Project #1

Student Names and SFUIDs:

Chenyu Ru	301323578
Jiangpei Chen	301326516

Purpose: This pair-project focuses on documentation for a software package.

Tasks:

Phase One (Today – Wednesday January 29th):

- Find a partner for this week-long project. Please do not ask me to pair you up. If you cannot find a partner, then you will have to work alone.
- Become familiar with the concept of pair-programming – know the pros and cons and be able to classify it within the context of software engineering practices.
- Read and understand *everything* in the following blog post:

  https://www.ybrikman.com/writing/2014/05/05/you-are-what-you-document/

Phase Two (Due next Friday, February 7th):

- You will then be set upon a week-long task next class (Friday, June 3rd) involving the *three types of documentation* referred to in the post above.
- Submission instructions will be provided next week by the TA.
- You will have the rest of next week to work on this project outside of class in your pairs.
- Use this document as your cover page

IMPORTANT NOTE: I do not make attendance mandatory. However, it is the student's responsibility to find out what was discussed from your peers. Assignments and in-class activities are designed so that attendance is NOT mandatory. However, those that do not attend will find the assignment somewhat difficult.

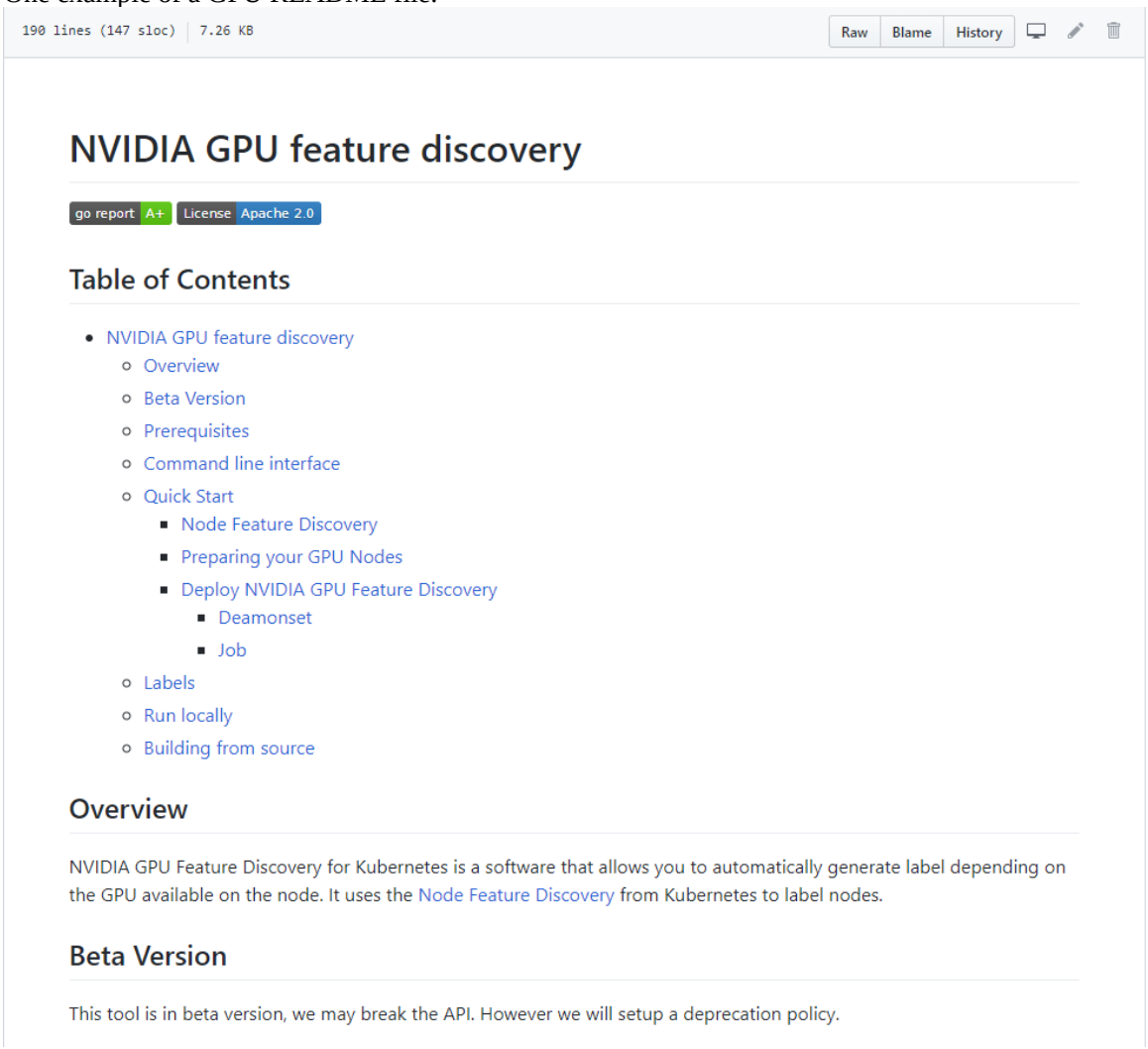
As the summary of the principle reference *You are what you document*, it summarized the documents into three different types. Written documentation, Code documentation, Community documentation. *Each type of documentation solves a different problem, so most projects should include some mix of all three types.*
https://www.ybrikman.com/writing/2014/05/05/you-are-what-you-document/

