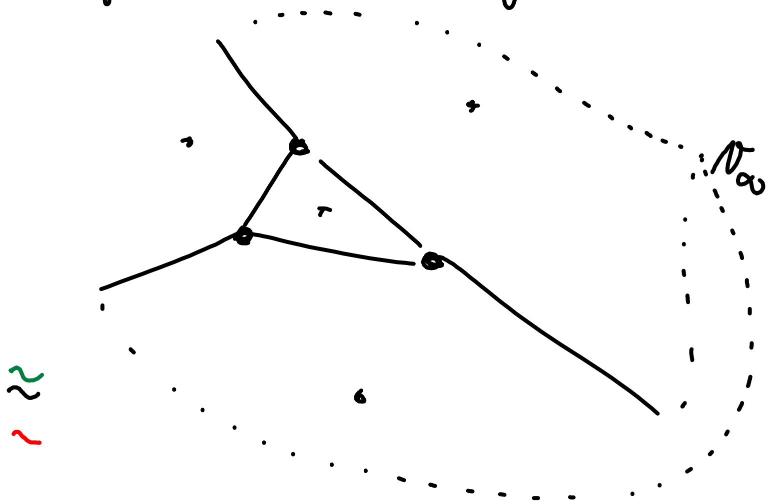
DIAGRAMY Z VORONOIA (1)

$P = \{p_1, p_2, \dots p_n\}$ množina bodů v rovině

Chceme rovinu rozdělit na oblasti $V(p_i)$ bodů, které mají nejblíže k p_i

Diagram Voronoia je rovinný graf o n oblastech, n_v vrcholech a n_e hranách



Přidáním v_∞ získáme souvislý rovinný graf, pro který platí Eulerova věta

$$(n_v+1) - n_e + n = 2$$

$$4 - 6 + 4 = 2$$

2

Z každého vrcholu vychází aspoň 3 hrany

st. vrcholu ≥ 3

Současně

$$\sum st = 2n_e$$

$$\underline{3(n_v+1)} \leq \sum st = \underline{2n_e}$$

$$2 = (n_v+1) - n_e + n \leq (n_v+1) - \frac{3}{2}(n_v+1) + n = -\frac{1}{2}n_v - \frac{1}{2} + n$$

Úpravou

$$\frac{1}{2}n_v \leq n - \frac{1}{2} - 2$$

$$\boxed{n_v \leq 2n-5}$$

3

$$\underline{2} = (n_v+1) - n_e + n \le \frac{2}{3} n_e - n_e + n = \underline{-\frac{1}{3} n_e + n}$$

$$\frac{1}{3} n_e \le n - 2$$

$$n_e \le 3n - 6$$

<u>Lemma</u> Diagram Voronoia pro n bodů má nejvýše $3n-6$ hran a $2n-5$ uzlů.

<u>Označení</u> $C_P(q)$ = kruh se středem q a maximálním poloměrem takovým, že uvnitř neleží bod z množiny P

$P = \{p_1, p_2, \dots, p_n\}$

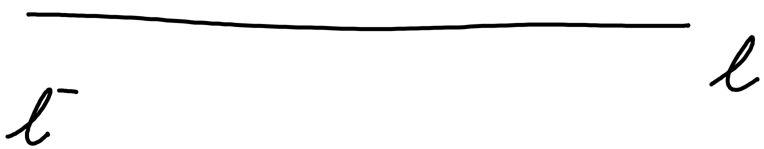
4

Věta. q leží na hraně diagramu Voronoia právě když $C_P(q)$ má na hranici aspoň dva body množiny P

