Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra matematiky

Diplomová práce

Zobecněné Voroneho diagramy

Plzeň, 2012 Jana Homrová

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracovala samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne …… Jana Homrová ……....

Poděkování

Děkuji RNDr, Světlaně Tomiczkové, PhD. za poskytnutý čas, odbornou pomoc, za ochotu a trpělivost při vedení mé diplomové práce.

Zobecněné Voroneho diagramy

Anotace

Tato diplomová práce se zabývá teorií potřebnou pro studium zobecněných Voroneho diagramů. Nejprve se zaměřuje na Obecný Voroneho diagram, pro který stručně uvede jeho definice a vlastnosti. Dále pro tento diagram popíše tři různé algoritmy, konkrétně Inkrementální algoritmus, "Plane sweep" algoritmus a Metodu rostoucích regionů. Posléze se zaměří na zobecněné Voroneho diagramy, a to na Voroneho diagram v L_1 metrice, Voroneho diagram kružnic a Laguerre Voroneho diagram. Pro každý z těchto diagramů zpracuje teorii a minimálně jeden ze zmíněných algoritmů. Součástí práce je implementace Inkrementálního algoritmu pro Laguerre Voroneho diagram v programu Mathematica 7.

Klíčová slova

Obecný Voroneho diagram, Voroneho diagram v L_1 metrice, Voroneho diagram kružnic, Laguerre Voroneho diagram, Inkrementální algoritmus, “Plane sweep“ algoritmus, Metoda rostoucích regionů

Generalized Voronoi diagrams

Annotation

The thesis deals with the theory needed for studying generalized Voronoi diagrams.
At the beginning it focuses on the Ordinary Voronoi diagram and briefly introduces its definition and characteristics. After that three different algorithms for this diagram are described, specifically Incremental algorithm, “Plane sweep“ algorithm and the Method of growing regions. Then it concentrates on generalized Voronoi diagrams, which are Voronoi diaram in L_1 distance, Voronoi diagram of circle and Laguerre Voronoi diagram. For each of these diagrams a theory and at least one of the algorithms are processed. One part of the thesis is the implementation of Incremental algorithm for Laguerre Voronoi diagram in software Mathematica 7.

Keywords

Ordinary Voronoi diagram, Voronoi diagram in L_1 distance, Voronoi diagram of circle, Laguerre Voronoi diagram, Incremental algorithm, “Plane sweep“ algorithm, Method of growing regions

Obsah

1 Úvod

Tato diplomová práce se zabývá teorií potřebnou pro studium zobecněných Voroneho diagramů. Dále se věnuje vybraným typům zobecněných Voroneho diagramů a uvádí pro ně vhodné algoritmy, popř. modifikaci známých algoritmů.

Pro tuto práci byla vybrána tři různá zobecnění Voroneho diagramů a to Voroneho diagram v L_1 metrice, Voroneho diagram kružnic a Laguerre Voroneho diagram.

Diplomová práce navazuje na bakalářskou práci [4], která se zabývala Voroneho diagramy, definicí základních pojmů a vlastností. Dále v ní byly uvedeny některé typy algoritmů pro konstrukci Voroneho diagramů a některé typy zobecnění.

Kapitoly diplomové práce se postupně věnují jednotlivým Voroneho diagramům.

Stručně je uveden Obecný Voroneho diagram, který byl podrobněji řešen v bakalářské práci [4], jeho základní terminologie a tři různé algoritmy pro jeho konstrukci.

V další kapitole je rozepsán Voroneho diagram v L_1 metrice a Metoda rostoucích reginů pro tento typ diagramu. Je popsána modifikace metody a její znázornění v programu Geogebra.

Následně je popsán Voroneho diagramem kružnic, hojně se, v této kapitole, využívá článek [1]. Pro tento diagram je uvedena modifikace všech tří algoritmů, které jsou v této práci řešeny a metoda rostoucích regionů je opět znázorněna v programu Geogebra.

Poslední zobecnění, kterému je věnována pozornost je Laguerre Voroneho diagram. Je pro něj popsán Inkrementální algoritmus, jehož implementace v programu Mathematica 7, je součástí práce. Uvádí se podrobný popis algoritmu a řešení problémů, které se při jeho implementaci vyskytly.

V závěrečné kapitole je uveden souhrn algoritmů, jejich výhod a nevýhod. Současně jsou zmíněna možná využití Voroneho diagramů.

Hlavními zdroji jsou [1] od autorů Li Jin, Domguk Kim, Lisen Mu, Deok-Soo Kim, Shi-Min Hu, [9], jehož autorem je K. Sugihara, [5] od autorů F. Aurenhammer a R. Klein, [10], jehož autory jsou Hiroshi Imai, Masao Iri a Kazuo Murota.

2 Obecný Voroneho diagram

Nechť $S = \{P_1, P_2, \ldots, P_n\}$ je množina generujících bodů v rovině, kde $n \geq 2$. Eukleidovská vzdálenost dvou bodů $P = (p_1, p_2)$ a $Q = (q_1, q_2)$ je definována vztahem $d_e(P, Q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$.

Definice 2.1 *Mějme dva generující body $P_1, P_2 \in S$ a libovolný bod X v rovině.*
***Otevřenou polorovinu** bodu P_1 definujeme*

$$h(P_1, P_2) = \{X | d_e(P_1, X) < d_e(P_2, X)\}. \tag{1}$$

***Uzavřenou polorovinu** bodu P_1 definujeme*

$$h'(P_1, P_2) = \{X | d_e(P_1, X) \leq d_e(P_2, X)\}. \tag{2}$$

*Přímka jež odděluje polorovinu $h(P_1, P_2)$ obsahující bod P_1 od poloroviny $h(P_2, P_1)$ obsahující bod P_2, je osa úsečky bodů P_1, P_2. **Osu úsečky** bodů P_1, P_2 definujeme*

$$B(P_1, P_2) = \{X \mid d_e(P_1, X) = d_e(P_2, X)\}. \tag{3}$$

•

Definice 2.2 ***Voroneho buňka** $\nu(P_i)$ bodu P_i vzhledem k množině $S = \{P_1, P_2, \ldots, P_n\}$ je definována*

$$\nu(P_i) = \bigcap_{P_j \in S, P_j \neq P_i} h(P_i, P_j). \tag{4}$$

•

Z definice 2.2 plyne, že Voroneho buňka $\nu(P_i)$, kde $P_i \in S$, je průnikem $n - 1$ otevřených polorovin $h(P_i, P_j), i \neq j$, obsahujících generující bod P_i. Lze ukázat, že Voroneho buňka $\nu(P_i)$ je otevřená a konvexní [5], [4].

Definice 2.3 *Hranice mezi Voroneho buňkami se nazývají **Voroneho hrany** a v následujícím textu je budeme značit e.*

•

Voroneho hrany jsou úsečky, polopřímky a ve speciálním případě přímky. Voroneho hrana je přímkou v případě, kdy všechny generující body $P_i \in S$ jsou kolineární.

Definice 2.4 ***Voroneho diagram** množiny S generujících bodů P_i je sjednocením Voroneho buněk $\nu(P_i)$ a Voroneho hran.*

•

Poznámka 1 *Jestliže Voroneho hrana e je hranicí mezi $\nu(P_i)$ a $\nu(P_j)$, kde $i \neq j$, pak platí $e \subset B(P_i, P_j)$.*

•

Definice 2.5 *Průsečík Voroneho hran nazýváme* ***Voroneho vrchol*** *a v následujícím textu jej budeme značit V.*

•

2.1 Algoritmy pro konstrukci Obecných Voroneho diagramů

Existuje několik druhů algoritmů, pomocí kterých lze zkonstruovat Voroneho diagram zadané množiny S generujících bodů. Jednotlivé algoritmy se mouhou lišit svou složitostí.

Pro další úvahy budeme předpokládat splnění následující podmínky

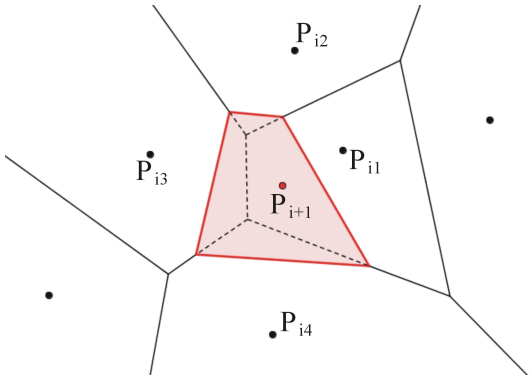
- Voroneho diagram není degenerovaný (*tzn. žádné čtyři generující body neleží na společné kružnici*) [4].

2.1.1 Inkrementální algoritmus pro Obecný Voroneho diagram

Jedná se o algoritmus, který vzhledem ke své jednoduchosti patří mezi nejpoužívanější algoritmy pro konstrukci Voroneho diagramů.

Nechť máme množinu generujících bodů $S = \{P_1, P_2, \ldots, P_n\}$, pro kterou hledáme příslušný Voroneho diagram. Nejprve nalezneme Voroneho diagram pro dva generující body. Tento zjednodušený diagram pak postupně modifikujeme přidáváním dalších generujících bodů z množiny S.

Hlavní část tedy spočívá v transformaci diagramu z i na $i + 1$ generujících bodů, pro $i = 2, \ldots, n$, viz obr. 1.



Obrázek 1: Inkrementální algoritmus

Algoritmus (Inkrementální pro Obecný Voroneho diagram)
Vstup: Množina generujících bodů S = $\{P_1, P_2, \ldots, P_n\}$.
Výstup: Voroneho diagram množiny S.

1. **Inicializace**
   (a) vytvoření fronty F a vložení generujících bodů $P_1, P_2, \ldots, P_n$ do fronty
2. **Postup algoritmu**
   (a) výběr dvou generujících bodů P_1, P_2 z fronty F a vytvoření Voroneho diagramu pro tyto generující body
   (b) výběr bodu P_{i+1} z fronty F, kde i udává počet generujících bodů stávajícího Voroneho diagramu
   (c) určení umístění bodu P_{i+1} ve stávajícím Voroneho diagramu, označení generujícího bodu Voroneho buňky, ve které se bod P_{i+1} nachází, bude P_{ij}, kde $j = 1, 2, \ldots$ a udává počet Voroneho buněk, které budou ovlivněny bodem P_{i+1}
      i. jestliže bod P_{i+1} leží na Voroneho hraně: zvolení libovolné z Voroneho buněk, které náléží daná hrana
      ii. jestliže bod P_{i+1} neleží na Voroneho hraně: určení v jaké Voroneho buňce, stávajícího Voroneho diagramu, se bod P_{i+1} nachází
   (d) nalezení osy úsečky $B(P_{i+1}, P_{ij})$
   (e) nalezení průsečíků osy úsečky $B(P_{i+1}, P_{ij})$ s hranicí Voroneho buňky $\nu(P_{ij})$
   (f) rozhodnutí o počtu průsečíků osy úsečky $B(P_{i+1}, P_{ij})$ s hranicí Voroneho buňky $\nu(P_{ij})$
      i. neexistuje žádný průsečík
         A. pokračování bodem (2b)
      ii. existuje pouze jeden průsečík
         A. tento průsečík určí následující Voroneho buňku, která bude ovlivněna bodem P_{i+1}, označení generujícího bodu této buňky bude $P_{i(j+1)}$
         B. nalezení osy úsečky $B(P_{i+1}, P_{i(j+1)})$ a jejích průsečíků s hranicí Voroneho buňky $\nu(P_{i(j+1)})$
         C. rozhodnutí o počtu průsečíků
            - existuje jeden průsečík: pokračování bodem (2g)
            - existují dva průsečíky: pokračování následujícím bodem
         D. zvolení průsečíku, který neleží na společné hraně Voroneho buněk $\nu(P_{ij}), \nu(P_{i(j+1)})$, pokračování bodem (2(f)iiA)
      iii. existují dva průsečíky

A. výběr jednoho libovolného průsečíku, čímž je určena Voroneho buňka, která jako další bude ovlivněna generujícím bodem P_{i+1}, označení generujícího bodu této buňky bude $P_{i(j+1)}$
B. nalezení osy úsečky $B(P_{i+1}, P_{i(j+1)})$ a jejích průsečíků s hranicí Voroneho buňky $\nu(P_{i(j+1)})$
C. rozhodnutí o počtu průsečíků
   - existuje jeden průsečík: zvolení druhého z původně nalezených průsečíků příslušných Voroneho buňce $\nu(P_{i1})$ a pokračování předcházejícím bodem
   - existují dva průsečíky: pokračování následujícím bodem
D. zvolení průsečíku, který neleží na společné hraně dvou Voroneho buněk $\nu(P_{ij}), \nu(P_{i(j+1)})$
E. ověření zda zvolený průsečík náleží hraně Voroneho buňky $\nu(P_{i1})$
   - if YES: přejít na bod (2g)
   - if NO: pokračování bodem (2(f)iiiB)

(g) vyrušení hran uvnitř nově vzniklé Voroneho buňky $\nu(P_{i+1})$

(h) ověření zda $i + 1 = n$

i. if Yes: konec algoritmu

ii. if No: pokračování bodem (2b)

Složitost tohoto algoritmu je obecně $O(n^2)$, ve speciálních případech může být i $O(n)$.

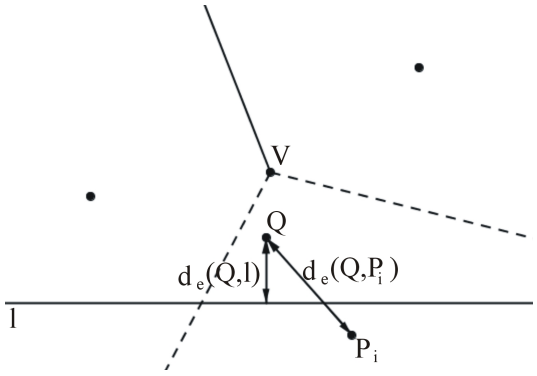
2.1.2 "Plane sweep"algoritmus pro Obecný Voroneho diagram

Tento algoritmus známe také pod názvem Fortuneho nebo Sweepline algoritmus.

Algoritmus je založen na využití takzvané zametací přímky. Zametací přímka je přímka, jež může být v rovině umístěna libovolně, nicméně pro jednoduchost se umisťuje horizontálně (*popř. vertikálně*). Zametací přímka se pohybuje v rovině shora dolů (*popř. zprava doleva*) [2], [4]. Základní přístup algoritmu se zabývá průsečíky zametací přímky s Voroneho diagramem.

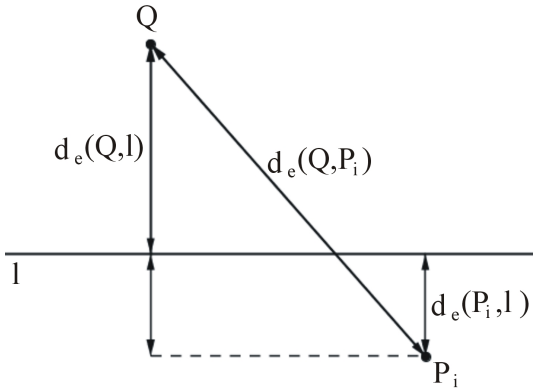
Nechť máme množinu generujících bodů $S = \{P_1, P_2, \ldots, P_n\}$ v rovině. Zametací přímka l je umístěna horizontálně a pohybuje se shora dolů. Zde nastává problém, Voroneho diagram $Vor(S)$ nacházející se nad zametací přímkou l závisí na generujících bodech nacházejících se pod l. Jinak řečeno, zametací přímka dosáhne vrcholu V Voroneho buňky $\nu(P_i)$ dříve, než dosáhne generujícího bodu P_i odpovídajícího buňce $\nu(P_i)$, viz obr. 2. Nemáme tedy potřebné informace k určení Voroneho vrcholu V [2].

Z tohoto důvodu nepatrně pozměníme základní přístup algoritmu. Namísto uchovávání informací o průsečících Voroneho diagramu $Vor(S)$ se zametací přímkou l, budeme uchovávat informace o té části $Vor(S)$, která není závislá na generujících bodech pod zametací přímkou l [2], [4].



Obrázek 2: Základní přístup algoritmu Plane sweep

Označme l^+ uzavřenou polorovinu nad zametací přímkou l a l^- otevřenou polorovinu pod l. Ptáme se, která část Voroneho diagramu $Vor(S)$ není ovlivněna generujícími body pod zametací přímkou l? Jinými slovy, pro které body $Q \in l^+$ známe jejich nejbližší generující bod [2]?



Obrázek 3: Vzdálenost bodu $Q \in l^+$ od zametací přímky a od generujícího bodu $P \in l^-$

Lemma 2.1 *Nechť máme libovolný bod $Q \in l^+$ a generující bod $P_i \in l^-$. Pak vzdálenost $d_e(Q, P_i) > d_e(Q, l)$.*

•

Důkaz Mějme libovolný bod $Q \in l^+$ a generující bod $P_i \in l^-$. Je zřejmé, že

$$d_e(Q, P_i) \geq d_e(Q, l) + d_e(P_i, l),$$

viz obr. 2, 3. Protože Eukleidovská vzdálenost nemůže být záporná a protože bod $P_i \in l^-$, pak platí $d_e(P_i, l) > 0$. Na základě toho můžeme psát

$$d_e(Q, P_i) \geq d_e(Q, l) + d_e(P_i, l) > d_e(Q, l)$$

a tedy platí

$$d_e(Q, P_i) > d_e(Q, l),$$

čímž je lemma dokázáno.

∘

Věta 2.1 *Bod Q leží nad zametací přímkou l, jestliže existuje generující bod $P_i \in l^+$, pro který platí, že $d_e(Q, P_i) < d_e(Q, l)$.*

•

Důkaz Z lemmatu 2.1 plyne, že pokud by bod Q ležel pod zametací přímkou l, pak vzdálenost $d_e(Q, P_i) > d_e(Q, l)$.

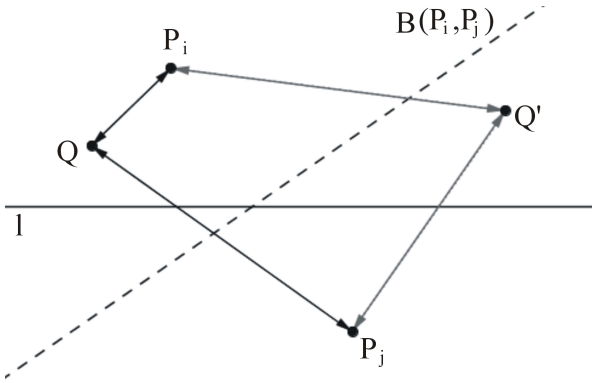
∘

Poznámka 2 *Pokud platí, že generující bod $P_j \in l^-$, $P_i \in l^+$, libovolný bod roviny $Q \in l^+$ a pokud platí*

$$d_e(Q, P_j) \geq d_e(Q, l) \geq d_e(Q, P_i), \tag{5}$$

pak bod Q nemůže být ovlivněn generujícím bodem P_j a tedy $Q \in \nu(P_i)$, viz obr. 4.

•



Obrázek 4: Vzdálenost bodu $Q \in l^+$ od generujících bodů P_i, P_j

Na základě těchto informací můžeme zadefinovat tzv. beach line.

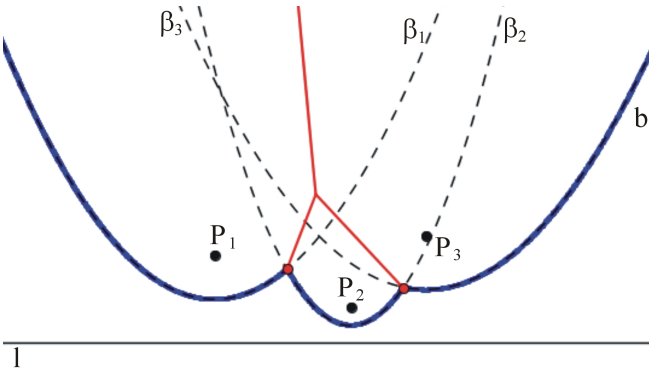
Definice 2.6 ***Beach line*** *b je křivka ohraničující body $Q \in l^+$, které nemohou být ovlivněny generujícími body $P_j \in l^-$.*

•

Poznámka 3 *Beach line pro Voroneho diagram bodů je křivka skládající se z parabolických oblouků β_i, viz obr. 5.*

•

Víme, že množina všech bodů, které mají stejnou vzdálenost od jednoho pevného bodu roviny a jedné pevně zvolené přímky v rovině, je parabola. Tato skutečnost platí pro libovolný generující bod $P_i \in l^+$.



Obrázek 5: Beach line pro Voroneho diagram bodů

Věta 2.2 *Průsečíky parabolických oblouků, které jsou součástí beach line zároveň leží na hranách Voroneho diagramu. Při pohybu zametací přímky tvoří tyto průsečíky Voroneho diagram, viz obr. 5.*

•

Důkaz Mějme beach line b pro dva generující body P_i, P_j a odpovídající zametací přímku l. Parabolické oblouky β_i, β_j náleží beach line a jsou příslušné generujícím bodům P_i, P_j. Průsečík dvou parabolických oblouků označíme V_p.

Pro libovolný bod $Y_i \in \beta_i$ platí

$$d_e(Y_i, l) = d_e(Y_i, P_i).$$

Obdobně, pro libovolný bod $Y_j \in \beta_j$ platí

$$d_e(Y_j, l) = d_e(Y_j, P_j).$$

Protože bod V_p leží na parabolickém oblouku β_i platí pro něj

$$d_e(V_p, l) = d_e(V_p, P_i),$$

současně bod V_p leží na parabolickém oblouku β_j a tedy pro něj platí

$$d_e(V_p, l) = d_e(V_p, P_j).$$

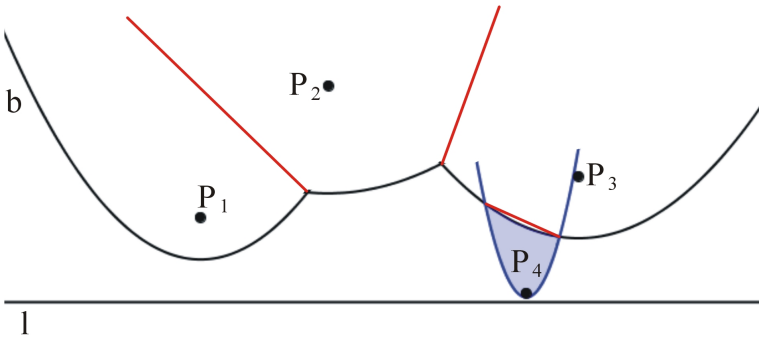
Po úpravě zístáváme rovnost

$$d_e(V_p, P_i) = d_e(V_p, P_j).$$

Z rovnosti je vidět, že průsečík V_p, dvou beach line oblouků β_i, β_j, má stejnou vzdálenost od dvou generujících bodů P_i, P_j. To znamená, že bod V_p leží na Voroneho hraně mezi dvěmi buňkami $\nu(P_i), \nu(P_j)$

○

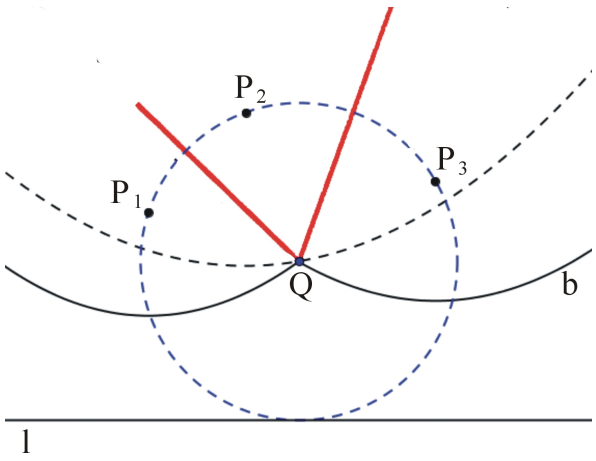
Při pohybu zametací přímky l se odpovídajícím způsobem mění struktura beach line b. Pro strukturu b jsou důležité dvě základní operace a to "site event"(*tj. objevení nového generujícího bodu P_i na zametací přímce l*) a "circle event"(*tj. zánik parabolického oblouku*).



Obrázek 6: Operace "site event"

Při operaci "site event"vzniká nový parabolický oblouk, který se stává součástí beach line, viz obr. 6 . Naopak při operaci "circle event"parabolický oblouk zaniká. To se děje ve chvíli, kdy tři parabolické oblouky $\beta_1, \beta_2, \beta_3$, příslušné ke třem generujícím bodům P_1, P_2, P_3, procházejí společným bodem Q. Bod Q má stejnou vzdálenost od daných generujících bodů a jedná se tedy o Voroneho vrchol, viz obr. 7.

Poznámka 4 *Bližší popis algoritmu pro Obecný Voroneho diagram je možné nalézt v [4], [2].*



Obrázek 7: Operace "cross event"

Algoritmus ("Plane sweep"pro Obecný Voroneho diagram)
Vstup: Množina generujících bodů S = $\{P_1, P_2, \ldots, P_n\}$.
Výstup: Voroneho diagram množiny S.

1. **Inicializace**
   (a) vytvoření zametací přímky l a beach line b, která je v počátku nevlastní
   (b) vložení generujících bodů P_i do fronty F
2. **Zpracování událostí**
   (a) výběr bodu P_i z fronty F
   (b) Switch(P_i)
      (c) CASE: Site event
         i. najít odpovídající beach line oblouk β
         ii. rozdělit β do dvou beach line oblouků β_l, β_r a vložit mezi ně nový beach line oblouk
         iii. vytvořit Voroneho hranu e
         iv. spojit Voroneho hranu e s beach line b
      (d) CASE: Circle event
         i. načtení odpovídající beach line trojice $(\beta_l, \beta, \beta_r)$
         ii. ověření, zda aktuální beach line trojice je platná
            A. if YES
               - načtení dvou Voroneho hran $e_i, e_j, i \neq j$ spojených s β

- vytvoření Voroneho vrcholu V
- spojení hran e_i, e_j ve vrcholu V
- vytvoření nové Voroneho hrany z Voroneho vrcholu V, kde V je koncovým vrcholem nově vzniklé Voroneho hrany
- připojení dvou beach line oblouků β_l, β_r k sousednímu oblouku β
- vyjmutí beach line oblouku β z fronty F

B. if NO

- pokračování v algoritmu

Ukončení algoritmu

1. Voroneho vrcholy spojeny s beach line nechť odpovídají nekonečnu
2. vymazání beach line b a zametací přímky l

Složitost tohoto algoritmu je $O(n\log n)$.

V následující kapitole se budeme věnovat Metodě rostoucích regionů. Metoda je zde zařazena především pro svoji názornost. V praxi se příliš nepoužívá, neboť je její náročnost velká. V této práci je modifikována a znázorněna v programu Geogebra pro dva typy zobecněných Voroneho diagramů. V následující kapitole, pro ukázku, popíšeme tuto metodu podrobněji a navrhneme možný přístup k její případné implementaci. Při jejím využití pro zobecněné Voroneho diagramy se o ni již blíže zmiňovat nebudeme.

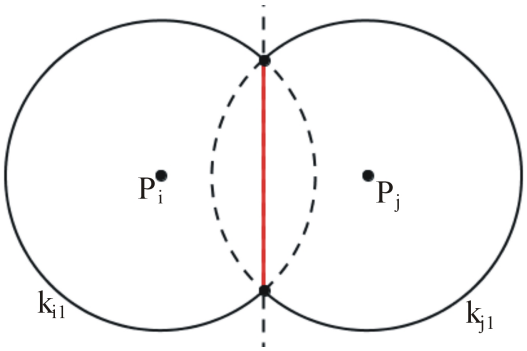
2.1.3 Metoda rostoucích regionů

Mějme množinu generujících bodů $S = \{P_1, P_2, \ldots, P_n\}$. Hlavní myšlenka metody rostoucích regionů spočívá ve vytvoření kružnic $k_i, i = 1, 2, \ldots, n$, v každém generujícím bodě $P_i \in S$. Kružnice k_i mají v počátku nulový poloměr, který se postupně zvětšuje. V bodech, ve kterých se kružnice pro dva generující body $P_i, P_j, i \neq j$, protnou, se tvoří Voroneho hrany [11].

