Sparivanje usmjerenih acikličkih grafova

Blažević, Mislav

Sveučilište J.J.Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike

Mislav Blažević

Sparivanje usmjerenih acikličkih grafova

Završni rad

Osijek, 2020.

Sveučilište J.J.Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike

Mislav Blažević

Sparivanje usmjerenih acikličkih grafova

Završni rad

Mentor: izv.prof.dr.sc. Domagoj Matijević

Osijek, 2020.

Sažetak. U ovom radu se bavimo problemom pronalaska maksimalnog sparivanja usmjerenih acikličkih grafova. Analizirat ćemo problem i prezentirat neke algoritme za rješavanje tog problema.
Ključne riječi: sparivanja, usmjereni aciklički grafovi, stabla

Abstract. In this paper we study the problem of finding maximum weight matching of directed acyclic graphs. We will analyze the problem and present some algorithms for solving the problem.
Keywords: matchings, directed acyclic graphs, trees

Sadržaj

1 Uvod

Uspoređivanje grafova je problem koji se javlja u raznim primijenjenim područjima. Primjerice, u bioinformatici je od interesa problem uspoređivanja filogenetskih stabala dobivenih različitim metodama,[15] a u kemoinformatici problem uspoređivanja kemijskih spojeva, koje također možemo tretirati kao grafove.[20] Postoje mnogi načini da formaliziramo koncept sličnosti grafova. Na primjer, pomoću pronalaska izomorfizma, maksimalnog zajedničkog grafa, minimalnog grafa koji sadrži oba grafa, računanjem edit udaljenosti, ili čak statističkim metodama kao što je uspoređivanje distribucija stupnjeva vrhova itd.[21] U ovom radu, motivirani problemima iz bioinformatike, ograničavamo se na uspoređivanje usmjerenih acikličkih grafova i to metodom pronalaska maksimalnog sparivanja koje čuva odnos među vrhovima.

2 Osnovni pojmovi

2.1 Grafovi

Definicija 2.1. ***Graf*** *G je uređeni par (V, E) gdje je V konačan skup a E relacija na V. Elemente skupa V zovemo* ***vrhovi*** *a elemente skupa E* ***bridovi*** *grafa G. Ukoliko je relacija E simetrična, kažemo da je G* ***neusmjeren****, inače da je* ***usmjeren****.*

Definicija 2.2. ***Šetnja*** *duljine $k \in \mathrm{N}_0$ u grafu G je uređena $k+1$-torka $(v_0, ..., v_k)$ takva da je $(v_{i-1}, v_i) \in E$ za $i = 1, ..., k$.* ***Put*** *je šetnja čiji su elementi međusobno različiti.* ***Podput*** *puta duljine k je put $(v_l, ..., v_r)$, $0 \leq l \leq r \leq k$. Za šetnju kažemo da je* ***ciklus*** *ukoliko je $v_0 = v_k$ i kažemo da je usmjereni graf* ***acikličkі*** *ukoliko ne sadrži ciklus. Nadalje, kažemo da je put duljine k* ***maksimalan*** *ukoliko ne postoji put duljine $k' > k$ čiji je on podput.*

Definicija 2.3. *Neka je $G = (V, E)$ usmjeren aciklički graf i $v, w \in V$. Ukoliko postoji put $(v_0, ..., v_k)$ u G takav da je $v_0 = v$ i $v_k = w$, reći ćemo da je vrh v* ***predak*** *vrha w, a vrh w* ***potomak*** *vrha v što ćemo označavati $v < w$.*

Definicija 2.4. *Neka je $G = (V, E)$ usmjeren aciklički graf. Vrh $r \in V$ takav da ne postoji $v \in V$ za koji vrijedi $v < r$ ćemo zvati* ***korijen*** *i skup svih korijena usmjerenog acikličkog grafa G ćemo označavati $r(G)$. Vrh $l \in V$ takav da ne postoji vrh $v \in V$ za koji vrijedi $l < v$ ćemo zvati* ***list*** *i skup svih listova usmjerenog acikličkog grafa G ćemo označavati $l(G)$.*

Definicija 2.5. *Neka je $G = (V, E)$ usmjeren aciklički graf. Za graf $G^* = (V, E^*)$ kažemo da je* ***tranzitivni zatvarač*** *grafa G ukoliko je $(v, w) \in E^*$ ako i samo ako je vrh v predak vrha w u grafu G.*

Definicija 2.6. *Neka je $G = (V, E)$ usmjeren aciklički graf. Za $A \subseteq V$ kažemo da je* ***antilanac*** *ukoliko vrijedi $v \not< w$ i $w \not< v$ za sve $v, w \in A$.*

Definicija 2.7. *Kažemo da su grafovi $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$* ***izomorfni****, u oznaci $G_1 \simeq G_2$, ukoliko postoji bijekcija $f : V_1 \to V_2$ takva da je $(u, v) \in E_1 \iff (f(u), f(v)) \in E_2$ za sve $u, v \in V_1$.*

