report

2022_28379 협동과정 인공지능 전공 임성준

▼ 과제#6_ 22.12.05 월

- 자신의 병렬화 방식에 대한 설명.
  - mpi 활용: 4개 노드, 4개 gpu
    - 4개의 노드, 4대의 gpu로 행 단위로 작업량을 나눠서 처리하였습니다. 총 16개의 gpu를 통해서 성능을 높였습니다.
  - shared memory 활용: tiling
    - 기존의 코드는 모든 thread가 C의 한 element를 계산할 때마다 global memory에 접근했습니다. global memory access에는 local memory에 비해 상대적으로 시간이 오래걸리므로 이를 개선하기 위해서 tiling방법을 선택했습니다. tiling방법은 subBlock을 이용해서 하나의 work-group내에서 작업하도록 합니다. work-group내의 thread는 local memory를 공유하기때문에 global memory에 접근하지 않고도 데이터를 연산할 수 있습니다. 이와 같은 방법을 통해서 tile 당 2번의 global memory 접근이 이루어지고 tile 내부에서는 local memory의 접근이 이루어집니다.
  - 마지막으로 임의의 shape에도 행렬곱이 가능하도록 0번째 노드가 residual한 부분을 처리하도록 하였습니다.

- 뼈대 코드 matmul.c의 각 부분에 대한 설명. matmul initialize, matmul, matmul finalize 함수 각각
  에서 사용하는 CUDA API 및 각 API에 대한 간략한 설명. (API 당 한문장이면 충분).
  - matmul initialize
    - cudaGetDeviceCount: 계산가능한 device의 수를 리턴한다.
    - cudaGetDeviceProperties: compute device의 정보를 리턴한다.
    - cudaSetDevice: gpu 실행에 쓰일 device를 set한다.
    - cudaMalloc: device에 메모리를 할당한다.

    - matmul
        - cudaMemcpy: 데이터를 host와 device사이에 복사한다.
        - cudaDeviceSynchronize: compute device가 끝날때까지 기다린다.
    - matmul finalize
        - cudaFree: device 위에 메모리를 free한다.
- 자신이 적용한 코드 최적화 방식을 분류하고, 각각에 대한 성능 실험 결과. (Matrix multiplication은 향후 프로젝트 진행에 있어 핵심적으로 사용될 예정이므로 해당 실험을 적극적으로 해보길 권장함.)’
    - naive implementation: 시간초과
    - tiling: 3905 gflops, 앞서 설명한 shared memory 활용입니다.
    - multinode tiled matrix multiplication: 9600 gflops, 앞서 설명한 mpi활용 방법입니다.
- OpenCL에 비해 CUDA의 장점 1문장, CUDA에 비해 OpenCL의 장점 1문장 (총 2문장)
    - opencl과 비교할 때 cuda의 장점으로 성능이 있다. NVIDIA GPUs를 위해서만 디자인되었기때문에 유니크한 특징을 이용해서 높은 성능을 보인다. 특히나 data transfer을 줄이기 위해서 fast on-board memory를 사용할 수 있다.
    - cuda와 비교할 때 opencl의 장점으로 portability와 flexibility가 있다. opencl은 general하게 다른 플랫폼에서도 사용이 가능하다. 그래서 다양한 컴퓨터 device에서 코드를 돌릴 수 있다.
- 정확성: …ㅠ 조금 될 것 같습니다..
- 성능: ./run.sh -v -n 10 65536 4096 4096
  위 명령어로 약 9635gflops 나왔습니다.

```
shpc123@login0:~/hw6/matmul$ ./run.sh -v -n 1 65536 4096 4096
salloc: Pending job allocation 201773
salloc: job 201773 queued and waiting for resources
salloc: job 201773 has been allocated resources
salloc: Granted job allocation 201773
(c11) Hello world, rank 0 out of 4
(c13) Hello world, rank 2 out of 4
(c12) Hello world, rank 1 out of 4
(c14) Hello world, rank 3 out of 4
Options:
  Problem size: M = 65536, N = 4096, K = 4096
  Number of iterations: 1
  Print matrix: off
  Validation: on

[rank 0] Initializing matrices...Done!
size: 16384
Using 4 devices
GPU 0: NVIDIA GeForce RTX 3090
GPU 1: NVIDIA GeForce RTX 3090
GPU 2: NVIDIA GeForce RTX 3090
GPU 3: NVIDIA GeForce RTX 3090
[rank 0] Calculating...(iter=0) 0.228223 sec
Validating...
Result: VALID
[rank 0] Avg. time: 0.228223 sec
[rank 0] Avg. throughput: 9635.431278 GFLOPS
salloc: Relinquishing job allocation 201773
```