Line arrangement problem

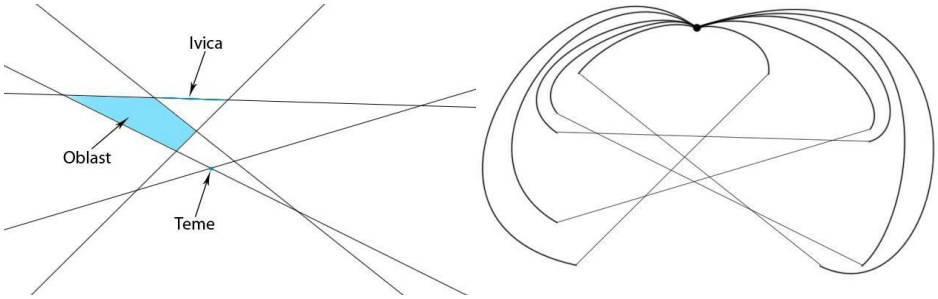
Djordje Jovanovic

Mentor: prof. Dragan Urošević

Sadržaj

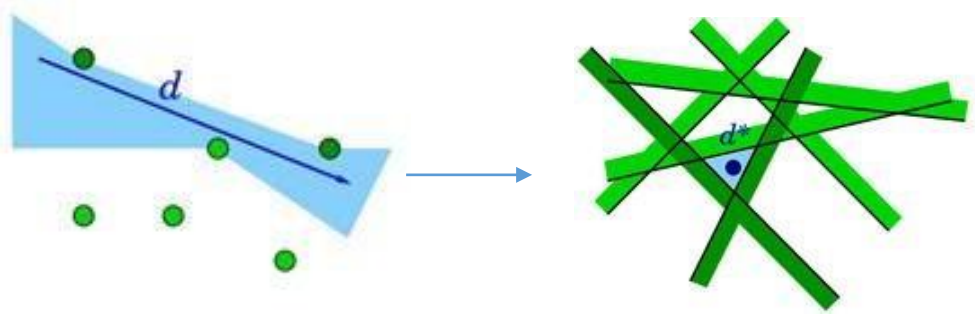
1 Uvod

Razmatramo skup linija H u ravni. Te linije dele ravan na određeni način i to je raspored linija. Presekom linija formiraju se temena, ivice se formiraju između parova uzastopnih temena, a oblasti između linija. Oblasti i ivice ne moraju biti ograničene, međutim ono što možemo uraditi je da dodamo tačku u beskonačnosti, i sve neograničene ivice povežemo sa tom tačkom i tako dobijemo ispravan planarni graf (Ilustracija 1: Osnovni elementi rasporeda i dodavanje tačke u beskonačnosti. Postoji više načina da se taj raspored predstavi, a jedan primer je dvostruko povezana lista ivica (DCEL).



Ilustracija 1: Osnovni elementi rasporeda i dodavanje tačke u beskonačnosti

Raspored linija u ravni je jedna najizučavanijih struktura u kompjuterskoj geometriji. Primena ove strukture postoji u rešenjima mnogih problema kao što su graf vidljivosti, planiranje kretanja, HSR (Hidden Surface Removal) i još mnogi drugi, naročitu u kombinaciji sa dualnom ravni gde se, umesto kolekcijom linija, radi sa kolekcijom tačaka. U dualnoj ravni, oblast unutar ćelije predstavlja kolekciju linija koje imaju iste tačke iznad i ispod (Ilustracija 2: Oblast rasporeda linija predstavlja skup linija koje imaju iste tačke ispod i iznad.



Ilustracija 2: Oblast rasporeda linija predstavlja skup linija koje imaju iste tačke ispod i iznad

Za raspored kažemo da je jednostavan ako se svaki par linija preseca u nekoj tački i ako se nikoje tri linije ne presecaju u istoj tački. U narednom poglavlju će biti predstavljena svojstva složenosti jednostavnog rasporeda linija u ravni tj. broj temena, ivica i oblasti. Zatim će uslediti pregled nekih od jednostavnijih metoda određivanja rasporeda linija. Na kraju će biti opisan algoritam koji je i predmet ovog rada.

2 Složenost rasporeda linija

Složenost je povezana sa brojem temena, ivica i oblasti. Kako razmatramo jednostavan raspored, pokazaćemo da su te količine $\Theta(n^2)$.

Lema 1: Neka je $A(H)$ jednostavan raspored od n linija H u ravni. Tada važi:

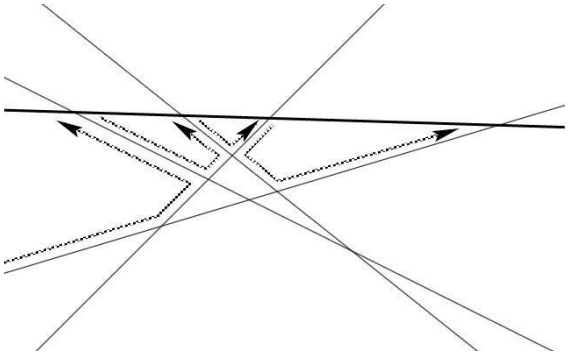
i. Broj temena (ne računajući teme u beskonačnosti) u $A(H)$ iznosi $\binom{n}{2}$
ii. Broj ivica u $A(H)$ iznosi n^2
iii. Broj oblasti u $A(H)$ iznosi $\binom{n}{2} + n + 1$

Dokaz: Broj temena je jasan na osnovu činjenice da se svaki par linija preseca u jednoj tački, jer nema paralelnih linija. Da bismo dokazali da je broj ivica n^2 koristimo indukciju. Baza za $n = 1$ je trivijalna, jedna linija jedna ivica. Kada dodamo liniju u raspored od $n - 1$ linija u kome imamo $(n-1)^2$ ivica, tačno jedna ivica na svakoj od liija se deli, što nam daje $(n-1)$ novih ivica. Takođe nova linija je podeljena na n ivica presecima sa svakom od $n - 1$ linija. Ukupno imamo $(n-1)^2 + (n-1) + n = n^2$ ivica. Broj oblasti sledi iz Ojlerove formule $v - e + f = 2$, gde je v broj temena, e broj ivica i f broj oblasti. Kako smo dodali tačku u beskonačnosti, $v = 1 + \binom{n}{2}$ i $e = n^2$. Pa je broj oblasti: $f = 2 - v + e = 2 - \left(1 + \binom{n}{2}\right) + n^2 = 2 - \left(1 + \frac{n(n-1)}{2}\right) + n^2 = 1 + \frac{n^2}{2} + \frac{n}{2} = 1 + \frac{n(n-1)}{2} + n = \binom{n}{2} + n + 1$.

Ove formule se mogu generalizovati i za više dimenzije. Složenost rasporeda n hiperravni u $\mathbb{R}^d$ je $\Theta(n^d)$. Stoga su ove strukture praktične samo za relativno male dimenzije kada n nije previše veliko.

3 Konstrukcija rasporeda

Kako je složenost samog raspored kvadratna, nema potrebe tražiti algoritam sa složenošću manjom od kvadratne. Najjednostavniji algoritam je inkrementalni algoritam. Linije se dodaju jedna po jedna u raspored (npr. DCEL struktura). U i-tom dodavanju, prvo treba pronaći oblast koja je najdalje levo, kroz koju linija prolazi. Te oblasti su ograničene ivicama koje su poluprave, kojih ima $O(i)$, pa je složenost ovog koraka $O(i)$. Zatim idemo po obodu oblasti, ivicu po ivicu, dok ne naiđemo na onu sa kojom se nova linija preseca. Ta ivica se podeli na dva dela, doda se teme u preseku i nastavi se dalje samo sad sa druge strane ivice (Ilustracija 3: Inkrementalni algoritam. Ubacivanje linije.).



Ilustracija 3: Inkrementalni algoritam. Ubacivanje linije.

Imamo n koraka. Kako smo sigurni da svaki korak ima linearnu složenost? Zone teorema nam dokazuje da je složenost koraka $O(i)$.

Za neku liniju l i raspored linija $A(H)$, zona $Z_{A(H)}(l)$ linije l u rasporedu $A(H)$ je skup svih strana tj. oblasti iz $A(H)$ koje preseca l.

