-uri a aceleiasi zone de memorie

d.

Un apel al functiei sleep

-\_-

------------------------------------------------------------------------------------

La initierea unei bariere in pthread este obligatoriu sa se specifice:

a.

Cate procesoare exista pe masina locala

b.

ID-ul unui thread master ce va fi primul deblocat din apelul pthread\_barrier\_wait()

c.

**Numarul de thread-uri ce vor astepta la bariera yas asta e**

d.

Timpul minim pentru care orice thread va fi blocat la apelul pthread\_barrier\_wait()

c+

-----------------------------------------------------------------------------------

La initierea unei bariere in pthread este obligatoriu sa se specifice:

a.

ID-ul unui thread master ce va fi primul deblocat din apelul pthread\_barrier\_wait()

b.

Cate procesoare exista pe masina locala

c.

**Numarul de thread-uri ce vor astepta la bariera**

d.

Timpul minim pentru care orice thread va fi blocat la apelul pthread\_barrier\_wait()

------------------------------------------------------------------------------------

Cum se porneste un thread in Pthread?

a.

Apel pthread\_start()

b.

#pragma omp parallel

c.

**Apel pthread\_create()**

d.

Apel pthread()

c e sigur

c crea za, ea intreba cum se porneste, create il porneste :))

c

Firul de execuție se **pornește** prin apelul metodei start() a acestuia. de pe net

“The function to start the thread is pthread\_create”

oki

------------------------------------------------------------------------------------

Un program ce foloseste MPI se compileaza folosind mpicc / Un program ce foloseste Pthread se compileaza folosind gcc

a.

fals / adevarat

b.

**adevarat / adevarat**

c.

fals / fals

d.

adevarat / fals

------------------------------------------------------------------------------------

Ce se va intampla daca porniti un program Pthread cu mai multe thread-uri decat core-uri?

a.

Programul nu compileaza

b.

Programul porneste si creste automat numarul de core-uri

c.

**Programul va functiona normal**

d.

Programul nu va putea porni

a sau c? stiu ca era cv cu oversubscribe; daca folosesti oversubscribe ruleaza cu cate coreuri ii dai, vezi ca asta e la mpi

c

------------------------------------------------------------------------------------

Fie o variabila globala a initializata cu 0 si doua thread-uri care executa (fara sincronizare) operatia a = a + 2. Considerand ca operatia de scriere in memorie este atomica ca, multimea completa de valori pe care le poate avea variabila in urma executiei ambelor thread-uri este:

a.

**{2, 4} gresit fmm**

b.

{0, 4}

c.

{4}

d.

{0, 2, 4}

------------------------------------------------------------------------------------

Ce face aplicatia mpirun?

a.

Porneste cate o aplicatie MPI pe fiecare sistem disponibil

b.

Alearga MPI-ul (run MPI, run)

c.

Porneste un program MPI multi-thread

d.

**Porneste un program distribuit MPI cred ca asta, imo**

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

In cadrul modelului genetic Master-Slave,

a.

functia de fitness se evalueaza la nivelul masterului si operatorii de selectie si evolutie se aplica pe toata populatia

b.

functia de fitness se evalueaza in paralel si operatorii de selectie si evolutie se aplica pe sub-populatii

c.

functia de fitness se evalueaza la nivelul masterului si operatorii de selectie si evolutie se aplica pe sub-populatii

d.

functia de fitness se evalueaza in paralel si operatorii de selectie si evolutie se aplica pe toata populatia

------------------------------------------------------------------------------------

Ce face aplicatia mpirun?

**a.**

**Porneste un program distribuit MPI**

b.

Alearga MPI-ul

c.

Porneste un program MPI multi-thread

d.

Porneste cate o aplicatie MPI pe fiecare sistem disponibil

-\_-

alearga MPI ul =))))) a/d

------------------------------------------------------------------------------------

La initierea unei bariere in pthread este obligatoriu sa se specifice:

a.

Cate procesoare exista pe masina locala

**b.**

**Numarul de thread-uri ce vor astepta la bariera**

c.

ID-ul unui thread master ce va fi primul deblocat din apelul pthread\_barrier\_wait()

d.

Timpul minim pentru care orice thread va fi blocat la apelul pthread\_barrier\_wait()

------------------------------------------------------------------------------------

Care este rezultatul operației de Crossover in doua puncte, după poziția a4a si a9a pentru următorii 2 cromozomi:

Cromozom1: 1101 | 00100 | 110100

Cromozom2: 1010 | 11000 | 011110

a.

Descendent1: 1010 00100 011110

Descendent2: 1101 11000 110100

b.

Descendent1: 1000 00100 011110

Descendent2: 1111 11000 110100

c.

Descendent1: 1011 00100 011110

Descendent2: 1100 11000 110100

d.

Descendent1: 1010 11000 110100

Descendent2: 1101 00100 011110

a

------------------------------------------------------------------------------------

In cadrul modelului genetic Master-Slave,

**a.**

**functia de fitness se evalueaza in paralel si operatorii de selectie si evolutie se aplica pe toata populatia**

b.

functia de fitness se evalueaza la nivelul masterului si operatorii de selectie si evolutie se aplica pe toata populatia

c.

functia de fitness se evalueaza la nivelul masterului si operatorii de selectie si evolutie se aplica pe sub-populatii

d.

functia de fitness se evalueaza in paralel si operatorii de selectie si evolutie se aplica pe sub-populatii

-------------- ----------------------------------------------------------------------

Numarul de mesaje pentru algoritmul LeLann este:

a.

O(N^3)

b.

O(N^4)

**c.**

**O(N^2) asta 100% , am cautat in curs**

d.

