Univerzitet u Beogradu,

Elektrotehnički fakultet

Složenost algoritama i odabrane metode optimizacije

Seminarski rad

Problem triobojivosti grafa

Autori:

Anja Kovačević, 2018/0112

Aleksa Peškir, 2018/0131

Maj 2020.

Sadržaj

1. Uvod

Tema ovog rada je problem bojenja grafova, specijalno problem triobojivosti grafa. Upoznaćemo se sa matematičkom osnovom ovog problema, a posebnu pažnju ćemo posvetiti veoma važnim pojmovima: hromatskom broju i hromatskom polinomu grafa. Zatim ćemo detaljno predstaviti neke od algoritama koji se koriste za bojenje grafova i analiziraćemo njihovu složenost. Za kraj, daćemo kratak pregled mogućih primena ovog koncepta u raznim oblastima.

1.1. Uvod u teoriju grafova

Teorija grafova je oblast matematike koja se bavi proučavanjem grafova i njihovih osobina. O velikom značaju teorije grafova govori činjenica da su njena dostignuća našla primenu u širokom spektru oblasti: u analizi električnih kola, u opisu modela i struktura podataka u računarstvu, kod mobilnih mreža itd.

U narednom delu definisaćemo neke osnovne pojmove vezane za graf i na ove definicije oslonićemo se u daljoj analizi problema bojenja grafa.

Definicija 1.1. Graf G je uređeni par G=(V,E) koji se sastoji od konačnog nepraznog skupa čvorova V(G) i konačnog skupa grana E(G).

Napomena 1.1: Ukoliko se ne kaže drugačije, u daljem tekstu sa n ćemo označavati broj čvorova grafa, a sa m broj grana.

Definicija 1.2. Dva čvora u i w su **susedna** ako su povezana granom e (pišemo e = (u,w)). Čvorovi grane nazivaju se **krajevi**.

Definicija 1.3. Grana koja spaja čvor sa samim sobom naziva se **petlja**.

Definicija 1.4. Grane grafa su **paralelne** ukoliko povezuju iste čvorove, odnosno imaju zajedničku početnu i krajnju tačku.

Definicija 1.5. Graf koji nema ni petlje ni paralelne grane naziva se **prost** graf. **Multigraf** je graf kod kojeg se pojavljuju paralelne grane ili petlje.

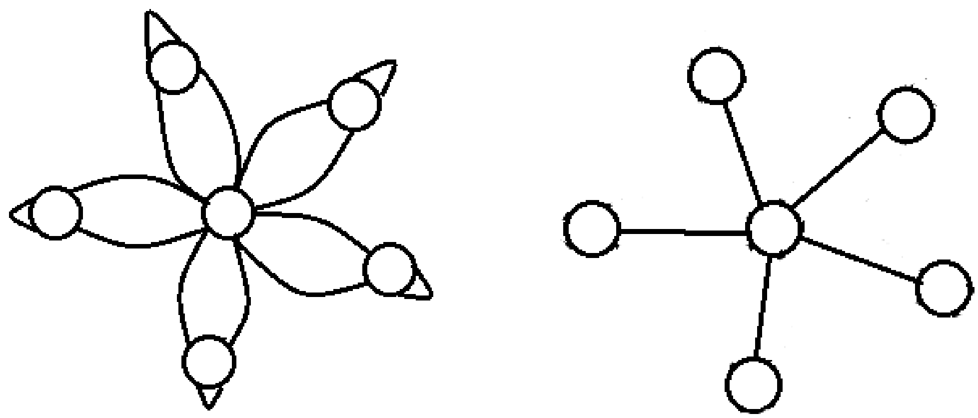
Definicija 1.6. Kondenzacija je postupak eliminacije paralelnih grana i petlji. Pri eliminaciji paralelnih grana zadržava se jedna grana između odgovarajućih čvorova.

Napomena 1.2: Kondenzacijom se od multigrafa dobija prost graf (*slika 1.1*).

Definicija 1.7. Graf H=(W,F) je **podgraf** grafa G=(V,E) ukoliko je W ⊆ V i F ⊆ E.

Napomena 1.3: Pošto je H takođe graf, grane iz skupa F imaju krajeve u skupu W.

Definicija 1.8. Stepen čvora *v* grafa G (u oznaci d(*v*)) predstavlja broj njemu susednih čvorova. Minimalan stepen čvora u grafu označava se sa δ(G), a maksimalan stepen čvora sa Δ(G).



Slika 1.1 – Mulitgraf (levo) i prost graf dobijen od njega kondenzacijom (desno)

Definicija 1.9. Put je niz čvorova grafa $\{v_1, v_2, \dots, v_{k-1}, v_k\}$ takav da za svaki čvor iz niza v_i, za i = 1, 2, … , k-1, postoji grana koja ga povezuje sa čvorom v_{i+1}.

Definicija 1.10. Graf G nazivamo **regularnim** grafom ako svaki čvor u grafu ima isti broj suseda, tj. važi δ(G) = Δ(G).

Definicija 1.11. Regularni grafovi čiji su svi čvorovi stepena 2 nazivaju se **konture**. Oznaka za konturu sa n čvorova je C_n.

Napomena 1.4: Očigledno je da je za postojanje konture neophodno da n bude veće od 3.

Definicija 1.12. Graf je **povezan** ako su svaka dva čvora povezana nizom čvorova i grana.

Definicija 1.13. Povezan graf koji ne sadrži nijednu konturu kao svoj podgraf naziva se **stablo**.

Napomena 1.5: Stablo od n čvorova sadrži tačno n – 1 granu (ako bi sadržalo manje grana, neki čvor bi ostao nepovezan sa ostatkom grafa, a ako bi sadržalo više, napravila bi se kontura). U ovom radu smatraćemo da ako tražimo stablo od nekog grafa G, ono mora sadržati sve čvorove tog grafa (u anglosaksonskoj literaturi razdvajaju se pojmovi *tree* i *spanning tree,* prvi ne zahteva korišćenje svih čvorova iz G, dok drugi zahteva).

Definicija 1.14. Kompletan graf K_n je graf sa n čvorova u kom su svaka dva čvora povezana granom.

Napomena 1.6: Ako graf nije povezan, onda se on može rastaviti na **komponente povezanosti**, disjunktne po čvorovima. Osobine povezanih grafova važe i za pojedinačne komponente povezanosti budući da su one povezani grafovi.

Definicija 1.15. Neka je dat graf G=(V,E). N**ezavisan skup čvorova** je skup čvorova grafa G u kojem ne postoje dva čvora koja su povezana.

Definicija 1.16. Mera nezavisnosti grafa G predstavlja maksimalnu veličinu nezavisnog skupa čvorova tog grafa. Oznaka mere nezavisnosti je α(G). Važi:

$$1 \leq \alpha(G) \leq n$$

gde je n broj čvorova grafa G.

Napomena 1.7: Za primer prostog grafa sa slike 1.1 mera nezavisnosti je 5 (svi čvorovi osim centralnog čvora pripadaju nezavisnom skupu čvorova).

Definicija 1.17. Graf je **planaran** ukoliko se može predstaviti u ravni tako da se nijedna grana ne seče sa drugom granom.

Napomena 1.8: Grafovi sa slike 1.1 jesu planarni grafovi.

U nastavku ćemo se detaljnije baviti planarnim grafovima koji će nam biti izuzetno važni za neke od algoritama za bojenje grafova.

Teorema 1.1. (Ojlerova formula) Neka je G povezan planaran graf sa n čvorova i m grana, predstavljen tako da mu se grane ne seku. Neka graf G deli ravan na r oblasti. Tada važi :

$$r = m – n + 2.$$

Dokaz. Dokazaćemo teoremu indukcijom po broju grana grafa. Baza indukcije je slučaj m = n – 1 što je minimalan broj grana koji povezuje n čvorova. Tada je G stablo, pa ne sadrži nijednu konturu, tj. broj oblasti je 1. Zaista, važi 1 = (n – 1) – n + 2. Pretpostavimo da G ima više od n – 1 grane, tj. da sadrži bar neku konturu. Eliminišemo jednu granu iz grafa formirajući graf G’ koji je povezan, ima n čvorova, m – 1 granu i deli ravan na r – 1 oblasti. Postavimo indukcionu hipotezu: za graf G’ važi Ojlerova formula, tj. važi r – 1 = (m – 1) – n + 2. Lako je pokazati da tada Ojlerova formula važi i za graf G: dodavanjem 1 sa obe strane jednakosti dobijamo r = m – n + 2, što je i trebalo pokazati.

Lema 1.1. Neka je G prost, povezan i planaran graf sa $n \geq 3$ čvorova i m grana, predstavljen tako da mu se grane ne seku. Neka graf G deli ravan na r oblasti. Tada važi:

$$m \leq 3n - 6.$$

Dokaz. Neka je f_i broj grana koje okružuju oblast r_i (ako postoji grana oko koje se sa obe strane nalazi ista oblast, takvu granu brojimo dva puta). Tada je $\sum_{i=1}^{r} f_i = 2m$ pošto smo svaku granu brojali dva puta. Kako je G prost graf i $n \geq 3$, svaka oblast je ograničena sa najmanje 3 grane. Koristeći Ojlerovu formulu i množeći je sa 3 sa obe strane jednakosti dobijamo: $3r = 3m - 3n + 6$, a pošto je $f_i \geq 3$ važi: $3r \leq \sum_{i=1}^{r} f_i = 2m$. Kombinujući ova dva izraza dobijamo $3m - 3n + 6 \leq 2m$, tj. posle sređivanja dobijamo oblik dat u iskazu teoreme.

Napomena 1.9: Pomoću ove leme možemo pokazati da graf K_5 nije planaran. Naime, graf K_5 ima $n = 5$ čvorova i $m = 10$ grana, pa pošto je $10 \geq 15 - 6$, po uslovu leme 1.1 ovakav graf nije planaran.

Lema 1.2. Svaki planaran graf G ima bar jedan čvor stepena ≤ 5.

Dokaz. Možemo pretpostaviti da je G povezan graf (ukoliko nije, možemo pojedinačno posmatrati njegove komponente povezanosti). Pretpostavimo da je $d(v_i) > 5$ za svaki čvor v_i grafa G. Tada je $2m = \sum_{i=1}^{n} d(v_i) \geq 6n$. Po lemi 1.1, $3n - 6 \geq m$, pa množenjem sa 2 dobijamo $6n-12 \geq 2m$. Sledi: $6n \leq 2m \leq 6n-12$, što je kontradikcija.

2. Uvod u bojenje grafova

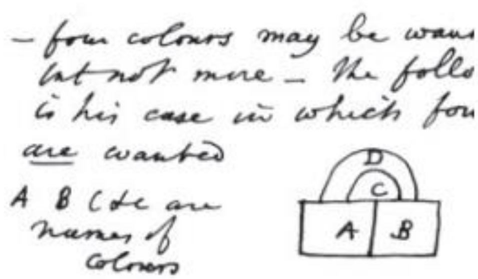
U ovom delu obratićemo pažnju na istorijski kontekst bojenja grafova, a zatim ćemo dati kratak pregled pojmova koji će biti ključni za dalje razmatranje u odeljcima 3. i 4. koji se bave važnim teoremama i algoritmima za bojenje grafova.