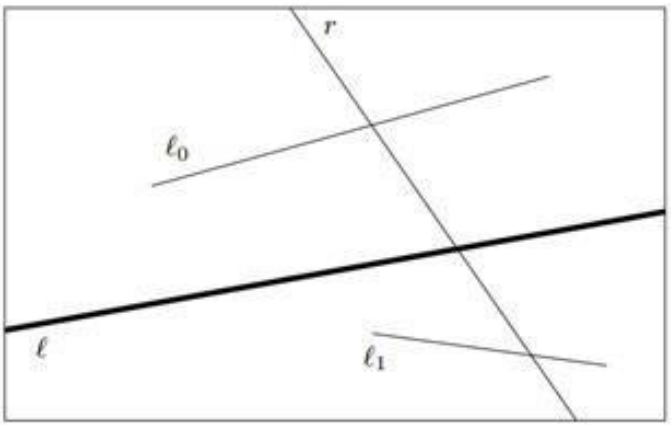
Zone teorema: Za dati raspored $A(H)$ od n linija iz $\mathbb{R}^2$ i linije l (koja ne mora biti neka iz skupa H), broj ivica na obodu svih oblasti zone $Z_{A(H)}(l)$ je najviše $6n$.

Dokaz: Pretpostavimo da je l horizontalna i da nijedna linija iz H nije horizontalna. Prva pretpostavka se može postići rotiranjem svih linija, a druga tako što odlučimo da je levi kraj horizontalne ivice viši od desnog kraja.

Za svaku oblast iz $Z_{A(H)}(l)$ podelimo ivice koje je ograničavaju najvišom i najnižom tačkom.
Indukcijom ćemo dokazati da ivica, koje ograničavaju oblast sa leve strane, ima najviše $3n$. Simetrično će važiti i za ivice koje ograničavaju oblast sa desne strane.

Za $n = 1$, postoji tačno jedna ivica u $Z_{A(H)}(l)$ koja ograničava neku oblast sa leve strane, i $1 \leq 3n = 3$. Pretpostavimo da je tvrdnja tačna za $n - 1$.

Razmotrićemo liniju r iz H koja najdalje desno seče liniju l i raspored $A(H\backslash\{r\})$ (Ilustracija 4). U zoni $Z_{A(H\backslash\{r\})}(l)$ imamo najviše $3n - 3$ ivica koje ograničavaju sa leve strane. Dodavanjem linije r najviše se mogu dodati tri nove ivice koje ograničavaju sa leve strane. Linija r preseca najviše dve ivice (koje ograničavaju sa leve strane i neka su to l_0 i l_1) oblasti koja je najdalje desno u zoni, i samim tim svaka je podeljena na dva dela. Sama linija r doprinosi još jednu ivicu, i ta linija ne može doprineti nijednoj drugoj oblasti pored najdesnije. Levo od r može doprineti samo broju ivica koje ograničavaju sa desne strane, a desno od r sve ostale oblasti su zaklonjene od l ili linijom l_0 ili l_1. Pa je tako ukupni broj ivica u $Z_{A(H)}(l)$ najviše $3 + 3n - 3 = 3n$.



Ilustracija 4

Memorijska složenost inkrementalnog algoritma je isto $O(n^2)$jer se cela struktura mora čuvati. Klasično prebrisavanje vertikalnom linijom bi zahtevalo samo $O(n)$ prostora, međutim vremenska složenost bi bila $O(n^2 \log n)$. Ukratko, postupak je da se sortiraju linije po nagibu, u toku prebrisavanja čuvamo trenutne ivice koje su presečene tom vertikalnom liniju, a u priority queue-u čuvamo presečna temena između uzastopnih linija(u trenutnom preseku). Da bi smo

pronašli sledeće teme, potrebno nam je $O(\log n)$. vremena, i kako ima ukupno $O(n^2)$ koraka to je složenost veća od optimalne koju tražimo.

4 Algoritam

Sledi opis algoritma koji je osmislio Edelsbrunner [1]. Neka imamo n linija u ravni.
Pretpostavimo da se nikoje tri linije ne presecaju istoj tački, i da nijedna linija nije vertikalna, kao i da nema paralelnih linija. Svaka linija preseca $n - 1$ liniju i samim tim je podeljena na n ivica. Ako koristimo vertikalnu pokretnu liniju za prebrisavanje, u suštini mi sortiramo n^2 presečnih tačaka. Topološkim prebrisavanjem linija preseka nije prava linija. Ideja je da se koristi zakrivljena linija (topološka linija) sa određenim svojstvima koja simuliraju vertikalnu liniju. Koristeći takvu liniju prilikom prebrisavanja, složenost je $O(n^2)$ vremena i svega $O(n)$ prsotora. Topološka linija je monotona linija u y- smeru, što znači da je bilo koja horizontalna linija preseca tačno jednom. Pored toga topološka linija preseca i svaku od n linija tačno jednom. Predstavljena je nizom ivica koje sadrže tačku preseka sa svakom od linija. Ova presečna linija, kao i vertikalna linija, ide od $-\infty$ do ∞ po y osi. Presek ima ista svojsvta kao i vertiklana linija po definicij. Prebrisavanje počinje se presekom koji je najdalji levo. Taj presek uključuje sve polu-beskonačne ivice i pomera se desno sve do preseka koji je nadalji desno. Koraci pomeranja se zovu elementarim koracima i pod tim podrazumevamo da je topološka linija prešla preko temena(tačke preseka) rasporeda. Da bi presek ostao topoliška linija, elementarni korak mora preći preko tačke preseka dve uzastopne ivice u trenutnom preseku, jer u suprotnom će neke linije rasporeda seći u više od jedne tačke.

4.1 Oznake i definicije

Oznake korišćene su preuzete iz originalnog rada.

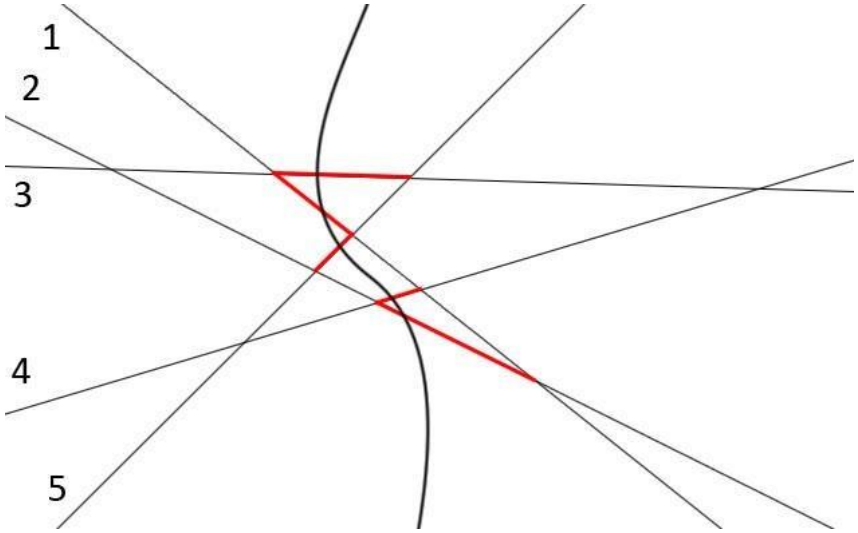
H skup n linija $\{l_1, l_2, \dots, l_n\}$ u ravni.

$A(H)$ označava raspored linija iz skupa H.

Definišemo relaciju „iznad“ između elemenata $A(H)$. Element A je iznad elementa B ako projekcije A i B na x osu imaju presek, i ako za svako x tog preseka, sve tačke A su iznad svih tačaka B. Slično se definiše i relacija „ispod“ gde je element A ispod elementa B ako projekcije A i B na x osu imaju presek, i ako za svako x tog preseka, sve tačke A su ispod svih tačaka B. Tako imamo tačno jednu oblast koja nije ispod nijedne druge oblasti koju ćemo ožnačiti sa T, i tačno jedna oblast koja nije iznad nijedne druge oblasti ožnačenu sa B. Dokaz je trivijalan.

Presek je niz ivica $(c_1, c_2, \dots, c_n)$ iz $A(H)$ takve da

i. c_1 je ivica oblasti T i c_n je ivica oblasti B, i
ii. Za svako i takvo da $1 \leq i \leq n - 1$ važi da su ivice c_i i c_{i+1} ivice oblasti R_i, i c_i je iznad R_i a i c_{i+1} ispod R_i, gde su sa R_i označene redom oblasti koje topoloska linija seče.



