GPU Profiling pytorch programs

24 nov. 2018

https://docs.google.com/presentation/d/1DCe2gjAhfAHqpG47ZiUGlQrCHT_dDwjGpDgXcsjoLyQ/edit?usp=sharing

Rencontre datascience
2018-10-24
Martin Sotir

CRIM

Occupancy ?

```
> watch -n 1 nvidia-smi
```

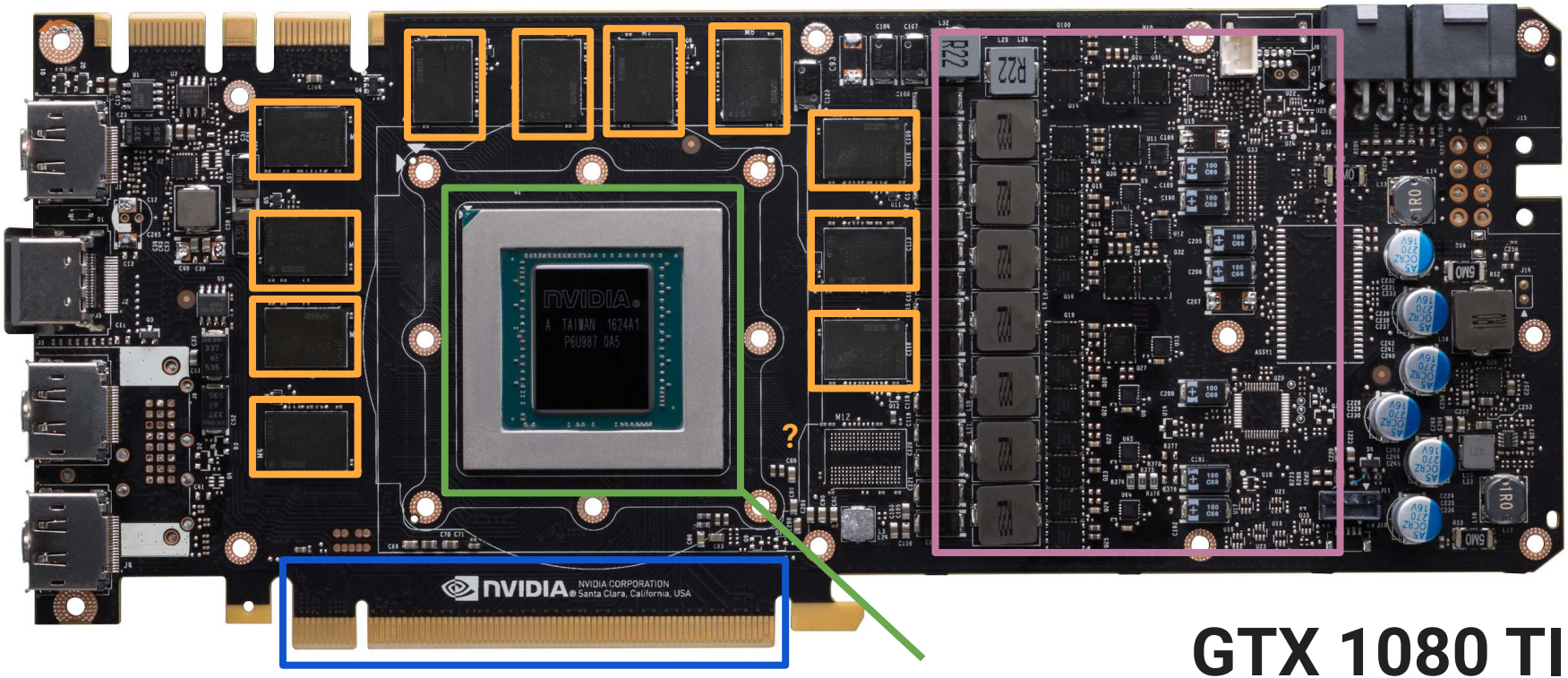
```
Every 1.0s: nvidia-smi

Wed Oct 24 07:00:40 2018
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 396.26                 Driver Version: 396.26                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce GTX 108...  Off  | 00000000:00:06.0 Off |                  N/A |
| 49%   84C    P2   197W / 250W |   2379MiB / 11178MiB |     97%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    0     22841      C   python                                     2369MiB |
+-----------------------------------------------------------------------------+
```

???

GPU Usage != occupancy ?