2.1. Istorijat

Problem bojenja grafova nastao je u 19. veku, kada je matematičar Fransis Gatri[1] 1852. godine primetio da su potrebne samo četiri boje da se oboji karta sveta [1]. Gatri je uputio pismo profesoru Londonskog univerziteta, Avgustu De Morganu[2], u nadi da će dobiti povratnu informaciju o tome koji je uzrok te činjenice. Pošto profesor de Morgan nije znao odgovor,

[1] Francis Guthrie (1831 – 1899), južnoafrički matematičar i botaničar
[2] Augustus De Morgan (1806 – 1871), engleski matematičar

pitanje je dalje prosleđeno u Dablin, profesoru Vilijamu Hamiltonu[3], međutim, on nije bio zainteresovan. Formulisan je takozvani problem četiri boje: *da li se svaka geografska karta može obojiti sa najviše četiri boje tako da susedne države ne budu obojene istom bojom?* Engleski matematičar Hejli[4] je predložio taj problem Londonskom matematičkom društvu 1878. godine. Godinu dana kasnije, Alfred B. Kempe[5] je pokušao da dokaže ovu teoremu, ali mu je pronađena greška u dokazu. Međutim, algoritam za bojenje koji je osmislio u svom dokazu i dan danas se koristi u različitim softverima i kompajlerima (ovaj algoritam biće detaljno razmatran u poglavlju 4)[2].



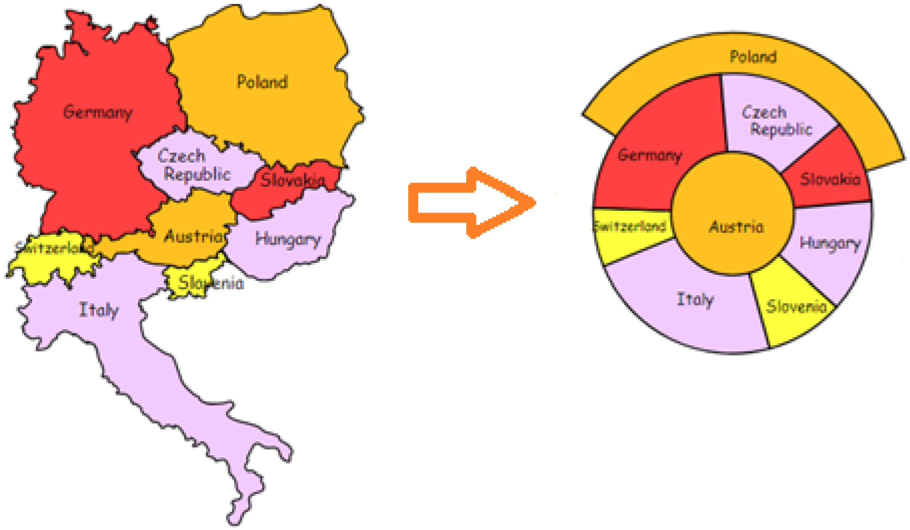
Slika 2.1 – Deo De Morganovog pisma Vilijamu Hamiltonu

Matematičar Persi Hivud[6], koji je pronašao grešku u Kempeovom dokazu, koristeći Kempeov algoritam rešio je problem 5-obojivosti.

Pošto se svaka karta može nacrtati kao planaran graf, problem bojenja karte može se svesti na problem bojenja čvorova planarnog grafa tako što se svaka regija na karti predstavi sa po jednim čvorom grafa, pri čemu su dva čvora povezana ako i samo ako su regije koje su njima predstavljene susedne.

Kenet Epl[7] i Volfgang Haken[8] su 1977. godine koristeći Kempeove i Hivudove teoreme kao osnovu konačno dokazali teoremu o četiri boje [11].

[3] Sir William Rowan Hamilton (1805 – 1865), irski matematičar, fizičar i astronom
[4] Arthur Cayley (1821 – 1895), engleski matematičar
[5] Alfred B. Kempe (1849 – 1922), engleski matematičar
[6] Percy John Heawood (1861 – 1955), engleski matematičar
[7] Kenneth Ira Appel (1932 – 2013), američki matematičar
[8] Wolfgang Haken (1928 -), američki matematičar



Slika 2.2 - Predstavljanje dela Evrope kao planarni graf

U narednom delu bavićemo se definisanjem nekih važnih pojmova vezanih za bojenje grafova.

2.2. Kratak pregled pojmova

Definicija 2.1. ***Obojen graf*** je graf kod kojeg je svakom čvoru dodeljena boja. Graf je ***pravilno obojen*** ako su svaka dva susedna čvora grafa različite boje.

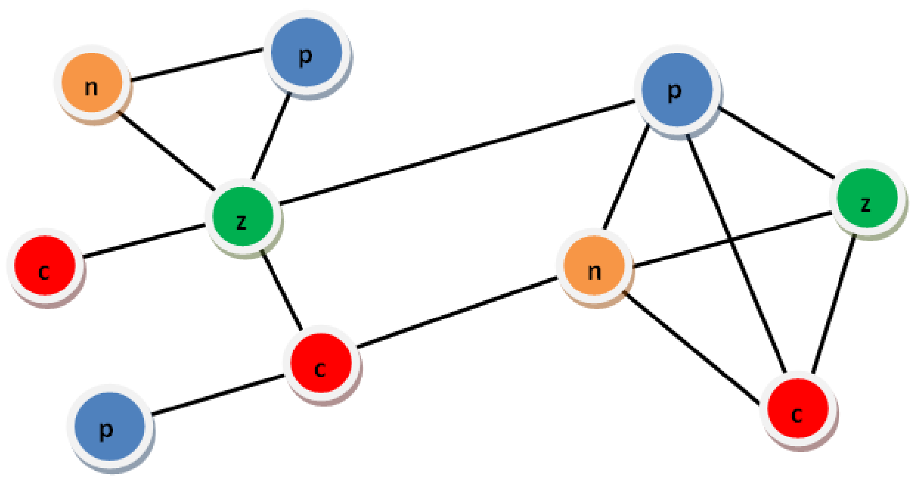
Napomena 2.1: Dodeljivanje „boje“ čvoru grafa predstavlja samo njegovo označavanje, gde oznaka ne mora stvarno biti boja već na primer prirodan broj ili slovo abecede. Koncept bojenja je tu da bi se sam problem mogao lakše vizualizovati.

Napomena 2.2: Nepovezani grafovi se mogu obojiti tako što se pojedinačno oboje sve njegove komponente povezanosti. Zbog toga ćemo nadalje razmatrati samo grafove koji su povezani i prosti.

Napomena 2.3: Treba napomenuti da pravilno obojen graf može biti potpuno ili delimično obojen. Potpuno i pravilno obojen graf podrazumeva da je svakom čvoru dodeljena odgovarajuća boja, dok kod delimično ali pravilno obojenog grafa postoji bar jedan čvor kojem nije dodeljena nijedna boja (ali su zato svi drugi čvorovi pravilno obojeni).

Napomena 2.4: U nastavku ćemo, kao što se to obično čini kada se ne pominju drugi načini bojenja, pod pojmom obojen graf smatrati pravilno obojen graf.

Jedan primer pravilno obojenog grafa dat je na slici 2.3. Kao oznake su izabrane **p**lava, **c**rvena, **n**arandžasta i **z**elena.



Slika 2.3 – Pravilno obojen graf

Napomena 2.5: Pojam nezavisnog skupa čvorova je veoma blizak konceptu bojenja grafova. Naime, ako je graf pravilno obojen, svaki čvor tog grafa obojen istom bojom pripada nezavisnom skupu čvorova. Skup čvorova koji su obojeni istom bojom nadalje ćemo zvati *klasom boje*.

Definicija 2.2. ***Hromatski broj*** grafa G je najmanji broj boja koji je potreban da bi se graf pravilno obojio. Oznaka hromatskog broja je $\chi(G)$.

*Napomena 2.6***:** Nije teško uočiti da za hromatski broj mora da važi nejednakost veoma slična kao u razmatranju mere nezavisnosti**:**

$$1 \leq \chi(G) \leq n$$

Kao primer grafa čiji je hromatski broj jednak 1 navodimo trivijalan primer grafa sa n čvorova koji su međusobno nepovezani, pa svaki od njih možemo obojiti istom bojom. Primetimo da je za ovakav graf mera nezavisnosti jednaka n. Nasuprot tome, kompletan graf K_n ima hromatski broj n i meru nezavisnosti jednaku 1.

Napomena 2.7: Možemo uspostaviti još jednu paralelu između hromatskog broja i mere nezavisnosti grafa. Naime, dok nam je veličina od interesa za određivanje hromatskog broja *minimalan* broj boja koje možemo koristiti, za meru nezavisnosti grafa to je *maksimalna* veličina nezavisnog skupa čvorova tog grafa.

Posmatrajući graf sa slike 2.3, iako odmah na prvi pogled možemo zaključiti da je on pravilno obojen, nije tako lako zaključiti koji je njegov hromatski broj. Obojili smo ga koristeći 4 boje, međutim, ne možemo odmah reći da je njegov hromatski broj jednak 4, jer ga je možda moguće obojiti sa manje boja.

Teoreme date u narednom odeljku pomoći će nam da odgovorimo na pitanje koliko je najmanje boja moguće koristiti za bojenje datog grafa, tj. koliki je njegov hromatski broj.

Još jedno pitanje od interesa je na koliko načina možemo obojiti neki graf G korišćenjem k boja. Naravno, ako je k manje od hromatskog broja grafa, očigledno je da je broj načina jednak 0. Ovaj problem možemo posmatrati i kao traženje funkcije $P_G(k)$ koja za eksplicitno uneseno k daje broj načina da se graf pravilno oboji korišćenjem k boja. Takva funkcija se naziva **hromatski polinom**. Osobine hromatskog polinoma ćemo razmotriti u narednom odeljku.

3. Teoreme o bojenju grafova

Ovaj odeljak započećemo jednom jednostavnom, ali značajnom teoremom koja daje odgovor na pitanje *kakva je veza između hromatskog broja grafa i hromatskog broja njegovog podgrafa*.

Teorema 3.1. Ako je H podgraf grafa G, onda $\chi(H) \leq \chi(G)$.

Dokaz. Bilo kakvo pravilno bojenje grafa G obezbeđuje pravilno bojenje i podgrafa H, tako što se jednostavno dodele iste boje čvorovima H kao što je to učinjeno kod grafa G. To znači da je za bojenje podgrafa H potrebno najviše $\chi(G)$ boja ili manje, što teorema i tvrdi.

Napomena 3.1: Ova teorema je često od interesa i u "obrnutom" poretku. Na primer, ako u okviru grafa G uočimo podgraf H oblika K_m, onda je $\chi(H) = m$, pa je po teoremi 3.1 $\chi(G) \geq m$. Ovo razmatranje je veoma važno, jer može da nam da donju granicu mogućeg hromatskog broja grafa, pa nam, u kombinaciji sa teoremama 3.3 – 3.5 koje govore o gornjoj granici, značajno sužava opseg za traženje hromatskog broja.