Ilustracija 5: Topološka linija i odgovarajući presek

Sledi opis stabala horizonta koji će biti korišćeni u ovom algoritmu kao pomoćne strukture. Stablo gornjeg horizonta se konstruiše produžavanjem ivica preseka na desno. Kada se neke dve ivice susretnu, samo ona ivica sa većim nagibom nastavlja na desno. Stablo donjeg horizonta se na sličan način konstruiše, jedina razlika je u tome da kada se dve ivice susretnu, ona ivica sa manjim nagibom nastavlja na desno.

Formalno, neka $(m_1, m_2, \dots, m_n)$ označava linije koje sadrže ivice preseka $(c_1, c_2, \dots, c_n)$, tačka p na liniji m_i pripada stablu gornjeg horizonta $T^+(C)$ preseka C ako

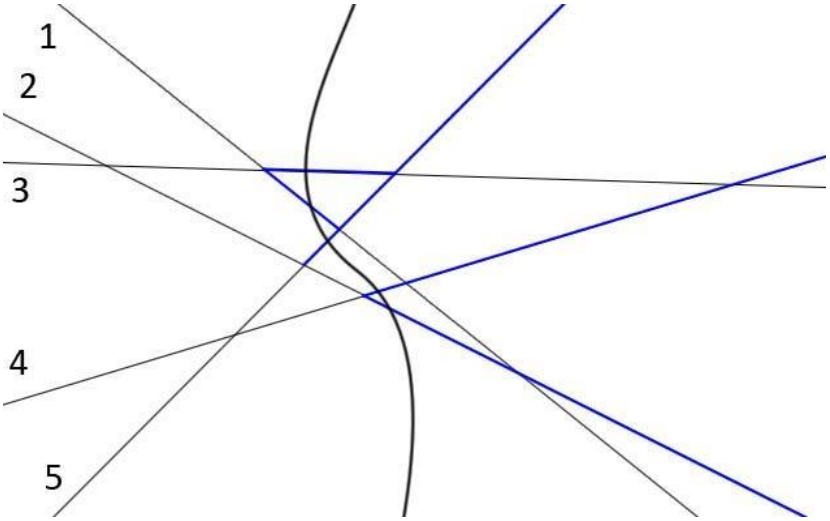
i. p je iznad svih linija m_j za $j > i$, i
ii. p je ispod svih linija m_k takvih da je $k < i$ i da im je nagib veći od nagiba linije m_i.

Na ilustraciji broj Ilustracija 6: Stablo gornjeg horizonta je prikazano stablo $T^+(C)$ za presek iz primera, a na ilustraciji broj Ilustracija 7: Stablo donjeg horizonta simetrično definisano stablo donjeg horizonta $T^-(C)$. Kao što se vidi sa slike, ivice preseka se nalaze u oba stabla, bez levog kraja.

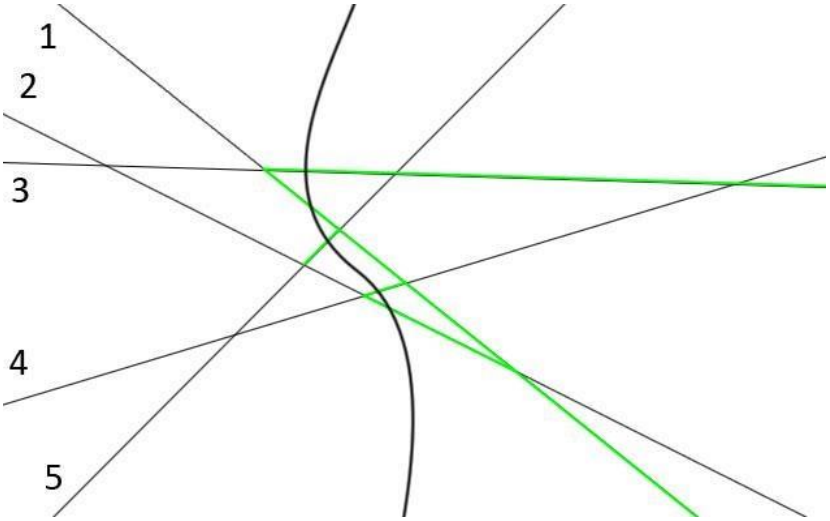
Kako se može desiti da se ovo stablo pretvori u šumu, dodaje se pomoćna vertikalna linija m_0 u $x = +\infty$ koja se u stablu gornjeg horizonta posmatra kao linija sa najvećim nagibom, a u stablu donjeg horizonta kao linija sa najmanjim nagibom.

Sa $(s_1^+, s_2^+, \dots, s_n^+)$ označićemo segmente stabla gornjeg horizonta (segmenti stabla donjeg horizonta su označeni sa minusom) koje pripadaju linijama $(m_1, m_2, \dots, m_n)$.

Važno zapažanje koje će biti od koristi kasnije, je da važi $C = T^+(C) \cap T^-(C)$, tj. svaka ivica preseka je presek odgovarajućih segmenata.



Ilustracija 6: Stablo gornjeg horizonta

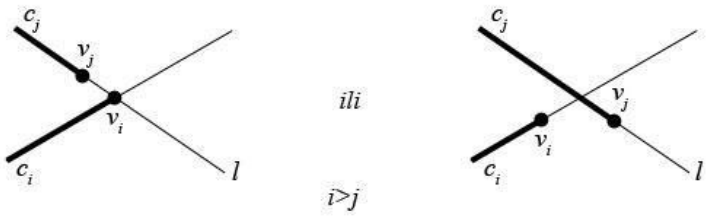


Ilustracija 7: Stablo donjeg horizonta

Pre samog opisa postupka, potrebno je još pozabaviti se pitanjem da li je uvek moguće učiniti elementarni korak. Da li uvek postoji tačka preko koje možemo preći kao što je ranije opisano.

Lema 2: Uvek će postojati dve uzastopne ivice preseka sa zajedničkim desnim krajem, osim preseka koji je nadalji desno.

Dokaz: Pretpostavimo da postoje tačke koje još uvek nisu obrađene, i da ne postoji par ivica c_k, c_{k+1} koji dele zajednički desni kraj. Neka je c_i ivica preseka čiji desni kraj se nalazi najviše levo. Neka je c_j ivica preseka na liniji l koja seče ivicu c_i u njenom desnom kraju u tački v_i (Ilustracija 8: Desni kraj ivice preseka). Desni kraj ivice c_j (v_j) je ili desno, ili levo od desnog kraja c_i (v_i). Ako je v_j desno od v_i, onda topološka linija preseca liniju l više puta. Ovo ne može biti slučaj. Ako je v_j levo od v_i, onda je v_j desni kraj koji se nalazi najviše levo što je kontradiktorno sa prvom pretpostavkom. To znači da je $v_i = v_j$ i da je desna krajnja tačka koja se nalazi najviše levo uvek elementarni korak.



Ilustracija 8: Desni kraj ivice preseka

4.2 Postupak

4.2.1 Strukture podataka:

$E[1: n]$ je niz koji sadrži jednačine linija. $E[i] = (a_i, b_i)$ ako i-ta linija l_i skupa H ima jednačinu $y = a_i * x + b_i$

$HTU[1: n]$ je niz koji predstavlja stablo gornjeg horizonta. $HTU[i]$ je par (λ_i, ρ_i) indeksa koji pokazuju na linije koje ograničavaju segment linije l_i u stablu sa leve i desne strane.

$HTL[1: n]$ predstavlja stablo donjeg horizonta i definisan je slično kao i stablo gornjeg horizonta.

I je skup celih brojeva, predstavljen kao stek. Ako se i nalazi u I, onda c_i i c_{i+1} dele zajednički desni kraj.

$M[1: n]$ je niz koji sadrži trenutni redosled indeksa koje formiraju linije $m_1, m_2, \dots, m_n$ preseka.

