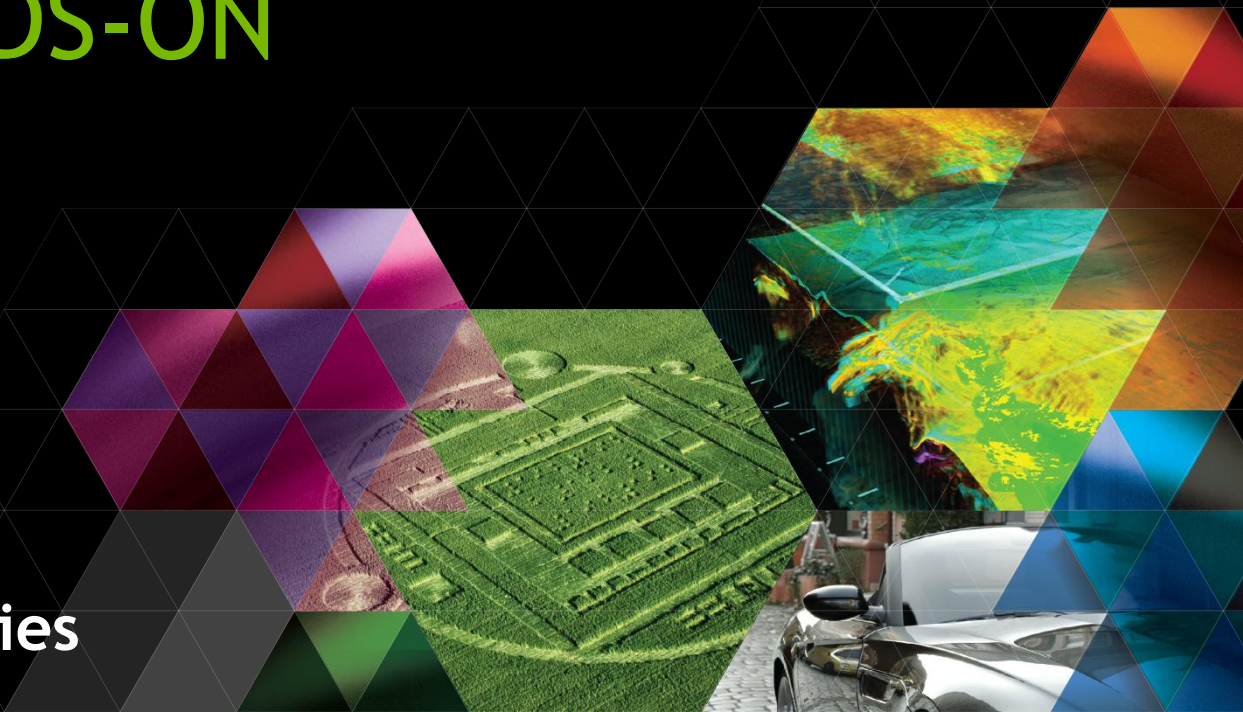
GPU TECHNOLOGY CONFERENCE

NVIDIA

OPENACC HANDS-ON



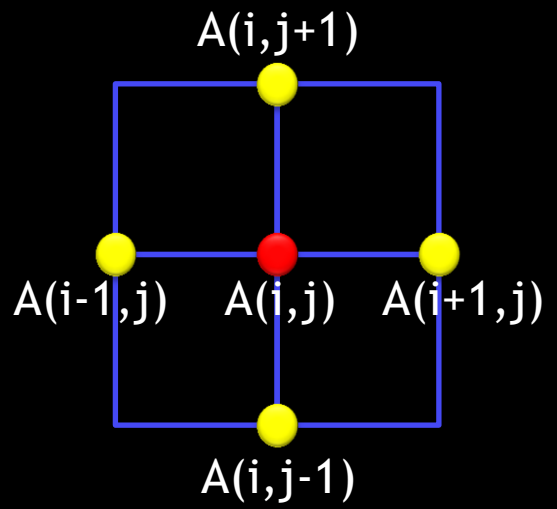
Akira Naruse
NVIDAI Developer Technologies

