

**Uniwersytet Jagielloński**  
Wydział Matematyki i Informatyki

**Piotr Helm**  
Nr albumu: 1132708

---

ANALIZA ALGORYTMÓW BUDOWANIA  
CORESETU DLA PROBLEMU K-MEANS

---

**Praca licencjacka**  
**na kierunku INFORMATYKA ANALITYCZNA**

Praca wykonana pod kierunkiem  
**dr Iwona Cieřlik**  
Instytut Informatyki Analitycznej

Kraków, wrzesień 2020

### Streszczenie

W pracy przedstawimy stan wiedzy na temat budowania coresetów w kontekście problemu  $k$ -means. W szczególności omówimy techniki konstrukcji coresetów takie jak geometryczna dekompozycja oraz losowe próbkowanie z [13].

Celem pracy jest implementacja technik budowania coresetów oraz przeprowadzenie eksperymentów, które zbadają jakość uzyskanych coresetów [13] [5]. W tym celu porównamy wyniki działania algorytmu Lloyd'a rozwiązującego problem  $k$ -means na całym zbiorze danych z jego działaniem na uzyskanych różnymi metodami coresetach.

## 0. Spis treści

<b>1</b>	<b>Wstęp</b>	<b>2</b>
<b>2</b>	<b>Notacja i niezbędne definicje</b>	<b>4</b>
2.1	K-means . . . . .	4
2.2	Coreset . . . . .	5
<b>3</b>	<b>Lightweight Coreset</b>	<b>8</b>
3.1	Lightweight coreset . . . . .	8
3.2	Konstrukcja . . . . .	9
3.3	Analiza . . . . .	10
<b>4</b>	<b>Geometryczna Dekompozycja</b>	<b>14</b>
4.1	Algorytm Gonzalez’a . . . . .	14
4.2	Konstrukcja kraty wykładniczej . . . . .	16
4.2.1	Szybka aproksymacja dla problemu K-means. . . . .	16
4.2.2	Konstrukcja zbioru dobrych punktów dla $X$ . . . . .	17
4.2.3	Krata wykładnicza. . . . .	19
4.3	Heurystyka single swap . . . . .	21
4.4	Podsumowanie . . . . .	26
<b>5</b>	<b>Analiza Implementacji</b>	<b>29</b>
<b>6</b>	<b>Bibliografia</b>	<b>30</b>

## 1. Wstęp

*More is more* to jedna z podstawowych doktryn związanych z szeroko rozumianym Big Data. Więcej danych to więcej informacji, które analizujemy licząc na poznanie ukrytych zależności. W erze Big Data skalowalność rozwiązań jest szczególnie ważna, dlatego celem wielu naukowców jest dostarczenie kompromisu pomiędzy szczegółowością informacji a wymaganiami pamięciowymi. Tutaj warto zwrócić uwagę na dużą wartość takich rozwiązań w praktycznych zastosowaniach.

*Sketch-and-solve* to popularny paradygmat, który zakłada separację algorytmu agregującego dane od właściwego algorytmu analizującego. Główną ideą jest redukcja danych tak aby rozmiar zbudowanego zbioru nie były zależny od rozmiaru wejściowych danych lub tylko *trochę* od nich zależał. Następnie aplikowany jest właściwy algorytm, który jest mniej zależny od początkowego rozmiaru danych. W rezultacie wykonuje swoją pracę szybciej, a niekiedy nawet lepiej. Dodatkową zaletą jest fakt, że w większości przypadków nie jest konieczna modyfikacja algorytmu analizującego.

Niestety w kontekście tego paradygmatu największym wyzwaniem jest znalezienie kompromisu pomiędzy stratą jakości danych a ich rozmiarem. To jak określamy charakterystykę ważnych informacji jest ściśle zależne od aplikacji danych. Coresety są strukturą algorytmiczną, która ma na celu indentyfikację takich cech oraz określenie akceptowalnego kompromisu dla różnych funkcji celu.

Mówiąc ogólniej, mamy na wejściu zbiór danych  $A \subset U$ , gdzie  $U$  oznacza jakieś uniwersum, zbiór potencjalnych rozwiązań  $C$  pewnego problemu oraz funkcję celu  $f : P(U) \times C \rightarrow \mathbb{R}_{\geq 0}$ , gdzie  $P(U)$  oznacza zbiór potęgowy dla uniwersum. Chcemy znaleźć istotnie mniejszy zbiór danych  $S \subset U$ , dla którego dla każdego potencjalnego rozwiązania  $c \in C$  daje wartość  $f(S, c)$  *dobrze* aproksymujące  $f(A, c)$ . A dokładniej:

$$|f(A, c) - f(S, c)| < \epsilon f(A, c)$$

gdzie  $\epsilon > 0$ .

Algorytmy budujące coresety są aplikowalne do wielu problemów klasteryzacji. W tej pracy skupimy się na konstrukcjach dla problemu  $k$ -means, który należy do klasy problemów *NP-trudnych* [1]. Najpopularniejszym algorytmem heurystycznym dla tego problemu jest algorytm Lloyd'a [11]. Z uwagi na to, że złożoność algorytmu istotnie zależy od rozmiaru wejściowych danych, jest on idealnym kandydatem do optymalizacji poprzez odpowiednią konstrukcję core-setu.

W rozdziale 3 opiszemy budowę *lightweight coresetu* z pracy [5]. Następnie, w rozdziale 4 przedstawimy konstrukcję *coresetu* bazującą na geometrycznej dekompozycji problemu korzystając z badań [13]. W ostatnim, 5 rozdziale omówimy przeprowadzone eksperymenty dla zaimplementowanych konstrukcji coresetów oraz porównamy ich wyniki.

## 2. Notacja i niezbędne definicje

W tej pracy przyjmujemy, że działamy w przestrzeni euklidesowej w skończonym wymiarze  $d > 0$ .

**Definicja 2.1.** Niech  $x = (x_1, \dots, x_d) \in \mathbb{R}^d$ . Normą typu  $l_2$  nazywamy odwzorowanie  $\|\cdot\| : \mathbb{R}^d \rightarrow \mathbb{R}$  określone wzorem:

$$\|x\|_2 = \|x\| = \sqrt{\sum_{i=1}^d x_i^2}$$

Założmy, że mamy dany konkretny problem optymalizacyjny.

**Definicja 2.2.**  $\alpha$ -aproksymacją dla problemu optymalizującego nazywamy wielomianowy algorytm, który dla każdej instancji problemu minimalizacyjnego oblicza rozwiązanie, którego wartość spełnia:

$$A \leq \alpha \cdot OPT$$

gdzie  $A$  jest wartością obliczonego rozwiązania, a  $OPT$  jest wartością optymalnego rozwiązania. Zmienną  $\alpha$  nazywamy *współczynnikiem aproksymacji*. Analogicznie możemy zdefiniować  $\alpha$ -aproksymację dla instancji problemu maksymalizacyjnego, gdzie wartość  $A$  obliczonego rozwiązania spełnia:

$$A \geq \frac{1}{\alpha} \cdot OPT$$

**Definicja 2.3.** Wielomianowym schematem aproksymacji dla problemu optymalizującego nazywamy zbiór algorytmów, który dla danego  $\epsilon > 0$  zawiera algorytm aproksymacyjny  $A$  o współczynniku aproksymacji równym  $(1 + \epsilon)$ . Złożoność algorytmu  $A$  jest wielomianowa ze względu na rozmiar instancji problemu podanej na wejściu  $A$  dla danego  $\epsilon$ .

**Definicja 2.4.** Centroidem dla skończonego zbioru punktów  $x_1, \dots, x_k \in \mathbb{R}^d$  nazywamy:

$$C = \frac{x_1 + \dots + x_k}{k}$$

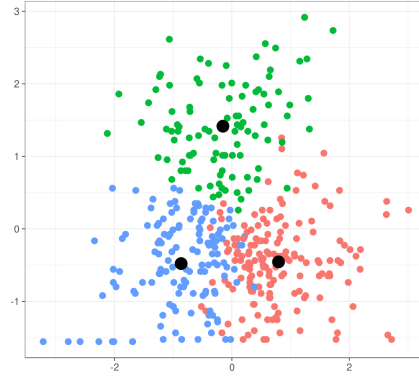
### 2.1. K-means

Zacznijmy od zdefiniowania problemu, dla którego będziemy analizować konstrukcje coresetów.

**Definicja 2.5.** *Problem k-means.* Niech  $X$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$ . Dla danego  $X$  chcemy znaleźć zbiór  $k \in \mathbb{N}$  punktów  $Q \subset \mathbb{R}^d$ , który minimalizuje funkcję  $\phi_X(Q)$  zdefiniowaną następująco:

$$\phi_X(Q) = \sum_{x \in X} d(x, Q)^2 = \sum_{x \in X} \min_{q \in Q} \|x - q\|^2$$

Definicja 2.4 zakłada, że działamy w przestrzeni euklidesowej. Uogólnioną wersję można zdefiniować analogicznie, zamieniając  $d$  na odpowiednią funkcję miary w danej przestrzeni.



**Rysunek 2.1:** Przykład interpretacji problemu  $k$ -means. Czarne punkty to centroidy, które stanowią zbiór  $Q$  optymalizujący funkcję  $\phi$ .

Problem  $k$ -means jest jednym z elementarnych problemów machine learningu. Używamy go m.in w data miningu, segmentacji obrazu czy klasyfikacji danych.

**Definicja 2.6.** *Problem k-means - wersja ważona.* Niech  $X_w$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$  oraz niech  $w$  będzie funkcją  $w : X_w \rightarrow \mathbb{R}_{\geq 0}$ . Dla danego  $X_w$  oraz funkcji  $w$  chcemy znaleźć zbiór  $k \in \mathbb{N}$  punktów  $Q \subset \mathbb{R}^d$ , który minimalizuje funkcję  $\phi_{X_w}(Q)$  zdefiniowaną następująco:

$$\phi_{X_w}(Q) = \sum_{x \in X_w} w(x) d(x, Q)^2$$

Wartość funkcji  $\phi$  dla optymalnego rozwiązania oznaczamy  $\phi_{opt}^k(X)$  lub  $OPT$ . W pracy często będziemy korzystać ze stwierdzenia *optymalne rozwiązanie*, które oznacza  $k$  elementowy zbiór  $C_{opt}$ , dla którego wartość funkcji  $\phi$  jest zminimalizowana. Funkcję  $\phi$  w literaturze nazywamy błędem kwantyzacji.

**Definicja 2.7.** *Klasterem* nazywamy skończony zbiór punktów  $x_1, \dots, x_k \in \mathbb{R}^d$ , które łączy pewna wspólna cecha. W naszym kontekście wspólną cechą będzie przyporządkowanie do tego samego centroidu.

## 2.2. Coreset

To jak definiujemy coreset ściśle zależy od problemu, który optymalizujemy. Zaczniemy od podstawowej definicji coresetu dla problemu  $k$ -means.