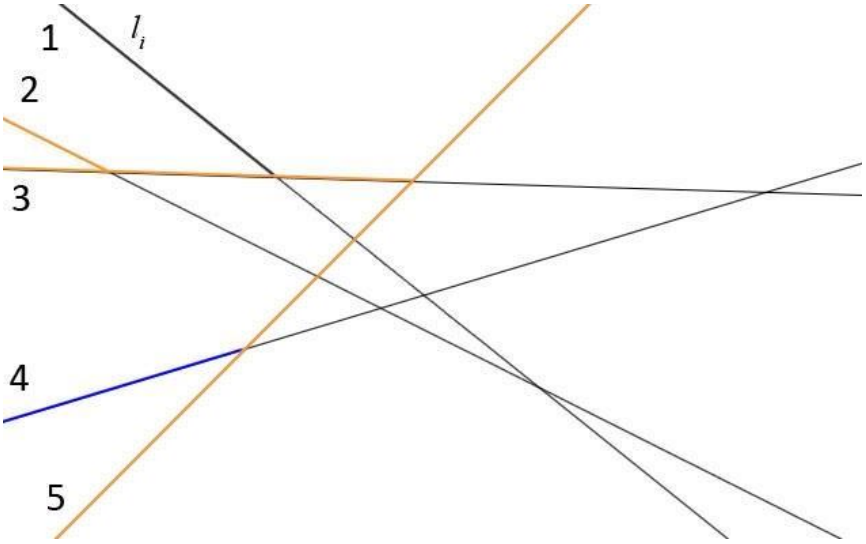
$N[1: n]$ je niz parova indeksa koje pokazuju na linije koje ograničavaju svaku ivicu preseka.

Oznake za stabla su preuzete iz originalnog rada, HTU (Horizon Tree Upper) za stablo gornjeg horizonta i HTL (Horizon Tree Lower) za stablo donjeg horizonta. Ako je segment iz HTU linije l_i najdalji levo, onda je $\lambda_i = -1$ (označava da taj segment nije ograničen sa leve strane), a ako je najdalji desno onda je $\rho_i = 0$ (označava da taj segment nije ograničen sa desne strane). Ista konvencija važi i za strukturu M.

4.2.2 Inicijalizacija:

Algoritam počinjemo formiranjem početnog stanja stabla gornjeg horizonta, za presek koji je najdalji levo. Prvo je potrebno sortirati linije skupa H po nagibu. Kao što smo već utvrdili, presek koji je najdalji levo se sastoji od ivica koje se protežu u beskonačnost sa leve strane. Sa desne strane su ograničene sa prvom linijom na koju naiđu sa većim nagibom.

Ubacujemo liniju po liniju u stablo, počevši od linije sa najvećim nagibom. Pretpostavimo da je linija l_i sledeća na redu i da su sve ostale linije sa većim nagibom ubačene u stablo. Te linije formiraju „gornji zaliv“ u koji će l_i udariti.



Ilustracija 9

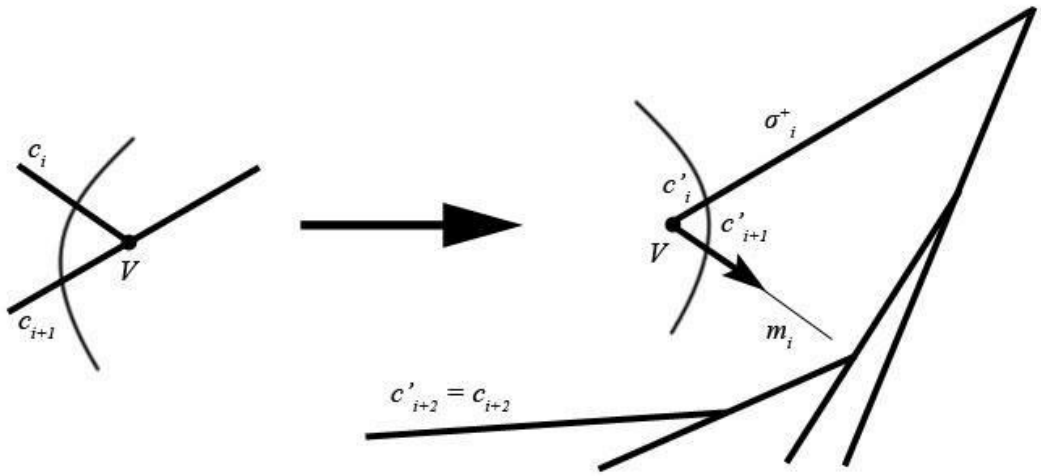
Određujemo gde će l_i udariti tako što zaliv obrađujemo u smeru obrnutom od kazaljke na satu. Prednost ovoga je što se svaka ivica koja se obradi prestaje da bude deo zaliva, izbacuje se iz njega i ne mora se više razmatrati. Kada naiđemo na ivicu koju l_i preseca, ažuriramo zaliv tako što podelimo presečenu ivicu na dva dela i dodamo novonastalu ivicu. Novonastalu ivicu takođe dodajemo i u HTU strukturu. Na Ilustracija 9 vidimo jedan primer gde je l_i linija sa rednim brojem 1, zaliv čine linije 2, 3 i 5 dok linija 4 nije deo njega. Složenost formiranja stabla je $O(n)$, ukoliko su linije već sortirane, u suprotnom nam je za sortiranje potrebno $O(n \log n)$. Početno stanje stabla donjeg horizonta se formira na sličan način.

Strukture M i N predstavljaju ivice preseka koje se mogu dobiti od segmenta stabla HTU i HTL tako što jednostavno izaberemo kraći segment svake linije. Kada dobijemo ivice preseka, skup I se može trivijalno dobiti. I ovaj deo inicijalizacije ima linearnu složenost $O(n)$ pa je cela inicijalizacija linearna osim ukoliko je potrebno sortiranje.

4.2.3 Elementarni korak:

Sa steka I preuzmemo sledeći indeks i. Znamo da c_i i c_{i+1} dele zajedniči desni kraj V, i samim tim možemo uraditi elementarni korak preko V(Ilustracija 10: Ažuriranje gornjeg stabla horizonta). Sa s i σ ćemo označiti segmente stabla pre i posle elementarnog koraka. Stablo gornjeg horizonta HTU se menja na sledeći način. σ_i^+ se lako dobije od s_{i+1}^+ . Deo s_{i+1}^+ levo od V se odseče i dobili smo σ_i^+ . Dobijanje σ_{i+1}^+nije toliko lako i zahteva kompleksniju analizu.

Kao i kod inicijalizacije, potrebno je odrediti gde će nastavak linije m_i nadesno udariti zaliv stabla ograničenog sa c_{i+1} i c_{i+2}. Obrađujemo zaliv tako što počnemo od c_{i+2} i nastavljamo u smeru suprotnom od kazaljke na satu sve dok ne naiđemo na segment koji se preseca sa m_i. Linija m_i mora preseći neki segmet tog zaliva, jer je c_{i+2} ispod c_i samim tim ispod m_i i zaliv je povezan sa σ_i^+ koji je iznad m_i.



Ilustracija 10: Ažuriranje gornjeg stabla horizonta

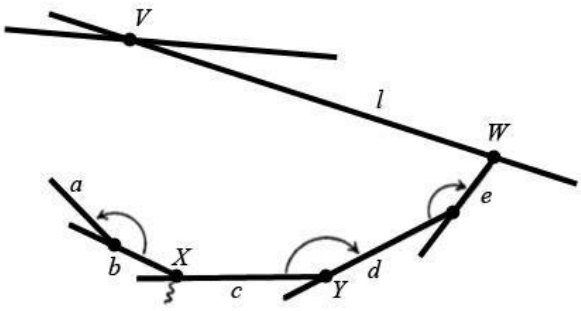
Stablo se zatim može ažurirati u $O(1)$ vremenu. Celokupna složenost ažuriranja stabla je $O(n)$. HTL stablo se na sličan način ažurira.

Na osnovu dva stabla, trivijalno se ažurira i sam presek (uzimajući kraće segmente), kao i stek I, gde je potrebno proveriti da li dve nove ivice imaju zajedničke desne krajeve. Složenost elementarnog koraka je linerna u najgorem slučaju.

Elementarnih koraka ima $O(n^2)$, i svaki korak je u najgorem slučaju $O(n)$ pa je time granica $O(n^3)$, što je previše uzimajući u obzir da je $O(n^2)$ optimalno. Međutim, u sledećem odeljku ćemo videti da je složenost ipak amortizovana.

4.3 Složenost

Lema 3: Za raspored od n linija, totalni broj obrada ivica stabla gornjeg horizonta tokom izvršavanja elementarnih koraka preko tačaka koje se nalaze na liniji l iz H je $O(n)$.



