VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Algoritmų analizės projektinis namų darbas
Grafo viršūnių spalvinimo algoritmai

Atliko: 3 kurso 3 grupės studentas
Aurimas Petrėtis

Vilnius
2020

Turinys

Įvadas

Grafo G = (V, E) viršūnių spalvinimu vadinsime spalvų priskyrimą iš spalvų aibės S kiekvienai viršūnei iš aibės V tokį, kad kiekvienos briaunos galai būtų nudažyti skirtinga spalva. Šiame darbe bus nagrinėjami viršūnių spalvinimo algoritmai.

Algoritmai

Šiame darbe nagrinėjami viršūnių spalvinimo naudojant paieška su grįžimu ir penkių spalvų viršūnių spalvinimo algoritmai. Funkcija C(n) parodo, kiek blogiausiu atveju grafe iš n viršūnių viršūnėms reikės priskirti naują spalvą.

Paieška su grįžimu realizuota trijomis funkcijomis: algoritmas pradedamas iškviečiant funkciją *coloringWithBacktracking*, kuri inicializuoja masyvą *vcr*, skirtą viršūnių priskyrimams, ir iškviečia funkciją *colorB*. Funkcija *colorB* rekursyviai generuoja viršūnių spalvinimus tol, kol galiausiai gaunamas tinkamas spalvinimas ir jis grąžinamas kaip atsakymas. Ar viršūnės nuspalvinimas tinkamas patikrina funkcija *coloringPossible*, palygindama jį su kitomis nuspalvintomis viršūnėmis.

Paieška su grįžimu grąžina pirmą rastą spalvinimą iš ne daugiau kaip 5 spalvų. Tačiau jeigu testuojamas plokščias grafas, pagal keturių spalvų teoremą jam nudažyt užtenka keturių spalvų. Blogiausias atvejis bus, jeigu algoritmas turės perrinkti visas įmanomas viršūnių spalvinimo kombinacijas, kurių iš viso yra 5^n. Tačiau, jei toks spalvinimas egzistuoja, algoritmas sustos, kai bus rastas pirmas spalvinimas. Jeigu tas grafas plokščias, sprendinys tikrai nebus blogesnis už {1, 2, 3, 4, 4, ..., 4}., čia laikoma, kad galima gauti kitą sprendinį sukeitus spalvas vietomis, bet toks sprendinys bus rastas pirmas. Tokiu atveju algoritmui tektų perrinkti $\mathrm{C(n)} = (2 \cdot 5^{n-2} + 3 \cdot 5^{n-3} + 4 \cdot 5^{n-4}) + (5^{n-4} + 5^{n-5} + \ldots + 5)$ atvejų. Tada $C(n) = 5^{n-4} \cdot ((2 \cdot 5^2 + 3 \cdot 5 + 4) + (\frac{5}{4} - \frac{1}{4 \cdot 5^{n-5}}))$. Tada $C(n) \sim (50 + \frac{1}{4}) \cdot 5^{n-3}$. Vaikščiojimas tarp viršūnių yra tik dalis veiksmų, kurie įeina į žingsnių skaičių L(n). Tada žingsnių skaičius L(n) bus ne mažesnis už C(n).

Penkių viršūnių spalvinimo algoritmo realizacija: algoritmas pradedamas iškviečiant funkciją *fiveColorColoring*, kuri , kuri inicializuoja masyvą *coloring* (pradinės reikšmės 0), skirtą viršūnių priskyrimams, inicializuoja loginį masyvą *usedVertices* (pradinės reikšmės *true*), kuris fiksuoja ar viršūnės ištrintos ar ne, ir iškviečia funkciją *fiveColRec*. Funkcija *fiveColRec* paprastai nuspalvina viršūnes, jei jų yra ne daugiau, nei 5, kitu atveju rekursyviai kreipiasi į save, pašalindama viršūnę, kurios laipsnis ne didesnis nei 5. Šioje vietoje, kai algoritmas grįžta iš rekursijos kviečiama funkcija *setProperColor*. Joje bandoma nuspalvinti tikrinamą viršūnę spalva, kurios neturi nė viena kaimynė. Jeigu nėra tinkamos spalvos, viršūnei paliekama pradinė nulinė spalva (tada gaunamas dalinis beveik pilnas spalvinimas). Galiausiai viršūnė atgal grąžinama į grafą ir grįžtama iš rekursijos. Kaip rezultatas grąžinamas viršūnių spalvinimas masyve *coloring*.

