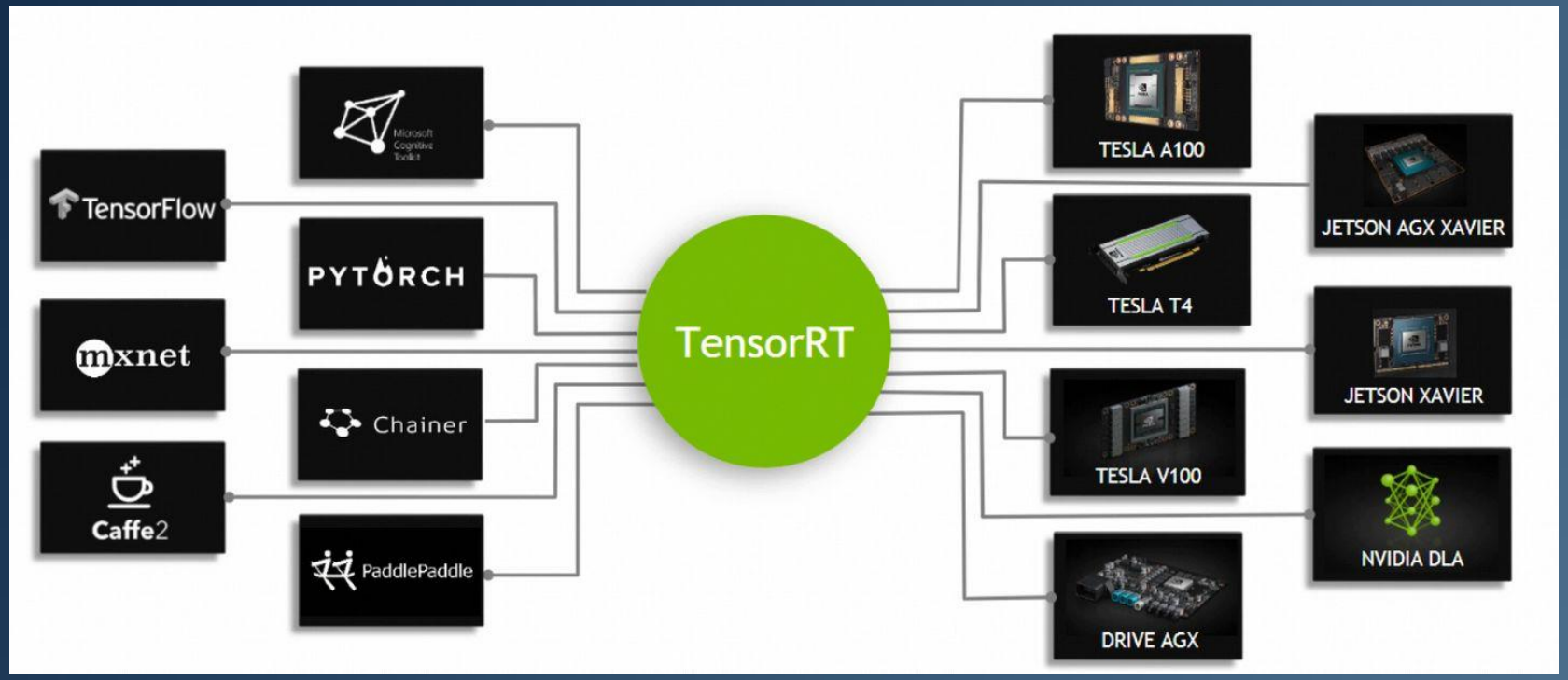
TensorRT 量化部署

带你认识 TensorRT

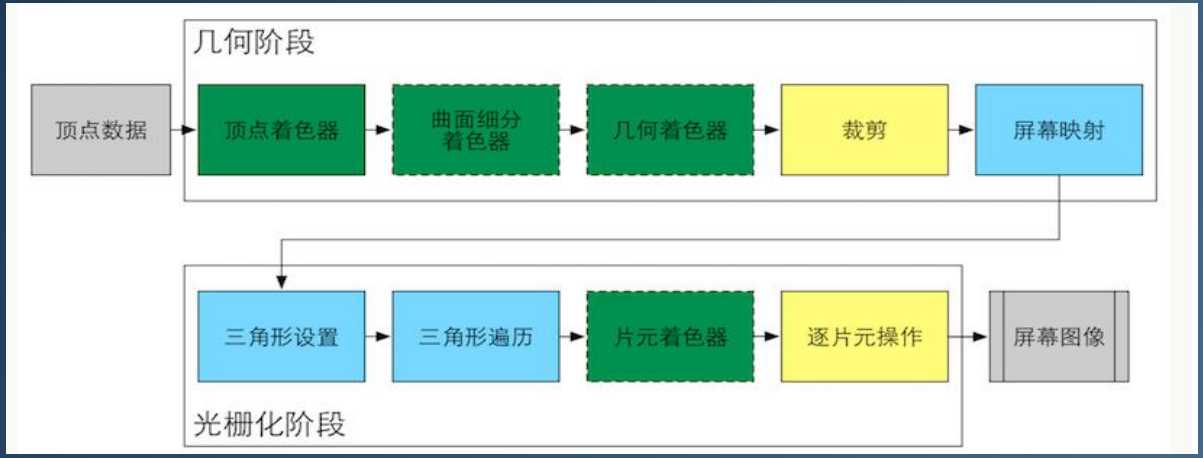


什么是 GPGPU

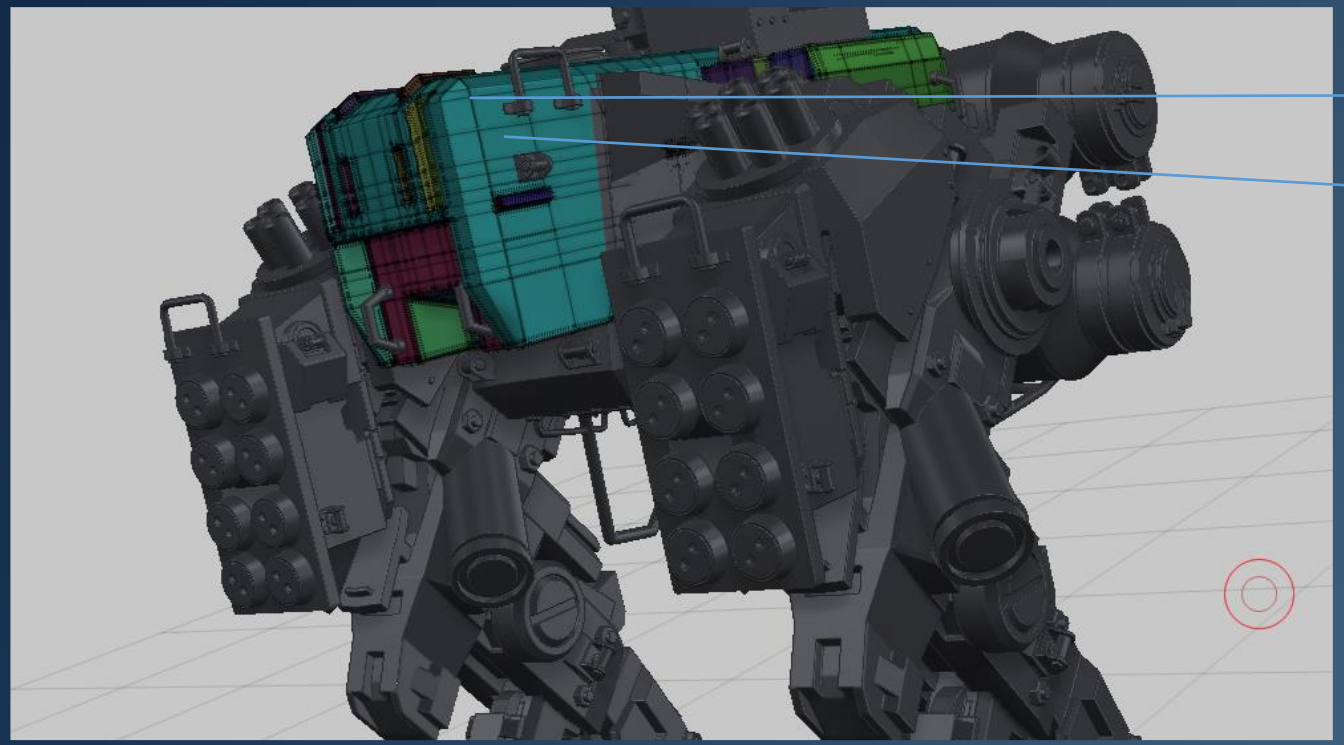
General-purpose computing on graphics processing units

通用图形处理器（general-purpose graphics processing unit，GPGPU）