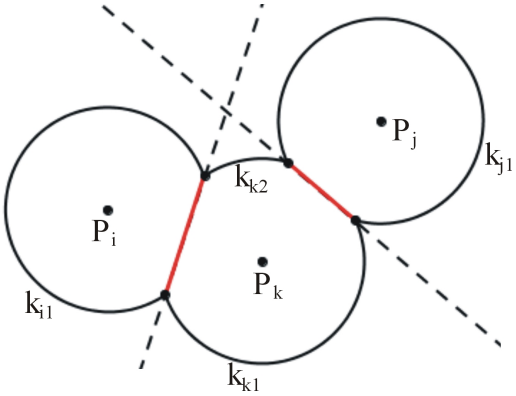
Poznámka 5 *Uvědomme si, že kružnice k_i se středy v generujících bodech $P_i \in S$ mají stejné poloměry. Pro každé dvě kružnice $k_i, k_j, i \neq j$, se středy v bodech P_i, P_j, platí $r_i = r_j$, kde r_i, r_j jsou poloměry těchto kružnic.*

•

Ještě než budeme pokračovat v popisu metody zavedeme množinu N, jež obsahuje kružnice k_i, které dosud nemají žádný společný bod s jinou kružnicí a současně obsahuje oblouky kružnic k_{ij}, které již mají společný průnik s jinou kružnicí. Oblouky, které množina N obsahuje, jsou pouze ty, které leží vně protínajících se kružnic, viz obr. 8.



Obrázek 8: Průnik dvou kružnic k_i, k_j



Obrázek 9: Rozdělení kružnice k_k na více oblouků

V množině N uchováváme pouze vnější oblouky protínajících se kružnic. Vnitřní oblouky zanedbáváme, neboť zasahují do Voroneho buňky pro jiný generující bod, díky tomu jsou pro účely metody nepodstatné, viz obr. 8.

Poznámka 6 *Uvědomme si, že jedna kružnice k_k může být rozdělena na více oblouků, jak je vidět na obrázku 9.*

•

Věta 2.3 *Mějme dva generující body $P_i, P_j \in S, i \neq j$ a dvě kružnice k_i, k_j jejichž středy jsou generující body P_i, P_j a jejich průnik je neprázdný. Pak průsečík $K^1_{i,j}$ (resp. průsečíky $K^1_{i,j}, K^2_{i,j}$) kružnic k_i, k_j leží na ose úsečky $B(P_i, P_j)$.*

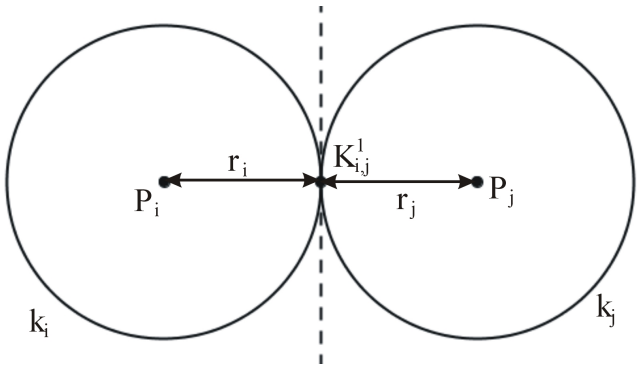
•

Důkaz Mějme dvě kružnice k_i, k_j jejichž středy jsou generující body P_i, P_j a jejich průnik je neprázdný. Pak pro průsečík $K^1_{i,j}$ (*resp. průsečíky* $K^1_{i,j}, K^2_{i,j}$) kružnic k_i, k_j platí

$$d_e(K^m_{i,j}, P_i) = r_i,$$

$$d_e(K^m_{i,j}, P_j) = r_j,$$

kde r_i, r_j jsou poloměry kružnic k_i, k_j a $m = 1$ (*resp.* $m = 1, 2$).



Obrázek 10: Dotyk dvou kružnic k_i, k_j

Poloměry r_i, r_j kružnic k_i, k_j se rovnají, $r_i = r_j$, viz poznamka 5. Na základě toho můžeme psát

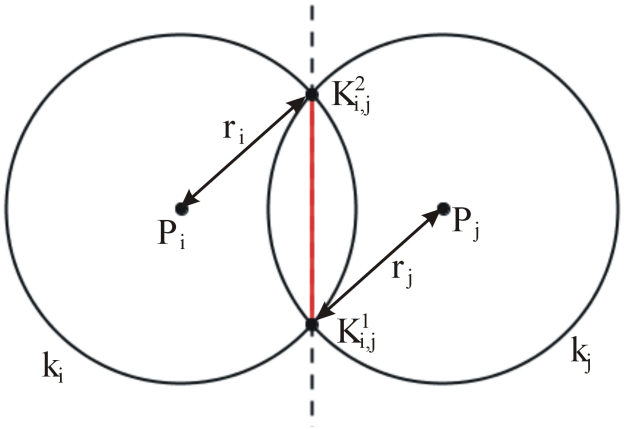
$$d_e(K^m_{i,j}, P_i) = d_e(K^m_{i,j}, P_j), \tag{6}$$

kde $m = 1$ (*resp.* $m = 1, 2$).

Průsečík $K^m_{i,j}$ kde $m = 1$ (*resp. m=1,2*) má stejnou vzdálenost od dvou generujících bodů P_i, P_j.

Platí tedy, že průsečíky $K^m_{i,j}$, kde $m = 1$ (*resp.* $m = 1, 2$), dvou kružnic k_i, k_j leží na Voroneho hraně, viz obr. 10 (*resp. obr. 11*).

○



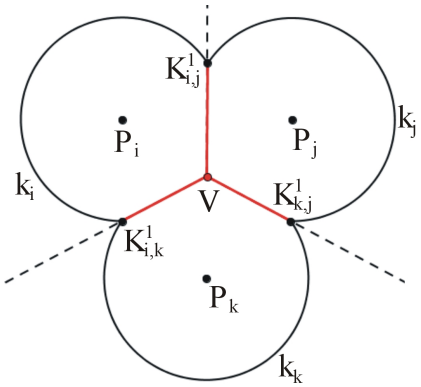
Obrázek 11: Průnik dvou kružnic k_i, k_j

Průsečíky $K^1_{i,j}, K^2_{i,j}$ dvou kružnic k_i, k_j, při zvětšování poloměru kružnic, vykreslují osu úsečky $B(P_i, P_j)$. Naším cílem není vykreslit celou osu úsečky, ale pouze její část

a to právě tu část, která je Voroneho hranou mezi generujícími body P_i, P_j. Potřebujeme tedy definovat podmínku, která v určité chvíli zastaví vykreslování Voroneho hrany. Pro tento účel uvedeme následující větu.

Věta 2.4 *Mějme tři generující body $P_i, P_j, P_k \in S$. Vytvoříme kružnice k_i, k_j, k_k se středy v generujících bodech P_i, P_j, P_k, s poloměry r_i, r_j, r_k, které jsou v počátku algoritmu rovny nule a postupně se zvětšují. Jestliže bod V je společným průnikem tří kružnic k_i, k_j, k_k, pak V je Voroneho vrchol, viz obr. 12.*

•



Obrázek 12: Společný průnik tří kružnic k_i, k_j, k_k

Důkaz Voroneho vrchol je bod, pro který platí, že má stejnou vzdálenost od tří generujících bodů. Pro Voroneho vrchol A a tři generující body $P_i, P_j, P_k \in S$ tedy platí

$$d_e(A, P_i) = d_e(A, P_j) = d_e(A, P_k).$$

V našem případě bod V leží na kružnicích k_i, k_j, k_k, které mají středy v generujících bodech P_i, P_j, P_k. Pro bod V platí

$$d_e(V, P_i) = r_i,$$

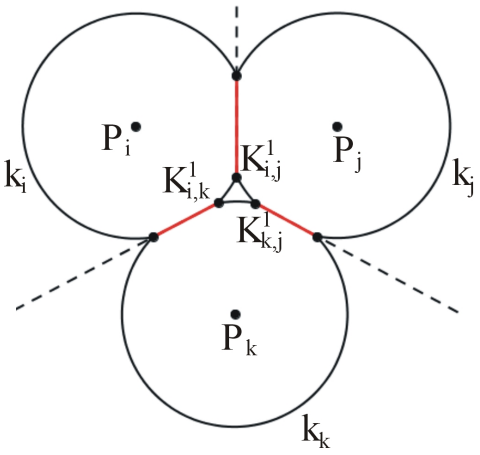
$$d_e(V, P_j) = r_j,$$

$$d_e(V, P_k) = r_k.$$

Protože poloměry r_i, r_j, r_k kružnic k_i, k_j, k_k jsou stejné, platí pro bod V následující rovnost

$$d_e(V, P_i) = d_e(V, P_j) = d_e(V, P_k)$$

a tedy bod V je Voroneho vrcholem, viz obr. 12.



Obrázek 13: Vykreslování Voroneho hran před vznikem Voroneho vrcholu

○

Nyní již máme potřebné informace, abychom mohli definovat podmínku, která zastaví vykreslování Voroneho hrany. Mějme tři generující body $P_i, P_j, P_k \in S$ a kružnice k_i, k_j, k_k se středy v generujících bodech P_i, P_j, P_k. Pro poloměry r_i, r_j, r_k těchto kružnic platí, že se rovnají $r_i = r_j = r_k$. Nechť $K^1_{i,j}$ je jeden z průsečíků kružnic k_i, k_j, $K^1_{i,k}$ je jeden z průsečíků kružnic k_i, k_k, $K^1_{k,j}$ je jeden z průsečíků kružnic k_k, k_j, viz obr. 13. Průsečíky $K^1_{i,j}, K^1_{i,k}, K^1_{k,j}$ vykreslují Voroneho hranu, dokud se kružnice k_i, k_j, k_k neprotnou v jednom bodě, tzn. dokud pro průsečíky platí $K^1_{i,j} \neq K^1_{i,k} \neq K^1_{k,j}$. V bodě ve kterém se kružnice protnou, vznikne nový Voroneho vrchol V a platí pro něj $V = K^1_{i,j} = K^1_{i,k} = K^1_{k,j}$, viz obr. 13.

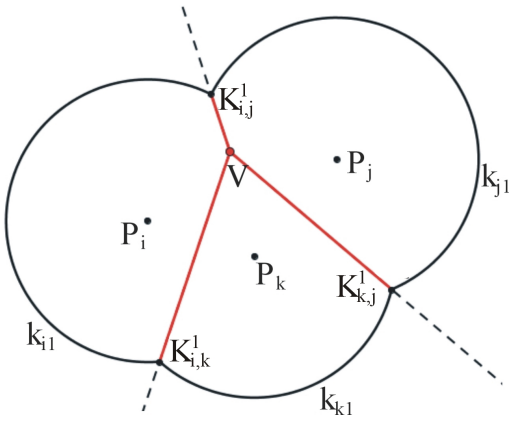
Na základě těchto informací můžeme říci, že metoda rostoucích regionů má dvě hlavní operace a to *průnik kružnic* a *zánik oblouku.*

Průnik kružnic Tato operace nastane ve chvíli, kdy průnik dvou kružnic se stane neprázdným. Namísto kružnice budeme pracovat s obloukem. Průsečíky, které vykreslují Voroneho hranu, mohou být dva,viz obr. 9 na straně 17, nebo pouze jeden, viz obr. 14.

Zánik oblouku Tato operace nastává ve chvíli, kdy zaniká jeden oblouk, viz obr. 9 na straně 17, nebo i více oblouků najednou, viz obr. 13. Kružnice se protnou v jednom bodě, v tomto bodě se vytvoří nový Voroneho vrchol.

Nyní uvedeme bližší popis algoritmu.

Inicializace Nejprve vytvoříme dvě fronty N, M, kružnice k_i se středy v generujících bodech $P_i \in S$ a poloměrem $r_i = 0$ a zvolíme konstantu t. O tuto konstantu t zvětšujeme,



Obrázek 14: Vykreslování Voroneho hrany pouze jedním průsečíkem

v každém kroku metody, poloměr kružnic. Do fronty N vložíme kružnice k_i, fronta M zůstane prozatím prázdná, budeme ji využívat později. Konstantu t volíme dostatečně malou.

Postup algoritmu Algoritmus je založen na hledání průsečíků kružnic. Pokud mají kružnice nulový poloměr nemají žádné průsečíky. Z toho důvodu nejprve zvětšíme poloměr kružnic k_i o konstantu t, platí tedy $r_i = r_i + t$. Následně hledáme, zda existuje průnik dvou kružnic $k_i, k_j, i \neq j$. Pokud existuje průnik dvou kružnic $k_i, k_j, i \neq j$ vložíme jejich průsečíky do fronty M, pokud neexistuje, zvětšíme poloměr kružnic k_i o konstantu t a znovu prohledáváme.

Kružnice k_i, v průběhu algoritmu, budou nahrazeny oblouky $k_{i,s}$, kde s udává počet oblouků, na které je daná kružnice k_i rozdělena. Může se tedy stát, že průnik dvou oblouků $k_{i,s}, k_{j,s}$ může mít pouze jeden společný bod a tento bod nemusí být nutně bod dotykový.

Po naplnění fronty M nastává druhá část algoritmu, která vezme první průsečík z fronty M a rozhoduje o tom, zda se jedná o operaci *průnik kružnic* nebo *zánik oblouku*. Po proběhnutí jedné ze dvou operací je vždy daný průsečík vymazán z fronty M. Tento postup opakujeme dokud není fronta M prázdná, následně opět zvětšíme poloměr kružnic k_i o konstantu t.

Ukončení algoritmu Ve chvíli, kdy jsou ve frontě N pouze oblouky se středy v generujících bodech ležících na hranách konvexního obalu, nemůže již nastat operace *zánik oblouku*. Průsečíky těchto oblouků nadále vykreslují Voroneho hranu. A Voroneho buňka odpovídající generujícímu bodu, jež je součástí hranice konvexního obalu množiny S, je neomezená. Proto, bude algoritmus dokončen vymazáním zbylých obouků z fronty N.

Algoritmus (rostoucích regionů pro Obecný Voroneho diagram)
Vstup: Množina generujících bodů S = $\{P_1, P_2, \ldots, P_n\}$.
Výstup: Voroneho diagram množiny S.

1. **Inicializace**
   (a) vytvoření fronty N a fronty M
   (b) vytvoření kružnic k_i se středy v generujících bodech $P_i \in S, i = 1, 2, \ldots, n$ a poloměry $r_i = 0$
   (c) vložení kružnic do fronty N a volba konstanty t
2. **Postup algoritmu**
   (a) zvětšení poloměru r_i kružnice k_i o konstantu t, tedy $r_i = r_i + t$
   (b) existuje neprázdný průnik libovolných dvou kružnic (*resp. oblouků*) k_i, k_j z fronty N
      i. if YES: označ průsečíky $K_{i,j}^{a,s}$, kde s udává počet kroků algoritmu, a kde $a = 1$ pro jeden společný průsečík dvou kružnic (*resp. oblouků*) a $a = 1, 2$ pro dva společné průsečíky dvou kružnic (*resp. oblouků*), vlož tyto průsečíky do fronty M
         A. výběr průsečíku $K_{i,j}^{a,s}$ z fronty M
         B. Switch($K_{i,j}^{a,s}$)
            - CASE: Průnik kružnic
              - rozhodni, zda průsečík $K_{i,j}^{a,s}$ je dotykový bod dvou kružnic (*resp. oblouků*) k_i, k_j
                * if YES: nedělej nic
                * if NO: spoj $K_{i,j}^{a,s}$ s $K_{i,j}^{a,(s-1)}$
              - nahrazení kružnic (*resp. oblouků*) k_i, k_j ve forně N novými oblouky k_i, k_j
              - vymazání průsečíku $K_{i,j}^{a,s}$ z fronty M
            - CASE: Zánik oblouku
              - průsečík $K_{i,j}^{a,s}$ je nový Voroneho vrchol
              - vymazání zaniklých oblouků z fronty N
              - vymazání průsečíku $K_{i,j}^{a,s}$ z fronty M
         C. rozhodnutí zda je fronta M prázdná
            - if YES: pokračování bodem 2a
            - if NO: pokračování bodem 2(b)iA
      ii. if NO: pokračování bodem 2a

3. **Ukončení algoritmu**

   (a) průsečíky oblouků z fronty N nechť jsou nevlastní

   (b) vymazání oblouků

Metoda rostoucích regionů, tak jak je zde prezentována, má několik nedořešených problémů. V následující části některé z těchto problémů uvedeme a současně navrhneme jejich možná řešení.

Problémy metody rostoucích regionů a jejich možná řešení Hlavní část metody spočívá v tom, že hledáme průsečíky kružnic (*resp. oblouků*). V popisu metody je psáno, že hledáme průsečíky libovolných dvou kružnic (*resp. oblouků*). Podle toho bychom museli hledat průsečíky každých dvou kružnic v množině N. To by pro větší množství generujících bodů bylo časově velmi náročné, proto by bylo vhodné omezit se na hledání průsečíků pouze těch kružnic, které se mohou vzájemně ovlivnit.

Jedno z možných řešení, jak omezit počet kružnic, pro které hledáme průnik, je jejich seřazení. Jednalo by se vlastně o dvě různá seřazení. Nejprve bychom seřadili všechny kružnice podle x-ové suřadnice a uložili je do množiny X. Následně seřadily kružnice podle y-ové souřadnice a uložili je do množiny Y. Předpokládejme, že kružnice k, pro kterou hledáme kružnice, se kterými kružnice k sousedí, je v množině X na pozici i (*resp. v množině Y na pozici j*). Pak bychom mohli hledat průnik kružnice k_i (*resp.* k_j), kde i, j označují pozici kružnice k v množinách X, Y, s kružnicemi v jejím okolí, tedy s kružnicemy $k_{i\pm1}, k_{j\pm1}$. S těmito vybranými kružnicemi budeme testovat, zda má kružnice k společný průnik.

V některých specialních případech, případně pro hustě rozmístěné generující body, by nemuselo stačit testovat společný průnik kružnice k pouze s jejími sousedními kružnicemi $k_{i\pm1}, k_{j\pm1}$ v množinách X a Y, ale bylo by vhodné testovat průnik kružnice k i s dalšími kružnicemi v jejím okolí, v množinách X, Y. Kružnice se, kterými bychom testovaly by mohly být, např. $k_{i\pm2}, k_{j\pm2}, k_{i\pm1}, k_{j\pm1}$. Uvědomme si, že i přes to, není tato myšlenka zcela stoprocentní.

Další z možných problémů je, jak rozhodnout, která operace, zda *průnik kružnic* nebo *zánik oblouku*, se pro průsečík $K^{a,s}_{i,j}$ vykoná. Při vyhledávání průsečíku kružnic, hledáme vždy průsečíky dvou krůžnic. Operace *zánik oblouku* nastává ve chvíly, kdy se v jednom bodě protínají tři kružnice. Toho můžeme využít, neboť to znamená, že pokud pro průsečík $K^{a,s}_{i,j}$ nastává operace *zánik oblouku*, pak v množine M je jiný průsečík $K^{a,s}_{i,k}$, pro který platí $K^{a,s}_{i,j} = K^{a,s}_{i,k}$. Můžeme tedy prohledat množinu M a zeptat se, zda jsou v ní průsečíky, které se rovnají, pokud ano jedná se o operaci *zánik oblouku*, pokud ne jedná se o operaci *průnik kružnic*.

To nás, ale přivádí k dalšímu z možných problémů. Průsečíky kružnic, které mají být novým Voroneho vrcholem se nemusejí rovnat. Mohou se lišit v setinách či méně, podle toho, jak malá je zvolena konstanta t. Tento problém můžeme vyřešit tak, že

budeme brát přesnost např. na dvě desetiná místa.

V této části jistě nebyly uvedeny všechny možné problémy, které při metodě rostoucích regionů mohou nastan. Nicméně snaha byla uvést ty nejhlavnější z problémů.

3 Voroneho diagram v L_1 metrice

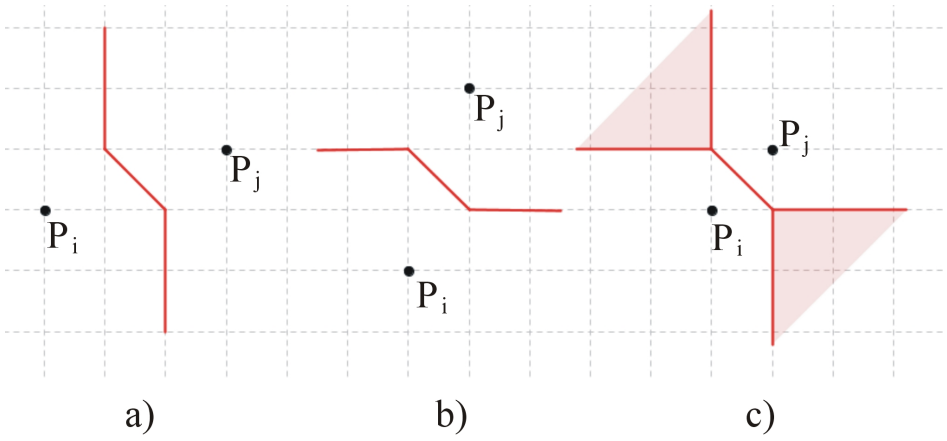
Metriku L_1 můžeme také najít pod názvem Manhattanská metrika (*"Manhattan distance"*). Jedná se o jednu z L_p metrik, kde $p = 1$ [12], [5].

Definice 3.1 *Mějme dva body $P = (p_1, p_2), Q = (q_1, q_2)$, pak L_1 vzdálenost bodů P, Q definujeme vztahem*

$$d_1(P, Q) = |p_1 - q_1| + |p_2 - q_2|.$$

•

Mějme $S = \{P_1, P_2, \ldots, P_n\}$ množinu generujících bodů. Pak Voroneho diagram v L_1 metrice je definován obdobně, jako Obecný Voroneho diagram s tím rozdílem, že namísto Eukleidovské metriky využívá metriku L_1. Osou úsečky $B_1(P_i, P_j)$ již nejsou obecně přímky, ale lomené čáry, viz obr. 15, [5],[12].



Obrázek 15: Osa úsečky dvou bodů P_i, P_j v L_1 metrice

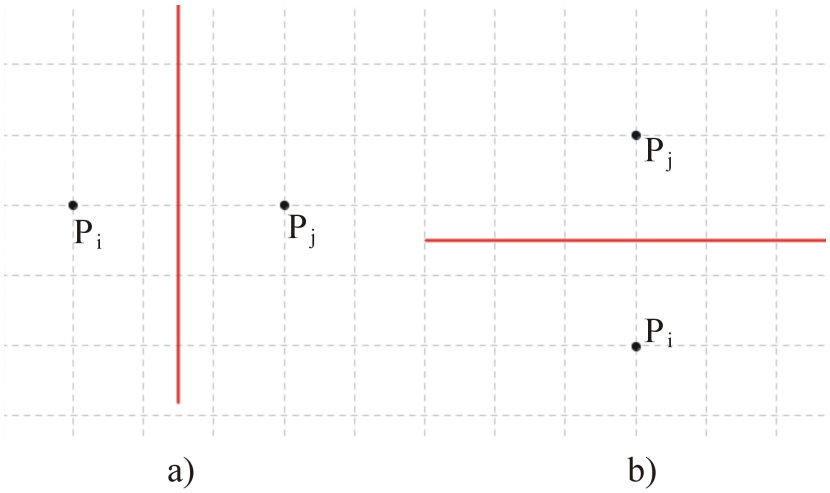
Ve specialních případech je osa úsečky $B_1(P_i, P_j)$ pro dva generující body P_i, P_j přímka, viz obr. 16

Poznámka 7 *Všimněme si, že osa úsečky $B_1(P_i, P_j)$, dvou generujících bodů $P_i = (p_{i_x}, p_{i_y}), P_j = (p_{j_x}, p_{j_y})$, bude přímkou pokud $p_{i_x} = p_{j_x}$, viz obr. 16b), nebo pokud $p_{i_y} = p_{j_y}$, viz obr. 16a).*

•

V případě, kdy generující body P_i, P_j jsou vrcholy čtverce ležící na úhlopříčce, osa úsečky mezi nimi není lomenou čarou, viz obr. 15c). Součástí osy úsečky je i část roviny. Takovýto případ rozmístění generujících bodů ztěžuje vykreslení Voroneho diagramu v L_1 metrice. Z toho důvodu množinu S generujících bodů omezíme [12].

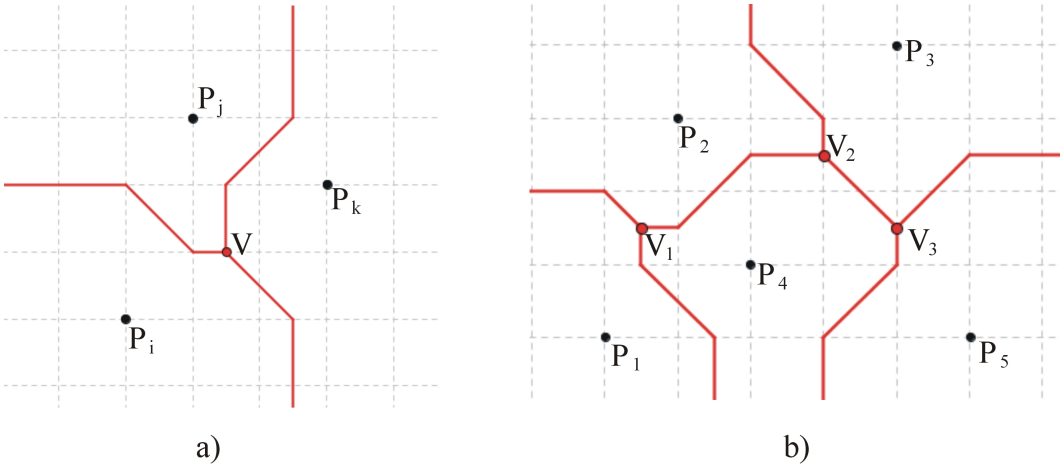
Pro generující body množiny S budeme předpokládat, že



Obrázek 16: Osa úsečky dvou bodů P_i, P_j v L_1 metrice

- žádné dva a více generujících bodů neleží na přímce, která se souřadnicovými osami svírá úhel $\pm 45°$.

Jak již bylo řečeno, osa úsečky $B_1(P_i, P_j)$ dvou generujících bodů, s různými x-ovými a y-ovými souřadnicemi, je lomená čára. Tato lomená čára se skládá z horizontálních, vertikálních a diagonálních částí přímek, viz obr. 15. Diagonální část svírá se souřadnicovými osami úhel $\pm 45°$.



Obrázek 17: Voroneho diagram: a) pro tři generující body P_i, P_j, P_k v L_1 metrice; b) v L_1 metrice

Poznámka 8 *Pro Voroneho vrchol V v metrice L_1 platí*

$$d_1(V, P_i) = d_1(V, P_j) = d_1(V, P_k)$$

a příklad je znázorněn na obrázku 17a).

•

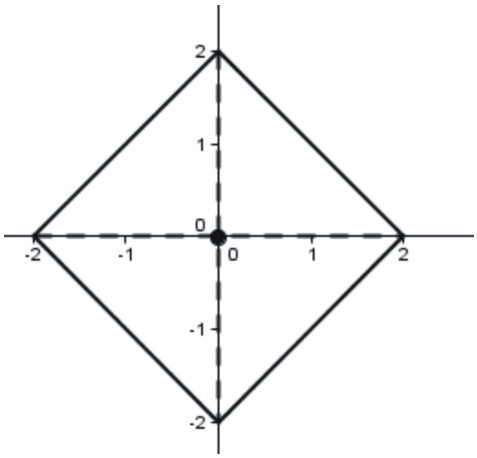
Na obrázku 17b) je příklad Voroneho digramu v L_1 metrice.