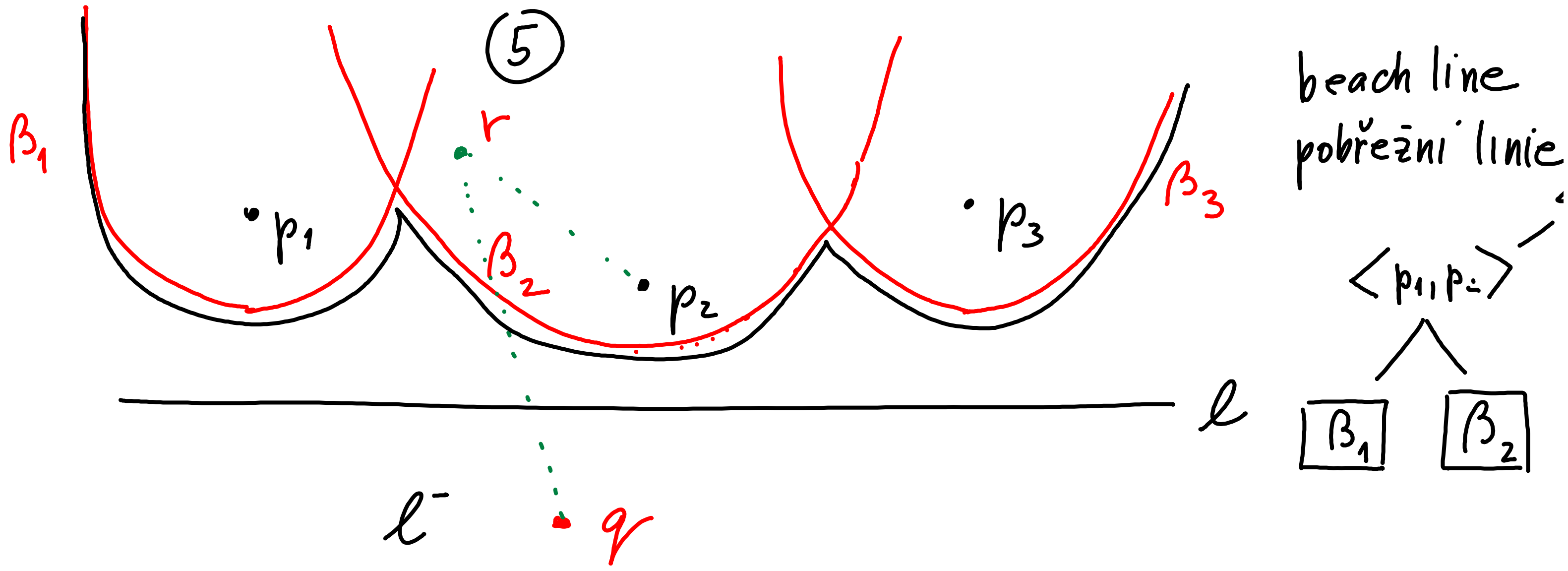
q je vrcholem diagramu Voronoia právě když $C_P(q)$ má na hranici aspoň 3 body

Obrázky

Algoritmus – metodou zametací přímky



Úloha může být vyřeše
v te oblasti nad ℓ, kde ne
vliv body z dolní polorovi
ℓ^-.



Geometrické místo bodů, které mají stejnou vzdálenost od přímky ℓ a bodu p_i, $(i=1,2,3)$, je parabola.