GDDR5X memory - **11 GB**
480 GB/s bandwidth

VRM -Voltage Regulator Module



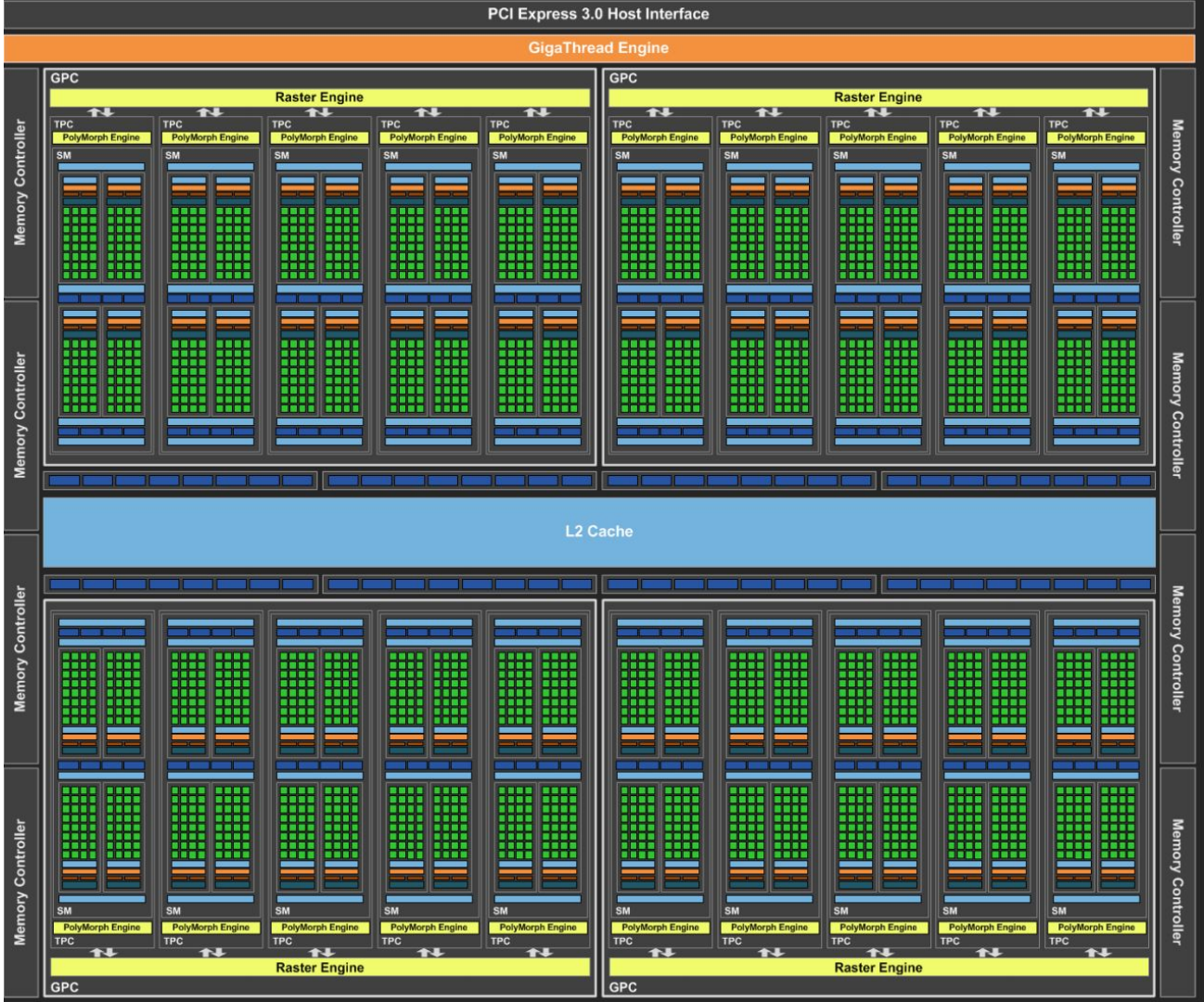
GTX 1080 TI

Bus PCI-E v3 x16 = 16 x 985 MB/s
= 15.7 GB/s*
GPU ⟷ CPU bus async I/O

Multiprocessors
1.2GHz - 2GHz

source : nvidia.com

Multiprocessors? (SM)