**Definicja 2.8.**  $(\epsilon, k)$ -aproxymacja zbioru centroidów. Niech  $X$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$ . Skończony zbiór  $A \subset \mathbb{R}^d$  nazywamy  $(\epsilon, k)$ -aproxymacją zbioru centroidów, gdzie  $\epsilon \in (0, 1)$  oraz  $k \in \mathbb{N}$ , jeżeli istnieje  $k$  elementowy podzbiór  $C \subseteq A$ , dla którego zachodzi:

$$|\phi_X(C_{opt}) - \phi_X(C)| \leq \epsilon \phi_X(C_{opt})$$

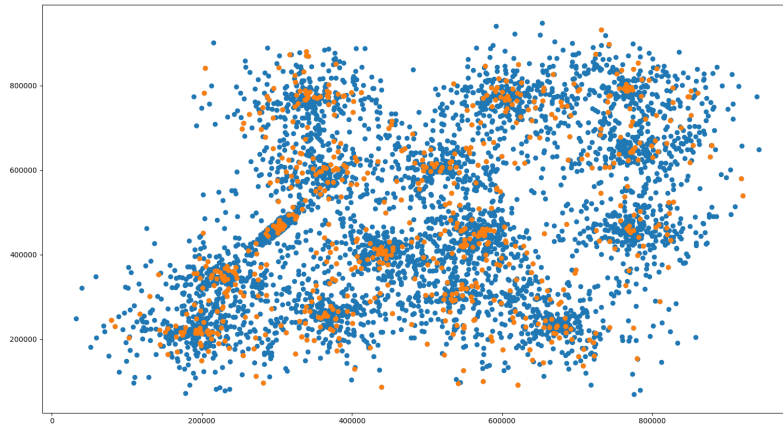
gdzie  $C_{opt}$  optymalizuje wartość funkcji  $\phi$ .

Intuicyjnie  $(\epsilon, k)$ -aproxymacje zbioru centroidów możemy rozumieć jako zbiór  $A \subset \mathbb{R}^d$ , który zawiera  $k$  elementowy podzbiór  $C \subseteq A$  *dobrze* aproxymujący  $C_{opt}$ . Definicja 2.8 pojawia się tutaj z dwóch powodów. Po pierwsze jest ona pewnego rodzaju prekursorem coresetów, a po drugie jest ona konieczna w definicji 2.10.

**Definicja 2.9.** *Coreset*. Niech  $X$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$ . Skończony zbiór  $C \subset \mathbb{R}^d$  nazywamy  $(\epsilon, k)$  coresetem, gdzie  $\epsilon \in (0, 1)$ , jeżeli dla dowolnego  $k \in \mathbb{N}$  elementowego zbioru  $Q \subset \mathbb{R}^d$  zachodzi:

$$|\phi_X(Q) - \phi_C(Q)| \leq \epsilon \phi_X(Q)$$

Zauważmy, że taka definicja daje nam bardzo mocne gwarancje teoretyczne. Wartość funkcji  $\phi_C(Q)$  aproxymuje  $\phi_X(Q)$  ze współczynnikiem aproxymacji równym  $(1 + \epsilon)$  dla dowolnego zbioru  $k$  kandydatów  $Q$  na rozwiązanie problemu  $k$ -means. Jest to na tyle istotne, że w literaturze odróżnimy taką wersję nazywając ją *strong coresetem*. Definicja 2.9 jest wariantem ciągłym, czyli punkty należące do coresetu są dowolnymi punktami przestrzeni, w szczególności nie muszą należeć do zbioru  $X$ . W praktyce stosuje się wariant dyskretny, który zakłada, że  $C \subseteq X$ . Tak samo jak w problemie  $k$ -means, coreset może być zdefiniowany w wariacie ważonym. Oznacza to, że dla zbioru  $C$  z definicji 2.9 definiujemy funkcję wag  $w : C \rightarrow \mathbb{R}_{\geq 0}$ . Reszta pozostaje bez zmian, ponieważ funkcja  $\phi$  uwzględnia  $C$  będące zbiorem ważonym.



**Rysunek 2.2:** Pomarańczowe punkty tworzą coreset dla zbioru niebieskich punktów. Parametry:  $k = 15$ ,  $\epsilon = 0.1$ ,  $n = 15000$ .



**Definicja 2.10.** *Coreset - weak (w wariancie dyskretnym).* Niech  $X$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$ . Niech  $C \subseteq X$  oraz  $A$  jest  $(\epsilon, k)$ -aproxymacją zbioru centroidów dla zbioru  $X$ , gdzie  $\epsilon \in (0, 1)$ . Jeżeli, dla dowolnego  $k \in \mathbb{N}$  elementowego podzbioru  $Q \subseteq A$  zachodzi:

$$|\phi_X(Q) - \phi_C(Q)| \leq \epsilon \phi_X(Q)$$

to parę  $(C, A)$  nazwiemy słabym coresetem.

*Słabość* definicji 2.10 wynika z konieczności budowy zbioru  $A$ . Intuicyjnie zbiór  $A$  pomaga coresetowi w zachowaniu odpowiedniego współczynnika aproksymacji, ponieważ w zbiorze  $A$  istnieje *dobry* kandydat na rozwiązanie. W pracy korzystamy wyłącznie z definicji 2.9. Z uwagi na to, że nasza praca ma charakter przeglądowy zdefiniowaliśmy wszystkie możliwe warianty coresetów.

### 3. Lightweight Coreset

Budowa coresetów w kontekście problemu  $k$ -means ma bardzo długo historię. Za przełomową pracę uznaje się [12], która jako pierwsza przedstawiła wielomianowy schemat aproksymacji o złożoności  $O(n\epsilon^{-2k^2d} \log^k n)$  budujący coreset dla problemu  $k$ -means.

Wyróżnia się trzy techniki budowania coresetów:

- Geometryczna dekompozycja problemu.
- Losowe próbkowanie zbioru.
- Zawansowane metody algebraiczne.

Pierwsza i trzecia technika cechuje się mocnymi gwarancjami teoretycznymi. Niestety większość rozwiązań jest mało praktyczna i kosztowna czasowo. Losowe próbkowanie w praktyce daje bardzo przyzwoite wyniki jednak bardzo często nie daje nam żadnej gwarancji odnośnie optymalności rozwiązania. Autorzy pracy [5] zaproponowali rozwiązanie o nazwie *lightweight coreset*, które w swoich założeniach ma łączyć:

- Prostą implementację.
- Gwarancje teoretyczne.
- Szybkie działanie oparte na próbkowaniu zbioru danych.

#### 3.1. Lightweight coreset

Zacznijmy od wprowadzenia definicji *lightweight coresetu*.

**Definicja 3.1.** *Lightweight coreset dla problemu  $K$ -means.* Niech  $\epsilon > 0$  oraz  $k \in \mathbb{N}$ . Niech  $X$  będzie  $n$  elementowym zbiorem punktów z  $\mathbb{R}^d$  wraz ze średnią  $\mu(X) = \frac{1}{n} \sum_{i=1}^n x_i$ . Zbiór  $C \subset \mathbb{R}^d$  jest  $(\epsilon, k)$  lightweight coresetem jeżeli dla dowolnego  $k \in \mathbb{N}$  elementowego zbioru  $Q \subset \mathbb{R}^d$  zachodzi:

$$|\phi_X(Q) - \phi_C(Q)| \leq \frac{\epsilon}{2} \phi_X(Q) + \frac{\epsilon}{2} \phi_X(\mu(X))$$

Jak możemy zauważyć definicja (3.1) trochę się różni od (2.9). Notacje *lightweight* coresetu możemy interpretować jako relaksację gwarancji teoretycznych zdefiniowanych w (2.9). Wprowadza ona oprócz błędu multiplikatywnego, błąd addytywny.

Główną motywacją stojącą za konstrukcjami coresetów jest to, żeby rozwiązanie obliczone na tym zbiorze było konkurencyjne z rozwiązaniem optymalnym dla całego zbioru danych. Dlatego w konsekwencji *lightweight* udowodnimy następujące twierdzenie.

**Twierdzenie 3.1.** [5] Niech  $\epsilon \in (0, 1]$ . Niech  $X$  będzie skończonym zbiorem danych z  $\mathbb{R}^d$  oraz niech  $C$  będzie  $(\epsilon, k)$  *lightweight coresetem* dla  $X$ . Optymalne rozwiązanie problemu  $K$ -means dla  $X$  oznaczamy  $Q_X^*$ . Optymalne rozwiązanie problemu  $K$ -means dla  $C$  oznaczamy  $Q_C^*$ . Dla takich założeń zachodzi:

$$\phi_X(Q_C^*) \leq \phi_X(Q_X^*) + 4\epsilon\phi_X(\mu(X))$$

*Dowód.* Zgodnie z własnością *lightweight coresetu* otrzymujemy:

$$\phi_C(Q_X^*) \leq (1 + \frac{\epsilon}{2})\phi_X(Q_X^*) + \frac{\epsilon}{2}\phi_X(\mu(X))$$

oraz

$$\phi_C(Q_C^*) \geq (1 - \frac{\epsilon}{2})\phi_X(Q_C^*) - \frac{\epsilon}{2}\phi_X(\mu(X))$$

Wiemy z definicji, że  $\phi_C(Q_C^*) \leq \phi_C(Q_X^*)$  oraz  $1 - \frac{\epsilon}{2} \geq \frac{1}{2}$ . A więc:

$$\begin{aligned} \phi_X(Q_C^*) &\leq \frac{1 + \frac{\epsilon}{2}}{1 - \frac{\epsilon}{2}}\phi_X(Q_X^*) + \frac{\epsilon}{1 - \frac{\epsilon}{2}}\phi_X(\mu(X)) \\ &\leq (1 + 2\epsilon)\phi_X(Q_X^*) + 2\epsilon\phi_X(\mu(X)) \end{aligned}$$

Zauważając, że:

$$\phi_X(Q_X^*) \leq \phi_X(\mu(X))$$

dowodzimy tezę twierdzenia.  $\square$

Twierdzenie 1 dowodzi, że kiedy wartość  $\epsilon$  maleje koszt optymalnego rozwiązania otrzymanego na zbiorze  $C$  zbiega do kosztu rozwiązania otrzymanego na całym zbiorze danych.

### 3.2. Konstrukcja

Opisana w tym podrozdziale konstrukcja *lightweight coresetu* oparta jest na próbkowaniu z uwzględnieniem ważności danego punktu. Niech  $q$  będzie dowolnym rozkładem prawdopodobieństwa na zbiorze  $X$  oraz niech  $Q \subset R^d$  będzie dowolnym potencjalnym zbiorem rozwiązań mocy  $k$ . Wtedy funkcję  $\phi$  możemy zapisać jako:

$$\phi_X(Q) = \sum_{x \in X} q(x) \frac{d(x, Q)^2}{q(x)}$$

Wynika z tego, że funkcja  $\phi$  może być aproksymowana poprzez wylosowanie  $m$  punktów z  $X$  korzystając z  $q$  i przypisując im wagi odwrotnie proporcjonalne do

$q$ . W szczególności musimy zagwarantować, jednostajność wyboru dowolnego zbioru  $k$  punktów  $Q$  z odpowiednim prawdopodobieństwem  $1 - \delta$ , gdzie  $\delta \in (0, \frac{1}{2}]$ . Funkcja  $q$  może mieć wiele form, autorzy [5] rekomendują postać:

$$q(x) = \frac{1}{2} \frac{1}{|X|} + \frac{1}{2} \frac{d(x, \mu(X))^2}{\sum_{x' \in X} d(x', \mu(X))^2}$$

---

#### Algorytm 1

---

Algorytm na wejściu otrzymuje zbiór  $X \subset \mathbb{R}^d$ .

