

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 przedstawię stan wiedzy na temat budowania coresetów w kontekście algorytmu K-means. W szczególności poruszę techniki takie jak geometryczna dekompozycja oraz losowe próbkowanie bazując na badaniach [7].

Celem pracy jest przedstawienie wyników teoretycznych [7] jak i implementacja technik budowania coresetów, które mogą mieć zastosowanie praktyczne [2] [5]. Następnie porównam ich działanie testując je na zbiorach punktów z dwuwymiarowej przestrzeni euklidesowej.

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 ich rozmiar nie był zależny od 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 , zbiór potencjalnych rozwiązań C oraz funkcję celu f . Chcemy znaleźć istotnie mniejszy zbiór danych S , który dla każdego potencjalnego rozwiązania $c \in C$ daje $f(S, c)$ *dobrze* aproksymujące $f(A, c)$.

Algorytmy budujące coresety są aplikowalne do wielu problemów klasteryzacji. W tej pracy skupię się na konstrukcjach dla problemu K-means, który należy do klasy problemów *NP-trudnych*. Najpopularniejszym algorytmem heurystycznym dla tego problemu jest algorytm Lloyd'a. Z uwagi na jego niską skalowalność jest on idealnym kandydatem do optymalizacji poprzez odpowiednią konstrukcję coresetu.

NOTACJA I NIEZBĘDNE DEFINICJE

K-MEANS

Zacznijmy od zdefiniowania problemu dla którego będziemy analizować konstrukcje coresetów.

Definicja 2.1. *Problem K-means.* Niech X to zbiór punktów z \mathbb{R}^d . Dla danego X chcemy znaleźć zbiór k punktów Q , który minimalizuje funkcję $\phi_X(Q)$ zdefiniowaną następująco:

$$\phi_X(Q) = \sum_{x \in X} d(x, Q)^2 = \sum_{x \in X} \min_{q \in Q} \|x - q\|_2^2$$

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

Definicja 2.2. *Problem K-means - wersja ważona.* Niech C to zbiór punktów z \mathbb{R}^d oraz niech w będzie funkcją $C \rightarrow \mathbb{R}_{\geq 0}$. Dla danego X oraz funkcji w chcemy znaleźć zbiór k punktów Q , który minimalizuje funkcję $\phi_X(Q)$ zdefiniowaną następująco:

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

Ewaluację funkcji ϕ dla optymalnego rozwiązania oznaczamy $\phi_{OPT}^k(X)$. Funkcję ϕ w literaturze nazywamy błędem kwantyzacji.

CORESET

To jak definiujemy coreset ściśle zależy od problemu, który optymalizujemy. Zacznijmy od podstawowej definicji coresetu dla problemu *K-means*.

Definicja 2.3. *Coreset.* Niech X to zbiór punktów z \mathbb{R}^d oraz niech Q to dowolny podzbiór X rozmiaru co najwyżej k . Zbiór C nazywamy (ϵ, k) coresetem jeżeli zachodzi:

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

Zauważmy, że taka definicja daje nam bardzo mocne gwarancje teoretyczne. Ewaluacji coresetu $\phi_C(Q)$ musi aproksymować $\phi_X(Q)$ z dokładnością $1 + \epsilon$ jednocześnie dla dowolnego zbioru k kandydatów na rozwiązanie Q w kontekście całego zbioru X . Jest to na tyle istotne, że w literaturze odróżnia się tą wersję nazywając ją *Strong Coreset*.

Definicja 2.4. *Coreset - weak.* Niech X to zbiór punktów z \mathbb{R}^d oraz niech Q będzie optymalnym rozwiązaniem. Słabym coresetem nazywamy zbiór C dla którego zachodzi:

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

LIGHTWEIGHT CORESET

Budowa coresetów w kontekście problemu K -means ma bardzo długą historię. Za przełomową pracę uznaje się [6], która jako pierwsza pokazuje konstrukcję $(1 + \epsilon)$ aproksymacji dla problemu K -means. Bazuje ona na budowie zbioru centroidów, czyli geometrycznym wariancie coresetu. Wyróżnia się trzy techniki budowania coresetów.

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

Pierwsza z technik cechuje się mocnymi gwarancjami teoretycznymi. Niestety większość rozwiązań jest mało praktyczna i kosztowna czasowo. Losowe próbowanie w praktyce daje bardzo przyzwoite wyniki jednak nie daje nam żadnej gwarancji odnośnie optymalności rozwiązania. Autorzy pracy [2] zaproponowali rozwiązanie o nazwie *lightweight coreset*, które w swoich założeniach ma łączyć:

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

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 to zbiór punktów z \mathbb{R}^d wraz ze średnią $\mu(X)$. Zbiór C jest (ϵ, k) lightweight coresetem dla X , jeżeli dla dowolnego zbioru $Q \subset X$ o mocy co najwyżej k 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 (2.2). Notacje *lightweight coresetu* możemy interpretować jako relaksację gwarancji teoretycznych zdefiniowanych w (2.2). Wprowadza ona oprócz błędu multiplikatywnego, błąd addytywny. Składnik $\frac{\epsilon}{2}\phi_X(Q)$ pozwala na odpowiednie skalowanie błędu aproksymacji dla funkcji ϕ . Druga część $\frac{\epsilon}{2}\phi_X(\mu(X))$ odpowiada za skalowalność rozwiązania zgodnie z wariancją danych.