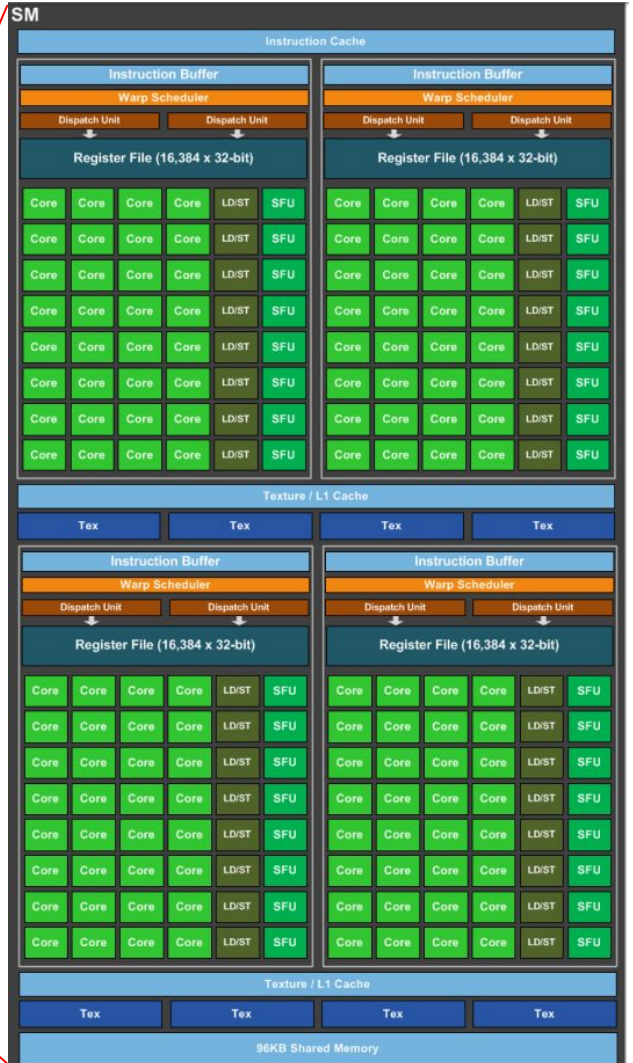


source : nvidia.com

Multiprocessors (SM) ?

28 multiprossors in 1080TI

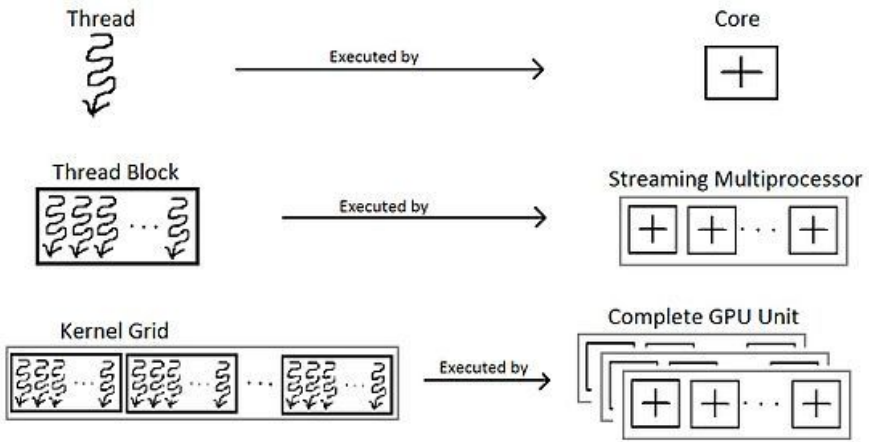
- Wrap ?
- Thread ?
- Core ?
- Block ?
- Grid ?
- Kernel ?

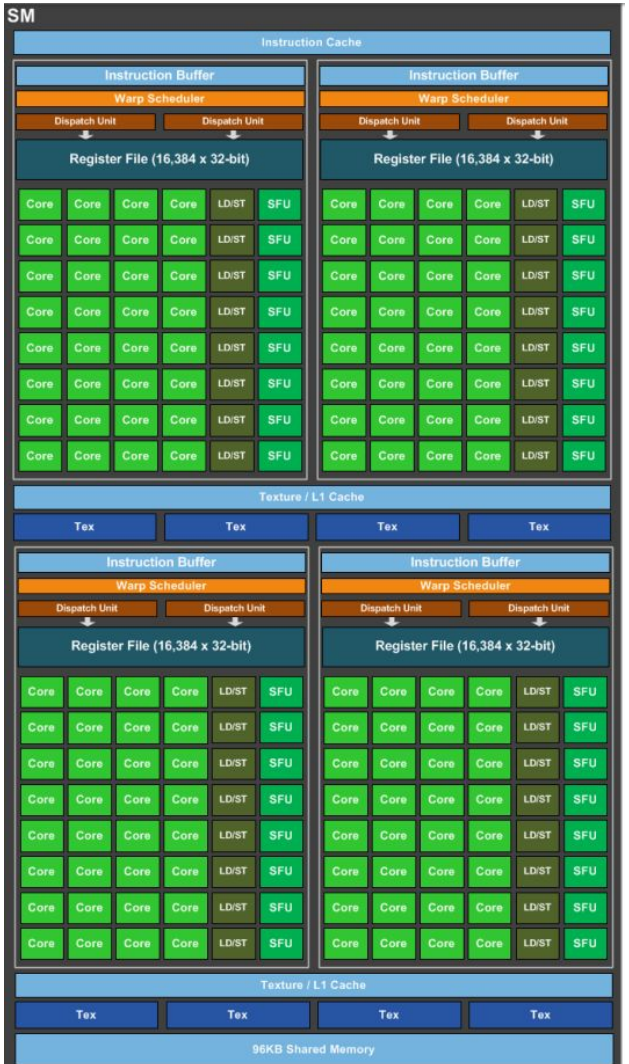


Streaming Multiprocessors (SM)

Hardware perspective : SM ? block ? Wrap ? core ?
… vs software perspective : Kernel ? Grid ? Thread ?