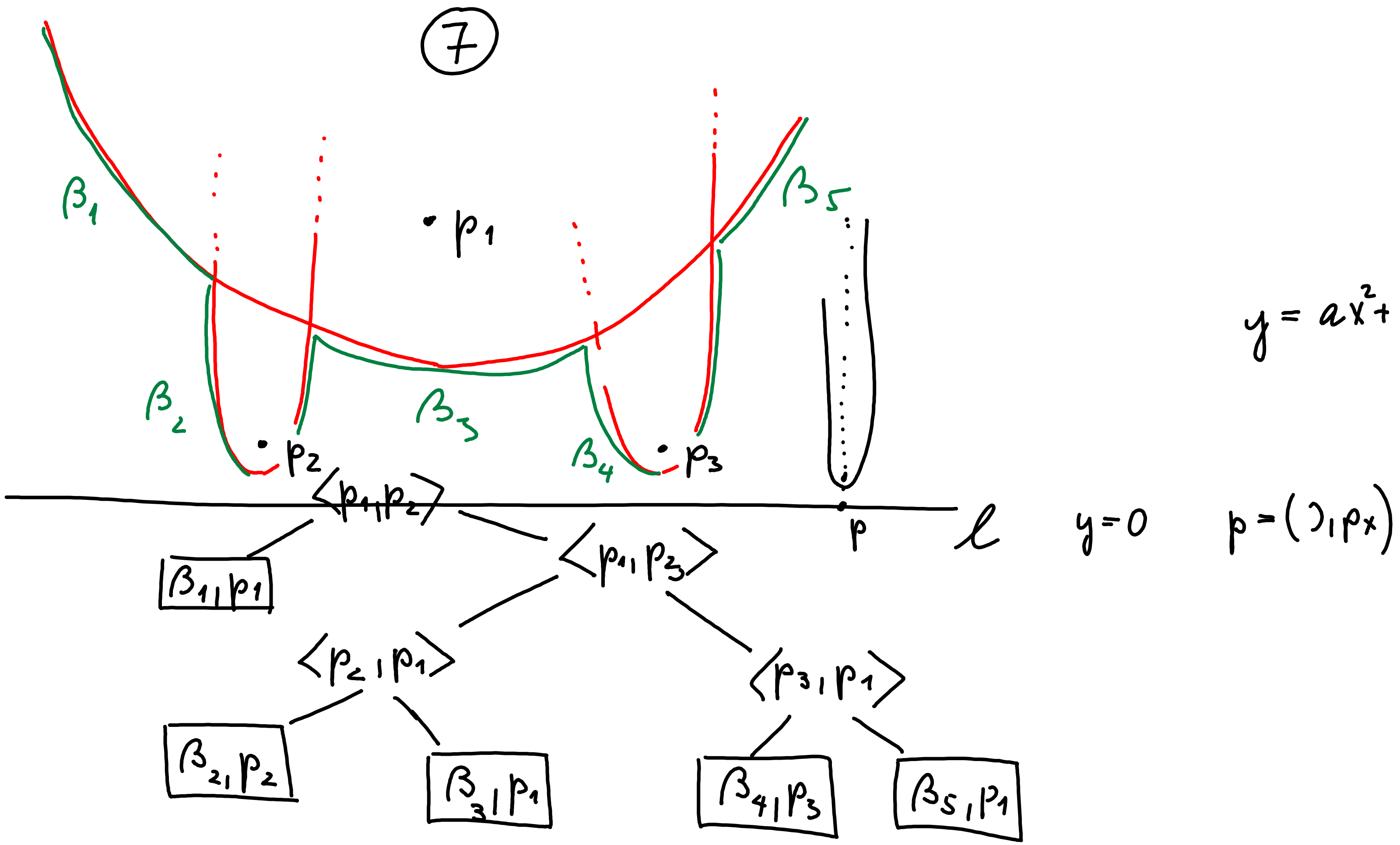
Body nad oblouky parabol mají blíže k p_1, nebo p_2 nebo p_3 než k ℓ, tj. než k nějakému bodu q v ℓ^-. Bod r nemůže být v oblasti Voronoi bodu pod ℓ^-.

6

Pobřežní linie je křivka složena z oblouků parabol, které jsou určeny jako geom. místo bodů stejně vzdálených od bodu $p \in P$, který leží nad zametací přímkou ℓ, a přímky ℓ

2 struktury

T binární vyvážený strom – určuje pořadí oblouků parabol v pobřežní linii. V listech jsou oblouky parabol společně s bodem, který je určuje. V uzlech jsou „rozhraní" mezi parabolami.


7
β_1
p_1
β_5
$y = ax^2 +$
β_2
β_3
β_4
p_2
p_3
$\langle p_1, p_2 \rangle$
p
ℓ
$y=0$
$p = (\,, p_x)$
$\langle p_1, p_3 \rangle$
β_1, p_1
$\langle p_2, p_1 \rangle$
$\langle p_3, p_1 \rangle$
β_2, p_2
β_3, p_1
β_4, p_3
β_5, p_1

