Paralelização do algoritmo de clusterização kmeans

Darwin Saire Pilco[1] **RA: 163137**

[1]Instituto de Computação – Universidade Estadual de Campinas (UNICAMP)

Resumo. *Na atualidade se esta incrementando a quantidade de informação que se pode manipular, tornar-se um desafio o tratamento de dados e a diminuição de tempo de execução dos programas. O objetivo deste trabalho e apresentar idéia inicial para fazer uma paralelização do algoritmo de classificação kmeans. Embora a idéia é bastante simples os resultados dizem que foi capaz de alcançar aceleração (speedup).*

1. Formulação do problema.

A grande quantidade de informação e grande numero de atributos associados a cada instancia dos dados, aumentam o tempo e dificultade dos algoritmos de classificação. Os dados com alta numero de atributos tem uma maior dificuldade na classificação.
O algoritmo kmeans têm boa resposta e é bastante utilizado, mas uma das suas principais desvantagens é o tempo excessivo que precisa.

2. O algoritmo kmeans

Kmeans é o método de classificação, que tem como objetivo a partição de um conjunto de n elementos em k grupos, em que cada um dos elementos pertence para o grupo mais proximo deles. Kmeans é um dos algoritmos que têm boa resposta as provas, mas uma das suas principais desvantagens é o tempo excessivo que precisa para convergir quando o número de atributos(dimensões) e numero de iterações são grandes, porque tem uma complexidade de $O(n.k.d.I)$, onde n é o numero de elementos, k é o numero de cluster, d é o numero de dimensões(atributos) para cada elemento e I é o número de iterações.
O algoritmo Kmeans pode ser realizada em quatro passos.

1. Selecione a posição inicial dos centroides.
2. Para cada um dos elementos encontrar o centróide mais próximo a eles.
3. Atualizar os novos centróides.
4. Repita a partir do passo 2.

A paralelização das etapas do kmeans são de cor verde na figura 1.

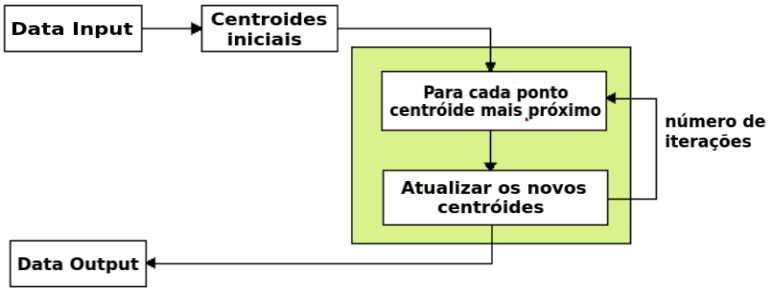

Figure 1. Etapas de paralelização são de cor verde

3. A Idea de paralelização de kmeans

A etapa 2 a cada um dos elementos encontrar o centróide mais próximo a cada um deles.

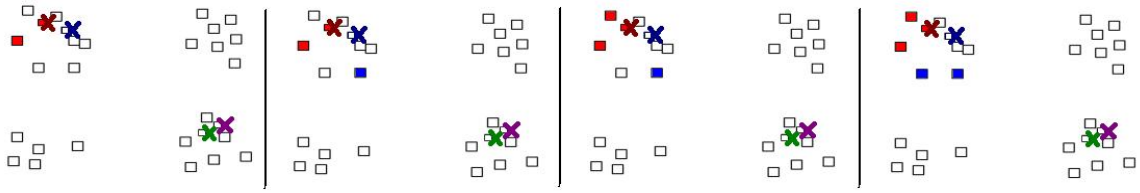

Figure 2. Etapas 2 do kmeans

O algoritmo tem que comparar cada um dos elementos(n) com cada um dos cluster(k) y pegar o mais proximo e vai fazer isso I veces. Onde em cada iteração do cluster vai mudar.

