# Implementace paralelního algoritmu K-means

Cílem druhého projektu do předmětu paralelní a distribuované algoritmy (PRL) bylo implementovat paralelní algoritmus K-means s využitím knihovny OpenMPI. Tato dokumentace popisuje využitý shlukovací algoritmu spolu s jeho složitosti, komunikaci mezi jednotlivými procesy a příklad použití vytvořeného programu.

## 1 K-means

V této sekci ukážeme sekvenční a paralelní algoritmus K-means a jejich složitosti. Shlukování bylo prováděno na posloupnosti jednorozměrných hodnot. Za účelem projektu jsou vytvářeny 4 shluky (v důkazech složitosti bude počítáno s $k = 4$). Jako první hodnoty středů shluků jsou zvoleny první 4 hodnoty shlukované posloupnosti.

### 1.1 Sekvenční K-means

Na vstupu sekvenčního algoritmu je posloupnost (množina) jednorozměrných bodů $points$ a počet tvořených shluků. Jako středy ($means$) shluků je vybráno $k$ prvních bodů z posloupnosti $points$. Každý shluk ($cluster$) obsahuje množinu přiřazených bodů. Pro $i \in \mathbb{N} > 1$ platí $\forall n, m \in \mathbb{N} \leq k : cluster_i(n) \cap cluster_i(m) \neq \emptyset \iff n = m$ a také $\bigcup_{\forall n \in \mathbb{N} \leq k} cluster_i(n) = points$. V každé iteraci algoritmu jsou body přiřazeny do shluku, který má nejbližší střed. Pokud během iterace nedošlo ke změně žádného shluku, je algoritmus ukončen.

**Algorithm 1:** Sekvenční K-means

**Input:** $points$, $k \in \mathbb{N}$
**Output:** $cluster : k \rightarrow 2^{points}$

1 **forall** $p_n \in \{p_1, \ldots, p_k\} \subseteq points$ **do**
2     $mean(n) \leftarrow p_n$
3     $cluster_0(n) \leftarrow \emptyset$
4 **end**
5 $i \leftarrow 1$
6 **repeat**
7     **forall** $p \in points$ **do**
8         $c \leftarrow argmin_{n \in \mathbb{N}: n \leq k}(|mean(n) - p|)$
9         $cluster_i(c) \leftarrow cluster_i(c) \cup \{p\}$
10     **end**
11     **forall** $n \in \mathbb{N} : n \leq k$ **do**
12         $means(n) \leftarrow \frac{1}{|cluster_i(n)|} \sum_{p \in cluster_i(n)} p$
13     **end**
14     $i \leftarrow i + 1$
15 **until** $\forall n \in \mathbb{N} \leq k : cluster(n)_i = cluster_{i-1}(n)$
16 **return** $cluster_i$

#### 1.1.1 Časová složitost

Časová složitost sekvenčního algoritmu K-means, pro $K = 4$, je $O(n^2)$, kde $n$ je počet shlukovaných bodů.

*Důkaz.* Časová složitost inicializace shluků (1–4) je $O(k)$, kde $k$ je počet shluků. Složitost smyčky aktualizující shluky (6–15) je $O(n + nk) = O(n)$, kde $n$ je počet bodů. V [2] bylo ukázáno, že konvergence algoritmu nastává po $O(n^{kd})$, kde $d$ je počet dimenzí. Tato hranice byla v [1] upřesněna pro speciální případy, kdy $k < 5$ na $O(n)$ iterací. Z předchozího vyplývá, že celková časová složitost algoritmu 1 je $O(n^2)$. □

## 1.2 Paralelní K-means

Paralelní algoritmu K-means využívá $n$ procesů, kde $n$ je shodné s počtem shlukovaných bodů. Na vstupu algoritmu je posloupnost (množina) jednorozměrných bodů $points$ a počet tvořených shluků $k$. Jako středy ($means$) shluků je vybráno $k$ prvních bodů z posloupnosti $points$. Tyto středy jsou následně pomocí operace *Broadcast* rozeslány ostatním procesům. Každý procesor obsluhuje jeden vstupní bod. Procesor $i$ obsluhuje bod $p_i$. Každý proces obsahuje lokální proměnné $C^{(i)}$ a $changed^{(i)}$. $C^{(i)}$ určuje příslušnost bodu $p_i$ do shluku. $changed^{(i)}$ určuje, zda v současné iteraci algoritmu změnil bod $p_i$ shluk. Po ukončení rozmístění bodů do shluků jsou pomocí operace *Reduce* získány velikosti (počet bodů ve shluku) a sumy bodů v každém shluku. Pomocí těchto hodnot proces *root* přepočítá hodnoty středů shluků a operací *Broadcast* je rozešle všem procesům pro další iteraci výpočtu. Všechny lokální hodnoty $changed^{(i)}$ jsou operací *ORReduce* zredukovány do proměnné $changed$. Pokud je $changed$ rovno *True* (nějaký bod změnil shluk) je započata nová iterace. Výpočet je ukončen pokud už žádným bod nemění shluk.