3.1 Metoda rostoucích regionů pro Voroneho diagram v L_1 metrice

Metoda rostoucích regionů pro Voroneho diagram v L_1 metrice je obdobná, jako pro Obecný Voroneho diagram. Metoda rostoucích regionů vytváří v každém generujícím bodě množiny S kružnici s nulovým poloměrem, který se postupně zvětšuje. Průsečíky kružnic, popř. jejich oblouků, vykreslují Voroneho hrany, dokud nevznikne nový Voroneho vrchol, ten vzniká ve společném průniku tří kružnic, popř. jejich oblouků.

Poznámka 9 *Uvědomme si, jak vypadá kružnice v L_1 metrice. Kružnice v L_1 metrice je čtverec, jehož úhlopříčky jsou rovnoběžné se souřadnicovými osami, viz obr. 18.*

•



Obrázek 18: Kružnice v L_1 metrice

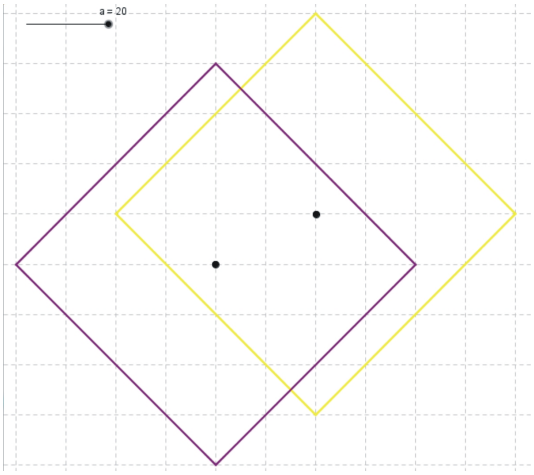
Součástí práce je znázornění této metody v programu Geogebra. Protože Geogebra není program uzpůsobený k programování, bylo nutno metodu rostoucích regionů přizpůsobit pro její znázornění, následujícím zůsobem.

3.1.1 Popis metody rostoucích regionů pro program Geogebra

Hlavní myšlenka metody rostoucích regionů, při konstrukci Voroneho diagramu v programu Geogebra, je zachována. Průsečíky dvou kružnic náleží Voroneho hraně a současně průnik tří kružnic tvoří Voroneho vrchol.

Nicméně v programu nevykreslujeme Voroneho hrany, ale Voroneho buňky.
To umožňuje funkce *stopa*. Její vlastností je, že při pohybu objektu, u něhož je funkce *stopa* zapnuta, se vykresluje stopa tohoto objektu.

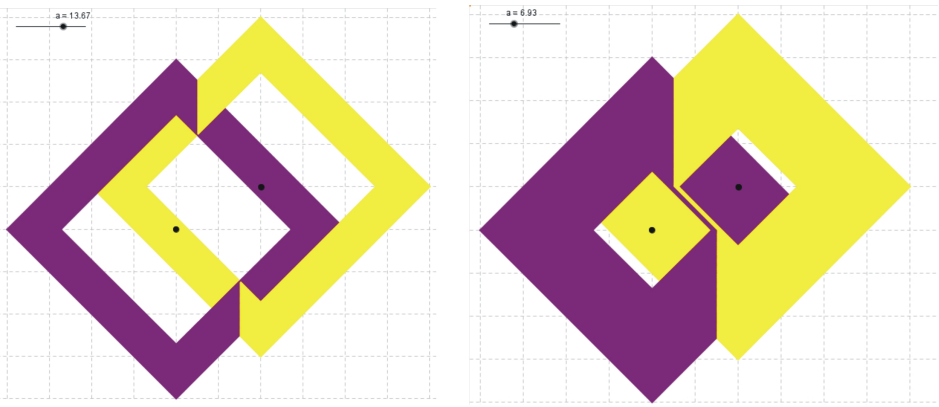
Metoda rostoucích regionů, při průniku kružnic, pracuje pouze s vnějšími oblouky těchto kružnic. Tuto myšlenku nemůžeme využít, neboť Geogebra nám neumožňuje, v průběhu procesu, zanedbat část kružnice a pracovat pouze s oblouky. Z toho důvodu nebudeme v počátku volit poloměr kružnic nulový, ale naopak nastavíme poloměr na konkrétní hodnotu, ideálně tak aby se kružnice s poloměrem a protínaly, tato hodnota je zvolena $a = 20$, viz obr. 19. Poloměr pak necháme zmenšovat do $a = 0$.



Obrázek 19: Nastavení poloměru na konkrétní hodnotu v progranu Geogebra

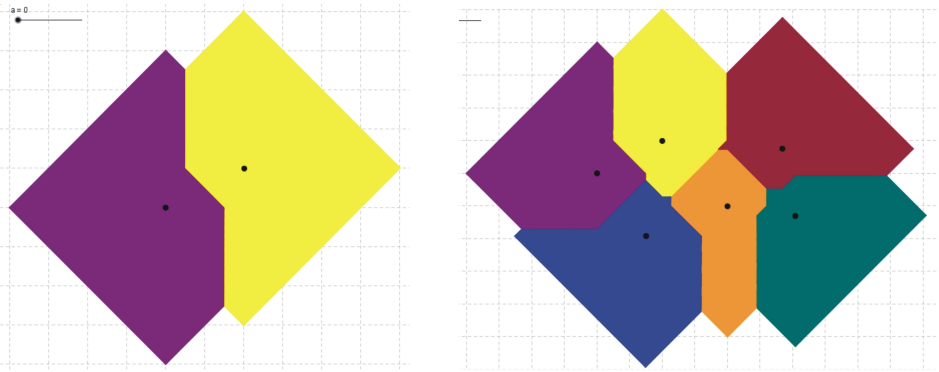
Průnik kružnic je neprázdný. Při zmenšování poloměru nám vnitřní oblouk také zanechává stopu. Díky tomu, že namísto zvětšování poloměru, poloměr snižujeme bude stopa, jež zanechávají vnitřní oblouky překreslena, viz obr. 20.

Výsledný Voroneho diagram v L_1 metrice pro dva generující body je znázorněn na obrázku 21a). Na obrázku 21b) je příklad Voroneho diagramu v L_1 metrice.



a) b)

Obrázek 20: Překrelení stopy vnitřních oblouků, kde: a) $a = 13,67$; b) $a = 6,93$



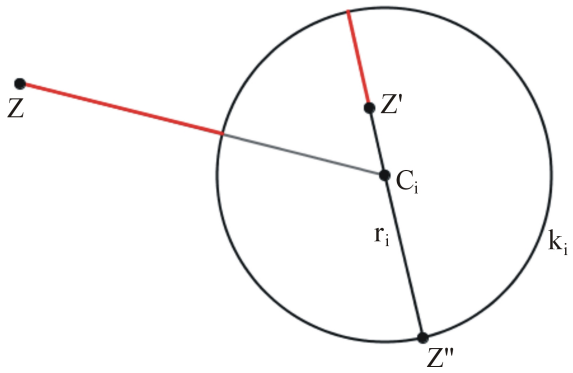
a) b)

Obrázek 21: Voroneho diagram v L_1 metrice: a) pro dva generující body; b) pro šest generujících bodů

4 Voroneho diagram kružnic

Mějme $S = \{k_1, k_2, \ldots, k_n\}$ množinu generujících kružnic $k_i = (C_i, r_i)$, kde $C_i = (x_i, y_i)$ jsou středy kružnic a r_i jejich poloměry.

Nechť S je množina generujících kružnic s různými poloměry v rovině. Pro každý bod Z roviny definujeme $d_e(Z, k_i)$, jako Eukleidovskou vzdálenost bodu Z od nejbližšího bodu kružnice k_i.



Obrázek 22: Vzdálenost bodu Z od kružnice k_i

Definice 4.1 *Mějme libovolný bod Z roviny a generující kružnici $k_i \in S$. Pak vzdálenost bodu Z a generující kružnice k_i definujeme, jako nejkratší Eukleidovskou vzdálenost bodu Z od kružnice k_i. Tuto vzdálenost budeme značit $d_e(Z, k_i)$ a platí*

$$d_e(Z, k_i) = |d_e(Z, C_i) - r_i| \tag{7}$$

kde C_i je střed generující kružnice k_i a r_i je její poloměr, viz obr. 22.

•

Poznámka 10 *Vzdálenost libovolného bodu Z roviny od generující kružnice $k_i \in S$ se středem v bodě C_i, z definice 4.1, lze rozdělit na tři případy.*
Bod Z leží vně generující kružnice k_i, pak platí

$$d_e(Z, k_i) = d_e(Z, C_i) - r_i,$$

bod Z leží uvnitř generující kružnice k_i, pak platí

$$d_e(Z, k_i) = r_i - d_e(Z, C_i),$$

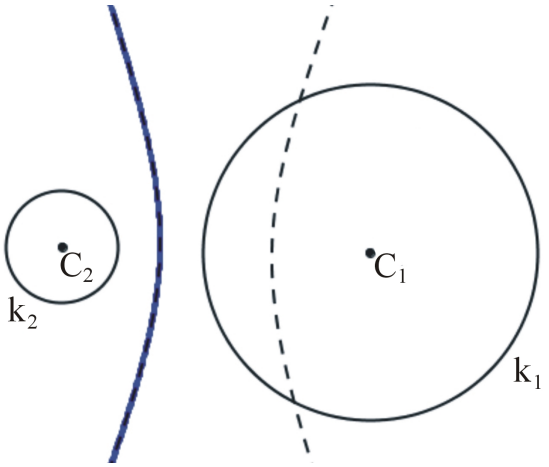
bod Z leží na generující kružnici k_i, pak platí

$$d_e(Z, k_i) = d_e(Z, C_i) - r_i = r_i - d_e(Z, C_i) = 0.$$

V následujícím textu budeme předpokládat, že množina S generujících kružnic má následující vlastnosti.
Generující kružice $k_i \in S$

- jsou disjunktní, tj. $k_i \cap k_j = \emptyset$, pro každé $i \neq j$
- jejich poloměry jsou nenulové, tedy $r_i \neq 0$
- žádná generující kružnice neleží uvnitř jiné generující kružnice.

Hrany Voroneho diagramu pro množinu generujících bodů, jsou částmi přímek. Pokud budeme konstruovat Voroneho diagram pro množinu generujících kružnic pak Voroneho hrana již nebude částí přímky, ale bude se jednat o část hyperboly,
viz obr. 23.



Obrázek 23: Voroneho hrana mezi dvěmi generujícími kružnicemi

Lemma 4.1 *Množina bodů, které mají stejnou vzdálenost od dvou neprotínajících se kružnic k_1, k_2 se středy v bodech C_1, C_2 a s různými poloměry r_1, r_2, tvoří část hyperboly s ohnisky v bodech C_1, C_2, viz obr. 23.*

•

Důkaz Mějme dvě kružnice k_1, k_2 se středy v bodech C_1, C_2 a poloměry r_1, r_2, kde $r_1 \neq r_2$.
Hledáme množinu bodů X_i, které mají stejnou vzdálenost od kružnic k_1, k_2.

Z definice hyperboly, jestliže bod X_i leží na hyperbole, pak platí

$$|d_e(C_1, X_i) - d_e(C_2, X_i)| = a, \tag{8}$$

kde a je konstanta a C_1, C_2 jsou ohniska hyperboly. Vzdálenost $d_e(C_1, X_i)$ je součet poloměru kružnice k_1 se vzdáleností w_i bodu X_i od kružnice k_1, tedy $w_i = d_e(X_i, k_1)$. Obdobně, $d_e(C_2, X_i)$ je součet poloměru kružnice k_2 se vzdáleností v_i bodu X_i od kružnice k_2, tedy $v_i = d_e(X_i, k_2)$. Na základě toho, že hledáme množinu bodů, která má stejnou vzdálenost od dvou kružnic k_1, k_2, platí

$$w_i = v_i.$$

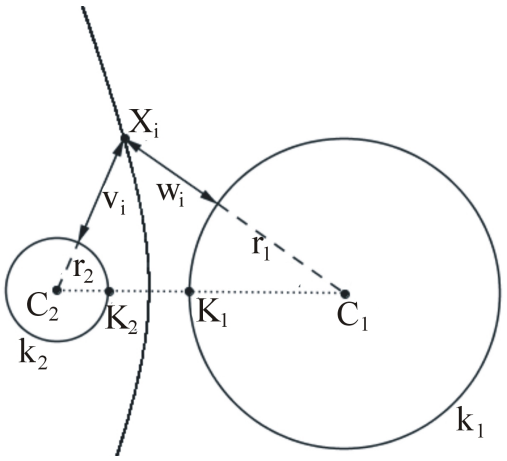
Rovnost (8) lze upravit následujícím způsobem

$$|(r_1 + w_i) - (r_2 + w_i)| = |r_1 - r_2| = a.$$

Tím získáváme konstatní rozdíl vzdáleností pro každý bod X_i hledané množiny a tedy výsledná množina je hyperbola, viz obr. 24.

Zájímá nás pouze jedna větev této hyperboly, která má ohniska ve středech C_1, C_2 kružnic k_1, k_2. Střed hyperboly je středem úsečky C_1C_2 a vrchol dané větve hyperboly je střed úsečky K_1K_2, kde body K_1, K_2 jsou průsečíky úsečky C_1C_2 s kružnicemi k_1, k_2, viz obr. 24.

○



Obrázek 24: Vzdálenost bodu X_i od dvou kružnic

Poznámka 11 *Pokud se poloměry obou kružnic k_i, k_j rovnají, tedy $r_i = r_j$, pak množina bodů, které mají stejnou vzdálenost od kružnic k_i, k_j je přímka. Tato přímka $B(C_i, C_j)$ je osou úsečky C_i, C_j, kde body C_i, C_j jsou středy kružnic k_i, k_j.*

•

Věta 4.1 *Nechť generující kružnice $k_i \in S$ jsou disjunktní, jejich poloměry $r_i \neq 0$ a žádná z generujících kružnic neleží uvnitř jiné generující kružnice. Pak hrany Voroneho diagramu kružnic jsou tvořeny částmi hyperbol, popř. přímek.*

•

Důkaz Vyplývá z lemmatu 4.1.

○

Poznámka 12 *Voroneho diagram kružnic v rovině je speciálním případem Aditivního Voroneho diagramu, kde každému generujícímu bodu je přiřazena váha. Váha generujícího bodu odpovídá poloměru kružnice. Bližší informace o Aditivním Voroneho diagramu je možné si přečíst v [4].*

•

4.1 Algoritmy pro Voroneho diagram kružnic

Mějme $S = \{k_1, k_2, \ldots, k_n\}$ množinu generujících kružnic $k_i = (C_i, r_i)$, kde $C_i = (x_i, y_i)$ jsou středy kružnic a r_i jejich poloměry, n je počet generujících kružnic. Pro jednoduchost budeme předpokládat, že generující kružnice jsou disjunktní, jejich poloměry jsou nenulové, tedy $r_i \neq 0$ a žádná generující kružnice neleží uvnitř jiné.

Poznámka 13 *Bod Z náleží Voroneho buňce $\nu(k_i)$, pro generující kružnici k_i, pouze tehdy, platí-li, že $d_e(Z, k_i) < d_e(Z, k_j)$. V případě rovnosti bude bod Z ležet na Voroneho hraně dvou Voroneho buněk odpovídajících generujícím kružnicím k_i, k_j.*

•

V následujících dvou podkapitolách uvedeme dva algoritmy pro konstrukci Voroneho diagramu kružnic a to Inkrementální algoritmus, viz kapitola 2.1.1 a algoritmus Plane sweep, viz kapitola 2.1.2. V obou případech algoritmy popíšeme pro výše zavedenou množinu generujících kružnic S, tedy pro generující kružnice, které neleží uvnitř jiných generujících kružnic, jsou disjunktní a jejich poloměry jsou nenulové. Následně, pro algoritmus Plane sweep shrneme rozdíly a postup pro modifikovanou množinu S, kde připustíme nulové poloměry, průniky kružnic a případy, kdy jedna kružnice obsahuje jinou.

4.1.1 Inkrementální algoritmus pro Voroneho diagram kružnic

Inkrementální algoritmus pro Voroneho diagram množiny $S = \{k_1, k_2, \ldots, k_n\}$ generujících kružnic je obdobný, jako algoritmus pro Voroneho diagram množiny bodů, viz kapitola 2.1.1. Hlavní rozdíl spočívá v tom, že hrany Voroneho diagramu kružnic již nejsou částmi přímek, ležících na ose úsečky $B(P_i, P_j)$, kde $i \neq j$, nýbrž částmi hyperbol, viz lemma 4.1.

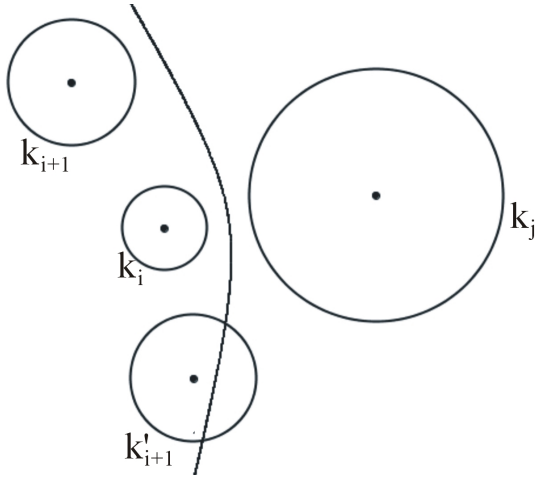
Poznámka 14 *Všimněme si, že množina bodů, která má stejnou vzdálenost od dvou kružnic s různým poloměrem je hyperbola, lépe řečeno jedna její větev. Současně, ale tato hyperbola pro dvě generující kružnice nahrazuje funkci osy úsečky $B(P_i, P_j)$ pro*

dva body P_i, P_j, kde $i \neq j$. Proto ji označíme $B_h(k_i, k_j)$, kde $k_i, k_j \in S$ jsou generující kružnice pro $i \neq j$ a platí

$$B_h(k_i, k_j) = \{X | d_e(X, k_1) = d_e(X, k_2)\},$$

kde bod X je bod hyperboly. Potom hrany Voroneho diagramu kružnic jsou částmi těchto hyperbol $B_h(k_i, k_j)$.

•



Obrázek 25: Příslušnost nově vložené generující kružnice k_{i+1} Voroneho buňce

Algoritmus pro množinu $S = \{k_1, k_2, \ldots, k_n\}$ generujících kružnic se ovšem neliší od algoritmu pro $S = \{P_1, P_2, \ldots, P_n\}$ generujících bodů.

Inkrementální algoritmus pro množinu $S = \{k_1, k_2, \ldots, k_n\}$ generujících kružnic spočívá v nalezení Voroneho diagramu pro dvě generující kružnice a jeho následné modifikaci z i na $i + 1$ generujících kružnic, pro každé $i = 2, 3, \ldots, n$.

Při přidání nové generující kružnice k_{i+1} do stávajícího Voroneho diagramu bude nutné provést její lokalizaci, tzn. nalézt Voroneho buňku $\nu(k_i)$, ve které se nově přidaná generující kružnice k_{i+1} nachází. Na obrázku 25 je vidět, že nově přidaná generující kružnice k_{i+1} může ležet zcela uvnitř Voroneho buňky $\nu(k_i)$, nebo může protínat hranici Voroneho diagramu mezi dvěmi, popřípadě třemi Voroneho buňkami, za předpokladu, že Voroneho diagram je nedegenerovaný.

O příslušnosti nově přidané generující kružnice k_{i+1} můžeme rozhodnout na základě její vzdálenosti od generujících kružnic k_i, stávajícího Voroneho diagramu, popřípadě na základě vzdálenosti středu C_{i+1}, nově přidané generující kružnice k_{i+1} od stávajících generujících kružnic k_i.

Proto, abychom mohli rozhodnout o příslušnosti generující kružnice k_{i+1} k dané Voroneho buňce, potřebujeme definovat vzdálenost dvou kružnic.

Definice 4.2 *Mějme dvě neprotínající se generující kružnice $k_i, k_j \in S$ se středy v bodech C_i, C_j a poloměry r_i, r_j. Pak* ***vzdálenost dvou kružnic*** *označme $d_e(k_i, k_j)$ a platí*

$$d_e(k_i, k_j) = d_e(C_i, C_j) - r_i - r_j. \tag{9}$$

•

Lemma 4.2 *Nově přidaná generující kružnice k_{i+1} náleží Voroneho buňce $\nu(k_i)$, stávajícího Voroneho diagramu, pokud platí*

$$d_e(k_{i+1}, k_i) < d_e(k_{i+1}, k_j), \tag{10}$$

popřípadě

$$d_e(C_{i+1}, k_i) < d_e(C_{i+1}, k_j), \tag{11}$$

kde $i \neq j$, jak je vidět na obrázku 25.

•

Důkaz Vycházíme z definice Voroneho buňky, viz definice 2.2. Voroneho buňka $\nu(k_i)$ obsahuje právě ty body X roviny, které mají od k_i menší vzdálenost, než od libovolné jiné kružnice množiny S. Platí tedy, že $X \in \nu(k_i)$, pokud

$$d_e(X, k_i) < d_e(X, k_j). \tag{12}$$

Jestliže bod X, v nerovnici (12) nahradíme středem C_{i+1} příslušné generující kružnice k_{i+1}, pak můžeme říct, že $C_{i+1} \in \nu(k_i)$, pokud platí $d_e(C_{i+1}, k_i) < d_e(C_{i+1}, k_j)$, kde $i \neq j$. Což je nerovnice (11).

Současně si uvědomme, že vzdálenost dvou kružnic k_i, k_j je
$d_e(k_i, k_j) = |d_e(C_i, C_j) - r_i - r_j|$, kde C_i, C_j jsou středy kružnic k_i, k_j a r_i, r_j jsou jejich poloměry.

Jestliže bod X, v nerovnici (12), nahradíme příslušnou generující kružnicí k_{i+1}, pak můžeme říci, že $k_{i+1} \in \nu(k_i)$, pokud platí $d_e(k_{i+1}, k_i) < d_e(k_{i+1}, k_j)$, kde $i \neq j$. Což je nerovnice (10).

◦

Algoritmus (Inkrementální pro Voroneho diagram kružnic)
Vstup: Množina generujících kružnic S = $\{k_1, k_2, \ldots, k_n\}$.
Výstup: Voroneho diagram množiny S.

1. **Inicializace**

   (a) vytvoření fronty F a vložení generujících kružnic $k_1, k_2, \ldots, k_n$ do fronty

2. **Postup algoritmu**

(a) výběr dvou generujících kružnic k_1, k_2 z fronty F a vytvoření Voroneho diagramu pro tyto generující kružnice

(b) výběr kružnice k_{i+1} z fronty F, kde i udává počet generujících kružnic stávajícího Voroneho diagramu

(c) určení lokalizace kružnice k_{i+1} ve stávajícím Voroneho diagramu, označení generující kružnice Voroneho buňky, ve které se kružnice k_{i+1} nachází, bude k_{ij}, kde $j = 1, 2, \ldots$ a udává počet Voroneho buněk, které budou ovlivněny kružnicí k_{i+1}

   i. jestliže střed C_{i+1} generující kružnice k_{i+1} leží na Voroneho hraně: zvolení libovolné z Voroneho buněk, které náléží daná hrana

   ii. jestliže střed C_{i+1} generující kružnice k_{i+1} neleží na Voroneho hraně: určení v jaké Voroneho buňce, stávajícího Voroneho diagramu, se kružnice k_{i+1} nachází

(d) nalezení osy mezi dvěmi kružnicemi $B_h(k_{i+1}, k_{ij})$

(e) nalezení průsečíků osy $B_h(k_{i+1}, k_{ij})$ s hranicí Voroneho buňky $\nu(k_{ij})$

(f) rozhodnutí o počtu průsečíků osy $B_h(k_{i+1}, k_{ij})$ s hranicí Voroneho buňky $\nu(k_{ij})$

   i. neexistuje žádný průsečík

      A. pokračování bodem (2b)

   ii. existuje pouze jeden průsečík

      A. tento průsečík určí následující Voroneho buňku, která bude ovlivněna kružnicí k_{i+1}, označení generující kružnice této buňky bude $k_{i(j+1)}$

      B. nalezení osy $B_h(k_{i+1}, k_{i(j+1)})$ a jejích průsečíků s hranicí Voroneho buňky $\nu(k_{i(j+1)})$

      C. rozhodnutí o počtu průsečíků

         - existuje jeden průsečík: pokračování bodem (2g)
         - existují dva průsečíky: pokračování následujícím bodem

      D. zvolení průsečíku, který neleží na společné hraně Voroneho buněk $\nu(k_{ij}), \nu(k_{i(j+1)})$, pokračování bodem (2(f)iiA)

   iii. existují dva průsečíky

      A. výběr jednoho libovolného průsečíku, čímž je určena Voroneho buňka, která jako další bude ovlivněna generující kružnicí k_{i+1}, označení generující kružnice této buňky bude $k_{i(j+1)}$

B. nalezení osy $B_h(k_{i+1}, h_{i(j+1)})$ a jejích průsečíků s hranicí Voroneho buňky $\nu(k_{i(j+1)})$

C. rozhodnutí o počtu průsečíků

- existuje jeden průsečík: zvolení druhého z původně nalezených průsečíků příslušných Voroneho buňce $\nu(k_{i1})$ a pokračování předcházejícím bodem
- existují dva průsečíky: pokračování následujícím bodem

D. zvolení průsečíku, který neleží na společné hraně dvou Voroneho buněk $\nu(k_{ij}), \nu(k_{i(j+1)})$

E. ověření zda zvolený průsečík náleží hraně Voroneho buňky $\nu(k_{i1})$

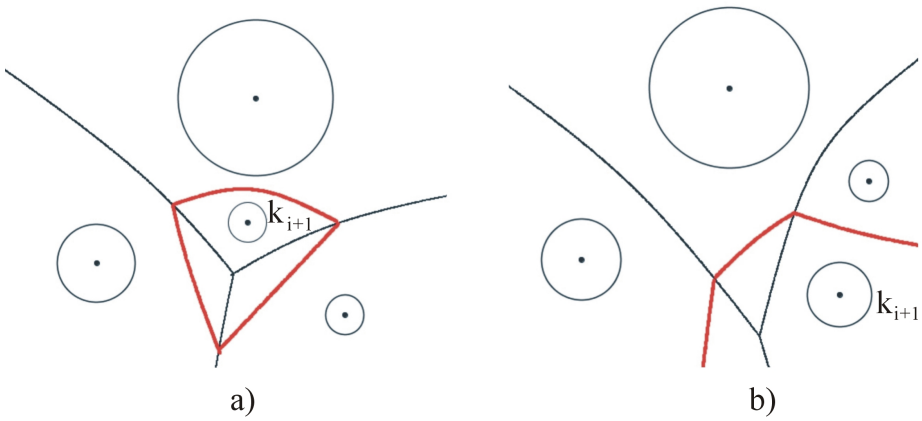
- if YES: přejít na bod (2g)
- if NO: pokračování bodem (2(f)iiiB)

(g) vyrušení hran uvnitř nově vzniklé Voroneho buňky $\nu(k_{i+1})$ a pokračování bodem (2b)

(h) rozhodnutí zda $i + 1 = n$

i. if Yes: konec algoritmu

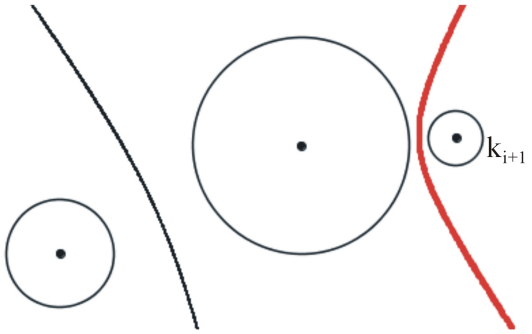
ii. if No: pokračování bodem (2b)



Obrázek 26: Inkrementální algoritmus pro Voroneho diagram kružnic

Poznámka 15 *Všimněme si, že při hledání průsečíků osy dvou generujících kružnic $B_h(k_{i+1}, k_{i1})$ s hranicí Voroneho buňky $\nu(k_{i1})$ mohou nastat tři případy. Průsečík osy $B_h(k_{i+1}, k_{i1})$ s Voroneho buňkou $\nu(k_{i1})$ neexistuje, viz obr. 27, je pouze jeden, viz obr. 26b), nebo jsou právě dva, viz obr. 26a).*

•



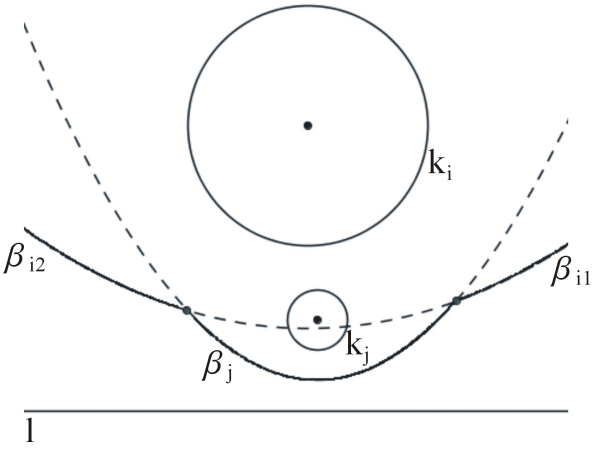
Obrázek 27: Inkrementální algoritmus pro Voroneho diagram kružnic

4.1.2 Algoritmus "Plane sweep"pro Voroneho diagram kružnic

Algoritmus Plane sweep pro Voroneho diagram kružnic využívá stejného přístupu jako algoritmus pro Obecný Voroneho diagram, viz kapitola 2.1.2.

