Tema 3 – Algoritmi paraleli in sisteme distribuite

responsabili: prof. Ciprian Dobre, ing. Alexandru Hogea, Cristian Condruz

Premisa tema:

In aceasta tema veti folosi atat conceptele de programare paralela dobandite pana acum, cat si noile concepte de programare distribuita insusite pe parcursul laboratoarelor de MPI.

Obiectivele temei:

1. sa invatati sa utilizati MPI
2. sa implementati un algoritm bazat pe sisteme distribuite
3. sa folositi fire de executie intr-un sistem distribuit
4. sa aplicati si sa *replicati* modelul *producator consumator* intre mai multe noduri distribuite ale unui cluster
5. sa sincronizati resurse procesate in paralel si distribuit

Enunt:

Se doreste implementarea unui procesator de text distribuit. Procesatorul va trebui sa transforme textul primit, aplicand diverse reguli pe cuvinte in functie de genul unui paragraf. Genurile sunt ***horror***, ***comedy***, ***fantasy***, ***science-fiction***. Paragrafele *horror* vor trebui sa aiba *consoanele dublate* in fiecare cuvant (*dublarea se va face doar in minuscula*). Paragrafele *comedy* vor avea *fiecare litera de pe pozitie para facuta majuscula*. Paragrafele fantasy vor trebui sa aiba prima litera a fiecarui cuvant facuta *majuscula.* Paragrafele science-fiction for trebui sa aiba fiecare al 7lea cuvant *inversat*.

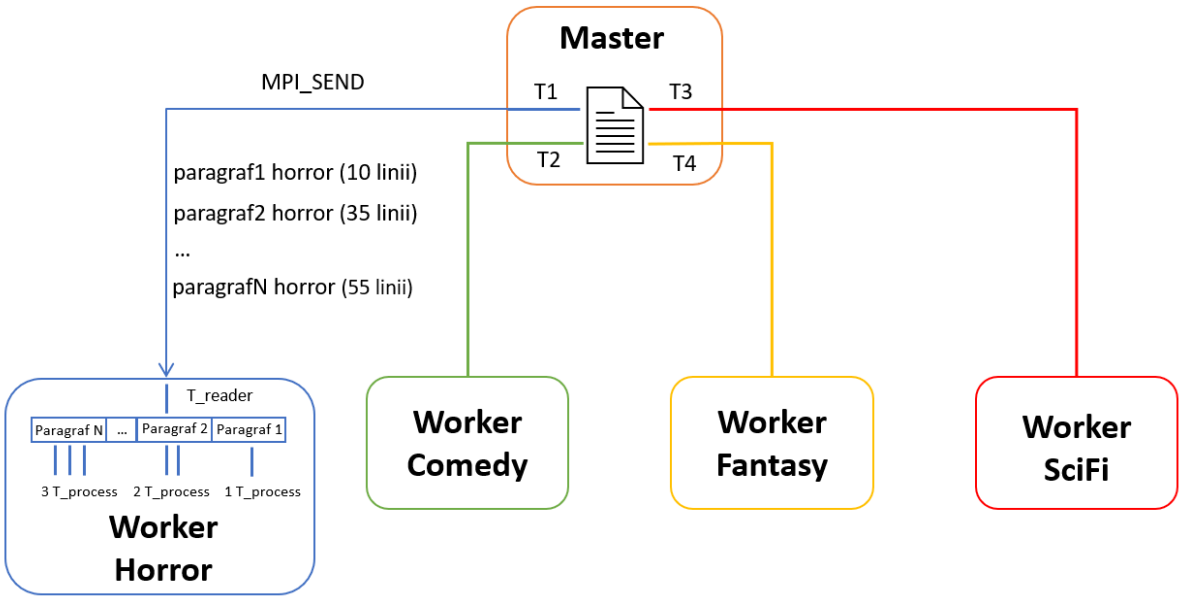
Procesatorul de text va fi bazat pe 5 noduri de MPI (un nod master si 4 noduri worker, pt fiecare gen in parte). Nodul master va trebui sa ruleze exact 4 fire de executie (pthreads), pentru a procesa fisierul de intrare in paralel si pentru a comunica cu workerii in paralel (cate un fir de executie per worker).

Workerii vor primi textul, ***paragraf cu paragraf***, in functie de genul paragrafului.