例題: JACOBI ITERATION

```
while ( error > tol ) {
  error = 0.0;

  for (int j = 1; j < N-1; j++) {
  for (int i = 1; i < M-1; i++) {
      Anew[j][i] = (A[j][i+1] + A[j][i-1] +
                    A[j-1][i] + A[j+1][i]) * 0.25;
      error = max(error, abs(Anew[j][i] - A[j][i]));
  }}

  for (int j = 1; j < N-1; j++) {
  for (int i = 1; i < M-1; i++) {
      A[j][i] = Anew[j][i];
  }}
}
```



始めに

- コードサンプル: C and Fortran
- Task[1-5]
  - ソースファイル (jacobi.[c|f90]) の内容を理解
  - 変更前に make & run、実行結果・性能を確認
  - 課題に従って、ソースファイルを変更
    - Makefileの変更は不要 (変更してもOK)
  - make & run、変更前の実行結果・性能と比較
- Task[1-5]-solution
  - 解答例

GPU使用状況の確認方法(1)

- nvprof を使用

```
$ nvprof ./jacobi
...
Time(%)      Time     Calls       Avg       Min       Max  Name
 44.56%  2.55738s      1800  1.4208ms  2.2400us  1.5996ms  [CUDA memcpy DtoH]
 44.43%  2.55007s      1800  1.4167ms     799ns  1.5980ms  [CUDA memcpy HtoD]
  7.00%  401.79ms       200  2.0090ms  2.0079ms  2.0127ms  jacobi_48_gpu
  3.17%  181.95ms       200  909.74us  908.75us  910.80us  jacobi_57_gpu
  0.84%  48.214ms       200  241.07us  240.31us  242.10us  jacobi_48_gpu_red
...
Time(%)      Time     Calls       Avg       Min       Max  Name
 46.90%  1.95212s      4796  407.03us  1.5120us  1.6080ms  cuEventSynchronize
 35.09%  1.46060s      1400  1.0433ms  1.7290us  1.6331ms  cuStreamSynchronize
 10.83%  450.75ms       200  2.2538ms  2.2430ms  2.4390ms  cuMemcpyDtoHAsync
  3.69%  153.70ms         1  153.70ms  153.70ms  153.70ms  cuCtxCreate
  1.13%  47.151ms      2400  19.646us  7.5880us  54.392us  cuMemcpy2DAsync
  0.74%  30.985ms      4798  6.4570us  2.4050us  48.221us  cuEventRecord
  0.56%  23.224ms      1000  23.223us  10.041us  35.594us  cuMemcpyHtoDAsync
```

GPU使用状況の確認方法(2)

- 環境変数 PGI_ACC_TIME を使用

```
$ PGI_ACC_TIME=1 ./jacobi
...
  jacobi  NVIDIA  devicenum=0
    time(us): 54,861
    44: data region reached 200 times
        44: data copyin transfers: 800
             device time(us): total=11,278 max=22 min=12 avg=14
        55: data copyout transfers: 800
             device time(us): total=9,209 max=52 min=4 avg=11
    44: compute region reached 200 times
        48: kernel launched 200 times
            grid: [32x4094]  block: [128]
             device time(us): total=4,672 max=96 min=21 avg=23
            elapsed time(us): total=7,728 max=113 min=35 avg=38
        48: reduction kernel launched 200 times
            grid: [1]  block: [256]
             device time(us): total=1,763 max=14 min=6 avg=8
```

TASK 1

- 課題: Kernelsディレクティブを使って、計算をGPUにオフロードする

- Kernelsディレクティブを2箇所に挿入
  - 配列Anewの更新ループ
  - 配列Aの更新ループ

COMPILER MESSAGE

```
$ make
...
     44, Generating present_or_copyout(Anew[1:4094][1:4094])
         Generating present_or_copyin(A[:][:])
         Generating Tesla code
     46, Loop is parallelizable
     48, Loop is parallelizable
         Accelerator kernel generated
         46, #pragma acc loop gang /* blockIdx.y */
         48, #pragma acc loop gang, vector(128) /* blockIdx.x threadIdx.x */
             Max reduction generated for error
     55, Generating present_or_copyin(Anew[1:4094][1:4094])
         Generating present_or_copyout(A[1:4094][1:4094])
         Generating Tesla code
     56, Loop is parallelizable
     57, Loop is parallelizable
         Accelerator kernel generated
         56, #pragma acc loop gang /* blockIdx.y */
```

