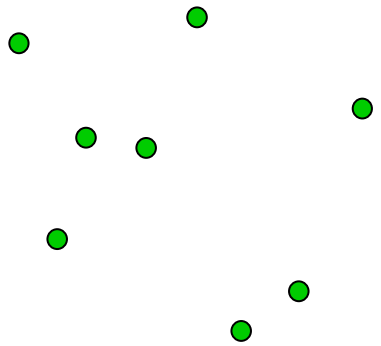
Wielościany Voronoi
i
triangulacja Delaunay'a

Gieorgij Fieodosjewicz Voronoi (Woronoj)
(1868-1908)

Borys Mikołajewicz Delaunay (*Delone*)
(1890-1980)

Wielościany Voronoi oraz triangulacja Delaunay'a zostaną przedstawione w płaszczyźnie R^2, choć można je uogólnić do dowolnej przestrzeni R^k

Niech S będzie zbiorem skończonym rozłącznych punktów $P_1,P_2,...,P_n$ leżących na płaszczyźnie R^2.
O punktach tych zakłada się, że nie wszystkie są współliniowe.
Zbiór takich punktów nazywany jest dalej ***chmurą punktów***.

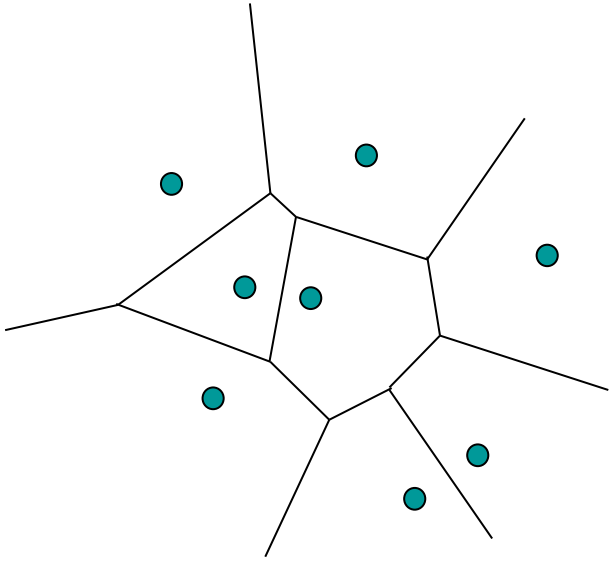


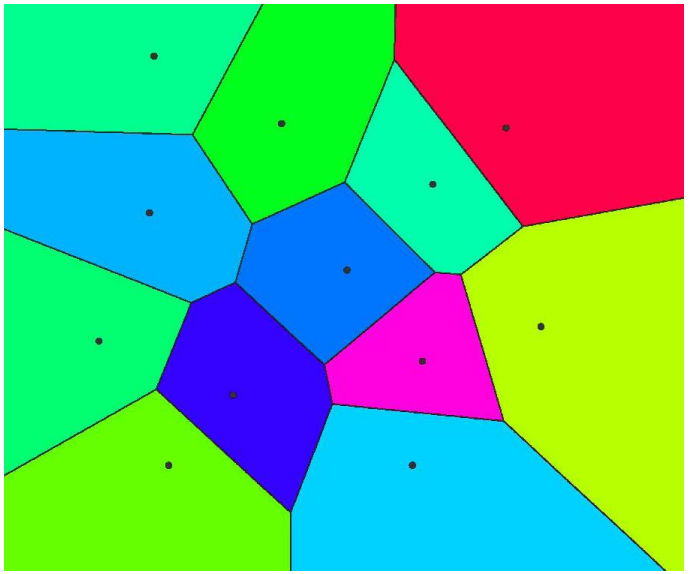
Diagramy Voronoi

Dla chmury n punktów $S = \{ P_1, P_2, \ldots, P_n \}$
definiuje się ***wieloboki Voronoi*** jako:

$$V_i = \{ x \in \mathbf{R}^2 : d(x,P_i) < d(x,P_j) \ \forall\, j \neq i \ \ j = 1,\ldots,n \}$$

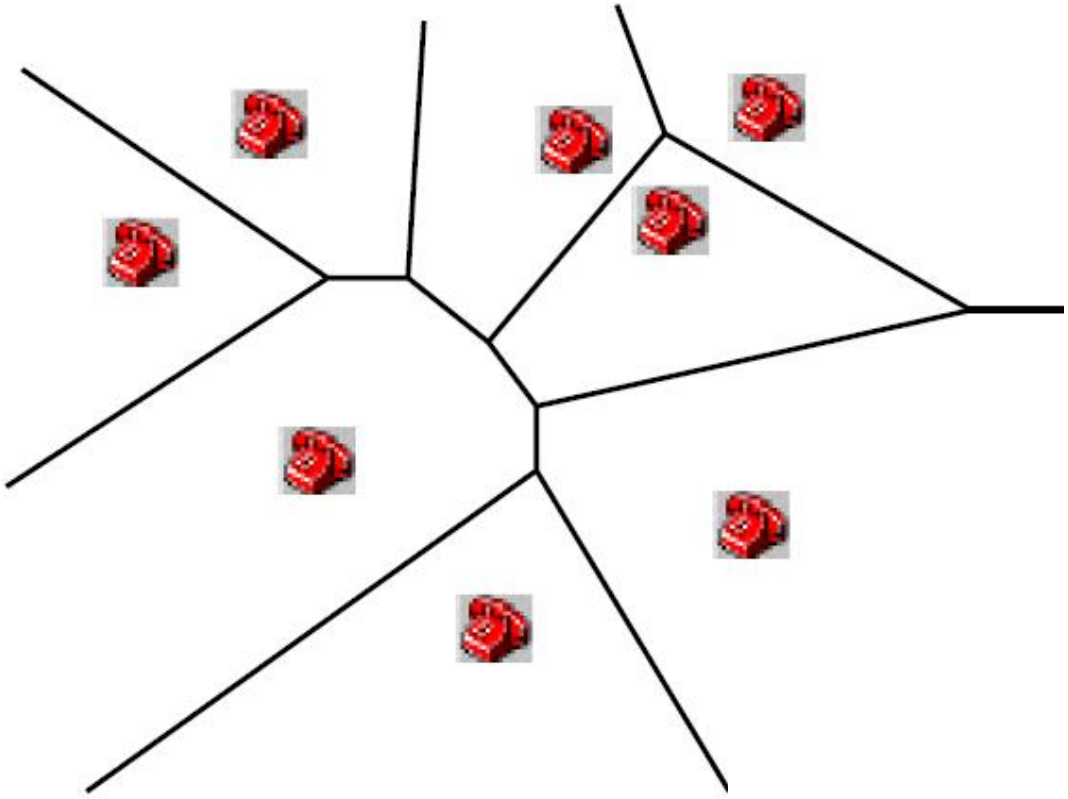
gdzie: $d(.,.)$ – oznacza odległość.





Zastosowania wieloboków Voronoi

Problem urzędu pocztowego – znaleźć najbliższy urząd pocztowy



Zastosowania wieloboków Voronoi

Planowanie przestrzenne:

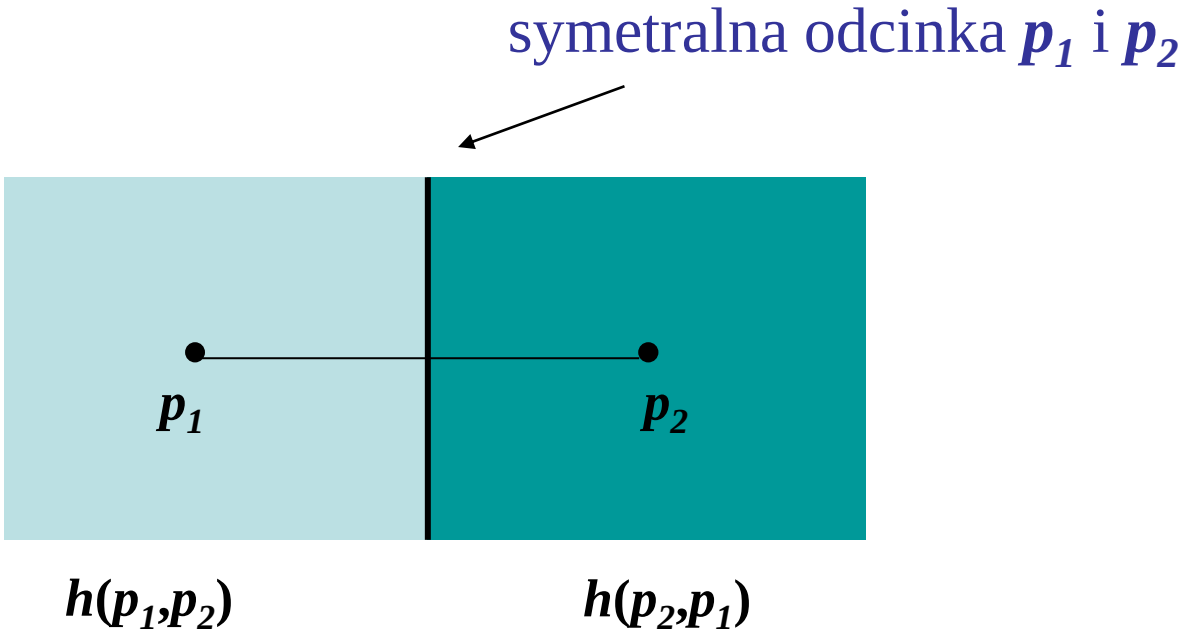
sieć centrów zbytu i usług: np. w celu oszacowania liczby klientów
– wyznaczyć takie rejony, dla których odległość do danego centrum jest mniejsza niż do każdego innego.

funkcja położenia: np. koncern chce otworzyć kolejną stację.
W celu zminimalizowania ingerencji w obszar istniejącej stacji, umiejscowić stację najdalej jak się da od najbliższej istniejącej stacji, tzn.
– na wierzchołku diagramu Voronoi
(Wierzchołek Voronoi określa środek największego pustego koła pomiędzy punktami.)

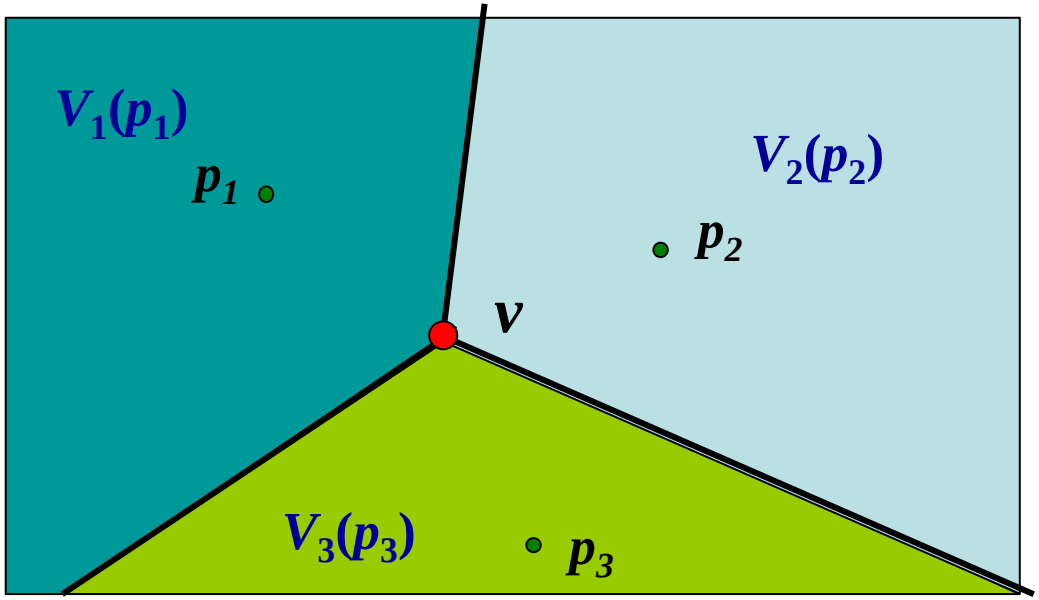
planowanie ścieżek: zadane środki przeszkód, które chcemy ominąć
– krawędzie diagramu Voronoi definiują możliwe ścieżki.

Inne – chemia, archeologia, leśnictwo, …

Dla dwóch punktów:



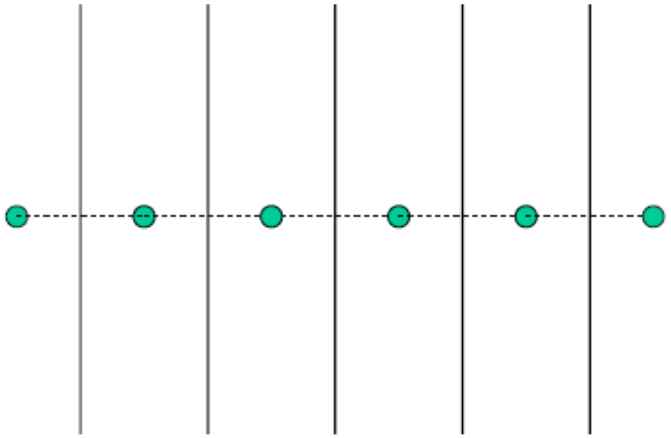
Dla trzech punktów:



$\boldsymbol{V}$ – komórka (wielobok, obszar) Voronoi

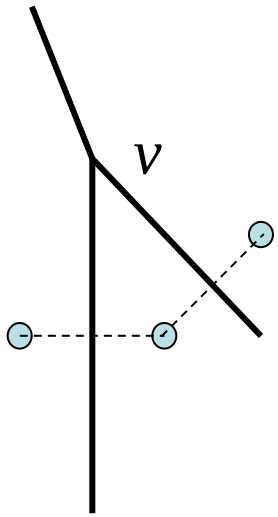
$\boldsymbol{v}$ – wierzchołek wieloboku

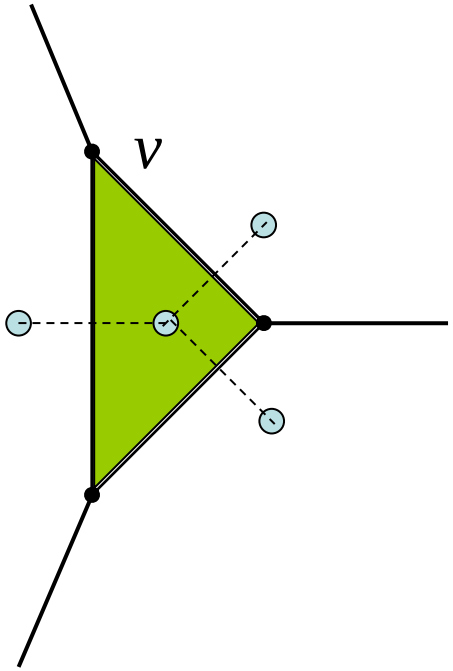
$$V_i(p_i) = \bigcap_{1 \le j \le n,\, j \ne i} h(p_i, p_j)$$

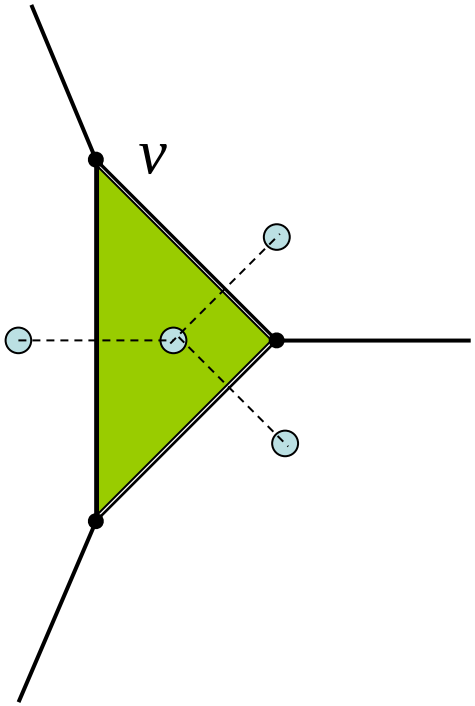


Jeśli punkty są współliniowe,
to krawędzie diagramu Voronoi
są prostymi

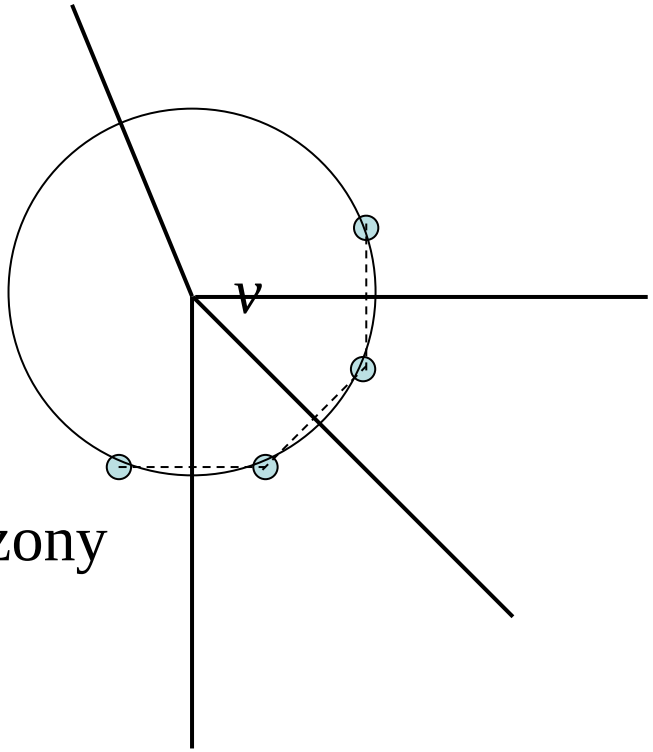
W przeciwnym razie
krawędzie są albo odcinkami,
albo półprostymi





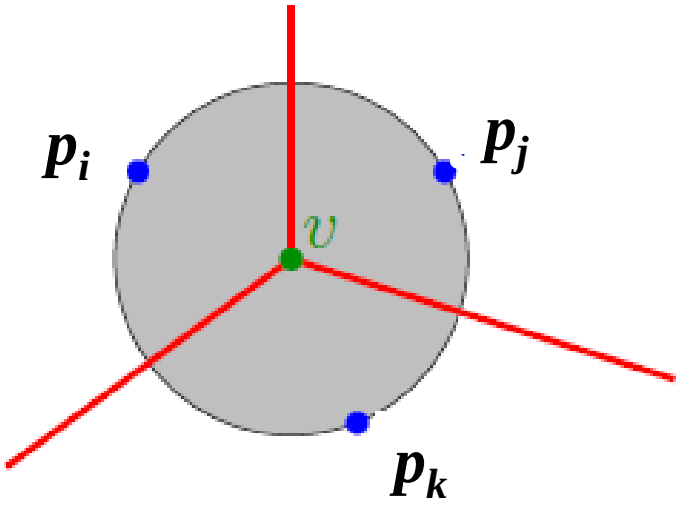


obszar jest ograniczony

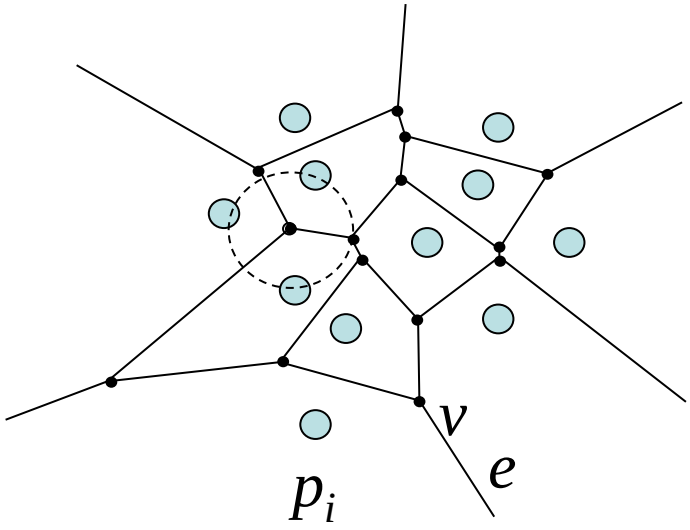


obszar nie jest ograniczony

Wierzchołek wieloboku Voronoi
jest równoodległy od każdego punktu p_i, p_j, p_k.