2.2 Mreže

Definicija 2.8. *Neka je $G = (V, E)$ usmjeren graf takav da $(u, v) \in E \implies (v, u) \notin E$ i postoje $s, t \in V$ takvi da vrijedi $s < v < t$ za sve $v \in V \setminus \{s, t\}$ i neka je $c : V \times V \to \mathrm{R}^+$ funkcija koja zadovoljava $c(u, v) = 0$ za $(u, v) \notin E$. Uređenu četvorku (G, c, s, t) zovemo* ***protočna mreža****, funkciju c* ***kapacitet****, vrh s* ***izvor*** *a vrh t* ***ponor****.*

Definicija 2.9. *Funkciju $f : V \times V \to \mathrm{R}^+$ koja zadovoljava*

1. *$0 \leq f(u, v) \leq c(u, v)$ za sve $u, v \in V$*
2. *$\sum_{v \in V} f(v, u) = \sum_{v \in V} f(u, v)$ za sve $u \in V \setminus \{s, t\}$*

zovemo ***tok****, broj $|f| = \sum_{v \in V} f(s, v) - \sum_{v \in V} f(v, s)$ zovemo* ***vrijednost*** *toka f i za tok kažemo da je* ***maksimalan*** *ukoliko je njegova vrijednost najveća među svim tokovima mreže G.*

Definicija 2.10. *Neka je (G, c, s, t) protočna mreža i f tok. Funkciju*

$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & (u,v) \in E \\ f(v,u) & (v,u) \in E \\ 0 & inače \end{cases}$$

*zovemo **rezidualni kapacitet**, protočnu mrežu (G_f, c_f, s, t), gdje je $G_f = (V, E_f)$, $E_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$, zovemo **rezidualna mreža**, svaki put $p = (v_0, ..., v_k)$ u G_f takav da je $v_0 = s$ i $v_k = t$ zovemo **uvećavajući put** a broj $c_f(p) = \min_{i=1,..,k} c_f(v_{i-1}, v_i)$ **rezidualni kapacitet puta** p.*

Lema 2.1. *Neka je f tok u protočnoj mreži (G, c, s, t) i f' tok u rezidualnoj mreži (G_f, c_f, s, t). Tada je*

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & (u,v) \in E \\ 0 & inače \end{cases}$$

tok u G i njegova vrijednost je $|f| + |f'|$.

Lema 2.2. *Neka je f tok u protočnoj mreži (G, c, s, t) i p uvećavajući put u (G_f, c_f, s, t). Tada je*

$$f_p(u,v) = \begin{cases} c_f(p) & (u,v) \text{ je na putu } p \\ 0 & inače \end{cases}$$

tok u G_f i njegova vrijednost je $c_f(p) > 0$.

Definicija 2.11. ***Rez** (S, T) u protočnoj mreži (G, c, s, t), $G = (V, E)$ je particija skupa V u skupove S i $T = V \setminus S$ takve da je $s \in S$ i $t \in T$. Broj*

$$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

*zovemo **kapacitet** reza (S, T) i za rez kažemo da je **minimalan** ukoliko je njegov kapacitet najmanji među svim rezovima mreže G.*

Teorem 2.1. *[14] Neka je f tok u mreži G. Sljedeće tvrdnje su ekvivalentne:*

1. *f je maksimalan*
2. *G_f ne sadrži uvećavajući put*
3. *$|f| = c(S,T)$ za neki rez (S,T)*

Teorem 2.1 govori da uzastopnom primjenom lema 2.2 i 2.1 možemo pronaći maksimalan tok a time i minimalni rez. Ispostavlja se da ukoliko na svakom koraku izaberemo najkraći uvećavajući put, algoritam će imati složenost $O(VE^2)$.[14]

Teorem 2.2. *[13][9] Neka je $G = (V, E)$ graf, $G^* = (V, E^*)$ njegov tranzitivni zatvarač i $w : V \to \mathrm{Z}^+$ težinska funkcija. Za sve $v \in V$ neka su v' i v'' odgovarajući vrhovi. Konstruirajmo protočnu mrežu (H, c, s, t), $H = (U, F)$ na sljedeći način:*

$$U = \{s, t\} \cup \{v' : v \in V\} \cup \{v'' : v \in V\}$$

$$F = \{(s, v') : v \in V\} \cup \{(v'', t) : v \in V\} \cup \{(k', l'') : (k, l) \in E^*\}$$

$$c(s, v') = c(v'', t) = w(v) \text{ za sve } v \in V$$

i neka su svi ostali kapaciteti jednaki ∞. Nadalje, neka je (S, T) particija skupa U dobivena primjenom teorema 2.1 na tu mrežu. Tada je $A = \{v \in V : v' \in S \wedge v'' \in T\}$ antilanac koji maksimizira $\sum_{v \in A} w(v)$.

