# [Vllm - Discussion] Async Tensor Parallelism + Sequence Parallelism on Llama70B

## Benchmark `Llama-3.1-70B-Instruct` using vLLM Open Source Script:

As the benchmark_latency.py is under deprecation, vLLM recommend to use benchmark_serving.py
So the below result is using benchmark_serving.py

We are not able to see the performance gain (TTFT) after apply SP + Async TP.  Details are below.

**SET UP**

**Without Async TP + SP**

```
#Terminal 1 - Server
python3 -m vllm.entrypoints.openai.api_server \
    --port 3000\
    --model "meta-llama/Llama-3.1-70B-Instruct" \
    -tp 8 \
    --disable-log-requests \
    -O3 \
    --dtype "auto" \
    --enable-chunked-prefill \
    --gpu-memory-utilization 0.9 \
    --max-num-batched-tokens 10240 \
    --max-model-len 131072 \
    --max_seq_len_to_capture 131072 \
    --max-num-seqs 1 \
    --block-size 16 \
    --no-enable-prefix-caching \
    -O '{"level":3,"compile_sizes":[8192]}'


#Terminal 2 - Client
vllm bench serve \
    --port 3000 \
    --model "meta-llama/Llama-3.1-70B-Instruct" \
    --request-rate 1 \
    --num-prompts 20 \
    --random-input-len 8192 \
    --random-output-len 1 \
    --tokenizer "meta-llama/Llama-3.1-70B-Instruct" \
    --ignore-eos
```

**With Async TP + SP**

```
#Terminal 1 - Server
python3 -m vllm.entrypoints.openai.api_server \
    --port 3000\
    --model "meta-llama/Llama-3.1-70B-Instruct" \
    -tp 8 \
    --disable-log-requests \
    -O3 \
    --dtype "auto" \
    --enable-chunked-prefill \
    --gpu-memory-utilization 0.9 \
    --max-num-batched-tokens 10240 \
    --max-model-len 131072 \
    --max_seq_len_to_capture 131072 \
    --max-num-seqs 1 \
    --block-size 16 \
    --no-enable-prefix-caching \
    -O '{"level":3,"compile_sizes":[8192],"pass_config":{"enable_async_tp":true,"enabl


#Terminal 2 - Client
vllm bench serve \
    --port 3000 \
    --model "meta-llama/Llama-3.1-70B-Instruct" \
    --request-rate 1 \
    --num-prompts 20 \
    --random-input-len 8192 \
    --random-output-len 1 \
    --tokenizer "meta-llama/Llama-3.1-70B-Instruct" \
    --ignore-eos
```