- SM = 4 wraps (thread blocks)
- wrap = group 32 cores (== threads)
- Kernel = mini-program / function





nvprof - collect human readable stats

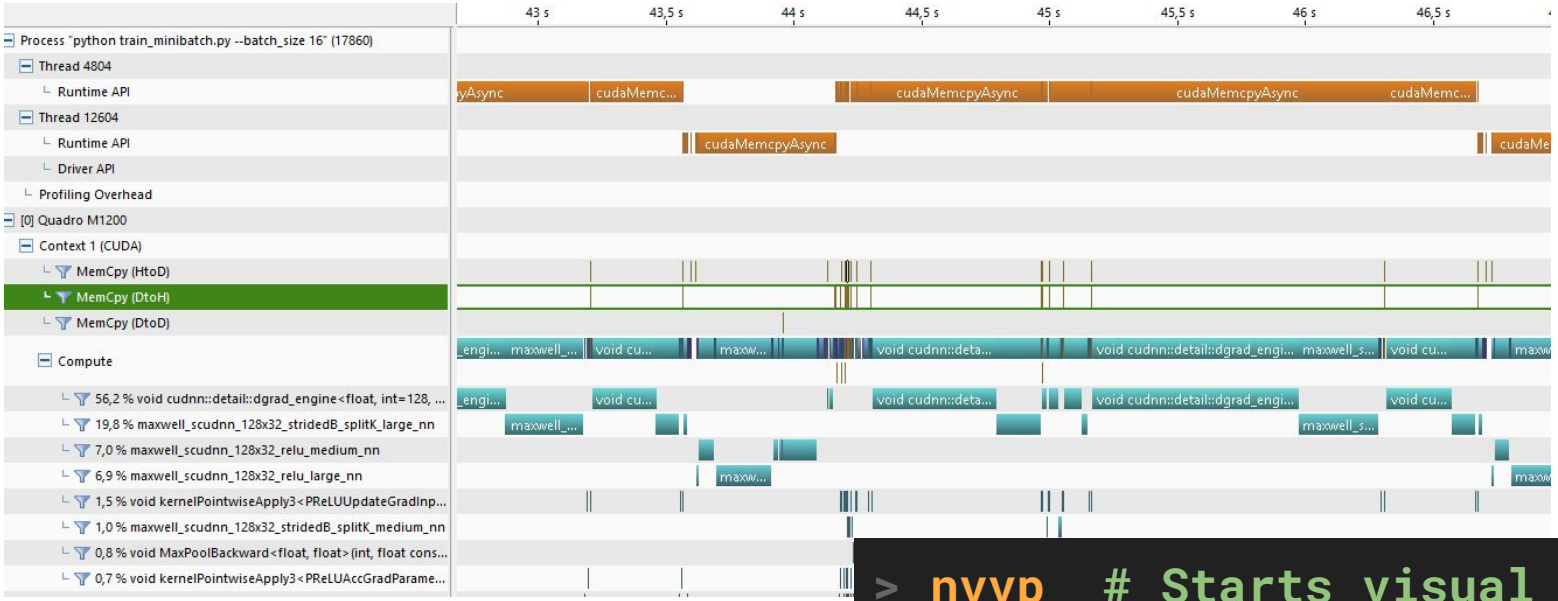
```
> nvprof --log-file profile.log python train.py
```

nvprof is usually available in /usr/local/cuda/bin

```
==20510== Profiling result:
Time(%)      Time     Calls       Avg       Min       Max  Name
 48.97%  15.0900s       618  24.417ms  696.98us  80.159ms  void cudnn::detail::dgrad_engine<float, int=128, int=6, i
 16.63%  5.12344s       618  8.2904ms  231.62us  27.441ms  maxwell_scudnn_128x32_stridedB_splitK_large_nn
  8.88%  2.73659s       835  3.2773ms  191.94us  11.317ms  maxwell_scudnn_128x32_relu_medium_nn
  8.59%  2.64821s       334  7.9288ms  338.22us  18.684ms  maxwell_scudnn_128x32_relu_large_nn
  3.75%  1.15434s      1648  700.45us  12.001us  1.7575ms  void calc_bias_diff<int=2, float, float, int=128, int=0>(
  2.65%  815.20ms       334  2.4407ms  553.81us  4.3517ms  void cudnn::detail::dgrad2d_alg1_1<float, int=0, int=6, i
  1.25%  386.56ms      2671  144.73us  2.4640us  628.37us  void add_tensor_kernel_v3<int=2, float, float, int=128, i
  1.03%  317.38ms      2838  111.83us     928ns  558.39us  void kernelPointwiseApply2<LeakyReLUUpdateOutput<float>,
  0.97%  299.27ms       619  483.48us  90.371us  1.1689ms  maxwell_scudnn_128x32_stridedB_splitK_medium_nn
  0.95%  291.26ms      1648  176.73us  1.0880us  834.97us  void kernelPointwiseApply3<LeakyReLUUpdateGradInput<float
  0.70%  214.34ms       167  1.2835ms  628.09us  1.5355ms  maxwell_scudnn_128x64_relu_small_nn
  0.56%  172.94ms       206  839.51us  69.731us  1.8050ms  maxwell_scudnn_128x32_stridedB_small_nn
  0.56%  171.60ms       103  1.6661ms  814.68us  1.9186ms  maxwell_scudnn_128x64_stridedB_splitK_large_nn
  0.51%  156.42ms       412  379.66us  115.52us  653.53us  void MaxPoolBackward<float, float>(int, float const *, lo
  0.46%  140.24ms       668  209.94us  64.867us  314.57us  void MaxPoolForward<float, float>(int, float const *, int
  0.44%  135.19ms       840  160.94us     672ns  3.9193ms  [CUDA memcpy HtoD]
  0.40%  123.27ms       206  598.42us  32.833us  1.3077ms  maxwell_scudnn_128x32_stridedB_splitK_interior_nn
```

