과제 #4

M1522.006700 확장형 고성능 컴퓨팅 (001)
M3239.005400 데이터사이언스를 위한 컴퓨팅 2 (001)

Due: 2023년 11월 15일(수) 23:59:59

1 (30점) Computing π with MPI

여러 노드와 여러 쓰레드를 사용해 π를 구하는 프로그램 `monte_carlo.c` 를 작성하라. 뼈대 코드와 Makefile 이 실습 서버 로그인 노드의 `/shpc23/skeleton/hw4/monte_carlo` 디렉토리에 제공된다. MPI를 이용해 노드 간 병렬화를 수행하고, OpenMP 혹은 Pthread를 이용해 노드 내 병렬화를 수행하라. `monte_carlo.c`을 제외한 파일은 수정 불가능하다 (채점 시 `monte_carlo.c` 를 제외한 파일들은 뼈대코드의 것을 사용함).

Slurm 작업 스케줄러에 작업을 제출하는 `./run.sh` 스크립트가 제공된다. 실행 예시는 다음과 같다. `NODES` 환경변수를 이용해 MPI 프로그램을 실행시킬 노드 개수를 지정하는 것에 유의하라.

```
$  NODES=2 ./run.sh -t 32 1000000
(a08) Hello world, rank 0 out of 2
(a09) Hello world, rank 1 out of 2
Options:
  Number of points: 1000000
  Number of threads per process: 32

[rank 0] Estimated PI value: 3.1397880000
[rank 0] Relative error: -5.7444e-04
[rank 0] Validation: PASS

$ NODES=4 ./run.sh -t 32 10000000
(a08) Hello world, rank 0 out of 4
(a09) Hello world, rank 1 out of 4
(a10) Hello world, rank 2 out of 4
(a11) Hello world, rank 3 out of 4
Options:
  Number of points: 10000000
  Number of threads per process: 32

[rank 0] Estimated PI value: 3.1412076000
```

```
[rank 0] Relative error: -1.2257e-04
[rank 0] Validation: PASS
```

순차버전의 구현은 뼈대 코드에 이미 주어져 있다. 본 문제는 병렬화된 구현의 정확성만 평가한다.

보고서에 다음 질문들에 대한 답을 서술하라

- 자신의 병렬화 방식에 대한 설명. MPI를 이용해 여러 노드(프로세스)에 작업을 어떻게 배분하는지 위주로 설명.

보고서 (10점) 명시된 질문들에 대한 답으로 평가.

정확성 (20점) 제공된 `./run_validation.sh` 스크립트를 이용해 채점할 예정. 병렬화하지 않고 순차 구현을 그대로 사용할 경우에는 0점 처리.

2 (70점) Matrix Multiplication with MPI

두 FP32 행렬의 곱을 여러 노드의 여러 쓰레드들이 협업해 계산하는 프로그램 `matmul.c` 를 작성하라. 뼈대 코드와 Makefile 이 실습 서버 로그인 노드의 `/shpc23/skeleton/hw4/matmul` 디렉토리에 제공된다. 뼈대 코드를 이해한 뒤, `matmul.c` 파일을 작성하면 된다. `matmul.c` 를 제외한 파일은 수정 불가능하다 (채점 시 `matmul.c` 를 제외한 파일들은 뼈대코드의 것을 사용함).

Slurm 작업 스케줄러에 작업을 제출하는 `./run.sh` 스크립트가 제공된다. 실행 예시는 다음과 같다. `NODES` 환경변수를 이용해 MPI 프로그램을 실행시킬 노드 개수를 지정하는 것에 유의하라.

본 과제에서는 구현의 편의성을 위해 M, N, K는 각각 **1024 이상의 2의 지수승**으로만 주어진다고 가정한다. 성능 평가는 $M = 16384, N = K = 4096$인 경우만 고려한다.

```
$ NODES=4 ./run.sh -n 10 -v -t 32 16384 4096 4096
salloc: Pending job allocation 530918
salloc: job 530918 queued and waiting for resources
salloc: job 530918 has been allocated resources
salloc: Granted job allocation 530918
(a08) Hello world, rank 0 out of 4
(a11) Hello world, rank 3 out of 4
(a10) Hello world, rank 2 out of 4
(a09) Hello world, rank 1 out of 4
Options:
  Problem size: M = 16384, N = 4096, K = 4096
  Number of threads per process: 32
  Number of iterations: 10
  Print matrix: off
  Validation: on

[rank 0] Initializing matrices...Done!
[rank 0] Calculating...(iter=1) 0.202684 sec
```

```
[rank 0] Calculating...(iter=2) 0.189279 sec
[rank 0] Calculating...(iter=3) 0.198201 sec
[rank 0] Calculating...(iter=4) 0.193058 sec
[rank 0] Calculating...(iter=5) 0.193211 sec
[rank 0] Calculating...(iter=6) 0.202826 sec
[rank 0] Calculating...(iter=7) 0.189476 sec
[rank 0] Calculating...(iter=8) 0.194473 sec
[rank 0] Calculating...(iter=9) 0.193830 sec
[rank 0] Calculating...(iter=10) 0.233732 sec
Validating...
Result: VALID
[rank 0] Avg. time: 0.199077 sec
[rank 0] Avg. throughput: 2761.524938 GFLOPS
```