Hlavní část algoritmu spočívá ve využití tzv. zametací přímky l jež se pohybuje shora dolů a tzv. beach line b, jež se skládá z parabolických oblouků β_i, kde každý oblouk odpovídá jiné generující kružnici k_i. Může se stát, že některý z parabolických oblouků β_i bude rozdělen jiným parabolickým obloukem β_j na dvě části, viz obr. 28.

Hlavní rozdíl mezi algoritmem Plane sweep pro Obecný Voroneho diagram a algoritmem pro Voroneho diagram kružnic spočívá ve vytvoření beach line b. Navíc ke dvěma základním operacím "site event" *(tj. objevení nového generujícího bodu na zametací přímce l)* a "circle event" *(tj. zánik parabolického oblouku β_i)* přibude operace "merge event". Tato operace nastává ve chvíli, kdy zametací přímka opouští generující kružnici. O jednotlivých operacích budeme hovořit později.



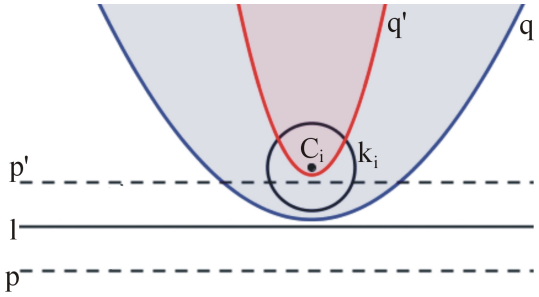
Obrázek 28: Rozdělení parabolického oblouku β_i jiným obloukem β_j na beach line

Lemma 4.3 *Množina bodů $X_i \in l^+$, která má stejnou vzdálenost od kružnice $k_i \in S$ a přímky l je parabola. Ohniskem této paraboly je střed $C_i \in l^+$ kružnice k_i a řídící přímka p je rovnoběžná s přímkou l ve vzdálenosti r_i, kde r_i je poloměr kružnice k_i,viz obr. 29.*

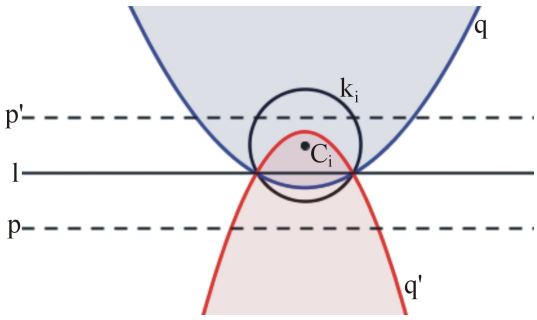
•

Ještě než provedeme důkaz lemmatu 4.3, ukážeme si možné polohy vzniklé paraboly vzhledem k její řídící přímce.

Všimněme si, že řídící přímka, z lemmatu 4.3, může ležet nad zametací přímou l, tedy $p' \in l^+$, nebo může ležet pod l, tedy $p \in l^-$. Sestrojíme paraboly q, q' pro obě řídící přímky p, p', pro případ, kdy generující kružnice k_i neprotíná zametací přímku l a pro případ, kdy k_i protíná zametací přímku l. Oba tyto případy jsou znázorněny na obrázcích 29, 30.



Obrázek 29: Paraboly q, q' s řídícími přímkami p, p' v případě, že zametací přímka neprotíná generující kružnici



Obrázek 30: Paraboly q, q' s řídícími přímkami p, p' v případě, že zametací přímka protíná generující kružnici

Jak je vidět na obrázku 29, pokud zametací přímka l neprotíná generující kružnici k_i, pak obě paraboly q, q' s ohniskem ve středu C_i kružnice k_i a řídícími přímkami p, p', jsou konvexní (*tzn. že vrchol paraboly má nejnižší y-ovou souřadnici, oproti ostatním bodů ležícím na danné parabole*) a parabola q' leží celá uvnitř paraboly q. Z toho důvodu parabola q' není pro naše účely využitelná, mimo jiné proto, že nesplňuje požadavek na stejnou vzdálenost od kružnice k_i jako od zametací přímky l.

Na obrázku 30 je druhý případ, kdy zametací přímka l protíná generující kružnici k_i. V tomto případě je parabola q konvexní a parabola q' je konkávní (*tzn. že vrchol paraboly má nejvyšší y-ovou souřadnici, oproti ostatním bodů ležícím na danné parabole*). Obě tyto paraboly splňují požadavek na stejnou vzdálenost od generující kružnice k_i jako od zametací přímky l. Vzhledem k tomu, že algoritmus Plane sweep neuchovává informace pod l a beach line b je sestavena pouze z parabolických oblouků nacházejících se nad zametací přímkou l, pak konkávní parabola q' nám doplní chybějící část beach line uvnitř generující kružnice k_i.

Nyní již máme potřebné informace a můžeme provést důkaz lemmatu 4.3.

Důkaz Nechť máme kružnici k_i se středem v bodě C_i, poloměrem $r_i \neq 0$ a nechť máme zametací přímku l a přímky p, p', kde $p, p' \parallel l$ a vzdálenost pl, resp. $p'l$ je rovna r_i, současně platí $p \in l^-$ a $p' \in l^+$.

Z definice paraboly, jestliže bod X_i leží na parabole s ohniskem C_i a řídící přímkou p, platí

$$d_e(X_i, C_i) = d_e(X_i, p). \tag{13}$$

Z definice 4.1 víme, že vzdálenost vnějšího bodu X_i od kružnice k_i je

$$d_e(X_i, k_i) = d_e(X_i, C_i) - r_i \tag{14}$$

a vzdálenost vnitřního bodu X_i od kružnice k_i je

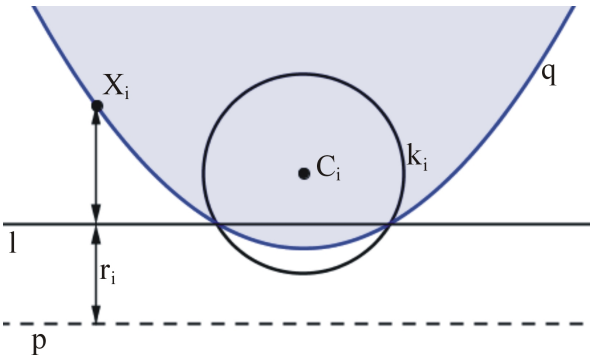
$$d_e(X_i, k_i) = r_i - d_e(X_i, C_i). \tag{15}$$

Důkaz rozdělíme na dvě části. Na část, kdy nás zajímají pouze vnější body kružnice k_i a na část, kdy nás budou zajímat pouze vnitřní body X_i kružnice k_i.

Nyní nás budou zajímat pouze body, které leží vně kružnce k_i a tedy pro vzdálenost vnějšího bodu X_i od kružnice k_i platí rovnost (14).

Po úpravě dostáváme

$$d_e(X_i, C_i) = d_e(X_i, k_i) + r_i. \tag{16}$$



Obrázek 31: Vzdálenost bodu X_i od řídící přímky p

Přímky p, l jsou od sebe vzdáleny o poloměr r_i, viz obr. 31, a lze tedy psát

$$d_e(X_i, p) = d_e(X_i, l) + r_i. \tag{17}$$

Po dosazení vztahů (16) a (17) do rovnice (13) dostaneme

$$d_e(X_i, k_i) + r_i = d_e(X_i, l) + r_i$$

a tedy platí

$$d_e(X_i, k_i) = d_e(X_i, l). \tag{18}$$

Z rovnice (18) vidíme, že vzdálenost vnějších bodů X_i od kružnice je stejná, jako vzdálenost vnějších bodů X_i od přímky l. Tím jsme dokázali, že množina vnějších bodů X_i, která má stejnou vzdálenost od kružnice k_i a od přímky l tvoří parabolu.

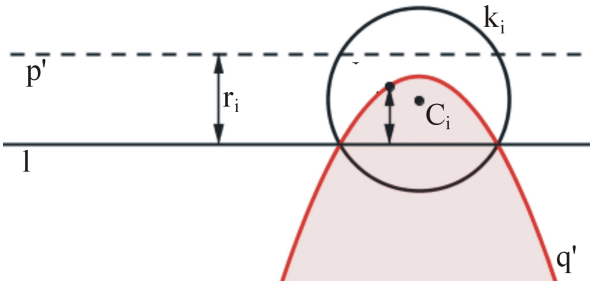
Nyní se zaměříme na body, které leží uvnitř kružnce k_i a tedy pro vzdálenost vnitřního bodu X_i od kružnice k_i platí rovnost (15).

Po úpravě dostáváme

$$d_e(X_i, C_i) = r_i - d_e(X_i, k_i). \tag{19}$$

Přímky p', l jsou od sebe vzdáleny o poloměr r_i, viz obr. 32, a lze tedy psát

$$d_e(X_i, p') = r_i - d_e(X_i, l). \tag{20}$$



Obrázek 32: Vzdálenost bodu X_i od řídící přímky p'

Po dosazení vztahů (19) a (20) do rovnice (13) dostaneme

$$r_i - d_e(X_i, k_i) = r_i - d_e(X_i, l)$$

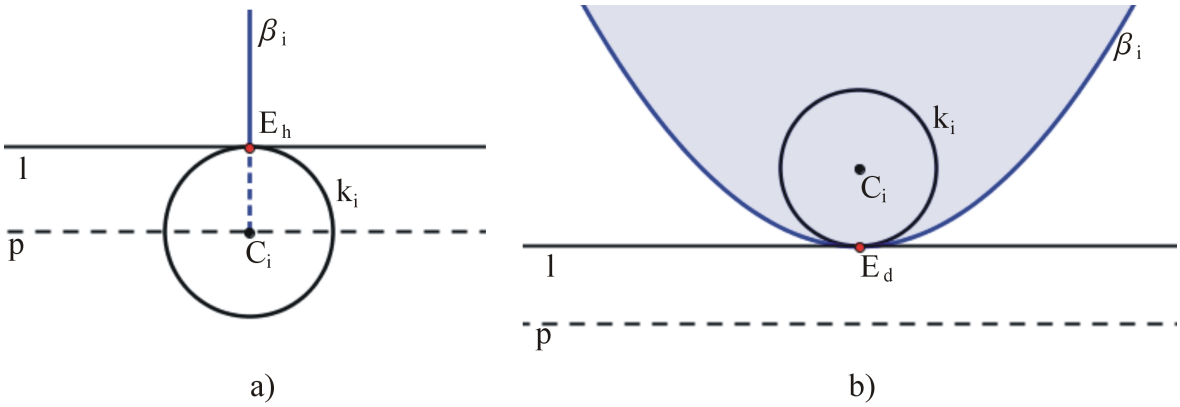
a tedy platí

$$d_e(X_i, k_i) = d_e(X_i, l). \tag{21}$$

Z rovnice (21) vidíme, že vzdálenost vnitřních bodů X_i od kružnice je stejná, jako vzdálenost vnitřních bodů X_i od přímky l. Tím jsme dokázali, že množina vnitřních bodů X_i, která má stejnou vzdálenost od kružnice k_i a od přímky l tvoří parabolu.

○

Poznámka 16 *Všimněme si, že pokud je zametací přímka l tečnou kružnice k_i v jejím horním extrému E_h, pak ohnisko C_i paraboly leží na své řídící přímce p a vzniklá parabola je degenerovaná (tzn. jedná se o polopřímku, která je kolmá na l má počáteční bod na přímce l a na kružnici k_i v jejich dotykovém bodě, tedy v horním extrému kružnice k_i), viz obr. 33a).*



Obrázek 33: Parabola pro zametací přímku l v: a) hodním extrému E_h; b) dolním extrému E_d

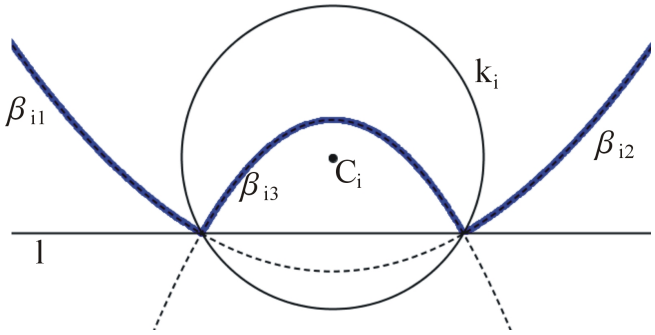
Zametací přímka l má vůči generující kružnici k tři různé polohy

1. zametací přímka l neprotíná kružnici k, tedy nemají žádný společný bod
2. zametací přímka l je tečnou kružnice k, tedy mají jeden společný bod a to horní nebo dolní extrém
3. zametací přímka l protíná kružnici k, tedy mají dva společné průsečíky.

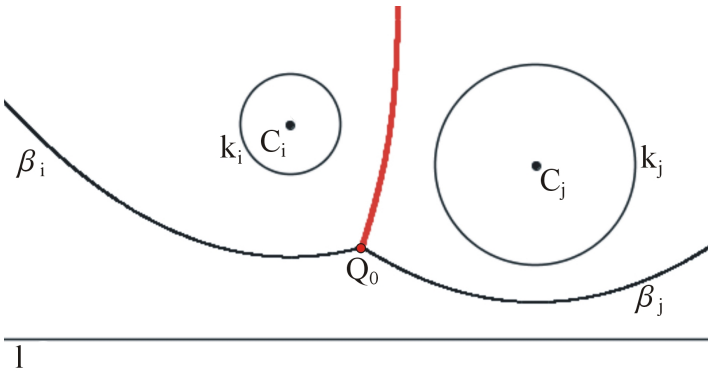
V prvním případě vykreslujeme parabolu, její část se stává součástí beach line. Druhý případ se dá rozdělit na dvě části, kdy zametací přímka má bod dotyku s generující kružnicí k v horním nebo dolním extrému, viz obr. 33a), 33b). Pokud má zametací přímka l bod dotyku s generující kružnicí k v horním extrému, pak vzniklá parabola je degenerovaná, viz obr. 33a. Pokud se zametací přímka l dotýká generující kružnice k v dolním extrému, pak vrchol vzniklé paraboly leží na zametací přímce a je jím právě dolní extrém generující kružnice k, viz obr. 33b). V případě třetím se vrchol paraboly a její část nachází pod zametací přímkou l, což je pro účel algoritmu nepodstatná část. Díky tomu, že se část paraboly nachází pod zametací přímkou l, je beach line b neúplná. Nicméně uvnitř generující kružnice k lze také najít množinu bodů, která má stejnou vzdálenost od generující kružnice k a od zametací přímky l, tedy parabolu. Parabola vzniklá uvnitř generující kružnice je konkávní a její část nacházející se nad zametací přímkou l je součástí beach line b, viz obr. 34.

Věta 4.2 *Průsečík Q_0 dvou sousedních parabolických oblouků β_i, β_j leží na Voroneho hraně,tj. Q_0 má stejnou vzdálenost od dvou nejbližších generujících kružnic k_i, k_j, viz obr. 35.*

•



Obrázek 34: Beach line b



Obrázek 35: Voroneho hrana pro Voroneho diagram kružnic

Důkaz Mějme beach line b pro dvě generující kružnice k_i, k_j a odpovídající zametací přímku l. Parabolické oblouky β_i, β_j náleží beach line a jsou příslušné generujícím kružnicím k_i, k_j. Průsečík dvou parabolických oblouků označíme Q_0.

Pro libovolný bod $X_i \in \beta_i$ platí

$$d_e(X_i, l) = d_e(X_i, k_i).$$

Obdobně, pro libovolný bod $X_j \in \beta_j$ platí

$$d_e(X_j, l) = d_e(X_j, k_j).$$

Protože bod Q_0 leží na parabolickém oblouku β_i platí pro něj

$$d_e(Q_0, l) = d_e(Q_0, k_i),$$

současně bod Q_0 leží na parabolickém oblouku β_j a tedy pro něj platí

$$d_e(Q_0, l) = d_e(Q_0, k_j).$$

Po úpravě zístáváme rovnost

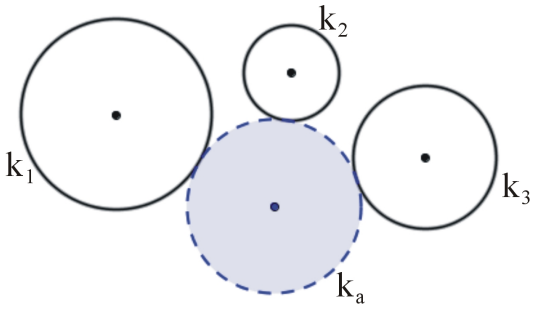
$$d_e(Q_0, k_i) = d_e(Q_0, k_j).$$

Z rovnosti je vidět, že průsečík Q_0, dvou beach line oblouků β_i, β_j má stejnou vzdálenost od dvou generujících kružnic k_i, k_j. To znamená, že bod Q_0 leží na Voroneho hraně mezi dvěmi Voroneho buňkami $\nu(k_i), \nu(k_j)$.

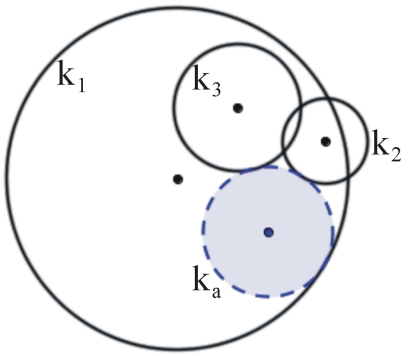
○

Jak jsme již zmínili dříve, algoritmus Plane sweep pro Voroneho diagram kružnic má tři typy operací. Ještě než popíšeme dané tři typy operací uvedeme si význam pojmu Apolloniova kružnice, jež budeme v následujícím textu potřebovat.

Apolloniovou úlohou rozumíme úlohu zabývající se sestrojením kružnice, která se dotýká tří zadaných kružnic, bodů nebo přímek. V této práci využijeme pouze úlohu o sestrojení kružnice, která se dotýká tří zadaných kružnic, jež jsou v tomto případě kružnice generující. Pak kružnici, která se dotýká tří zadaných generujícíck kružnic $k_i \in S$ nazveme Apolloniova kružnice.



Obrázek 36: Apolloniova kružnice vně všech tří generujících kružnic

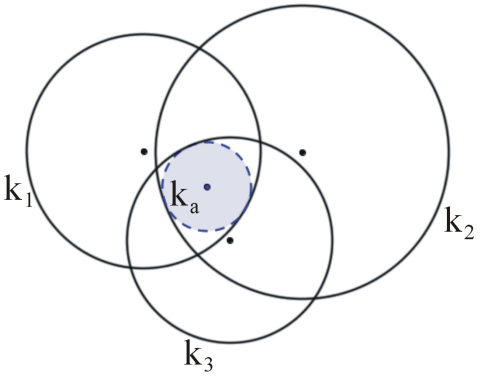


Obrázek 37: Apolloniova kružnice vně dvou generujících kružnic a uvnitř jedné

Apolloniova kružnice může být umístěna vně všech tří generujících kružnic, viz obr. 36, uvnitř všech tří generujících kružnic, viz obr. 38, a vně nebo uvnitř pouze některých, viz obr. 37. Pro naše účely, kdy se generující kružnice neprotínají, budeme používat Apolloniovu kružnici, která je vně všech tří generujících kružnic.
Bližší informace lze nalézt v [7], [6].

Nyní již můžeme zavést a popsat tři typy operací jež jsou charakteristické pro algoritmus Plane sweep

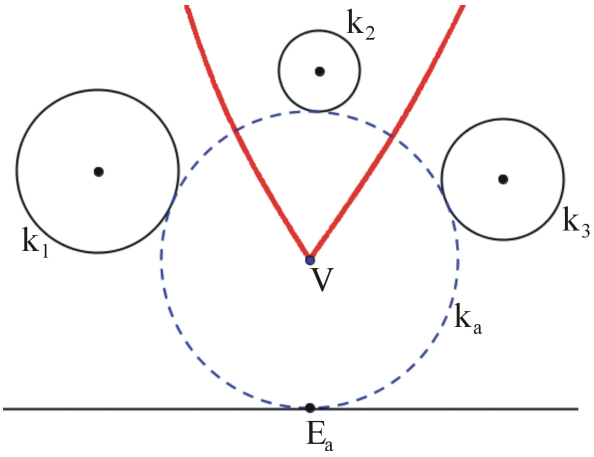
Site event tato operace nastane ve chvíli, kdy zametací přímka l poprvé protne generující kružnici, bod ve kterém se tak stane nazýváme horní extrém E_h



Obrázek 38: Apolloniova kružnice uvnitř všech tří generujících kružni

Merge event operace nastává ve chvíli, kdy zametací přímka l opouští generující kružnici, vzniká dotyk v dolním bodě generující kružnice, tento dolní bod generující kružnice je brán ve směru pohybu zametací přímky a nazýváme ho dolní extrém E_d

Circle event zametací přímka opouští Apolloniovu kružnici tří generujících kružnic, významným bodem je zde dolní bod E_a Apolloniovy kružnice, viz obr. 39.



Obrázek 39: Operace "circle event"

Poznámka 17 *Všimněme si, že zametací přímka l v horním (resp. dolním) extrému je v tomto bodě tečnou dané generující kružnice.*

•

Zatímco první dva typy operací jsou na výpočet poměrně jednoduché, u operace "circle event"to neplatí. Pro výpočet je nutné nejprve vypočíst Apolloniovu kružnici ke třem generujícím kružnicím z množiny S [1].

V našem případě máme tři generující kružnice, které se neprotínají v žádném bodě a současně žádná z kružnic neleží unvitř jiné generující kružnice. Počet Apolloniových kružnic sestrojených k takto zadaným generujícím kružnicím je právě osm. Nicméně není nutné pracovat se všemi osmi Apolloniovými kružnicemi pro tři generující kružnice, neboť orientace beach line oblouků nám poskytuje informace o požadované Apolloniově kružnici.

Lemma 4.4 *Pokud parabola $\beta_i \in b$ je konvexní, pak hledaná Apolloniova kružnice je vně všech odpovídajících generujících kružnic a zároveň žádná z generujících kružnic neleží uvnitř Apolloniovy kružnice. Pokud je parabola $\beta_i \in b$ konkávní, pak se Apolloniova kružnice nachází uvnitř generující kružnice k_i odpovídající β_i [1].*

•

V našem případě nás bude zajímat pouze Apolloniova kružnice, která se bude nacházet vně všech tří generujících kružnic. Druhý případ nastává pro generující kružnice, které se protínají. Bude řešeno později.

Při inicializaci bude nutno seřadit body podle určitého pravidla, toto pravidlo zavedeme následovně.

Definice 4.3 *Bod $P = [x_p, y_p]$ zařadíme před bod $Q = [x_q, y_q]$, pokud platí $y_p \leq y_q$ a $x_p < x_q$.*

•

Inicializace Nejprve musíme zjistit a zapsat do fronty F horní, resp. dolní extrémy a seřadit je podle pravidla uvedeného v definici 4.3. Nakonec vytvoříme zametací přímku l a beach line b.

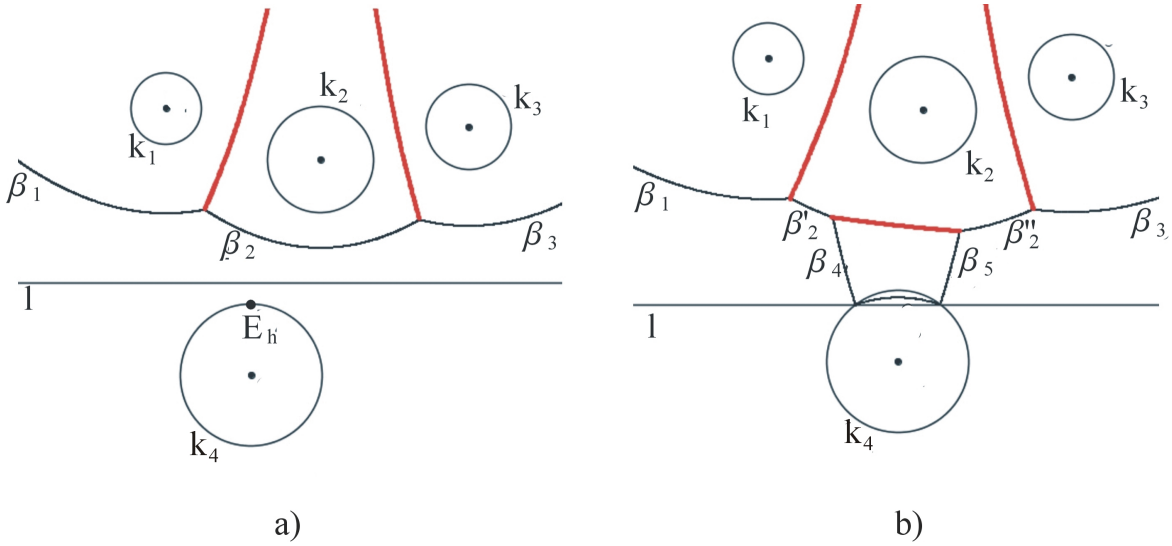
Zpracování jednotlivých operací Jakmile je fronta F vytvořena, algoritmus začne se zpracováváním jednotlivých operací postupně tak, jak jsou seřazeny ve frontě F. Každá operace je spojena se svou vlastní akcí, ta se skládá ze tří hlavních prvků

- modifikace beach line b, buď přidáním nebo odebráním některého parabolického oblouku β_i
- modifikace fronty F, odstraněním nežádoucí Apolloniovy kružnice a přidáním nové, vhodnější Apolloniovy kružnice
- modifikace Voroneho diagramu tím, že přidáme nebo sloučíme některé Voroneho hrany.

Vyjímku, pro námi zadanou množinu S, tvoří operace "merge event"u které se vyskytuje pouze první akce, tedy přidání nebo odebrání parabolického oblouku. Pro zadanou množinu S, kde budeme připouštět protínání kružnic atd., bude řešeno později, platí všechny tři typy akcí.