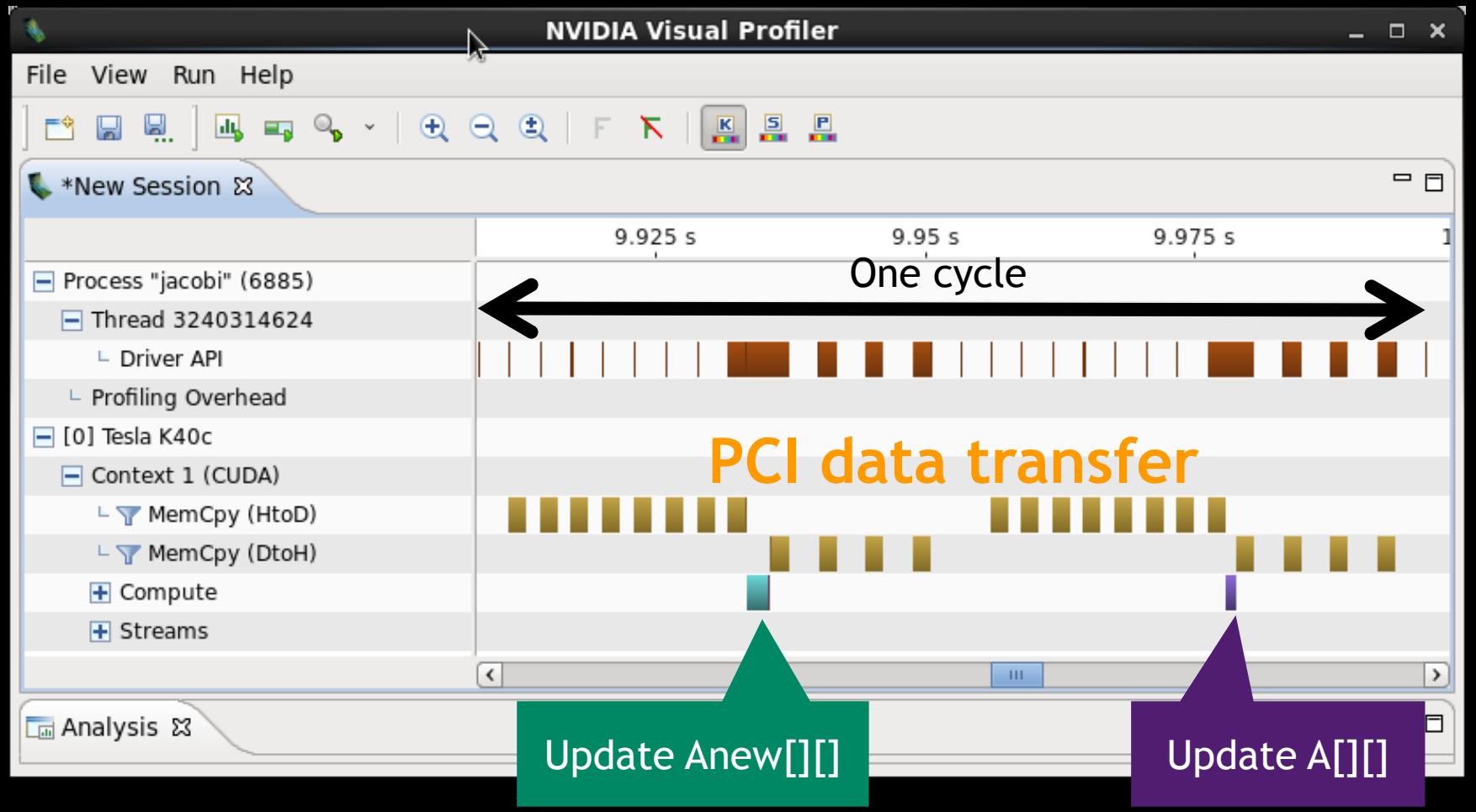
TASK 2

- 課題: コンパイラメッセージをソースコードに反映する

- Kernelsディレクティブ
  - Dataクローズ (pcopy, pcopyin, pcopyout)
- Loopコンストラクト
  - Reductionクローズ
  - Gangクローズ
  - Vectorクローズ