nvprof - visual profiler

```
> nvprof -o profile.prof python train.py
```




```
> nvvp  # Starts visual profiler
```

nvprof - GPU / system stats

```
> nvprof --system-profiling -o profile.prof python train.py
```

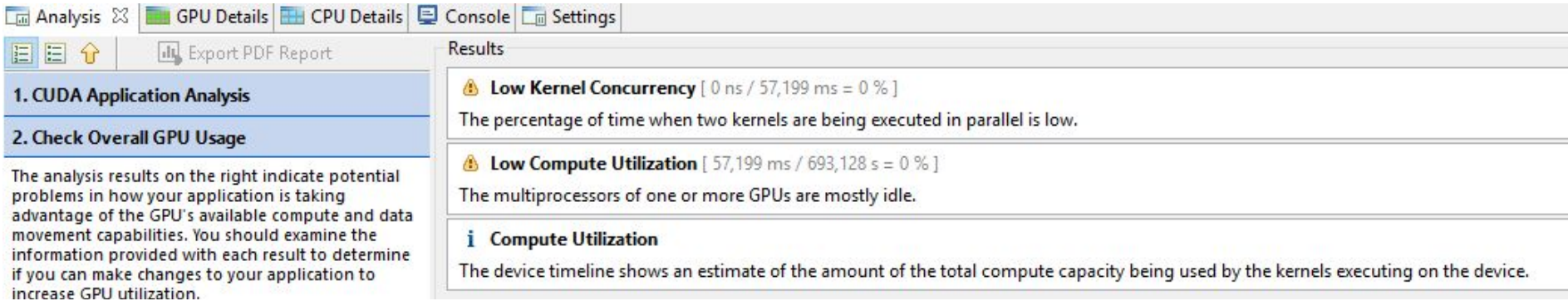
[0] GeForce GTX 1080 Ti	
Attributes	
Compute Capability	6.1
Maximums	
Threads per Block	1024
Threads per Multiprocessor	2048
Shared Memory per Block	48 KiB
Shared Memory per Multiprocessor	96 KiB
Registers per Block	65536
Registers per Multiprocessor	65536
Grid Dimensions	[2147483647, 65535, 65535]
Block Dimensions	[1024, 1024, 64]
Warps per Multiprocessor	64
Blocks per Multiprocessor	32
Half Precision FLOP/s	88,592 GigaFLOP/s
Single Precision FLOP/s	11,34 TeraFLOP/s
Double Precision FLOP/s	354,368 GigaFLOP/s
Multiprocessor	
Multiprocessors	28
Clock Rate	1,582 GHz
Concurrent Kernel	true
Max IPC	6
Threads per Warp	32

nvprof - cross platform profiling

Important for cross plateform analysis: use --export-profile instead of -o ?