2.3 Sparivanja

Definicija 2.12. *Za neusmjeren graf $G = (V, E)$ kažemo da je **bipartitan** ukoliko postoji particija (V_1, V_2) skupa V takva da za sve $(u, v) \in E$ vrijedi $u \in V_1 \iff v \in V_2$.*

Definicija 2.13. *Neka je $G = (V, E)$ bipartitan graf s biparticijom (V_1, V_2). Skup $M \subseteq E \cap V_1 \times V_2$ takav da za sve $(u, v), (u', v') \in M$ vrijedi $u = u' \iff v = v'$ zovemo **sparivanje**. Ukoliko je dodatno definirana težinska funkcija $w : V_1 \times V_2 \to \mathrm{Q}$ za koju je $w(u, v) = 0$ ukoliko $(u, v) \notin E$ onda broj $\sum_{(u,v)\in M} w(u, v)$ zovemo **težina sparivanja** i najveću težinu sparivanja ćemo označavati $\sigma(V_1, V_2, w)$.*

Propozicija 2.1. *Neka je $G = (V, E)$ bipartitan graf s biparticijom (V_1, V_2), $v \in V_1$ proizvoljan i $w : V_1 \times V_2 \to \mathrm{Q}$ težinska funkcija. Tada vrijedi:*

$$\sigma(V_1, V_2, w) = \max \begin{cases} \max\limits_{v \in V_2} \sigma(V_1 \setminus \{u\}, V_2 \setminus \{v\}, w) + w(u, v) \\ \sigma(V_1 \setminus \{u\}, V_2, w) \end{cases}$$

Algoritam iz propozicije 2.1 je eksponencijalne složenosti, ali je praktičan ukoliko barem jedna strana grafa ima jako malen broj vrhova. Inače koristimo algoritam 1, koji je polinomne složenosti.[12]

Podatci: Graf $G = (V, E)$ s biparticijom (V_1, V_2) i težinska funkcija $w : V_1 \times V_2 \to \mathrm{Q}$
Rezultat: Sparivanje M grafa G_1 najveće težine

1 $M \leftarrow \emptyset$;
2 **dok** *postoji put s linije 7* **čini**
3 $\quad E' \leftarrow \{(v, u) : u \in V_1, v \in V_2, (u, v) \in M\} \cup \{(u, v) : u \in V_1, v \in V_2, (u, v) \notin M\}$;
4 $\quad l(u, v) \leftarrow \begin{cases} w(u, v) & u \in V_1, v \in V_2, (u, v) \in M \\ -w(u, v) & u \in V_1, v \in V_2, (u, v) \notin M \end{cases}$;
5 $\quad V_1^M \leftarrow \{u \in V_1 : (u, v) \notin M \text{ za sve } v \in V_2\}$;
6 $\quad V_2^M \leftarrow \{v \in V_2 : (u, v) \notin M \text{ za sve } u \in V_1\}$;
7 $\quad P \leftarrow$ najkraći $V_1^M - V_2^M$ put u grafu (V, E') s duljinama bridova l;
8 $\quad M \leftarrow \{e :$ ili e u M ili e na putu $P\}$;
9 **kraj**

Algoritam 1: Mađarska metoda

2.4 Šume

Definicija 2.14. *Neka je $G = (V, E)$ usmjeren aciklički graf. Ukoliko je $(u, v) \in E$ reći ćemo da je u **roditelj** vrha v i da je v **dijete** vrha u. Skup roditelja vrha v ćemo označavati $p(v)$ a skup djece $c(v)$ i za G ćemo reći da je **šuma** ukoliko je $|p(v)| \leq 1$ za sve $v \in V$. Šumu $G = (V, E)$ takvu da postoji točno jedan $r \in V$ takav da je $p(r) = \emptyset$ ćemo zvati **stablo ukorijenjeno u vrhu** r ili, kraće, **stablo**.*

Definicija 2.15. *Neka je $G = (V, E)$ graf i $S \subseteq V$. Za graf $G[S] = (S, E[S])$ gdje je $E[S] = \{(u, v) \in E : u, v \in S\}$ kažemo da je **podgraf induciran skupom** S.*

Definicija 2.16. *Neka je $G = (V, E)$ šuma, $v \in V$. Podgraf induciran skupom $\{w \in V : v \leq w\}$ ćemo zvati **podstablo inducirano vrhom** v i označavati $\tau(v)$.*

Definicija 2.17. *Neka je $G = (V, E)$ šuma. Za $b \in V$ ćemo reći da je **razgranat** ukoliko je $|c(b)| > 1$. Neka je dodatno $v \in V$ vrh koji nije razgranat. Razgranat vrh $b \in V$, $v < b$ takav da niti jedan $w \in V$ za koji vrijedi $v < w < b$ nije razgranat ćemo označavati $b(v)$ a s $\pi(v)$ ćemo označavati podgraf induciran skupom*

$$S = \begin{cases} \{w \in V : v \leq w \leq b(v)\} & b(v) \text{ postoji} \\ \{w \in V : v \leq w\} & \text{inače} \end{cases}$$

3 Sparivanje usmjerenih acikličkih grafova

Definicija 3.1. *Neka su $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$ usmjereni aciklički grafovi. Za $M \subseteq V_1 \times V_2$ kažemo da je **sparivanje usmjerenih acikličkih grafova** G_1 i G_2 ili, kraće, **sparivanje** ukoliko vrijedi:*

