ci1316 - PROGRAMAÇÃO PARALELA - 2o semestre de 2025 - por W.Zola/UFPR

trab2-knnMPI: Dados 2 conjuntos P e Q de pontos no multi-dimensionais
Para cada ponto em Q, determinar o conjuntos de K vizinhos mais próximos (KNN) em P
Fazer esse trabalho com processos MPI

Histórico:
- v1.0 a versão inicial

Data do enunciado: 24/out/2025
Data da entrega: **10/nov/2025 (são 17 dias!)**

Esse trab pode ser em grupo de no MAXIMO 2 alunos, o nome dos autores devem estar no relatório e no progr. Fonte, a entrega será via UFPR Virtual
---
Objetivo do trabalho 2:
O objetivo deste exercício é fazer um trabalho que use paralelismo em MPI no cluster com rede ethernet e verificar a aceleração da nossa solução. OBS: **Todas as medições devem ser feitas no cluster w, MAS** você pode **desenvolver** em qualquer computador que tenha MPI.

.**O trabalho:**

A entrada para nossa função paralela são dois conjuntos de pontos de D dimensoes cada ponto (i.e. são pontos de muitas dimensões).
Dados os conjuntos de n pontos P e e nq pontos em Q, denominados respectivamente
P = Pontos da base (dataset) (P é uma **matriz de floats**)
Q = Pontos de consulta, send nq=|Q| o tamanho de Q. (Q é uma **matriz de floats**)

Dado um número inteiro k

Para cada ponto em Q, queremos determinar quais são seus k vizinhos mais próximos no conjunto P.

Fazer uma funçao knn que recebe como parâmetros: Q, nq, P, n, D, k

sendo: nq = numero de pontos em Q
n = numero de pontos em P
D = número de dimensoes dos pontos
k tamanho dos conjuntos de pontos vizinhos buscados

Note que o resultado (saída) da funçao knn deve ter nq conjuntos de tamanho k
Cada conjunto deve ter o indice de seus k vizinhos mais próximos

Ou seja, a saida de knn pode ser uma matriz de nq linhas por k colunas.

A figura na pagina seguinte ilustra o problema.

Nosso programa MPI deve obter as matrizes Q e P.
O conteúdo dessas matrizes poderia ser obtido de arquivos MAS
como o conjunto P pode ser grande, a leitura pode demorar,
então vamos gerar esses conjunto com dados de maneira aleatoria no processo 0 MPI (raiz).
Usando uma função geraConjuntoDeDados(C, nc, d)
que gera um conjuto qualquer **C** de **nc** pontos aleatorios de **d** dimensoes.

Todos as nossas coordenadas dos porntos em Q e P devem ser float.

Somente o processo 0 deve gerar os conjuntos Q e P.

Para nossas experiencias queremos um conjunto Q de 128 pontos de 300 dimensoes

e um conjunto P de 400mil pontos de 300 dimensoes

- SOBRE A IMPLEMENTACAO:

- Sua solução DEVE usar a adequadamente implementaçao do max-HEAP (chave, valor) e operacão deccreaseMax conforme figura dada explicando o funcionamento do max-heap (também foi dada uma implementação sequencial simples como referência)

- Voce DEVE usar a operação scatter de MPI na matriz **Q** da raiz para os nodos e a operação broadcast para enviar a matriz **P** para todos os nodos, e gather para "juntar" o resultado R na RAIZ

- todas as distancias calculadas, comparadas e mantidas no heap são **distancias quadraticas (distancia ao quadrado d^2, ou seja, NAO vamos usar operacoes de raiz quadrada** e nosso programa rodará mais rapido

- Todos os resultados reportados devem ser medidos no nosso cluster XEON e você deve fornecer os scripts para compilacao e para rodar suas experiencias, bem como as planilhas openoffice com seus resultados e graficos ALEM do relatório

- Os parametros para rodar o programa devem ser precedidos por seus nomes seguidos de = (em qualquer ordem), por exemplo:

  mpirun -np 4 knn-mpi nq=n1 npp=n2 d=n3 k=n4

  onde: (n1, n2, n3 e n4 são os números)

  nq = numero de pontos em Q (tamanho do conjunto de pesquisa)

  npp = numero de pontos em P (dataset)

  d = numero de dimensoes dos pontos (dimensionalidade)

  k = tamanho de cada conjunto de vizinhos (k vizinhos por ponto)

- ex:

  mpirun -np 4 knn-mpi nq=128 npp=400000 d=300 k=1024

  Calcula os 1024 vizinhos mais próximos de 128 pontos de 300 dimensoes em base de dados de 400mil pontos

- SOBRE AS EXPERIÊNCIAS:

- TODAS os resultados reportados DEVEM ser
- Faremos a computacao total para 3 casos apenas:
-
- Experiencia 1:
- Rodar o programa para APENAS 1 processo MPI e medir o tempo da computaçao de knn
-
- Experiencia 2:
- Rodar o programa para 8 processos MPI no mesmo host e medir o tempo da computaçao de knn
-
- Experiencia 3:
- Rodar o programa para um total de 8 processos MPI em 4 hosts diferentes e medir o tempo da computaçao de knn (ou seja, serão 2 processo por host)
-
- Reportar a aceleração (speedup) da sua implementaçao paralela em relação a 1 processo apenas, nos casos da experiencia 2 e 3.