8

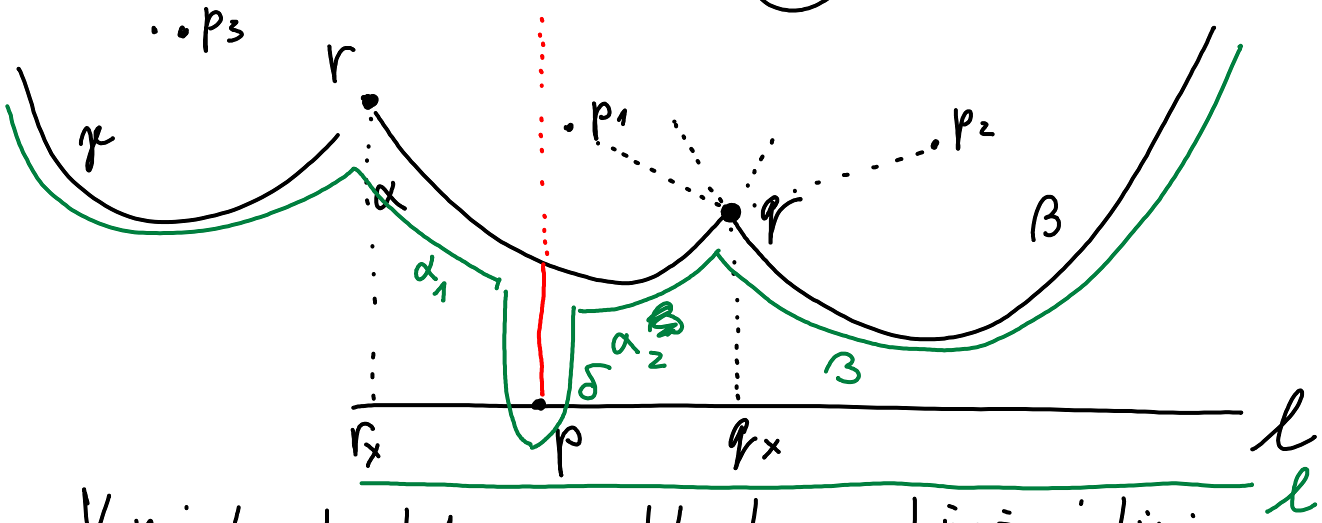
Fronta událostí je tvořena body takovými, že při průchodu zametací přímky nad nimi vzniká oblouk paraboly nebo zaniká.

Body prvého typu se nazývají site events (místní události).
druhého typu se nazývají circle events (kruhové události)

Je lehké zpozorovat, že každý bod $p \in P$ je místní událostí.

$\boxed{\text{Lemma}}$: Nové oblouky v pobřežní linii vznikají pouze při přechodu zametací přímky přes bod $p \in P$. (Žádné další místní události nejsou)

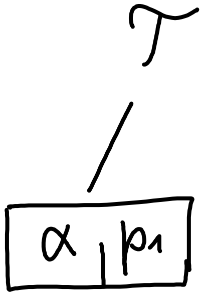
(9)



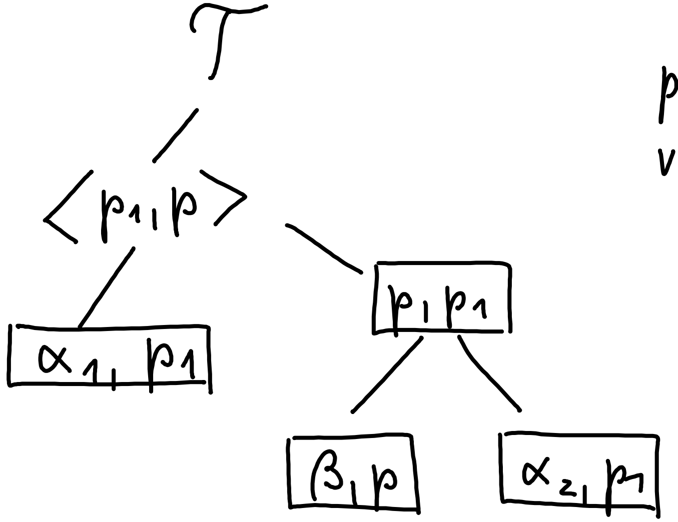
V místech, kde se oblouky pobřežní linie střetávají, jsou body ležící na hranách diagramu Voronoia.