O(N)

------------------------------------------------------------------------------------

In cazul algoritmului Bully, un proces devine lider daca:

primeste OK de la majoritatea proceselor

**b.**

**trimite mesaje de Alegere si nu ii se raspunde**

c.

descopera primul ca actualul lider este picat

d.

primeste OK de la toate proceselor

------------------------------------------------------------------------------------

Care dintre urmatoarele operatii efectuate asupra unei variabile globale NU genereaza o conditie de cursa, daca tid este ID-ul thread-ului curent?

a.

a += 2

b.

a = tid + 1

c.

a = tid ? a + 1 : a

d.

**a = 3**

d + 3

------------------------------------------------------------------------------------

Alegerea liderului cu algoritmul arbore:

a.

radacina arborelui initiaza algortimul de alegere

b.

doar frunzele pot initia faza de wake-up

c.

fiecare nod isi seteaza mai intai un parinte

d.

**un proces incepe algoritmul de alegere dupa ce primete wake-up pe fiecare canal**

-------------------------------------------------------------------------------

Care din urmatoarele este un efect al unei bariere?

a.

Un thread care ajunge la o bariera trece intotdeauna imediat mai departe

b.

Mai multe thread-uri nu pot rula simultan o bucata de cod delimitata de doua bariere

c.

Operatiile de dupa bariera devin atomice

**d.**

**Tot codul de dinainte de bariera se executa de catre toate thread-urile inainte de tot codul de dupa bariera**

------------------------------------------------------------------------------------

In pthread, alocarea de cuante de timp pe CPU pentru thread-uri se face de catre:

a.

Compilator

**b.**

**Sistemul de operare +3**

c.

Programator

d.

MMU

-\_-

------------------------------------------------------------------------------------

Avem un program cu un deadlock.

a.

Acesta se va bloca intotdeauna

**b.**

**Uneori acesta se va bloca la nesfarsit**

c.

Acesta nu va compila

d.

Programul va termina mereu executarea afisand rezultatul corect

[Șterge alegerea mea](https://curs.upb.ro/mod/quiz/attempt.php?attempt=434303&cmid=189836&page=3#)

------------------------------------------------------------------------------------

Ce poate cauza un comportament nedeterminist al unui program paralel?

a.

Un apel al functiei sleep

**b.**

**Scrierea simultana a doua thread-uri in aceeasi zona de memorie**

c.

Citirea simultana de catre doua thread-uri a aceleiasi zone de memorie

d.

Folosirea unei regiuni critice

b

------------------------------------------------------------------------------------

Atunci cand se intoarce, apelul MPI\_Send indica faptul ca:

a.

Receptorul a apelat MPI\_Recv si a inceput sa receptioneze datele in cadrul programului MPI

b.

Receptorul a apelat MPI\_Recv si a receptionat integral datele in programul MPI

c.

Informatia a ajuns in bufferele receptorului

d.

**Informatia a fost copiata in bufferele de trimitere** //asta

d cred (sigur e asta)

eu am pus a

------------------------------------------------------------------------------------

Succesiunea a doua operatii atomice este atomica / Succesiunea a trei operatii atomice este atomica

a.

adevarat / adevarat

b.

fals / adevarat

**c.**

**fals / fals**

d.

adevarat / fals

------------------------------------------------------------------------------------

Care dintre urmatoarele operatii creeaza o conditie de cursa atunci cand este apelata simultan de mai multe thread-uri?

**a.Inmultirea cu 3 a unei variabile** // asta

b.Citirea unei variabile

c.Scrierea aceleiasi constante intr-o variabila

d.Efectuarea operatiei a = 0

?? a a cred eu, a 100%

------------------------------------------------------------------------------------

Care este eficienta pentru sortarea Parallel Rank Sort in cazul rularii pe P procesoare? Se considera N dimensiunea vectorului de sortat

a.

logN/P

b.

logP

c.

**logN/N + 4**

d.

logN

e c sigur

pentru P procesoare?? cred ca e A +1

------------------------------------------------------------------------------------

Care este rezultatul operației de Crossover intr-un punct, după poziția a8a pentru următorii 2 cromozomi:

Cromozom1: 10 01 10 01 | 01 11 01

Cromozom2: 11 11 11 10 | 00 01 11

a.

Descendent1: 11 01 10 01 00 01 11

Descendent2: 11 11 11 10 01 11 01

b.

Descendent1: 01 01 10 01 00 01 11

Descendent2: 11 01 11 10 01 11 01

c.

**Descendent1: 10 01 10 00 00 01 11**

**Descendent2: 11 11 11 11 01 11 01**

d.

Descendent1: 11 11 10 01 00 11 11

Descendent2: 11 11 11 10 01 11 01

nu e a? cred ca a. pare gresit al doilea bit de la desc 1. pun a si dau flag

------------------------------------------------------------------------------------

Numarul de mesaje pentru algoritmul LeLann-Chang-Robert, cazul cel mai favorabil, este:

a.

2N-2

b.

N(N+1)/2

c.

N(N-1)/2

d.

**2N-1**

d?

+1 pentru d

------------------------------------------------------------------------------------

Ce complexitate are Odd Even Transposition Sort pentru n=p (unde n este numarul de elemente de sortat si p este numarul de thread-uri)?

a.

O( n \* log n )

b.

O( n ^ 2 )

c.

**O( n )**

d.

O( log n )

------------------------------------------------------------------------------------

Numarul de mesaje pentru algoritmul Hirschberg-Sinclair, cazul cel mai defavorabil, este:

**a.**

**O(NlogN)**

b.

O(logN)

c.

O(N^2)

d.

O(N)

------------------------------------------------------------------------------------

In practica, timpul de executie scade intotdeuna liniar cu numarul de procesoare. / Inmultirea de matrici este un exemplu de algoritm care NU poate fi paralelizat.

a.

adevarat / fals

b.

fals / adevarat

c.

**fals / fals**

d.

adevarat / adevarat

------------------------------------------------------------------------------------

Ce face MPI\_Comm\_size?

a.

Face broadcast in toata reteaua de procese

b.

**Returneaza numarul de procese asociate cu un comunicator**

c.