是一种近年来出现的计算机芯片，它的出现给高性能计算带来了重大突破。

这种功能强大的芯片是在前一个十年里作为高端计算机游戏的图形处理引擎引入的，
是一种大规模并行处理器。它不仅有助于复杂的浮点计算处理，而且容易编程。



什么是 GPGPU



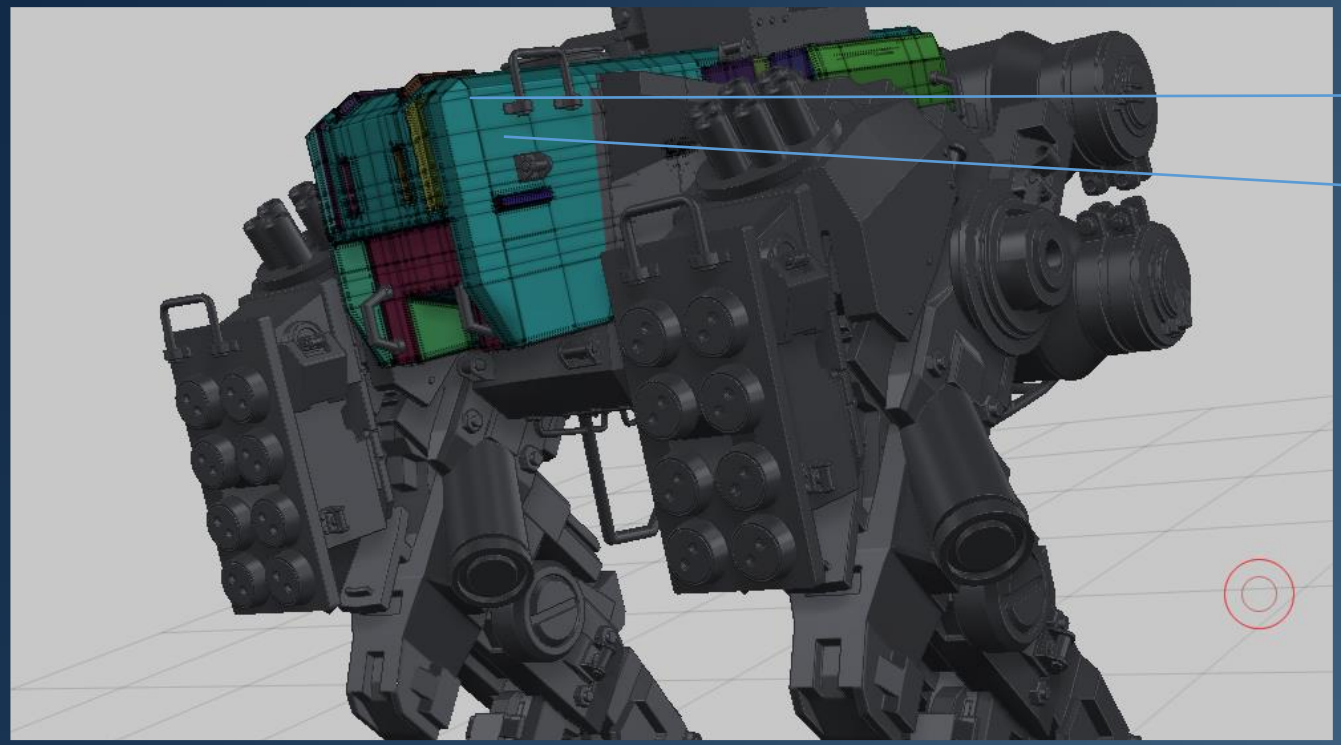
图形渲染要为每一个顶点调用顶点着色器

图形渲染要为每一个像素调用像素着色器

```
for Vertex in model:
    VertexShader(Vertex)

for pixel in view:
    FragementShade(pixel)
```