Vždy umíme spočítat, pod kterým obloukem bod p leží

10



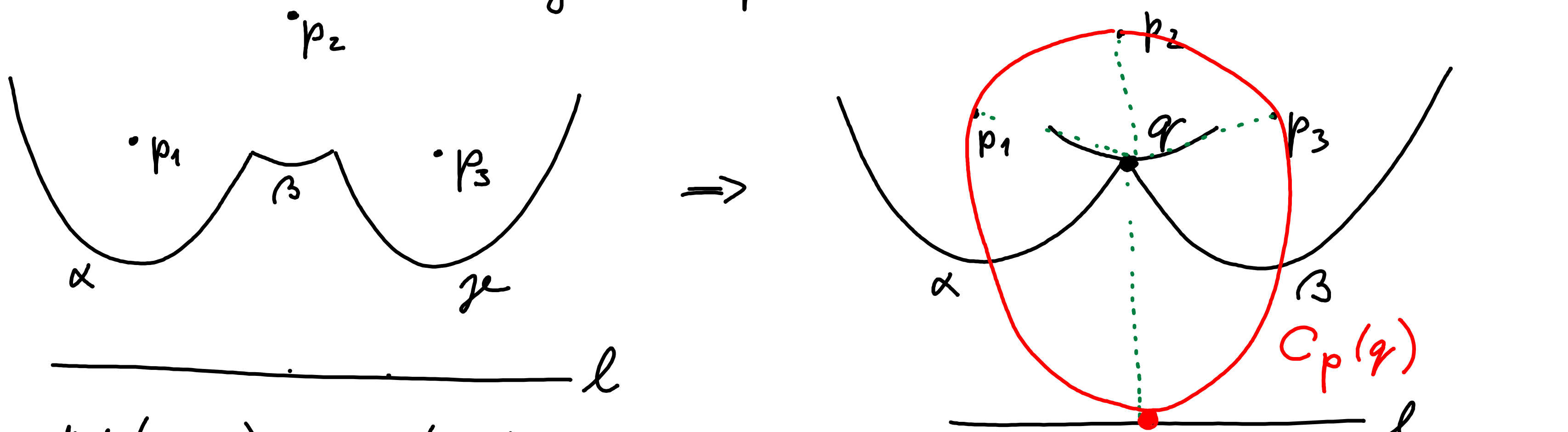
po průchodu přímky
l bodem p



provedem[e]
vyvážení

11

Zánik oblouku paraboly v pobřežní linii



kruhová událost r

$dist(q, p_1) = dist(q, \ell) = dist(q, p_2) = dist(q, p_3)$

q je střem kruhu $C_p(q)$, tudíž q je vrcholem diagramu Voronoia

Vjaké poloze musí být přímka ℓ, aby tato situace nastala.

№12

Kruhová událost, kde zaniká oblouk β příslušný p_2 je bod r ležící na kružnici určené body p_1, p_2, p_3 (vidící body 3 po sobě jdoucích oblouků paraboly) s minimální y-ovou souřadnicí.

Lemma: Toto je jediný možný způsob zániku oblouku paraboly

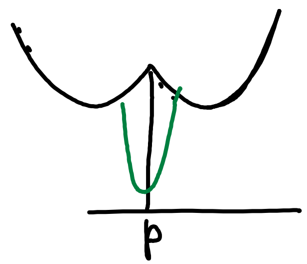
Jak se mění fronta událostí. Na začátku jsou v ní všechny body z množiny P (místní události) V průběhu algoritmu přidáváme a opět vyndaváme kruhové události podle aktuálních pořadí oblouků v pobřežní linii.

13

Kolik nejvíce oblouků může být v pobřežní linii?

Induktivně. 1 bod v P ... ... 1 oblouk

Při vzniku nového oblouku se některý ze stávajících rozdělí na 2. Tedy přibudou nejvýše 2 oblouky



Proto počet oblouků v pobřežní linii je nejvýše $\boxed{2n-1}$

Důležité pro odvození časové náročnosti

Celková čas. náročnost je $O(n \log n)$, paměťová $O(n)$.

14

Algoritmus str 30, 31, 32.