1.环境准备

镜像位置在 `hjbog-srdc-25:/mnt/raid0/hangy/rtp_llm_perf_0.0.0.tar` ，解压之后可以得到镜像 `rtp_llm_perf:0.0.0` 。使用如下命令启动docker container:

```
docker run --privileged --cap-add=SYS_PTRACE --security-opt seccomp=unconfined \
--ulimit nofile=655350:655350 --ulimit memlock=-1 --cpu-shares=15360 --cpu-quota=9600000 \
--cpu-period=100000 --memory=500000m -d --network=host --device=/dev/kfd --device=/dev/dri \
--ipc=host --group-add video --name <container_name> rtp_llm_perf:0.0.0 /sbin/init
```

2.启动RTP-LLM server

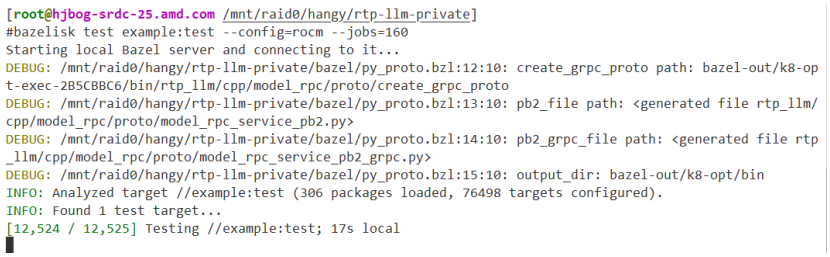
准备代码并checkout到 `develop/qwen3-rocm-main_more_opt` 分支

```
git clone https://github.com/alibaba/rtp-llm.git
cd rtp-llm
git checkout develop/qwen3-rocm-main_more_opt
```

编译及启动server有两种方法

2.1 启动方法一

- 运行编译命令 `bazelisk test example:test --config=rocm --jobs=160` ，该命令会编译一个bazel py_test目标，我们并不需要这个测试的结果，**只需要这个命令产生的sandbox环境**，当编译日志出现 `Testing //example:test;` 时即可终止程序，如下图所示



- cd到sandbox环境： `cd bazel-bin/example/test.runfiles/rtp_llm/`
- 启动server：

```
# for bf16 model
LD_PRELOAD=/opt/rocm/lib/libhipblaslt.so \
USE_ASM_PA=1 REUSE_CACHE=0 \
ENABLE_MERGE_W13=1 USE_SWIZZLEA=1 USE_AITER_PA=1 \
DEVICE_RESERVE_MEMORY_BYTES=-21474836480 \
AITER_ASM_DIR=/root/.cache/bazel/_bazel_root/e409d0075c719073d2cc406d6a042faf/external/aite
MAX_SEQ_LEN=10000 CONCURRENCY_LIMIT=1024 \
WORLD_SIZE=1 DP_SIZE=1 TP_SIZE=1 \
START_PORT=8099 WARM_UP=0 ACT_TYPE=bf16 \
TOKENIZER_PATH=<model_path> \
CHECKPOINT_PATH=<model_path> \
MODEL_TYPE=qwen_3 \
FT_SERVER_TEST=1  \
SEQ_SIZE_PER_BLOCK=16 \
LD_LIBRARY_PATH=/opt/rocm/lib:/opt/conda310/lib/:/usr/local/nvidia/lib64:/usr/lib64:/usr/lo
/opt/conda310/bin/python3.10 -m rtp_llm.start_server

# for fp8 model
LD_PRELOAD=/opt/rocm/lib/libhipblaslt.so \
USE_ASM_PA=1 REUSE_CACHE=0 QUANTIZATION=FP8 KV_CACHE_DTYPE=fp8 \
ENABLE_MERGE_W13=1 USE_SWIZZLEA=1 USE_AITER_PA=1 \
DEVICE_RESERVE_MEMORY_BYTES=-21474836480 \
AITER_ASM_DIR=/root/.cache/bazel/_bazel_root/e409d0075c719073d2cc406d6a042faf/external/aite
MAX_SEQ_LEN=10000 CONCURRENCY_LIMIT=1024 \
WORLD_SIZE=1 DP_SIZE=1 TP_SIZE=1 \
START_PORT=8099 WARM_UP=0 ACT_TYPE=bf16 \
TOKENIZER_PATH=<model_path> \
CHECKPOINT_PATH=<model_path> \
MODEL_TYPE=qwen_3 \
FT_SERVER_TEST=1  \
SEQ_SIZE_PER_BLOCK=16 \
LD_LIBRARY_PATH=/opt/rocm/lib:/opt/conda310/lib/:/usr/local/nvidia/lib64:/usr/lib64:/usr/lo
CUDA_VISIBLE_DEVICES=7 /opt/conda310/bin/python3.10 -m rtp_llm.start_server
```

请注意设置 START_PORT， TOKENIZER_PATH， CHECKPOINT_PATH 等变量

NOTE：bazel编译产生的hashid可能不一样(上面命令中的 e409d0075c719073d2cc406d6a042faf)，请按需调整

- 验证server是否启动成功，可以使用命令 `netstat -tnlp | grep <START_PORT>` ，如果有显示进程，则说明server启动成功

2.2 启动方法二

- 运行编译命令 `bazelisk build rtp_llm:rtp_llm --config=rocm --jobs=160`
- 在路径 `bazel-bin/rtp_llm` 下会生成一个whl包，安装该whl包
- 剩余步骤与方法一相同

3.生成trace

在镜像 `rtp_llm_perf:0.0.0` 的路径 `/root` 下有三个脚本:

```
/root
    run_perf_dense.py
    run_perf_dense.sh
    trace_parse.py
```

依据server启动命令，对应修改脚本 `run_perf_dense.sh` 中的变量 `START_PORT` ，`TOKENIZER_PATH` 和 `CHECKPOINT_PATH` ，运行 `sh run_perf_dense.sh` 即可在server的启动路径下生成一个trace json文件

4.解析trace文件

- 使用脚本 `/root/trace_parse.py` 可以对trace json进行解析
- 也可以使用Perfetto UI对trace json进行可视化