什么是 GPGPU



图形渲染要为每一个顶点调用顶点着色器

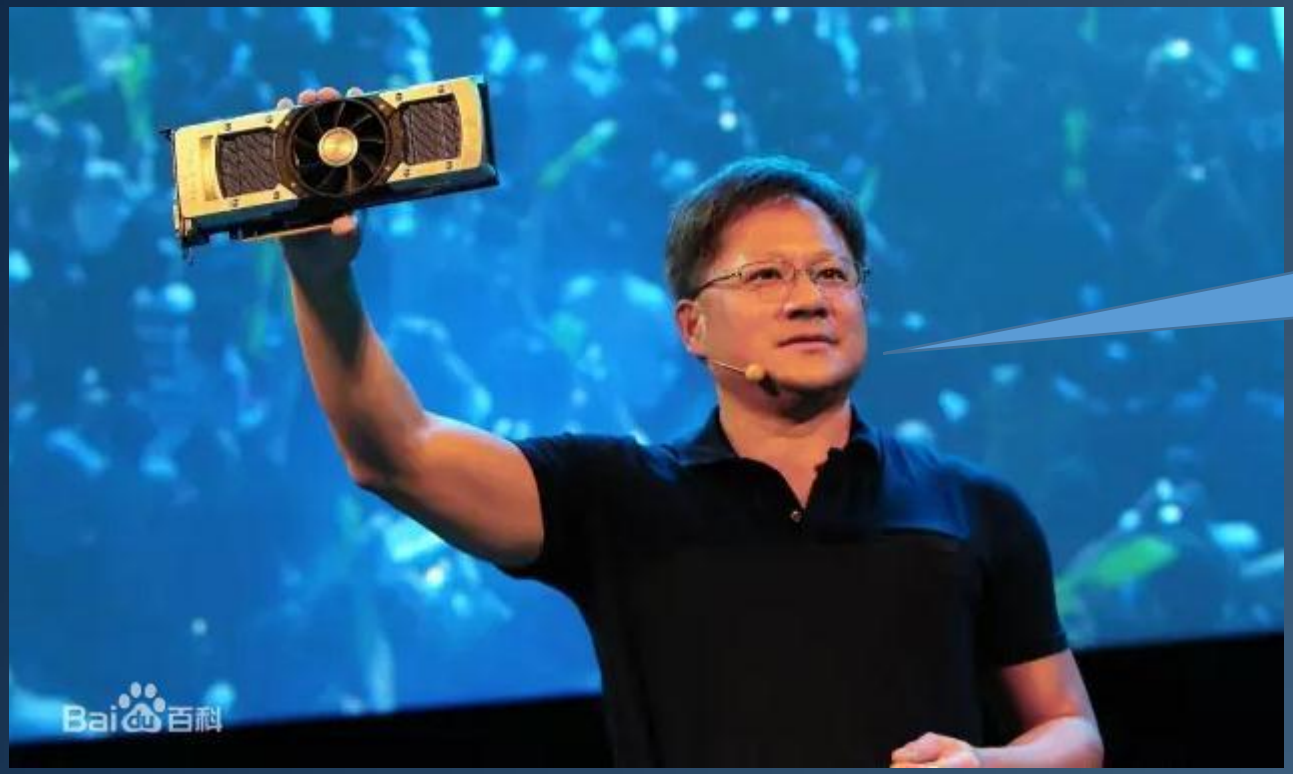
图形渲染要为每一个像素调用像素着色器

```
for Vertex in model:
    VertexShader(Vertex)

for pixel in view:
    FragementShade(pixel)
```

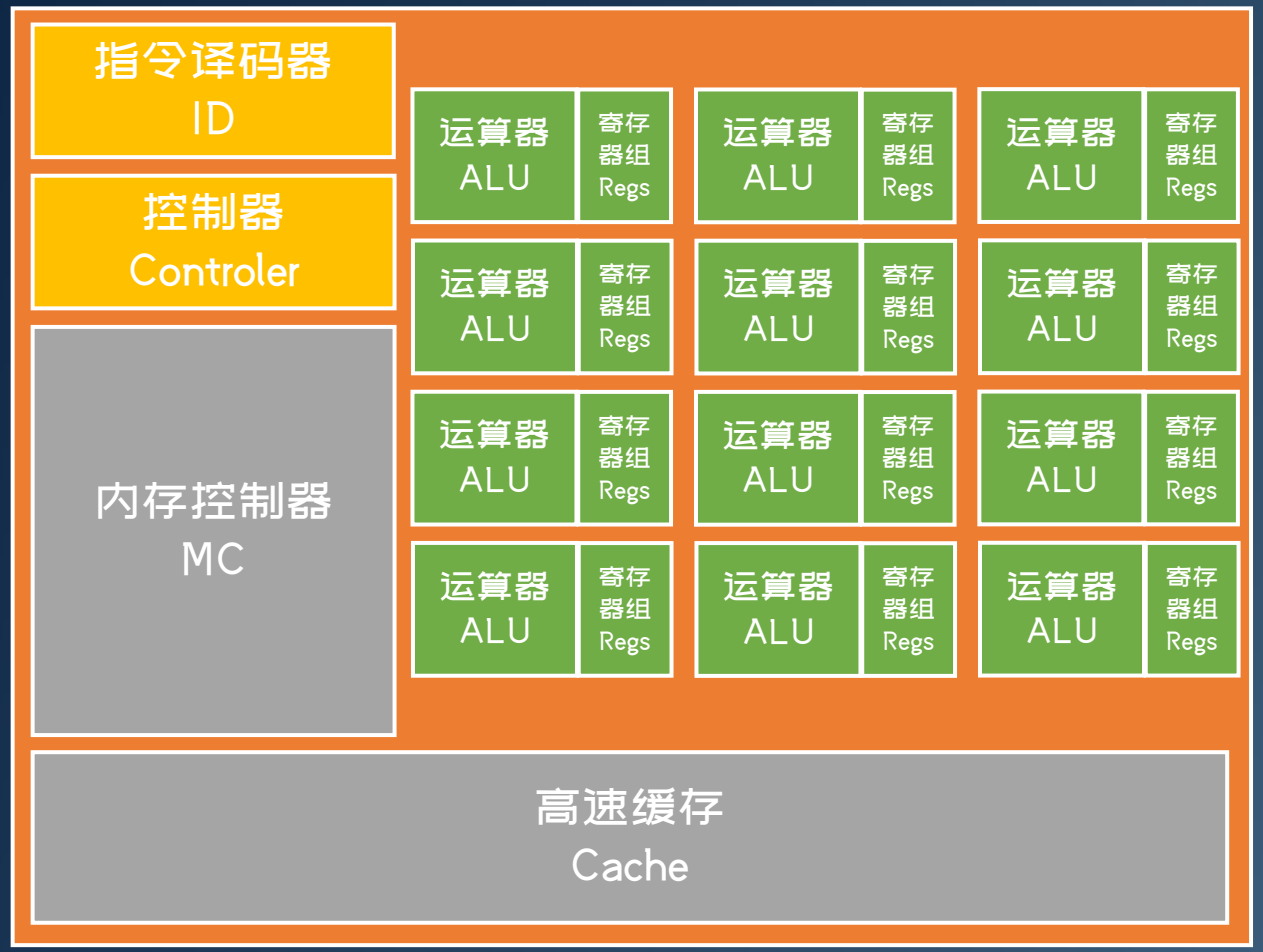
我们需要并行!