Returneaza rangul procesului curent

d.

Trimite un mesaj de la un proces x pana la un proces y

------------------------------------------------------------------------------------

Ce face MPI\_Comm\_rank?

a.

Returneaza numarul de procese active

b.

Returneaza numarul total de procese

c.

**Returneaza rangul procesului curent**

d.

Trimite un mesaj de la un proces x pana la un proces y

------------------------------------------------------------------------------------

Ce inseamna scalabilitate buna in programarea paralela?

a.

Un program poate fi rulat pe mai multe thread-uri

b.

**Pe masura ce numarul de thread-uri pe care se executa un program creste, durata de rulare scade**

c.

Un program nu poate fi rulat pe mai multe thread-uri

d.

Pe masura ce numarul de thread-uri pe care se executa un program creste, durata de rulare creste

b cred

---------------------------------

Execuția in paralel a operației logice a|b (unde a si b sunt doua numere întregi) este un exemplu de paralelism la nivel de:

a.

**bit**

b.

task

c.

CPU

d.

instrucțiune

a cred +

------------------------------------------------------------------------------------

In pthread, alocarea de cuante de timp pe CPU pentru thread-uri se face de catre:

a.

MMU

b.

Compilator

c.

Programator

**d.**

**Sistemul de operare // cred+1**

------------------------------------------------------------------------------------

Care din urmatoarele solutii de sincronizare pe baza de ceas este descentralizata?

**a.**

**nici o varianta listata aici**

b.

Algoritmul lui Cristian

c.

Algoritmul Berkeley

d.

Network Time Protocol

hese clocks are mostly adjusted by centralized systems such as Network Time Protocol (NTP) because these adjustments are indispensable to reduce the effects of random drifts on clocks

-\_-

cred ca e d, thoNetwork Time Protocol (algoritm descentralizat)

------------------------------------------------------------------------------------

Ce se poate intampla daca facem foarte multe send-uri dintr-un proces MPI, dar nu facem recv in procesul destinatie?

a.

**Primele mesaje trimise se vor pierde +3**

b.

Nimic notabil// cred asa zic si eu dar nu sunt sigur

c.

Operatia de send poate deveni blocanta la umplerea bufferului de sistem

d.

Vom primi un buffer overflow

Mai degraba C

------------------------------------------------------------------------------------

Problema filosofilor este un exemplu clasic de problema pentru:

a.

Alegerea liderului

b.

**Deadlock 100%**

c.

Stabilirea topologiei

d.

Nimic

b cred

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

Cum este definiti timpul total de executie al unul algoritm distribuit?

a.

**Timpul scurs de la inceperea executiei primului proces pana la terminarea executiei ultimului proces**

b.

Suma timpilor de executie a tuturor proceselor

c.

Media dintre timpii de executie a tuturor proceselor

d.

Timpul scurs de la terminarea executiei primului proces pana la inceperea executiei ultimului proces

a cred, da

------------------------------------------------------------------------------------

In Java notify() pune un thread din starea waiting in running / notifyAll() trezeste toate threadurile din waiting, punandu-le in running

**a.**

**adevarat / adevarat**

b.

fals / fals

c.

adevarat / fals // eu cred ca asta +1

d.

fals / adevarat

a??

nu e c ? notifyAll le trezeste pe toate, dar numai unul intra in running, eu asa stiu

notifyAll (): toate thread urile sunt scoase din wait set ul

lui m dar numai unul va putea ob ț ine z ă vorul

------------------------------------------------------------------------------------

Minim de cate semafoare este nevoie pentru a rezolva problema bărbierului?

*(Se considera cele 3 constrangeri ale problemei discutate la curs: barbierul fara clienti, venirea unui client, respectiv frizerie plina)*

a.

2

b.

4

**c.**

**3**

d.

1

-\_-

e 3, nvm

------------------------------------------------------------------------------------

Care este complexitatea Parallel Merge Sort?

a.

O( log n^2 )

b.

O( n ^ 2 )

c.

**O( n )**

d.

O( n \* log n )

------------------------------------------------------------------------------------

Ce face MPI\_Barrier?

a.

Sincronizeaza threadurile asociate unui proces

**b.**

**Sincronizeaza procesele dintr-un comunicator**

c.

Este echivalentul unui mutex in MPI

d.

Sincronizeaza threadurile dintr-un comunicator

e b?

**MPI\_Barrier**. Blocks until all processes in the communicator have reached this routine

------------------------------------------------------------------------------------

Considerand relatia de petrecut înainte (->) si doua evenimente a şi b:

a.

**dacă a -> b şi b -> c, atunci a -> c**

b.

dacă a şi b nu sunt din acelasi proces, nu se poate stabili o relatia intre ele

c.

dacă a reprezintă transmiterea unui mesaj iar b recepţia aceluiaşi mesaj la destinatie, atunci b -> a

d.

dacă a şi b sunt pe acelasi proces, nu se poate aplica aceasta relatie

Sigur a?? da

------------------------------------------------------------------------------------

Alegeți afirmația incorecta referitor la tehnica Split binary semaphore:

a.

Instrucțiunile între operațiile P (luare resursa) și V (eliberare resursa) se execută în excludere mutuală

b.

**Fiecare cale de execuție începe cu o operație P (luare resursa) pe unul dintre semafoare din set și se termină cu o operație V (eliberare resursa) pe respectivul semafor**

c.

Întotdeauna, cel mult unul dintre semafoarele din set este 1

d.

Inițial unul dintre semafoare din set este 1

HELP????????? b cred +1 +1

Algoritmul Pulsatiilor: Care este complexitatea in timp?

a.

O(N)

b.

O(D\*logN)

c.

**O(D)**

d.

O(D\*N)

cred ca e c, in curs apare O(2D) +1

PUNEM FLAG? SAU O(D)

Eu zic ca o(D) ca reduci asimptomatic de la O(2d) la o(d)

------------------------------------------------------------------------------------

La algoritmii cu mesaje de sondaj cu ecou, in cazul topologiilor de tip arbore (initiatorul fiind radacina arborelui):

a.