**procedure** LIGHTWEIGHT

$\mu \leftarrow$  średnia dla  $X$

**for**  $x \in X$  **do**

$$q(x) = \frac{1}{2} \frac{1}{|X|} + \frac{1}{2} \frac{d(x, \mu(X))^2}{\sum_{x' \in X} d(x', \mu(X))^2}$$

$C \leftarrow$  próbka  $m$  ważonych punktów z  $X$ , gdzie każdy punkt  $x$  ma wagę  $\frac{1}{mq(x)}$  oraz jest wylosowany z prawdopodobieństwem  $q(x)$

**return** lightweight coreset  $C$

---

Pierwszy składnik rozkładu  $q$  to rozkład jednostajny, który zapewnia, że każdy punkt jest wylosowany z niezerowym prawdopodobieństwem. Drugi składnik uwzględnia kwadrat odległości punktu od średniej  $\mu(X)$  dla całego zbioru. Intuicyjnie, punkty, które są daleko od średniej  $\mu(X)$  mogą mieć istotny wpływ na wartość funkcji  $\phi$ . Musimy więc zapewnić, odpowiednią częstotliwość wyboru takich punktów. Jak pokazuje pseudokod, implementacja takiej konstrukcji jest całkiem prosta. Algorytm przechodzi przez zbiór danych jedynie dwukrotnie, a jego złożoność to  $O(nd)$ , gdzie  $n$  to rozmiar wejściowych danych,  $d$  to wymiar przestrzeni. Warto zwrócić uwagę na to, że nie mamy zależności od  $k$  co jest kluczowe w konsekwencji praktyczności takiego rozwiązania.

### 3.3. Analiza

W tym podrozdziale pokażemy, że zaproponowany w poprzedniej części algorytm oblicza lightweight coreset dla odpowiedniego  $m$ .

**Twierdzenie 3.2.** [5] *Niech  $\epsilon > 0$ ,  $\delta > 0$  oraz  $k \in \mathbb{N}$ . Niech  $X$  będzie skończonym zbiór punktów z  $\mathbb{R}^d$  oraz  $C \subseteq X$  to zbiór zwracany przez algorytm dla:*

$$m \geq c \frac{dk \log k + \log \frac{1}{\delta}}{\epsilon^2}$$

*gdzie  $c$  jest stałą. Wtedy z prawdopodobieństwem co najmniej  $1 - \delta$  zbiór  $C$  jest  $(\epsilon, k)$  lightweight coresetem dla  $X$ .*

*Dowód.* Zaczniemy od ograniczenia  $q(x)$  dla każdego punktu  $x \in X$ . W tym celu definiuje funkcję:

$$f(Q) = \frac{1}{2|X|} \phi_X(Q) + \frac{1}{2|X|} \phi_X(\mu(X))$$

gdzie  $\mu(X)$  to średnia zbioru  $X$  oraz dowodnimy następujący lemat.

**Lemat 1.** [5] Niech  $X$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$  wraz ze średnią  $\mu(X)$ . Dla każdego  $x \in X$  oraz  $Q \subset \mathbb{R}^d$  zachodzi:

$$\frac{d(x, Q)^2}{f(Q)} \leq \frac{16d(x, \mu(X))^2}{\frac{1}{|X|} \sum_{x' \in X} d(x', \mu(X))^2} + 16$$

*Dowód.* Z nierówności trójkąta oraz z faktu, że  $(|a| + |b|)^2 = 2a^2 + 2b^2$ , otrzymujemy

$$d(\mu(X), Q)^2 \leq 2d(x, \mu(X))^2 + 2d(x, Q)^2$$

Uśrednienie dla wszystkich  $x \in X$ , implikuje:

$$\begin{aligned} d(\mu(X), Q)^2 &\leq \frac{2}{|X|} \sum_{x \in X} d(x, \mu(X))^2 + \frac{2}{|X|} \sum_{x \in X} d(x, Q)^2 \\ &= \frac{2}{|X|} \phi_X(\mu(X)) + \frac{2}{|X|} \phi_X(Q) \end{aligned}$$

To implikuje, że dla każdego  $x \in X$  oraz  $Q \subset \mathbb{R}^d$  zachodzi:

$$\begin{aligned} d(x, Q)^2 &\leq 2d(x, \mu(X))^2 + 2d(\mu(X), Q)^2 \\ &\leq 2d(x, \mu(X))^2 + \frac{4}{|X|} \phi_X(\mu(X)) + \frac{4}{|X|} \phi_X(Q) \end{aligned}$$

Dzieląc powyższą nierówność przez wyżej zdefiniowaną funkcję  $f(Q)$  dostajemy:

$$\begin{aligned} \frac{d(x, Q)^2}{f(Q)} &\leq \frac{2d(x, \mu(X))^2 + \frac{4}{|X|} \phi_X(\mu(X)) + \frac{4}{|X|} \phi_X(Q)}{\frac{1}{2|X|} \phi_X(Q) + \frac{1}{2|X|} \phi_X(\mu(X))} \\ &\leq \frac{2d(x, \mu(X))^2 + \frac{4}{|X|} \phi_X(\mu(X))}{\frac{1}{2|X|} \phi_X(\mu(X))} + \frac{\frac{4}{|X|} \phi_X(Q)}{\frac{1}{2|X|} \phi_X(Q)} \\ &\leq \frac{16d(x, \mu(X))^2}{\frac{1}{|X|} \sum_{x' \in X} d(x', \mu(X))^2} + 16 \end{aligned}$$

co kończy dowód lematu.  $\square$

Powyższy lemat implikuje, że stosunek pomiędzy kosztem kontrybucji  $d(x, Q)^2$  jednego punktu  $x \in X$  a  $f(Q)$  jest ograniczony dla każdego  $Q \subseteq X$  przez:

$$s(x) = \frac{16d(x, \mu(X))^2}{\frac{1}{|X|} \sum_{x' \in X} d(x', \mu(X))^2} + 16$$

Niech  $S = \frac{1}{|X|} \sum_{x \in X} s(x)$ . Zauważmy, że:

$$\begin{aligned} S &= \frac{1}{|X|} \sum_{x \in X} s(x) = \frac{1}{|X|} \sum_{x \in X} \left( \frac{16d(x, \mu(X))^2}{\frac{1}{|X|} \sum_{x' \in X} d(x', \mu(X))^2} + 16 \right) \\ &= \frac{1}{|X|} \sum_{x \in X} \left( \frac{16d(x, \mu(X))^2}{\frac{1}{|X|} \sum_{x' \in X} d(x', \mu(X))^2} \right) + \frac{1}{|X|} \sum_{x \in X} 16 \end{aligned}$$

$$= \frac{16 \sum_{x \in X} d(x, \mu(X))^2}{\sum_{x' \in X} d(x', \mu(X))^2} + 16 = 32$$

dla każdego zbioru  $X$ . Dzięki temu możemy zapisać rozkład  $q$  jako:

$$q(x) = \frac{1}{2} \frac{1}{|X|} + \frac{1}{2} \frac{d(x, \mu(X))^2}{\sum_{x' \in X} d(x', \mu(X))^2} = \frac{s(x)}{S|X|}$$

dla każdego  $x \in X$ . Teraz zdefiniujmy funkcję:

$$g_Q(x) = \frac{d(x, Q)^2}{f(Q)s(x)}$$

dla każdego  $x \in X$  oraz  $Q \subset \mathbb{R}^d$ . Zauważmy, że dla dowolnego zbioru  $Q \subset \mathbb{R}^d$  zachodzi:

$$\begin{aligned} \phi_X(Q) &= \sum_{x \in X} d(x, Q)^2 = S|X|f(Q) \sum_{x \in X} \frac{s(x)}{S|X|} \frac{d(x, Q)^2}{f(Q)s(x)} \\ &= S|X|f(Q) \sum_{x \in X} q(x)g_Q(x) \end{aligned}$$

Następnie podstawiamy z definicji wartości oczekiwanej:

$$\mathbb{E}_q[g_Q(x)] = \sum_{x \in X} q(x)g_Q(x)$$

dzięki temu przekształcamy ostatnie równanie:

$$\phi_X(Q) = S|X|f(Q)\mathbb{E}_q[g_Q(x)]$$

Następnym krokiem jest ograniczenie wartości  $\mathbb{E}_q[g_Q(x)]$ . Autorzy [5] nie dowodzą wprost tego ograniczenia, powołując się na inne prace [10]. Dowód jest bardzo skomplikowany i wykracza tematyką istotnie poza ramy tej pracy, więc go pomijamy. Korzystamy z finalnego ograniczenia:

$$|\mathbb{E}_q[g_Q(x)] - \frac{1}{|C|} \sum_{x \in X} g_X(x)| \leq \frac{\epsilon}{32}$$

Powyższe ograniczenie jest prawdziwe z prawdopodobieństwem  $1 - \delta$  dla dowolnego  $Q \subset \mathbb{R}^d$  o rozmiarze nie większym niż  $k$ . Mnożąc obie strony nierówności przez  $32|X|f(Q)$  otrzymujemy:

$$|32|X|f(Q)\mathbb{E}_q[g_Q(x)] - \frac{32|X|f(Q)}{|C|} \sum_{x \in X} g_X(x)| \leq \epsilon|X|f(Q)$$

Niech  $(C, u)$  będzie ważonym zbiorem, gdzie dla każdego  $x \in C$  definiujemy funkcję  $u(x) = \frac{1}{|C|q(x)}$ . Wynika z tego, że:

$$\begin{aligned} \frac{32|X|f(Q)}{|C|} \sum_{x \in X} g_X(x) &= \sum \frac{1}{|C|q(x)} d(x, Q)^2 \\ &= \sum u(x) d(x, Q)^2 = \phi_C(Q) \end{aligned}$$

A więc otrzymujemy:

$$|32|X|f(Q)\mathbb{E}_q[g_Q(x)] - \phi_C(Q)| \leq \epsilon|X|f(Q)$$

$$|\phi_Q(Q) - \phi_C(Q)| \leq \frac{\epsilon}{2}\phi_X(Q) + \frac{\epsilon}{2}\phi_X(\mu(X))$$

co kończy dowód twierdzenia 3.2.

□

## 4. Geometryczna Dekompozycja

W tej części pracy przedstawimy konstrukcję budowy coresetu bazującą na geometrycznej dekompozycji problemu. Punktem wyjścia były badania [13], na których bazuje opisany w podrozdziale 4.4 algorytm. Zgodnie z nimi budowa coresetu w kontekście problemu K-means to wieloetapowy proces, który jest sekwencją algorytmów z prac: [8] [9] [4] [13]. W rozdziale przyjmujemy następujący porządek analizy konstrukcji:

- W sekcjach 4.1, 4.2, 4.3 opiszemy konstrukcje pomocnicze pochodzące z prac: [8], [9], [4].
- W sekcji 4.4 opiszemy właściwy algorytm z pracy [13].

