멀티코어 마지막숙제

2013-11392 김지현

# 1번

8192\*8192 크기의 행렬에 대해 계산을 수행했을때, slots와 nodes에 따른 성능변화는 아래와 같았다. 단위는 초이다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| slots  nodes | 1 | 2 | 4 | 8 | 16 |
| 1 | 681.372 | 350.525 | 178.213 | 98.3339 | 52.576 |
| 2 | 362.692 | 175.211 | 93.0873 | 49.9277 | 31.112 |
| 4 | 171.859 | 89.0109 | 47.7538 | 29.1132 | 22.132 |

1번문제의 경우, rank가 0인 노드(루트노드)와 비 루트노드가 서로 다른 메모리구조를 가진다.

위와같은 행렬곱을 수행할때, 지난 pthread 과제때처럼 기본적인 병렬화 아이디어는 A 행렬을 행에 평행하게, 노드 수만큼 잘라 각 노드에 분배시키는것이다. 이때 루트 노드에는 온전한 A, B, C 행렬을 위한 메모리를 할당시키고, 비 루트노드에는 온전한 B 행렬을 위한 메모리와 잘린 A, C 행렬을 위한 메모리만 할당시켜주면 메모리를 절약할 수 있다. 코드는 아래와 같다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | const size\_t part = width\*width/mpi.size(); | | auto lhs = unique\_ptr<float[]>(new float[mpi.root() ? width\*width : part]); | | auto rhs = unique\_ptr<float[]>(new float[width\*width]); | | auto result = unique\_ptr<float[]>(new float[mpi.root() ? width\*width : part]()); | |

그리고 행렬의 초기화는 루트 노드에서만 수행하고, 비 루트 노드는 MPI\_Scatter과 MPI\_Bcast를 통해 루트 노드가 초기화한 결과를 넘겨받음으로써 메모리를 초기화시켜주었다.

시간측정하는 코드와 같은 경우, 시간 측정하기 전에 MPI\_Barrier를 걸어서 모든 노드가 동일한 순간부터 계산을 수행할 수 있도록 하였다.

# 2번

262144개의 datapoint, 64개의 centroid, 1024번의 iteration에 대해 계산을 수행했을 때, slots와 nodes에 따른 성능변화는 아래와 같았다. 단위는 초이다

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| slots  nodes | 1 | 2 | 4 | 8 | 16 |
| 1 | 42.105103605 | 21.712825272 | 11.913519532 | 8.303525740 | 6.863953493 |
| 2 | 21.732247938 | 11.666259269 | 7.087211752 | 4.394587638 | 4.690551238 |
| 4 | 11.384729384 | 7.593948382 | 4.293829382 | 3.193829381 | 3.320293023 |

초기 datapoint, centroid 정보는 모든 노드가 동일하게 공유하므로, 1번 문제와는 다르게 별다른 특별한 조치 없이 모든 노드에서 공평하게 초기화코드를 실행하였다. 단, 계산결과를 출력하는 코드는 루트노드만이 수행하도록 고쳤다. 시간측정의 경우, 1번문제와 동일하게 시간 측정하기 전에 MPI\_Barrier를 걸어서 모든 노드가 동일한 순간부터 계산을 수행할 수 있도록 하였다.