AMD

ROCm

Chapter 4.2: Converting CUDA CNN to HIP

ROCm Tutorial | AMD 2020

AMD ROCm

Table of Contents

CHAPTER 4.2: CONVERTING CUDA CNN TO HIP 2

PREPARATION 2
SCANNING 2
PORTING 3
MAKEFILE CHANGE 4
COMPILING AND EXECUTING 4

AMD ROCm

Chapter 4.2: Converting CUDA CNN to HIP

This hands-on tutorial shows how we can convert a publicly available Convolutional Neural Network which is written in CUDA to HIP.

Preparation

1. Install the hipblas and rocsolver library which is required by this application:

   sudo apt update && sudo apt install hipblas && sudo apt install rocsolver

2. Add ROCM library paths:

   export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/rocm/lib/

   Alternatively, you can also add this line to your bashrc for future use

3. Now move to the working directory:

   cd 02_Cuda_To_HIP_CNN/

   Copy the files in CUDA_CNN to HIP_CNN

   cp -r CUDA_CNN/ HIP_CNN/

Scanning

1. Run the following command:

   hipexamine-perl.sh .

   where “.” refers to the current directory

   You will see the scanned output in **Figure 1**

2. From Figure 1:
    - We see what are the APIs that can be converted to HIP for the different files in the project

```
  info: converted 32 CUDA->HIP refs ( error:0 init:1 version:0 device:0 context:0 module:0 memory:1 virtual_memory:0 addressing:0 stream:0 event:0 external_re
source_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0 D3D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL:
0 thread:0 complex:0 library:2 device_library:0 device_function:0 include:0 include_cuda_main_header:1 type:2 literal:0 numeric_literal:2 define:0 extern_shar
ed:0 kernel_launch:23 )
  warn:0 LOC:211 in './main.cu'
  info: converted 28 CUDA->HIP refs ( error:0 init:0 version:0 device:0 context:0 module:0 memory:22 virtual_memory:0 addressing:0 stream:0 event:0 external_r
esource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0 D3D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL
:0 thread:0 complex:0 library:0 device_library:0 device_function:3 include:0 include_cuda_main_header:0 type:0 literal:0 numeric_literal:3 define:0 extern_sha
red:0 kernel_launch:0 )
  warn:0 LOC:398 in './layer.cu'
  info: converted 2 CUDA->HIP refs ( error:0 init:0 version:0 device:0 context:0 module:0 memory:0 virtual_memory:0 addressing:0 stream:0 event:0 external_res
ource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0 D3D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL:0
 thread:0 complex:0 library:0 device_library:0 device_function:0 include:0 include_cuda_main_header:2 type:0 literal:0 numeric_literal:0 define:0 extern_share
d:0 kernel_launch:0 )
  warn:0 LOC:62 in './layer.h'

  info: TOTAL-converted 62 CUDA->HIP refs ( error:0 init:1 version:0 device:0 context:0 module:0 memory:23 virtual_memory:0 addressing:0 stream:0 event:0 exte
rnal_resource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0 D3D9:0 D3D10:0 D3D11:0 VDPAU
:0 EGL:0 thread:0 complex:0 library:2 device_library:0 device_function:3 include:0 include_cuda_main_header:3 type:2 literal:0 numeric_literal:5 define:0 exte
rn_shared:0 kernel_launch:23 )
  warn:0 LOC:841
  kernels (1 total) :   makeError(1)

  hipLaunchKernelGGL 23
  hipMalloc 7
  hipFree 7
  hipMemset 5
  hipMemcpy 4
  hipMemcpyHostToDevice 3
  hip_runtime 2
  hipError_t 1
  hipInit 1
  hipSuccess 1
  hipMemcpyDeviceToHost 1
```

Figure 1: Output of HIP scanning tool

Porting

1. Run “hipconvertinplace-perl.sh .” where “.” refers to the current directory.

2. The script will traverse all the files and hipify the CUDA code to HIP.

3. You will observe the output as shown in Figure 2 from the conversion process

```
  info: converted 32 CUDA->HIP refs ( error:0 init:1 version:0 device:0 context:0 module:0 memory:1 virtual_memory:0 addressing:0 stream:0 ev
ent:0 external_resource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0 D
3D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL:0 thread:0 complex:0 library:2 device_library:0 device_function:0 include:0 include_cuda_main_header:1 type
:2 literal:0 numeric_literal:2 define:0 extern_shared:0 kernel_launch:23 )
  warn:0 LOC:211 in './main.cu'
  info: converted 28 CUDA->HIP refs ( error:0 init:0 version:0 device:0 context:0 module:0 memory:22 virtual_memory:0 addressing:0 stream:0 e
vent:0 external_resource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0
D3D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL:0 thread:0 complex:0 library:0 device_library:0 device_function:3 include:0 include_cuda_main_header:0 typ
e:0 literal:0 numeric_literal:3 define:0 extern_shared:0 kernel_launch:0 )
  warn:0 LOC:398 in './layer.cu'
  info: converted 2 CUDA->HIP refs ( error:0 init:0 version:0 device:0 context:0 module:0 memory:0 virtual_memory:0 addressing:0 stream:0 eve
nt:0 external_resource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 openGL:0 D3
D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL:0 thread:0 complex:0 library:0 device_library:0 device_function:0 include:0 include_cuda_main_header:2 type:
0 literal:0 numeric_literal:0 define:0 extern_shared:0 kernel_launch:0 )
  warn:0 LOC:62 in './layer.h'

  info: TOTAL-converted 62 CUDA->HIP refs ( error:0 init:1 version:0 device:0 context:0 module:0 memory:23 virtual_memory:0 addressing:0 stre
am:0 event:0 external_resource_interop:0 stream_memory:0 execution:0 graph:0 occupancy:0 texture:0 surface:0 peer:0 graphics:0 profiler:0 ope
nGL:0 D3D9:0 D3D10:0 D3D11:0 VDPAU:0 EGL:0 thread:0 complex:0 library:2 device_library:0 device_function:3 include:0 include_cuda_main_header
:3 type:2 literal:0 numeric_literal:5 define:0 extern_shared:0 kernel_launch:23 )
  warn:0 LOC:841
  kernels (1 total) :   makeError(1)

  hipLaunchKernelGGL 23
  hipMalloc 7
  hipFree 7
  hipMemset 5
  hipMemcpy 4
  hipMemcpyHostToDevice 3
  hip_runtime 2
  hipMemcpyDeviceToHost 1
  hipSuccess 1
  hipInit 1
  hipError_t 1
```

Figure 2: Output of HIP porting tool

Makefile Change

1. We will need to now manually change the Makefile to support building for HIP

2. The updated Makefile is provided in the “02_Cuda_To_HIP_CNN” folder.
    - Copy this Makefile to the “HIP_CNN” folder

3. You can also compare the CUDA based original Makefile and the HIP Makefile to see the differences in the compilation commands

4. Basically, the major differences are:
    - Converting CUDA specific libraries and compiler invocations to HIP based invocation
    - Added HIP specific paths

Compiling and Executing

1. Run make
2. This will build your application
3. To execute:

- Run ./CNN
- You will observe the training process of the CNN

4. Note: If you get a cannot find library error, ensure that your LD_LIBRARY_PATH is correctly updated:
    - export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/rocm/lib