HW5 report

2022-25448
박기범

1(a)

```
shpc033@login0:~/hw5/matmul$ srun --partition=class1 --gres=gpu:4 nvidia-smi
Sat Nov 25 17:26:18 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 520.61.05    Driver Version: 520.61.05    CUDA Version: 11.8     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA TITAN RTX    On   | 00000000:18:00.0 Off |                  N/A |
| 41%   36C    P8    35W / 280W |      0MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA TITAN RTX    On   | 00000000:3B:00.0 Off |                  N/A |
| 41%   33C    P8    25W / 280W |      0MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  NVIDIA TITAN RTX    On   | 00000000:86:00.0 Off |                  N/A |
| 40%   33C    P8    28W / 280W |      0MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  NVIDIA TITAN RTX    On   | 00000000:AF:00.0 Off |                  N/A |
| 41%   32C    P8    24W / 280W |      0MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

shpc033@login0:~/hw5/matmul$ srun --partition=class1 --gres=gpu:4 nvidia-smi -q

==========NVSMI LOG==========

Timestamp : Sat Nov 25 17:28:06 2023
Driver Version : 520.61.05
CUDA Version : 11.8

Attached GPUs : 4
GPU 00000000:18:00.0
    Product Name : NVIDIA TITAN RTX
    Product Brand : Titan
    Product Architecture : Turing
    Display Mode : Disabled
    Display Active : Disabled

```
    Persistence Mode                    : Enabled
    MIG Mode
        Current                         : N/A
        Pending                         : N/A
    Accounting Mode                     : Disabled
    Accounting Mode Buffer Size         : 4000
    Driver Model
        Current                         : N/A
        Pending                         : N/A
    Serial Number                       : 1324419071615
    GPU UUID                            : GPU-15e04221-2ac6-4d81-5941-8614ed1afbb4
    Minor Number                        : 0
    VBIOS Version                       : 90.02.2E.00.0C
    MultiGPU Board                      : No
    Board ID                            : 0x1800
    GPU Part Number                     : 900-1G150-2500-000
    Module ID                           : 0
    Inforom Version
        Image Version                   : G001.0000.02.04
        OEM Object                      : 1.1
        ECC Object                      : N/A
        Power Management Object         : N/A
    GPU Operation Mode
        Current                         : N/A
        Pending                         : N/A
    GSP Firmware Version                : N/A
    GPU Virtualization Mode
        Virtualization Mode             : None
        Host VGPU Mode                  : N/A
    IBMNPU
        Relaxed Ordering Mode           : N/A
    PCI


shpc033@login0:~/hw5/matmul$ srun --partition=class1 --gres=gpu:4 clinfo
Number of platforms                               1
  Platform Name                                   NVIDIA CUDA
  Platform Vendor                                 NVIDIA Corporation
  Platform Version                                OpenCL 3.0 CUDA 11.8.88
  Platform Profile                                FULL_PROFILE
  Platform  Extensions                                                    cl_khr_global_int32_base_atomics
cl_khr_global_int32_extended_atomics  cl_khr_local_int32_base_atomics  cl_khr_local_int32_extended_atomics
cl_khr_fp64  cl_khr_3d_image_writes  cl_khr_byte_addressable_store  cl_khr_icd  cl_khr_gl_sharing
```

cl_nv_compiler_options cl_nv_device_attribute_query cl_nv_pragma_unroll cl_nv_copy_opts cl_nv_create_buffer cl_khr_int64_base_atomics cl_khr_int64_extended_atomics cl_khr_device_uuid cl_khr_pci_bus_info cl_khr_external_semaphore cl_khr_external_memory cl_khr_external_semaphore_opaque_fd cl_khr_external_memory_opaque_fd

```
  Platform Host timer resolution                  0ns
  Platform Extensions function suffix             NV

  Platform Name                                   NVIDIA CUDA
Number of devices                                 4
  Device Name                                     NVIDIA TITAN RTX
  Device Vendor                                   NVIDIA Corporation
  Device Vendor ID                                0x10de
  Device Version                                  OpenCL 3.0 CUDA
  Driver Version                                  520.61.05
  Device OpenCL C Version                         OpenCL C 1.2
  Device Type                                     GPU
  Device Topology (NV)                            PCI-E, 18:00.0
  Device Profile                                  FULL_PROFILE
  Device Available                                Yes
  Compiler Available                              Yes
  Linker Available                                Yes
  Max compute units                               72
  Max clock frequency                             1770MHz
  Compute Capability (NV)                         7.5
  Device Partition                                (core)
    Max number of sub-devices                     1
    Supported partition types                     None
    Supported affinity domains                    (n/a)
  Max work item dimensions                        3
  Max work item sizes                             1024x1024x64
  Max work group size                             1024
  Preferred work group size multiple              32
  Warp size (NV)                                  32
  Max sub-groups per work group                   0
  Preferred / native vector sizes
```

첫번째 커맨드의 경우 간단한 클락과 모델 명등만 적히는데 반해

-q옵션은 쿼리 모드로써 더 자세한 설명이 나온다

Clinfo는 opencl을 지원하는 기기들의 설명으로 이 경우에는 엔비디아 지피유가 나오게 된다.

(b)
4개의 titan RTX 24GB gpu가 장착되어 있습니다.

(c) 24576MiB의 용량을 가집니다.

(d) nvidia-smi 에서 280w를 확인할 수 있고
-q에서 Max clock frequency 1770MHz 임을 확인할 수 있습니다.

Max work item dimension = 3
Max work item sizes. = 1024x1024x64 =2^16
Max work group size = 1024
입니다.

• *자신의 병렬화 방식에 대한 설명.*
대부분의 병렬화 관련 내용은 kernel.cl에 적혀있습니다
```
#define size 32
#define NUM_ELEMS_PER_THREAD 1
```
로 각 타일을 32 x 32로 나누는것이 핵심이고
```
__local float As[size][size];
 __local float Bs[size][size];
```
를 통하여 워크로드를 적당히 쪼갭니다.

