Vilniaus Universitetas

Matematikos ir Informatikos Fakultetas

Laimonas Beniušis, Kompiuterių Mokslas 1

Lygiagrečiųjų skaičiavimų tyrimas

Užduotis 4

Rūšiavimas liejimo būdu

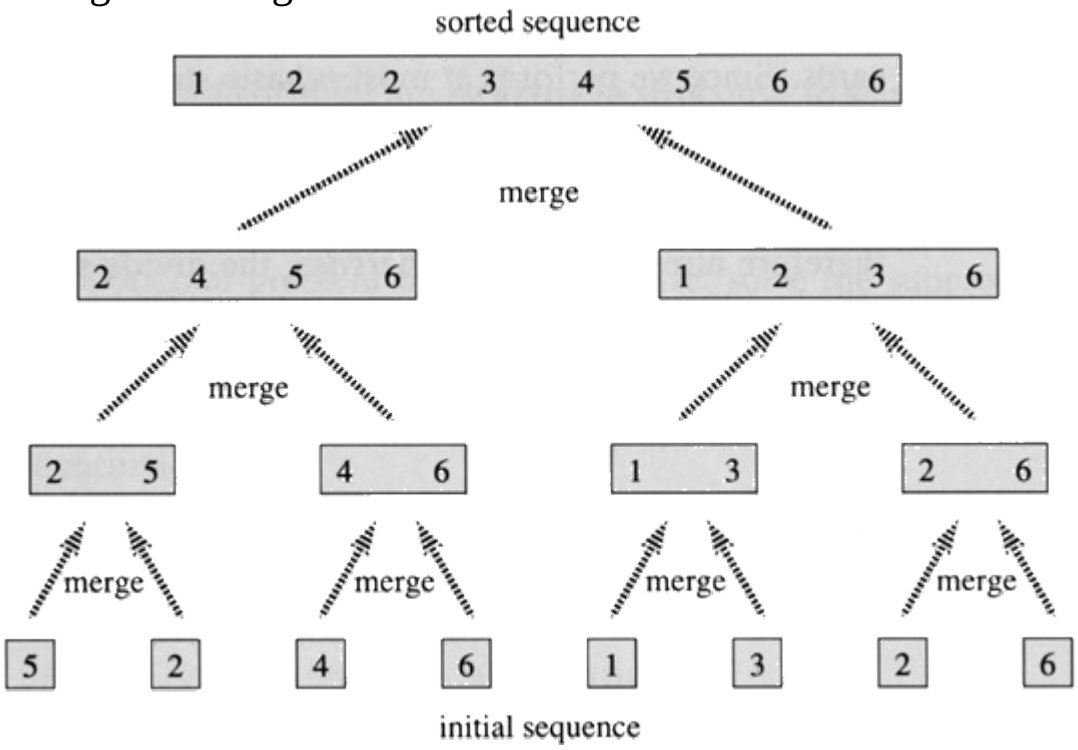
Merge sort

Turinys

Užduotis

Duomenų rūšiavimas liejimo būdu multiprocesinėje aplinkoje, pasinaudojant MPI technologijomis.

“Merge Sort” algoritmas:



Kada yra dalinami maži masyvai, dažnai neefektyvu toliau dalinti iki tol kol dydis pasiekia 1, todėl yra naudojami paprastesni algoritmai (pvz “Insertion sort”), kurie surūšiuoja šiuos mažus masyvus. Vėliau juos belieka sulieti.

Algoritmo versijos:

1. Išankstinio paskirstymo liejimas
2. Patobulintas išankstinio paskirtymo liejimas
3. Medžio formos priklausomybių liejimas

Išankstinio paskirstymo liejimas

Naudojama MPI_Scatter ir MPI_Gather

NP – naudojami procesai

N – masyvo dydis

Nerūšiuotas masyvas išskaidomas į N/NP dydžio gabaliukus, kuriuos atskiri procesai surūšiuoja.

Tada surūšiuoti gabaliukai surenkami į finalinį masyvą.

Pagrindinis procesas surūšiuoja finalinį masyvą.

Patobulintas išankstinio paskirtymo liejimas

Didžioji dalis sutampa su išankstinio paskirstymo liejimo algoritmu. “Merge Sort” funkcija naudoja “Insertion Sort”, kada masyvo dydis pasiekia <=16.

Finalinio masyvo liejimo metu, naudojama kitokia “Merge Sort” funkcija, kuri skanuoja masyvą ieškodama “tarpo” liejimo metu, tokio būdu yra minimizuojamos masyvo dalinimo operacijos.