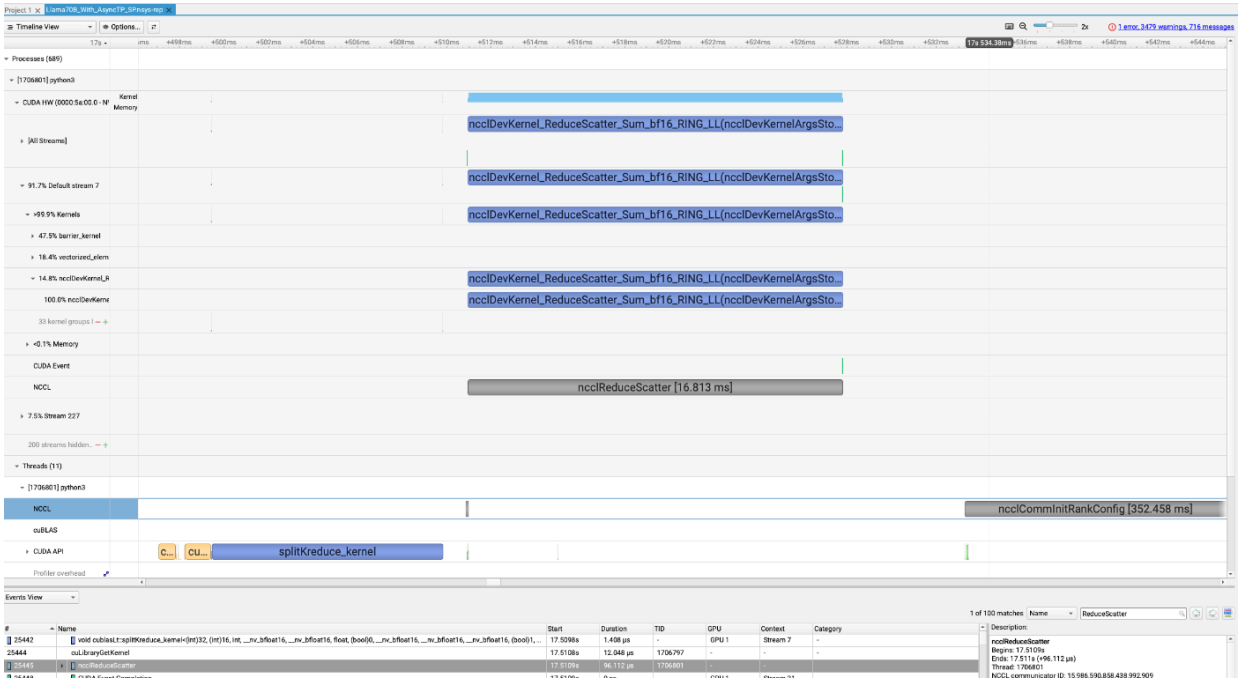
| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Purpose | # | Model | TP | Input_Token_Length | Output_Token_Length | batch_size | requests | prefix_caching | chunk prefill | compile_size | Async TP + SP | Mean TTFT (ms) | Median TTFT (ms) | P99 TTFT (ms) | |
| 2 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 1 | 1 | 20 | No | Yes | [8192] | No | 459.71 | 354.99 | 1157.72 | |
| 3 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 1 | 1 | 20 | No | Yes | [8192] | Yes | 459.78 | 355.11 | 1158.17 | |
| 4 | | | | | | | | | | | | | -0.01523 | -0.0338 | -0.03887 | No gain |
| 5 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 300 | 1 | 20 | No | Yes | [8192] | No | 24061.76 | 23779.28 | 47047.53 | |
| 6 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 300 | 1 | 20 | No | Yes | [8192] | Yes | 23990.1 | 23708.73 | 46898.13 | |
| 7 | | | | | | | | | | | | | 0.29782 | 0.29669 | 0.31755 | No gain |
| 8 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 1 | 8 | 20 | No | Yes | [8192] | No | 471.83 | 355.4 | 1189.17 | |
| 9 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 1 | 8 | 20 | No | Yes | [8192] | Yes | 470.72 | 353.87 | 1182.13 | |
| 10 | | | | | | | | | | | | | 0.23525 | 0.4305 | 0.59201 | No gain |
| 11 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 300 | 8 | 20 | No | Yes | [8192] | No | 478.42 | 364.35 | 1191.6 | |
| 12 | | | Llama-3.1-70B-Instruct | 8 | 8192 | 300 | 8 | 20 | No | Yes | [8192] | Yes | 477.81 | 362.67 | 1187.76 | |
| 13 | | | | | | | | | | | | | 0.1275 | 0.4611 | 0.32226 | No gain |
| 14 | | | Llama-3.1-70B-Instruct | 8 | 16384 | 300 | 8 | 20 | No | Yes | [16384] | No | 1865.19 | 1695.19 | 3863.96 | |
| 15 | | | Llama-3.1-70B-Instruct | 8 | 16384 | 300 | 8 | 20 | No | Yes | [16384] | Yes | 1867.31 | 1700.65 | 3865.22 | |
| 16 | | | | | | | | | | | | | -0.11366 | -0.32209 | -0.03261 | Regression |

## NSys Kernel Profiling for Lllama70B

By conducting the NSys profiling with 4 layer of Llama-3.1-70B-Instruct, the Sequence Parallelism PR and Async Tensor Parallelism PR did take effect to split and fuse kernels:

- without Async TP + SP: ReduceScatter is 0,  AllGather is 64, AllReduce is 448.
- with Async TP + SP:  ReduceScatter is 100, AllGather is 116, AllReduce is 232.

However, the expected overlap between communication and computation seems not happening at all expected places. For instance, `ReduceScatter` kernels are only introduced with Async TP + SP, NSys graphs show **no compute operations occurring concurrently** during the `ReduceScatter` phase.