- ＋α: Gang,Vectorクローズで色々なパラメータを試す

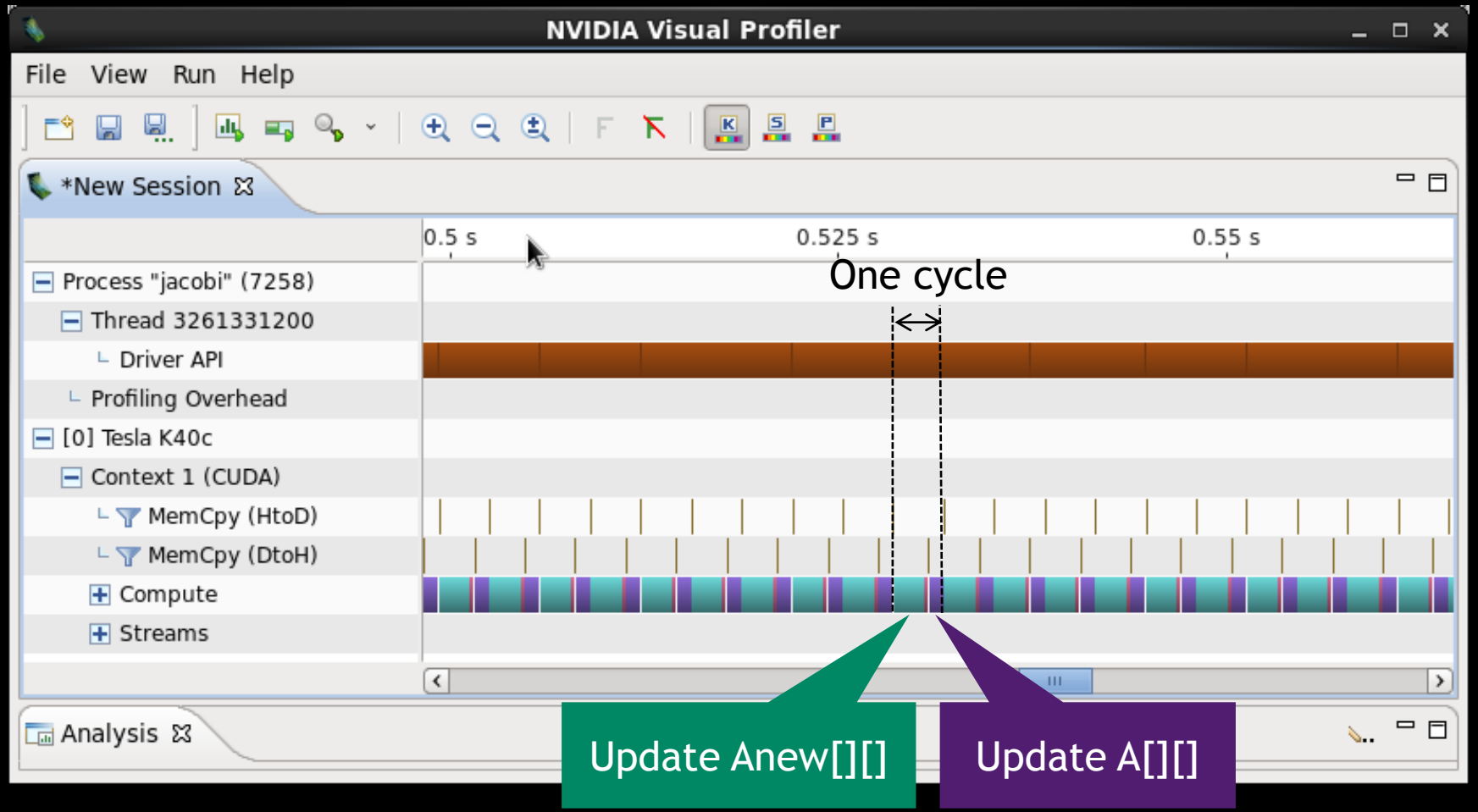
TASK 2 (NVVP)