(S1) $x = x' \iff y = y'$ *za sve* $(x, y), (x', y') \in M$

(S2) $x < x' \iff y < y'$ *za sve* $(x, y), (x', y') \in M$

Nadalje, neka je $w : V_1 \times V_2 \to \mathrm{Z}^+$ *težinska funkcija. Broj*

$$w(M) := \sum_{(x,y)\in M} w(x, y)$$

*ćemo zvati **težina sparivanja** M i za sparivanje M ćemo reći da je **optimalno** ukoliko je njegova težina najveća među svim sparivanjima grafova G_1 i G_2. Težinu optimalnog sparivanja grafova G_1 i G_2 ćemo označavati $\delta(G_1, G_2)$.*

Jasno je da ukoliko iz definicije 3.1 uklonimo zahtjev (S2), tako definiran problem pronalaska optimalnog sparivanja se trivijalno svodi na elementarni problem pronalaska sparivanja najveće težine u bipartitnom grafu koji je opisan u poglavlju 2.3. Međutim, nas ne zanimaju bilo kakva sparivanja usmjerenih acikličkih grafova, nego samo ona koja čuvaju strukturu u smislu zahtjeva (S2) jer je to jedan način da formaliziramo pojam sličnosti grafova, koji je od interesa u primjeni.[11][7][8][1][16][15]

Za početak promotrimo slučaj $w(x, y) = 1$ za sve $(x, y) \in V_1 \times V_2$. Primijetimo da je tada problem pronalaska optimalnog sparivanja ekvivalentan pronalasku najvećeg podgrafa grafa G_1^* koji je izomorfan nekom podgrafu grafa G_2^*. Neka je M optimalno sparivanje. Broj $S(G_1, G_2) := \frac{|M|}{|V_1|+|V_2|-|M|}$[1] nazovimo **koeficijent sličnosti** grafova G_1 i G_2. Jasno je da za tako definiran koeficijent sličnosti vrijedi $0 < S(G_1, G_2) \leq 1$ i $S(G_1, G_2) = 1 \iff G_1 \simeq G_2$.

Općenito, težina $w(u, v)$ predstavlja sličnost vrhova $u \in V_1$ i $v \in V_2$, a jedan praktični problem koji često želimo riješiti jest klasifikacija ulaznog grafa pronalaskom najsličnijeg među referentnim grafovima. Međutim, postoji poteškoća pri rješavanju tog problema o kojoj govori sljedeći teorem.

Teorem 3.1. *Pronalazak optimalnog sparivanja je MAX $\mathcal{SNP}$-hard.*

Dokaz. Problem neuređene edit udaljenosti[1][2] stabala je MAX $\mathcal{SNP}$-hard[3] i pokazano je,[4] analogno kao i u uređenom slučaju,[5] da edit udaljenost δ stabala $T_1 = (V_1, E_1)$ i $T_2 = (V_2, E_2)$ zadovoljava

$$\gamma(M) := \sum_{u\in I_1} \gamma(l(u), \epsilon) + \sum_{v\in I_2} \gamma(\epsilon, l(v)) + \sum_{(u,v)\in M} \gamma(l(u), l(v))$$

$$\delta(T_1, T_2) = \min_{M \text{ sparivanje}} \gamma(M)$$

gdje su $I_1 \subseteq V_1$, $I_2 \subseteq V_2$ skupovi vrhova koji se ne pojavljuju u M, $\gamma(a, b)$ cijena zamjene oznake a oznakom b, $\gamma(a, \epsilon)$, $\gamma(\epsilon, a)$ redom cijene uklanjanja i umetanja vrha s oznakom a i $l(v)$ oznaka vrha v. Stoga je jasno da ukoliko uzmemo $w(u, v) := \gamma(l(u), \epsilon)+\gamma(\epsilon, l(v))-\gamma(l(u), l(v))$, optimalno sparivanje M usmjerenih acikličkih grafova T_1 i T_2 će zadovoljavati $\delta(T_1, T_2) = \gamma(M)$. □

[1] Ne uzimamo u obzir grafove bez vrhova

4 Sparivanje kao problem cjelobrojnog programiranja

S obzirom na teorem 3.1, ne možemo se nadati pronalasku polinomnog algoritma za problem optimalnog sparivanja pa ga je opravdano pokušati riješiti metodama cjelobrojnog programiranja.[8][7]

Neka su $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$ usmjereni aciklički grafovi, $w : V_1 \times V_2 \to \mathrm{Z}^+$ težinska funkcija i $x_{u,v}$ indikatorska varijabla:

$$x_{u,v} = \begin{cases} 1, & (u,v) \text{ je u sparivanju;} \\ 0, & \text{inače.} \end{cases}$$

