Documentatie Proiect 1: Client-Server

Timpii de executie:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Reader | Writers | ∆t (ms) | ∆x (s) | T (ms) |
| 4 | 4 | 1 | 1 | 159790 |
| 1 | 2 | 159632 |
| 2 | 1 | 158093 |
| 2 | 2 | 158165 |
| 4 | 1 | 157865 |
| 4 | 2 | 158653 |
| 2 | 2 | 1 | 1 | 152135 |
| 1 | 2 | 151356 |
| 2 | 1 | 151630 |
| 2 | 2 | 150356 |
| 4 | 1 | 153654 |
| 4 | 2 | 156481 |
| 4 | 2 | 1 | 1 | 162541 |
| 1 | 2 | 156542 |
| 2 | 1 | 160751 |
| 2 | 2 | 162548 |
| 4 | 1 | 163524 |
| 4 | 2 | 156684 |
| 4 | 8 | 1 | 1 | 153452 |
| 1 | 2 | 156847 |
| 2 | 1 | 152203 |
| 2 | 2 | 153214 |
| 4 | 1 | 149652 |
| 4 | 2 | 150861 |

Configuratia optima:

Cel mai rapid timp de executie (T) este cu 4 cititori si 8 scriitori, cu un ∆t de 4 ms si un ∆x de 1 s.

Consistenta intre rulari:

Timpii de executie sunt relativ consistenti in diferite configuratii, cu variatii minore. Aceasta ar putea insemna ca sistemul este stabil si previzibil sub diferitele conditii testate.

Limbaj folosit: Java

Mod de rezolvare:

Pentru conexiunea dintre server si clienti am folosit socket-uri (ServerSocket, Socket din java.net). Clientii trimit cereri de UPDATE\_SCORE, RANKING si FINAL\_RANKING. Serverul raspunde cu clasamentul pe tari partial pentru cererea de tip RANKING si cu cel final pentru FINAL\_RANKING.

Folosind coada si lista din laboratul 5, mecanismele folosite sunt asemanatoare (variabile conditionale, lock-uri, etc). Serverul, dupa ce porneste, asteapta sa primeasca mesaje de la clienti, in momentul cand primeste un mesaj, daca acesta este UPDATE\_SCORE, adauga in coada scorurile primite de la participanti. Daca mesajul este RANKING, incearca calcularea clasamentrului, iar daca mesjul este FINAL\_RANKING, trimite clasamentul final. Clasamentul partial se calculeaza folosind un future, care se creeaza, urmand ca mai apoi sa se masoare timpul de cand a fost creat pana cand a fost obtinuta valoarea lui, iar daca acesta este mai mic decat delat t, se returneaza raspunsul.

Concluzie: In urma analizei efectuate, constatam ca in timp ce variatia numarului de readeari nu influenteaza asa tare rularea, eventuali timpi mai mici, au fost obtinuti folosing mai multe threaduri pentru writeri, de unde putem deduce ca de cele mai multe ori, nu producerea pune neaparat probleme, ci consumul.