In this project, I will give some examples for those three different types of documents of GPU as the coprocessors that enhances the core functionality of a CPU.

Written documentation

Written documentation is usually the most straight forward documentation that we feel, when we mention "software documentation". For some examples of written documentation, I choose README, reference guides (Manual).

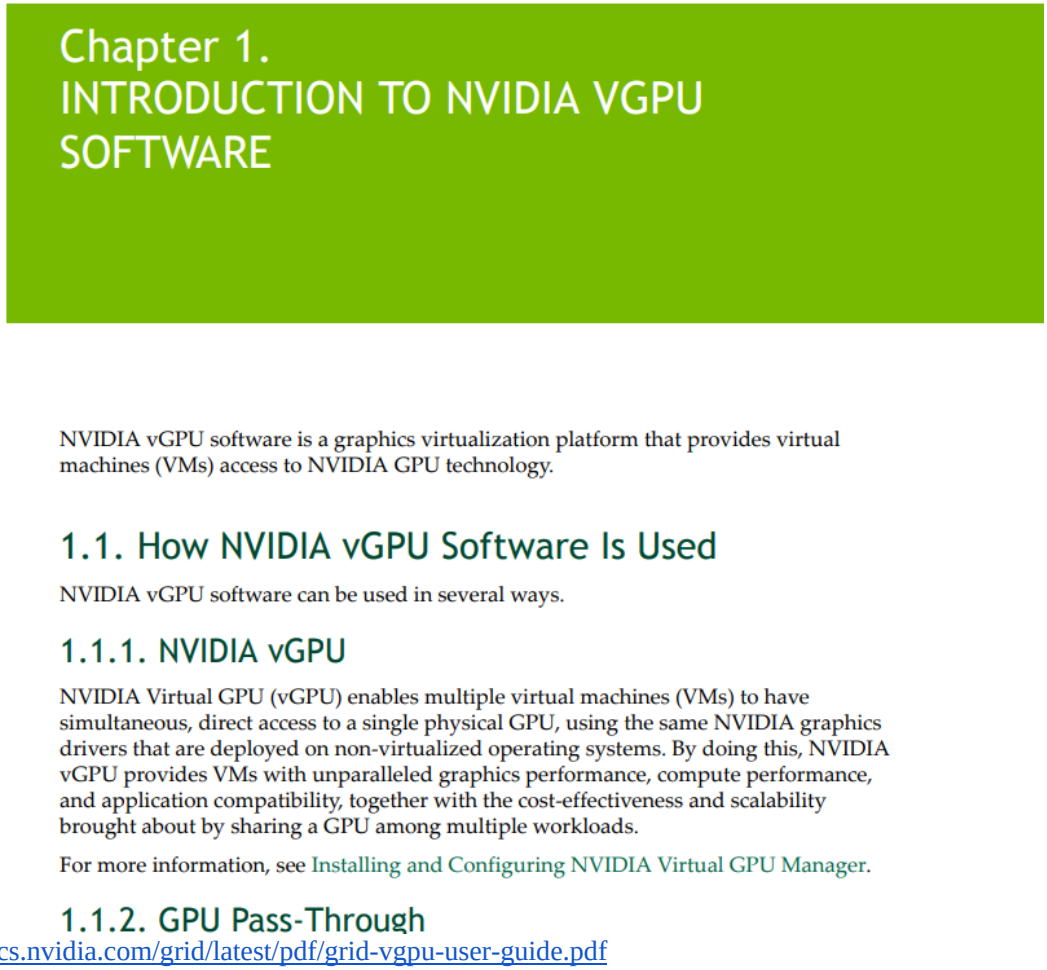
README is more like a brief introduction of that software. It basically includes Description, Quick examples, Quick start, Further documentation, Project organization, Legal notices as the definition of *You are what you document.*
One example of a GPU README file.



https://github.com/NVIDIA/gpu-feature-discovery/blob/master/README.md

Other example of a NVIDIA GPU README files
https://download.nvidia.com/XFree86/Linux-x86_64/165.33.09/README/
https://github.com/NVIDIA/gpu-operator/blob/master/README.md

Then as reference guides, I think it more likes a detailed of a manual of the software or hardware.
It contains all the topics and the users can use this reference guides to find all the information of the software (hardware) and can navigate the functionality, implement training, troubleshooting as well.
One simple reference guide of NVIDIA virtual GPU



Chapter 1. INTRODUCTION TO NVIDIA VGPU SOFTWARE

NVIDIA vGPU software is a graphics virtualization platform that provides virtual machines (VMs) access to NVIDIA GPU technology.

