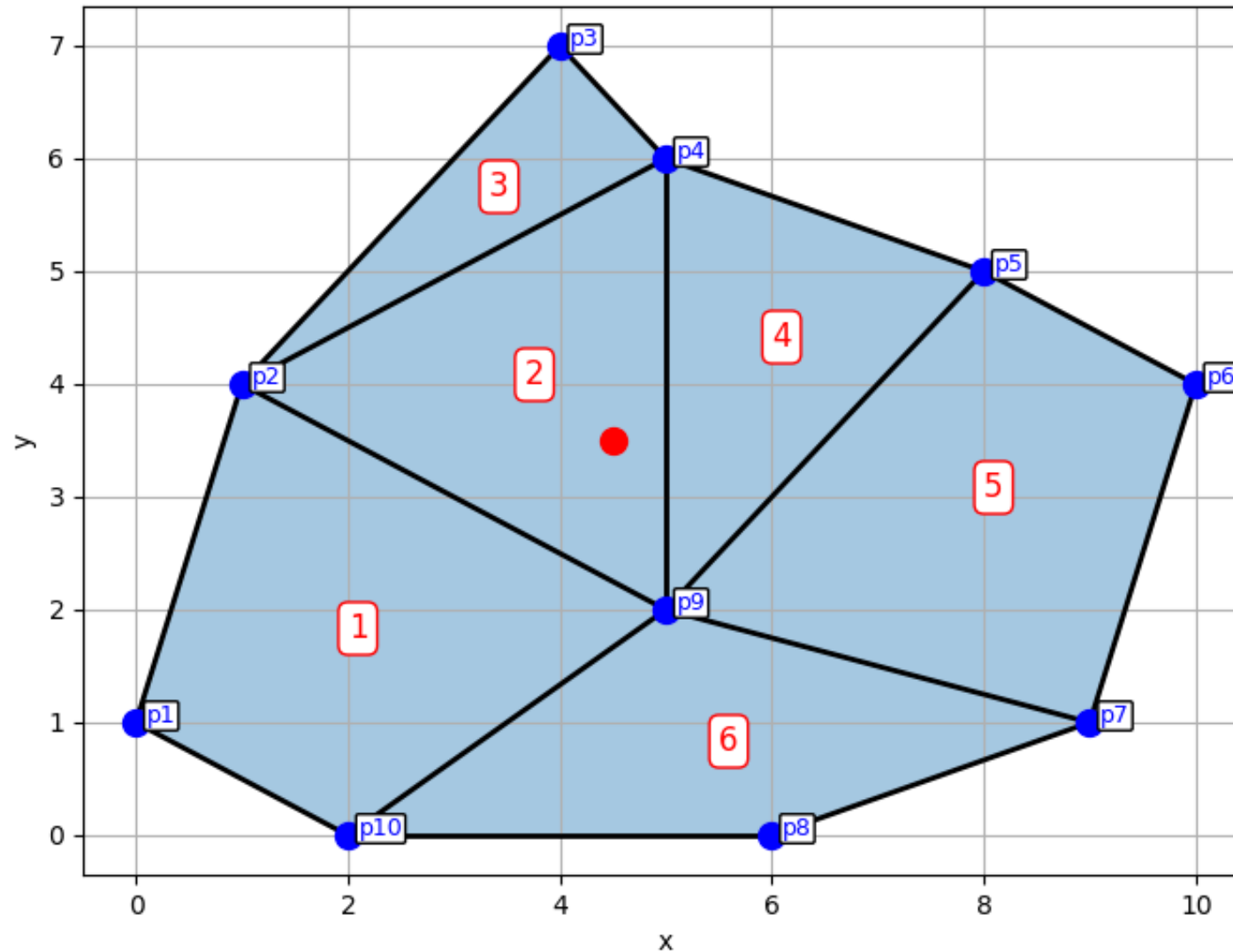