Mesajele de sondaj se propaga de la radacina la frunze / Mesajele de ecou se propaga de la initiator la frunze

b.

Mesajele de sondaj se propaga de la frunze la initiator / Mesajele de ecou se propaga de la frunze la initiator

c.

Mesajele de sondaj se propaga de la frunze la initiator / Mesajele de ecou se propaga de la initiator la frunze

d.

**Mesajele de sondaj se propaga de la radacina la frunze / Mesajele de ecou se propaga de la frunze la initiator**

------------------------------------------------------------------------------------

Alegeți varianta ce nu reprezintă un pas in Sortarea (crescătoare) folosind Pipeline. Primul proces (Rank 0) va primi rând pe rând cate un element. Fiecare proces:

a.

**transmite valoarea mai mica vecinului din dreapta**

b.

compară valoarea cu cea memorata local

c.

acceptă o valoare de la vecinul din stânga

d.

va transmite spre ieșire (Rank 0) o valoare primita de la vecinul din dreapta

------------------------------------------------------------------------------------

La algoritmii cu mesaje de sondaj cu ecou, in cazul topologiilor de tip arbore (initiatorul fiind radacina arborelui):

a.

Mesajele de sondaj se propaga de la radacina la frunze / Mesajele de ecou se propaga de la initiator la frunze

**b.**

**Mesajele de sondaj se propaga de la radacina la frunze / Mesajele de ecou se propaga de la frunze la initiator**

c.

Mesajele de sondaj se propaga de la frunze la initiator / Mesajele de ecou se propaga de la frunze la initiator

d.

Mesajele de sondaj se propaga de la frunze la initiator / Mesajele de ecou se propaga de la initiator la frunze

-\_-

------------------------------------------------------------------------------------

*Detecția terminării folosind marcaje:* Un proces porneste algoritmul de terminare:

a.

dupa ce primeste marcaje de la toti vecinii

b.

dupa ce primeste semnal de la parinte

c.

dupa ce primeste marcaj de la parinte

d.

dupa ce primeste marcaje de la toti vecinii, mai putin parintele

c? in curs zice de la prim, prim = parinte?

dar semnal sau **marcaj** ?da c am pus si io

------------------------------------------------------------------------------------

MPI este un standard pentru programarea multi-thread / MPI vine de la Message Passing Interface.

a.adevarat / fals

b.adevarat / adevarat

c.fals / fals

**d.fals / adevarat**

d?? a doua e adev, prima nush

eu zic ca b

In Java, ce face metoda submit(Runnable task) din ExecutorService?

a.

**Creeaza un nou task in thread pool**

b.

Creeaza un nou thread pool, in care vom avea task-ul respectiv

c.

Submite task-ul dat ca parametru in thread pool

d.

Porneste executia tuturor task-urilor din thread pool

help???

eu zic ca c

Prin metoda *submit()*, se introduce un task nou în pool, corespunzător directorului părinte -> e luat din lab

meris frumos, pwp

------------------------------------------------------------------------------------

Care este complexitatea inmultirii a doua matrici in paralel?

a.

O(N\*P)

b.

O(N)

c.

**O(N^3/P)**

d.

O(N^2/P)

AICI?

PLS?

------------------------------------------------------------------------------------

Ce complexitate are Odd Even Transposition Sort pentru n=p (unde n este numarul de elemente de sortat si p este numarul de thread-uri)?

a.

O( n \* log n )

b.

O( n ^ 2 )

**c.**

**O( n )**

d.

O( log n )

sigur e c?

------------------

In problema Producători și consumatori, M Producători, N Consumatori si k dimensiune buffer, de cate semafoare este nevoie pentru implementare, daca nu se mai poate folosi nici un alt mecanism de sincronizare?

a.

3

b.

**4**

c.

1

d.

2

3 ? Cineva ?

3 sau 4 ?

4 cred

4+1. raspuns 4

Care este un dezavantaj la soluția Reader priority (prioritate Cititorilor) pentru problema Cititori si Scriitori?

a.

exista posibilitatea ca cititorii sa nu mai aibă ce citii

b.

cititorii pot citi valori greșite

c.

un flux continuu de cititori pot bloca ulterior pe toți potențialii scriitori

d.

nici un scriitor nu va putea modifica niciodată resursa comuna

help???

C) e si in curs, ca daca se citeste in continuu, se poate sa nu intre scriitorii

Ce face o bariera?

a.

Forteaza re-activarea unui thread blocat pe o resursa

b.

Separa codul in bucati

c.

**Blocheaza thread-ul curent pana ce un numar dat de thread-uri ajunge la un apel al acesteia**

d.

Blocheaza thread-urile o perioada scurta de timp

c?

------------------------------------------------------------------

In MPI cum se primesc mesajele trimise prin apelul functiei MPI\_Bcast?

a.

Prin apel MPI\_Send

b.

Prin apel MPI\_Recv

c.

Prin apel MPI\_Get

d.

**Prin apel MPI\_Bcast**

------------------------------------------------------------------------------------

Considerând următorul pseudo-cod:

chan Can (int);

process P1{ int v1 =0;

v1 = v1+7; (ev1)

send Can(v1);} (ev2)

process P2{ int v2 =0;

v2 = v2+1; (ev3)

receive Can(v2); (ev4)

v2++;} (ev 5)

Se poate garanta ca pentru orice execuție:

a.

evenimentele ev1 si ev4 sunt concurente

b.

evenimentele ev2 si ev5 sunt concurente

c.

evenimentele ev2 si ev4 sunt concurente

**d.**

**evenimentele ev2 si ev3 sunt concurente**

stie cineva aici?

help???

------------------------------------------------------------------------------

Care este eficienta pentru sortarea OETS in cazul rularii pe P procesoare? Se considera N dimensiunea vectorului de sortat

a.

logP

b.