什么是 GPGPU



3.2.3 专用芯片架构

ASIC Archticture



- 在GPU的基础上，我们可以更快吗？
- 虽然GPU已经移除了大量指令集系统，但对于特定应用而言，仍然有很多指令是不必须的。
- GPU为了图形运算，还有一些特殊器件包括texture memeory，ray tracing core等等，这些东西也可以被移除。
- GPU的吞吐大，但延迟通常较高，功耗也不占优势能否从芯片设计的角度解决这些问题？

GPU 与科学计算

for Vertex in model:
    VertexShader(Vertex)

for pixel in view:
    FragementShade(pixel)

可编程

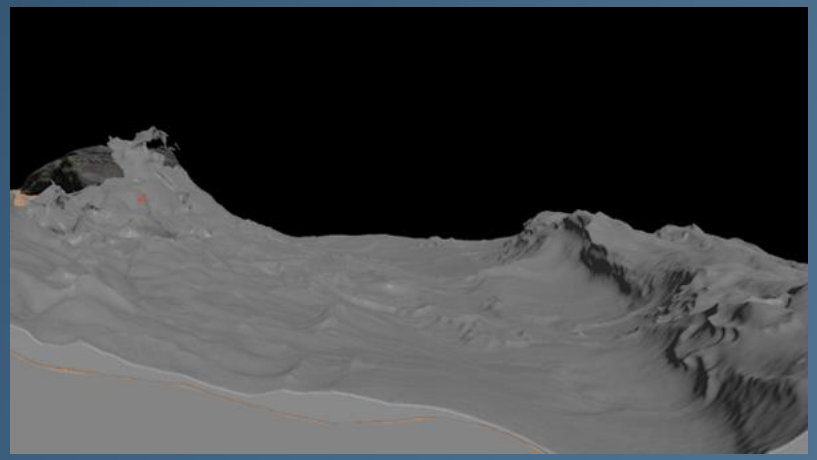
3.2.2 SPH方法求解纳维-斯托克斯方程

CUDA example: Solving NS Equation by SPH method.

$$\rho \frac{Dv}{Dt} = \rho g - \nabla p + \nabla^2 v$$

- 方程描述了流体粒子加速度$\frac{Dv}{Dt}$，与密度ρ、重力g、压力p之间的关系

- 在实际计算中，常用SPH对其进行模拟，算法流程：
  1. 初始化粒子集合S，其中包含100万个粒子
  2. 对于每一个粒子$s_i \in S$，计算其附近密度 ρ_{s_i} 与压强 p_{s_i}
  3. 将 p_{s_i} ρ_{s_i} 代入方程，求解 $\frac{Dv}{Dt}|_{s_i}$
  4. 更新粒子 s_i 的位置与速度。
  5. 更新完所有粒子位置后，返回第二步进行循环。



GPU 与科学计算

- 有限元分析
- 粒子群模拟
- 元胞自动机
- 微分方程
- 矩阵运算

可以用这种编程范式来实现

```
for Vertex in model:
    VertexShader(Vertex)

for pixel in view:
    FragementShade(pixel)
```

GPU 与科学计算

- 有限元分析
- 粒子群模拟
- 元胞自动机
- 微分方程
- 矩阵运算

CUDA（Compute Unified Device Architecture），是显卡厂商NVIDIA推出的运算平台。

CUDA™是一种由NVIDIA推出的通用并行计算架构，该架构使GPU能够解决复杂的计算问题。 它包含了CUDA指令集架构（ISA）以及GPU内部的并行计算引擎。

开发人员可以使用C语言来为CUDA™架构编写程序，所编写出的程序可以在支持CUDA™的处理器上以超高性能运行。CUDA3.0已经开始支持C++和FORTRAN。

GPU 与科学计算

- 有限元分析
- 粒子群模拟
- 元胞自动机
- 微分方程
- 矩阵运算

CUDA（Compute Unified Device Architecture），是显卡厂商NVIDIA推出的运算平台。

CUDA™是一种由NVIDIA推出的通用并行计算架构，该架构使GPU能够解决复杂的计算问题。 它包含了CUDA指令集架构（ISA）以及GPU内部的并行计算引擎。

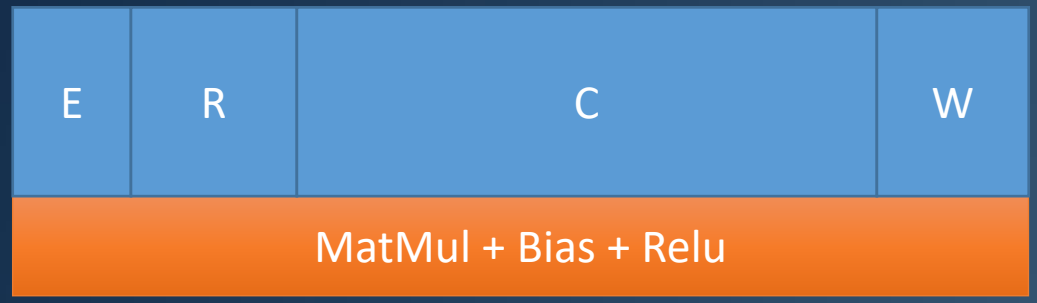
开发人员可以使用C语言来为CUDA™架构编写程序，所编写出的程序可以在支持CUDA™的处理器上以超高性能运行。CUDA3.0已经开始支持C++和FORTRAN。