NVIDIA Visual Profiler
File View Run Help
*New Session
9.925 s
9.95 s
9.975 s
Process "jacobi" (6885)
Thread 3240314624
Driver API
Profiling Overhead
[0] Tesla K40c
Context 1 (CUDA)
MemCpy (HtoD)
MemCpy (DtoH)
Compute
Streams
Analysis
One cycle
PCI data transfer
Update Anew[][]
Update A[][]

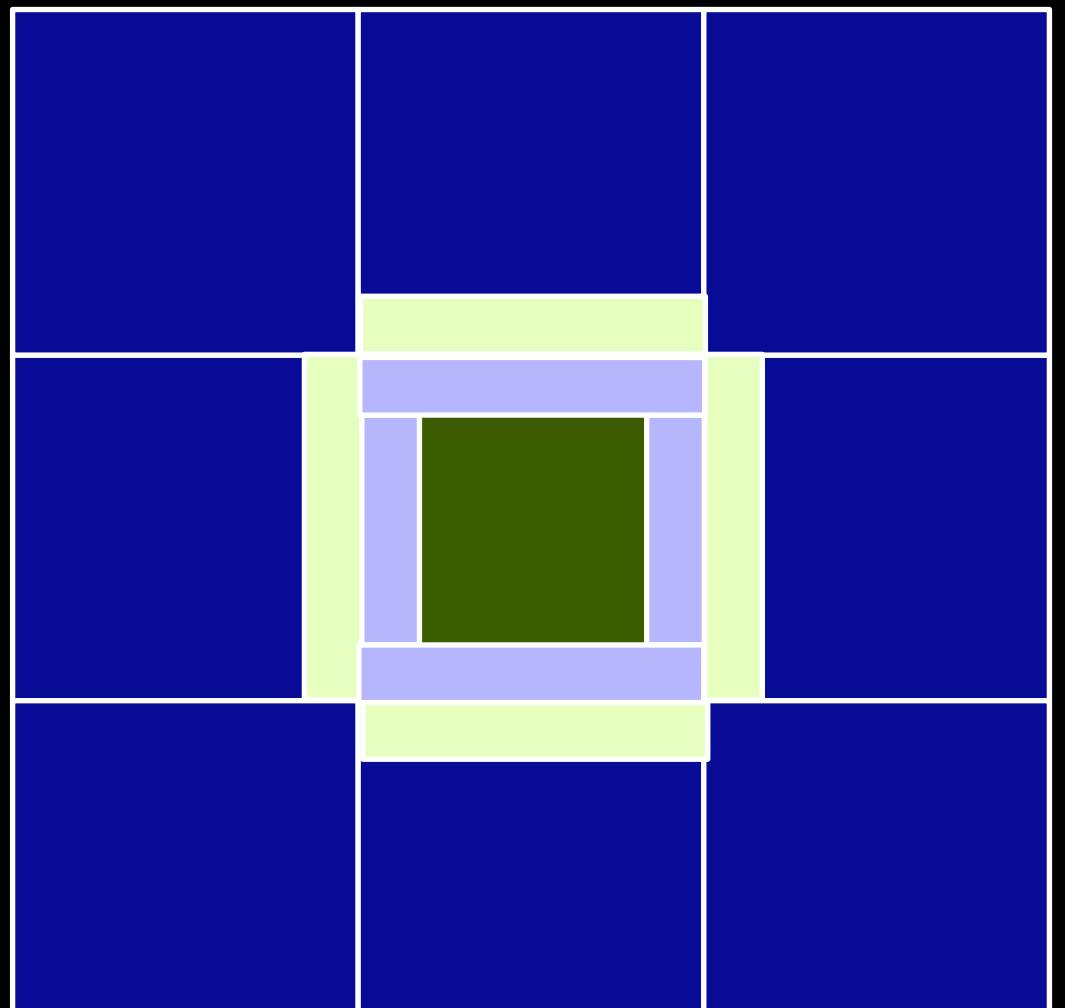
TASK 3

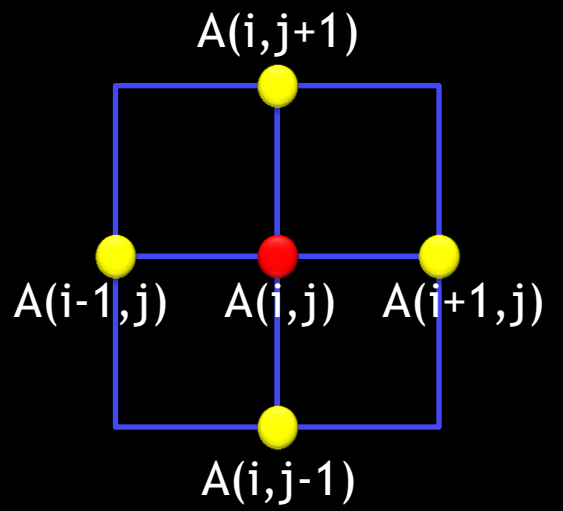
- 課題: Data ディレクティブを使って、PCIデータ転送回数・量を減らす
  - データをGPUに常駐させる
- Dataディレクティブ
  - Dataクローズ (pcopy, create)

TASK 3 (NVVP)



MPI並列 (HALO EXCHANGE)





- ブロック分割
- 各プロセスは1ブロック担当
- 境界部(halo)のデータ交換

MPI JACOBI ITERATION

```
#pragma acc data pcopy(A) create(Anew)
while ( error > tol ) {

  pack_data_at_boundary( send_buf, A, ... );

  exchange_data_by_MPI( recv_buf, send_buf, ... );


  unpack_data_to_halo( A, recv_buf, ... );

  #pragma acc kernels pcopy(Anew) pcopyin(A)
  calc_new_A( Anew, A, ... );

  #pragma acc kernels pcopy(A) pcopyin(Anew)
  update_A( A, Anew );
}
```