**logN/N**

c.

logN/P

d.

logN

------------------------------------------------------------------------------------

Care este timpul de trimitere a unui mesaj intr-un sistem distribuit, unde timpul de serializare este de 30 unități, un caracter se trimite in 6 unități si dimensiunea mesajului este 8?

a.

188

b.

78

c.

246

d.

44

A gasit cineva formula? nup

Tmsg = ts + tw\*L deci 78?

------------------------------------------------------------------------------------

Care este condiția pentru adăugarea unei noi regine (la problema Reginelor)? Oricare doua regine nu trebuie sa fie:

a.

**pe aceeași linie, coloana, diagonala principala sau secundara**

b.

pe aceeași linie, coloana sau diagonala principala

c.

pe aceeași linie sau coloana

d.

pe aceeași linie, coloana sau diagonala secundara

De ce poate aparea exceptia IllegalMonitorStateException in Java atunci cand apelam wait() pe un obiect?

a.

Thread-ul curent a fost intrerupt

b.

Obiectul respectiv nu a fost alocat

**c.**

**Thread-ul curent nu detine monitor lock-ului obiectului pe care apelam wait()**

d.

Nu a apelat nimeni notify() intr-o perioada data de timp

help?? c

------------------------------------------------------------------------------------

Care este timpul de executie in cazul descopunerii distribuite uni-dimensionale pe randuri a problemeii lui Floyd de aflare a drumui minim? Se considera o matrice NxN si P procese.

*t\_msg - este timpul de transmitere a unui mesaj (o linie din matrice) catre urmatorul proces destinatie*

*for [k = 0 to N-1]*

*for [i = local\_i\_start to local\_i\_end]*

*for [j = 0 to N-1]*

*I[i,j]k+1 = min(I[i,j]k, I[i,k]k + I[k,j]k)*

a.

**t\_iteratie\*N^3/P + N\*logP\*t\_msg**

b.

t\_iteratie\*N^3/P + N\*logP

c.

t\_iteratie\*N^3/P + N\*t\_msg

d.

t\_iteratie\*N^3/P + logP\*t\_msg

------------------------------------------------------------------------------------

*Detecția terminării folosind marcaje:* Un proces porneste algoritmul de terminare:

a.

dupa ce primeste marcaje de la toti vecinii

b.

dupa ce primeste semnal de la parinte

c.

dupa ce primeste marcaj de la parinte

**d.**

**dupa ce primeste marcaje de la toti vecinii, mai putin parintele**

-\_-idk a/d cred ca d

------------------------------------------------------------------------------------

Apelul this.wait() intr-o metoda non-statica sincronizata produce exceptie / In Java se poate initializa un Semaphore cu o valoare negativa (corespunzator numarului de permise)

**a.**

**fals / adevarat**

b.

adevarat / adevarat

c.

fals / fals

d.

adevarat / fals

-\_-

Ce complexitate are Odd Even Transposition Sort pentru n=p (unde n este numarul de elemente de sortat si p este numarul de thread-uri)?

**a.**

**O( n )**

b.

O( n \* log n )

c.

O( log n )

d.

O( n ^ 2 )

-\_-

In modelul distribuit de comunicare prin pasarea mesajelor:

**a.**

**se garantează ca mesajele trimise pe canal nu se vor pierde**

b.

memoria este grupată și partajabilă între procese

c.

se garantează respectarea ordinii de trimitere doar daca procesele comunica in mod sincron

d.

nu se garantează ca doua mesaje trimise unul după altul către aceeași destinație vor ajunge la procesul destinatar respectând ordinea de trimitere

-\_-

Care este rezultatul apelarii metodei .run() a unui thread Java?

a.

Deadlock, indiferent de felul in care este definita clasa MyThread.

b.

Generarea unei exceptii de tipul InvalidCallException.

**c.**

**Executia, pe firul curent, a continutului metodei.**

d.

Pornirea unui nou thread care va executa Runnable-ul definit de catre clasa.

-\_-

m as duce pe c, boon

In cazul algoritmului unda de tip arbore cu N noduri, timpul de execuție este:

a.

O(N)

b.

O(logN)

c.

O(N^2)

**d.**

**O(D)**

-\_-

Care este complexitatea pentru a calcula (eficient) in paralel distanta din fiecare punct al unei liste pana la sfarsitul acesteia? Lista are N elemente.

**a.**

**O(logN)**

b.

O(NlogN)

c.

O(N^2)

d.

O(N)

-\_-

Care este complexitatea Parallel Merge Sort?

a.

O( n ^ 2 )

b.

O( n \* log n )

**c.**

**O( n )**

d.

O( log n^2 )

-\_-

are u sure?no b/c, dar mai probabil c

Cine decide pe ce core va rula un anumit thread?

a.

Thread-ul in sine

**b.**

**Sistemul de operare**

c.

Nu decide nimeni, se alege aleator

d.

Procesorul

-\_-

Care este una dintre diferențele dintre un semafor binar si un mutex?

a.

semaforul binar oferă siguranță la ștergere, task-ul care deține semaforul binar nu pate fi șters (însă nu si mutex-ul)

b.

daca un task care deține semaforul binar se termina din diferite motive, SO poate reda resursa semaforului (însă nu si la mutex)

c.

nu exista nici o diferența

**d.**

**apare eroare daca un task care nu deține mutexul încearcă sa îl elibereze (însă nu si la semafor binar)**

-\_-

aia

da

asa ziceam si eu

forta

------------------------------------------------------------------------------------

De cate procese este nevoie pentru implementarea calcului polinomial cu Pipeline, considerând ca ultima putere a lui x este 6:

a.

**cel mult 7, se vor împarți in mod egal coeficienții**

b.

