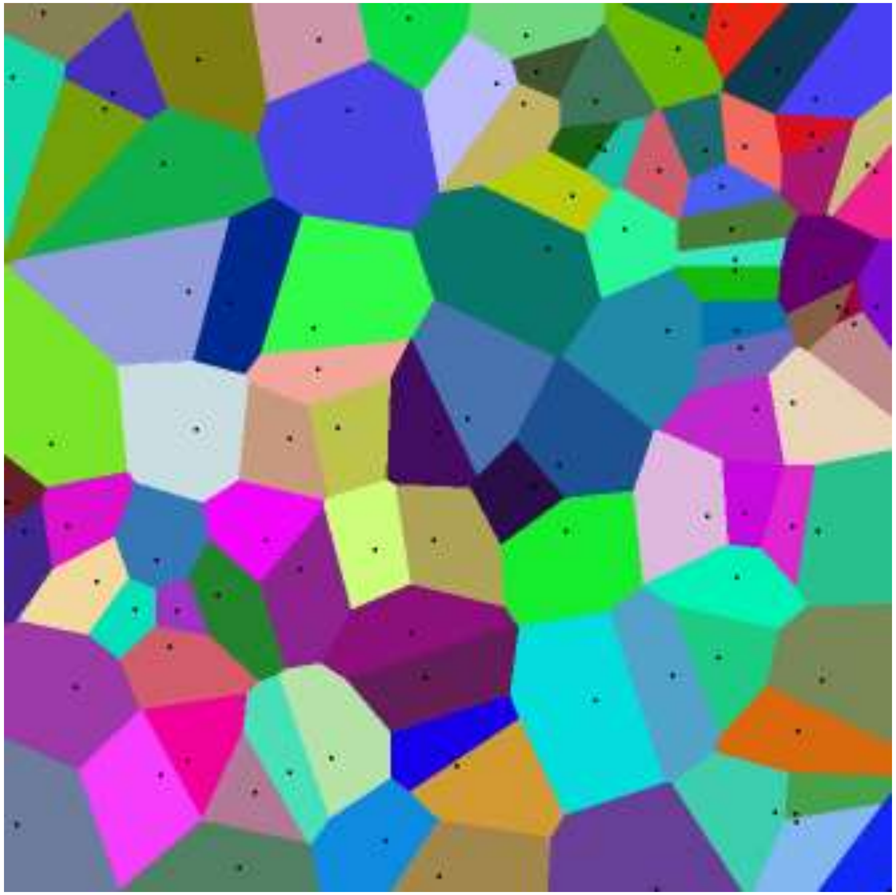
Diagram Voronoi

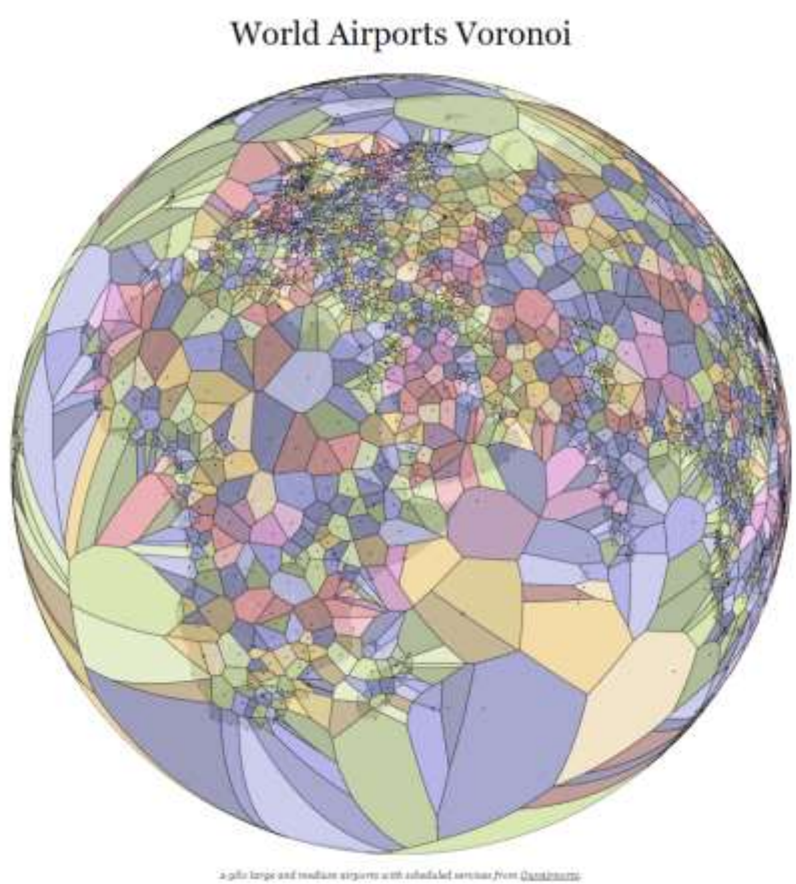
Diagram Voronoi – dla zbioru *n* punktów podział płaszczyzny na *n* obszarów, w taki sposób, że każdy punkt w dowolnym obszarze znajduje się bliżej określonego punktu ze zbioru *n* niż od pozostałych *n-1* punktów

Diagram Voronoi w metryce euklidesowej



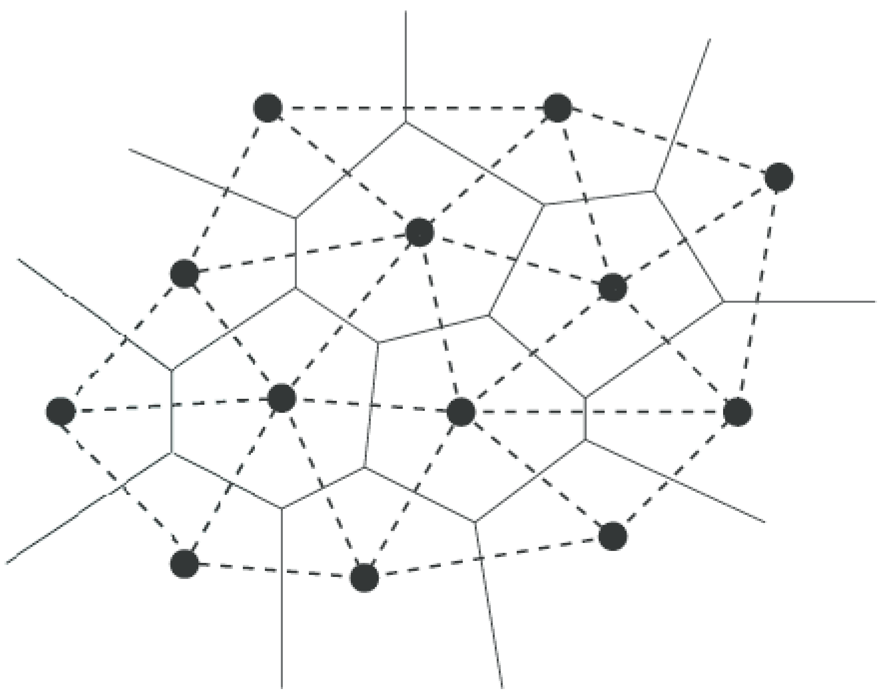
Przykładowe Zastosowania

Knuth’s Post Office Problem – znając lokalizację placówek poczty która jest najbliżej twojego domu?