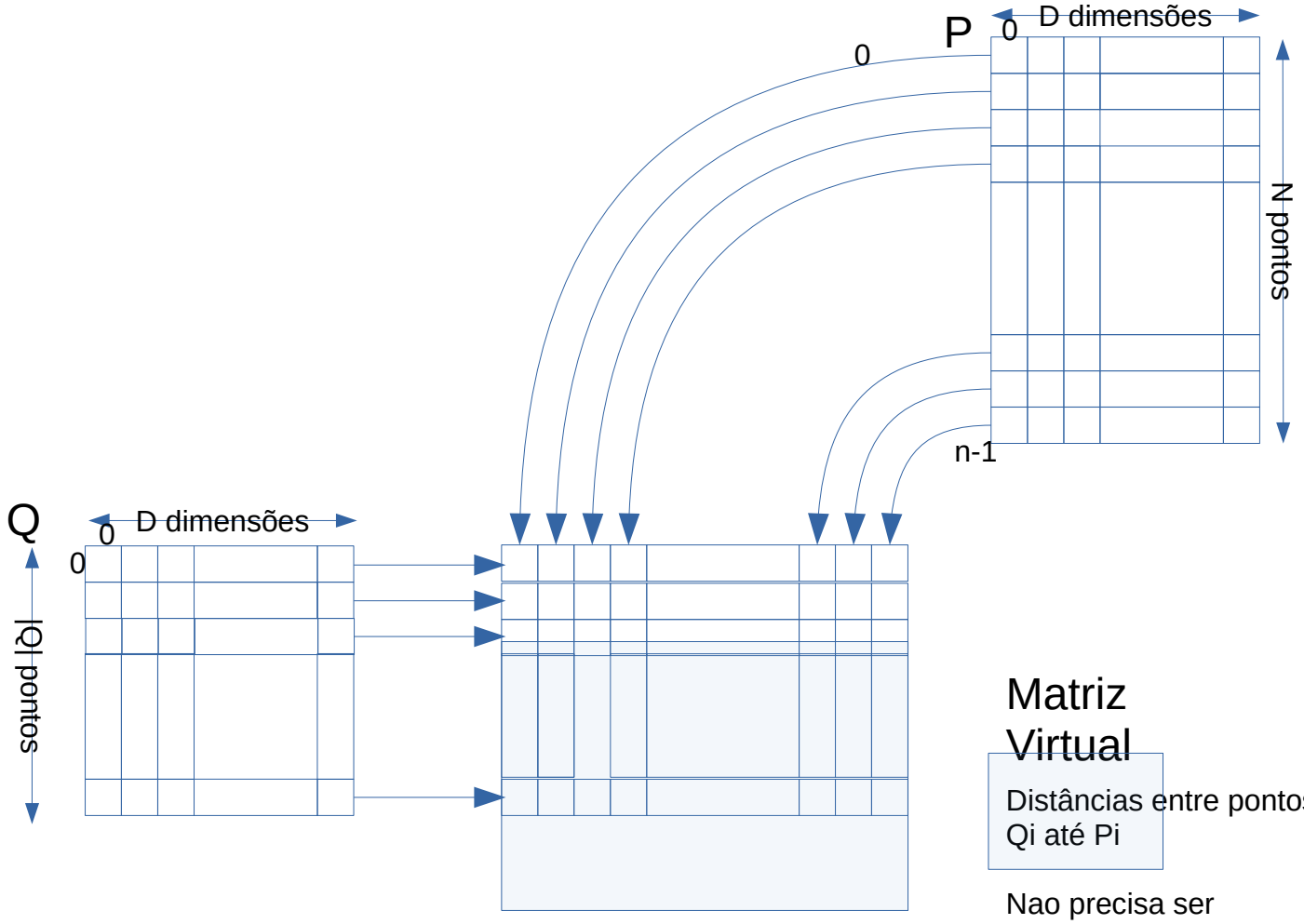
- SOBRE O RELATÓRIO do trabalho: ... (**próxima página**)

- SOBRE O RELATÓRIO do trabalho:
  - Fazer um relatório (PDF) com suas conclusões. Descreva as características importantes da sua CPU usada. Inclua em apendice ao relatório o texto da saida do comando lscpu bem como a figura do comando lstopo para sua CPU usada.
- SOBRE A funçao de vefificaçao a ser chamada
  - Incluir ao final do seu programa, ANTES de terminar o MPI e ANTES de desalocar suas matrizes, para rodar APENAS no processo 0 MPI, a funcao verificaKNN(...) que DEVE ter o mesmo prototipo abaixo. Voce deve colocar a funcao inicial de verificacao abaixo em um arquivo chamado verificaKNN.c e fazer o include desse arquivo no INICIO do seu programa.
  - A funçao inicial está abaixo, esta apenas vai imprimir o vetor de resultados. Para a verificacao adequada o prof irá substituir essa versao por outra que fara a verificacao (durante a correção do trabalho).

```
void verificaKNN( float *Q, int nq, float *P,  int n, int D, int k, int *R ) {
    // note que R tem nq linhas por k colunas, para qualquer tamanho de k (colunas)
    // entao é linearizado para acesso como um VETOR
    printf( "  ------------ VERIFICA KNN ---------------- " );
    for( int linha=0; linha<nq; linha++ )  {
       printf( "knn[%d]: ", linha );
       for( int coluna=0; coluna<k; coluna++ )
           printf( "%d ", R[ linha*k+coluna ] );
       printf( "\n" );
    }
}
```

  - OBS: na página seguinte temos a ilustraçao do prolema de calcular a distância entre TODOS os pares de pontos

ilustraçao do problema de calcular a distância entre TODOS os pares de pontos



Matriz Virtual

Distâncias entre pontos Qi até Pi

Nao precisa ser armazenada em memoria, basta calcular as distancias indicadas entre os pontos

Para comparar distancias, podemos evitar a raiz quadrada comparando o quadrado das distancias (sem extrair a raiz)