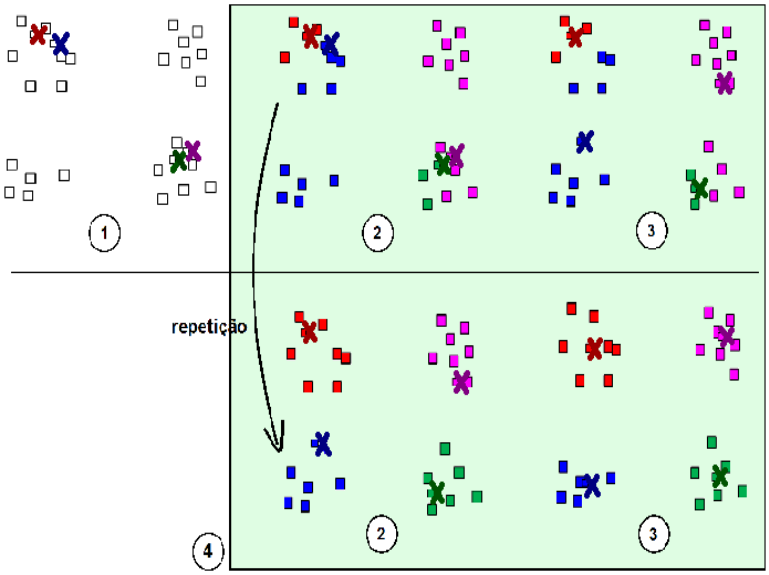

Figure 3. Algoritmo kmeans

Na seção 4 podemos ver os segmentos hotpoint, então a idéia para paralelizar o algoritmo é só na etapa 2 da image, não na etapa 2 e 3. Para a paralelização com Openmp colocamos um pragma na iteracão de todos os elementos, o qual chama a mais funções. Basicamente foi dividir o número de elementos entre o número de threads que são usados. Para cuda a ideia foi pegar cada elemento com uma thread, e usamos bloques de 256 threads.

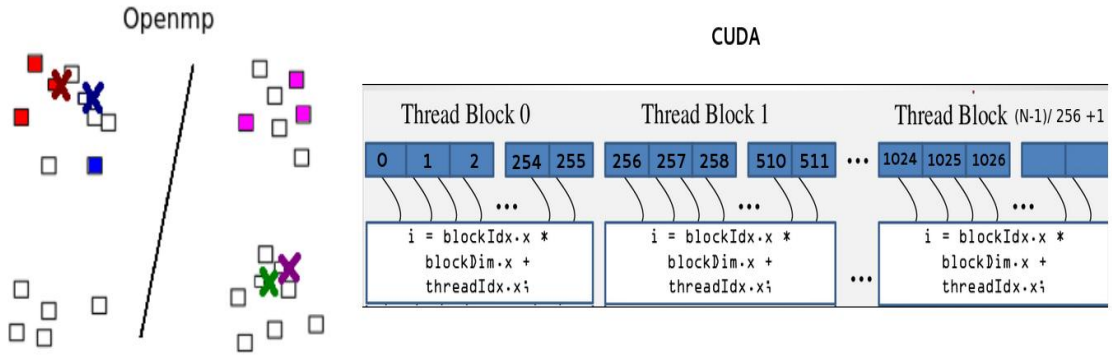

Figure 4. Paralelização algoritmo kmeans

4. Profile

No profile pomos ver os hotpoint do algoritmo kmeans, e como a gente pensou foi no segmento da iteração dos numero de elementos. Em seguida, o que foi feito foi para paralelizar esse laço.