Hint: 노드간 통신은 최소화하는 것이 좋다. 어떻게 하면 노드 간 통신을 최소화 시킬 수 있을 지 생각해보자.

Hint: 쓰레드 간의 동기화는 최소화하는 것이 좋다. 어떻게 하면 각 쓰레드가 서로 독립적인 일을 할 수 있을 지 생각해 보자.

Hint: x86 intrinsics 를 사용해 FMA instruction 을 사용할 수 있다. 실습 서버의 CPU는 AVX512 instruction set 까지 지원한다. 512bit 레지스터를 나타내는 `__m512` 자료형 및 대응되는 함수를 사용하면 된다. 다음 document 들을 참고.

https://www.intel.com/content/www/us/en/docs/intrinsics-guide/index.html

https://www.cs.virginia.edu/~cr4bd/3330/F2018/simdref.html

https://www.codeproject.com/Articles/874396/Crunching-Numbers-with-AVX-and-AVX

Hint: Vector instruction 을 사용할 때는 인자로 들어가는 주소의 alignment 에 주의할 것. Alignment 조건을 지키지 않으면 segmentation fault 가 발생할 수 있다.

보고서에 다음 질문들에 대한 답을 서술하라

- 자신의 병렬화 방식에 대한 설명.
- 성능 최적화를 위한 적용한 방법 및 고려 사항들에 대한 논의.
- 자신이 사용한 통신 패턴에 대한 설명. (e.g., Scatter, Gather, Broadcast 등)
- M, N, K가 2의 지수승이 아닐 경우 strip-mining의 복잡성에 대한 논의.

본 문제는 성능 평가를 한다. 채점 기준은 다음과 같다.

보고서 (20점) 명시된 질문들에 대한 답으로 평가.

성능 (50점) 실습 서버에서 4노드, 노드당 32 스레드, $M = 16384, N = K = 4096$ 옵션을 주고 실행했을 때, 1,000 GFLOPS를 넘으면 만점. 그 이하는 비율에 따라 점수를 부여한다 (e.g., 900 GFLOPS인 경우 성능 점수의 90%를 부여). 답이 틀린 경우 0점. 제공된 `./run_performance.sh` 스크립트를 이용해 채점할 예정임. 주어진 `threads_per_node` 인자보다 많은 스레드를 생성해 계산하는 경우 0점. `matmul` 이 호출되는

main 스레드는 개수에 포함하지 않음.

3 제출 방법

- 과제 제출은 실습 서버에서 이루어진다.
- 보고서는 pdf 형식으로 만들어 `report.pdf` 이름으로 제출한다. 제출할 `report.pdf` 파일이 위치한 디렉토리에서 `shpc-submit submit hw4 report.pdf` 명령을 실행한다.
- 제출할 `monte_carlo.c` 파일이 위치한 디렉토리에서 `shpc-submit submit hw4 monte_carlo.c` 명령을 실행한다.
- 제출할 `matmul.c` 파일이 위치한 디렉토리에서 `shpc-submit submit hw4 matmul.c` 명령을 실행한다.
- 파일들이 잘 제출되었는지 확인을 위해 `shpc-submit status` 명령을 실행한다.
- 과제 마감 기한이 지난 뒤 다시 제출 명령을 실행하면 마지막 제출시간이 변경되므로 주의할 것.
- 과제 마감 기한이 지난 뒤 파일이 수정된 경우 grace day 를 사용한 것으로 간주한다.

4 주의 사항

- 뼈대 코드를 각자의 홈 디렉토리로 복사해 가 작업하도록 한다.
- 실습용 서버에서 과제를 수행하도록 한다. 소스 코드를 제출하는 과제의 경우 실습용 서버에서 작동하지 않으면 점수를 받을 수 없다.
- 보고서는 간략하게 필요한 내용만 적는다.