Przykładowe Zastosowania cd.

Tworzenie triangulacji Delaunay – możemy łatwo otrzymać triangulację Delaunay łącząc początkowe punkty których obszary mają wspólną krawędź.



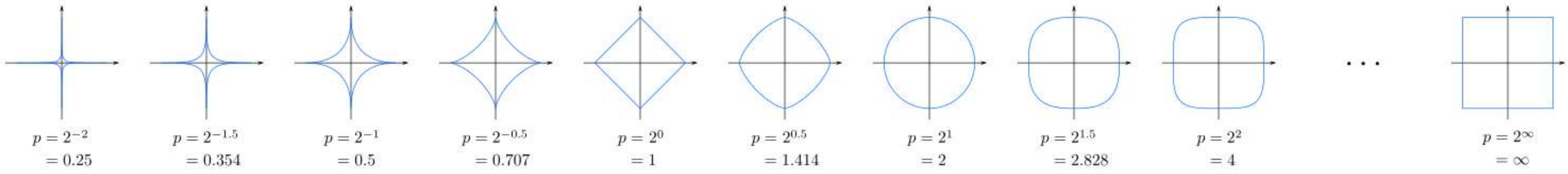
Odległość Minkowskiego

Odległość Minkowskiego (stopnia p) dla punktów:

$$X = (x_1, x_2, \ldots, x_n) \qquad Y = (y_1, y_2, \ldots, y_n) \in \mathbb{R}^n$$

Jest zdefiniowana jako:

$$D(X,Y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{1/p}$$

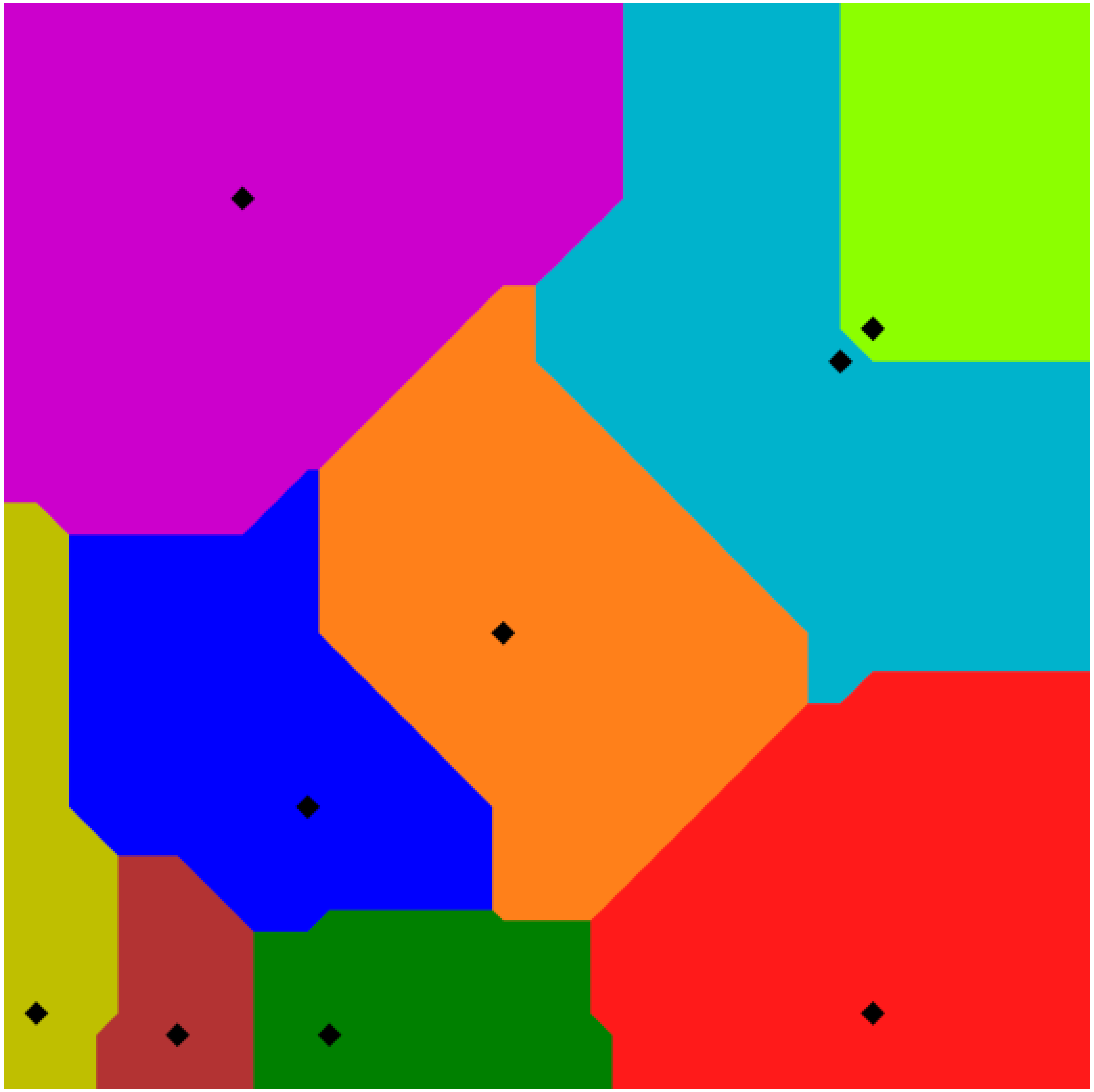


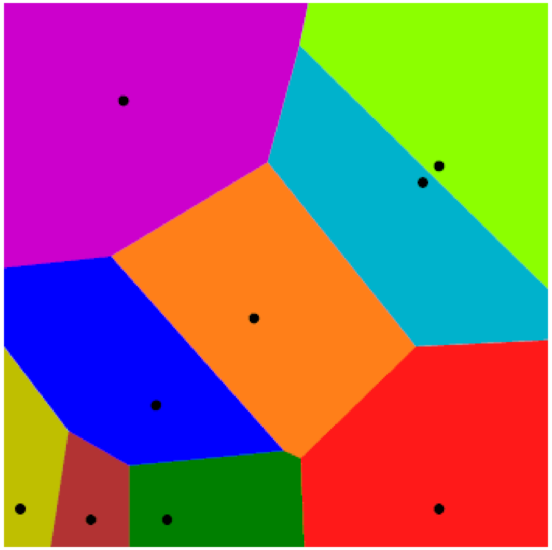
Metryka Miasta

Odległość w metryce miasta to odległość minkowskiego stopnia 1. W naszym wypadku rozpatrujemy punkty na płaszczyźnie co upraszcza wzór dystansu do

$$D(a, b) = |x_a - x_b| + |y_a - y_b|$$

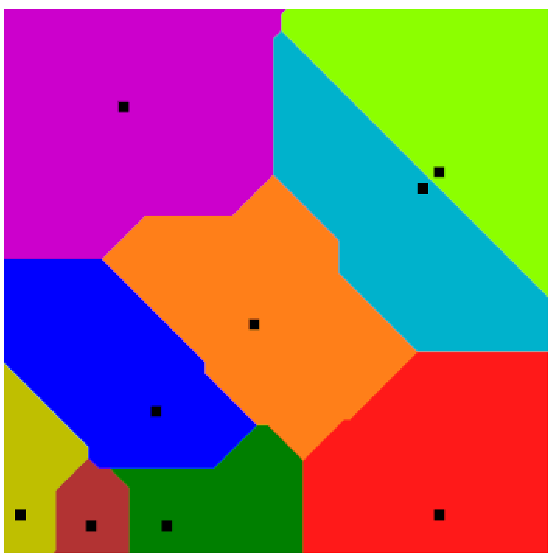
Diagram Voronoi w metryce miasta





Euklidesowa *(p=2)*

Taksówkowa *(p=1)*



Maksimum *(p->inf)*

(p=0.707)

Algorytmy Obliczające Diagram Voronoi

Algorytm Fortuny – algorytm zamiatania.