### 4.1. Algorytm Gonzalez’a

Pierwszy algorytm, który opiszemy to *Farthest point algorithm* z pracy [8]. Jest to pierwszy algorytm aproksymacyjny rozwiązujący problem  $k$ -centrów ze współczynnikiem aproksymacji równym 2. Jego złożoność to  $O(nk)$ , gdzie  $n$  jest liczbą punktów danych na wejściu. Algorytm Gonzalez’a jest nam potrzebny, ponieważ w rozdziale 4.2, w którym budujemy kratę wykładniczą istotnie korzystamy z rozwiązania problemu  $k$ -centrów.

Na potrzeby tego rozdziału wprowadzimy kilka definicji.

**Definicja 4.1.** *Problem  $k$ -centrów.* Niech  $X$  będzie skończonym zbiorem punktów z  $\mathbb{R}^d$ . Dla danego  $X$  chcemy znaleźć zbiór  $k \in \mathbb{N}$  punktów  $Q \subset \mathbb{R}^d$ , który minimalizuje funkcję  $\rho_X(Q)$  zdefiniowaną następująco:

$$\rho_X(Q) = \max_{x \in X} \min_{q \in Q} |x - q|$$

**Definicja 4.2.** Niech  $G = (V, E, W)$  będzie ważonym nieskierowanym grafem ze zbiorem wierzchołków  $V$ , krawędzi  $E$  oraz funkcją  $W : E \rightarrow \mathbb{R}^+$  przyporządkowującą wagi krawędziom. W tym rozdziale utożsamiamy wagi z dystansem pomiędzy dwoma punktami.

**Definicja 4.3.** Podział zbioru wierzchołków na  $k \in \mathbb{N}$  zbiorów  $B_1, \dots, B_k$ , nazywamy  *$k$ -splitem*.

**Definicja 4.4.** Zbiory  $B_i$  podziału  $k$ -split nazywamy *klastrami*.



Dla każdego  $k$ -splitu definiujemy funkcję celu  $f : B_1, \dots, B_k \rightarrow \mathbb{R}_{\geq 0}$ . W tym rozdziale zakładamy, że funkcją celu jest  $\max(M_1, \dots, M_k)$ , gdzie  $M_i$  to największa waga krawędzi pomiędzy dowolnymi dwoma punktami z  $B_i$ . Tak zdefiniowana funkcja celu jest odpowiednikiem funkcji  $\rho$  zdefiniowanej dla problemu  $k$ -centerów.

**Definicja 4.5.** *Problem klastrowania.* Dla danego grafu  $G$ , funkcji celu  $f$  oraz  $k \in \mathbb{N}$  znaleźć  $k$ -split, dla którego funkcja  $f$  jest zminimalizowana. Dla przykładu: znaleźć  $k$ -split  $(B_1^*, \dots, B_k^*)$  taki, że

$$f(B_1^*, \dots, B_k^*) = \min\{f(B_1, \dots, B_k) \mid (B_1, \dots, B_k) \text{ to } k\text{-split dla } G\}$$

Niech  $S \subset \mathbb{R}^d$  będzie zbiorem, na którym chcemy rozwiązać problem  $k$ -centrów oraz niech  $T$  będzie podzbiorem  $S$ . Zakładamy, że  $|S| > k$  ponieważ w przeciwnym przypadku problem podziału jest trywialnie rozwiązywalny.

**Definicja 4.6.** Zbiór  $T$  nazywamy  $(k+1)$  kliką wysokości  $h$  jeżeli moc zbioru  $T$  jest równa  $k+1$  oraz odległość pomiędzy parą dwóch różnych punktów jest równa co najmniej  $h$ .

Niech  $OPT(S)$  oznacza optymalne rozwiązanie problemu  $k$ -centrów dla zbioru  $S$ . Udowodnimy teraz następujący lemat, który jest potrzebny w dowodzie ograniczającym współczynnik aproksymacji algorytmu.

**Lemat 2.** [8] Jeżeli w zbiorze  $S$  istnieje  $(k+1)$  klika wysokości  $h$ , to  $OPT(S) \geq h$ .

*Dowód.* Klika ma  $k+1$  elementów, zatem co najmniej 2 z nich znajdują się w jednym klastrze. Waga krawędzi pomiędzy tymi punktami to co najmniej  $h$  co implikuje, że  $OPT(S) \geq h$ .  $\square$

Algorytm Gonzaleza składa się z fazy inicjalizującej oraz  $k-1$  faz powiększających. W fazie inicjalizującej wszystkie elementy są przypisane do zbioru  $B_1$ , który jest pierwszym klastrem. Jeden z elementów tego zbioru oznaczamy jako  $(t_1)$  - środek klastra  $B_1$ . Wybór tego elementu jest losowy. Podczas  $j$  fazy powiększającej, niektóre elementy z istniejącego podziału na klastry  $B_1, \dots, B_j$  trafiają do nowego zbioru  $B_{j+1}$ . Dodatkowo jeden z elementów nowego zbioru będzie oznaczony jako  $(t_{j+1})$  - środek klastra  $B_{j+1}$ . Budowę zbioru  $B_{j+1}$  rozpoczynamy od wyboru punktu  $v$ , który należy do jednego ze zbiorów  $B_1, \dots, B_j$  oraz jego odległość do środka klastra, do którego należy jest największa spośród wszystkich punktów z  $B_1, \dots, B_j$ . Taki punkt będzie oznaczony jako  $(t_{j+1})$ , czyli jest środkiem klastra  $B_{j+1}$ . Każdy punkt, dla którego dystans do  $v$  jest nie większy niż dystans do środka klastra, w którym się znajduje zostaje przeniesiony do  $B_{j+1}$ . Algorytm 2 buduje jakiś  $k$ -split. Teraz pokażemy, że dla takiego  $k$ -splitu wartość funkcji celu  $f(S)$  jest ograniczona przez  $2 \cdot OPT(S)$ .

Niech  $v \in B_j$  oraz  $1 \leq j \leq k$  będzie wierzchołkiem, którego odległość do  $(t_j)$  jest maksymalna. Niech  $h$  będzie tą odległością. Ponieważ dla zbioru wag krawędzi zachodzi nierówność trójkąta to:

$$W(E(x, y)) \leq W(E(x, t_i)) + W(E(y, t_i)) \leq 2h$$

gdzie  $t_i$  to środek klastra dla punktów  $x, y \in B_i$ . A więc możemy ograniczyć wartość naszej funkcji celu przez  $f(S) \leq 2h$ . Ponieważ  $v$  nigdy nie zostało

---

**Algorytm 2**

---

Algorytm na wejściu otrzymuje zbiór  $S \in \mathbb{R}^d$ .

Niech  $T$  będzie szukanym zbiorem  $k$ -centrów.

Dla każdego punktu  $p \notin T$ , algorytm trzyma  $neighbor(p)$ , czyli najbliższy punkt w  $T$  dla  $p$  oraz  $dist(p)$ , czyli odległość od punktu  $p$  do  $neighbor(p)$ .

**procedure** FARTHEST POINT ALGORITHM

$T \leftarrow \emptyset$

$dist(p) \leftarrow \infty$  for all  $p \in S$

**while**  $|T| \leq k$  **do**

$D \leftarrow \max\{dist(p) | p \in S - T\}$

wybierz  $v$  z  $S - T$  tak, aby  $dist(v) = D$

add  $v$  to  $T$

zaktualizuj  $neighbor(p)$  oraz  $dist(p)$  dla każdego  $p \in S - T$

**return**  $T$

---

wybrane na środek klastra to wiemy, że  $W(t_i, t_j) \geq h$  dla  $i \neq j$ . Niech  $T = \{t_1, \dots, t_k, v\}$ . Z definicji wiemy, że  $T$  jest  $(k+1)$  kliką wagi  $h$ , więc z lematu 2,  $OPT(S) \geq h$ . A więc ograniczenie wartości funkcji celu to  $f(S) \leq 2h \leq 2 \cdot OPT(S)$ .

## 4.2. Konstrukcja kraty wykładniczej

W tym podrozdziale omówimy część pracy [9]. Tematem pracy są konstrukcje algorytmów dla problemów  $k$ -means oraz  $k$ -median. W kontekście problemu  $k$ -means autorzy zaproponowali algorytm rozwiązujący problem  $k$ -means bazujący na budowie coresetu, który uzyskuje lepszą złożoność od [12]. Matoušek w swojej pracy przedstawił wielomianowy schemat aproksymacji, którego złożoność to  $O(n\epsilon^{-2k^2d} \log^k n)$ . Autorzy [9] poprawili tę złożoność uzyskując  $O(n + k^{k+2}\epsilon^{-(2d+1)k} \log^{k+1} n \log^k \frac{1}{\epsilon})$ . Schemat konstrukcji jest następujący:

- Obliczmy szybką ale niedokładną aproksymację dla problemu  $k$ -means z pewną dużą wartością  $k$ .
- Obliczoną aproksymację przekształcamy w  $(\epsilon, k)$  coreset używając kraty wykładniczej.

### 4.2.1. Szybka aproksymacja dla problemu K-means.

Zacznijmy od pierwszej części. Dokładniej, udowodnimy następujące twierdzenie.

**Twierdzenie 4.1.** [9] Dla danego zbioru  $n$  punktów  $P \subset \mathbb{R}^d$  oraz parametru  $k \in \mathbb{N}$  możemy obliczyć zbiór  $X$  o mocy  $O(k \log^3 n)$ , dla którego

$$\phi_P(X) \leq 20\phi_{opt}^k(P)$$

Czas działania algorytmu to  $O(n)$  dla  $k = O(n^{\frac{1}{4}})$  oraz  $O(n \log(k \log n))$  w przeciwnym przypadku.

Niech  $P \subset \mathbb{R}^d$  będzie danym na wejściu zbiorem  $n$  punktów. Chcemy szybko obliczyć aproksymację dla problemu  $k$ -means na tym zbiorze, gdzie rozmiar wynikowego zbioru punktów będzie rzędu  $O(k \log^3 n)$ .

**Definicja 4.7.** *Złe/dobre punkty.* Dla zbioru punktów  $X$ , punkt  $p \in P$  nazywamy *złym* jeżeli

$$d(p, X) \geq 2d(p, C_{opt})$$

gdzie  $C_{opt}$  jest zbiorem punktów realizującym  $\phi_{opt}^k(P)$ . Punkt jest *dobry* jeżeli nie jest zły.

Na początku opiszemy procedurę, która dla danego  $P$ , wyznacza zbiór punktów  $X$  o mocy rzędu  $O(k \log^2 n)$  oraz zbiór  $P' \subset P$ . Zbiór  $P'$  będzie zawierać *dobre* punkty dla zbioru  $X$ .