Workerii vor avea ***maxim P fire de executie***, unde P este nr de fire de executie disponibile pe sistemul pe care ruleaza nodul. Din cele P fire de executie, un fir de executie se va ocupa de receptarea datelor de la master, si celelalte P-1 fire de executie vor procesa datele primite. Pentru fiecare 20 de linii in plus la paragraf, se va crea un fir de executie nou (de exemplu, daca paragraful primit are 21 de linii, vor trebui sa existe 2 fire de executie de procesare deci, in total, 3 fire de executie – 1 reader + 2 de procesare -). Se considera linie *textul paragrafului, mai putin* ***genul****.*

Procesatorul va scrie intr-un fisier text de iesire textul modificat, in format identic cu formatul de intrare (se va pastra inclusiv si genul paragrafului).

Firul de executie principal (main) nu intra in calculul numarului de fire de executie ce trebuie rulate (4 pe nodul master, respectiv maxim P pe workeri).



Date de intrare:

Datele de intrare vor fi reprezentate de un fisier text, in care vor fi scrise paragrafe si genurile lor dupa cum urmeaza:

horror
Aceasta tema pare
Imposibil de grea si
Speram ca nu e imposibil sa o facem

comedy
da, dar e tema 3 si mereu temele 3 au fost usoare
plus ca e scrisa de Hogea, toate temele lui sunt accesibile

fantasy
cel
mai
probabil
se
face in 2-3 ore maxim

science-fiction
stiti ca tema asta contine si MPI si Pthreads, nu?

horror
Da...

Asa cum observati, paragrafele ***incep intotdeauna*** cu genul lor, dupa care se pune un \n. Doua paragrafe sunt despartite de ***doua \n.***

Date de iesire:

Datele de iesire vor fi reprezentate de un fisier text, in care va fi scris textul procesat. ***ORDINEA PARAGRAFELOR SE VA RESPECTA!***

horror
Acceasstta ttemma pparre
Immppossibbill dde ggrrea ssi
Sspperramm cca nnu e immppossibbill ssa o ffaccemm

comedy
dA, dAr e tEmA 3 sI mErEu tEmElE 3 aU fOsT uSoArE
pLuS cA e sCrIsA dE HOgEa, tOaTe tEmElE lUi sUnT aCcEsIbIlE

fantasy
Cel
Mai
Probabil
Se
Face In 2-3 Ore Maxim

science-fiction
stiti ca tema asta contine si IMP si Pthreads, nu?

horror
Dda...

TL;DR enunt:

- procesator de text
  - text impartit in paragrafe
  - fiecare paragraf are un gen
  - procesare in functie de gen
- 5 noduri MPI
  - 1 nod master, care citeste textul in paralel, in 4 fire de executie pthreads si trimite catre workeri paragrafe de genul respectiv
  - 4 noduri worker, care primesc paragrafe in cate un fir de executie pthread si proceseaza paragrafele in P-1 fire de executie
    - P este minim 2, maxim nr de fire de executie disponibile pe sistemul gazda
    - Pentru fiecare 20 de linii din paragraf, se mai foloseste inca un fir de executie pentru procesare

        - Genul, scris pe prima linie, nu intra in calculul numarului de linii
        - Paragraful **poate sa aiba** mai mult de (P-1) * 20 de linii
        - Nodul worker **nu poate sa aiba** mai mult de P fire de executie
    - se pastreaza ordinea paragrafelor procesate in textul output

Punctare:

- **4p**, daca se implementeaza solutia folosind **doar** MPI
- **6p**, daca se realizeaza citirea si trimiterea textului catre workeri in paralel, in master
- **8p**, daca se separa logica de citire si procesare in workeri
- **10p**, daca tema scaleaza pe texte mari

Se vor aplica depunctari de pana la **6p** daca nu se respecta numarul de procese (5, 1 master, 4 workers) sau nu se respecta limitarile de fire de executie (daca un paragraf are 1000 de linii, sa nu se porneasca 50 de fire de executie, ci NR_MAX_THREADS).
Daca ordinea paragrafelor, la scriere, nu este respectata, punctajul temei va fi anulat complet.

Arhiva:

Arhiva se va numi NUME_PRENUME_GRUPA_TEMA3_APD_2020**.zip** si va contine urmatoarele:

- fisierele sursa (.c, .cpp)
- fisiere header (*daca este cazul*)
- Makefile
- Readme
- *Feedback.txt (optional, cine doreste)*

In readme se va dovedi scalabilitatea solutiei (daca exista)


MERRY
CHRISTMAS
& HAPPY NEW YEAR