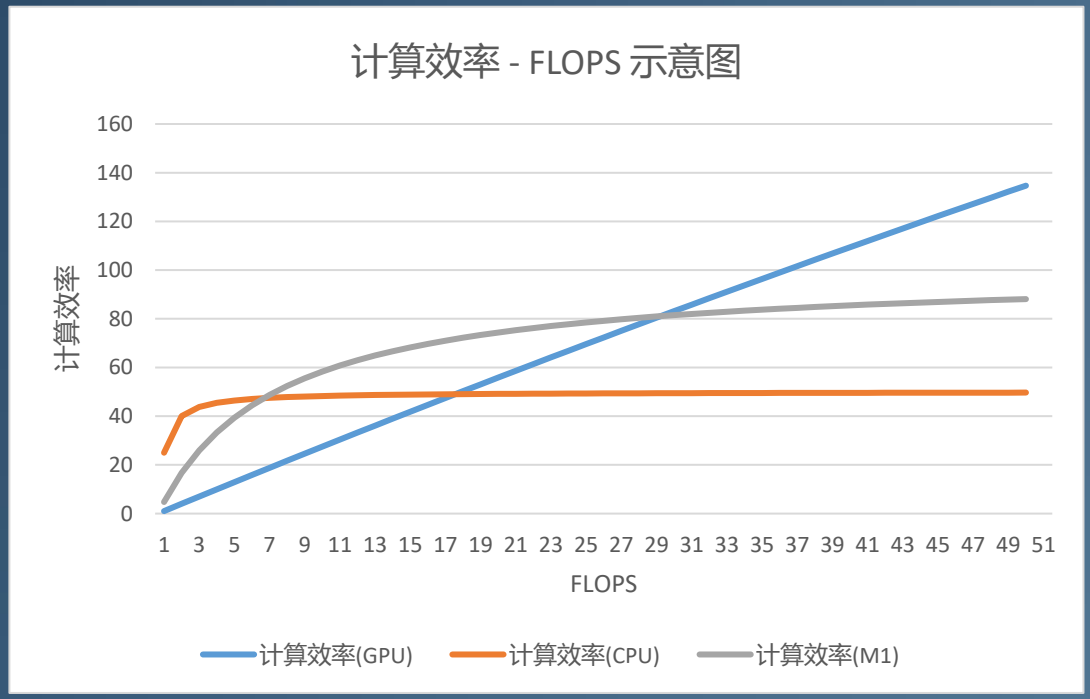
8.1.4 提升算子计算效率

Operator Efficiency

- 提升算子效率，计算量并非越小越好，反而计算量大的算子执行效率更高。

$$\text{计算效率} = \frac{FLOPS}{\text{运行时间}}$$





8.1.4 提升算子计算效率

Operator Efficiency

- 提升算子效率，计算量并非越小越好，反而计算量大的算子执行效率更高。
- 你应该减少算子数量，提高计算密集程度。

Conv (3*3)
in_channel=3
out_channel=3

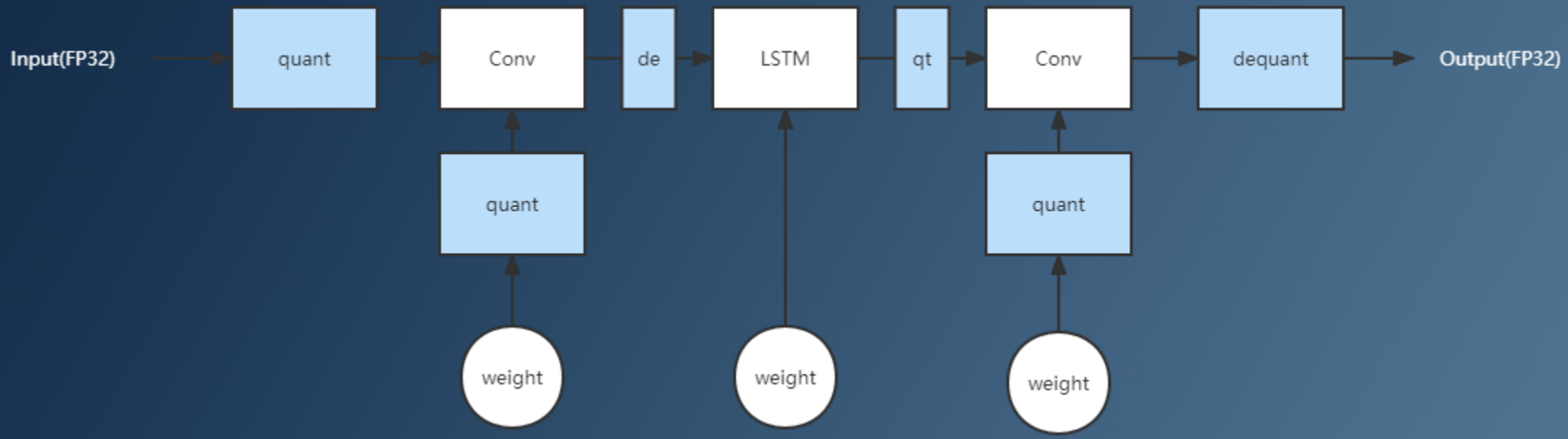
Conv (3*3)
in_channel=3
out_channel=3

Conv (3*3)
in_channel=64
out_channel=64

8.1.5 连贯量化区

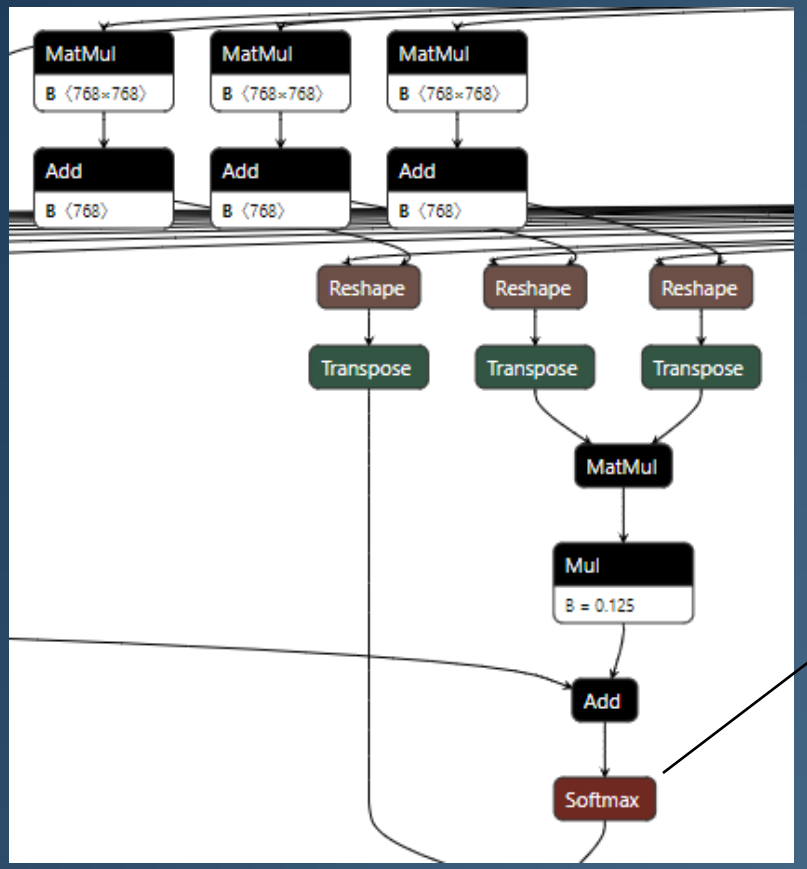
Continuous Quantiation

- 不要在网络中过度使用不可量化算子，比如sigmoid，div，softmax。



8.1.5 连贯量化区
Continuous Quantiation

- 不要在网络中过度使用不可量化算子，比如sigmoid，div，softmax。



FP32 Only!

8.1.6 网络结构与图融合

Graph Fusion & Network Sturcture

- 网络结构设计不能破坏图融合
- 量化点插入不能破坏图融合

13.2.4.1. Layer Fusion

TensorRT attempts to perform many different types of optimizations in a network during the build phase. In the first phase, layers are fused together whenever possible.

Fusions transform the network into a simpler form but preserve the same overall behavior. Internally, many layer implementations have extra parameters and options that are not directly accessible when creating the network. Instead, the fusion optimization step detects supported patterns of operations and fuses multiple layers into one layer with internal options set.

8.1.6 网络结构与图融合

Graph Fusion & Network Sturcture

Supported Layer Fusions：

- ReLU ReLU Activation
- Convolution and ReLU Activation
- Convolution and GELU Activation

The precision of input and output should be the same; with both of them FP16 or INT8. The Activation layer must be GELU type. TensorRT should be running on a Turing or later device with CUDA version 10.0 or later.

- Convolution and Clip Activation
- Scale and Activation
- Convolution And ElementWise Operation
- Padding and Convolution/Deconvolution
- Shuffle and Reduce
- Shuffle and Shuffle