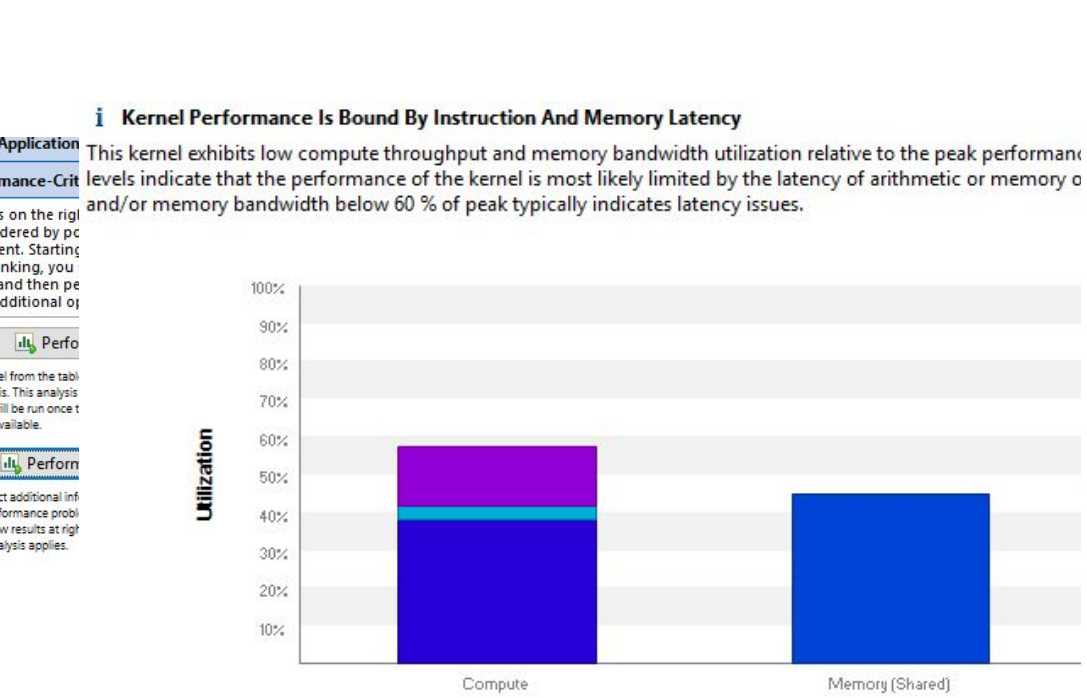
```
> nvprof --export-profile profile.prof python train.py
```

nvprof - visual profiler



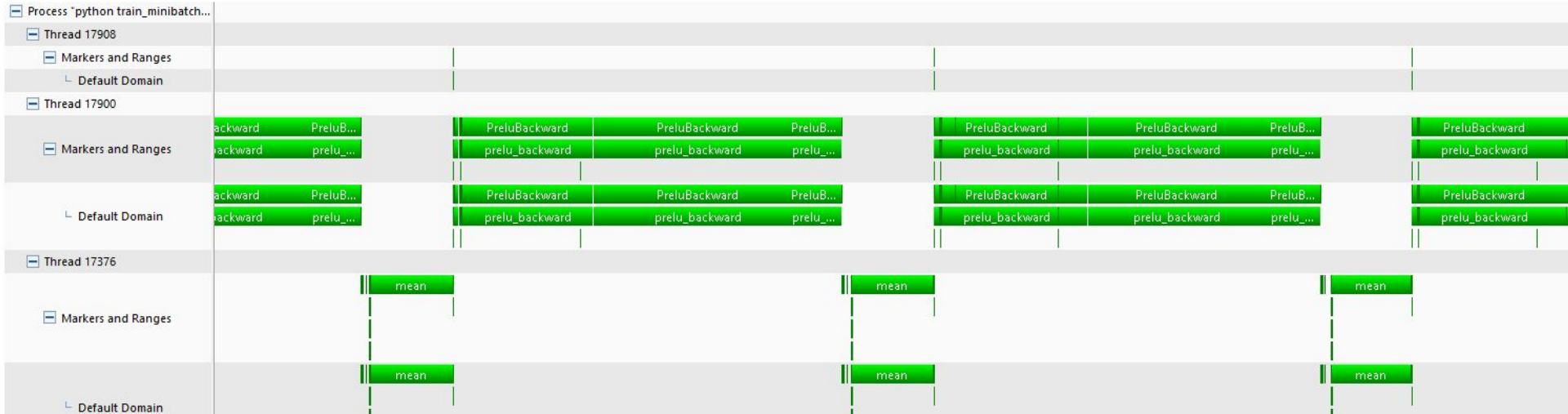
nvprof - fine-grained kernel analysis

```
> nvprof --analysis-metrics -o profile.prof python train.py
```