Konstrukcję zbioru  $X$  zaczynamy od obliczenia 2-aproksymacji problemu k-centrów dla zbioru  $P$ . Niech obliczony zbiór centrów to  $V$ . Taką aproksymację dla  $k = O(n^{\frac{1}{4}})$  możemy obliczyć w czasie  $O(n)$  oraz dla  $k = \Omega(n^{\frac{1}{4}})$  w czasie  $O(n \log k)$  [7]. Są to wyniki teoretyczne, które zakładamy na potrzebę analizy. W podrozdziale 4.1 przedstawiliśmy algorytm aproksymacyjny rozwiązujący ten problem w czasie  $O(nk)$  dla dowolnego  $k$ . Algorytmy o lepszej złożoności są w istotny sposób bardziej skomplikowane, dlatego w naszej implementacji zastosowaliśmy algorytm Gonzaleza. Niech  $L$  będzie promieniem takiej aproksymacji, czyli największą odległością pomiędzy punktem  $v \in V$  a punktem  $p \in P - V$ . Ponieważ algorytm z pracy [7] bazuje na algorytmie przedstawionym w podrozdziale 4.1 z pracy [8] to dystans pomiędzy dowolną parą punktów z  $V$  wynosi co najmniej  $L$ . To implikuje następujące ograniczenia:

$$\left(\frac{L}{2}\right)^2 \leq \phi_{opt}^k(V) \leq \phi_{opt}^k(P) \leq nL^2$$

Dolne ograniczenie argumentujemy tym, że możemy ograniczyć  $\phi_{opt}^k(V)$  przez  $d(v_1, v_2)^2$ , gdzie  $v_1, v_2 \in V$ . Górne ograniczenie wynika z faktu, że promień aproksymacji jest równy  $L$ , więc dla  $n$  elementowego zbioru  $P$  wartość funkcji  $\phi_{opt}^k(P)$  nie może być większa niż  $nL^2$ .

Następnym krokiem konstrukcji jest wylosowanie  $\rho = \gamma k \log^2 n$  punktów ze zbioru  $P$ , gdzie  $\gamma$  jest stałą, która wynika z analizy, którą zaraz przeprowadzimy. Niech  $Y$  będzie zbiorem wylosowanych punktów z  $P$  oraz  $X = Y \cup V$  będzie zbiorem środków klastów. Dla  $\rho > n$  przyjmujemy  $X = P$ .

Konstrukcja zbioru  $X$  jest stosunkowo prosta. Dużo cięższym zadaniem jest zbudowanie zbioru  $P'$ , który jest zbiorem *dobrych* punktów dla  $X$ .

#### 4.2.2. Konstrukcja zbioru dobrych punktów dla $X$ .

Rozpatrzmy zbiór  $C_{opt}$ , który jest optymalnym zbiorem centrów problemu  $k$ -means dla zbioru  $P$  i liczby  $k$ . Dla każdego  $c_i \in C_{opt}$  tworzymy kulę  $B_i$  o środku w  $c_i$ . Każda taka kula będzie zawierać  $\eta = \frac{n}{20k \log n}$  punktów z  $P$ . Jeżeli zastosowana do wylosowania zbioru  $X$  stała  $\gamma$  jest odpowiednio duża to

z wysokim prawdopodobieństwem w każdym  $B_i$  jest przynajmniej jeden punkt z  $X$ . Dokładniej:

$$X \cap B_i \neq \emptyset \text{ dla } i = 1 \dots k$$

Niech  $P_{bad}$  będzie zbiorem złych punktów  $P$  w kontekście zbioru  $X$ . Załóżmy, że dla każdego  $B_i$  istnieje punkt  $x_i \in X$ , który  $x_i \in B_i$ . Zauważmy, że dla każdego punktu  $p \in P \setminus B_i$ , dla którego  $x_i$  jest najbliższym punktem z  $X$  mamy  $\|p - x_i\| \leq 2\|p - c_i\|$ . W szczególności takie punkty są *dobre*, jeżeli  $c_i$  jest najbliższym punktem z  $C_{opt}$ . Zatem z wysokim prawdopodobieństwem jedyne *złe* punkty będą w kulach  $B_i$  dla  $i = 1, \dots, k$ . To implikuje, że z wysokim prawdopodobieństwem liczba złych punktów w  $P$  dla zbioru  $X$  to co najwyżej  $\beta = k\eta = \frac{n}{20 \log n}$ .

W takim razie złych punktów nie jest dużo. Mimo tego bezpośrednie wyznaczenie tych punktów jest skomplikowane. Autorzy pracy [9] budują zbiór  $P'$  tak aby koszt złych punktów w  $P'$  był jak najmniejszy. Koszt punktu  $p$  w tym kontekście oznacza jaki wkład ma punkt  $p$  w wartość  $\phi_{P'}(X)$ . Dla każdego punktu w  $P$  obliczamy najbliższego sąsiada w  $X$ . Niech  $r(p) = d(p, X)$  dla każdego punktu  $p \in P$ . Teraz podzielimy  $P$  na zbiory według następującej formuły:

$$P[a, b] = \{p \in P \mid a \leq r(p) \leq b\}$$

A dokładniej:

$$\begin{aligned} P_0 &= P\left[0, \frac{L}{4n}\right] \\ P_\infty &= P\left[2Ln, \infty\right] \\ P_i &= P\left[\frac{2^{i-1}L}{n}, \frac{2^i L}{n}\right] \end{aligned}$$

dla  $i = 1, \dots, M$ , gdzie  $M = 2\lceil \lg n \rceil + 3$  oraz  $L$  to promień aproksymacji. Taki podział możemy wykonać w czasie linowym. Niech  $P_\alpha$  będzie ostatnim zbiorem, który zawiera więcej niż  $2\beta = \frac{n}{10 \log n}$  punktów. Szukany zbiór zdefiniujemy następująco:

$$P' = V \cup \bigcup_{i \leq \alpha} P_i$$

Chcielibyśmy aby  $|P'| \geq \frac{n}{2}$  oraz  $\phi_{P'}(X) = O(\phi_{P'}(C_{opt}))$ . Teraz udowodnimy, że faktycznie tak zdefiniowane  $P'$  spełnia powyższe założenia.

*Dowód.* Moc zbioru  $P'$  jest na pewno równa conajmniej  $\left(n - |P_\infty| - M \frac{n}{10 \log n}\right)$ , gdzie ostatni składnik jest mocą wszystkich  $M$  zbiorów  $P_i$ , które zawierają nie więcej niż  $2\beta$  punktów. Zauważmy, że  $P_\infty \subseteq P_{bad}$  oraz  $|P_{bad}| \leq \beta$ . Zatem:

$$\begin{aligned} |P'| &\geq n - \frac{n}{20 \log n} - M \frac{n}{10 \log n} \\ &= n - \left(\frac{n}{10 \log n}\right) \left(M + \frac{1}{2}\right) \\ &= n - \left(\frac{n}{10 \log n}\right) \left(2\lceil \lg n \rceil + 3 + \frac{1}{2}\right) \end{aligned}$$

$$\geq \frac{n}{2}$$

Jeżeli  $\alpha > 0$ , to  $|P_\alpha| \geq 2\beta = \frac{n}{10 \log n}$ . Z uwagi na to jak budujemy  $P'$  w najgorszym przypadku wszystkie złe punkty będą w  $P_\alpha$ . Wtedy takie punkty będą miały największy wpływ na funkcję  $\phi$ . Niech  $Q' = P_\alpha \setminus P_{bad}$ . Dla dowolnych punktów  $p \in P' \cap P_{bad}$  oraz  $q \in Q'$ , mamy  $d(p, X) \leq 2d(q, X)$ , ponieważ  $d(p, X) \leq \frac{2^\alpha L}{n}$  oraz  $d(q, X) \geq \frac{2^{\alpha-1} L}{n}$ . Dodatkowo  $|Q'| > |P_{bad}|$ , a więc:

$$\phi_{P' \cap P_{bad}}(X) \leq 4\phi_{Q'}(X) \leq 16\phi_{Q'}(C_{opt}) \leq 16\phi_{P'}(C_{opt})$$

Teraz możemy wyprowadzić następujące ograniczenie:

$$\begin{aligned} \phi_{P'}(X) &= \phi_{P' \cap P_{bad}}(X) + \phi_{P' \setminus P_{bad}}(X) \\ &\leq 16\phi_{P'}(C_{opt}) + 4\phi_{P'}(C_{opt}) = 20\phi_{P'}(C_{opt}) \end{aligned}$$

Jeżeli  $\alpha = 0$ , to dla dowolnego punktu  $p \in P'$  mamy:

$$(d(p, X))^2 \leq n \left( \frac{L}{4n} \right)^2 \leq \frac{L^2}{16n} \leq \frac{L^2}{4n}$$

Zatem:

$$\phi_{P'}(X) \leq \frac{L^2}{4} \leq \phi_V(C_{opt}) \leq \phi_{P'}(C_{opt})$$

bo  $V \subseteq P'$ . □

Powyższa analiza dowodzi poprawności konstrukcji dla zbiorów  $X$  i  $P'$ . Podsumowując otrzymujemy zbiór  $X$  o mocy  $O(k \log^2 n)$  oraz zbiór  $P'$ , dla którego mamy  $\phi_{P'}(X) \leq 20\phi_{P'}(C_{opt})$ . Czas działania tego algorytmu to  $O(n)$  dla  $k = O(n^{\frac{1}{4}})$  oraz  $O(n \log(k \log n))$  w przeciwnym przypadku [9]. Aby otrzymać taką złożoność kluczowe jest szybkie obliczenie najbliższych sąsiadów punktów. Autorzy [9] proponują zastosowanie algorytmu przedstawionego w pracy [2].

Wróćmy teraz do twierdzenia 4.1, które zostało zdefiniowane na początku podrozdziału 4.2. Chcemy znaleźć zbiór  $X'$  o mocy  $O(k \log^3 n)$ , dla którego jak najwięcej punktów ze zbioru  $P$  jest dobrych. Powyżej zdefiniowaną procedurę powtarzamy dla zbioru  $P_1 = P \setminus P'$ . Analogicznie otrzymamy zbiór  $P'_1$  oraz  $X_1$ . Kolejny raz aplikujemy procedurę na zbiorze  $P_2 = P \setminus (P' \cup P'_1)$ . Ponieważ za każdym razem zbiór  $P_i$  zmniejsza się co najmniej o połowę to taki proces zakończy się po  $O(\log n)$  powtórzeniach. Finalnie otrzymamy zbiór  $X' = X \cup X_1 \dots X_i$  o mocy  $O(k \log^3 n)$ , dla którego  $\phi_P(X') \leq 20\phi_P(C_{opt})$ . Złożoność pozostanie taka sama, czyli  $O(n)$  dla  $k = O(n^{\frac{1}{4}})$  oraz  $O(n \log(k \log n))$  w przeciwnym przypadku.

#### 4.2.3. Krata wykładnicza.

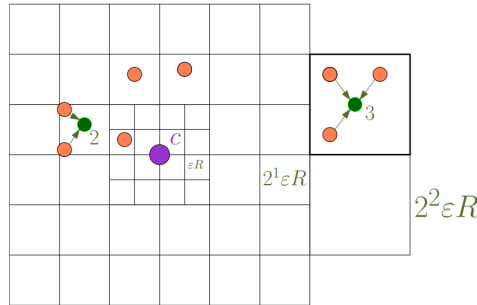
Przejdźmy teraz do kluczowej części tego podrozdziału, czyli budowy kraty wykładniczej, która jest autorską konstrukcją [9]. Krata wykładnicza jest obiektem geometrycznym zbudowanym z komórek, które są hipersześcianami. Każda krata posiada punkt początkowy, od którego zaczynamy konstrukcję przylegających do siebie komórek o zwiększających się długościach boków. Komórki

zawierają punkty z danego na wejściu zbioru danych oraz punkt początkowy jest ustalonym punktem z tego zbioru. Konstrukcję dzielimy na  $\log n$  poziomów, gdzie  $n$  to moc wejściowego zbioru. Komórki o tych samych długościach boków należą do tego samego poziomu. Na rysunku 4.1 widzimy 3 poziomową kartę wykładniczą w  $\mathbb{R}^2$ . Komórki są kwadratami a punktem początkowym jest  $c$ .