U *Napomeni 2.5* pomenuta je jedna veoma važna veza između nezavisnog skupa čvorova i bojenja grafa koja se koristi u dokazu naredne teoreme.

Teorema 3.2. Za graf G sa n čvorova važi nejednakost: $\frac{n}{\alpha} \leq \chi$, gde je χ hromatski broj, a α mera nezavisnosti grafa G.

Dokaz. Pretpostavimo da je graf G obojen sa χ različitih boja. Pošto je svaka klasa boje nezavisna, broj čvorova koji joj pripadaju nije veći od veličine α. Označimo klase boja sa V_1, V_2, V_3, ... V_χ . Tada važi:

$$n = \sum_{i=1}^{\chi} |Vi| \leq \chi \cdot \alpha$$

Što je i trebalo dokazati.

Sledeća tri veoma značajne teoreme govore o gornjem ograničenju hromatskog broja grafa G.

Teorema 3.3. Za svaki graf G važi $\chi(G) \leq \Delta + 1$, gde je Δ najveći stepen čvora grafa G.

Dokaz. Izaberimo proizvoljan čvor grafa G i obojimo ga sa jednom od Δ +1 boja. Zatim, izaberimo neki od neobojenih čvorova i obojimo ga bojom različitom od boja njemu susednih čvorova. Ponavljamo postupak sve dok u potpunosti ne obojimo graf. Pošto je bilo koji čvor povezan sa najviše Δ čvorova, najviše Δ različitih boja je iskorišćeno za bojenje njemu susednih čvorova, pa će za bojenje takvog čvora uvek biti dostupna bar jedna boja, čime je dokaz završen.

Napomena 3.2: Za graf K_n smo već rekli da ima hromatski broj jednak n, a najveći stepen čvora ovog grafa je $\Delta = n - 1$. Primetimo da za K_n važi $\chi(K_n) = \Delta + 1$, što je gornja granica iz prethodne teoreme. Još jedan takav primer je i graf C_{2n+1} koji predstavlja konturu sa neparnim brojem čvorova. Najveći stepen čvora ovog grafa je 2, ali je njegov hromatski broj jednak 3 (proizvoljno izaberemo prvi čvor i obojimo ga jednom bojom, njemu susedni čvor moraćemo obojiti drugom bojom itd, naizmeničnim menjanjem boja uspećemo da obojimo sve čvorove osim poslednjeg koji će biti okružen prvim i pretposlednjim čvorom obojenim različitim bojama, pa za njegovo bojenje moramo iskoristiti treću boju). Primetimo da su oba ova grafa regularna. Štaviše, ispostavlja se da su ova dva grafa jedina za koje važi jednakost u gornjoj teoremi. Hromatski brojevi svih ostalih grafova iznose najviše Δ što tvrdi i teorema 3.5 (Bruksova teorema).

Teorema 3.4. Ako graf G nije regularan, $\chi \leq \Delta$.

Teorema 3.5. (Bruksova teorema) Ako graf G nije graf K_n i C_{2n+1} **,** $\chi \leq \Delta$.

Dokaz teoreme 3.4 zahteva poznavanje algoritma za bojenje grafova, pa ćemo se zbog toga na njega vratiti u odeljku 4. Taj dokaz je ujedno i deo dokaza Bruksove teoreme koju u ovom radu nećemo u potpunosti dokazivati zbog dužine dokaza. Detaljan dokaz Bruksove teoreme može se pogledati u [6].

Sada ćemo posvetiti malo više pažnje funkciji $P_G(k)$ pomenutoj u prethodnom odeljku. Pogledajmo za početak dva uvodna primera.

Primer 3.1. Neka graf G ima n čvorova i nijednu granu, tj. svi čvorovi su međusobno nepovezani. Na koliko načina možemo obojiti ovakav graf korišćenjem k boja? Hromatski broj ovakvog grafa je 1 kao što smo to zaključili u *Napomeni 2.6.* Ako koristimo samo jednu boju za bojenje grafa, broj načina je očigledno 1. Dakle, $P_G(1) = 1$. Međutim, šta se dešava ako za k odaberemo broj različit od 1? Potrebno je izvesti opšti izraz za hromatski polinom $P_G(k)$. Odaberimo proizvoljan čvor grafa. Njega je moguće obojiti na k načina. Zatim proizvoljno odaberemo naredni čvor. Njega takođe možemo obojiti na k načina, jer nije povezan sa prvim čvorom. Ako nastavimo ovakav postupak sve do n-tog čvora, zaključićemo da za svaki od n čvorova imamo k različitih načina da ga obojimo, tj. tražena funkcija je $P_G(k) = k^n$.

Primer 3.2. Posmatramo graf G koji je oblika K_n. Za kompletan graf važi $\chi(G) = n$. Očigledno za $k < n$, $P_G(k) = 0$. Kao u prethodnom primeru, prvoizabrani čvor možemo obojiti na k načina. Međutim, već za drugi čvor imamo smanjenu mogućnost izbora boja, tj. možemo izabrati k-1 boju, jer je jedna boja već iskorišćena na njemu susedan prvi čvor. Nadalje, za svaki novoizabrani čvor imamo po jedan manje način za bojenje. Svih n čvorova, dakle, bojimo na $P_G(k) = k(k-1)(k-2)...(k-n+1)$ načina.

Možemo uočiti da je u oba primera kao rezultat dobijen polinom po promenljivoj k. Zbog toga se $P_G(k)$ i naziva hromatskim polinomom. Međutim, da li je baš za svaki graf G $P_G(k)$ polinomska funkcija i da li možemo unapred odrediti stepen takvog polinoma bez poznavanja tačnog oblika cele funkcije? O tome govori naredna teorema.

Teorema 3.6. Za sve grafove G sa n čvorova, P_G je polinom stepena n.

Dokaz. Dokaz ćemo izvršiti matematičkom indukcijom po broju grana grafa G. Kao bazu indukcije iskoristićemo primer 3.1 gde smo pokazali da je hromatski polinom grafa sa n čvorova i nijednom granom $P_G(k) = k^n$ što jeste polinom stepena n. Uočimo dva susedna čvora u grafu G i obeležimo ih sa *v* i *w*, a granu koja ih spaja sa *e*. Sa G – e označićemo graf G kojem je oduzeta grana *e*. Na koliko načina njega možemo obojiti? Svakako ga možemo obojiti svim kombinacijama kao i G i još dodatnim kombinacijama koje podrazumevaju da uklanjanjem grane *e* čvorovi *v* i *w* mogu da se oboje istom bojom. Koliko je takvih dodatnih kombinacija? Pođimo ponovo od grafa G i zamenimo mu čvorove *v* i *w* jednim čvorom (označićemo ga sa *x*) pri čemu ćemo zadržati sve grane ka ostalim susednim čvorovima (ako su i *v* i *w* bili povezani sa nekim čvorom *u*, *x* će biti povezan samo jednom granom sa *u*, jer se bavimo samo prostim grafovima). Ovako dobijen graf dobiće oznaku G/e. Broj načina da obojimo graf G/e upravo je jednak traženom dodatnom broju kombinacija. Zbog čega je to tako? Bojenje čvora *x* nekom od k boja je u stvari ekvivalentno bojenju čvorova *v* i *w* istom tom bojom (zbog očuvanja grana sa susednim čvorovima u G/e u ostatku grafa se ništa ne menja). Dakle, važi:

$$P_G = P_{G-e} – P_{G/e}$$

Postavićemo indukcionu hipotezu: neka tvrđenje važi za grafove G – e i G/e koji imaju manje grana od grafa G. P_{G-e} je tada polinom stepena n, a $P_{G/e}$ je polinom stepena n – 1 (uklonili smo 2 čvora, a dodali smo 1, pa je broj čvorova u G/e jednak n – 1). Pokažimo sada da je P_G polinom stepena n: pošto važi $P_G = P_{G-e} – P_{G/e}$, a hromatski polinomi grafova G – e i G/e su stepena n i n – 1 redom, sledi da je stepen polinoma P_G jednak n, što je i trebalo pokazati.

Napomena 3.3: Množeći članove u zagradama u primeru 3.2. dobijamo polinom n-tog stepena, što je u skladu sa teoremom 3.6.

Za kraj ćemo, bez dokazivanja, navesti par zanimljivih svojstava hromatskog polinoma.

Svojstvo 3.1. Koeficijent uz član k^n hromatskog polinoma jednak je 1.

Svojstvo 3.2. Slobodan član hromatskog polinoma jednak je 0.

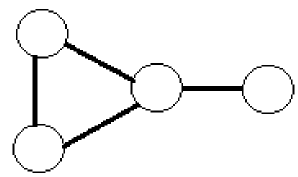
Svojstvo 3.3. Koeficijent uz član k hromatskog polinoma grafa G ima nenultu vrednost ako i samo ako je graf povezan.

Svojstvo 3.4. Neka je graf G nepovezan graf sa komponentama povezanosti C_1, C_2, ... C_m. Tada je P_G jednak proizvodu svih P_{Ci}, i = 1, 2, ..., m.

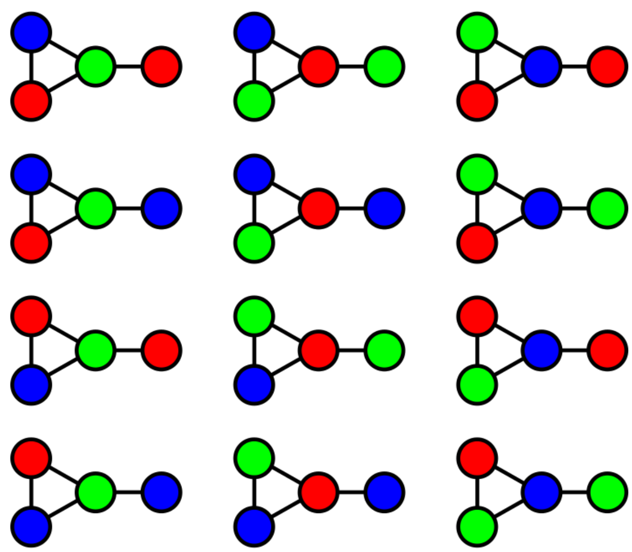
Napomena 3.4: Ovo svojstvo možemo intuitivno dokazati. Komponente grafa G su međusobno nepovezane, pa su nezavisne jedna od druge, te ih nezavisno možemo i bojiti. Prvu komponentu bojimo na $P_{C1}(k)$ načina, drugu na $P_{C2}(k)$ načina,..., m-tu na $P_{Cm}(k)$ načina, pa po principu proizvoda množenjem broja načina bojenja pojedinačnih komponenata dobijamo broj načina bojenja grafa G.

4. Algoritmi za bojenje grafova

Posmatrajmo graf sa slike 4.1 i pokušajmo da ga obojimo koristeći tri boje (crvenu, plavu i zelenu). Graf je planaran i ima 4 čvora, a najveći stepen čvora je 3. Ako za taj postupak ne koristimo nikakav poseban algoritam, nego nasumično bojimo čvorove, ipak ćemo relativno lako doći do rešenja. Od 81 (3^4) mogućih kombinacija bojenja, 12 će dati pravilno obojen graf (slika 4.2).