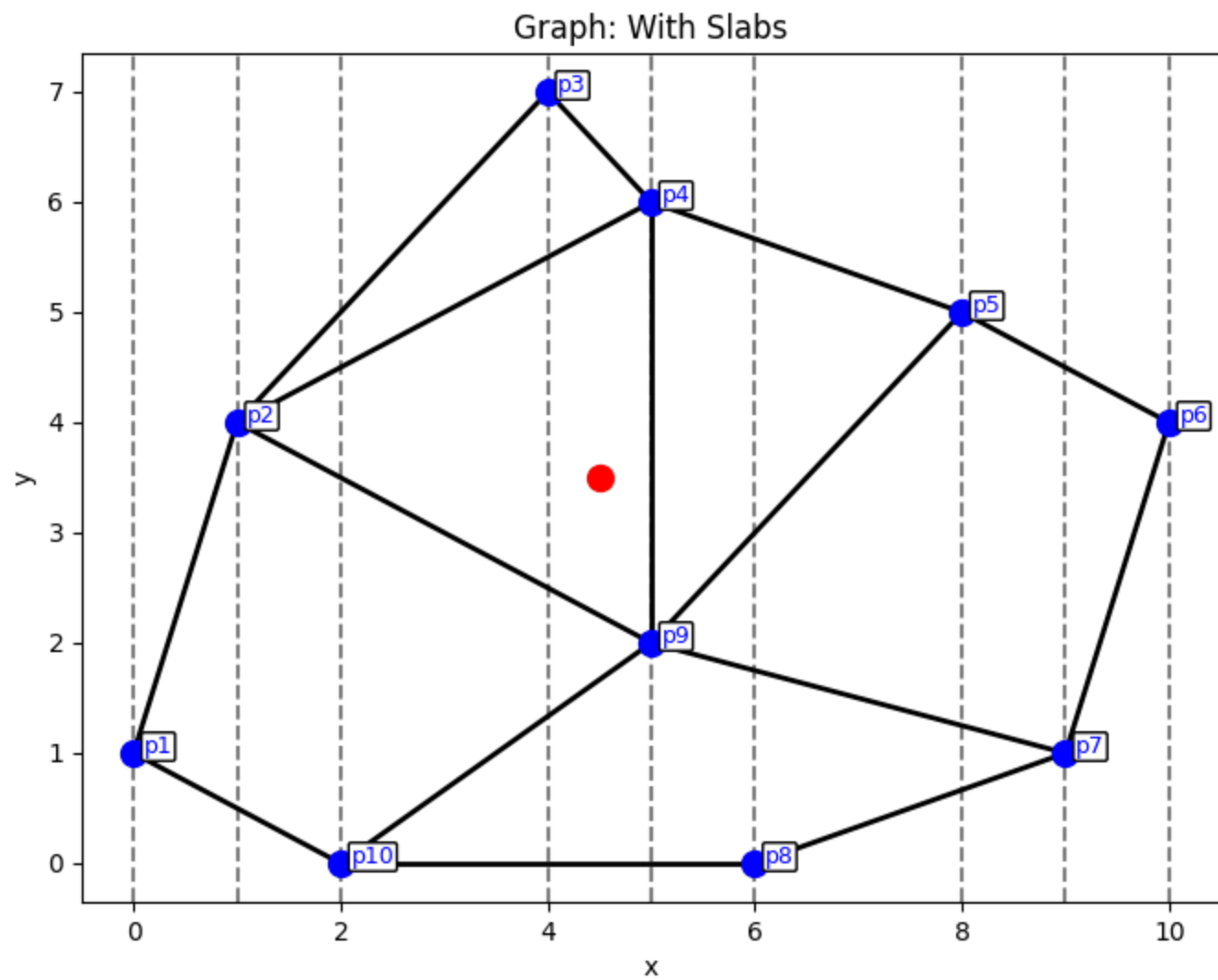