Niech  $P \subset \mathbb{R}^d$ ,  $|P| = n$  oraz niech  $A = \{x_1, \dots, x_m\}$  będzie zbiorem punktów, dla którego zachodzi  $\phi_P(A) \leq c\phi_{opt}^k(P)$ , gdzie  $c$  jest stałą. Nasze  $A$  otrzymamy z konstrukcji opisanej w 4.2.1 dla parametru  $k = O(k \text{poly} \log n)$  oraz zbioru  $P$ , gdzie  $\text{poly} \log n = \bigcup_{c \geq 1} O(\log^c n)$ . Przedstawiony w tym podrozdziale algorytm oblicza  $(k, \epsilon)$ -coreset, gdzie  $\epsilon \in (0, 1)$  oraz  $c = 20$ .

Niech  $R = \sqrt{\frac{\phi_P(A)}{cn}}$  będzie dolnym ograniczeniem dla  $R_{opt}^\phi(P, k) = \sqrt{\frac{\phi_{opt}(P, k)}{n}}$ . Niech  $P_i$  będzie podzbiorem punktów z  $P$ , dla których punkt  $x_i \in A$  jest najbliższym sąsiadem. Dla dowolnego  $p \in P_i$ , mamy  $\|px_i\| \leq \sqrt{xn}R$ , ponieważ  $\|px_i\|^2 \leq \phi_P(A)$  dla  $i = 1, \dots, m$ .

Kolejnym krokiem będzie budowa kraty wykładniczej wokół każdego punktu  $x_i$ . Niech  $Q_{i,j}$  będzie kwadratem o boku długości  $R2^j$  o środku w punkcie  $x_i$  dla  $j = 0, \dots, M$ , gdzie  $M = \lceil 2 \lg(cn) \rceil$ , który jest równoległy do osi układu współrzędnych dla danej przestrzeni. Następnie, niech  $V_{i,0} = Q_{i,0}$  oraz niech  $V_{i,j} = Q_{i,j} \setminus Q_{i,j-1}$  dla  $j = 0, \dots, M$ . Kolejnym krokiem będzie przekształcenie  $V_{i,j}$  w kratę, której komórki będą długości  $r_j = \frac{\epsilon R 2^j}{10cd}$  oraz niech  $G_i$  oznacza wynikową kratę wykładniczą dla  $V_{i,0}, \dots, V_{i,M}$ . Mając  $G_i$  obliczamy dla każdego punktu z  $P_i$  komórkę, do której należy. Dla każdej niepustej komórki z kraty wybieramy losowy punkt do niej należący, który będzie jej reprezentantem. Do takiego punktu przypisujemy wagę, która będzie równa liczbie punktów z komórki, dla której jest reprezentantem. Po przejściu całej kraty otrzymamy zbiór  $S_i$  takich punktów. Definiujemy  $S = \bigcup_i S_i$ , który jest  $(\epsilon, k)$  coresetem. Dowód tego, że zbiór  $S$  jest  $(\epsilon, k)$  coresetem pomijamy z uwagi na jego mechaniczność i żmudność. Zauważmy, że  $|S| = O\left(\frac{|A| \log n}{(c\epsilon)^d}\right)$ , ponieważ każda krata ma  $\log n$  poziomów, a każdy poziom stałą liczbę komórek.



**Rysunek 4.1:** Krata wykładnicza, gdzie  $c = x_i$ . W najbliższym otoczeniu punktu  $c$  widzimy kratę  $V_{i,0}$  z długością komórki równej  $r_0 = \epsilon R$ . Dla uproszczenia pomijam zmienne w mianowniku wcześniej podanej definicji  $r_j$ . Kolejny poziom to krata  $V_{i,1}$  o długości komórek równej  $r_1 = 2\epsilon R$ . Zielone punkty to reprezentanci danej komórki, którym przyjmujemy odpowienie wagi równe ilości punktów w komórce.

### 4.3. Heurystyka single swap

W tym podrozdziale opiszemy heurystykę single swap [4], która jest przykładem techniki *local search*. Algorytm jest  $(25 + \epsilon)$  aproksymacją problemu  $k$ -means, która zakłada, że na wejściu dostajemy zbiór kandydatów na centra  $C$  oraz zbiór  $n$  punktów  $P \subset \mathbb{R}^d$ . Autorzy [4] w celu wyznaczenia  $C$  wykorzystują algorytm z pracy [12], którą zastąpimy pracą [9] z uwagi na lepsze gwarancje teoretyczne. Korzystając z algorytmu opisanego w podrozdziale 4.2.3 wyznaczamy zbiór  $C$ , który będzie obliczonym  $(\epsilon, k)$ -coresetem.

Heurystyka *single swap* działa poprzez wybranie początkowego zestawu  $k$  centrów  $S$  ze zbioru kandydatów na centra  $C$ , a następnie wielokrotnej próbie ulepszenia rozwiązania poprzez usunięcie jednego centrum  $s \in S$  i zastąpienie go innym centrum  $s' \in C - S$ . Początkowy stan  $S$  może zostać zainicjalizowany losowo albo może być obliczony za pomocą omówionego w podrozdziale 4.1 algorytmu Gonzaleza. Niech  $S' = S - \{s\} \cup \{s'\}$  będzie nowym zbiorem centrów, gdzie taką operację na zbiorze  $S$  nazywamy *wymianą* a punkty  $s, s'$  *swap parą*. Jeżeli  $\phi_P(S') \leq \phi_P(S)$  to zastępujemy zbiór  $S$  zbiorem  $S'$ , w przeciwnym przypadku  $S$  pozostaje bez zmian. W praktyce taki proces powtarzamy do momentu, kiedy  $|\phi_P(S') - \phi_P(S)| < \epsilon$ . Formalnie możemy udowodnić, że dla każdego  $\epsilon > 0$ , po wielomianiowej liczbie wymian punktów  $s, s'$  algorytm zakończy swoje działanie. Autorzy nie dowodzą tego wprost ale powołują się na pracę [3].

Dla uproszczenia zakładamy, że algorytm kończy się kiedy pojedyncza wymiana elementów  $s, s'$  nie poprawia wyniku. Taki zbiór centrów nazwiemy *1-stable*.

Wprowadźmy notację:

$$\Delta(u, v) = d(u, v)^2 = \sum_{i=0}^d (u_i - v_i)^2 = (u - v) \cdot (u - v)$$

gdzie punkty interpretujemy jako wektory oraz operacja  $(\cdot)$  jest iloczynem skalarnym.

$$\begin{aligned} \Delta(S, v) &= \sum_{u \in S} d(u, v)^2 \\ \phi_P(S) = \Delta(S) &= \sum_{q \in P} \Delta(q, s_q) = \sum_{q \in P} d(q, s_q)^2 \end{aligned}$$

gdzie  $s_q$  to najbliższy punkt dla  $q$  w  $S$  oraz  $P \subset \mathbb{R}^d$ .

**Definicja 4.8.** Zbiór  $S$  nazywamy *1-stable*, jeżeli:

$$\Delta(S - \{s\} \cup \{c\}) \leq \Delta(S)$$

dla dowolnych  $s \in S$  oraz  $c \in C_{opt}$ .

**Definicja 4.9.** Niech  $N_S(s)$  oznacza sąsiedztwo punktu  $s$ , czyli zbiór punktów z  $P$ , dla których punkt  $s$  jest najbliższym spośród punktów z  $S$ .

W ramach tego podrozdziału udowodnimy następujące twierdzenie.

**Twierdzenie 4.2.** [9] Niech  $S$  będzie zbiorem  $k$  punktów spełniającym definicję 1-stable oraz niech  $C_{opt}$  będzie optymalnym zbiorem dla problemu  $k$ -means. Wtedy zachodzi następującą nierówność:

$$\Delta(S) \leq 25\Delta(C_{opt})$$

**Lemat 3.** [9] Dla danego skończonego podzbioru  $S$  punktów z  $\mathbb{R}^d$ , niech  $c$  będzie centroidem dla  $S$ . Wtedy dla dowolnego  $c' \in \mathbb{R}^d$ ,  $\Delta(S, c') = \Delta(S, c) + |S|\Delta(c, c')$ .

*Dowód.* Z definicji  $\Delta(S, c')$  otrzymujemy:

$$\begin{aligned} \Delta(S, c') &= \sum_{u \in S} \Delta(u, c') = \sum_{u \in S} (u - c')(u - c') \\ &= \sum_{u \in S} ((u - c) + (c - c'))((u - c) + (c - c')) \\ &= \sum_{u \in S} ((u - c)(u - c)) + 2((u - c)(c - c')) + ((c - c')(c - c')) \\ &= \Delta(S, c) + 2\left((c - c') \sum_{u \in S} (u - c)\right) + |S|((c - c')(c - c')) \\ &= \Delta(S, c) + |S|\Delta(c, c') \end{aligned}$$

Ostatnie przejście korzysta z faktu, że jeżeli  $c$  jest centroidem  $S$  to z definicji zachodzi  $\sum_{u \in S} (u - c) = 0$ .  $\square$

Na potrzebę dowodu założymy, że znamy zbiór  $C_{opt}$ . Dla każdego optymalnego  $c \in C_{opt}$  wyznaczamy  $s_c$ , który jest najbliższym punktem w  $S$  dla punktu  $c$ . Punkt  $s_c$  nazywamy heurystycznym centrum dla punktu  $c$ . W takim kontekście powiemy, że  $c$  jest *schwywane* przez  $s_c$ . Tutaj warto zaznaczyć, że każde optymalne centrum jest schwywane przez jedno heurystyczne centrum, ale każde heurystyczne centrum może być schwywane przez kilka optymalnych centrów. Heurystyczne centrum nazwiemy *samoctnym* jeżeli nie jest schwywane przez żadne optymalne centrum.

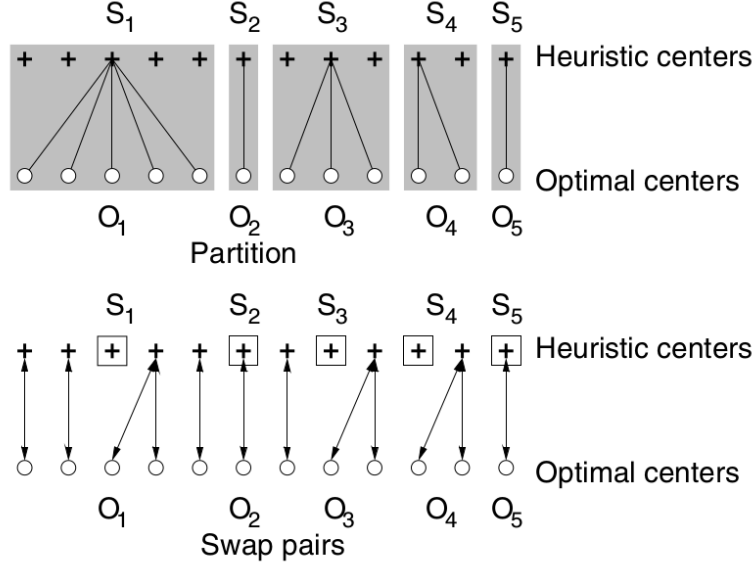
*Dowód.* Dowód twierdzenia 4.2 zaczniemy od zdefiniowania podziału  $S$  oraz  $C_{opt}$  na zbiory  $S_1, \dots, S_r$  oraz  $O_1, \dots, O_r$  dla pewnego  $r$ , gdzie  $|S_i| = |O_i|$  dla  $i = 1, \dots, r$ .

