Paralelná implementácia algoritmu K-means
PRL - Paralelní a distribuované algoritmy

Meno a priezvisko: Richard Seipel
Login: xseipe00

1 Popis algoritmu

Algoritmus K-means slúži na rozdelenie zadaných dát do K zhlukov. Na začiatku sú zvolené počiatočné hodnoty stredov zhlukov. Môžu byť vybrané napríklad náhodne alebo v prípade tohto projektu, ako prvých n hodnôt zo zadaných dát. Spracovávané dáta môžu mať ľubovoľný počet rozmerov, v našom prípade sú vstupné dáta jednorozmerné. Algoritmus spočíta vzdialenosť každého prvku od zvolených stredov zhlukov a priradí ho k zhluku s najbližším stredom. Po zaradení všetkých prvkov z dát sú vypočítané nové stredy zhlukov. Ak sa hodnoty stredov zmenili, algoritmus znova priradí prvky k zhlukom s najbližším stredom a pokračuje, až kým sa hodnoty stredov zhlukov neustália (nové stredy sú rovnaké ako predchádzajúce).

2 Paralelná implementácia

Paralelizácia algoritmu primárne spočíva v paralelnom výpočte vzdialenosti od stredov pre každé číslo (prvok) z dát v samostatnom procese. Na začiatku sú v hlavnom procese načítané vstupné dáta a overená ich dĺžka vo vzťahu k počtu spustených procesov. Taktiež sú ním vypočítané počiatočné hodnoty stredov zhlukov. Aby bolo možné zabezpečiť paralelizáciu, sú metódou `MPI_Scatter` vstupné dáta z hlavného procesu rozdelené medzi všetky procesy. Následne sú operáciou `MPI_Bcast` počiatočné hodnoty stredov z hlavného procesu zaslané všetkým ostatným procesom. Každý proces tak má priradený jeden prvok z dát v premennej `assigned_number` a prístup k všetkým stredom v K prvkovom poli `centroids`. Každý proces vykoná výpočet vzdialenosti a uloží index najbližšieho stredu do premennej `nearest_centroid`.

S vypočítanými indexami zhlukov v každom procese (pre danú hodnotu), je potrebné vypočítať hodnoty nových stredov. Pre tento výpočet je potrebné získané výsledky agregovať. Hodnotu jedného stredu dosiahneme podelením súčtu hodnôt v zhluku ich počtom. Pre získanie súčtu aj počtu hodnôt v jednotlivých zhlukoch, je použitá operácia `MPI_Reduce`. Tá pracuje s trojicou K prvkových polí.

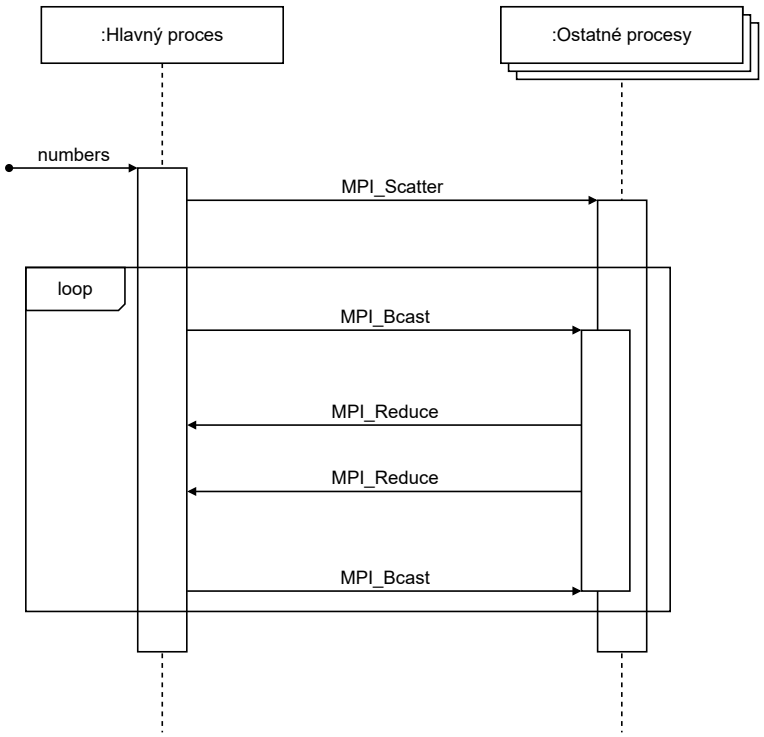
Prvým je lokálne pole `clusters_with_one_number`, do ktorého pri výpočte súčtu, každý proces vloží svoje priradené číslo z dát, na miesto (index) zhluku s najbližším stredom. Následne sú polia všetkých procesov sčítané operáciou `MPI_Reduce` s parametrom `MPI_SUM`. Výsledok je uložený v hlavnom procese do druhého poľa `sums_of_clusters`. To vo výsledku obsahuje súčty hodnôt pre každý z K zhlukov. Obdobne prebehne aj výpočet počtu prvkov, pre ktorý je znovu využité prvé pole, do ktorého sú tentokrát priradzované namiesto hodnôt prvkov len hodnoty `1`. Výsledné počty prvkov zhlukov sú ukladané v hlavnom procese do poľa `size_of_clusters`.