Pradiniai duomenys

Starting graph: Edges, Vertices, Position Point and Regions



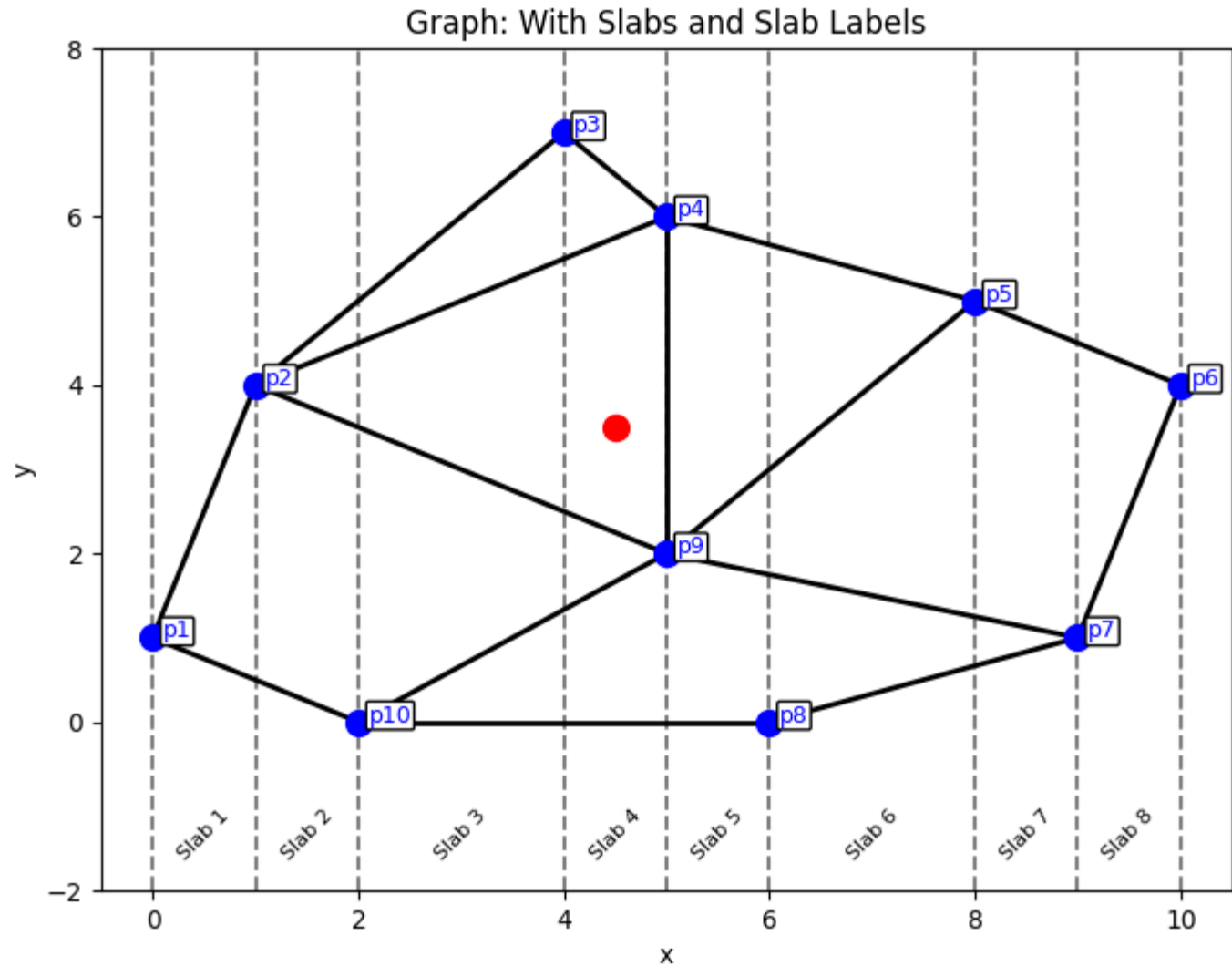
Nubraižome vertikalias linijas kiekvienai viršūnei, grafas pasidalina į „Slabs“ - plokštės