1 proces

c.

cel mult 6, se vor împarți in mod egal coeficienții

d.

exact 6, fiecare proces poate sa aibă exact un coeficient

-\_-a? no idea

------------------------------------------------------------------------------------

In algoritmul lui Huang, pentru a detecta terminarea:

a.

se folosesc mesaje de tip marcaj

b.

**se folosesc ponderi**

c.

se folosesc mesaje de tip marcaj si semnal

d.

se folosesc mesaje de tip semnal

-stie cineva? e bif W = 1 → declară “terminarea”

**b**

------------------------------------------------------------------------------------

La algoritmii cu mesaje de sondaj cu ecou, in cazul topologiilor de tip arbore (initiatorul fiind radacina arborelui):

a.

Mesajele de sondaj se propaga de la radacina la frunze / Mesajele de ecou se propaga de la initiator la frunze

**b.**

**Mesajele de sondaj se propaga de la radacina la frunze / Mesajele de ecou se propaga de la frunze la initiator**

c.

Mesajele de sondaj se propaga de la frunze la initiator / Mesajele de ecou se propaga de la initiator la frunze

d.

Mesajele de sondaj se propaga de la frunze la initiator / Mesajele de ecou se propaga de la frunze la initiator

------------------------------------------------------------------------------------

Stabilirea empirica a scalabilitatii unei implementari presupune:

a.

Rularea programului serial pe o masina cu foarte multe CPU-uri pentru a vedea efectul vectorizarilor efectuate de compilator.

**b.**

**Rularea cu numar diferit de thread-uri/procese si urmarirea timpului de executie.**

c.

Rularea programului cu input-uri de dimensiuni diferite si urmarirea timpului de executie pentru 1 thread.

d.

Rularea de un numar mare de ori a programului pentru a vedea daca da acelasi rezultat.

b? da

------------------------------------------------------------------------------------

Care este complexitatea de difuzare a unei valori intr-un sistem SIMD - EREW cu P procesoare?

a.

O(P)

b.

1

c.

O(PlogP)

d.

O(logP)

O parere?

din curs:

• Influenteaza performanta algoritmilor

• Exemplu: citirea valorii unei variabila partajate

• Un pas in CREW, CRCW

• log2N pași in EREW, ERCW

Deci d

------------------------------------------------------------------------------------

In Java folosim synchronized pentru a marca o zona critica / In Pthread folosim pthread\_barrier\_t pentru a marca o zona critica

**a.**

**adevarat / fals asta e clar**

b.

adevarat / adevarat

c.

fals / adevarat

d.

fals / fals

------------------------------------------------------------------------------------

In Java folosim synchronized pentru a marca o zona critica / In Pthread folosim pthread\_barrier\_t pentru a marca o zona critica

**a.**

**adevarat / fals**

b.

fals / fals

c.javaseri

fals / adevarat

d.

adevarat / adevarat

a cred

------------------------------------------------------------------------------------

Ce complexitate are cautarea paralela pe un vector cu N elemente si un sistem cu N/2 procesoare?

**a.O(1)**

b.O(N)

c.O(log(log(N)))

d.O(log(N))

Implementarea Java Thread face ca toate thread-urile sa se execute pe un singur CPU indiferent de cate procesoare are masina pe care ruleaza / Folosind Java Threads programatorul poate crea mai multe thread-uri decat numarul de CPU-uri de pe masina

**a.**

**fals / adevarat**

b.

adevarat / adevarat

c.

adevarat / fals

d.

fals / fals

a?

------------------------------------------------------------------------------------

In modelul distribuit de comunicare prin pasarea mesajelor:

a.

se garantează respectarea ordinii de trimitere doar daca procesele comunica in mod sincron

b.

se garantează ca mesajele trimise pe canal nu se vor pierde

c.

nu se garantează ca doua mesaje trimise unul după altul către aceeași destinație vor ajunge la procesul destinatar respectând ordinea de trimitere

d.

memoria este grupată și partajabilă între procese

b cred, eu cred ca e c

------------------------------------------------------------------------------------

Care dintre următoarele NU este o proprietate a algoritmilor undă?

a.

Tranzitivitate

b.

Terminare

c.

**Consistență**

d.

Decizie

a sau c???? C

Care este complexitatea algoritmului de Căutare binara paralela in sistemele SIMD - EREW (Exclusive Read Excusilve Write)? Se considera P procesoare si o secvență ordonata de N numere.

a.

O(NlogN)

b.

O(logN)

c.

O(logP)

d.

O(log(N/P))

**cineva? helppp cred ca d ++1 tot d**

Pentru analiza unui algoritm paralel consideram: P=numar procesoare, T=timpul de executie a algoritmului paralel si G=timpul de executie a celui mai rapid algoritm secevential. Costul (C) reprezinta:

a.

T\*P/G

b.

P\*G

c.

**T\*P**

d.

G/T\*P

c

=====================

In comunicarea intre procese, overhead-ul este definit ca:

**a.**

**durata pentru care procesorul este angajat în transmiterea sau recepția unui mesaj**

b.

cantitatea de resurse folosita de un nod de procesare in mediul distribuit

c.

intervalul minim de timp cat un proces trebuie sa aștepte între două transmiteri succesive

d.

întârzierea de transmitere a unui mesaj

which one? plss

Care este semnificatia unei sectiuni synchronized in Java?

a.

Sectiunea synchronized nu trebuie deloc executata

b.

Mai multe thread-uri pot executa sectiunea synchronized la acelasi moment de timp

c.

Un singur thread poate executa sectiunea synchronized la acelasi moment de timp

d.

In Java nu exista conceptul de synchronized

**cineva?**

**c cred +1**

In Java nu se poate folosi pentru sincronizare:

a.

O instanta a unui obiect diferit de cel care obtine lock-ul

b.

O clasa

c.

Instanta curenta a obiectului care obtine lock-ul

d.