Algorytm uzyskania diagram za pomocą triangulacji – mając stworzoną triangulajce znajdź punkty rownoodległe od wierzchołków trójkątów, te punkty to wierzchołki diagramu.

Algorytm na znajdowanie diagramu voronoi w metryce miasta

Poniżej przedstawimy algorytm na szukanie diagramu voronoi. Użyliśmy metryki miasta w której dystans jest liczony jako: $|x_a - x_b| + |y_a - y_b|$

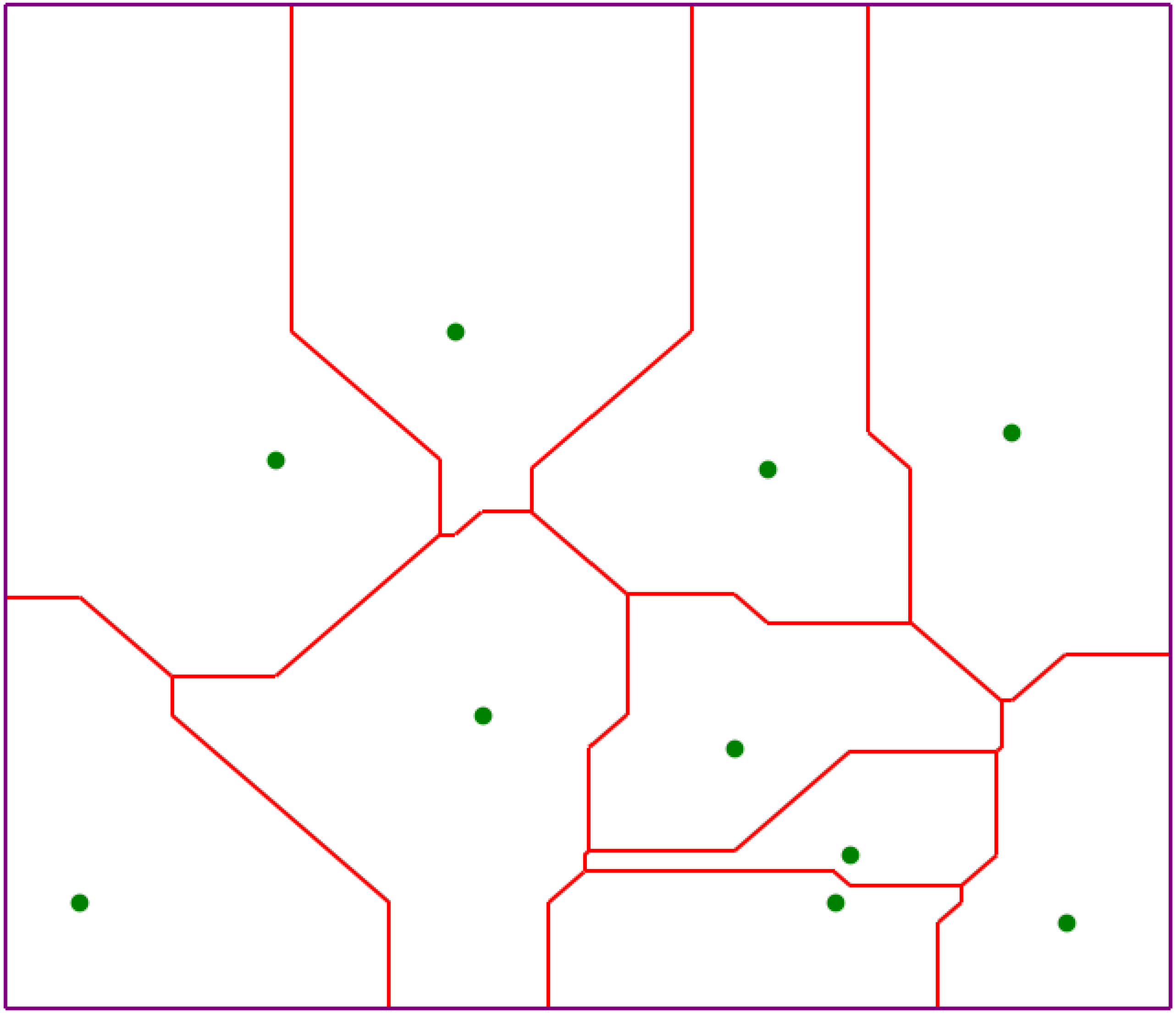
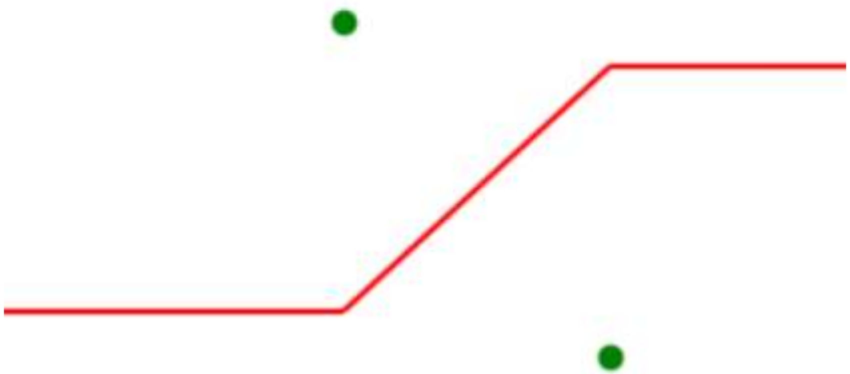
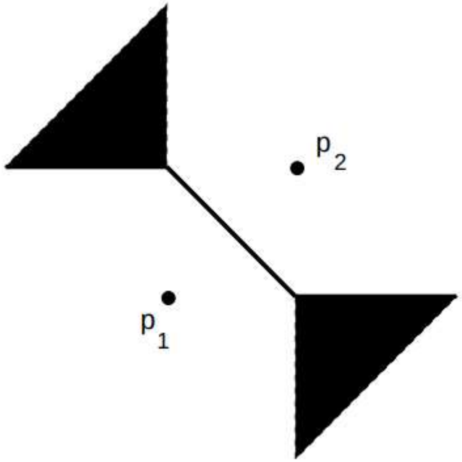