Koło o środku *v* przechodzące przez punkty p_i, p_j, p_k
nie zawiera w swoim wnętrzu żadnego innego punktu *S*

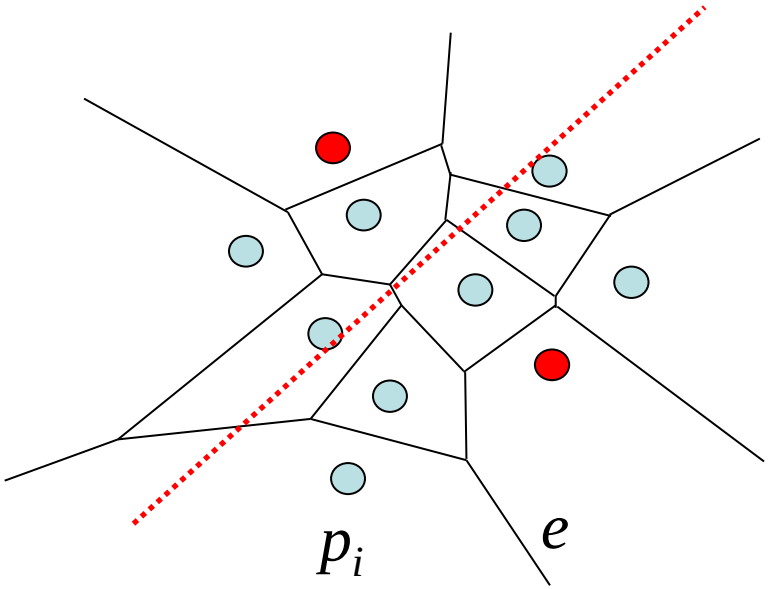


V – komórka (wielobok, obszar) Voronoi
v – wierzchołek wieloboku, środek okręgu opisanego na trójkącie p_1, p_2, p_3

Do każdego obszaru Voronoi należy dokładnie jeden punkt z S.
Punkty z S nazywamy ***generatorami*** (***centrami***)
odpowiednich obszarów Voronoi.

Punkty należące do brzegów obszarów Voronoi tworzą
diagram Voronoi Vor(S)

Uwaga:



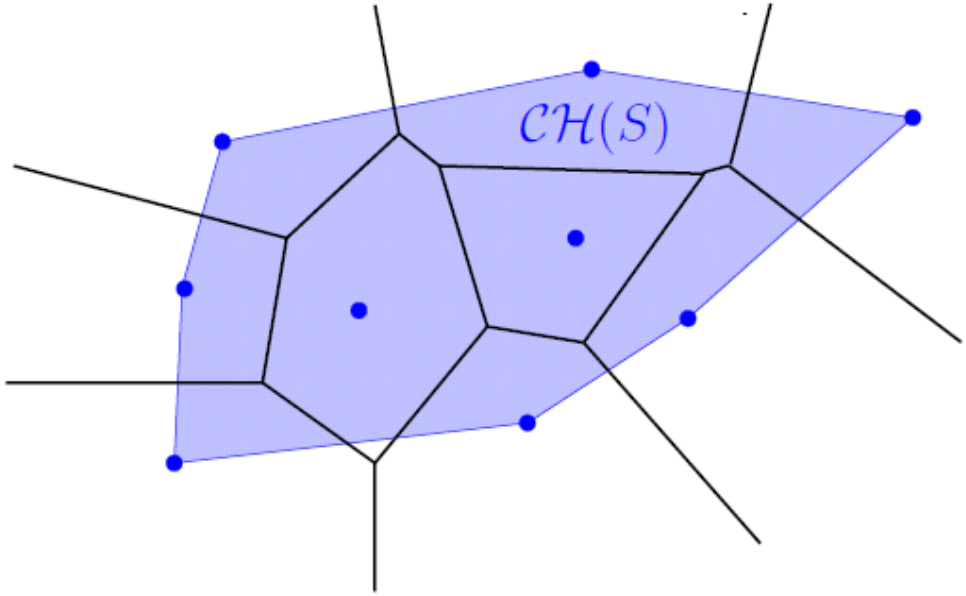
Nie wszystkie symetralne są bokami wielokątów Voronoi.

Komórka Voronoi jest wielokątem wypukłym

(czasem nieograniczonym)

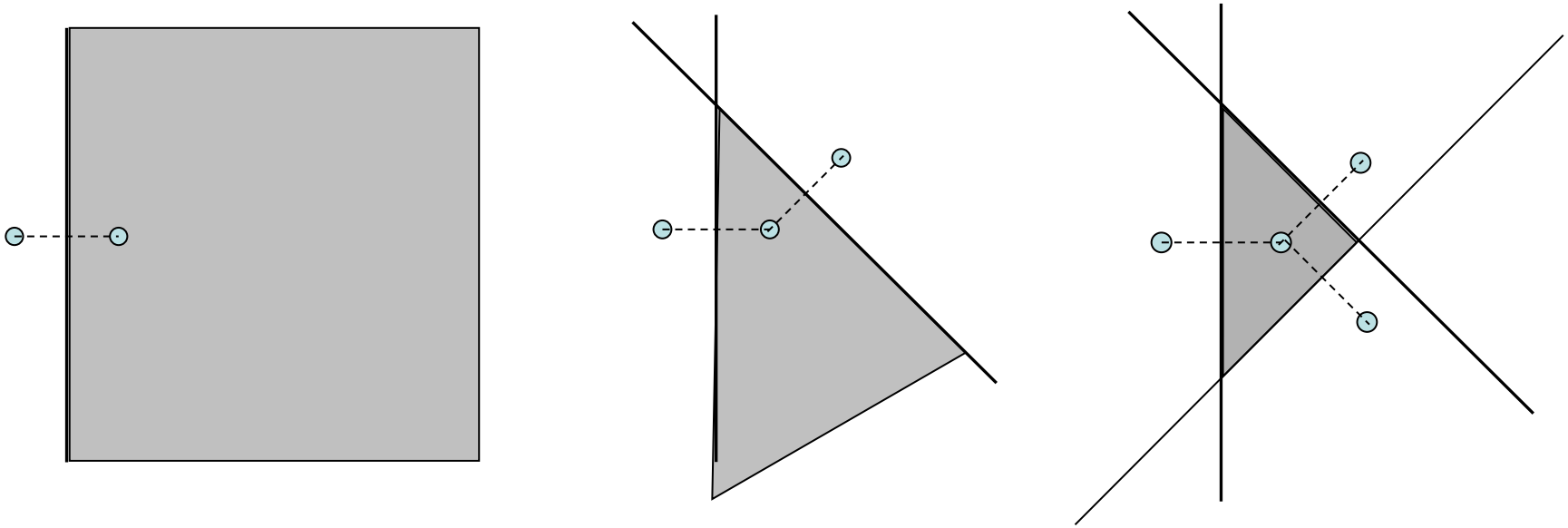
Dokładniej:

- jeśli $V(P_i)$ jest ograniczony
  – jest wielokątem wypukłym,
- jeśli $V(P_i)$ jest nieograniczony
  – P_i jest wierzchołkiem otoczki $CH(S)$



Tworzenie diagramu – I sposób

Dodajemy kolejne punkty S i znajdujemy przecięcia półpłaszczyzn



Złożoność $O(n^2 \log n)$

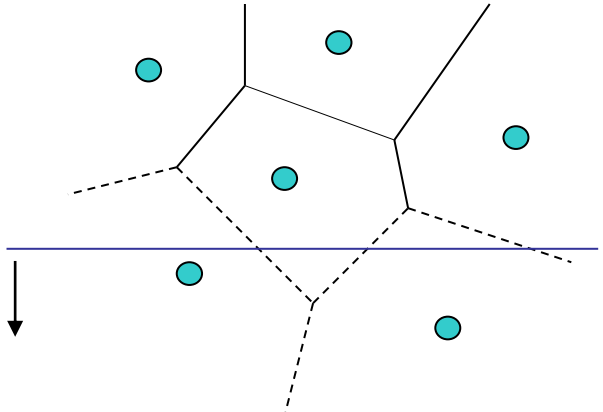
Algorytm zamiatania – Fortune'a

Złożoność O(n log n)

Miotła ***l*** przesuwana poziomo z góry na dół

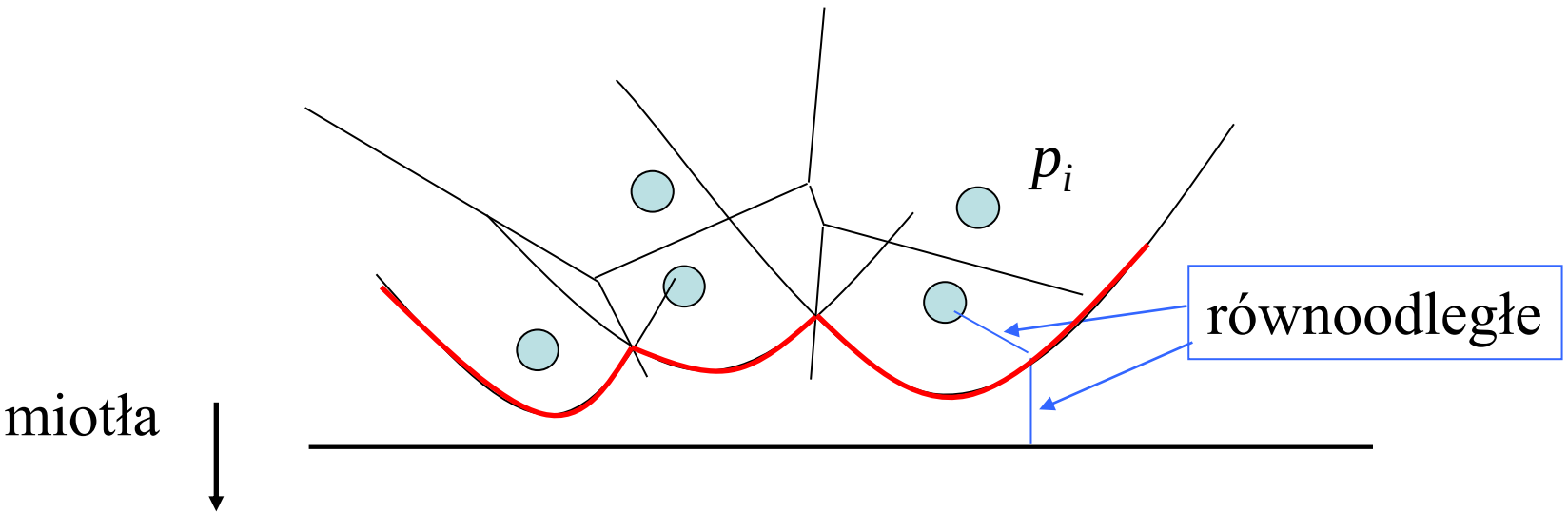
Problem:

Część Vor(S) powyżej miotły
zależy nie tylko od centrów powyżej ***l***,
ale też od tych poniżej miotły!



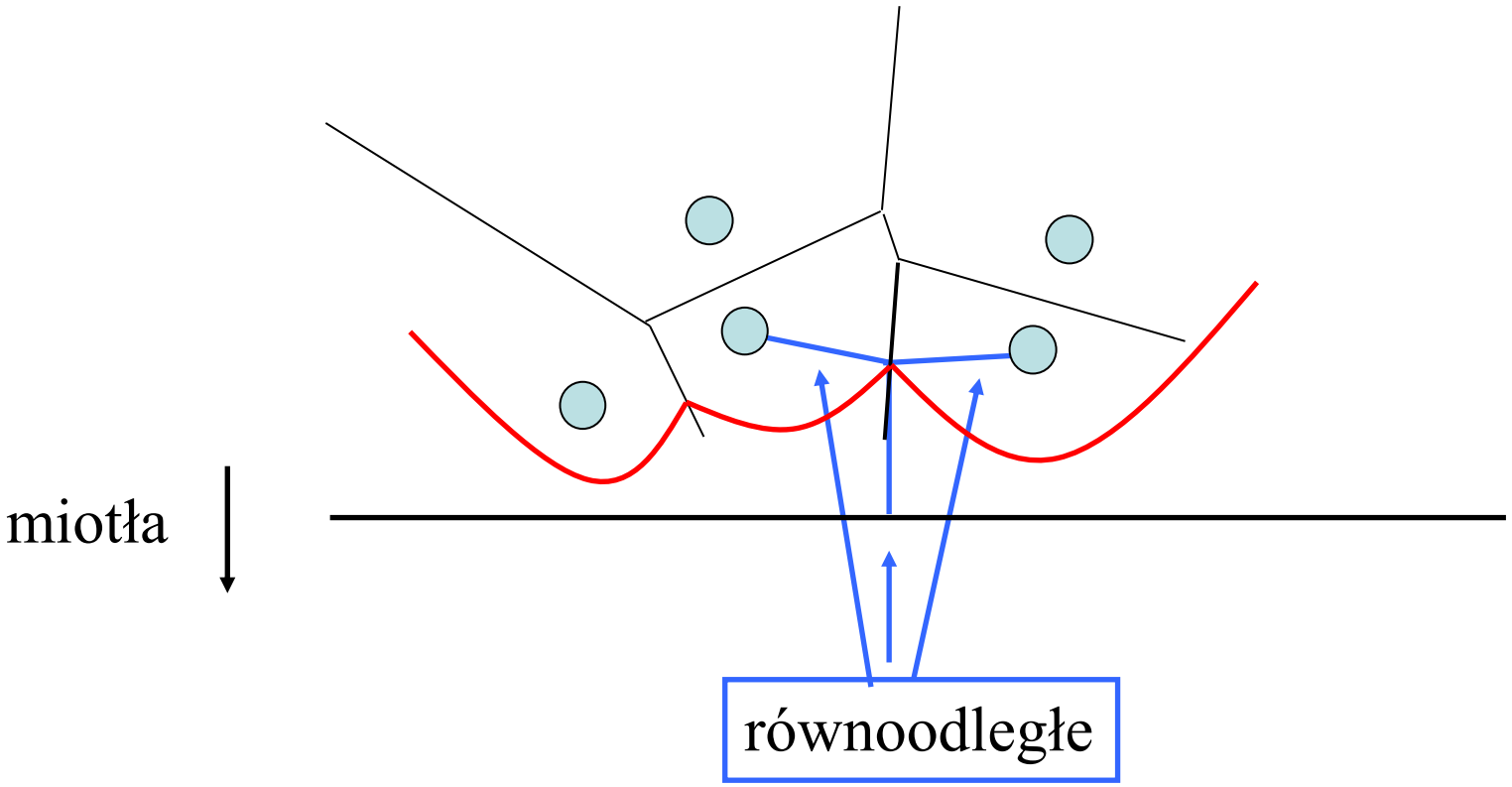
Zamiast przechowywać informacje o przecięciach diagramu z miotłą
– informacje o części diagramu, która nie może zostać zmieniona
przez centra poniżej miotły

Które punkty są bliżej do centrum powyżej miotły niż do samej miotły?

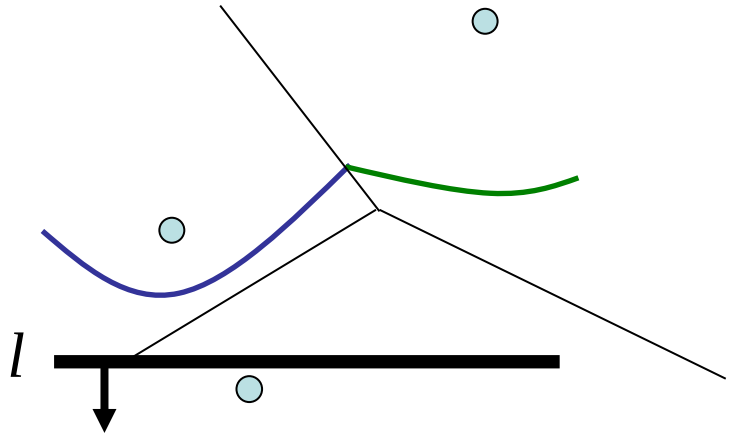


Ciąg parabolicznych łuków tworzy linię brzegu ograniczającą położenia takich punktów.

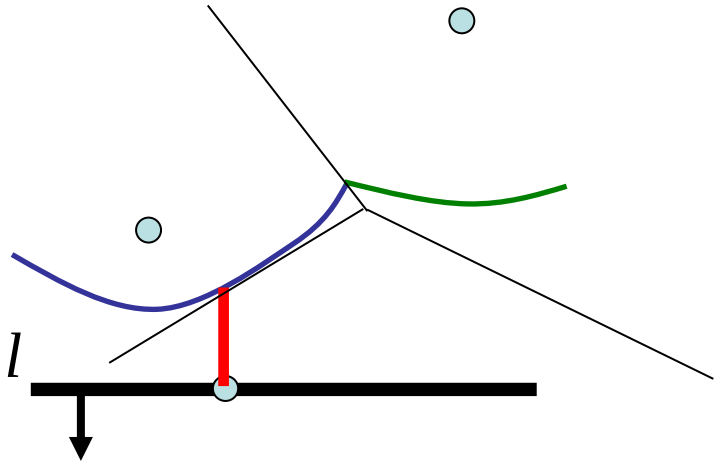
Punkty „***załamania***” parabol wytyczają diagram Voronoi



Zdarzenie punktowe



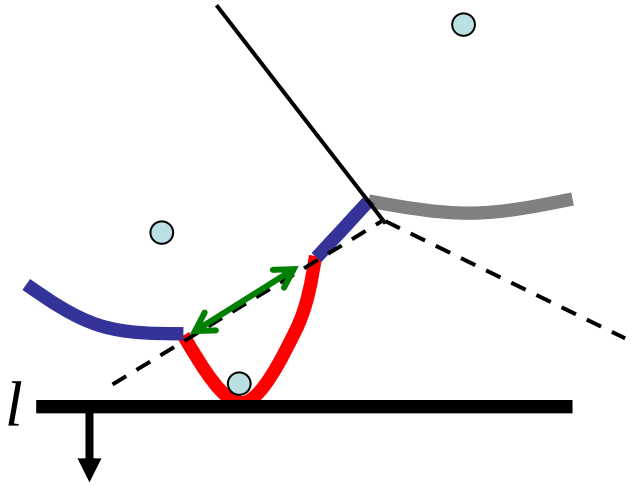
Nowy łuk powstaje podczas napotkania nowego punktu



Początkowo jest to parabola zdegenerowana do odcinka pionowego.