Nadalje, s $\mathcal{I}$ označimo skup **nekompatibilnih** parova, tj. $\mathcal{I}$ sadrži $\{(u,v),(u',v')\}$ ako i samo ako nije zadovoljen uvjet $u \leq u' \iff v \leq v'$. Problem pronalaska optimalnog sparivanja tada možemo formulirati na sljedeći način:

$$\max \sum_{u \in V_1} \sum_{v \in V_2} w(u,v) x_{u,v} \tag{I1}$$

$$\text{t.d.} \sum_{v \in V_2}^{V_2} x_{u,v} \leq 1 \qquad \forall u \in V_1 \tag{I2}$$

$$\sum_{u \in V_1} x_{u,v} \leq 1 \qquad \forall v \in V_2 \tag{I3}$$

$$x_{u,v} + x_{u',v'} \leq 1 \qquad \forall \{(u,v),(u',v')\} \in \mathcal{I} \tag{I4}$$

$$x_{u,v} \in \{0,1\} \tag{I5}$$

Koristeći tu ILP formulaciju i IBM CPLEX nije moguće izračunati optimalno sparivanje grafova s više od ∼200 vrhova.[7] Stoga se okrećemo rješavanju relaksacije tog problema.

4.1 Relaksacija cjelobrojnog programa

Prvo uočimo da je zahtjev (S2) iz definicije 3.1, a time i skup nekompatibilnih parova, moguće rastaviti na sljedeći način:

(C1) $(x < x' \text{ ili } x' < x) \iff (y < y' \text{ ili } y' < y)$ za sve $(x,y),(x',y') \in M$

(C2) $[(x < x' \text{ ili } x' < x) \iff (y < y' \text{ ili } y' < y)] \implies [x < x' \iff y < y']$ za sve $(x,y),(x',y') \in M$

To jest, usporedivi vrhovi moraju biti spareni s usporedivim vrhovima (C1), i njihov redoslijed ne smije biti promjenjen (C2). Osim što ćemo zamijeniti uvjet integralnosti s $x_{u,v} \in [0,1]$, umjesto uvjeta kompatibilnosti parova I4, po potrebi ćemo dodavati uvjete oblika

$$\sum_{(u,v) \in Q} x_{u,v} \leq 1$$

gdje je Q skup bridova takav da niti jedan njegov dvočlani podskup ne zadovoljava zahtjev (S2). Takvi uvjeti su jači i osiguravaju manji *integrality gap* od uvjeta kompatibilnosti parova.

Jedna klasa skupova bridova takvih da niti jedan par ne zadovoljava uvjet (C1) se sastoji od skupova oblika $P \times A$ gdje je P put u jednom grafu a A antilanac u drugom. U slučaju kada su G_1 i G_2 stabla, problem pronalaska skupa iz te klase koji maksimizira ukupnu težinu $\sum_{u \in P} \sum_{v \in A} x^*_{u,v}$ moguće je pronaći u $O(|V_1||V_2|)$, o čemu govori sljedeća propozicija.

Propozicija 4.1. *[7] Neka su $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ stabla, $P = (u_1, ..., u_n)$ put u G_1, x^* frakcionalno rješenje. Za $v \in V_2$ označimo $w_P(v) := \sum_{u \in V_1} x^*_{u,v}$ i s $\alpha_P(G_2)$ težinu antilanca $A \subseteq V_2$ koji maksimizira ukupnu težinu $\sum_{v \in A} w_P(v)$. Vrijedi*

$$w_{(u_1,...,u_n)}(v) = w_{(u_1,...,u_{n-1})}(v) + x^*_{u_n,v}$$

$$\alpha_P(G_2) = \max \begin{cases} \sum\limits_{w \in c(v)} \alpha_P(\tau(w)) \\ w_P(v) \end{cases}$$

U općenitom slučaju, problem pronalaska puta u G_1 i antilanca u G_2 koji maksimiziraju ukupnu težinu je $\mathcal{NP}$-hard.[7] Pomoću teorema 2.2 je moguće efikasno pronaći antilanac maksimalne težine u proizvoljnom usmjerenom acikličkom grafu, ali broj puteva u usmjerenom acikličkom grafu općenito eksponencijalano ovisi o broju vrhova. Iz tog razloga, u praktične svrhe napustimo problem pronalaska maksimalnog skupa u toj klasi i ograničimo broj puteva koje uzimamo u obzir a time i broj antilanaca koje tražimo prilikom separacije. Najjednostavniji način da to učinimo je da pohlepnim izborom maksimalnih puteva pokrijemo sve parove $(u_1, u_2) \in V_1 \times V_1$, $u_1 < u_2$. Time slabimo relaksaciju, ali u praksi neznatno zbog velikog preklapanja puteva.

Podatci: $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$, frakcionalno rješenje x^*
Rezultat: Skup H (nekih) hiperravnina koje separiraju x^* i optimalno rješenje. Ako algoritam vrati $H = \emptyset$, onda x^* zadovoljava I4 za uvjet (C1).

za svaki $(u, v) \in V_1 \times V_1,\ u < v$ **čini**
  $M[u][v] \leftarrow false$;
kraj
$H \leftarrow \emptyset$;
za svaki $(u, v) \in V_1 \times V_1,\ u < v$ **čini**
  ako $M[u][v] = false$ **onda**
    $P \leftarrow$ proizvoljan maksimalan put u G_1 koji sadrži u i v;
    $A \leftarrow x'_j$-maksimalan antilanac u G_2;
    ako $\sum\limits_{(u,v) \in P \times A} x^*_{u,v} > 1$ **onda**
      $H \leftarrow H \cup \left\{ \sum\limits_{(u,v) \in P \times A} x_{u,v} \leq 1 \right\}$;
    kraj
    za svaki $(u, v) \in P \times P,\ u < v$ **čini**
      $M[u][v] \leftarrow true$;
    kraj
  kraj
kraj