Diagram voronoi w metryce miasta

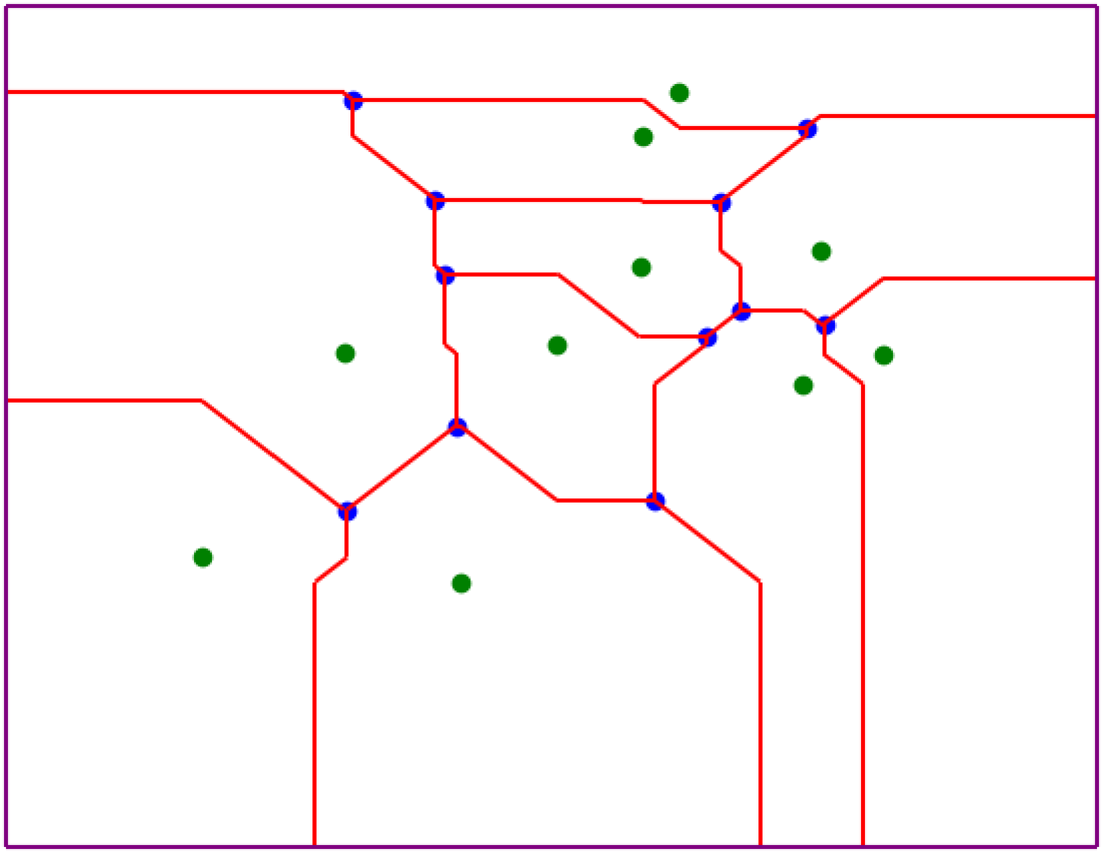
Symetralną dwóch punktów a i b będziemy nazywać zbiór linii na płaszczyźnie dla których każdy punkt leży w równej odległości od punktów a i b. Linie te będą zawsze leżały pod kątami które są wielokrotnościami 45 stopni (jeśli na przykładach będzie inaczej to tylko i wyłącznie ponieważ skala osi x nie będzie równa skali osi y)



Wykluczamy przypadek gdy punkty leżą na wierzchołkach kwadratu. Wtedy symetralna jest płaszczyzną



Nasz algorytm będzie najpierw szukał wierzchołków voronoi, potem poprzez umiejętne ich łączenie będzie znajdywał krawędzie



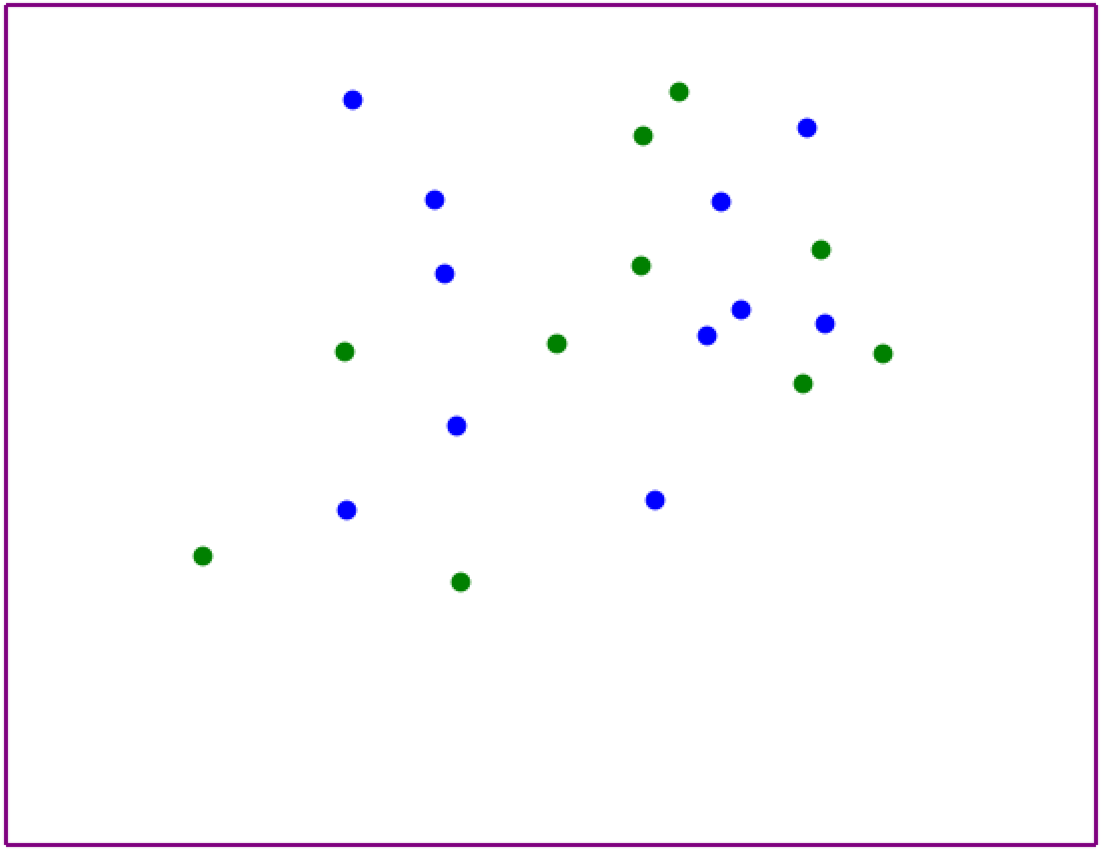
Na niebiesko zaznaczone są wierzchołki voronoi, a na czerwono krawędzie. Punkty zielone to punkty wejściowe

Nasz algorytm znajduje wierzchołki voronoi wykorzystując fakt iż każdy z tych wierzchołków leży w równej odległości od 3 punktów ze zbioru wejściowego i są to najbliższe punkty dla tego wierzchołka.