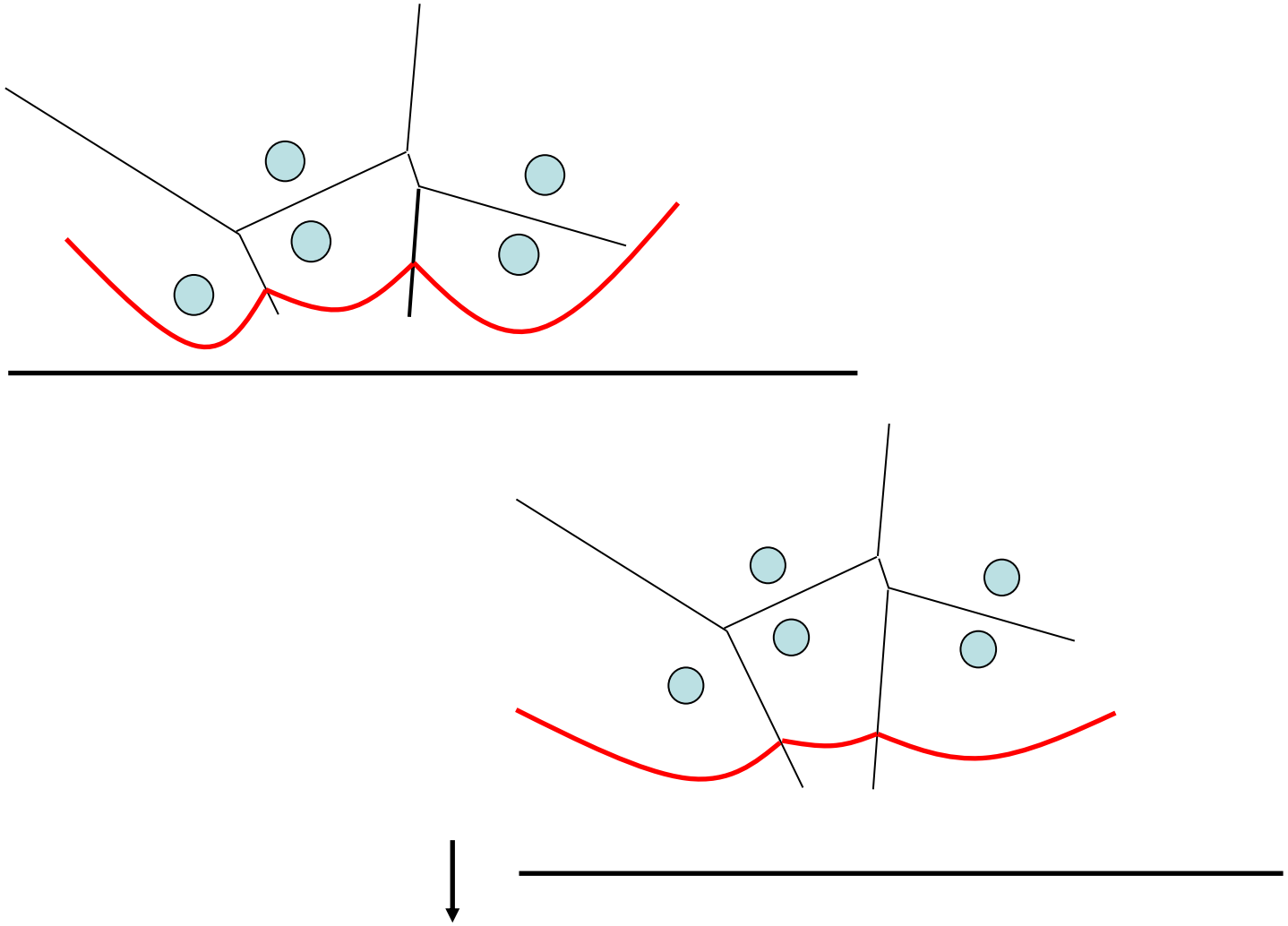
Zdarzenie punktowe

- Pojawia się nowy łuk.
- Pierwotny łuk ponad punktem zostaje podzielony na dwa nowe.
- Zaczyna być wyznaczana nowa krawędź diagramu.

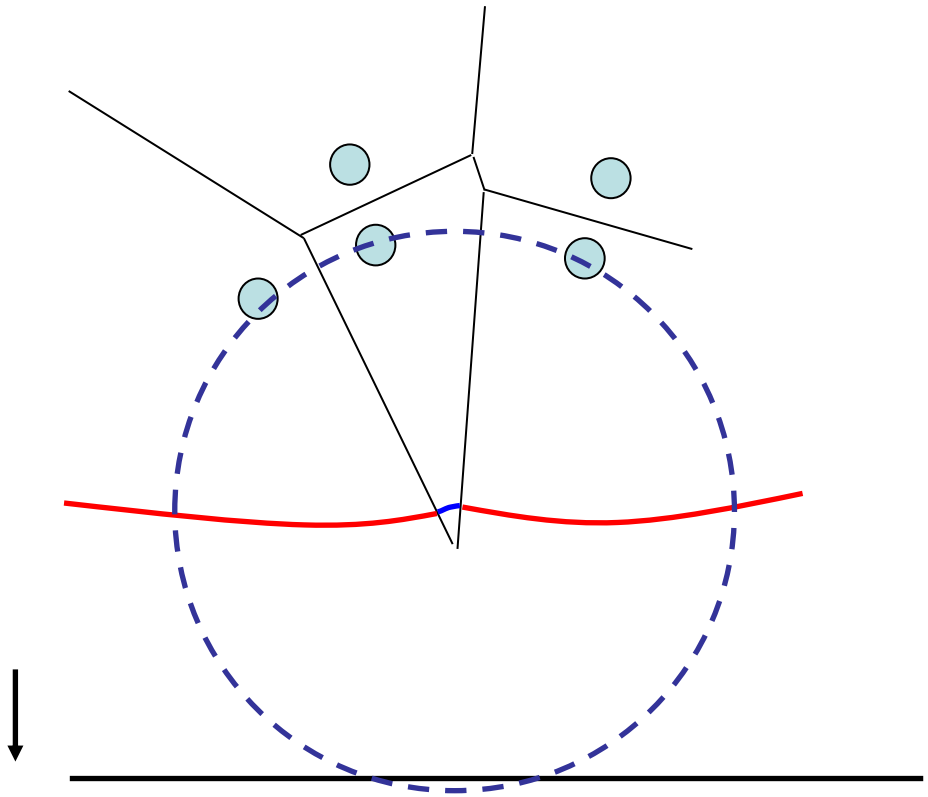


Każde napotkane centrum daje przyrost o jeden łuk oraz podział co najwyżej jednego istniejącego łuku.

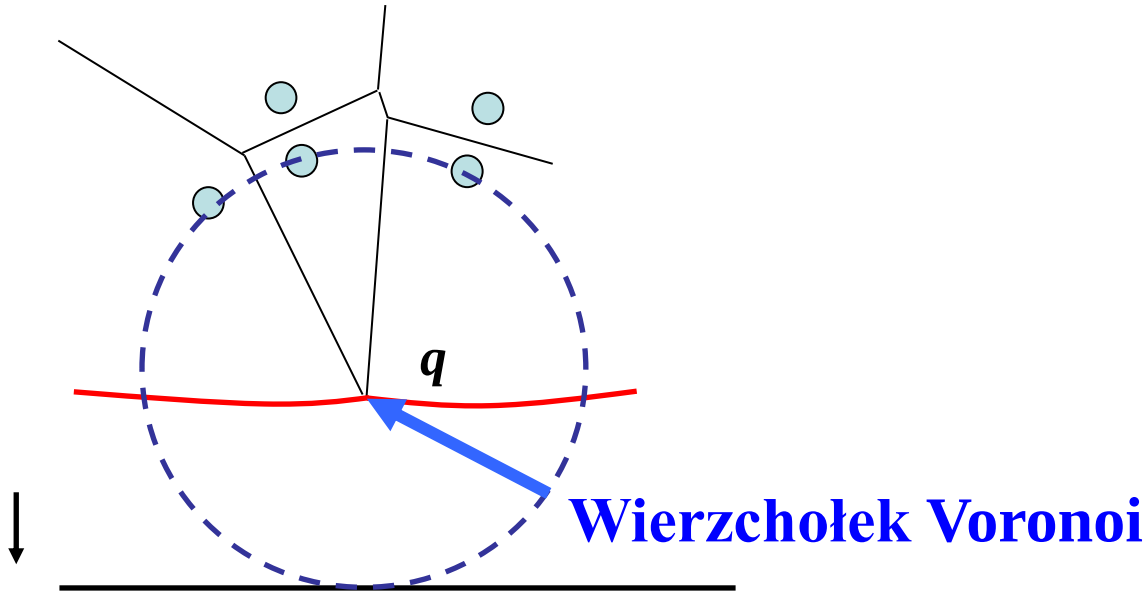
Łuki stają się coraz bardziej płaskie, jak miotła przesuwa się w dół



Środkowy łuk zaczyna zanikać.



Zdarzenie okręgowe



- Łuk znika – parabole definiowane przez trzy centra przechodzą przez wspólny punkt ***q***
- Punkt ***q*** jest równoodległy od centrów i miotły.
- Istnieje okrąg przechodzący przez centra, styczny do miotły, ze środkiem w ***q***.
- Punkt ***q*** jest wierzchołkiem diagramu Voronoi.

Własności linii brzegu

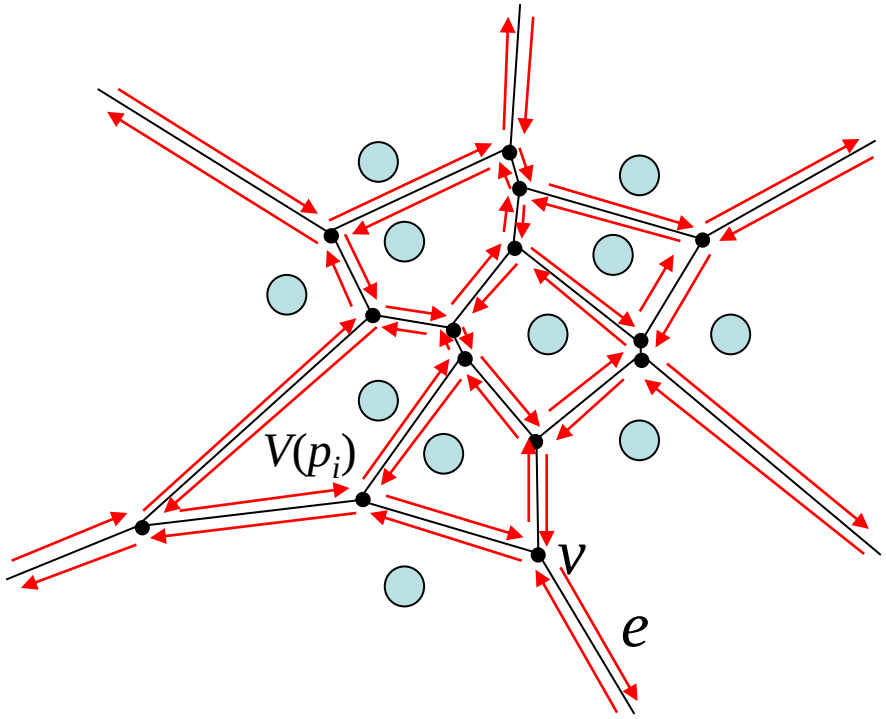
- **Krawędzie Voronoi** wykrywane zostają przez punkty załamania w miarę przesuwania się prostej zamiatającej w dół płaszczyzny.
  - Powstawanie nowego punktu(ów) załamania (z utworzonego nowego łuku lub połączenia dwu istniejących punktów załamania) tworzy nową krawędź
- **Wierzchołki Voronoi** są wykrywane, gdy dwa punkty załamania łączą się.

Struktury danych

- Bieżący stan **diagramu Voronoi (*D*)**
  - Podwójnie łączona lista krawędzi, wierzchołków, komórek
- Bieżący stan **linii brzegowej (struktura stanu *T*)**
  - Monitoruje punkty załamania
  - Monitoruje łuki znajdujące się na linii brzegowej
- **Struktura zdarzeń (*Q*)**
  - Kolejka priorytetowa posortowana malejąco względem współrzędnej y

Podwójnie łączona lista krawędzi

- Z każdą krawędzią związane dwie przeciwnie zorientowane ***półkrawędzie***.
- Łańcuch skierowanych przeciwnie do ruchu wskazówek zegara półkrawędzi tworzy komórkę Voronoi.

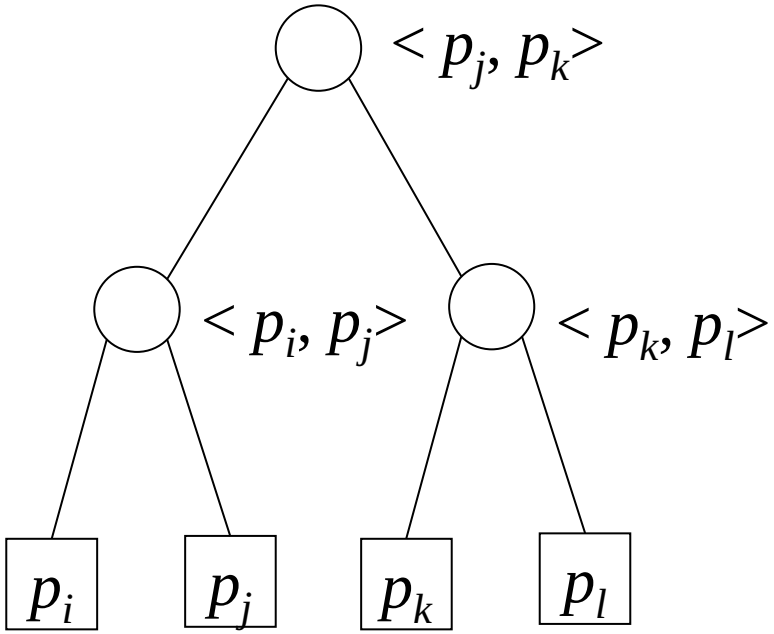


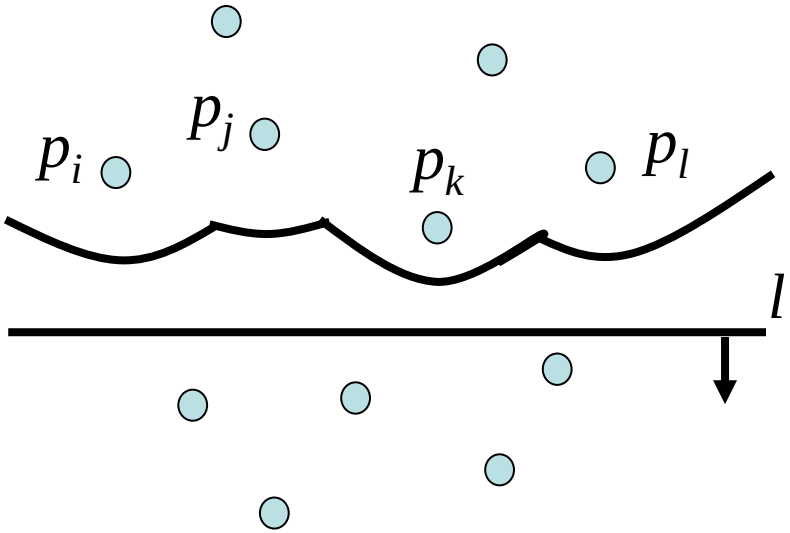
Linia brzegowa – struktura stanu
Zrównoważone drzewo wyszukiwań binarnych

- **Węzły wewnętrzne**

  reprezentują punkty załamania między dwoma łukami

- Każdy **liść** przechowuje centrum, które definiuje łuk





Struktura zdarzeń

Kolejka zdarzeń składa się z:

- **zdarzeń punktowych** (podczas napotkania przez prostą zamiatającą nowego punktu) – przechowujemy samo centrum,
- **zdarzeń okręgowych** (podczas gdy prosta zamiatająca wykrywa dolny punkt *pustego okręgu stycznego do* 3 lub więcej centrów) – najniższy punkt okręgu ze wskaźnikiem do liścia w T reprezentującego łuk znikający w tym zdarzeniu.

Zdarzenia kolejkowane są względem współrzędnych y

Zdarzenia punktowe są znane z góry, ale okręgowe nie.