Algoritam 2: Separacija

Propozicija 4.2. *[7] Neka su G_1 i G_2 usmjereni aciklički grafovi, $|r(G_1)| = 1$, x^* frakcionalno rješenje, $r \in r(G_1)$, $l_j \in l(G_2)$. Za proizvoljne $u \in V_1$ i $v \in V_2$ označimo s $D[u][v]$ težinu x^*-maksimalnog skupa $Q \subseteq \{w : r \leq w \leq u\} \times \{w : v \leq w \leq l_j\}$ takvog da niti jedan njegov dvočlani podskup ne zadovoljava (C2). Vrijedi*

$$D[u][v] = x^*_{u,v} + \max \begin{cases} \max\limits_{w \in p(u)} D[w][v] \\ \max\limits_{w \in c(v)} D[u][w] \end{cases}$$

Te dvije klase ne pokrivaju sve skupove u parovima nekompatibilnih bridova, ali su nam dovoljne da u praksi pronađemo dobra cjelobrojna rješenja.

4.2 Dobivanje cjelobrojnog rješenja

Prvo uočimo da postoji vrlo jednostavan aproksimacijski algoritam za pronalazak sparivanja.

Podatci: $G_1 = (V_1, E_1), G_2 = (V_2, E_2), w : V_1 \times V_2 \to \mathrm{Z}^+$
Rezultat: $M \subseteq V_1 \times V_2$ sparivanje grafova G_1 i G_2

$M \leftarrow \emptyset$;
$P \leftarrow V_1 \times V_2$ sortiran silazno po $w(u, v)$;
za svaki $(u, v) \in P$ **čini**
  ako $u < u' \iff v < v'$ *za sve* $(u', v') \in M$ **onda**
    $M \leftarrow M \cup \{(u, v)\}$;
  kraj
kraj

Algoritam 3: Pohlepno sparivanje

Težina pronađenog sparivanja je očigledno barem $\frac{\max_{M \text{ sparivanje}} w(M)}{\min(|V_1|,|V_2|)}$ jer pronađeno sparivanje sadrži najteži brid. Ta težina se postiže prilikom sparivanja dva puta s težinom 1 na svim bridovima osim brida između korijena i lista na kojem je težina $1 + \epsilon$. Međutim, kao što ćemo vidjeti u poglavlju 6, u praksi daje daleko bolje rezultate.

Pohlepni algoritam nam je koristan ne samo kao donja međa za metodu grananja i ograničavanja nego i kao *sanity check*, jer je svaki sofisticiraniji algoritam koji u praksi ne producira bolja rješenja od tako jednostavnog i efikasnog algoritma zapravo besmislen.

Osim metode grananja i ograničavanja, cjelobrojno rješenje možemo dobiti zaokruživanjem optimalnog frakcionalnog rješenja x^*. Jedan način zaokruživanja je da brid koji odgovara varijabli $x_{u,v}$ odaberemo s vjerojatnošću $x^*_{u,v}$ i postupimo pohlepno, slično kao i u algoritmu 3, kako bi smo dobili dopustivo rješenje.

Alternativno, nakon pronalaska optimalnog rješenja relaksacije, možemo uvesti nelinearne uvjete integralnosti oblika $x_{u,v}(1 - x_{u,v}) = 0$ i metodama nelinearnog programiranja pronaći (ne nužno optimalno) rješenje.[7] Takav pristup, zajedno s jakom relaksacijom, daje dobre eksperimentalne rezultate.

5 Posebni slučajevi sparivanja

U nekim posebnim slučajevima, problem pronalaska optimalnog sparivanja je moguće efikasno riješiti pomoću dinamičkog programiranja, o čemu govore sljedeće dvije propozicije.

Propozicija 5.1. *Neka su $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$ stabla, $V_1 = \{u_1, ..., u_n\}$, $u_1 < ... < u_n$, $V_2 = \{v_1, ..., v_m\}$, $v_1 < ... < v_m$ i $w : V_1 \times V_2 \to \mathrm{Z}^+$ težinska funkcija. Optimalno sparivanje grafova G_1 i G_2 moguće je tada izračunati u $\mathcal{O}(nm)$ i vrijedi:*

$$\delta(G_1, G_2) = \begin{cases} \max \begin{cases} w(u_1, v_1) + \delta(\tau(u_2), \tau(v_2)) \\ \delta(\tau(u_2), G_2) \\ \delta(G_1, \tau(v_2)) \end{cases} & |V_1| \neq 1 \text{ i } |V_2| \neq 1 \\ \max\limits_{v \in V_2} w(u_1, v) & |V_1| = 1 \\ \max\limits_{u \in V_1} w(u, v_1) & |V_2| = 1 \end{cases}$$

