编写模型配置

一、最小模型配置

必要参数：

platform/backend：用于指定后端，大部分情况二选一，特殊情况需要特殊对待，见后面。

max_batch_size：指定最大 batch。

input、output：输入输出 Tensor 的名字和信息。

注意，对于 Tensorrt，TensorFlow save-model，onnx 模型，config.pbtxt 不是必须的，只要启动指定 **--strict-model-config=false**

二、platform 和 backend 的异同

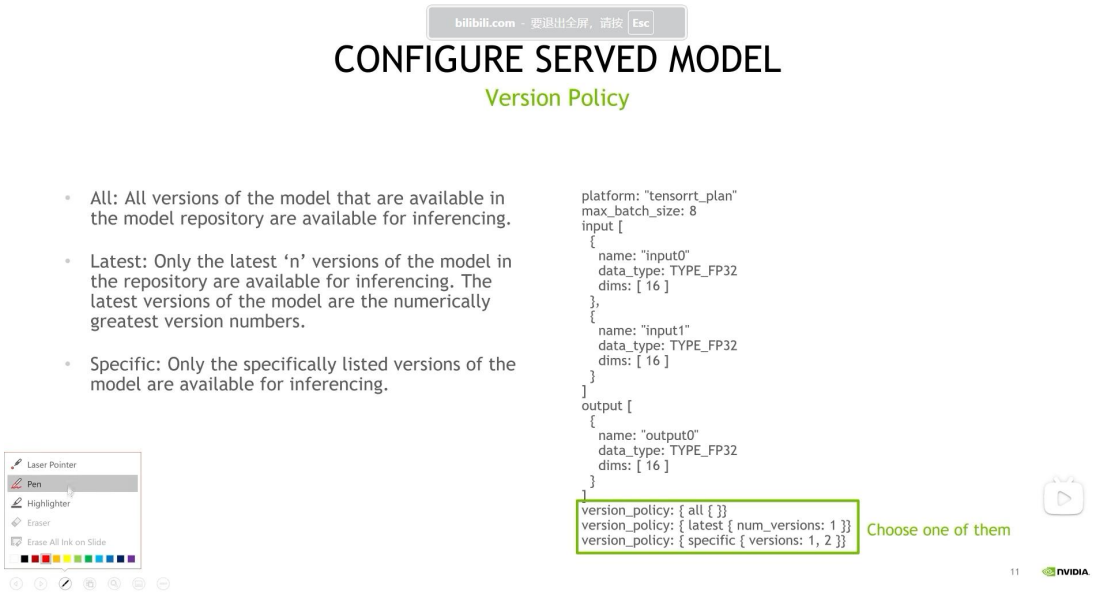
对于 **Tensorrt、onnxrt、pytorch**，这两种参数二选一即可。

对于 **TensorFlow** 必须指定 platform，backend 可选。

对于 **openvino，python，dali**，只能使用 backend。

对于**Custom**,21.05 版本之前,可以通过 platform 参数设置为 custom 表示;之后必须通过 backend 字段进行指定，值为你的 custom backend library 的名字。