**Un tip de date primitiv**

cred ca e a aici

Consideram un sistem cu n procesoare pe care dorim sa realizam in paralel suma partiala a elementelor unui vector ce contine n elemente. Sumele partiale se vor stoca la nivelul fiecarui proces. Care este regula ca la un anumit pas (j) un proces sa lucreze? (j porneste cu valoarea 1)

a.

rank\_proces - pow(2,j-1) >= 1

b.

rank\_proces - pow(2,j) < 1

c.

rank\_proces - pow(2,j) >= 1

d.

rank\_proces mod pow(2,j) == 0

a

Algoritmi distribuiti

------------------------------------------------------------------------------------

Cand avem in Java o metoda de tipul "public synchronized void f()" intr-o clasa C, pe monitor lock-ul carui obiect se face sincronizarea?

a.

f

b.

this

c.

C.class

d.

Un obiect nou creat special

b cred ++

c zic

------------------------------------------------------------------------------------

Avem un semafor in Java declarat cu "Semaphore s = new Semaphore(-2)". De cate "s.release()" este nevoie astfel incat un thread care apeleaza "s.acquire()" sa se deblocheze din asteptare?

a.

2

b.

1

c.

0

d.

**3**

------------------------------------------------------------------------------------

Un thread poate fi creat implementand interfata Runnable / Thread-urile sunt pornite apeland metoda thread.start()

a.

fals / adevarat ? trb new() ca sa fie creat

b.

adevarat / adevarat

c.

adevarat / fals

d.

fals / fals

b cred

Ce inseamna ca un lock este re-entrant?

a.

Daca un thread a luat deja lock-ul respectiv, va putea sa-l ia de oricate ori cat timp il detine

b.

Mai mult thread-uri pot sa intre in regiunea critica delimitata de acel lock

c.

Un thread care detine lock-ul se va bloca daca va incerca sa-l ia din nou

d.

Niciun thread nu poate sa intre in regiunea critica delimitata de acel lock

Ce este aici??a cred +1

------------------------------------------------------------------------------------

Cum este definita eficienta paralela ( E )? Se considera T - Timpul total necesar execuției algoritmului paralel; P - Numărul de procesoare utilizate G =Timp execuție cel mai rapid algoritm secvențial,

a.

**G/(T\*P)**

b.

T\*P

c.

G\*P/T

d.

T\*P/G

------------------------------------------------------------------------------------

Considerati urmatorul cod pentru sumele prefix, executat pe un sistem MIMD: process

suma[k=1 to n] {

for (j = 1; j < sup(log2 n); j++) {

temp[k] = a[k];

if (k - 2j-1 >= 1)

a[k] = temp[k-2j-1] + a[k];

}

}

Este nevoie sa se adauga mecanisme de sincronizare?

a.

Da, 1 mutex

b.

**Da, 2 bariere**

c.

Da, 1 bariera

d.

Nu

------------------------------------------------------------------------------------

In cazul algoritmului unda de tip arbore cu N noduri, numărul de mesaje trimise in sistem este:

a.N^2

b.N+2

c.2N

**d.N**

e d?algoritmul foloseşte cel mult N mesaje

------------------------------------------------------------------------------------

Considerati urmatorul cod pentru sumele prefix, executat pe un sistem SIMD:

process suma[k=1 to n] {

for (j = 1; j < sup(log2 n); j++) {

temp[k] = a[k];

if (k - 2j-1 >= 1)

a[k] = temp[k-2j-1] + a[k];

}

}

Este nevoie sa se adauga mecanisme de sincronizare?

a.

Da, 1 bariera

b.

Da, 1 mutex

c.

**Da, 2 bariere**

d.

Nu

anyone?

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

Cum se poate delimita o zona critica in Java?

a.

Folosind pthread\_lock()/pthread\_unlock()

b.

Folosind o bariera ciclica

**c.**

**Folosind synchronized**

d.

Folosind o bariera

c cred

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

Teorie Probleme

(cine vrea, poate contribui)

Algoritmi paraleli

------------------------------------------------------------------------------------

4. (0.5 puncte) Enumerați si definiți cele 3 proprietăți ale unui algoritm unda.

R:

‒ terminare fiecare calcul este finit

‒ decizie fiecare calcul conține cel puțin un eveniment de decizie decide

‒ dependență în fiecare calcul , fiecare decide este precedat cauzal de un

eveniment în fiecare proces

------------------------------------------------------------------------------------

Dați exemplu de un **race condition**. Explicați de ce e **race condition**.

Ex: a = a + 2 nu este operatie atomica. Are in spate 3 operatii atomice.

● Totul tine de cum thread-urile sunt planificate. Un core core este mai aglomerat, mai lent, etc

------------------------------------------------------------------------------------

3. (0.5 puncte) Definiți *(cate o fraza)* următoarele tipuri de defecte: Crash (Căderi), Omission (Omisiune), Timing (Temporizare)

R:

**Crash Fault (Caderi):** la un moment dat un proces/componenta devine

nefuncțional/nefuncțională. În ultimul pas, înainte de a se defecta, se poate să fi trimis numai un subset din mesaje

**Omission Fault** ( la un moment dat un proces nu reușește sa trimită sau sa primească anumite date/mesaje

**Timing Fault (Temporizare)**: Un proces/componenta răspunde înafara unui

interval de timp setat

------------------------------------------------------------------------------------