Algorytm

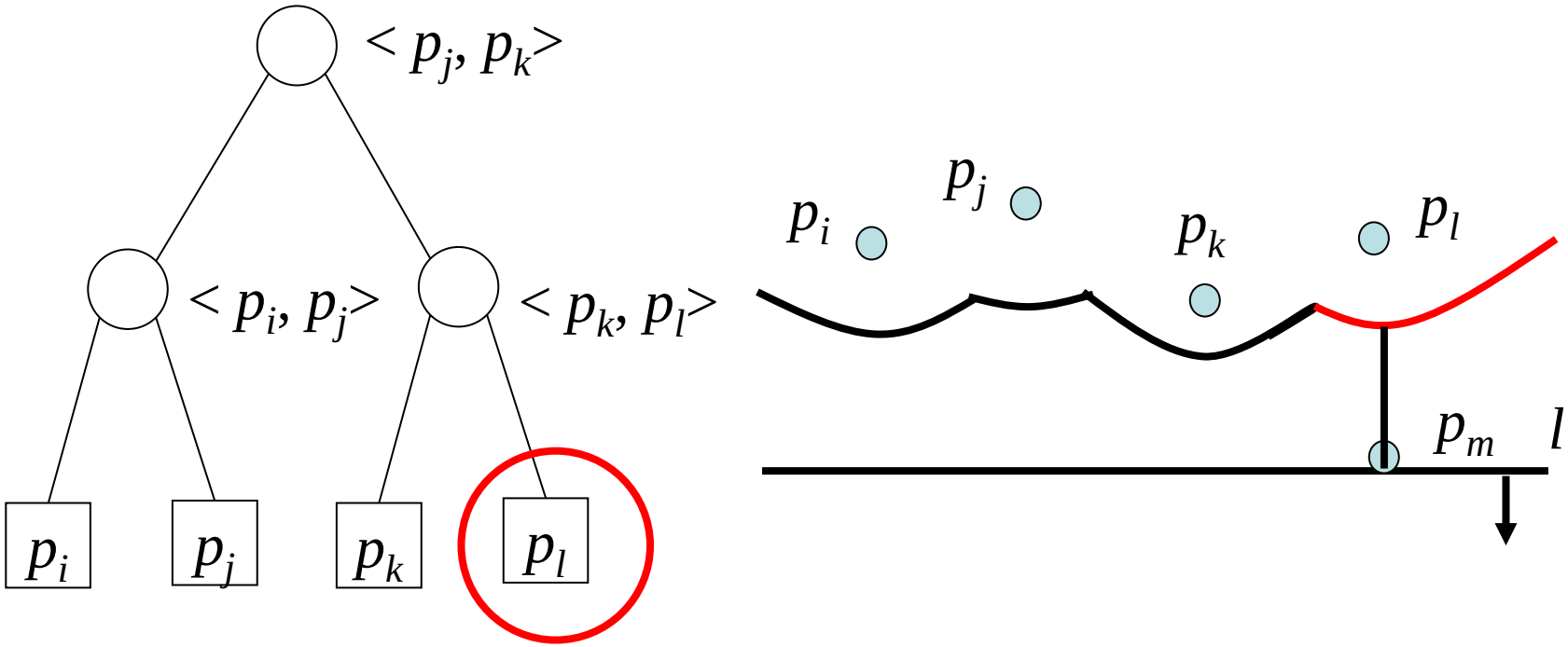
- Inicjuj kolejkę zdarzeń ***Q*** wszystkimi zdarzeniami punktowymi
- Inicjuj pustą strukturę stanu ***T***
- Inicjuj pustą strukturę *D*

- **while** Q nie jest pusta:
  if nowe zdarzenie jest punktowe
  then HANDLEPOINTEVENT
  else HANDLECIRCLEEVENT;

Obsługa zdarzenia punktowego

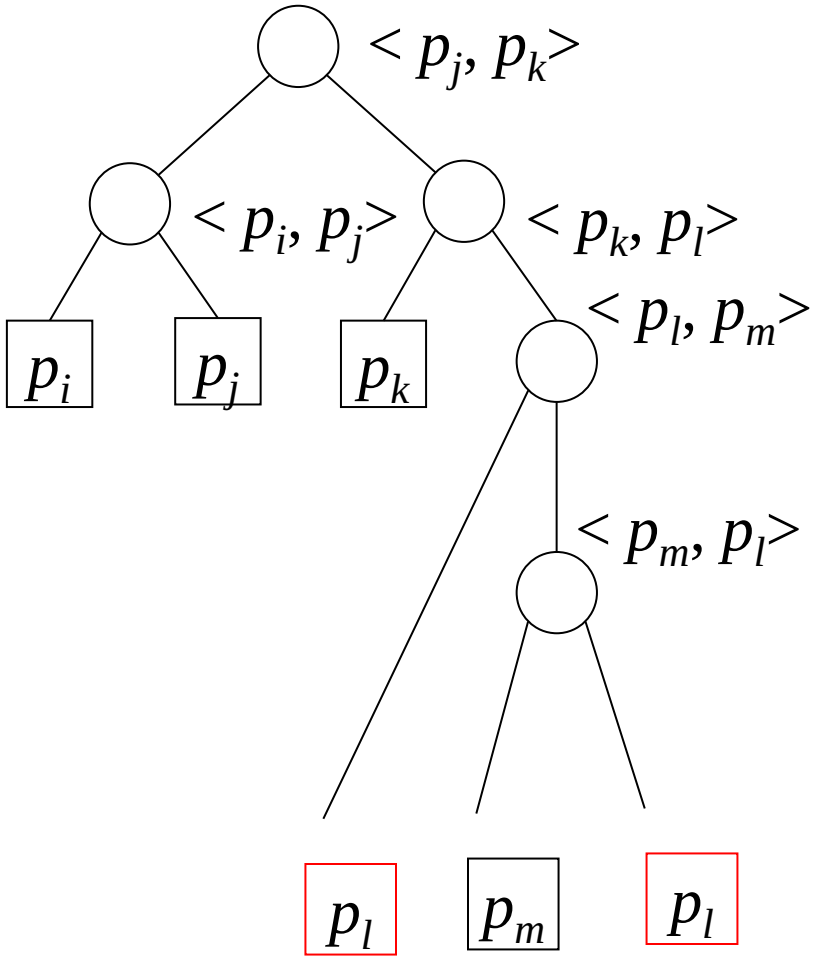
1. Odszukanie łuku (o ile istnieje) ponad nowym punktem
2. Podział łuku poprzez zamianę liścia poddrzewem reprezentującym nowy łuk i jego punkty załamania
3. Dodanie nowej półkrawędzi do struktury D
4. Sprawdzenie czy zachodzi zdarzenie *potential circle event*, jeżeli tak to należy je dodać do kolejki zdarzeń

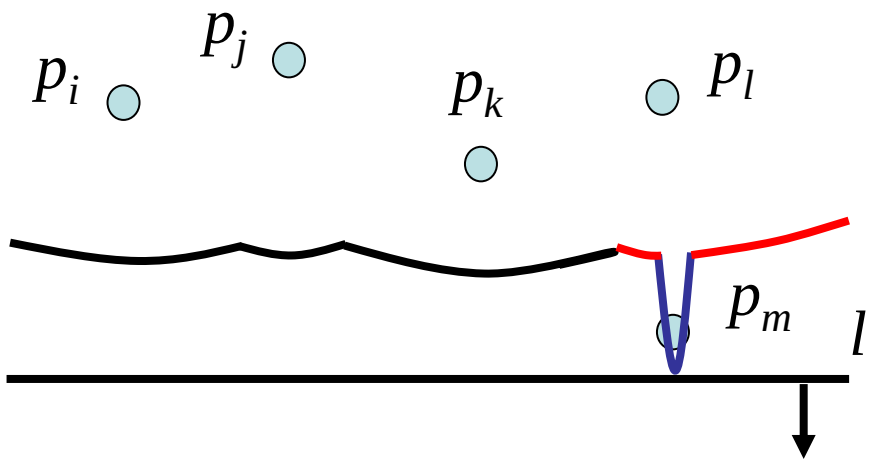
Odszukanie istniejącego łuku ponad nowym punktem



Podział łuku

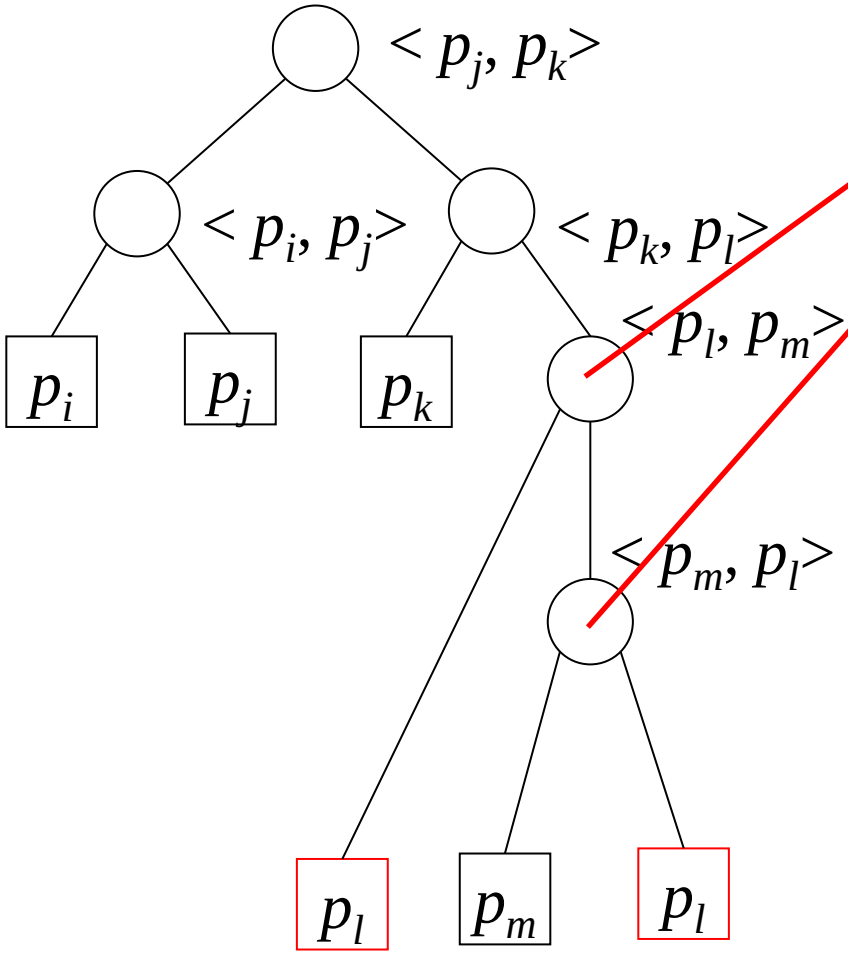
Odpowiadający liść zostaje zamieniony nowym poddrzewem





Różne łuki można zidentyfikować poprzez ten sam punkt!

Dodanie nowego rekordu krawędzi w strukturze *D*

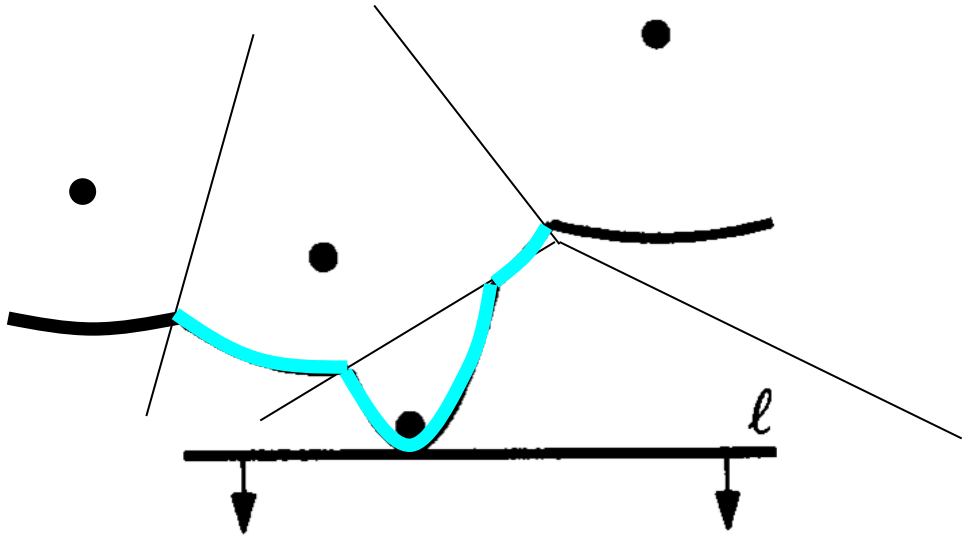


Wskaźniki do rekordów półkrawędzi

Wykonaj na T operację równoważenia

Sprawdzenie możliwych zdarzeń *Potential Circle Events*

Skanowanie trójek kolejnych łuków i sprawdzenie czy punkty załamania zbiegają się

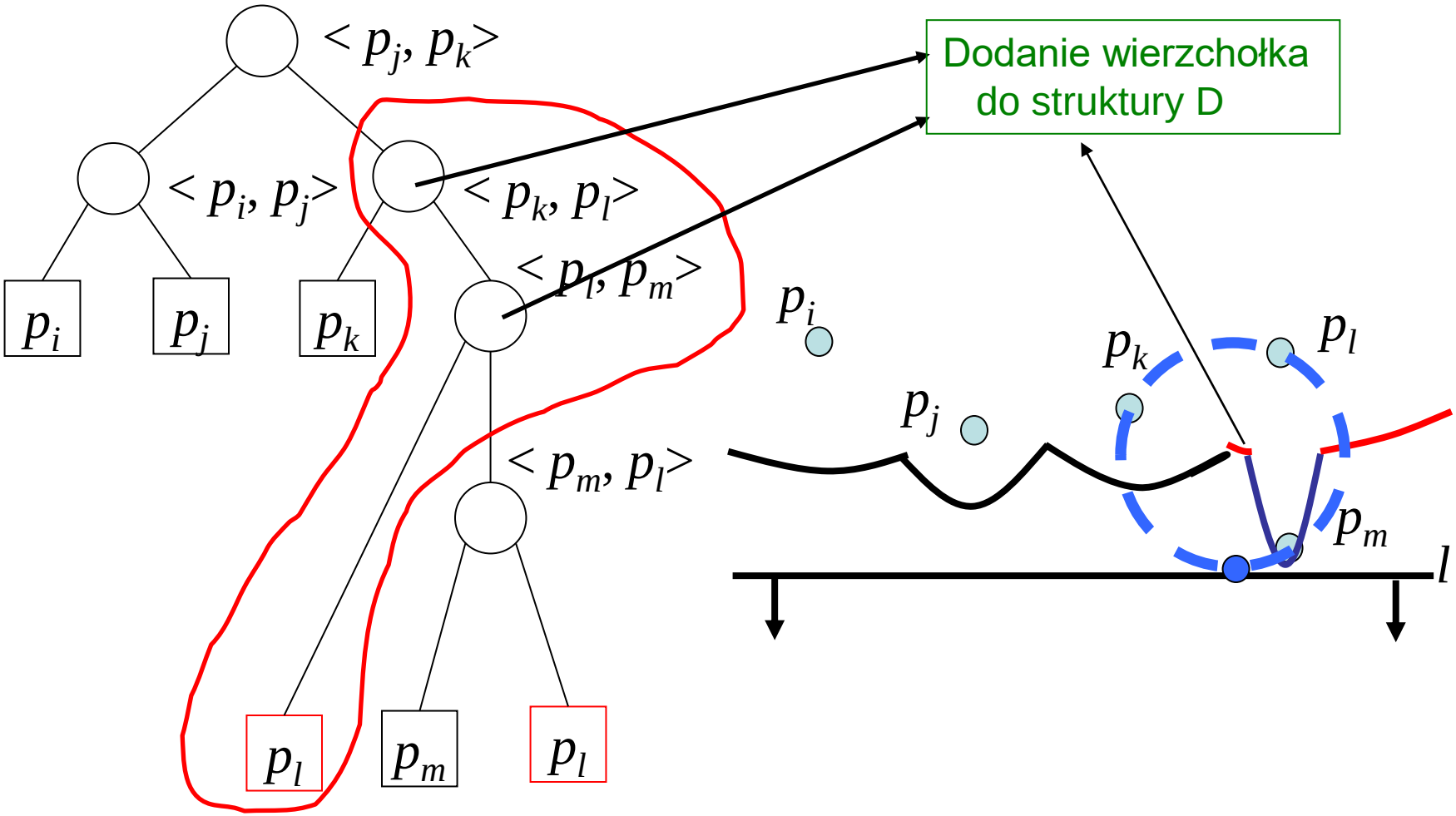


- Jeśli wyznacza zdarzenie okręgowe – wprowadzić do Q

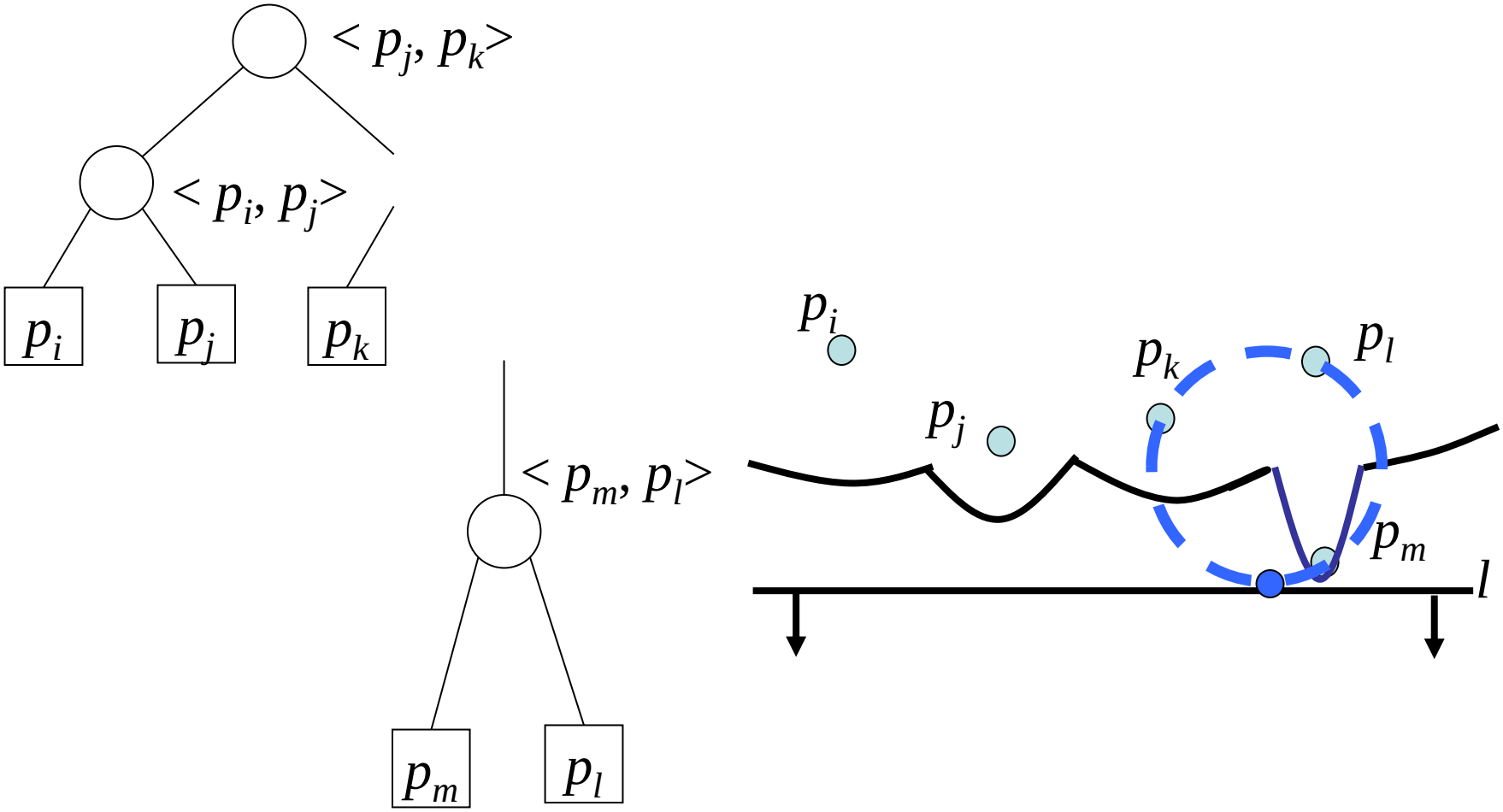
Obsługa zdarzenia okręgowego

1. Dodanie wierzchołka do struktury D
2. Usunięcie środkowej znikającej paraboli
3. Dodanie nowej krawędzi do struktury
4. Sprawdzenie czy zachodzi zdarzenie *potential circle event*, jeżeli tak to należy je dodać do kolejki zdarzeń