**Algorithm 2:** Paralelní K-means

```
Input: points, k ∈ ℕ
Output: cluster : k → 2^points

forall p_n ∈ {p_1, ..., p_k} ⊆ points do
    mean(n) ← p_n
    Broadcast(means(n))
end
forall p_i ∈ {p_1, ..., p_n} = points do in parallel
    C^(i) ← 0                              // Přiřaď 0 do proměnné shluku C procesu i.
    changed^(i) ← True                     // Přiřaď True do proměnné changed procesu i.
end
changed ← True
while changed do
    forall p_i ∈ {p_1, ..., p_n} = points do in parallel
        c ← argmin_{n∈ℕ:n≤k}(|mean(n) − p_i|)
        changed^(i) ← c ≠ C^(i)                                        // Logický výraz
    end
    cluster_size ← GetSizesOfClusters()    // Redukcí získej velikosti všech shluků.
    cluster_sum ← GetSumInClusters()         // Redukcí získej sumu v každém shluku.
    forall n ∈ ℕ : n ≤ k do
        means(n) ← cluster_sum(n)/cluster_size(n)
        Broadcast(means(n))
    end
    changed ← ORReuce(changed^(1), ..., changed^(n))    // Redukce procesových changed.
end
forall n ∈ ℕ : n ≤ k do
    cluster(n) = {p_i ∈ points | C^(i) = n}
end
return cluster
```

### 1.2.1 Časová složitost

Časová složitost paralelního algoritmu K-means, pro $K = 4$, je $O(n \cdot log\, n)$, kde $n$ je počet shlukovaných bodů.

*Důkaz.* Časová složitost výběru středů a jejich rozeslání procesorům (1–4) je $O(log\, n)$. Složitost inicializace procesových proměnných (5–8) je $O(1)$. Smyčka pro výpočet nových středů má časovou složitost $O(k) = O(1)$. Časová složitost Reduce pro získání $cluster\_size$ (15), $cluster\_size$ (16) a $changed$ (21) je $O(log\, n)$. Přepočítání středů shluků a jejich rozeslání procesům (17–20) má časovou složitost $O(log\, n)$. Protože z [1] víme, že ke konvergenci pro $k < 5$ dochází po $O(n)$ iteracích hlavního cyklu (10–22), tedy celková časová složitost paralelního algoritmu je $O(n \cdot log\, n)$. □

### 1.2.2 Cena

Cena paralelního algoritmu K-means, pro $K = 4$, je $O(n^2 \cdot log\, n)$, kde $n$ je počet shlukovaných bodů. Algoritmus není optimální.

*Důkaz.* Dříve jsme ukázali, že časová složitost paralelního algoritmu K-means je $O(n \cdot log\, n)$, kde $n$ je počet shlukovaných bodů. Při použití $p$ procesů, kde $p = n$ je cena paralelního algoritmu $O(pn \cdot log\, n) = O(n^2 \cdot log\, n)$. Lze viděl, že paralelní algoritmus není efektivní, protože cena sekvenčního algoritmu je $O(n^2)$. □

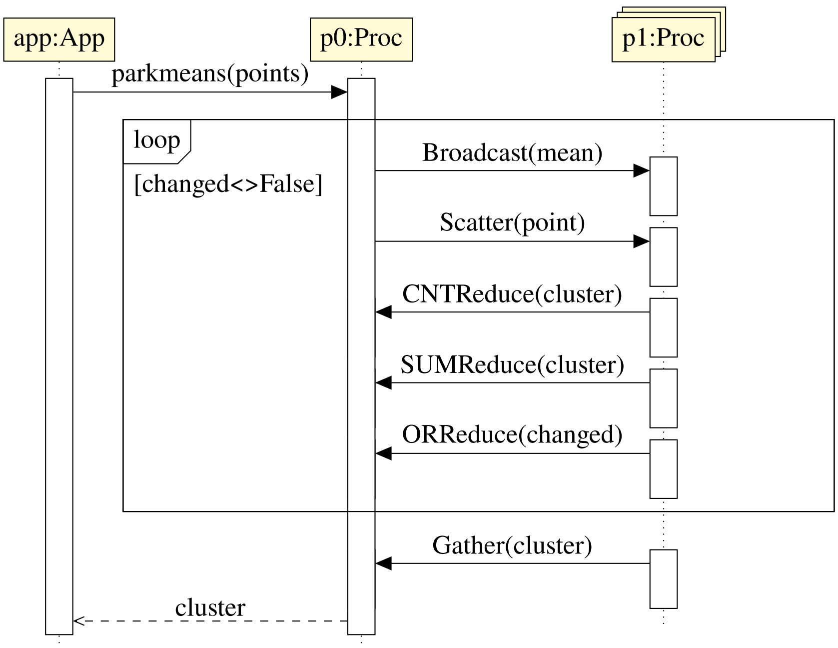
### 1.2.3 Prostorová složitost

Prostorová složitost paralelního algoritmu K-means, pro $K = 4$, je $O(n)$, kde $n$ je počet shlukovaných bodů.

*Důkaz.* Prostorová složitost proměnné $means$ je $O(k) = O(1)$. Proměnná $cluster$, a všechny lokální proměnné $c$ a $changed$ mají dohromady prostorovou složitost $O(3n) = O(n)$. Proměnné $cluster\_size$ a $cluster\_sum$ mají dohromady prostorovou složitost $O(1)$. Celková prostorová složitost paralelního algoritmu je tedy $O(n)$. □

## 2 Komunikace

Tato sekce pomocí sekvenčního diagramu popisuje komunikaci mezi aplikací a jednotlivými procesy v průběhu paralelního algoritmu K-means pro $n$ shlukovaných bodů.



## 3 Závěr

bla bla

## Reference

[1] David Arthur and Sergei Vassilvitskii. How slow is the k-means method? In *Proceedings of the Twenty-Second Annual Symposium on Computational Geometry*, SCG ’06, page 144–153, New York, NY, USA, 2006. Association for Computing Machinery.

[2] Mary Inaba, Naoki Katoh, and Hiroshi Imai. Applications of weighted voronoi diagrams and randomization to variance-based-clustering. pages 332–339, 06 1994.