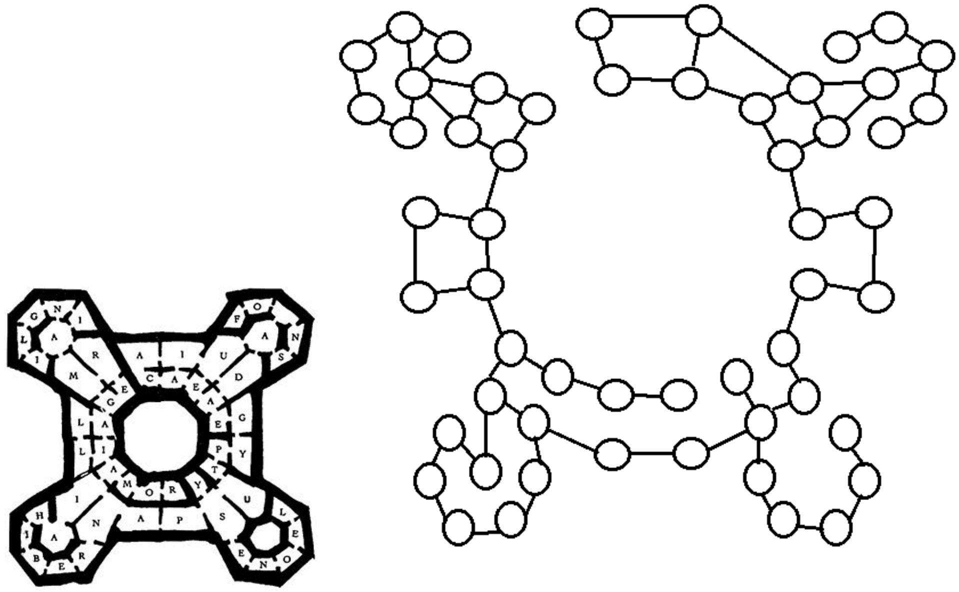
Slika 4.1 – Primer grafa



Slika 4.2 – Pravilno obojen graf sa slike 4.1

Međutim, ako pokušamo da obojimo neki znatno komplikovaniji graf sa mnogo više čvorova (na primer graf sa slike 4.3), videćemo da to nije tako lako bez korišćenja nekog od algoritama kojima ćemo se baviti u ovom poglavlju.

Graf sa slike 4.3 nastao je iz plana čuvene biblioteke iz *Imena Ruže* Umberta Eka. Sobe su predstavljene čvorovima, a prolazi između njih predstavljaju odgovarajuće grane grafa.



Slika 4.3 – Plan Biblioteke i graf nastao po ugledu na njega

Ovaj graf ima 55 čvorova i samim tim ukupan broj kombinacija bojenja je 3^{55} tj. $1.7444921 \cdot 10^{26}$. Nije teško zaključiti da postupak nasumičnog bojenja, a zatim odbacivanja nepravilnih rešenja nije baš efikasan. Zbog toga, u narednom delu, predstavićemo neke od najčešće korišćenih algoritama za bojenje grafova.

4.1. Kempeov algoritam

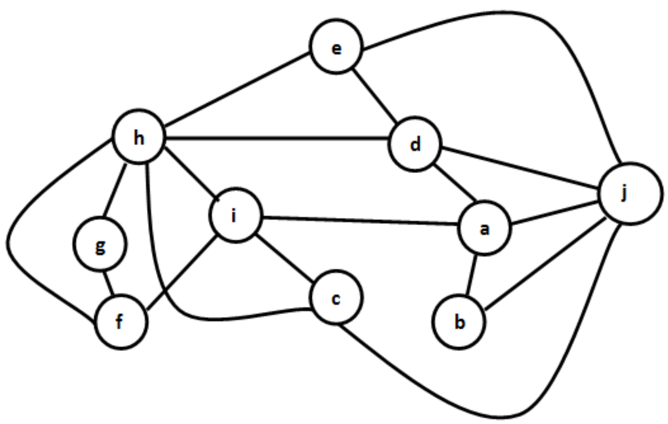
Alfred B. Kempe osmislio je jedan od najednostavnijih algoritama za bojenje grafa. U početku je za bojenje koristio 6 boja, zatim 5, a kasnije je njegov algoritam uopšten za k boja. Predstavićemo originalni Kempeov algoritam za bojenje sa 6 boja, a zatim ćemo ga uopštiti i koristiti za bojenje sa 3 boje.

Postupak bojenja po Kempeu:

1) Svaki planaran graf ima bar jedan čvor stepena ≤ 5. Otkloniti takav čvor.
2) Obojiti ostatak grafa rekurzivnim pozivom Kempeovog algoritma.
3) Vratiti čvor.
4) Vraćeni čvor je susedan sa najviše 5 čvorova za čije bojenje je iskorišćeno najviše 5 boja iz palete. Obojiti vraćeni čvor jednom od neiskorišćenih boja.

U opštem slučaju, kada koristimo k boja za bojenje, tražimo čvorove stepena < k. Demonstriraćemo upotrebu algoritma na primeru grafa sa slike 4.4. Naš algoritam sastojaće se iz dve faze (u nastavku ćemo ga zvati *dvofazni algoritam*):

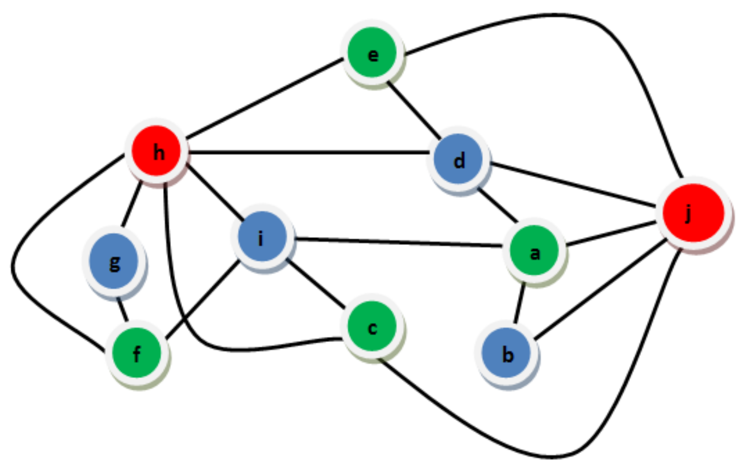
1) Uklonićemo sve čvorove grafa pri čemu ćemo zapamtiti redosled uklanjanja.
2) Obojićemo čvorove grafa po redosledu obrnutom od redosleda uklanjanja.



Slika 4.4 – Primer grafa

Graf sa slike 4.4 ima 10 čvorova (označili smo ih slovima od *a* do *j*). Izvršimo analizu stepena čvorova: čvorovi *b* i *g* su stepena 2, *c, f* i *e* su stepena 3, *j* je stepena 5, *h* je stepena 6, dok su *a, d* i *i* stepena 4. Prema Kempeovom algoritmu, izbacićemo sve čvorove stepena manjeg od 3, što su *b* i *g* (usvojićemo baš takav raspored izbacivanja). Primetimo da izbacivanjem čvorova *b* i *g* smanjujemo stepene čvorova *a, j, f* i *h*. Čvor *f* je sada stepena 2, pa i njega možemo ukloniti iz grafa (nizu izbačenih čvorova *b, g* dodajemo i ***f***). Došli smo u situaciju da više nijedan čvor nije stepena manjeg od 3. U tom slučaju, izbacićemo iz grafa proizvoljan čvor, izabraćemo na primer čvor *i* (niz: *b, g, f,* ***i***). Izbacivanjem čvora *i*, čvor *c* postaje stepena 2, pa ćemo izbaciti i njega (niz: *b, g, f, i,* ***c***). Izbacivanjem čvora *c*, čvor *h* postaje stepena 2, pa se i on izbacuje (niz: *b, g, f, i, c,* ***h***). Uklanjanjem *h*, čvor *e* postaje stepena 2, pa i njega dodajemo u niz izbačenih čvorova: *b, g, f, i, c, h,* ***e***. Ostaju čvorovi *d, a* i *j* koji su svi sada stepena 2 (mogu se izbacivati proizvoljnim redosledom) tako da konačan niz čvorova glasi: *b, g, f, i, c, h, e,* ***d, a, j***. Ovim postupkom završili smo prvi korak algoritma i prelazimo na drugi.

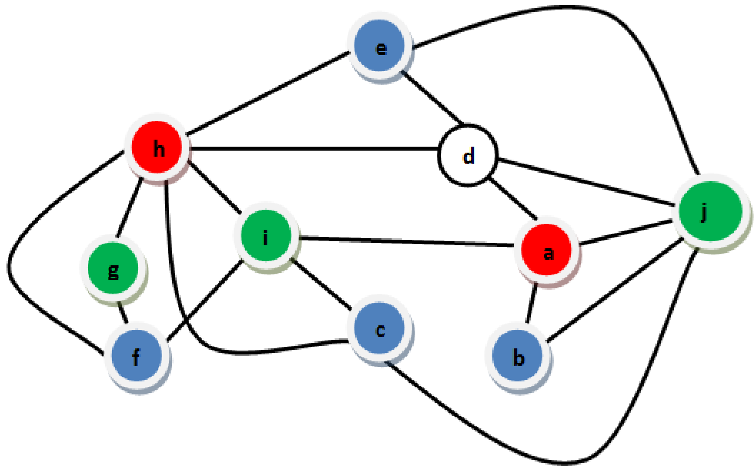
U drugom koraku algoritma, posmatramo dobijeni niz izbačenih čvorova i na osnovu njega formiramo novi, tako što obrnemo redosled članova niza, tj. novi niz glasi: *j, a, d, e, h, c, i, f, g, b*. Za bojenje grafa koristićemo kao i u prvom primeru crvenu, zelenu i plavu. Prvi čvor koji bojimo je čvor *j*. Za njega biramo proizvoljnu boju, na primer crvenu. Zatim, čvoru *a* dodeljujemo boju koja se razlikuje od njemu susednog čvora *j*, na primer zelenu. Ostale čvorove takođe bojimo tako da se njihova boja razlikuje od boja njima susednih čvorova, pri tom prateći redosled čvorova koje bojimo po nizu koji smo prethodno formirali. Na taj način, čvoru *d* moramo dodeliti plavu, čvoru *e* zelenu, a čvoru *h* crvenu boju. Za čvor c imamo dve mogućnosti, budući da ga zasad okružuju samo crveni čvorovi. Obojićemo ga u zeleno. Dolazimo do čvora *i* koji je jedini bio stepena 3 kada smo ga u prvom koraku izbacivali iz grafa i baš zbog toga nije zagarantovano da ćemo moći da ga obojimo, međutim, u ovom slučaju možemo, jer je okružen sa dva zelena i jednim crvenim čvorom, pa ga možemo obojiti u plavo. Dalje, po istom principu kao i ranije, bojimo čvor *f* u zeleno, čvor *g* u plavo i čvor *b* u plavo, sa čim završavamo bojenje grafa i zaključujemo da je ono pravilno (slika 4.5).



Slika 4.5 – Pravilno obojen graf sa slike 4.4

Diskusija: Šta ako bismo umesto korišćenja Kempeovog algoritma za određivanje redosleda bojenja čvorova koristili neki proizvoljan redosled?