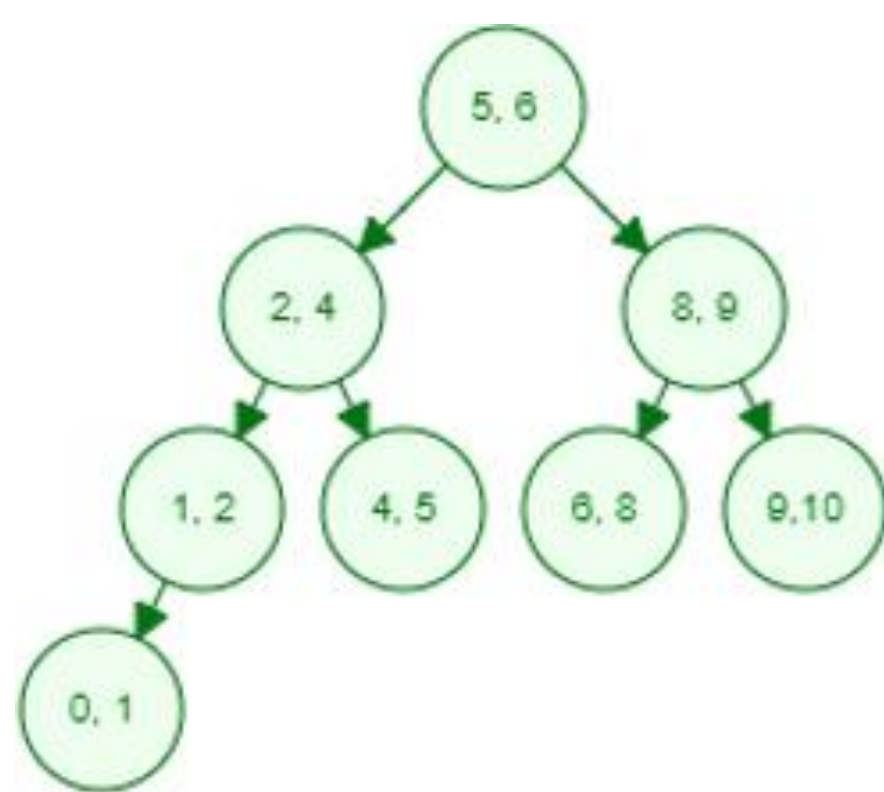


Kiekvienos plokštės viduje padalijimo kraštinės elgiausi ypatingai: kadangi padalijimo viršūnės plokštėje neegzistuoja (tik tarp plokščių), kraštai arba visiškai kerta plokštę, arba neegzistuoja joje.

Taip sudarytų plokščių kraštai bus išdėstyti iš viršaus į apačią, nes jie nekerta vienas kito. Tai leidžia laikyti kraštus, kertančius kiekvieną plokštę, surūšiuota tvarka.

- Slab 1: From $x = 0$ to $x = 1$
- Slab 2: From $x = 1$ to $x = 2$
- Slab 3: From $x = 2$ to $x = 4$
- Slab 4: From $x = 4$ to $x = 5$
- Slab 5: From $x = 5$ to $x = 6$
- Slab 6: From $x = 6$ to $x = 8$
- Slab 7: From $x = 8$ to $x = 9$
- Slab 8: From $x = 9$ to $x = 10$





X-X

Slab 1: 0, 1

Slab 2: 1, 2

Slab 3: 2, 4

Slab 4: 4, 5

Slab 5: 5, 6

Slab 6: 6, 8

Slab 7: 8, 9

Slab 8: 9, 10

Mūsu taškas
 X, Y - (4.5, 3.5)