Site event Nastává ve chvíli, kdy zametací přímka l protne novou generující kružnici v horním etrému E_h, viz obr. 40. V tuto chvíli vznikne nová parabola, která je degenerovaná. Následně, při této operaci, vznikají tři parabolické oblouky $\beta_4, \beta_5, \beta_6$, odpovídající generující kružnici k_4, jež rozdělují beach line oblouk β_2 na dvě části β_2' a β_2'', viz obr. 40b). Oblouky β_4, β_6 jsou částmi, levé a pravé větve, téže paraboly, viz lemma 4.3.

To, který beach line oblouk bude rozdělen nově přidanými parabolickými oblouky, lze rozhodnout např. na základě neprázdného průniku beach line oblouku β_i s degenerovanou parabolou.



Obrázek 40: Operace "site event"pro Voroneho diagram kružnic: a) před operací; b) po operaci

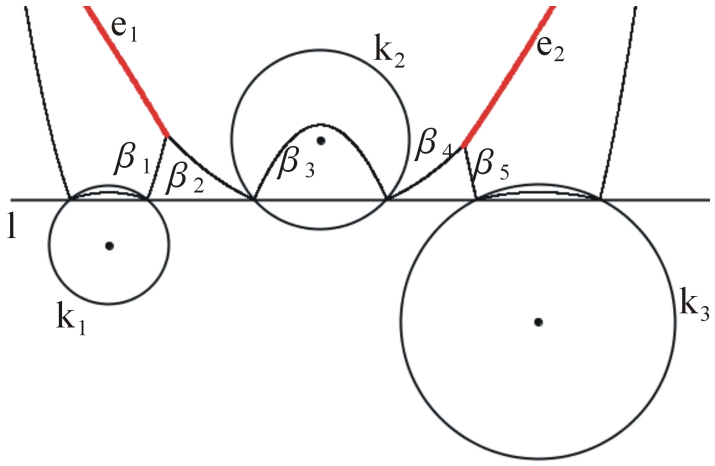
Sousedními oblouky beach line oblouku β_2 jsou oblouky β_1 a β_3. Beach line oblouky $\beta_1, \beta_2, \beta_3$ odpovídají po řadě generujícím kružnicím k_1, k_2, k_3. Sousedních oblouků oblouku β_2 využijeme pro vytvoření nových Apolloniových kružnic. Dvě nové Apolloniovy kružnice vytvoříme pro generátory k_1, k_2, k_4 a k_3, k_2, k_4 odpovídající po řadě beach line obloukům $\beta_1, \beta_2', \beta_4$ a $\beta_3, \beta_2'', \beta_6$, čehož se v dalších krocích využívá v operaci "circle event". Pro první trojici oblouků označíme významný bod Apolloniovy kružnice E_{h1}, pro druhou trojici významný bod Apolloniovy kružnice označíme E_{h2}. Oba tyto body zařadíme do fronty F.

Frontu F modifikujeme pokaždé, kdy je definována nová Apolloniova kružnice, popřípadě pokud zaniká Apolloniova kružnice nebo kdykoli je změněna beach line odstraněním nebo vložením nových oblouků.

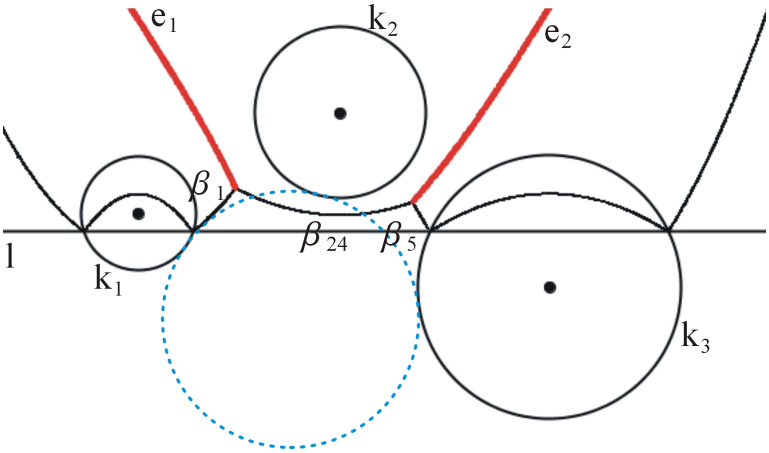
Operace "site event"odstraňuje jednu Apolloniovu kružnici z fronty F a vytváří až dvě nové Apolloniovy kružnice. Množství přidávaných Apolloniových kružnic závisí na tom, v jaké fázy algoritmu se nacházíme a také na tom, kolik máme objevených

generujících kružnic [1].

Merge event Pro operaci "merge event", znázorněné na obrázcích 41, 42, je podstatný dolní extrém E_d, generující kružnice k_2. Operaci dělíme na dva případy, kdy uvnitř generující kružnice leží pouze jeden beach line oblouk nebo více oblouků beach line. Druhý případ nastává ve chvíli, kdy se dvě a více generujících kružnic protíná, bude řešeno později.



Obrázek 41: Operace "merge event"pro Voroneho diagram kružnic, před zpracováním operace



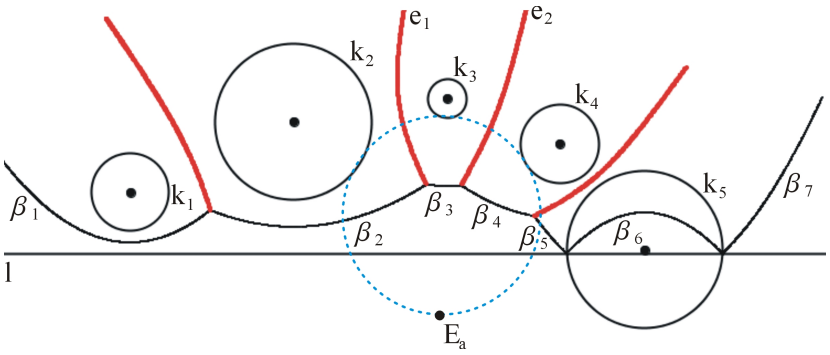
Obrázek 42: Operace "merge event"pro Voroneho diagram kružnic, po zpracováním operace

Při operaci "merge event"dochází k zániku beach line oblouku β_3. Po zániku beach line oblouku β_3, který se nacházel uvnitř kružnice k_2 dojde ke spojení sousedních beach

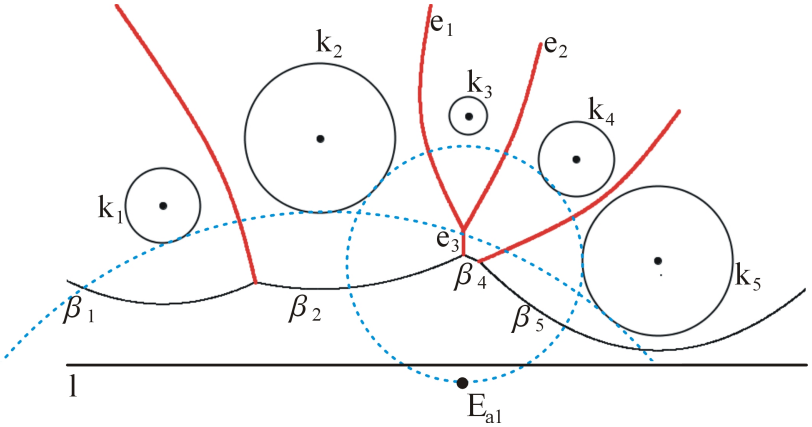
line oblouků β_2 a β_4, které leží vně generující kružnice k_2. Tím je vytvořen nový beach line oblouk β_{24}.

V tomto kroku může dojít k zániku některých Apolloniových kružnic, a to právě těch, které souvisely s odstraněným beach line obloukem β_3, proto musejí být odstraněny z fronty F. To nastává pouze v případě, kdy se generující kružnice protínají, bude řešeno později. Nicméně ve frontě F musíme nahradit beach line oblouky β_2, β_4 sloučeným beach line obloukem β_{24} [1].

Circle event Nechť k_a je Apolloniova kružnice s odpovídajícím významným bodem E_a.



Obrázek 43: Operace "circle event"pro Voroneho diagram kružnic před operací



Obrázek 44: Operace "circle event"pro Voroneho diagram kružnic po operaci

Jak je vidět na obrázcích 43, 44, Apolloniova kružnice k_a se dotýká tří generujích kružnic k_2, k_3, k_4 a odpovídá tedy beach line obloukům $\beta_2, \beta_3, \beta_4$. Při této operaci zaniká beach line oblouk β_3 a ve středu Apolloniovy kružnice k_a vznikne nový Voroneho vrchol

V. Ten je ukončením stávajících hran e_1, e_2 a současně je počátkem nové Voroneho hrany e_3.

Apolloniova kružnice k_a zaniká, je tedy vyjmuta z fronty F a současně jsou definovány nejvýše dvě nové Apolloniovy kružnice. Počet nových Apolloniových kružnic závisí na tom kolik generujících kružnic je již prohledaných. Pokud například v počátku algoritmu pracujeme pouze se třemi generujícími kružnicemi nevzniká mi žádná nová Apolloniova kružnice.

Nově vzniklé Apolloniovy kružnice jsou definovány pomocí beach line oblouků, které zbyly po zániku oblouku β_3. Nalevo od oblouku β_3 to jsou oblouky β_1, β_2 a napravo jsou to oblouky β_4, β_5. První z Apolloniových kružnic k_{a1} je tvořena pro $\beta_1, \beta_2, \beta_5$, s významným bodem E_{a_1} a druhá k_{a2} je tvořena pro $\beta_2, \beta_4, \beta_5$, s významným bodem E_{a_2}. Oba tyto body přidáme do fronty F.

Ukončení algoritmu Poslední operací algoritmu je vždy "merge event"nebo "circle event". Po zpracování poslední události ve frontě F by měl být Voroneho diagram hotov. Nicméně stále máme zametací přímku l a beach line b, která i nadále tvoří Voroneho hrany, tedy prostor pod beach line je stále nedokončen.

Voroneho buňka odpovídající generující kružnici, jež je součástí hranice konvexního obalu celé množiny generátorů S, je neomezená. Z toho důvodu bude algoritmus dokončen odstraněním beach line b a zametací přímky l.

Algoritmus ("Plane sweep"pro Voroneho diagram kružnic 1)
Vstup: Množina generujících kružnic S = $\{\mathrm{k}_1, k_2, \ldots, k_n\}$.
Výstup: Voroneho diagram množiny S.

1. **Inicializace**
   (a) vytvoření zametací přímky l a beach line b, která je v počátku nevlastní
   (b) vložení horních E_h a dolních E_d extrémů, pro operace site event a merge event, do fronty F
2. **Zpracování událostí**
   (a) výběr události E z fronty F
   (b) Switch(E)
   - CASE: Site event
     i. najít odpovídající beach line oblouk β
     ii. rozdělit β do dvou beach line oblouků β_l, β_r a vložit mezi ně nový beach line oblouk
     iii. vytvořit Voroneho hranu e
     iv. spojit Voroneho hranu e s beach line b

v. vytvořit nové Apolloniovy kružnice, jejich množství závisí na počtu již projetých generujících kružnic,
a vložit je do fronty F

- CASE: Merge event
  i. najít odpovídající bech line oblouk β_c a dva oblouky β_l, β_r spojených s β_c
  ii. β_l, β_r jsou vně odpovídající kružnice
    A. vyjmout β_c
    B. sloučit oblouky β_l, β_r a vytvořit jediný oblouk β_{lr}
  iii. vytvoření jedné Apolloniovy kružnice, pro circle event, definované trojicí beach line oblouků $(left(\beta_{lr}), \beta_{lr}, right(\beta_{lr}))$ [1] a jejich vložení do fronty F
- CASE: Circle event
  i. načtení odpovídající beach line trojice $(\beta_l, \beta, \beta_r)$
  ii. ověření, zda aktuální beach line trojice je platná [2]
    A. if YES
      – načtení dvou Voroneho hran $e_i, e_j, i \neq j$ spojených s β
      – vytvoření Voroneho vrcholu v
      – spojení hran e_i, e_j ve vrcholu v
      – vytvoření nové Voroneho hrany z Voroneho vrcholu v, kde v je koncovým vrcholem nově vzniklé Voroneho hrany
      – připojení dvou beach line oblouků β_l, β_r k sousednímu oblouku β
      – vytvoření dvou Apolloniových kružnic definovaných trojicemi $(left(\beta_l), \beta_l, \beta_r)$ a $(\beta_l, \beta_r, right(\beta_r))$ a jejich vložení do fronty F
      – vyjmutí beach line oblouku β z fronty F
    B. if NO
      – pokračování v algoritmu

3. **Ukončení algoritmu**

(a) Voroneho vrcholy spojeny s beach line nechť odpovídají nekonečnu

(b) vymazání beach line b a zametací přímky l

[1] $left(\beta_{lr}), rigrt(\beta_{lr})$ označují levý a pravý sousední oblouk β_{lr}.
[2] tzn. zda Apolloniova kružnice odpovídá dané trojici.

4.1.3 Algoritmus "Plane sweep"pro Voroneho diagram kružnic se změnou základních vlastností množiny S

V předchozí kapitole 4.1.2 jsme popsali algoritmus Plane sweep pro množinu $S = \{k_1, k_2, \ldots, k_n\}$ generujících kružnic, které měly tu vlastnost, že se neprotínaly, jejich poloměry byly nenulové a žádná z generujících kružnic neobsahovala jinou generující kružnici.

Nyní si uvedeme změny, které v algoritmu nastanou, pokud předchozí vlastnosti množiny S generujících kružnic změníme. Tedy pro generující kružnice $k_i \in S$ bude nadále platit, že

- mohou mít neprázdný průnik (*tzn. dva body, kružnice se protínají; jeden bod, kružnice se dotýkají*), pouze za předpokladu, že žádné tři a více generujících kružnic nemají společný bod
- poloměry generujících kružnic mohou být nulové, tedy $r_i = 0$ (*tzn. množina S může obsahovat body*)
- generující kružnice k_i může ležet uvnitř jiné generující kružnice k_j, kde $i \neq j$
- žádné čtyři generující kružnice nemají společnou Apolloniovu kružnici.

Mějme množinu $S = \{k_1, k_2, \ldots, k_n\}$ generujících kružnic $k_i = (C_i, r_i)$, kde $C_i = (x_i, y_i)$ je střed kružnice a r_i je její poloměr. Z celkového počtu n generujících kružnic k_i množiny S existuje g kružnic, které nemají žádný společný bod s jinou generující kružnicí k_j z množiny S. Počet generujících kružnic, které mají společný bod s jinou generující kružnicí z množiny S je $l = n - g$ a současně m je počet společných bodů těchto kružnic. Každé dvě kružnice s neprázdným průnikem mají dva (*resp. jeden*) společný bod. Tyto body rozdělují každou kružnici na dva (*resp. jeden*) oblouk.

Věta 4.3 *Mějme množinu $S = \{k_1, k_2, \ldots, k_n\}$ generujících kružnic, kde n je počet těchto kružnic. Nechť g je počet generujících kružnic, které nemají žádný společný bod s jinou generující kružnicí z množiny S. Počet kružnic s neprázdným průnikem je $l = n - g$ a m je počet společných bodů všech těchto kružnic. Pak celkový počet oblouků pro $l = n - g$ kružnic je $2m$ [1].*

•

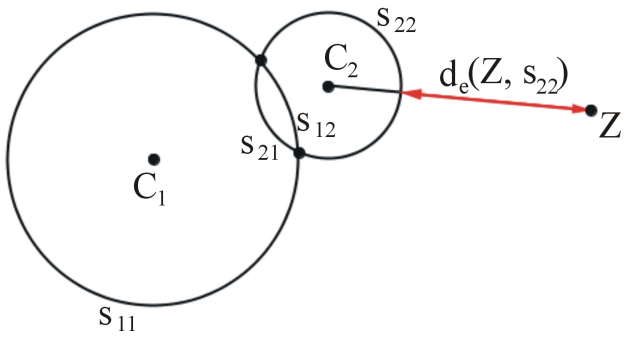
Důkaz Každé dvě kružnice s neprázdným průnikem mají dva (*resp. jeden*) společný bod, tyto body rozdělují každou kružnici na dva (*resp. jeden*) oblouk. Proto pro m průsečíků získáváme $2m$ oblouků. Toto platí pouze v případě, že žádné tři nebo více generujících kružnic se neprotínají v jednom bodě, což je předpoklad množiny S.

○

V následujícím textu budeme používat označení generující místo. Tímto pojmem se rozumí buď kružnice, která nemá žádný spoječný bod s jinou generující kružnicí nebo oblouk na generující kružnici rozdělené průsečíkem, popř. dotykovým bodem.

Množina $S' = \{s_1, s_2, \ldots, s_{g+2m}\}$ je množina všech generujících míst definovaných v množině S.

Nejkratší Eukleidovskou vzdálenost mezi libovolným bodem Z roviny a generujícím místem $s_i \in S'$ označme $d_e(Z, s_i)$, viz obr. 45.



Obrázek 45: Vzdálenost bodu Z od generujícího místa s_{22}

Věta 4.4 *Bod Z bude náležet Voroneho buňce $\nu(s_i)$, pro generující místo s_i, pouze tehdy, bude-li platit, že $d_e(Z, s_i) < d_e(Z, s_j)$, kde $i \neq j$. V případě rovnosti bude bod Z ležet na Voroneho hraně.*

•

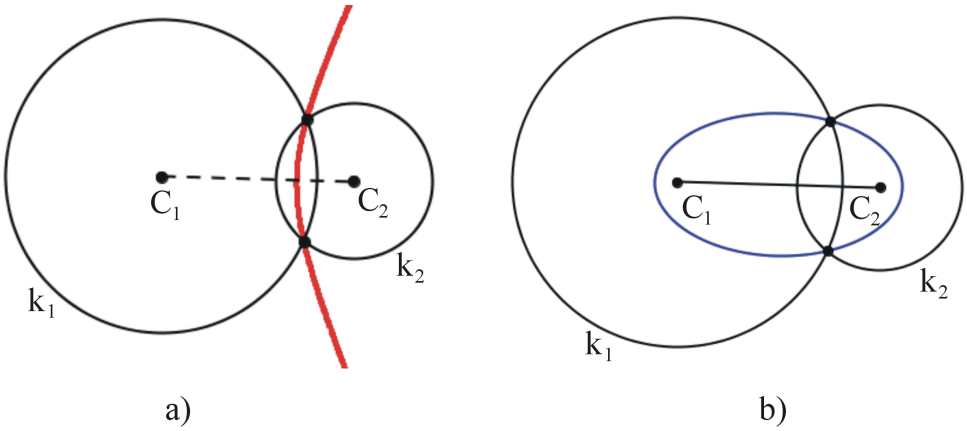
Zatímco hrany Obecného Voroneho diagramu jsou částmi přímek, hrany Voroneho diagramu kružnic jsou částmi kuželoseček. V úvodu kapitoly 4 jsme uvedli lemma 4.1, to pojednávalo o tom, že množina bodů, která má stejnou vzdálenost od dvou neprotínajících se kružnic s různými poloměry je hyperbola. V případě, kdy hledáme množinu bodů, která má stejnou vzdálenost od dvou protínajících se kružnic, tato skutečnost zcela neplatí. Pro dvě protínající se kružnice nastanou současně dvě možnosti, viz následující lemma 4.5.

Lemma 4.5 *Množiny bodů které májí stejnou vzdálenost od dvou protínajících se kružnic s různými poloměry jsou*

hyperbola *v případě, že se hledaná množina nachází uvnitř nebo vně obou kružnic, viz obr. 46a)*

elipsa *v případě, že se hledaná množina nachází uvnitř jedné kružnice a vně druhé nebo naopak, viz obr. 46b).*

•



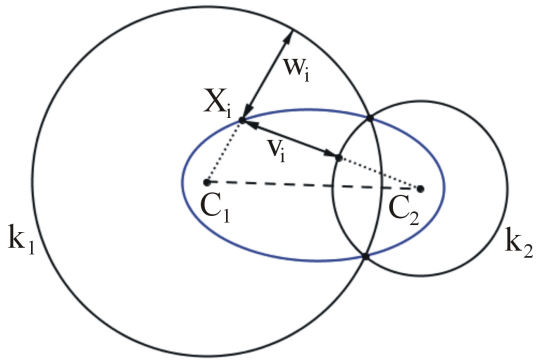
Obrázek 46: Množina bodů se stejnou vzdáleností od dvou protínajících se kružic: a) hyperbola; b) elypsa

Důkaz Mějme dvě protínající se kružnice k_1, k_2 se středy v bodech C_1, C_2 a poloměry r_1, r_2, kde $r_1 \neq r_2$. Hledáme množiny bodů X_i, které mají stejnou vzdálenost od kružnic k_1, k_2.

První případ, kdy je hledaná množina hyperbolou, je dokázán v lemmatu 4.1. Druhý případ, kdy je hledaná množina elipsou, dokážeme následujícím způsobem.

Z definice elipsy, bod X_i je bodem elipsy, kde C_1, C_2 jsou ohniska elipsy, pokud platí

$$d_e(X_i, C_1) + d_e(X_i, C_2) = \text{konst.} \tag{22}$$



Obrázek 47: Vzdálenost bodu X_i od středů C_1, C_2 generujících ružnic k_1, k_2

Pokud budeme předpokládat, že bod X_i se nachází vně kružnice k_1 a současně leží uvnitř kružnice k_2, viz obr. 47, pak platí, že $d_e(X_i, C_1)$ je součet poloměru r_1 kružnice k_1 se vzdáleností v_i bodu X_i od kružnice k_1, tedy $v_i = d_e(X_i, k_1)$. Ovšem, pokud bod X_i se nachází uvnitř kružnice k_2, pak $d_e(X_i, C_2)$ je rozdíl poloměru r_2 kružnice k_2 a vzdáleností w_i bodu X_i od kružnice k_2, tedy $w_i = d_e(X_i, k_2)$.

Z předchozích informací můžeme psát, že

$$d_e(X_i, C_1) = d_e(X_i, k_1) + r_1 \quad (= v_i + r_1)$$

a zároveň

$$d_e(X_i, C_2) = r_2 - d_e(X_i, k_2) \quad (= r_2 - w_i).$$

Po úpravě získáváme

$$v_i = d_e(X_i, C_1) - r_1$$

a zároveň

$$w_i = r_2 - d_e(X_i, C_2).$$

Na základě toho, že hledáme množinu bodů, která má stejou vzdálenost od dvou kružnic k_1, k_2, platí

$$w_i = v_i.$$

Po dosazení získáváme rovnost

$$d_e(X_i, C_1) - r_1 = r_2 - d_e(X_i, C_2)$$

a po úpravě dostaneme rovnost

$$d_e(X_i, C_1) + d_e(X_i, C_2) = r_2 + r_1.$$

Tato rovnost odpovídá rovnosti (22), kde konstanta je $r_2 + r_1$.

Postup důkazu platí i pro případ, kdy se bod X_i nachází uvnitř kružnice k_1 a současně leží vně kružnice k_2. Rozdíl pak bude v tom, že $d_e(X_i, C_1)$ je rozdíl poloměru r_1 kružnice k_1 a vzdálenosti w_i bodu X_i od kružnice k_1 a $d_e(X_i, C_2)$ je součet poloměru r_2 kružnice k_2 a vzdálenosti v_i bodu X_i od kružnice k_2. Po vyjádření a úpravách opět dostaneme rovnost (22).

Tím získáváme konstatní součet vzdáleností pro každý bod X_i hledané množiny, výsledná množina je tedy elipsou.

○

Následující věta nám ukazuje vlastnosti bodů ležících na Voroneho hraně dvou protínajících se kružnic, vzhledem k tomu na jaké části hrany leží.

Algoritmus Plane sweep pro Voroneho diagram kružnic, obdobně jako v předchozích případech, využívá zametací přímky l, pomocí níž vytváří beach line b, jejíž jednotlivé parabolické oblouky označujeme β_i.

Algoritmus Plane sweep pro Obecný Voroneho diagram má pouze dva typy operací a to "site event"a "circle event". Algoritmus Plane sweep pro Voroneho diagram kružnic s omezenými vlastnostmi množiny S generujících kružic, jak byl definován v kapitole 4.1.2, má operace tři, a to "site event", "circle event"a "merge event". Algoritmus Plane sweep pro Voroneho diagram kružnic s vlastnostmi, jež byly definovány v této kapitole má operace čtyři, má navíc operaci "cross event".

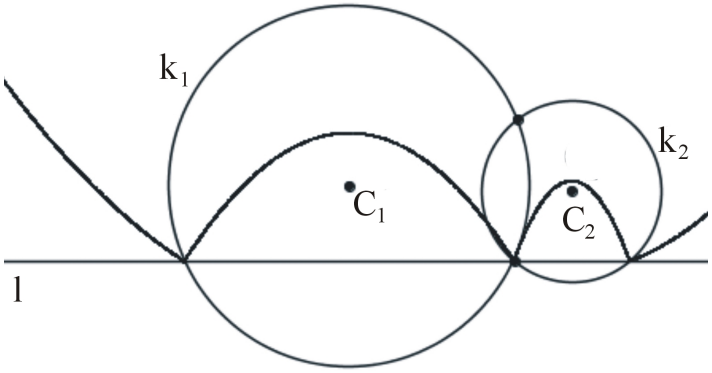
Jak jsme již zmínili, algoritmus využívá čtyři typy operací

Cross event operace nastane ve chvíli, kdy se zametací přímka l bude nacházet v průsečíku dvou generujících kružnic, viz obr. 48

Site event tato operace nastane ve chvíli, kdy zametací přímka l poprvé protne generující kružnici, bod ve kterém se tak stane nazýváme horní extrém E_h

Merge event operace nastává ve chvíli, kdy zametací přímka l opouští generující kružnici, vzniká dotyk v dolním bodě generující kružnice, tento dolní bod generující kružnice je brán ve směru pohybu zametací přímky a nazýváme ho dolní extrém E_d

Circle event zametací přímka opouští Apolloniovu kružnici, významným bodem je zde dolní bod Apolloniovy kružnice E_a.



Obrázek 48: Operace "cross event"

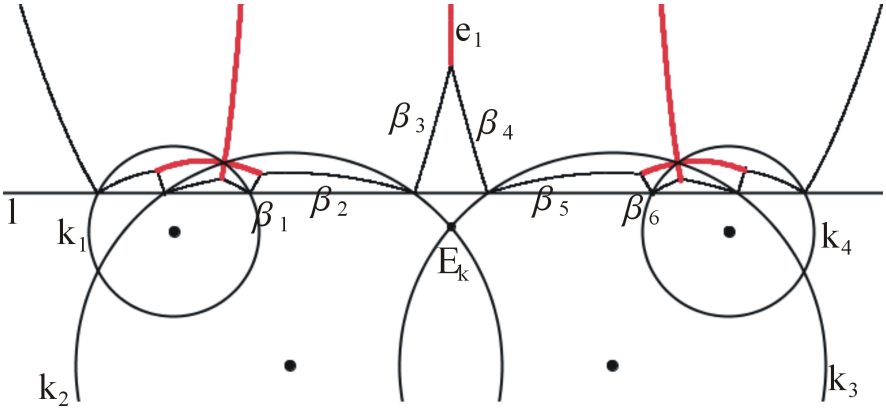
Inicializace Nejprve musíme zjistit a zapsat do fronty F horní a dolní extrémní body, dále všechny průsečíky a vše seřadit sestupně, viz definice 4.3. Nakonec vytvoříme zametací přímku l a beach line b.