Na primer, uzmimo kao raspored čvorova sledeći niz *a, j, h, g, i, c, e, b, f, d* (dok smo korišćenjem Kempeovog algoritma dobili niz: *j, a, d, e, h, c, i, f, g, b*). Pokušajmo da obojimo isti graf kao u prethodnom primeru, samo sa novim redosledom bojenja čvorova. Čvor *a* bojimo crvenom bojom, čvor *j* zelenom, čvor *h* crvenom, čvor *g* zelenom, čvor *i* zelenom, čvor *c* plavom, čvor *e* plavom, čvor *b* plavom, čvor *f* plavom. Dolazimo do čvora d koji ne možemo obojiti nijednom bojom, jer je povezan sa čvorovima svih raspoloživih boja (slika 4.6). Za ovako obojen graf kažemo da je pravilno delimično obojen.



Slika 4.6 – Graf obojen sa izmenjenim prvim korakom algoritma

Zaključujemo da Kempeov algoritam daje bolje rezultate u bojenju grafa nego uzimanje nasumičnog redosleda bojenja čvorova. Zbog čega je to tako? Uočimo šta u stvari radimo kada primenjujemo Kempeov algoritam. Uklanjanjem čvorova stepena manjeg od 3 (primetimo da nam takvi čvorovi nisu ni problematični za bojenje jer za njih uvek možemo izabrati boju), smanjujemo stepen njima susednih čvorova. Tada se nekim od susednih čvorova stepen spusti ispod 3, pa ih po Kempeu možemo ukloniti, time smanjujući stepen njima susednim čvorovima itd. Primetimo da iz ovog razloga između čvorova postoji veza u smislu da jedan čvor ne možemo izbaciti iz grafa dok ne izbacimo drugi. Nasuprot tome, kada nasumično biramo redosled čvorova, takva veza između čvorova ne postoji. Možemo reći da Kempeov algoritam, u stvari, daje prednost u bojenju čvorovima većeg stepena. Zbog čega je to važno pokažimo na prethodnom primeru grafa sa slike 4.4. Posmatrajmo broj čvorova kod kojih nam izbor boje nije bio ograničen na jednu, kod prvog i kod drugog algoritma. Svakako, boje prva dva čvora se biraju nasumično (za prvi čvor možemo izabrati bilo koju od 3 boje, dok za drugi biramo od 2 ili 3 boje u zavisnosti od povezanosti sa prvim čvorom) i u jednom i u drugom slučaju. Međutim pored toga, u prvom slučaju jedinu „nedoumicu“ imali smo kod čvora *c* kojeg smo izbacili neposredno posle čvora *i* kod kojeg je zadržan stepen 3 (da smo za njega izabrali plavu umesto zelene, ne bismo mogli da obojimo čvor *i*). U drugom slučaju, takvih „nedoumica“ bilo je mnogo više, na primer čvor *h* je imao mogućnost bojenja sa sve 3 boje, zatim čvor *g* mogao je da se oboji sa 2 boje itd. Povećanjem broja „nedoumica“, povećava se i verovatnoća da graf bude delimično obojen umesto potpuno, samim tim, više puta moramo ponoviti ceo postupak ispočetka da bismo došli do pravilno potpuno obojenog grafa. Za kraj razmatranja ovog algoritma, daćemo još par napomena.

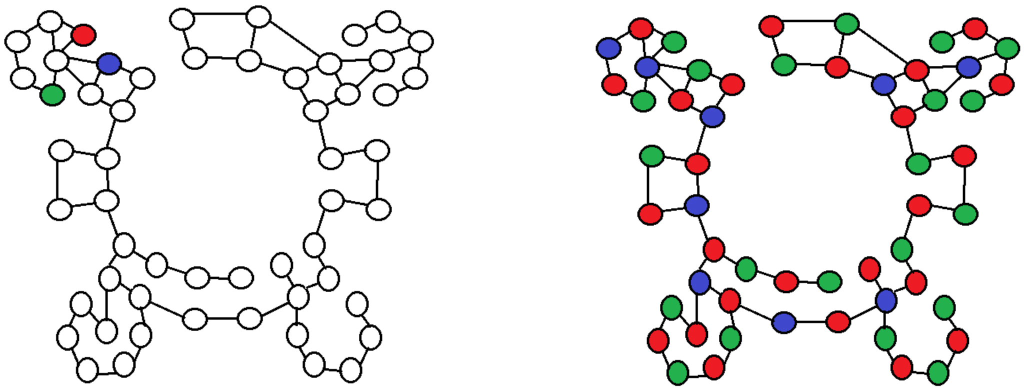
Napomena 4.1: Iako Kempeov algoritam daje raspored po kojem bojimo čvorove, on ne kaže ništa o biranju konkretne boje za čvor (osim naravno činjenice da ne smemo birati boju koja je već iskorišćena kod susednog čvora) tj. ne daje prednost nijednoj boji kada imamo nedoumicu u bojenju. To smo videli kada smo razmatrali bojenje grafa sa slike 4.4. Naime, čvor *c* smo mogli obojiti i zelenom i plavom bojom od kojih je samo jedna boja dala korektno rešenje – zelena, ali Kempeov algoritam ni po čemu nije rekao zašto treba izabrati zelenu, a ne plavu boju.

Napomena 4.2: Dvofazni algoritam koji smo koristili i u prvom i u drugom slučaju uvek daje pravilno obojene grafove, bili oni delimično obojeni ili potpuno [2].

Napomena 4.3: Primenjujući Kempeov algoritam na graf sa slike 4.3, možemo ukloniti sve čvorove tako da oni u momentu uklanjanja budu stepena manjeg od 3. Zato, u trenutku bojenja određeni čvor je susedan sa najviše dva već obojena čvora, pa ga možemo bez ikakvih problema obojiti. Za razliku od toga, kod grafa sa slike 4.4 imali smo situaciju da je čvor *i* u trenutku bojenja bio okružen sa 3 već obojena čvora (*c*, *a* i *h*) i zato nismo bili sigurni da li ćemo moći da ga obojimo (njegova obojivost zavisila je od izbora boje čvora *c*, dok takva zavisnost ne postoji kod bojenja grafa sa slike 4.3). Inspirisana ovim zaključkom je sledeća ideja:

Ideja: Graf se može potpuno i pravilno obojiti korišćenjem 3 boje ako po Kempeovom algoritmu možemo ukloniti sve čvorove tako da oni u momentu uklanjanja budu stepena manjeg od 3. Napomenimo da ako po Kempeovom algoritmu **ne** možemo ukloniti sve čvorove tako da oni u momentu uklanjanja budu stepena manjeg od 3, to ne znači da graf nije triobojiv (primer je graf sa slike 4.4). Drugim rečima, bez bojenja možemo proveriti obojivost jedne klase grafova. Ovaj pristup je suštinski ekvivalentan teoretskom pristupu: *Graf je obojiv ako i samo ako je obojiv korišćenjem nekog algoritma*, a praktično može biti veoma koristan u proveravanju obojivosti grafa, pogotovo ako se to radi bez korišćenja računara. Primetimo da se ovakav pristup može i uopštiti, tj. ne mora se koristiti samo za 3 boje.

Napomena 4.4: Takođe, na primeru grafa 4.3 možemo ponovo pokazati zašto je bolje koristiti Kempeov algoritam nego nasumičan redosled bojenja. Korišćenjem Kempeovog algoritma, za bilo koji izbor boja koji odgovara algoritmu, dobija se pravilno i potpuno obojen graf iz već pomenutog razloga. Za razliku od toga, kod nasumičnog biranja redosleda može se dogoditi situacija gde je čvor stepena 5 već okružen sa 3 različito obojena čvora, pa se on ne može obojiti, tj. dobijamo delimično obojen graf (primer takvog izbora čvorova su čvorovi koji čine severoistočni toranj *Biblioteke* – slika 4.7).



Slika 4.7 – problem u bojenju centralnog čvora severoistočnog tornja kod nasumičnog biranja rasporeda bojenja (levo), pravilno i potpuno obojen graf korišćenjem Kempeovog algoritma (desno)

Naredni algoritam koji ćemo razmatrati je algoritam pohlepnog bojenja.

4.2. Algoritam pohlepnog bojenja

U opštem slučaju, za bojenje grafa algoritmom pohlepnog bojenja može se koristiti k boja. Međutim, mi ćemo se kao i do sada ograničiti na problem triobojivosti grafa.

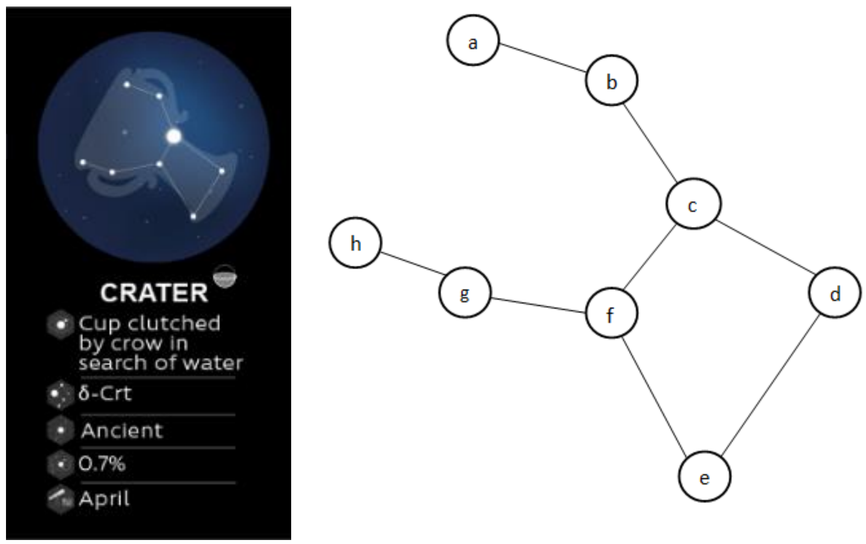
Da bismo graf obojili algoritmom pohlepnog bojenja, njegov skup čvorova mora biti uređen skup, odnosno čvorovi moraju na neki način biti numerisani. Na primer, čvorove grafa sa slike 4.4 možemo numerisati na način dat Kempeovim algoritmom ($\{j, a, d, e, h, c, i, f, g, b\}$) ili nasumično kako smo to učinili u ranijem razmatranju ($\{a, j, h, g, i, c, e, b, f, d\}$). Napomenimo da se u okviru uređenog skupa ne smeju pojaviti dva ista čvora, tj. svaki čvor se u uređenom skupu mora pojaviti tačno jednom. Ovakav graf se u anglosaksonskoj literaturi naziva *ordered graph*.

Sada kada imamo numerisane čvorove, možemo primeniti algoritam pohlepnog bojenja koji glasi:

1) Utvrditi redosled boja u paleti (jednoznačno numerisati svaku od boja).
2) Prvi čvor u numerisanom nizu čvorova obojiti prvom bojom.
3) Sledećem čvoru u nizu dodeliti boju sa najmanjim indeksom koja nije već iskorišćena na susedne čvorove.
4) Ponavljati korak pod 3) sve dok graf ne bude potpuno obojen.