Po vykonaní týchto operácií, je z hodnôt v dvoch výsledných poliach možné dostať K výpočtami hodnoty nových stredov. Po ich výpočte sú výsledné hodnoty uložené

do poľa `new_centroids`. Výsledky sú porovnané s doterajšími hodnotami stredov v poli `centroids`. Ak sú rozdielne, doterajšie hodnoty sú nahradené novými a následne sú v novom cykle opäť rozoslané všetkým procesom pre nové výpočty vzdialeností.

Ak sú nové hodnoty rovnaké ako doterajšie, prebehne ukončenie cyklu počítania nových stredov vo všetkých procesoch. To je zabezpečené operáciou `MPI_Bcast`, ktorá hodnotu premennej `continue_loop` odošle každému procesu. Ak sa teda hodnoty stredov v určitej iterácii nezmenia, premenná `continue_loop` je nastavená na nulu a pri jej kontrole na začiatku každého cyklu je cyklus v každom z procesov ukončený.

Po ukončení výpočtu stredov zhlukov sú v jednom prechode poľa zhlukovaných hodnôt, všetky hodnoty roztriedené do výsledných zhlukov. Tie sú následne vypísané na výstup spolu s ich stredmi.



Obr. 1: Zasielanie správ medzi procesmi

3 Zložitosť algoritmu

Časová zložitosť načítania a kontroly vstupu je $O(n)$. Kopírovanie hodnôt stredov je $O(K)$, kde K je konštanta. Zložitosť operácie `MPI_Scatter` je $O(\log n)$. Nasleduje cyklus hľadania nových stredov, ktorého súčasťou sú operácie `MPI_Bcast` a `MPI_Reduce`. Obe využívajú pre šírenie informácie medzi procesmi stromový prístup, vďaka ktorému majú obe zložitosť $O(\log n)$. Ďalšie operácie v cykle, ako výpočet, porovnávanie a kopírovanie hodnôt stredov, či vyplnenie pomocného poľa nulami majú zložitosť $O(K)$. Po ukončení cyklu prebehne finálne zaradenie hodnôt do zhlukov a ich výpis, obe so zložitosťou $O(n)$.

Ak neberieme do úvahy načítanie vstupu a výpis výstupu, asymptotická časová zložitosť algoritmu je $O(i * \log n)$, kde i je počet iterácií. Cena algoritmu pri použití n procesorov je potom $n * i * \log n$.

Asymptotická priestorová zložitosť je $O(n * K)$, keďže priestorovo najnáročnejšími operáciami sú uloženie hodnôt K stredov v každom z n procesov a pomocné pole K hodnôt v každom z procesov, nad ktorým má byť vykonaná redukcia. Ostatné operácie majú priestorovú zložitosť $O(n)$, kde je buď jedna hodnota ukladaná v n procesoch alebo pole n hodnôt v jednom procese. Prípadne zložitosť $O(K)$, pri uložení poľa K hodnôt v hlavnom procese.

Výsledky hodnotím ako dostatočné a jednoznačne lepšie ako pri sekvenčnom riešení. Prístup použitia n procesorov pre n hodnôt však nie je veľmi prakticky využiteľný pre vyššie počty vstupných hodnôt.