Zpracování jednotlivých operací Jakmile je fronta F hotova, algoritmus začne se zpracováváním jednotlivých operací postupně tak jak jsou seřazeny ve frontě F. Každá operace je spojena se svou vlastní akcí, ty se skládají ze tří hlavních částí

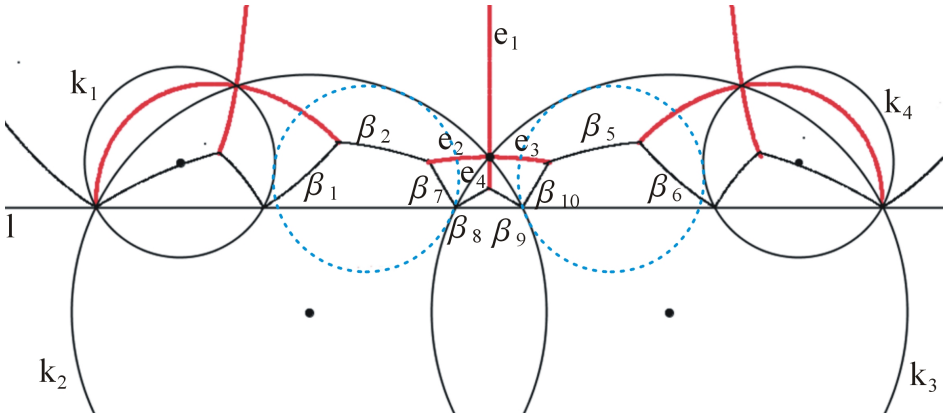
- modifikace beach line b, buď přidáním nebo odebráním některého oblouku β_i
- modifikace fronty F, odstraněním nežádoucí Apolloniovy kružnice a přidáním nové, vhodnější Apolloniovy kružnice
- modifikace Voroneho diagramu přidáním nebo sloučením některé Voroneho hrany [1].

Operace "site event", "merge event"a "circle event"jsou diskutovány v předchozí podkapitole 4.1.2 a zde je tedy nebudeme uvádět.

Cross event Na obrázku 49 je operace "cross event"definována dvěmi generujícími kružnicemi $k_2 = (C_2, r_2)$ a $k_3 = (C_3, r_3)$. Těsně před tím než nastane operace "cross event"se k bodu E_k,tj. průsečík dvou generujících kružnic, přibližují dva průsečíky beach line oblouků β_2, β_3 a β_4, β_5. Tyto průsečíky současně leží na generujících kružnicích k_2, k_3. Jakmile se zametací přímka l, současně s průsečíky beach line oblouků β_2, β_3 a β_4, β_5, dostane do bodu E_k, parabolické oblouky β_3, β_4 zaniknou a vytvoří se nové parabolické oblouky $\beta_7, \beta_8, \beta_9, \beta_{10}$, průsečík E_k se stane novým Voroneho vrcholem.



Obrázek 49: Operace "cross event"pro Voroneho diagram, před zpracováním operace



Obrázek 50: Operace "cross event"pro Voroneho diagram, po zpracování operace

Při operaci "cross event"nezanikají žádné Apolloniovy kružnice současně po přidání čtyř nových parabolických oblouků do beach line, mohou vzniknout nové Apolloniovy kružnice. Na obrázku 50 jsou nové Apolloniovy kružnice definovány pomocí beach line oblouků $\beta_1, \beta_2, \beta_7$ a $\beta_{10}, \beta_5, \beta_6$.

Průsečík dvou kružnic E_k je novým Voroneho vrcholem a je ukončením stávající hrany e_1. Z nového Voroneho vrcholu E_k povedou tři nové Voroneho hrany e_2, e_3 a e_4, kde hrany e_2, e_3 jsou částmi téže elipsy mezi generujcími kružnicemi k_2, k_3 a hrana e_4 je pokračování hyperboly mezi generujcími kružnicemi k_2, k_3. Koncový Voroneho vrchol nových Voroneho hran zatím není znám. Stupeň Voroneho vrcholu E_k je čtyři [1].

Ukončení algoritmu Poslední operací algoritmu je vždy "merge event"nebo "circle event". Po zpracování poslední události ve frontě F by měl být Voroneho diagram hotov. Nicméně stále máme zametací přímku l a beach line b, která i nadále tvoří Voroneho hrany, tedy prostor pod beach line je stále nedokončen.

Voroneho buňka odpovídající generující kružnici, jež je součástí hranice konvexního obalu celé množiny generátorů S, je neomezená. Z toho důvodu bude algoritmus dokončen odstraněním beach line b a zametací přímky l.

V následující části uvedeme popis algoritmu Plane sweep rozšířený o operaci "cross event". Postup algoritmu z kapitoly 4.1.2 je nezměněn až na vložení operace "cross event"mezi operace "merge event"a "circle event". Nebudeme tu tedy algoritmus uvádět celý, nýbrž pouze jeho část zabývající se operací "cross event".

Algoritmus ("Plane sweep"pro Voroneho diagram kružnic 2)
Vstup: Množina generujících kružnic S = $\{k_1, k_2, \ldots, k_n\}$.
Výstup: Voroneho diagram množiny S.

1. **Inicializace**

⋮

2. **Zpracování operací**

   (a) výběr události E z fronty F

   (b) Switch(E)

⋮

- CASE: Cross event
  i. nalezení dvou odpovídajících beach line oblouků β_l, β_r
  ii. vytvoření čtyř beach line oblouků a jejich vhodného napojení
  iii. vytvoření tří Voroneho hran a jejich správné napojení
  iv. vytvoření dvou Apolloniových kružnic a jejich vložení do fronty F
  v. odebrání dvou beach line oblouků β_l, β_r z fronty F

⋮

4.1.4 Metoda rostoucích regionů pro Voroneho diagram kružnic

Metoda rostoucích regionů pro Voroneho diagram kružnic je obdobná, jako metoda pro Obecný Voroneho diagram. Vytvoříme kružnice s nulovými poloměry a se středy v generujících bodech, poloměr kružnic postupně zvětšujeme. Průsečíky kružnic (*popř. oblouků*), vykreslují Voroneho hrany, dokud nevznikne nový Voroneho vrchol. Ten vzniká ve společném průniku tří kružnic (*popř. oblouků*).

Množina S pro Voroneho diagram kružnic obsahuje, na rozdíl od Obecného Voroneho diagramu, generující kružnice. Kružnice l_i, které vytváříme musíme tedy vytvořit pro generující kružnice k_i. Mějme generující kružnici $k_i = (C_i, r_i) \in S$, kde $C_i = (x_i, y_i)$ je střed generující kružnice a r_i je její poloměr, $i = 1, 2, \ldots, n$. Pak kružnice l_i má střed v bodě C_i a počáteční poloměr $r_{l,i}$ kružnice l_i není roven nule, ale rovná se poloměru kružnice k_i, platí tedy $r_{l,i} = r_i$.

Poloměry vytvořených kružnic l_i se zvětšují stejnou rychlostí, ale mají rozdílnou hodnotu v počátku algoritmu. Pro poloměry $r_{l,i}, r_{l,j}$ dvou vytvořených kružnic l_i, l_j v generujících kružnicích k_i, k_j s poloměry r_i, r_j platí

$$r_{l,i} - r_i = r_{l,j} - r_j.$$

Pro množinu S generujících kružnic k_i zavedeme několik omezení

- průnik dvou libovolných kružnic $k_i, k_j \in S$ je prázdná množina
- žádná generující kružnice neleží uvnitř jiné.

Součástí této práce je znázornění metody rostoucích regionů, pro Voroneho diagram kružnic, v programu Geogebra. V následující části uvedeme stručný popis metody rostoucích regionů pro Voroneho diagram kružnic v programu Geogebra.

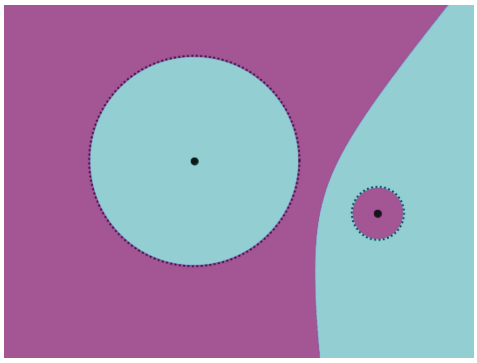
Popis metody rostoucích regionů pro program Geogebra Jak je uvedeno v kapitole 3.1.1, hlavní myšlenka metody rostoucích regionů, při konstukci Voroneho diagramu v programu Geogebra, je zachována. Rozdíl je v tom, že v počátku algoritmu, není poloměr kružnic l_i roven nule, ale poloměr kružnice l_i bude roven konstantě a. Poloměr kružnic následně zmenšujeme. Konstanta, které se poloměr v počátku algoritmu rovná, je v tomto případě volena $a = 20$ a zmenšujeme ji do $a = 0$. Kružnice l_i při změnšování poloměru, vykreslují svoji stopu, tímto způsobem je pro každou generující kružnici vykreslena Voroneho buňka.

Poznámka 18 *Uvědomme si, že poloměr $r_{l,i}$ vytvořené kružnice l_i pro generující kružnici k_i je $r_{l,i} = r_i + a$.*

•

Pokud by se změnšování poloměru zastavilo na hodnotě $a = 0$, pak vytvořené kružnice l_i by nevykreslily Voroneho buňku uvnitř generující kružnice k_i, viz obr. 51.

Tento problém můžeme vyřešit několika způsoby.



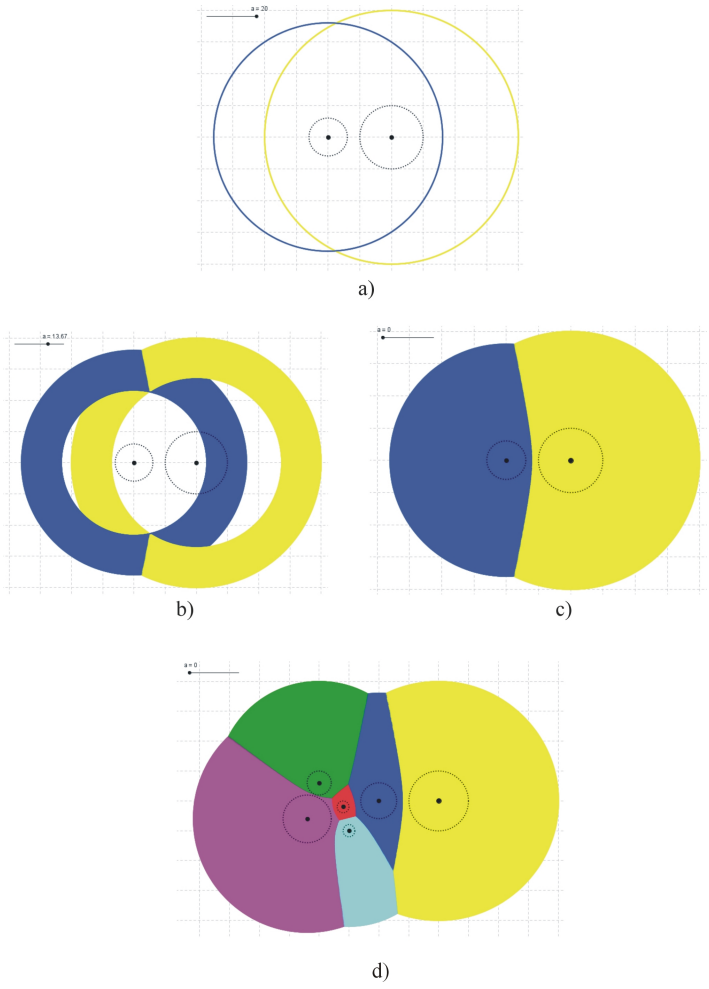
Obrázek 51: Voroneho diagram pro dvě generující kružnice s nevykresleným vnitřkem generujících kružnic

Prvním z možných řešení je zmenšit dolní hranici a a to minimálně o poloměr největší generující kružnice. Pokud by byl poloměr r_i největší generující kružnice k_i roven např. pěti, pak bychom a snižovali až do hodnoty $a = -5$.

Dalším možným řešením je zvolit si konstantu v, pro níž platí $v \geq r_i, i = 1, 2, \ldots, n$. Potom pro poloměr $r_{l,i}$ vytvořené kružnice l_i platí

$$r_{l,i} = a - (v - r_i).$$

Při znázornění metody rostoucích regionů je použito druhé řešení. Na obrázku 52a)-c) je ukázán postup vykreslování Voroneho diagramu kružnic pro dvě generující kružnice. Na obrázku 52d) je příklad Voroneho diagramu pro šest generujících kružnic.



Obrázek 52: Voroneho diagram pro dvě generující kružnice kde: a) a=20; b) a=13,67; c) a=0; d) Voroneho diagram pro šest generujících kružnnic

5 Laguerre Voroneho diagram

Laguerre Voroneho diagram můžeme najít také pod názvem Silový (*"Power"*) Voroneho diagram.

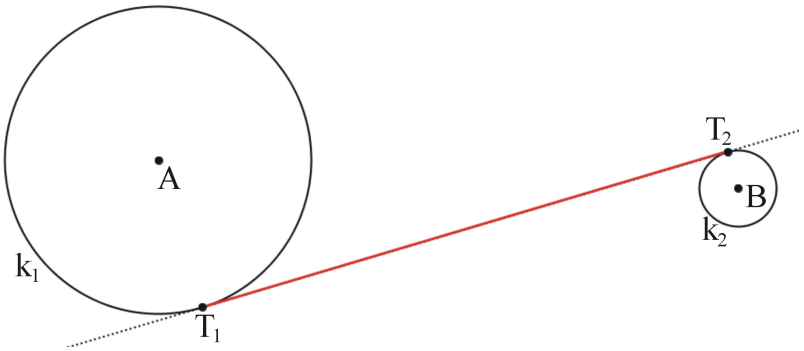
Mějme $S = \{P_1, P_2, \ldots, P_n\}$ množinu generujících bodů $P_i = (x_i, y_i, \sqrt{w_i})$, kde (x_i, y_i) jsou souřadnice generujícího bodu P_i a w_i je váha přiřazená danému generujícímu bodu. Váha w_i udává schopnost bodu P_i ovlivňovat své okolí [5], [9], [10].

Před tím, než se začneme plně věnovat Laguerre Voroneho diagramu uvedeme něco málo k Laguerre geometrii a ukážeme si možný geometrický přístup k množině S.

Laguerre geometrie Mějme tří-dimenzionální vektorový prostor nad reálným tělesem, kde vzdálenost $d(P, Q)$ dvou bodů $P = (x_p, y_p, w_p), Q = (x_q, y_q, w_q)$ je definována $d(P, Q)^2 = (x_p - x_q)^2 + (y_p - y_q)^2 - (w_p - w_q)^2$ [10].

Bod $A = (x, y, w)$, v Laguerre geometrii, koresponduje s kružnicí k v Eukleidovské geometrii, kde střed této kružnice má souřadnice (x, y) a její poloměr je $|w|$.

Vzdálenost dvou bodů $A = (x_A, y_A, w_A), B = (x_B, y_B, w_B)$ v Laguerre geometri, koresponduje se vzdáleností dotykových bodů T_1, T_2 společné tečny odpovídajících dvou kružnic $k_1 = (x_A, y_A), k_2 = (x_B, y_B)$ s poloměry $|w_A|, |w_B|$, viz obr. 53. Obdobně lze definovat vzdálenost bodu $C = (x_C, y_C, w_C)$ a bodu $Q = (x_Q, y_Q)$, jako vzdálenost dotykového bodu T tečny ke kružnici $k = (x_C, y_C)$ s poloměrem $|w_C|$ z bodu Q. Tato vzdálenost se nazývá Laguerre vzdálenost, podrobně ji zavedeme později. Bližší informace lze najít v [5], [9],[10],[13].



Obrázek 53: Laguerre vzdálenost dvou kružnic

Geometrická interpretace Na základě informací uvedených u Laguerre geometrie je možné si množinu S generujících bodů $P_i = (x_i, y_i, \sqrt{w_i})$ představit jako množinu kružnic p_i, kde $K_i = (x_i, y_i)$ jsou souřadnice středu a $\sqrt{w_i}$ jsou poloměry kružnic p_i.

Pro lepší pochopení a práci s Laguerre Voroneho diagramem modifikujeme množinu generujících bodů S. Nadále o množině generujících bodů S budeme mluvit jako

o množině generujících kružnic $S' = \{p_1, p_2, \ldots, p_n\}$, kde $p_i = (K_i, \sqrt{w_i})$ je generující kružnice se středem v bodě $K_i = (x_i, y_i)$ a poloměrem $\sqrt{w_i}$.

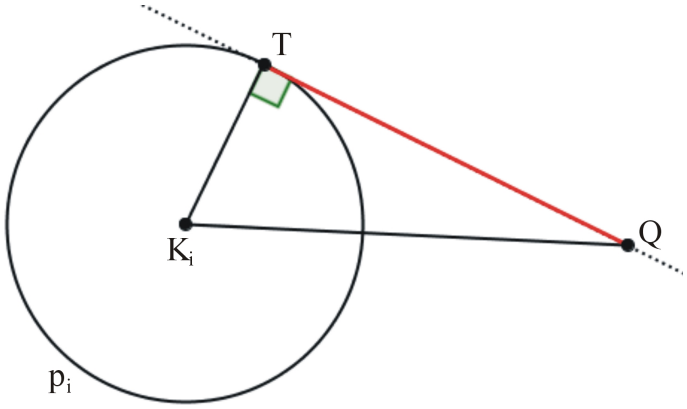
Definice 5.1 *Mějme libovolný bod Q roviny a generující kružnici $p_i = (K_i, \sqrt{w_i}) \in S'$, která má střed v bodě $K_i = (x_i, y_i)$ a poloměr $\sqrt{w_i}$. Pak* ***dotykový bod*** *T bude jeden z dotykových bodů tečny z bodu Q ke kružnici p_i, viz obr. 54.*

•

V následujícím textu odvodíme Laguerre vzdálenost, kterou budeme prozatím značit $d'_L(Q, p_i)$, kde $Q = (x_Q, y_Q)$ je libovolný bod roviny a $p_i = (K_i, \sqrt{w_i})$ je generující kružnice.

Poznámka 19 *Uvědomme si, že Laguerre vzdálenost není nic jiného než vzdálenost libovolného bodu Q roviny od příslušného dotykového bodu T ležícího na generující kružnici $p_i = (K_i, \sqrt{w_i})$. Platí tedy, že $d'_L(Q, p_i) = d_e(Q, T)$, viz obr. 54.*

•



Obrázek 54: Laguerre vzdálenost libovolného bodu Q roviny od generující kružnice p_i

Z obrázku 54 je vidět, že Laguerre vzdálenost bodu Q od generující kružnice $p_i = (K_i, \sqrt{w_i})$ můžeme zjistit pomocí Pythagorovy věty. Na základě toho můžeme psát

$$d_e(Q, K_i)^2 = d_e(Q, T)^2 + (w_i)^2.$$

Současně platí $d'_L(Q, p_i) = d_e(Q, T)$, viz poznámka 19. Po přepsání dostáváme

$$d_e(Q, K_i)^2 = d'_L(Q, p_i)^2 + w_i.$$

Po úpravách a vyjádření Laguerre vzdálenosti $d'_L(Q, p_i)$ dostáváme rovnost

$$d'_L(Q, p_i) = \sqrt{d_e(Q, K_i)^2 - w_i}. \tag{23}$$

Poznámka 20 *Laguerre vzdálenost libovolného bodu Q roviny od generující kružnice p_i se liší podle jeho polohy vůči generujcící kružnici. Pokud bod Q leží vně generující kružnice p_i, pak platí $d'_L(Q,p_i)^2 > 0$. Pokud bod Q leží na generující kružnici p_i, pak platí $d'_L(Q,p_i)^2 = 0$. A pokud bod Q leží uvnitř generující kružnice p_i, pak platí $d'_L(Q,p_i)^2 < 0$.*

•

Laguerre vzdálenost by měla být definována rovností (23). Nicméně Laguerre vzdálenost z rovnosti (23) zavedeme bez odmocniny. To uděláme úmyslně, neboť z poznámky 20 je vidět, že pro bod Q nacházející se uvnitř generující kružnice je Laguerre vzdálenost záporná. Protože chceme definovat Laguerre vzdálenost i pro body ležící uvnitř generující kružnice zavedeme Laguerre vzdálenost následovně [9], [5].

Definice 5.2 *Mějme libovolný bod Q roviny a generující kružnici $p_i = (K_i, \sqrt{w_i}) \in S'$, kde $K_i = (x_i, y_i)$ je střed kružnice p_i a $\sqrt{w_i}$ je poloměr této kružnice. Pak* ***Laguerre vzdálenost*** *definujeme*

$$d_L(Q,p_i) = d_e(Q,K_i)^2 - w_i. \tag{24}$$

•

Pro odvozenou Laguerre vzdálenost v rovnosti (23) a pro definovanou Laguerre vzdálenost v rovnosti (24) platí $d'_L(Q,p_i)^2 = d_L(Q,p_i)$.

Věta 5.1 *Osa dvou generujících kružnic $p_i, p_j \in S'$ je přímka a budeme ji značit $B_L(p_i,p_j)$. Pak hrany Laguerre Voroneho diagramu jsou částmi těchto přímek.*

•

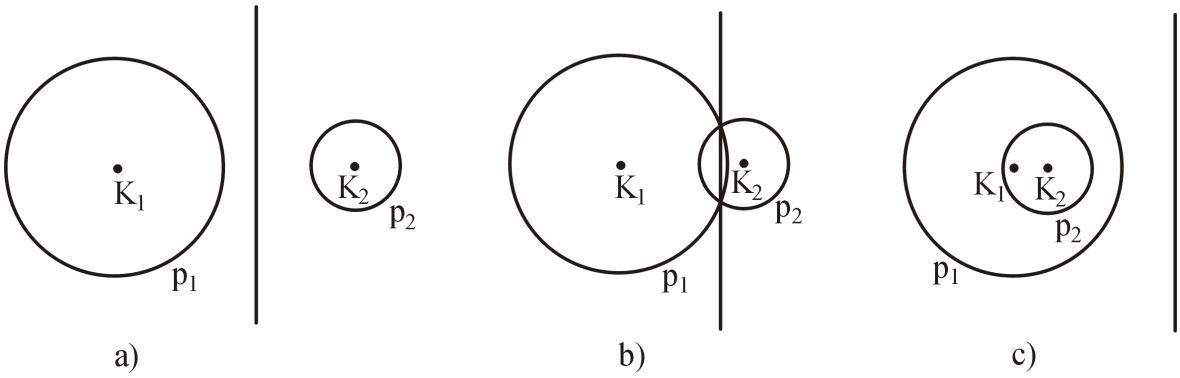
Důkaz V úvodu této kapitoly jsme zavedli množinu generujících bodů S, kterou jsme poté, pro lepší přístup k Laguerre Voroneho diagramu, modifikovali na množinu generujících kružnic S'. Nicméně, zde bude výhodnější vrátit se k množině generujících bodů S.

Hledáme množinu bodů Y_i, které mají stejnou Laguerre vzdálenost od dvou generujících bodů $P_1, P_2 \in S$. Můžeme tedy psát

$$d_L(Y_i,P_1) = d_L(Y_i,P_2).$$

Množina bodů, která má stejnou vzdálenost od dvou pevně zvolených bodů je přímka. Z toho plyne, že množina bodů Y_i, které mají stejnou Laguerre vzdálenost od dvou bodů je také přímka.

○



Obrázek 55: Laguerre Voroneho hrana v závislosti na poloze generujících kružnic

Na obrázku 55 je ukázáno, jak vypadá hrana Laguerre Voroneho diagramu pro dvě generující kružnice $p_1, p_2 \in S'$ v závislosti na jejich vzájemné poloze [10].

Mějme dvě generující kružnice $p_1 = (K_1, \sqrt{w_1}), p_2 = (K_2, \sqrt{w_2}) \in S'$. V případě, že generující kružnice nemají žádný společný bod a jedna neleží uvnitř druhé, pak Voroneho hrana protíná úsečku K_1K_2, viz obr. 55a). Pokud mají generující kružnice společný bod (*resp. body*), pak Voroneho hrana prochází tímto společným bodem (*resp. společnými body*), viz obr. 55b). Pokud generující kružnice nemají společný bod a současně jedna z generujících kružnic leží uvnitř druhé, pak Voroneho hrana neprotíná úsečku K_iK_j, viz obr. 55c), [10].

Definice 5.3 *Mějme dvě generující kružnice $p_i, p_j \in S'$ a libovolný bod Q roviny, pak* ***otevřenou Leguerre polorovinu*** *kružnice $p_i \in S'$, kde kružnice p_i je interpretací generujícího bodu $P_i \in S$ s váhou $\sqrt{w_i}$, definujeme*

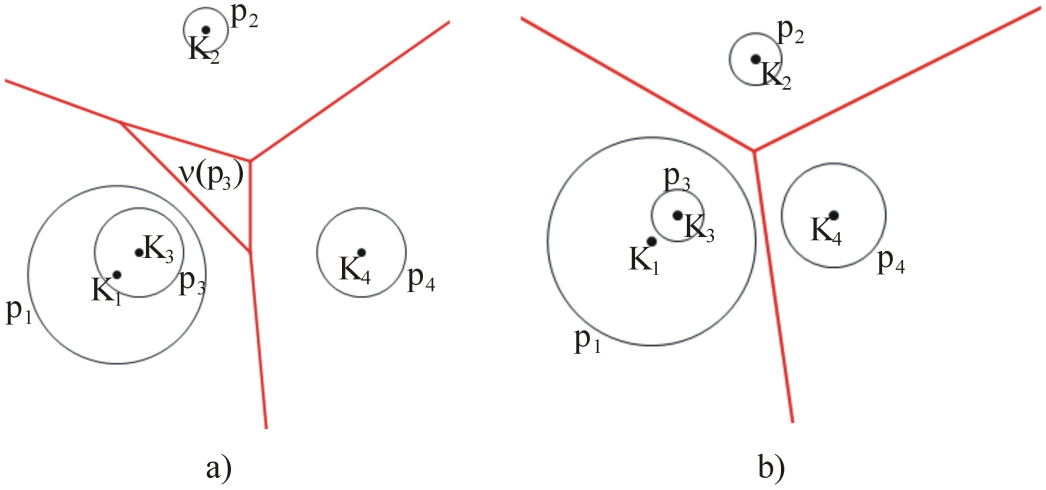
$$h_L(p_i, p_j) = \{Q | d_L(Q, p_i) < d_L(Q, p_j)\}.$$

Laguerre Voroneho buňku *kružnice p_i definujeme*

$$\nu_L(p_i) = \bigcap_{p_j \in S', p_j \neq p_i} h_L(p_i, p_j).$$

•

Obrázky 56a), b) jsou příklady Laguerre Voroneho diagramu. Všimněme si, že generující kružnice některých Voroneho buněk neobsahují své generátory, např. Voroneho buňka $\nu_L(p_3)$, viz obr. 56a). Takovéto generující kružnice se označují jako *"substantial circle"*. Obdobně se může stát, že některé generující kružnice budou natolik ovlivněny zbylými generátory, že jejich Voroneho buňka bude prázdná množina. V diagramu se to projeví tak, že nebudou mít svou vlastní Voroneho buňku, např. generující kružnice p_3 na obrázku 56b). Takovéto generující kružnice se označují jako *"trivial circle"* [10].



Obrázek 56: Laguerre Voroneho diagram, kde generující kružnice je :a) "substantial circle"; b) "trivial circle"

V následující kapitole uvedeme Inkrementální algoritmus pro Laguerre Voroneho diagram, jehož implementace je součástí této práce. Popíšeme postup algoritmu a současně uvedeme některé problémy a jejich řešení, které se vyskytly při jeho implementaci.