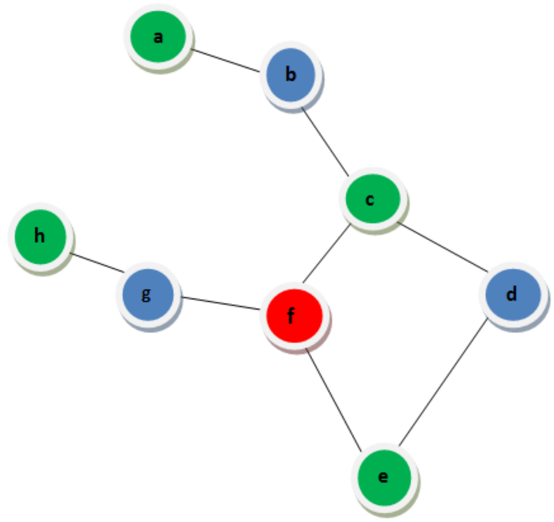
Napomena 4.5 – korak 4): Graf će moći da bude obojen ukoliko je on obojiv uzimajući u obzir njegov hromatski broj i redosled čvorova.

Demonstriraćemo algoritam na primeru grafa sazvežđa *Crater* (slika 4.8). Čvorove grafa označićemo slovima od *a* do *h*.



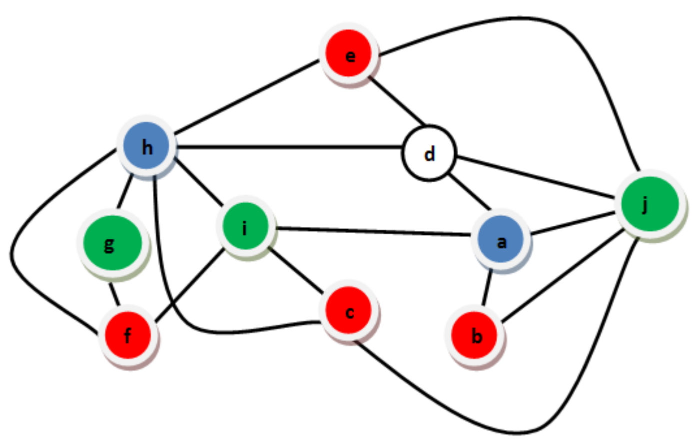
Slika 4.8 – Primer grafa

Prvo, usvojićemo redosled bojenja čvorova. Neka to bude nasumičan redosled, npr. ({*d*, *e*, *g*, *f*, *c*, *b*, *a*, *h*}). Zatim, treba odrediti redosled boja u paleti. Pošto nijedna od boja nema neki poseban značaj, učinićemo to proizvoljno npr. {plava, zelena, crvena}. Prvi čvor (***d***) bojimo prvom bojom u nizu tj. plavom. Drugi čvor (***e***) bojimo drugom bojom u nizu – zelenom, jer je čvor *d* koji je susedan čvoru *e* već obojen plavom. Treći čvor (***g***) bojimo ponovo prvom bojom – plavom (jer nije susedan ni sa jednim od prethodno obojenih čvorova, pa po algoritmu prednost ima čvor najmanjeg indeksa tj. indeksa 1). Četvrti čvor (***f***) moramo obojiti crvenom bojom jer su njemu susedni čvorovi ***g*** i ***e*** već obojeni plavom i zelenom. Dalje, istim postupkom bojimo čvorove: ***c*** zelenom, ***b*** plavom, ***a*** zelenom i ***h*** zelenom. Ovim smo potpuno i pravilno obojili graf (slika 4.9).



Slika 4.9 – Pravilno obojen graf sa slike 4.8

Obojimo sada i graf sa slike 4.4 algoritmom pohlepnog bojenja. Usvojićemo nasumičan redosled čvorova koji smo već pomenuli ({*a*, *j*, *h*, *g*, *i*, *c*, *e*, *b*, *f*, *d*}) i isti redosled boja u paleti kao u prethodnom primeru. Istim postupkom kao malopre, čvor *a* bojimo plavom, *j* zelenom, *h* plavom, *g* zelenom, *i* zelenom, a čvorove *c*, *e*, *b* i *f* bojimo crvenom bojom (slika 4.10). Vidimo da čvor *d* nećemo moći da obojimo nijednom bojom (što je bio slučaj i u prethodnom razmatranju – slika 4.6).

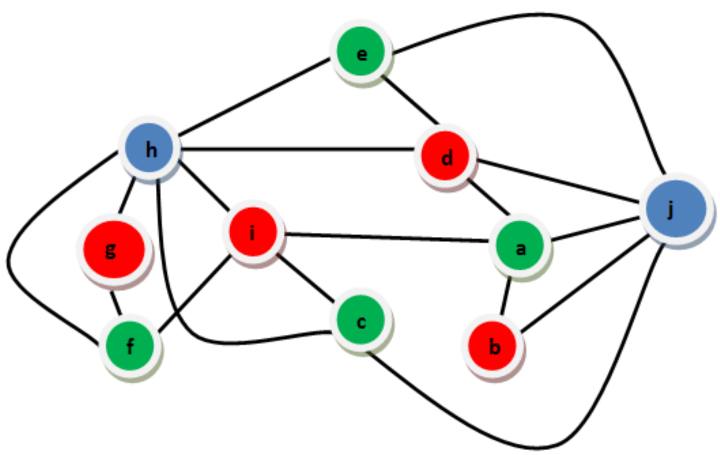


Slika 4.10 – Bojenje grafa sa slike 4.4 algoritmom pohlepnog bojenja kada se usvoji nasumičan redosled čvorova isti kao za graf za slike 4.6

Kako smo ovaj graf već uspeli pravilno da obojimo sa 3 boje kada smo koristili Kempeov algoritam, znamo da je on sigurno obojiv sa 3 boje (lako možemo da pokažemo da sa 2 boje već nije obojiv: bojeći na primer čvor *a* plavom i čvor *j* zelenom zaključujemo da čvor *d* moramo obojiti nekom trećom bojom; iz ovog razmatranja sledi da je hromatski broj ovog grafa upravo 3). Znajući da je hromatski broj grafa odgovarajući za bojenje sa 3 boje, iz *Napomene 4.5* zaključujemo da je problem u rasporedu čvorova za bojenje.

Diskusija: Ako bismo za dobijanje rasporeda čvorova koristili Kempeov algoritam, a za samo bojenje algoritam pohlepnog bojenja, da li bismo dobili bolji rezultat nego korišćenjem nasumičnog redosleda?

Usvojićemo redosled čvorova po Kempeovom algoritmu ({*j, a, d, e, h, c, i, f, g, b*}). Redosled boja u paleti ostaje isti kao i do sada ({plava, zelena, crvena}). Prethodno objašnjenim postupkom obojićemo graf. Rezultat bojenja prikazan je na slici 4.11.



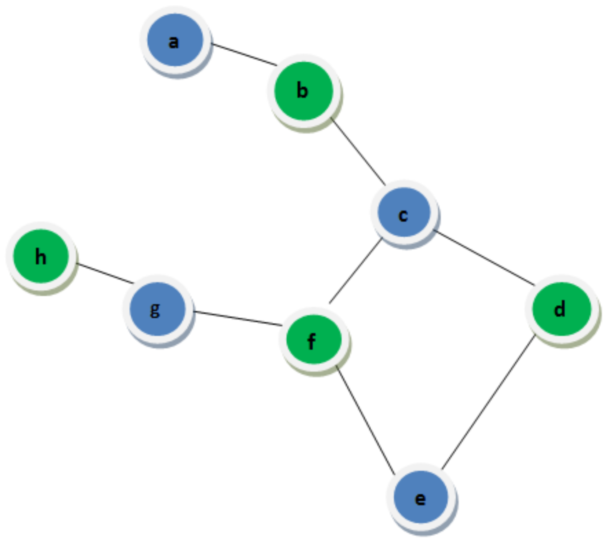
Slika 4.11 – Graf sa slike 4.4 obojen kombinovanim algoritmom

Vidimo da smo korišćenjem kombinovanog algoritma dobili bolji rezultat od prethodnog, jer smo dobili pravilno i potpuno obojen graf.

Primetimo da smo na ovaj način ponovo koristili dvofazni algoritam, samo smo na drugi način kombinovali algoritme u okviru faza.

Diskusija: Da li smo graf sa slike 4.8 mogli obojiti sa manje boja nego što jesmo korišćenjem algoritma pohlepnog bojenja?

Odgovor na ovo pitanje je potvrdan. Umesto korišćenja tri boje, mogli smo koristiti samo dve i jedno takvo rešenje prikazano je na slici 4.12.



Slika 4.12 – Graf sa slike 4.8 obojen korišćenjem hromatskog broja boja

Ova osobina je jedna od mana algoritma pohlepnog bojenja. Međutim, ispostavlja se da ovaj problem može biti rešen odgovarajućim izborom rasporeda čvorova pri bojenju. O tome govori naredna teorema:

Teorema 4.1. Za svaki graf G postoji raspored čvorova takav da se korišćenjem algoritma pohlepnog bojenja dobija graf obojen sa hromatskim brojem boja.

Dokaz. Po definiciji hromatskog broja znamo da bar jedno takvo bojenje postoji. Dodelimo proizvoljan redosled bojama u paleti. Razmatramo skup svih načina da se oboji graf G i od njih odaberemo one koji koriste hromatski broj boja. Od njih dalje biramo one u kojima je prva boja u paleti zastupljena maksimalan broj puta, zatim iz tako suženog skupa dalje biramo one u kojima je druga boja u paleti zastupljena maksimalan broj puta itd. Ovim postupkom doći ćemo do jednog odgovorajućeg bojenja grafa. Sada treba naći redosled čvorova koji je potrebno koristiti da bi se dobio tako obojen graf. Znamo da su svi čvorovi obojeni drugom bojom iz palete u trenutku bojenja morali imati bar jedan susedan čvor obojen prvom bojom iz palete (inače bi i oni bili obojeni prvom bojom). Ovo razmatranje važi i za preostale boje. Zbog toga, redosled čvorova biramo tako što prvo biramo čvorove obojene prvom bojom iz palete, zatim čvorove obojene drugom itd. (ređamo čvorove po prioritetu boja kojim su obojeni, od najvećeg prioriteta ka najmanjem). Kada smo rasporedili sve čvorove, primenjujući algoritam pohlepnog bojenja dobijamo bojenje grafa od kojeg smo krenuli, što znači da je moguće izabrati takav raspored čvorova da algoritam pohlepnog bojenja koristi hromatski broj boja, što je i trebalo pokazati.

U narednom delu uvešćemo još jedan algoritam koji će nam u kombinaciji sa algoritmom pohlepnog bojenja pomoći da dokažemo teoremu 3.4.

4.3. Algoritam raspoređivanja čvorova

Ovaj algoritam pripada grupi algoritama koji pronalaze odgovarajući raspored čvorova koji kasnije možemo koristiti za bojenje grafa tj. pripada prvoj fazi dvofaznog algoritma.