三、max_batch_size&input&output



情况 1：

max_batch_size 为一个大于 0 的常数，Input 和 output 指定名字，数据类型，数据形状。

注意：dims 在指定的时候忽略 batch_size 的维度。

情况 2：

max_batch_size 等于 0。表示模型的输入和输出是不包括 batch_size 那个维度的。

这个时候维度信息就是真实的维度信息。

情况 3：

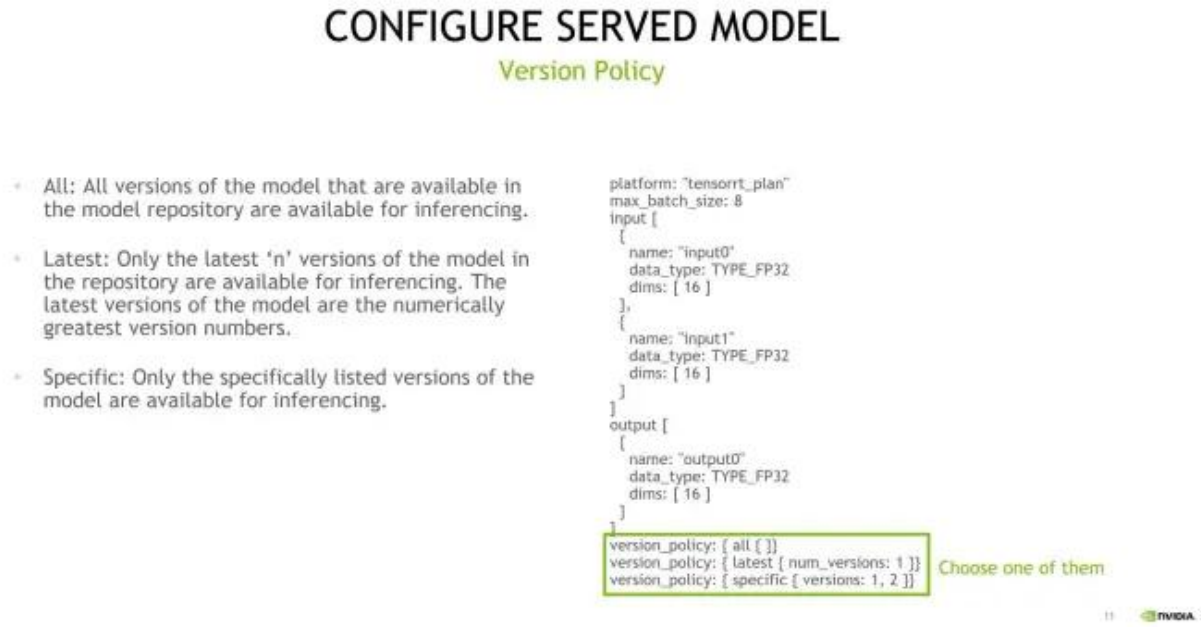
pytorch 特殊情况，torchscript 模型不保存输入输出的名字，因此对输入输出名称有特殊规定，"字符串__数字"。