Medžio formos priklausomybių liejimas

Kol yra napanaudotų procesų, daliname masyvą pusiau ir duodame tam procesui rūšiuoti.

Naudojamas MPI_Isend (neblokuojančiai siunčia) ir MPI_Recieve

Kadangi yra optimaliai dalinami rūšiavimai, nereikia skanuoti kur dalinti masyvus finaliniame rūšiavime. Naudojamas “Insertion Sort” mažiems masyvams rūšiuoti.

Pvz. Jeigu NP – 9 N - dydis

0 → 1,2,4,8

1 → 3,5

2 → 6

3 → 7

Pasidalinta taip:

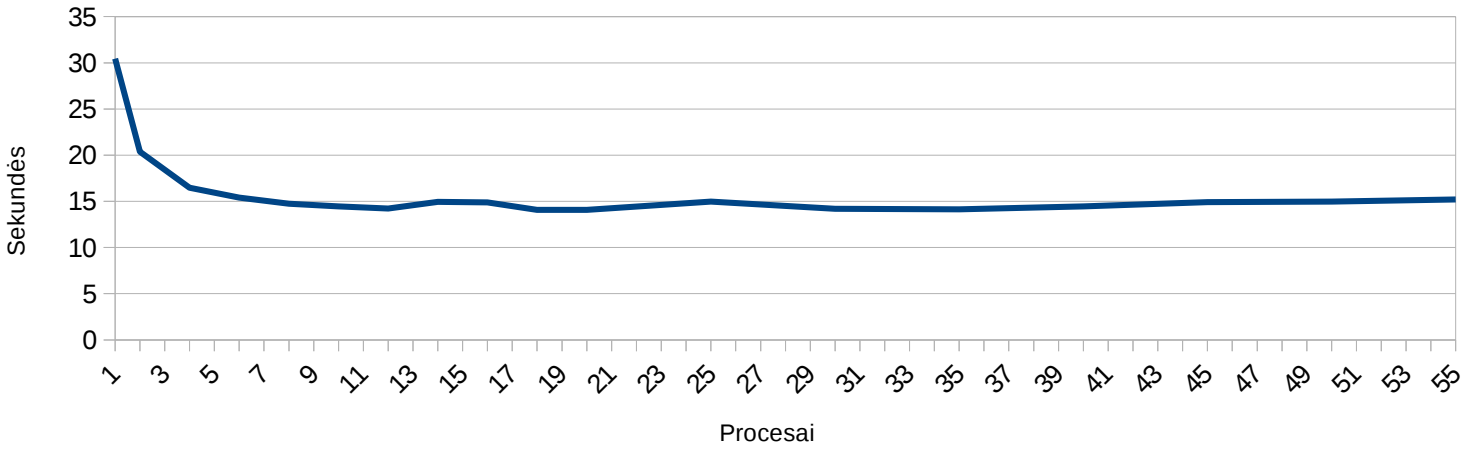
N / 16 dalį gavo 2 procesai: 0,8

N / 8 dalį gavo 7 procesai: 1,2,3,4,5,6,7

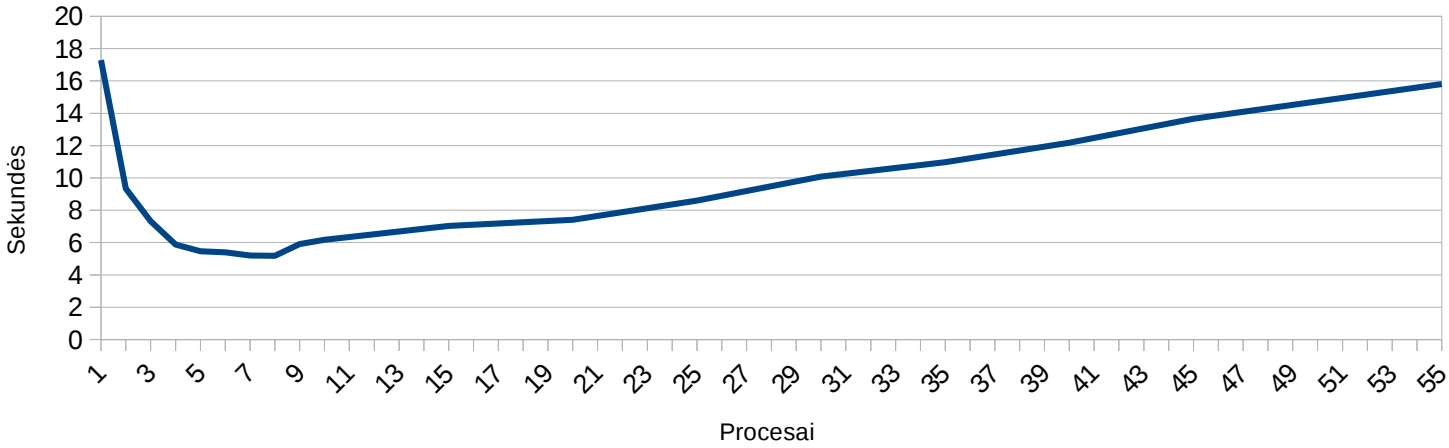
Testavimas

Masyvo dydis 50 000 000

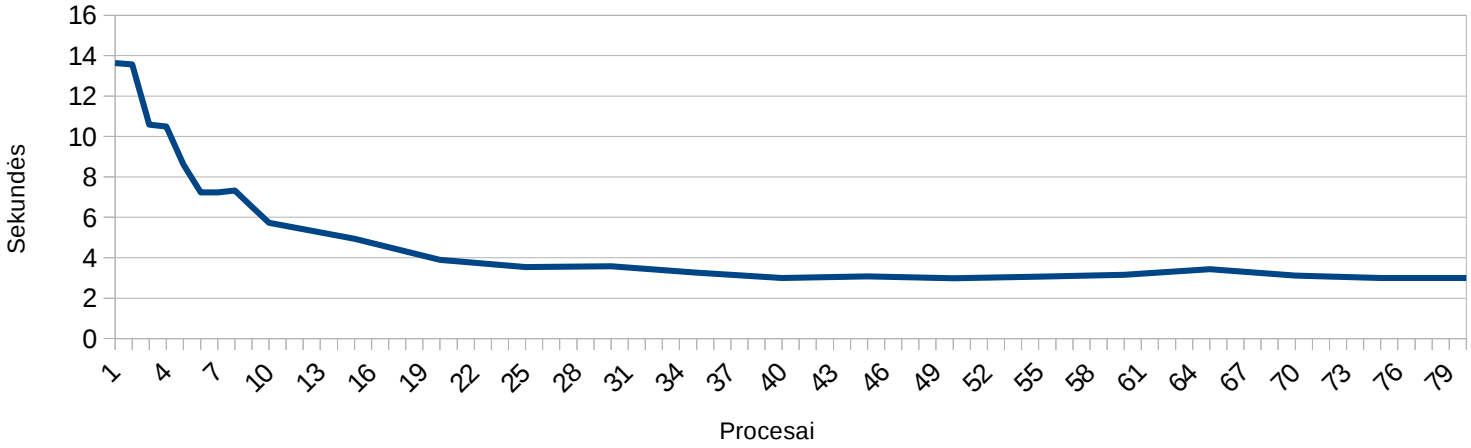
1-as



2-as



3-ias



Spartėjimas

Algoritmas 1

Procesų intervale [1, 5] spartėjimas yra 10%

Procesų intervale [6, 12] spartėjimas sumžėja iki 1-3%

Toliau spartėjimas sustoja ar netgi tampa neigiamas

Algoritmas 2

Procesų intervale [1,5] spartėjimas yra 10-30%

Procesų intervale [6,8] spartėjimas yra 1-4%

Toliau spartėjimas tampa neigiamas.

Algrotimas 3

Procesų intervale [1,2] spartėjimas yra 1%

Procesų intervale [3,6] spartėjimas yra 20%

Procesų intervale [7,25] spartėjimas yra 0-1%

Toliau spartėjimas svyruoja ties 0%.

Išvados

Vienintelis 1-o algoritmo privalumas yra tai, kad jis yra gana paprastas, tačiau labai neefektyvus.

2-as algoritmas yra labai efektyvus, kada turime mažai masyvo skilčių, t.y. NP<10. Šis algoritmas nedaug skiriasi nuo 1-ojo, todėl taip pat yra gana paprastas.

3-as algoritmas yra efektyvus kada turime NP>9. Kada NP < 10 darbo padalinimas nesutaupo pakankamai laiko, kad komunikacija tarp procesų būtų pateisinama, tačiau kai NP >9, darbo išlygiagretinimo efektyvumas aplenkia 2-ą algoritmą, tačiau daugiau negu 40 procesų naudoti nėra verta.