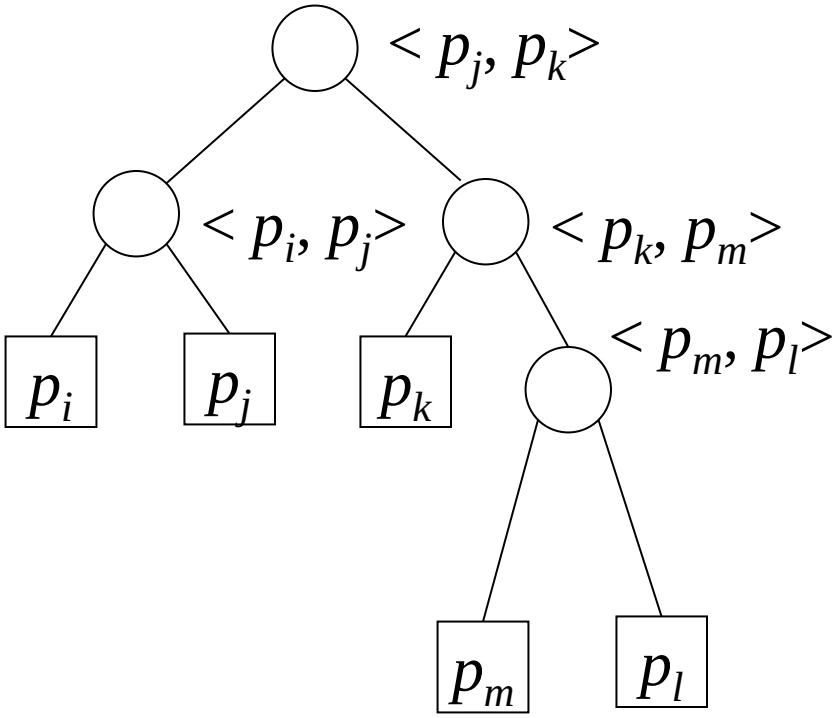
Obsługa zdarzenia okręgowego

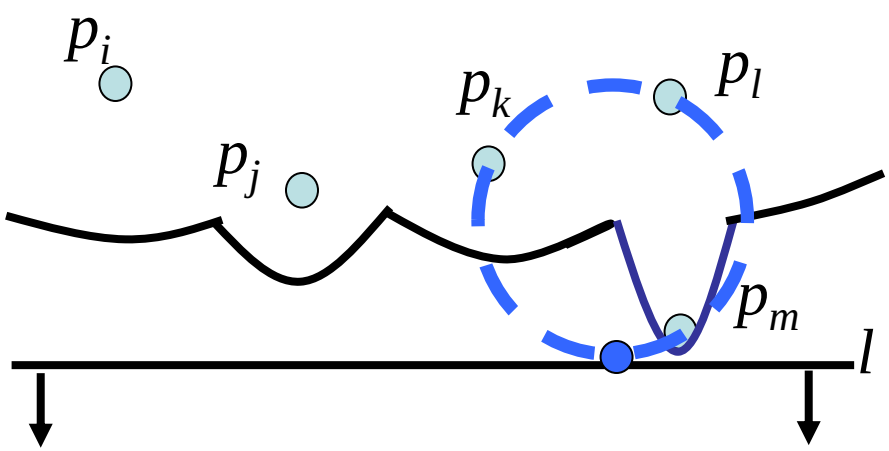


Usunięcie znikającego łuku

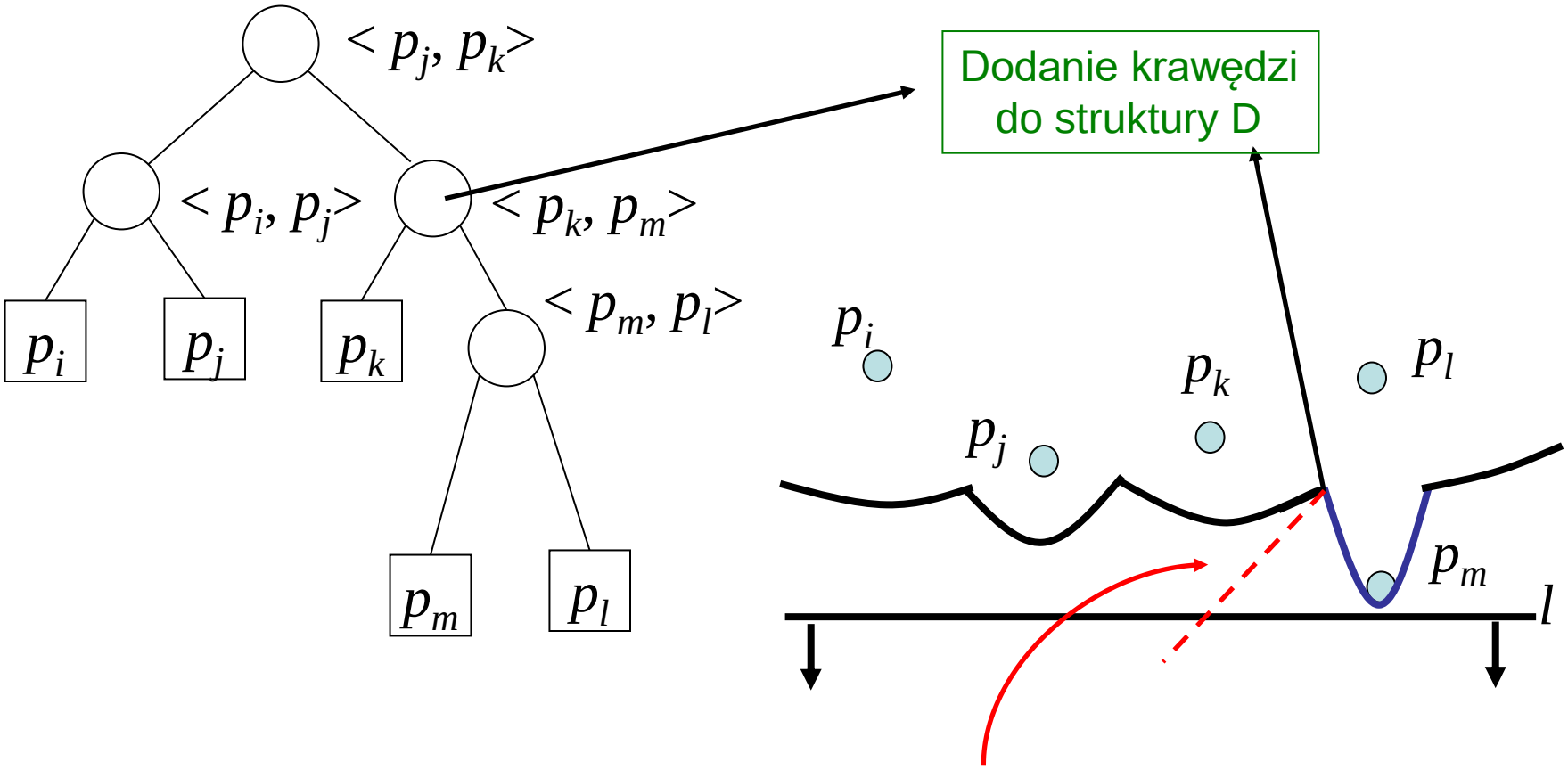


Usunięcie znikającego łuku



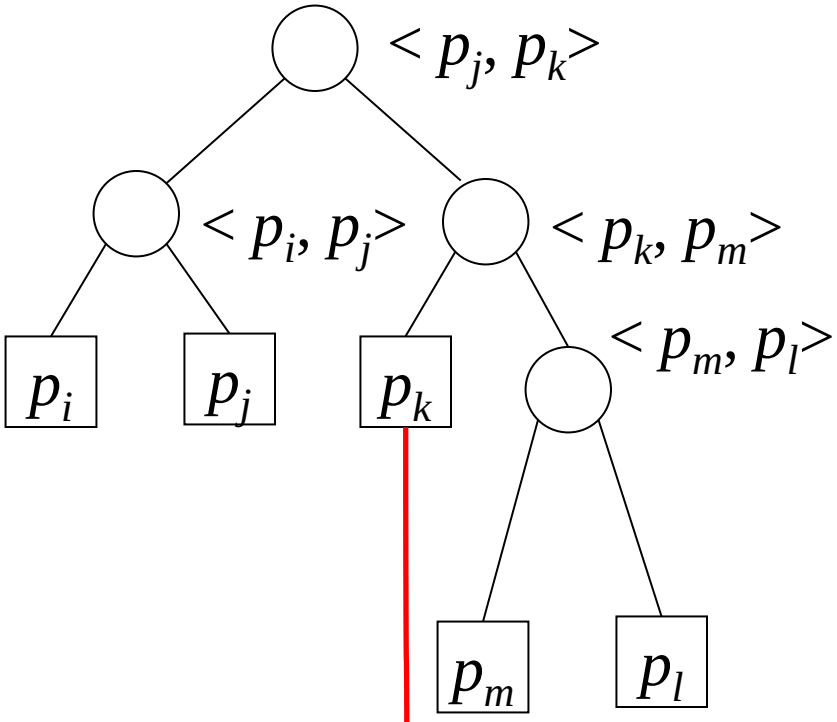


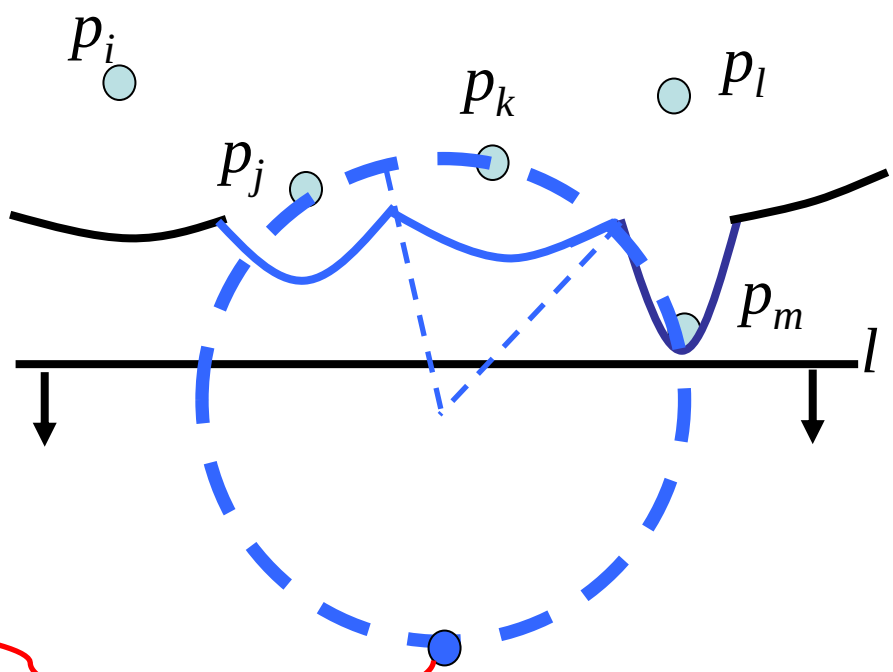
Utworzenie nowego rekordu krawędzi

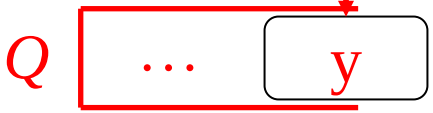


Nowa krawędź z nowego punktu załamania < p_k, p_m>

Sprawdzenie nowych trójek dla *potential circle events*

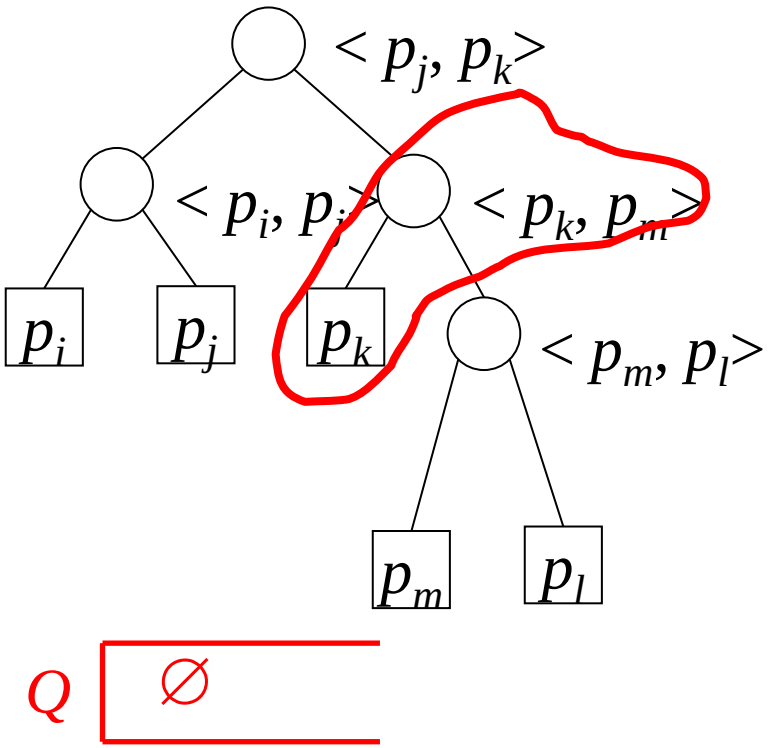


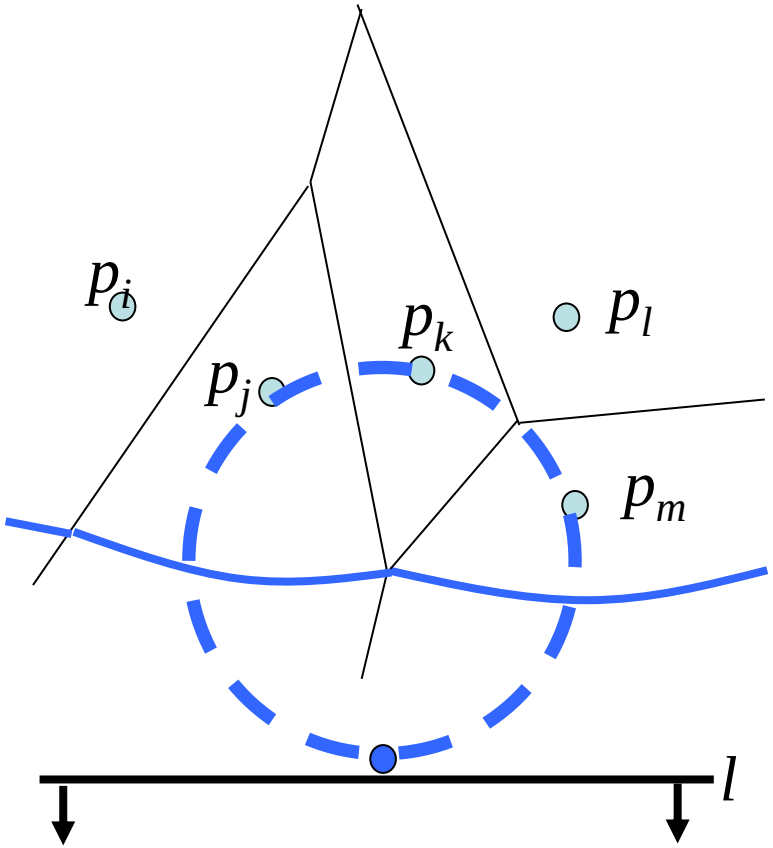




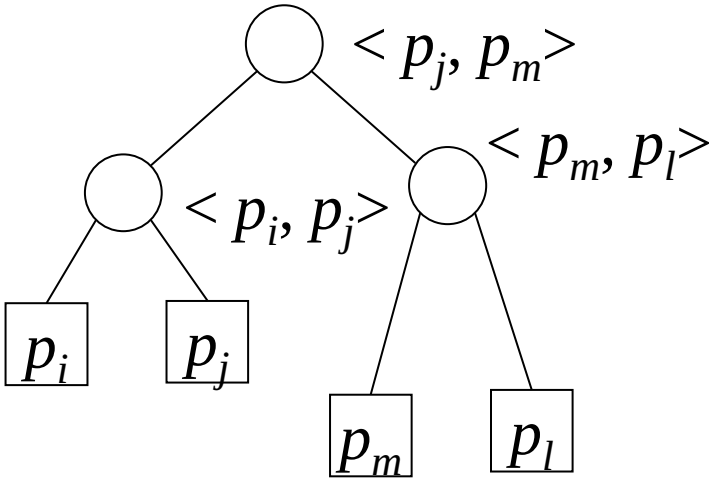
Nowe zdarzenie okręgowe

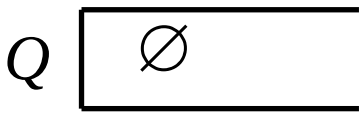
Zakończenie działania

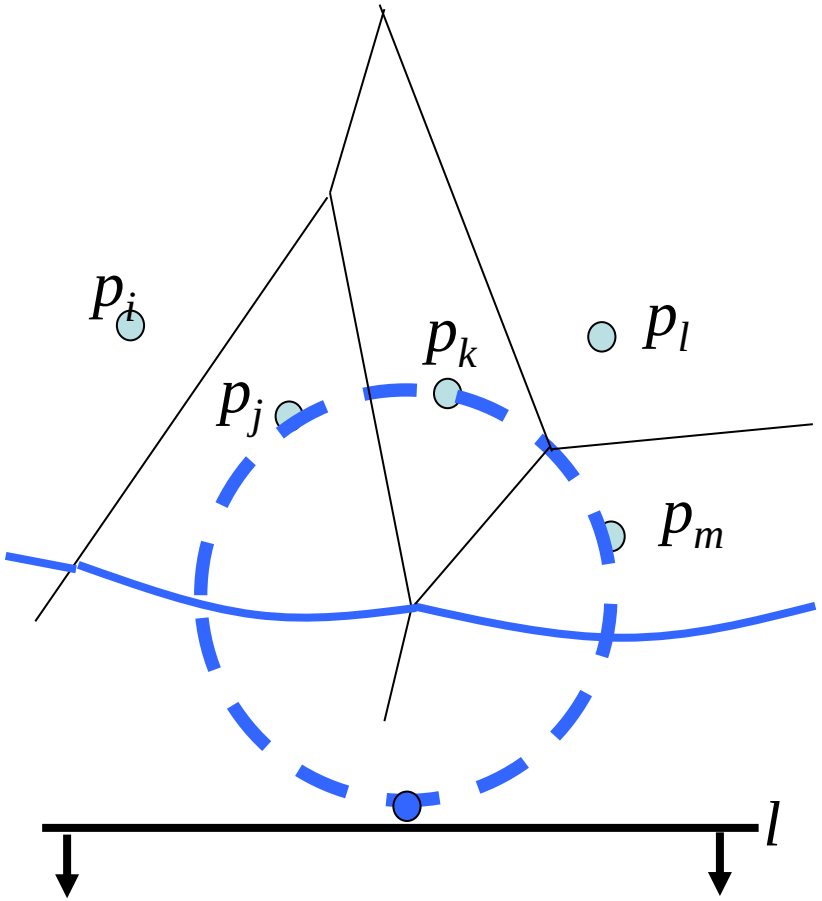




Zakończenie działania







Algorytm kończy działanie gdy Q = ∅, ale linia brzegowa i punkty załamania nadal podążają za krawędziami Voronoi.

Należy zakończyć te nieskończone krawędzie prostokątem ograniczającym.