1.1. How NVIDIA vGPU Software Is Used

NVIDIA vGPU software can be used in several ways.

1.1.1. NVIDIA vGPU

NVIDIA Virtual GPU (vGPU) enables multiple virtual machines (VMs) to have simultaneous, direct access to a single physical GPU, using the same NVIDIA graphics drivers that are deployed on non-virtualized operating systems. By doing this, NVIDIA vGPU provides VMs with unparalleled graphics performance, compute performance, and application compatibility, together with the cost-effectiveness and scalability brought about by sharing a GPU among multiple workloads.

For more information, see Installing and Configuring NVIDIA Virtual GPU Manager.

1.1.2. GPU Pass-Through

https://docs.nvidia.com/grid/latest/pdf/grid-vgpu-user-guide.pdf

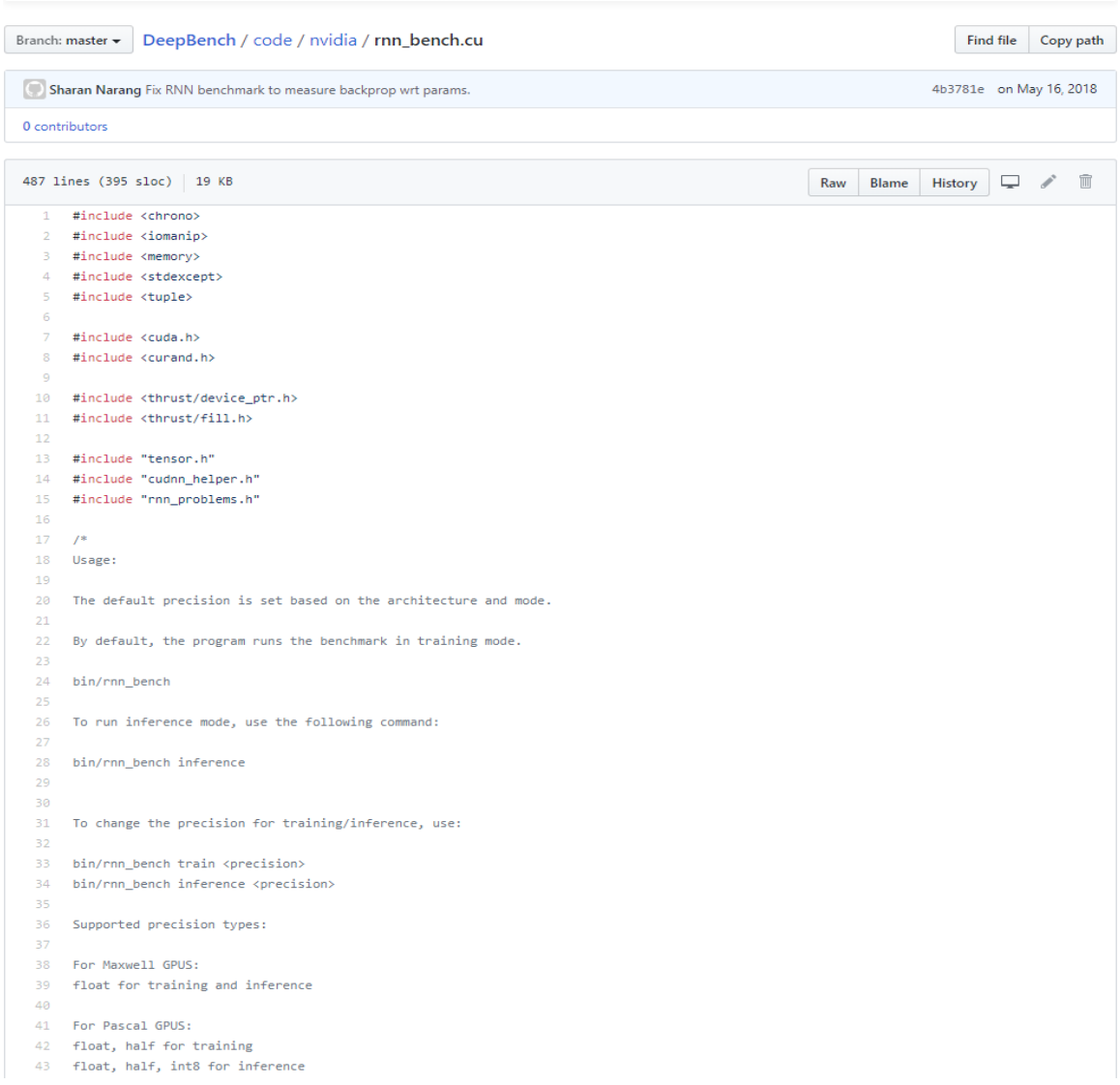
Other examples of NVIDIA GPU.
https://download.nvidia.com/XFree86/Linux-x86_64/165.33.09/README/
https://docs.nvidia.com/grid/latest/