支持可变 shape，设置为-1。

情况 4：

reshape 参数：对输入输出进行 reshape。

四、模型版本管理——version_policy



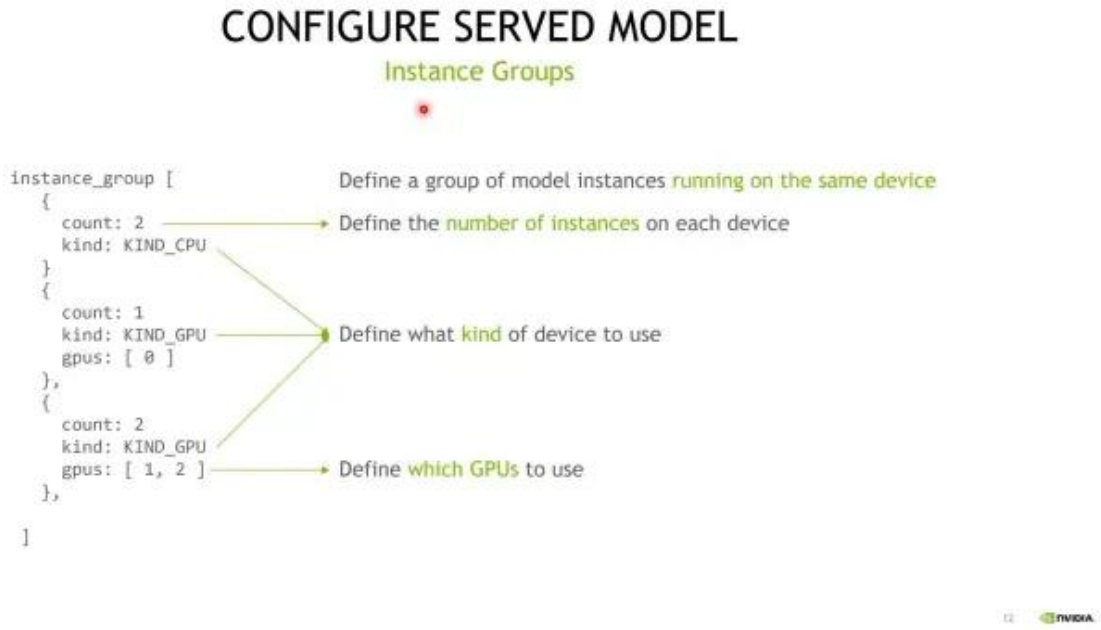
version_policy 参数，策略：

all：加载所有版本的模型。

latest：加载最新的模型（可多个，版本号越大越新）

specific：指定特定的版本。

五、Instance Groups



对应 triton 的并行计算能力特性，这个参数主要用来配置在指定设备上运行多个实例，提高模型服务能力，增加吞吐。

Instance Groups 配置跑在同样设备上的一组模型实例。

count：同时开启的模型数量。

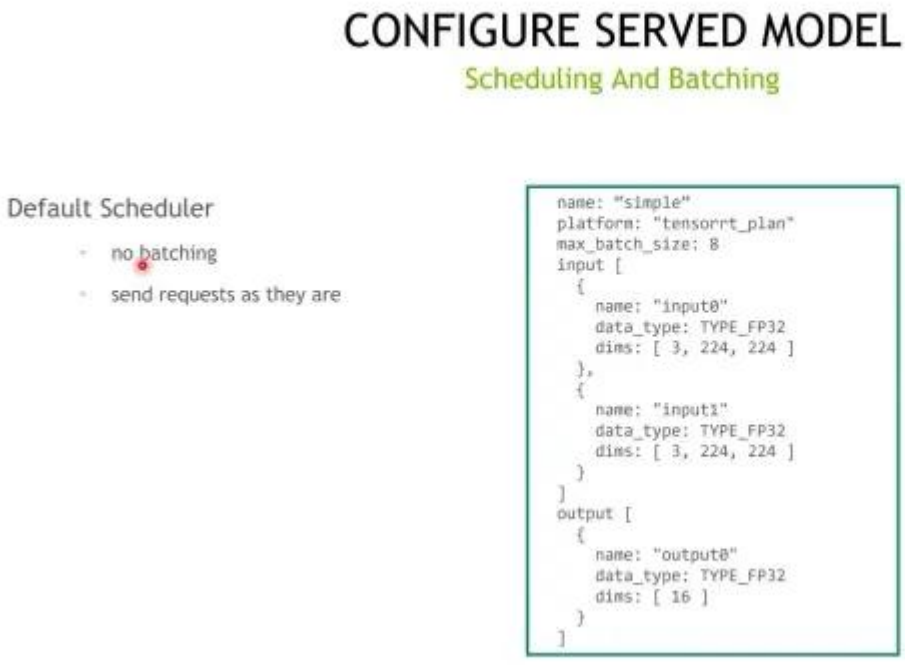
kind：指定设备类型。

gpus：指定 GPU 编号，如果不指定这个参数，triton 会在每个 GPU 上跑相应数量的 instance。可配置多组。

六、Scheduling And Batching

Scheduling：指定调度策略来应对请求。

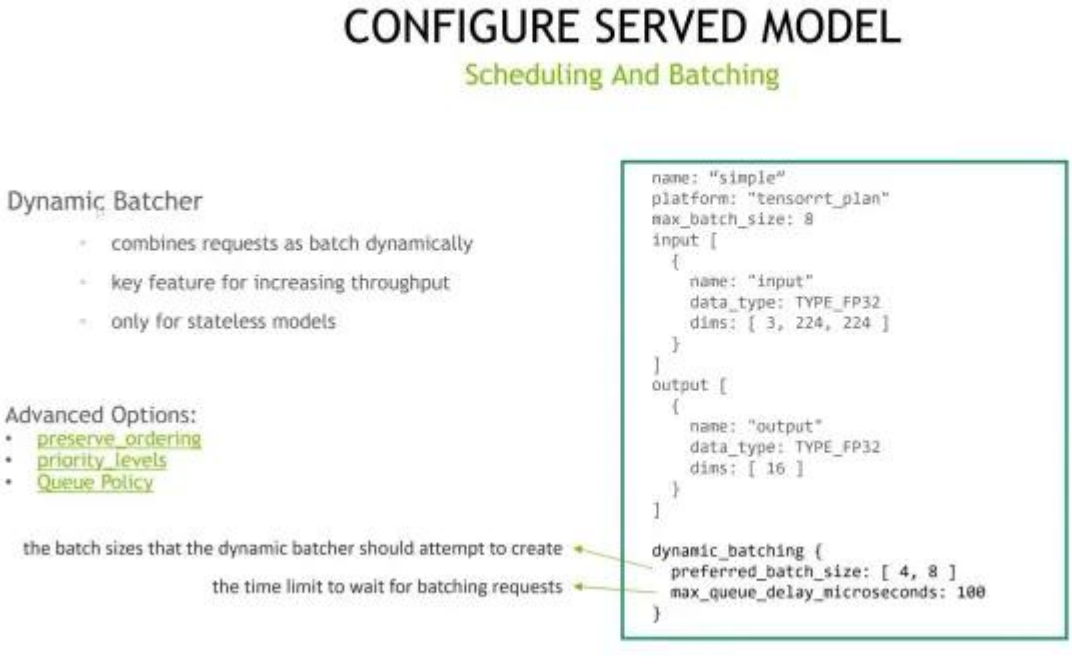
6.1 Default Scheduler



不做 batching；

输入进来是多少就按照多少去推理；

6.2 Dynamic Batcher



- 在服务端将多少个 batch_size 比较小的 input_tensor 合并为一个 batch_size 比较大的 input_tensor；
- 提高吞吐率的关键手段；