(NSys Llama70B_With_AsyncTP_SP.nsys-rep  4 Layers Llama70B,  Input 8192, Output 1 --- **Prefill ONLY** )

**SET UP**

```
nsys profile --trace=nvtx,cuda,cudnn,cublas \
--delay 30 \
-o /data/NSysReport/Llama70B_Without_AsyncTP_SP.nsys-rep \
--trace-fork-before-exec=true \
--cuda-graph-trace=node \
--force-overwrite=true \
python3 smoke_test.py --input 8192 --output 1
```

```
hidden_laterys = 4
Input_token_length = 8192
Output_token_length = 1
Barch_size = 1
TP = 4
```

**WITHOUT ASYNC TP + SP**

```
config = CompilationConfig(
    level=3,
    custom_ops=["+rms_norm"],
    compile_sizes=[4, 8, 16],
    splitting_ops=[],
)

config.pass_config.enable_sequence_parallelism= False
config.pass_config.enable_async_tp= False

kwargs = dict(
    model="/data/Llama-3.1-70B-Instruct",
    tokenizer="/data/Llama-3.1-70B-Instruct",
    tokenizer_mode="auto",
    seed=0,
    max_num_seqs=1, #Batch_Size = 1
    max_model_len=ModelConstants.MAX_TOTAL_TOKEN_LENGTH,
    max_num_batched_tokens=ModelConstants.MAX_NUM_BATCHED_TOKENS,
    gpu_memory_utilization=ModelConstants.VLLM_GPU_MEMORY_UTILIZATION,
    swap_space=ModelConstants.VLLM_SWAP_SPACE,
    tensor_parallel_size=4, #TP = 4
    trust_remote_code=False,
    disable_log_stats=False,
    dtype="auto",
    enable_prefix_caching=ModelConstants.ENABLE_PREFIX_CACHING, #disabled
    kv_cache_dtype=ModelConstants.KV_CACHE_DTYPE,
    block_size=ModelConstants.GPU_VLLM_BLOCK_SIZE,
    enable_chunked_prefill=ModelConstants.ENABLE_CHUNK_PREFILL, #enabled
    collect_time_per_step=False,
    enforce_eager=False, #newly_added
    compilation_config=config #newly_added
)
if USE_DUMMY_WEIGHT:
    kwargs["load_format"] = "dummy"
```

**NSYS ARTIFACTS**

Llama70B_Without_AsyncTP_SP.nsys-rep

**WITH ASYNC TP + SP**

```
config = CompilationConfig(
    level=3,
    custom_ops=["+rms_norm"],
    compile_sizes=[4, 8, 16],
    splitting_ops=[],
)

config.pass_config.enable_sequence_parallelism= True
config.pass_config.enable_async_tp= True

kwargs = dict(
    model="/data/Llama-3.1-70B-Instruct",
    tokenizer="/data/Llama-3.1-70B-Instruct",
    tokenizer_mode="auto",
    seed=0,
    max_num_seqs=1, #Batch_Size = 1
    max_model_len=ModelConstants.MAX_TOTAL_TOKEN_LENGTH,
    max_num_batched_tokens=ModelConstants.MAX_NUM_BATCHED_TOKENS,
    gpu_memory_utilization=ModelConstants.VLLM_GPU_MEMORY_UTILIZATION,
    swap_space=ModelConstants.VLLM_SWAP_SPACE,
    tensor_parallel_size=4, #TP = 4
    trust_remote_code=False,
    disable_log_stats=False,
    dtype="auto",
    enable_prefix_caching=ModelConstants.ENABLE_PREFIX_CACHING, #disabled
    kv_cache_dtype=ModelConstants.KV_CACHE_DTYPE,
    block_size=ModelConstants.GPU_VLLM_BLOCK_SIZE,
    enable_chunked_prefill=ModelConstants.ENABLE_CHUNK_PREFILL, #enabled
    collect_time_per_step=False,
    enforce_eager=False, #newly_added
    compilation_config=config #newly_added
)
if USE_DUMMY_WEIGHT:
    kwargs["load_format"] = "dummy"
```

**NSys Artifacts**

Llama70B_With_AsyncTP_SP.nsys-rep