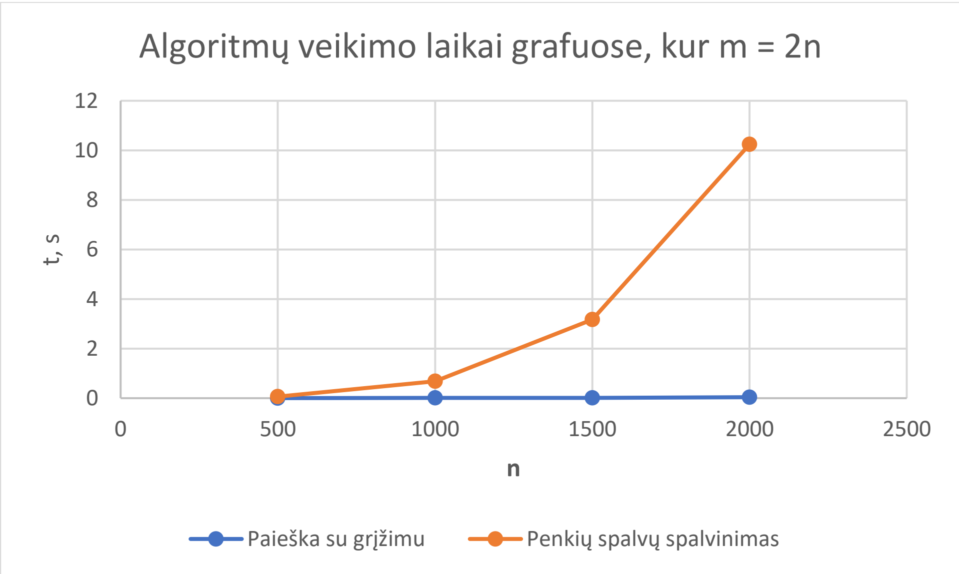
Penkių viršūnių spalvinimo algoritmo sudėtingumas: masyvų *coloring* ir *usedVertices* inicializacija reikalauja 2n žingsnių. Funkcija *fiveColorRec* rekursyviai iškviečiama n-5 kartus. Jei viršūnių ne daugiau 5 dalyje bus įvykdyta $n - 4 + 1 + 2n + 2 + n + 5 + 5 = 4n + 9$ žingsnių. Toliau einantis ciklas, ieškantis tinkamos viršūnės spalvinimui bus tikrinamas (n-5)(2n+2) kartus, o vykdomas (n-5)n kartų. Tikrinimas, ar viršūnė tinkama, užima 3 žingsnius. Jei taip, vienetu sumažinamas naudojamų viršūnių skaičius (1 žingsnis), masyve *usedVertices* nustatoma, kad viršūnė ištrinta, o vėliau grįžus iš rekursijos atstatoma (2 žingsniai), kviečiama funkcija *setProperColor*, kuri nustato viršūnei tinkamą spalvą (blogiausiu atveju ties paskutine viršūne susikerta ir neranda tinkamos spalvos – tada bus (2*6 + 5*(1 + (2n+2) + 5n + 2+1) = 35n + 42 žingsniai). Iš viso 2n + (4n+9) + (n-5)((2n+2) + n(1 + 2 + (35n+42))) = (6n+9) + (n-5)($35n^2$+47n+2) = (6n+9) + ($35n^3$ + $47n^2$ + 2n - $175n^2$ – 235n - 10) = $35n^3$ - $128n^2$ - 229n – 1 žingsnių, čia n ne mažesnis už 5. Tada $L(n) \sim 35 \cdot n^3$.

Eksperimentai

Konstruojant grafą iš n briaunų iš viso yra n(n-1)/2 galimų briaunų. Iš jų buvo renkamasi m atsitiktinių. Tačiau nebuvo užtikrinama, kad konstruojamas grafas bus planarus.

Gauti rezultatai:

1 pav. Algoritmų veikimo laikai grafuose, kur m = n



2 pav. Algoritmų veikimo laikai grafuose, kur m = 2n

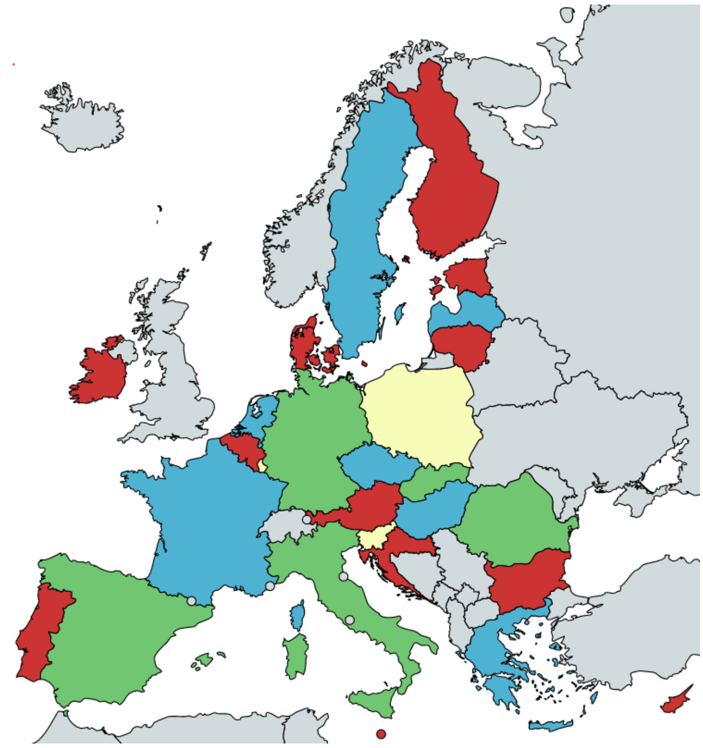
Iš 1 pav. ir 2 pav. matyti, kad retuose grafuose, kur briaunų skaičius lygus n arba 2n penkių spalvų spalvinimo algoritmas užtrunka ilgiau. Tą lemia jo kubinis sudėtingumas nuo laiko. Tuo tarpu paieškos su grįžimu algoritmas veikia palyginti greičiau, nes retuose grafuose dėl briaunų atsiradimo tenka padaryti mažai korekcijų, kad būtų surastas tinkamas spalvinimas.