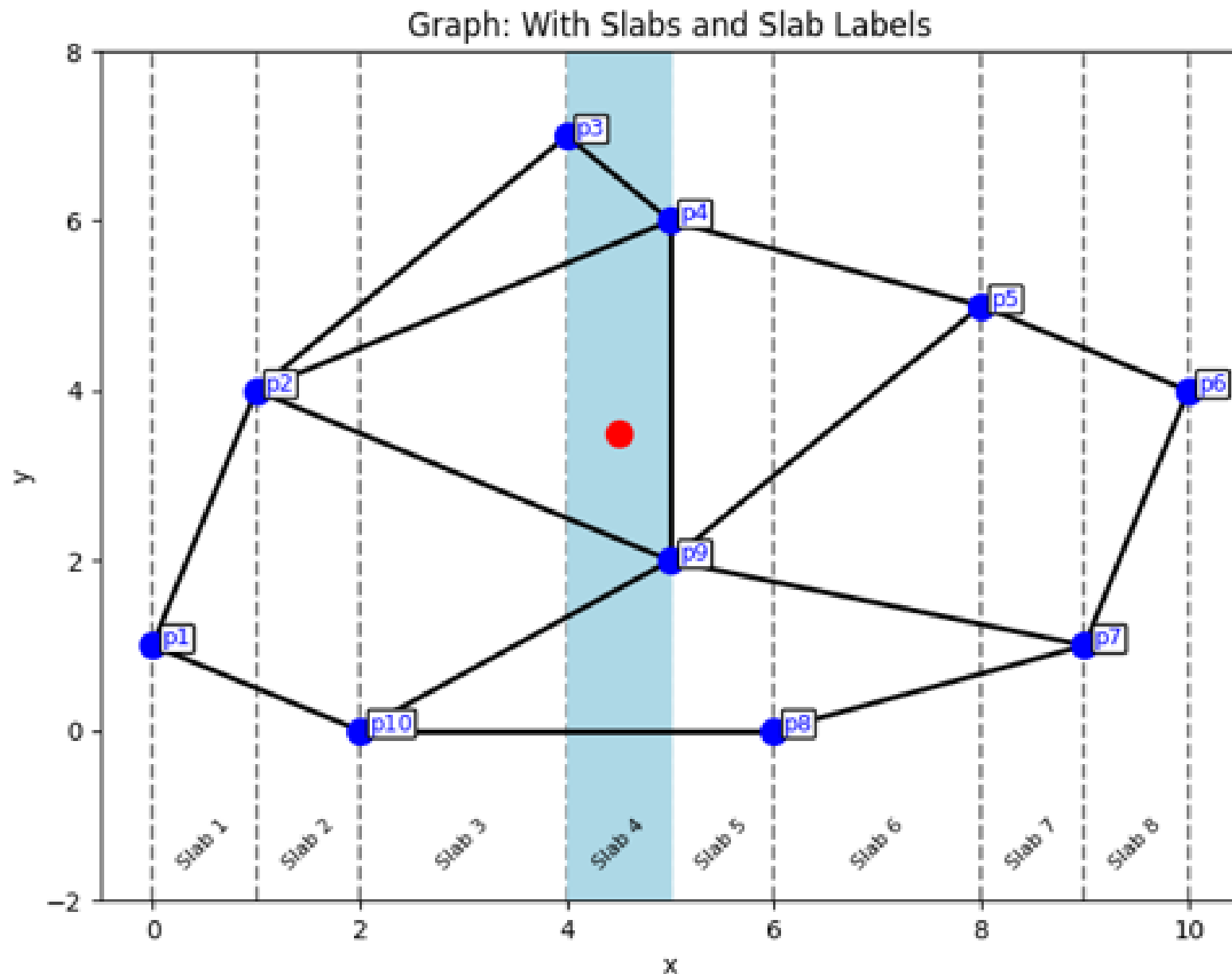
4.5

$[5; 6] > 4.5$

$[2; 4] < 4.5$

$[4; 5] \in 4.5$ – Slab 4

$O(\log n)$



Identifikave kurioje plokštėje esame galime sudaryti binary search tree pagal y ašį, su regionais. Atlikę paiešką pagal y sužinosime kuriame regione esame.