- Scale

A Scale layer that adds 0, multiplied by 1, or computes powers to the 1 can be erased.

8.1.6 网络结构与图融合

Graph Fusion & Network Sturcture

- Convolution and Scale
- Reduce

A Reduce layer that performs average pooling will be replaced by a Pooling layer.

- Convolution and Pooling

The Convolution and Pooling layers must have the same precision. The Convolution layer may already have a fused activation operation from a previous fusion.

- Depthwise Separable Convolution

A depthwise convolution with activation followed by a convolution with activation may sometimes be fused into a single optimized DepSepConvolution layer. The precision of both convolutions must be INT8 and the device's computes capability must be 7.2 or later.

- SoftMax and Log

It can be fused into a single Softmax layer if the SoftMax has not already been fused with a previous log operation.

- SoftMax and TopK

Can be fused into a single layer. The SoftMax may or may not include a Log operation.

- FullyConnected

The FullyConnected layer will be converted into the Convolution layer, all fusions for convolution will take effect.

8.1.6 网络结构与图融合

Graph Fusion & Network Sturcture

13.2.4.3. PointWise Fusion

Multiple adjacent PointWise layers can be fused into a single PointWise layer, to improve performance.

The following types of PointWise layers are supported, with some limitations:

- Activation

Every ActivationType is supported.

- Constant

Only constant with a single value (size == 1).

- ElementWise

Every ElementWiseOperation is supported.

- PointWise

PointWise itself is also a PointWise layer.

- Scale

Only support ScaleMode::kUNIFORM.

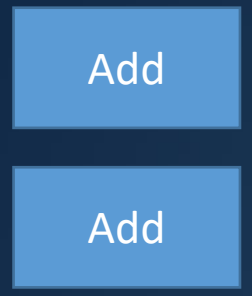
- Unary

Every UnaryOperation is supported.

The size of the fused PointWise layer is not unlimited, therefore, some PointWise layers may not be fused.

