Implementacija paralelizacije operacija nad podacima u binarnom stablu

Ovaj program implementira razne operacije nad binarnim stablom:

- Sekvencijalnu i paralelnu pretragu.
- Sekvencijalno i paralelno sumiranje vrednosti cvorova.
- Brojanje parnih i neparnih vrednosti cvorova.
- Pronalazenje minimalne i maksimalne vrednosti.

Sve operacije koriste OpenMP za paralelizaciju. Cilj je uporediti performanse sekvencijalne i paralelne obrade sa razlicitim brojem niti (2, 4 i 6), kako bi se izracunalo ubrzanje postignuto paralelizacijom.

Ako je dubina podstabla veca od zadatog praga, primenice se paralelizacija, inace ce se obaviti sekvencijalna obrada. Ovim nacinom je izbegnuto nepotrebno kreiranje i sinhronizacija izmedju taskova (utice na performanse) za malu dubinu podstabala.

Opis rada programa

Svaki cvor stabla sadrzi vrednost (*data*), koja je broj od 0 do 9 i pokazivace na levo i desno podstablo (*left* i *right*).
Dubina binarnog stable u ovom programu je 24.
Funkcija generate_tree rekurzivno kreira stablo sa zadatom dubinom. Svaki cvor dobija vrednost na osnovu dubine (*depth % 10*).

1. Pretraga binarnog stabla

- **Sekvencijalna pretraga (search_tree_sequential)**:
  Rekurzivno se prolazi kroz sve cvorove stabla.
- **Paralelna pretraga (search_tree_parallel)**:
  Koristeci OpenMP pragme, kreiraju se paralelni taskovi za obradu levog i desnog podstabla. Taskovi se kreiraju samo ako je dubina veca od praga (*threshold*), cime se izbegava prevelika sinhronizacija.

OpenMP pragme:

- #pragma omp task shared(left_count) – kreira task za pretragu levog podstabla.

- #pragma omp task shared(right_count) – kreira task za pretragu desnog podstabla.
- #pragma omp taskwait – ceka da oba taska zavrse pre nego sto kombinuje rezultate.
- #pragma omp atomic – obezbedjuje atomicnu operaciju za azuriranje globalnog brojaca.

---

2. Sumiranje vrednosti cvorova

- **Sekvencijalno sumiranje (calculate_sum_sequential)**:
  Prolazi kroz celo stablo rekurzivno, vracajuci zbir vrednosti svih cvorova.
- **Paralelno sumiranje (calculate_sum_parallel)**:
  OpenMP taskovi omogucavaju istovremeno sabiranje vrednosti iz levog i desnog podstabla.

OpenMP pragme:

- #pragma omp task shared(left_sum) – kreira task za sabiranje vrednosti u levom podstablu.
- #pragma omp task shared(right_sum) – kreira task za sabiranje vrednosti u desnom podstablu.
- #pragma omp taskwait – sinhronizuje taskove kako bi se kombinovali rezultati.

---

3. Brojanje parnih i neparnih vrednosti

- **Sekvencijalno brojanje (count_even_odd_sequential)**:
  Sekvencijalno prolazi kroz stablo i povecava brojac parnih ili neparnih vrednosti.
- **Paralelno brojanje (count_even_odd_parallel)**:
  Paralelno broji parne i neparne vrednosti pomocu OpenMP taskova.

OpenMP pragme:

- #pragma omp task shared(left_even, left_odd) – kreira task za brojanje parnih i neparnih vrednosti u levom podstablu.
- #pragma omp task shared(right_even, right_odd) – kreira task za brojanje parnih i neparnih vrednosti u desnom podstablu.
- #pragma omp taskwait – ceka zavrsetak taskova.
- #pragma omp atomic – azurira globalne brojace parnih i neparnih vrednosti.

---

4. Pronalazenje minimalne i maksimalne vrednosti

- **Sekvencijalno pronalazenje (find_min_max_sequential)**:
  Sekvencijalno prolazi kroz stablo i azurira minimalnu i maksimalnu vrednost.
- **Paralelno pronalazenje (find_min_max_parallel)**:
  Koristi OpenMP taskove za istovremeno pronalazenje minimalne i maksimalne vrednosti u levom i desnom podstablu.

OpenMP pragme:

- #pragma omp task shared(left_min, left_max) – kreira task za pronalazenje min/maks u levom podstablu.
- #pragma omp task shared(right_min, right_max) – kreira task za desno podstablo.
- #pragma omp taskwait – sinhronizuje taskove.
- #pragma omp critical – kombinuje rezultate iz svih taskova koristeci kriticnu sekciju.

Merenje performansi

Merenje vremena izvrsavanja sekvencijalne i paralelne obrade vrsimo koristeci **omp_get_wtime()**.
Ubrzanje racunamo tako sto podelimo vreme sekvencijalne obrade sa vremenom paralelne obrade.

Ispis rezultata

- **Za sekvencijalnu i paralelnu pretragu:**
  - Broj pojavljivanja trazene vrednosti.
  - Vreme izvrsavanja sekvencijalne i paralelne pretrage.
  - Ubrzanje za 2, 4 i 6 niti.
- **Za sumiranje, brojanje parnih/neparnih vrednosti i pronalazenje min/maks vrednosti:**
  - Rezultati sekvencijalne i paralelne verzije.
  - Ubrzanje za svaku operaciju.

Testiranje

Testiranje se vrsi za stablo dubine 24 i trazenu vrednost 3.
Rezultati pokazuju da paralelizacija znacajno smanjuje vreme izvrsavanja, posebno za duboka stabla.

```
__________ Testiranje sa 2 niti __________

Pretraga: Serijska = 2099202, Paralelna = 2099202
Ubrzanje pretrage: 1.87
Sumiranje: Serijska = 33226426, Paralelna = 33226426
Ubrzanje sumiranja: 1.62
Brojanje: Serijski (Parni = 5592405, Neparni = 11184810), Paralelni (Parni = 5592405, Neparni = 11184810)
Ubrzanje brojanja: 2.00
Min/Max: Serijski (Min = 0, Max = 9), Paralelni (Min = 0, Max = 9)
Ubrzanje Min/Max: 2.44
_________________________________________
```

```
__________ Testiranje sa 4 niti __________

Pretraga: Serijska = 2099202, Paralelna = 2099202
Ubrzanje pretrage: 1.86
Sumiranje: Serijska = 33226426, Paralelna = 33226426
Ubrzanje sumiranja: 2.11
Brojanje: Serijski (Parni = 5592405, Neparni = 11184810), Paralelni (Parni = 5592405, Neparni = 11184810)
Ubrzanje brojanja: 2.19
Min/Max: Serijski (Min = 0, Max = 9), Paralelni (Min = 0, Max = 9)
Ubrzanje Min/Max: 2.44
_________________________________________
```

```
__________ Testiranje sa 6 niti __________

Pretraga: Serijska = 2099202, Paralelna = 2099202
Ubrzanje pretrage: 2.99
Sumiranje: Serijska = 33226426, Paralelna = 33226426
Ubrzanje sumiranja: 3.04
Brojanje: Serijski (Parni = 5592405, Neparni = 11184810), Paralelni (Parni = 5592405, Neparni = 11184810)
Ubrzanje brojanja: 2.99
Min/Max: Serijski (Min = 0, Max = 9), Paralelni (Min = 0, Max = 9)
Ubrzanje Min/Max: 2.92
_________________________________________
```