경희대학교 전자공학과 종합설계

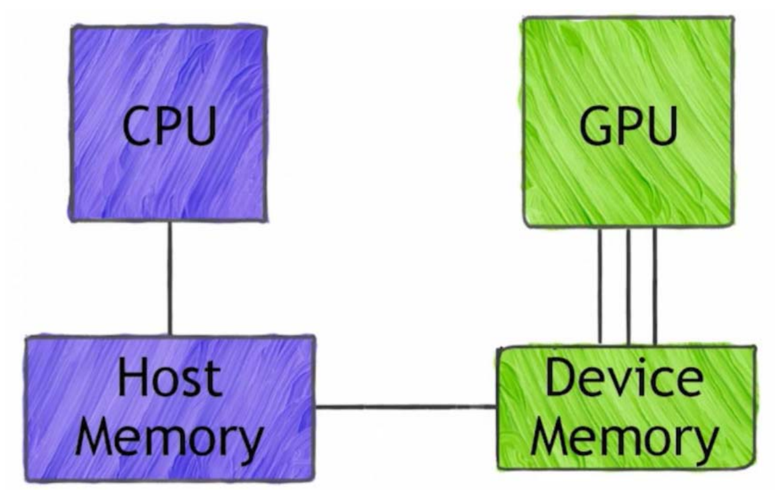
GPU를 이용한 가속화 제어 모델 설계

지도교수 김종한 교수님

전자공학과 강준희, 류찬종, 신유경

설계 배경

데이터를 연산하는 데에는 많은 방법들이 있다. 여기서 우리는 크게 CPU를 이용한 연산과 GPU를 이용한 연산에 주목했다. CPU 연산을 이용한 알고리즘은 직렬적으로 연산을 처리하고, GPU 연산은 데이터를 병렬으로 처리한다. 따라서 우리는 행렬 연산에 이를 적용하여 가속화된 제어 모델을 설계하고자 하였다.



설계 요약

먼저 행렬의 크기를 변화시키면서 몇 가지 연산을 실행해보았다. 그리고 각 연산에 걸리는 시간을 비교해 보았다.

또한, 우리가 구상한 아이디어를 적용하기 위해 PQP(Parallel Quadratic Programming) 알고리즘을 가져왔다. 천 개 이상의 행과 열을 가지는 행렬 연산에서 우리가 구상한 알고리즘으로 가속화가 가능한지 살펴보았다.

설계 내용

1. 더하기(addition) 연산

수행 연산 : (1700, 1500) + (1700, 1500) = (1700, 1500)

```
numpy    :  0.008975982666015625
addGPU   :  0.015962600708007812
pycuda   :  0.019973278045654297
```

2. 곱하기(dot product) 연산

수행 연산 : (1700, 1500) • (1500, 1700) = (1700, 1700)

1) pycuda with module

```
CPU takes :  0.036896467208862305
GPU takes :  0.026901006698608
```

2) skcuda(scikit-cuda)

```
cpu time  :   0.036936521530151367
gpu time  :   0.021940231323242188
```

3. 스택(stack) 연산

수행 연산 : (16,1)에 (16,1)를 200번 쌓음

```
hstack              :  0.0029668807983398438
concatenate         :  0.0019948482513427734
hstack(sparse)      :  0.1266725063323974
```

4. 전치(transpose) 연산

수행 연산 : (1700, 1500) -> (1500, 1700)

```
numpy.transpose   :  0.009950637817382812
numpy.T           :  0.004013299942016602
pycuda            :  1.1608686447143555
```

설계 결과

```
mat_formation   :  0.6352910995483398
loop_time       :  1.6785149574279785
mat_mul         :  0.18248724937438965
total           :  2.496293306350708
PQP_formulation :  0.22639131546020508
PQP_iteration   :  3.409879684448242
```

<기존의 PQP 알고리즘 연산 시간>

```
mat_formation   :  0.6692101955413818
loop_time       :  1.6426322460174561
mat_mul         :  0.08674478530883789
total           :  2.398587226867676
PQP_formulation :  0.20148658752441406
PQP_iteration   :  3.424813985824585
```

<GPU 연산을 활용한 연산 시간>

결과를 분석하면 적절하게 CPU 연산과 GPU 연산을 섞어 다시 작성한 알고리즘의 경우 matrix multiplication time이 절반으로 감소한 것을 볼 수 있었다. 그러나 1000번 iteration하는 부분에서의 시간이 오래 걸리기 때문에 이를 해결하기 위한 노력이 필요하다.

결론

제어 모델을 설계하는데에 있어 중요한 연산 시간을 줄이고자 하였다. 병렬으로 연산을 처리하는 GPU를 사용하면 데이터 양이 많을 때, 빠른 연산을 수행할 수 있다. 하지만 모든 경우에 GPU가 빠른 것은 아니다. PQP알고리즘을 이용한 설계 결과에서 GPU 연산과 CPU연산을 적절히 섞어 사용한다면 보다 빠른 연산이 가능함을 알 수 있었다. 이는 처음의 가정에 부합하였다. 다만 속도가 행렬의 크기, 즉 데이터 양에 많이 좌우되기 때문에 iteration 동작에서의 한계를 개선 할 방안이 필요하다.