8.1.6 网络结构与图融合

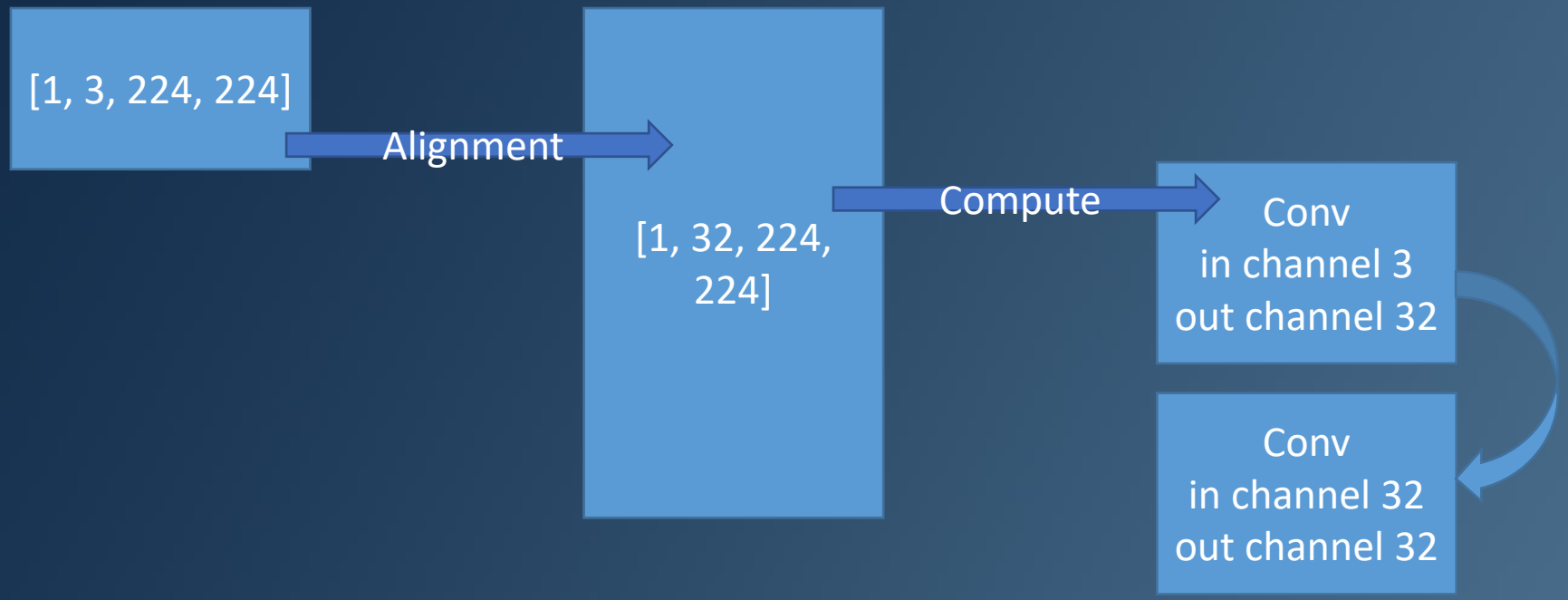
Graph Fusion & Network Sturcture



8.1.7 Tensor 对齐

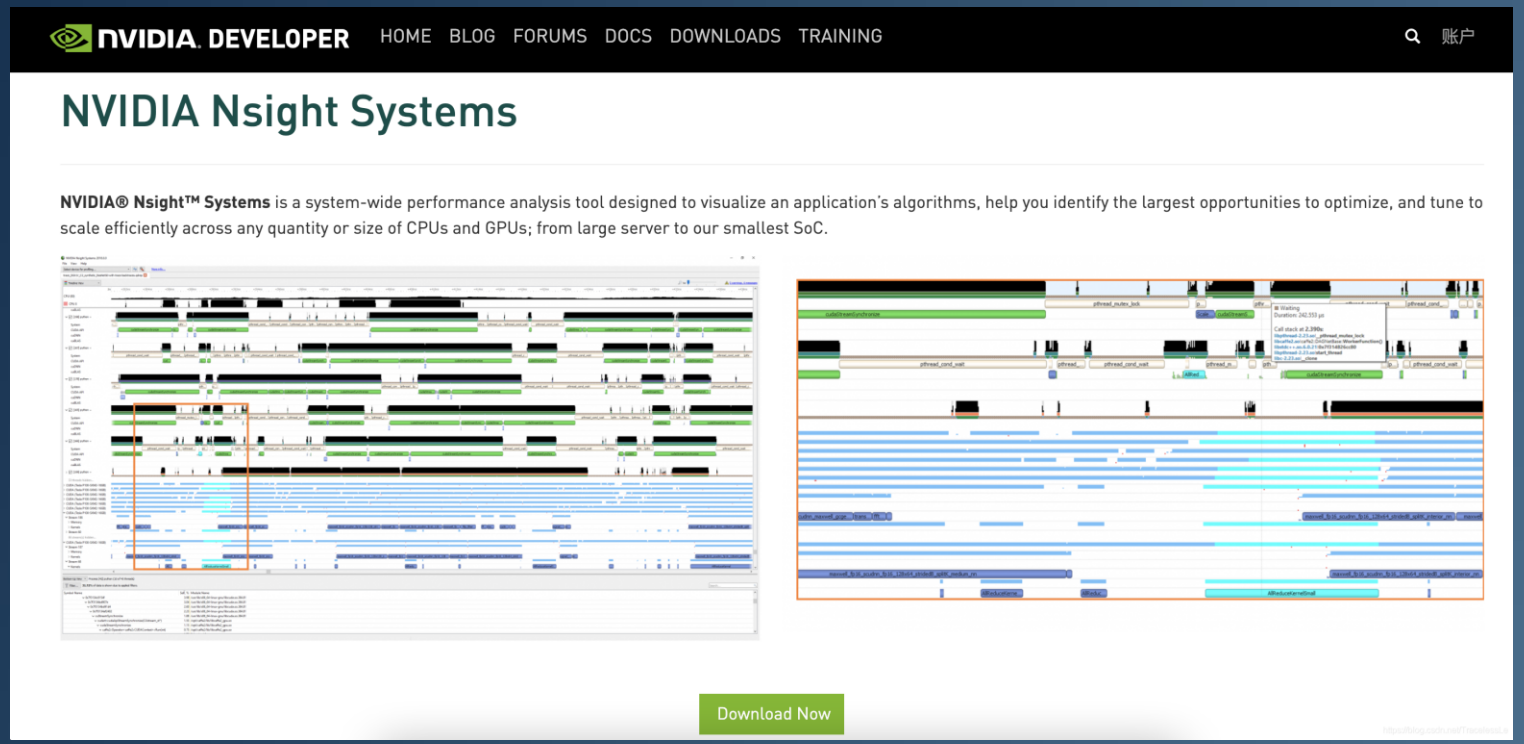
Graph Fusion & Network Sturcture

- 32 channel 对齐(tensor core int8)
- C，H，W 都要对齐(最好都为16的倍数)