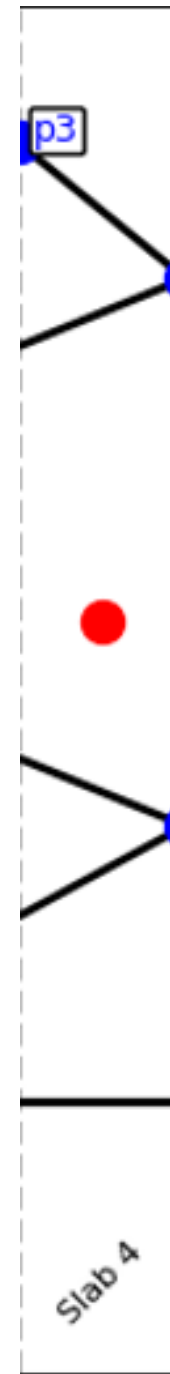
Paieška pagal x surasti plokštę “Slab” - $O(\log n)$

Paieška pagal y surasti regioną - $O(\log n)$

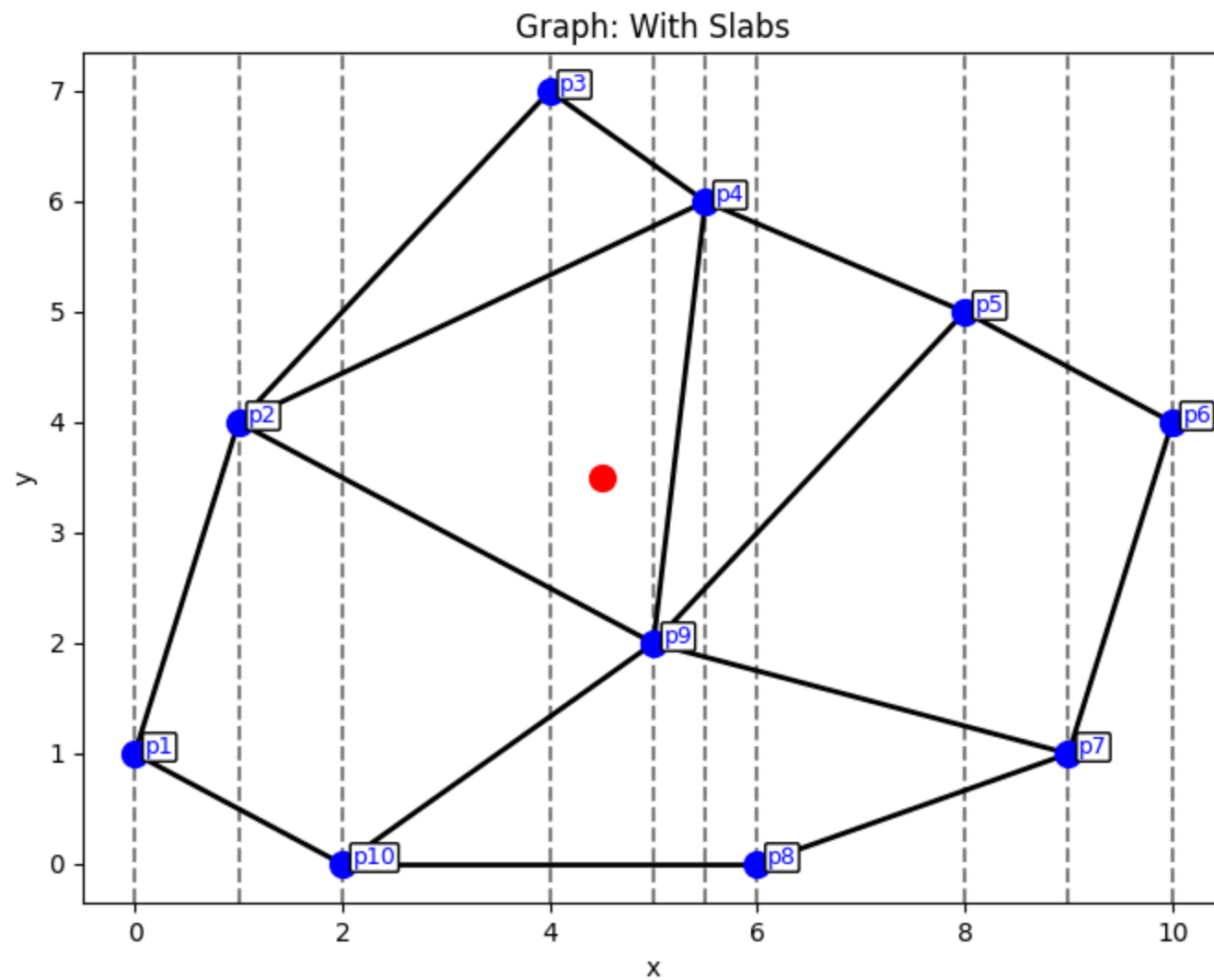
Tokio algoritmo sudėtingumas $O(\log n)$