5.1 Inkrementální algoritmus pro Laguerre Voroneho diagram

Mějme $S' = \{p_1, p_2, \ldots, p_n\}$ množinu generujících kružnic $p_i = (K_i, \sqrt{w_i})$, kde $K_i = (x_i, y_i)$ jsou středy kružnic a $\sqrt{w_i}$ jsou jejich poloměry. Pro jednoduchost budeme předpokládat následující omezení množiny S'

- žádné tři a více generující kružnice nemají společný průnik
- žádné dvě generující kružnice p_i, p_j neleží současně uvnitř jiné generující kružnice p_k, kde $i \neq j \neq k$
- žádné tři středy generujících kružnic nejsou kolineární
- žádné dva středy generujících kružnic nemají stejnou x-ovou souřadnici
- Laguerre Voroneho diagram není degenerovaný, *tzn. stupeň každého Voroneho vrcholu je tři*
- žádná generující kružnice není "trivial circle", *tzn. Voroneho diagram neobsahuje kružnice, jejichž Voroneho buňka je prázdná množina.*

Inkrementální algoritmus pro Laguerre Voroneho diagram je analogický jako Inkrementální algoritmus pro Obecný Voroneho diagram, viz kapitola 2.1.1. Hrany obou

těchto Voroneho diagramů, Obecného i Laguerre, jsou částmi přímek. Hlavní rozdíl je v určování vzdálenosti, ta se u Laguerre Voroneho diagramu určuje jako vzdálenost libovolného bodu Q roviny od dotykového bodu T_i generující kružnice p_i,
viz definice 5.2.

Poznámka 21 *Bod Q náleží Laguerre Voroneho buňce $\nu_L(p_i)$, pro generující kružnici p_i, pokud platí $d_L(Q, p_i) < d_L(Q, p_j)$. V případě rovnosti bod Q leží na Voroneho hraně mezi buňkami $\nu(p_i), \nu(p_j)$.*

•

Inkrementální algoritmus pro Laguerre Voroneho diagram spočívá v nalezení Laguerre Voroneho diagramu pro tři generující kružnice $p_i \in S'$, kde $i = 1, 2, 3$, a následně postupném přidávání dalších generujících kružnic $p_j \in S'$, kde $j = 4, 5, \ldots, n$, n udává počet generujících kružnic v množině S'. Pro každou nově přidanou generující kružnici určíme její lokalizaci, tzn. ve které Laguerre Voroneho buňce nově přidaná generující kružnice leží. Následně prohledáváme a vytváříme osy $B_L(p_{k+1}, p_l)$, kde k je počet generujících kružnic stávajícího Voroneho diagramu a l udává index generující kružnice stávajícího diagramu, která je ovlivněna nově přidanou generující kružnicí. Po ukončení prohledávání přejdeme k začištění hran a vymažeme ty části hran, které již nesplňují vlastnosti Voroneho hran.

Lemma 5.1 *Nově přidaná generující kružnice $p_{k+1} = (K_{k+1}, \sqrt{w_{k+1}})$, kde K_{k+1} je střed generující kružnice p_{k+1}, leží uvnitř Laguerre Voroneho buňky $\nu_L(p_l)$, stávajícího Laguerre Voroneho diagramu, pokud platí*

$$d_L(K_{k+1}, p_l) < d_L(K_{k+1}, p_j), \tag{25}$$

kde $p_l \neq p_j$.

•

Důkaz Budeme vycházet z definice Voroneho buňky, viz definice 5.3. Laguerre Voroneho buňka $\nu_L(p_i)$ obsahuje právě ty body Q roviny, které mají od generující kružnice p_i menší Laguerre vzdálenost než od jiné generující kružnice z množiny S'. Platí tedy, že $Q \in \nu_L(p_i)$, pokud platí

$$d_L(Q, p_i) < d_L(Q, p_j). \tag{26}$$

Jestliže bod Q v nerovnici (26) nahradíme středem K_{k+1} nově přidané generující kružnice p_{k+1}, pak platí

$$d_L(K_{k+1}, p_i) < d_L(K_{k+1}, p_j), \tag{27}$$

kde $i \neq j$. Což je nerovnice (25).

◦

Algoritmus (Inkrementální pro Laguerre Voroneho diagram)
Vstup: Množina generujících kružnic S' = $\{p_1, p_2, \ldots, p_n\}$.
Výstup: Voroneho diagram množiny S'.

1. **Inicializace**

   (a) vytvoření fronty F a vložení generujících kružnic $p_1, p_2, \ldots, p_n$ do fronty

   (b) seřazení generujících kružnic p_i, kde $i = 1, 2, \ldots, n$, od nejnižší x-ové souřadnice středů kružnic

2. **Postup algoritmu**

   (a) výběr prvních tří generujících kružnic p_1, p_2, p_3 z fronty F a vytvoření Laguerre Voroneho diagramu pro tyto generující kružnice

   (b) výběr kružnice p_{k+1} z fronty, kde k udává počet generujících kružnic stávajícího Laguerre Voroneho diagramu

   (c) určení lokalizace kružnice p_{k+1} ve stávajícím Laguerre Voroneho diagramu

   (d) nalezení osy $B_L(p_{k+1}, p_l)$ dvou kružnic

   (e) nalezení průsečíků osy $B_L(p_{k+1}, p_l)$ s hranicí Laguerre Voroneho buňky $\nu_L(p_l)$, kde hranice je množina hran ohraničující danou Voroneho buňku

   (f) rozhodnutí o počtu průsečíků osy úsečky $B_L(p_{k+1}, p_l)$ s hranicí Laguerre Voroneho buňky $\nu_L(p_l)$

      i. existuje pouze jeden průsečík

         A. tento průsečík určí následující Laguerre Voroneho buňku $\nu_L(p_m)$, která bude ovlivněna generující kružnicí p_{k+1}

         B. nalezení osy $B_L(p_{k+1}, p_m)$ a jejích průsečíků s hranicí Laguerre Voroneho buňky $\nu_L(p_m)$

         C. rozhodnutí o počtu průsečíků

            - existuje jeden průsečík: pokračování bodem (2g)
            - existují dva průsečíky: zvolení průsečíku, který neleží na společné hranici dvou buněk $\nu_L(p_l), \nu_L(p_m)$, pokračování bodem (2(f)i)

      ii. existují dva průsečíky

         A. výběr jednoho libovolného průsečíku, čímž je určena Laguerre Voroneho buňka $\nu_L(p_m)$, která jako další bude ovlivněna nově přidanou generující kružnicí p_{k+1}

         B. nalezení osy $B_L(p_{k+1}, p_m)$ a jejích průsečíků s hranicí Voroneho buňky $\nu(p_m)$

C. rozhodnutí o počtu průsečíků

- existuje jeden průsečík: zvolení druhého průsečíku z původně nalezených a pokračování předcházejícím bodem
- existují dva průsečíky: zvolení průsečíku, který neleží na společné hranici dvou buněk $\nu_L(p_l), \nu_L(p_m)$

(g) vyrušení hran uvnitř nově vzniklé Voroneho buňky νp_{k+1} a pokračování bodem (2b)

Inkrementální algoritmus pro Laguerre Voroneho diagram tak, jak zde byl do této chvíle prezentován, neinformuje o tom, jakým způsobem řešit dílčí části algoritmu, jako například lokalizaci nově přidané generující kružnice nebo začištění hran. V následující podkapitole uvedeme řešení některých těchto základních částí Inkrementálního algoritmu tak, jak byly řešeny při jeho implementaci.

5.1.1 Řešení dílčích částí Inkrementálního algoritmu pro Laguerre Voroneho diagram

Vstupem algoritmu je seznam generujících kružnic $p_i = (x_i, y_i, \sqrt{w_i}) \in S'$, $i = 1, 2, \ldots, n$. Seznam těchto generujících kružnic zadáváme do položky $vstup = \{p_1, p_2, \ldots, p_n\}$, např.

$$vstup = \{\{0, -8, 1\}, \{-6, 0, 6\}, \{-1, 7, 2\}, \{1, 1, 4\}, \{10, -5, 3\}\}.$$

Aby se nestalo, že některé generující kružnice budou mimo výstupní zorné pole je v úvodu programu možnost si velikost pole zvolit. Velikost zorného pole volíme v položce $pole = \{a, b\}$, kde a udává spodní mez x-ové, y-ové osy a b udává horní mez x-ové, y-ové osy, např.

$$pole = \{-15, 15\}.$$

Velikost pole je úmyslně nastavena tak, aby se dala navolit podle potřeby uživatele. Je to z toho důvodu, aby bylo možné zorné pole zvětšit na požadovanou velikost.

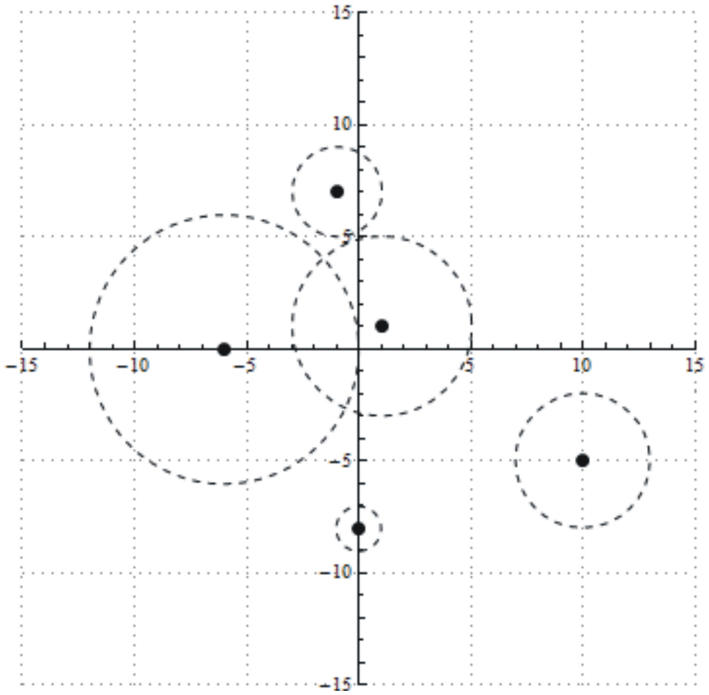
Zorné pole, se vstupními generujícími kružnicemi je vidět na obrázku 57.

Jedna z hlavních částí programu řeší přidávání generujících kružnic a následné prohledávání pro nalezení Laguerre Voroneho buňky nově vložené generující kružnice. Abychom omezili počet možností, kam se může nově přidaná generující kružnice umístit, a tím zjednodušili prohledávání, je úvodní množina vstupních generujících kružnic seřazena podle x-ových souřadnic jejich středů, od nejnižší po nejvyšší hodnotu. Seřazená množina vstupních bodů je v programu označena a.

Vstupní množina, která je zde uvedena jako příklad, bude po seřazení vypadat

$$a = \{\{-6, 0, 6\}, \{-1, 7, 2\}, \{0, -8, 1\}, \{1, 1, 4\}, \{10, -5, 3\}\}.$$

Po seřazení vstupní množiny generujících kružnic vezmeme první tři generující kružnice a vytvoříme pro ně Laguerre Voroneho diagram. To, jakým způsobem Laguerre Voroneho diagram pro tři kružnice vzniká, popíšeme v následující části.



Obrázek 57: Zorné pole se vstupními generujícími kružnicemi

Laguerre Voroneho diagram pro tři kružnice První tři generující kružnice, pro které vytváříme Laguerre Voroneho diagram, vložíme do nového seznamu. Tento nový seznam, prvních tří generujících kružnic, je v programu značen v a pro náš příklad vypadá

$$v = \{\{-6, 0, 6\}, \{-1, 7, 2\}, \{0, -8, 1\}\}.$$

Abychom byli schopni vykreslit Laguerre Voroneho diagram pro tři kružnice, potřebujeme znát osu $B_L(p_i, p_j), i \neq j$, kde $i, j = 1, 2, 3$.

Nalezení osy dvou generujících kružnic Pro nalezení osy dvou libovolných generujících kružnic je v programu vytvořena procedura ***Bisector[a1,a2]***, kde $a1, a2$ jsou vstupy procedury v podobě dvou generujících kružnic mezi nimiž hledáme osu. Abychom byli schopni nalézt rovnici osy potřebujeme znát další informace. V proceduře je tento problém řešen tak, že najdeme dva body, které jsou součástí osy dvou generujících kružnic. Tyto dva body nalezneme díky jejich vlastnostem. Pro body, které leží na ose platí, že mají stejnou vzdálenost od dvou generujících kružnic. Této vlastnosti využijeme. Chceme najít souřadnice bodu $A = (x, y)$, který má stejnou vzdálenost od dvou generujících kružnic $p_i = (K_i, \sqrt{w_i}), p_j = (K_j, \sqrt{w_j})$, kde $K_i = (x_i, y_i), K_j = (x_j, y_j)$ jsou středy generujících kružnic. Budeme řešit soustavu dvou rovnic

$$\begin{array}{rcl} \mathrm{d}_L(A, p_i) & = & \text{konst.} \\ \mathrm{d}_L(A, p_j) & = & \text{konst.} \end{array}$$

Konstanta je volena tak, aby nenastal případ, kdy soustava nemá řešení. Abychom toto zajistili je konstanta pro každé dvě generující kružnice p_i, p_j volena různě

$$konst. = \sqrt{w_i} + \sqrt{w_j} + |x_i - x_j| + |y_i - y_j|.$$

Řešením soustavy rovnic jsou dva body ležící na ose dvou generujících kružnic. Díky těmto bodům nalezneme obecnou rovnici osy a současně ji vykreslíme.

Všechny osy, pro libovolné dvě generující kružnice, které jsou v programu počítány, jsou označovány jako bla_i a uchovávají v sobě informace o dvou bodech, jež jsou řešením výše uvedené soustavy rovnic, obecnou a středovou rovnici osy, vykreslení dané osy a generující kružnice mezi nimiž byla osa úsečky vypočtena.

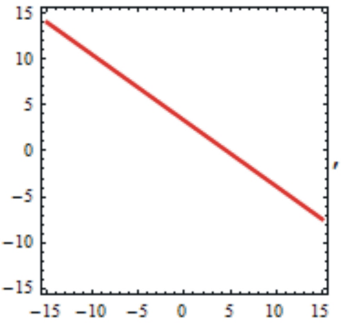
Pro dvě generující kružnice $p_1 = \{-6, 0, 6\}, p_2 = \{-1, 7, 2\}$ je osa

$$bla_1 = Bisector[p_1, p_2]$$

a její výstup je vidět na obrázku 58.

$$\left\{\left\{\left\{\frac{1}{74}\left(-179 + 7\sqrt{29455}\right), \frac{1}{74}\left(371 - 5\sqrt{29455}\right)\right\}, \left\{\frac{1}{74}\left(-179 - 7\sqrt{29455}\right), \frac{1}{74}\left(371 + 5\sqrt{29455}\right)\right\}\right\},\right.$$

$$-\frac{7}{74}\left(371 - 5\sqrt{29455}\right) - \frac{5}{74}\left(-179 + 7\sqrt{29455}\right) + 5\,x1 + 7\,y1,$$



$$\left.\frac{1}{5}\left(\frac{7}{74}\left(371 - 5\sqrt{29455}\right) + \frac{5}{74}\left(-179 + 7\sqrt{29455}\right) - 7\,y1\right), \{-6, 0, 6\}, \{-1, 7, 2\}\right\}$$

Obrázek 58: Výstup busectoru bla_1 pro dvě generující kružnice p_1, p_2

Nyní se vraťme zpět k nalezení Laguerre Voroneho diagramu pro tři generující kružnice. Pro nalezení tří os bla_i mezi generujícími kružnicemi $p_i, i = 1, 2, 3$, použijeme proceduru *Bisector*. V případě, že středy generujících kružnic nejsou kolineární, hledáme Voroneho vrchol. Voroneho vrchol je průnikem tří os pro tři generující kružnice. Najdeme ho jako řešení soustavy rovnic pro dvě osy. Obecnou rovnici osy označme bla_{i2} a vypočtěme soustavu rovnic

$$\begin{aligned} \mathrm{bla}_{i2} &= 0 \\ \mathrm{bla}_{j2} &= 0 \end{aligned}$$

Tento průsečík je v programu pojmenován *vvrchol* a uchovává v sobě informace o osách $bla_i, i = 1, 2, 3$, které jím prochází.

Nyní již máme zjištěny informace o třech osách bla_i pro tři generující kružnice. Abychom mohly dokončit Laguerre Voroneho diagram pro tři generující kružnice, potřebujeme osy začistit. Voroneho vrchol nám každou z os rozděluje na dvě polopřímky, začištěním je zde myšleno vymazání té polopřímky, která není součástí Voroneho hrany. Postup začištění popíšeme v následující části.

Začištění pro tři generující kružnice Pro začištění Laguerre Voroneho diagramu pro tři generující kružnice je v programu vytvořena procedura ***Zacisteni[zk1, zk2,zk3,vrchol,b1,b2,b3]***, kde $b1, b2, b3$ jsou vstupy procedury v podobě tří generujících kružnic, pro které začišťujeme, *vrchol* je Voroneho vrchol jež jsme vypočetli výše a $zk1, zk2, zk3$ jsou body ležící na dané ose.

Body $zk1, zk2, zk3$ jsou body, které leží po řadě na osách $bla_i, i = 1, 2, 3$, a jejich x-ová (*popř. y-ová*) souřadnice je rovna x-ové (*popř. y-ové*) souřadnici Voroneho vrcholu plus dostatečně velká konstanta. V programu jsme konstantu zvolili rovnu sto. Pro hledání těchto bodů *zk* je v programu vytvořena procedura ***BodBisectoru[v1,v2,blai,vrcholi]***, kde vstup $v1, v2$ jsou generující kružnice, *blai* je osa pro generující kružnice $v1, v2$ a *vrcholi* je Voroneho vrchol, body této procedury označujeme zk_i. Procedura *BodBisectoru*, pro dvě generující kružnice, které nemají stejnou y-ovou souřadnici, počítá soustavu rovnic

$$\begin{array}{lcl} \mathrm{bla}_{i2} & = & 0 \\ \mathrm{x1} & = & \mathrm{x}_{vrcholi} + konst. \end{array}$$

kde bla_{i2} označuje obecnou rovnici osy s neznámými $x1, y1$, $konst. = 100$ a $x_{vrcholi}$ označuje x-ovou souřadnici Voroneho vrcholu.

Pokud budou mít dvě generující kružnice stejnou y-ovou souřadnici bude soustava rovnic vypadat

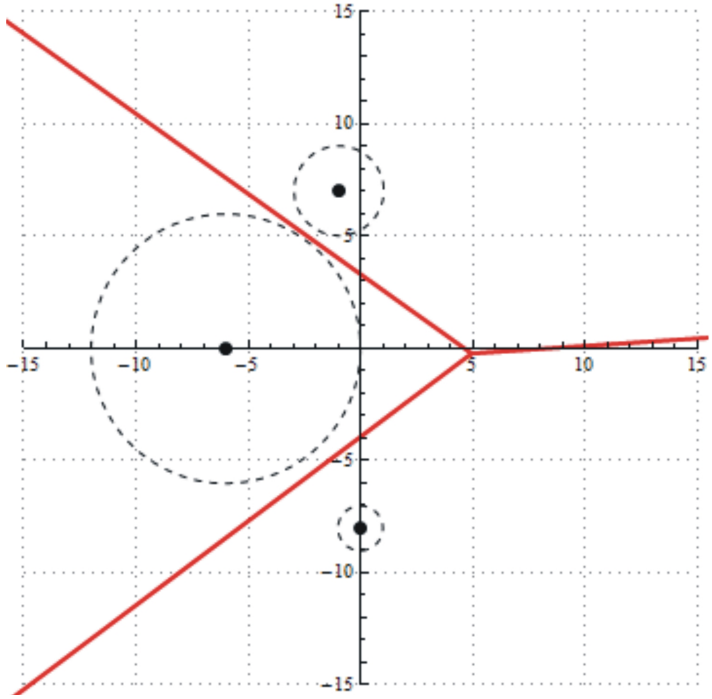
$$\begin{array}{lcl} \mathrm{bla}_{i2} & = & 0 \\ \mathrm{y1} & = & \mathrm{y}_{vrcholi} + konst. \end{array}$$

kde $konst. = 100$. Konstanta, která se přičítá, je úmyslně volena vyšší, je to z toho důvodu, aby se hledaný bod nenacházel uvnitř některé generující kružnice, to by následně mohlo vést k chybnému výpočtu v proceduře *Zacisteni*.

Pokud máme zjištěny body $zk_i, i = 1, 2, 3$, můžeme přistoupit k využití procedury *Zacistení*. Tato procedura porovnává Laguerre vzdálenost bodu zk_i od jedné generující kružnice p_m, která je jednou z tvořících kružnic osy bla_i a od jiné generující kružnice p_n, která není tvořící kružnice osy bla_i. Na základě těchto porovnání vykreslí správnou polopřímku dané osy. Aby bod zk_i byl součástí Voroneho hrany musí pro něj platit

$$d_L(zk_i, K_m) < d_L(zk_i, K_n),$$

kde K_m je střed generující kružnice pro osu bla_i a K_n je střed generující kružnice, která není tvořící generující kružnicí pro osu bla_i. Na obrázku 59 je ukázán diagram pro tři generující kružnice.



Obrázek 59: Laguerre Voroneho diagram pro tři generující kružnice

Nyní již máme vytvořen Laguerre Voroneho diagram pro tři generující kružnice. V této chvíli přidáme do stávajícího diagramu, v tomto případě do diagramu pro tři generující kružnice, další kružnici v pořadí, v našem příkladě je to čtvrtá kružnice ze seřazeného seznamu a, a to $p_4 = \{1, 1, 4\}$.

Pro zacyklení a další využití jsou všechny osy uchovávány v seznamu, který je pojmenovám $primky = \{bla_i\}, i = 1, 2, \ldots, k$, kde k je celkový počet os, které jsou během průběhu algoritmu vypočteny. Kromě os budeme uchovávat i seznam Voroneho vrcholů. V programu jsou celkem dva takovéto seznamy, první je pojmenován l a uchovává všechny Voroneho vrcholy, ktreré byly v průběhu algoritmu vypočteny, i ty, které již Voroneho vrcholy nejsou, druhý seznam je pojmenován ll a uchovává pouze platné Voroneho vrcholy.

Cyklus probíhá od $j = 1$ do $j \leq n-3$, kde n je počet vstupních generujících kružnic.

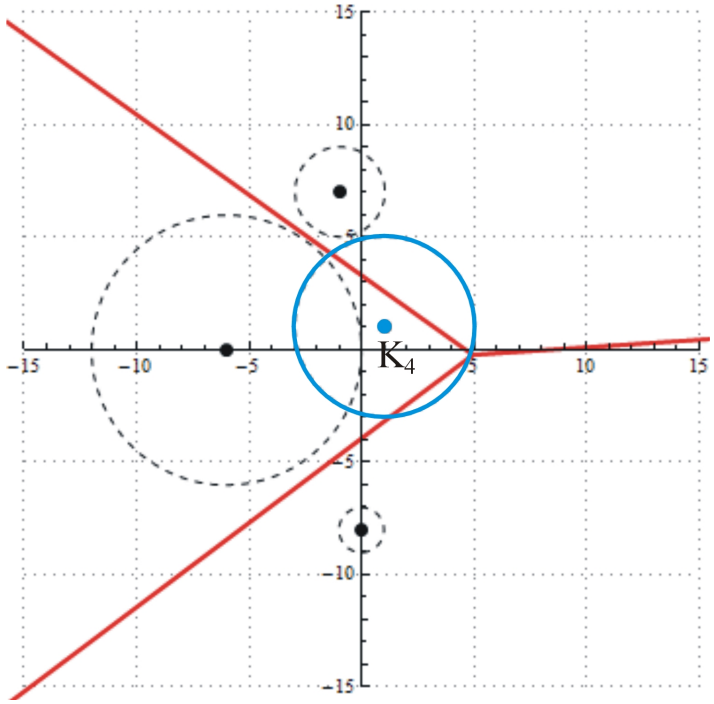
Po přidání čtvrté generující kružnice je potřeba určit lokalizaci, tzn. určit Voroneho buňku, ve které nově přidaná generující kružnice leží. To je řešeno následovně.

Lokalizace Pro lokalizaci nově přidané generující kružnice použijeme opět řazení množiny generujících kružnic. Řadit nebudeme celou množinu vstupních kružnic, ale

pouze ty kružnice pro které je Voroneho diagram již hotov. Tento proces je v programu pojmenován $serazeni_i$. Pro přidání čtvrté kružnice $p_4 = \{1, 1, 4\}$, tedy budeme řadit pouze první tři generující kružnice. Kružnice řadíme od nejmenší po největší Laguerre vzdálenost středu nově přidané generující kružnce od ostatních generujících kružnic. V našem příkladě bude seřazení prvních tří kružnic vypadat následovně

$$serazeni_1 = \{\{-6, 0, 6\}, \{-1, 7, 2\}, \{0, -8, 1\}\}.$$

Je jasné, že pokud řadíme generující kružnice stávajícího Voroneho diagramu podle jejich nejmenší Laguerre vzdálenosti od nově přidané generující kružnice, pak kružnice na první pozici v $serazeni_1$ je generující kružnice v jejíž Voroneho buňce nově přidaná generující kružnice leží,viz obr. 60.



Obrázek 60: Lokalizace nově vložené generující kružnice p_4

Po určení generující kružnice $p_i, i = 1, 2, 3$, v jejíž Voroneho buňce nově vložená kružnice p_4 leží, najdeme osu $B_L(p_i, p_4)$ pomocí procedury $Bisector$, viz výše. Tato nově nalezená osa je pojmenována bla_4. Abychom mohli pokračovat v algoritmu musíme najít průsečíky osy $B_L(p_i, p_4)$ s hranicí Voroneho buňky kružnice p_i.

Nalezení průsečíků V první řadě si uvědomme, že průsečíky osy $B_L(p_i, p_4)$ s hranicí Voroneho buňky kružnice p_i jsou současně nové Voroneho vrcholy, které přidáme do seznamu Voroneho vrcholů l. Myšlenka nalezení Voroneho vrcholů spočívá v tom, že

nalezneme průsečíky $B_L(p_i, p_4)$ se všemi ostatními osami. Budeme tedy řešit soustavy rovnic

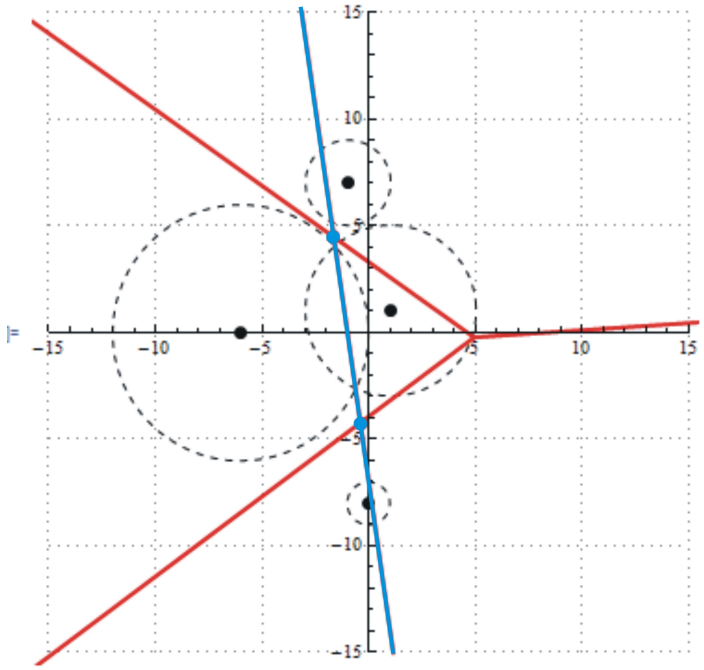
$$\begin{array}{rcl} \text{bla}_i & = & 0 \\ \text{bla}_4 & = & 0 \end{array}$$

kde bla_i jsou rovnice os stávajícího Laguerre Voroneho diagramu. Po nalezení všech těchto průsečíků budeme testovat, který z nich splňuje podmínky na Voroneho vrchol. K tomuto účelu je v programu vytvořena procedura ***TestPrusecikuOb[prusecikyi, serazenii]***, kde vstupem procedury je seznam všech nalezených průsečíků a seznam $serazeni_1$. V proceduře je počítána Laguerre vzdálenost každého jednotlivého průsečíku od generující kružnice v $serazeni_1$, tyto hodnoty seřadíme od nejnižší po nejvyšší pro každý z průsečíků. Dále testujeme, zda vzdálenost jednotlivých průsečíků od příslušných generujících kružnic, tzn. od generujících kružnic, pro které jsou vytvořeny osy, jejichž průnikem je konkrétní průsečík, je stejná a současně, zda je menší než nejmenší Laguerre vzdálenost průsečíku od všech generujících kružnic, pokud toto platí vypíšeme $True$ pokud toto neplatí vypíšeme $False$.