1. Dla każdej trójki punktów ze zbioru wejściowego sprawdź czy istnieje punkt równoodległy od nich trzech (proponowany wierzchołek)
2. Jeśli istnieje sprawdź czy istnieje punkt ze zbioru wejściowego który leży bliżej proponowanego wierzchołka niż dowolny z trójki punktów.
3. Jeśli nie istnieje taki punkt dodaj proponowany wierzchołek do zbioru wierzchołków voronoi. Zapisujemy także informacje o trzech punktach które posłużyły do stworzenia wierzchołka voronoi. (poźniej będzimy nazywać je punktami tworzącymi)

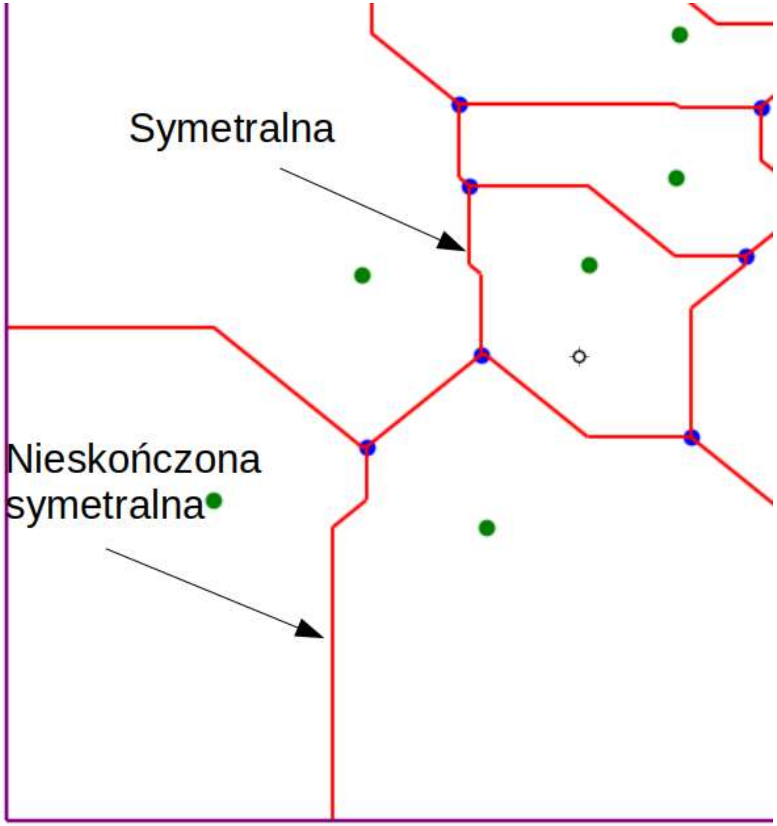
Jak znaleźć punkt równoodległy od 3 punktów w metryce miasta?

Idea jest bardzo prosta. W dla zadanych punktów (a,b,c) konstruujemy dowolne 2 symetralne pomiędzy tymi punktami oraz szukamy punktu przecięcia tych dwóch symetralnych. Jeśli takowy istnieje to jest on równoodległy od 3 zadanych punktów

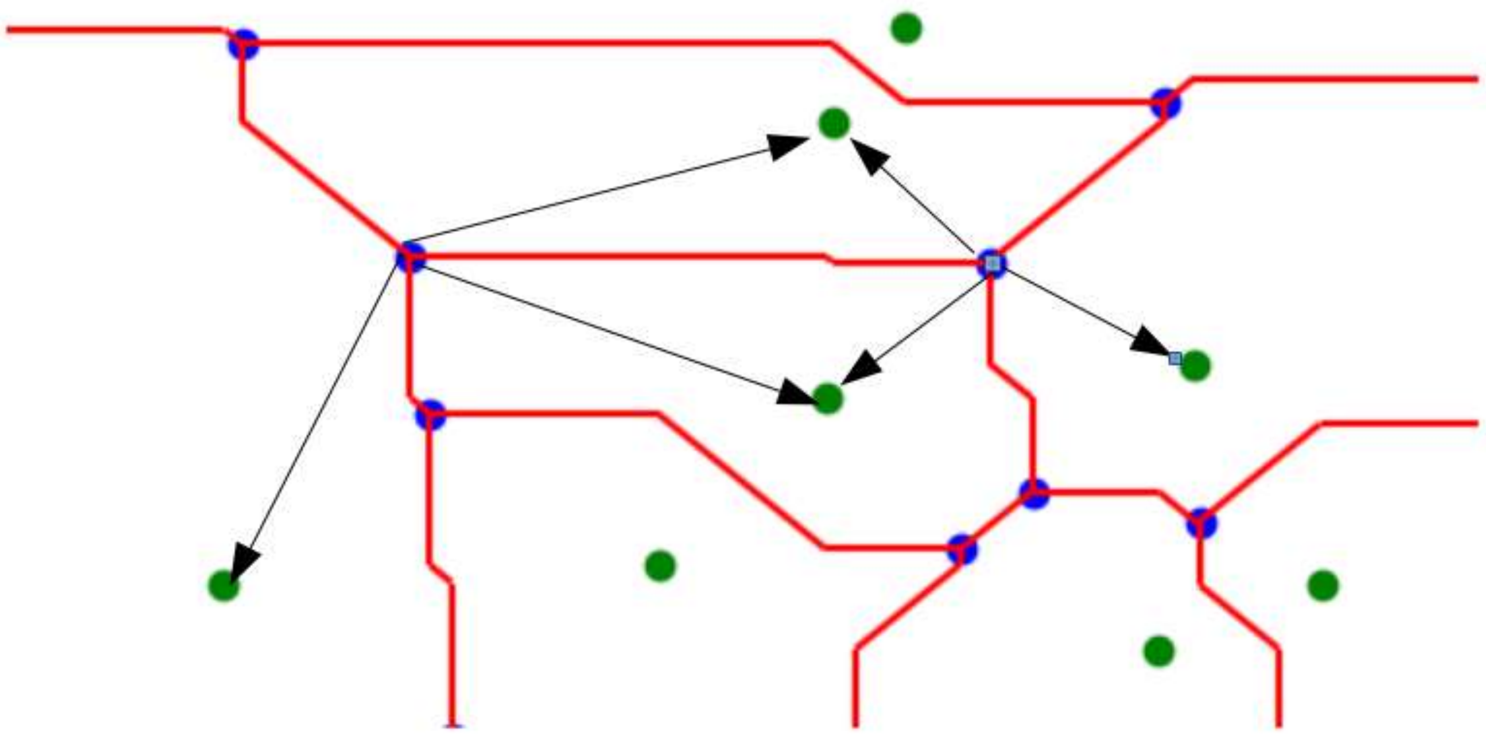


W ten prosty sposób znajdujemy wszystkie wierzchołki voronoi (zaznaczone na niebiesko)

Następnie szukamy krawędzi voronoi. Dzielą się one na dwa rodzaje
1. Symetralne pomiędzy dwoma punktami ze zbioru wejściowego, kończące się na dwóch wierzchołkach voronoi
2. Nieskończone symetralne pomiędzy dwoma punktami ze zbioru wejściowego, zaczynające się w wierzchołku voronoi. Z oczywistych względów u nas reprezentowane jako kończące się na granicy obszaru na którym konstruowaliśmy diagram