3 pav. Algoritmų veikimo laikai grafuose su 100 viršūnių

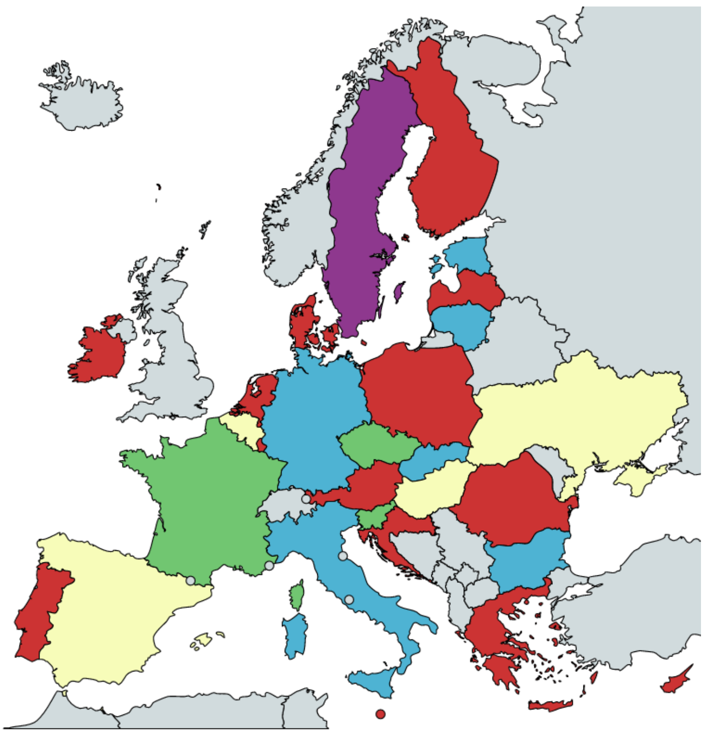
Iš 3 pav. matyti, kad didinant briaunų skaičių m nekeičiant viršūnių skaičiaus paieškos su grįžimu algoritmas pasiekia lūžio tašką, kada pradeda staigiai augti veikimo laikas. Tada jam tenka perrinkti gerokai daugiau variantų, kol randa tinkamą spalvinimą. Dar labiau didinant briaunų skaičių algoritmas iš vis nebaigdavo darbo, nes būdavo sugeneruojamas grafas, kuriam spalvinimas neegzistuoja, todėl paieška su grįžimu bandydavo perrinkti visus variantus, o jų yra iš viso 5^{100}.

Pavyzdys

Algoritmai ištestuoti ir su realiu pavyzdžiu: Europos Sąjungos žemėlapiu. Čia grafas buvo formuotas taip, kad viršūnes atitinka valstybės, o briaunos nubrėžtos tik tarp tų viršūnių, kurias atitinkančios valstybės tarpusavyje ribojasi. Pirmai spalvai priskirta raudona spalva, antrai – mėlyna, trečiai – žalia, ketvirtai – geltona, penktai – violetinė spalva.



4 pav.EU žemėlapio spalvinimas naudojant paiešką su grįžimu



5 pav. EU žemėlapio spalvinimas naudojant penkių spalvų spalvinimą

Išvados

Paieškos su grįžimu sudėtingumas eksponentinis, o penkių spalvų algoritmo sudėtingumas polinominis. Didinant briaunų skaičių grafe paieška su grįžimu pasiekia tašką, nuo kurio neįmanoma sulaukti algoritmo darbo pabaigos. Tačiau retuose grafuose greičiau užduotį atlieka paieškos su grįžimu algoritmas. Beje, penkių spalvų algoritmas ne visada randa visišką 5 spalvų spalvinimą (jeigu tikrinamos viršūnės visos penkios kaimynės yra skirtingų spalvų). Tam, kad spalvinimas būtų iki galo sutvarkytas galimos įvairios modifikacijos, pavyzdžiui, aptinkant ir naudojant Kempės grandines (angl. *Kempe chain*) ir surandant dvi viršūnes, kurios guli skirtinguose blokuose, padalytuose grandinės ir tikrinamos viršūnės.

Priedai

Visi failai, kodas ir programos paleidimo instrukcija yra sukelti į gitą: https://github.com/Aurimasjar/Vertex-Coloring