Accelerate Python Analytics on GPUs with RAPIDS

Jinhui Qin
SHARCNET | Compute Ontario | Compute Canada
Western University

Outline

- Overview
- What is available in RAPIDS?
- Getting started with RAPIDS on Graham
    - Based on a singularity container for RAPIDS

Overview

- Data Science
  - a field about extracting knowledge and insights from data
- Python is widely used in the data analytics communities
  - e.g. Numpy, Pandas, Scikit-learn, etc.
- How to process lots of data?

Overview

- Distributed data analysis
  - A cluster of many machines
  - Distributed software framework
    - E.g. Dask, Spark
- How about using GPU accelerators?

Overview

- GPU accelerators
  - Specialized processing units with highly parallel structures
  - Used in scientific and engineering computing
    - parallel algorithms
    - programming skills, e.g. CUDA, C/C++, etc.
- How about using GPUs in the Python world?

Overview

- Using GPUs in Deep learning
  - Intensive computing workload
  - Distributed across multi-GPU nodes
  - E.g. Pytorch, Tensorflow, etc.
- How about using GPUs in analytics with Python in general?
  - E.g. heavy workload in extracting, filtering, and transforming data
  - Nice to have Python libraries with familiar APIs to execute on GPUs

RAPIDS

- A suite of open source software libraries from NVIDIA
- Executing data science and analytics pipelines on GPUs
- Built on NVIDIA CUDA primitives
- Provided with user-friendly Python interface
  - E.g. Pandas-like, Scikit-learn-like, etc.
- Scallabe with Dask integration
  - Supports multi-GPU on a server or on a GPU cluster
    - E.g. cuDF, single GPU DataFrame library
    - E.g. dask-cuDF, GPU DataFrame across multi-GPU or a cluster

What are available in RAPIDS?

- cuDF
  - GPU DataFrame library with a *pandas-like* API
- cuML
  - GPU accelerated ML library with a *scikit-learn-like* API
- cuGraph
  - GPU accelerated graph analytics library
- CLX (Cyber Log Accelerators)
  - GPU accelerated cybersecurity analysis
- cuSpatial
  - GPU accelerated library for GIS workflow
- cuxfilter
  - For GPU DataFrame visualization
- cuSignal
  - GPU accelerated signal processing via cuPy, Numba, etc.
- More ...
  - Libraries for development
  - Examples, use cases, etc.
  - API docs: https://docs.rapids.ai/api

GPU resources on general purpose clusters

Cluster	# of nodes	GPUs / node	GPU type	GPU mem	Total GPUs
Beluga	172	4	V100	16GB	688
Cedar	146	4	P100	12GB or 16GB	1352
	192	4	V100	32GB	
Graham	160	2	P100	12GB	490
	7	8	V100	16GB	
	30	4	T4	16GB	

Request GPUs with **--gres** in a Slurm job script

https://docs.computecanada.ca/wiki/Using_GPUs_with_Slurm

Get started

- RAPIDS Installation
  - Conda
  - Build from source
  - Docker images
- Working with a Singularity container
  - Build a singularity container from a docker container
  - CC docs: https://docs.computecanada.ca/wiki/Singularity

Build a singularity container for RAPIDS

- Where to find a docker image for RAPIDS:
  - NGC: https://ngc.nvidia.com/catalog/containers/ e.g. search for RAPIDS
  - Docker Hub: https://hub.docker.com/r/rapidsai/rapidsai-dev for RAPIDS-dev
- Build a singularity image, called *rapids.sif*, from a docker image
  - E.g. a docker pull tag: docker pull rapidsai/rapidsai-core-dev
  - E.g. to build a .sif image: singularity build *rapids.sif* docker://rapidsai/rapidsai-core-dev
- Some notes
  - Base image: for submitting a job execution to slurm scheduler
  - Runtime image: for interactively working on Jupyter Notebook
  - Devel image: for RAPIDS development, e.g. working with cuda toolkit, etc.
  - Require relatively large resources to build a singularity image

Explore RAPIDS Singularity container

- Start RAPIDS shell with a singularity image, e.g. named *rapids.sif*

  [user@cluster]$ **module load singularity**

  [user@cluster]$ **singularity shell rapids.sif**

  Singularity>

- Initiate Conda and activate RAPIDS env

  Singularity> **source /opt/conda/etc/profile.d/conda.sh**

  Singularity> **conda activate rapids**

  (rapids) Singularity>

Explore RAPIDS Singularity container

- List available Python packages

  (rapids) Singularity> **conda list**

- Deactivate RAPIDS env and exit from the container

  (rapids) Singularity> **conda deactivate**

  Singularity> **exit**

  [user@gra-loin]$

Work interactively on a GPU node

- Request an interactive session on a GPU node
  - using salloc and with --gres option, e.g. request a T4 type GPU on Graham

[user@gra-login]$ **salloc --time=1:00:00 --account=def-user --ntasks=1 --cpus-per-task=2 --mem=10G --gres=gpu:t4:1**

[user@gra####]$

- Start RAPIDS shell on GPU node

[user@gra####]$ **module load singularity**

[user@gra####]$ **singularity shell --nv -B /home -B /project -B /scratch rapids.sif**

Singularity>

- Check GPU status

Singularity> **nvidia-smi**

Work interactively on a GPU node

- Launch Jupyter Notebook server manually in container shell

  Singularity> **source /opt/conda/etc/profile.d/conda.sh**

  Singularity> **conda activate rapids**

  (rapids) Singularity> **jupyter-lab --ip $(hostname -f) --no-browser**

- Launch Jupyter Notebook server by running the container image

  [user@gra####]$ **singularity exec --nv -B /home -B /project -B /scratch rapids.sif**

- Connect to Jupyter Notebook
  - Setup SSH tunnel on your local computer
  - CC docs for Jupyter, i.e. Step 6 at https://docs.computecanada.ca/wiki/Jupyter

Submit a RAPIDS job to Slurm Scheduler

Example job submission script, *submit.sh*

```
#! /bin/bash
#SBATCH --time=dd:hh:mm
#SBATCH --account=def-someuser
#SBATCH  --ntasks=1
#SBATCH  --cpus-per-task=2
#SBATCH  --mem=10G
#SBATCH  --gres=gpu:t4:1

module load singularity

singularity run -B /home  -B /scratch --nv rapids.sif  /path/to/run_script.sh
```

Example job execution script, *run_script.sh*

```
#! /bin/bash

source /opt/conda/etc/profile.d/conda.sh
conda activate rapids
nvidia-smi

python /path/to/my_rapids_code.py
```

Note: /scratch is mounted as run_script.sh and my_rapids_code.py are located on /scratch in the example

[user@cluster]$ **sbatch submit.sh**

- submit.sh: job submission script
- rapids.sif: singularity image for RAPIDS
- run_script.sh: script to run in the container
- my_rapids_code.py: python code programmed with RAPIDS

Helpful links

- RAPIDS docs
  - https://docs.rapids.ai/
- Getting started with examples
  - https://github.com/rapidsai/notebooks
- More use cases and blogs
  - https://medium.com/rapids-ai

Next: Demo on Graham