```
if (row < M && col < N) {
                float sum = 0;
                for (int t = 0; t < (K + size - 1) / size; t++) {
```
for 루프를 같은 블럭끼리 먼저 수행하여 로컬리티를 높이는 방향으로 만들었습니다.

• *뼈대 코드 matmul.c의 각 부분에 대한 설명. matmul initialize, matmul, matmul finalize 함수 각각에서 사용하는 OpenCL API 및 각 API에 대한 간략한 설명. (API 당 한문장이면 충분).*

matmul_initialize
clGetPlatformIDs: 사용 가능한 OpenCL 플랫폼의 수를 검색하고, 플랫폼이 발견되면 변수 platform에 저장합니다.
clGetDeviceIDs: 지정된 유형의 사용 가능한 OpenCL 디바이스를 가져옵니다.(여기선 GPU
clCreateContext: 지정된 디바이스 또는 여러 디바이스에 대한 OpenCL 컨텍스트생성
clCreateCommandQueue: 특정 디바이스에 커맨드 큐를 만들어 놓습니다.
create_and_build_program_with_source: clCreateProgramWithSource와 clBuildProgram을 래핑하는 사용자 정의 함수로, 파일에서 OpenCL 커널 코드를 로드하고 컴파일 하게 됩니다.
clCreateKernel: 컴파일된 프로그램에서 "sgemm"이라는 함수에 대한 커널 객체를 생성

matmul:

clEnqueueWriteBuffer: 호스트 디바이스 메모리 버퍼에 명령어를 전달합니다.
clSetKernelArg: 커널의 인자 값들을 전달합니다.
clEnqueueNDRangeKernel: 디바이스에서 커널을 실행하도록 명령을 큐에 넣습니다.
clFinish: 커맨드 큐에 이전에 큐에 넣은 모든 OpenCL 명령이 연관된 디바이스로 발행되어 완료될 때까지 기다려 동기화를 기다립니다.
clEnqueueReadBuffer: 디바이스 메모리의 버퍼에서 호스트 메모리의 버퍼로 데이터를 읽도록 명령을 큐에 넣습니다.

matmul_finalize:

clReleaseMemObject: 메모리 참조 카운터를 감소시킵니다.
clReleaseKernel: 커널 참조 수를 감소시킵니다. 참조 수가 0이 되면 객체와 그 리소스는 해제
clReleaseProgram: 프로그램 릴리스(참조) 카운터를 감소합니다
clReleaseCommandQueue: 커맨드 큐 카운터를 내려 놓습니다.
clReleaseContext: 컨텍스트 참조 수를 내려 놓습니다
모든 릴리스는 0이되면 리소스가 풀리게 됩니다.

• *자신이 적용한 코드 최적화 방식을 분류하고, 각각에 대한 성능 실험 결과. (Matrix multiplication은 향후*
프로젝트에 핵심적으로 사용될 예정이므로 해당 실험을 적극적으로 해보길 권장함.)

커널을 가장 단순하게 로컬리티를 제외하고 짜게 되면

```
__kernel void sgemm(__global float *A, __global float *B, __global float *C, int M, int N, int K) {
  // Get global position in X and Y direction
  int row = get_global_id(0);
  int col = get_global_id(1);

  // Bound check for row and col
  if(row < M && col < N) {
    float sum = 0.0;
    for(int k = 0; k < K; ++k) {
      sum += A[row * K + k] * B[k * N + col];
    }
    C[row * N + col] = sum;
  }
}
```

정도로만 짜면 됩니다.
이대로 그냥 돌린다면 단순하게 300 정도의 성능이 나오게 됩니다.
단순하게 생각해보아도 이는 훨씬 부족한 성능이므로

타일링을 적용하면

8. 1000

16. 1210

32 1330

정도의 성능이 나오게 되는데 단순히 생각해보아도 워크로드가 커질수록 당연히 효율적인 잡의 스케줄링이 가능합니다.

여기서 더 늘리고 싶었으나 하드웨어 오류에서 64 64는 불가능하고 32 32 가 최대라는 아웃풋을 보게 되었습니다.

여기서 현재 제 함수는 한번에 하나의 값만을 계산하는 중인데 이를 한번에 여러개의 답을 내도록 수정하려 하였으나 그러면 오히려 500정도로 성능이 감소하여 일단은 보류 하였습니다.