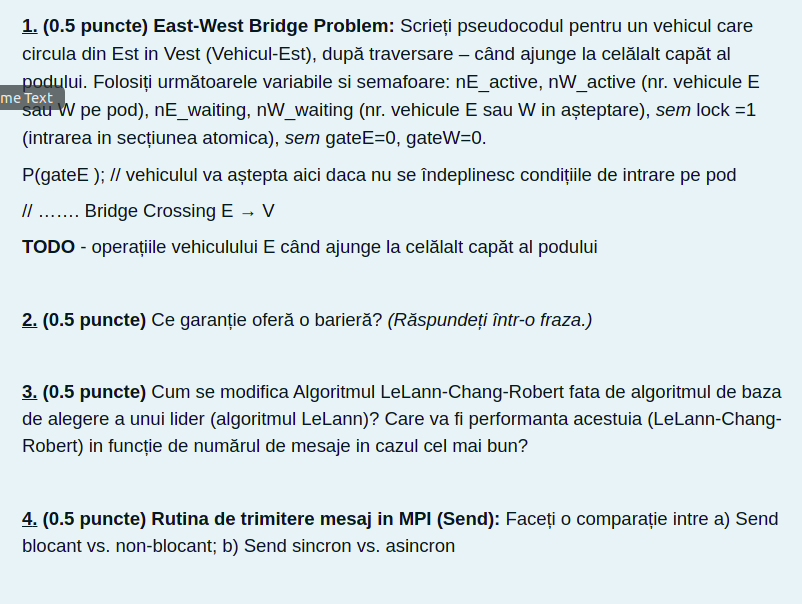
2. (0.5 puncte) Ce garanție oferă o barieră? *(Răspundeți într-o fraza.)*

*R: o barieră garantează că toate thread urile execută tot codul de dinainte*

*de barieră înainte de a executa orice linie de cod de după barieră*

----------------------------------------------------------------------------------

------------------------------------------------------------------------------------

-

1. P( lock ); // lock is a mutex and is initializes to 1.

if (( nw\_active + nw\_waiting == 0 ) && ( nr\_active < 3 )) {

nr\_active++; // notify we are active

V( r\_sem ); // allow ourself to get through

} else

nr\_waiting++; // we are waiting

V( lock );

P( r\_sem ); // readers will wait here, if they must wait.

READING...

P( lock );

nr\_active--;

if (( nr\_active == 0 ) && ( nw\_waiting > 0 )) { // if we are the last reader

count = 0; // makes sure we only at most 3 writers in

while (( nw\_waiting > 0 ) && ( count < 3 )) {

V( w\_sem ); // wake a writer;

w\_active++; // one more active writer

w\_waiting--; // one less waiting writer.

count++;

}

} else if (( nw\_waiting == 0 ) && ( nr\_waiting > 0 )) { // allow another waiting reader to go in, if

no waiting writers

V( r\_sem );

w\_active++; // one more active reader

w\_waiting--; // one less waiting reader.

}

V( lock );

-----------------------------------------------------------------------------------

1. (0.5 puncte) Ce garanție oferă o zonă critică? *Răspundeți într-o fraza.*

La un moment de timp, in aceasta zona se poate afla cel mult un thread.

2. (0.5 puncte) Considerați problema *Cititori-Scriitori*, varianta ce oferă prioritate scriitorilor. Scrieți pseudocodul pentru Cititor după ce a citit din resursa comuna (TODO):

process Cititor[i=1 to m] {

... // pseudocod de inceput

citește din resursa comună;

TODO – operațiile făcute de Cititor după ce a accesat resursa comuna

Folosiți următoarele variabile si semafoare: nr, nw (nr. cititori/scriitori ce folosesc resursa), dr,dw (nr. cititori/scriitori întârziați), *sem* e (intrarea in secțiunea atomica), *sem* r, *sem* w. Specificați cu cat se inițializează cele 3 semafoare.

process

Cititor [i=1 to m]{

while

true

P(e);

if

nw > 0 or dw >

dr

= dr + 1; V(e); P(r);

}

nr = nr + 1;

if

dr > 0) { dr = dr 1; V(r); }

else

if dr == 0) V(e);

citește din resursa comună;

P(e);

nr = nr

1;

if

(nr == 0 and dw > 0) { dw = dw 1; V(w); }

else

if (nr > 0 or dw == 0) V(

}

}

3. (0.5 puncte) Cum se modifica Algoritmul LeLann-Chang-Robert fata de algoritmul de baza de alegere a unui lider (algoritmul LeLann)? Care va fi performanta acestuia (LeLann-Chang-Robert) in funcție de numărul de mesaje in cazul cel mai bun?

4. (0.5 puncte) Specificați care sunt pașii realizați de un proces pentru terminare in cazul Detecției terminării folosind marcaje.

------------------------------------------------------------------------------------

1. (0.5 puncte) Considerați problema *Producător-Consumator* cu M producători, N consumatori si buffer de dimensiune k>1. Corectați următorul pseudocod pentru Consumator, prin adăugarea de mecanisme de sincronizare (semafoare). Orice semafor folosit trebuie si inițializat.

process Consumator[i=1 to N] {

typeT w;

while (true) {

w = buf[prim];

prim = prim mod k + 1;

consuma(w);

} }

2. (0.5 puncte) Ce garanție oferă o barieră? *(Răspundeți într-o fraza.)*

*Asigura ca niciun thread nu poate trece mai departe de punctul in care este plasata decat atunci cand toate threda-urile gestionate de bariera ajung in acel punct*

3. (0.5 puncte) Definiți *(cate o fraza)* următoarele tipuri de defecte: Crash (Căderi), Omission (Omisiune), Timing (Temporizare)

R:

**Crash Fault (Caderi):** la un moment dat un proces/componenta devine

nefuncțional/nefuncțională. În ultimul pas, înainte de a se defecta, se poate să fi trimis numai un subset din mesaje

**Omission Fault** ( la un moment dat un proces nu reușește sa trimită sau sa primească anumite date/mesaje

**Timing Fault (Temporizare)**: Un proces/componenta răspunde înafara unui

interval de timp setat

4. (0.5 puncte) Enumerați si definiți cele 3 proprietăți ale unui algoritm unda.

E mai sus, dar dau copy-paste și aici.

R:

‒ terminare fiecare calcul este finit

‒ decizie fiecare calcul conține cel puțin un eveniment de decizie decide

‒ dependență în fiecare calcul , fiecare decide este precedat cauzal de un

eveniment în fiecare proces

------------------------------------------------------------------------------------

------------------------------------------------------------------------------------

Algoritmi distribuiti