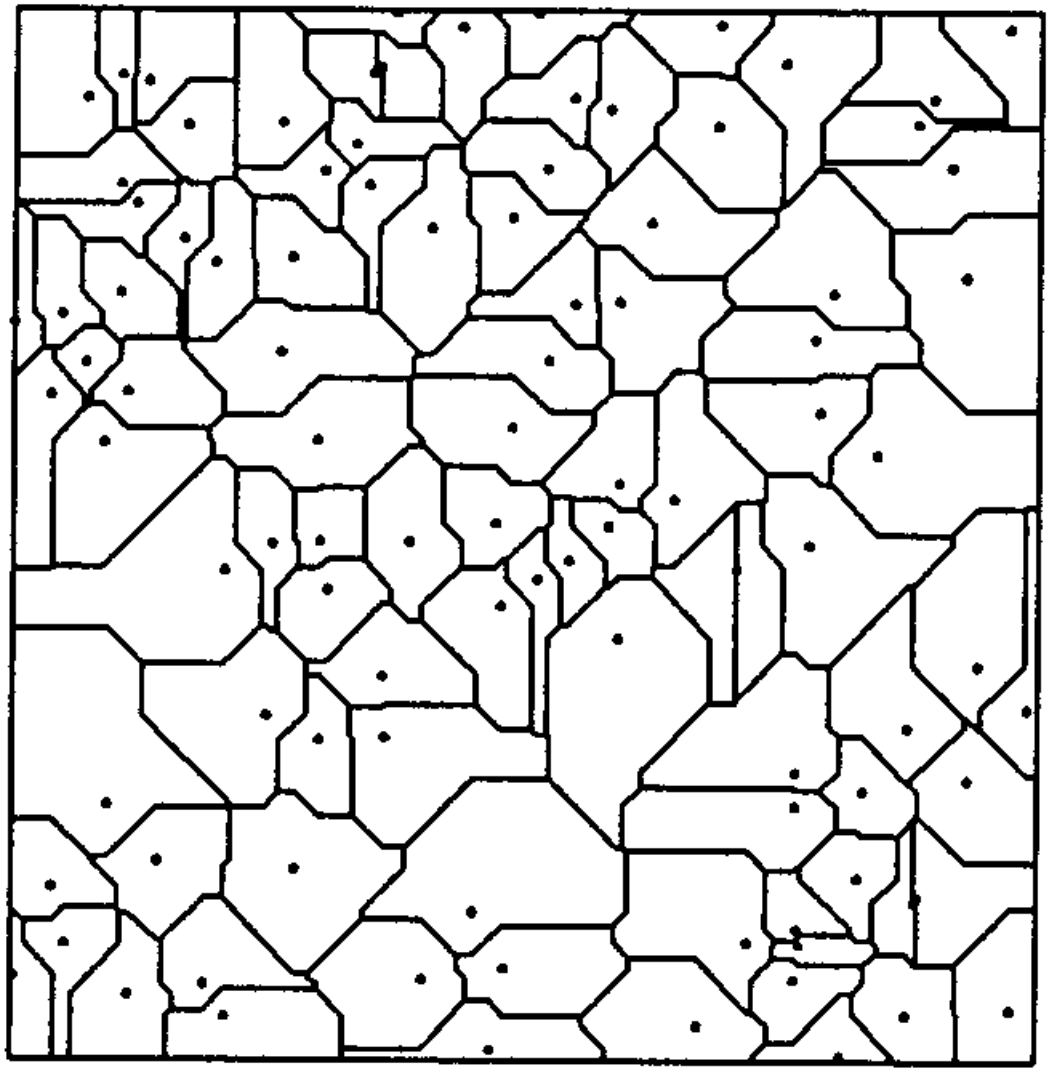


Diagram Voronoi dla innej metryki

TRIANGULACJA DELAUNAY'A

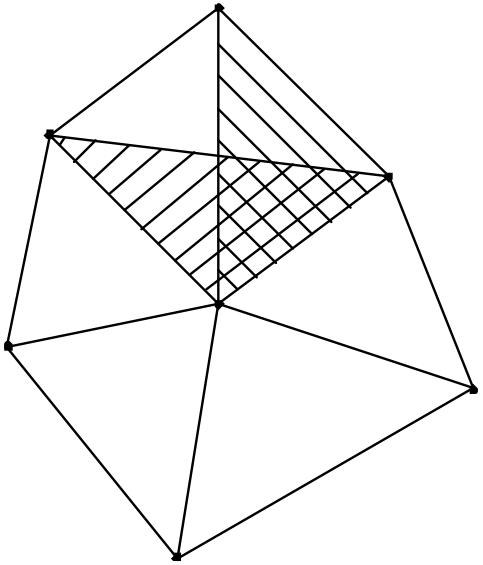
Triangulacja

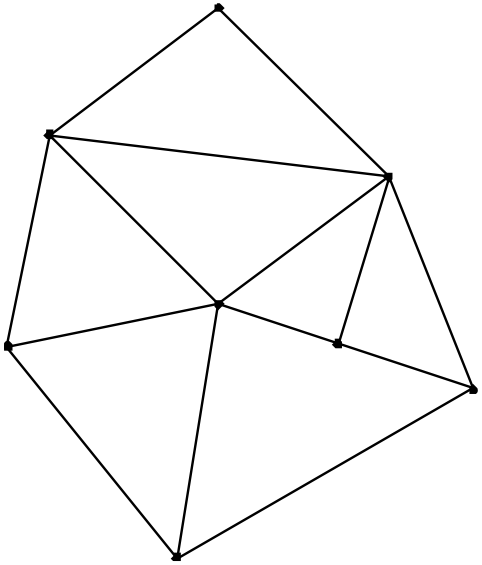
Niech:
S – zadana chmura punktów,
otoczka wypukła $HC(S)$ definiuje obszar Ω (powłoka),
K – określa sympleks w danej przestrzeni

Definicja:

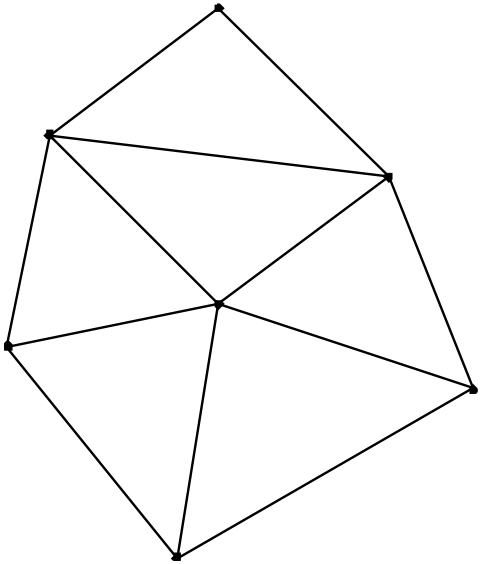
T nazywamy triangulacją Ω, jeżeli spełnione są następujące warunki:

- Zbiór wierzchołków elementów $T = S$,
- $\Omega = \bigcup_{K \in T} K$
- Każdy element K ma niepuste wnętrze,
- Wnętrza różnych elementów są rozłączne ,
- Przecięcie dwóch różnych elementów jest:
  - (a) albo puste
  - (b) albo zredukowane do jednego punktu,
    który jest wspólnym wierzchołkiem tych elementów
  - (c) albo zredukowane do jednej krawędzi,
    która jest wspólną krawędzią tych elementów
  - (d) albo zredukowane do jednej ściany (w 3D),
    która jest wspólną ścianą tych elementów.



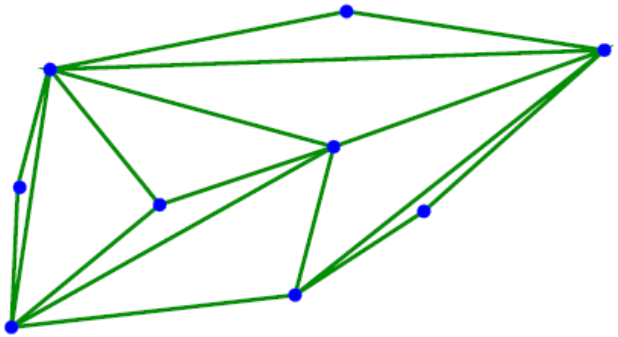


niepoprawne



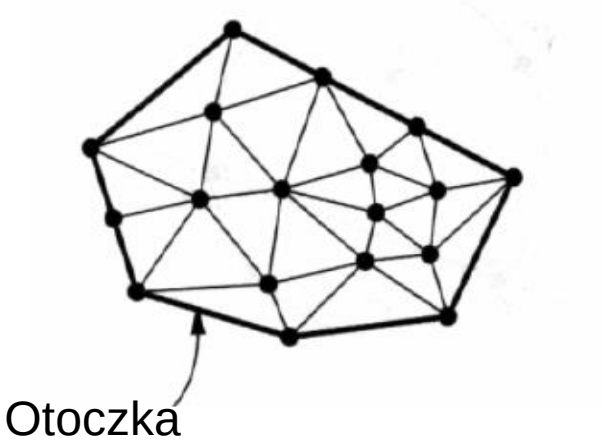
poprawna

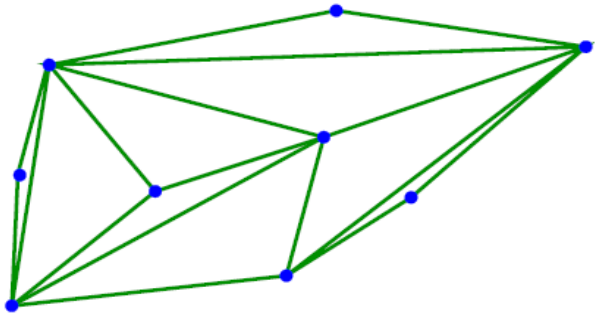
Triangulacja powłoki chmury punktów

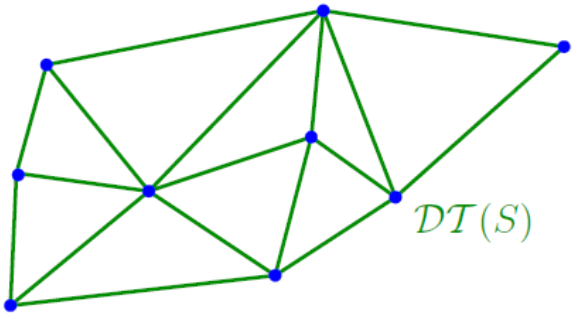


Dla danej chmury *S* punktów znaleźć powłokę wypukłą, która będzie odpowiednio podzielona na trójkąty

W 2D dla chmury *n* punktów
każda triangulacja zawiera
$2n - 2 - k$ trójkątów
$3n - 3 - k$ krawędzi,
gdzie *k* – liczba wierzchołków otoczki





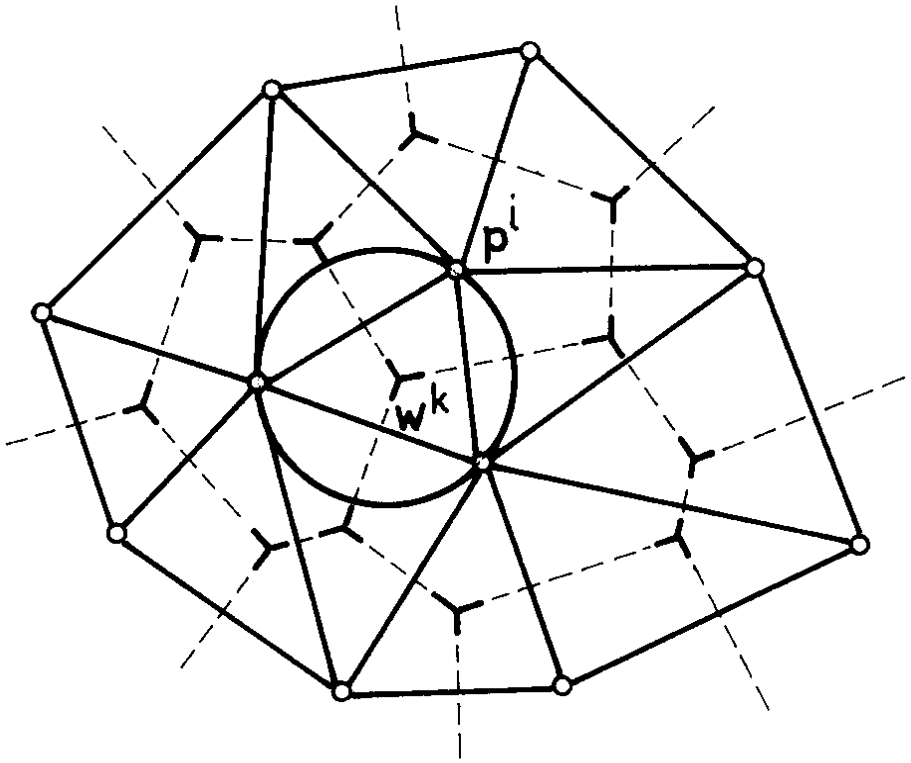


Triangulacja Delaunay'a ***DT(S)***

Ta sama chmura punktów – a triangulacja wygląda lepiej

Na dodatek ma wiele interesujących własności

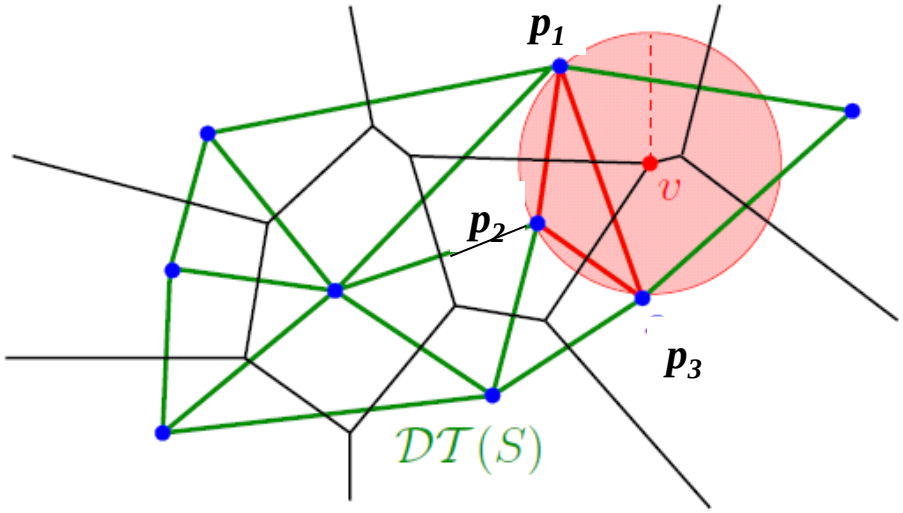
Triangulacja Delaunay'a



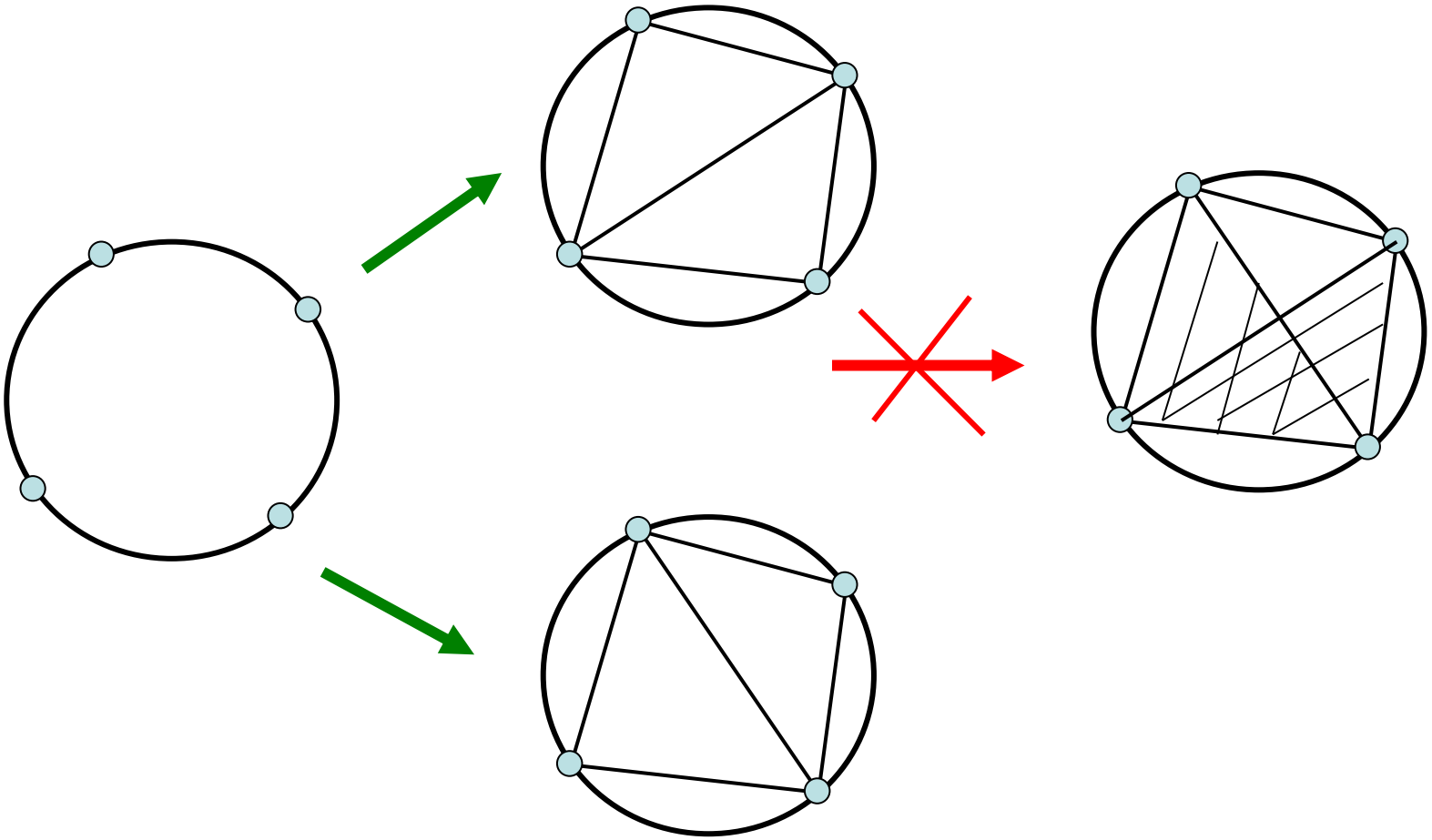
Wieloboki Voronoi (linia przerywana)
oraz dualna wobec nich
triangulacja Delaunay'a (linia ciągła)

Kryterium kuli opisanej

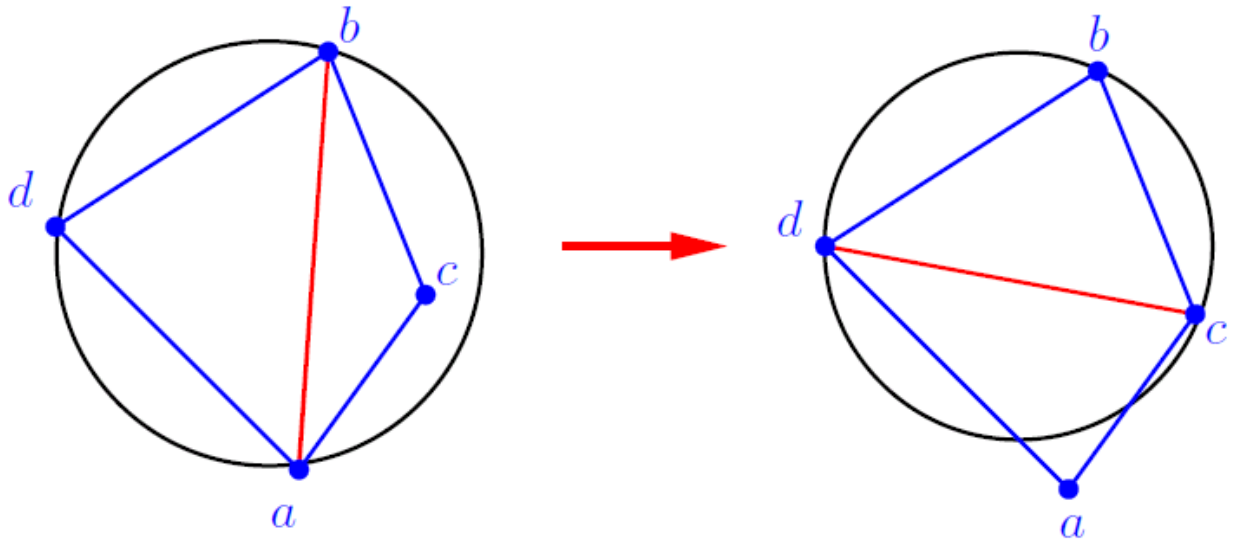
Niech S oznacza chmurę n punktów w R^k.
Triangulacja Delaunay'a jest jedyną triangulacją powłoki wypukłej zbioru S, w której żaden punkt chmury nie leży we wnętrzu kuli opisanej na dowolnym k-sympleksie.