Da bismo primenili ovaj algoritam za bojenje grafa G potrebno je da:

1) Izaberemo stablo grafa G (označićemo ga sa G_t). Izabrano stablo G_t zadržava sve čvorove iz G i čuva osobinu njihove susednosti. Smatraćemo da su sve grane u G_t iste, jedinične dužine.
2) Izaberemo bilo koji čvor v_0 iz grafa G_t.
3) Tražimo čvor koji zadovoljava sledeći uslov: *Put od čvora v_0 do tog čvora ima najveću dužinu*. Ukoliko više takvih čvorova postoji, biramo bilo koji od njih. Izabrani čvor postavimo na prvo mesto u nizu koji formiramo.
4) Tražimo neraspoređeni čvor koji zadovoljava isti uslov. Ponovo, ako ima više takvih čvorova, izaberemo bilo koji od njih. Postavimo izabrani čvor na sledeće mesto u nizu.
5) Ponavljamo korak 4) sve dok ne rasporedimo sve čvorove osim v_0.
6) Čvor v_0 postavimo na poslednje mesto u nizu.

Sada možemo dokazati teoremu 3.4:

Dokaz. Pošto po uslovu teoreme graf G nije regularan, bar jedan čvor u G je stepena manjeg od Δ. Izaberimo bilo koji takav čvor i obeležimo ga sa v_0. Primenimo algoritam raspoređivanja čvorova, a zatim i algoritam pohlepnog bojenja. Zbog samog redosleda izbora čvorova po algoritmu raspoređivanja čvorova, svaki čvor osim v_0 imaće za suseda čvor većeg indeksa u nizu. U skladu sa navedenim i činjenicom da svaki čvor u G ima najviše Δ suseda, u trenutku bojenja čvor w će imati najviše $\Delta - 1$ prethodno obojenih čvorova, što znači da će za njegovo bojenje biti dostupna bar jedna boja. Kada po algoritmu pohlepnog bojenja stignemo do poslednjeg čvora, čvora v_0, on će imati manje od Δ suseda koji su svi već obojeni, pa za bojenje v_0 na raspolaganju imamo bar jednu boju. Dakle, G može biti pravilno i potpuno obojen korišćenjem najviše Δ boja.

Za kraj ovog odeljka, predstavićemo još jedan jednostavan algoritam koji nam može poslužiti za izračunavanje hromatskog broja grafa. Pri objašnjavanju, koristićemo oznake i zaključke iz dokaza teoreme 3.6.

4.4. Algoritam za izračunavanje hromatskog broja

Za početak, uočimo neka dva nesusedna čvora v i w (ukoliko su svi čvorovi međusobno susedni, onda je G kompletan graf, a njegov hromatski broj već znamo, pa ćemo ovakve grafove nadalje izuzeti iz razmatranja). Sa G + $\{v,w\}$ = G + e, gde je e = $\{v,w\}$ grana koja ima svoje krajeve u v i w, označićemo graf koji je dobijen iz G dodavanjem grane e, slično kao što smo to učinili za operaciju oduzimanja grane e (G – e) u dokazu teoreme. Usvojićemo i oznaku G/e iz navedenog dokaza. Posmatrajmo dva različita skupa kombinacija za pravilno bojenje grafa G:

1) U pravilno obojenom grafu G čvorovi *v* i *w* su različite boje

Primetimo da ovakve kombinacije bojenja odgovaraju i grafu G + e. Takođe, bilo koje pravilno bojenje grafa G + e ujedno je i pravilno bojenje grafa G u kojem v i w imaju različite boje. Zaključujemo da je $\chi(G + e)$ minimalan broj boja potreban za bojenje grafa G tako da v i w budu različite boje.

2) U pravilno obojenom grafu G čvorovi *v* i *w* su iste boje

Primetimo da ovakve kombinacije bojenja sada odgovaraju grafu G/e gde je čvor x (dobijen sažimanjem čvorova v i w) obojen istom bojom kao v i w. Takođe, bilo koje bojenje grafa G/e daje pravilno bojenje grafa G u kojem su v i w obojeni isto kao x. Zaključujemo da je $\chi(G/e)$ minimalan broj boja potreban za bojenje grafa G tako da v i w budu iste boje.

Pošto su 1) i 2) jedini mogući slučajevi, zaključujemo da će traženi hromatski broj grafa G biti jednak $\min(\chi(G + e), \chi(G/e))$.

Konačno, možemo primeniti na prethodno razmatranje postupak rekurzije na sledeći način: uvođenjem smene $G_1 = G + e$ i $G_2 = G/e$ (pri čemu e sada označava neku drugu granu koja ispunjava isto što i prethodna) dobijamo $\chi(G_1) = \min(\chi(G_1 + e), \chi(G_1/e))$ i $\chi(G_2) = \min(\chi(G_2 + e), \chi(G_2/e))$, zatim za izračunavanje $\chi(G_1)$ i $\chi(G_2)$ uvodimo smenu $G_3 = G_1 + e$, $G_4 = G_1/e$, $G_5 = G_2 + e$ i $G_6 = G_2/e$ itd. (primetimo da se postupak grana, za graf G imali smo njegova dva pomoćna grafa G_1 i G_2, za G_1 imamo grafove G_3 i G_4 itd). Ponavljanjem ovog postupka možemo doći do hromatskog broja grafa G (potrebno je izvršiti ceo postupak dovoljan broj puta da bi se na kraju svake grane dobio graf dovoljno jednostavan za bojenje – u opštem slučaju graf kojem već znamo hromatski broj ili specijalno, kompletan graf). Zbog čega se sa dovoljnim brojem ponavljanja postupka dobija kompletan graf? Intuitivno govoreći, graf oblika $G_i + e$ ima veći broj grana, dok G_i/e ima manji broj čvorova od polaznog grafa G_i, pa kada dovoljno puta dodamo po granu i oduzmemo po čvor u graničnom slučaju dobijamo kompletan graf. Jednom kada dođemo do kompletnog grafa K_m nema potrebe da dalje ponavljamo postupak, jer znamo da je hromatski broj grafa K_m jednak m.

U sledećem odeljku bavićemo se složenošću algoritama za bojenje i određivanje hromatskog broja proizvoljnog grafa.

5. Složenost algoritama

Za primenu nekog algoritma od izuzetnog je značaja njegova složenost. Algoritmi koji imaju složenost O(n!) ili $O(m^n)$ nisu od interesa za implementaciju, jer nisu isplatljivi. Procenom složenosti različitih algoritama i mnogim drugim pitanjima bavi se teorija kompleksnosti. U okviru nje se algoritmi i problemi klasifikuju u klase kompleksnosti. Jedna takva klasa problema je NP klasa kojoj pripadaju problemi odlučivanja koji se mogu rešiti ***N**edeterminističkim algoritmom za **P**olinomijalno vreme*. Za neki problem se kaže da je NP-težak problem ukoliko je svaki problem iz klase NP polinomijalno svodljiv na taj problem. Problem je NP-kompletan ukoliko pripada klasi NP i ukoliko je NP-težak.

Problem k-obojivosti grafa (uključujući i 3-obojivost) pripada klasi NP-kompletnih problema, dok je problem izračunavanja hromatskog broja grafa NP-težak problem. U ovom radu nećemo se baviti dokazom da je problem 3-obojivosti NP-kompletan (dokaz se može pogledati u [1]).

Ovaj segment podelićemo na dva dela. U prvom delu bavićemo se složenošću izračunavanja hromatskog broja, dok će u drugom delu akcenat biti na složenosti nekih od algoritama za bojenje grafova kojima smo se bavili u odeljku 4.

5.1. Složenost izračunavanja χ(G)

U opštem slučaju, izračunati hromatski broj za proizvoljan graf G je veoma složeno i zahteva korišćenje velikog broja operacija. Za primenu bojenja grafova o kojoj će biti više reči u odeljku 6, potrebno je odrediti hromatski broj u konačnom vremenu. Dakle, potreban nam je neki valjan algoritam koji će u konačnom vremenu kao rezultat dati hromatski broj zadatog grafa. Jedan od takvih algoritama je dat na kraju prethodnog odeljka.

Teorema 5.1. Algoritam iz prethodnog odeljka ispravno računa hromatski broj grafa G u konačnom vremenu.

Dokaz. Pretpostavimo da graf G ima n čvorova i m grana. Broj nesusednih čvorova (oznaka: na(G)) je tada jednak $\binom{n}{2} - m$. Dokaz ćemo izvršiti indukcijom po na(G). Baza indukcije: Ako je na(G) = 0, G je kompletan graf pa znamo bez daljeg računanja da je njegov hromatski broj jednak n. Za nenulte vrednosti na(G) važi:

1) na(G + e) = $\binom{n}{2} - (m + 1)$ = na(G) – 1 < na(G)
2) na(G/e) = $\binom{n-1}{2} - (m - c) \leq \binom{n-1}{2} - m + n - 2 = \frac{(n-1)(n-2)}{2} - m + n - 2 =$

$= \frac{n(n-1)}{2} - \frac{2(n-1)}{2} - m + n - 2 = \binom{n}{2} - m - 1 = na(G) - 1 < na(G)$

gde je c broj zajedničkih suseda čvorova v i w (za koji znamo da je manji ili jednak broju čvorova n umanjenom za 2 čvora – v i w).

Sada možemo postaviti indukcionu hipotezu: algoritam ispravno izračunava χ(G + e) i χ(G/e) u konačnom vremenu. Pošto je χ(G) = min(χ(G + e), χ(G/e)), za izračunavanje hromatskog broja grafa G potreban je samo još jedan korak, dakle i on se ispravno izračunava u konačnom vremenu.

Još jedan način izračunavanja hromatskog broja je izračunavanje preko datog hromatskog polinoma. Postupak je veoma jednostavan: kao argumente $P_G(k)$ ubacujemo redom vrednosti 1, 2, 3,... – sve dok kao rezultat ne dobijemo nenultu vrednost. Možemo odrediti i složenost takvog izračunavanja. Uzmimo opšti oblik $P_G(k) = k^n + a_{n-1}k^{n-1} + \cdots + a_1 k$ (primetimo da smo u određivanju ovakvog oblika primenili teoremu 3.6 i svojstva 3.1 i 3.2). Za jedno fiksirano k, moramo iskoristiti n – 1 sabiranje i (n – 1) + (n – 2 + 1) + ... + 1 = n – 1 + $\sum_{m=1}^{n-1} m = n - 1 + \frac{n(n-1)}{2} = \frac{n^2 + n - 2}{2}$ množenja. To čini ukupno $\frac{n^2 + 3n - 4}{2}$ operacija za jedno fiksirano k. U najgorem slučaju ovakvo izračunavanje moraćemo da ponovimo n puta (jer je χ(G) ≤ n). Složenost ovakvog izračunavanja je polinomska: $O(n^3)$. Ovakva složenost smatra se dovoljno dobrom za implementaciju na računaru. Međutim, šta ako hromatski polinom nije unapred zadat već ga je potrebno odrediti, kao što je to obično slučaj? Za njegovo određivanje možemo koristiti formulu izvedenu u dokazu teoreme 3.6: $P_G(k) = P_{G-e}(k) - P_{G/e}(k)$ i nju primeniti u postupku rekurzije slično kao što smo to učinili u algoritmu za određivanje hromatskog broja. Postupak treba ponavljati sve dok na kraju ne dobijemo graf bez ijedne grane, jer za njega unapred znamo hromatski polinom (za takav graf od n čvorova $P_G(k) = k^n$).