GPU
1.送信データの梱包

MPI
2.データの交換

GPU
3.受信データの開梱

TASK 4

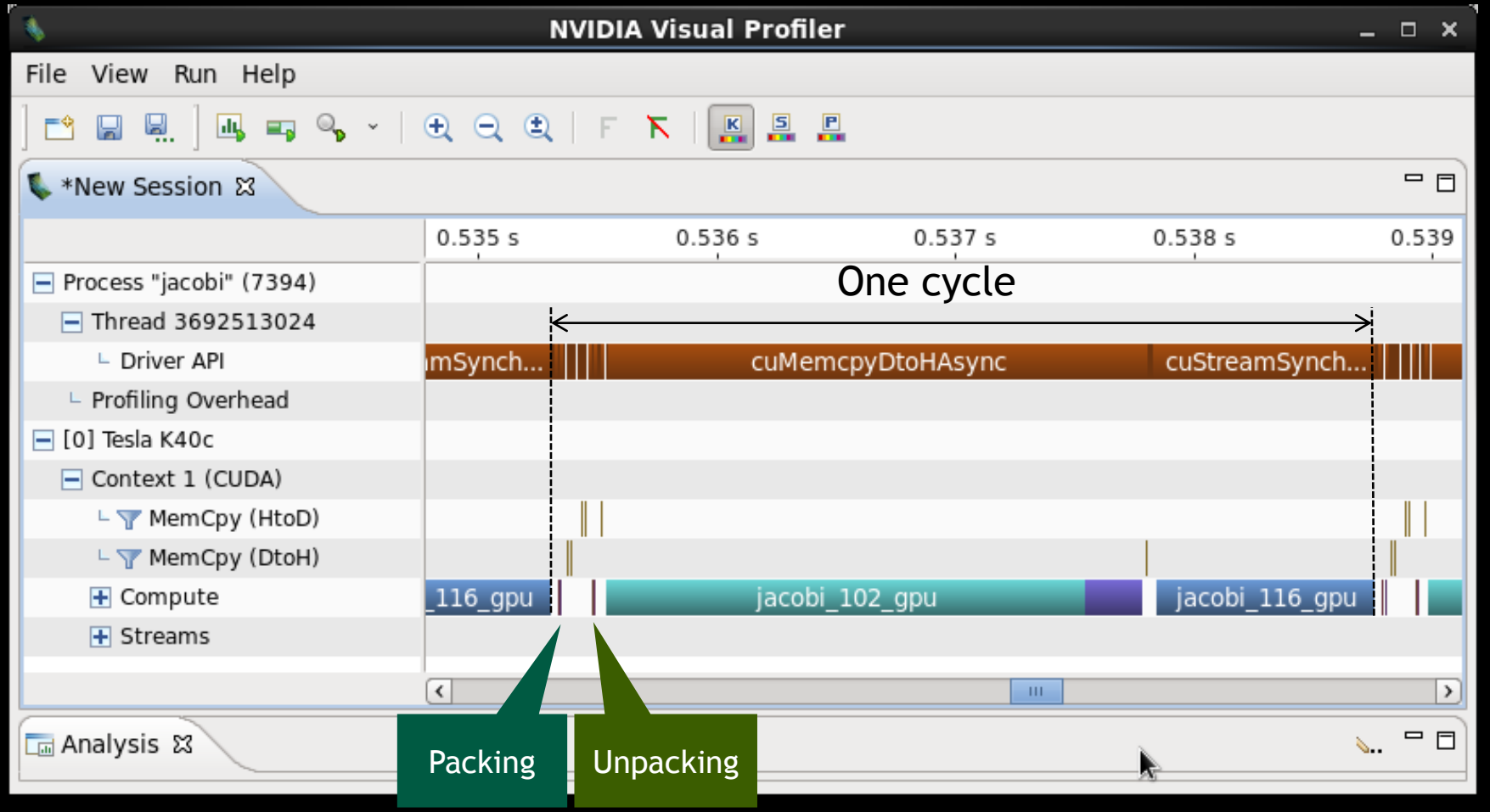
- 課題: MPIでノード間通信するデータの処理(packingとunpacking)をGPUにオフロードする

  (*) 全員がMPI実行すると大変なので、1プロセスで模擬。

- 送信データのpacking処理を、GPUにオフロード
- 送信データをPCI転送(CPU ← GPU) … copyout

  (通信模擬部分(メモリコピー)はCPU上で実行)
- 受信データをPCI転送(CPU → GPU) … copyin
- 受信データのunpacking処理を、GPUにオフロード

TASK 4 (NVVP)


NVIDIA Visual Profiler
File View Run Help
*New Session
0.535 s
0.536 s
0.537 s
0.538 s
0.539
One cycle
Process "jacobi" (7394)
Thread 3692513024
Driver API
Profiling Overhead
[0] Tesla K40c
Context 1 (CUDA)
MemCpy (HtoD)
MemCpy (DtoH)
Compute
Streams
mSynch...
cuMemcpyDtoHAsync
cuStreamSynch...
_116_gpu
jacobi_102_gpu
jacobi_116_gpu
Analysis
Packing
Unpacking