Triangulacja Delaunay’a jest jednoznaczna, gdy żadne cztery punkty chmury nie leżą na jednym okręgu

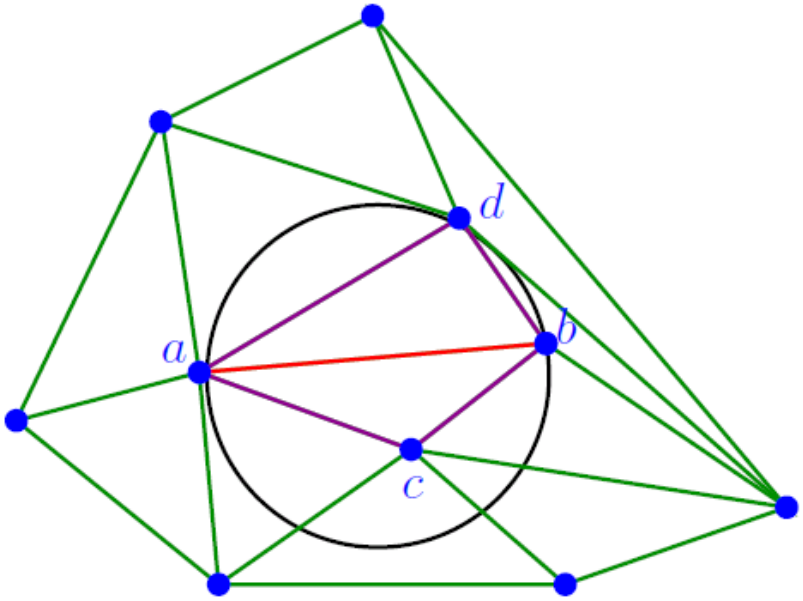


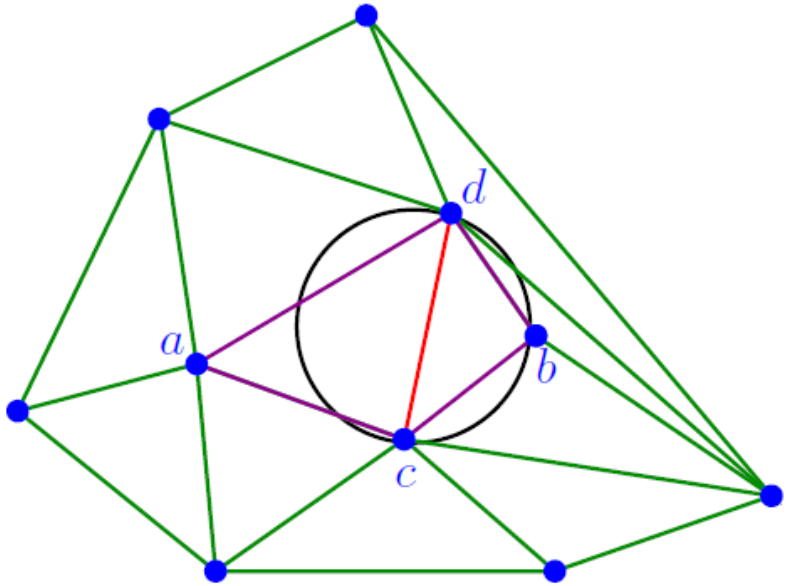
Zamiana przekątnych



ab jest „nielegalna”

cd jest „lokalnie Delaunay’a”


d
a
b
c


d
a
b
c

Dla przestrzeni R^2 można również wykazać
inne własności tej triangulacji, które nie mają odpowiedników
w dowolnej przestrzeni *k*-wymiarowej.

Kryterium lokalnej równoboczności triangulacji
opartej na zadanym układzie węzłów:

Jest to taka triangulacja,
która maksymalizuje minimum sześciu kątów
w każdym dwóch przylegających trójkątów.

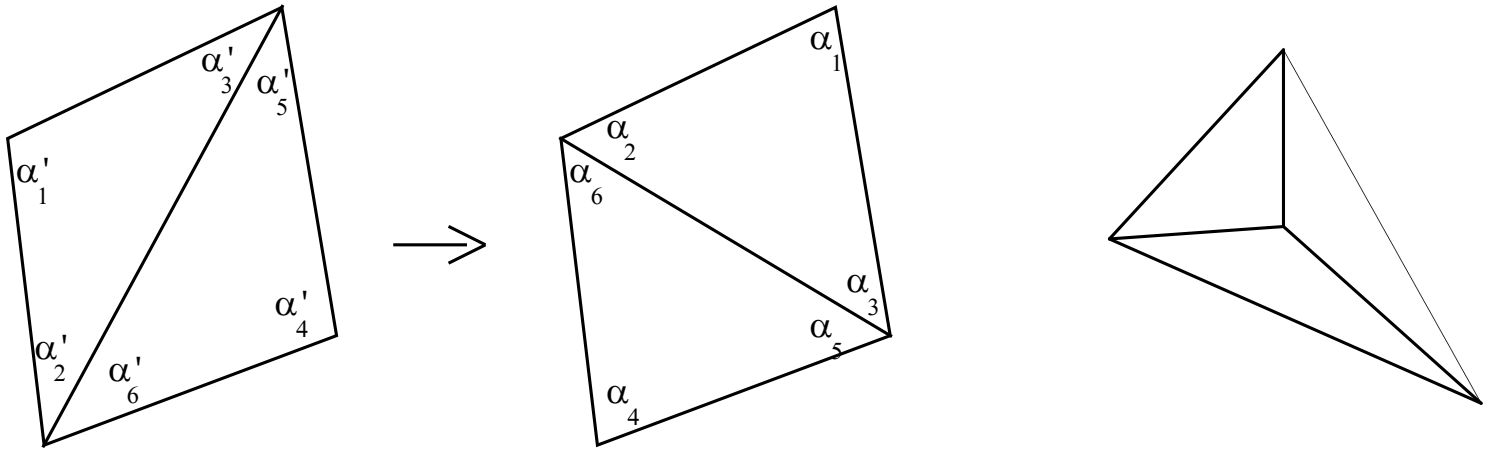
Można wykazać, że ta własność
(nazywana **kryterium max-min**)
prowadzi do triangulacji Delaunay'a.

Twierdzenie:

Niech T będzie dowolną triangulacją powłoki wypukłej chmury S,
w której punkty S są wierzchołkami tej triangulacji.
T jest triangulacją Delaunay'a wtedy i tylko wtedy, gdy
dowolna para przylegających trójkątów Δ_1 i Δ_2 spełnia jeden z warunków:

(a) $\overline{Q} = \overline{\Delta}_1 \cup \overline{\Delta}_2$ nie jest wypukłe,

(b) suma kątów w $\overline{Q}$ oddzielonych krawędzią wspólną $\overline{\Delta}_1$ oraz $\overline{\Delta}_2$
jest większa bądź równa sumie pozostałych kątów w $\overline{Q}$.



Idea metody:

- Wyznacz dowolną triangulację chmury *S*.
- Jeśli wszystkie krawędzie *T* są lokalnie Delaunay'a, zakończ.
- W przeciwnym przypadku zamień „nielegalną" krawędź.
- Powtarzaj aż wszystkie krawędzie będą lokalnie Delaunay'a.

Złożoność obliczeniowa ?

Algorytm iteracyjny triangulacji

Idea Bowyera i Watsona (rok 1981)

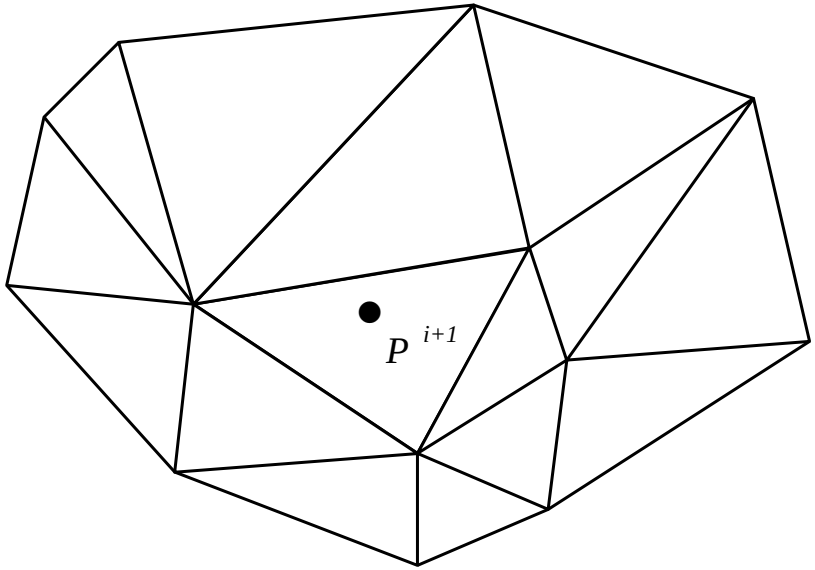
Pierwszym krokiem algorytmu jest skonstruowanie
powłoki pokrywającej punkty chmury.

Można to zrealizować poprzez utworzenie trójkąta
(dodając trzy nowe punkty pomocnicze),
który zawiera w swoim wnętrzu wszystkie punkty chmury

Jest to początkowa triangulacja T_0.

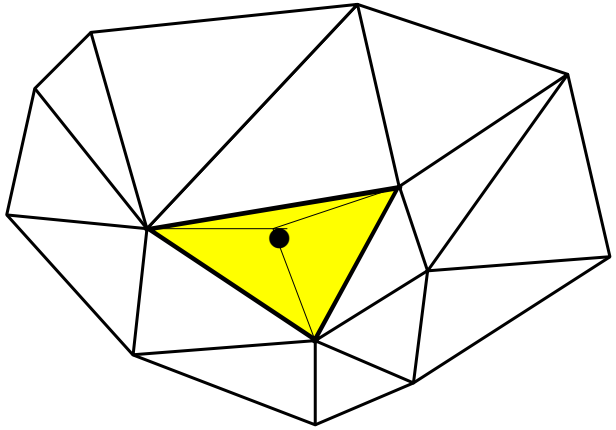
Kolejne triangulacje T_{i+1} mogą być konstruowane na dwa sposoby.

Triangulacja T_i i nowo wprowadzany punkt P_{i+1}



SPOSÓB I:

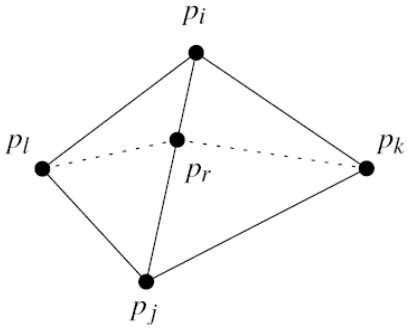
- Do istniejącej triangulacji Delaunay'a T_i dorzucany jest punkt P_{i+1}.



- Wyszukany zostaje trójkąt Δ_k, którego wnętrze zawiera ten punkt.

- Δ_k dzieli się na trzy trójkąty, których wierzchołkiem jest P_{i+1}.

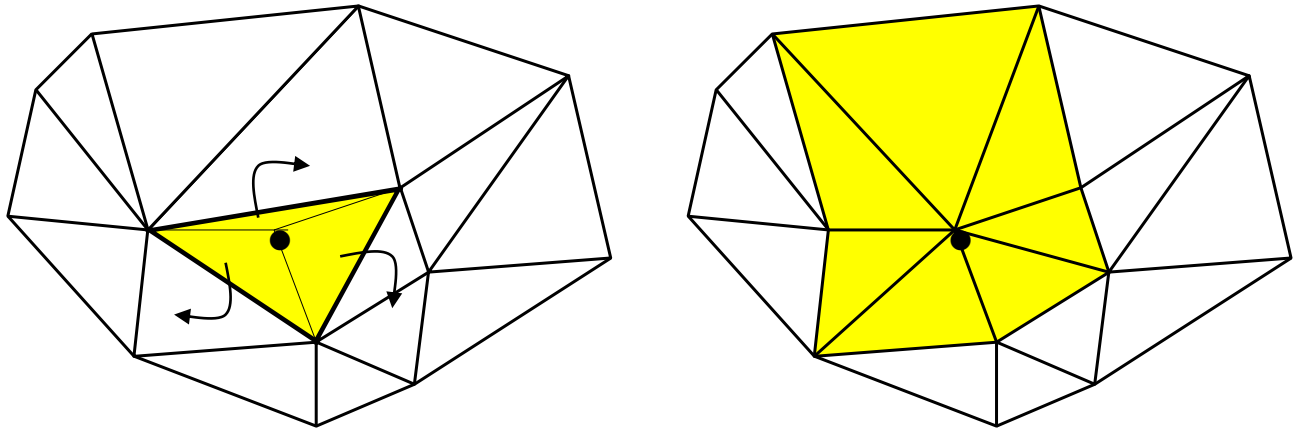
Specjalnego traktowania wymaga przypadek,
gdy nowo wprowadzany punkt P_{i+1}
znajdzie się na którejś krawędzi istniejącej triangulacji T_i.

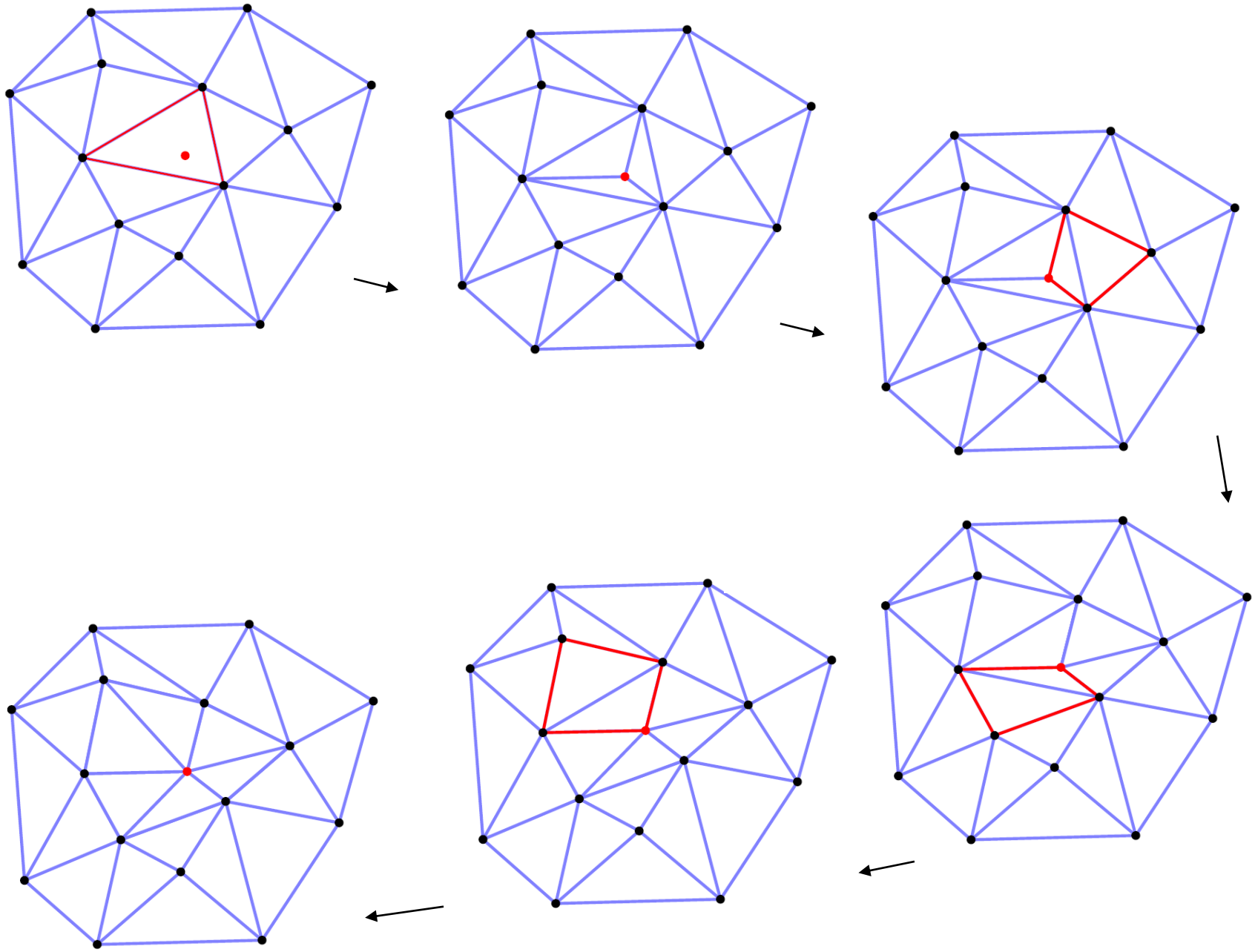


SPOSÓB I:

- Następnie przeszukuje się trójkąty sąsiednie:
  jeśli w czworoboku powstałym
  z danego trójkąta oraz mu przyległego
  nie spełnione jest kryterium (np. kątów, kuli),
  to dokonuje się zmiany przekątnej.

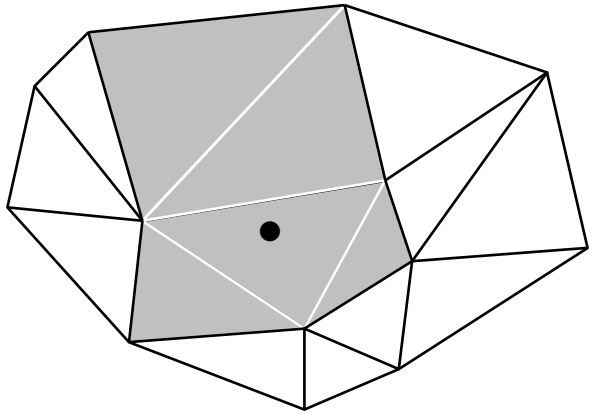
- Systematyczne weryfikowanie dalszych przylegających trójkątów doprowadza do triangulacji T_{i+1} Delaunay'a.