Pro naše účely potřebujeme znát souřadnice Voroneho vrcholů, proto je v programu procedura ***VorVrcholy2[testprusecikui, prusecikyi]***, jejíž vstupem je výstup předešlé procedury $TestPrusecikuOb$ a seznam všech nalezených průsečíků. V Proceduře $VorVrcholy2$ procházíme výstup procedury $TestPrusecikuOb$ a pokud je roven $True$, tak daný průsečík vypíšeme, pokud je roven $False$, pak neděláme nic. Výstupem procedury $VorVrcholy2$ jsou pouze Voroneho vrcholy jež vzniknou průnikem osy $B_L(p_i, p_4)$ s hranicí Voroneho buňky $\nu(p_i)$, kde p_i je kružnice v jejíž Voroneho buňce nově vložená generující kružnice p_4 leží. V programu je Voroneho vrchol pojmenován $vrcholy_i$, viz obr. 61.

Nyní máme nalezené Voroneho vrcholy, v našem příkladě jsou dva. Pokud jsou nalezeny dva Voroneho vrcholy musíme, pro další prohledávání, zvolit jeden z nich. V programu vždy volíme Voroneho vrchol, který je v seznamu $vrcholy_i$ na první pozici. Podle obecných informací, uvedených u Inkrementálního algoritmu, tyto Voroneho vrcholy určují Voroneho buňku, která jako další bude ovlivněna nově vloženou generující kružnicí. To, jak je problém řešen, nyní uvedeme.

Určení následující Voroneho buňky Pro určení následující buňky je v programu vytvořena procedura ***LokBisectoru2[vrcholyi, a4]***, kde vstupem jsou $vrcholy_1$ a nově přidaná kružnice p_4. Procedura $LokBisectoru2$ testuje, zda Laguerre vzdálenost Voroneho vrcholu od generujících kružnic, vyjma nově přidané kružnice a té v jejíž buňce nově přidaná generující kružnice leží, je stejná jako Laguerre vzdálenost Voroneho vrcholu od nově přidané generující kružnice, pokud ano, pak se odvoláme na proceduru $Bisector$ a vytvoříme novou osu $bla_5 = B_L(p_j, p_4)$. To, od kterých kružnic měřím Laguerre vzdálenost Voroneho vrcholu, je informace, která je uchovávána u Voroneho vrcholu. U každého Voroneho vrcholu jsou uchovávány informace o jeho souřadnicích, o třech osách, jež se protínají v daném Voroneho vrcholu. A jak bylo řečeno dříve,



Obrázek 61: Laguerre Voroneho diagram pro tři kružnice s nově vloženou kružnicí p_4, osou $B_L(p_i, p_4)$ a Voroneho vrcholy

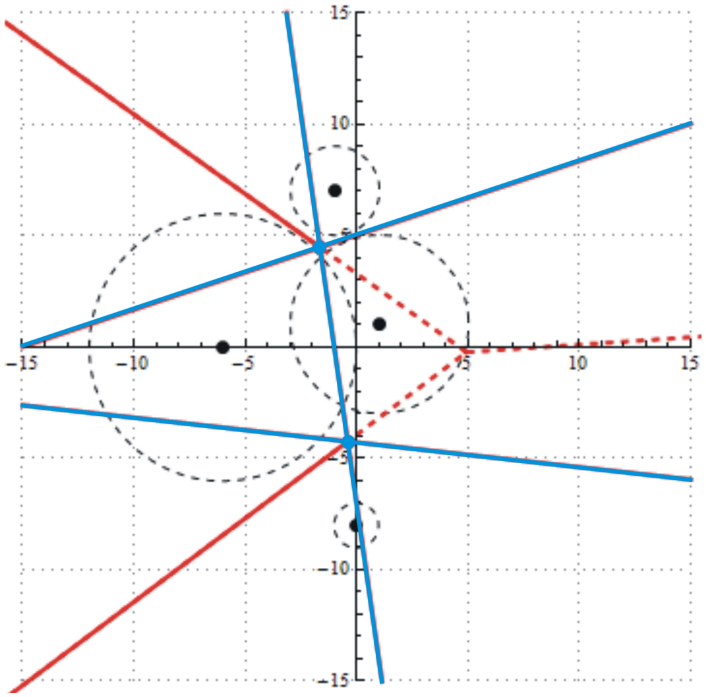
u každé osy jsou uchovávány informace o generujících kružnicích, pro které jsou osy vytvořeny.

Tento postup opakujeme pro druhý nalezený vrchol. Tím získáme další osu. Laguerre Voroneho diagram v tuto chvíly, je vidět na obrázku 62.

Čárkovaná část je část, která bude při začištění odstraněna. Princip začištění uvedeme později.

Jak již bylo řečeno dříve, program, který je součástí této práce, neřeší specialní případy. Množina generujících kružnic S' je omezena již na začátku této kapitoly, nicméně program má ještě jedno omezení. Toto omezení se týká počtu buněk do kterých může nově přidaná kružnice zasahovat, počet těchto buněk je nejvýše tři. Díky řazení vstupní množiny podle x-ových souřadnic středů generujících kružnic mohou ve výsledném diagramu být Voroneho buňky, jejichž ohraničení je tvořeno více než třemi částmi os, ale při samotném přidání nové kružnice toto možné není. V případě, že by nově přidaná kružnice zasahovala do více jak tří Voroneho buněk, program vypíše chybovou hlášku.

Poslední krok, který zbývá, je začištění hran. Princip začištění os, na kterých leží pouze jeden Voroneho vrchol je stejný jako začištění Laguerre Voroneho diagramu pro tři generující kružnice, viz kap. 5.1.1 na str. 72.



Obrázek 62: Laguerre Voroneho diagram pro tři kružnice s nově vloženou kružnicí p_4, osami úseček $B_L(p_i, p_4), i = 1, 2, 3$, a Voroneho vrcholy vzniklé na hranici Voroneho buňky $\nu_L(p_i)$

Začištění Procedura *Zacisteni* ze strany 72 začišťuje osy tří generujících kružnic najednou. Nyní, pro naše účely, je výhodnější začišťovat hrany postupně. Pro tento účel je v programu vytvořena procedura ***Zacisteni2[zk4, vrchol2, b1, b2, b3, b4]***, kde $vrchol2$ je Voroneho vrchol, ve kterém začisťujme, $zk4$ je bod ležící na dané ose a jeho x-ová souřadnice je rovna x-ové souřadnici Voroneho vrcholu plus konstanta, viz procedura *BodBisectoru*, která je popsána na straně 72. Vstupy $b1, b2, b3, b4$ procedury $Zacisteni2$ jsou generující kružnice, kde $b1$ je jedna z generujících kružnic pro kterou je daná osa vytvořena, $b2$ je nově přidaná generující kružnice a $b3, b4$ jsou generující kružnice, které netvoří danou osu.

Abychom zkrátili a zpřehlednili zápis programu je vytvořena procedura ***PripadTreti[vrcholy, serazeni]***, kde vstupem procedury je Voroneho vrchol a $serazeni_i$. Procedura $PripadTreti$ sjednocuje a zpracovává dvě předešlé procedury *BodBisectoru* a $Zacisteni2$.

Pokud na ose leží dva Voroneho vrcholy, pak pro ně vytvoříme úsečku. Důležitá je v tomto případě myšlenka, že každá osa je pojmenována $bla_i, i = 1, 2, \ldots, k$, kde k je celkový počet os, které program vypočetl. Každá začištěná hrana je pojmenována $zabla_i, i = 1, 2, \ldots, k$ a indexem vždy odpovídá příslušné ose. Při přidání nové kružnice jsou vždy přidány nové hrany bla_j a také jsou začištěny $zabla_j$. Některé hrany se při přidání nové kružnice nezmění, jejich původní začištění se tedy také nezmění. Některé

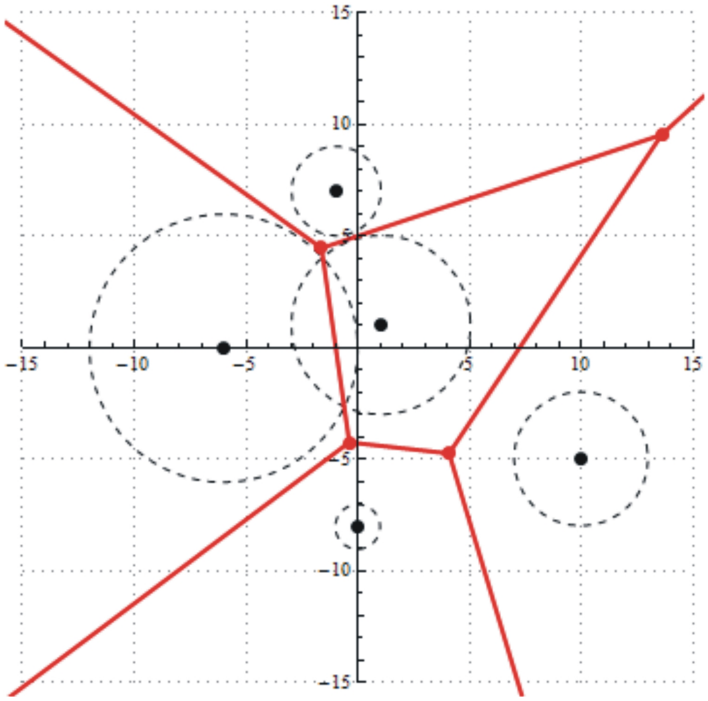
hrany se při přidání nové kružnice změní, jejich původní začištění $zabla_j$ bude přepsáno novým. Pomocí přepisování začištěných hran jsou hrany po ukončení cyklu správně začištěny. Jediné co zbývá je, tyto hrany vykreslit.

Problém vykreslení konečného Laguerre Voroneho diagramu spočíval v tom, že některé hrany nejsou pouze začištěny, ale při přidání nové generující kružnice budou vymazány. Taková hrana je vidět na obrázku 62. Z toho důvodu je potřeba ze seznamu začištěných hran $zabla_i$ tyto hrany vypustit. To je vyřešeno tak, že máme seznam všech Voroneho vrcholů ll a každý vrchol uchovává informace o osách, které jím prochází. Pokud vypíšeme pro každý Voroneho vrchol indexy jeho os a pro každý takovýto index dáme vykreslit začištěnou hranu $zabla_i$, pak již máme Laguerre Voroneho diagram hotov.

Pro množinu vstupních generujících kružnic, jež zde byla dána jako příklad,

$$vstup = \{\{0, -8, 1\}, \{-6, 0, 6\}, \{-1, 7, 2\}, \{1, 1, 4\}, \{10, -5, 3\}\},$$

je Laguerre Voroneho diagram znázorněn na obrázku 5.1.2.



Obrázek 63: Laguerre Voroneho diagram pro pět generujících kružnic

5.1.2 Řešené příklady

V této kapitole uvedeme několik příkladů pro zajímavě zvolené množiny generujících kružnic. A vzhledem k omezení algoritmu ukážeme i příklady množin s větším množstvím generujících kružnic.

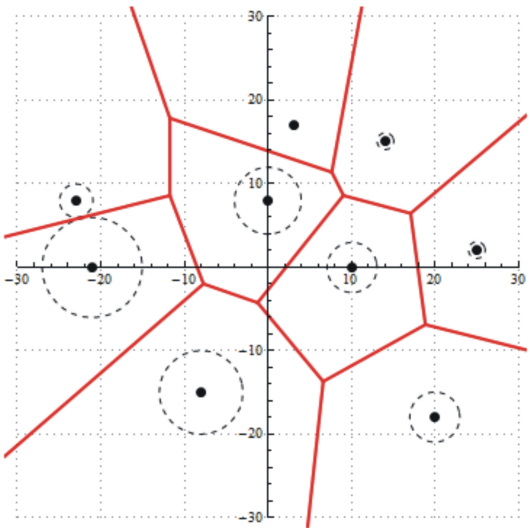
V počátcích implementace byl algoritmus řešen tak, aby fungoval i ve specialních případech, jako např. při kolinearitě středů kružnic. Postupně bylo od tohoto plánu upuštěno s tím, že pokud bude dostatek času mohou se některé specialní připady v programu dořešit. Z tohoto důvodu algoritmus zvládá vykreslit některé konkrétní příklady specialních případů, ale ne pro všechny možné kombinace umístění generujících kružnic daného specialního případu.

První dva příklady jsou ukázkou Voroneho diagramu pro množinu S' s větším množstvím generujících kružnic.

Příklady tři, čtyři a pět jsou ukázkou zajímavě zvolené množiny S'. Každý z těchto tří příkladů nějak porušuje předpoklady kladené na množinu S', viz. kapitola 5.1 na straně 66. Zvolení takovýchto příkladů je možné, právě díky tomu, že v počátcích implementace byl algoritmus řešen i pro specialní případy.

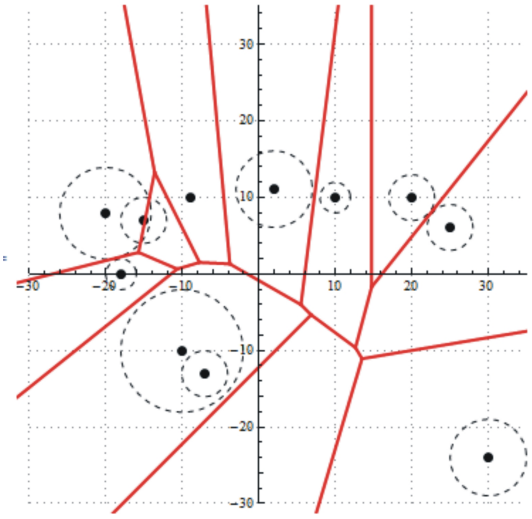
1. příklad $pole = \{-30, 30\}$

$vstup = \{\{-23, 8, 2\}, \{-8, -15, 5\}, \{-21, 0, 6\}, \{0, 8, 4\}, \{3, 17, 0\}, \{10, 0, 3\}, \{14, 15, 1\}, \{20, -18, 3\}, \{25, 2, 1\}\}$



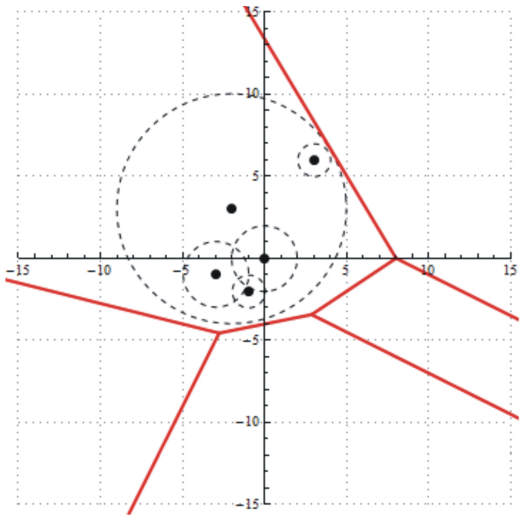
2. příklad $pole = \{-30, 35\}$

$vstup = \{\{-20, 8, 6\}, \{-15, 7, 3\}, \{-18, 0, 2\}, \{-10, -10, 8\}, \{-9, 10, 0\}, \{-7, -13, 3\}, \{2, 11, 5\}, \{10, 10, 2\}, \{20, 10, 3\}, \{25, 6, 3\}, \{30, -24, 5\}\}$



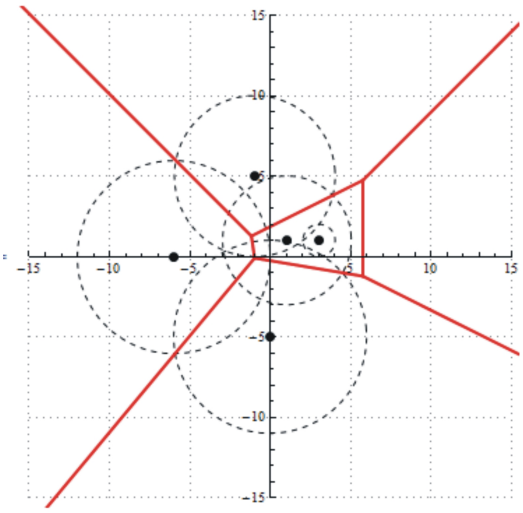
3. příklad $pole = \{-15, 15\}$

$vstup = \{\{-2, 3, 7\}, \{0, 0, 2\}, \{3, 6, 1\}, \{-3, -1, 2\}, \{-1, -2, 1\}\}$



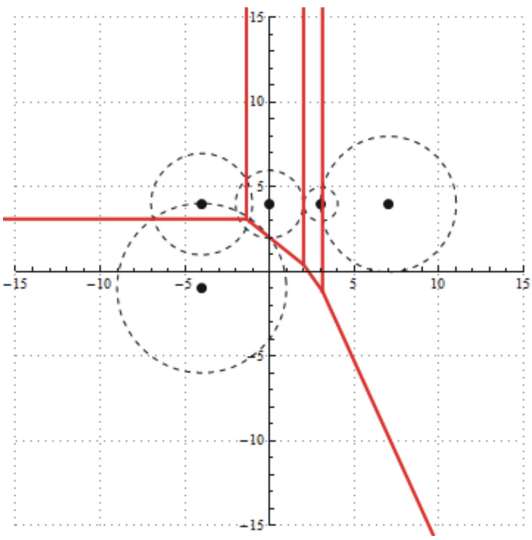
4. příklad $pole = \{-15, 15\}$

$$vstup = \{\{0, -5, 6\}, \{-6, 0, 6\}, \{-1, 5, 5\}, \{1, 1, 4\}, \{3, 1, 1\}\}$$



5. příklad $pole = \{-15, 15\}$

$$vstup = \{\{-4, 4, 3\}, \{-4, -1, 5\}, \{0, 4, 2\}, \{3, 4, 1\}, \{7, 4, 4\}\}$$



6 Shrnutí algoritmů a využití Voroneho diagramů

V následujících podkapitolách shrneme informace o algoritmech použitých v této práci a uvedeme jejich výhody a nevýhody. Dále uvedeme několik příkladů využití Voroneho diagramů v praxi.

6.1 Shrnutí algoritmů

V této práci jsme se zabývali třemi různými algoritmy, Inkrementálním algoritmem, algoritmem "Plane sweep"a Metodou rostoucích regionů. Uvedeme zhodnocení těchto algoritmů, jejich časovou složitost a výhody, či nevýhody jejich použití.

Voroneho diagramy mohou být konstruovány v nejlepším případě s časovou složitostí $O(n \log n)$. Algoritmy, které mají tuto složitost jsou pro konstrukci Voroneho diagramu optimální [2], [5].

Inkrementální algoritmus Jedná se o jeden z hojně využívaných algoritmů pro konstrukci Voroneho diagramů. Algoritmus v první řadě vytvoří Voroneho diagram pro dva, popř. tři, generující body a následně, v každém kroku, přidá generující bod, pro nějž modifikuje Voroneho diagram a začistí hrany, uvnitř nově vzniklé Voroneho buňky. Výhodou algoritmu, oproti jiným algoritmům, je, že je založen na přidávání generujících bodů. Pokud pro již vytvořený Voroneho diagram množiny generujících bodů budume chtít množinu vstupních bodů navýšit, tak nemusíme přepočítávat celý diagram, ale můžeme vycházet z již vytvořeného Voroneho diagramu. Složitost Inkrementálního algoritmu je $O(n^2)$, pro obecně zadanou množinu generujících bodů S [2], [5].

"Plane sweep"algoritmus Tento algoritmus využívá tzv. zametací přímky a beach line, která se skládá z parabolických oblouků. Průsečíky těchto oblouků vykreslují Voroneho hranu. Složitost tohoto algoritmu je $O(n \log n)$ a je optimální [1], [2], [5].

Metoda rostoucích regionů Tato metoda zde byla uvedena pouze pro její náročnost. Nicméně v kapitole 2.1.3 je uveden možný postup sestavení algoritmu. Postup algoritmu je zde uveden pouze na ukázku, neboť pro vlastní implementaci je metoda poměrně náročná a nevhodná.

6.2 Využití Voroneho diagramů

V následujícím textu uvedeme možná využití některých zobecněných Voroneho diagramů. Ty jsou obecně využívány v různých odvětvích vědy, např. v biologii, sociálních a ekonomických vědách, v počítačové grafice atd.

Voroneho diagram v L_1 metrice Voroneho diagramy se využívájí k řešení problémů ideálního rozmístění různých typů středisek, např. nákupní střediska, nemocnice atd. popřípadě zjištění, které středisko máme nejblíže domovu. Při řešení takového typu problému není vhodné využívat Obecné Voroneho diagramy, ty pracují s Eukleidovskou metrikou, tzn. vzdušnou vzdáleností dvou míst. Pro případ, kdy budeme ve městě s pravoúhlými ulicemi a budeme chtít zjistit, které středisko máme nejblíže, využijeme Voroneho diagram v L_1 metrice.

Voroneho diagram kružnic Bývá využíván v krystalografii, biologii a chemii. V biologii simuluje růst buněk, živočišných společenstev (*např. mořských korálů*) či krystalů [8]. Jak zde bylo zmíněno, Voroneho diagram kružnic se využívá v biologii, chemii a dalších vědách. Nicméně pro tyto vědy je výhodné Voroneho diagram kružnic zobecnit v prostoru, tedy Voroneho diagram sfér. Bližší informace o využití Voroneho diagramu sfér v chemii můžeme nalézt v [11].

7 Závěr

Cílem této práce bylo zpracovat teorii potřebnou pro studium zobecněných Voroneho diagramů, pro vybrané typy zobecnění uvést vhodné algoritmy, popř. modifikaci známých algoritmů a ve vhodném software provést implementaci vybraného algoritmu.

V jednotlivých kapitolách se diplomová práce věnovala různým typům Voroneho diagramů a algoritmům pro jejich konstrukci. Tato práce vychází především z anglicky psaných článků, které jsou vyjmenovány v seznamu literatury.

Úvodní kapitola se stručně věnovala Obecnému Voroneho diagramu a třem algoritmům pro jeho konstrukci. Těmito algoritmy jsou Inkrementální algoritmus, algoritmus "Plane sweep"a metoda rostoucích regionů. Jedná se spíše o připomenutí teorie potřebné pro Voroneho diagramy a jejich algoritmy, neboť podrobnější informace lze nalézt v bakalářské práci [4], na kterou tato práce navazuje. Vyjma metody rostoucích regionů, která zde byla popsána zcela nově.

V následujících kapitolách byly podrobně popsány tři různé typy zobecněných Voroneho diagramů a to Voroneho diagram v L_1 metrice, Voroneho diagram kružnic a Laguerre Voroneho diagram. V každé z těchto kapitol byla zpracována teorie k danému diagramu a byl udeveden minimálně jeden algoritmus, který byl vhodně modifikován pro jeho použití na daném zobecněném Voroneho diagramu.

Pro každý z výše uvedených zobecněných Voroneho diagramů byl ve vhodném software zpracován jeden algoritmus. Pro Voroneho diagram v L_1 metrice a pro Voroneho diagram kružnic byla řešena metoda rostoucích regionů v programu Geogebra. Tato metoda byla uvedena spíše jako názorná, tomu odpovídá i zvolený software.

Pro Laguerre Voroneho diagram byl implementován Inkrementální algoritmus v programu Mathematica 7. V počátcích implementace byl algoritmus řešen tak, aby fungoval i ve specialních případech, jako např. při kolinearitě středů kružnic. Postupně bylo od tohoto plánu upuštěno s tím, že pokud bude dostatek času mohou se některé specialní připady v programu dořešit. Z tohoto důvodu algoritmus zvládá vykreslit některé konkrétní příklady specialních případů, ale ne pro všechny možné kombinace umístění generujících kružnic daného specialního případu.

Všechny programy spolu s manuálem jsou na přiloženém CD.

Tato práce se věnovala pouze třem typům zobecněných Voroneho diagramů. Což je velmi úzký výběr. V další práci by bylo možno se věnovat jiným zajímavým typům zobecněných Voroneho diagramů, jako například Krystalický Voroneho diagram, Voroneho diagram úseček, popř. Voroneho diagram pro smíšenou vstupní množinu (*kružnice, úsečky, mnohoúhelníky*) atd.

8 Přehled použitého značení

S	...	množina generujících prvků
l	...	zametací přímka
l^+	...	polorovina nad zametací přímkou
l^-	...	polorovina pod zametací přímkou
Vor(S)	...	Voroneho diagram množiny S
P_i	...	generující body
e_i	...	Voroneho hrana
$\nu(P_i)$	...	Voroneho buňka
$d_e(P, Q)$	...	Eukleidovská vzdálenost
b	...	Beach line
β_i	...	parabolické oblouky na Beach line
k_i	...	generující kružnice
F	...	fronta událostí
k_a	...	Apolloniova kružnice
E_h	...	horní extrém generující kružnice
E_d	...	dolní extrém generující kružnice
E_a	...	významný bod Apolloniovy kružnice
V	...	Voroneho vrchol
$B(P_i, P_j)$	...	osa úsečky
$h(P_i, P_j)$	...	otevřená polorovina pro bod P_i
$h'(P_i, P_j)$	...	uzavřená polorovina pro bod P_i
$B_h(k_i, k_j)$	...	hyperbola tvořící osu mezi dvěmi kružnicemi
T	...	dotykový bod
$B_L(p_i, p_j)$	...	osa úsečky v Laguerre geometrii
$h_L(p_j, p_j)$	...	Laguerre otevřená polorovina kružnice p_i
$\nu_L(p_i)$	...	Laguerre Voroneho buňka kružnice p_i

9 Seznam použité literatury

Reference

[1] Li Jin; Donguk Kim; Lisen Mu; Deok-Soo Kim; Shi-Min Hu: A sweepline algorithm for Euclidean Voronoi diagram of circles. Computer-Aided Design 38. 2006

[2] de Berg, M.; van Kreveld, M.; Overmars, M.; Schwarzkopf, O.: Computation Geometry. Algorithms and Applications. Berlin. Springer Verlag 1997. ISBN 3-540-65620-0

[3] Sugihara, K.: Voronoi Diagrams, Department of Mathematical Engineering and Information Physics, University of Tokyo, Hongo, Bunkyo-ku, Tokyo 113-8656, Japan

[4] Homrová, J.: Voroneho diagramy. bakalářská práce. ZČU. Plzeň. 2008

[5] Aurenhammer, F.; Klein, R.: Voronoi diagrams. Technische Universität Graz a Fern Universität Hagen

[6] Patáková, E.: Apolloniovy úlohy. diplomová práce. ZČU. Plzeň. 2005

[7] Liška, P.: Apolloniova úloha. bakalářská práce. Masarykova univerzita. Brno. 2007

[8] Bayer, T.: Voronoi diagram. součást přednášky. PřF UK. Praha. 2010

[9] Sugihara, K.: Laguerre Voronoi diagram on the sphere. Jurnal for Geometry and Graphics. Volume 6. p.69-81. 2002

[10] Hiroshi Imai; Masao Iri; Kazuo Murota: Voronoi diagram in the Laguerre geometry and its applications. SIAM J. COMPUT. Volume 14. p.93-105. 1985

[11] Beneš, P.: Voroného diagramy v molekulární chemii. diplomová práce. Masarykova univerzita. Brno. 2006

[12] Hodorkovsky, D.: 2-Point Site Voronoi Diagrams. Researh Thesis. Haifa. 2005

[13] Anton, F.; Bereg, S.: The Fitting Line Problem in the Laguerre Geometry. 16th Canadian Conference on Computational Geometry. Montreal. 2004