Propozicija 5.2. *Neka su $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$ stabla, $V_1 = \{u_1, ..., u_n\}$, $u_1 < ... < u_n$, $r_2 \in r(G_2)$ i $w : V_1 \times V_2 \to \mathrm{Z}^+$ težinska funkcija. Optimalno sparivanje grafova G_1 i G_2 moguće je tada izračunati u $\mathcal{O}(nm)$ i vrijedi:*[2]

$$\delta(G_1, G_2) = \begin{cases} \max \begin{cases} w(u_1, r_2) + \max\limits_{k \in c(r_2)} \delta(\tau(u_2), \tau(k)) \\ \max\limits_{k \in c(r_2)} \delta(G_1, \tau(k)) \\ \delta(\tau(u_2), G_2) \end{cases} & |V_1| \neq 1 \text{ i } |V_2| \neq 1 \\ \max \begin{cases} \max\limits_{k \in c(r_2)} \delta(G_1, \tau(k)) \\ w(u_1, r_2) \end{cases} & |V_1| = 1 \\ \max\limits_{k \in V_1} w(k, r_2) & |V_2| = 1 \end{cases}$$

Teorem 3.1 govori da je problem neuređene edit udaljenosti stabala poseban slučaj optimalnog sparivanja. S obzirom da za problem uređene edit udaljenosti postoji $\mathcal{O}(n^2m^2)$ algoritam,[1][5] postavlja se prirodno pitanje je li neuređena edit udaljenost stabala jednaka najmanjoj među uređenim edit udaljenostima uređenih stabala koje dobijemo permutirajući djecu nekih vrhova stabala koje uspoređujemo. Odgovor je ne. Shasha *et al.*[6] su pronašli protuprimjer. Međutim, postoje polinomni algoritmi za rješavanje tog problema ukoliko je, primjerice, broj razgranatih vrhova konstantan. Tada problem možemo svesti na prethodna dva slučaja na sljedeći način:

Teorem 5.1. *[19] Neka su $F_1 = (V_1, E_1)$ i $F_2 = (V_2, E_2)$ šume s t_1 odnosno t_2 razgranatih vrhova i $w : V_1 \times V_2 \to \mathrm{Z}^+$ težinska funkcija. Optimalno sparivanje grafova G_1 i G_2 moguće je tada izračunati u $O(n^{3(t_1+t_2)})$ i vrijedi:*

$$\delta(F_1, F_2) = \max \begin{cases} \max\limits_{u \in \{u \in r(F_1) : \exists b(u)\}} \delta(F_1 \setminus \pi(u), F_2) \\ \max\limits_{v \in \{v \in r(F_2) : \exists b(v)\}} \delta(F_1, F_2 \setminus \pi(v)) \\ \sigma(r(F_1), r(F_2), w') \end{cases} \tag{1}$$

$$w'(u, v) = \max \begin{cases} \max\limits_{s \in c(b(u))} \delta(\pi(u) \cup \tau(s), \tau(v)) \\ \max\limits_{s \in c(b(v))} \delta(\tau(u), \pi(v) \cup \tau(s)) \\ \delta(\tau(u) \setminus \pi(u), \tau(v) \setminus \pi(v)) + \delta(\pi(u), \pi(v)) \end{cases} \tag{2}$$

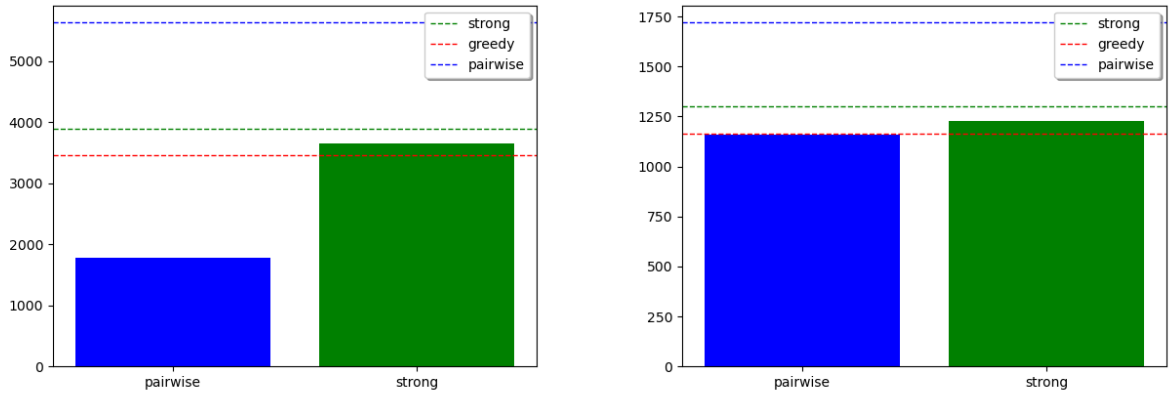
[2] $\max\limits_{x \in \emptyset} f(x) := 0$

6 Implementacija

C++ implementaciju algoritama moguće je pronaći na `https://github.com/krofna/dagmatch`. Implementacija koristi činjenicu da je neke od najskupljih operacija poput algoritma 2 moguće paralelizirati te je stoga praktična na modernim računalima.