Code Documentation

Code Documentation is more like a core documentation of a software, it is more self-documented as the software itself. But not only the core code of a software can be regarded as code documentation.
Any code that that can help the programmers, testers or debuggers to navigate their demands, can be traded as Code Documentation.
I will choose code comments and testing cases as examples of Code Documentation.

Code Comments
Code comment is an interpretation of the real code by using human-languages. It can help the developer itself or other programmers to understand how the specific function works and what is the logic behinds.

One example of the code comment for a benchmark function in GPU development.



Branch: master ▾ DeepBench / code / nvidia / rnn_bench.cu — Find file | Copy path

Sharan Narang Fix RNN benchmark to measure backprop wrt params. 4b3781e on May 16, 2018

0 contributors

487 lines (395 sloc) | 19 KB — Raw | Blame | History

```
#include <chrono>
#include <iomanip>
#include <memory>
#include <stdexcept>
#include <tuple>

#include <cuda.h>
#include <curand.h>

#include <thrust/device_ptr.h>
#include <thrust/fill.h>

#include "tensor.h"
#include "cudnn_helper.h"
#include "rnn_problems.h"

/*
Usage:

The default precision is set based on the architecture and mode.

By default, the program runs the benchmark in training mode.

bin/rnn_bench

To run inference mode, use the following command:

bin/rnn_bench inference


To change the precision for training/inference, use:

bin/rnn_bench train <precision>
bin/rnn_bench inference <precision>

Supported precision types:

For Maxwell GPUS:
float for training and inference

For Pascal GPUS:
float, half for training
float, half, int8 for inference
```

https://github.com/baidu-research/DeepBench/blob/master/code/nvidia/rnn_bench.cu

Other examples of code comments
https://github.com/tbennun/mgbench/blob/master/src/L0/devinfo.cpp

Test cases
Test case is a special code that can demonstrate the expected functionality of code. It can assist the developers to quickly detect the error or bugs especially if it is a huge project.

One example of a test file in GPU development.

Kernel Code - test.cl

The kernel code is generated from the unmodified source code of the tested C program. The tool turns the C source into OpenCL code, by turning the `main` function into an OpenCL kernel function. It also modifies the way inputs are being read, by replacing references to `argv` and standard input with references to the auto-generated **input** structure.

To illustrate this, consider the resulting OpenCL code for **add.c**:

```
1 int add(int a, int b){
2   return a + b;
3 }
4
5 __kernel void main_kernel(
6           __global struct input* inputs,
```

https://wyaneva.github.io/papers/thesis2016.pdf

Another example of a test file

```
#define ARRAY_SIZE 9

void quickSort(int[], int, int);
int partition(int[], int, int);

int main(void) {
  // sample input array as test case
  int a[] = { 7,12,1,-2,0,15,4,11,9 };
  quickSort(a, 0, ARRAY_SIZE-1);
  return 0;
}
```

Figure 2: Harness for testing Quicksort with one test input

```
#define ARRAY_SIZE 9
#define NUM_TESTS 256

__device__ void quickSort(int[], int, int);
__device__ int partition(int[], int, int);

// GPU function
__global__ void compute(int *tests) {
  // The thread ID identifies the test case
  int test_case = threadIdx.x*ARRAY_SIZE;
  quickSort(tests+test_case, 0, ARRAY_SIZE-1);
}

int main(void) {
  int host_inputs[NUM_TESTS][ARRAY_SIZE] = {....};
  int *device_inputs;

  cudaMalloc((void **)&device_inputs, sizeof(host_inputs));
  cudaMemcpy(device_inputs, host_inputs,
             sizeof(host_inputs), cudaMemcpyHostToDevice);

  // Number of blocks is 1 and number of threads per block is 256.
  compute<<<1, NUM_TESTS>>>(device_inputs);
  return 0;
}
```