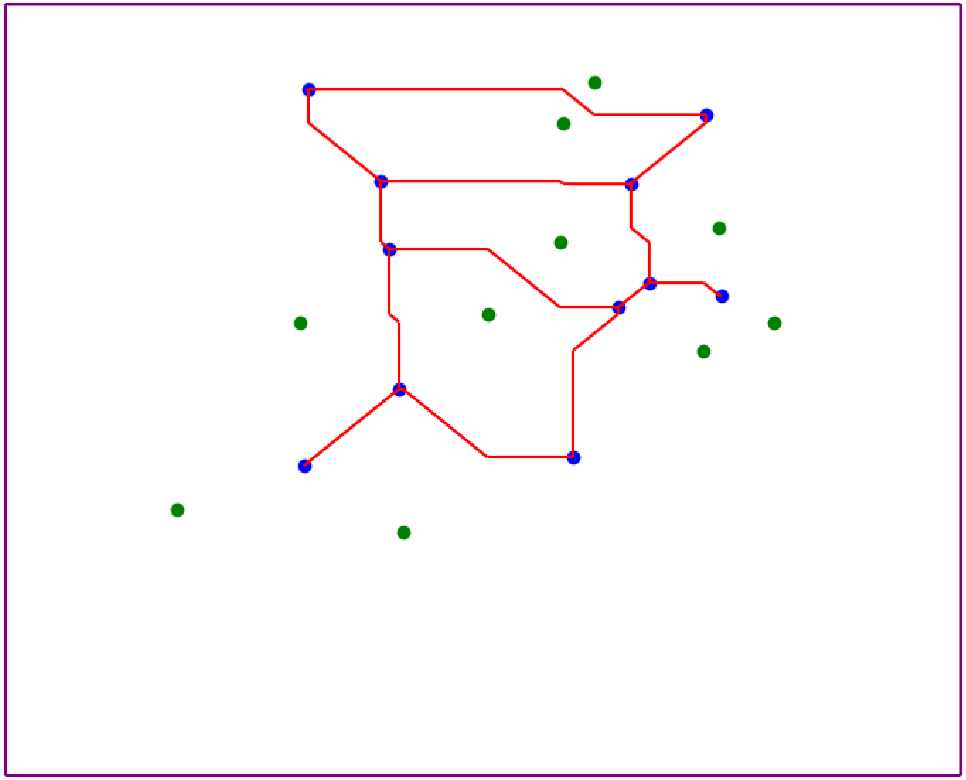


Na początek zaczynamy od szukania skończonych symetralnych. Końce takiej symetralnej to będą wierzchołki voronoi które zostały stworzone za pomocą dwóch wspólnych punktów tworzących ze zbioru wejściowego i trzeciego różnego



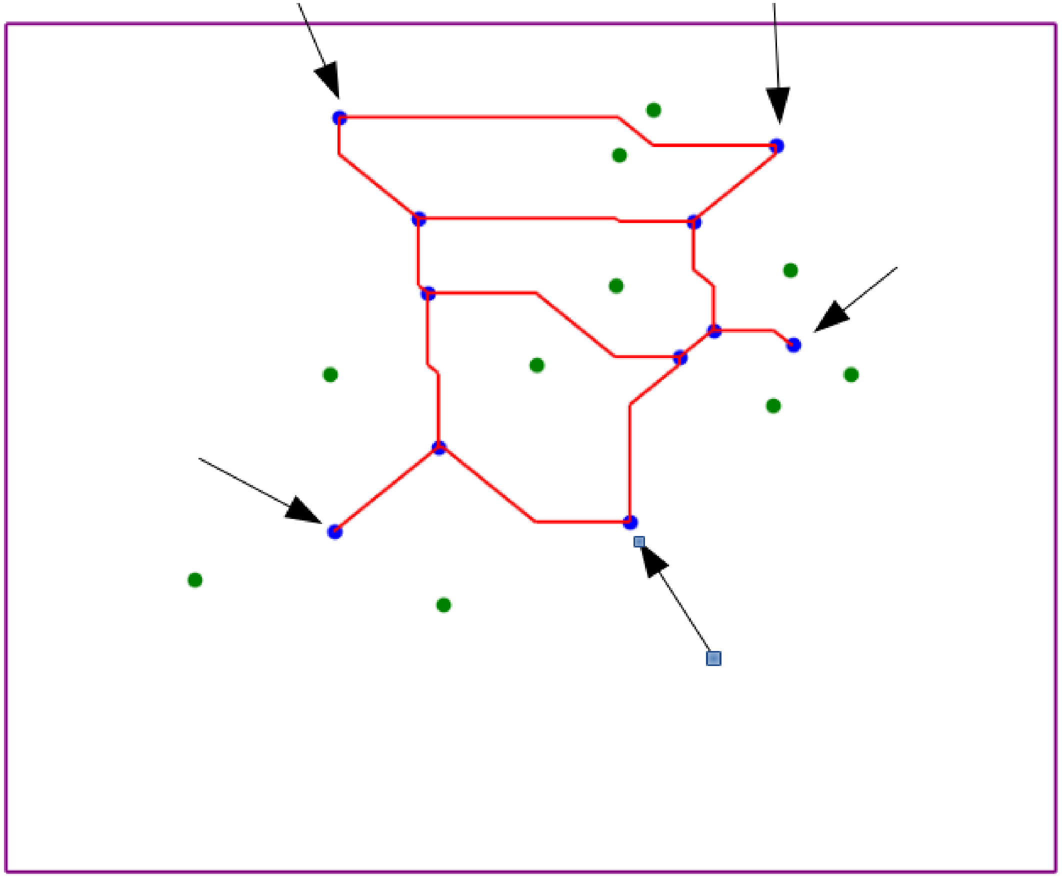
Strzałkami zaznaczyłem za pomocą których punktów te dwa wierzchołki zostały stworzone. Widzimy że punkty są wspólne więc wiemy że pomiędzy tymi punktami będzie leżała krawędź voronoi

Przebiegamy po każdej parze z wierzchołków Voronoi. Jeśli mają dwa wspólne punkty tworzące to konstruujemy symetralną za pomocą tych dwóch punktów która kończy się na wierzchołkach voronoi. Do każdego wierzchołka zapisujemy informacje o stworzonej krawędzi.