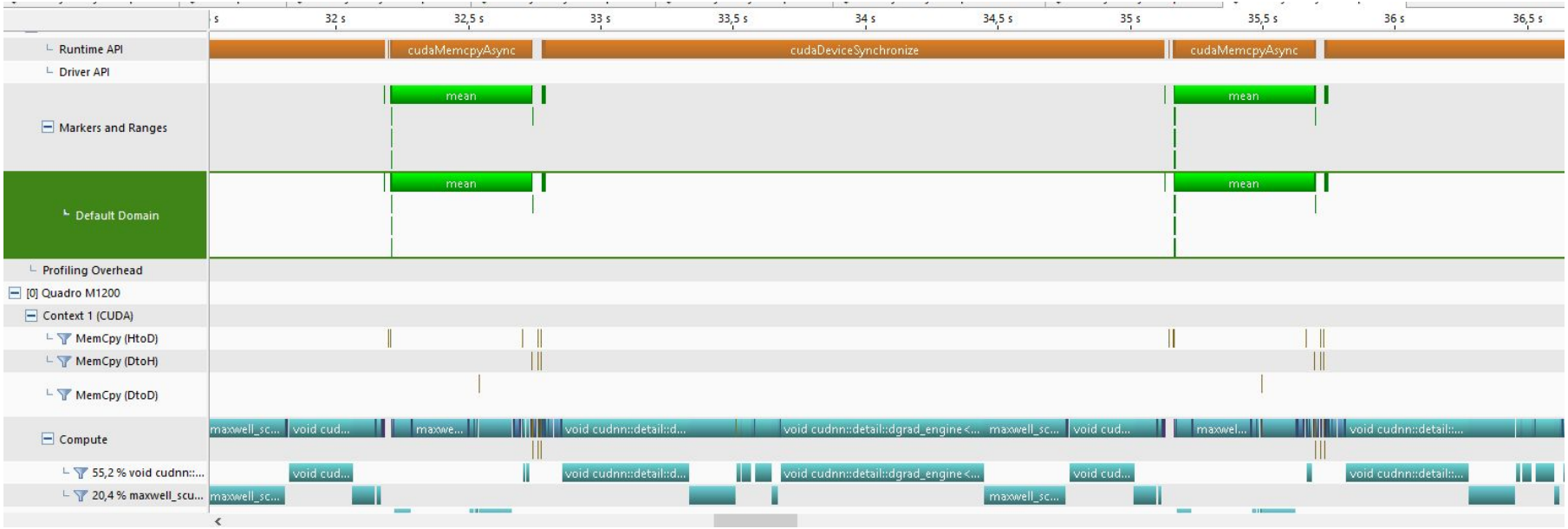


nvprof - Add pytorch annotations

```
with torch.autograd.profiler.emit_nvtx(enabled=True):
        density_map = net(im_data, gt_data, gt_class_label, class_wts, nb_pixels)
        loss = net.loss
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
```



nvprof - Add pytorch annotations



nvprof - Add pytorch annotations

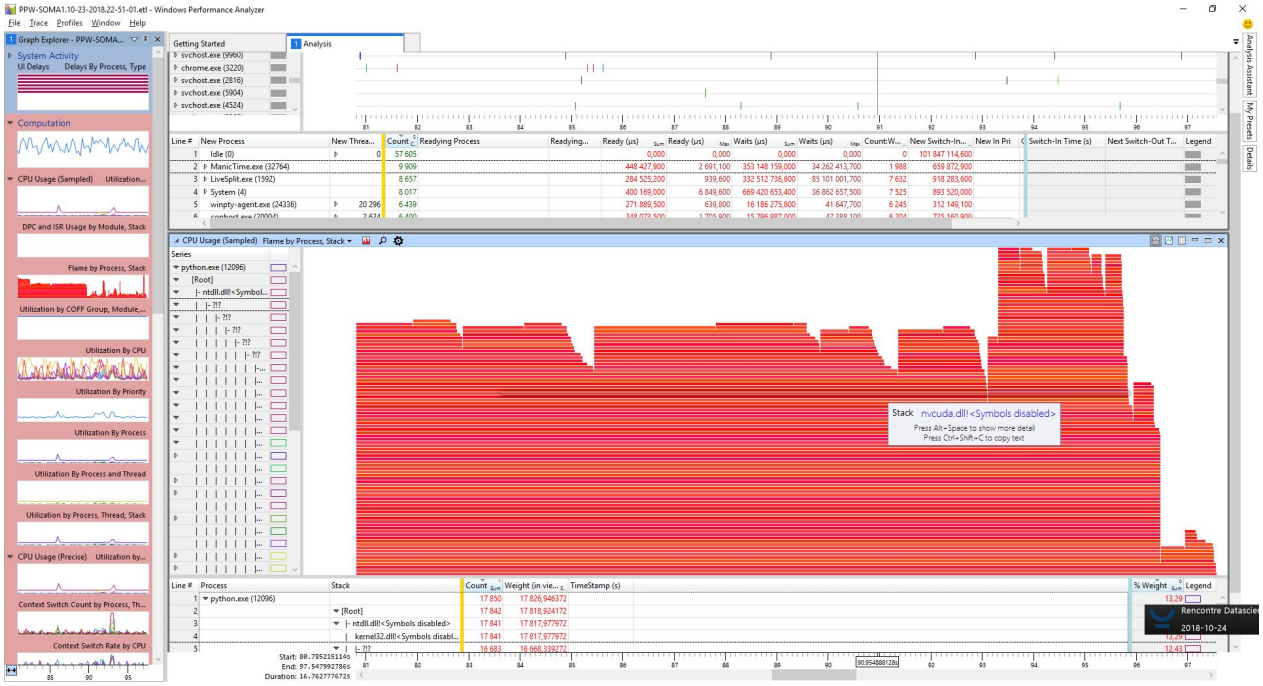
kernprof

```
> pip install line_profiler
> kernprof -l -o train.py.lprof train.py
> python -m line_profiler train.py.lprof
```

```
Line #      Hits         Time  Per Hit   % Time  Line Contents
==============================================================
    50                                           @profile
    51                                           def build_loss(self, density_map, density
    52
    53      1000     679970.0    680.0      0.8          loss_mse = self.loss_mse_fn(density_m
    54
    55                                               # manipulate loss_mse to get sum of s
    56      1000    1220858.0   1220.9      1.4          loss_mse = loss_mse.sum(dim=3).sum(di
    57
    58                                               # divide total squared error by numbe
    59      1000     430174.0    430.2      0.5          loss_mse = loss_mse / nb_pixels
    60
    61                                               # return mean of error as loss for ba
    62      1000   78014267.0  78014.3     87.2          loss_mse = torch.sum(loss_mse)
    63
    64      1000    9120701.0   9120.7     10.2          cross_entropy = self.loss_bce_fn(density_cls_score, gt_cls_label)
    65
    66      1000      17252.0     17.3      0.0          return loss_mse, cross_entropy
```

```
@profile
def forward(self, im_data, gt_data=None, g
    density_map, density_cls_score = self.
    density_cls_prob = F.softmax(density_c

    if self.training:

        self.loss_mse, self.cross_entropy
            density_map, density_cls_prob,

    return density_map
```