Dla każdego heurystycznego centrum  $s_i$ , które schwywało jakąś liczbę  $m \geq 1$  optymalnych centrów, tworzymy zbiór  $S_i$ , który będzie zawierał  $s_i$  oraz dowolne  $m - 1$  osamotnionych heurystycznych centrów. Analogicznie, zbiór  $O_i$  będzie zawierał wszystkie optymalne centra schwywane przez  $s_i$ . Rysunek 4.2 obrazuje tak zdefiniowany podział.

Wygenerujemy teraz *swap par* dla utworzonego podziału. Dla każdego podzbioru podziału takiego, że  $|S_i| = |O_i| = 1$ , tworzymy z ich elementów parę. Dla każdego podzbioru podziału, który zawiera więcej schwytych centrów, czyli  $|S_i|, |O_i| \geq 1$ , tworzymy pary między osamotnionymi heurystycznymi centrami z  $S_i$  a optymalnymi centrami z  $O_i$ . Każde optymalne centrum jest związane z



jednym heurystycznym oraz każde osamotnione centrum jest przyporządkowane co najwyżej dwóm optymalnym centróm. Centra łączymy dowolnie. Rysunek 4.2 przedstawia przykładowe swap pary.



Rysunek 4.2

Wyznamy teraz ograniczenie górne na zmianę funkcji  $\Delta$  po wymianie punktów ze swap pary  $(s, o)$ . Zaczniemy od obliczenia najbliższych centrów z  $S - \{s\} \cup \{o\}$  dla zadanego na wejściu zbioru  $P$ . Niech  $N_X(x)$  będzie *śsiedztwem* punktu  $x$ , czyli podzbiorem punktów z  $P$ , dla których  $x$  jest najbliższym punktem spośród punktów z  $X$ . Dla punktów, które należą do  $N_{C_{opt}}(o)$  zmiana  $\Delta$  będzie następująca:

$$\sum_{q \in N_{C_{opt}}(o)} (\Delta(q, o) - \Delta(q, s_q))$$

Każde  $q \in N_S(s) \setminus N_{C_{opt}}(o)$  straciło przypisane mu centrum  $s$  zatem punkt  $q$  musi otrzymać nowe centrum. Niech  $o_q$  będzie oznaczało najbliższe centrum dla punktu  $q$ . Skoro  $q \notin N_{C_{opt}}(o)$  to  $o_q \neq o$ , zatem  $s$  nie schwyciło  $o_q$ . Zatem po skorzystaniu ze swap pary  $(s, o)$ ,  $s_{o_q}$ , najbliższe heurystyczne centrum dla  $o_q$  nadal istnieje. Zmiana  $\Delta$  po wyborze nowych centrów jest co najwyżej równa:

$$\sum_{q \in N_S(s_i) \setminus N_{C_{opt}}(o_i)} (\Delta(q, s_{o_q}) - \Delta(q, s))$$

Na tym etapie dowodu konieczne będzie wprowadzenie dwóch lematów.

**Lemat 4.** [9] Niech  $S$  będzie zbiorem  $k$  punktów spełniającym definicję 1-stable oraz niech  $C_{opt}$  będzie optymalnym zbiorem dla problemu  $k$ -means. Wtedy zachodzi następującą nierówność:

$$0 \leq \Delta(C_{opt}) - 3\Delta(S) + 2R$$

gdzie,  $R = \sum_{q \in P} \Delta(q, s_{o_q})$ .

*Dowód.* Na potrzeby dowodu ustalmy swap parę  $(s, o)$ . Ponieważ  $S$  jest 1-stable to:

$$\begin{aligned} 0 &\leq \Delta(S) - \Delta(S - \{s\} \cup \{o\}) \\ &= \sum_{q \in N_{C_{opt}}(o)} (\Delta(q, o) - \Delta(q, s_q)) + \sum_{q \in N_S(s) \setminus N_{C_{opt}}(o)} (\Delta(q, s_{o_q}) - \Delta(q, s)) \end{aligned}$$

Aby rozszerzyć sumę na wszystkie możliwe swap pary zauważmy, że dla każdego optymalnego centrum, jest ono wymienione tylko raz. Zatem każdy punkt  $q$  kontrybuuje w pierwszej sumie tylko raz. Po drugie zważmy, że różnica w drugiej sumie jest zawsze niezerowa dlatego rozszerzając zakres sumowania możemy tylko zwiększyć sumaryczny wynik.

$$\begin{aligned} 0 &\leq \sum_{q \in P} (\Delta(q, o_q) - \Delta(q, s_q)) + \sum_{q \in P} (\Delta(q, s_{o_q}) - \Delta(q, s_q)) \\ 0 &\leq \sum_{q \in P} \Delta(q, o_q) - 3 \sum_{q \in P} \Delta(q, s_q) + 2 \sum_{q \in P} \Delta(q, s_{o_q}) \\ 0 &\leq \Delta(C_{opt}) - 3\Delta(S) + 2R \end{aligned}$$

□

Wcześniej zdefiniowane  $R$  nazywamy sumarycznym kosztem przepisania centrów. Korzystając z lematu 3 przekształcamy:

$$\begin{aligned} R &= \sum_{o \in C_{opt}} \sum_{q \in N_{C_{opt}}(o)} \Delta(q, s_o) = \sum_{o \in O} \Delta(N_{C_{opt}}(o), s_o) \\ &= \sum_{o \in C_{opt}} (\Delta(N_{C_{opt}}(o), o) + |N_o(C_{opt})| \Delta(o, s_o)) \\ &= \sum_{o \in C_{opt}} \sum_{q \in N_{C_{opt}}(o)} (\Delta(q, o) + \Delta(o, s_o)) \\ &\leq \sum_{o \in C_{opt}} \sum_{q \in N_{C_{opt}}(o)} (\Delta(q, o) + \Delta(o, s_q)) \\ &= \sum_{q \in P} (\Delta(q, o_q) + \Delta(o_q, s_q)) \end{aligned}$$

gdzie ostatnia nierówność bazuje na fakcie, że dla każdego  $q \in N_{C_{opt}}(o)$  mamy  $\Delta(o, s_o) \leq \Delta(o, s_q)$ . Następnie korzystamy z nierówności trójkąta.

$$\begin{aligned} R &\leq \sum_{q \in P} \Delta(q, o_q) + \sum_{q \in P} (d(o_q, q) + d(q, s_q))^2 \\ &= \sum_{q \in P} \Delta(q, o_q) + \sum_{q \in P} (d(o_q, q)^2 + 2d(o_q, q)d(q, s_q) + d(q, s_q)^2) \\ &= 2 \sum_{q \in P} \Delta(q, o_q) + \sum_{q \in P} \Delta(q, s_q) + 2 \sum_{q \in P} d(o_q, q)d(q, s_q) \\ &= 2\Delta(C_{opt}) + \Delta(S) + 2 \sum_{q \in P} d(o_q, q)d(q, s_q) \end{aligned}$$

**Lemat 5.** [9] Niech  $\langle o_i \rangle$  oraz  $\langle s_i \rangle$  będą ciągami liczb rzeczywistych, dla których zachodzi:

$$\delta^2 = \frac{\sum_i s_i^2}{\sum_i o_i^2}$$

dla pewnego  $\delta > 0$ . Wtedy:

$$\sum_{i=1}^n o_i s_i \leq \frac{1}{\delta} \sum_{i=1}^n s_i^2$$

*Dowód.* Z nierówności Schwarza:

$$\begin{aligned} \sum_{i=1}^n o_i s_i &\leq \left( \sum_{i=1}^n o_i^2 \right)^{\frac{1}{2}} \left( \sum_{i=1}^n s_i^2 \right)^{\frac{1}{2}} \\ &= \left( \frac{1}{\delta^2} \sum_{i=1}^n s_i^2 \right)^{\frac{1}{2}} \left( \sum_{i=1}^n s_i^2 \right)^{\frac{1}{2}} \\ &= \frac{1}{\delta} \sum_{i=1}^n s_i^2 \end{aligned}$$

□

Niech  $\langle o_i \rangle$  będzie ciągiem  $d(q, o_q)$  oraz niech  $\langle s_i \rangle$  będzie ciągiem  $d(q, s_q)$  dla wszystkich  $q \in P$ . Z tego wynika, że współczynnik aproksymacji możemy przedstawić jako:

$$\delta^2 = \frac{\Delta(S)}{\Delta(C_{opt})} = \frac{\sum_{q \in P} d(q, s_q)^2}{\sum_{q \in P} d(q, o_q)^2} = \frac{\sum_{i=1}^n s_i^2}{\sum_{i=1}^n o_i^2}$$

gdzie  $S$  jest zbiorem 1-stable uzyskanym przez wcześniej przedstawiony algorytm. Korzystając z lematu 5:

$$\begin{aligned} R &\leq 2\Delta(C_{opt}) + \Delta(S) + 2 \sum_{q \in P} d(o_q, q) d(q, s_q) \\ &\leq 2\Delta(C_{opt}) + \Delta(S) + \frac{2}{\delta} \sum_{q \in P} d(q, s_q)^2 \\ &= 2\Delta(C_{opt}) + \Delta(S) + \frac{2}{\delta} \Delta(S) \\ &= 2\Delta(C_{opt}) + \left(1 + \frac{2}{\delta}\right) \Delta(S) \end{aligned}$$

Z lematu 4 wiemy, że:

$$\begin{aligned} 0 &\leq \Delta(C_{opt}) - 3\Delta(S) + 2R \\ &= \Delta(C_{opt}) - 3\Delta(S) + 2\left(2\Delta(C_{opt}) + \left(1 + \frac{2}{\delta}\right)\Delta(S)\right) \\ &\leq 5\Delta(C_{opt}) - \left(1 - \frac{4}{\delta}\right)\Delta(S) \end{aligned}$$

Powyższą nierówność możemy przekształcić do postaci:

$$\begin{aligned}\frac{5}{1 - \frac{4}{\delta}} &\geq \frac{\Delta(S)}{\Delta(C_{opt})} = \delta^2 \\ 5 &\geq \delta^2 \left(1 - \frac{4}{\delta}\right) \\ 0 &\geq (\delta - 5)(\delta + 1)\end{aligned}$$

To implikuje, że  $\delta \leq 5$ , zatem współczynnik omawianego algorytmu możemy ograniczyć przez  $\delta^2 \leq 25$ , co kończy dowód twierdzenia 4.2.  $\square$

#### 4.4. Podsumowanie

W tej części przedstawimy algorytm budowania coresetu z [13]. Algorytm ten rozpoczynamy od wykonania 10-aproksymacji dla problemu  $k$ -means korzystając z pracy [4]. W części 4.3 opisaliśmy konstrukcję, której użyjemy w implementacji. Ma ona trochę większy współczynnik aproksymacji ale jak sami autorzy [13] stwierdzają w swojej pracy, nie ma to dużego znaczenia w kontekście całości. Dowolna aproksymacja o stałym błędzie może zostać użyta w tej metodzie. Na potrzeby analizy zakładamy, że wykonujemy algorytm 10-aproksymacyjny dla problemu  $k$ -means i uzyskaliśmy zbiór  $C'$  oraz, że mamy dany na wejściu zbiór punktów  $A \subset \mathbb{R}^d$ .