Bet sudaryti tokius binary search trees užtruktų $O(n^2 \log n)$

Bei duomenų struktūra užimtų $\Theta(n^2)$



Pastebējimas
kiekviena plokštēs dalis yra trapecija.



Galime supaprastinti struktūrą. Vertikalias linijas braižant iki pirmos kraštinės arba išorinio stačiakampio R.

Gautos trapecijos visada turės:

Viršaus kraštinę – $\text{top}(\Delta)$

Apačios kraštinę - $\text{bottom}(\Delta)$

Kairiausią tašką - $\text{leftp}(\Delta)$

Dešiniausią tašką - $\text{rightp}(\Delta)$

Turime 5 dėl šonų taškų apžvelkime kairės pusės atvejį:

- Viršus ir apačia susijungią į tašką $\text{leftp}(\Delta)$
- Viršus kairiausia kraštinė yra taškas $\text{leftp}(\Delta)$
- Apačios kairiausia kraštinė yra taškas $\text{leftp}(\Delta)$
- Taškas $\text{leftp}(\Delta)$ yra ant kairiausios kraštinės
- Taško kairiausia kraštinė yra ant išorinio stačiakampio R.

Dešinei pusei $\text{rightp}(\Delta)$ atvejai yra simetriški.

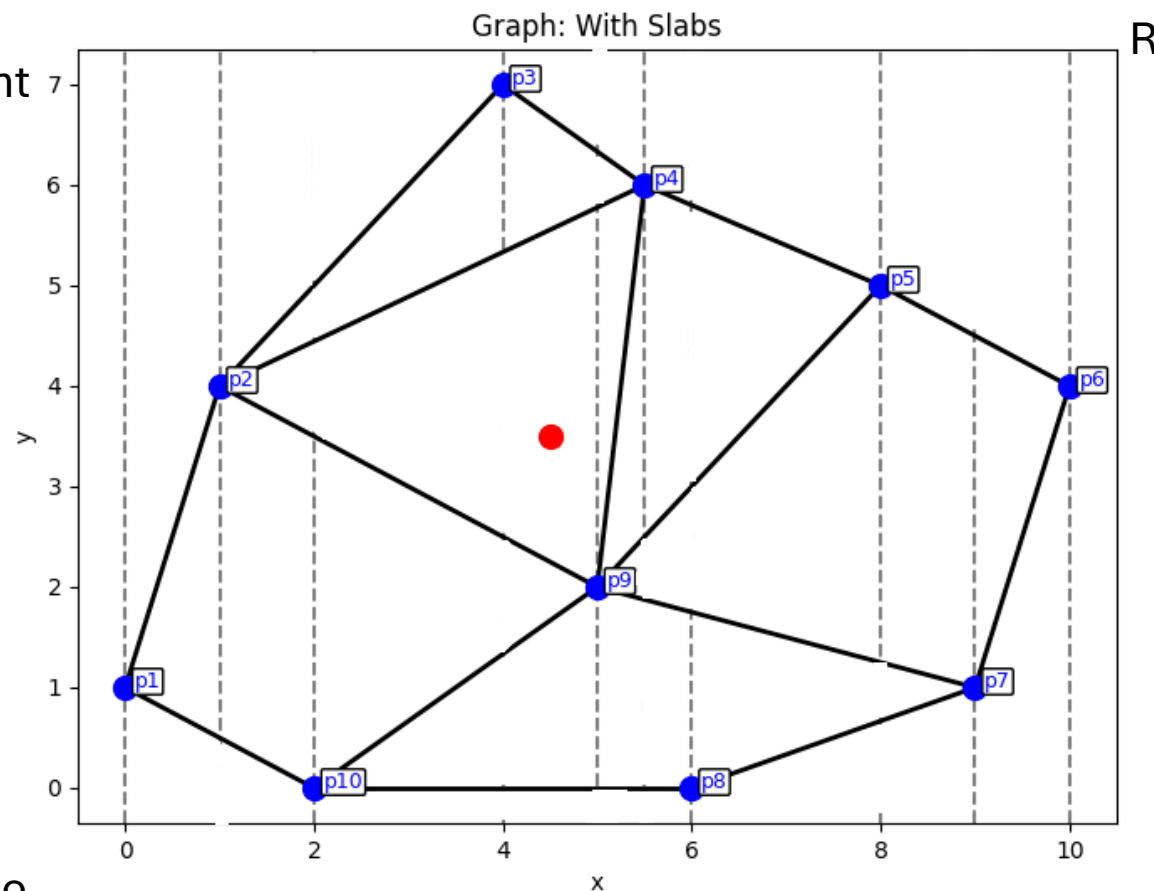
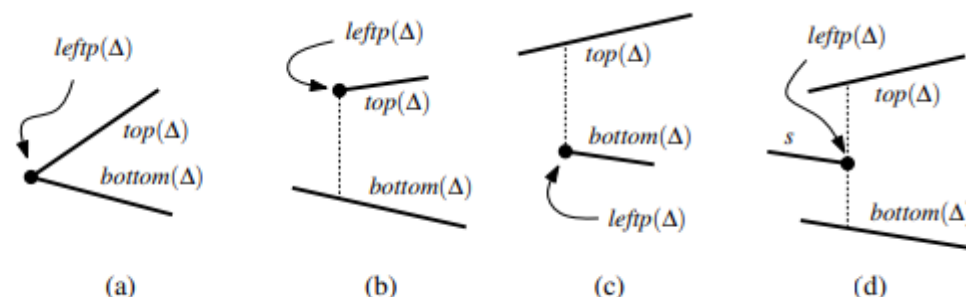


Figure 6.4
Four of the five cases for the left edge
