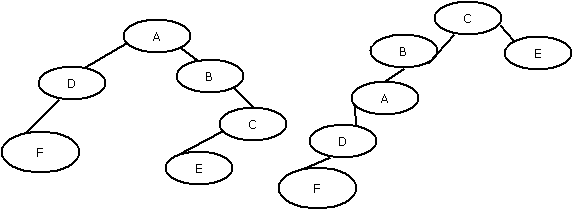
|  |
| --- |
| Vilniaus universitetas  Matematikos ir informatikos fakultetas |
| Algoritmų analizė |
| Laboratorinis darbas |
| 3 kursas, 1 grupė, Kompiuterių mokslas |
| **Laurynas Paradauskas** |
| **4/10/2012** |

|  |
| --- |
|  |

**Uždavinio formuluotė**

**Duota:** Du neorientuoti medžiai *T*1 ir *T*2, turintys *n* viršūnių ir *m* briaunų. (Medžiai gali būti realizuoti tėvų sąrašu).

**Rasti:** Nustatyti, ar šie medžiai yra izomorfiški.

****

**Realizuotų algoritmų aprašymai**

**1 algoritmas. Radix sort**

1. Nusiskaitome duomenis iš 2 failų (failuose medžiai vaizduojami tėvų nuorodomis), susidedam sąrašą.
2. Tada paleidžiam 2 gijas, kurios rušiuoja medžių viršūnes pagal tėvų nuorodas. Rūšiavimas realizuojamas:
   1. M=10, N=1 iš pradžių.
   2. Praeiname pirmą kartą sąrašą ir susidedame į sunumeruotus kibirus gautas nuorodas pagal taisyklę:

vieta := (vaikas.getTėvas mod M)/N

kibirai[vieta].add(vaikas);

* 1. Tada einame į laukimo barjerą, kol atsiras 2 laukiančios gijos.
  2. Sulaukus 2 gijų patikriname, ar kibirai1 – pirmos gijos, kibirai2 – antros gijos dydžiai yra lygūs, tai yra:
     1. Tikriname, ar kibirai1[vieta].size ,
     2. Atitinkamai su kibirai2[vieta].size ,
     3. Jeigu bent 1 vietos dydžio nerandame kitame, tada algoritmas nutraukiamas ir medžiai yra neizomorfiški.
  3. Gavau “true” reikšmę iš tikrinimo, tęsiame algoritmą, tol kol pradinio masyvo dydis nebus lygus kibirai[0].size. Tai yra, M = M\*10, N=N\*10 ir einame į **b** dalį.

1. Negavus jokios klaidos, spausdiname rezultatą, jog grafai izomorfiški ir atspausdiname darbo trukmę.

Eksperimentų lentelė:

|  |  |
| --- | --- |
| Vaikų skaičius | Darbo trukmė (sekundėmis) |
| 6 | 0 |
| 60 | 0,016 |
| 120 | 0,016 |
| 300 | 0,015 |
| 600 | 0,016 |
| 6000 | 0,063 |
| 1800 | 0,406 |
| 30000 | 2,485 |
| 60000 | 22,937 |
| 72000 | 25,344 |
| 90000 | 41,516 |
| 120000 | 111,984 |

Kadangi medžiuose vaikas gali turėti tik 1 tėvą, tai briaunų skaičius priklauso nuo vaikų skaičiaus ir yra lygus M=N-1 , kur M – briaunų skaičius, o N – vaikų skaičius. Todėl sudėtingumas priklauso tik nuo vaikų skaičiaus.

Iš darbo laiko kreivės matome, kad uždavinio sudėtingumas – eksponentinis. Sudėtingumo analizei buvo naudojami atsitiktinai sugeneruoti tėvų nuorodų sąrašas. Darbo trukmę galėtume sumažinti surasdami geresnį algoritmą „kibirų“ dydžių tikrinimui.

Uždavinio sudėtingumas: O(N2)

**2 algoritmas. Tėvų skaičiavimas**

Skaičiuojame, kiek kiekvieno grafo viršūnė turi tėvų nerūšiuotame sąraše.

Eksperimentų lentelė:

|  |  |
| --- | --- |
| Vaikų skaičius | Darbo trukmė (sekundėmis) |
| 6 | 0,016 |
| 60 | 0,016 |
| 120 | 0,125 |
| 300 | 3,406 |
| 600 | 52,562 |
| 6000 | Daugiau nei 780 |

Dabar kaip ir pirmo algoritmo atveju gavome eksponentinį darbo laiko augimą, tačiau šiuo atveju kuomet vaikų skaičius 6000 programos darbą nutraukėme nelaukdami programos pabaigos.

**Išvados:** gerai apgalvotas algoritmas žymiai sutrumpina darbo laiką ir leidžia programai dirbti su didesniu kiekiu duomenų. Medžių izomorfizmo tikrinimo atveju sudėtingumas ir darbo laikas priklauso tik nuo 1 parametro – vaikų skaičiaus, nes briaunų skaičius yra atitinkamai lygus iš vaikų skaičiaus atėmus 1.

**Programos naudojimo instrukcija**

Programa neturi grafinės vartotojo sąsajos. Dirbama su ja konsoliniu rėžimu.

Paleisti programą: java Main

Grafus pateikiame aprašydami tėvų sąrašus 2 atskiruose failuose.

Pasileidę programą suvedame duomenų failų vardus ir laukiame rezultato.