Figure 3: CUDA test harness for Quicksort with 256 tests

http://www.kroening.com/papers/ase2014.pdf

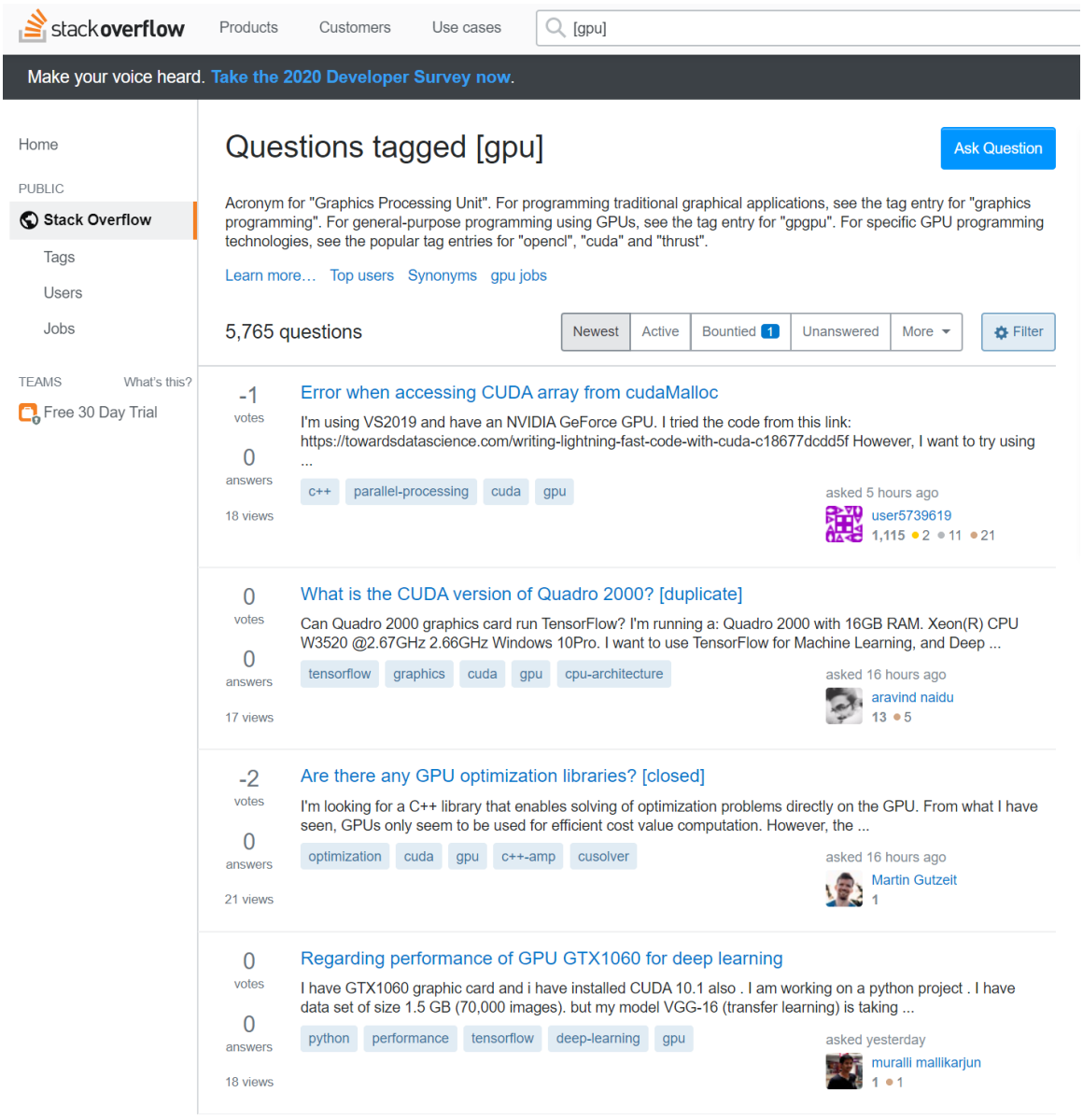
Other examples
http://www.prace-ri.eu/IMG/pdf/wp67.pdf

Community documentation

This is also an important documentation for a software. It is used for the people who involved with the project discuss questions and take some help.

And here is an example of stack overflow, which is a website used for people post questions and share their idea, solutions etc.

Here is the website:
https://stackoverflow.com/questions/tagged/gpu



Another example for stack overflow
https://discourse.julialang.org/t/stack-overflow-on-cuda/10735