of trapezoid Δ



Naujos struktūros vietos sudėtingumas

Ką turime:

1.Trapecijų skaičius: Poligone yra daugiausiai $3n+1$ trapecijų.

Įrodymas knygoje.

2.Kiekviena trapecija saugo:

1. Nuorodas į jos **viršutinę** ir **apatinę** atkarpas.
2. Nuorodas į jos **kairįjį** ir **dešinįjį** taškus.
3. Nuorodas į daugiausiai **keturias kaimynines trapecijas** (gretimas trapecijas).

* Nesaugoma viršūnės ar kraštinės: Saugomos tik trapecijos jų formos(kraštinės) bei kaimyninės trapecijos.

Vietos kiekis vienai trapecijai:

• Viršutinės, apatinės, kairiųjų ir dešiniųjų ribas, bei keturias kaimynines trapecijas, tai sudaro $4+4=8$ nuorodų vienai trapecijai.

* Kiekviena nuoroda yra pastovaus dydžio, todėl vietos kiekis vienai trapecijai yra pastovus, nepriklauso nuo n .

Bendra vieta:

Trapecijų skaičius yra $O(n)$, nes jų daugiausia turėsime - $3n+1$

Kiekvienai trapecijai reikia $O(1)$ vietos saugoti jos savybėms ir nuorodoms.

$$\Theta(n) * \Theta(1) = \Theta(n)$$

Pagerinome vietos sudėtingumą iš polinominio sudėtingumo $\Theta(n^2)$ iki tiesinio $\Theta(n)$