Genera profiling on windows - xperf



Fichier Options Affichage

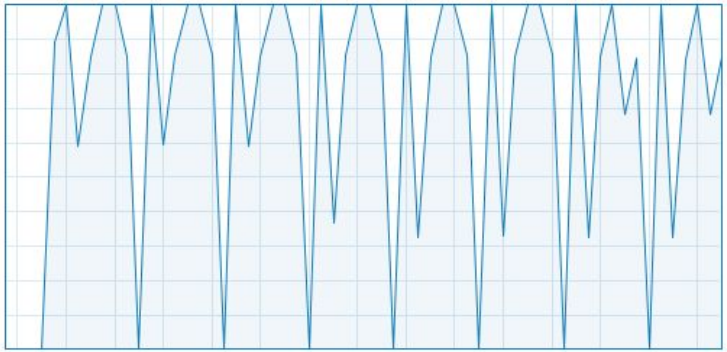
Processus | Performance | Historique des applications | Démarrage | Utilisateurs | Détails | Services

- Processeur
  25% 2,37 GHz
- Mémoire
  21,0/31,9 Go (66%)
- Disque 0 (C:)
  1%
- Ethernet
  Non connecté
- Ethernet
  E : 0 R : 0 Kbits/s
- Ethernet
  Non connecté
- Wi-Fi
  E : 0 R : 0 Kbits/s
- Ethernet
  E : 0 R : 0 Kbits/s
- Processeur graphiqu
  Intel(R) HD Graphics 630
  6%
- Processeur graphiqu
  NVIDIA Quadro M1200
  77%

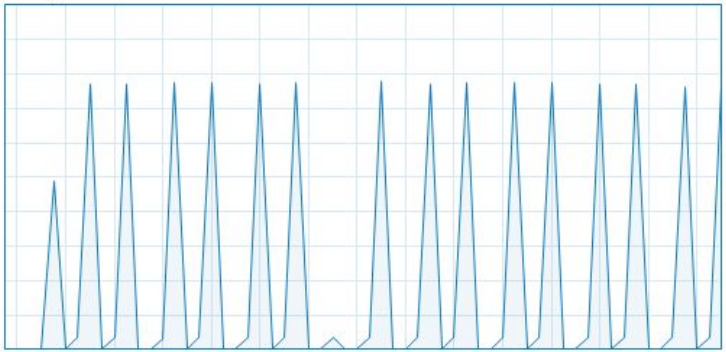
Processeur graphique

NVIDIA Quadro M1200

Compute_0 85%



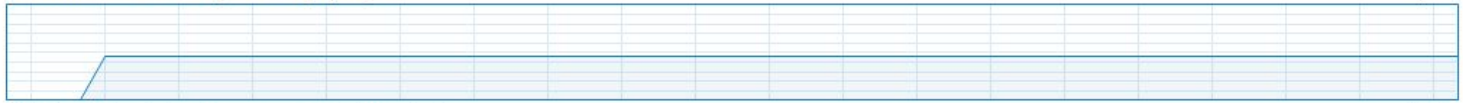
Copy 77%



Video Encode 0%

Video Decode 0%

Utilisation de la mémoire du processeur graphique dédié 4,0 Go



Utilisation de la mémoire du processeur graphique partagé 15,9 Go