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* udowodnię następujące twierdzenie.

Twierdzenie 3.1. *Niech $\epsilon \in (0, 1]$. Niech X będzie dowolnym zbiorem danych oraz niech C będzie (ϵ, 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. □

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

KONSTRUKCJA

Konstrukcja oparta jest na próbkowaniu z uwzględnieniem ważności danego punktu. Niech $q(x)$ będzie dowolnym rozkładem prawdopodobieństwa na zbiorze X oraz niech $Q \subset R^d$ będzie dowolnym potencjalnym zbiorem rozwiązań mocy k . Wtedy funkcję ϕ możemy zapisać jako:

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

Wynika z tego, że funkcja ϕ może być aproksymowana poprzez wylosowanie m punktów z X korzystając z $q(x)$ i przypisując im wagi odwrotnie proporcjonalne do $q(x)$. Dla dowolnej liczby próbek m oraz dla dowolnego rozkładu $q(x)$ możemy otrzymać sprawiedliwy (unbiased) estymator dla funkcji ϕ . Niestety, nie jest to wystarczające aby spełnić definicję (3.1). W szczególności musimy zagwarantować, jednostajność wyboru dowolnego zbioru k punktu Q z odpowiednim prawdopodobieństwem $1 - \delta$. Funkcja $q(x)$ może mieć wiele form, autorzy rekomendują postać:

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

Algorithm 1

procedure LIGHTWEIGHT ▷ Require: Set of data points X , coreset size m
 $\mu \leftarrow$ mean of X
 for $x \in X$ **do**
 $q(x) = \frac{1}{2} \frac{1}{|X|} + \frac{1}{2} \frac{d(x, \mu)^2}{\sum_{x' \in X} d(x', \mu)^2}$
 $C \leftarrow$ sample m weighted points from X where each point x has weight $\frac{1}{mq(x)}$ and is sampled with probability $q(x)$
 return lightweight coreset C

Pierwszy składnik rozkładu $q(x)$ 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 ϕ . Musimy więc zapewnić, odpowiednią częstotliwość wyboru takich punktów. Jak pokazuje pseudokod, implementacja takiej konstrukcji jest całkiem prosta. Zauważmy, że jest ona też bardzo praktyczna. Algorytm przechodzi przez zbiór danych jedynie dwukrotnie, a jego złożoność to $O(nd)$. Nie mamy zależności od k co jest kluczowe w konsekwencji praktyczności takiego rozwiązania.

ANALIZA

W tej części chciałbym udowodnić, że zaproponowany w poprzedniej części algorytm oblicza lightweight coreset dla odpowiedniego m .

Twierdzenie 3.2. *Niech $\epsilon > 0$, $\delta > 0$ oraz $k \in \mathbb{N}$. Niech X to zbiór punktów z \mathbb{R}^d oraz C zbiorem, który dostajemy z algorytmu dla*

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

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

Dowód. Na początku wyprowadzę rozkład dla punktów $x \in X$ uwzględniający ich wagę. Następnie pokażę, że wybierając wystarczającą liczbę punktów z tego rozkładu uzyskuje się (ϵ, k) lightweight coreset.

Zacznę od ograniczenia wagi 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 dowodzę następujący lemat.

Lemat 1. *Niech X to zbiór 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

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

Biorąc średnie wszystkich $x \in X$ otrzymujemy:

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

To implikuje, że dla każdego $x \in X$ oraz $Q \subset \mathbb{R}$ 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 jednego punktu $x \in X$ a $f(Q)$ jest ograniczona dla każdego $Q \subset X$ przez:

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

Zdefiniuje $S = \sum_{x' \in X} s(x')$ zauważając, że $S = 32$ dla każdego zbioru X . Dzięki temu mogę zapisać rozkład $q(x)$ jako:

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

dla każdego $x \in X$. Rozpatruję funkcję:

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

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

$$\phi_X(Q) = \sum_{x \in X} d(x, Q)^2 = S|X|f(Q) \sum_{x \in X} \frac{s(x)}{S|X|} \frac{d(x, Q)^2}{f(Q)s(x)}$$

$$= S|X|f(Q) \sum_{x \in X} q(x)g_Q(x)$$

Wprowadźmy notację:

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

dzięki której 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 [2] nie dowodzą wprost tego ograniczenia, powołując się na inne prace. Dowód jest bardzo skompilowany i wykracza tematyką istotnie poza ramy tej pracy więc go pomijam. Korzystam 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. □