Ilustracija 11

Dokaz: Posmatraćemo primer zaliva sa slike (Ilustracija 11). Elementarni korak se dešava preko tačke V na liniji l. Obrađuju se ivice na linijama a, b, c, d i e u smeru suprotnom od kazaljke na satu dok ne naidjemo na presek sa linijom l. Prebrojaćemo obrađene ivice, tokom svih elementarnih koraka na liniji l, tako što će svaka od obrađenih ivica zadužiti neku od linija, a zatim ćemo dokazati da nijedna od linija, osim linije l, ne može biti zadužena više od jednom. Linija l ima $n-1$ temena, samim tim imamo $n-1$ elementarnih koraka koji uključuju liniju l i ukupno imamo $n-1$ zaliva. Prvu i poslednju ivicu pridružujemo liniji l, tj. te ivice zadužuju l.

Svaki zaliv je konveksni niz ivica kojima nagib monotono raste. Ako neka ivica m, iz zaliva, ima nagib manji od l, ta ivica zadužiće liniju koja sadrži prethodnu ivicu, a ako je nagib veći zadužiće liniju koja sadrži sledeću ivicu. Prva i poslednja ivica, kao što smo već rekli, zadužuju liniju l. U primeru, b zadužuje a, c zadužuje d i d zadužuje e (ivice pre temena X imaju manji nagib, a ivice posle X imaju veći nagib od linije l).

Koliko puta jedna prava može biti zadužena tokom elementarnih koraka na liniji l? Razmotrimo prvo linije sa nagibom većim od l, kao što je linija d u primeru. Ivica koja je zadužila d se nalazi na liniji c. Linija koja zadužuje d mora imati nagib između l i d. Pretpostavimo da je ovo poslednji put da se d zadužuje tokom elementarnih koraka na liniji l.

Svaki zaliv ograničava oblast koja je formirana presecima poluravni ispod l sa svim poluravnima iznad linija koje slede posle l u presek. Presek linija c i d, Y i linija c i l jos uvek nisu obrađeni. To znači da u svim prethodnim koracima, linije l, c i d se pojavljuju u tom redosledu. Deo linije d, levo od Y, tokom svih tih prethodnih koraka je zbog toga zagrađen linijom c i ne može biti deo zaliva, i samim tim ne može biti zadužen od strane neke ivice. Ni deo desno od Y ne može biti zadužen. Da bi se to desilo, mora postojati linija c' koja preseca d desno od Y. Po sistemu zaduživanja c' mora presecati l ispod W. Iz toga sledi da je c' ispod l u trenutnom koraku i da c' zagrađuje tačku Y što je kontradikcija.

Analogno dokazujemo i za linije manjeg nagiba, kao što je linija a, samo sada razmatramo kada se prvi put ta linija zadužila za neku ivicu, i zašto ne može kasnije biti ponovo zadužena. Ovim je dokazana linearnost obrade svih zaliva tokom svih elementarnih koraka na određenoj liniji l.

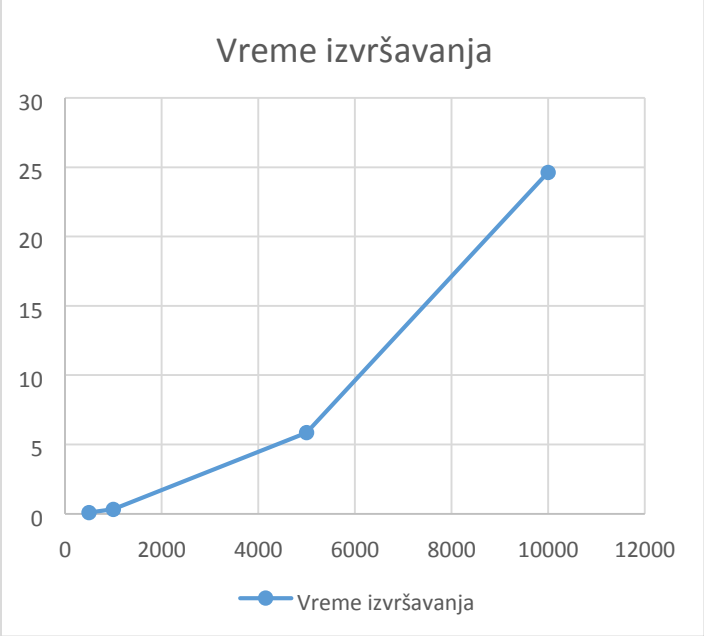
Iz ove leme sledi da je složenost ažuriranja stabla gornjeg horizonta tokom topološkog prebrisavanja $O(n^2)$. Analogno i za stablo donjeg horizonta. Samim tim je i cela složenost algoritma $O(n^2)$. U bilo kom trenutku, strukture koje se koriste su linearne, pa je potreban samo $O(n)$dodatni prostor.

4.4 Eksperimentalni rezultati

Za potrebe ovog rada implementiran je ovaj algoritam u Java programskom jeziku. Api se može koristiti u drugim aplikacijama, a takođe je i napravljena vizuelizacija algoritma korak po korak koja koristi taj api. Na grafiku (Ilustracija 12: Grafik vremena izvršavanja. Vreme je u sekundama.) su prikazana merenja vremena izvršavanja algoritma nad skupovima različitog broja linija. Linije su generisane nasumično, stim da nema paralelnih, vertikalnih niti da ima tri linije koje se seku u istoj tački. Takođe postoji i provera da li je topološka linija stigla do

najdesnijeg kraja. Testovi su radjeni na računaru koji ima procesor AMD FX(tm)-8350 Eight-Core Processor 4.00GHz. Uradjeno je deset merenja za istu veličinu skupa linija i uzet je prosek. U tabeli je prikazano tih deset merenja. Vremena su u sekundama.

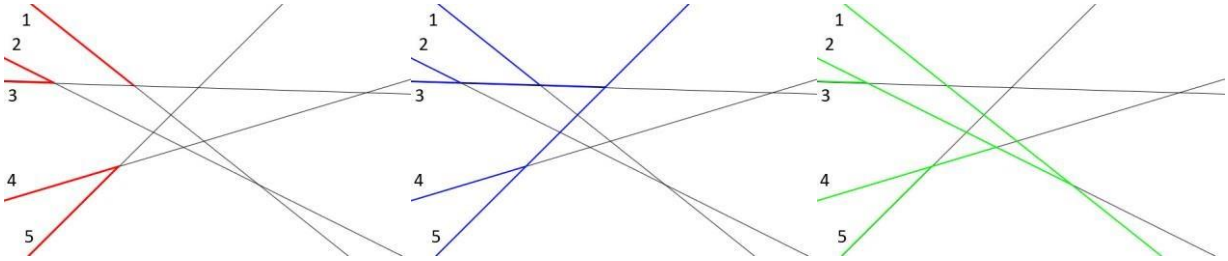
RB	500	1000	5000	10000
1	0.086481522	0.305970172	5.875202435	24.85988729
2	0.093683345	0.321593597	5.817452427	24.43543341
3	0.094647923	0.319280663	5.822176561	24.68757985
4	0.089452989	0.310779861	5.749640108	24.70286203
5	0.094453834	0.322316174	5.889452571	24.63029575
6	0.087297967	0.306778794	5.912020662	24.54021943
7	0.089756101	0.304660439	5.83895963	24.55860019
8	0.094357524	0.310383862	5.852184565	24.43552386
9	0.095065924	0.309993239	5.897963149	24.81722342
10	0.086855033	0.318066752	5.851504032	24.60709846
PROSEK	0.091205216	0.3129823553	5.850655614	24.62747237



Ilustracija 12: Grafik vremena izvršavanja. Vreme je u sekundama.