Geometryczna dekompozycja bazuje na dyskretyzacji punktów z  $A$ , czyli na zgrupowaniu ze sobą najbliższych punktów, a następnie zbudowaniu ważonego zbioru punktów  $S$  o zredukowanym rozmiarze. Taką technikę mogliśmy już zobaczyć w części 4.2, gdzie odpowiednio grupowaliśmy punkty w komórki kraty wykładniczej.

Praca [13] przedstawia inną technikę, która bazuje na budowaniu kul o wykładniczo rosnącym promieniu wokół każdego punktu z  $C'$ . Dla przybliżenia idei konstrukcji założymy, że znamy  $OPT = \phi_{opt}^k(A)$ . Algorytm zaczyna od budowy kul o promieniach równych  $\frac{1}{n}OPT$  a kończy na promieniach równych  $10OPT$ , gdzie  $n$  to moc zbioru  $A$ . Dla takich kul budujemy  $\epsilon$ -pokrycie kuli. W pracy pod pojęciem *kula* rozumiemy sferę w przestrzeni o wymiarze  $d$ .

**Lemat 6.** [14] *Niech  $U$  będzie sferą jednostkową w  $\mathbb{R}^d$ . Wtedy dla dowolnego  $\epsilon \in (0, 1)$ , istnieje  $\epsilon$ -pokrycie  $B$  o rozmiarze  $\left(1 + \frac{2}{\epsilon}\right)^d$ , czyli dla każdego punktu  $p \in U$  zachodzi:*

$$\min_{b \in B} \|p - b\| \leq \epsilon$$

Niestety nie istnieją efektywne metody budowania takich  $\epsilon$ -pokryć. My w analizie przyjmujemy, że  $|B| = \epsilon^{-O(d)}$ . Jako referencję jak zbudować taką konstrukcję autorzy [13] sugerują analizę pracy [6]. Jest to problematyczne w kontekście implementacji jednak tę kwestię poruszymy w następnym rozdziale.

**Lemat 7.** [13] Niech  $a, b, c$  będą punktami z  $\mathbb{R}^d$ . Wtedy dla dowolnego  $\epsilon \in (0, 1)$  zachodzi:

$$\left| \|a - c\|^2 - \|b - c\|^2 \right| \leq \frac{12}{\epsilon} \|a - b\|^2 + 2\epsilon \|a - c\|^2$$

**Lemat 8.** Niech  $A$  będzie zbiorem  $n$  punktów z  $\mathbb{R}^d$ ,  $B^i$  będzie kulą o promieniu  $r_i = \frac{2^i}{n} \sum_{x \in A} \|x\|^2$  o środku w punkcie o współrzędnych zerowych  $(0, \dots, 0)$  oraz niech  $S^i$  będzie  $\frac{\epsilon}{3}$ -pokryciem kuli  $B^i$  dla  $i = 1, \dots, \log 10n$ . Zdefiniujemy  $S = \bigcup_{i=0}^{\log 10n} S^i$ . Wtedy:

$$\sum_{x \in A} \min_{s \in S} \|x - s\|^2 \leq \epsilon^2 \sum_{x \in A} \|x\|^2$$

*Dowód.* Niech  $A_{close}$  będzie podzbiorem punktów z  $A$ , dla których kwadrat normy euklidesowej jest równy co najwyżej  $\frac{1}{n} \sum_{x \in A} \|x\|^2$  oraz niech  $A_{far}$  będzie zbiorem pozostałych punktów zbioru  $A$ . Ponieważ  $|A_{close}| \leq n$  oraz z definicji  $\epsilon$ -pokrycia wynika, że:

$$\sum_{x \in A_{close}} \min_{s \in S^0} \|x - s\|^2 \leq |A_{close}| \frac{1}{n} \frac{\epsilon^2}{9} \sum_{x \in A_{close}} \|x\|^2 \leq \frac{\epsilon^2}{9} \sum_{x \in A_{close}} \|x\|^2$$

Dla każdego punktu  $x$  ze zbioru  $A_{far}$  istnieje takie  $i$ , że  $x \in B^i \setminus B^{i-1}$  dla  $i \in \{1, \dots, \log 10n\}$ . Zatem:

$$\min_{s \in S^i} \|x - s\|^2 \leq \frac{\epsilon^2}{9} r_i^2 \leq \frac{4\epsilon^2}{9} r_{i-1}^2 \leq \frac{4\epsilon^2}{9} \|x\|^2$$

Sumując po wszystkich punktach otrzymujemy:

$$\sum_{x \in A} \min_{s \in S} \|x - s\|^2 \leq \frac{\epsilon^2}{9} \sum_{x \in A_{close}} \|x\|^2 + \frac{4\epsilon^2}{9} \sum_{x \in A_{far}} \|x\|^2 < \epsilon^2 \sum_{x \in A} \|x\|^2$$

□

Powyżej zdefiniowana procedura zaczyna konstrukcję w punkcie o współrzędnych zerowych  $(0, \dots, 0)$ . W dowodzie twierdzenia 4.3 wykorzystamy lemat 8, aplikując go do każdego punktu z  $C'$ .

**Twierdzenie 4.3.** Dla dowolnego zbioru  $n$  punktów  $A \subset \mathbb{R}^d$  istnieje coreset dla problemu  $k$ -means zawierający  $O(k\epsilon^{-d} \log n)$  punktów, gdzie  $d$  to wymiar (skończony) przestrzeni.

*Dowód.* Dla każdego z  $k$  centrów ze zbioru  $C'$ , który obliczyliśmy korzystając z algorytmu 10-aproksymacyjnego [4], tworzymy  $\log 10n$  kul o różnych promieniach. Dla każdej takiej kuli o promieniu  $r$  obliczamy  $\frac{\epsilon}{16}r$ -pokrycie oraz dla każdego punktu  $x \in A$ , niech  $B(x)$  będzie najbliższym punktem w sumie wszystkich pokryw kuli  $S$ . Z lematu 8 wynika, że:

$$\sum_{x \in A} \|x - B(x)\|^2 \leq \left(\frac{\epsilon}{16}\right)^2 \sum_{x \in A} \|x\|^2$$

Zauważmy, że koszt punktów  $A_c \subseteq A$ , dla których  $c \in C'$  jest centrem wynosi  $\sum_{x \in A_c} \|x - c\|^2$ . Jeżeli założymy, że punkt  $c$  jest początkiem przestrzeni to  $\sum_{x \in A_c} \|x - c\|^2 = \sum_{x \in A_c} \|x\|^2$ . Zatem:

$$\sum_{x \in A} \|x - B(x)\|^2 \leq \left(\frac{\epsilon}{16}\right)^2 \cdot 10 \cdot OPT$$

Teraz, rozpatrzmy dowolny zbiór  $k$  centrów  $C$ :

$$\begin{aligned} & \left| \sum_{x \in A} \min_{c \in C} \|x - c\|^2 - \sum_{s \in S} \min_{c \in C} \|s - c\|^2 \right| \\ & \leq \left| \sum_{x \in A} \min_{c \in C} \|x - c\|^2 - \sum_{x \in A} \min_{c \in C} \|B(x) - c\|^2 \right| \\ & \leq_{\text{lemat 7}} \frac{12}{\epsilon} \sum_{x \in A} \|x - B(x)\|^2 + 2\epsilon \sum_{x \in A} \min_{c \in C} \|x - c\|^2 \\ & \leq \frac{12}{\epsilon} \left(\frac{\epsilon}{16}\right)^2 \cdot 10 \cdot OPT + 2\epsilon \sum_{x \in A} \min_{c \in C} \|x - c\|^2 \\ & \leq 2\epsilon \cdot OPT + 2\epsilon \sum_{x \in A} \min_{c \in C} \|x - c\|^2 \\ & \leq 4\epsilon \sum_{x \in A} \min_{c \in C} \|x - c\|^2 \end{aligned}$$

gdzie ostatnia nierówność zachodzi ponieważ  $OPT \leq \sum_{x \in A} \min_{c \in C} \|x - c\|^2$  dla dowolnego zbioru center  $C$ .

Skalując  $\epsilon$  przez  $\frac{1}{4}$  kończymy dowód, a szukany zbiór to  $S$ . Rozmiar coresetu  $S$  to  $O(k\epsilon^{-d} \log n)$ , ponieważ obliczamy  $k \log 10n$  razy  $(\frac{\epsilon}{64})$ - pokrycie o rozmiarze  $\epsilon^{-O(d)}$ .  $\square$

## 5. Analiza Implementacji

TBA

## 6. Bibliografia

- [1] ALOISE, D., DESHPANDE, A., HANSEN, P., AND POPAT, P. Np-hardness of euclidean sum-of-squares clustering. *Machine Learning* 75 (05 2009), 245–248.
- [2] ARYA, S., MOUNT, D. M., NETANYAHU, N. S., SILVERMAN, R., AND WU, A. Y. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM* 45, 6 (Nov. 1998), 891–923.
- [3] ARYA, V., GARG, N., KHANDEKAR, R., MEYERSON, A., MUNAGALA, K., AND PANDIT, V. Local search heuristic for k-median and facility location problems. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2001), STOC '01, Association for Computing Machinery, p. 21–29.
- [4] ARYA, V., GARG, N., KHANDEKAR, R., MEYERSON, A., MUNAGALA, K., AND PANDIT, V. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.* 33 (2004), 544–562.
- [5] BACHEM, O., LUCIC, M., AND KRAUSE, A. Scalable k-means clustering via lightweight coresets.
- [6] CHAZELLE, B. *The Discrepancy Method: Randomness and Complexity*. Cambridge University Press, 2000.
- [7] FEDER, T., AND GREENE, D. Optimal algorithms for approximate clustering. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1988), STOC '88, Association for Computing Machinery, p. 434–444.
- [8] GONZALEZ, T. F. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.* 38 (1985), 293–306.
- [9] HAR-PELED, S., AND MAZUMDAR, S. On coresets for k-means and k-median clustering. 291–300.
- [10] LI, Y., LONG, P. M., AND SRINIVASAN, A. Improved bounds on the sample complexity of learning. *Journal of Computer and System Sciences* 62, 3 (2001), 516 – 527.
- [11] LLOYD, S. Least squares quantization in pcm. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137.



- [12] MATOUSEK, J. On approximate geometric k-clustering.
- [13] MUNTEANU, A., AND SCHWIEGELSHOHN, C. Coresets-methods and history: A theoreticians design pattern for approximation and streaming algorithms. *Künstliche Intell.* 32, 1 (2018), 37–53.
- [14] PISIER, G. *The Volume of Convex Bodies and Banach Space Geometry*. Cambridge Tracts in Mathematics. Cambridge University Press, 1989.