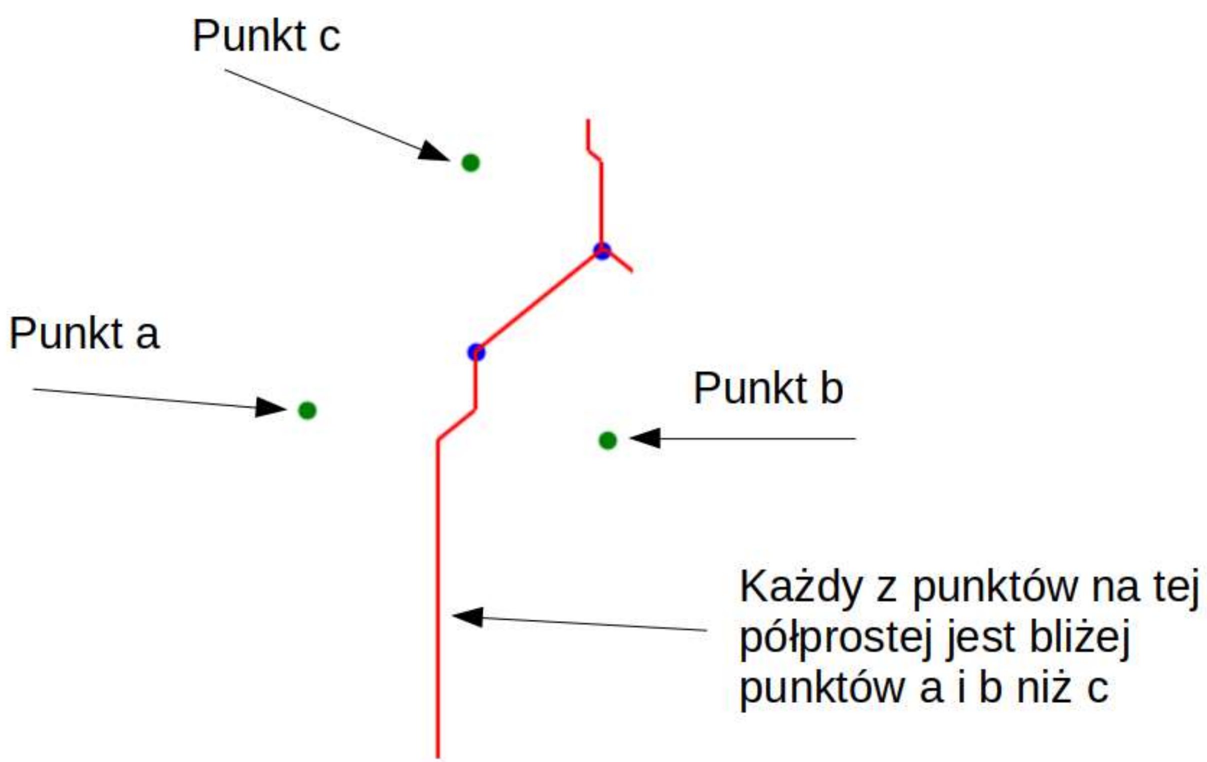
Teraz pozostały nam do stworzenia jeszcze nieskończone półproste

Przebiegamy ostatni raz po wierzchołkach voronoi. Patrzymy czy skonstruowaliśmy symetralną pomiędzy każdą parą punktów (a,b) tworzących wierzchołek. Jeśli takowa symetralna nie istnieje jest to dla nas informacją że z tego wierzchołka będzie wychodziła nieskończona symetralna stworzona za pomocą punktów (a,b). Dodajemy taką symetralną do zbioru krawędzi voronoi

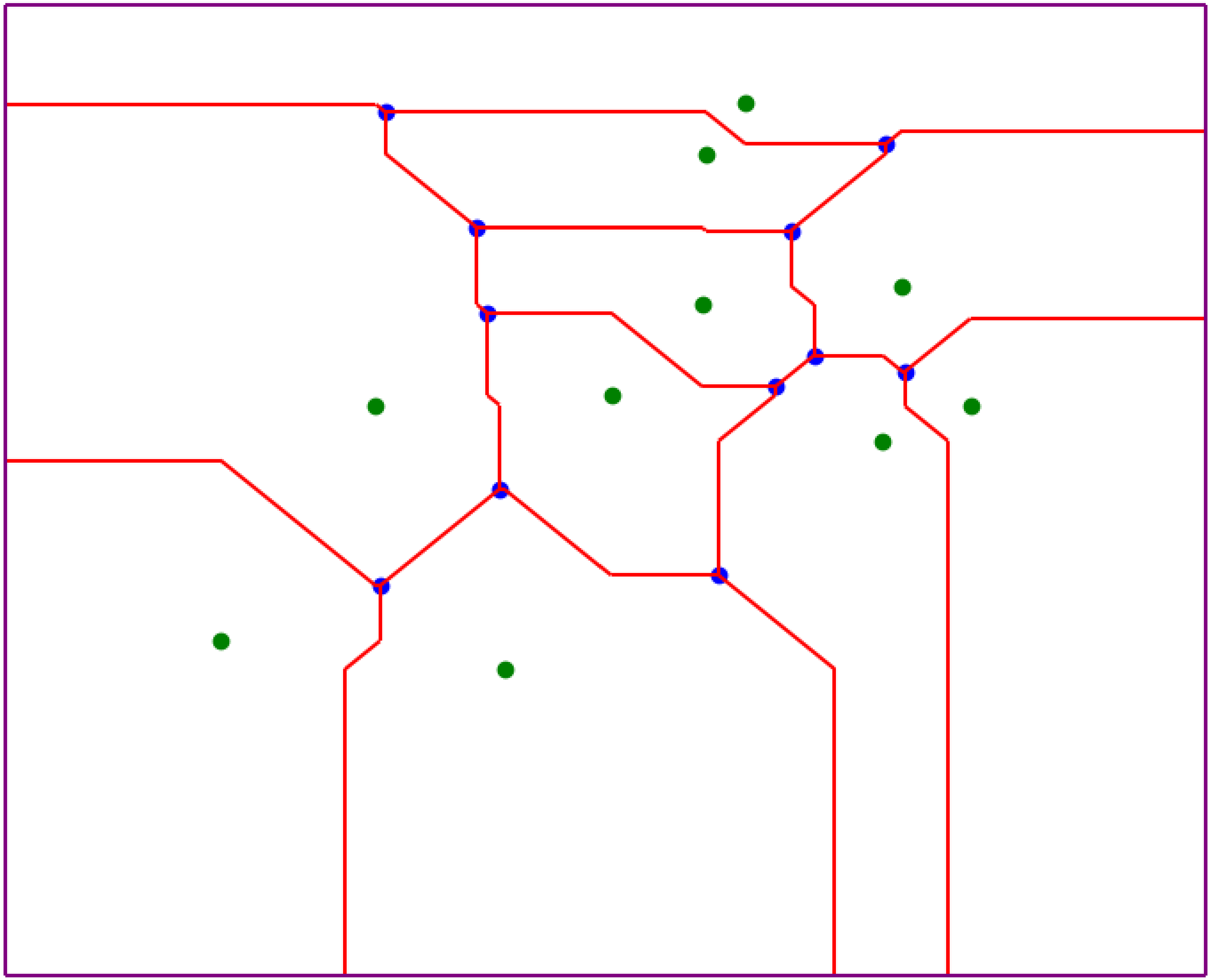


Zaznaczone wierzchołki które będą początkiem nieskończonych symetralnych.

Kolejnym problemem będzie znalezienie którą część nieskończonej symetralnej wziąć. Jeden ograniczający punkt stworzy nam przecież dwie nieskończone symetralne!
Do rozwiązania tego problemu posłużą nam punkty tworzące.
Niech (a,b,c) to będą punkty tworzące, Za pomocą punktów a i b została stworzona nieskończona symetralna. Rozwiązaniem tego problemu jest punkt c. Widzimy że każdy punkt nieskończonej symetralnej musi być bliżej punktów a i b niż c. Musimy więc wziąć dowolny punkt z obu symetralnych i wybrać ten dla którego punkty a i b są bliższe.