- 只适合无状态模型；

子参数：

preferr_batch_size：期望达到的 batch_size 是多少，多个值；

max_queue_delay_microseconds：打成 batch 的时间限制，微秒；

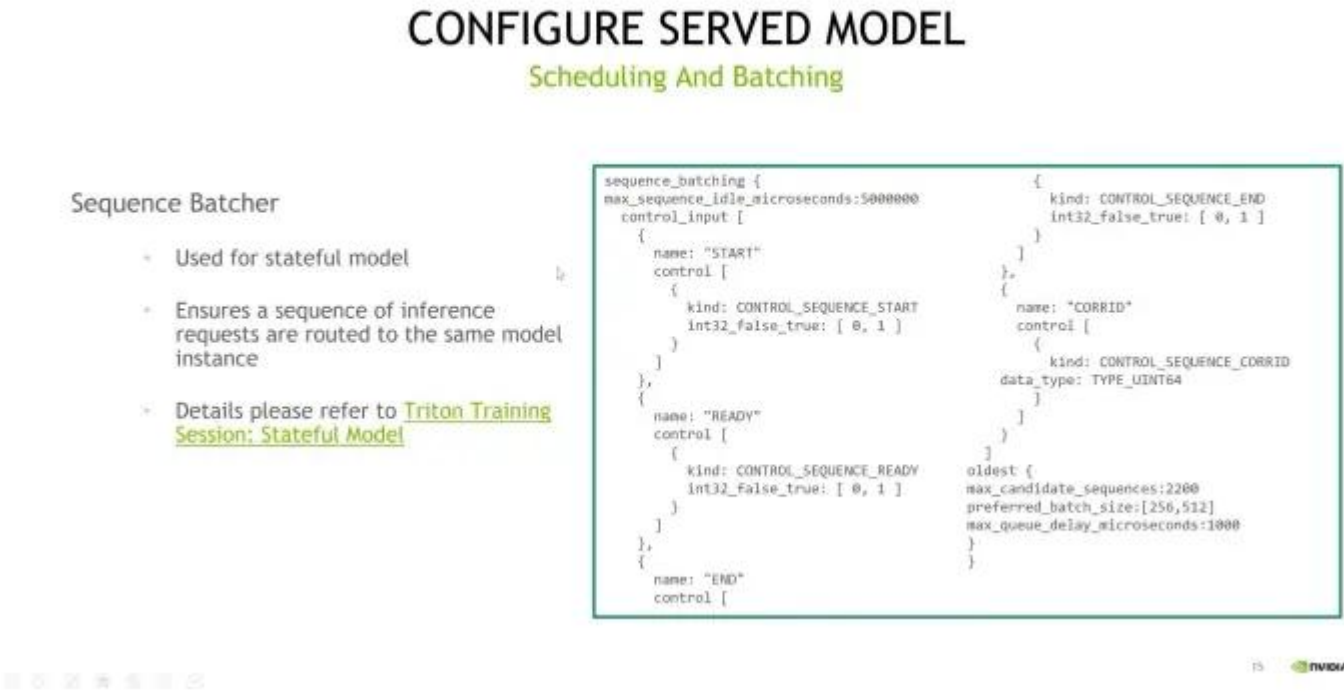
高级子参数：

preserver_ordering：请求进来的顺序和响应出去的顺序保持一致；

priority_levels：定义不同优先级请求处理顺序；

Queue_Policy：设置请求等待队列行为；

6.3 Sequence Batcher



- 专门用于 stateful model 的一种调度器；
- 确保同一序列的推理请求能够路由到同样的模型实例上推理；

6.4 Ensemble Scheduler

- 组合不同的模块，形成 pipeline；
- 后面详细介绍。

七、Optimization Policy

CONFIGURE SERVED MODEL

Optimization Policy

ONNX with TensorRT Optimization

- Make use of TRT backend for ONNX

```
optimization { execution_accelerators {
  gpu_execution_accelerator : [ {
    name : "tensorrt"
    parameters { key: "precision_mode" value: "FP16"
}
    parameters { key: "max_workspace_size_bytes"
value: "1073741824" }
    }]
}}
```

TensorFlow with TensorRT Optimization

- Make use of TF-TRT

```
optimization { execution_accelerators {
  gpu_execution_accelerator : [ {
    name : "tensorrt"
    parameters { key: "precision_mode" value: "FP16"
}}]
}}
```


- Onnx 模型优化——TRT backend for ONNX；
- TensorFlow 模型优化——TF-TRT；

八、Model Warmup

CONFIGURE SERVED MODEL

Model Warmup

- Initialization may be deferred until the model receives its first few inference requests
- model_warmup makes Triton not show the model as ready until warmup has completed
- will cause Triton to be less responsive to model update

```
model_warmup [
  {
    batch_size: 64
    name: "warmup_requests"
    inputs {
      key: "input"
      value: {
        random_data: true
        dims: [ 299, 299, 3]
        data_type: TYPE_FP32
      }
    }
  }
]
```


指定模型热身的参数；

- 初始化可能延迟，直到收到前面几个推理请求；

- 热身完成后，Triton 的服务才是 Ready 状态；
- 模型加载会变长；