4.5 Primer

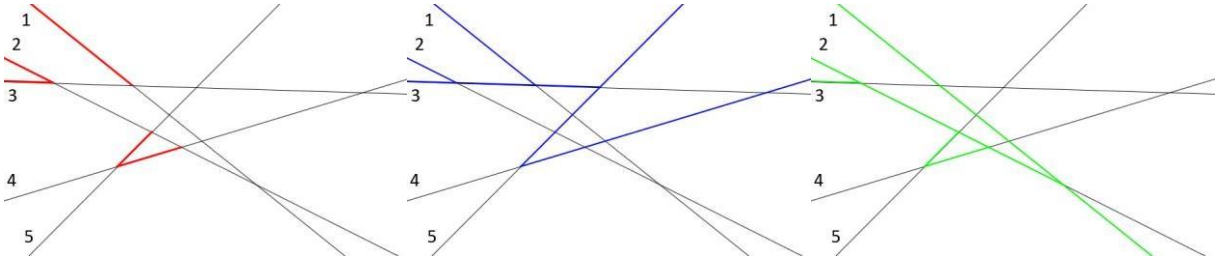
Sledi primer ovog postupka korak po korak, i to konkretno do preseka koji je služio kao primer u ovom radu. U tabelama su predstavljene strukture koje su opisane ranije. 1)



	E		HTU		λ_i	ρ_i	HTL		λ_i	ρ_i	N		M	I
1	a_1	b_1	-1	3	1	-1	0	1	-1	3	1	1	1 2	
2	a_2	b_2	-1	3	2	-1	1	2	-1	3	2	2	2 4	
3	a_3	b_3	-1	5	3	-1	2	3	-1	2	3	3		
4	a_4	b_4	-1	5	4	-1	2	4	-1	5	4	4		
5	a_5	b_5	-1	0	5	-1	4	5	-1	4	5	5		

Ovaj korak se razlikuje od drugih jer je ovo inicijalizacija. Prvo su formirana stabla. Radi boljeg pregleda, linije su numerisane od one sa manjim nagibom ka onoj sa većim. Tako su u stablu gornjeg horizonta linije ubacivane počevši od linije pet, pa onda redom do linije broj jedan. Za stablo donjeg horizonta, prvo je ubačena linija jedan, što se može videti po tome što je ta linija beskonačna. Presek, tj. strukture N i M se formiraju uzimajući kraći segment iz oba stabla.
Zajedničke desne krajeve imaju druga i četvrta ivica, pa zato na steku I imamo dva i četiri.

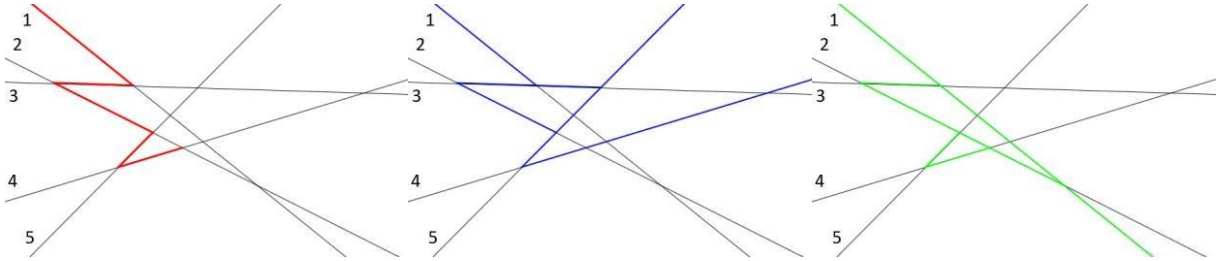
2)



	E		HTU		λ_i	ρ_i	HTL		λ_i	ρ_i	N		M	I
1	a_1	b_1	-1	3	1	-1	0		-1	3		1	1	
2	a_2	b_2	-1	3	2	-1	1		-1	3		2	2	
3	a_3	b_3	-1	5	3	-1	2		-1	2		3	3	
4	a_4	b_4	5	0	4	5	2		4	2		4	5	
5	a_5	b_5	4	0	5	4	2		5	2		5	4	

1	1 2	
2	2	
3	3	
4	4	
5	5	

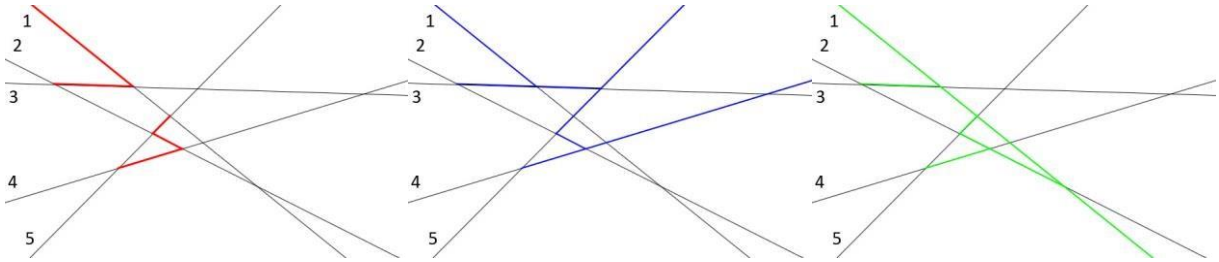
Sa steka se uzima spremno teme u ovom slučaju između ivice 4 i 5, i preko njega se vrši elementarni korak. U strukturi M 4 i 5 menjaju mesta. $HTU[5]$ se samo skratio sa leve strane, a $HTU[4]$ nastavljen. Slično je $HTL[4]$ skraćen, a $HTL[5]$ nastavljen. Određivanje dokle idu nastavljene linije je opisan ranije. $N[4]$ i $N[5]$ predstavljaju kraće segmente i to konkretno $N[4]$ je segment $HTL[5]$, jer je kraći od $HTU[5]$, i $N[5]$ je $HTL[4]$, jer je kraći od $HTU[4]$. $N[4]$ nema zajednički desni kraj sa $N[3]$ a $N[5]$ je poslednja ivica pa ne dodajemo novo teme na stek. 3)



	E			HTU		λ_i	ρ_i	HTL		λ_i	ρ_i	N		M		I
1	a_1	b_1	1	-1	3	1	-1	0	1 1	-1	3	1	1	2 2		
2	a_2	b_2	3	3	5	2	3	1		2	1	2	3			
3	a_3	b_3		2	5	3	2	1		3	5	3	2			
4	a_4	b_4		5	0	4	5	2		4	2	4	5			
5	a_5	b_5		4	0	5	4	2		5	2	5	4			
3							3									
4							4									
5							5									

U ovom koraku, dve nove ivice imaju zajedničke desne krajeve, što zaključujemo na osnovu toga što je desni kraj $N[1] = M[2]$ i desni kraj $N[2] = M[1]$ kao i desni kraj $N[3] = M[4]$ i desni kraj $N[4] = M[3]$. Dva nova temena su dodata, i zato imamo 1 i 3 na steku.

4)



	E		HTU	λ_i	ρ_i	HTL	λ_i	ρ_i	N	M	I
1	a_1	b_1									
2	a_2	b_2									

3	a_3	b_3
4	a_4	b_4
5	a_5	b_5

-1	3
5	4
2	5
5	0
2	0

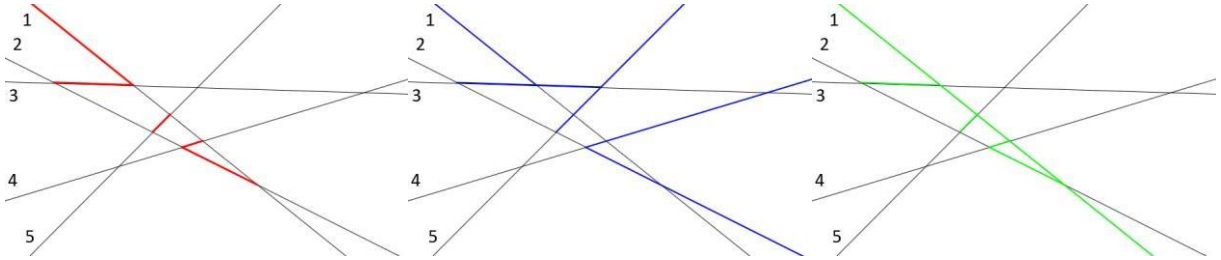
1	-1	0
2	5	1
3	2	1
4	5	2
5	2	1

-1	3
2	1
2	1
5	4
5	2

1	1
2	3
3	5
4	2
5	4

1		1 1
2		2 4
3		3
4		4
5		5

U ovom koraku se najbolje vidi primer gde nastavljeni segment nije jednostavno prekinut susednom ivicom. Nastavak linije pet u HTL stablu, tj. segmenta $HTL[5]$ udara u zaliv sastavljen od $HTL[3]$, što je u stvari ivica $N[2]$, i $HTL[1]$. Segment $HTL[3]$ nije presečen linijom pet, pa se zato nastavlja po zalivu gde nailazimo na $HTL[1]$ koji jeste presečen linijom pet. 5)