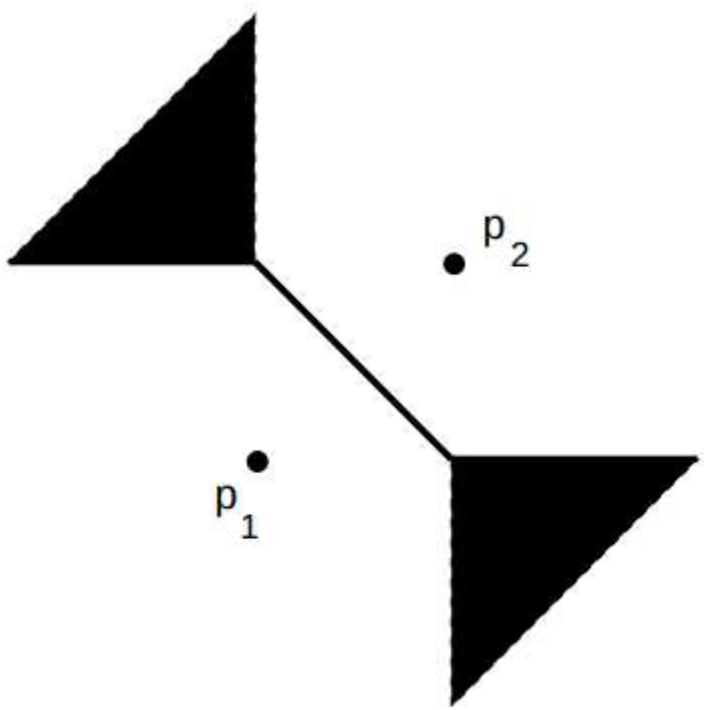
W ten sposób skonstruowaliśmy cały diagram voronoi!



Problemy algorytmu

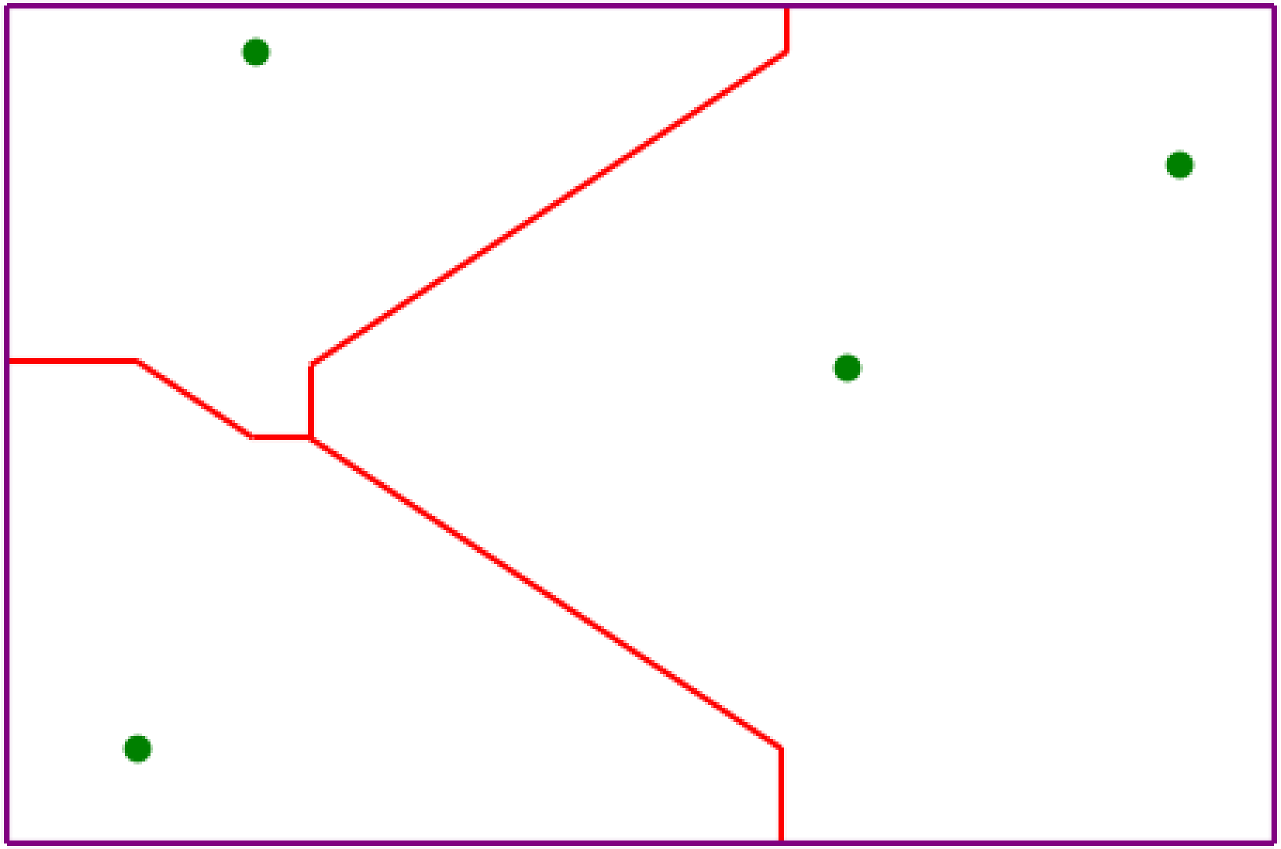
1. Algorytm jest wolny. Szukanie krawędzi ma złożoność $O(n^2)$ (za pomocą odpowiednich struktur danych możliwe byłoby uzyskanie złożoności O(n), jednak nie zrobiliśmy tego w naszej implementacji), ale szukanie wierzchołków ma złożoność $O(n^4)$. Jest to spowodowane najpierw szukaniem każdej trójki z punktów wejściowych ($O(n^3)$), a potem sprawdzanie czy znaleziony punkt jest na pewno wierzchołkiem voronoi (O(n)). Ten proces można byłoby przyspieszyć za pomocą triangulacji Delonuaya (na metryce miasta oczywiście). Dałaby nam ona informację o tym jakie punkty tworzą wierzchołki voronoi i zredukowałoby złożoność z $O(n^4)$ do O(n)

2. Algorytm nie działa dla przypadku gdy dwa punkty położone blisko siebie znajdują się na wierzchołkach kwadratu $|x_a - x_b| == |y_a - y_b|$



Każdy z punktów leżących na czarnej płaszczyźnie jest równoodległy od punków p_1 i p_2

3. Algorytm nie znajduje krawędzi voronoi gdy dla zadanego punktu nie wyznaczymy żadnego wierzchołka voronoi. Taki przypadek jest mało prawdopodobny gdy punktów jest więcej niż tylko kilka, jednak może wystąpić.



Nie istnieje wierzchołek voronoi dla punktu najbardziej po lewej stronie

Dziekujemy za uwagę

Kacper Sala

Kuba Wydra