8.1.8 Profiling is all your need

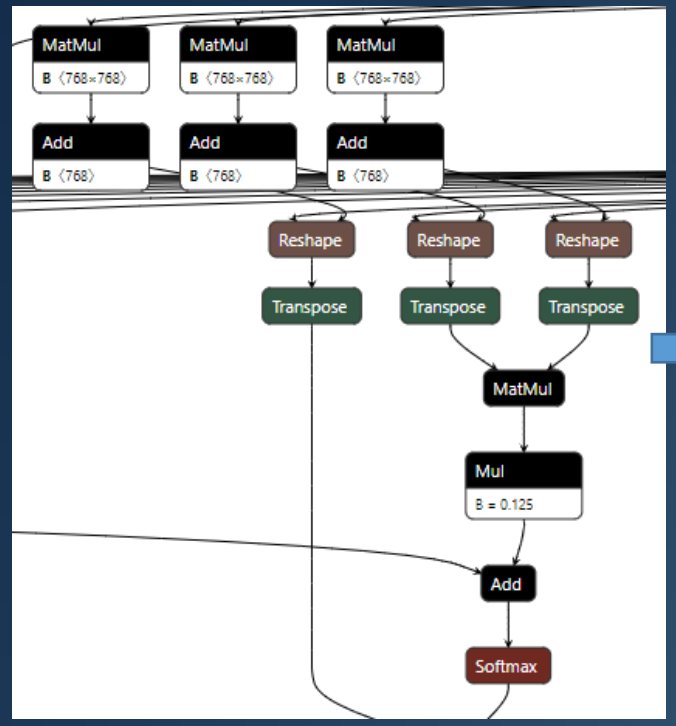
- Profiler with TensorRT



8.1.9 自定义算子
Custimized Plugin

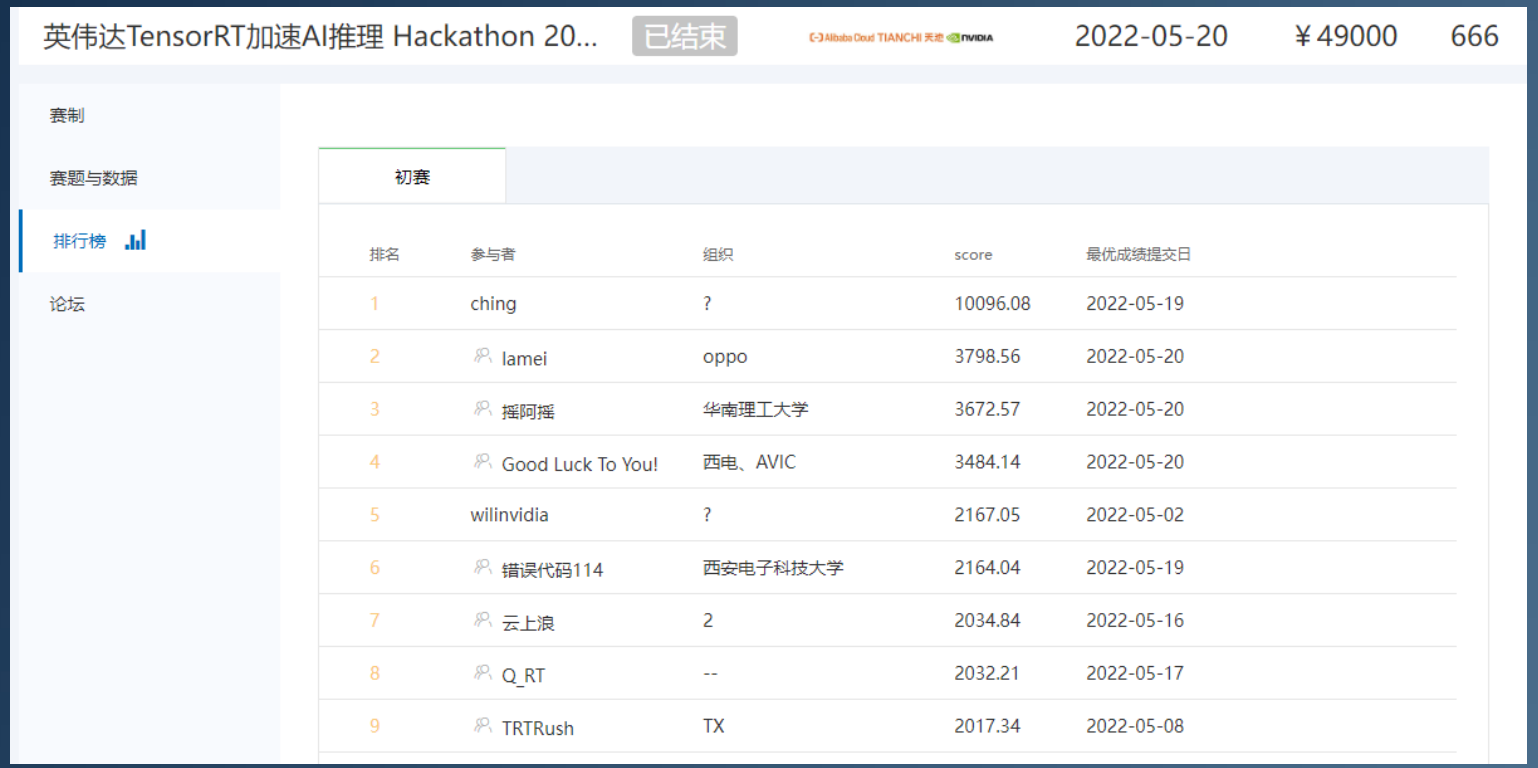
- 必要时自己去写 plugin

  https://github.com/NVIDIA/TensorRT/tree/main/plugin



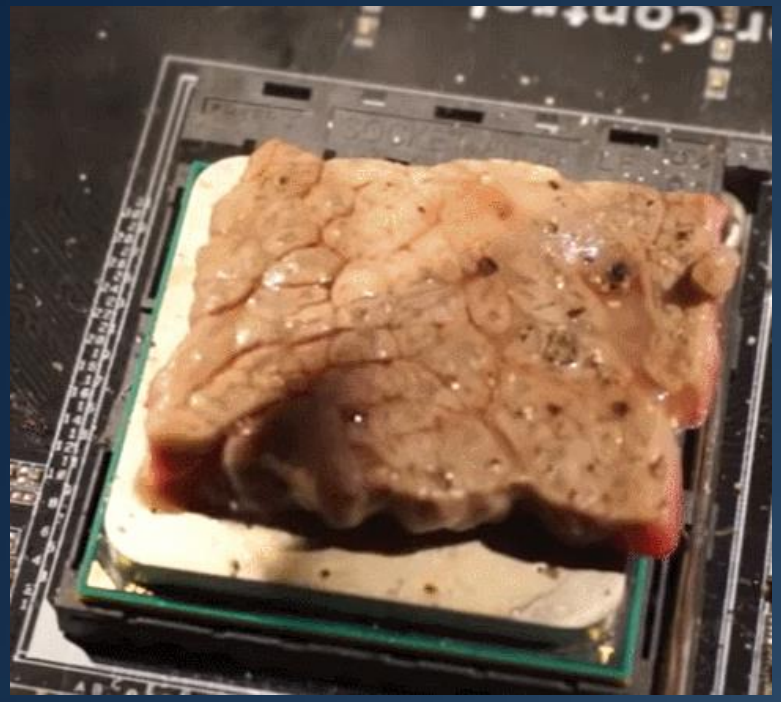
GPU 与科学计算

- Profiler with TensorRT



排名	参与者	组织	score	最优成绩提交日
1	ching	?	10096.08	2022-05-19
2	lamei	oppo	3798.56	2022-05-20
3	摇阿摇	华南理工大学	3672.57	2022-05-20
4	Good Luck To You!	西电、AVIC	3484.14	2022-05-20
5	wilinvidia	?	2167.05	2022-05-02
6	错误代码114	西安电子科技大学	2164.04	2022-05-19
7	云上浪	2	2034.84	2022-05-16
8	Q_RT	--	2032.21	2022-05-17
9	TRTRush	TX	2017.34	2022-05-08

联系我们 https://github.com/openppl-public



广告位招租

微信群

QQ群（入群密令OpenPPL）