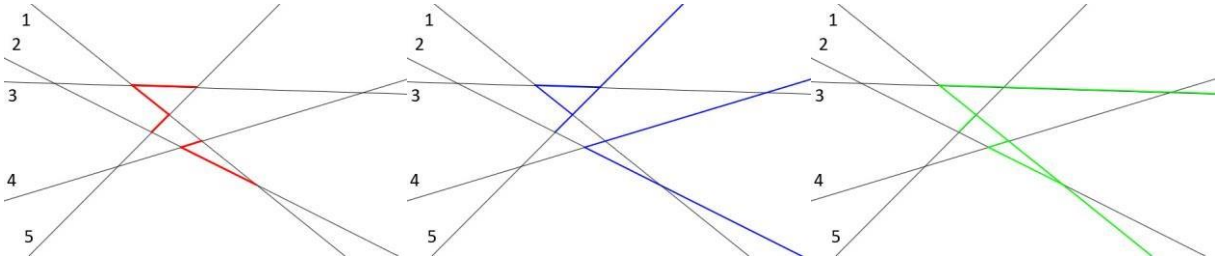


	E			HTU		λ_i	ρ_i	HTL		λ_i	ρ_i	N		M	I
1	a_1	b_1		-1	3	1	-1	0		-1	3		1	1	
2	a_2	b_2		4	0	2	4	1		2	1		2	3	
3	a_3	b_3		2	5	3	2	1		2	1		3	5	
4	a_4	b_4		2	0	4	2	1		2	1		4	4	
5	a_5	b_5		2	0	5	2	1		4	1		5	2	

1		1 1
2		2
3		3
4		4
5		5

Ovde se najbolje vidi ranija zabrinutost da HTU više nije stablo, nego da može postati šuma, međutim ako se setimo, dodali smo se pomoćnu vertikalnu liniju u $x = +\infty$, i u ovom slučaju, $HTU[5]$, $HTU[4]$ i $HTU[2]$ udaraju u tu liniju što je označeno time što je ρ jednako nula.

6)



	E	
1	a_1	b_1
2	a_2	b_2
3	a_3	b_3
4	a_4	b_4
5	a_5	b_5

HTU	
3	5
4	0
1	5
2	0
2	0

	λ_i	ρ_i
1	3	0
2	4	1
3	1	0
4	2	1
5	2	1

HTL

λ_i	ρ_i
1	5
3	5
2	1
2	1
4	1

N

1	3
2	1
3	5
4	4
5	2

M

I

1	1 2
2	2
3	3
4	4
5	5

Dolazimo do stanja korišćenog kao primer. Primećujemo da se u svakom koraku menjalo $O(1)$ elemenata, posle prolaza kroz zaliv, za čiji obilazak je već dokazana složenost. Postupak se nastavlja sve dok nema više spremnih temena, i u tom slučaju presek će se sastojati od poluprava, slično kao početno stanje, samo što će sve ići desno u beskonačnost.

4.6 Specijalni slucajevi

Za specijalne slučajeve kao što su paralelne linije, vertikalne linije i vise od dve linije koje se seku u jednoj tački, Edelsbrunner [1] je predložio metod koji olakšava implementaciju time što nije potrebno zasebno posmatrati svaki specijalani slučaj. Ta metoda je zapravo primitivna procedura koja posmatra paralelne linije kao da nisu paralelne i koja posmatra tri linije koje se seku u jednoj tačku kao da se ne seku. To znači da se pojavljuju ivice dužine nula i tačke preseka u beskonačnosti.Primitivna procedura prima i indekse od 1 do n kao parametar. Ako je linija l_i data jednačinom:

$$a_i x + b_i y + c_i = 0$$

za $1 \leq i \leq n$ i $(a_i, b_i) \neq (0, 0)$ definišemo sledeću liniju

$$l_i(\varepsilon): a_i' x + b_i' y + c_i' = 0$$

gde je $a_i' = a_i + \varepsilon^{2^{3i}}$, $b_i' = b_i + \varepsilon^{2^{3i-1}}$, $c_i' = c_i + \varepsilon^{2^{3i-2}}$, za neko $\varepsilon > 0$ koje je dovoljno malo.

Umesto da se odredi raspored za skup H određuje se raspored za $H(\varepsilon) = \{l_i(\varepsilon) \mid 1 \leq i \leq n\}$ i $\varepsilon > 0$ dovoljno malo. Lako se dokazuje da $H(\varepsilon)$ nema dve paralelne linije ni tri linije koje se seku jednoj tački. To znači da se sve odluke, kao što je određivanje da li l_i preseca l_j iznad l_k, zasnivaju na $l_i(\varepsilon)$, a ne na l_i, što se može svesti na određivanje znaka sledeće determinante:

$$\begin{vmatrix} a_i' & b_i' & c_i' \\ a_j' & b_j' & c_j' \\ a_k' & b_k' & c_k' \end{vmatrix}$$

što se može uraditi bez konkretne vrednosti za ε. Međutim, svako računanje se mora izvesti bez gubitka preciznosti. Ova tehnika se zove simbolična perturbacija. Više detalja u radu „Simulation of simplicity“ [2].

U radu [3] Rafalin Souvaine i Streinu su predložili određene modifikacije algoritma kako bi se rešili specijalni slučajevi. Oznake su preuzete is originalnog rada. Koristi se dosta istih struktura. $E[]$, za čuvanje jednačina linija, $M[]$, za redosled linija preseka, $HTL[]$ i $HTU[]$, za stabla horizonta. Elementi struktura $N[]$ i I imaju jos jedno dodatno polje. $N[i]$ je sada trojka (λ_i, $r_{up,i}$, $r_{down,i}$) indeksa. Ako je desni kraj $N[i]$ generisan presekom sa linojm iznad onda je $r_{up,i}$ indeks te linije, u suprotnom je $null$. Ako je desni kraj $N[i]$ generisan presekom sa linojm ispod onda je $r_{down,i}$ indeks te linije, u suprotnom je $null$. Bar jedan od $r_{up,i}$, $r_{down,i}$ nije $null$ a moguće je da oba nisu $null$ u slučaju preseka tri ili više linija. λ_i je bilo koja linija koja koja preseca ivicu u levom kraju. Sa poznatim HTU i HTL ovi indeksi se računaju u konstantnom vremenu.

Elementi steka I su sada parovi celih brojeva (i, k), što predstavlja ivice m_i, m_{i+1}, … , m_{i+k} preseka koje imaju zajednički desni kraj i taj desni kraj je tačka koja je spremna da bude sledeći elementarni korak.

Potrebna je i nova struktura. $MATCH[i]$ je par indeksa koji pokazuju na najvišu i na najnižu ivicu preseka koja deli desni kraj kao i ivica na liniji l_i. $MATCH[i]$ se resetuje na par (i, i) svaki put kada l_i učestvuje u elemntarnom koraku. $MATCH$ se ažurira na kraju elementarnog koraka kada se pronađu novi zajednički desni krajevi. Ivice duž preseka koje dele desni kraj formiraju najviše jednu povezanu komponentu (pogledati Lemu 4 dole) pa samim tim ažuriramo samo dve krajnje ivice, kada dodajemo novu ivicu u niz onih koji dele zajednički desni kraj. One elemente koji se nalaze između ignorišemo.

Određivanje spremnih temena:

Definiše se ***odgovarajući par*** linija kao uzastopni par linija l_i, l_j gde je $r_{up,i}$ jednak j, i gde je $r_{down,j}$ jednak i. Kada najviše dve linije učestvuju u preseku, odgovarajući par implicira spremno teme. U opštem slučaju, niz uzastopnih linija l_i, …, l_j preseka gde su sve susedne linije odgovarajući parovi, i kada gornja linija l_i ima $r_{up,i}$ jednak null i kada donja linija l_j ima $r_{down,j}$ jednak null onda to implicira spremno teme. Nakon svakog ažuriranja preseka, proveravamo da li dve nove susedne linije prave odgovarajući par. Ako je tako, taj novi par ili dopunjava već

postojeći niz odgovarajućih parova, ili započinje novi niz. Ažuriranje i $MATCH$ strukture i provera da li je niz završen se izvodi u konstantnom vremenu(pogledati Lemu 6 dole).

Paralelne linije i identične linije:

Paralelne linije imaju presek u beskonačnosti. Njihova presečna tačka se ne tretira drugačije od preseka nekog drugog para linija. Identične linije se tretiraju kao jedna linija, i pretpostavlja se da će se aplikacija koja poziva topološko prebrisavanje pobrinuti za uticaj duplih linija.