Slika 1 prikazuje rezultate pokretanja implementacije algoritama za sparivanje na paru stabala (lijevo) i paru usmjerenih acikličkih grafova (desno) koji se pojavljuju u primjeni. Plave, zelene i crvene isprekidane crte predstavljaju redom težine optimalnog frakcionalnog rješenja s uvjetima kompatibilnosti parova, težine frakcionalnog rješenja s jačim uvjetima iz poglavlja 4.1 i težine cjelobrojnog pohlepnog rješenja dobivenog pomoću algoritma 3. Plavi i zeleni stupci predstavljaju cjelobrojna rješenja pronađena metodom nelinearne optimizacije koristeći pronađeno frakcionalno rješenje kao početnu aproksimaciju. Vidimo da je na tim primjerima optimalno frakcionalno rješenje s uvjetima kompatibilnosti parova beskorisno kao početna aproksimacija za nelinearni rješavač koji u tom slučaju pronalazi lošija rješenja od pohlepnog algoritma dok je frakcionalno rješenje s jačim uvjetima dovoljno dobra početna aproksimacija u svrhu pronalaska boljeg cjelobrojnog rješenja.

Slika 1: Rezultati implementacije



Literatura

[1] Akutsu, Tatsuya. (2010). Tree Edit Distance Problems: Algorithms and Applications to Bioinformatics. IEICE Transactions. 93-D. 208-218.

[2] Bille, P. (2005). A survey on tree edit distance and related problems. Theoretical Computer Science, 337(1-3), 217–239

[3] Zhang, K., Jiang, T. (1994). Some MAX SNP-hard results concerning unordered labeled trees. Information Processing Letters, 49(5), 249–254

[4] Zhang, K., Statman, R., Shasha, D. (1992). On the editing distance between unordered labeled trees. Information Processing Letters, 42(3), 133–139.

[5] Zhang, K., Shasha, D. (1989). Simple Fast Algorithms for the Editing Distance between Trees and Related Problems. SIAM Journal on Computing, 18(6), 1245–1262.

[6] Shasha, D., Wang, J. T.-L., Kaizhong Zhang, Shih, F. Y. (1994). Exact and approximate algorithms for unordered tree matching. IEEE Transactions on Systems, Man, and Cybernetics, 24(4), 668–678.

[7] V. Hoan Do, M. Blažević, P. Monteagudo, L. Borozan, K. Elbassioni, S. Laue, F. Rojas Ringeling, D. Matijević, S. Canzar, Dynamic pseudo-time warping of complex single-cell trajectories, 23nd Annual International Conference on Research in Computational Molecular Biology, The George Washington University, 2019, 294-297

[8] Böcker S., Canzar S., Klau G.W. (2013) The Generalized Robinson-Foulds Metric. In: Darling A., Stoye J. (eds) Algorithms in Bioinformatics. WABI 2013. Lecture Notes in Computer Science, vol 8126. Springer, Berlin, Heidelberg

[9] L.R. Ford and D.R. Fulkerson, Flows in Networks, Princeton Univ. Press, Princeton NJ., 1962.

[10] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. 1993. Network Flows: Theory, Algorithms, and Applications. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

[11] Kramer, M., Dutkowski, J., Yu, M., Bafna, V., Ideker, T. (2014). Inferring gene ontologies from pairwise similarity data. Bioinformatics, 30(12), i34–i42.

[12] A. Schrijver, Combinatorial Optimization — Polyhedra and Efficiency, Springer, Berlin, 2003.

[13] Frank, A. (1999). Finding Minimum Generators of Path Systems. Journal of Combinatorial Theory, Series B, 75(2), 237–244.

[14] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. Introduction to Algorithms, Third Edition (3rd ed.). The MIT Press.

[15] Robinson, D. F., Foulds, L. R. (1981). Comparison of phylogenetic trees. Mathematical Biosciences, 53(1-2), 131–147.

[16] Bogdanowicz, Damian, and Krzysztof Giaro. On a matching distance between rooted phylogenetic trees, International Journal of Applied Mathematics and Computer Science 23, 3: 669-684

[17] Alexander Schrijver. 1986. Theory of Linear and Integer Programming. John Wiley Sons, Inc., New York, NY, USA.

[18] Bogdanowicz, D., Giaro, K., Wróbel, B. (2012). TreeCmp: Comparison of Trees in Polynomial Time. Evolutionary Bioinformatics, 8, EBO.S9657.

[19] Halldórsson, M. M., Tanaka, K. (1996). Approximation and special cases of common subtrees and editing distance. Lecture Notes in Computer Science, 75–84.

[20] Hattori M., Okuno Y., Goto S., Kanehisa M. Heuristics for chemical compound matching. Genome Inform 2003, 14:144–153.

[21] Zager, L. A., Verghese, G. C. (2008). Graph similarity scoring and matching. Applied Mathematics Letters, 21(1), 86–94.

[22] Vijay V. Vazirani. 2001. Approximation algorithms. Springer-Verlag, Berlin, Heidelberg.