TASK 5の前に

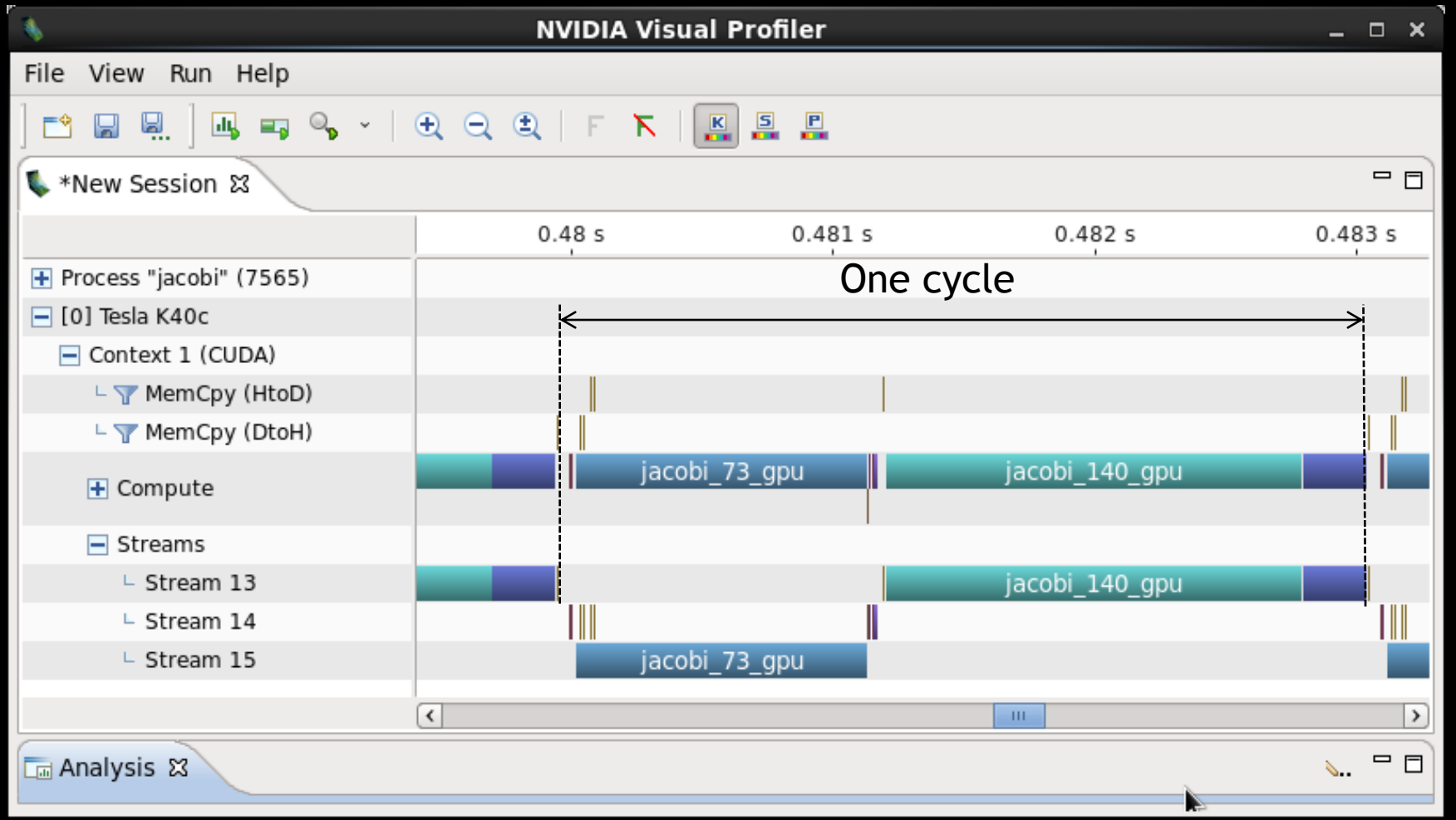
- Updateディレクティブ
  - Host(list) … listに指示したデータをGPUからホストに転送
  - Device(list) … listに指示したデータをホストからGPUに転送
- Collapseクローズ
  - Loopコンストラクトに付けることができる、指定数のループをまとめる

TASK5

- 課題: GPUカーネル実行とPCIデータ転送をオーバーラップさせる、Asyncクローズ とWaitディレクティブを使用

- Asyncクローズ
  - KernelsディレクティブやDataディレクティブに付けることができる
  - Async(N) … ID=Nのキューに処理を投入する (同キュー内では逐次実行)
- Waitディレクティブ
  - Wait(list) … listに指示したキューの処理が完了するまで待つ

TASK 5 (NVVP)


NVIDIA Visual Profiler
File View Run Help
*New Session
0.48 s
0.481 s
0.482 s
0.483 s
Process "jacobi" (7565)
[0] Tesla K40c
Context 1 (CUDA)
MemCpy (HtoD)
MemCpy (DtoH)
Compute
Streams
Stream 13
Stream 14
Stream 15
One cycle
jacobi_73_gpu
jacobi_140_gpu
jacobi_140_gpu
jacobi_73_gpu
Analysis