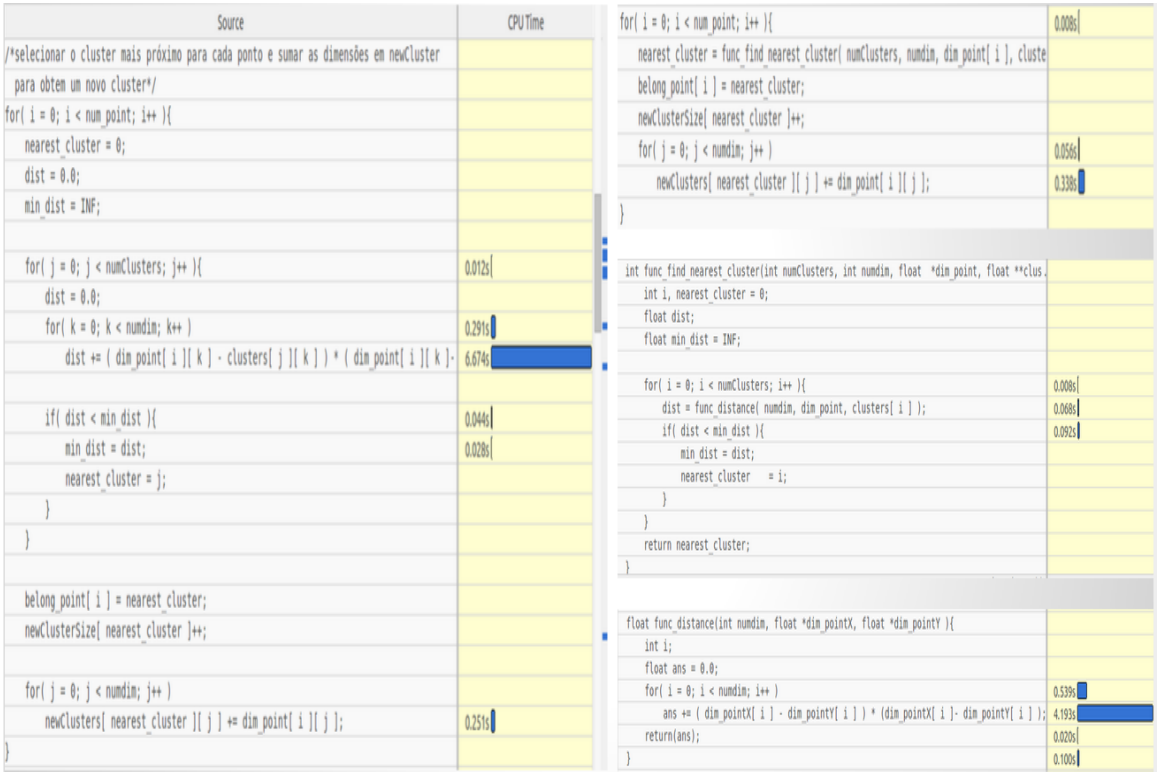

Figure 5. Hotpoint do algoritmo kmeans

5. Speedup

A base de dados foi http://cs.joensuu.fi/sipu/datasets/
A base content (N) 1024 elementos, a cantidade de atributos é especificada por o nome e content (k) 16 clusters

	dim128.txt		dim256.txt		dim512.txt		dim1024.txt	
	Tempo(s)	SpeeUp	Tempo(s)	SpeeUp	Tempo(s)	SpeeUp	Tempo(s)	SpeeUp
sequencial	8,2399		16,1725		32,1575		64,3928	
omp nt = 2	4,4694	1,84	8,4827	1,91	16,5957	1,9377	33,3151	1,9325
omp nt = 4	2,682	3,07	4,625	3,4968	8,9564	3,5905	17,4236	3,7171
omp nt = 8	1,923	4,28	2,8297	5,7151	5,1127	6,2897	9,7582	6,5988
omp nt = 16	1,8488	4,46	2,1247	7,6116	3,1544	10,1948	6,0392	10,6642
gpu	2,11	3,91	4,27	4,2743	7,33	4,487	14,16	4,5475

Figure 6. Tempo de execução e speedup

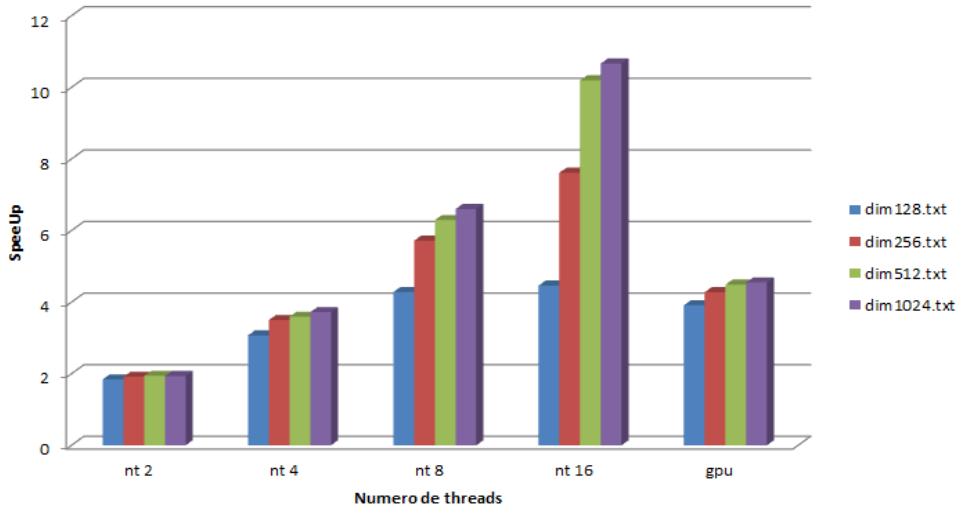

Figure 7. Tabela dos Speedup's