SPOSÓB II:

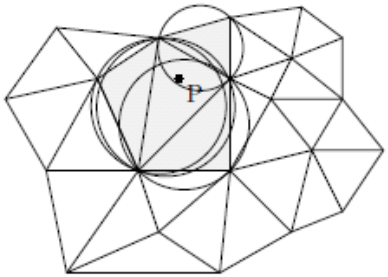
- Do istniejącej triangulacji Delaunay'a T_i dorzucany jest punkt P_{i+1}.



- Poszukuje się wszystkich trójkątów, których koła opisane zawierają ten punkt.

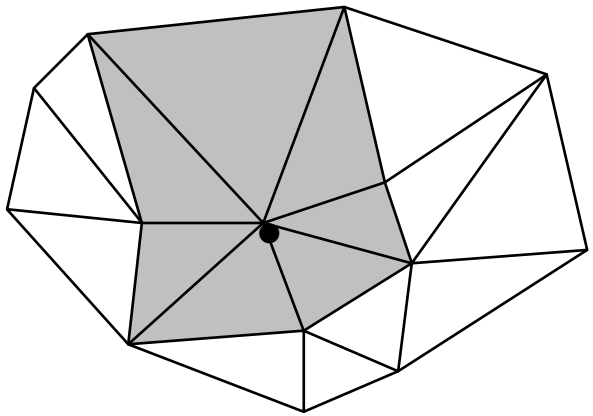
Tworzą one zbiór: $T^* = \{ \Delta \in T_i : d(P_{i+1}, S_\Delta) < R_\Delta \}$

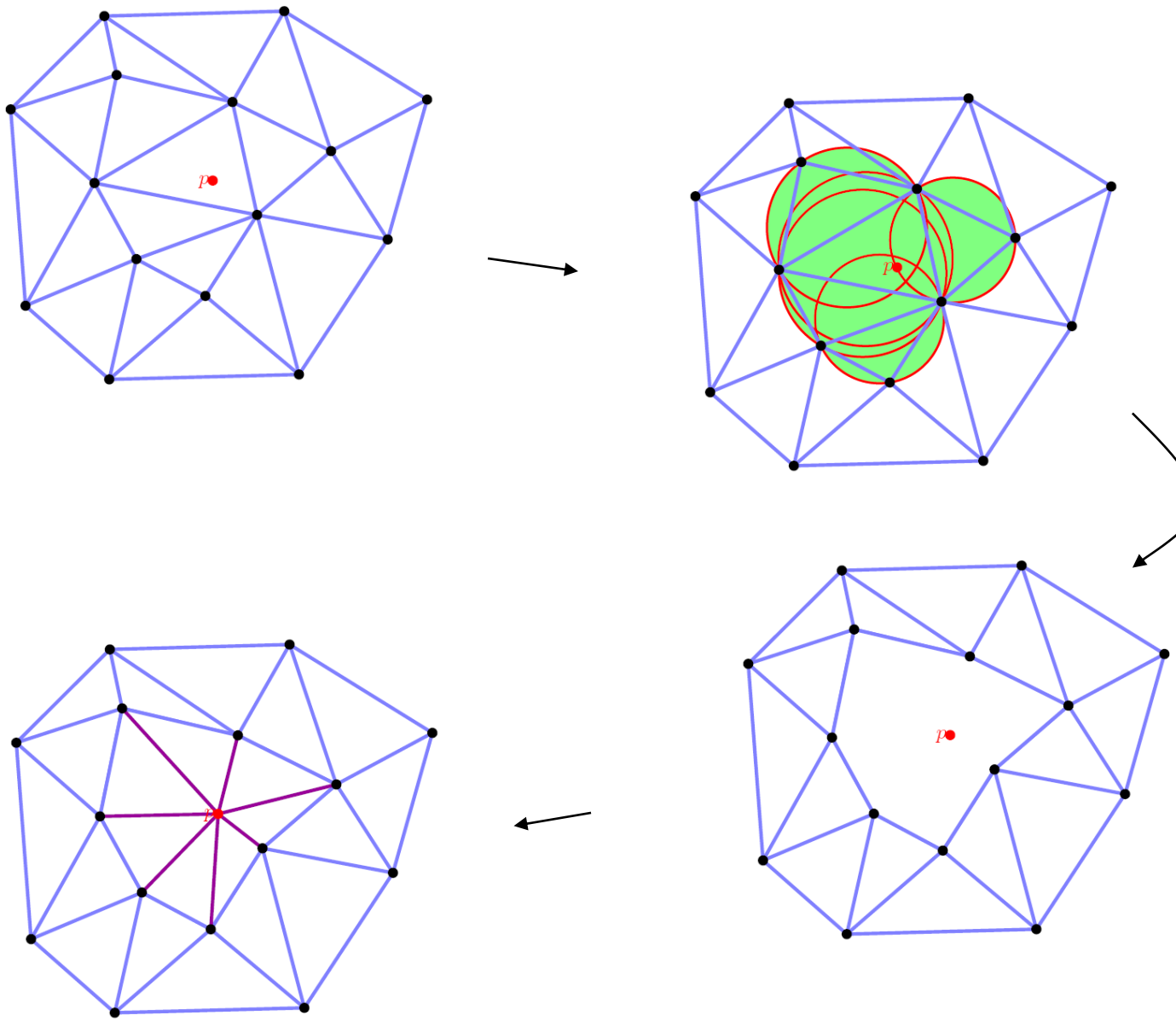
R_Δ - promień koła opisanego,
S_Δ - środek koła opisanego



SPOSÓB II:

- Trójkąty z T^* zostają usunięte z triangulacji T_i.
- Wnęka γ powstała po ich usunięciu jest spójnym obszarem wielobocznym zawierającym P_{i+1}. Punkt P_{i+1} jest widzialny z wszystkich punktów brzegu $\partial\gamma$, czyli każdy wierzchołek $\partial\gamma$ może być z nim połączony odcinkiem leżącym wewnątrz wnęki

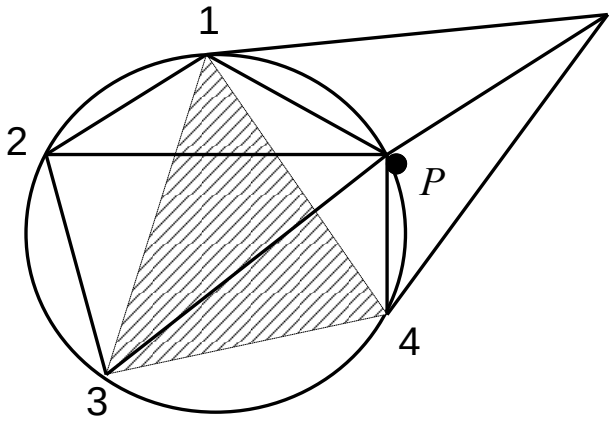



p
p
p
p

Trudność leży w teście przynależności do koła opisanego

Złe ustawienie tego testu doprowadzić może do:

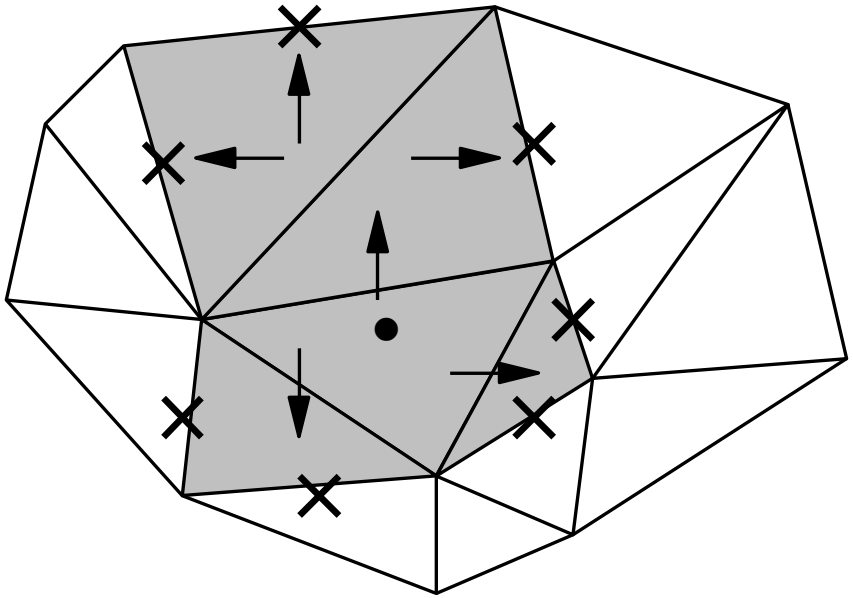
Punkt *P* nie „znalazł” się wewnątrz okręgu opisanego na trójkącie $\Delta(1,3,4)$
i tym samym element ten nie został usunięty, w przeciwieństwie do pozostałych trójkątów opartych na punktach leżących na okręgu.



⇓

zachodzenie na siebie elementów i nieciągłość triangulacji

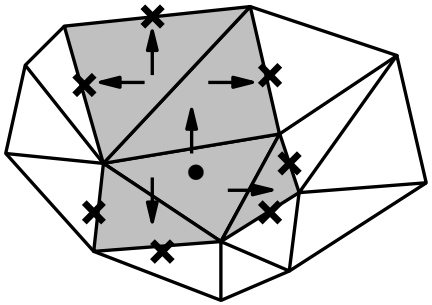
Poszukiwanie podobszaru do retriangulacji poprzez sąsiedztwo topologiczne



- znajdź trójkąt Δ_k zawierający wprowadzany punkt P_{i+1}
  i wstaw jego numer na listę elementów do usunięcia,

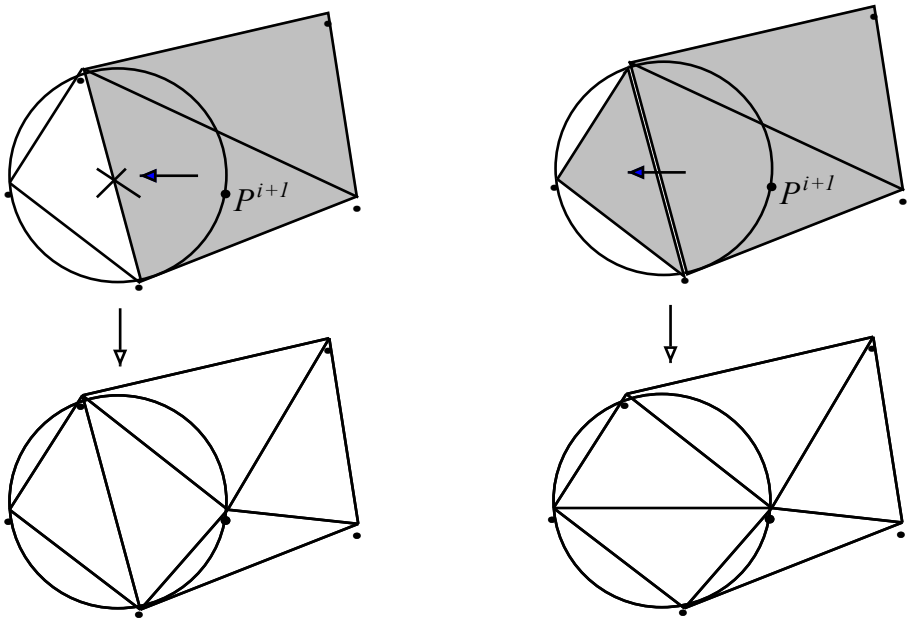
- wstaw na stos numery elementów topologicznie sąsiednich do Δ_k,

- dopóki stos nie jest pusty:
  - pobierz numer elementu j ze stosu i sprawdź, czy $P_{i+1} \in C(\Delta_j)$
  - jeśli tak:
    wstaw numer trójkąta na listę elementów do usunięcia,
    wstaw na stos numery elementów sąsiednich,
    które nie były dotychczas przeglądane.

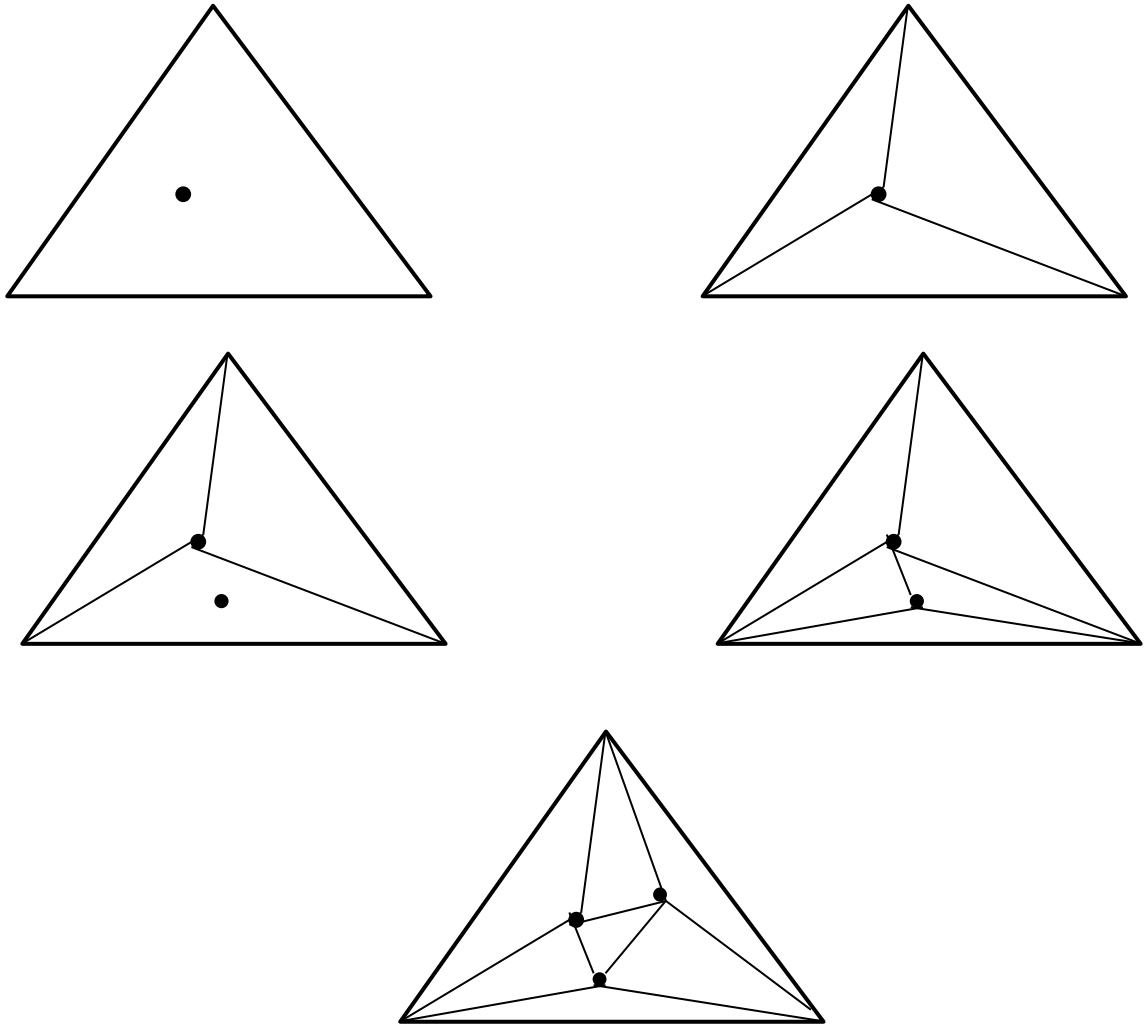


Jeśli dla zadanej precyzji obliczeń okaże się, że koło opisane na danym trójkącie nie zawiera dodanego punktu, elementy sąsiednie nie zostaną zmodyfikowane.

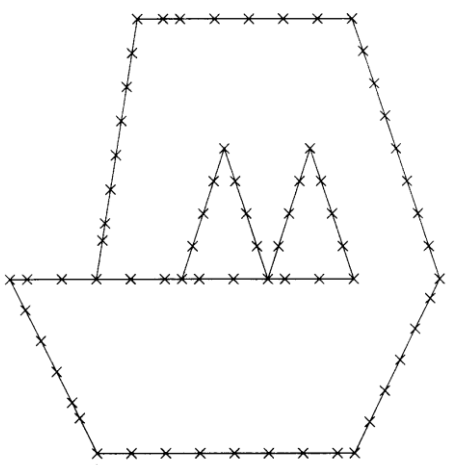
W przypadku punktów leżących na tym samym okręgu prowadzi to do równoważnych konstrukcji.

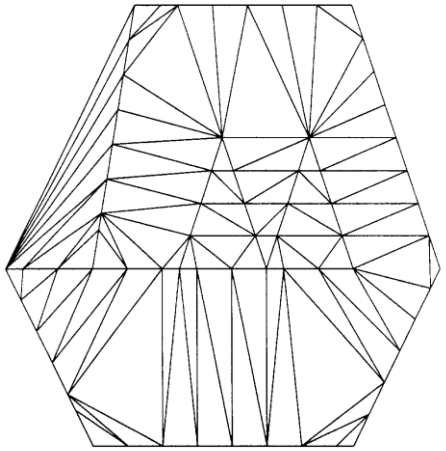


Algorytm iteracyjny triangulacji



Po wprowadzeniu do triangulacji wszystkich punktów chmury,
usuwamy trójkąty których choć jeden wierzchołek
jest punktem pomocniczym
(wierzchołkiem trójkąta zawierającego wszystkie punkty chmury).

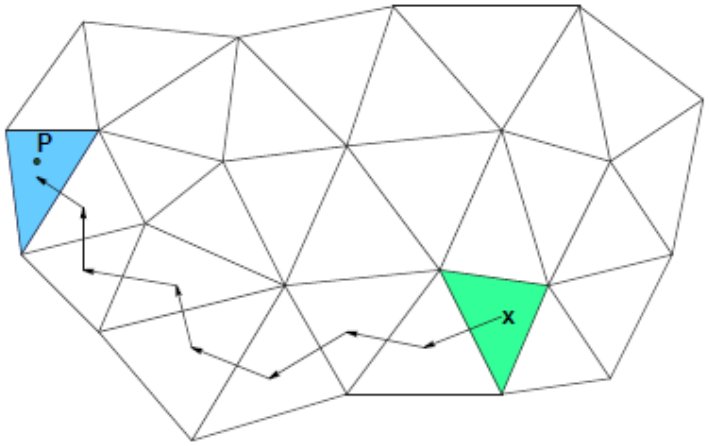


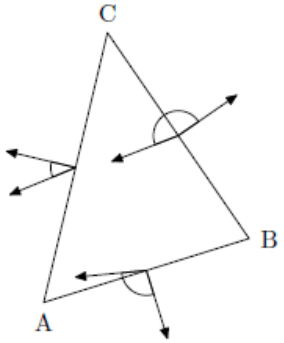


Efektywność algorytmu zależy też od sposobu lokalizacji trójkąta zawierającego dodawany punkt

Przykładowo:

Ustal startowy trójkąt jako aktualny,
Dopóki aktualny trójkąt nie zawiera zadanego punktu:
wybierz krawędź aktualnego trójkąta w kierunku punktu
ustal trójkąt sąsiadujący poprzez tę krawędź jako aktualny





Wybór trójkąta startowego?