GEOMETRYCZNA DEKOMPOZYCJA

W tej części pracy przedstawię konstrukcję budowy coresetu bazującą na geometrycznej dekompozycji problemu. Punktem wyjścia były badania [7], na których bazuje opisany poniżej algorytm. Zgodnie z nimi budowa coresetu w kontekście problemu K-means to wieloetapowy proces, który jest sekwencją algorytmów z prac: [4] [5] [1] [7]. W rozdziale przyjmuję następujący porządek analizy konstrukcji:

- Na początku opiszę konstrukcję pomocnicze w kolejności: [4], [5], [1].
- W sekcji *Podsumowanie* opiszę właściwy algorytm z pracy [7]

ALGORYTM GONZALEZ'A

Pierwszy algorytm, który opiszę to *Farthest point algorithm* z pracy [4]. Jest to pierwszy algorytm aproksymacyjny rozwiązujący problem k-centrów z błędem nie większym niż $2OPT$, gdzie OPT to optymalne rozwiązanie. Jego złożoność to $O(nk)$, gdzie n to liczba punktów.

Na potrzeby tego rozdziału wprowadzę kilka definicji

Definicja 4.1. 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 utożsamiam wagi z dystansem pomiędzy dwoma punktami.

Definicja 4.2. Podział zbioru wierzchołków na k zbiorów B_1, \dots, B_k , nazywamy *k-split*.

Definicja 4.3. Zbiory B_i podziału k-split nazywamy *klastrami*.

Dla każdego k-splitu definiujemy funkcję celu $f : B_1, \dots, B_k \rightarrow \mathbb{R}^+$. 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 .

Definicja 4.4. *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

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

Niech S to będzie zbiór który chcemy sklastrować oraz niech T będzie podzbiorem S . Zakładamy, że $|S| > k$ ponieważ w przeciwnym przypadku problem jest trywialnie rozwiązywalny.

Definicja 4.5. 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 . Udowodnię teraz następujący lemat, który jest potrzebny w dowodzie błędu aproksymacji algorytmu.

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

Dowód. Wartość funkcji celu dla kliki wysokości h to conajmniej h ponieważ kilka ma $k + 1$ elementów to 2 z nich wylądują w jednym klastrze. W takim razie waga krawędzi pomiędzy tymi punktami to conajmniej h co implikuje, że $OPT(S) > h$. \square

Algorytm składa się z fazy inicjalizującej oraz $k - 1$ faz powiększających. W fazie inicjalizującej wszystkie elementy są przypisane do zbioru B_1 , który jest pierwszym klastrzem. 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 odległość do centrum jego aktualnego klastra jest największa spośród wszystkich punktów. Taki punkt będzie oznaczony jako (t_{j+1}) , czyli jest centrem klastra B_{j+1} . Każdy punkt dla którego dystans do v jest nie większy niż dystans do centra klastru w którym się znajduje zostaje przeniesiony do B_{j+1} .

Algorithm 2

For each point not in T , the algorithm keeps $neighbor(p)$, the nearest point in T , and $dist(p)$, the distance from p to $neighbor(p)$.

procedure FARTHEST POINT ALGORITHM

$T \leftarrow \emptyset$

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

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

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

choose v from $S - T$ such $dist(v) = D$

add v to T

update $neighbor(p)$ and $dist(p)$ for all $p \in S - T$

return T

Taki algorytm buduje jakiś k -split. Teraz pokażę, że dla takiego k -splitu wartość funkcji celu jest ograniczona przez $2OPT(S)$.

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 centrum klastra dla punktów $x, y \in B_i$. A więc możemy ograniczyć wartość naszej funkcji celu przez $\leq 2h$. Ponieważ v nigdy nie zostało wybrane na

centrum klasta 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 $\leq 2h \leq 2OPT(S)$.

KONSTRUKCJA KRATY WYKŁADNICZEJ

TBA

HEURYSTYKA SINGLE SWAP

TBA

PODSUMOWANIE

TBA

ANALIZA IMPLEMENTACJI

TBA

BIBLIOGRAFIA

- [1] 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.
- [2] BACHEM, O., LUCIC, M., AND KRAUSE, A. Scalable k-means clustering via lightweight coresets.
- [3] 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.
- [4] GONZALEZ, T. F. Clustering to minimize the maximum intercluster distance. *Theor. Comput. Sci.* 38 (1985), 293–306.
- [5] HAR-PELED, S., AND MAZUMDAR, S. On coresets for k-means and k-median clustering. 291–300.
- [6] MATOUSEK, J. On approximate geometric k-clustering.
- [7] 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.

SPIS TREŚCI

1	Wstęp	1
2	Notacja i niezbędne definicje	2
2.0.1	K-means	2
2.0.2	Coreset	2
3	Lightweight Coreset	4
3.1	Lightweight coreset	4
3.2	Konstrukcja	5
3.3	Analiza	6
4	Geometryczna Dekompozycja	9
4.1	Algorytm Gonzalez’a	9
4.2	Konstrukcja kraty wykładniczej	11
4.3	Heurystyka single swap	11
4.4	Podsumowanie	11
5	Analiza Implementacji	12
6	Bibliografia	13