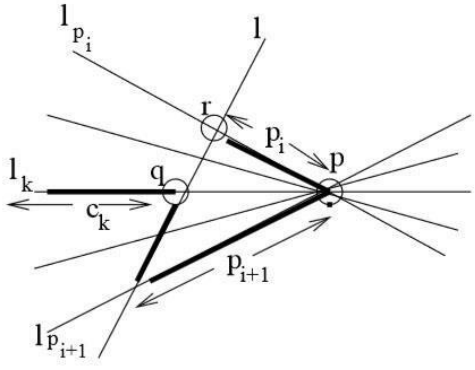
Dodatne promene u algoritmu:

Provere nazvane 'iznad' i 'bliži' se koriste kada se strukture prave i ažuriraju. Svako poređenje ima tri ishoda umesto dva: TRUE, FALSE i EQUAL(Jednki). The EQUAL stanje nije deo originalnog algoritma jer se generiše samo u specijalnim slučajevima. Da bi se obradile paralelne linije, dodati su specjialni testovi, koji proveravaju da li su linije o kojima je reč paralelne. Provere se izvode tokom inicijalizacije.

Svaki korak ažuriranja zahteva dve ili više promene, u zavistnosti od temena. Različite strukture zahtevaju različite strategije ažuriranja. Jedna metoda je da se zameni svaka linija od i do j u odgovarajućem nizu (nizu sa odgovarajućim parovima). Ova procedura se koristi da bi se ažurirali levi krajevi preseka i stabla donjeg i gornjeg horizonta. Druga metoda je da se linije obrađuju jedna po jedna. Ta metoda se koristi da bi se ažurirali desni krajevi preseka i stabla dognjeg i gornjeg horizonta. Stabla se moraju obrađivati uzastopno, u suprotnom presek neće biti ažuriran kako treba.

Analiza složenosti:

Lema 4: Skupi ivica na preseku koje imaju zajednički desni kraj formiraju najviše jednu povezanu komponentu



Ilustracija 13

Dokaz: Pretpostavimo da postoje ivice koje sadrže p kao njihov desni kraj i koje formiraju više od jedne komponente(Ilustracija 13). Ivice p_i i p_{i+1} se nalaze na linijama l_{pi} i l_{pi+1}. Kako postoji više od jedne povezane komponente, postoji bar jedna linija l_k izmedju l_{pi} i l_{pi+1} koja sadrži tačku p, ali čija trenutna ivica c_k nema p kao desni kraj. Neka je desni kraj c_k, q, određen linijom

l. Ako je nagib linije l manji (respektivno veći) od linije l_k, onda l preseca l_{pi} (respektivno l_{pi+1}) u tački r koja je između p i q. Kako q još uvek nije procesuirana, tačka r nije spremna da bude procesuirana, i zato ivica p_i (respektivno p_{i+1}) ne može da bude deo preseka. Kontradikcija.

Lema 5: Ukupna složenost ažuriranja HTU i HTL tokom svih elementarnih koraka je $O(n^2)$

Dokaz: slično kao i u originalnom algoritmu.

Lema 6: Ukupna složenost upoređivanja susednih ivica (pronalaženje spremnih temena) tokom svih elementarnih koraka je $O(n^2)$

Dokaz: Svaka ivica, c, koja se završava na nekom raskršću može generisati najviše tri provere. Prva provera je sa ivicama iznad i ispod. Ako je odgovarajuća ivica m pronađena, $MATCH[m]$ će takođe biti proverena. Iz leme 4, najviše jedna ivica koja ograničava c iznad i ispod može biti pogodak, u suprotnom postoji više od jedne povezane komponente. Zbog toga najviše tri provere postoje. Nema potrebe vršiti dodatne provere, jer ako postoji još neko odgovaranje, trebalo je da je već ranije pronađeno. Ukupan broj ivica koje nailaze na raskršća je n^2 pa je samim tim ukupna složenost $O(n^2)$.

Lema 7: Ukupna složenost je $O(n^2)$

Dokaz: Inicijalizacija struktura je linerna nakon sortiranja linija po nagibu koje ima složenost $O(n \log n)$. Svaki elementarni korak ima konstantnu amortizovanu složenost, što proizilazi iz lema 5 i 6. Elementarnih koraka ima $O(n^2)$, i tako je ukupna složenost $O(n^2)$.

Generalno postoji problem sa brojevima sa beskonačnom presiznošću. U praksi brojevi su predstavljeni sa pokretnim zarezom što može stvoriti greške kod zaokruživanja. U mnogim aplikacijma ovo nije problem, međutim u računarskoj geometriji to može biti veliki problem ukoliko se algoritm npr. grana na osnovu određene vrednosti. Ovi problemi su i dalje aktivni deo istraživanja.

5 Zaključak

U ovom radu se diskutovalo o rasporedu linija i postupcima konstruisanja, počevši od opisa i definicija rasporeda, preko jednostavnog inkrementalnog algoritma do topološkog prebrisavanja, koji se i dan danas koristi i citira. Govorilo se o problemima na koja je moguće naići, i kako ih rešiti u zavisnosti od zahteva same aplikacije. Diskutovano je i o ograničenjima, složenostima, kako vremenskim tako i prostornim, u svrhu lakšeg odabira algoritma i postupka u konkretnim primenama i aplikacijama. Dati su i dokazi za te složenosti. Opisana je i vrlo važna Zone teorema. Dati su primeri struktura za čuvanje rasporeda linija, kao i struktura korišćenih u algoritmima. Algoritam topološkog prebrisavanja je generalizacija algoritma koji su predstavili Bentley i Ottmann [4]. [1] Edelsbrunnerov algoritam se može proširiti do algoritma za enumeraciju temena u rasporedu n hiperravni u $\mathbb{R}^d$ sa $O(n^d)$ vremenskom i $O(n)$ prostornom složenošću. Neki od drugih algoritama zasnovanih na topološkom prebrisavanju su [5], [6] i [7]. Avis i Fukuda [8] su razvili algoritam za enumeraciju u $O(n^2k)$ vremena i $O(n)$ prostora svih k

temena rasporeda n hiperravni u $\mathbb{R}^d$ gde svako teme dodiruje d hiperravni. Taj algoritam je vrlo koristan kada ima puno paralelnih hiperravni. Drugi srodni radovi su [9] i [10]. Postoji puno otvorenih problema i mesta za dalji razvoj, npr. problem nalaženja najkraćeg puta u rasporedu linija koji je predložio Marc van Kreveld, kao i razvoj algoritama za određivanje rasporeda u višim dimenzijama. Algoritam topološkog prebrisavanja je bio vrlo značajan pored toga se i dan danas puno citira u mnogim radovima, sada već kao jedan od klasičnih tehnika.

6 Reference

[1] H. Edelsbrunner / L. J. Guibas, „Topologically sweeping an arrangement," u *Proceedings of the eighteenth annual ACM symposium on Theory of computing*, ACM, 1986, pp. 389-403.

[2] H. Edelsbrunner / E. P. Mücke, „Simulation of simplicity: a technique to cope with degenerate cases in geometric algorithms," *ACM Transactions on Graphics (TOG),* t. 9, br. 1, pp. 66-104, 1990.

[3] E. Rafalin, D. Souvaine / I. Streinu, „Topological sweep in degenerate cases," u *Algorithm Engineering and Experiments*, Springer, 2002, pp. 155-165.

[4] J. L. Bentley / T. A. Ottman, „Algorithms for reporting and counting geometric intersections,," *Computers, IEEE Transactions on,* t. 100, br. 9, pp. 643-647, 1979.

[5] E. G. Anagnostou, L. J. Guibas / V. G. Polimenis, „Topological sweeping in three dimensions," u *Algorithms*, Springer, 1990, pp. 310-317.

[6] T. Asano, L. J. Guibas / T. Tokuyama, „Walking in an arrangement topologically," *International Journal of Computational Geometry & Applications,* t. 4, br. 02, pp. 123-151, 1994.

[7] H. Edelsbrunner / D. L. Souvaine, „Computing least median of squares regression lines and guided topological sweep," *Journal of the American Statistical Association,* t. 85, br. 409, pp. 115-119, 1990.

[8] D. Avis / K. Fukuda, „A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra," *Discrete & Computational Geometry,* t. 8, br. 1, pp. 295313, 1992.

[9] D. Avis / K. Fukuda, „Reverse search for enumeration," *Discrete Applied Mathematic,* t. 65, br. 1, pp. 21-46, 1996.

[10] K. Fukuda, T. M. Liebling / F. Margot, „Analysis of backtrack algorithms for listing all vertices and all faces of a convex polyhedron,“ *Computational Geometry,* t. 8, br. 1, pp. 112, 1997.