Navedeni postupci mogu se koristiti za proveru da li je graf obojiv korišćenjem 3 boje. U prvom razmatranom algoritmu, ukoliko za hromatski broj grafa dobijemo broj veći od 3, znamo da graf nije triobojiv. U drugom algoritmu, sve što je potrebno da uradimo je da izračunamo $P_G(3)$, ukoliko dobijemo nenultu vrednost znamo da je graf sigurno triobojiv.

5.2. Složenost algoritama za bojenje grafova

Prvi algoritam koji smo pomenuli u analizi algoritama za bojenje grafova je postupak nasumičnog bojenja. Ovaj algoritam iako veoma jednostavan ima veoma veliku složenost. Prvo dodeljujemo nasumično boje svim čvorovima grafa kojih je n. Neka je dodeljivanje boje čvoru elementaran korak. Pošto različitih kombinacija bojenja korišćenjem 3 boje ima 3^n (u opštem slučaju k^n), za generisanje svih mogućih kombinacija potrebno je $n3^n$ elementarnih koraka. Zatim, treba proći kroz svako rešenje i utvrditi da li je graf pravilno obojen. Da bismo utvrdili da li je graf pravilno obojen, treba za svaki čvor proći kroz sve preostale čvorove da bi se pronašli svi njemu susedni čvorovi, a zatim za njih izvršiti proveru da li su odgovarajuće boje. Dakle, n čvorova · ((n – 1) pristup čvoru za proveru susednosti + maksimalno (n – 1) provera boje) daje još dodatnu složenost od $O(n^2)$, pa je ukupna složenost ovakvog algoritma $O(n3^n)$. Algoritmi ovakve složenosti nikako nisu praktični za računarsku implementaciju.

Drugi algoritam koji smo koristili je Kempeov algoritam. Pokušajmo da analiziramo njegovu složenost, pa zatim da ga uporedimo sa složenošću prethodnog algoritma. Biramo proizvoljan čvor od zadatih n čvorova, zatim prolazimo kroz preostalih n – 1 čvorova i proveravamo da li su ovi čvorovi susedni sa izabranim čvorom. Ukoliko jesu, stepen datog čvora povećavamo za 1. Zatim, vršimo proveru da li je stepen čvora manji od 3 (u opštem slučaju manji od k). Ukoliko jeste, čvor obrišemo i postavimo ga u niz koji formiramo za kasnije bojenje grafa. Ako nije, ponovimo postupak za neki drugi čvor (u najgorem slučaju ponovićemo ovakav postupak n – 1 puta). Dakle, za prvi čvor (ukoliko je dobro izabran) imamo: (n – 1) proveru susednosti + maksimalno 1 sabiranje pri formiranju stepena čvora + 1 proveru stepena + 1 brisanje + 1 postavljanje u niz, što je n + 3 elementarnih koraka. Ukoliko čvor nije dobro izabran, imamo 2n – 2 koraka (u ovo smo uračunali najgori slučaj od n – 2 sabiranja, jer stepen čvora može biti najviše n – 1). Sada razmatramo najgori mogući slučaj, tj. da smo u prvoj iteraciji postupka morali da prođemo kroz svih n čvorova da bismo našli odgovarajući čvor, u drugoj iteraciji postupka kroz svih n – 1 preostalih čvorova itd. Složenost takvog algoritma bi bila: $\big((2n-2)(n-1)+n+3\big)+\ \big((2n-4)(n-2)+\ n+2\big)+\cdots=\ \sum_{i=1}^{n}(2(n-i)^2+n+4-i)=\ 2\sum_{i=1}^{n}(n-i)^2+\sum_{i=1}^{n}(n+4-i)$, što sređivanjem daje polinomsku složenost $O(n^3)$. Dalje, kada imamo redosled čvorova, potrebno je obojiti čvorove. Za prvi čvor u nizu imamo samo jedan korak – bojenje čvora, za drugi moramo proveriti susednost prvog čvora, izvršiti najviše jednu proveru boje i obojiti ga – 3 koraka, a počevši od trećeg čvora, za i-ti čvor treba proveriti susednost za i – 1 čvorova, izvršiti najviše 2 provere boje i obojiti dati čvor – i + 2 koraka. Ovaj postupak daje složenost $O(n^2)$, pa je ukupna složenost Kempeovog algoritma po ovakvoj proceni $O(n^3)$. Zbog znatno manje složenosti od prethodnog algoritma, Kempeov algoritam je znatno pogodniji za implementaciju od nasumičnog bojenja.

Napomena 5.1: U ovoj analizi složenosti, u obzir smo uzeli samo grafove koji se sigurno mogu potpuno obojiti korišćenjem tri boje (pogledati napomenu 4.3 i ideju za proveru triobojivosti ispod nje), tj. zahtevali smo da svaki čvor pre nego što ga uklonimo iz grafa bude stepena manjeg od 3. Da smo imali situaciju da cela jedna iteracija nema takav čvor, uklonili bismo proizvoljan čvor iz grafa i nastavili postupak. U slučaju da na kraju dođemo u situaciju da ne možemo da obojimo neki čvor, morali bismo da ponovimo ceo postupak ispočetka menjajući boje čvorovima grafa, sve dok ne dođemo do pravilno i potpuno obojenog grafa, ukoliko je naravno moguće do njega doći (zbog toga je preporučljivo prvo proveriti hromatski broj grafa).

Treći algoritam koji smo razmatrali je algoritam pohlepnog bojenja. Pokušaćemo da odredimo njegovu složenost. Ovaj algoritam zahteva već gotov raspored čvorova, neka to za sada bude nasumični raspored (neka je numerisanje čvora elementarni korak kao i u prethodnom razmatranju – to čini n koraka). Takođe, izvršimo numeraciju boja (još 3 koraka). Nadalje, složenost se analizira isto kao u koraku bojenja u prethodnom algoritmu čime se dobija složenost $O(n^2)$. Međutim, imajmo u vidu da posle ovako izvršenog postupka nećemo sa sigurnošću imati pravilno i potpuno obojen graf (to smo videli i na primeru grafa sa slike 4.4). Mogućih permutacija čvorova ima n!, pa bismo u najgorem slučaju morali da ponovimo isti postupak bojenja n! puta (po teoremi 4.1 znamo da će postojati odgovarajući raspored čvorova). To daje složenost od $O(n^2 n!)$ koja je prevelika za implementaciju. Zbog toga, algoritam pohlepnog bojenja treba koristiti zajedno sa još nekim algoritmom za dobijanje rasporeda čvorova, npr. Kempeovim algoritmom ili algoritmom raspoređivanja čvorova. Ovakve kombinovane algoritme već smo razmatrali u prethodnom poglavlju. Konkretno, kombinovani algoritam Kempeov – pohlepno bojenje dao bi bolji rezultat od pojedinačno korišćenih algoritama u slučaju bojenja grafa sa slike 4.4 koji je imao jedan čvor koji u momentu uklanjanja nije bio stepena manjeg od 3 (u slučaju Kempeovog algoritma u zavisnosti od izbora boja za pojedinačne čvorove, u opštem slučaju morali bismo ponoviti postupak više puta, dok smo sa kombinovanim algoritmom odmah dobili pravilno i potpuno obojen graf).

Napomena 5.2: Treba napomenuti da smo u ovom poglavlju vršili samo **procenu** složenosti algoritama bez ikakvih optimizacija. Sama složenost zavisi od načina implementacije, pa tako neke implementacije ovih algoritama imaju manju složenost nego što smo je za njih procenili, npr. za algoritam pohlepnog bojenja može se realizovati složenost $O(n + m)$ [3]. Naš cilj u ovom poglavlju bio je da prikažemo razliku između složenosti nasumičnog bojenja sa jedne strane i korišćenih algoritama sa druge strane, kako bismo opravdali korišćenje razmatranih algoritama.

6. Primena bojenja grafova

Bojenje grafova ima svoju primenu u širokom spektru oblasti. Jedna velika grupa problema koji mogu da se modeluju konceptom bojenja grafova su problemi raspoređivanja. Sledećih par primera ilustruju primenu bojenja u rešavanju baš takvih problema.

Primer 6.1. Bojenje grafova možemo koristiti da napravimo raspored predavanja za studente. Predstavićemo predavanja kao čvorove grafa, pri čemu su čvorovi susedni ako odgovarajuća predavanja imaju preklapanje u studentima koji ih slušaju. Tada, koristeći termine predavanja kao boje, možemo obojiti graf tj. napraviti raspored tako da se nijednom studentu predavanja ne preklapaju.

Primer 6.2. Ako radio stanice predstavimo kao čvorove grafa pri čemu su dva čvora susedna ukoliko su odgovarajuće stanice dovoljno blizu da interaguju jedna sa drugom, bojenjem grafa možemo raspodeliti radne učestanosti tako da radio stanice nesmetano funkcionišu.

Primer 6.3. Ukoliko čvorovi grafa predstavljaju semafore na raskrsnici pri čemu su dva čvora susedna ako odgovarajući semafori ne smeju istovremeno pokazivati zeleno svetlo, bojenjem grafa se može regulisati saobraćaj na raskrsnici.

Još jedna važna primena bojenja grafova je u računarstvu. U cilju smanjenja vremena izvršavanja dela koda, neki kompajleri koriste tehniku alokacije registra koja se može modelovati bojenjem grafova. Kompajler pravi takozvani graf interferencije kod kojeg čvorovi predstavljaju promenljive, a dva čvora su povezana granom ako se odgovarajuće promenljive moraju koristiti istovremeno. Ako se graf može obojiti sa k boja, tada bilo koji skup promenljivih koje zahtevaju istovremeno korišćenje može stati u najviše k registara.

7. Zaključak

Bojenje grafova je veoma kompleksan multidisciplinaran problem. U ovom radu pokušali smo da damo kratak, ali celovit prikaz dela tog problema. Uz veoma važnu matematičku podlogu, priložili smo i zanimljive primere grafova i neke od primena bojenja, kao i zaključke koji mogu biti interesantni za dalje bavljenje ovom oblašću.

8. Literatura

[1] Čisar Angela, *Algoritmi bojenja čvorova u grafovima*, master rad, Univerzitet u Novom Sadu, Prirodno - matematički fakultet, 2016

[2] https://www.cs.princeton.edu/~appel/Color.pdf

[3] Marek Kubale, *Graph Colorings*, American Mathematical Society, 2004

[4] http://poincare.math.rs/~jelenagr/AIDA/cas11.pdf

[5] R.M.R. Lewis, *A guide to Graph Colouring – Algorithms and Applications*, Springer, 2016

[6] https://www.whitman.edu/Documents/Academics/Mathematics/stevens.pdf

[7] https://www.whitman.edu/mathematics/cgt_online/book/section05.01.html

[8] https://www.whitman.edu/mathematics/cgt_online/book/section05.08.html

[9] https://www.whitman.edu/mathematics/cgt_online/book/section05.09.html

[10] https://www.whitman.edu/mathematics/cgt_online/book/section05.10.html

[11] https://www.britannica.com/science/four-color-map-problem